2012

Development of a Cell-Centred Finite Difference Numerical Methodology on Triangulated Domains

James Situ
University of Windsor

Development of a Cell-Centred Finite Difference Numerical Methodology on Triangulated Domains

by

JAMES J. SITU

A Thesis

Submitted to the Faculty of Graduate Studies

through Mechanical, Automotive and Materials Engineering

in partial fulfillment of the requirements for

the Degree of Master of Applied Science at the

University of Windsor

Windsor, Ontario, Canada

2011

Development of a Cell-Centred Finite Difference Numerical Methodology on Triangulated Domains

by

James J. Situ

APPROVED BY:

____
Dr. M.S. Monfared, External Department Reader
Department of Mathematics and Statistics
University of Windsor

____
Dr. N.G. Zamani, Internal Department Reader
Department of Mechanical, Automotive and Materials Engineering
University of Windsor

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

____
Dr. V. Stoilov, Chair of Defence
Department of Mechanical, Automotive and Materials Engineering
University of Windsor

Author's Declaration of Originality

I, James Jianmin Situ, hereby certify that I am the sole author of this thesis. This thesis includes original research that has been previously published/submitted for publication in peer reviewed conference proceedings, as follows,

Thesis Chapter	Publication title/full citation	Publication status
Chapter 2 & 4	Situ, J.J. and Barron, R.M. *Development of a Cell-Centred Finite Difference Scheme for Numerical Simulations on Triangulated Domains.* 19th Annual Conf. of the CFD Society of Canada, Montreal, April 2011.	*Presented*
Chapter 2 & 4	Situ, J.J. and Barron, R.M. *Cell-Centred Finite Numerical Methodology for Solving Partial Differential Equations on Triangulated Domains.* Applied Mathematics, Modeling and Computational Science Conference, Waterloo, July 2011.	*Under review*

I certify that the above material describes work completed during my registration as a graduate student at the University of Windsor.

I declare 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, I certify that I have not included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act.

Provisional Patent Application 61/457,589 has been filed on April 22, 2011 for Patent Protection for: "C**ell-Centered Finite Difference Method for Numerical Solution of Partial Differential Equations on Arbitrary Mesh Systems"**

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 in any form for another degree or diploma to any other University or Institution.

Abstract

A cell-centred finite difference (CCFD) method for unstructured mesh topology is proposed and applied to model partial differential equations (PDEs) governing fluid flow and solid mechanics phenomena. The numerical method implements a finite difference approximation at cell centroids by taking differencing points along orthogonal Cartesian axes localized within each cell. The predominant advantage of this method is that it can be applied to arbitrary mesh topologies, including structured, unstructured and hybrid meshes. Either a direct or iterative approach is used to solve the system of equations developed by the proposed method. The numerical method is designed to solve a variety of physical phenomena governed by PDEs, such as electrostatic potential in electromagnetic fields, stress and strain in structural mechanics and wave phenomena in physics. The focus of the thesis research is to investigate the application of this methodology in heat transfer and fluid mechanics problems. This new finite difference methodology is applied to typical “benchmark” problems in such fields, covering the representative of different types of PDEs with initial and boundary conditions. Solutions obtained are compared to exact solutions if available from analytical methods or to the results from other reliable numerical simulations.

Acknowledgement

It is a pleasure to thank many people who made this thesis possible.

I am indebted in the preparation of this thesis to my supervisor, Dr. Ronald M. Barron of University of Windsor, whose patience and kindness as well as his indepth academic knowledge, have been invaluable to me. With his enthusiasm, his inspiration and his great efforts to explain concepts effectively, he helped me expand my knowledge greatly in the engineering numerical analysis field. Throughout the period of writing this thesis, he provided encouragement, advice and lots of innovative ideas. Without him, I would not have been able to complete my thesis.

I would like to thank my committee members, Dr. Nader G. Zamani and Dr. Mehdi S. Monfared, who are always available to provide support to my research when I needed it. Also, lots of constructive suggestions from their feedbacks help improve the quality of my thesis.

Lastly but not least, I wish to thank many friends, and especially my parents, who give me constant support, both spiritually and materially.

Table of Contents

List of Figures

List of Tables

Nomenclature

CC	Cell centroid; cell centre
IP	Interpolation point
OA	Order of accuracy
RE	Relative error
RD	Relative difference
RMSE	Root mean square error
RMSD	Root mean square difference
SOR	Successive over-relaxation scheme
ζ	Spatial coordinate in computational stencil
η	Spatial coordinate in computational stencil
a_i	Coefficients of polynomial transformation from *x* to ζ
b_i	Coefficients of polynomial transformation from *y* to η
γ_i	Coefficients in finite difference equation
ω	Relaxation parameter
$\forall$	For all points
Ω	Interior of a domain
$\partial\Omega$	Boundary of a domain
$\cup$	Union of sets
$\emptyset_i^k$	Dependent variable at node/cell centroid index *i* at *k*th iteration
$\emptyset_{e,i}$	Exact value at index *i*
$\left.\frac{\partial^k \emptyset}{\partial x^k}\right\|_i$	*k*th spatial derivative of Φ with respect to *x* at node/cell centroid index *i*
$\left.\frac{\partial^k \emptyset}{\partial t^k}\right\|^n$	*k*th time derivative of Φ at time step *n*
g(x,y)	Neumann boundary condition function
x	Independent variable
y	Independent variable
q	Triangular quality

t	Time variable
N^i	Test function
W^i	Weight function
$N_i(x,y)$	Shape function
L	Operator
α	General differencing point index
S	South differencing point
N	North differencing point
W	West differencing point
E	East differencing point
M	Mid differencing point
L_{AB}	Distance between two points A and B
Δx	Adjacent differencing points spacing along *x* direction
Δy	Adjacent differencing points spacing along *y* direction
Δt	Discretised time step size

CHAPTER 1 – OVERVIEW OF PARTIAL DIFFERENTIAL EQUATIONS AND NUMERICAL TECHNIQUES

Partial differential equations (PDEs) arise in connection with various thermofluid and solid mechanics problems. The governing PDEs are derived from physical principles and lead to initial and boundary value problems in both time and spatial domains.

1.1 Preliminary Concepts of Partial Differential Equations

A partial differential equation is defined as an equation involving one or more partial derivatives of a function of two or more independent variables. The order of the highest derivative is called the order of the equation [1]. The solution of a PDE in a domain Ω is a function that has all partial derivatives appearing in the equation and satisfies the equation everywhere in Ω. However, a solution of a PDE is generally not unique. A unique solution may be obtained by the use of additional information imposed by the physical conditions, i.e. boundary conditions that give the values of the required solution on the boundary and/or initial conditions that prescribe the value of the solution at initial time t = 0. Some mathematical theorems describe the criteria for solution existence and uniqueness of linear PDEs, but these theorems do not generally apply to nonlinear PDEs.

1.2 Classification of PDEs

A PDE for the function $\emptyset(x_1,..x_n)$ has the form

$$F\left(x_1,..,x_n,\emptyset,\frac{\partial\emptyset}{\partial x_1},\frac{\partial^2\emptyset}{\partial x_1^2},\frac{\partial^2\emptyset}{\partial x_1\partial x_2},....\right)=0 \tag{1.1}$$

The PDE is linear if it is of the first degree in the dependent variable $\emptyset$ and its partial derivatives $\partial^k\emptyset/\prod_i^n\partial x_i^{k_i}$. The independent variables x_i's can represent spatial coordinates, time or other physical parameters, such as pressure, temperature, etc. A nonlinear PDE contains the product of the dependent variable with itself or one of its derivatives. If each term of the equation (1.1) contains either the dependent variable or one of its derivatives, the equation is said to be homogeneous; otherwise it is said to be nonhomogeneous [1].

In addition to the distinction between linear and nonlinear PDEs, further classification of PDEs is essential for computational scientists and engineers working on numerical simulation. Linear, second order PDEs can be classified as parabolic, hyperbolic or elliptic based on the characteristic curves associated with the equation. Consider the following second order linear PDE

$$A\frac{\partial^2 \emptyset}{\partial x^2} + B\frac{\partial^2 \emptyset}{\partial x \partial y} + C\frac{\partial^2 \emptyset}{\partial y^2} + D\frac{\partial \emptyset}{\partial x} + E\frac{\partial \emptyset}{\partial y} + F\emptyset + G = 0 \tag{1.2}$$

Discontinuities in the second order derivatives of the dependent variable may arise across the characteristics. Characteristic curves can be real or imaginary depending on the discriminant value of the second order derivative coefficients. The second order PDE (1.2) is classified according to the sign of the expression ($B^2 - 4AC$) as follows:

1. Elliptic if $B^2 - 4AC < 0$. An elliptic PDE has no real characteristic curves and any disturbance is propagated instantly in all direction within the region [2]. The solution domain is a closed region. This type of PDE usually arises in physical application of diffusion processes into an equilibrium state, such as a steady state temperature distribution or fluid motion at subsonic speed.
2. Parabolic if $B^2 - 4AC = 0$. The solution domain is an open region and such PDEs only exhibit one characteristic curve. The solution marches downstream within the domain from prescribed initial conditions while satisfying the specified boundary conditions [2]. For a physical interpretation, parabolic PDEs arise in time-dependent diffusion problems, such as unsteady heat conduction. Mathematically, parabolic PDEs serve as a transition from hyperbolic PDEs to elliptic PDEs.
3. Hyperbolic if $B^2 - 4AC > 0$. A hyperbolic PDE has two real characteristic curves and the solution domain exhibits a disconnected conic section. Hyperbolic PDEs usually arise in connection with mechanical oscillations, such as a vibrating string or plate, or in convection driven transport problems.

1.3 Analytical Solution Method

Although analytical solutions for most PDEs are not obvious and may not even exist, some rudimentary approaches are used to solve some well-imposed, linear PDEs. The general analytical approaches to solve such PDEs include separation of variables, conformal mapping, infinite series, coordinate and dependent variable transformations and perturbation methods. Analytical solutions are available for some of the problems considered in this thesis, and will be used as needed.

1.4 Experimental Solution Method

Experiments, as an alternative to numerical simulation, are often used for validation of simulation results for a physical problem governed by PDEs. Experimental fluid mechanics provides information regarding a particular flow field and thus experimental data is used along with computational solutions of the equations for design purposes. Nevertheless, limitations on hardware, such as wind tunnel size and measurement resolution, sometimes make it impractical to perform an experiment. Huge costs may also be encountered and some experiments are not possible to conduct, such as solar or galactic events and nuclear explosions. For these reasons, numerical simulations are used by engineers to reconstruct the physical condition under the appropriate boundary and initial conditions.

1.5 Numerical Solution Method

Numerical methods specify a finite discretized domain from the continuum physical domain and each finite discretization unit is analyzed individually. From a numerical methods perspective, there are three well-established primary methodologies for solving PDEs in a pre-defined mesh topology; finite difference, finite volume and finite element methods. Finite difference (FD) and finite element (FE) methods are usually applied in solid or fluid mechanics, while the finite volume (FV) method is popular in fluid mechanics. Lohner [3] has classified the three numerical methods by choice of the trial and test functions N^i and W^i based on a weighted residual formula.

1.5.1 Finite Difference Method

The FD method takes N^i as a polynomial and $W^i = \delta(x_i)$, where δ is the delta function, such that the operator approximation $L(\emptyset^h)$ is enforced at a finite number of locations in space. The choice of polynomial N^i determines the order of accuracy of the resulting stencil of the operation. The traditional FD methods are commonly used in Computational Fluid Dynamics (CFD) for problems that exhibit a moderate degree of geometrical complexity, or within multiblock solvers. The discretization stencils are derived for structured grids with uniform element size *h*. For this reason, for complex geometries, FD methods usually require transformation from an arbitrary physical domain to a structured and uniform computational domain.

1.5.2 Finite Volume Method

The FV (and FE) methodologies on the other hand have the capability of handling unstructured or hybrid mesh systems. Most commercial and research CFD codes for solving fluid flow and heat transfer problems are based on the FV methodology because of the clear relationship between the numerical algorithm and the underlying physical conservation principle [3]. The FV method employs integration of the governing equations over all finite control volumes of the domain. In Lohner's definition, FV methods are obtained by taking polynomial N^i and $W^i = 1$ if integration is within the element and 0 otherwise. Since the test function is set in Kronecker delta form in each of the respective elements, any integration by parts over the control volume reduces to element boundary integrals. This implies that only the normal fluxes through the element faces appear in the discretization [4]. However, discretization in the time domain for time-dependent problems is one of the limitations in FV because of global conservation. Some revised FV techniques handle this problem, such as the integrated space-time (IST) FV method proposed by Zwart [5]. In his dissertation, a space-time meshing algorithm and a solver were developed for the IST FV method. Particular application of this method is when conservation in time is important, such as moving boundary problems involved free surface flow. Other limitations involve the development of higher-order methods, such as the use of compact Hermitian schemes available in a FD formulation. Higher-order methods are particularly important where very high solution accuracy is needed, such as in computational aeroacoustics and direct numerical simulation of turbulent flows. It is also difficult to implement the FV method on higher-order PDEs. One popular strategy in the FV method to handle higher-order derivatives is to evaluate a derivative in a pass over the mesh and then obtain the next order derivative in the subsequent pass, until the highest-order derivative has been reached [4].

1.5.3 Finite Element Method

In comparison, the FE method develops an equilibrium equation by inputting element shape functions into the weak formulation of the PDEs. The FE method can be summarized as the projection of the weak form of the differential equation onto a finite-dimensional function space, as a combination of linear piecewise basis functions [5]. In Lohner's definition of the Galerkin FE method, the polynomial trial function N^i is set to be the test function, i.e. $N^i = W^i$ [4]. This method is widely used for thermal problems, structural dynamics, potential flows and electrostatics. However, special treatments are needed to ensure a conservative solution. Lube and Rapin [6] presented different techniques to handle the mass conservation in advection-diffusion problems, such as higher-order approximations and constructing Scott-Vogelius elements. Surana et al. [7] presented k-version of the FE method in gas dynamics for higher-order global differentiability numerical solutions. The article addressed a FE approximation scheme of differential equations by space-time coupled processes in order to preserve the physics and mathematics of the initial/boundary value problem.

1.5.4 Mesh-Free Numerical Method

Besides the numerical methods based on a pre-generated mesh structure, there are methods that do not require mesh generation, such as smoothing particle hydrodynamics (SPH) and material particle semi-implicit (MPS) formulations. Regarding the SPH method, it was developed to avoid the limitation of mesh tangling encountered in extreme deformation problems. Absence of grid generation is the major advantage for the SPH formulation compared to the traditional ALE (Arbitrary-Lagrangian-Eulerian) formulation used in many fluid-structure problems. The SPH technique allows to obtain numerical solutions of the continuum equations by defining the variables at a set of suitable moving points and reconstructing the continuous field by means of interpolation functions centred on each moving points [8]. However, there are limited actual applications that exist in such a method. For example, the SPH method mostly applies to fluid-strucutre interaction problems with large deformation. It has difficulty to capture turbulence effect in high Reynolds number flows. Also, some drawbacks need to be further investigated, such as uneven particle distribution determined by characteristic length and inter-particle distance discrepancies caused by large variations [9].

1.6 Thesis Overview

In the present research work, a cell-centred finite difference (CCFD) method is developed for 2D arbitrary unstructured and hybrid mesh topologies. The primary objective of this thesis is to develop the necessary equations, discuss the important features of the method and demonstrate its potential applicability. Chapter 2 describes the algorithmic development of the methodology and formulation of specific approximation schemes. Chapter 3 provides a preliminary overview of related subjects, including mesh topology information, methods of solving systems of equations, manufacturing of solutions, assessment criteria for the CCFD method and post-processing interpolation. Chapter 4 applies the developed formulation to benchmarked two-dimensional PDE problems with Dirichlet and Neumann boundary conditions, covering a spectrum of typical equations and boundary conditions with different geometric domains. The last chapter summarizes all the findings, analyzes sources of error and concludes with proposed future improvements.

CHAPTER 2 – DEVELOPMENT OF CCFD NUMERICAL METHOD

2.1 Algorithm for PDE Numerical Method

Specific numerical methods for PDEs involve formulation of the problem in particular differential equation forms. The FE method develops the weak form of the differential equation onto a finite-dimensional space, while the FV method applies integration over the finite control volume based on laws of conservation of mass, momentum and energy. The formulation is then applied to a predefined mesh structure or particles over the interior domain. This leads to solving a system of algebraic equations to reach the final solutions, either by direct or iterative methods. The general numerical process is shown in the following figure:

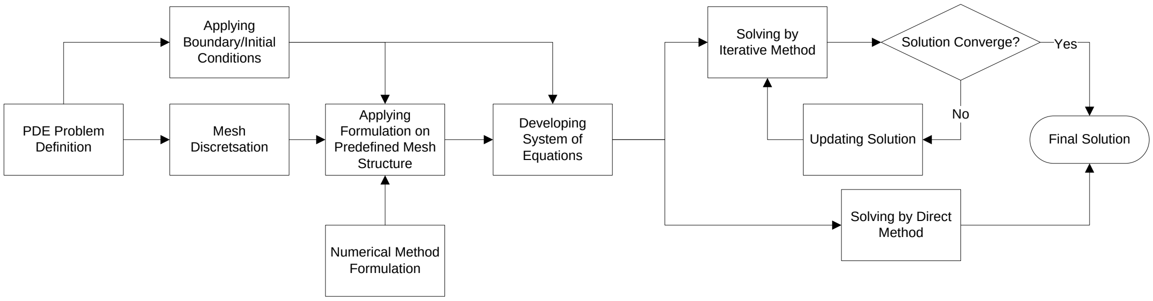

Applying Boundary/Initial Conditions
PDE Problem Definition
Mesh Discretsation
Applying Formulation on Predefined Mesh Structure
Developing System of Equations
Numerical Method Formulation
Solving by Iterative Method
Solution Converge?
Yes
No
Updating Solution
Final Solution
Solving by Direct Method

Figure 2.1 Algorithmic process for solving PDE by numerical method

The CCFD formulation is first tested by direct scheme using Gaussian elimination in a predefined simple mesh structure. Once results are validated by comparing to the exact solution, an arbitrary mesh topology can be solved by Jacobi iterative scheme. Point Gauss-Seidel and Successive Over-Relaxation (SOR) methods are also used to further improve the convergence rate.

2.2 Implementation of Finite Difference Formula in CCFD Numerical Scheme

In the CCFD method, the PDE is evaluated at the centroid of each cell. Second order partial derivatives are approximated using a central finite difference formula at the cell centre. The order of accuracy of the differencing formulae depends on the number of differencing points taken in each finite stencil. Considering a second-order central finite differencing at a cell centroid in the spatial domain, the expression for the second derivative with respect to *x* is

$$\left.\frac{\partial^2 f}{\partial x^2}\right|_{ij} = \frac{f_{i+1,j} - 2f_{ij} + f_{i-1,j}}{(\Delta x)^2} + O(\Delta x)^2 \qquad (2.1)$$

where the *y* value is held constant, i.e., $y = y_j$. Equation (2.1) approximates the second order derivative at (x_i, y_j) by taking forward and backward differencing point values at (x_{i+1}, y_j) and (x_{i-1}, y_j) and is of the order of $O(\Delta x)^2$, provided the points (x_{i+1}, y_j), (x_i, y_j) and (x_{i-1}, y_j) are equally spaced. For derivatives with respect to *y*, a similar finite difference approximation is applied. In the CCFD formulation, the finite differencing points are confined to remain within each cell. This can be achieved by setting up a local Cartesian system with cell centroid at the origin and differencing points at the intersection between the axes and cell edges. The figure below illustrates the localized finite difference stencil in an element ΔABC:

Figure 2.2 Differencing points in second order central FD stencil

The differencing points are denoted as west (W), east (E), south (S) and north (N) relative to the cell centroid, following the usual notation of the FV method.

2.3 FD Stencil Polynomial Transformation

Equation (2.1) presented in the previous section requires equally spaced grid points. However, in general, for the numerical procedure depicted in Fig. 2.2, the differencing point locations on a FD stencil are not uniformly distributed, eg., length of line segment from CC to W is not the same as from CC to E. For this reason, a transformation is required to bring an arbitrary FD stencil to an equally spaced FD stencil. For simplicity, one can use the polynomial transformation $T: \mathbb{R}^2 \rightarrow \mathbb{R}^2$ which maps the spatial variables independently and is expressed by the following equations:

$$\begin{aligned} x = x(\zeta) = a_n\zeta^n + \cdots + a_1\zeta^1 + a_0; \\ y = y(\eta) = b_n\eta^n + \cdots + b_1\eta^1 + b_0; \end{aligned} \tag{2.2}$$

The coefficients a_i's and b_i's depend on the coordinates of the differencing points referenced to a fixed global coordinate system, and localized Cartesian axes are aligned along each FD stencil. The order n of the polynomial expression depends on the order of approximation for each cell-centred finite difference expression. For example, second-order central differencing at the cell centre requires a quadratic transformation, while a fourth-order FD scheme uses a quartic transformation. The polynomial mapping is applied to the PDE as well as the FD stencil, i.e., the PDE is transformed from the (*x*, *y*) physical domain to the (ζ, η) computational domain. The major advantage of implementing this polynomial transformation is that the same order of accuracy can be maintained in the (ζ, η) FD equation as in the original (*x*, *y*) FD approximation formula.

Let's consider second-order central FD scheme in a 2D steady problem. Quadratic transformation $T: \mathbb{R}^2 \rightarrow \mathbb{R}^2$ is sufficient to bring physical coordinates (*x*, *y*) into the computational coordinates (ζ, η) with equally spaced grid points. The origin in the local Cartesian plane is set at the cell centroid and all the differencing points are mapped within a unit square domain in a localized computational stencil (ζ, η). The quadratic transformation maps the three differencing points in each direction independently, i.e. W, CC and E in *x*-direction and S, CC and N in *y*-direction. The transformation of the stencil is illustrated in the following figure:

[redacted]

Figure 2.3 Quadratic map transformation

By substituting the coordinates of the differencing points into the transformation equations (2.2) in quadratic form (n = 2), the coefficients are determined as follows:

$$\begin{aligned} a_0 = x_{CC}; \quad a_1 = \frac{1}{2}(x_E - x_W); \quad a_2 = \frac{1}{2}(x_E + x_W) - x_{CC} \\ b_0 = y_{CC}; \quad b_1 = \frac{1}{2}(y_N - y_S); \quad b_2 = \frac{1}{2}(y_N + y_S) - y_{CC} \end{aligned} \tag{2.3}$$

Similarly, a fourth-order central differencing approximation at the cell centroid employs a quartic transformation. Five differencing points are taken in each direction, which are classified as W, MW, CC, ME and E in x-direction and S, MS, CC, MN and N in y-direction. The coefficients of the transformation are determined by substitution of the differencing points' coordinates into eqn. (2.2) in quartic form (n = 4) and the result in x-direction transformation is:

$$\begin{gathered} a_0 = x_{CC};\ a_1 = -\frac{1}{6}x_E + \frac{4}{3}x_{ME} - \frac{4}{3}x_{MW} + \frac{1}{6}x_W; \\ a_2 = -\frac{1}{6}x_E + \frac{8}{3}x_{ME} - 5x_{CC} + \frac{8}{3}x_{MW} - \frac{1}{6}x_W; \\ a_3 = \frac{2}{3}x_E - \frac{4}{3}x_{ME} + \frac{4}{3}x_{MW} - \frac{2}{3}x_W; \\ a_4 = \frac{2}{3}x_E - \frac{8}{3}x_{ME} + 4x_{CC} - \frac{8}{3}x_{MW} + \frac{2}{3}x_W; \end{gathered} \tag{2.4}$$

Coefficients in y-direction have corresponding pattern with E replaced by N and W by S, which gives,

$$\begin{gathered} b_0 = y_{CC};\ b_1 = -\frac{1}{6}y_N + \frac{4}{3}y_{MN} - \frac{4}{3}y_{MS} + \frac{1}{6}y_S; \\ b_2 = -\frac{1}{6}y_N + \frac{8}{3}y_{MN} - 5y_{CC} + \frac{8}{3}y_{MS} - \frac{1}{6}y_S; \\ b_3 = \frac{2}{3}y_N - \frac{4}{3}y_{MN} + \frac{4}{3}y_{MS} - \frac{2}{3}y_S; \\ b_4 = \frac{2}{3}y_N - \frac{8}{3}y_{MN} + 4y_{CC} - \frac{8}{3}y_{MS} + \frac{2}{3}y_S; \end{gathered} \tag{2.5}$$

2.4 Evaluating Differencing Points

A system of equations is to be developed from the finite difference formulation of the governing PDEs. This involves imposing boundary and initial conditions on the discretised domain. In addition to parameter inputs, the values of the dependent variables at the differencing points will be needed during the calculations. These values are expressed in terms of neighbouring nodal values or cell centroid values, or a combination of both. The locations of the differencing points for each cell are determined first. There are three different possibilities where a differencing point might lies: at an interior edge, coinciding exactly at an interior node, or at a boundary edge or boundary node. To determine a differencing point condition, the boundary and the interior domain are identified first. If a differencing point lies at a Dirichlet boundary, it is evaluated directly from the specified boundary value. On the other hand, differencing points that lie on a Neumann boundary are evaluated by the Neumann boundary point approximation scheme, which will be discussed in section 2.7. For differencing points located in the interior domain, the value of the dependent variable is expressed in terms of nodal and centroid values based on an approximation scheme (discussed in section 2.4.1), if it is located on a line segment. If a differencing point coincides at an interior node, then the nodal value is assigned to the differencing point. The following figure illustrates the process of evaluating a differencing point:

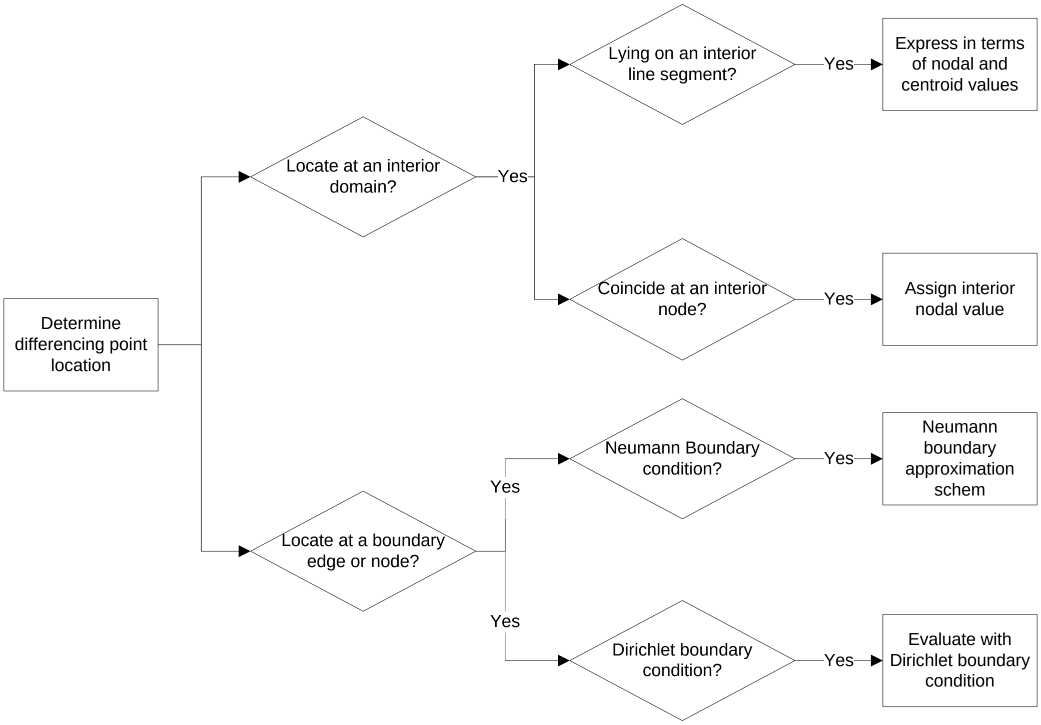

Determine differencing point location
Locate at an interior domain?
Yes
Lying on an interior line segment?
Yes
Express in terms of nodal and centroid values
Coincide at an interior node?
Yes
Assign interior nodal value
Locate at a boundary edge or node?
Yes
Neumann Boundary condition?
Yes
Neumann boundary approximation schem
Yes
Dirichlet boundary condition?
Yes
Evaluate with Dirichlet boundary condition

Figure 2.4 Process of evaluating a differencing point

Parametric equations are used to determine if a differencing point lies on a line segment. In Fig. 2.5, the relationship between a differencing point α and a line segment AB can be expressed as

$$\begin{bmatrix} x_\alpha \\ y_\alpha \end{bmatrix} = (1-t)\begin{bmatrix} x_A \\ y_A \end{bmatrix} + t\begin{bmatrix} x_B \\ y_B \end{bmatrix}$$

$$\begin{cases} \alpha \text{ is at Node A if } t = 0 \\ \alpha \text{ is at Node B if } t = 1 \\ \alpha \text{ is at line segment AB if } 0 < t < 1 \\ \alpha \text{ is not on line segment AB otherwise} \end{cases} \qquad (2.6)$$

Figure 2.5 Parametric representation of a line segment AB

where α represents any differencing point in any cell, and $L_{A\alpha}$ and $L_{B\alpha}$ are distances from α to A and B, respectively. Each differencing point in the interior domain is checked using these parametric equations to determine its location relative to the nodes and edges of the cell being considered, and its value is evaluated accordingly based on the process chart, Fig 2.4 above.

2.4.1 Approximation Scheme at Interior Line Segment

If a differencing point lies at an interior line segment, it must be expressed in terms of nodal and centroid values based on some approximation scheme. The approximation can be solely confined within the cell or accompanied with effect from the neighbouring cell. Several approximation schemes are illustrated in the following sections.

2.4.1.1 End Node Weighted Average Approximation

The value at a differencing point in a finite stencil can be evaluated by a weighted average of the two end nodal values on an edge where the differencing point lies. In this approximation scheme, values at differencing points are solely determined within the cell. The general formula for this approximation scheme can be represented as a piecewise step function in the following:

$$\emptyset_\alpha = \begin{cases} \dfrac{\dfrac{\emptyset_A}{L_{A\alpha}} + \dfrac{\emptyset_B}{L_{B\alpha}}}{\dfrac{1}{L_{A\alpha}} + \dfrac{1}{L_{B\alpha}}} & if\ 0 < t < 1 \\ \emptyset_A & if\ t = 0 \\ \emptyset_B & if\ t = 1 \end{cases} \tag{2.7}$$

where *t* is determined from parametric equations (2.6) This approximation using the length-weighted average of the two end nodal values on the line segment is first order accurate. In this case, the entire computation is confined within the cell. However, neighbouring effects from boundary conditions and/or adjacent cells have no direct influence in approximating differencing point values. Thus, the inaccuracy in this approximation scheme has more significant effect when a discontinuity arises or large gradient takes place. The results of the approximation will be discussed with examples in later sections.

2.4.1.2 Approximation by Interior Triangular Interpolation Function

To perform approximation within the cell, a three-point triangular interpolation function can be developed to evaluate differencing point values. The method is taken from the finite element numerical method in approximating a functional value at an arbitrary point within a cell, using the following equation [10]

$$\emptyset(x, y) = \sum_{i=1}^{n} \emptyset_i(x, y)\, N_i(x, y) \tag{2.8}$$

where the variable $\emptyset_i's$ are the nodal values and the functions $N_i(x, y)$ are referred to as shape functions that depend on the geometry of a cell. In the first-order triangular element, the shape functions are determined as follows

$$\begin{gathered} N_i(x, y) = \frac{1}{2A}(\alpha_i + \beta_i x + \gamma_i y) \\ \alpha_i = x_j y_k - x_k y_j \\ \beta_i = y_j - y_k \\ \gamma_i = x_k - x_j \end{gathered} \tag{2.9}$$

where (i,j,k) is the cyclic permutation of (1,2,3) for the nodal index of a triangle and $i \neq j \neq k$, and A is the area of the triangle. In this approximation scheme, differencing point value at a cell boundary (i.e. lying on an interior line segment) has the same value as computed by the length-weighted average of end nodal values in the previous approximation scheme discussed in section 2.4.1.1. Nevertheless, function values at points that lie in the interior region of the cell generally have better approximation than the length-weighted average scheme. This method is especially beneficial to approximate higher-order derivative terms or higher-order accurate approximations for low order derivatives that involve differencing points in the interior region of the cell.

2.4.1.3 Centroid and Nodal Weighted Average Approximation

In order to consider the neighbouring effect in approximating the differencing point values, both inscribed and adjacent cell centroid values are taken into account in addition to the two end nodal values. Location of the differencing point on a line segment is determined by the parametric equation (2.6). Each interior line segment will have two cells attached to it and both cell centroid values are used in the approximation of the differencing point value. The approximation scheme is illustrated in the following figure for a differencing point S lying on a line segment AB.

Figure 2.6 Differencing point approximation

Figure 2.6 considers the south differencing point S for ΔABC. The south point lies on the line segment AB and ΔABD is the adjacent triangle with the common edge AB. In this approximation scheme, four distances are required; distances to the two end nodes L_{AS} and L_{BS} as well as the distances to the two cell centroids $L_{CC1\text{-}S}$ and $L_{CC2\text{-}S}$. A similar length-weighted average formula as eqn. (2.7) is used to approximate the function value at the differencing point lying on an interior line segment:

$$\emptyset_{\alpha} = \frac{\frac{\emptyset_A}{L_{A\alpha}} + \frac{\emptyset_B}{L_{B\alpha}} + \frac{\emptyset_{CC1}}{L_{CC1-S}} + \frac{\emptyset_{CC2}}{L_{CC2-S}}}{\frac{1}{L_{A\alpha}} + \frac{1}{L_{B\alpha}} + \frac{1}{L_{CC1-S}} + \frac{1}{L_{CC2-S}}} \tag{2.10}$$

Recall that if the differencing point locates at a node instead of on the line segment, the nodal value is assigned to the differencing point. In this approximation scheme, the adjacent centroid value is included in the evaluation of the differencing point to account for the neighbouring cell effect. In general, such an approximation scheme is more accurate than the two end node weighted average scheme, as demonstrated in later examples.

2.4.1.4 Approximation by Quadrilateral Interpolation Function

For better accuracy to approximate a differencing point value, a quadrilateral interpolation function can be constructed by taking the two edge end nodes, inscribed centroid and adjacent centroid as vertices. This approximation scheme is illustrated in Fig. 2.7.

Figure 2.7 Quadrilateral interpolation scheme

In this figure, the south differencing point of ΔABC is taken as an example. The quadrilateral constructed to evaluate the south point involves two end nodes, A and B, as well as two centroids CC1 and CC2. The value at the differencing point is determined from equation (2.8) as a linear combination of interpolation functions. Because of the arbitrary shape, an isoparametric transformation is required to map the arbitrary quadrilateral element to a unit square master element with origin at the centroid, which is illustrated in Fig. 2.8.

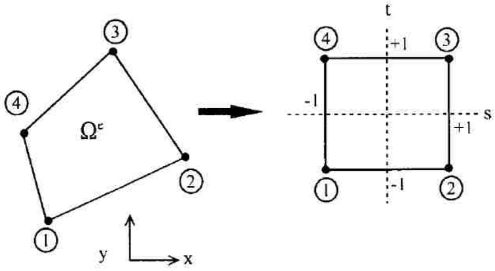


Figure 2.8 Four nodal quadrilateral element and master element [10]

The interpolation functions can be simply constructed in the master element as:

$$N_1(s,t) = \frac{1}{4}(s-1)(t-1);\ \ N_2(s,t) = \frac{1}{4}(s+1)(1-t)$$
$$N_3(s,t) = \frac{1}{4}(s+1)(t+1);\ N_4(s,t) = \frac{1}{4}(1-s)(t+1) \qquad (2.11)$$

The idea behind the mapping is to transform (*x*, *y*) domain into (*s*, *t*) master domain by the following equations,

$$x = x(s,t) = \sum_{i=1}^{4} x_i N_i(s,t)\,; \qquad y = x(s,t) = \sum_{i=1}^{4} x_i N_i(s,t) \qquad (2.12)$$

Given the coordinates (*x*, *y*) of the differencing point, new coordinates (*s*, *t*) in the master domain are required to be determined from equation (2.12). The system of equations is nonlinear, involving two equations and two unknowns. Extensive computation is needed to calculate the new coordinate (*s*, *t*) of the differencing point. Thus, this approximation scheme will not be investigated in the current research. Once having the differencing point coordinates determined in the master element, the differencing point value can be determined in a similar way as eqn. (2.8):

$$\emptyset(s,t) = \sum_{i=1}^{4} \emptyset_i N_i(s,t) \qquad (2.13)$$

2.4.2 Differencing Points Evaluated at a Dirichlet Boundary

The values of the dependent variables at points that lie on a boundary on which Dirichlet conditions are prescribed are given as functional values in terms of their coordinates.

Dirichlet boundary conditions may represent streamfunction or velocity values in fluid mechanics or temperature distribution in heat transfer. In solid mechanics, zero Dirichlet boundary condition can be interpreted as the boundary being clamped without displacement under loading conditions. If a differencing point lies at a Dirichlet boundary, it is directly evaluated from the Dirichlet boundary condition based on its position.

2.4.3 Differencing Points Evaluated at a Neumann Boundary

The Neumann boundary condition involves specification of the value of the first derivative at the boundary. In thermal-fluid applications, Neumann boundary conditions represent flux across the boundary, while it represents distributing load in solid mechanics.

[redacted]

Figure 2.9 Normal gradient in Neumann boundary bondition

In Fig. 2.9, a Neumann boundary condition in a right triangle is taken to illustrate the procedure. Normal gradients, $\partial f/\partial x$ and $\partial f/\partial y$ on the two edges are included in the consideration since the Neumann boundaries are along one of the Cartesian axes. The normal gradient on the hypotenuse, on the other hand, involves components in *x*- and *y*-direction and the formulation for this kind of Neumann boundary condition is not included in this thesis.

For Neumann boundaries aligned with the Cartesian axis, the function value at the differencing point can be approximated by first-order or second-order one-sided finite difference approximation schemes.

2.4.3.1 First-Order Backward Differencing Approximation

To illustrate the procedure, suppose the east differencing point lies on a Neumann boundary. As a simple approximation, the first derivative can be evaluated by a first-order backward difference formula, which can be expressed in terms of the backward adjacent differencing point in the finite stencil. The first-order backward difference formula in *x*-direction is

$$\left.\frac{\partial f}{\partial x}\right|_{ij} = \frac{f_{ij} - f_{i-1,j}}{\Delta x} + O(\Delta x) \tag{2.14}$$

Since the Neumann boundary condition is along the *x*-axis, the normal gradient $\partial f/\partial x$ can be expressed as a function of y, $g_1(y)$. Then the value for f_{ij} can be written as

$$f_{ij} \approx g_1(x, y) * \Delta x + f_{i-1,j} \tag{2.15}$$

Note that $f_{i\text{-}1,\mathrm{j}}$ represents the backward differencing point value. In a second-order central FD stencil, $f_{i\text{-}1,\mathrm{j}}$ can be defined as the centroid value, while $f_{i\text{-}1,\mathrm{j}}$ represents the adjacent mid differencing point in a fourth-order central FD stencil. Similar formulae can be used to approximate the normal gradient to the *y*-direction by substituting $\partial f/\partial y = g_2(x)$ into the backward finite difference equation.

2.4.3.2 Second-Order Backward Differencing Approximation

For a more accurate approximation, a second-order three-point backward finite differencing scheme can be used to approximate the value at differencing points lying on a Neumann boundary. The second-order backward differencing formula in *x*-direction is

$$\left.\frac{\partial f}{\partial x}\right|_{ij} = \frac{3f_{ij} - 4f_{i-1,j} + f_{i-2,j}}{(\Delta x)^2} + O(\Delta x)^2 \tag{2.16}$$

Note that $f_{i\text{-}1,\mathrm{j}}$ and $f_{i\text{-}2,\mathrm{j}}$ represent the adjacent backward differencing point value. In the second-order central differencing scheme on the finite stencil, $f_{i-1,\mathrm{j}}$ represents the value at the cell centroid and $f_{i-2,\mathrm{j}}$ is the differencing point value intersecting at an interior edge along the differencing direction. In a fourth-order central differencing on the finite stencil, $f_{i-1,\mathrm{j}}$ represents the mid differencing point and $f_{i-2,\mathrm{j}}$ represents the cell centroid value along the differencing direction. The figure below illustrates the second-order and fourth-order central differencing within a cell.

[redacted] [redacted]

Figure 2.10 2nd order and 4th order finite difference scheme

Note that the spacing Δx between two neighbouring points are equal after the finite stencil polynomial transformation, which was mentioned in section 2.3. Given the normal gradient function value as $\partial f/\partial x = g_1(y)$, the equation (2.16) can be rearranged to solve for f_{ij} as the following

$$f_{ij} \approx \frac{g_1(y) * (\Delta x)^2 + 4f_{i-1,j} - f_{i-2,j}}{3} \tag{2.17}$$

For the differencing points lying on a Neumann boundary perpendicular to the *y*-axis, similar formulae apply. Generally, the second-order backward differencing approximation scheme is more accurate than the first-order approximation.

2.5 Central Finite Differencing at Cell Centroid

Once differencing points have been determined within each cell, the cell centroid value can be evaluated by applying the governing PDEs at the cell centroid and using an appropriate finite difference formula on the Cartesian system in the computation domain. The governing PDEs are rewritten in terms of computational coordinates (ζ, η) which are related to the physical coordinates (*x*, *y*) by the polynomial transformation discussed in section 2.3. The general second order PDEs in (1.2) can be rewritten in terms of ζ and η as:

$$\bar{A}\frac{\partial^2\emptyset}{\partial\zeta^2} + \bar{B}\frac{\partial^2\emptyset}{\partial\zeta\partial\eta} + \bar{C}\frac{\partial^2\emptyset}{\partial\eta^2} + \bar{D}\frac{\partial\emptyset}{\partial\zeta} + \bar{E}\frac{\partial\emptyset}{\partial\eta} + \bar{F}\emptyset + G = 0 \tag{2.18}$$

where the coefficients $\bar{A}, \bar{B},$ etc., depend on the metrics of the transformation. Other PDEs can be converted in a similar way to (ζ, η). Once the PDEs are converted into the computational coordinates (ζ, η), the central finite difference formula can be applied at each cell centroid with equal spacing between grid points. Depending on the approximation scheme employed for differencing points, the cell centroid values can be evaluated, but the values are solely dependent upon the neighbouring cell centroid values and the nodal values (the vertices of the triangular cell), which is expressed by the following relation:

$$\emptyset_{cc} = f(\emptyset_{Acc}s, \emptyset_V s) = f(\emptyset_{DP}s) \tag{2.19}$$

where $\emptyset_{Acc}s$ are all the associated adjacent cell centroid values of a specific triangle, $\emptyset_V s$ are the vertices values and $\emptyset_{DP}s$ are corresponding differencing point values. In particular, applying the second-order central finite difference scheme to a Poisson equation, the cell centroid values can be expressed as differencing point values in the following generalized equation,

$$\gamma_{cc}\emptyset_{cc} = \gamma_E\emptyset_E + \gamma_W\emptyset_W + \gamma_N\emptyset_N + \gamma_S\emptyset_S - rhs_{CC} \tag{2.20}$$

where rhs_{cc} is the right-hand-side of the Poisson equation evaluated at the cell centroid and $\gamma_{cc}, \gamma_E, \gamma_W, \gamma_N$ and γ_S are coefficients determined from the quadratic transformation, and are given by

$$\begin{gathered}\gamma_{CC} = \frac{8}{(x_E - x_W)^2} + \frac{8}{(y_N - y_S)^2} \\ \gamma_W = \frac{8(x_E - x_{CC})}{(x_E - x_W)^3};\ \gamma_E = \frac{8(x_{CC} - x_W)}{(x_E - x_W)^3} \\ \gamma_S = \frac{8(y_N - y_{CC})}{(y_N - y_S)^3};\ \gamma_N = \frac{8(y_{CC} - y_S)}{(y_N - y_S)^3}\end{gathered} \tag{2.21}$$

Similarly, a fourth-order central finite difference scheme at cell centroids yields the following generalized equation:

$$\begin{aligned}\gamma_{cc}\emptyset_{cc} = \gamma_E\emptyset_E + \gamma_{ME}\emptyset_{ME} + \gamma_W\emptyset_W + \gamma_{MW}\emptyset_{MW} + \gamma_N\emptyset_N \\ + \gamma_{MN}\emptyset_{MN} + \gamma_S\emptyset_S + \gamma_{MS}\emptyset_{MS} - rhs_{CC}\end{aligned} \tag{2.22}$$

The coefficients γ_i depended on the quartic transformation coefficients, the a_i's and b_i's given by eqns. (2.4) and (2.5). After some calculations, we obtain

$$\begin{gathered}\gamma_W = -\frac{1}{3}\left(\frac{1}{a_1^2} + \frac{a_2}{a_1^3}\right);\ \gamma_E = -\frac{1}{3}\left(\frac{1}{a_1^2} - \frac{a_2}{a_1^3}\right); \\ \gamma_S = -\frac{1}{3}\left(\frac{1}{b_1^2} + \frac{b_2}{b_1^3}\right);\ \gamma_N = -\frac{1}{3}\left(\frac{1}{b_1^2} - \frac{b_2}{b_1^3}\right); \\ \gamma_{MW} = \frac{1}{3}\left(\frac{16}{a_1^2} + \frac{8a_2}{a_1^3}\right);\ \gamma_{ME} = \frac{1}{3}\left(\frac{16}{a_1^2} - \frac{8a_2}{a_1^3}\right); \\ \gamma_{MS} = \frac{1}{3}\left(\frac{16}{b_1^2} + \frac{8b_2}{b_1^3}\right);\ \gamma_{MS} = \frac{1}{3}\left(\frac{16}{b_1^2} - \frac{8b_2}{b_1^3}\right); \gamma_{CC} = \frac{10}{a_1^2} + \frac{10}{b_1^2}\end{gathered} \tag{2.23}$$

2.6 Interior Nodes and Neumann Boundary Points Approximation

Having determined all cell centroid values as in the previous section, nodal values are to be evaluated next. Nodal points on a Dirichlet boundary have specific fixed values evaluated in terms of their coordinates. Nodal points in the interior domain and along a Neumann boundary can be evaluated by the length-weighted average approximation of all the neighbouring cell centroid values. The equation used to determine an interior or Neumann boundary nodal value is

$$\emptyset_i \approx (\sum_{k=1}^{N_i} \frac{\emptyset_k}{L_{ik}}) \ / \ (\sum_{k=1}^{N_i} \frac{1}{L_{ik}}) \qquad for\ all\ i \in I \cup NB \tag{2.24}$$

where $I \cup NB$ is the set of all interior and Neumann boundary nodes. The index *k* represents the cell centroid index of a cell attached to node *i* and N_i is the maximum number of cells attached to node *i*. Alternatively, for a structured mesh, Neumann boundary points can be evaluated by the first-order or second-order one-sided finite difference formulae as in eqn. (2.15) or (2.17).

2.7 Development of System of Equations

A system of linear algebraic equations can now be developed to solve the governing PDEs by combining the generalized cell-centred equations (2.20) or (2.22) and the nodal value equations (2.24).

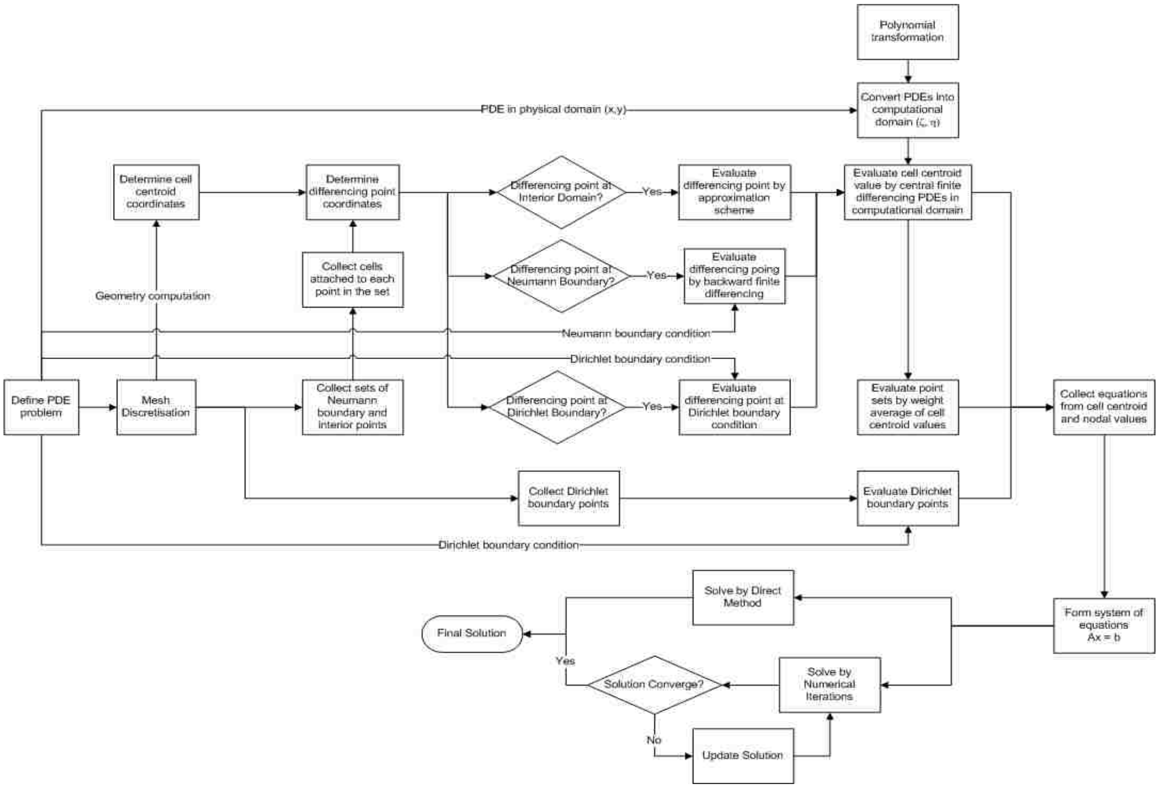
Suppose a mesh contains M triangular cells and N nodes (including interior and boundary nodes). A system of N+M equations can be assembled and expressed in matrix form as:

$$A\vec{\phi} = \vec{b} \tag{2.25}$$

where *A* is the coefficient matrix, $\vec{b}$ is a column vector containing the constant values in each equation (boundary values and *rhs*), and $\vec{\phi} = [\phi_{CC,1}, \cdots, \phi_{CC,N}, \ \phi_{N,1}, \cdots, \ \phi_{N,M}]^T$ is a variable vector containing all cell centroid and nodal values to be determined. The column vectors of the coefficient matrix are linearly independent in *N*+*M* dimensional space and therefore $\vec{\phi}$ has a unique solution from the system of equations (2.25). Either a direct method or iterative method can be used to solve such a system of equations.

2.8 Summary of CCFD Formulation

Sections 2.2 – 2.6 outline the derivation procedures for developing the system of equations and the flow chart in Fig. 2.11 summarizes the development process.



Polynomial transformation
Convert PDEs into computational domain (ζ, η)
PDE in physical domain (x,y)
Determine cell centroid coordinates
Determine differencing point coordinates
Differencing point at Interior Domain?
Yes
Evaluate differencing point by approximation scheme
Evaluate cell centroid value by central finite differencing PDEs in computational domain
Collect cells attached to each point in the set
Differencing point at Neumann Boundary?
Yes
Evaluate differencing poing by backward finite differencing
Geometry computation
Neumann boundary condition
Dirichlet boundary condition
Define PDE problem
Mesh Discretisation
Collect sets of Neumann boundary and interior points
Differencing point at Dirichlet Boundary?
Yes
Evaluate differencing point at Dirichlet boundary condition
Evaluate point sets by weight average of cell centroid values
Collect equations from cell centroid and nodal values
Collect Dirichlet boundary points
Evaluate Dirichlet boundary points
Dirichlet boundary condition
Solve by Direct Method
Final Solution
Yes
Solution Converge?
Solve by Numerical Iterations
Form system of equations Ax = b
No
Update Solution

Figure 2.11 Process of forming system of equations by CCFD scheme

The system of equations consists of equations for cell centroid values and equations for nodal values. Cell centroid values are illustrated along the top branch in Fig. 2.11 and are derived from finite difference approximations of the PDEs being solved. Nodal values are shown along the bottom two branches with Dirichlet boundary points and the combined set of interior and Neumann boundary points. The dependent variables at Dirichlet boundary points are evaluated from the Dirichlet boundary conditions defined by the PDE problem. In comparison, function values at the set of interior and Neumann boundary points are determined by the length-weighted average of surrounding cell centroid values. Once both cell centroid and nodal equations have been derived, the system of equations is developed by formulating the coefficient matrix A and constant column vector $\vec{b}$ in the matrix form. The next chapter introduces a preliminary overview of some topics that are relevant to the CCFD formulation.

CHAPTER 3 – PRELIMINARY OVERVIEW

3.1 Mesh Overview

Numerical methods based on the spatial subdivision of a domain into polyhedra imply the need to generate a mesh. The mesh topology being studied in this thesis is based on a triangulated domain created by Delaunay triangulation, which maximizes the minimum angle of each triangle in the domain. Compared with the advancing front triangulation technique, Delaunay triangulation generally yields a better quality mesh due to two main reasons [4]:

1. Iteratively checks the discrepancy between the desired and actual element shape and size of the current mesh.
2. Points are introduced into the regions where the discrepancy exceeds a user-defined tolerance.

Delaunay triangulation in a plane ensures that the circumcircle associated with each triangle contains no other mesh points in its interior domain [11]. Note that triangular elements may create problems in structural loading applications, such as plane stress problems. One problem is the geometric modeling of curved edges. The surface of a model with a large curvature may appear reasonably modeled, whereas the surface of a hole is poorly modeled. A second problem is that the strains in various regions of the actual structure may be changing rapidly and the constant strain element will only provide an approximation of the average strain at the centre of the element. For example, loading in a nutshell will have poor approximation result when using triangular elements. This problem can be solved either by increasing the number of elements (i.e. mesh density), or alternatively replacing the triangular element with a better element, such as an eight-noded quadrilateral [12].

3.1.1 Mesh Quality

The mesh quality can be evaluated with respect to two parameters: skewness and aspect ratio. Skewness of a cell can be determined by the deviation from a normalized angle as [13]

$$Skewness = max\left[\frac{\theta_{max} - \theta_e}{180 - \theta_e}, \frac{\theta_e - \theta_{min}}{\theta_e}\right] \tag{3.1}$$

where $\theta_e = \frac{Sum\ of\ internal\ polyhedral\ angles}{\#\ of\ sides\ in\ polyhedron}$ is the normalized angle of a cell (i.e. 60^o for tetrahedral and triangular elements and 90^o for quadrilateral and hexagonal elements). Equation (3.1) implies that good cell quality results when skewness is close to 0 and bad cell quality results when skewness is close to 1. For example, perfect cells with zero skewness are equilateral triangles in a triangulated domain and rectangles in a quadrilateral domain. As a general rule for acceptable mesh quality, skewness of hexagonal, triangular and quadrilateral elements should not exceed 0.8.

Aspect ratio on the other hand is defined as the ratio of the longest side to the shortest side of a mesh element. For an acceptable range of the aspect ratio, it should not be greater than a value of 40, but the range can vary based on the characteristics of a physical problem. By combining the effects of skewness and aspect ratio, the quality of a triangle can be measured by the following equation [5]

$$q = \frac{4A\sqrt{3}}{h_1^2 + h_2^2 + h_3^2} \tag{3.2}$$

where A is the triangle area and h_1, h_2 and h_3 are the side lengths of the triangle. If q > 0.6, the triangle is of acceptable quality. Note that an equilateral triangle has perfect quality with $q = 1$ when $h_1 = h_2 = h_{3.}$

3.1.2 Mesh Classification

Mesh types can be classified according to the following categories [4]:

1. Conformality
2. Surface or body alignment
3. Topology
4. Element Type

In principle, any of the four classifications can be combined randomly. However, only a few main combinations are normally considered in the discretisation, which are as follows:

a) Multiblock grids: Conformal, surface-aligned, macro-unstructured and micro-structured grids consisting of quadrilaterals or bricks.
b) Adaptive Cartesian grids: Non-conformal, non-surface-aligned, micro-unstructured grids consisting of quadrilaterals or bricks.
c) Unstructured uniform-element grids: Conformal, surface-aligned, micro-unstructured grids consisting of triangles or tetrahedra.

In this thesis, an unstructured uniform-element mesh is implemented in solving physical problems by the Cell-Centred Finite Difference scheme and other types of numerical methodologies.

3.1.2.1 Conformality

Conformal meshes are characterized by continuous neighbouring elements across all edges and faces [4]. Non-conforming meshes exhibit edges and faces that do not match perfectly between neighbouring elements, which results in hanging nodes or overlapped zones, as illustrated in Fig. 3.1.

Figure 3.1 Non-conformal mesh

Note in Fig. 3.1, that node A is a hanging node and face F1 becomes an overlapped zone since the quadrilateral element does not match with the triangular elements at node A.

3.1.2.2 Surface or Body Alignment

Surface or body alignment refers to boundary faces matching exactly with grid points in the surface domain [4]. If faces are crossed by the surface, the mesh is referred to as being non-aligned. The following figures illustrated examples of surface aligned and non-surface aligned meshes.

[redacted] [redacted]

Figure 3.2 Examples of surface aligned and non-surface aligned meshes

Note that edges crossing the curve γ will create a node in the surface aligned mesh, but not in the non-surface aligned mesh.

3.1.2.3 Mesh Topology

Mesh topology refers to the structure or order of the elements [4]. Three types of mesh topology are:

1. Micro-structured: each interior node has the same number of neighbours.
2. Micro-unstructured: each interior node may have a different number of neighbours.
3. Macro-unstructured, micro-structured: also refer as a staggered grid system. Mesh is assembled from groups of micro-structured subgrids.

The following figure shows the three types of mesh topology.

[redacted] [redacted] [redacted]

Figure 3.3 Micro-structured, micro-unstructured and macro-unstructured, micro-structured meshes

Note that each interior node in the micro-structured mesh at the left of Fig. 3.3 has six neighbouring cells and degrees. Interior nodes in the micro-unstructured mesh in the middle figure have an arbitrary number of neighbouring cells and degrees. The figure on the right is macro-unstructured overall, but locally micro-structured.

3.1.2.4 Mesh Element

The mesh element describes the polyhedra used to discretize space [4]. Typical element types include triangles and quadrilaterals for 2D domains and tetrahedral, prisms and bricks for 3D domains.

3.2 Solution of System of Equations

Once the system of equations has been developed on the discretised mesh, there are two basic approaches to the solution of such systems: directly and iterative. Some methods of direct approach are Gaussian elimination, Crout and Cholesky lower-upper matrix decomposition, while some iterative approaches are Jacobi, Gauss-Seidel and Successive Relaxation methods. Both approaches are discussed in the following sections.

3.2.1 Direct Solvers

Direct solvers require large storage space for memory and large computational power if the system of equations becomes very large. However, the rapid increase in computer memory and improvement in CPU power have led to a revival of direct solvers.

Gauss elimination is the classic direct solver. The key idea of this method is to perform elementary row operations to reduce the coefficient matrix in the system to an upper triangular matrix. Then by back substitution or further row operations, the solution is obtained. The main disadvantage of this method is that it requires a large number of computations to perform elementary row operations.

Crout's method is an alternative direct approach. In this method, the coefficient matrix is decomposed into an upper and lower triangulation portion, i.e. $A\vec{\Phi} = LU\vec{\Phi} = \vec{b}$. Once the decomposition is complete, the system is solved first by forward substitution $L\vec{v} = \vec{b}$ and followed by backward substitution $U\vec{\Phi} = \vec{v}$.

The basic concept of the direct solver involves inversion of the coefficient matrix A to solve the system, i.e. $\vec{\Phi} = A^{-1}\vec{b}$. The Gauss elimination and Crout elimination methods are used extensively in structural finite element analysis.

3.2.2 Iterative Solvers

Compared to direct solvers involving matrix inversion, iterative procedures for solving a system of equations require much less computational power and memory storage. Iterative approaches are simple and easy to program. An initial solution is assigned first and new values are computed based on an iterative formulation. Then the procedure is repeated until a specified convergence criterion has been reached. Various formulations of the iterative approach based on the Cell-Centred Finite Difference numerical methodology are explored in this thesis, including Jacobi, Gauss-Seidel and Successive Over-Relaxation (SOR) methods, which are discussed in the following sections.

3.2.2.1 Jacobi Iterative Method

In the Jacobi iterative method, current evaluations at the finite differencing points are done using the cell centroid values and nodal values from the previous iteration. Then, the new cell centroid values are computed from the finite difference equations approximating the PDEs, such as the central finite difference formula mentioned in section 2.1, followed by determining the nodal values in the mesh using the weighted average of the neighbouring cell centre values, as in eqn. (2.24). Suppose that a second-order central finite difference scheme is implemented at the cell centres and differencing points are evaluated by centroid-nodal weighted average mentioned in section 2.4.1.3. Then, eqns. (2.10), (2.20) and (2.24) can be rewritten based on the Jacobi iterative approach as:

$$\emptyset_{\alpha}^{k+1} = \frac{\dfrac{\emptyset_A^k}{L_{A\alpha}} + \dfrac{\emptyset_B^k}{L_{B\alpha}} + \dfrac{\emptyset_{CC1}^k}{L_{CC1-S}} + \dfrac{\emptyset_{CC2}^k}{L_{CC2-S}}}{\dfrac{1}{L_{A\alpha}} + \dfrac{1}{L_{B\alpha}} + \dfrac{1}{L_{CC1-S}} + \dfrac{1}{L_{CC2-S}}} \tag{3.3}$$

$$\gamma_{CC}\emptyset_{CC}^{k+1} = \gamma_E\emptyset_E^k + \gamma_W\emptyset_W^k + \gamma_N\emptyset_N^k + \gamma_S\emptyset_S^k - rhs_{cc} \tag{3.4}$$

$$\emptyset_i^{k+1} \approx \left(\sum_{m=1}^{N_i} \frac{\emptyset_{CC,m}^{k+1}}{L_{im}}\right) \Big/ \left(\sum_{m=1}^{N_i} \frac{1}{L_{im}}\right) \; for\ all\ i \in I \cup NB \tag{3.5}$$

The superscript *k*+1 indicates the current iteration and the updated iteration value is determined based on previous iteration values. The Jacobi method is classified as a cell-based iterative scheme, in which all cell centroid values are computed before updating the nodal values in the mesh. This iterative method is illustrated as follows:

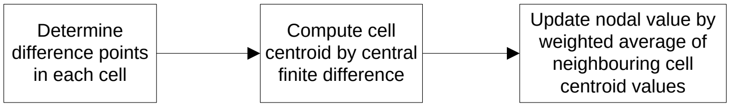

Determine difference points in each cell
Compute cell centroid by central finite difference
Update nodal value by weighted average of neighbouring cell centroid values

Figure 3.4 Iteration process of Jacobi method

Usually, the Jacobi iterative method requires the most number of iterations to arrive at the final solution, but it is generally more stable than other types of iterative methods.

3.2.2.2 Gauss-Seidel Iterative Method

The Gauss-Seidel iterative method implemented in the CCFD numerical methodology is classified as a nodal-based scheme. The Gauss-Seidel iterative process starts at one interior or Neumann boundary node by determining its neighbouring cells. For each neighbouring cell, differencing points are approximated using the updated nodal and cell centroid values in the current iteration if available, otherwise the values in the previous iteration are used. In this case, eqn. (3.3) for evaluating differencing points can be rewritten as:

$$\emptyset_{\alpha}^{k+1} = \frac{\dfrac{\emptyset_{A}^{k+1,\ k}}{L_{A\alpha}} + \dfrac{\emptyset_{B}^{k+1,\ k}}{L_{B\alpha}} + \dfrac{\emptyset_{CC1}^{k+1,\ k}}{L_{CC1-S}} + \dfrac{\emptyset_{CC2}^{k+1,\ k}}{L_{CC2-S}}}{\dfrac{1}{L_{A\alpha}} + \dfrac{1}{L_{B\alpha}} + \dfrac{1}{L_{CC1-S}} + \dfrac{1}{L_{CC2-S}}} \tag{3.6}$$

The superscript “*k*+1, *k*” indicates use of the updated value if available, or otherwise take the previous iteration value. The cell centre values and the nodal value are computed as in the Jacobi iterative method by eqn. (3.4) and (3.5). Once the current nodal value is updated, the process repeats for the next node, until all the values of interior and Neumann boundary nodes are updated in the domain, completing one iteration. The flow chart in Fig 3.5 further illustrates this iteration process.

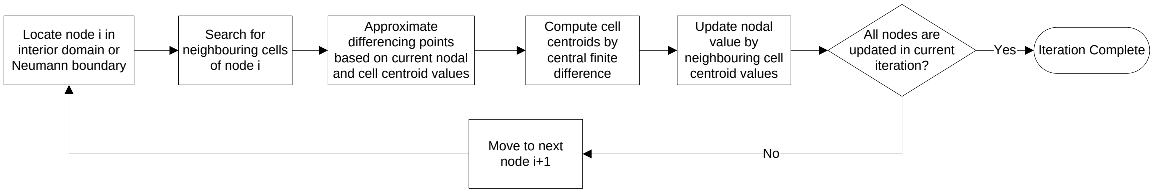

Locate node i in interior domain or Neumann boundary
Search for neighbouring cells of node i
Approximate differencing points based on current nodal and cell centroid values
Compute cell centroids by central finite difference
Update nodal value by neighbouring cell centroid values
All nodes are updated in current iteration?
Yes
Iteration Complete
No
Move to next node i+1

Figure 3.5 Iteration process of Gauss-Seidel method

Normally, the Gauss-Seidel iterative method performs better than the Jacobi method, having a faster convergence rate with less number of iterations, because Gauss-Seidel method implements updated nodal and cell centroid values in approximating differencing points. Notice that the method is convergent if the largest elements are located in the main diagonal of the coefficient matrix in the system of equations [2], which imposes a more restricted convergence requirement. Thus, the solution obtained by the Gauss-Seidel method may be less stable than the solution from the Jacobi method.

3.2.2.3 Successive Over-Relaxation (SOR) Iterative Method

The Successive Over-Relaxation (SOR) iterative method is a convergence accelerated version of the Gauss-Seidel iterative method. The iteration process is exactly the same as the Gauss-Seidel method. During the solution process in the SOR method, the solution procedure is accelerated by changing the direction of the updated solution. This is achieved by imposing a relaxation parameter ω on the previous nodal value in eqn. (3.5) to update the current nodal value. In SOR, the nodal value can be determined by the following equation:

$$\emptyset_i^{k+1} \approx \emptyset_i^k + \omega\left(\left(\sum_{m=1}^{N_i} \frac{\emptyset_{CC,m}^{k+1}}{L_{im}}\right) \Big/ \left(\sum_{m=1}^{N_i} \frac{1}{L_{im}}\right) - \emptyset_i^k\right) \, for\ all\ i \in I \cup NB \qquad (3.7)$$

Note that when 0 < ω < 1, the method is classified as under-relaxation. When ω > 1, it is classified as over-relaxation. When ω = 1, the successive relaxation method becomes the Gauss-Seidel method. By numerical experimentation, the value of the relaxation parameter can be estimated for optimal convergence rate. Differencing points are determined as in the Gauss-Seidel method by eqn. (3.6), and cell centroid values are computed by the finite difference approximation of the PDEs.

The direct approach by Gaussian elimination and the three iterative methods introduced above will be studied in detail in one example in a later section. Solutions will be checked in a simple mesh topology against the analytical solution.

3.3 Manufactured Solutions

Roache [14] argues that the method of manufactured solutions (MMS) provides a general procedure for generating solutions and produces a strong method for numerical algorithm code verifications. To solve a partial differential equation with corresponding initial and boundary conditions, the goal is to determine the analytical solution, consistent with these given conditions. In contrast, the method of manufactured solutions is used to determine the consistent boundary and initial conditions by substituting predefined analytical solutions into the partial differential equation. Schwer [15] commented that subsequent grid refinement will show an eventual convergence to the exact solution by most of the numerical methods. However, the direct use of the traditional error measures, such as relative error (RE), is inadequate and the MMS usually requires a check of mesh sensitivity for the effectiveness of the MMS between two grid structures, obtained from the following equation:

$$p = \frac{1}{ln\gamma} \ln \left(\frac{E_{grid,1}}{E_{grid,2}} \right) \tag{3.8}$$

where γ is the refinement ratio, $E_{grid,1}$ and $E_{grid,2}$ are global errors obtained from the numerical solutions to the manufactured test problem at two mesh refinements. The quantity *p* defined in the above equation is referred to as the order of accuracy (OA). For constant volume cells, i.e. uniform grids, the normalized global error is also the root mean square error (RMSE).

3.4 Interpolation Line Post-Processing

To perform post-processing of the numerical solution, one may want to examine the solution along a line passing through the physical domain. There is no guarantee that such a line contains nodes in the mesh, especially for an unstructured mesh. In this thesis, such lines are constructed using piecewise cubic Hermite interpolation within the solution domain. The numerical solution along a line is compared with analytical or other numerical method solutions. The procedure for this post-processing is described as follows:

1. Collect interpolation points at the intersections between the interpolation line and the cell edges. This includes any points at which the interpolation line coincides

with interior nodes. The following figure shows the interpolation points taken for post-processing, denoted by triangular symbols:

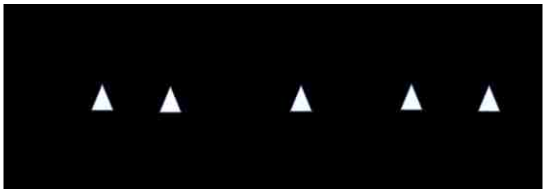


Figure 3.6 Interpolation points for post-processing

2. Determine value of the dependent variable at each interpolation point. Nodal value is assigned to an interpolation point if it coincides with a node in the mesh. Otherwise, for interpolation points lying on a line segment AB, the end nodal length-weighted average formula (2.7) is used to approximate its value, which is shown as follows:

$$\emptyset_{IP} = \frac{\frac{\emptyset_A}{L_{A,IP}} + \frac{\emptyset_B}{L_{B,IP}}}{\frac{1}{L_{A,IP}} + \frac{1}{L_{B,IP}}} \tag{3.9}$$

   where subscript *IP* stands for interpolation point.

3. Determine nodal relative error (NRE) and absolute error (AE) at each interpolation point by comparing with analytical solution or solutions from other numerical simulations. The NRE and AE are determined from

$$NRE = \begin{cases} \left|\frac{\emptyset_{e,i} - \emptyset_i}{\emptyset_{e,i}}\right|; & for\ \emptyset_{e,i} \neq 0 \\ 0; & for\ \emptyset_{e,i} = 0 \end{cases} \tag{3.10}$$

$$AE = \left|\emptyset_{e,i} - \emptyset_i\right| \tag{3.11}$$

   where N is number of interpolation points, $\emptyset_{e,i}$ is the exact nodal value and $\emptyset_i$ represents the computed nodal value. Note that the equation for relative error excludes the interpolation having zero values.

4. Compute NRE and AE at midpoint in each subinterval among the interpolation points by piecewise cubic Hermite interpolation. Hermite interpolation is classified as a second-order accurate approximation [11].

5. Piecewise cubic Hermite interpolation lines are drawn based on computed values of NRE and AE at the midpoint in each interpolated subinterval.

3.5 Assessment Criteria

The assessment criteria for the accuracy of the numerical solution can be divided into two categories, global and local assessments. Global measures include overall relative error (RE) and root mean square error (RMSE), which are defined as the average of the nodal relative error and absolute error given in eqn. (3.10) and (3.11), respectively. These can be expressed as

$$RE = \frac{1}{N} \sum_{i=1, \emptyset_{e,i} \neq 0}^{N} \left| \frac{\emptyset_{e,i} - \emptyset_i}{\emptyset_{e,i}} \right| \tag{3.12}$$

$$RMSE = \frac{1}{N} \sqrt{\sum_{i=1}^{N} \left(\emptyset_{e,i} - \emptyset_i\right)^2} \tag{3.13}$$

Through mesh refinement, accuracy of numerical solutions usually improve and the degree of improvement can be evaluated by the order of accuracy (OA) in eqn. (3.8). Local scale assessments on the other hand, include the nodal relative error (NRE) and absolute error (AE) as discuss in section 3.4. The local assessments measure the error distribution in the solution domain and can be illustrated in error contour plots. Besides the global and local assessments, computational performance can be evaluated through number of iterations for convergence and computational time.

3.6 Variations of CCFD Numerical Method

Several approximation schemes for evaluating differencing points and different approaches to solve the resulting system of equations have been discussed in previous sections. Variations of the CCFD method can be studied with different combinations based on choice of approximation scheme for differencing points, order of central finite difference schemes and direct or iterative solution approaches. A few of these variations will be studied in the “benchmarked” examples in the following chapter. Beside the variations of the methodology, assessment criteria discussed in the previous section are used to measure the performance of the numerical schemes.

CHAPTER 4 – APPLICATIONS TO PDES GOVERNING PHYSICAL PHENOMENA

4.1 Overview of Benchmarked Examples

The new CCFD methodology is applied to several "benchmark" problems representing different physical applications, covering the some typical PDEs and boundary conditions. To validate this new methodology, the solutions obtained will be compared to available exact solutions or results from other numerical simulations. The following table shows the different application problems to be solved:

Ex. #	PDE	Boundary Condition Type	Domain	Solution for Comparison
1	Laplace	Discontinuous, Dirichlet	Unit Square	Exact, FEM
2	Poisson	Continuous, Dirichlet	Rectangle	Exact, FEM
3	Poisson	Continuous, Dirichlet	Unit Disk	Exact, FEM
4	Laplace	Piecewise continuous, Dirichlet and Neumann	Rectangle	N/A
5	Laplace	Discontinuous, Dirichlet and Neumann	Hexagon	FEM
6	Diffusion - Convection	Continuous, Dirichlet	Unit Square	Exact, FEM
7	Unsteady Diffusion	Discontinuous, Dirichlet	Unit Square	Exact

Table 4.1 Application examples list

At least two meshes are employed to measure the mesh sensitivity of each problem. These meshes include a relatively coarse mesh, a finer mesh and one with improved cell quality. A clustered mesh is also investigated in Example #5. Assessment criteria on the performance of the CCFD method include nodal and overall relative error (RE) and root mean square error (RMSE) by comparing CCFD results with analytical or other numerical simulation results. Also, mesh sensitivity is studied via order of accuracy (OA) by determining the root mean square error ratio in coarse and refined meshes. In addition, convergence rate is also investigated among the iterative approaches of the CCFD method, measured by the number of iterations required for convergence and overall computational time.

Note that the convergence tolerance is set at 1.0e^{-6}. The average relative difference (ARD) at each iteration is determined by the average of the relative difference at all cell centroid values between current iteration n and previous iteration n – 1, from the following equation:

$$ARD = \frac{1}{M}\sum_{i=1}^{M}\left|\frac{\emptyset_{CC,i}^{n} - \emptyset_{CC,i}^{n-1}}{\emptyset_{CC,i}^{n}}\right| \tag{4.1}$$

The iteration will stop when the *ARD* reaches the specified tolerance.

4.1.1 Unit Square Discretisation

Two types of meshes in the unit square domain are going to be studied, which apply to Examples #1, #6 and #7. A coarse mesh and a refined mesh are generated by Delaunay triangulation, which are shown in the following figures:

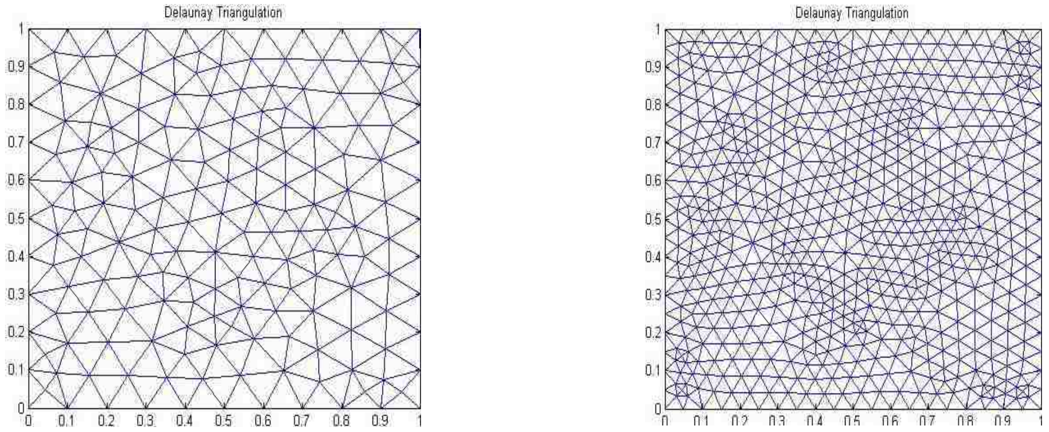


Figure 4.1 Coarse mesh and refined mesh

The coarse mesh has 185 nodes and 328 cells within the domain, while the refined mesh contains 697 nodes and 1312 cells with a refinement ratio of 4:1. The coarse mesh has 145 interior nodes and 40 boundary nodes, while the refined mesh contains 537 interior nodes and 160 boundary nodes. The refinement method is regular refinement, where all of the specified triangles are divided into four triangles of the same shape [5]. In addition, interior nodal positions have been adjusted to improve the quality of the mesh. The quality for each cell of the two meshes is measured by the eqn. (3.1) and is illustrated in the following contour figures.

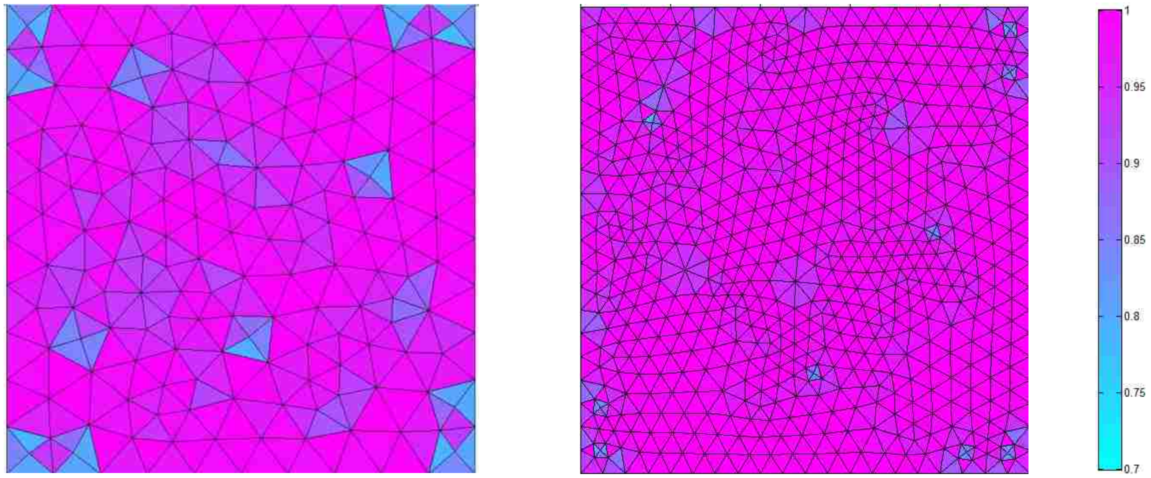
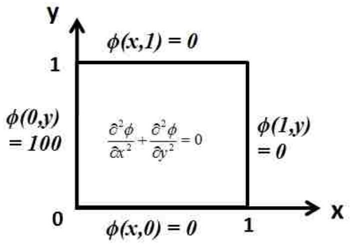


Figure 4.2 Quality of coarse and refined meshes

As seen from the above figure and legend at the side, perfect quality cells are shown in dark pink with aspect ratio of unity and zero skewness. Light blue colour indicates low quality cells that are only 70% as good as the perfect quality cells. In general, the refined grids with improved mesh quality will yield smoother solution contours and better solution accuracy and this can be measured by order of accuracy indicated in eqn. (3.8).

4.2 Example 1: Laplace Equation with Discontinuous Boundary Conditions

Laplace equation is an elliptic equation which models an equilibrium problem in steady-state. This model elliptic equation describes irrotational and incompressible flow in a fluid mechanics context and steady-state temperature distribution in heat transfer. Equation (4.2) and Fig. 4.3 define the boundary value problem in this example.


y
$\phi(x,1) = 0$
1
$\phi(0,y)$
$= 100$
$\frac{\partial^2\phi}{\partial x^2}+\frac{\partial^2\phi}{\partial y^2}=0$
$\phi(1,y)$
$= 0$
0
$\phi(x,0) = 0$
1
x

$$\begin{cases} \dfrac{\partial^2\emptyset}{\partial x^2} + \dfrac{\partial^2\emptyset}{\partial y^2} = 0; \;\; \forall(x,y) \in (0,1)\times(0,1) \\ \quad \emptyset(x,1) = \emptyset(1,y) = \emptyset(x,0) = 0 \\ \quad \emptyset(0,y) = 100 \end{cases} \tag{4.2}$$

Figure 4.3 Example 1 description

Dirichlet boundary conditions are given with discontinuities at the top left and bottom left corners. For the numerical method approach, average value of the two neighbouring boundary values is assigned at the discontinuities.

The analytical solution can be derived from superposition principle because of the linear property of the governing equation [16]. The complete analytical solution of this example is expressed as follows:

$$\emptyset(x,y) = \sum_{n=1}^{\infty} \sin\,(n\pi y)\left\{\frac{-200[(-1)^n - 1]}{n\pi}\operatorname{csch}(n\pi)\sinh[n\pi(1-x)]\right\} \tag{4.3}$$

The value of the exact solution is taken by the first 20 terms of the infinite series and the truncation error normally has value less than 1.0e^{-10}, which is negligible compared to the default tolerance 1.0e^{-6}. The analytical solution contour plot is shown in Fig. 4.4.

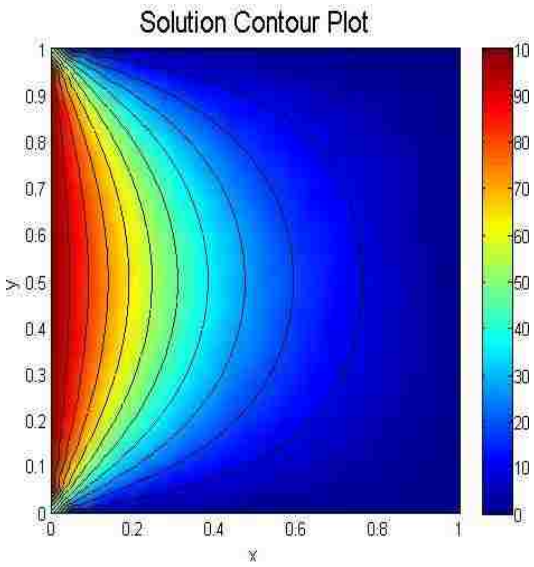


Figure 4.4 Example 1 – Analytical solution contour plot

For this example, the solutions from the numerical simulation are only slightly different from the analytical solution and therefore the difference between solution contour plots from numerical simulation and analytical solution are not noticeable (see below). Therefore, RE and AE contour plots are used to analyze the accuracy of the numerical simulation.

The problem is first solved directly by CCFD on two simple mesh structures, using centroid-nodal averaging to calculate the values at differencing points. Then, the problem is solved on two automatically generated meshes by FEM and CCFD, using a point-Jacobi algorithm. At the end, a successive over-relaxation (SOR) scheme is implemented, and the relaxation parameter ω is determined through numerical experimentation. The CCFD solutions are compared with the analytical solution to this boundary value problem.

4.2.1 Direct CCFD Approach

To demonstrate the CCFD procedure, two types of simple mesh structures are solved. First, the mesh is constructed with one interior node at the centre and the cell edges are generated by the diagonal of the unit square, as illustrated in Fig. 4.5. The solution at interior node P is to be determined.

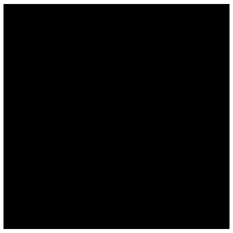


Figure 4.5 Example #1 – Mesh #1

The second-order central finite difference scheme is implemented at each cell centroid. After determining the quadratic mapping coefficients for each cell in the mesh, the coefficients in eqn. (2.20), computed from eqns. (2.21), become those shown in Table 4.2.

Cell #	γ_S	γ_N	γ_W	γ_E	γ_{CC}
1	9/8	9/8	8/3	4/3	25/4
2	8/3	4/3	9/8	9/8	25/4
3	9/8	9/8	4/3	8/3	25/4
4	4/3	8/3	9/8	9/8	25/4

Table 4.2 Coefficients of CCFD equations for mesh 1

The differencing points at each cell are approximated by the centroid-nodal weighted average scheme. By substitution of the coefficients from the above table, the system of equations obtained by applying central finite difference equation (2.20) for each cell, coupled with the weighted average formula (2.24) for the interior node, can be expressed as the matrix equation

$$\begin{bmatrix} 5.7041 & -0.2730 & 0 & -0.2730 & -1.7194 \\ -0.2730 & 5.7041 & -0.2730 & 0 & -1.7194 \\ 0 & -0.2730 & 5.7041 & -0.2730 & -1.7194 \\ -0.2730 & 0 & -0.2730 & 5.7041 & -1.7194 \\ 0.25 & 0.25 & 0.25 & 0.25 & -1 \end{bmatrix} \begin{bmatrix} \phi_{C1} \\ \phi_{C2} \\ \phi_{C3} \\ \phi_{C4} \\ \phi_{P} \end{bmatrix} = \begin{bmatrix} 305.2706 \\ 19.3019 \\ 0 \\ 19.3019 \\ 0 \end{bmatrix} \tag{4.4}$$

Solving this system of equations by Gauss elimination gives

$$\begin{bmatrix} \phi_{C1} \\ \phi_{C2} \\ \phi_{C3} \\ \phi_{C4} \\ \phi_{P} \end{bmatrix}_{CCFD2} = \begin{bmatrix} 62.4257 \\ 14.3338 \\ 8.9079 \\ 14.3338 \\ 25.003 \end{bmatrix} \tag{4.5}$$

The analytical values at the cell centroids and interior node P, based on eqn. (4.3), are calculated to be

$$\begin{bmatrix} \phi_{C1} \\ \phi_{C2} \\ \phi_{C3} \\ \phi_{C4} \\ \phi_{P} \end{bmatrix}_{Exact} = \begin{bmatrix} 67.8319 \\ 13.0718 \\ 6.0243 \\ 13.0718 \\ 25 \end{bmatrix} \tag{4.6}$$

showing a relative error of 0.012% at node P.

The second mesh is similar to the first, but includes an additional boundary node at the bottom edge, which is illustrated in Fig. 4.6. This mesh represents partial refinement of mesh 1.

Figure 4.6 Example #1 – Mesh 2

The coefficients in the central finite difference approximation of the Laplace equation, i.e. eqn. (2.20), determined from eqns. (2.21), are given in Table 4.3.

Cell #	γ_S	γ_N	γ_W	γ_E	γ_P
1	9/8	9/8	8/3	4/3	25/4
2	9/8	9/8	4/3	8/3	25/4
3	4/3	8/3	9/8	9/8	25/4
4	9/2	9/2	9/2	9/2	18
5	9/2	9/2	9/2	9/2	9/2

Table 4.3 Coefficients of CCFD equations for mesh 2

The resulting system of equations for the centroids and interior node values can be expressed in the following matrix equation:

$$\begin{bmatrix} 5.8271 & 0 & -0.2032 & 0 & -0.4393 & -1.6324 \\ 0 & 5.7574 & -0.2730 & -0.4393 & 0 & -1.6817 \\ -0.2730 & -0.2730 & 5.7041 & 0 & 0 & -1.7194 \\ 0 & -2.0283 & 0 & 13.3311 & -1.286 & -2.4138 \\ -2.0283 & 0 & 0 & -1.286 & 13.3311 & -6.9138 \\ 0.1883 & 0.1883 & 0.1883 & 0.2175 & 0.2175 & -1 \end{bmatrix} \begin{bmatrix} \phi_{C1} \\ \phi_{C2} \\ \phi_{C3} \\ \phi_{C4} \\ \phi_{C5} \\ \phi_P \end{bmatrix} = \begin{bmatrix} 296.5704 \\ 0 \\ 19.3019 \\ 0 \\ 90.8734 \\ 0 \end{bmatrix} \tag{4.7}$$

Solving the above system by Gaussian elimination, the result turns out as

$$\begin{bmatrix} \phi_{C1} \\ \phi_{C2} \\ \phi_{C3} \\ \phi_{C4} \\ \phi_{C5} \\ \phi_P \end{bmatrix}_{CCFD2} = \begin{bmatrix} 60.1646 \\ 8.1718 \\ 13.7600 \\ 8.3081 \\ 28.9971 \\ 23.5726 \end{bmatrix} \tag{4.8}$$

The analytical values at the cell centroids and interior node P are computed as

$$\begin{bmatrix} \phi_{C1} \\ \phi_{C2} \\ \phi_{C3} \\ \phi_{C4} \\ \phi_{C5} \\ \phi_P \end{bmatrix}_{Exact} = \begin{bmatrix} 67.8319 \\ 6.0243 \\ 13.0718 \\ 6.9666 \\ 23.9373 \\ 25 \end{bmatrix}. \tag{4.9}$$

The interior nodal value predicted by the CCFD method has a 5.7% error compared with the analytical value. The major sources of error of the CCFD computational values on these two meshes are mainly due to the coarse discretisation of the solution domain. In the next sections, auto-generated grids are studied by a FEM direct solver and iterative CCFD approaches.

4.2.2 FEM Solution

The two mesh topologies in Fig. 4.1 are solved using different numerical methods. In this section the given problem in eqn. (4.1) is solved by the finite element method from a built-in functional toolbox in MATLAB [11]. Figure 4.7 illustrates the local RE contour plots by comparing the FEM results with the analytical solution.

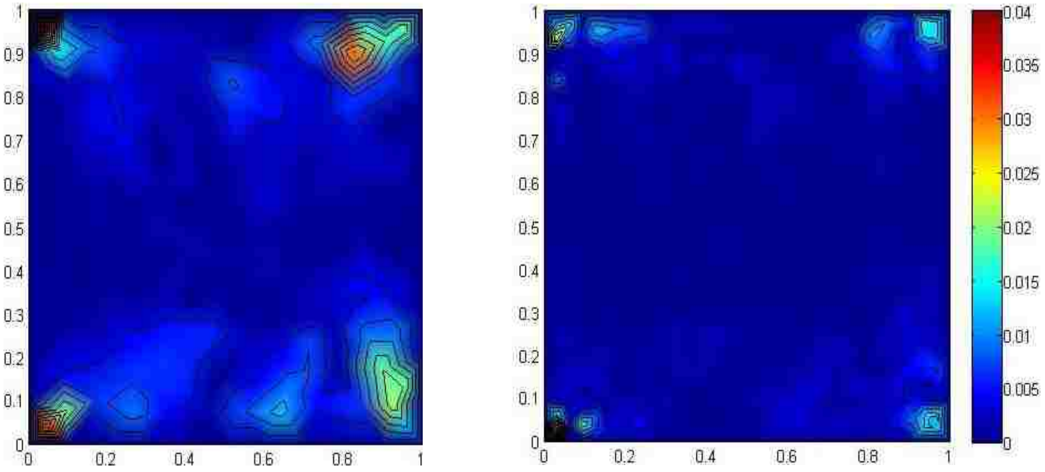


Figure 4.7 Example 1 – FEM RE contour plots on coarse and refined meshes

Both RE contour plots show a symmetric pattern with respect to the midsection horizontal line at y = 0.5, with the highest errors of 4% on the coarse grid and 3.5% on the refined grid occuring at the corners where the discontinuities are located. The refined grid shows lower overall RE than the coarse grid. The following table shows the general FEM results of the two grids.

Mesh	Relative Error	RMS Error	Order of Accuracy
Coarse	0.43%	0.0212	1.03
Refined	0.14%	0.0051	

Table 4.4 FEM results for Example 1

The RE on the refined mesh improves about three times compared with the RE on the coarse mesh, while the RMSE reduces about 4 times at a value of 0.0051 on the refined mesh. Calculating the solution improvement due to the given refinement ratio of 4 using eqn. (3.8), the order of accuracy is about 1.03.

4.2.3 Numerical Solution by CCFD Method

Using the same mesh topologies as above, the Laplace equation (4.2) is now solved by the CCFD method, and the results are compared to the FEM solution and the analytical solution (4.3). The CCFD solution contours for both coarse and refine meshes are shown in Fig. 4.8.

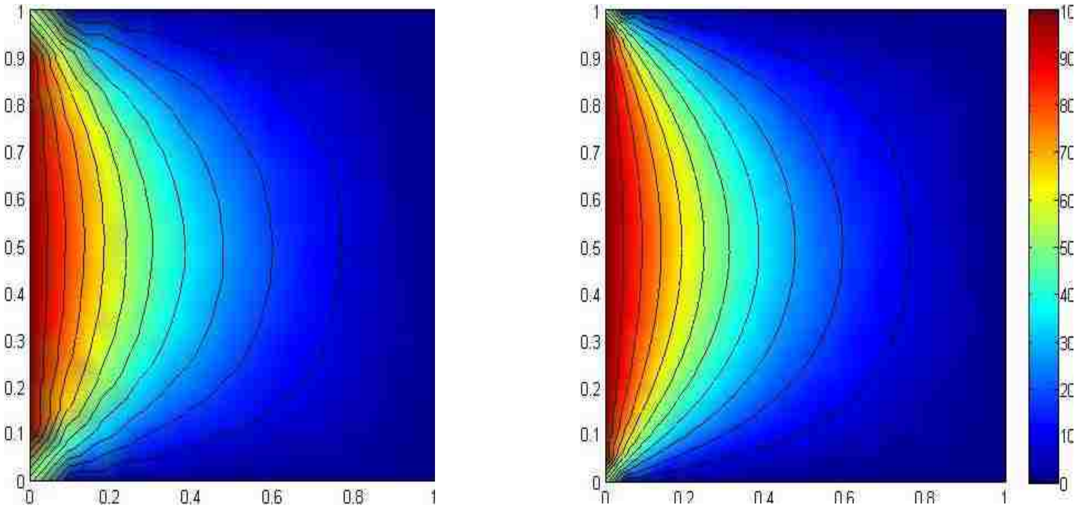


Figure 4.8 Example 1 – CCFD solution contour plots on coarse and refined meshes

The general performance of the solver is measured using the RE, RMSE, OA and number of iterations for convergence, which are shown in Table 4.5.

Mesh	RE	RMSE	OA	Iterations
Coarse	4.81%	0.0357	0.356	715
Refined	2.85%	0.0218		2238

Table 4.5 Performance of CCFD solver for Example 1

The solution on the refined mesh is more accurate than on the coarse mesh based on the lower values of RE and RMSE, but at a cost of larger number of iterations for solution convergence. The low value for order of accuracy (OA) indicates low accuracy improvement by mesh refinement. Figure 4.8 shows the nodal RE locations on the two meshes.

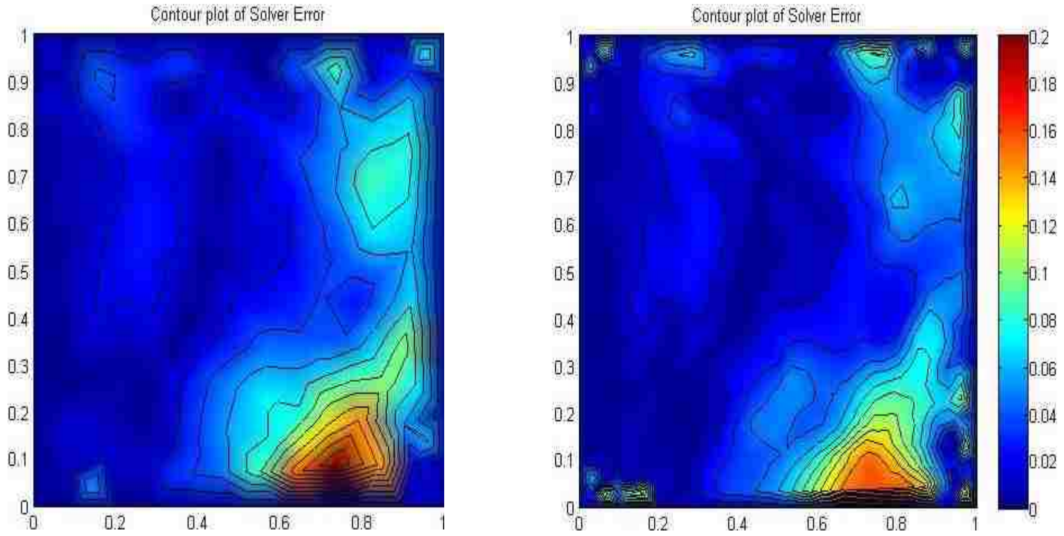


Figure 4.9 Example 1 – CCFD RE contour plots on coarse and refined meshes

The RE plots for both coarse and refined meshes show a similar contour pattern, with the highest RE region at the bottom right corner of the domain. The highest RE value improves from 20% for the coarse mesh solution to 16% for the refined mesh solution. Compared with the FEM solution, the CCFD solution has lower accuracy, which is indicated by higher RE and RMSE values. In addition, solution accuracy appears to be less sensitive to mesh refinement in the CCFD method than in the FEM method.

The solutions are interpolated along horizontal lines to compare the CCFD against FEM and exact solutions. The interpolation lines are drawn at the bottom (y = 0.05), middle (y = 0.5) and top (y = 0.95) region of the domain. Both sets of numerical solution values are interpolated by the procedures mentioned in section 3.4.

The bottom interpolations along y = 0.05 for computed solution values on both meshes are compared to the exact solution in Fig. 4.10. The interpolations at y = 0.5 and y = 0. 95 for the computed solution values on both meshes show a similar pattern as in the bottom interpolation, as illustrated in Figs. 4.11 and 4.12.

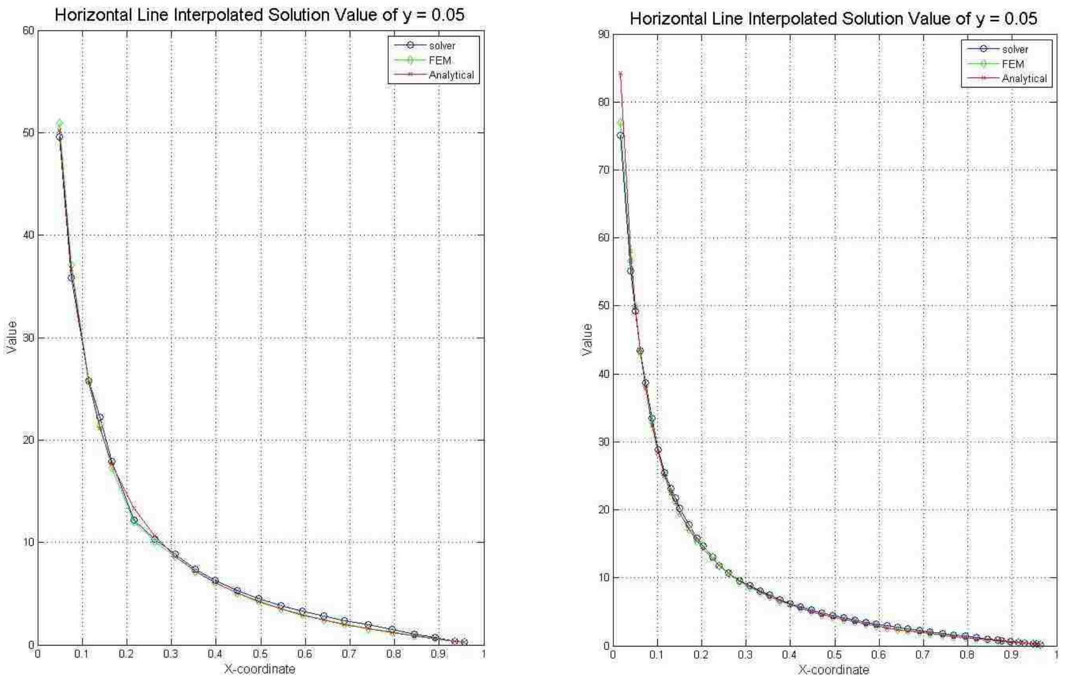


Figure 4.10 Example 1 – Comparison of exact, FE and CCFD solutions along y = 0.05 on coarse and refined meshes

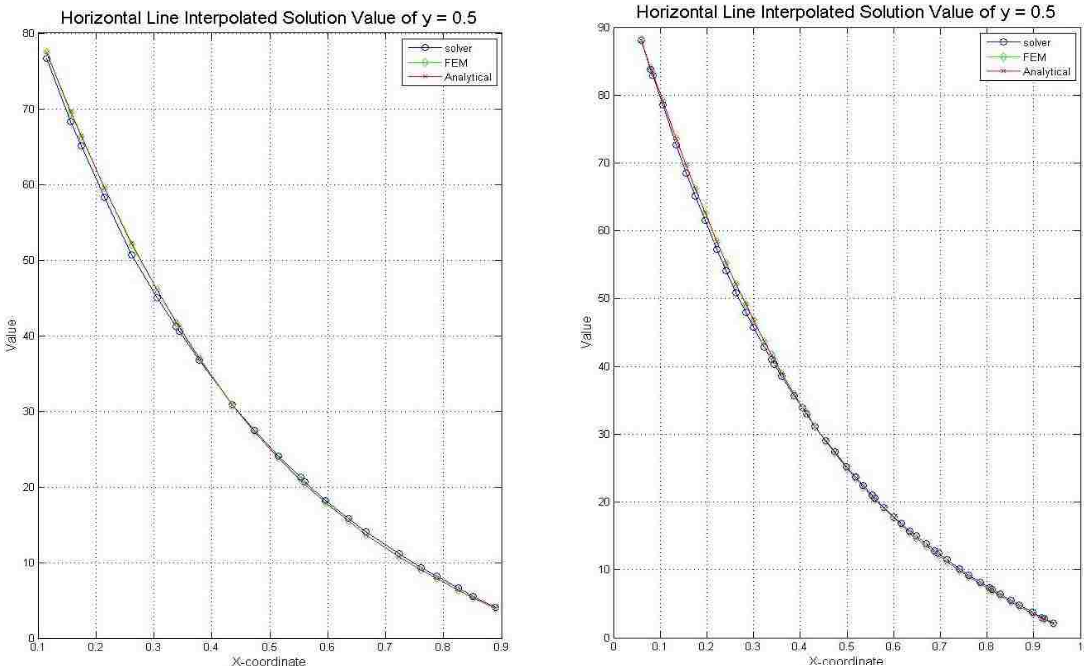


Figure 4.11 Example 1 – Comparison of exact, FE and CCFD solutions along y = 0.5 on coarse and refined meshes

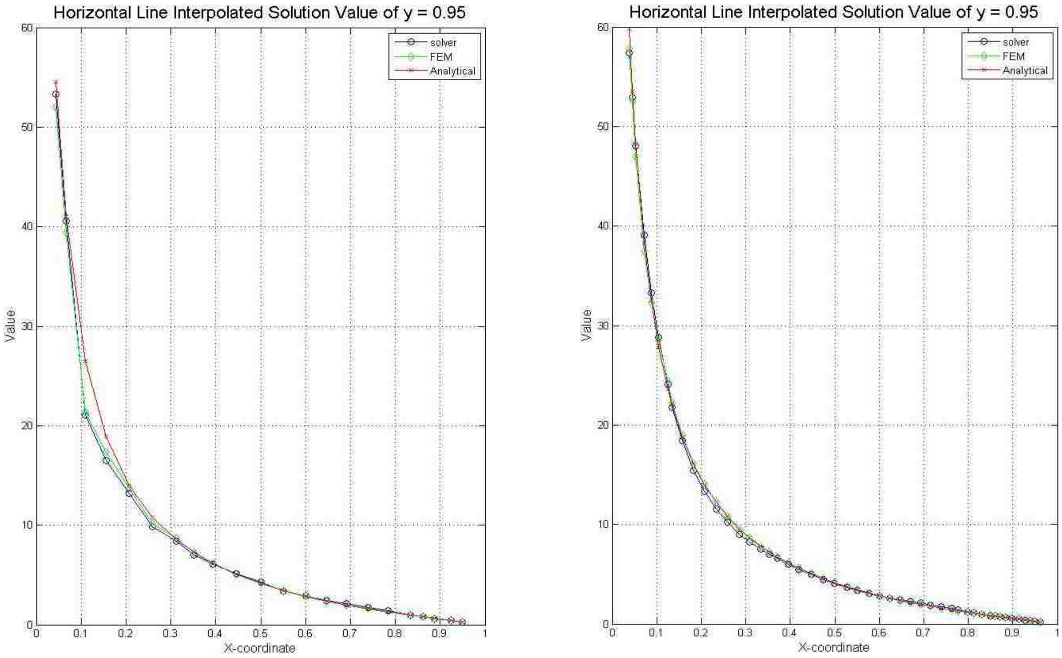


Figure 4.12 Example 1 – Comparison of exact, FE and CCFD solutions along y = 0.95 on coarse and refined meshes

These figures clearly show that the accuracy of the CCFD solution is comparable to that of the FE solution, and that the accuracy is improved with mesh refinement. The interpolated solution values on both meshes show a steep gradient in the leftmost region of the domain. The gradient gradually reduces as *x* increases, indicating that the solution changes more rapidly near the left boundary where the nonhomogeneous condition is applied, then falls gradually to zero at the right side of the domain.. Normally, the higher absolute errors occur in the high gradient region, while less absolute errors are observed in the low gradient region. This effect has been illustrated in Fig. 4.13, which displays the absolute relative error plots on both coarse and refined meshes. The largest absolute errors are observed at both discontinuity corners and in the high gradient region.

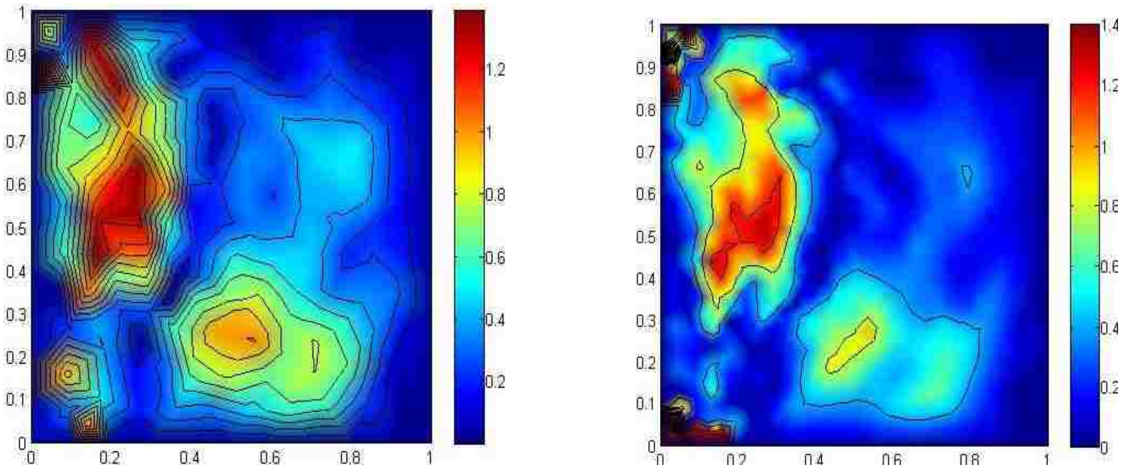


Figure 4.13 Example 1 – Absolute error contour plots on coarse and refined mesh

The CCFD formulation and the SOR iterative scheme are used to determine the optimal relaxation parameter ω which requires the least computational power, i.e. least number of iterations for convergence and computational time. The computational time is used to measure the entire computational process of the solver, excluding the preliminary geometry definition. Both coarse and refined mesh topologies in Figure 4.2 are studied. The solver implements a nodal advancing scheme like the Gauss-Seidel iterative approach instead of a cell advancing scheme as in the Jacobi method. In addition, the convergence rate is expected to further increase by coupling with a relaxation parameter ω to change direction of the next iterative solution. Overall RE and RMSE are measured to assess the solution accuracy, while number of iterations and computational time are used to determine the optimal relaxation factor. The results of numerical experiments on the coarse mesh are shown in Table 4.6

ω	**0.6**	**0.8**	**1.0**	**1.2**	**1.4**	**1.6**	**1.8**	**2.0**	**2.2**	**2.25**	**2.3**
RE	3.56%	3.57%	3.57%	3.57%	3.57%	3.57%	3.57%	3.57%	3.57%	3.57%	N/A
RMSE	0.0481	0.0481	0.0481	0.0481	0.0481	0.0481	0.0481	0.0481	0.0481	0.0481	N/A
Iterations	871	734	651	595	555	524	500	481	466	462	N/A
Time (s)	9.145	9.8705	6.4680	5.9575	5.6014	5.3833	5.1015	4.8299	4.6520	4.5342	N/A

Table 4.6 Effect of relaxation parameter (coarse mesh)

The solution remains the same by adjusting the relaxation parameter, which is consistent with the behaviour in other numerical formulations, as the REs and RMSEs remain unchanged. The numerical experiments reveal that under-relaxation requires more computational power than the over-relaxation iterative approach. The optimal value of the relaxation parameter is about 2.25 with 462 iterations and 4.5 seconds of computational time. For relaxation parameters greater than this value, i.e. ω > 2.25, CCFD iterations become divergent. Comparing the computational cost in the numerical experiments, the optimal relaxation parameter saves about half of the computational power required when ω = 0.6.

Similar numerical experiments were also conducted on the refined mesh with different relaxation parameter and the results are shown in Table 4.7.

ω	**0.4**	**0.6**	**0.8**	**1.0**	**1.2**	**1.4**	**1.6**	**1.8**	**2.0**	**2.2**	**2.25**	**2.3**
RE	2.76%	2.78%	2.80%	2.80%	2.81%	2.81%	2.81%	2.82%	2.82%	2.82%	2.82%	N/A
RMSE	0.0214	0.0214	0.0214	0.0214	0.0214	0.0214	0.0214	0.0214	0.0214	0.0214	0.0214	N/A
Iterations	3504	2695	2274	2015	1839	1712	1616	1540	1479	1429	1418	N/A
Time (s)	165.85	125.19	105.34	94.40	86.16	81.20	75.33	73.11	68.89	68.39	66.21	N/A

Table 4.7 Effect of relaxation parameter (refined mesh)

The optimal relaxation parameter for the refined mesh has exactly the same value as for the coarse mesh in Table 4.6. The convergence rate is improved about 2.5 times at optimal ω compared to the under-relaxed result at ω = 0.4. Nevertheless, a slight decrease of solution accuracy is the price paid for the improvement of convergence rate. The RE is increased from 2.76% at ω = 0.4 to 2.82% at optimal ω, while the RMSE remains unchanged. By comparing results from the two mesh topologies, more iterations and computational time are required to solve in the refined mesh than in the coarse mesh. However, solution accuracy is improved by mesh refinement, since the RE is reduced from 3.57% to 2.82% and RMSE decreases from 0.0481 on the coarse mesh to 0.0214 on the refined mesh.

From the results of numerical experiments on both meshes, the relaxation parameter has an optimal value at 2.25 for fastest convergence rate and least computational power. This optimal value is different from the value found in other numerical methods. For example, Hoffman [2] commented that the optimal relaxation parameter in traditional FD methods usually occurs between zero and two. Since the computational algorithm differs depending on numerical method, the optimal relaxation parameter may vary and usually must be determined by numerical experiments.

4.3 Example 2: Poisson Equation in a Rectangle with Dirichlet BC

The governing equation of this problem is given by Poisson equation with a variable sourcing term in a half unit square domain. The boundary is subjected to piecewise continuous functions. The problem is described by eqn. (4.10) and Fig. 4.14.

Figure 4.14 Example 2 description

$$\begin{cases} \dfrac{\partial^2 \emptyset}{\partial x^2} + \dfrac{\partial^2 \emptyset}{\partial y^2} = -\{2 + \pi^2 x(1-x)\}\cos(\pi y)\,; \\ \qquad \forall (x,y) \in (0,1) \times \left(0, \dfrac{1}{2}\right) \\ \emptyset(0,y) = \emptyset(1,y) = \emptyset\left(x, \dfrac{1}{2}\right) = 0; \\ \emptyset(x,0) = x(1-x) \end{cases} \tag{4.10}$$

The height of the rectangular domain is half of the length. All prescribed boundary conditions in the problem are derived from a manufactured solution by imposing a predefined analytical solution

$$\emptyset(x,y) = \ x(1-x)\cos\,(\pi y) \tag{4.11}$$

The analytical solution contour plot of the problem is shown in Fig. 4.15.

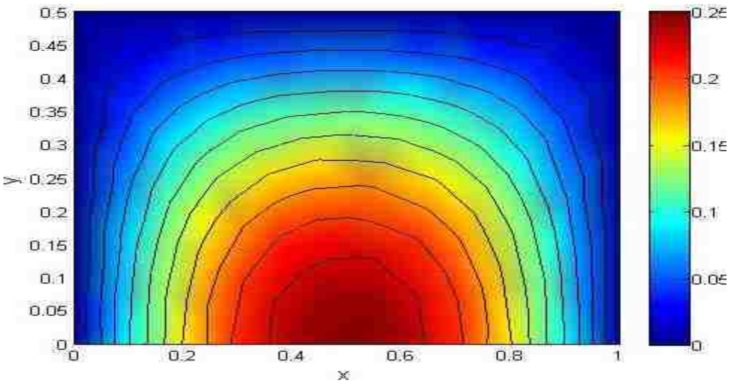


Figure 4.15 Example 2 - Contour plot of analytical solution

The solution domain is enclosed by zero boundary conditions at the left, top and right boundaries. The highest solution value is located at the centre of the lower boundary. Both FEM and CCFD numerical methods are used to solve this problem on two types of mesh topologies, coarse and refined as shown in Fig. 4.16.

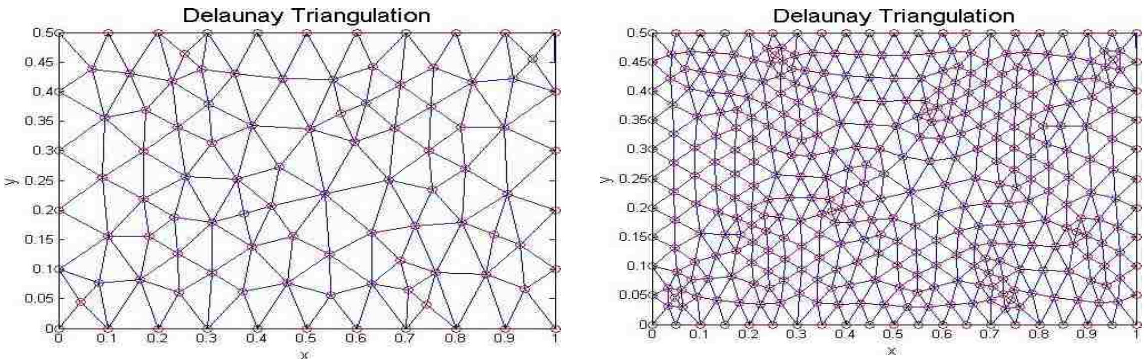


Figure 4.16 Example 2 - Mesh discretisation

The quality of both meshes is evaluated by eqn. (3.1) and displayed in Fig. 4.17.

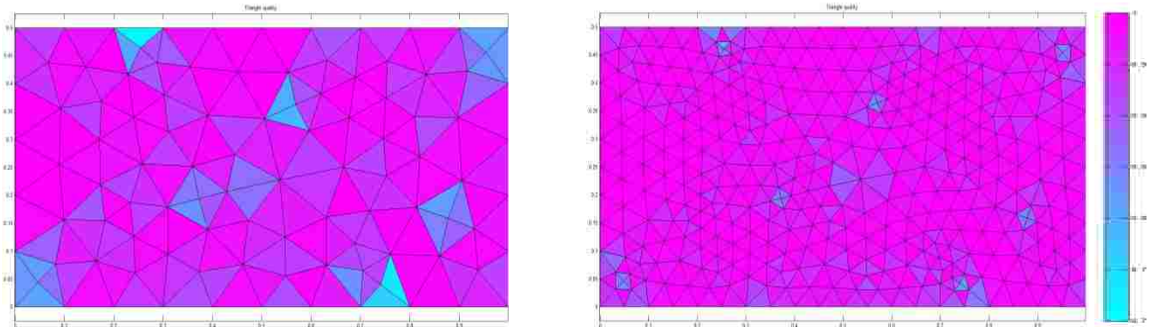


Figure 4.17 Example 2 - Mesh quality in coarse and refined discretisation

The coarse mesh at the left contains 30 boundary nodes, 70 interior nodes and 168 cells, while the refined mesh at the right of Fig. 4.16 contains 60 boundary nodes, 307 interior nodes and 672 cells. The mesh refinement ratio between these two mesh structures is 4:1. The numerical results obtained by the two methodologies are compared with the analytical solution.

4.3.1 Numerical Solution by FEM

The problem is first solved by the FEM approach and the numerical solution is compared with the analytical solution. The results are shown in Table 4.8.

Mesh	RE	RMSE	OA
Coarse	0.67%	7.6381e-05	1.6647
Refined	0.16%	7.5991e-06	

Table 4.8 Example 2 - General result by FEM solver

As the mesh refinement result indicates, solution accuracy is considerably improved. In the refined mesh result, the overall RE is reduced by ¾ and the overall RMSE is reduced by an order of magnitude compared to the coarse results. Figure 4.18 illustrates the RE contour plots for the FEM solution.

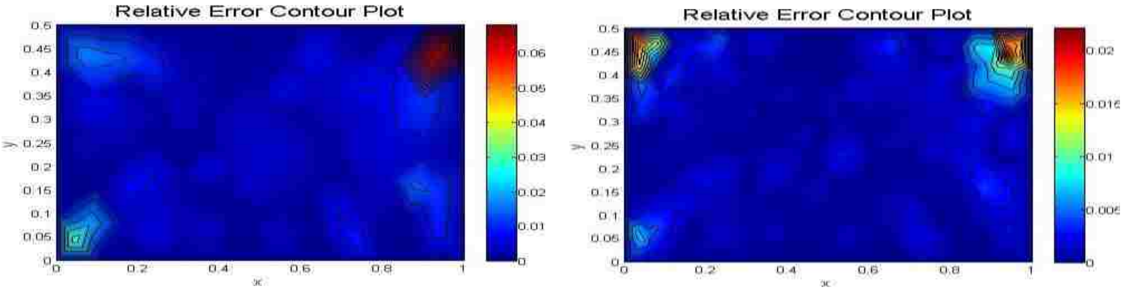


Figure 4.18 Example 2 - FEM RE contour plots on coarse and refined meshes

The RE contours show the error clustering at the top right corner of the domain, perhaps due to discretised error in the localized region with additional generated triangles. Improvement in solution accuracy by mesh refinement can also be verified by the legend scale in both contour plots.

4.3.2 CCFD Solution

The CCFD solution contours for both coarse and refined meshes are shown in Fig. 4.19. The differencing points have been evaluated using the centroid-nodal interpolation scheme.

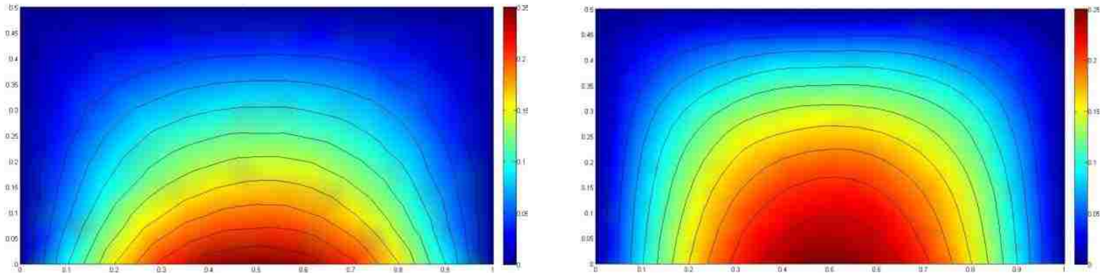


Figure 4.19 Example 2 – CCFD solution contour plots on both coarse and refined meshes

Mesh	RE	RMSE	OA	Iterations	Time (s)
Coarse	4.12%	6.4491e-04	1.219	318	0.3988
Refined	2.01%	1.1898e-04		1037	5.8697

Table 4.9 Example 2 - General result by CCFD solver

The results shown in Table 4.9 indicate that mesh refinement produces more improvement in solution accuracy than the results in the previous example, given by the higher order of accuracy. The refined mesh results show half of the overall RE and a quarter of the overall RMSE as the coarse mesh results. Figure 4.20 illustrates the RE contour plots in the domain.

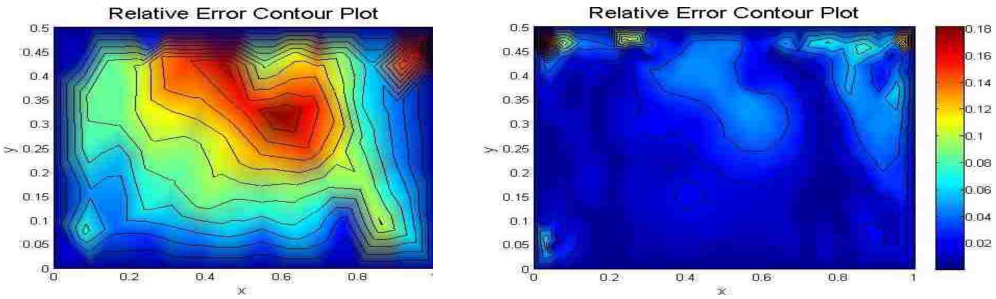


Figure 4.20 Example 2 – CCFD RE contour plots on coarse and refined meshes

The RE in the coarse mesh spreads out over the top half region, while the RE in the refined mesh clusters only at the top corners. The observations from the RE contour plots also verifies the solution accuracy improvement by mesh refinement.

In comparison, the absolute error contour plots on both coarse and refined meshes are shown in Fig. 4.21.

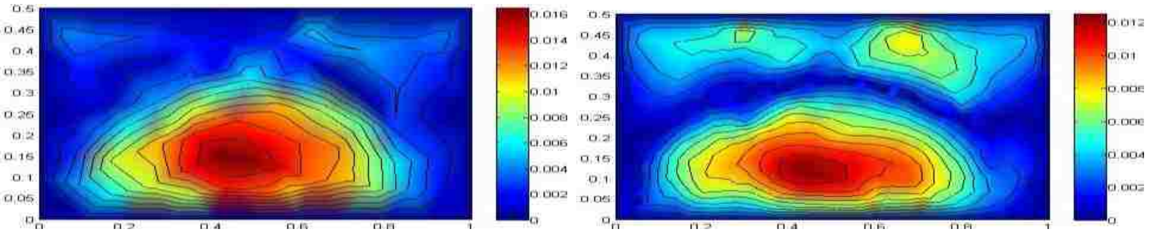


Figure 4.21 Example 2 – AE contour plots in both coarse and refined meshes

The AE contour plots on both the coarse and refined meshes exhibit a symmetric pattern across the vertical midsection $x = 0.5$. High AE values cluster at the middle region of the lower boundary where the solution values and gradients are highest.

4.4 Example 3: Poisson Equation in a Disk with Dirichlet Conditions

This example describes heat transfer in a unit disk with a constant source term [5]. The governing equation of the problem is the Poisson equation which models the heat diffusion. Zero Dirichlet conditions are prescribed at the boundary of the unit circle. The problem is described by eqn (4.12) and illustrated in Fig. 4.22.

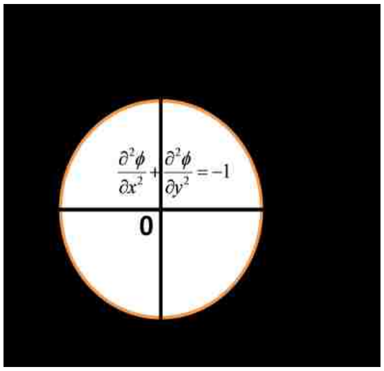

$\frac{\partial^2\phi}{\partial x^2}+\frac{\partial^2\phi}{\partial y^2}=-1$
0

Figure 4.22 Example 3 description

$$\begin{cases} \dfrac{\partial^2\emptyset}{\partial x^2}+\dfrac{\partial^2\emptyset}{\partial y^2}=-1;\ \ \forall(x,y)\in\Omega \\ \emptyset(x,y)=0;\ \ \forall(x,y)\in\partial\Omega \end{cases} \tag{4.12}$$

where Ω is interior domain of the unit disk.

The exact solution of this BVP is

$$\emptyset(x,y)=\frac{1-x^2-y^2}{4} \tag{4.13}$$

The accuracy of the FEM and CCFD numerical solutions can be evaluated for different meshes by comparison to the exact solution. The domain is discretised by an unstructured mesh, so the traditional FD method is unable to handle this type of mesh topology. The meshes generated within the interior domain are shown in Fig. 4.23.

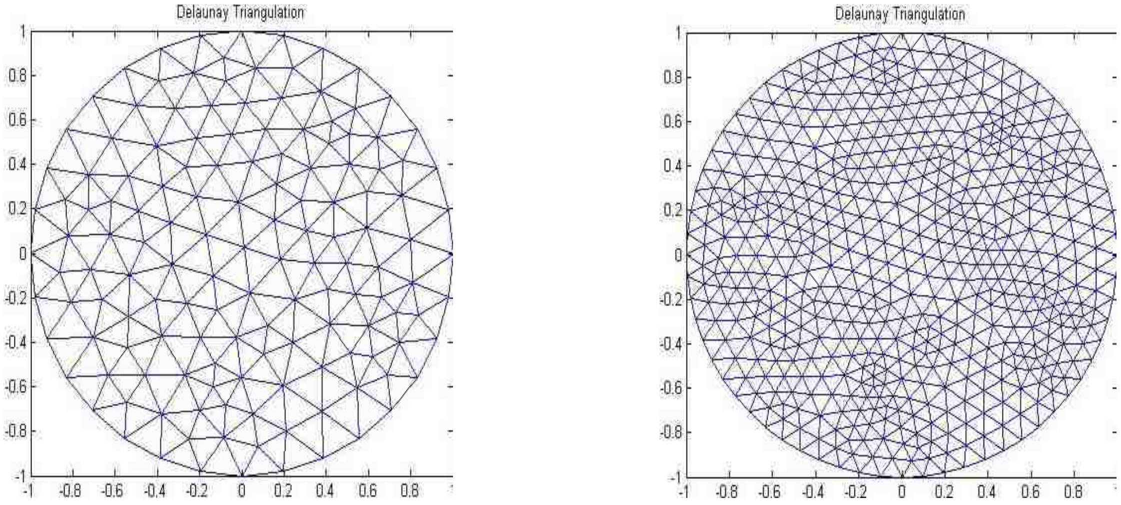


Figure 4.23 Example 3 - Coarse and refined mesh discretisations

The quality of both meshes is evaluated by eqn. (3.1) and represented on the colour scale of Fig. 4.24.

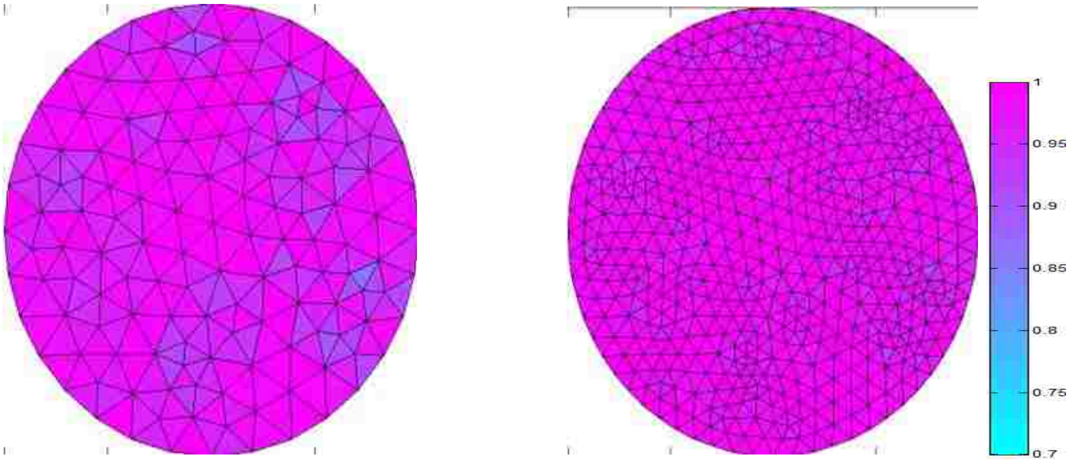


Figure 4.24 Example 3 - Coarse and refined mesh quality

The coarse mesh contains 36 nodes at the boundary and 108 nodes in the interior domain with 254 cells, while the refined mesh contains 72 boundary nodes and 469 interior nodes with 1016 cells. Therefore, the mesh refinement ratio is 4:1 as one cell in the coarse mesh is broken apart into four cells in the refined mesh. In addition, the mesh quality has also improved by refinement.

4.4.1 Solution by FEM

This problem is solved by FEM and the numerical solution is compared with the analytical solution. The results are shown in Table 4.10.

Mesh	RE	RMSE	OA
Coarse	0.49%	6.4407e-05	1.3863
Refined	0.12%	6.7501e-06	

Table 4.10 Example 3 - General result by FEM solver

As the mesh is refined, solution accuracy makes considerable improvement. The overall RE on the refined mesh becomes ¼ of the RE on the coarse mesh, while the RMSE is reduced by an order of factor 10 in the refined mesh. Figure 4.25 illustrates the RE contour plots for the FEM solution.

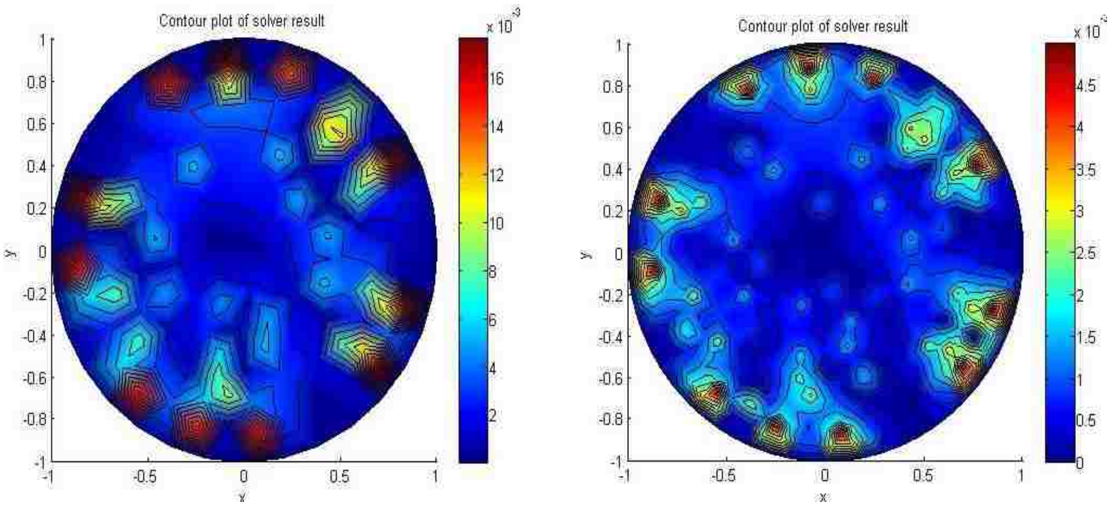


Figure 4.25 Example 3 – FEM RE contour plots on coarse and refined meshes

The contour plots show high RE located at the boundary of the unit disk and appear to be fairly symmetrical about the centre of the disk. Mesh refinement reduces the highest RE from 1.6% on the coarse mesh to 0.45% on the refined mesh.

4.4.2 Solution by CCFD

The CCFD method is used to solve this problem, by implementing the centroid-nodal differencing points' approximation and second-order central finite difference scheme. The solution contour plots for both coarse and refined meshes are shown in Fig. 4.26.

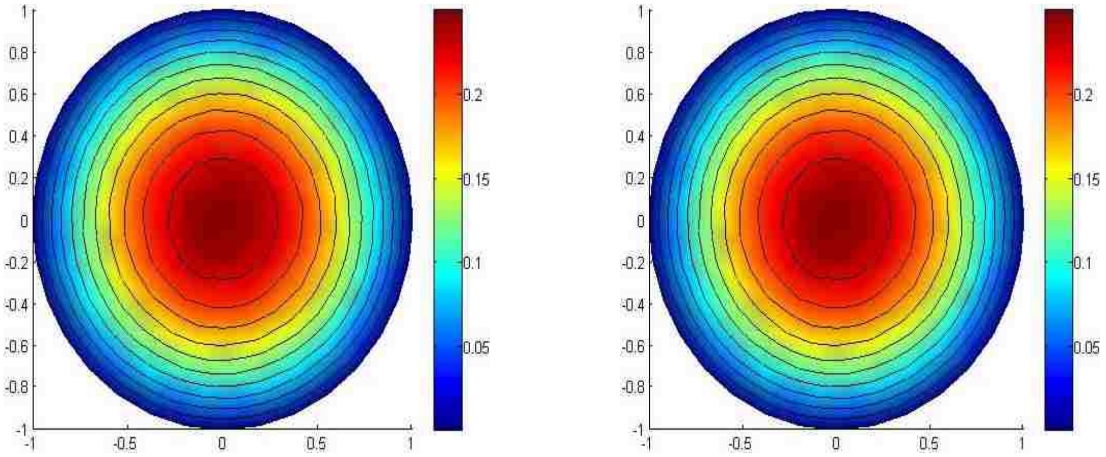


Figure 4.26 Example 3 – CCFD solution contour plots on coarse and refined meshes

High temperature value cluster at the centre core of the unit disk and gradually dissipates towards the boundary. The CCFD performance results are shown in Table 4.11.

Mesh	RE	RMSE	OA	Iterations	Time (s)
Coarse	3.43%	6.115e-04	0.8655	1029	1.8770
Refined	2.03%	1.842e-04		3679	26.8831

Table 4.11 Example 3 – General result by CCFD solver

The solution accuracy shows a slight improvement by mesh refinement, which is indicated by its order of accuracy and reduction in RE. The following Fig. 4.27 illustrates the RE contour plots in the geometric domain.

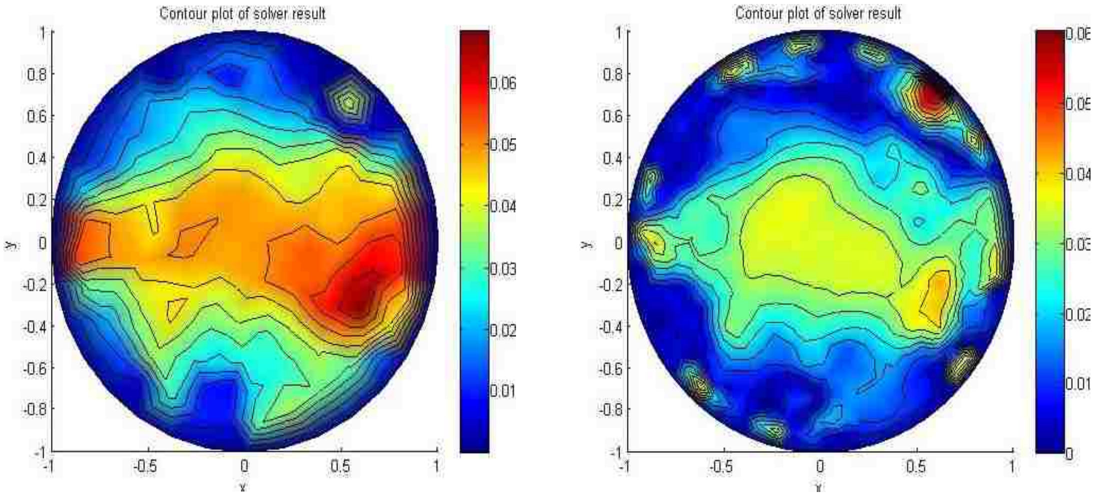


Figure 4.27 Example 3 – RE contour plots on coarse and refined meshes

Most of the RE in both sets of contours clusters at the central region of the disk. Improvement in solution accuracy is observed with lower RE on the refined mesh contours. Similarly, AE contour plots for both coarse and refined meshes are shown in Fig. 4.28. Similar error distribution can be observed in the these contour plots, as the error clusters at the central core region.

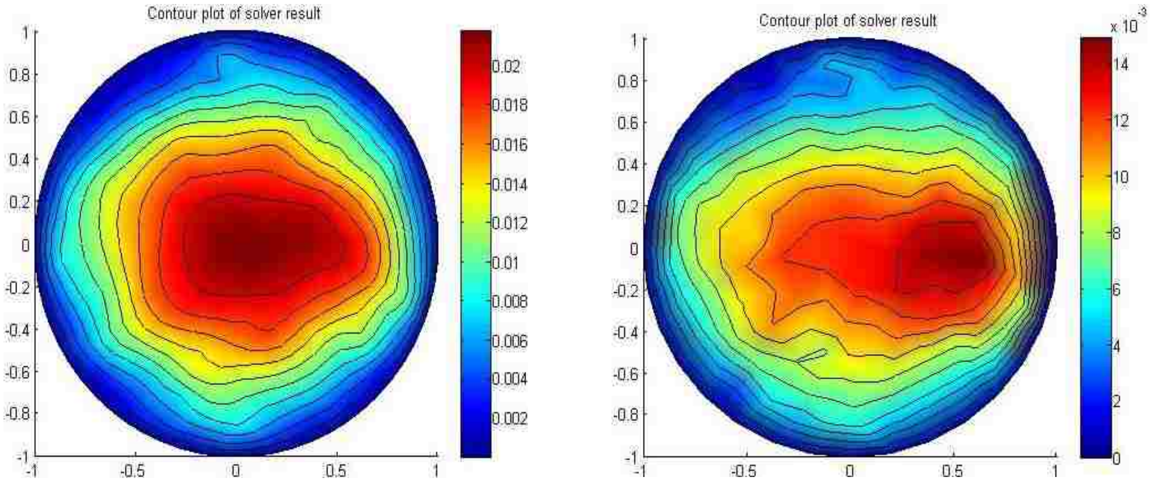


Figure 4.28 Example 3 – AE contour plots on coarse and refined meshes

4.5 Example 4: Laplace Equation for Potential Fluid Flow

This example describes potential fluid flow in a rectangular chamber that enters at a small inlet and leaves from an open-ended outlet [2]. The outlet is subjected to a Neumann boundary condition with zero flux, while the chamber walls and inlet are subjected to Dirichlet boundary conditions. The stream function of the flow is governed by the Laplace equation and the solution of the problem determines the streamline pattern within the chamber. The problem is described by eqn. (4.14) and illustrated in Fig. 4.29.

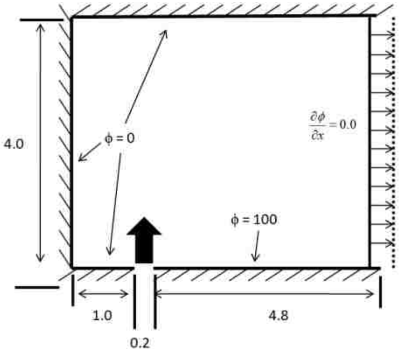


Figure 4.29 Example 4 description

$$\begin{cases} \dfrac{\partial^2 \emptyset}{\partial x^2} + \dfrac{\partial^2 \emptyset}{\partial y^2} = 0; \ \forall (x,y) \in (0,6) \times (0,4) \\ \emptyset(x,4) = \emptyset(0,y) = 0 \\ \emptyset(x,0) = 0; \ \forall x \in (0,1) \\ \emptyset(x,0) = 100; \ \forall x \in (1.2,6) \\ \dfrac{\partial \emptyset(6,y)}{\partial x} = 0; \ \forall y \in [0,4] \end{cases} \tag{4.14}$$

The domain is discretised with a structured mesh with 100 boundary nodes and 551 interior nodes. There are 31 nodes distributed uniformly in the *x*-direction and 21 nodes in the *y*-axis. Both vertical and horizontal directions have uniform equal spacing as Δ*x* = Δ*y* = 0.2. There are 1200 cells in the domain and each interior node has the same number of neighbouring cells, a common characteristic for structured meshes. The discretised mesh is shown in Fig. 4.30.

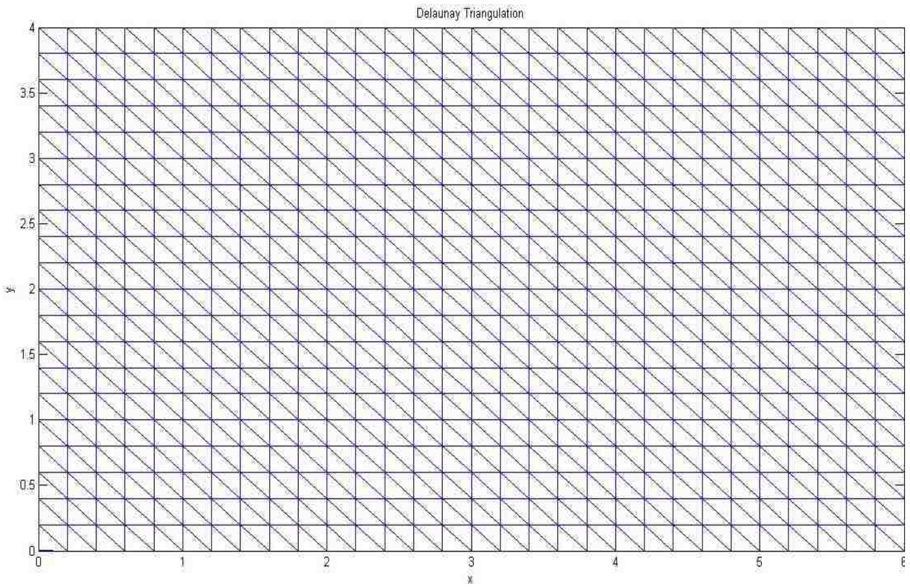


Figure 4.30 Example 4 - Mesh discretisation

This problem is solved by the CCFD method, using the centroid-nodal approximation scheme at the differencing points. Different approximation schemes are taken to evaluate the Neumann boundary condition points, as described in section 2.6, and the solution contour plots are compared.

4.5.1 CCFD Solution

For differencing points at the Neumann boundary, the 2nd-order backward finite difference scheme described in section 2.4.3.2 is implemented. Three approximation schemes are used to evaluate the Neumann boundary nodes, which are described as follows:

a) First-order backward difference scheme as in eq (2.20)
b) Second-order backward difference scheme as in eq (2.22)
c) Neighbouring cell centroids weighted average scheme

Schemes a) and b) are typically used in traditional FD methods for structured meshes, while scheme c) can be used to approximate Neumann boundary nodes in any arbitrary mesh topology. Differencing points in the inlet boundary are approximated by the weighted average of the neighbouring Dirichlet boundary conditions. The streamfunction contours obtained from the three approximation schemes in the CCFD method are shown in Figs. 4.31. 4.32 and 4.33.

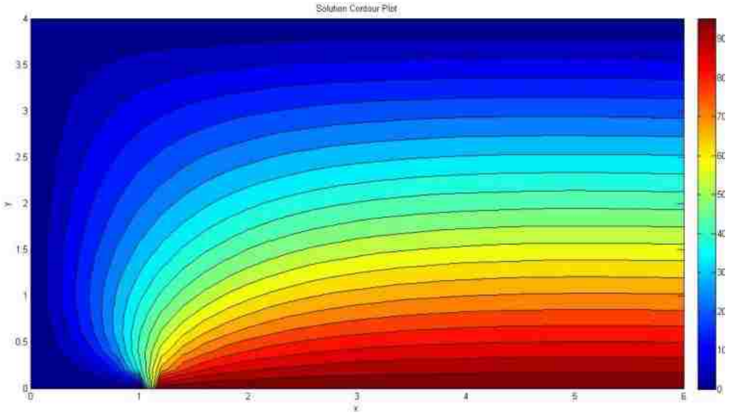


Figure 4.31 Example 4 - CCFD stream function contours by approximation scheme a)

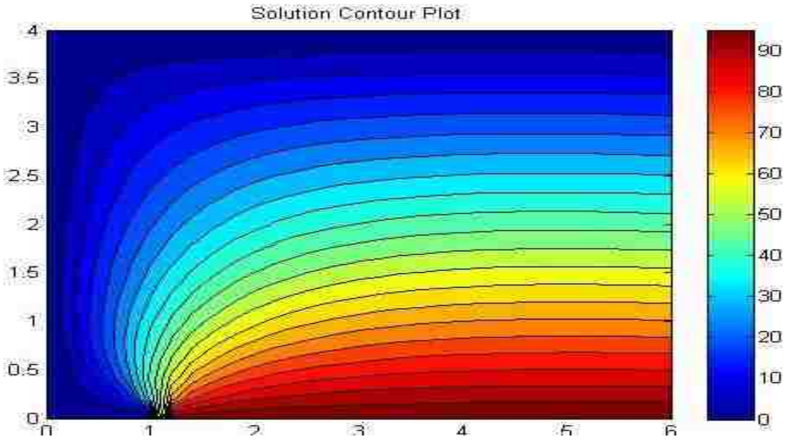


Figure 4.32 Example 4 - CCFD stream function contours by approximation scheme b)

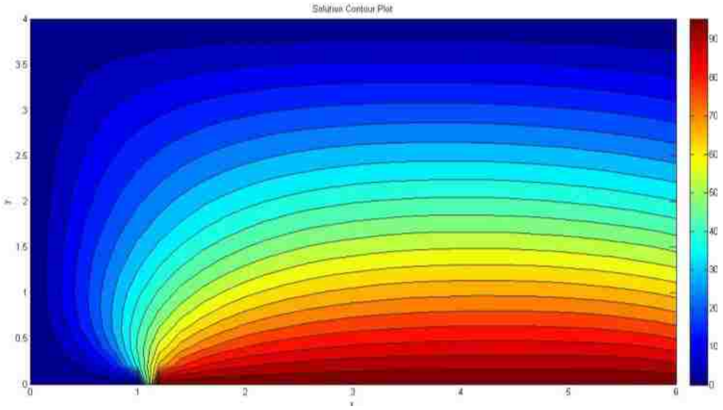


Figure 4.33 Example 4 - CCFD stream function contours by approximation scheme c)

Fluid enters from the inlet at the bottom left of the chamber. Fully developed flow is observed in the outlet region, consistent with the physical interpretation of the Neumann condition at the outlet. The stream function profiles at the Neumann boundary in the above three figures can be compared. The first-order backward approximation contour in Figure 4.31 shows that the streamlines are normal to the outlet boundary, as the boundary nodes have the same values as the neighbouring interior nodal values by its approximation scheme. Contours in Figure 4.32 by second-order backward approximation show a small step down effect in each streamline, but remains normal to the boundary. This is because the approximation takes consideration of two successive backward neighbouring nodal values. Streamline contours in Figure 4.33 show further bending to the outlet boundary, as the approximation takes the weighted average of the neighbouring cell values. From a physical interpretation, the first two sets of solution contours agree with the physical principle that the flux is normal to the boundary surface. However, approximation scheme c) has greater capacity to handle any arbitrary mesh topology, which will be described in the following example.

4.6 Example 5: Heat Transfer in Hexagonal Domain

The problem in this section considers 2D heat transfer in a hexagonal ring geometry shown in Fig. 4.34 [3]. The temperature distribution in the geometry is to be determined. Constant Dirichlet temperature boundary conditions and normal flux boundary conditions are assigned at the edges of the hexagon ring, as illustrated in Fig. 4.34. Discontinuities occur at the inner ring corners and average temperature of the neighbouring boundary values is assigned at these discontinuities. The governing equation is eqn. (4.15).

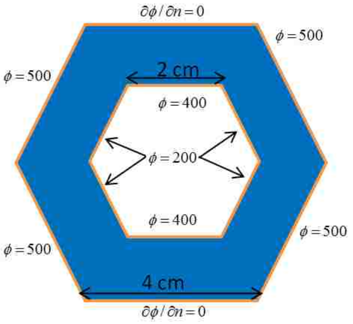


Figure 4.34 Example 5 description

$$\frac{\partial^2\emptyset}{\partial x^2}+\frac{\partial^2\emptyset}{\partial y^2}=0;\ \ \forall(x,y)\in\Omega \tag{4.15}$$

The governing equation is the elliptic Laplace equation, which models steady heat transfer in the domain. The problem only involves heat conduction, so it is a diffusion problem without any heat sources or sinks. A structured mesh cannot be constructed in this geometry in either Cartesian or cylindrical coordinates, so an unstructured mesh is used to discretise the domain. Three meshes with different refinement features are considered to solve this problem, as shown in Fig. 4.35.

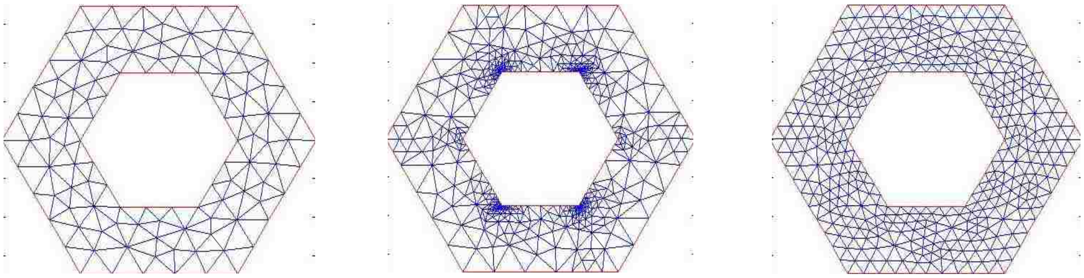


Figure 4.35 Example 5 - Coarse, clustered and refined mesh topologies

The detail information of each mesh is given in Table 4.12.

Mesh Information	**Coarse Mesh**	**Clustered Mesh**	**Refined Mesh**
Total # of Nodes	114	638	408
# of Interior Nodes	66	482	312
# of Boundary Nodes	48	156	96
# of Cells	180	1120	720

Table 4.12 Example 5 - Mesh information

The clustered mesh topology adapts the coarse mesh around the discontinuity region and the total number of nodes and cells are increased significantly. By comparison, the refined mesh topology adapts the coarse mesh in the entire domain by a refinement ratio of 4:1. The quality of the three mesh topologies are evaluated by eqn. (3.1) for each cell and the results are displayed in Fig. 4.36.

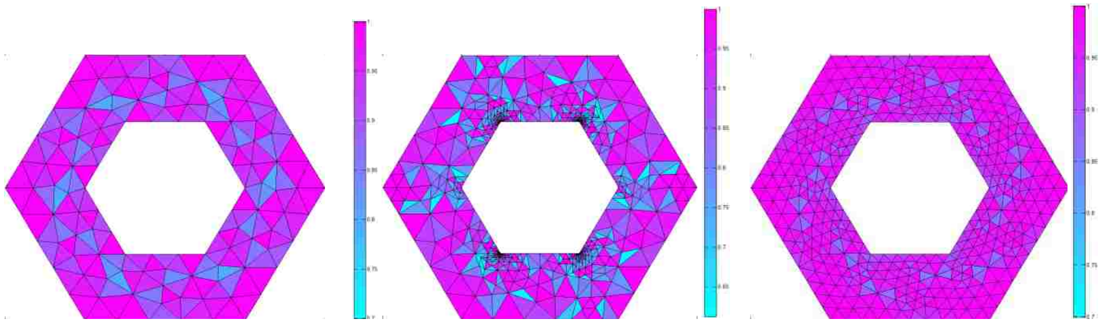


Figure 4.36 Example 5 - Quality of coarse, clustered and refined meshes

By comparing these figures, we see that there is no significant mesh quality improvement by refinement, since the quality figures for coarse and refined meshes, with the same scale imposed, show that the low-quality cells lie in the same region of the domain for both meshes. The quality is further deteriorated in the clustered mesh as some highly skewed triangles appear. The lowest-quality triangle in the clustered mesh is only 60% as good as the best quality triangle in the domain.

This problem is solved by the FEM and CCFD methods and the numerical results are compared. To explore the capability of the CCFD formulation by symmetry to reduce the size of the computational model, a half model and quarter model are also solved numerically and the solutions are compared with the full model results. Discretised meshes for half and quarter geometry are shown in Figs. 4.37 and 4.38.

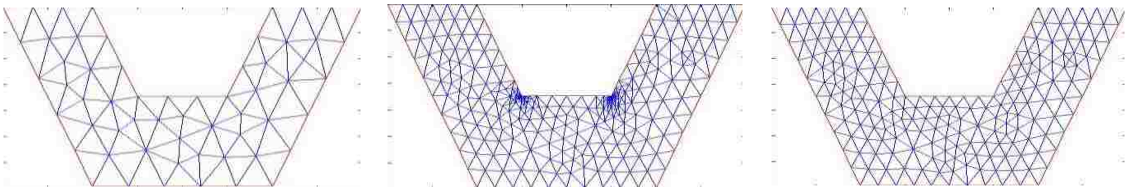


Figure 4.37 Example 5 - Coarse, clustered and refined meshes in half model

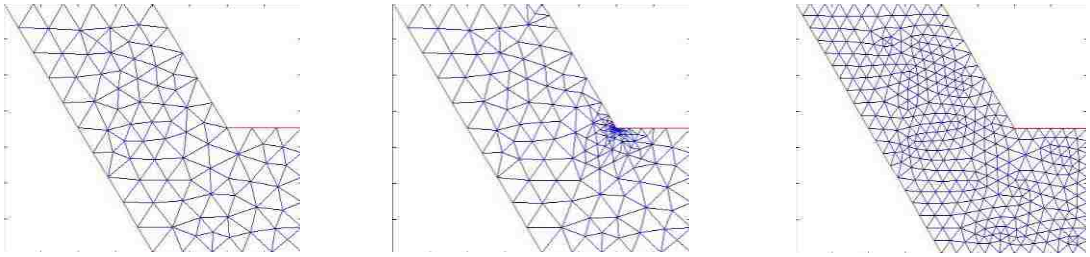


Figure 4.38 Example 5 - Coarse, clustered and refined meshes in quarter model

The detailed mesh information is given in Table 4.13.

Mesh Information	**Half Model**			**Quarter Model**		
	Coarse Mesh	**Clustered Mesh**	**Refined Mesh**	**Coarse Mesh**	**Clustered Mesh**	**Refined Mesh**
Total # of Nodes	59	337	203	99	187	360
# of Interior Nodes	29	259	143	66	141	294
# of Boundary Nodes	30	78	60	33	46	66
# of Cells	86	594	344	163	326	652

Table 4.13 Example 5 - Half model and quarter model mesh information

In the half model, the mesh size remains the same as in the full model, while the number of cells and nodes are reduced by half compared to the full model mesh. The quarter model on the other hand, uses mesh adaptation for the same mesh topology. The number of nodes and cells are twice as much as in the half model. Hence the mesh density is increased by four times in the quarter model compared with the half model and full model.

4.6.1 FEM Solution

The problem is solved by the FEM direct approach with full model, half model and quarter model for coarse, clustered and refined meshes. The numerical solution contours for each model are shown in Figs 4.39 – 4.41.

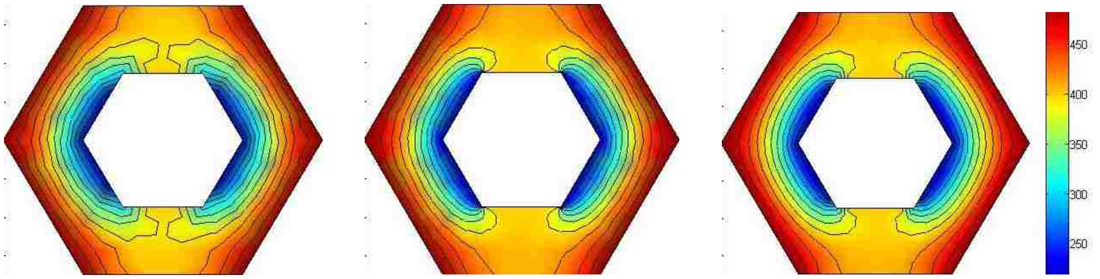


Figure 4.39 Example 5 – Full model FEM solution contours on coarse, clustered and refined meshes

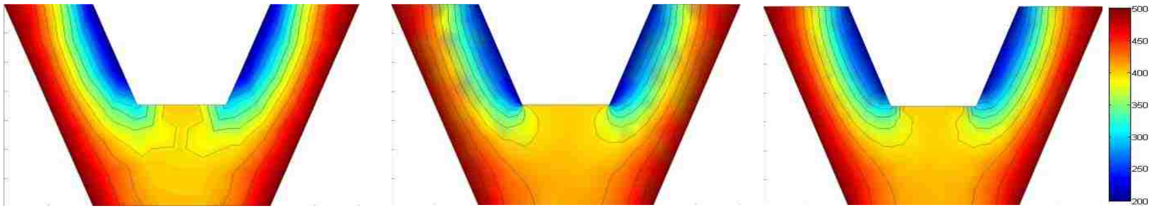


Figure 4.40 Example 5 – Half model FEM solution contours on coarse, clustered and refined meshes

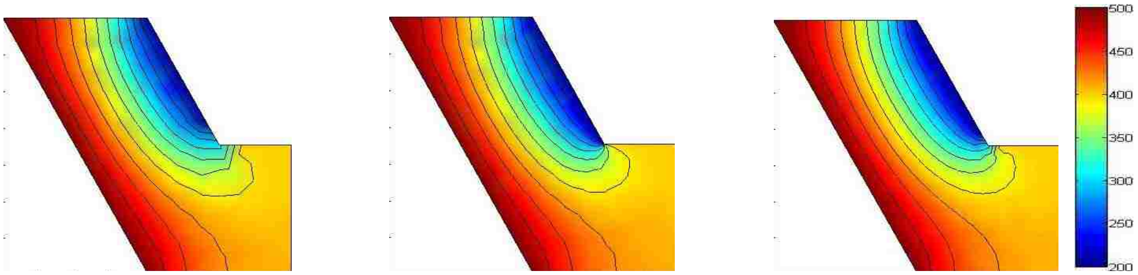


Figure 4.41 Example 5 – Quarter model FEM solution contours on coarse, clustered and refined meshes

The solution contours in all the models are similar, and there is a balance of the heat flux into and out of the domain along the Neumann boundaries. The clustered mesh and refined mesh contours have better transition and smoothness of temperature distribution in the corners where the boundary condition is discontinuous.

4.6.2 CCFD Solution

The CCFD method is used to solve this problem, by implementing centroid-nodal approximation at differencing points and 2nd-order central finite difference scheme to approximate the PDE. Because of the unstructured mesh, weighted average of neighbouring cell centroids are used to evaluate the Neumann boundary nodes. The results are compared with FEM solutions and shown in Table 4.14.

Model	Mesh	RD	RMSD	OA	Iterations	Time (s)
Full Model	Coarse	0.79%	0.4466	0.5484	268	0.5075
	Refined	0.80%	0.2088		884	6.0024
	Clustered	3.34%	0.6230	N/A	724	7.7118
Half Model	Coarse	0.88%	0.6843	0.6441	265	0.2623
	Refined	0.75%	0.2802		900	3.1380
	Clustered	2.06%	0.5343	N/A	686	5.4105
Quarter Model	Coarse	0.60%	0.3083	0.5116	896	1.4816
	Refined	0.60%	0.1517		3003	18.9048
	Clustered	2.67%	0.8830	N/A	959	3.1109

Table 4.14 Example 5 - General result by CCFD solver

The relative differences remain almost unchanged by mesh refinement, while relative mean square differences are reduced by half. On the other hand, the clustered mesh enlarges the mean difference between the FEM and CCFD results when local mesh refinement occurs in the regions of discontinuity on the boundaries. Figures 4.42 - 4.44 show the relative difference contours between FEM and CCFD results on all three mesh types and symmetric models.

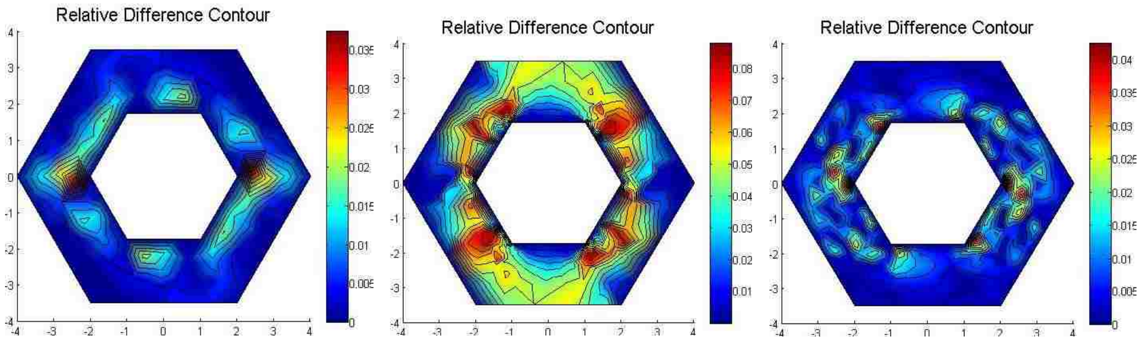


Figure 4.42 Example 5 - Full model RD on coarse, clustered and refined meshes

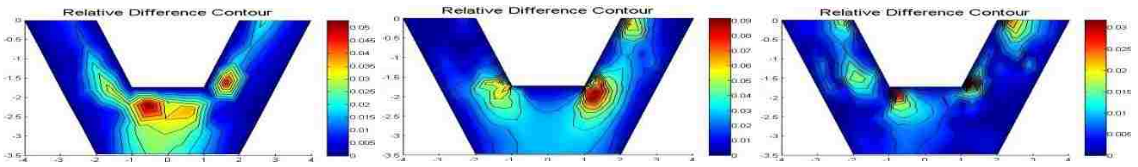


Figure 4.43 Example 5 - Half model RD on coarse, clustered and refined meshes

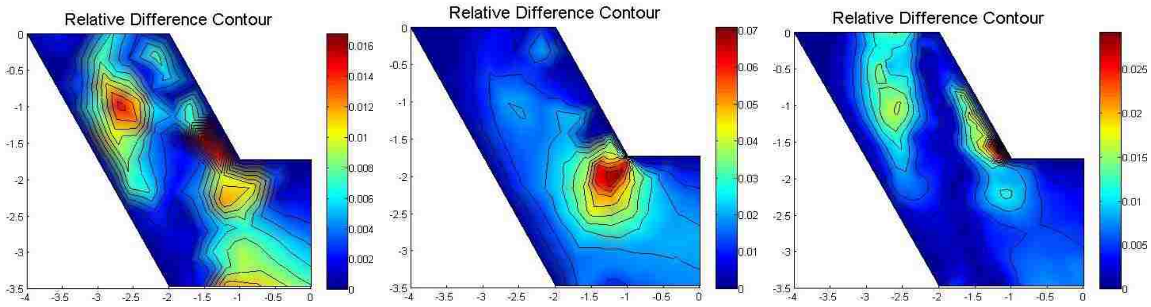


Figure 4.44 Example 5 - Quarter model RD on coarse, clustered and refined meshes

As a general pattern, the contour plots show high RD in discontinuous regions, particularly in clustered contours with local mesh refinement. In comparison between the coarse and refined contour plots, the legend scales of RD value show little difference but refined mesh RD plots have higher density contour lines than the coarse mesh plots. Since the full model and half model have same mesh density, both model contours have similar RD distribution. The quarter model on the other hand, has higher mesh density so the contour plots have lower value of RD distribution than both the full model and half model contour plots.

4.7 Example 6: Diffusion-Convection in a Unit Square with Dirichlet BC

Gupta et al. [17] has studied high order numerical schemes for convection-diffusion PDEs. The general PDE modeling the convection-diffusion problem is

$$\emptyset_{xx} + \emptyset_{yy} + p(x,y)\emptyset_x + q(x,y)\emptyset_y = f(x,y) \tag{4.16}$$

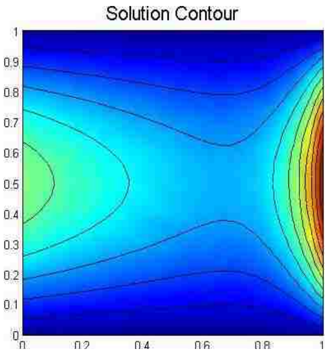
This equation is often used to describe transport phenomena. Consider the following boundary value problem in a unit square domain,

$$\begin{cases} \emptyset_{xx} + \emptyset_{yy} - \dfrac{1}{\varepsilon}\emptyset_x = 0 \\ \emptyset(x,0) = \emptyset(x,1) = 0\,; \quad \forall x \in [0,1] \\ \emptyset(0,y) = \sin(\pi y)\,; \quad \forall y \in [0,1] \\ \emptyset(1,y) = 2sin\,(\pi y)\,;\ \forall y \in [0,1] \end{cases} \tag{4.17}$$

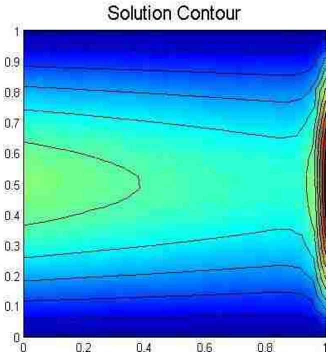
Comparison of (4.16) and (4.17) shows that $p(x,y) = -1/\varepsilon \equiv P$, $q(x,y) = 0$ and $f\,(x,y) = 0$, where P (or $1/\varepsilon$) in fluid flow context is interpreted as Reynolds number. This problem only considers the convection effect in x-direction without external source. The exact solution of BVP (4.17), given by

$$\emptyset(x,y) = \frac{e^{Px/2}\left\{sin(\pi y)2e^{-P/2}sinh(\sigma x) + sinh[\sigma(1-x)]\right\}}{\sinh\,(\sigma)} \tag{4.18}$$

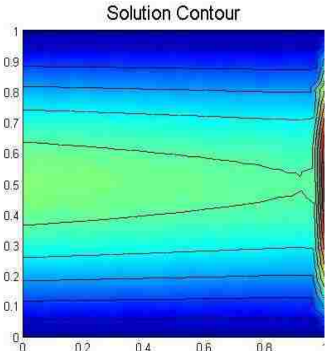
where $\sigma^2 = \pi^2 + P^2/4$, shows the presence of a boundary layer near x = 1. The thickness of the boundary layer is related to the Reynolds number P, so the boundary layer is expected to have an adverse effect on accuracy of the numerical solutions as P increases. The boundary layer effects for different value of P can be distinguished in the following exact solution contours, whose interior node values are determined based on the refined mesh in Fig. 4.1.



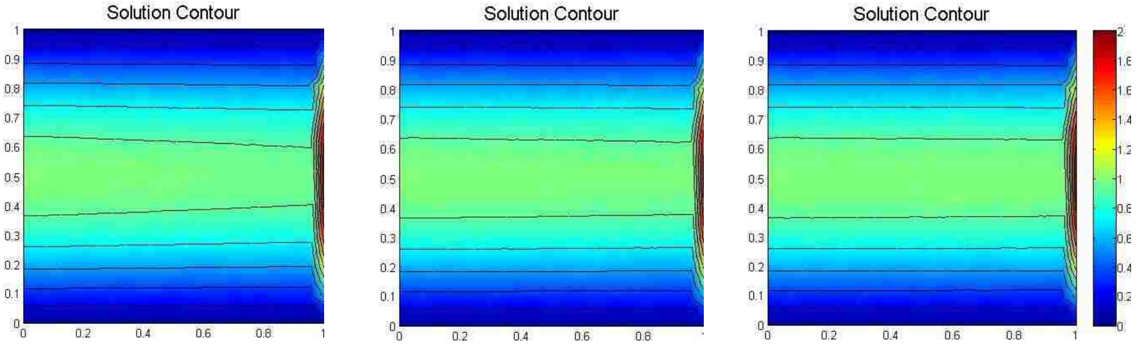
P = 10



P = 40



P = 100



P = 200 *P* = 500 *P* = 1000

Figure 4.45 Example 6 - Exact solution contours for different P values

Contour plots illustrate that the right boundary layer effect damps out sharply as *P* increases, while the left boundary layer effect has lower dissipation rate and eventually joins with the central part of the domain.

The boundary value problem eqn. (4.17) is solved numerically by CCFD for both mesh topologies shown in Fig 4.1.

4.7.1 CCFD Solution

The CCFD method is used to solve this problem, by implementing centroid-nodal differencing point approximations and 2nd-order central finite difference scheme for the second-order derivatives in eqn. (4.17). The first derivative convective term is approximated by 2nd-order central finite difference using

$$\left.\frac{\partial f}{\partial x}\right|_{ij} = \frac{f_{i+1,j} - f_{i-1,j}}{2(\Delta x)} + O(\Delta x)^2 \qquad (4.19)$$

Then the coefficients of the generalized equation with second-order central finite difference scheme in eqn. (2.21) are given as

$$\begin{aligned}
\gamma_{CC} &= \frac{8}{(x_E - x_W)^2} + \frac{8}{(y_N - y_S)^2} \\
\gamma_W &= \frac{4}{(x_E - x_W)^3}\left[2x_E - 2x_{CC} + \frac{P}{4}(x_E - x_W)^2\right] \\
\gamma_E &= \frac{-4}{(x_E - x_W)^3}\left[2x_W - 2x_{CC} + \frac{P}{4}(x_E - x_W)^2\right]
\end{aligned} \qquad (4.20)$$

$$\gamma_S = \frac{8(y_N - y_{CC})}{(y_N - y_S)^3};\ \gamma_N = \frac{-8(y_S - y_{CC})}{(y_N - y_S)^3}$$

Different magnitudes of Reynolds number (i.e. the ratio between convection to diffusion) are studied at P = 10, 40, 100 and 500 to examine different boundary layer effects. The result is given in Table 4.15.

P	Mesh	RE	RMSE	OA	Iterations	Time (s)
10	Coarse	24.85%	0.0126	0.5352	336	0.9025
	Refined	24.76%	0.0060		870	11.5147
40	Coarse	25.71%	0.0152	0.5291	209	0.5876
	Refined	25.53%	0.0073		608	8.2178
100	Coarse	16.78%	0.0112	0.6419	88	0.2276
	Refined	15.17%	0.0046		290	3.7968
500	Coarse	N/A	N/A	N/A	N/A	N/A
	Refined	6.67%	0.0032		85	1.0454

Table 4.15 Example 6 - General result by CCFD solver

Higher convection to diffusion ratio P (or higher Reynolds number) indicates reduction of RE and improvement of solution accuracy. Also, the required mesh density depends on this ratio, in which higher value P requires a more refined mesh to solve numerically by the CCFD method. For example, the solution does not converge on the coarse mesh for P = 500 when the flow is dominated by convection. This may be due to the use of the central difference approximation for the convective derivative term. Upwinding is more commonly used in fluid flow simulations since it correctly models the flow behaviour and produces stable solutions. Regarding mesh sensitivity, the overall RE shows little changes by mesh refinement, while the RMSE is reduced by half for every refinement. The RE contour plots for P = 10, 40 and 100 on both mesh topologies are shown in Figs. 4.46 - 4.48.

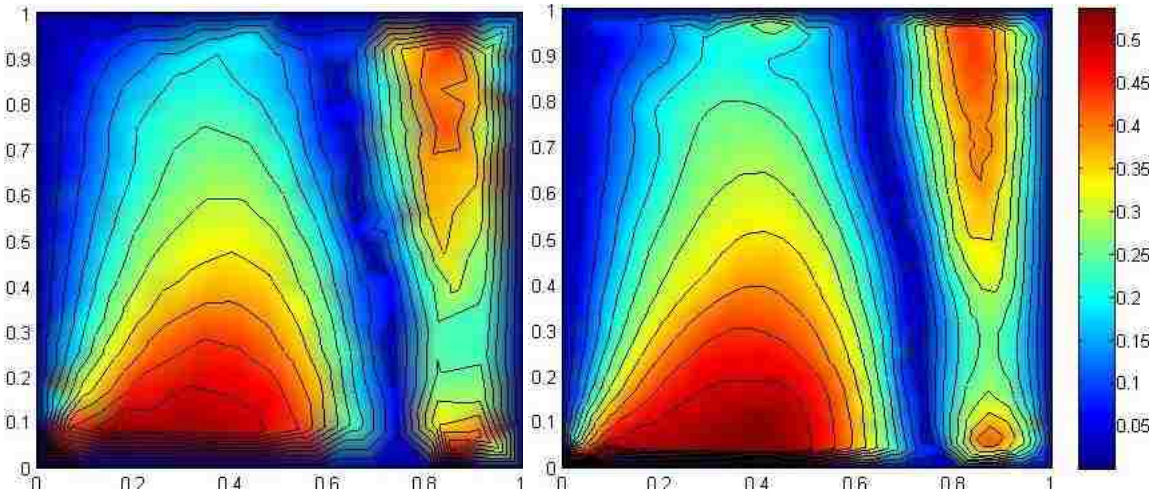


Figure 4.46 Example 6 - RE contours for P = 10 on coarse and refined meshes

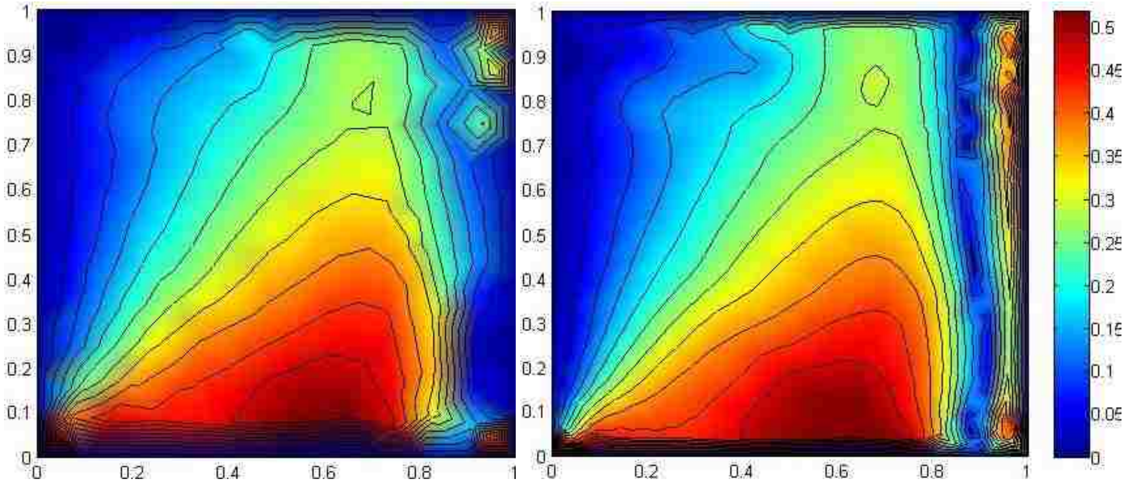


Figure 4.47 Example 6 - RE contours for P = 40 on coarse and refined meshes

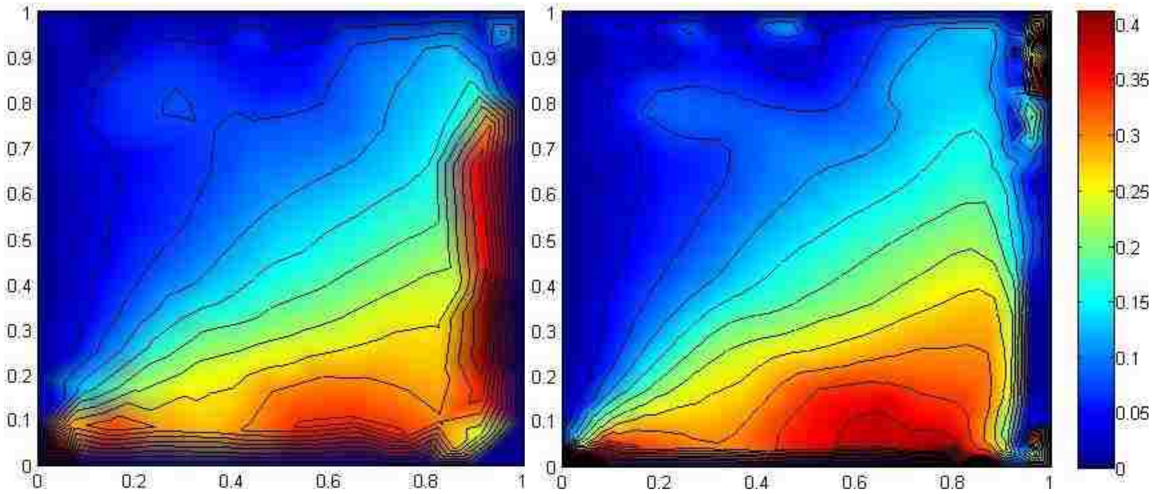


Figure 4.48 Example 6 - RE contours for P = 100 on coarse and refined meshes

High values of RE are mainly located around the lower and right boundary layers. Errors located at the lower boundary propagate towards the centre and gradually dissipates out. Errors at right boundary layer are mainly due to high localized gradient effects.

4.8 Example 7: Unsteady Diffusion in a Unit Square with Dirichlet BC

This initial-boundary value problem considers the time dependent version of the steady diffusion problem studied in Example #1, described by the following equation,

$$\begin{cases} \dfrac{\partial^2 \emptyset}{\partial x^2} + \dfrac{\partial^2 \emptyset}{\partial y^2} = \dfrac{\partial \emptyset}{\partial t};\ \ \forall (x,y) \in (0,1) \times (0,1) \\ \emptyset(x,1,t) = \ \emptyset(x,0,t) = 0;\ \forall x \in (0,1) \\ \emptyset(1,y,t) = 0, \emptyset(0,y,t) = 100;\ \forall y \in (0,1) \\ \emptyset(0{,}1,t) = \emptyset(0{,}0,t) = 50 \\ \emptyset(x,y,0) = 0;\ \ \forall (x,y) \in (0,1) \times (0,1) \end{cases} \tag{4.21}$$

The first-order time derivative in the governing equation classifies the PDE as parabolic type. When the solution reaches equilibrium as time approaches infinity, the solution of this problem is the same as the exact solution in steady state in eqn. (4.3).

For this unsteady problem, a time marching scheme is implemented in the CCFD formulation to solve the problem on both coarse and refined meshes shown in Figure 4.1. Applying the explicit Euler scheme to the governing equation (4.21), the resulting finite difference equation is

$$\emptyset_{CC}^{n+1} = \Delta t(\gamma_W \emptyset_W^n + \gamma_E \emptyset_E^n + \gamma_S \emptyset_S^n + \gamma_N \emptyset_N^n - \gamma_{cc} \emptyset_{cc}^n) + \emptyset_{cc}^n \tag{4.225}$$

where the coefficients are the same as in eqn. (2.21).

This explicit scheme appears to be unstable in the CCFD formulation, as the coefficients of the differencing points calculated from geometric definition are large numbers. Instead, a semi-implicit scheme has been devised to ensure a stable solution. This is achieved by evaluating the $\gamma_{cc} \emptyset_{cc}^n$ term on the RHS of eqn. (4.22) at time level *n*+1 instead of *n*, leading to

$$\emptyset_{CC}^{n+1} = \frac{\Delta t(\gamma_W \emptyset_W^n + \gamma_E \emptyset_E^n + \gamma_S \emptyset_S^n + \gamma_N \emptyset_N^n) + \emptyset_{cc}^n}{1 + \Delta t \gamma_{CC}} \tag{4.26}$$

CCFD is used to solve this problem with semi-implicit time marching scheme and the transient numerical solution is compared with the steady numerical solution and the exact solution in Example #1 at large time. The effect of different time step sizes is also investigated in the semi-implicit CCFD solver.

4.8.1 CCFD Solution

CCFD is implemented to solve this problem, by applying centroid-nodal differencing points' approximation, 2nd-order central finite difference scheme and the semi-implicit time marching scheme described above. The time domain is discretised uniformly with different time step sizes, for a total simulation time of 5 seconds. Comparison of results between the numerical transient solution and the exact solution at steady-state is shown in Table 4.16.

Time step	Mesh	RE	RMSE	OA	RD	RMSD	Time (s)
Δt = 0.02s N = 250	Coarse	6.24%	0.0767	-1.1835	6.47%	0.7731	6.7663
	Refined	58.34%	0.3957		58.11%	89.3382	9.8400
Δt = 0.01s N = 500	Coarse	2.00%	0.0275	-1.3403	0.29%	0.0015	11.8397
	Refined	30.55%	0.1763		30.24%	17.2782	20.8127
Δt = 0.005s N = 1000	Coarse	2.03%	0.0277	-0.3020	0.008435%	1.301e-06	12.6256
	Refined	6.95%	0.0421		6.54%	0.7302	30.9144
Δt = 0.0025s N = 2000	Coarse	2.07%	0.0277	-0.3437	0.009162%	1.5350e-06	75.7378
	Refined	1.82%	0.0172		0.22%	9.2784e-04	123.719

Table 4.16 Example 7 – General result by CCFD solver

Relative difference (RD) and root mean square difference (RMSD) are calculated by comparing the transient solution at the last time step to the steady numerical solution. An apparent result from the above table shows that the refined mesh requires a finer time step size to reach steady-state. This can be verified by the negative values for order of accuracy, which indicates an adverse solution accuracy effect by mesh refinement in the spatial domain. The coarse mesh reaches steady-state with time step size of 0.01s with RE of 2.00% and RD of 0.29%, while the refined mesh solution requires the step size of 0.0025s to reach equilibrium state with RE of 1.82% and RD of 0.22%. This is an indication of the relationship between mesh cell size, time step size and stability of the numerical algorithm. Nevertheless, this relationship is not explored in this thesis.

The following figures show the RE and RD contours for the transient solution at the last time step, for the coarse mesh.

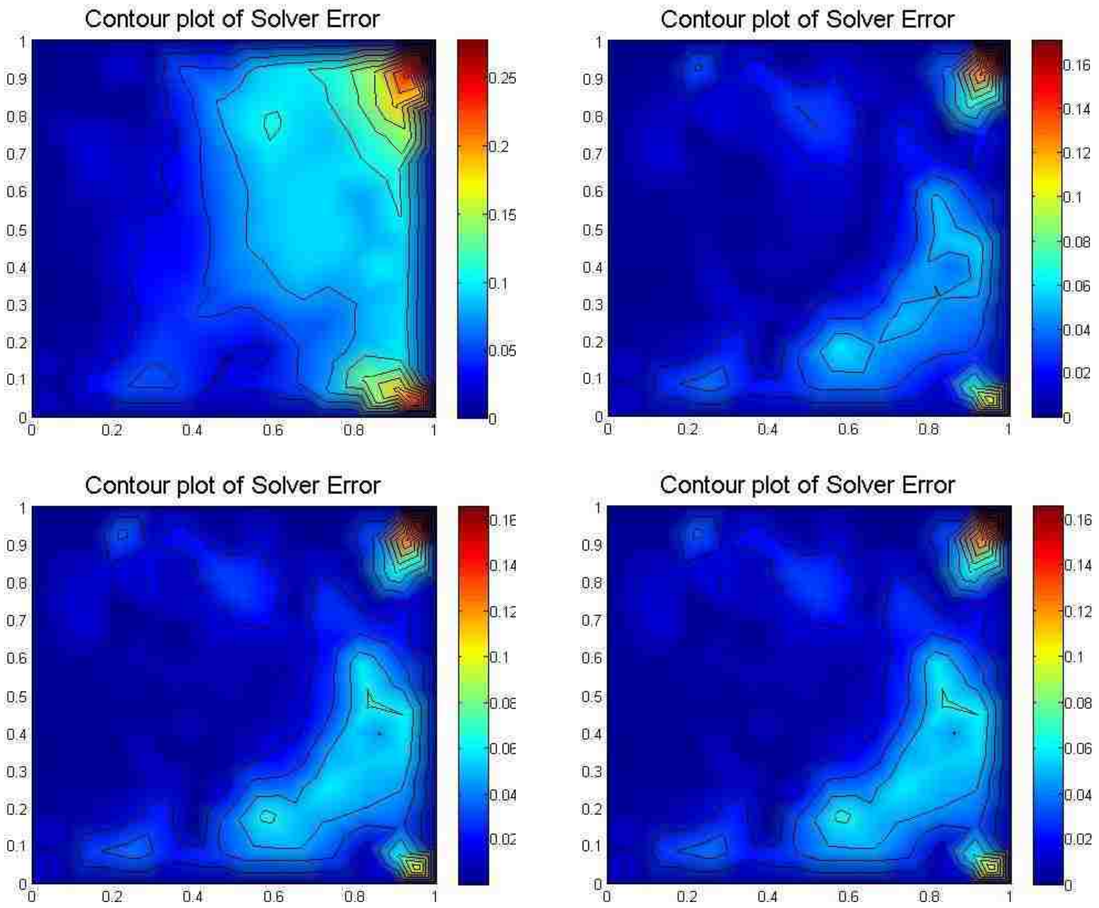
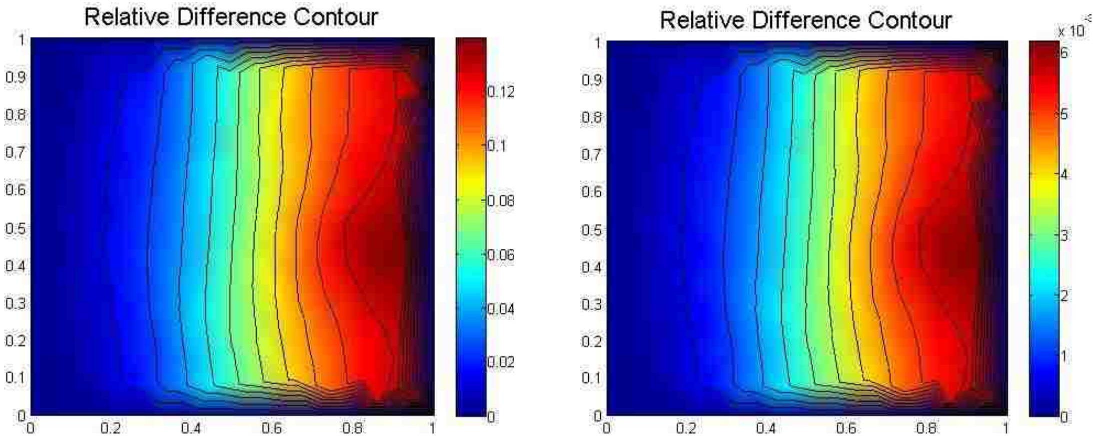


Figure 4.49 Example 7 - RE contours on coarse mesh with time step sizes of 20, 10, 5 and 2.5 ms



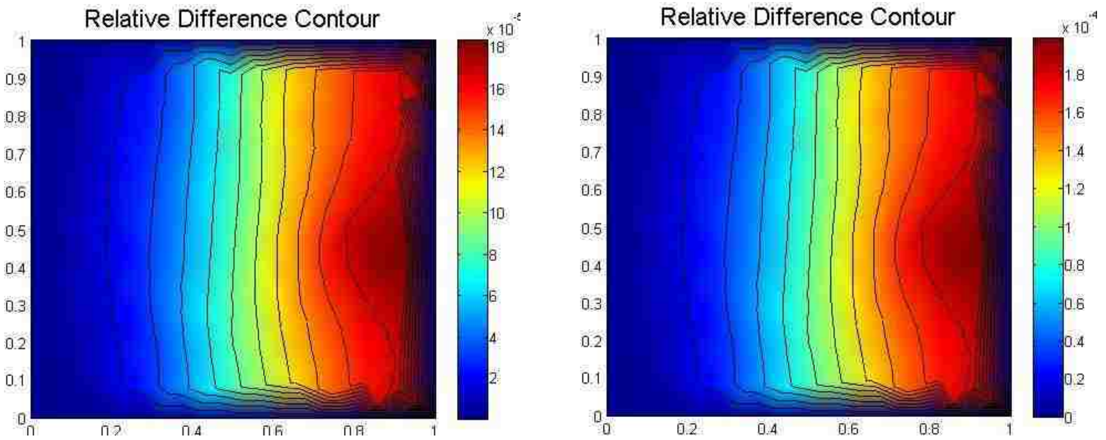
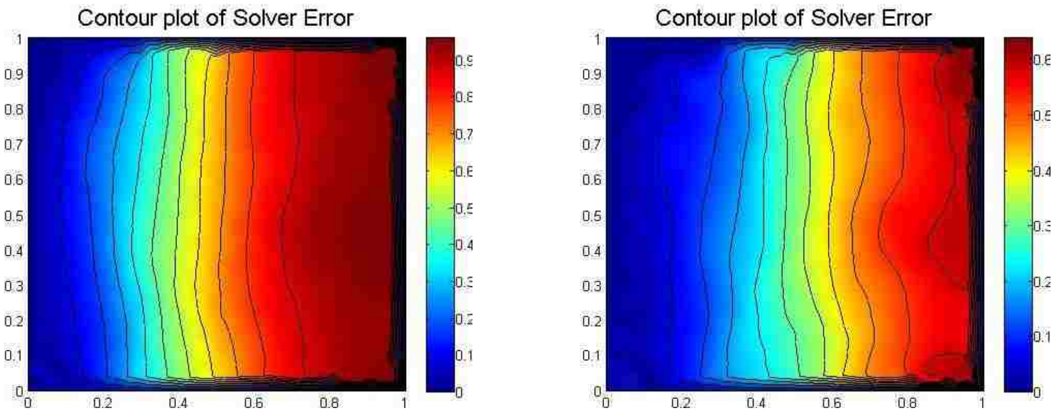


Figure 4.50 Example 7 - RD contours on coarse mesh with time step sizes of 20, 10, 5 and 2.5ms

The RE contours show no significant difference with the time step size ≤ 10 ms. High values of RE are clustered at both bottom and top right corners, where small temperature distribution occurs. The same error distributions also show in the RD contours compared with the steady-state numerical solution, since the RD is mostly located close to the right boundary. However, the RD values further reduce with smaller time step size, which can be seen from the legend scale at the right of the contours. For comparison, the RE and RD contours for the refined mesh topology are shown in Figs. 4.51 and 4.52.



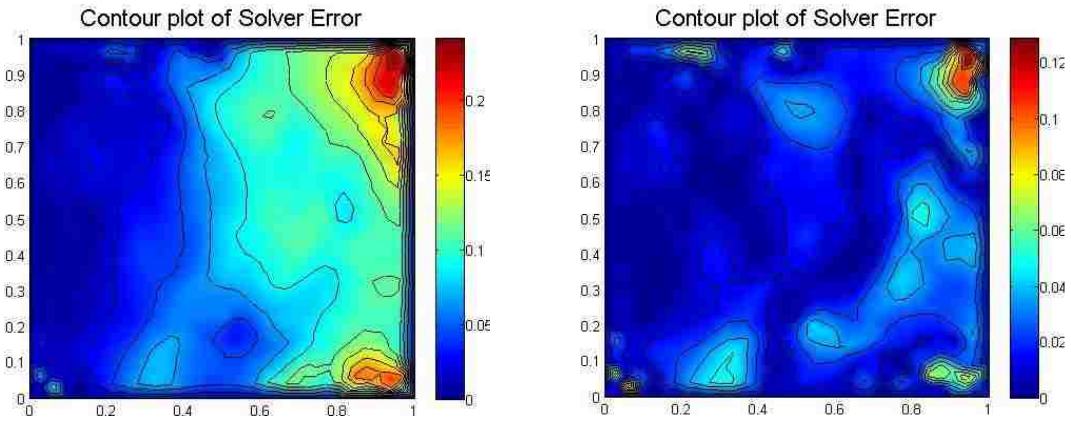


Figure 4.51 Example 7 - RE contours on refined mesh with time step sizes of 20, 10, 5 and 2.5ms

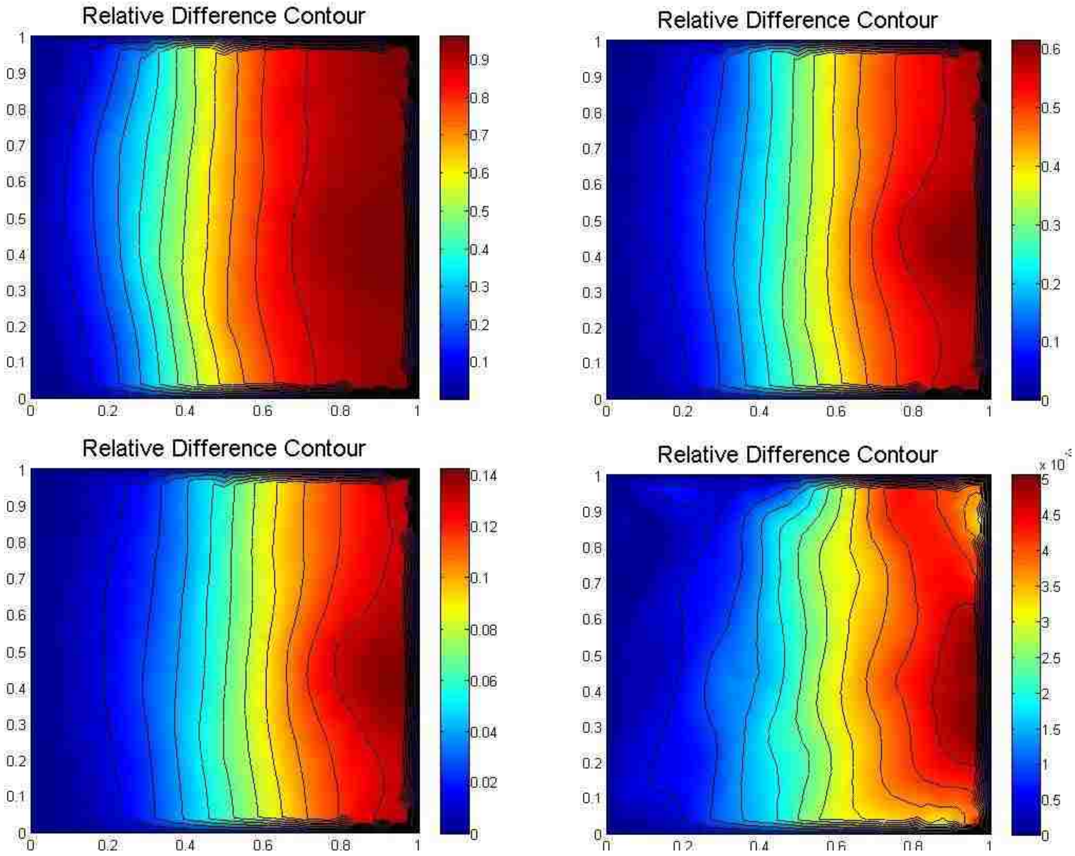


Figure 4.52 Example 7 - RE contours 0n refined mesh with time step sizes of 20, 10, 5 and 2.5ms

The transient solution on the refined mesh reaches steady-state with a time step size of 2.5 ms when the overall RE and RD reach an acceptable level within 2%. The RD contours show clustering at the far field region close to the right boundary, while the RE contours at Δt = 2.5 ms show clustering at right boundary corners.

CHAPTER 5 – DISCUSSION AND SUMMARY OF CCFD METHODOLOGY

5.1 Discussion of CCFD Results

In most of the examples studied in the previous chapter, solution accuracy is only slightly improved by mesh refinement, indicated by small values of order of accuracy. Instead, mesh refinement yields a smoother error distribution with well-defined contour lines. In other words, the local error in the coarse mesh solution is distributed across a broader region in the refined mesh solution. RE and RD contour plots illustrate that most of the error is concentrated at discontinuities and in high gradient regions. Numerical results indicate that locally clustering the mesh in regions where the boundary conditions are discontinuous may adversely affect solution accuracy compared to the accuracy obtained on a regular unstructured mesh.

Combinations of several important aspects of the CCFD method have been investigated, including

a) <u>Differencing points approximation schemes</u>

   Differencing points that are approximated by centroid-nodal weighted average scheme, which considers the neighbouring cell effect in the computation, generally has better solution accuracy than other approximation schemes.

b) <u>Iterative schemes</u>

   Regardless of the iterative schemes implemented in the method to solve the resulting system of linear algebraic equations, there is no difference in the numerical solution. However, the convergent rate is significantly improved for the successive over-relaxation iterative scheme compared to the other two methods.

c) <u>Time marching schemes</u>

   CCFD with an explicit Euler time marching scheme appears to be unstable, while a semi-implicit scheme ensures better stability and solution convergence.

5.2 Error and Uncertainties Analysis

Oberkampf and Trucano [18] gave definitions for error and uncertainty in numerical simulations. Error is defined as a recognizable deficiency in a physical model that is not caused by lack of knowledge, while uncertainly is stated as a potential deficiency in a physical model that is caused by lack of knowledge. The errors and uncertainties of the CCFD method can be diagnosed by the following Ishikawa Diagram (or Fishbone Diagram):

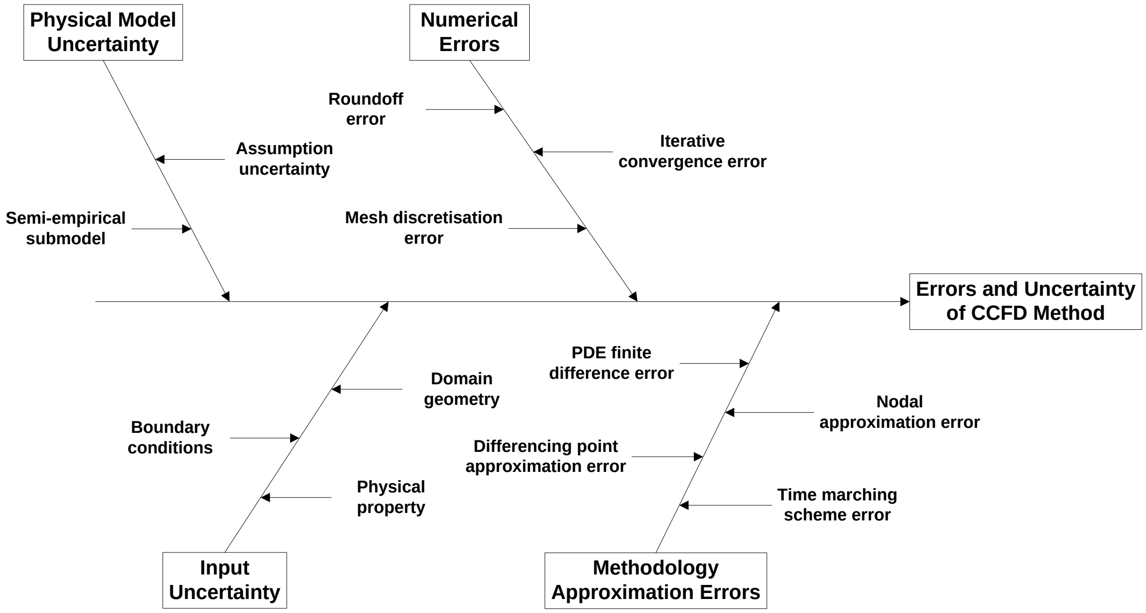

Physical Model Uncertainty
Numerical Errors
Roundoff error
Iterative convergence error
Assumption uncertainty
Semi-empirical submodel
Mesh discretisation error
Errors and Uncertainty of CCFD Method
PDE finite difference error
Domain geometry
Nodal approximation error
Boundary conditions
Differencing point approximation error
Physical property
Time marching scheme error
Input Uncertainty
Methodology Approximation Errors

Figure 5.1 Fishbone diagram of errors and uncertainties analysis

Physical model uncertainty describes the discrepancies between real physical parameters and parameter given in the actual model due to inadequate representation of physical processes or due to simplifying assumptions in the modeling process, such as steady flow and incompressible flow. Input uncertainty includes the inaccuracies due to limited information or approximated representation of geometry, boundary conditions and material properties. In general, uncertainties arise in a physical model due to lack of knowledge and exist in all types of numerical simulations.

On the other hand, numerical errors consist of round-off error, iterative convergence error and mesh discretisation error. Round-off error is the result of the computational representation of real numbers by a finite number of significant digits, which is controlled by the machine accuracy. Iterative convergence error refers to the residual between the last two iterations. In this thesis, the residual is calculated by average absolute mean difference in centroid values between two successive iterations. The smaller the convergence tolerance is, the numerical solution generally becomes more accurate but requires more computational cost for a higher number of iterations. Mesh discretisation error describes the mesh quality in the domain. Usually some cells have relative low quality with high skewness and aspect ratio, which may affect the numerical solution accuracy. Some approximation errors which are inherent in the CCFD method include FD errors, differencing points and nodal approximation errors and time marching scheme errors. FD error refers to the order of accuracy of FD formulation of the PDE applied at cell centroids. Differencing point and nodal approximation errors refer to the errors in the evaluation methods at differencing points and nodes. For best solution accuracy of the available approximation schemes studied in the previous chapter, differencing points are evaluated by centroid-nodal weighted average scheme, while nodal values are approximated by the weighted average of neighbouring cell centroids values. Errors in the time marching scheme include the order of accuracy in finite difference approximation for the transient derivatives in the PDE, which is related to the errors in time step size discretised in the time domain. Generally, a more refined time step size ensures the final transient solution reach steady-state, but requires more computational cost.

5.3 Advantage of CCFD Numerical Method

The new proposed CCFD numerical method has the following advantages when compare with other numerical methods:

- CCFD methodology can be applied on any type of mesh topology, including structured, unstructured or hybrid mesh, regardless of its conformality. Traditional FD can only be applied on structured grids.
- CCFD method provides computational values at both nodes and cell centroids, while other numerical methods only provide nodal values.
- Higher-order finite differencing schemes can be implemented at the cell centroid in each finite difference stencil, by taking more differencing points in the FD formulation. Although results in some examples show little apparent accuracy improvement by implementing a higher-order FD scheme, further study can be taken to investigate the causes. In contrast, FVM is often restricted with respect to developing higher-order schemes.

- CCFD method is able to solve on a domain where the physical phenomena are governed by multiple PDEs in different regions. For example, different regions may be subjected to different physical conditions or consist of different materials. This advantage is especially beneficiary to solve fluid-structure interaction problem, while FEM and FVM are generally applied separately in the solid and fluid regions, respectively.

5.4 CCFD Method Future Improvement

Solution accuracy of the CCFD numerical results needs further investigation. Higher-order approximation schemes at the differencing points, nodes and cell centroids can be developed to reduce errors in the CCFD computation.

A simple parabolic PDE problem has been studied to check the compatibility of the CCFD method for time-dependent problem. However, more efficient and stable time marching schemes should be further developed. In addition, von Neumann stability analysis can be performed to check for numerical stability and find the connection between time step size and cell size.

In chapter 4, the CCFD numerical method is used to solved heat transfer and fluid mechanics problems. These problems involve diffusion and convection, i.e. first and second derivatives with respect to single space variables. Other types of physical problems, for example solid mechanics problem involving plane stress or plain strain, are usually governed by coupled PDEs with cross-derivative terms. The coupled PDEs represent the loading conditions in different dimensions, while cross-derivative terms involve high-order derivatives with respect to different independent space variables, e.g. $\partial^2\emptyset/(\partial x\partial y)$. Therefore, in order to solve other types of physical applications, formulations of the CCFD method which deal with cross-derivative terms as well as coupled PDEs are required to be further investigated.

Last but not least, Neumann boundary conditions along a line that is not aligned with the Cartesian axes are not considered in the present thesis. An example of this condition is shown in Fig. 5.2.

[redacted]

Figure 5.2 Neumann boundary not aligned with Cartesian axes

In the domain ABCD, techniques for handling the arc AB and line segment BC, subjected to Neumann boundary conditions, have not yet been formulated in the CCFD method. Formulation of such Neumann boundary conditions is required to solve problems on complex geometry domains.

5.5 Conclusions

In this thesis, a new numerical methodology is proposed to solve governing PDEs in fluid flow and heat transfer problems, which is referred to as a Cell-Centred Finite Difference (CCFD) method. The main idea of the CCFD method is to implement a finite difference approximation of the PDEs at the cell centroid for each cell in a discretised mesh topology, while keeping the differencing stencil confined to the cell. Typical "benchmark" PDE problems are tested with the CCFD method and the results are compared with exact solutions or FEM solutions. The new proposed methodology has the advantages of a finite difference scheme but is not restricted to a structured mesh topology. Potentially, it may provide a powerful alternative for numerical simulation to solve general physical problems governed by PDEs.

REFERENCES

1. Kreyszig, E., *Advanced Engineering Mathematics*. 2nd ed., 1967, Columbus, OH: John Wiley and Sons Inc. 486 - 529.
2. Hoffmann, K.A. and S.T. Chiang, *Computational Fluid Dynamics*. 2nd ed. Vol. I. 2004, Wichita, KS: Engineering Education System..
3. Versteeg, H.K. and W. Malalasekera, *An Introduction to Computational Fluid Dynamics.* 2 ed., 2007. The Finite Volume Method. New York: Pearson Prentice Hall.
4. Lohner, R., *Applied CFD Techniques*. 2nd ed., 2008, West Sussex, England: John Wiley & Sons Ltd.
5. MATLAB, in *Partial Differential Equation Toolbox User's Guide*. 2010, The MathWorks, Inc.: Natick, MA.
6. Lube, G. and G. Rapin, *Residual-based stabilized higher-order FEM for a generalized Oseen problem.* Appl. Sci., 2006. **16**: p. 949-966.
7. Surana, K.S., S.Allu, P.W. Tenpas, J.N. Reddy. *k-version of finite element method in gas dynamics: higher-order global differentiability numerical solution.* International Journal for Numerical Methods in Engineering, 2007. **69**: p. 1109 - 1157.
8. Antoci, C., M. Gallati and S. Sibilla, *Numerical simulation of fluid-structure interaction by SPH.* Elsevier - Computers and Structures, 2007(85): p. 879-890.
9. Hallquist, J.O., *LS-DYNA Theory Manual*, in *Chapter 38 - SPH*. March 2006, Livermore Software Technology Corporation: Livermore.
10. Zamani, N.G. and W.B. Altenhof, eds. *Introduction to the Finite Element Analysis*. 2008, FEA Connenction: Windsor.
11. MATLAB, *Documentation*, in *Delaunay Triangulation and Interpolation*. 2010, MathWorks: Natick, MA.
12. Budynas, R.G., *Advanced Strength and Applied Stress Analysis*. 2nd ed. Introducing to the Finite Element Method. 1999, Boston, MA: McGraw-Hill.
13. Ansys, *Introduction to Ansys Meshing*. March 2010, Ansys Inc.
14. Roache, P.J., *Code verification by the method of manufactured solutions.* ASME Journal of Fluids Engineering, March 2002. **124**(1): p. 4-10.
15. Schwer, L., *Method of Manufactured Solutions: Demonstrations - Draft.* United States Association for Computation Mechanics, August 2002.
16. Folland, G.B., *Fourier Analysis and Its Application*. Some Boundary Value Problems. 1992: Thomson.
17. Gupta, M.M., R.P. Manohar and J.W.Stephenson, *A single cell high order scheme for the convection-diffusion equation with variable coefficient.* 1984(31): John Wiley & Sons, Ltd. p. 641-651.
18. Oberkampf, W.L. and T.G. Trucano, *Verification and validation in computational fluid dynamics.* Prog. Aerosp. Sci., 2002. **38**: p. 209-272.

APPENDIX A – MATLAB PDE Toolbox Overview

The MATLAB PDE toolbox generates an unstructured mesh within the defined domain and solves PDE problems with prescribed initial and/or boundary conditions by FEM. A sample of the GUI interface in the MATLAB PDE toolbox is shown in Fig. A.1.

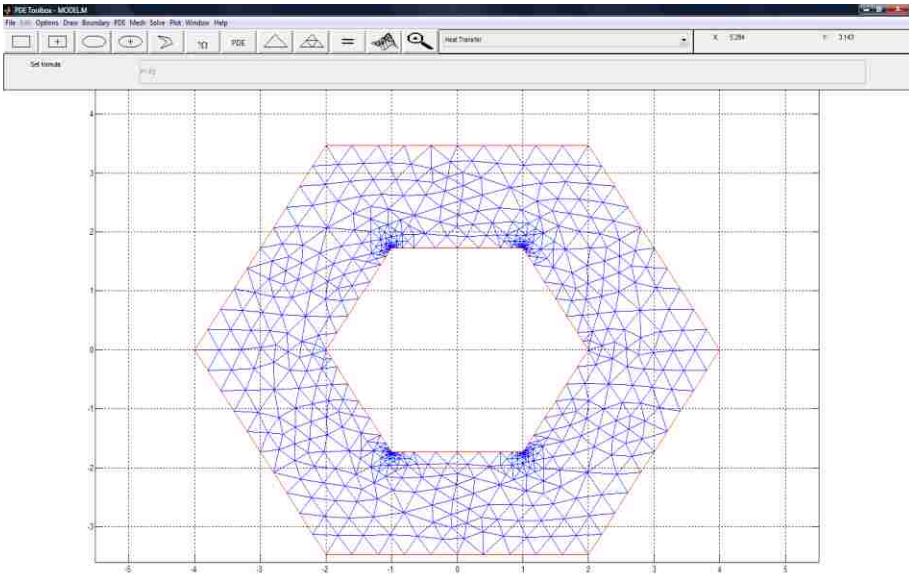


Figure A.1 MATLAB PDE toolbox GUI Interface

The procedures of using the PDE toolbox are as follows:

1) Create geometry for the PDE problem
2) Assign boundary conditions
3) Specify PDEs to be solved within the geometric domain
4) Generate unstructured mesh and make necessary refinement and quality improvements
5) Assign initial condition and specify time step size and time duration
6) Solve the PDE by FEM
7) Perform mesh adaptation if necessary

Once the PDE problem has been solved by the MATLAB PDE toolbox, the result is exported to the MATLAB command workspace for further analysis by MATLAB M-file.

APPENDIX B – CCFD Program Code Overview

The CCFD numerical methodology is programmed in MATLAB M-file, which contains three main components: geometry definition, computation process and post-processing. Geometry definition collects all the necessary information in a predefined or arbitrarily generated mesh topology. The main tasks for this component include the following:

1) Define all nodes within geometric domain
2) Create mesh topology among the defined nodes by Delaunay triangulation
3) Identify cell centroid in each triangle
4) Identify interior and boundary nodes
5) Assign initial and/or boundary conditions
6) Identify neighbouring triangles around each interior node
7) Determine distance of each neighbouring triangle centroid to an interior node, for every interior node
8) Determine differencing point information including: coordinates, end-edge nodes index and in-between distance, inscribed cell centroid index and in-between distance, adjacent cell centroid index and in-between distance
9) Determine coefficients in generalized polynomial transformed FD equations (2.20) and/or (2.23)

Computational time is determined during the computation process, which involves solving the PDE problems according to the following steps:

1) Determine differencing point values
2) Determine cell centroid value by applying generalized polynomial transformed FD equation
3) Update average relative difference value as in eqn. (4.1)
4) Evaluate interior and Neumann boundary condition nodes
5) Check if the average relative difference meets the convergence criterion. If yes, end iterative process. Otherwise go back to step #1.

Post-processing involves analysis of results and comparison between CCFD, FEM and analytical solutions. It contains the following aspects:

1) Compute exact nodal value if available

2) Compute nodal relative error (NRE) and nodal absolute error (AE) by comparing CCFD and exact solution by eqns. (3.11) and (3.12)
3) Generate RE and AE contour plots
4) Compute nodal relative difference (NRD) and nodal absolute difference (AD) at each node by comparing CCFD and FEM solution
5) Generate RD and AD contour plots
6) Compute overall RE, RD, RMSE and RMSD
7) Generate interpolation plots of solution values, NRE and NRD

APPENDIX C – Data Storage Allocation

All the necessary information to carry out the CCFD computation is shown in the following table.

Matrix Data Storage Allocation

Matrix	Description	1	2	3	4	5	6	7	8	9	10	11	Remark
A	Global node info	x-cod	y-cod	solver value	FEM value	exact value	RE	RD					Row # = # of nodes
I	Interior node info	global index	x-cod	y-cod	node value								Row # = # of interior nodes
B	Boundary node info	global index	x-cod	y-cod	node value								Row # = # of nodes; Max col = Max # of tris to a node
Tri	Triangle index nodes	node A	node B	node C									Max col = Max # of tris to an interior node
IC	Cell centroid information	x-cod	y-cod	current centroid value	updated centroid value								Row # = # of triangles
TI	Neigbouring triangle to interior node	tri #	tri #	tri #	tri #	tri #	tri #	….	….	….	….	….	Row # = # of triangles
LI	Distance from interior node to neibouring cell centroids	dis 1	dis 2	dis 3	dis 4	dis 5	dis 6	….	….	….	….	….	Infinity will be assigned if no more ∆s to the node; Row # = # of interior nodes; Max col = Max # of ∆s to interior node

S	South pt info	x-cod	y-cod	dis to node A	dis to node B	dis to node C	edge node 1	edge node 2	dis to inscribed cell centroid	Adjacent tri index	dis to adjacent cell centroid	value	Row # = # of triangles; Col 6 = 0 if lie on a node and Col 7 = node
N	North pt info	x-cod	y-cod	dis to node A	dis to node B	dis to node C	edge node 1	edge node 2	dis to inscribed cell centroid	Adjacent tri index	dis to adjacent cell centroid	value	
W	West pt info	x-cod	y-cod	dis to node A	dis to node B	dis to node C	edge node 1	edge node 2	dis to inscribed cell centroid	Adjacent tri index	dis to adjacent cell centroid	value	
E	East pt info	x-cod	y-cod	dis to node A	dis to node B	dis to node C	edge node 1	edge node 2	dis to inscribed cell centroid	Adjacent tri index	dis to adjacent cell centroid	value	
MS	Mid-south info	x-cod	y-cod	dis to node A	dis to node B				dis to node C	dis to inscribed cell centroid	dis to adjacent cell centroid	value	Row # = # of triangles
MN	Mid-north info	x-cod	y-cod	dis to node A	dis to node B				dis to node C	dis to inscribed cell centroid	dis to adjacent cell centroid	value	
MW	Mid-west info	x-cod	y-cod	dis to node A	dis to node B				dis to node C	dis to inscribed cell centroid	dis to adjacent cell centroid	value	
ME	Mid-east info	x-cod	y-cod	dis to node A	dis to node B				dis to node C	dis to inscribed cell centroid	dis to adjacent cell centroid	value	
M	Coefficients in polynomial mapping	S coef	MS coef	N coef	MN coef	W coef	MW coef	E coef	ME coef	IC coef	IC possion value		Row # = # of triangles

Table C.1 – Matrix data storage allocation

VITA AUCTORIS

James Situ was born in 1985 in China and immigrated to Canada in 2001. He graduated from High School in Dr. Norman Bethune C.I. in 2005. From there he went on to the University of Windsor, where he obtained a B.A.Sc. in Mechanical Engineering in 2008, B.A.Sc. in Industrial Engineering and B.Math in Mathematics Honours in 2009. He is currently a candidate for the Master degree in Mechanical Engineering at the University of Windsor and hopes to graduate in Spring 2011.