

**APPROXIMATE FACTORIZATION USING ACDI METHOD ON HYBRID GRIDS AND
PARALLELIZATION OF THE SCHEME**

**A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY**

BY

OĞUZ KAAAN ONAY

**IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
AEROSPACE ENGINEERING**

JANUARY 2013

Approval of the thesis:

**APPROXIMATE FACTORIZATION USING ACDI METHOD ON HYBRID GRIDS AND
PARALLELIZATION OF THE SCHEME**

submitted by **OĞUZ KAAN ONAY** in partial fulfillment of the requirements for the degree of
Master of Science in Aerospace Engineering Department, Middle East Technical University by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Ozan Tekinalp
Head of Department, **Aerospace Engineering**

Assoc. Prof. Dr. Oğuz Uzol
Supervisor, **Aerospace Engineering Dept., METU**

Assist. Prof. Dr. Nilay Sezer Uzol
Co-supervisor, **Mechanical Engineering Dept., TOBB-UET**

Examining Committee Members:

Prof. Dr. İsmail Hakkı Tuncer
Aerospace Engineering Dept., METU

Assoc. Prof. Dr. Oğuz Uzol
Aerospace Engineering Dept., METU

Prof. Dr. Serkan Özgen
Aerospace Engineering Dept., METU

Prof. Dr. Haluk Aksel
Mechanical Engineering Dept., METU

Assoc. Prof. Dr. D. Funda Kurtuluş
Aerospace Engineering Dept., METU

Date: 24.01.2013

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Oğuz Kaan Onay

Signature:

ABSTRACT

APPROXIMATE FACTORIZATION USING ACDI METHOD ON HYBRID GRIDS AND PARALLELIZATION OF THE SCHEME

Onay, Oğuz Kaan

M.Sc., Department of Aerospace Engineering

Supervisor: Assoc. Prof. Dr. Oğuz Uzol

Co-Supervisor: Assist. Prof. Dr. Nilay Sezer Uzol

January 2013, 98 pages

In this thesis study, a fast implicit iteration scheme called Alternating Cell Directions Implicit method is combined with Approximate Factorization scheme. This application aims to offer a mathematically well defined version of the Alternating Cell Directions Implicit Method and increase the accuracy of the iteration scheme that is being used for the numerical solutions of the partial differential equations.

The iteration scheme presented here is tested using unsteady diffusion equation, Laplace equation and advection-diffusion equation. The accuracy, convergence character and the stability character of the scheme compared with suitable iteration schemes for structured and unstructured quadrilateral grids. Besides, it is shown that the proposed scheme is applicable to triangular and hybrid polygonal grids.

A transonic full potential solver is generated using the current scheme. The flow around a 2-D cylinder is solved for subcritical and supercritical cases. Axi-symmetric flow around cylinder is selected as a benchmark problem since the potential flow around bodies with a blunt leading edge is a more challenging problem than slender bodies.

Besides, it is shown that, the method is naturally appropriate for parallelization using shared memory approach without using domain decomposition applications. The parallelization that is performed here is partially line, partially point parallelization. The performance of the application is presented for a 3-D unsteady diffusion problem using Cartesian cells and 2-D unsteady diffusion problem using both structured and unstructured quadrilateral cells.

Key words: Implicit Formulation, ACDI

ÖZ

HİBRİD ÇÖZÜM AĞLARI İÇİN DAHYKF KULLANILARAK YAKLAŞIK ÇARPANLARA AYIRMA VE ŞEMANIN PARALELLEŞTİRİLMESİ

Onay, Oğuz Kaan
Yüksek Lisans, Havacılık ve Uzay Mühendisliği
Tez Yöneticisi: Doç. Dr. Oğuz Uzol
Ortak Tez Yöneticisi: Yrd. Doç. Dr. Nilay Sezer Uzol
Ocak 2013, 98 sayfa

Bu tez çalışmasında Değişken Ardışık Hücre Yönlü Kapalı Formülasyon Metodu, Yaklaşık Çarpanlara Ayırma yaklaşımıyla birleştirilmiştir. Bu uygulama, Değişken Ardışık Hücre Yönlü Kapalı Formülasyon Metodu'nun matematiksel olarak daha iyi tanımlanmış bir versiyonunu ortaya koymayı ve kısmi diferansiyel denklemlerin çözümü için kullanılan bu metodun başarısını arttırmayı hedeflemektedir.

Burada sunulan iterasyon şeması daimi difüzyon denklemi, Laplace denklemi ve konveksiyon-difüzyon denklemi kullanılarak test edilmiştir. Şemanın doğruluk, yakınsama ve kararlılık karakterleri yapısal ve yapısal olmayan dörtgen sayısal ağlar kullanılarak uygun iterasyon şemalarıyla karşılaştırılmıştır. Ayrıca önerilen şemanın üçgen ve hibrit çokgen sayısal ağlara uygulanabilir olduğu gösterilmiştir.

Şema kullanılarak transonik tam potansiyel bir çözücü geliştirilmiştir. 2-B silindir etrafında kritik altı ve kritik üstü akış çözülmüştür. Problem olarak aksel simetrik silindir probleminin seçilmiş olmasının sebebi küt hücum kenarlı cisimler etrafındaki akış çözümlerinin ince cisimler etrafındaki akış çözümlerinden daha zor olmasıdır.

Aynı zamanda metodun doğası gereği paylaşımlı bellek kullanımında paralelleştirmeye uygun olduğu gösterilmiştir. Burada uygulanan paralelleştirme kısmen çizgi, kısmen nokta paralellidir. Uygulamanın performansı basit bir 3-B daimi difüzyon problemi üzerinde Kartezyen elemanlar kullanılarak ve 2-B daimi difüzyon denklemi üzerinde yapısal ve yapısal olmayan dört kenarlı elemanlar kullanılarak sunulmaya çalışılmıştır.

Anahtar Kelimeler: Kapalı Formülasyon, DAHYKF

To my precious Tuba...

ACKNOWLEDGEMENTS

I would like to thank my supervisors Assoc. Prof. Dr. Oğuz Uzol and Assist. Prof. Dr. Nilay Sezer Uzol for their supports during my master study. I also want to thank to Dr. Ali Ruhşen Çete for sharing his valuable experience on numerical methods and for his creative ideas.

Much appreciation is also expressed to my previous workmate Halit Kutkan because of sharing my workload in the office at the most tiring days of my study.

I wish to express my appreciation to my mother and father because of their neverending love and support. I hope my brother Cihan would forgive me. I couldn't deal with him enough. I will work hard to make up for our lost time!

My wife Tuba never gave up on me and she was always very patient and relaxing at the instants when I am nervous. I am very thankful to her for being in my life and for her precious love. The duration of this study again made me remember that marrying her was my best decision in my life.

TABLE OF CONTENTS

ABSTRACT	iv
ÖZ	v
ACKNOWLEDGEMENTS	vii
TABLE OF CONTENTS	viii
LIST OF TABLES	x
LIST OF FIGURES.....	xi
LIST OF SYMBOLS	xiv
LIST OF ABBREVIATIONS	xv
CHAPTERS	
1. INTRODUCTION.....	1
1.1 Background	1
1.2 Point Gauss Seidel iteration method (PGS).....	2
1.3 Line Gauss Seidel iteration method (LGS).....	3
1.4 Alternating directions implicit method (ADI)	4
1.5 Alternating cell directions implicit method (ACDI).....	5
1.6 Classical approximate factorization	7
1.7 Implicit Laasonen Method.....	8
1.8 Present approach and aims of the study	8
2. NUMERICAL METHOD	11
2.1 Approximate factorization method appropriate for ACDI	11
2.2 Approximate factorization with ACDI	14
2.2.1 Difference of the approach from fractional time stepping methods	18
2.2.2 Flux calculation using inverse distance weighting	20
2.2.3 Flux calculation using trapezoidal rule.....	21
2.2.4 Von Neumann stability analysis.....	23
2.3 Odd number of edges	25
2.4 K-exact least squares	27
2.5 Parallelization strategy	28
3. RESULTS AND DISCUSSION	31
3.1 Validation of the numerical approach using unsteady diffusion equation.....	31
3.1.1 Problem specifications for unsteady diffusion equation solution comparisons	31
3.1.2 Validation of the proposed approximate factorization	33
3.1.3 Validation of the proposed approximate factorization with ACDI.....	35
3.1.3.1 Comparison of different methods using structured grid	35
3.1.3.2 Comparison of different methods using unstructured grid	39
3.1.4 Comparison of different grid types.....	42
3.1.5 Comparison of inverse distance weighting and trapezoidal rule flux calculations	46
3.2 Comparison of convergence characters using steady Laplace equation	48
3.2.1 Problem specifications for convergence character comparisons with Laplace Equation ..	48
3.2.2 Comparison of convergence characters using structured grid	49
3.2.3 Comparison of convergence characters using unstructured grid	52
3.3 Time accurate comparisons using advection-diffusion equation.....	55
3.3.1 Problem specifications for advection-diffusion equation comparisons	55
3.3.2 Comparison for advection-diffusion equation using structured grid	57
3.3.3 Comparison for advection-diffusion equation using unstructured grid.....	60
3.4 Transonic full potential solver using AF ACDI	64

3.4.1 Problem specifications for full potential equation solution.....	66
3.4.2 Full potential equation solution using Crank-Nicolson AF ACDI.....	66
3.4.3 Full potential equation solution using simple AF ACDI.....	69
3.5 Performance of parallelization	72
3.5.1 Performance of parallelization for finite difference Cartesian case using proposed approximate factorization	72
3.5.2 Performance of parallelization for AF ACDI Using 2-D Quadrilateral structured and unstructured grids	76
4. CONCLUSIONS.....	79
5. FUTURE WORK.....	81
REFERENCES	83
APPENDIX A-LIST OF THE TEST CASES.....	85
APPENDIX B-GRIDS USED FOR THE TEST CASES	89
APPENDIX C-EQUATIONS USED FOR THE TEST CASES.....	93
APPENDIX D- NUMERICAL STABILITY TESTS	97

LIST OF TABLES

TABLES

Table 3.1 CPU time required for the methods, 20x20 structured cells.....	38
Table 3.2 CPU time required for the methods, 297 unstructured cells.....	41
Table 3.3 CPU times and speedup ratios for sweep parallelization (50 time iterations).....	75
Table 3.4 CPU times and speedup ratios for general parallelization (50 time iterations).....	75
Table 3.5 CPU times and speedup ratios using structured grid (10000 elements 1000 iterations).....	78
Table 3.6 CPU times and speedup ratios using structured grid (8828 elements 1000 iterations).....	78
Table 3.7 Selected points for the comparison of sequential and different number of CPU solutions.....	78
Table 3.8 Comparison of sequential and different number of CPU solutions at selected points.....	78

LIST OF FIGURES

FIGURES

Figure 1.1 Example solution domain for PGS.....	2
Figure 1.2 Example solution domain for LGS.....	3
Figure 1.3 Sweep directions for ADI.....	4
Figure 1.4 Example alternating cell directions for quadrilateral cells.....	5
Figure 2.1 Control Volume with N edges.....	15
Figure 2.2 Cell directions drawn travelling opposing edges of cells.....	17
Figure 2.3 Notation used for low order flux calculation on a sample polygonal cell and neighbor cell.....	21
Figure 2.4 Notation used for high order flux calculation on a sample quadrilateral cell and neighbor cells.....	22
Figure 2.5 Example Cartesian cell and its neighbors.....	23
Figure 2.6 Split edge of triangular element at the boundary and cell directions.....	26
Figure 2.7 Zero length edge of triangular element inside the domain and cell directions.....	26
Figure 2.8 Stencil used for a reconstruction at node.....	27
Figure 2.9 Difference between usage of shared and distributed memory approaches for parallelization.....	29
Figure 3.1 Heat conduction problem on a rectangular plate.....	31
Figure 3.2 Non-dimensional temperature contours-Analytical.....	33
Figure 3.3 Mean Error histories of the numerical solutions for 20x20 structured cells.....	34
Figure 3.4 Mean Error histories of the numerical solutions for 40x40 structured cells.....	34
Figure 3.5 Analytical results through $Y=0.5$ constant line.....	35
Figure 3.6 Error distribution through $Y=0.5$ constant line $t=0.1$ structured grid.....	36
Figure 3.7 Error distribution through $Y=0.5$ constant line $t=0.2$ structured grid.....	36
Figure 3.8 Error distribution through $Y=0.5$ constant line $t=0.3$ structured grid.....	37
Figure 3.9 Variation of the absolute error at point $X=0.5$ $Y=0.5$ structured grid.....	38
Figure 3.10 Error distribution through $Y=0.5$ constant line $t=0.1$ unstructured grid.....	39
Figure 3.11 Error distribution through $Y=0.5$ constant line $t=0.2$ unstructured grid.....	40
Figure 3.12 Error distribution through $Y=0.5$ constant line $t=0.3$ unstructured grid.....	40
Figure 3.13 Variation of the absolute error at point $X=0.5$ $Y=0.5$ unstructured grid.....	41
Figure 3.14 297 element unstructured grid.....	42
Figure 3.15 Non-dimensional temperature contours – Numerical.....	42
Figure 3.16 (a) 45 element structured grid. (b) 46 element trigonal grid. (c) 43 element quadrilateral unstructured grid. (d) 44 element hybrid polygonal grid.....	43
Figure 3.17 (a) Non-dimensional theta contours and (b) error contours for triangular grid (c) Non-dimensional theta contours and (d) error contours for quadrilateral grid and (e) Non-dimensional theta contours and (f) error contours for hybrid polygonal grid.....	44
Figure 3.18 (a) Non-dimensional theta contours and (b) error contours for structured grid.....	45
Figure 3.19 Cell directions for polygonal grid after splitting the edges of pentagonal elements.....	45
Figure 3.20 Error histories of the numerical solutions for different grid types.....	46
Figure 3.21 Error histories of the numerical solutions using higher and lower order flux calculation approaches with ACADI AF for unstructured grids.....	47
Figure 3.22 (a) 297 element and (b) 1283 element unstructured grids that used for the flux calculation method comparison.....	47
Figure 3.23 The stream function contours obtained using the analytical solution.....	49
Figure 3.24 Number of iteration required for the convergence versus time pseudo-step size-structured grid.....	50
Figure 3.25 CPU time required for the convergence versus pseudo-time step size-structured grid..	50
Figure 3.26 Convergence histories of simple AF ACADI and Point Gauss Seidel Method with optimum time step sizes with 500 element structured grid.....	51
Figure 3.27 Contours of stream function around 2-D cylinder obtained with using simple AF ACADI with 500 element structured grid.....	51

Figure 3.28 (a) 500 element structured grid used for the comparison of convergence behaviors of different methods (b) absolute error contours of stream function with the usage of simple AF ACDI.....	52
Figure 3.29 Number of iteration required for the convergence versus pseudo-time step size-unstructured grid.....	53
Figure 3.30 CPU time required for the convergence versus pseudo-time step size-unstructured grid.....	53
Figure 3.31 Convergence histories of simple AF ACDI and Point Gauss Seidel Method with optimum time step sizes for 789 element unstructured grid.....	54
Figure 3.32 Contours of stream function around 2-D cylinder obtained with using simple AF ACDI with 789 element unstructured grid.....	54
Figure 3.33 (a) 789 element unstructured grid used for the comparison of convergence behaviors of different methods (b) absolute error contours of stream function with the usage of simple AF ACDI.....	54
Figure 3.34 Contours of mass concentration-analytical.....	56
Figure 3.35 Comparison of concentration distributions obtained with analytical solution and numerical solution using simple AF ACDI for 16x32 structured cells dt=0.01.....	58
Figure 3.36 Comparison of concentration distributions obtained with analytical solution and numerical solution using Laasonen method for 16x32 structured cells dt=0.01.....	58
Figure 3.37 Comparison of concentration distributions obtained with analytical solution and numerical solution using Runge-Kutta order 4 method for 16x32 structured cells dt=0.01.....	58
Figure 3.38 Comparison of concentration distributions obtained with analytical solution and numerical solution using ACDI SIMPLE AF for 16x32 structured cells dt=0.25.....	59
Figure 3.39 Comparison of concentration distributions obtained with analytical solution and numerical solution using Laasonen method for 16x32 structured cells dt=0.25.....	59
Figure 3.40 Comparison of concentration distributions obtained with analytical solution and numerical solution using Runge-Kutta order 4 method for 16x32 structured cells dt=0.25.....	59
Figure 3.41 Comparison of concentration distributions obtained with analytical solution and numerical solution using ACDI SIMPLE AF for 624 element unstructured cells dt=0.01.....	61
Figure 3.42 Comparison of concentration distributions obtained with analytical solution and numerical solution using Laasonen method for 624 element unstructured cells dt=0.01.....	61
Figure 3.43 Comparison of concentration distributions obtained with analytical solution and numerical solution using Runge-Kutta order 4 method for 624 element unstructured cells dt=0.01.....	61
Figure 3.44 Comparison of concentration distributions obtained with analytical solution and numerical solution using ACDI SIMPLE AF for 624 element unstructured cells dt=0.25.....	62
Figure 3.45 Comparison of concentration distributions obtained with analytical solution and numerical solution using Laasonen method for 624 element unstructured cells dt=0.25.....	62
Figure 3.46 Unstable results of Runge-Kutta order 4 method for 624 element unstructured cells dt=0.25.....	62
Figure 3.47 (a) Grid used for unstructured grid comparisons (b) evolution of mass concentration for simple AF-ACDI method solution with dt=0.25s.....	63
Figure 3.48 (a) Structured grid and (b) unstructured grid used for the full potential equation solution.....	66
Figure 3.49 Mach contours around the cylinder and the appearance of the structured grid $Ma_\infty=0.25$ (Crank Nicolson AF ACDI solution).....	67
Figure 3.50 Mach contours around the cylinder and the appearance of the unstructured grid $Ma_\infty=0.25$ (Crank Nicolson AF ACDI solution).....	67
Figure 3.51 Comparison of pressure coefficients with the analytical results adapted from Saied and Alireza [29] $Ma_\infty=0.25$ (Crank Nicolson AF ACDI solution).....	67
Figure 3.52 Mach contours around the cylinder and the appearance of the structured grid $Ma_\infty=0.7$ (Crank Nicolson AF ACDI solution).....	68
Figure 3.53 Mach contours around the cylinder and the appearance of the unstructured grid $Ma_\infty=0.7$ (Crank Nicolson AF ACDI solution).....	68
Figure 3.54 Comparison of pressure coefficients with the full potential solutions adapted from Djojodihardjo and Widodo [30] $Ma_\infty=0.7$ (Crank Nicolson AF ACDI solution).....	69
Figure 3.55 Comparison of pressure coefficients with the analytical results adapted from Saied and Alireza [29] $Ma_\infty=0.25$ (Simple AF ACDI solution).....	70

Figure 3.56 Comparison of pressure coefficients with the analytical results adapted from Djojodihardjo and Widodo [30] $Ma_\infty=0.7$ (Simple AF ACIDI solution).....	71
Figure 3.57 Example residual histories for subcritical and supercritical cases using 30x50 structured grid.....	71
Figure 3.58 Mach contours around the cylinder and the appearance of the 30x60 structured grid that is clustered at shock region.....	72
Figure 3.59 Speedup ratios of parallelization trials and comparisons with theoretically 90% and 97% parallelized codes.....	74
Figure 3.60 Parallelization efficiency for sweep parallelization and general parallelization.....	74
Figure 3.61 Non-dimensional temperature contours-3-D Cartesian case.....	75
Figure 3.62 (a) 10000 element structured grid and (b) 8823 element unstructured grid used for the parallelization performance tests.....	76
Figure 3.63 Speedup ratios of parallelization trials and comparisons with theoretically 90% and 95% parallelized codes for structured and unstructured grids (1000 time iterations).....	77
Figure 3.64 Parallelization efficiency for structured and unstructured grids.....	77

LIST OF SYMBOLS

A	Area
Bi	Biot number
c	Concentration of mass
D	Diffusion coefficient
L	Length
M	Mass
N	Number of cells/nodes
P	Proportion
p	pressure
Pe	Pecklet Number
T	Temperature
u, v	Velocity components in x and y directions
X, Y	Non-dimensional x and y coordinates
α	Courant number
θ	Non-dimensional temperature
v	Von-Neumann number
ξ, ζ	Growth factors
ρ	Density
τ	Non-dimensional time
φ	Velocity potential
ψ	Stream function

LIST OF ABBREVIATIONS

ACDI	Alternating Cell Directions Implicit Method
ADI	Alternating Directions Implicit Method
AF	Approximate Factorization
Crank Nicolson AF ACDI	Crank Nicolson Approximate Factorization combined with Alternating Cell Directions Implicit Method
LGS	Line Gauss Seidel Iteration Method
PGS	Point Gauss Seidel Iteration Method
RK4	Runge-Kutta order 4 Method
Simple AF ACDI	Simple Approximate Factorization combined with Alternating Cell Directions Implicit Method

CHAPTER 1

INTRODUCTION

1.1 Background

The numerical solution of partial differential equations is the basis of the scientific computation applications. Solutions are being performed using discrete forms of the equations using finite cells instead of infinitesimal extents. The discretization approaches of equations specify the classification of the solution methods.

One of the classifications can be stated as having implicit formulation or explicit formulation. Implicit formulations result with linear sets of equations in matrix forms whereas explicit formulations use directly the previous iteration or time step values to obtain the current iteration or time step value. The explicit schemes are more common with unstructured grids [1],[2].

Implicit formulations are the approaches that are more difficult to program than explicit schemes and have better convergence character. Having a better convergence character can be used as an advantage if fast implicit methods like Alternating Directions Implicit (ADI) Method are applied. These kinds of fast implicit formulations are usually appropriate for structured grids [2],[3] (grids that are composed of ordered finite cells on the solution domain). The formulations of them generally use tri-diagonal or penta-diagonal matrix solutions and these kinds of matrices are relatively easier to solve.

Fully implicit schemes require more computational time since they end up with mass matrices for the solution and the efforts to obtain quicker results with these applications are generally focused on faster solution methods of mass matrices [4].

Alternating Cell Directions Implicit method is a fast implicit scheme that can be used for both structured and unstructured grids [3]. This ability is not very common for fast implicit schemes. Mavriplis [2] states that it is possible to generate grid directions with the edges of triangular elements and these directions can be used for line implicitness, but in fact these sweep directions are not unique for an unstructured grid and they are defined by a method selected by the programmer. One of the most important studies on line implicitness of unstructured grids is the usage of Hamiltonian Tours [5] of Hassan et al [6]. Venkatakrishnan states that [4] there might exist many of the Hamiltonian Tours and there might be no of them for a 2-D unstructured mesh.

The ACDI method is offered in order to combine the advantages of successful convergence characters of fast implicit methods with the easier meshing advantage of unstructured grids. The method is inspired from ADI Methods [7],[8],[9] but its degree of implicitness is in the order of Line Gauss Seidel Iteration method in case of the existence of quadrilateral elements. Since one of the objectives of this study is to increase the implicitness level of the ACDI method to the order of approximate factorization [10], classical approximate factorization approach will be explained briefly after Point Gauss Seidel, Line Gauss Seidel, Alternating Directions Implicit and Alternating Cell Directions Implicit Methods. Then the fully implicit Laasonen method will be explained briefly.

1.2 Point Gauss Seidel Iteration Method (PGS)

Laplace equation (1.1) is used as model equation to show the discretization of Point Gauss Seidel, Line Gauss Seidel and the Alternating Directions implicit methods.

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0 \tag{1.1}$$

If the node based discretization using second order central differences is written for the sample grid shown in Figure 1.1. as given below;

$$2 \left(1 + \left(\frac{\Delta x}{\Delta y} \right)^2 \right) T_{i,j}^{n+1} = \left(\frac{\Delta x}{\Delta y} \right)^2 (T_{i,j+1}^n + T_{i,j-1}^n) + T_{i+1,j}^n + T_{i-1,j}^n \tag{1.2}$$

If the equation (1.2) is inspected well it can be seen that the terms with the indices $(i-1,j)$ and $(j-1,i)$ are written implicitly to the right hand side of the equation. If the solution procedure begins from the corners where the boundary conditions exist as in Figure 1.1, then the implicitness of these terms can be provided for all of the discretized equations written for whole nodes inside the domain. This approach can be applied to unstructured grids as well as the structured grids and provides an enhanced convergence character to the method. Line Gauss Seidel and Alternating Directions Implicit methods have also relatively good convergence characters but they can not be applied to unstructured grids.

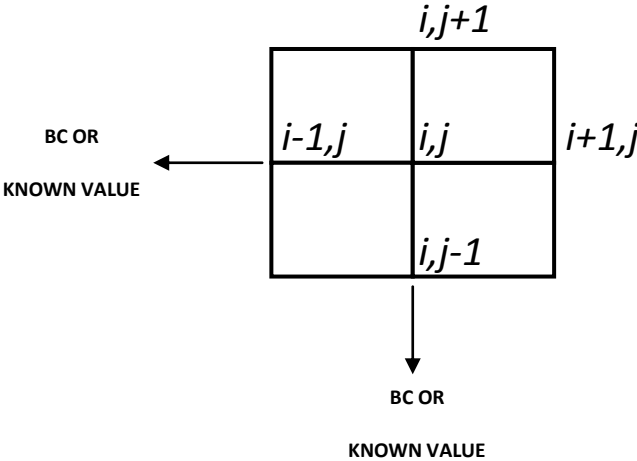


Figure 1.1 Example solution domain for PGS

1.3 Line Gauss Seidel Iteration Method (LGS)

The Laplace equation given with (1.1) is discretized as shown in equation (1.3) using finite difference approximation and node based approach.

$$T_{i-1,j}^{n+1} - 2 \left(1 + \left(\frac{\Delta x}{\Delta y} \right)^2 \right) T_{i,j}^{n+1} + T_{i+1,j}^{n+1} = - \left(\frac{\Delta x}{\Delta y} \right)^2 (T_{i,j+1}^n + T_{i,j-1}^n) \quad (1.3)$$

If equation (1.3) is written for the marked nodes given in Figure 1.2 the cells with $j+1$ indices remain explicit in the left hand side of the discretized equation. Each equation set written for constant j indices, generate a single tri-diagonal matrix to be solved. The equation sets are solved beginning with $j=1$ constant line and the solution of the j^{th} equation set requires the results of the $(j-1)^{\text{th}}$ tri-diagonal matrix solution.

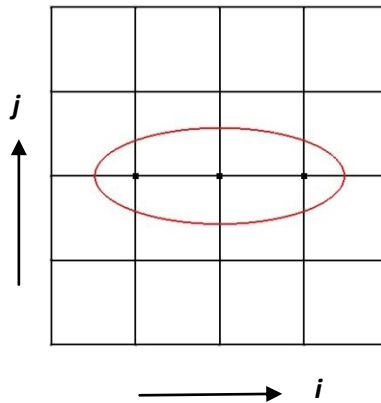


Figure 1.2 Example solution domain for LGS

1.4 Alternating Directions Implicit Method (ADI)

ADI Method uses two implicit iteration cycles for x and y directions over the domain. An example domain and solution directions are shown with Figure 1.3.

Each of the iteration cycles ends up in a solution for half iteration time step for a 2-dimensional domain. This solution is not meaningful in the manner of unsteadiness, although the semi-solution is called half iteration time step solution.

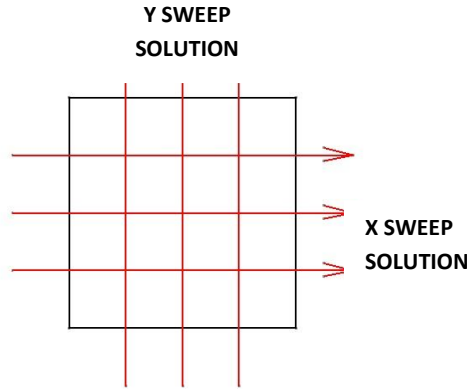


Figure 1.3 Sweep directions for ADI

If the Laplace equation given with (1.1) is discretized using node based finite difference approximation as shown below;

$$\begin{aligned}
 T_{i-1,j}^{n+1/2} - 2 \left(1 + \left(\frac{\Delta x}{\Delta y} \right)^2 \right) T_{i,j}^{n+1/2} + T_{i+1,j}^{n+1/2} &= - \left(\frac{\Delta x}{\Delta y} \right)^2 (T_{i,j+1}^n + T_{i,j-1}^{n+1/2}) \\
 \left(\frac{\Delta x}{\Delta y} \right)^2 T_{i,j-1}^{n+1} - 2 \left(1 + \left(\frac{\Delta x}{\Delta y} \right)^2 \right) T_{i,j}^{n+1} + T_{i,j+1}^{n+1} &= -T_{i+1,j}^{n+1/2} - T_{i-1,j}^{n+1}
 \end{aligned} \tag{1.4}$$

Both of the discretized equations given with (1.4) generate tri-diagonal matrices to be solved for each sweep. Having tri diagonal matrices lessens the computational time and programming effort.

The oncoming sweep solution has to use the results of the previous one for such a scheme. For example second x sweep solution can not be performed before the end of the first x sweep solution, also to begin y sweep calculations, x sweep calculations have to be completed for such a discretization. As obviously seen, the scheme requires structured grids and the application can also be used where the grid transformation is required [11].

1.5 Alternating Cell Directions Implicit Method (ACDI)

Alternating cell directions implicit method uses the cell directions that generated via travelling over the opposing edges of the unstructured quadrilateral cells instead of the x and y sweep directions of classical ADI method. This approach makes it useful for both structured and unstructured grids. Example cell directions passing through a sample cell are given in Figure 1.4

The method has previously been used to generate an incompressible Navier-Stokes solver and it has been proven that it gives more accurate results than Point Gauss Seidel Iteration Method [12]

The notation given with Figure 1.4 is for cell centered based finite volume approach of ACDI method. Only two cell directions can pass through each cell center of quadrilateral cells and these directions become unique for each mesh. The points P, M and N are the cell centers where the dependent variable is written implicitly into the discretized equation and all of these points take place over the cell direction shown with continuous red arrow. The dashed red arrow represents the other cell direction that also passes through the cell with center M. Continuing to write the discretized equation over the same solution band results with a tri-diagonal matrix to be solved.

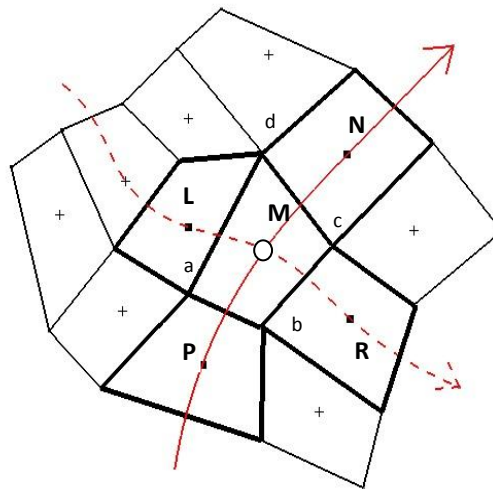


Figure 1.4 Example alternating cell directions for quadrilateral cells

Integrating the unsteady diffusion equation over the cell with center M yields:

$$\frac{\partial T}{\partial t} - \nabla^2 T = 0 \quad (1.5)$$

$$\int_A \frac{\partial T}{\partial t} dA = \int_A \nabla^2 T dA = \int_A \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \right) dA \quad (1.6)$$

Applying the Green's theorem gives:

$$\frac{\partial}{\partial t} \int_A T dA = \int_c \frac{\partial T}{\partial n} ds = \int_c \left(\frac{\partial T}{\partial x} \frac{\partial x}{\partial n} + \frac{\partial T}{\partial y} \frac{\partial y}{\partial n} \right) ds \quad (1.7)$$

Also:

$$\frac{dx}{dn} = \frac{dy}{ds} \quad \frac{dy}{dn} = -\frac{dx}{ds} \quad (1.8)$$

If the identities given with (1.8) are written into (1.7);

$$A_{abcd} \frac{\partial \bar{T}}{\partial t} = \int_c \left(\frac{\partial T}{\partial x} \frac{\partial y}{\partial s} + \frac{\partial T}{\partial y} \frac{\partial x}{\partial s} \right) ds = \int_{abcd} \frac{\partial T}{\partial x} dy - \int_{abcd} \frac{\partial T}{\partial y} dx \quad (1.9)$$

Where:

$$\bar{T} = \frac{1}{A} \int_A T dA$$

The integrations defined at boundary $abcd$ have to be calculated. The average fluxes at each edge of the cells are calculated using virtual areas. The corners of these virtual areas are the nodes of the edge, center of the cell itself and the center of the neighbor cell. For example MaPb is the face that is used for the flux calculation at the edge ab of the cell which has the center M at Figure 1.4.

The flux expressions are inserted into (1.9) and the values at points P, M and N are held implicit in the discretized equation. These points take place on the solution band of interest.

Writing the discretized equation through a solution band ends up in a tri-diagonal matrix. Solution of these tri-diagonal matrices for all solution bands gives a single time iteration step solution.

Solving the tri-diagonal matrices one by one gives the results for the current iteration or time step, but there is no appropriate or well-defined selection method for the sequence of matrix solutions. One of

the aims of this thesis study is to present a mathematically well defined approach that uses Alternating Cell Directions concept.

1.6 Classical Approximate Factorization

If the unsteady diffusion equation is discretized using Crank-Nicolson scheme and node based finite difference approach;

$$\frac{T_{i,j}^{n+1} - T_{i,j}^n}{\Delta t} = \frac{1}{2} \left[\frac{T_{i+1,j}^{n+1} - 2T_{i,j}^{n+1} + T_{i-1,j}^{n+1}}{\Delta x^2} + \frac{T_{i+1,j}^n - 2T_{i,j}^n + T_{i-1,j}^n}{\Delta x^2} + \frac{T_{i,j+1}^{n+1} - 2T_{i,j}^{n+1} + T_{i,j-1}^{n+1}}{\Delta y^2} + \frac{T_{i,j+1}^n - 2T_{i,j}^n + T_{i,j-1}^n}{\Delta y^2} \right] \quad (1.10)$$

Writing the equation in operator form and rearranging gives:

$$\left[1 - \frac{\Delta t}{2} \frac{\partial^2}{\partial x^2} - \frac{\Delta t}{2} \frac{\partial^2}{\partial y^2} \right] T_{i,j}^{n+1} = \left[1 + \frac{\Delta t}{2} \frac{\partial^2}{\partial x^2} + \frac{\Delta t}{2} \frac{\partial^2}{\partial y^2} \right] T_{i,j}^n \quad (1.11)$$

Approximately factorizing the equation above;

$$\left[1 - \frac{\Delta t}{2} \frac{\partial^2}{\partial x^2} \right] \left[1 - \frac{\Delta t}{2} \frac{\partial^2}{\partial y^2} \right] T_{i,j}^{n+1} = \left[1 + \frac{\Delta t}{2} \frac{\partial^2}{\partial x^2} \right] \left[1 + \frac{\Delta t}{2} \frac{\partial^2}{\partial y^2} \right] T_{i,j}^n \quad (1.12)$$

A solution procedure can be applied as shown below;

$$\begin{aligned} \left[1 - \frac{\Delta t}{2} \frac{\partial^2}{\partial x^2} \right] T_{i,j}^* &= \left[1 + \frac{\Delta t}{2} \frac{\partial^2}{\partial y^2} \right] T_{i,j}^n \\ \left[1 - \frac{\Delta t}{2} \frac{\partial^2}{\partial y^2} \right] T_{i,j}^{n+1} &= \left[1 + \frac{\Delta t}{2} \frac{\partial^2}{\partial x^2} \right] T_{i,j}^* \end{aligned} \quad (1.13)$$

Writing the discretized form which is an approximation of Crank-Nicolson scheme gives:

$$\begin{aligned} T_{i,j}^* - \frac{\Delta t}{2} \left[\frac{T_{i+1,j}^* - 2T_{i,j}^* + T_{i-1,j}^*}{\Delta x^2} \right] &= T_{i,j}^n + \frac{\Delta t}{2} \left[\frac{T_{i,j+1}^n - 2T_{i,j}^n + T_{i,j-1}^n}{\Delta y^2} \right] \\ T_{i,j}^{n+1} - \frac{\Delta t}{2} \left[\frac{T_{i,j+1}^{n+1} - 2T_{i,j}^{n+1} + T_{i,j-1}^{n+1}}{\Delta y^2} \right] &= T_{i,j}^* + \frac{\Delta t}{2} \left[\frac{T_{i+1,j}^* - 2T_{i,j}^* + T_{i-1,j}^*}{\Delta x^2} \right] \end{aligned} \quad (1.14)$$

The factorization yields with an additional term in the formulation and this additional term in the operator form is:

$$\frac{1}{4}\Delta t^2 \frac{\partial}{\partial x^2} \left[\frac{\partial}{\partial y^2} (T_{i,j}^{n+1} - T_{i,j}^n) \right] \quad (1.15)$$

The solution of (1.13) can be obtained using x and y sweeps that shown in Figure 1.3 as ADI scheme. Different than the ADI scheme x sweep solutions do not require any other x sweep solution but all x sweep solutions have to be completed before the calculation of the y sweep equations.

Classical Approximate Factorization is an approximation of fully implicit schemes rather than a line implicit scheme.

1.7 Implicit Laasonen Method

The model equation (1.5) is discretized as given below;

$$\frac{T_{i,j}^{n+1} - T_{i,j}^n}{\Delta t} = \frac{1}{2} \left[\frac{T_{i+1,j}^{n+1} - 2T_{i,j}^{n+1} + T_{i-1,j}^{n+1}}{\Delta x^2} + \frac{T_{i,j+1}^{n+1} - 2T_{i,j}^{n+1} + T_{i,j-1}^{n+1}}{\Delta y^2} \right] \quad (1.16)$$

As seen all the terms instead of the central node variable in the time derivative are written implicitly. If the equation (1.16) is written for a domain that is composed of $N_x N_y$ nodes $N_x N_y$ linear equations will be obtained, thus $(N_x N_y)$ by $(N_x N_y)$ matrix has to be solved to obtain the dependent variable values of the next time step. Hoffman [13] states that the implicit methods have outstanding convergence characters even with very large time step sizes, but the accuracy decreases with the increased time step size. Besides, solution of mass matrices take relatively long CPU times when compared to tri-diagonal or penta-diagonal matrix solutions. Most of the efforts on speeding up the fully implicit schemes are on fast solution of the mass matrices [4].

1.8 Present Approach and Aims of the Study

The main aim of this thesis study is to combine an approximate factorization scheme with ACDI [10] time iteration scheme in order to increase the implicitness level and the accuracy of the previous ACDI studies [3],[12]. Although the main study is carried on using cell centered finite volume discretization; the approximate factorization approach used for this study will also be explained for node based finite difference discretization on structured grids for clearness, also it is aimed to serve a mathematically well defined approach.

The unsteady diffusion equation, Laplace equation and unsteady advection-diffusion equation are selected as the model equations for the validation of the method. Two different approximate factorization methods are adapted to ACDI method. One of these factorizations uses the Crank-Nicolson discretization whereas the other one uses relatively simpler approach for the spatial derivatives, also two different flux calculation approaches are used to calculate the flux integrals at the boundaries of the cells. One of these methods uses inverse distance weighting interpolation and a finite difference approach which is relatively low order and is easier to program. The second method uses virtual faces on edges to calculate the flux integrals with trapezoidal rule and it is similar with the flux calculation of previous ACDI study. Inverse distance weighting flux calculation is more appropriate for three dimensional applications. Low order approach is presented since it is more useful and easy to program and high order one is represented in order to compare the success of the current approach with previous ACDI approach.

It is previously shown by Çete [3] that, the ACDI is appropriate for structured, unstructured quadrilateral and combination of triangular and quadrilateral quad-dominant unstructured grids. The approach is generalized for all kinds of polygonal cells with this study. The data structure of the previous programming approach is reconfigured to obtain a flexible data structure that suits different number of edge numbers on cells.

Another objective is to show that the approximate factorization used for this study, serves a scheme that can be easily parallelized for shared memory applications without using domain decomposition. Unsteady diffusion equation is solved using $100 \times 100 \times 100$ Cartesian cells and the speedup ratios are calculated up to 32 processors with a 48 GB RAM server. Also the performance of parallelization is shown for both structured and unstructured quadrilateral grids again using the unsteady diffusion equation.

CHAPTER 2

NUMERICAL METHOD

2.1 Approximate Factorization Method Appropriate for ACDI

Writing the unsteady diffusion equation in operator form where the spatial derivatives symbolize second order central differencing at nodes of equi-sized structural cells;

$$\frac{T_{i,j}^{n+1} - T_{i,j}^n}{\Delta t} - \left[\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \right] T_{i,j}^{n+1} = 0 \quad (2.1)$$

Rearranging gives;

$$\left[1 - \Delta t \frac{\partial^2}{\partial x^2} - \Delta t \frac{\partial^2}{\partial y^2} \right] T_{i,j}^{n+1} = T_{i,j}^n \quad (2.2)$$

Applying the simple factorization as shown below [14];

$$\left[1 - \Delta t \frac{\partial^2}{\partial x^2} \right] \left[1 - \Delta t \frac{\partial^2}{\partial y^2} \right] T_{i,j}^{n+1} = T_{i,j}^n \quad (2.3)$$

The additional term of factorization will be;

$$\Delta t^2 \frac{\partial^2}{\partial x^2} \left[\frac{\partial^2}{\partial y^2} (T_{i,j}^{n+1}) \right] \quad (2.4)$$

If semi-solutions for equation (2.3) are assigned as shown below;

$$T_1 = \left[1 - \Delta t \frac{\partial^2}{\partial y^2} \right] T_{i,j}^{n+1} \quad (2.5)$$

$$T_2 = \left[1 - \Delta t \frac{\partial^2}{\partial x^2} \right] T_{i,j}^{n+1} \quad (2.6)$$

Then equation (2.3) can be written as two independent equations;

$$\left[1 - \Delta t \frac{\partial^2}{\partial x^2} \right] T_1 = T_{i,j}^n \quad (2.7)$$

$$\left[1 - \Delta t \frac{\partial^2}{\partial y^2}\right] T_2 = T_{i,j}^n \quad (2.8)$$

Using same terms in the right hand sides of the equations and using independent semi-solutions at left hand sides makes the discretization appropriate for ACIDI.

The equations given as (2.7) and (2.8) result with tri-diagonal matrices if second order central difference is used for spatial derivatives. (2.7) has to be solved for all x sweep directions and (2.8) has to be solved for all y sweep directions, but the semi-solutions T_1 and T_2 do not require each other to be calculated. The procedure to obtain $T_{i,j}^{n+1}$ using these semi-solutions is as given below.

Summation of expressions (2.5) and (2.6) gives;

$$T_1 + T_2 = \left[2 - \Delta t \frac{\partial^2}{\partial x^2} - \Delta t \frac{\partial^2}{\partial y^2}\right] T_{i,j}^{n+1} \quad (2.9)$$

Rearranging (2.9) gives

$$T_1 + T_2 = T_{i,j}^{n+1} + \left[1 - \Delta t \frac{\partial^2}{\partial x^2} - \Delta t \frac{\partial^2}{\partial y^2}\right] T_{i,j}^{n+1} \quad (2.10)$$

The term given in operator form at right hand side of (2.10) is already equal to $T_{i,j}^n$.

$$T_{i,j}^{n+1} = T_1 + T_2 - T_{i,j}^n \quad (2.11)$$

As shown with equations (2.7),(2.8) and (2.11) none of the tri-diagonal matrix solutions use the semi-solution of each other until all x sweep and y sweep solutions are obtained. Although such an approximate factorization has a higher additional term error than classical Crank-Nicolson approximate factorization, it is easier to program. This approach will be named as **simple AF** for the rest of the study.

Also the approximate factorization that is used by Ramos [15] is adapted for the ACIDI and became useful for unstructured grids. The form given below becomes Crank Nicolson scheme if $\theta = 0.5$;

$$\frac{T_{i,j}^{n+1} - T_{i,j}^n}{\Delta t} = \theta \frac{\partial^2 T_{i,j}^{n+1}}{\partial x^2} + (1 - \theta) \frac{\partial^2 T_{i,j}^n}{\partial x^2} + \theta \frac{\partial^2 T_{i,j}^{n+1}}{\partial y^2} + (1 - \theta) \frac{\partial^2 T_{i,j}^n}{\partial y^2} \quad (2.12)$$

(2.12) can be rearranged as shown below;

$$\frac{\Delta T}{\Delta t} - \theta \frac{\partial^2 \Delta T}{\partial x^2} - \theta \frac{\partial^2 \Delta T}{\partial y^2} = \frac{\partial^2 T_{i,j}^n}{\partial x^2} + \frac{\partial^2 T_{i,j}^n}{\partial y^2} \quad (2.13)$$

$$\Delta T = T_{i,j}^{n+1} - T_{i,j}^n.$$

If a regular kind of approximate factorization is used for (2.13);

$$\left[1 - \Delta t \theta \frac{\partial^2}{\partial x^2}\right] \Delta T^* = \frac{\partial^2 T_{i,j}^n}{\partial x^2} + \frac{\partial^2 T_{i,j}^n}{\partial y^2} \quad (2.14)$$

$$\left[1 - \Delta t \theta \frac{\partial^2}{\partial y^2}\right] \Delta T^{n+1} = \Delta T^* \quad (2.15)$$

The factorization given with (2.13), (2.14) and (2.15) is inappropriate for ACIDI applications. It can not be used with alternating cell directions since it is not possible to define meaningful sequence selection for the sweep solutions for unstructured cells. Thus the importance of sequence of sweep solutions should be removed. Using a similar approach of equations (2.7) and (2.8) yields;

$$\left[1 - \Delta t \theta \frac{\partial^2}{\partial x^2}\right] \Delta T_1 = \frac{\partial^2 T_{i,j}^n}{\partial x^2} + \frac{\partial^2 T_{i,j}^n}{\partial y^2} \quad (2.16)$$

$$\left[1 - \Delta t \theta \frac{\partial^2}{\partial y^2}\right] \Delta T_2 = \frac{\partial^2 T_{i,j}^n}{\partial x^2} + \frac{\partial^2 T_{i,j}^n}{\partial y^2} \quad (2.17)$$

Where;

$$\Delta T_1 = \left[1 - \Delta t \theta \frac{\partial^2}{\partial y^2}\right] \Delta T \quad (2.18)$$

$$\Delta T_2 = \left[1 - \Delta t \theta \frac{\partial^2}{\partial x^2}\right] \Delta T \quad (2.19)$$

If similar derivation approaches of equation (2.11) are performed, the solution for next time step value will be;

$$\Delta T_{i,j}^{n+1} = \Delta T_1 + \Delta T_2 - \frac{\partial^2 T_{i,j}^n}{\partial x^2} - \frac{\partial^2 T_{i,j}^n}{\partial y^2} \quad (2.20)$$

If the expression $\left(-\frac{\partial^2 T_{i,j}^n}{\partial x^2} - \frac{\partial^2 T_{i,j}^n}{\partial y^2}\right)$ that takes place in the right hand side of the above equation is inspected, it will be seen that it is equal to the minus one times the right hand side of the equation (2.13) which is a known value of the general equation (2.13).

$$\frac{\partial^2 T_{i,j}^n}{\partial x^2} + \frac{\partial^2 T_{i,j}^n}{\partial y^2} = \text{RHS} \quad (2.21)$$

Then;

$$\Delta T_{i,j}^{n+1} = \Delta T_1 + \Delta T_2 - \text{RHS} \quad (2.22)$$

$$T_{i,j}^{n+1} = T_{i,j}^n + \Delta T_{i,j}^{n+1} \quad (2.23)$$

The additional term of factorization is;

$$\theta^2 \Delta t^2 \frac{\partial}{\partial x^2} \left[\frac{\partial}{\partial y^2} (T_{i,j}^{n+1} - T_{i,j}^n) \right] \quad (2.24)$$

And it is equal to the additional term of standard Crank-Nicolson approximate factorization where $\theta = 0.5$.

This factorization method will be named as **Crank Nicolson like AF** for the comparison of the results.

2.2 Approximate Factorization with ACDI

The method will be explained using cell centered finite volume approach. Using finite volume approach is more suitable since the success of the approach will be compared with previous ACDI method which also uses cell centered finite volumes.

$$\frac{\partial T}{\partial t} - \nabla^2 T = 0 \quad (2.25)$$

Integrating the equation over the control volume cell that shown in Figure 2.1 gives:

$$\int_A \frac{\partial T}{\partial t} dA - \int_A \nabla^2 T dA = 0 \quad (2.26)$$

Applying the Green's Theorem for the second term of left hand side and writing the first term in discrete form:

$$A \frac{\Delta \bar{T}}{\Delta t} - \oint_c \frac{\partial T}{\partial n} ds = 0 \quad (2.27)$$

Where \bar{T} is a mean value of dependent variable that placed at cell center such that:

$$\bar{T} = \frac{1}{A} \int_A T dA$$

If it is assumed that the fluxes at each cell edge is constant through the edge, where the cell has N edges like shown in Figure 2.1, (2.27) can be written in the form as shown:

$$A \frac{\Delta \bar{T}}{\Delta t} - \sum_{i=1}^N \left(\frac{\partial T}{\partial n} \right)_i \Delta s_i = 0 \quad (2.28)$$

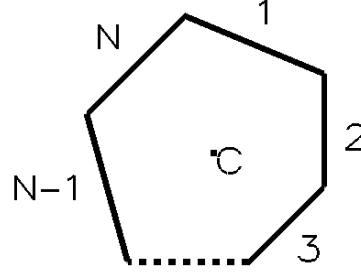


Figure 2. 1 Control Volume with N edges

If (2.28) is written in the form of:

$$A \frac{\Delta \bar{T}}{\Delta t} - \left[\left(\frac{\partial T}{\partial n} \right)_1 \Delta s_1 + \left(\frac{\partial T}{\partial n} \right)_2 \Delta s_2 + \dots + \left(\frac{\partial T}{\partial n} \right)_N \Delta s_N \right] = 0 \quad (2.29)$$

If the flux terms are grouped such that opposite edge fluxes take place in the same group where T^{n+1} is the next time step value of dependent variable at each cell center that is to be calculated, the resulting expression will be;

$$A \frac{T^{n+1} - T^n}{\Delta t} - \left\{ \left[\left(\frac{\partial T}{\partial n} \right)_1 \Delta s_1 + \left(\frac{\partial T}{\partial n} \right)_{1+N/2} \Delta s_{1+N/2} \right] + \left[\left(\frac{\partial T}{\partial n} \right)_2 \Delta s_2 + \left(\frac{\partial T}{\partial n} \right)_{2+N/2} \Delta s_{2+N/2} \right] + \dots + \left[\left(\frac{\partial T}{\partial n} \right)_{N/2} \Delta s_{N/2} + \left(\frac{\partial T}{\partial n} \right)_N \Delta s_N \right] \right\} = 0 \quad (2.30)$$

It is possible to rearrange (2.30) such that T^n takes place at the right hand side of the equation since it is a known value at the n^{th} time step:

$$T^{n+1} - \left\{ \left[\frac{\Delta t}{A} \left(\frac{\partial T}{\partial n} \right)_1 \Delta s_1 + \frac{\Delta t}{A} \left(\frac{\partial T}{\partial n} \right)_{1+N/2} \Delta s_{1+N/2} \right] + \left[\frac{\Delta t}{A} \left(\frac{\partial T}{\partial n} \right)_2 \Delta s_2 + \frac{\Delta t}{A} \left(\frac{\partial T}{\partial n} \right)_{2+N/2} \Delta s_{2+N/2} \right] + \dots + \left[\frac{\Delta t}{A} \left(\frac{\partial T}{\partial n} \right)_{N/2} \Delta s_{N/2} + \frac{\Delta t}{A} \left(\frac{\partial T}{\partial n} \right)_N \Delta s_N \right] \right\} = T^n \quad (2.31)$$

The fluxes are going to be calculated implicitly, thus it is possible to write the LHS in the operator form. For the sake of simplicity T^{n+1} will be represented as T and T^n will be represented as RHS beginning from this point.

$$\begin{aligned}
& \left\{ 1 - \left[\frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n} \right)_1 + \frac{\Delta t}{A} \Delta s_{1+N/2} \left(\frac{\partial}{\partial n} \right)_{1+N/2} \right] \right. \\
& \quad + \left[\frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n} \right)_2 + \frac{\Delta t}{A} \Delta s_{1+N/2} \left(\frac{\partial}{\partial n} \right)_{2+N/2} \right] \\
& \quad \left. + \dots \dots \left[\frac{\Delta t}{A} \Delta s_{N/2} \left(\frac{\partial}{\partial n} \right)_{N/2} + \frac{\Delta t}{A} \Delta s_N \left(\frac{\partial}{\partial n} \right)_N \right] \right\} T = RHS
\end{aligned} \tag{2.32}$$

The form given with equation (2.32) is for the application of **simple AF**. The derivations for usage of **simple AF** and **Crank-Nicolson like AF** are very similar to each other, thus the derivation of usage of **simple AF** is given only.

Equation given above offers a form that is suitable for approximate factorization for any kind of polygonal cells that have even number of edges, also it is possible to use the same factorization for cells that have odd number of edges via developing simple strategies to convert these cell edges to even numbered edges. Possible strategies for odd numbered edges will be discussed in oncoming parts of the study.

Applying simple AF to the form given as (2.3) gives:

$$\begin{aligned}
& \left(1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n} \right)_1 - \frac{\Delta t}{A} \Delta s_{1+\frac{N}{2}} \left(\frac{\partial}{\partial n} \right)_{1+\frac{N}{2}} \right) \\
& \left(1 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n} \right)_2 - \frac{\Delta t}{A} \Delta s_{2+\frac{N}{2}} \left(\frac{\partial}{\partial n} \right)_{2+\frac{N}{2}} \right) \dots \\
& \left(1 - \frac{\Delta t}{A} \Delta s_{N/2} \left(\frac{\partial}{\partial n} \right)_{N/2} - \frac{\Delta t}{A} \Delta s_N \left(\frac{\partial}{\partial n} \right)_N \right) T = RHS
\end{aligned} \tag{2.33}$$

Grouping fluxes written for opposite edges in the factorized equation lets the scheme to be applied using alternating cell directions that shown on even number edged polygonal cells in Figure 2.2.

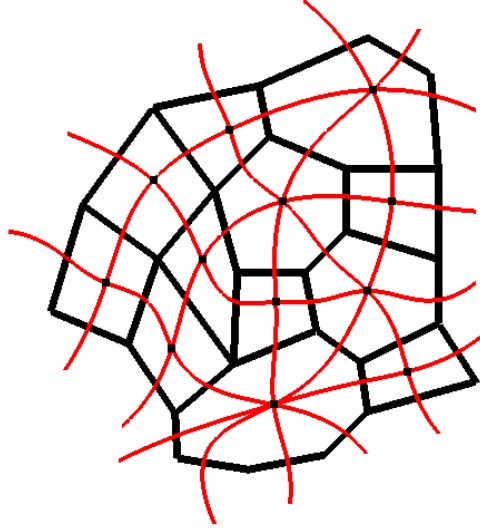


Figure 2. 2 Cell directions drawn travelling opposing edges of cells

It is possible to write the expression (2.33) separately for each factorized term via assigning semi-solutions that will be declared as $T_1, T_2, \dots, T_{N/2}$ such that:

$$\begin{aligned}
 \left(1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n} \right)_1 - \frac{\Delta t}{A} \Delta s_{1+\frac{N}{2}} \left(\frac{\partial}{\partial n} \right)_{1+\frac{N}{2}} \right) T_1 &= RHS \\
 \left(1 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n} \right)_2 - \frac{\Delta t}{A} \Delta s_{2+\frac{N}{2}} \left(\frac{\partial}{\partial n} \right)_{2+\frac{N}{2}} \right) T_2 &= RHS \\
 &\vdots \\
 \left(1 - \frac{\Delta t}{A} \Delta s_{N/2} \left(\frac{\partial}{\partial n} \right)_{N/2} - \frac{\Delta t}{A} \Delta s_N \left(\frac{\partial}{\partial n} \right)_N \right) T_{N/2} &= RHS
 \end{aligned} \tag{2.34}$$

Writing the operators as shown above and using semi-solutions is performed to obtain tri-diagonal matrices for the equation sets that will be written over the solution bands that shown in Figure 2.2.

The value of T has to be calculated after the semi-solutions are obtained. If the operators of (2.34) are named as X_i , $i = 1, 2, 3 \dots N/2$

$$\begin{aligned}
 X_2 X_3 \dots X_{N/2} T &= T_1 \\
 X_1 X_3 \dots X_{N/2} T &= T_2 \\
 &\vdots \\
 X_1 \dots X_j \dots X_{N/2} T &= T_i \quad i \neq j \\
 &\vdots \\
 X_1 \dots X_j \dots X_{N/2-1} T &= T_{N/2}
 \end{aligned} \tag{2.35}$$

If it is assumed that the expression $X_1 \dots X_j \dots X_{N/2} T = T_i$, $i \neq j$ is equal to the form before the factorization:

$$\begin{aligned}
& X_1 \dots X_j \dots X_{N/2} T \cong \\
& \left\{ 1 - \left\{ \left[\frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n} \right)_1 + \frac{\Delta t}{A} \Delta s_{1+N/2} \left(\frac{\partial}{\partial n} \right)_{1+N/2} \right] \dots \dots \right. \right. \\
& \quad + \left. \left[\frac{\Delta t}{A} \Delta s_i \left(\frac{\partial}{\partial n} \right)_i + \frac{\Delta t}{A} \Delta s_{i+N/2} \left(\frac{\partial}{\partial n} \right)_{i+N/2} \right] \right. \\
& \quad \left. \left. \dots \dots \left[\frac{\Delta t \Delta s_{N/2}}{A} \left(\frac{\partial}{\partial n} \right)_{N/2} + \frac{\Delta t}{A} \Delta s_N \left(\frac{\partial}{\partial n} \right)_N \right] \right\} \right\} T
\end{aligned} \tag{2.36}$$

If the expressions of (2.35) are summed using the assumption given as (2.36) the summation gives:

$$\frac{N}{2} T - \left(\frac{N}{2} - 1 \right) \sum_{i=1}^N \frac{\Delta t}{A} \Delta s_i \left(\frac{\partial T}{\partial n} \right)_i = \sum_{i=1}^N T_i \tag{2.37}$$

Rearranging (2.37) yields:

$$T + \left(\frac{N}{2} - 1 \right) \underbrace{\left(T - \sum_{i=1}^N \frac{\Delta t}{A} \Delta s_i \left(\frac{\partial T}{\partial n} \right)_i \right)}_{\text{RHS}} = \sum_{i=1}^N T_i \tag{2.38}$$

Then the value of T can be calculated using the simple expression below:

$$T = \sum_{i=1}^N T_i - \left(\frac{N}{2} - 1 \right) \text{RHS} \tag{2.39}$$

2.2.1 Difference of the Approach from Fractional Time Stepping Methods

The equation set that is written for a sample hexagonal element of the general form is given below;

$$\left(1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n} \right)_1 - \frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n} \right)_4 \right) T_1 = \text{RHS} \tag{a}$$

$$\left(1 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n} \right)_2 - \frac{\Delta t}{A} \Delta s_5 \left(\frac{\partial}{\partial n} \right)_5 \right) T_2 = \text{RHS} \tag{b} \tag{2.40}$$

$$\left(1 - \frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n} \right)_3 - \frac{\Delta t}{A} \Delta s_6 \left(\frac{\partial}{\partial n} \right)_6 \right) T_3 = \text{RHS} \tag{c}$$

If the equation set is inspected well with the factorized equation which is written together with the equation set (2.40-a), (2.40-b) and (2.40-c) it will be seen that;

$$\begin{aligned}
& \left(1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1 - \frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n}\right)_4\right) \\
& \left(1 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n}\right)_2 - \frac{\Delta t}{A} \Delta s_5 \left(\frac{\partial}{\partial n}\right)_5\right) \\
& \left(1 - \frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3 - \frac{\Delta t}{A} \Delta s_6 \left(\frac{\partial}{\partial n}\right)_6\right) T \\
& = \left(1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1 - \frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n}\right)_4\right) T_1 \\
& = \left(1 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n}\right)_2 - \frac{\Delta t}{A} \Delta s_5 \left(\frac{\partial}{\partial n}\right)_5\right) T_2 \\
& = \left(1 - \frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3 - \frac{\Delta t}{A} \Delta s_6 \left(\frac{\partial}{\partial n}\right)_6\right) T_3
\end{aligned} \tag{2.41}$$

The equation (2.40) shows that none of the equations (2.40-a), (2.40-b) and (2.40-c) are the approximations of the left hand side of the equation (2.41), but they are exactly equal to it by definition.

The tri-diagonal matrix solutions of (2.40-a), (2.40-b) and (2.40-c) give the values of the definitions given below;

$$\begin{aligned}
& \left(1 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n}\right)_2 - \frac{\Delta t}{A} \Delta s_5 \left(\frac{\partial}{\partial n}\right)_5\right) \left(1 - \frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3 - \frac{\Delta t}{A} \Delta s_6 \left(\frac{\partial}{\partial n}\right)_6\right) T = T_1 \\
& \left(1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1 - \frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n}\right)_4\right) \left(1 - \frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3 - \frac{\Delta t}{A} \Delta s_6 \left(\frac{\partial}{\partial n}\right)_6\right) T = T_2 \\
& \left(1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1 - \frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n}\right)_4\right) \left(1 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n}\right)_2 - \frac{\Delta t}{A} \Delta s_5 \left(\frac{\partial}{\partial n}\right)_5\right) T = T_3
\end{aligned}$$

The algebraic operations (2.35)-(2.39) are simply performed to obtain the updated cell center values using the previously obtained values of the definitions given above as T_1 , T_2 and T_3 .

Node based finite difference Crank-Nicolson version of the fractional time stepping method is adapted from Hoffman [13] and given below for the unsteady diffusion equation;

$$\frac{T_{i,j}^{n+1/2} - T_{i,j}^n}{\Delta t/2} = \frac{1}{2} \left[\frac{\partial^2}{\partial x^2} \right] T_{i,j}^{n+1/2} + \frac{1}{2} \left[\frac{\partial^2}{\partial x^2} \right] T_{i,j}^n \tag{2.42}$$

$$\frac{T_{i,j}^{n+1} - T_{i,j}^{n+1/2}}{\Delta t/2} = \frac{1}{2} \left[\frac{\partial^2}{\partial y^2} \right] T_{i,j}^{n+1} + \frac{1}{2} \left[\frac{\partial^2}{\partial y^2} \right] T_{i,j}^{n+1/2}$$

As seen from the equation set (2.42) the first of the equations does not include any of the y derivative terms and the second of the equations does not include any of the x derivative terms. The operators are split to obtain half time step solution in one direction firstly and the updated value in the other direction secondly. The bond between the equations of (2.42) is only provided with the half time step solution, but the equations of (2.40) are exactly equal to each other and the whole factorized equation itself. Each of them is set to provide the cell center values of different parts of the whole factorized equation (2.40).

2.2.2 Flux Calculation Using Inverse Distance Weighting

The notation that will be used for the flux calculation is given in Figure 2. 3. The subscripts M and N are going to be used for the coordinates of the center of the cell that the fluxes are being calculated for, and center of the neighbor cells relatively. The subscripts a and b are going to be used for the coordinates of the nodes that are on the cell edge that the flux calculation is being performed.

The points that shown with the coordinates (x_0, y_0) and (x_1, y_1) are the locations that the dependent variables will be moved using inverse distance weighting [16]. These points are perpendicularly $\Delta s/2K$ far away from the edge center that the flux is being calculated. If it is assumed that the flux is constant over the cell edge, the derivative to be calculated can be approximated as;

$$\frac{\partial T}{\partial n} \cong \frac{T_{(x_1, y_1)} - T_{(x_0, y_0)}}{\Delta s/K} \quad (2.43)$$

The coordinates (x_0, y_0) and (x_1, y_1) have to be calculated for the inverse distance weighted interpolation.

$$\begin{aligned} x_0 &= \frac{x_b + x_a}{2} - \frac{dx}{dn} \frac{\Delta s}{2K} \\ y_0 &= \frac{y_b + y_a}{2} - \frac{dy}{dn} \frac{\Delta s}{2K} \\ x_1 &= \frac{x_b + x_a}{2} + \frac{dx}{dn} \frac{\Delta s}{2K} \\ y_1 &= \frac{y_b + y_a}{2} + \frac{dy}{dn} \frac{\Delta s}{K} \end{aligned} \quad (2.44)$$

The derivative values on the edge can be approximated as;

$$\frac{dx}{dn} = \frac{dy}{ds} = \frac{y_a - y_b}{\Delta s}$$

$$\frac{dy}{dn} = -\frac{dx}{ds} = \frac{x_b - x_a}{\Delta s} \quad (2.45)$$

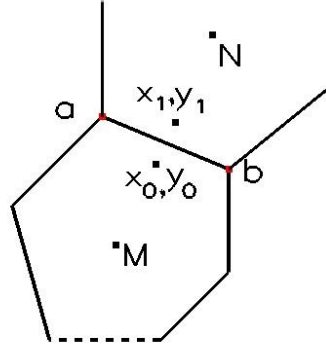


Figure 2. 3 Notation used for low order flux calculation on a sample polygonal cell and neighbor cell

If D_{M_0} , D_{M_1} are distances from the cell center to (x_0, y_0) , (x_1, y_1) relatively and D_{N_0} , D_{N_1} are distances from the neighbour cell center to (x_0, y_0) and (x_1, y_1) then extrapolated values will be:

$$T_{(x_0, y_0)} = \frac{D_{N_0}}{(D_{M_0} + D_{N_0})} T_C + \frac{D_{M_0}}{(D_{M_0} + D_{N_0})} T_N$$

$$T_{(x_1, y_1)} = \frac{D_{N_1}}{(D_{M_1} + D_{N_1})} T_C + \frac{D_{M_1}}{(D_{M_1} + D_{N_1})} T_N \quad (2.46)$$

The flux calculation approach shown here is a simple and easy to use method. Also it is easily applicable for three dimensional applications but its order of accuracy is not high enough for most of the fluid flow calculations.

2.2.3 Flux Calculation Using Trapezoidal Rule

The notation that will be used for the flux calculation is given in Figure 2.4. The subscript M will be used for the cell that the flux terms are being calculated. P and N are the centers of the neighbor cells which are on the solution band of interest. The values of these points are going to be included as unknown values in the calculation. L and R are the centers of the neighbor cells which take place on the other solution band that passes through the cell M.

The node values at points a, b, c and d will be used as known values, and thus the values at nodes have to be updated at each time step via interpolating the cell center values to these nodes.

$$\frac{\partial T}{\partial n} = \frac{\partial T}{\partial x} \frac{\partial x}{\partial n} + \frac{\partial T}{\partial y} \frac{\partial y}{\partial n} \quad (2.47)$$

The derivatives $\partial x/\partial n$ and $\partial y/\partial n$ are already approximated using the equations (2.45). The average values of fluxes are calculated using integrations with the help of virtual faces shown with dashed lines at Figure 2.4. For example derivative on the edge ab is calculated using the face MaPb. The average values of derivatives at edge ab:

$$\begin{aligned} \left(\frac{\partial \bar{T}}{\partial x}\right)_{ab} &= \frac{1}{A_{MaPb}} \left[\int_{MaPb} \frac{\partial T}{\partial x} dA = \int_{MaPb} T \frac{\partial x}{\partial n} ds = \int_{MaPb} T \frac{\partial y}{\partial s} ds = \int_{MaPb} T \cdot dy \right] \\ \left(\frac{\partial \bar{T}}{\partial y}\right)_{ab} &= \frac{1}{A_{MaPb}} \left[\int_{MaPb} \frac{\partial T}{\partial y} d = \int_{MaPb} T \frac{\partial y}{\partial n} ds = - \int_{MaPb} T \frac{\partial x}{\partial s} ds = - \int_{MaPb} T \cdot dx \right] \end{aligned} \quad (2.48)$$

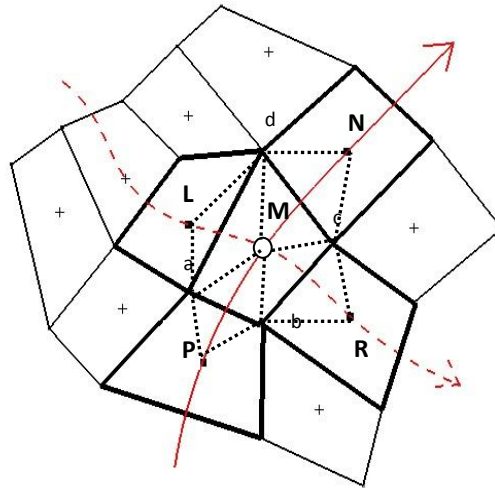


Figure 2.4 Notation used for trapezoidal rule flux calculation on a sample quadrilateral cell and neighbor cells

The integrations can be calculated numerically using trapezoidal rule:

$$\begin{aligned} \left(\frac{\partial \bar{T}}{\partial x}\right)_{ab} &= \frac{1}{A_{MaPb}} \left[\frac{(T_M + T_a)}{2} \Delta y_{Ma} + \frac{(T_a + T_p)}{2} \Delta y_{aP} + \frac{(T_p + T_b)}{2} \Delta y_{Pb} \right. \\ &\quad \left. + \frac{(T_b + T_M)}{2} \Delta y_{bM} \right] \end{aligned}$$

$$\left(\frac{\partial \bar{T}}{\partial y}\right)_{ab} = \frac{-1}{A_{MaPb}} \left[\frac{(T_M + T_a)}{2} \Delta x_{Ma} + \frac{(T_a + T_p)}{2} \Delta x_{aP} + \frac{(T_p + T_b)}{2} \Delta x_{Pb} + \frac{(T_b + T_M)}{2} \Delta y_{bM} \right] \quad (2.49)$$

If the expressions of (2.49) inserted into (2.47) and the $\partial T/\partial n$ calculations are used with equations given as (2.34) the equation sets will end up with tri-diagonal matrices for all alternating cell directions. The node values will take place explicitly in equation sets.

2.2.4 Von Neumann Stability Analysis

Von Neumann stability analysis is performed assuming Cartesian cells. Sample Cartesian cell and its neighbors are shown in Figure 2.5 with the notation used for the analysis. Analysis is carried on using inverse distance weighting flux calculation with simple AF.

Rewriting the equations in (2.34) quadrilateral cells:

$$\begin{aligned} T_1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial T_1}{\partial n}\right)_1 - \frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial T_1}{\partial n}\right)_3 &= RHS \\ T_2 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial T_2}{\partial n}\right)_2 - \frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial T_2}{\partial n}\right)_4 &= RHS \end{aligned} \quad (2.50)$$

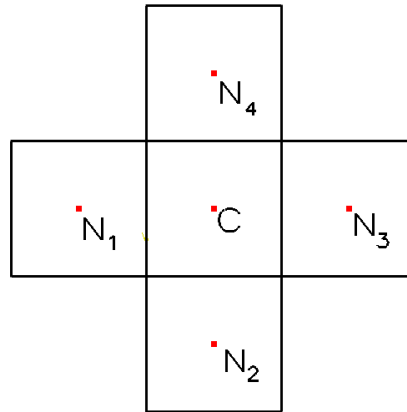


Figure 2.5 Example Cartesian cell and its neighbors

The low order flux terms can be inserted in the form of:

$$\left(\frac{\partial T_1}{\partial n}\right)_1 = w_{1N_1} T_{1N_1} - w_{1C} T_{1C} \quad (2.51)$$

w_{1N_1} and w_{1C} are the weights that used to calculate the flux term from the cell with cell center N_1 that shown in Figure 2.5. The weights for Cartesian cells are all equal to $1/\Delta s$. Then the flux terms can be inserted into equations (2.50) in the form as below:

$$\begin{aligned}
\left(\frac{\partial T_1}{\partial n}\right)_1 &= \frac{1}{\Delta S} (T_{1N_1} - T_{1C}) \\
\left(\frac{\partial T_1}{\partial n}\right)_3 &= \frac{1}{\Delta S} (T_{1N_3} - T_{1C}) \\
\left(\frac{\partial T_2}{\partial n}\right)_2 &= \frac{1}{\Delta S} (T_{2N_2} - T_{2C}) \\
\left(\frac{\partial T_2}{\partial n}\right)_4 &= \frac{1}{\Delta S} (T_{2N_4} - T_{2C})
\end{aligned} \tag{2.52}$$

Inserting the flux terms into equations given as (2.50) and using spatial and temporal indices gives:

$$\begin{aligned}
\left(1 + 2\frac{\Delta t}{A}\right) T_{1i,j}^{n+1} - \frac{\Delta t}{A} T_{1i-1,j}^{n+1} - \frac{\Delta t}{A} T_{1i+1,j}^{n+1} &= T_{i,j}^n \\
\left(1 + 2\frac{\Delta t}{A}\right) T_{2i,j}^{n+1} - \frac{\Delta t}{A} T_{2i,j-1}^{n+1} - \frac{\Delta t}{A} T_{2i,j+1}^{n+1} &= T_{i,j}^n
\end{aligned} \tag{2.53}$$

Inserting $T_{i,j}^n = \xi^n e^{ikx} e^{iky}$ into the first expression given with (2.53) to find out the growth rate of error at each time iteration gives;

$$\begin{aligned}
\left(1 + 2\frac{\Delta t}{A}\right) \xi^{n+1} e^{ikx} e^{iky} - \frac{\Delta t}{A} \xi^{n+1} e^{ik(x-\Delta x)} e^{iky} - \frac{\Delta t}{A} \xi^{n+1} e^{ik(x+\Delta x)} e^{iky} \\
= \xi^n e^{ikx} e^{iky}
\end{aligned} \tag{2.54}$$

Dividing both RHS and LHS of (2.54) by $\xi^n e^{ikx} e^{iky}$ yields:

$$\left(1 + 2\frac{\Delta t}{A}\right) \xi - \frac{\Delta t}{A} \xi e^{-ik\Delta x} - \frac{\Delta t}{A} \xi e^{ik\Delta x} = 1 \tag{2.55}$$

Rearranging (2.55) gives:

$$\xi \left[\left(1 + 2 \frac{\Delta t}{A}\right) - \frac{\Delta t}{A} (e^{-ik\Delta x} + e^{ik\Delta x}) \right] = 1 \quad (2.56)$$

Then the growth factor will be:

$$\xi = \frac{1}{\left(1 + 2 \frac{\Delta t}{A}\right) - 2 \frac{\Delta t}{A} (\cos k\Delta x)} \quad (2.57)$$

Similarly the growth factor for the second expression of (2.53) will be:

$$\zeta = \frac{1}{\left(1 + 2 \frac{\Delta t}{A}\right) - 2 \frac{\Delta t}{A} (\cos k\Delta y)} \quad (2.58)$$

The stability criterion is:

$$|\xi| \leq 1$$

$$|\zeta| \leq 1$$

Since;

$$-1 \leq \cos k\Delta x \leq 1 \quad -1 \leq \cos k\Delta y \leq 1 \quad (2.59)$$

Both of the growth factors satisfy the stability criterion, thus the scheme is unconditionally stable.

2.3 Odd Number of Edges

The ACDI scheme is appropriate for elements that have even number of edges as stated previously. The solution bands called cell directions are generated via travelling the opposing edges of the cells as shown in Figure 2. 2 and such generation is not possible for cells that have odd number of edges. Two different strategies that used to overcome this problem will be explained in this part.

One of these methods is splitting one of the edges of the cell into two and converting it to a cell that has even number of edges. This approach is only possible if the split edge takes place on the boundary of the domain. If the cell is inside the domain and one of the edges is split into two, the neighbor cell which has even number of edges will be converted to a cell that has odd number of edges which is not desired.

As seen from Figure 2.6 splitting the edge at the boundary causes having very sharp turns of the cell directions which is an undesirable situation. Although the new element has a poor quality and cell

directions of the element has very sharp turns, strategy is still useful and it is used for some of the solutions that are presented in the oncoming parts of the study.

The cell that has odd number of edges may take place inside the domain. In this situation one of the nodes of the cell is considered as a zero length edge and cell direction ends inside the domain at that edge. The edge will have two nodes that have the same coordinates. No boundary condition application is required since the flux will be zero at this edge.

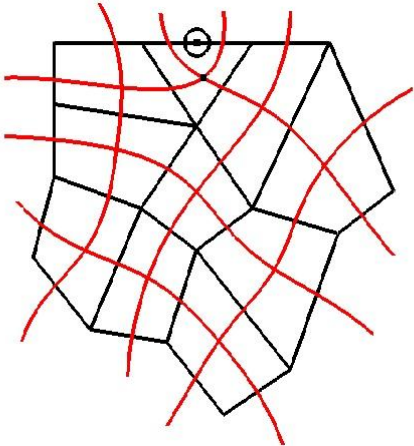


Figure 2.6 Split edge of triangular element at the boundary and cell directions

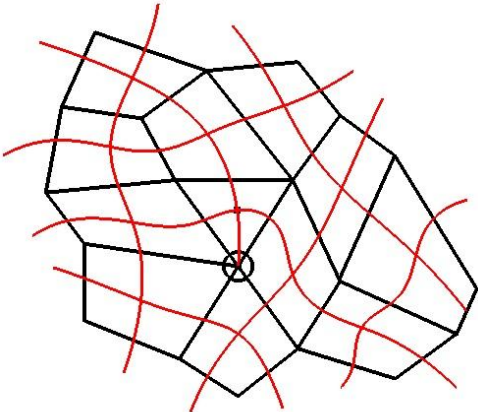


Figure 2. 7 Zero length edge of triangular element inside the domain and cell directions

An example cell that is converted to a quadrilateral element with a zero length edge and the cell directions are shown in Figure 2. 7.

Existence of a zero length edge breaks the cell direction as shown in Figure 2.7 and having more triangular elements inside the domain means being less implicit.

Also the selection of the location of zero length edge is random, thus the cell directions generated for the grid is not unique anymore. One of the aims of this study is to end up with a mathematically well defined version of previous ACDI studies. There exists no randomness of solution in case of having even number of edges all over the domain. The sequence of solution bands does not change the numerical results. But having odd number of edges inside the domain damages this property.

2.4 K-exact Least Squares Reconstruction

If the factorized equation for advection-diffusion equation that is shown in Appendix C is inspected well, it will be seen that the dependent variable should also be calculated at edge centers as well as the nodes of the cells. The node values required for trapezoidal rule flux calculation have been calculated using inverse distance weighting for diffusion equation comparisons but reconstruction at nodes and edge centers with inverse distance weighting will result in fluctuations for the solution of advection diffusion equation [22] [23], also low order reconstruction is not appropriate for the full potential solver that will be explained in the oncoming parts of the study. Neel [24] states that K-exact least squares method is one of the most suitable reconstruction schemes for full potential solvers for non-oscillatory results with a linear distribution. Thus K-exact least squares method is embedded to the AF ACDI code for non-oscillatory results for both advection-diffusion equation solution and transonic full potential solution.

An appropriate stencil is selected for the reconstruction of the variable on the desired point. The variables and coordinates of the points of the stencil are used to generate a k^{th} order polynomial for the reconstruction. A polynomial of k^{th} order provides $k+1^{\text{st}}$ order of accuracy.

The explanation of the least squares reconstruction application is adapted from Neel [24] for reconstruction of variables at nodes using the adjacent cell center values.

If the first order polynomial that will be used for the reconstruction in a 2-D domain is in the form;

$$P = K_0 + K_1x + K_2y \tag{2. 60}$$

Then there will be linear equations equal to the number of the cells in the stencil that will be used for the reconstruction. This situation results with an over-constraint set of equations. All the adjacent cells are used for the calculation of the variable on the node as shown Figure 2.8.

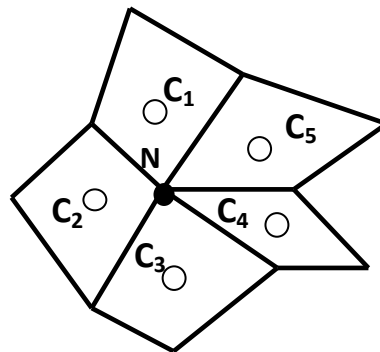


Figure 2.8 Stencil used for a reconstruction at node

Weighted reconstruction is used for such situations. Weights are generally set via using the inverse distance weighting method. The sum of the square errors at each cell center of the reconstruction will be;

$$error = \sum_{i=1}^N w_i (K_0 + K_1x + K_2y - c_i)^2 \quad (2.61)$$

Setting the derivative of the error with respect to the constant terms equal to 0 gives 3 linear equations with 3 unknowns;

$$\begin{aligned} \frac{\partial e}{\partial K_0} &= 2 \sum_{i=1}^N w_i (K_0 + K_1x + K_2y - c_i) = 0 \\ \frac{\partial e}{\partial K_1} &= 2 \sum_{i=1}^N w_i (K_0 + K_1x + K_2y - c_i)x_i = 0 \\ \frac{\partial e}{\partial K_2} &= 2 \sum_{i=1}^N w_i (K_0 + K_1x + K_2y - c_i)y_i = 0 \end{aligned} \quad (2.62)$$

2.5 Parallelization Strategy

If the finite difference formulations given as (2.16), (2.17) or the finite volume formulations given as (2.34) are observed, it will be seen that none of the matrices of sweep directions or matrices of cell directions require each other's solutions. It is possible to solve the tri-diagonal matrices independently. They do not have to wait another matrix solution.

Two different options are considered for the parallelization. One of them is MPI (Message Passing Interface) and the other one is Open MP (Open Multi Processing). MPI is distributed memory based [17] and OpenMP is a shared memory based library [18]. The difference between parallelization principles is shown in Figure 2.9.

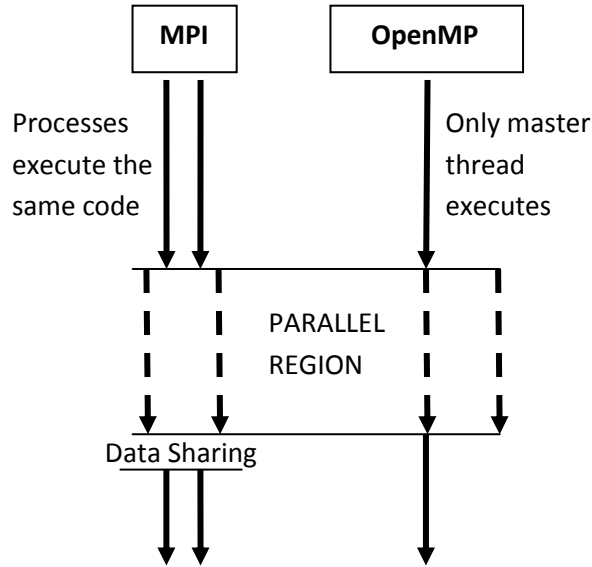


Figure 2.9 Difference between usage of shared and distributed memory approaches for parallelization

If distributed memory is used for the approach a 'broadcast' or a 'send-receive' operation will be a must during the data sharing part. Although data sharing part results with an overhead for all parallelization applications with distributed memory; this is not the biggest problem of ACIDI with MPI. If domain decomposition was being used, data sharing would take place only over the edges of the decomposed domains [19], but the matrices of sweep directions or cell directions return with semi-solutions instead of an updated value of the new solution. Thus all these semi-solutions have to be sent to the correct cell information and the overhead of the parallelization will be enormous.

But if shared memory is used and the matrices of the cell directions are calculated by different threads, all the threads will calculate independent semi-solutions of different cells. They record nothing new to each other's variable, thus no confusion occurs.

The dashed lines of OpenMP part in Figure 2.9 can be considered as two equi-sized cell directions. When they both end up, no data sharing procedure is required and simply the updated results are calculated by the Master thread using the equation (2.39);

$$T = \sum_{i=1}^N T_i - \left(\frac{N}{2} - 1\right) RHS$$

In fact it is not a must to update the values using only master thread. The part of the code that updates the results can also be parallelized with almost no effort. All the semi-solutions or RHS parts (which are required for the calculation of the new values at cell centers) of the updating equation can be taken into account independently since they all belong to a unique cell center. No data sharing between the cells are required and all the equations that are in the form of (2.39) are written independently for each

cell center. Then each of them can be calculated by a random thread. Then the approach will be parallel-by-line during the tri-diagonal matrix solutions and parallel-by-point while updating the cell center values [20].

If an implicit scheme is being used together with the domain decomposition approach, the implicitness of the solution will be harmed by the boundaries of the decomposed domains. Besides different decompositions with same number of CPU usage or usage of different numbers of decomposed domains might result with non-unique solution of the implicit scheme. Using a parallelization that offered in this part of the study gives exactly same results for any cases and the constructed implicitness is protected.

CHAPTER 3

RESULTS AND DISCUSSION

3.1 Validation of the Numerical Approach Using Unsteady Diffusion Equation

3.1.1 Problem Specifications for Unsteady Diffusion Equation Solution Comparisons

Firstly unsteady heat conduction problem on a rectangular plate is solved for the comparisons. The infinite wall boundary condition is used at horizontal boundaries and constant temperature boundary condition is applied at vertical walls. Representation of this 1-D problem is given in Figure 3.1.

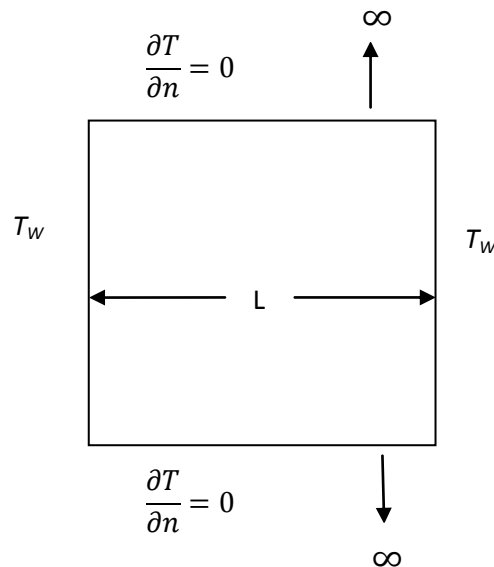


Figure 3.1 Heat conduction problem on a rectangular plate

The unsteady diffusion equation is as given below;

$$\frac{1}{\alpha} \frac{\partial T}{\partial t} - \nabla^2 T = 0 \quad (3.1)$$

In the equation given above α is the thermal diffusivity of the material. Dimensionless time for the unsteady diffusion equation is;

$$\tau = \frac{\alpha t}{k} \quad (3.2)$$

Also dimensionless distance is;

$$X = \frac{x}{L} \quad (3.3)$$

Non-dimensionalization of the temperature is performed as given below;

$$\theta(x, y, t) = \frac{T(x, y, t) - T_\infty}{T_i - T_\infty} \quad (3.4)$$

Then the non-dimensional form of the equation will be;

$$\frac{\partial^2 \theta}{\partial X^2} + \frac{\partial^2 \theta}{\partial Y^2} = \frac{\partial \theta}{\partial \tau} \quad (3.5)$$

The analytical solution of the problem is;

$$\begin{aligned} \theta(x, y, t) &= \sum_{n=1}^{\infty} A_n \exp(-\lambda_n \tau) \cos(\lambda_n X) \\ A_n &= \frac{4 \sin \lambda_n}{2\lambda_n + \sin 2\lambda_n} \\ \lambda_n \tan \lambda_n &= Bi \end{aligned} \quad (3.6)$$

Bi is the non-dimensional heat transfer coefficient Biot number.

$$Bi = \frac{hL}{k} \quad (3.7)$$

And it is taken as infinity for the numerical comparisons given.

Non-dimensional temperature contours at $\tau = 0.1, 0.2$ and 0.3 obtained using analytical formulation are given in Figure 3.2.

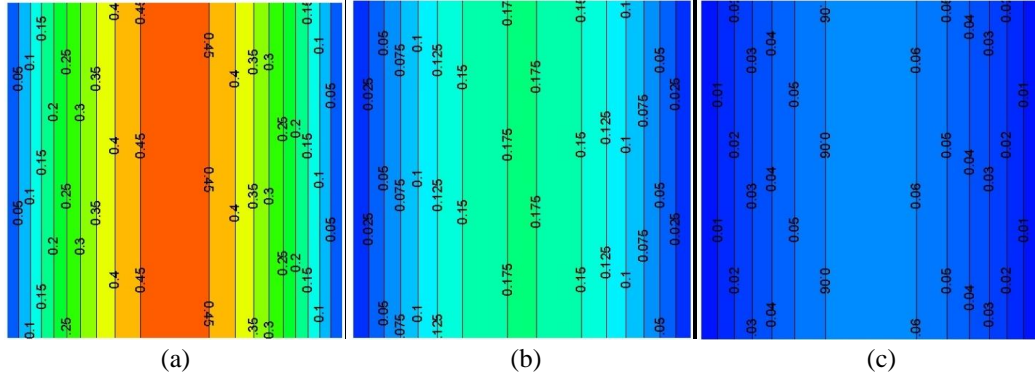


Figure 3.2 Non-dimensional temperature θ contours at (a) $\tau = 0.1$ (b) $\tau = 0.2$ (c) $\tau = 0.3$ using analytical formulation

3.1.2 Validation of the Proposed Approximate Factorization

The problem explained in part 3.1.1 is solved using classical Crank Nicolson Approximate Factorization, Crank-Nicolson like AF and simple AF for 20x20 and 40x40 structured grids.

Simple AF is the approximate factorization approach that is defined in Section 2.1 and the formulations are given as (2.3), (2.7) and (2.8). **Crank Nicolson like AF** is the approximate factorization approach that is defined in Section 2.1 and the formulations are given as (2.13), (2.16) and (2.17). These two methods are combined with ACIDI method in oncoming sections, thus their accuracies will be compared with the classical approximate factorization method that is defined in section 1.6 and the formulations can be seen in equations (1.12) and (1.14). The classical method is named as **Crank -Nicolson AF** for the comparisons of this section

Since it is not appropriate to use the classical approximate factorization methods with unstructured grids, only structured grids are used for this comparison.

The analytical solution is used to calculate the time accurate mean errors of the time integration schemes. The mean error is defined as the average of absolute differences of numerical and analytical solutions at nodes of the whole solution domain.

$$MEAN\ ERROR = \left(\sum_{i=1}^N |\theta_{i\ ANALYTICAL} - \theta_{i\ NUMERICAL}| \right) \times \frac{1}{N} \quad (3.8)$$

N represents the total number of nodes if the solution is node based or it represents the total number of cells if the solution is cell center based.

Node based finite difference approach is used for the solutions. Second order central differences are used for the spatial derivative terms. The mean error histories of the solutions up to $\tau = 0.5$ are plotted and compared.

It is seen in Figure 3.3 and Figure 3.4 that the behaviors of the Crank-Nicolson AF and Crank-Nicolson like AF schemes are very similar to each other for both of the structured grids. This means that, AF solution as proposed does not increase the order of error for structured grids for the current

case. Although simple AF method has higher order of error for both of the grids; it yields comparable results with Crank-Nicolson discretization.

Having higher order of error with simple AF method is an expected result. There exist explicit terms in the spatial discretization of Crank-Nicolson and these terms increase the order of the method.

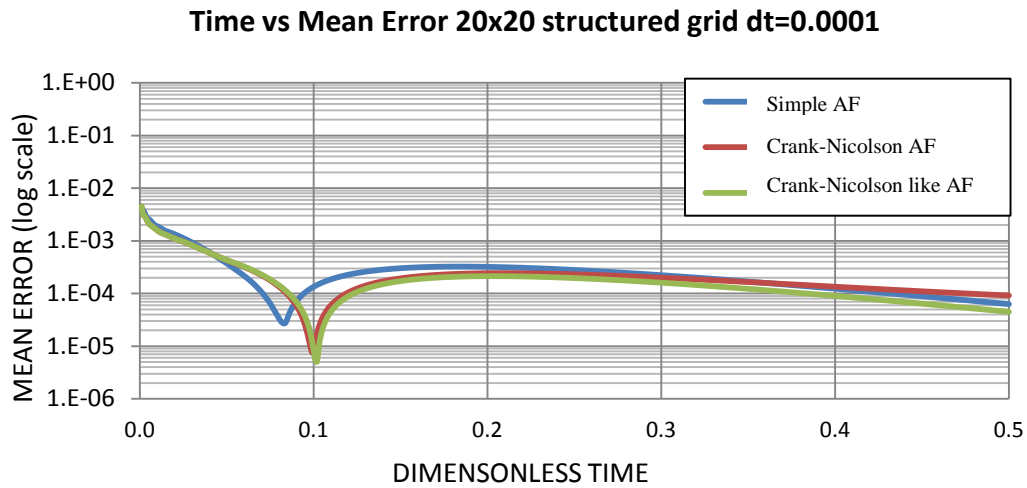


Figure 3.3 Mean Error histories of the numerical solutions for Crank-Nicolson Approximate Factorization and current Approximate Factorization methods using 20x20 structured cells with $\Delta \tau = 0.0001$ time step size

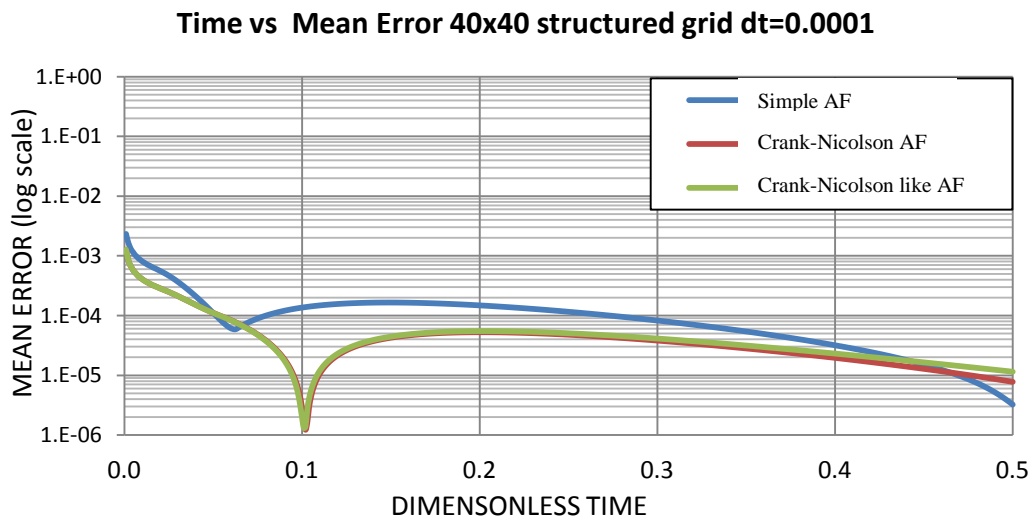


Figure 3.4 Mean Error histories of the numerical solutions for Crank-Nicolson Approximate Factorization and current Approximate Factorization methods using 40x40 structured cells with $\Delta \tau = 0.0001$ time step size

3.1.3 Validation of the Proposed Approximate Factorization with ACDI

Firstly time accurate results of Crank Nicolson AF-ACDI and simple AF-ACDI are compared with previous ACDI, Point-Gauss-Seidel, Line Gauss Seidel, Runge-Kutta order 4 and Full Implicit Laasonen Methods for structured cells. Then the comparison is carried on for quadrilateral unstructured cells.

Also analytic results of the problem is calculated at non-dimensional time $\tau = 0.1, 0.2$ and 0.3 . These analytic results are used to plot the error distributions through $Y=0.5$ constant line.

3.1.3.1 Comparison of Different Methods Using Structured Grid

The analytical results at $\tau = 0.1, \tau = 0.2$ and $\tau = 0.3$ are given in Figure 3.5.

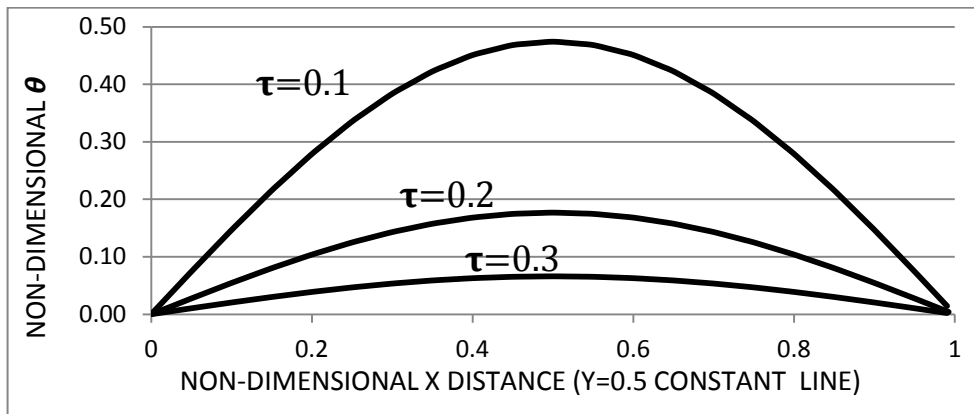


Figure 3.5 Analytical results through $Y=0.5$ constant line at $\tau = 0.1, \tau = 0.2$ and $\tau = 0.3$ dimensionless time

The absolute error distributions through $Y=0.5$ constant line are plotted for comparison of the Crank Nicolson AF-ACDI, simple AF-ACDI, ACDI, Point-Gauss-Seidel, Line Gauss Seidel, Runge-Kutta order 4 and Full Implicit Laasonen Methods in case of structured cells. Error at nodes is calculated using the formulation;

$$NODE\ ERROR = |\theta_{NODE\ ANALYTICAL} - \theta_{NODE\ NUMERICAL}| \quad (3.9)$$

Error distributions at $\tau = 0.1, \tau = 0.2$ and $\tau = 0.3$ are given in Figure 3.6, Figure 3.7 and Figure 3.8 respectively.

ERROR DISTRIBUTIONS AT Y=0.5 CONSTANT LINE $\tau=0.1$

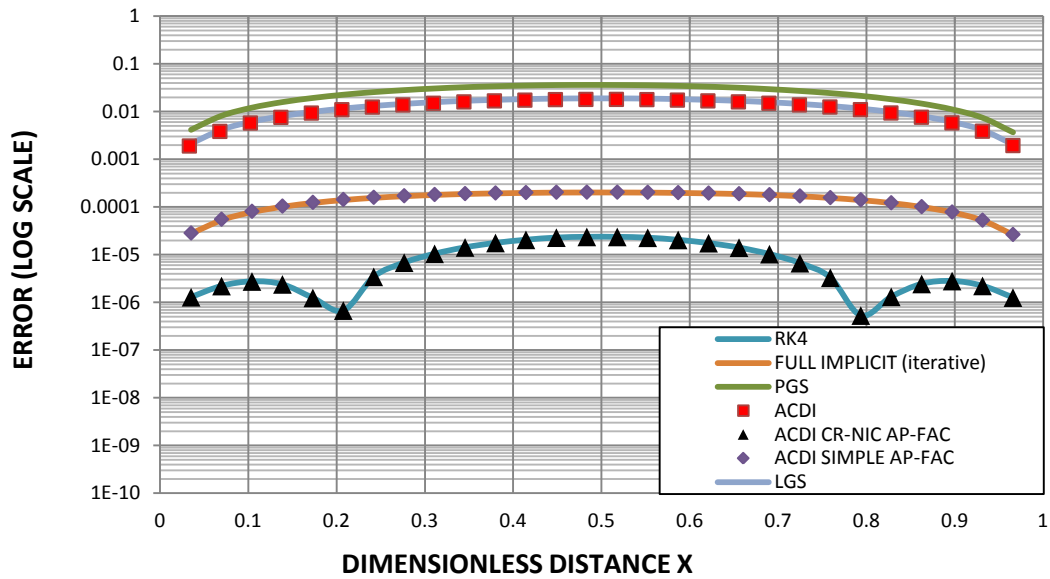


Figure 3.6 Error distribution through Y=0.5 constant line at $\tau = 0.1$, 20x20 structured grid, $\Delta \tau = 0.0001$

ERROR DISTRIBUTIONS AT Y=0.5 CONSTANT LINE $\tau=0.2$

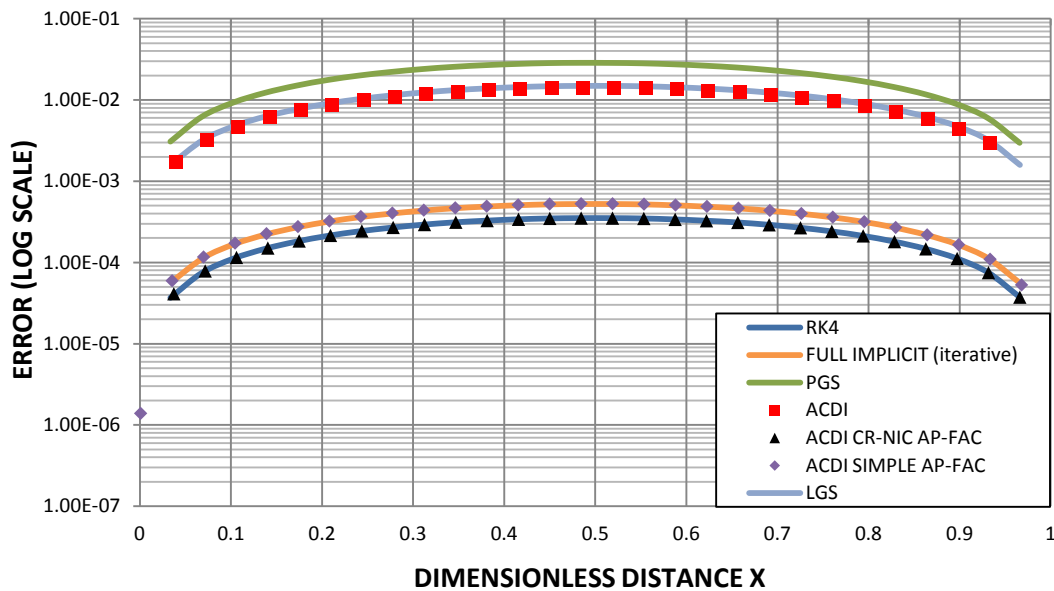


Figure 3.7 Error distribution through Y=0.5 constant line at $\tau = 0.2$, 20x20 structured grid, $\Delta \tau = 0.0001$

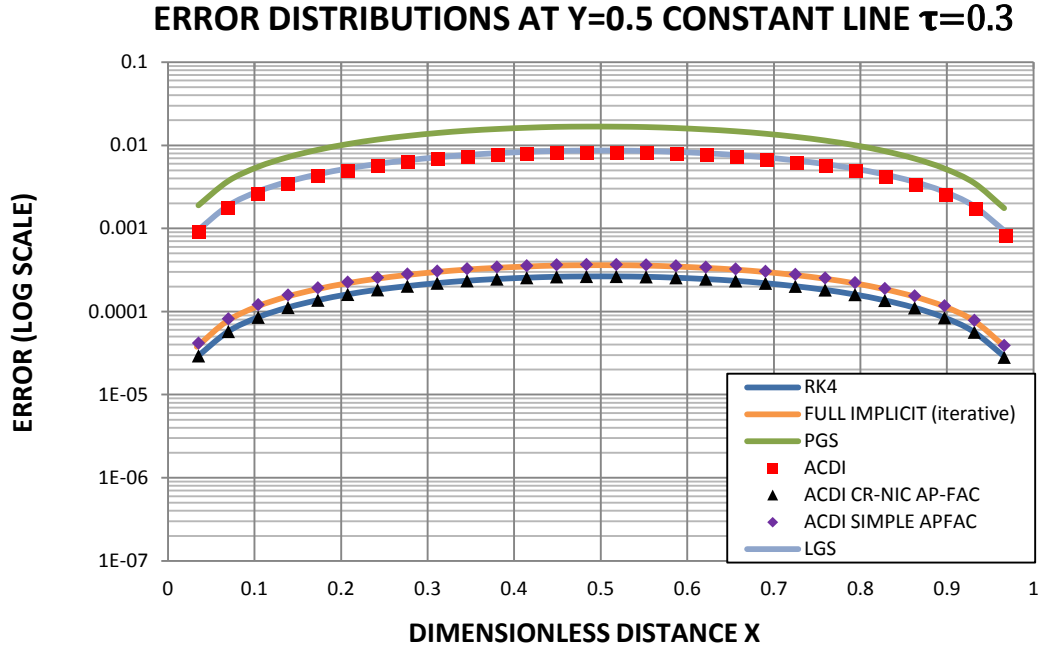


Figure 3.8 Error distribution through Y=0.5 constant line at $\tau = 0.3$, 20x20 structured grid, $\Delta \tau = 0.0001$

As seen from the Figures 3.6, 3.7 and 3.8 most accurate results are obtained with the methods Runge Kutta Order 4 and Crank-Nicolson AF ACDI. Interestingly the error distributions of these two methods are almost equal. Also the error distributions of simple AF ACDI and Full Implicit Laasonen Methods are almost same. Full implicit method is applied iteratively instead of a single large time step size to see the similarities between the AF ACDI and implicit approach.

It is possible to note that the explicit node terms added to the Crank-Nicolson scheme disturbs the implicitness of the Crank-Nicolson AF ACDI scheme while increasing its accuracy. The non-dimensional diffusion number which is the order of diffusivity of a property in a domain is defined as given below.

$$v = \alpha \frac{\Delta t}{\Delta x^2} \tag{3.10}$$

The non-dimensional Von-Neumann number for the current numerical solutions is 0.04.

As seen from the Figures 3.6, 3.7 and 3.8 the orders of the errors of Full Implicit Laasonen Method and Simple AF ACDI are almost equal to each other. It is possible to say that Simple AF-ACDI is an appropriate approximation of fully implicit scheme for diffusion problem solution with structured cells. Numerical tests show that simple AF ACDI is unconditionally stable for diffusion problem using structured cells.

Also CPU times of the solutions are obtained during the calculations. It is seen that Crank-Nicolson AF-ACDI is distinguishably faster than Runge-Kutta Order 4 method. The CPU time results are given in Table 3.1 for 20x20 structured grids. Runge-Kutta order 4 has 4 steps for the integration application and it means 4 times reconstruction of the variables. Also Crank-Nicolson AF ACDI do not deal with the whole discretized equation through the cell directions and handles only the half of the operations and this is another reason of being faster than Runge-Kutta order 4 method.

Table 3.1 CPU time required for the methods, 20x20 structured cells

METHOD USED	CPU TIME (s) FOR 1000 ITER
RK4	5.788
FULL IMPLICIT (iterative)	270.588
PGS	3.698
ACDI	4.492
ACDI CR-NIC AP-FAC	4.682
ACDI SIMPLE AP-FAC	4.152
LGS	4.140

The absolute error at selected point X=0.5 Y=0.5 is given in Figure 3.9 for different time step sizes. As seen from the figure the order of error of Crank Nicolson AF ACDI does not increase with the increasing time step size but it loses stability after a certain point.

The stability of Simple AF ACDI is not lost up to $\Delta \tau = 0.1$ and it gives meaningful results with only 2 iteration steps at $\tau = 0.2$ whereas the accuracy becomes poorer with the increasing time step size.

Both of the time accurate error distribution plots and order of accuracy plots proves that Crank-Nicolson AF ACDI shows the character of a high order explicit scheme whereas simple AF ACDI shows the character of a fast implicit scheme.

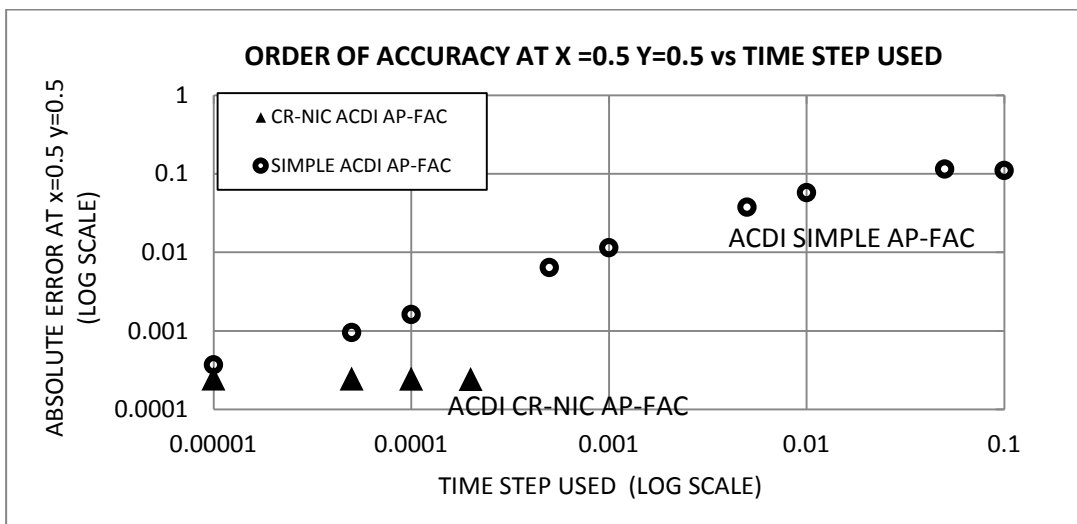


Figure 3.9 Variation of the absolute error at point X=0.5 Y=0.5 with the change of time step size for Crank Nicolson AF ACDI and Simple AF ACDI – solution at $\tau = 0.2$ with 20x20 structured grid

3.1.3.2 Comparison of Different Methods Using Unstructured Grid

Error distributions through $Y=0.5$ constant line are given in Figure 3.10, 3.11 and 3.12 at dimensionless time $\tau = 0.1, 0.2$ and 0.3 respectively. 297 element unstructured grid is used for the comparisons. The grid is given in Figure 3.14. Time step is selected as $\Delta \tau = 0.0001$ since all the integration schemes are stable for this time step.

It is seen that order of the accuracy is again almost same for Runge-Kutta order 4 and Crank-Nicolson AF ACIDI methods, also it can be observed in Table 3.2 that Crank-Nicolson AF ACIDI method is faster than RK4 method if the CPU time per iteration is taken into account.

The order of the error distributions were very close to each other for Simple AF ACIDI and Full Implicit Laasonen methods in case of structured cells. The order of the error is again in comparable orders for these two methods but the full implicit approximation of the simple AF ACIDI method is slightly disturbed for unstructured quadrilateral cells.

The order of the error is again lower than the previous ACIDI study for both of the AF ACIDI methods.

Non-dimensional θ contours at $\tau = 0.1, \tau = 0.2$ and $\tau = 0.3$ are given in Figure 3.15 for 297 element unstructured grid using Crank-Nicolson AF ACIDI.

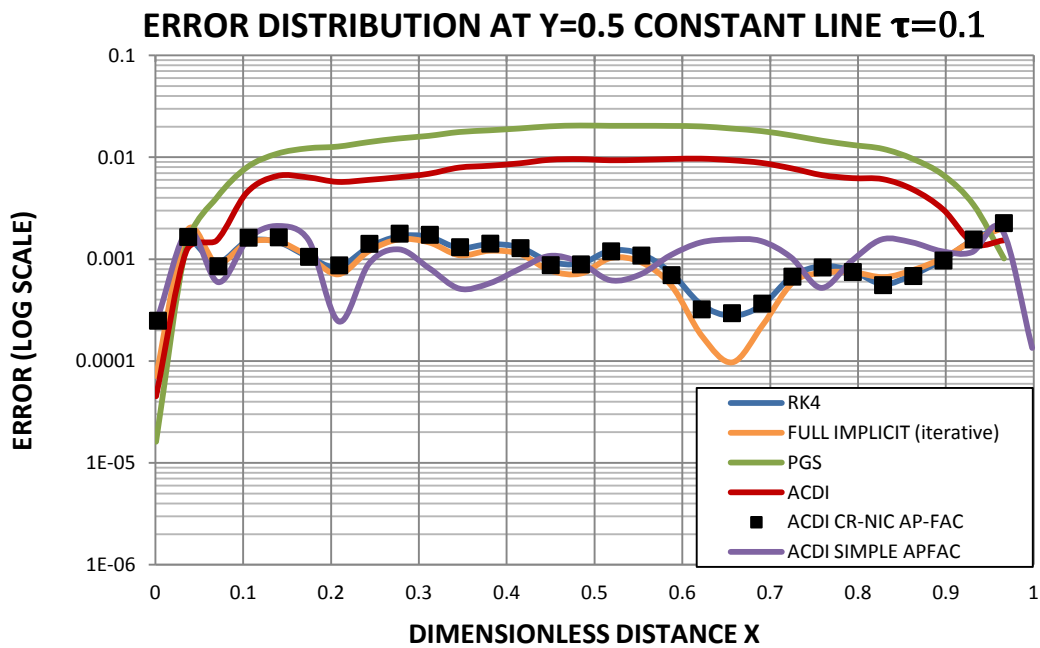


Figure 3.10 Error distribution through $Y=0.5$ constant line at $\tau = 0.1$, 297 elements unstructured grid, $\Delta \tau = 0.0001$

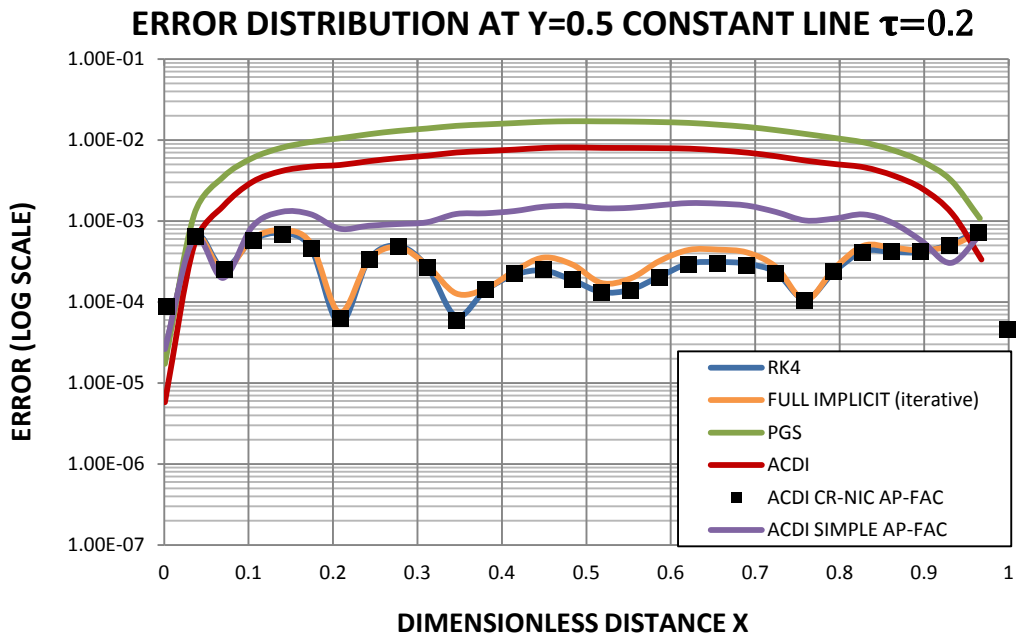


Figure 3.11 Error distribution through Y=0.5 constant line at $\tau = 0.2$, 297 element unstructured grid, $\Delta \tau = 0.0001$

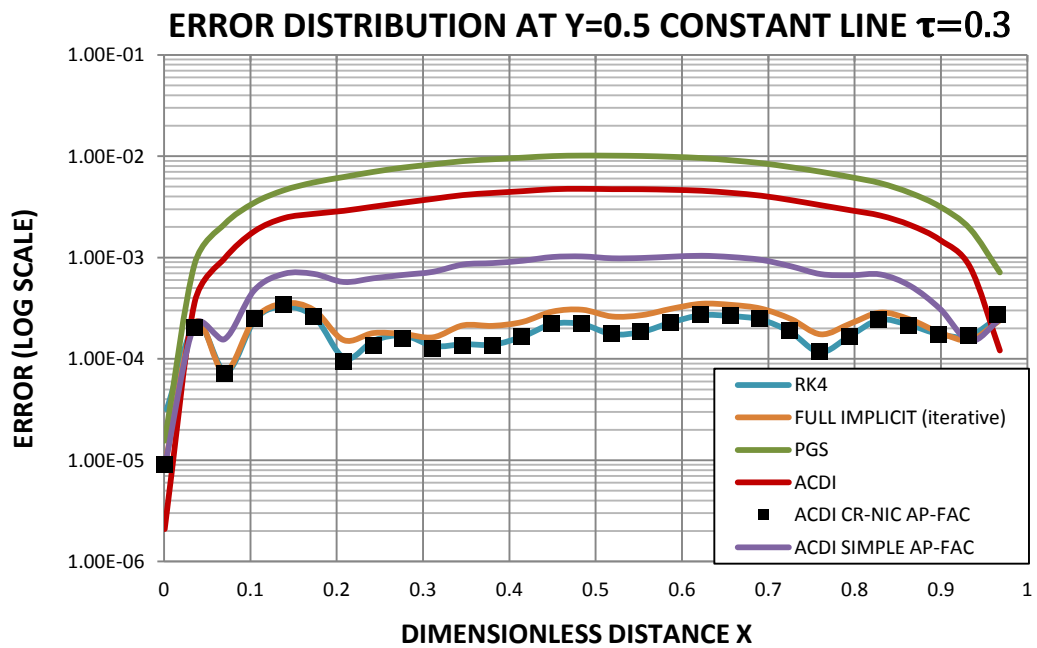


Figure 3.12 Error distribution through Y=0.5 constant line at $\tau = 0.3$, 297 element unstructured grid, $\Delta \tau = 0.0001$

Table 3.2 CPU time required for the methods, 297 unstructured cells

METHOD USED	CPU TIME (s) FOR 1000 ITER
RK4	3.8322
FULL IMPLICIT (iterative)	62.459
PGS	2.668
ACDI	3.052
ACDI CR-NIC AP-FAC	3.124
ACDI SIMPLE AP-FAC	2.828

The absolute error at selected point X=0.5 Y=0.5 is given in Figure 3.13 for different time step sizes.

The behaviors of the schemes are similar to structured grid case. The order of the absolute error do not increase with the increasing step size for Crank-Nicolson AF ACDI solution but the method has poor stability such as an explicit scheme.

The order of error increases with the increasing step size if the simple AF ACDI method is used. The method do not lose its stability up to $\Delta\tau = 0.1$ where the numerical results are calculated at $\tau = 0.2$.

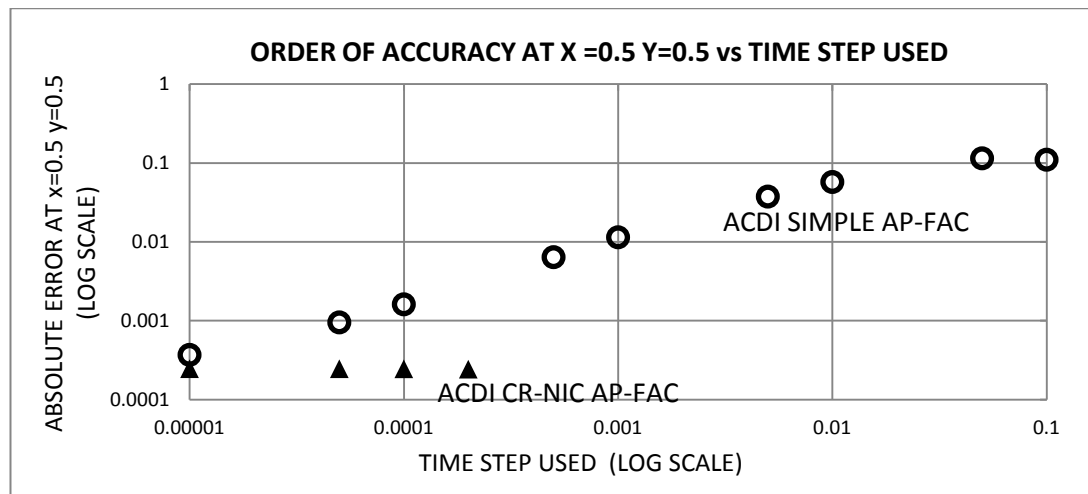


Figure 3.13 Variation of the absolute error at point X=0.5 Y=0.5 with the change of time step size for Crank Nicolson AF ACDI and Simple AF ACDI – solution at $\tau = 0.2$ with 297 element unstructured grid

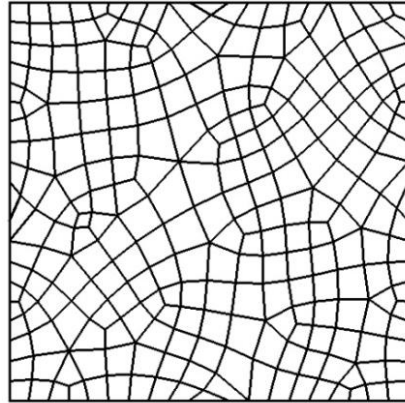


Figure 3.14 297 element unstructured grid that is used for time accurate comparisons

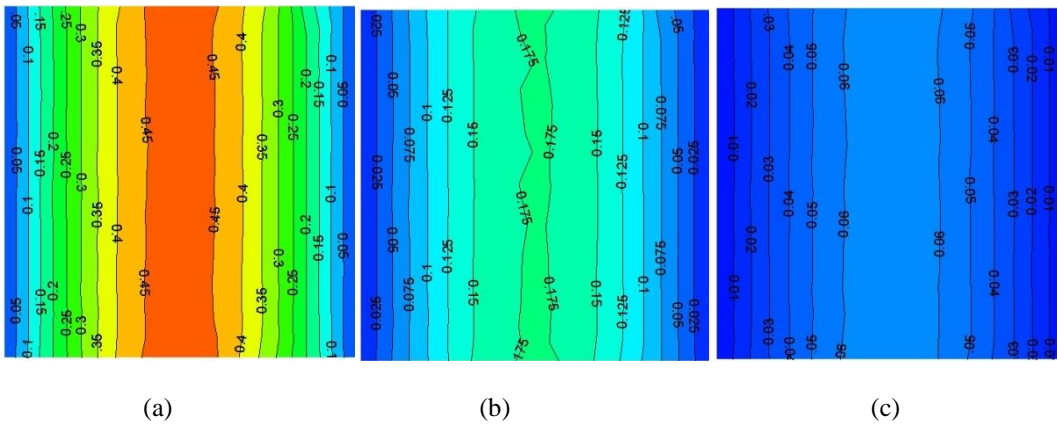


Figure 3.15 Non-dimensional temperature θ contours at (a) $\tau = 0.1$ (b) $\tau = 0.2$ (c) $\tau = 0.3$ for 297 element unstructured grid - Crank Nicolson ACDI AF

3.1.4 Comparison of Different Grid Types

The grids generated for the comparison are given in Figure 3.16. Trapezoidal rule flux calculation is used to obtain the numerical results. The resulting temperature contours and error distribution contours have been plotted at $t=0.1$ using $\Delta t=0.0001$ time step size, also average errors of the numerical solutions until $\tau=0.5$ have been compared in order to observe the behavior of the method for different cell types. Simple AF ACDI is used for the comparison.

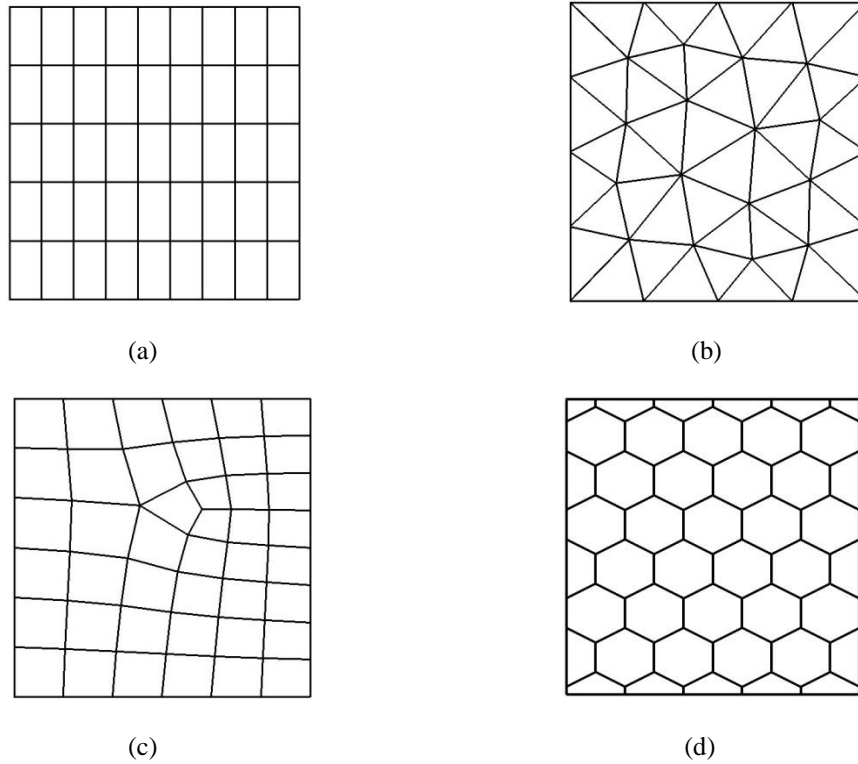


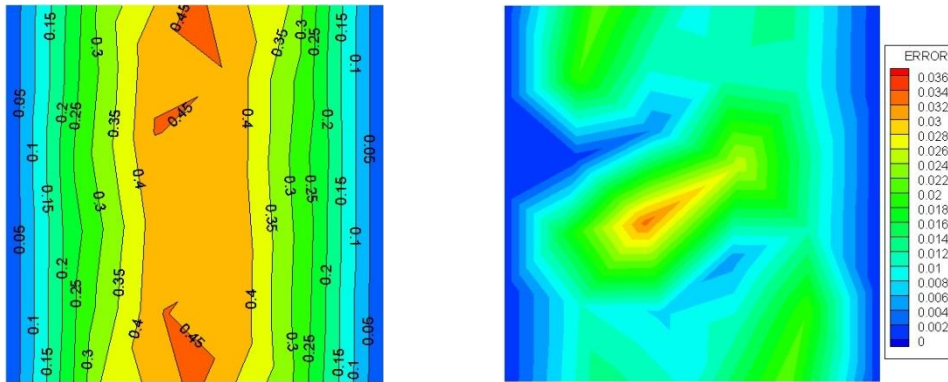
Figure 3.16 (a) 45 element structured grid. (b) 46 element triangular grid. (c) 43 element quadrilateral unstructured grid. (d) 44 element hybrid polygonal grid.

Non-dimensional temperature contours are given in Figure 3.17 for triangular, quadrilateral and hybrid polygonal grids at $\tau = 0.1$ non-dimensional time. Also absolute error distributions obtained using the analytical solution are represented. The non-dimensional temperature contours and absolute error contours for structured grid are given in Figure 3.18 separately. The result for structured grid is not given together with other grids since the error is in the order of 10^{-4} for structured grid whereas the error is in the order of 10^{-2} for other grids. The error distributions are not comparable and it is not possible to represent structured grid absolute error with unstructured grid absolute error distributions with the same legend.

If the results of unstructured grid solutions are compared, it will be observed that quadrilateral grid has the lowest error and triangular grid has the highest error.

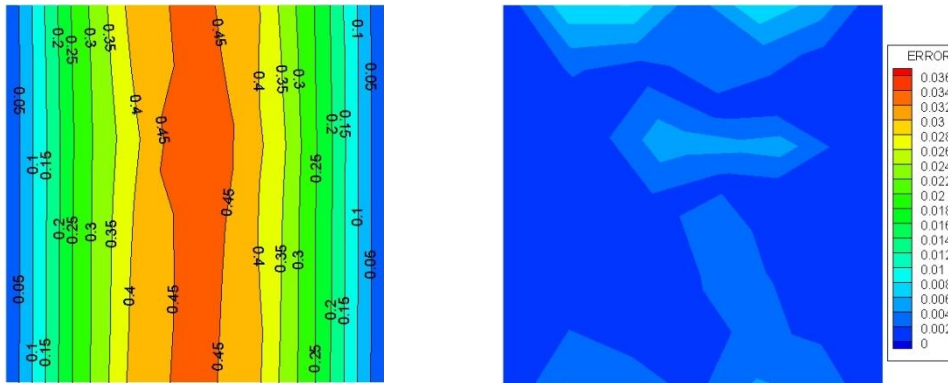
Hybrid polygonal grid does not generate high order of error in the regions where hexagonal elements exist. The high error regions take place over the pentagonal elements. The cell directions for this grid are given Figure in 3.19. It can be seen that the edges of the pentagonal elements are split into two to obtain elements with even number of edges. This application results with sharp turns of cell directions as well as low quality elements and also increased order of error.

Having highest error with triangular cells is also expected. The cell directions are broken inside the domain and this situation blocks the information flow as stated previously. The implicitness of the scheme is harmed with broken cell directions and the scheme becomes a hybrid of approximate factorization method and Point-Gauss-Seidel scheme.



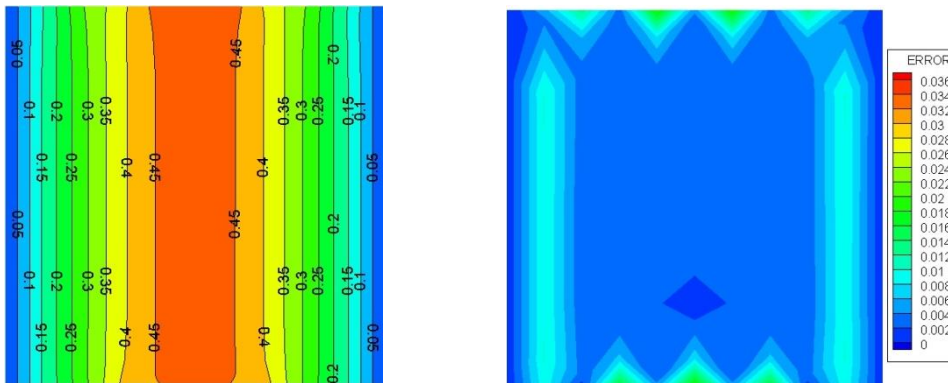
(a)

(b)



(c)

(d)



(e)

(f)

Figure 3.17 (a) Non-dimensional theta contours and (b) error contours for triangular grid (c) Non-dimensional theta contours and (d) error contours for quadrilateral grid and (e) Non-dimensional theta contours and (f) error contours for hybrid polygonal grid

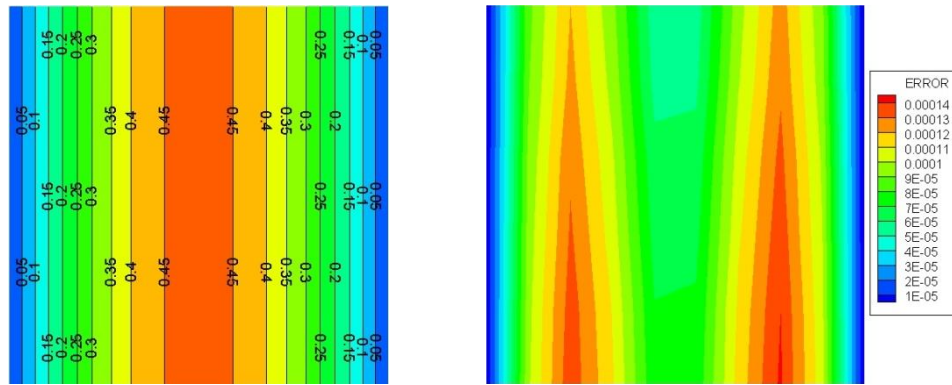


Figure 3.18 (a) Non-dimensional theta contours and (b) error contours for structured grid

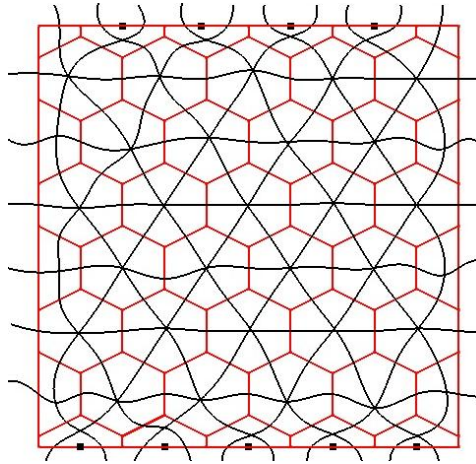


Figure 3.19 Cell directions for polygonal grid after splitting the edges of pentagonal elements

The mean error histories of different grid solutions are obtained up to $\tau=0.5$ using $\Delta \tau=0.0001$ time steps. The comparison of the mean error histories is given in Figure 3.20. The lowest error is obtained with structured grid and highest error is obtained with triangular grid up to $\tau=0.15$. Having even number of edges seems to be an advantage before approaching the steady state. By the way having regularly distributed cells become more important for more accurate results while approaching the steady state. Also it is possible to say that the arrangement of structured cells favors the gradient directions, thus it is an expected result to have relatively lower order of error with structured grids.

Mean error for different grid types $\Delta\tau=0.0001$

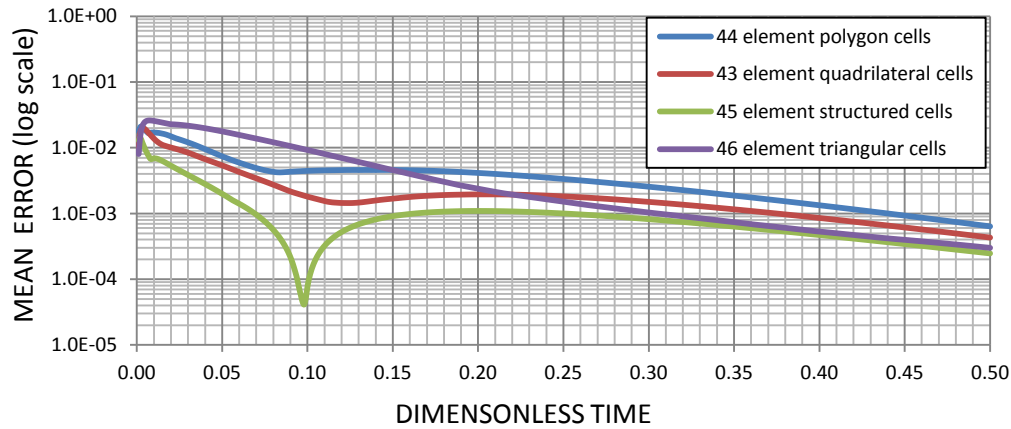


Figure 3.20 Error histories of the numerical solutions for different grid types

3.1.5 Comparison of Inverse Distance Weighting and Trapezoidal Rule Flux Calculations

In this part, effect of the order of the flux calculation is observed. The comparison is made for only 297 element unstructured quadrilateral grid and 1283 element unstructured quadrilateral grid. It is seen that the order of the flux calculation has almost no effect for structured cells for the problem of unsteady heat conduction on a rectangular plate, thus the error histories are not given for structured grids in this part.

The mean error curves of the inverse distance flux calculation results are very similar to each other for 297 element grid and 1283 element grid as seen in Figure 3.21. This means inverse distance flux calculation ends up in less dependency to grid size.

Applying higher order discretization decreases the order of the error for both of the grids as expected. But the decrease of the error is higher for the coarser grid.

The grids used for comparison is given in Figure 3.22.

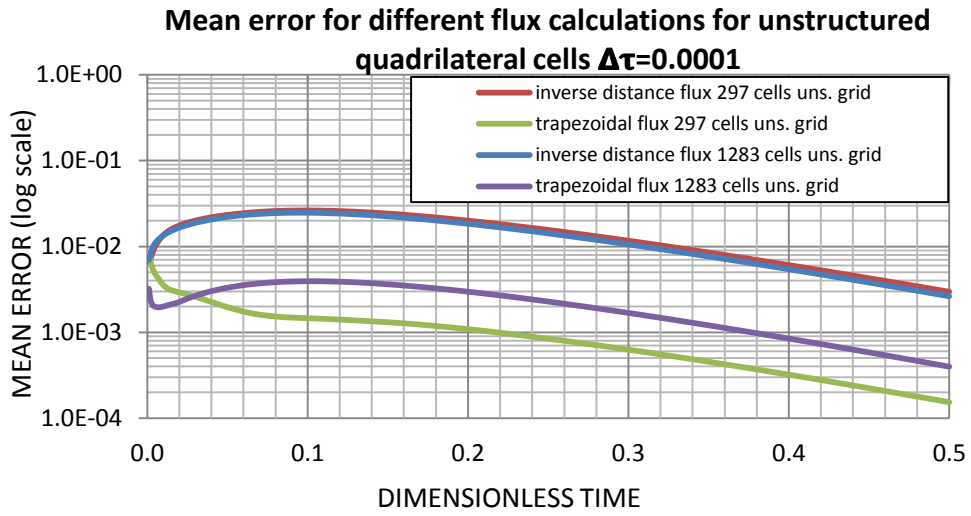


Figure 3.21 Error histories of the numerical solutions using higher and lower order flux calculation approaches with ACIDI AF for unstructured grids $\Delta\tau = 0.0001$ time step size

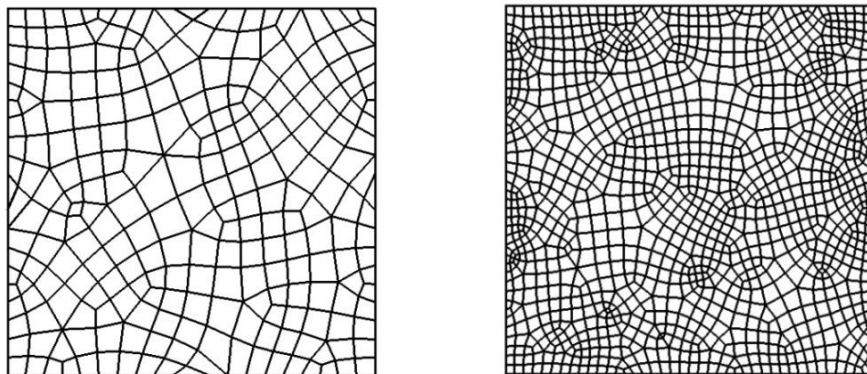


Figure 3.22 (a) 297 element and (b) 1283 element unstructured grids that used for the flux calculation method comparison

3.2 Comparison of Convergence Characters Using Steady Laplace Equation

In this section the method is applied to a steady problem. Incompressible potential flow around a cylinder is solved to observe the convergence behavior of the method for elliptic equations. The potential flow equation is solved using simple AF ACDI, Runge Kutta Order 4, Full Implicit Laasonen and Point Gauss Seidel iteration methods. Crank-Nicolson AF ACDI is not used for the comparisons since the method behaves like an explicit scheme. Full implicit method is again applied iteratively in order to reach a certain residual value of the solution.

The steady Laplace equation that is used for the solution of the potential equation is as given below;

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = 0 \quad (3.11)$$

ψ represents the stream function in the equation given above. The velocity component in x and y directions are;

$$u = \frac{\partial \psi}{\partial y} \quad \text{and} \quad v = -\frac{\partial \psi}{\partial x} \quad (3.12)$$

The equation is solved with the help of pseudo-time step. Equation can be rewritten as if it is an unsteady equation;

$$\frac{\partial \psi}{\partial t} - \frac{\partial^2 \psi}{\partial x^2} - \frac{\partial^2 \psi}{\partial y^2} = 0 \quad (3.13)$$

The solution converges to a steady state if pseudo-time step is used and the solutions before the convergence are meaningless in the manner of unsteadiness. As seen above the form of the equation with a virtually transient term is completely same with the unsteady diffusion equation and the factorization application is given in Appendix C.

3.2.1 Problem Specifications for Convergence Character Comparisons with Laplace Equation

Potential flow around a 2-D cylinder with a diameter of 1 m is solved for the comparisons. The free stream velocity is taken as 1 m/s and the analytic solution for the stream function is as given below.

$$\psi(x, y) = y - \frac{0.25y}{x^2 + y^2} \quad (3.14)$$

The stream function contours obtained using the analytical solution is as given below;

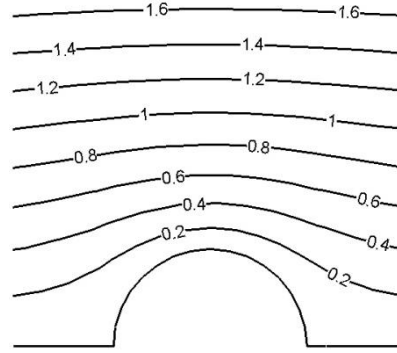


Figure 3.23 The stream function contours obtained using the analytical solution where the free stream velocity is 1 m/s

3.2.2 Comparison of Convergence Characters Using Structured Grid

The iteration numbers and CPU times of the methods to reach a previously defined root mean square residual are obtained using different pseudo-time step sizes for simple AF ACADI, Point Gauss Seidel Method, Full Implicit Laasonen and Runge Kutta order 4 methods. The residual value to be reached for the convergence is selected as 10^{-7} and the root mean square residual is defined as;

$$RMS\ RESIDUAL = \sqrt{\frac{1}{N} \sum_{i=1}^N (\psi_i^{n+1} - \psi_i^n)^2} \quad (3.15)$$

N represents the number of elements for cell centered calculations and number of node for node based calculations, n is the time indice and i is the element or node indice.

The 500 element structured grid is shown in Figure 3.28 (a). The Dirichlet boundary condition is used at the far field of the domain. Since the problem is symmetric, half of the domain is used for the solution and again Dirichlet boundary condition is used for the symmetry and wall faces where $\psi = 0$.

In Figure 3.24 required iteration number using different pseudo-time step sizes is plotted for simple AF ACADI, Point Gauss Seidel, Full Implicit and Runge-Kutta order 4 methods using 500 element structured grid.

It is seen that Runge-Kutta order 4 method has a poor stability character when it is compared with the other methods. The required pseudo-time step size for the convergence is larger than the other methods since small step sizes are required for the stability.

The iteration number required converges to a certain number with the increased pseudo-time step sizes for both of the Point Gauss Seidel and Full Implicit methods and both of the methods remain stable with the usage of relatively larger time step sizes.

Simple AF ACADI method has an optimum value of pseudo-time step size different than the Full Implicit and Point Gauss Seidel Methods. As seen from the figure the slope of the trend of the fully implicit scheme before converging to a certain iteration number and slope of the trend of the fully explicit scheme seem very close to each other. The Point Gauss Seidel Method leaves the trend before the simple AF ACADI and converges to a certain iteration number value whereas the number of time steps required increases for simple AF ACADI method after the optimum time step size value.

It is possible to conclude that simple AF ACDI is a better approximation of fully implicit scheme than Point Gauss Seidel until an optimum pseudo-time step size.

The convergence histories of Point Gauss Seidel and simple AF ACDI methods with the minimum iteration numbers required are plotted in Figure 3.26

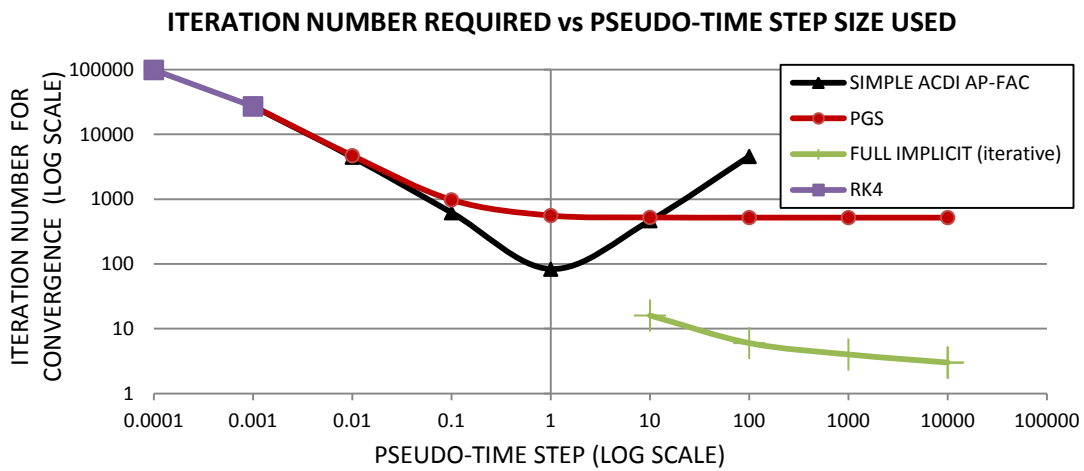


Figure 3.24 Number of iteration required for the convergence versus time pseudo-step size (s) using different methods for 500 element structured grid

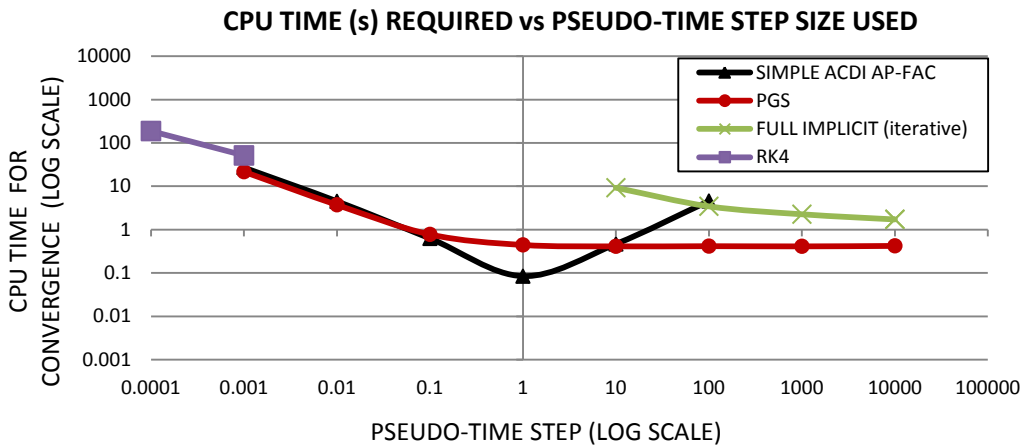


Figure 3.25 CPU time required for the convergence versus pseudo-time step size (s) using different methods for 500 element structured grid

In Figure 3.25 CPU time versus pseudo-time step size plots are given for the methods. It can be observed that Laasonen method loses its advantage of less iteration number requirement. The method inverts a mass matrix for each iteration step and this operation increases the requirement of CPU time for convergence.

In Figure 3.27 the stream function contours obtained with simple AF ACDI is given, also absolute error contours of numerical solution with simple AF-ACDI can be observed in Figure 3.28 (b). The absolute error at nodes is calculated using the formulation;

$$NODE\ ERROR = |\psi_{NODE\ ANALYTICAL} - \psi_{NODE\ NUMERICAL}| \quad (3.16)$$

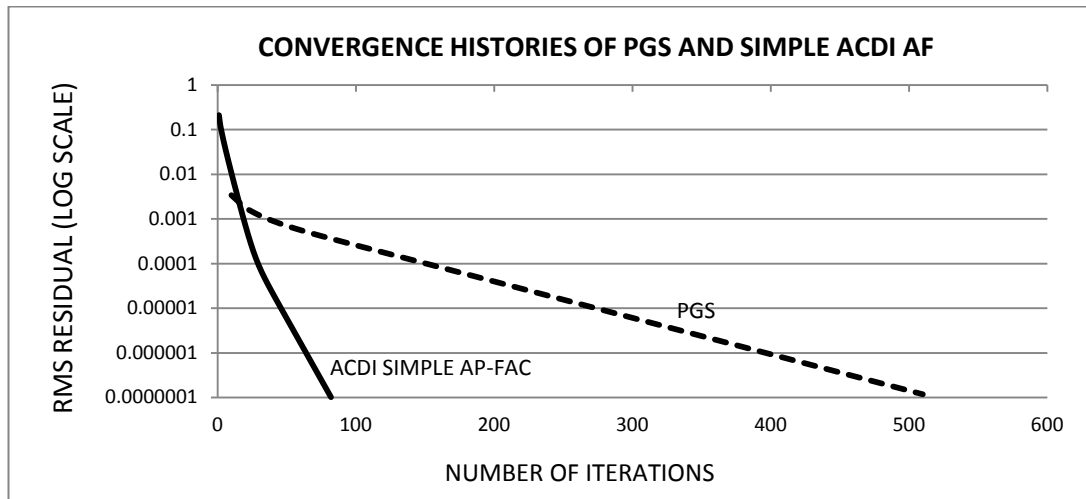


Figure 3.26 Convergence histories of simple AF ACDI and Point Gauss Seidel Method with optimum time step sizes with 500 element structured grid

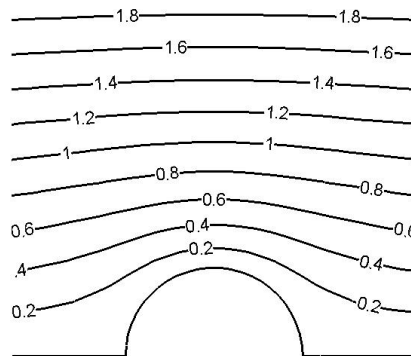


Figure 3.27 Contours of stream function around 2-D cylinder obtained with using simple AF ACDI with 500 element structured grid

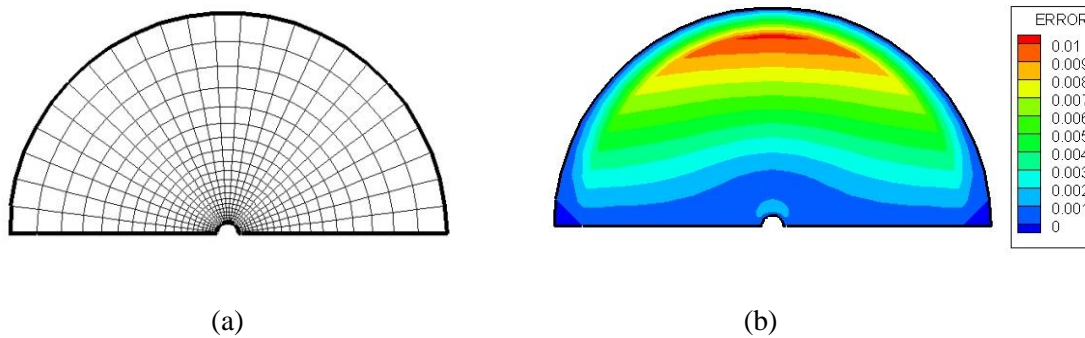


Figure 3.28 (a) 500 element structured grid used for the comparison of convergence behaviors of different methods (b) absolute error contours of stream function with the usage of simple AF ACDI

3.2.3 Comparison of Convergence Characters Using Unstructured Grid

The 789 element unstructured grid that is used for the numerical calculations are given in Figure 3.33 (a).

Iteration number required versus time step size for simple AF ACDI, Laasonen Method, Point Gauss Seidel and Runge Kutta order 4 methods are given for numerical solution of Laplace Equation with unstructured quadrilateral grid in Figure 3.29.

The curves obtained for the methods have similar behavior with the structured grid case results. Simple AF ACDI method has again an optimum value of pseudo-time step size but after this optimum value the scheme shows an unstable character instead of an increased iteration number requirement different than the structured grid case.

In case of usage of optimum pseudo-time step size simple AF ACDI gives convergent results about two times faster than the Point Gauss Seidel Method. The convergence histories of simple AF ACDI method and Point Gauss Seidel Method can be observed in Figure 3.31.

CPU time requirements with different time step sizes of the methods are given in Figure 3.30. Laasonen Method uses larger CPU time per iteration and Runge Kutta order 4 method requires relatively smaller time step sizes, thus both of the methods seem disadvantageous for obtaining convergent results faster.

The absolute error contours stream function is given in Figure 3.33 (b).

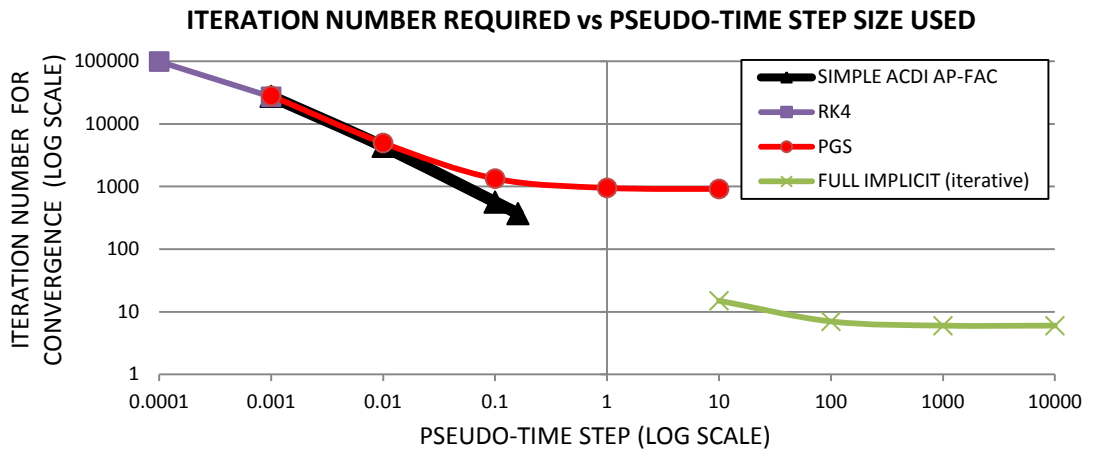


Figure 3.29 Number of iteration required for the convergence versus pseudo-time step size (s) using different methods for 789 element structured grid

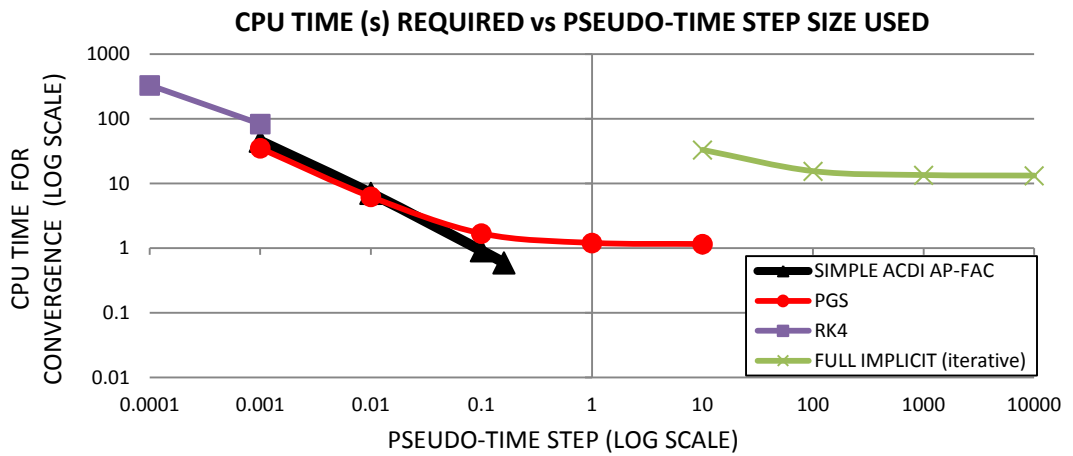


Figure 3.30 CPU time required for the convergence versus time step size (s) using different methods for 789 element unstructured grid

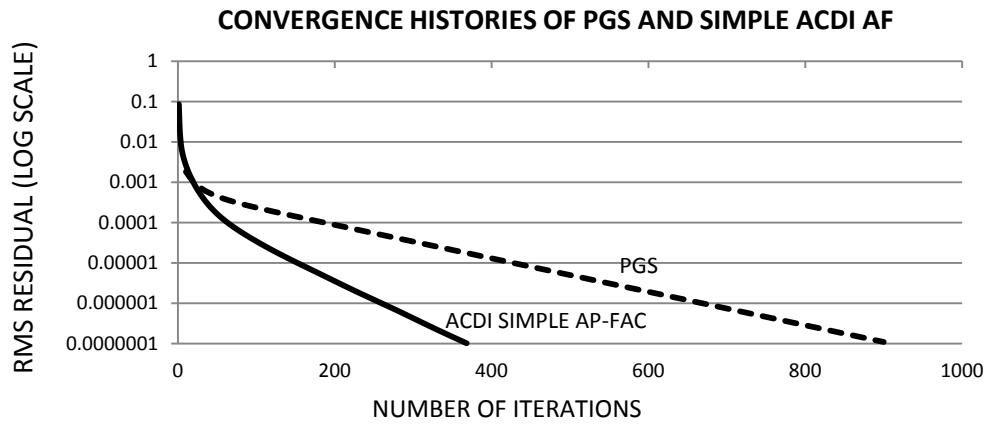


Figure 3.31 Convergence histories of simple AF ACDI and Point Gauss Seidel Method with optimum time step sizes for 789 element unstructured grid

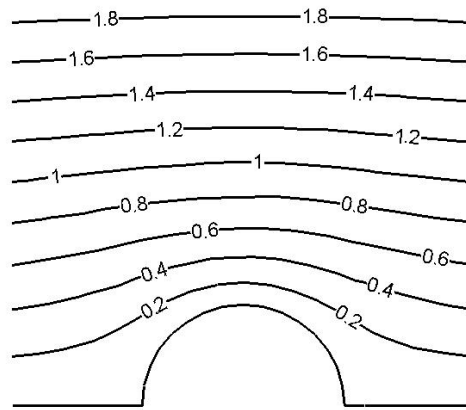


Figure 3.32 Contours of stream function around 2-D cylinder obtained with using simple AF ACDI with 789 element unstructured grid

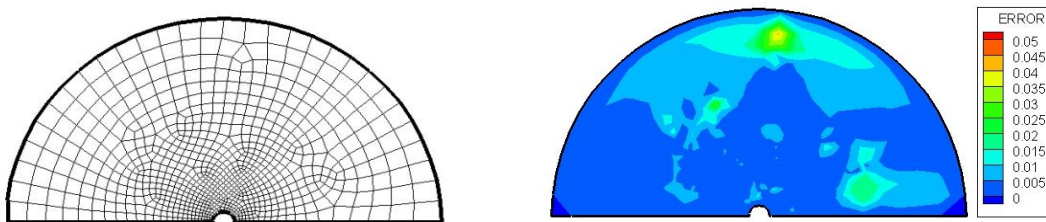


Figure 3.33 (a) 789 element unstructured grid used for the comparison of convergence behaviors of different methods (b) absolute error contours of stream function with the usage of simple AF ACDI

3.3 Time Accurate Comparisons Using Advection-Diffusion Equation

Alternating Cell Directions Implicit Method with Approximate Factorization is also tested for the time accurate solutions of well-known advection diffusion equation. Trials on Diffusion equation show that using Crank-Nicolson discretization results with a scheme that behaves like a high order explicit scheme with a relatively better stability behavior, but the aim of this thesis study is to represent a fast implicit scheme that is also useful for unstructured grids. Thus simple version of AF ACDI is tested for advection-diffusion equation instead of the Crank-Nicolson version.

If the advective terms are added to diffusion equation, the resulting partial differential equation will be;

$$\frac{\partial c}{\partial t} - D_x \frac{\partial^2 c}{\partial x^2} - D_y \frac{\partial^2 c}{\partial y^2} + u \frac{\partial c}{\partial x} + v \frac{\partial c}{\partial y} = 0 \quad (3.17)$$

Where D_x and D_y represent the diffusion coefficients in x and y directions and u and v represent the advection coefficients in x and y directions respectively. The dependent variable c can be considered as mass concentration for the model equation.

The model problem solved for the time integration comparisons do not include the advection in y direction, thus the equation reduces to;

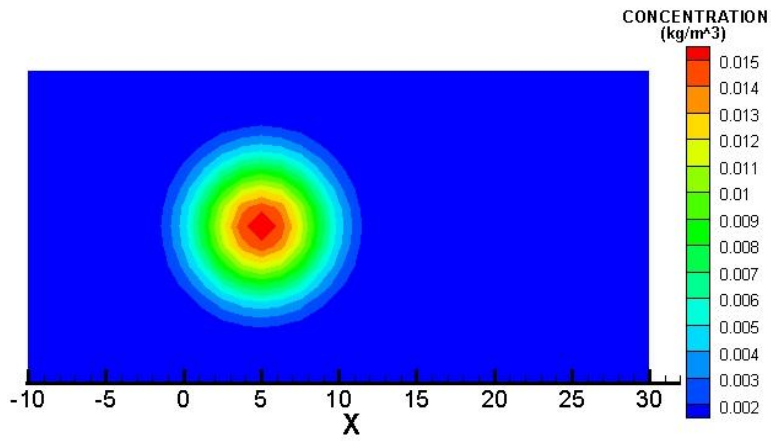
$$\frac{\partial c}{\partial t} - D_x \frac{\partial^2 c}{\partial x^2} - D_y \frac{\partial^2 c}{\partial y^2} + u \frac{\partial c}{\partial x} = 0 \quad (3.18)$$

3.3.1 Problem Specifications for Advection-Diffusion Equation Comparisons

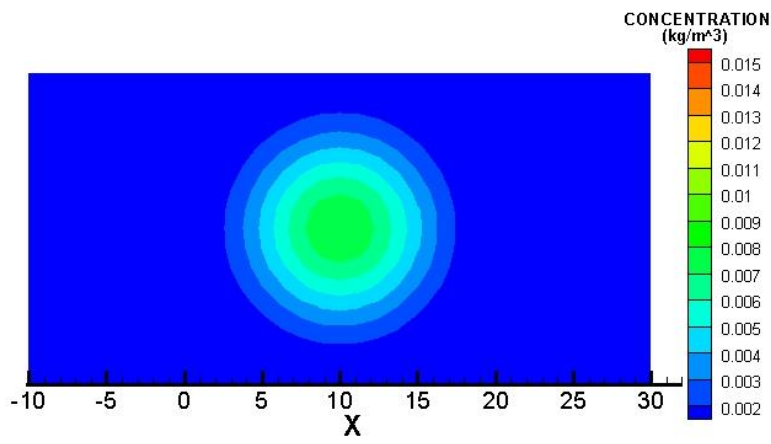
The diffusion and advection problem of an instantaneous point source is solved for the comparisons. The analytic solution that is used for comparisons assumes an infinite domain, thus the analytic solution is used as Dirichlet boundary condition at the boundaries in order not to deal with an extremely big domain with too many number of cells or complicated types of boundary conditions since the aim of the comparisons is simply to compare the performances of different iteration schemes. Time accurate analytic solution of the problem is [25];

$$c(x, y, t) = \frac{M}{L} \frac{1}{4\pi t \sqrt{D_x D_y}} \exp \left[-\frac{(x - x_0 - ut)^2}{4D_x t} - \frac{(y - y_0)^2}{4D_y t} \right] \quad (3.19)$$

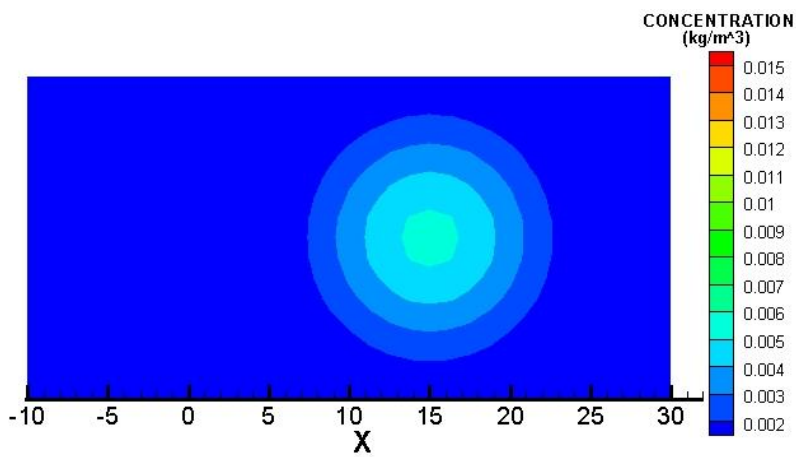
In the equation given above M/L is the mass released instantaneously and the unit for this term is kg/m since the problem is 2 dimensional. Diffusion and advection coefficients are taken as unity for numerical comparisons. The analytic solutions are plotted in Figure 3.34. The concentration distribution of 1 kg/m mass that is released from the point $(0,0)$ is given after 5 seconds, 10 seconds and 15 seconds.;



(a)



(b)



(c)

Figure 3.34 Contours of mass concentration at (a) $t = 5$ s (b) $t = 10$ (c) $t = 15$ s using analytical formulation

3.3.2 Comparison for Advection-Diffusion Equation Using Structured Grid

16x32 cell structured grid is used for the comparisons. Laasonen method is selected as fully implicit solution method and Runge-Kutta order-4 method is selected as fully explicit method. The numerical calculations are carried on using two different time steps which are mainly $\Delta t=0.01$ s and $\Delta t=0.25$ s. The non-dimensional Pecklet number for the case with unity diffusion and advection terms is $Pe = 1.25$. Pecklet number can be defined as the rate of advective behavior to diffusive behavior and calculated with the relation;

$$Pe = \frac{u\Delta x}{D} \quad (3.20)$$

The numerical schemes require upwinding methods for a stable solution where $Pe \geq 2$ [26]. Thus no upwinding applications are required for the numerical experiments of this section of the study.

The Courant number is a dimensionless number that defines the rate of transport per each time step and it is calculated with the definition;

$$\alpha = \frac{u\Delta t}{\Delta x} \quad (3.21)$$

The Courant number for the 16x32 structured grid is $\alpha=8 \times 10^{-3}$ if $\Delta t=0.01$ s and $\alpha=0.2$ if $\Delta t=0.25$ s.

In Figures 3.35, 3.36 and 3.37 the evolution of mass concentrations are given for simple AF ACDI, Full Implicit Laasonen and Runge-Kutta order 4 methods including the comparisons with analytical solution at $y=0$ constant line for $\Delta t=0.01$ s. It can be seen that all of the methods give slightly more diffusive results than the analytical solution and the solutions of the methods are very close to each other for relatively low Courant number.

The solutions at $y=0$ constant line are given in Figures 3.38, 3.39 and 3.40 for $\Delta t=0.25$ s. The results are relatively less diffusive than they should be and the accuracy of the Runge-Kutta order 4 method is better than the simple AF ACDI method and Laasonen method. It is possible to say that the behavior of the simple AF ACDI method is again close to full implicit Laasonen method. Increased time step decreases the accuracy of both of them. Also it is possible to say that decreased time step size results the schemes to behave more diffusive.

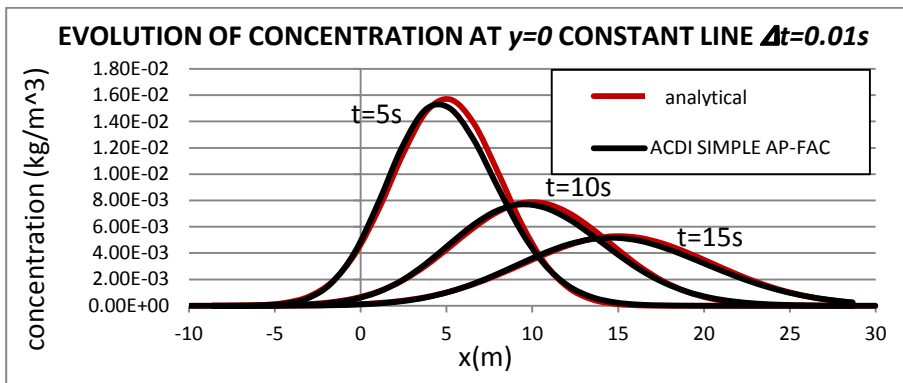


Figure 3.35 Comparison of concentration distributions obtained with analytical solution and numerical solution using simple AF ACDI for 16x32 structured cells and $\Delta t=0.01s$

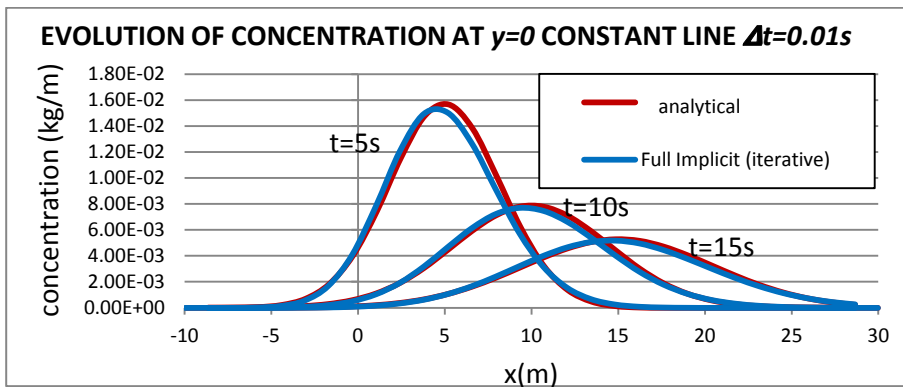


Figure 3.36 Comparison of concentration distributions obtained with analytical solution and numerical solution using Laasonen method for 16x32 structured cells and $\Delta t=0.01s$

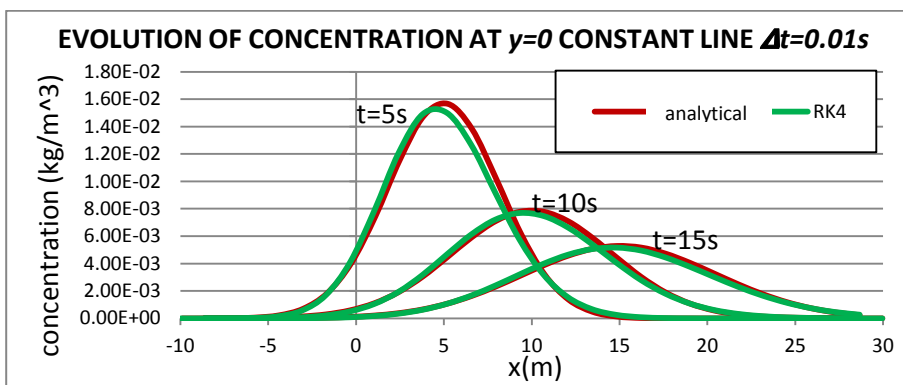


Figure 3.37 Comparison of concentration distributions obtained with analytical solution and numerical solution using Runge-Kutta order 4 method for 16x32 structured cells and $\Delta t=0.01s$

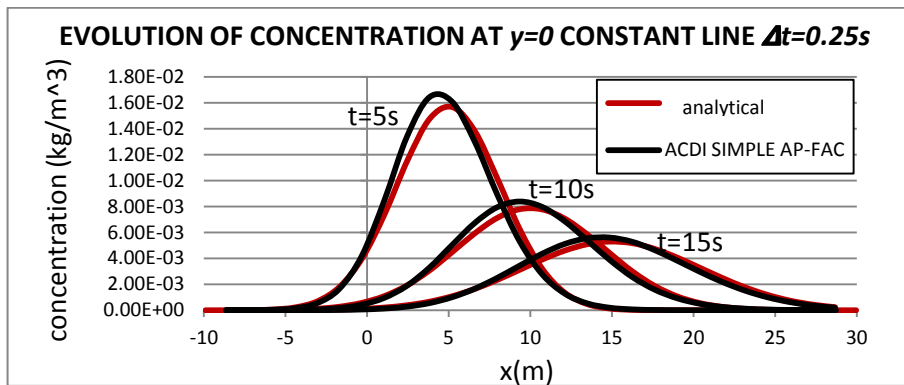


Figure 3.38 Comparison of concentration distributions obtained with analytical solution and numerical solution using ACIDI SIMPLE AF for 16x32 structured cells and $\Delta t=0.25s$

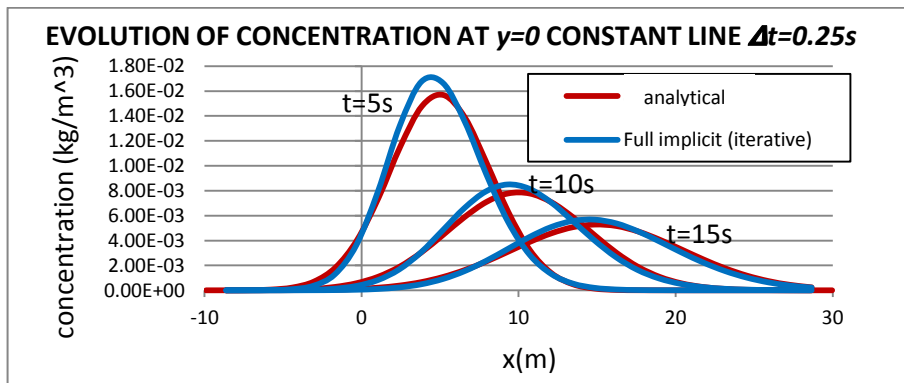


Figure 3.39 Comparison of concentration distributions obtained with analytical solution and numerical solution using Laasonen method for 16x32 structured cells and $\Delta t=0.25s$

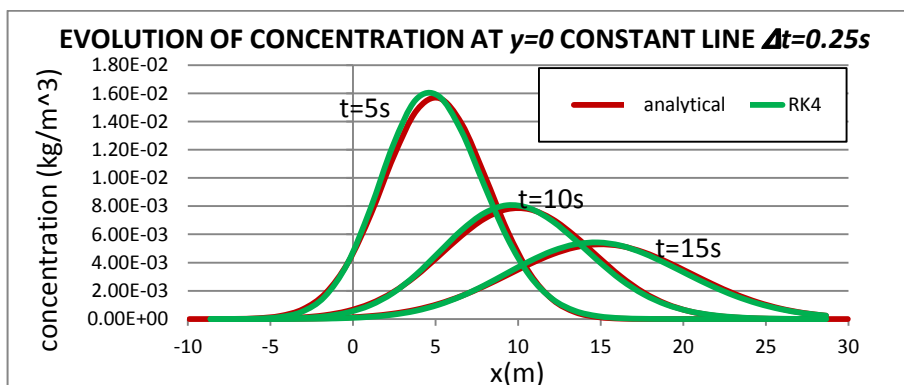


Figure 3.40 Comparison of concentration distributions obtained with analytical solution and numerical solution using Runge-Kutta order 4 method for 16x32 structured cells and $\Delta t=0.25s$

3.3.3 Comparison for Advection-Diffusion Equation Using Unstructured Grid

654 element unstructured grid that is used for the comparison of advection-diffusion equation solutions with different methods are given in Figure 3.47 (a). The numerical solutions are again obtained with simple AF ACDI, Full Implicit Laasonen and Runge Kutta order 4 methods using time steps $\Delta t=0.01$ s and $\Delta t=0.25$ s.

The maximum Courant number is $\alpha=0.02$ inside the domain where $\Delta t=0.01$ s. The numerical solutions at $y=0$ constant line are plotted and compared with the analytical solution at $t=5, 10$ and 15 s for all of the three methods and the results are given in Figure 3.41, 3.42 and 3.43. It can be seen that the solutions are again very close to each other but this time numerical solutions are less diffusive than they should be.

The numerical results obtained by using time step $\Delta t=0.25$ s are given in Figures 3.44, 3.45 and 3.46 including the comparisons with analytical solutions. Both Full Implicit and simple AF ACDI method gives relatively less diffusive numerical results then the exact solution and the accuracy of the numerical solutions decreases with the increased Courant number similarly to structured grid advection-diffusion solution. The maximum Courant number inside the domain is $\alpha=0.51$. For this Courant number with unstructured quadrilateral cells Simple AF ACDI and Full Implicit methods give more accurate results than the Runge-Kutta order 4 methods.

If the mass concentration plots at $y=0$ constant line are inspected it will be seen that very smooth results are obtained for both of the stable solutions of Full Implicit Laasonen and simple AF ACDI methods even with the unstructured cells. Also linear distribution of mass concentration contours can be observed in Figure 3.47 (b). The linear distribution of the variables at nodes is provided by the usage of k-exact least squares reconstruction.

The mass concentration contours at $t=5, t=10$ and $t=15$ s are given in Figure 3.47 (b) for simple AF ACDI solution.

The contours of mass concentration of simple AF ACDI method shows that the regions where relatively smaller cells take place are less diffusive then the larger cell size regions inside the domain. This situation supports the idea that lower Courant number solutions behave more diffusive then relatively higher Courant number solutions.

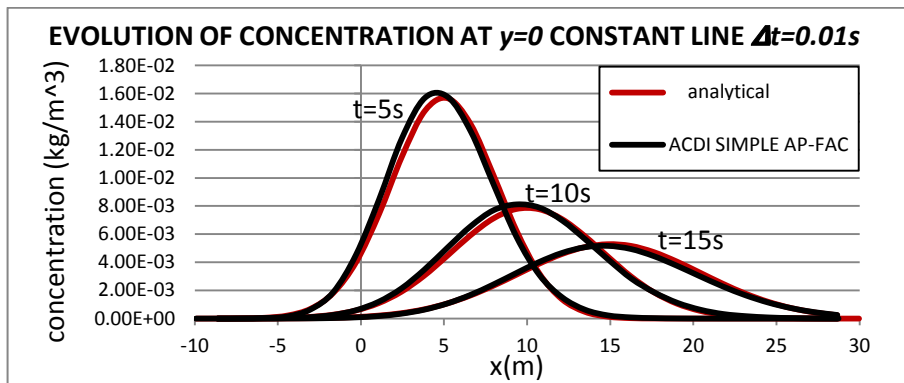


Figure 3.41 Comparison of concentration distributions obtained with analytical solution and numerical solution using ACIDI SIMPLE AF for 624 element unstructured cells and $\Delta t=0.01s$

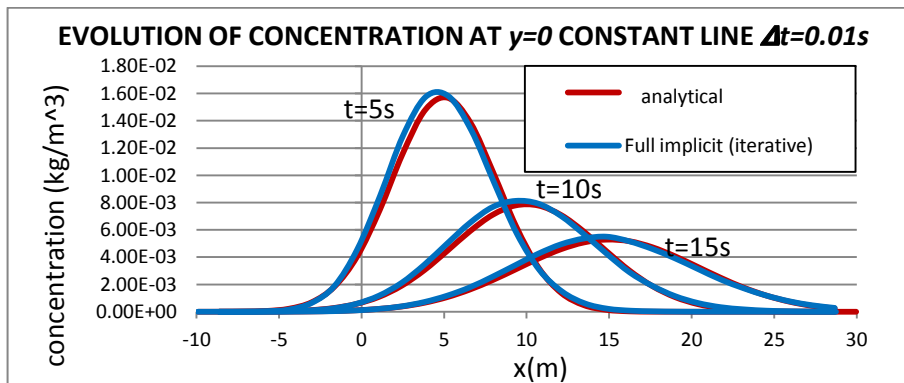


Figure 3.42 Comparison of concentration distributions obtained with analytical solution and numerical solution using Laasonen method for 624 element unstructured cells and $\Delta t=0.01s$

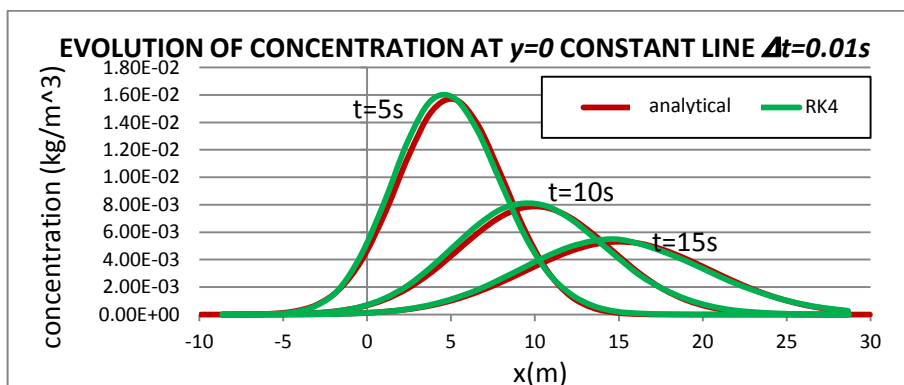


Figure 3.43 Comparison of concentration distributions obtained with analytical solution and numerical solution using Runge-Kutta order 4 method for 624 element unstructured cells and $\Delta t=0.01s$

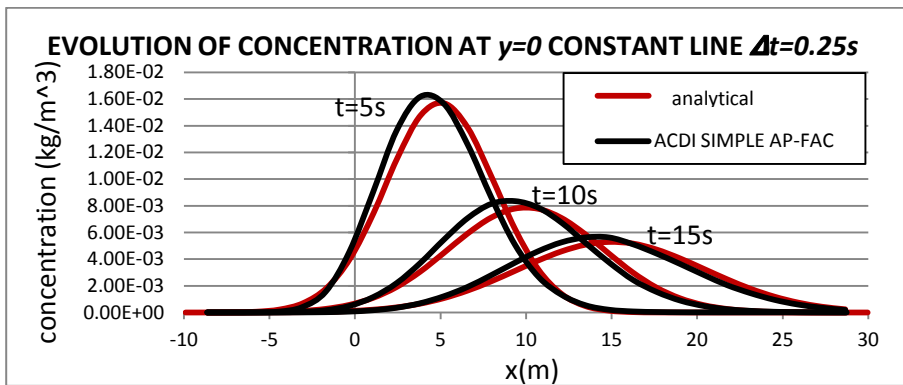


Figure 3.44 Comparison of concentration distributions obtained with analytical solution and numerical solution using ACIDI SIMPLE AF for 624 element unstructured cells and $\Delta t=0.25s$

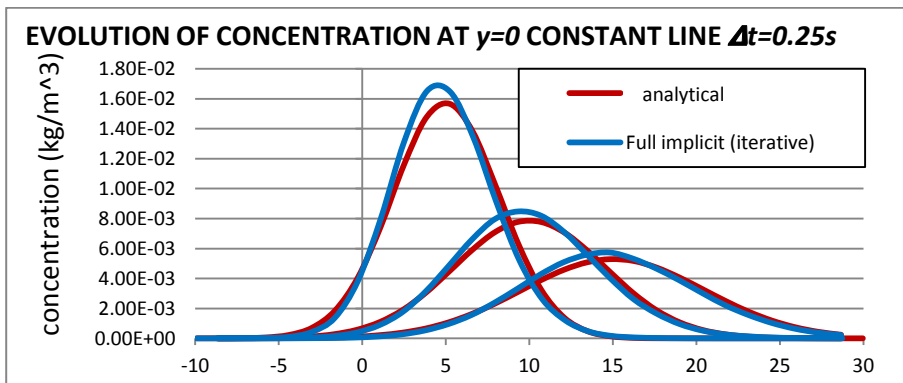


Figure 3.45 Comparison of concentration distributions obtained with analytical solution and numerical solution using Laasonen method for 624 element unstructured cells and $\Delta t=0.25s$

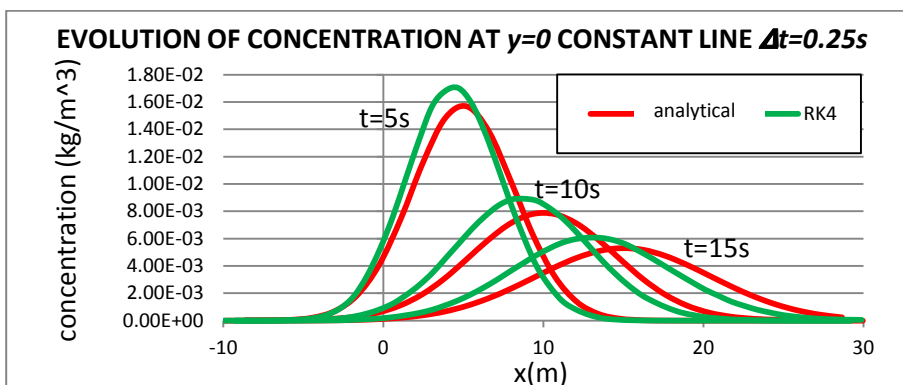
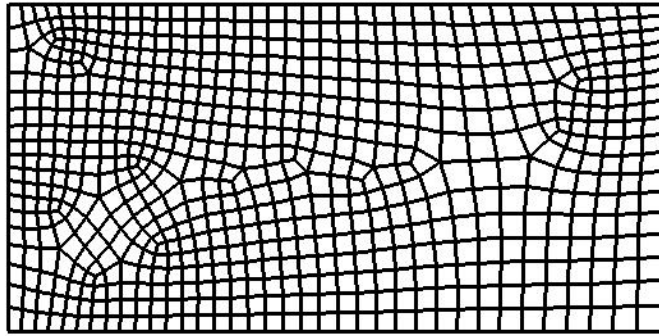
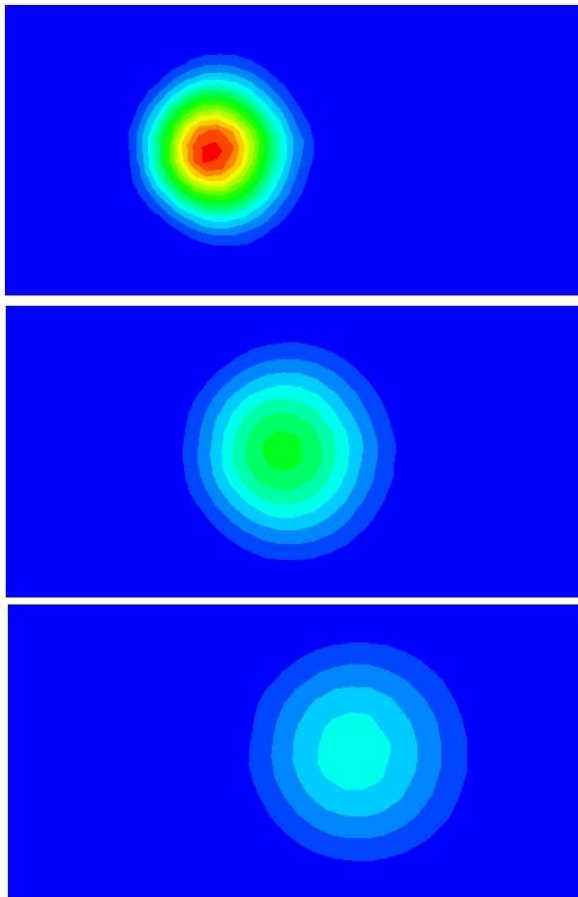


Figure 3.46 Comparison of concentration distributions obtained with analytical solution and numerical solution using Runge-Kutta order 4 method for 624 element unstructured cells and $\Delta t=0.25s$



(a)



(b)

Figure 3.47 (a) Grid used for unstructured grid comparisons (b) evolution of mass concentration for simple AF-ACDI method solution with $\Delta t=0.25s$

3.4 Transonic Full Potential Solver Using AF ACIDI

A full potential flow solver has also been generated using the AF ACIDI method. Full potential equation is a scalar and nonlinear equation which includes the assumptions of irrotational, isentropic and inviscid flow.

The full potential equation is usually used for subsonic and transonic flow problems since the accuracy of the solutions decreases for strong shock cases where the isentropic assumption through the shock is not valid anymore. Holst [27] states that the full potential solution gives relatively good approximations of Euler solution for the cases that Mach number does not exceed the value 1.3.

The unsteady and conservative form of the 2-D equation is;

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} = 0$$

$$u = \frac{\partial \varphi}{\partial x} \text{ and } v = \frac{\partial \varphi}{\partial y} \quad (3.22)$$

Where u is velocity component in x direction, v is velocity component in y direction, φ is the velocity potential and ρ is the density.

Then the steady form of the equation will be;

$$\frac{\partial}{\partial x} \left(\rho \frac{\partial \varphi}{\partial x} \right) + \frac{\partial}{\partial y} \left(\rho \frac{\partial \varphi}{\partial y} \right) = 0 \quad (3.23)$$

The relation between pressure and density is;

$$\frac{p}{p_\infty} = \left(\frac{\rho}{\rho_\infty} \right)^{\gamma} \quad (3.24)$$

P is the pressure and γ is the specific heat ratio. Calculation of the density using the free stream velocity is;

$$\rho^* = \left[1 + \frac{\gamma - 1}{2} Ma^2 (1 - q^{*2}) \right]^{\frac{1}{\gamma - 1}} \quad (3.25)$$

ρ^* and q^* are the non-dimensional density and non-dimensional free stream velocity which are calculated using the equations;

$$\rho^* = \frac{\rho}{\rho_\infty} \quad \text{and} \quad q^* = \frac{u^2 + v^2}{q_\infty} \quad (3.26)$$

Using artificial time for the iterations and integrating the conservative form of the full potential equation gives;

$$\int_A \frac{\partial \varphi}{\partial t} dA - \int_C \rho \frac{\partial \varphi}{\partial n} ds = 0 \quad (3.27)$$

The equation is non-linear because of the relation between ρ and φ . The problem is handled via using the density value as the constant value at a specific time on the cell edge of the interest. The factorized equation can be seen in Appendix C.

Trapezoidal rule is used for the flux calculations again but using the inverse distance weighting for the calculation of the node values results with fluctuations in the solution of the full potential equation. Thus K-exact least squares method is used for the reconstruction of variables at nodes and the center of the edges.

The behavior of the equation is elliptic for subsonic regions, parabolic for sonic line and hyperbolic for supersonic regions, thus density biasing is required for supersonic region . Well-known artificial viscosity method is used for stability in the supersonic regions of the solution. The density term is replaced with the biased term that is given as below [28];

$$\tilde{\rho} = \rho - \mu \frac{\partial \rho}{\partial s} \quad (3.28)$$

The derivative term given in the equation above is;

$$\frac{\partial \rho}{\partial s} = \frac{u}{q} \frac{\partial \rho}{\partial x} \Delta x + \frac{v}{q} \frac{\partial \rho}{\partial y} \Delta y \quad (3.29)$$

And μ is the switching function;

$$\mu = \max \left[0, 1 - \frac{M_c^2}{M^2} k \right] \quad (3.30)$$

M is the local Mach number, M_c is the cut-off Mach number which is about 0.98 and k is a programmer defined constant between 1 and 3.

3.4.1 Problem Specifications for Full Potential Equation Solution

The full potential solution of geometries with blunt leading edge is a more challenging problem than slender bodies, thus subcritical and supercritical cases for 2-D cylinder is selected as a benchmark problem. The pressure coefficients for subcritical case are compared with the analytical result adapted from the study of Saied and Alireza [29] and the pressure coefficients obtained for supercritical case are compared with the full potential solutions of Djojodihardjo and Widodo [30]. In case of the existence of angle of attack satisfaction of the Kutta condition is required, but only axi-symmetric flow is taken into account for the current study.

The flow around a 2-D cylinder with a diameter of 1 m is solved using two different grids for two different free stream Mach numbers. The free stream Mach numbers are selected as 0.25 and 0.7. First of the grids is a structured c-grid and composed of 1500 elements. Second grid used is an unstructured grid that is composed of 720 quadrilateral elements. The appearances of the grids used can be seen in Figure 3.48 (a) and Figure 3.48 (b). 50 elements are used around the semi-cylinder for both of the grids.

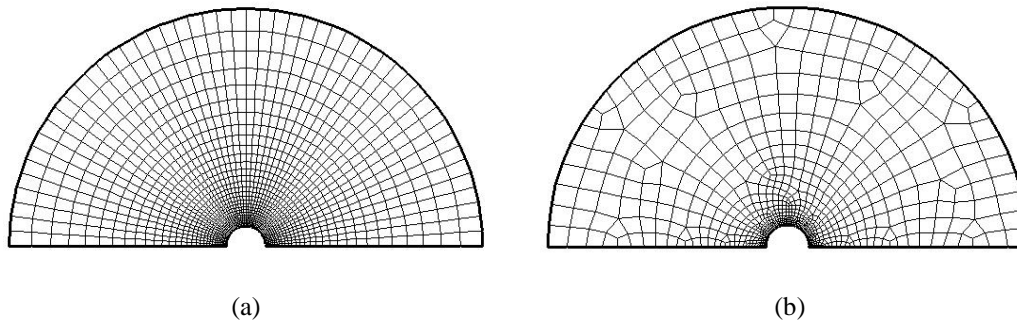


Figure 3.48 (a) Structured grid and (b) unstructured grid used for the full potential equation solution

3.4.2 Full Potential Equation Solution Using Crank-Nicolson AF ACDI

The plots of Mach contours obtained for $Ma_\infty=0.25$ are given in Figure 3.49 for structured grid and in Figure 3.50 for unstructured grid including the appearances of the grids around the cylinder. The effect of k-exact least squares reconstruction can be seen on the solution domain of quadrilateral unstructured grid. The linear distribution of the variables result with highly smoothed contours. The comparison of the numerically obtained pressure coefficients over the surface of the cylinder are compared with an analytical solution and shown in Figure 3.51 [29]. The results of structured and unstructured grid solutions are in a good agreement whereas the numerical pressure distribution results are slightly more diffusive than they should be.

Mach contours for supercritical case where $Ma_\infty=0.7$ are given in Figure 3.52 for structured grid and in Figure 3.53 for unstructured grid. Also the pressure coefficient distributions are compared with the full potential solution of Djojodihardjo and Widodo [30]. In Figure 3.54 it can be observed that both of the shocks that captured by using the current study are not steep enough, but rest of the pressure coefficient curves are in a good agreement. Soulis [28] states that the selection of the switching function k affects the accuracy in serious manner. The selection of this parameter or using relatively coarse mesh over the surface of the cylinder might be the reason of having a flat weak shock.

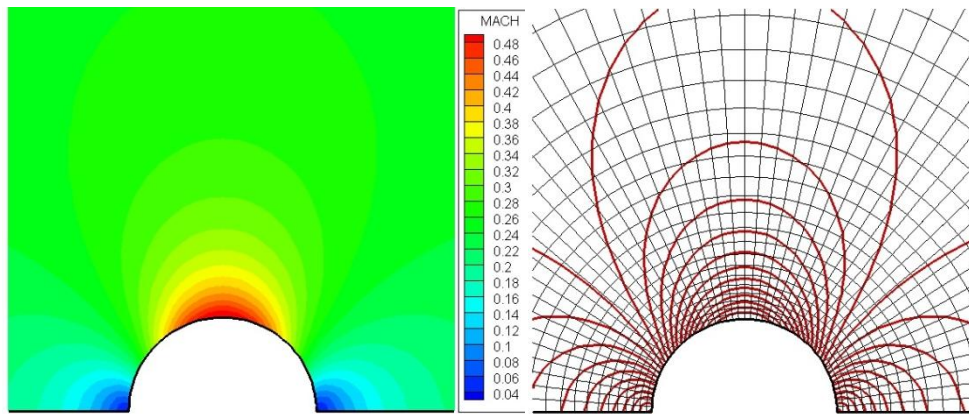


Figure 3.49 Mach contours around the cylinder and the appearance of the structured grid $Ma_\infty=0.25$ (Crank Nicolson AF ACDI solution)

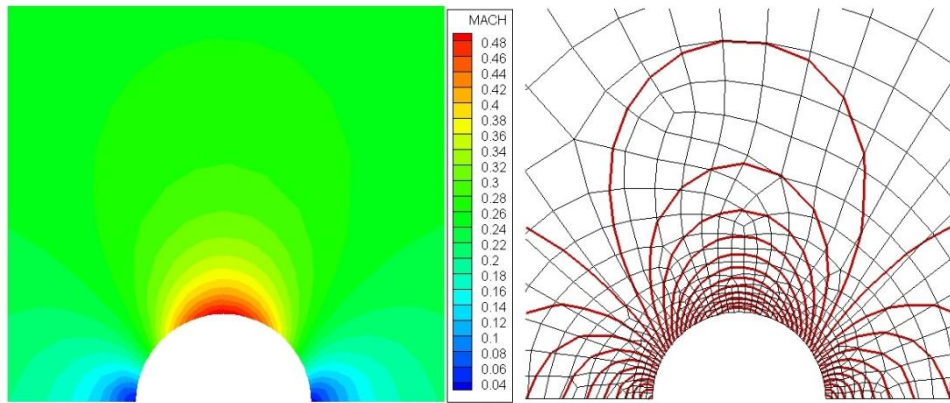


Figure 3.50 Mach contours around the cylinder and the appearance of the unstructured grid $Ma_\infty=0.25$ (Crank Nicolson AF ACDI solution)

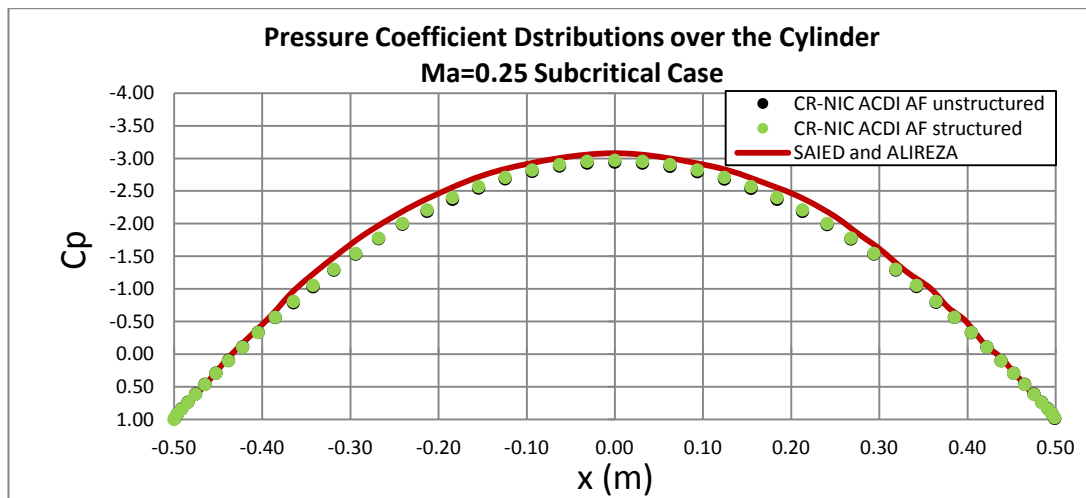


Figure 3.51 Comparison of pressure coefficients with the analytical results adapted from Saied and Alireza [29] $Ma_\infty=0.25$ (Crank Nicolson AF ACDI solution)

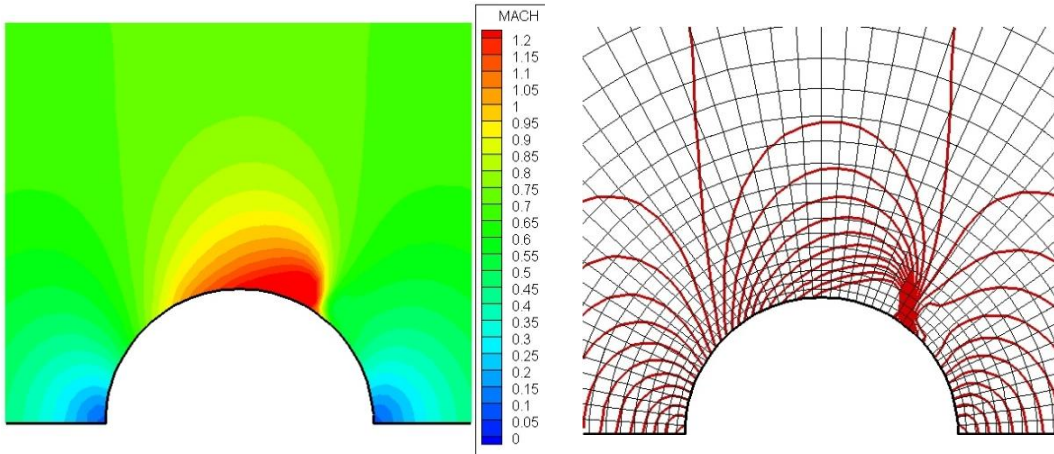


Figure 3.52 Mach contours around the cylinder and the appearance of the structured grid $Ma_\infty=0.7$ (Crank Nicolson AF ACDI solution)

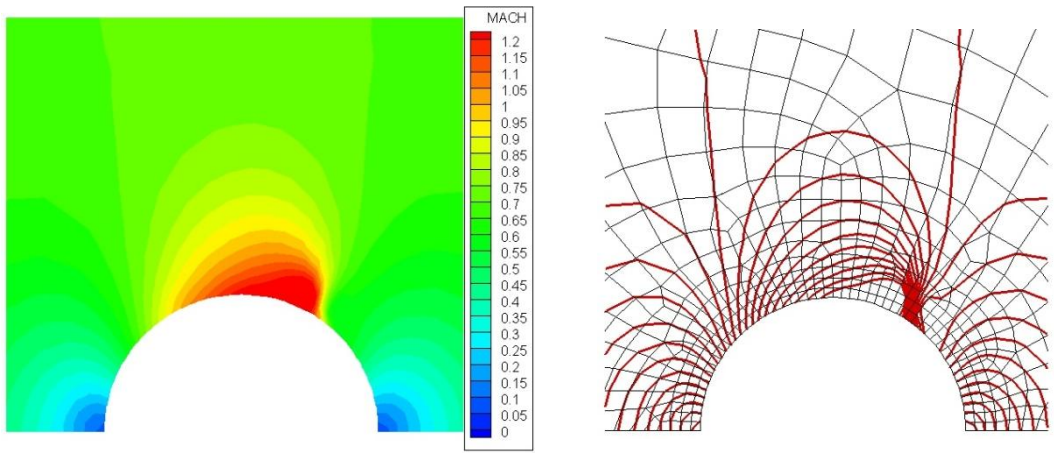


Figure 3.53 Mach contours around the cylinder and the appearance of the unstructured grid $Ma_\infty=0.7$ (Crank Nicolson AF ACDI solution)

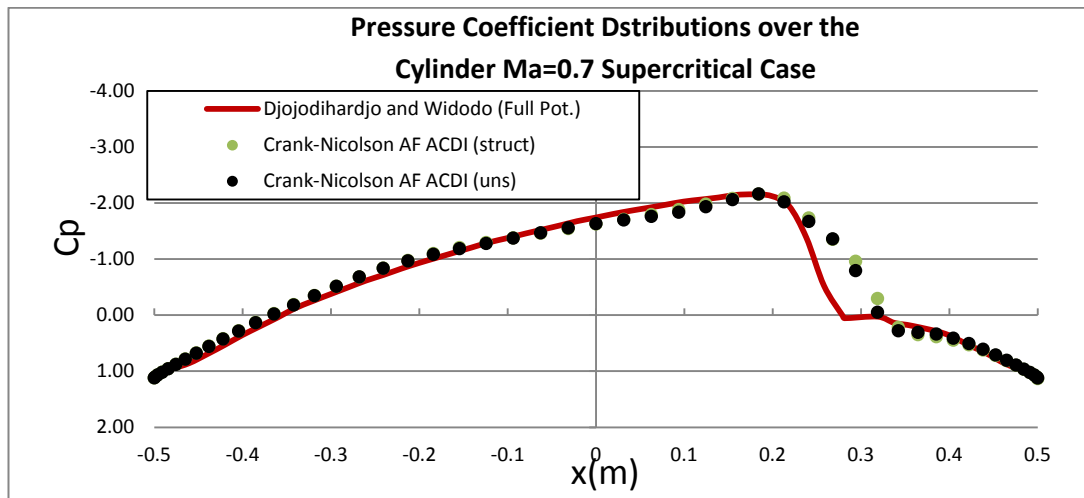


Figure 3.54 Comparison of pressure coefficients with the full potential solutions adapted from Djojodihardjo and Widodo [30] $Ma_\infty=0.7$ (Crank Nicolson AF ACDI solution)

The artificial viscosity term that is used for the density biasing is first order accurate. If Mach contours and the pressure coefficient distributions for supercritical solution are inspected well, it can be seen that the agreement of the pressure coefficient comparisons begin to break up beginning with the sonic line. Obviously the effect of the artificial viscosity term increases with the increasing Mach number. Thus the solution becomes closer to a first order accurate solution while approaching to weak shock region.

3.4.3 Full potential Equation Solution Using Simple AF ACDI

The trials on the solution of the full potential equation is also carried on using simple AF ACDI since the problem solved is a steady problem and the simple AF ACDI is expected to give more convergent results than Crank Nicolson AF ACDI. The grids that shown in Figure 3.48 are used again for $Ma_\infty=0.25$ and $Ma_\infty=0.7$. The comparisons of pressure coefficients with the analytic solution [29] where $Ma_\infty=0.25$ are given in Figure 3.55. The agreement with the analytic solution is worse than the Crank-Nicolson AF ACDI as expected but the solution is relatively more convergent than the Crank Nicolson AF ACDI. Example RMS residual history of the of the simple AF ACDI solution is given in Figure 3.57. The solution reaches to the level of 10^{-7} in about 750 iterations for 1500 element grid.

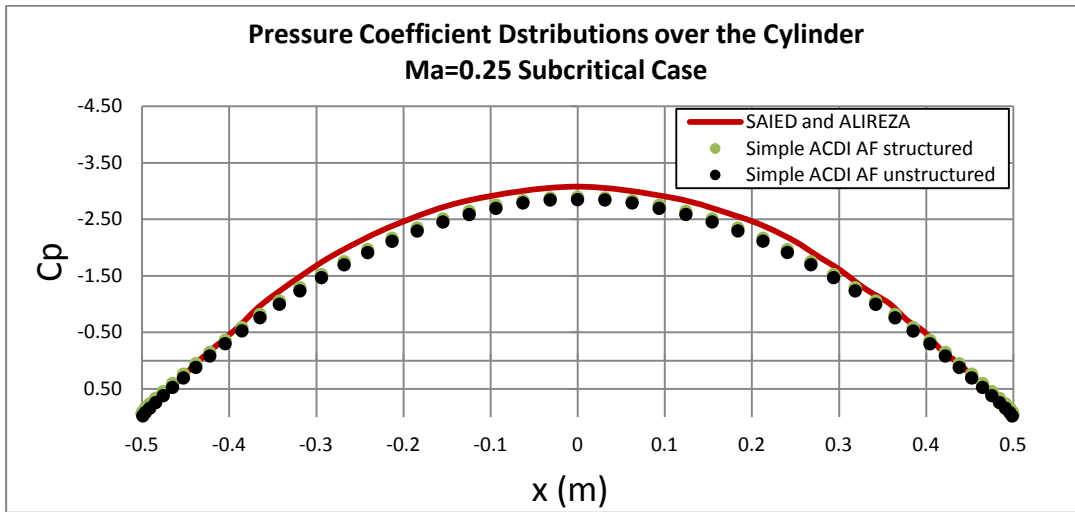


Figure 3.55 Comparison of pressure coefficients with the analytical results adapted from Saied and Alireza [29] $Ma_\infty=0.25$ (Simple AF ACDI solution)

The comparisons with the full potential solution of Dijojodihardjo and Widodo [30] are given in Figure 3.56. Weak shock is again more flat than expected.

In Figure 3.57 example residual histories for subcritical and supercritical cases are given. It is seen that very convergent solutions have been obtained for the subcritical case, but the first order artificial viscosity term shows its effect from the very beginning of the solution since the object has a very blunt leading edge.

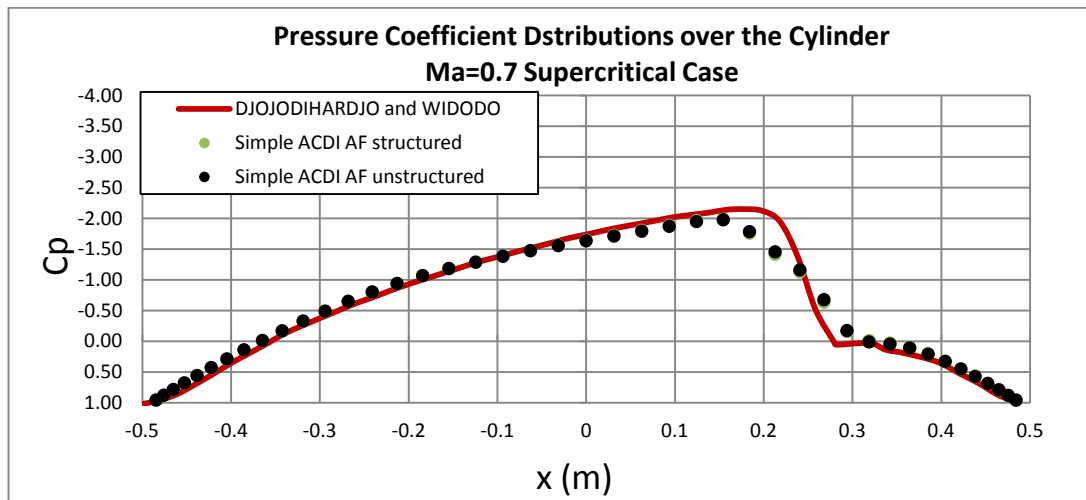


Figure 3.56 Comparison of pressure coefficients with the analytical results adapted from Djojodihardjo and Widodo [30] $Ma_{\infty}=0.7$ (Simple AF ACDI solution)

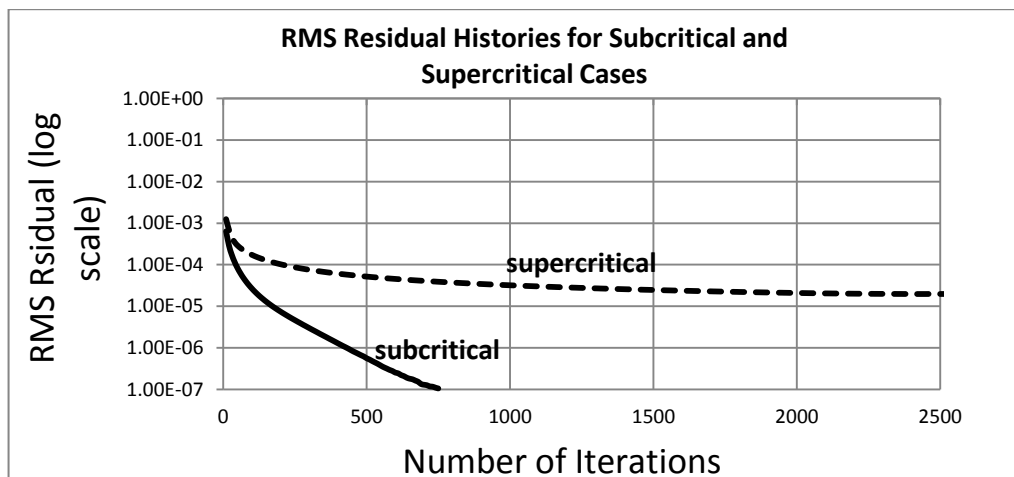


Figure 3.57 Example residual histories for subcritical and supercritical cases using 30x50 structured grid

In Figure 3.58 the Mach contours for a structured grid that is clustered at the shock region is given. Grid used has 30x60 structured cells. The solution is given in order to show that it is possible to obtain steeper weak shocks in case of finer grids.

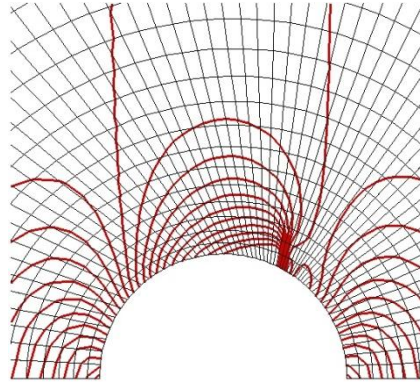


Figure 3.58 Mach contours around the cylinder and the appearance of the 30x60 structured grid that is clustered at shock region - $Ma_\infty=0.7$

3.5 Performance of parallelization

3.5.1 Performance of Parallelization for Finite Difference Cartesian Case Using Proposed Approximate Factorization

1-D unsteady heat conduction problem is solved in a 3-D domain using 100x100x100 Cartesian cells for the observation of the parallelization performance. $\Delta t=0.01$ time step size is used for 50 time steps. The calculations are performed using node based finite difference approach and simple version of the new approximate factorization is used. Three dimensional forms of the formulations (2.7) and (2.8) are;

$$\begin{aligned} \left[1 - \Delta t \frac{\partial^2}{\partial x^2} \right] T_1 &= T_{i,j}^n \\ \left[1 - \Delta t \frac{\partial^2}{\partial y^2} \right] T_2 &= T_{i,j}^n \\ \left[1 - \Delta t \frac{\partial^2}{\partial z^2} \right] T_3 &= T_{i,j}^n \end{aligned} \quad (3.31)$$

And the updated solution is;

$$T_{i,j}^{n+1} = T_1 + T_2 + T_3 - 2T_{i,j}^n \quad (3.32)$$

There is no importance of the sequence of the solution of the sweeps for such a factorization. In standard approaches x, y and z sweep solutions are calculated in order and all sweeps uses the previous sweep solution. For example, in standard ADI approaches firstly, first sweep direction of x sweeps is solved, but the factorization of the current study can begin with z sweep direction which passes through the middle of the domain and following a random sequence for the solution does not change the numerical results.

Thus the scheme lets the programmer to solve all tri-diagonal matrices of a common time step at the same time. This property also exists for Crank-Nicolson like AF ACDI which is used with finite volumes.

The code has run beginning with 1 CPU up to 32 CPU's using the powers of 2 as number of CPU's. CPU times of the solutions are recorded and the speedup ratios are calculated to observe the performance of the parallelization approach.

Firstly, only the sweep solutions have been sent to different processors. Secondly, update part of the code is parallelized as well as the sweeps. These two different applications have been performed in order to see the performance of sending sweep solutions to different processors.

Speedup ratio is the ratio between CPU times of serial and parallel running codes. Maximum speedup of a parallel code is calculated theoretically using the Amdahl's law [31];

$$S(N) = \frac{1}{(1 - P) + \frac{P}{N}} \quad (3.33)$$

Where P is the proportion of the code that is parallelized and N is equal to number of processors used. Amdahl's law is used to plot the speedup ratios of theoretically 90% and 97% parallelized codes. These plots are drawn together with the speedup ratios of numerical experiments in order to obtain an idea about the success of parallelization trials of the current study [32],[33].

It is seen that sending the sweep solutions to different processors results with a performance close to 90% parallelization. Parallelizing other serial parts of the time iteration increases the performance up to 97%. CPU number versus speed up ratio plots of the numerical experiments are given in Figure 3.59, also parallelization efficiency which is the ratio of speedup ratio to the CPU number is given in Figure 3.60.

CPU time's required for the numerical experiments are given in Tables 3.3 and 3.4.

The application gives high performance without using any domain decomposition, but the calculations presented in this part are performed using equi-sized sweep directions. Having different size of sweep directions would probably end up in more overhead since the sweep solutions have to wait for all sweep calculations to be completed to update the current time step results. For different size of sweeps a parallelization optimization might be required. But it can not be considered as a disadvantage, since domain decomposition applications also require an optimum decomposition to obtain decomposed domains which include close numbers of elements and minimized data sharing boundaries.

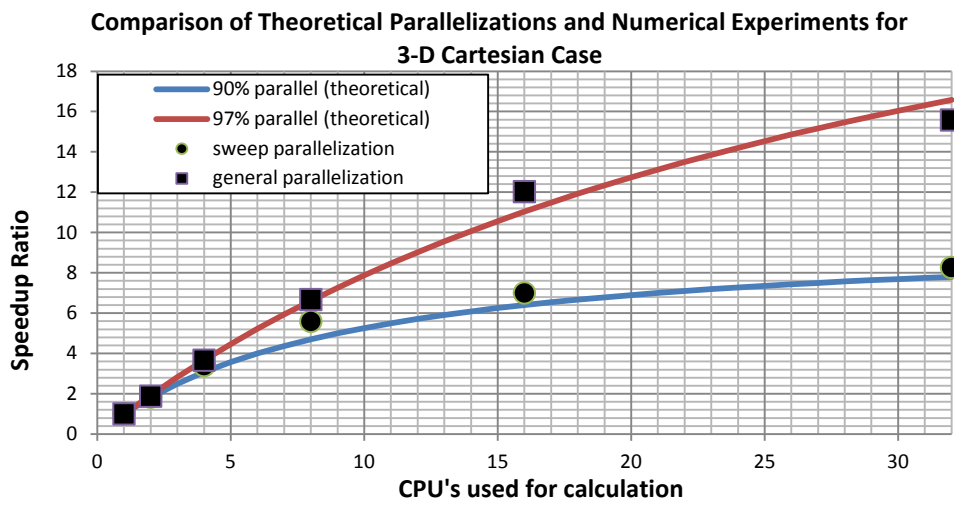


Figure 3.59 Speedup ratios of parallelization trials and comparisons with theoretically 90% and 97% parallelized codes

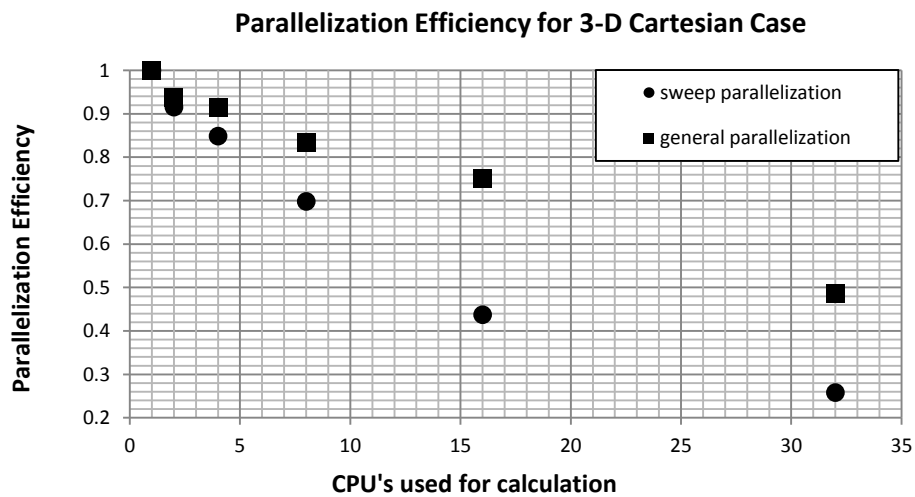


Figure 3.60 Parallelization efficiency for sweep parallelization and general parallelization

Table 3.3 CPU times and speedup ratios for sweep parallelization (50 time iterations)

CPU's	CPU time (s)	speedup ratio
1	23.221	1.000
2	12.684	1.831
4	6.841	3.394
8	4.157	5.586
16	3.318	6.998
32	2.811	8.261

Table 3.4 CPU times and speedup ratios for general parallelization (50 time iterations)

CPU's	CPU TIME (s)	speedup ratio
1	23.086	1.000
2	12.301	1.877
4	6.305	3.661
8	3.461	6.670
16	1.920	12.024
32	1.482	15.577

Non-dimensional temperature contours can be seen in Figure 3.60.

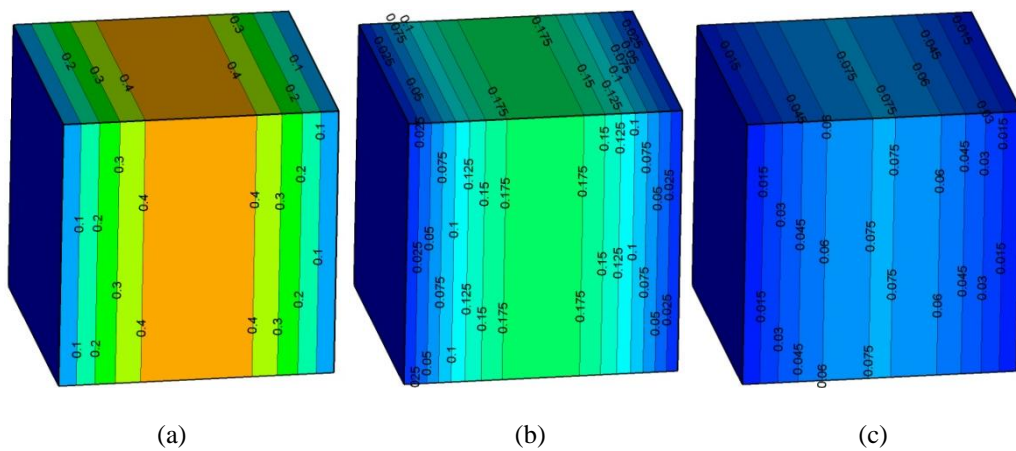


Figure 3.61 Non-dimensional temperature θ contours at (a) $\tau = 0.1$ (b) $\tau = 0.2$ (c) $\tau = 0.3$ for 3-D parallelization case using $\Delta\tau=0.01$

3.5.2 Performance of Parallelization for AF ACDI Using 2-d Quadrilateral Structured and Unstructured grids

A similar study of parallelization of node based 3-D Cartesian case is also carried on for simple AF ACDI method. 2-D structured grid and the unstructured grid used for the numerical tests are shown in Figure 3.602 (a) and 3.62 (b). Unsteady diffusion equation is used and CPU times are represented for 1000 iterations.

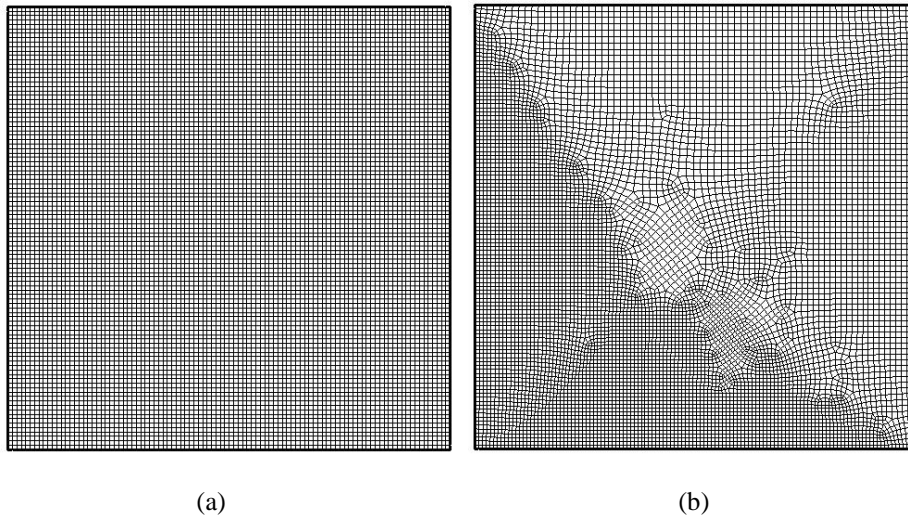


Figure 3.62 (a) 10000 element structured grid and (b) 8823 element unstructured grid used for the parallelization performance tests

The parallelization efficiencies are not as high as the node based case but the parallelization performances of structured and unstructured cases are very close to each other. Indeed having similar performances for structured grids and unstructured grids is an expected result. Although structured case that is shown in this part has cell directions that are equal in length, there exist enough cell directions for the unstructured grid to compensate the overhead of having cell directions in different lengths. The only time loss might exist at the very end of the sweep calculations and this effect can be observed in Figure 3.63. Parallelization efficiency plots for structured and unstructured quadrilateral cases are given in Figure 3.64.

The data structure of finite volume approach of AF ACDI is quite different than the finite difference case. Besides, there are more numerical operations at the update part of the finite volume AF ACDI solution than the finite difference solution. Having more loss of time and poorer parallelization performance than finite difference case is because of these extra operations of the update part of the finite volume AF ACDI.

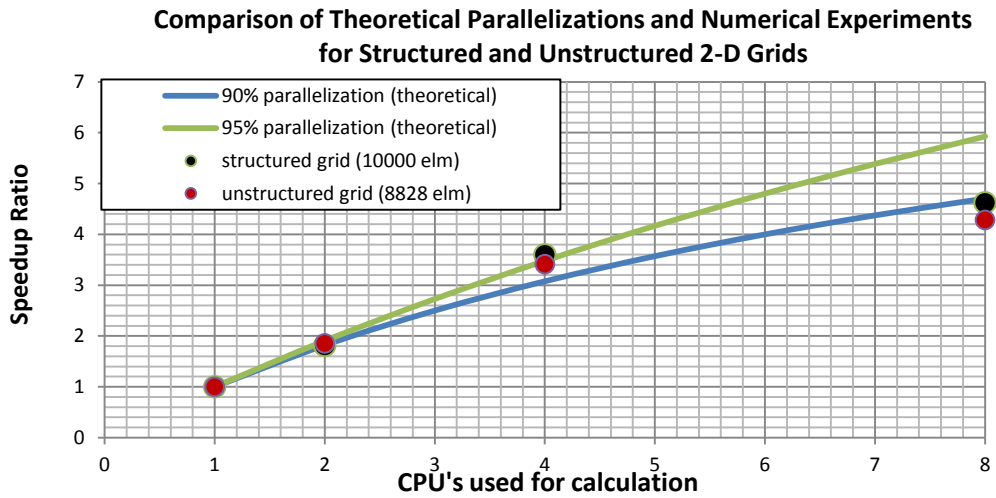


Figure 3.63 Speedup ratios of parallelization trials and comparisons with theoretically 90% and 95% parallelized codes for structured and unstructured grids (1000 time iterations)

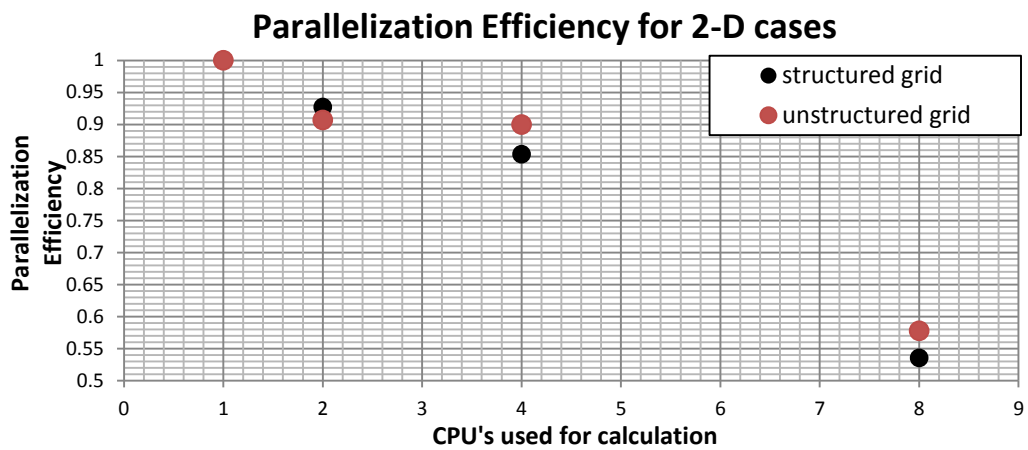


Figure 3.64 Parallelization efficiency for structured and unstructured grids

The CPU times and the speedup ratios for the numerical tests using structured and unstructured grids can be seen in Table 3.5 and Table 3.6.

Table 3.5 CPU times and speedup ratios using structured grid (10000 elements 1000 iterations)

CPU's	CPU TIME (s)	speedup ratio
1	8.633	1.000
2	4.760	1.814
4	2.399	3.599
8	1.868	4.622

Table 3.6 CPU times and speedup ratios using structured grid (8828 elements 1000 iterations)

CPU's	CPU TIME (s)	speedup ratio
1	7.778	1.000
2	4.195	1.854
4	2.279	3.413
8	1.816	4.283

Table 3.7 Selected points for the comparison of sequential and different number of CPU solutions

	X	Y
POINT 1	0.898101786880999	0.834186303294999
POINT 2	0.240029605709000	0.814992029195000
POINT 3	0.476599131213999	0.258333948568999

Table 3.8 Comparison of sequential and different number of CPU solutions at selected points

	POINT 1	POINT 2	POINT 3
sequential	0.149465956179728	0.325120573213070	0.473524291931471
4 CPU's	0.149466016191259	0.325120745813366	0.473525370048431
8 CPU'S	0.149537754060230	0.325273628421354	0.473740594093422

The numerical solutions obtained using sequential code and parallel codes using 4 and 8 CPU's are compared at 3 randomly selected points. The non-dimensional coordinates of these points are given in Table 3.7. The solutions are extracted from the 8823 quadrilateral unstructured grid solution at $\tau = 0.1$ using 10^4 time steps.

Non-dimensional temperature values at these selected points are given in Table 3.8. The contour plots of these three cases are not given since the difference between them are not clear. It can be observed that the largest difference between the solutions of the cases is in the order of 10^{-4} . The parallelization approach does not affect the numerical results in a serious manner.

CHAPTER 4

CONCLUSIONS

The main properties of the variations of the iteration method that offered and examined with this study can be summarized as below:

- The iteration method behaves like an approximation of fully implicit method up to an optimum time step value with a certain quadrilateral grid if the simple version of the Approximate Factorization is used. The order of accuracy decreases with the increased time step size up to this point. It does not matter if the grid is a quadrilateral structured grid or quadrilateral unstructured grid. After this optimum value the error term of the factorization causes unstability problems with unstructured grids.
- The iteration method behaves like a high order explicit method with an enhanced stability character if the Crank-Nicolson discretization is used. More than 50% percent of the discretized terms remain explicit with such a spatial discretization since the node values are added explicitly to the flux calculation operations as well as the explicit terms of the Crank-Nicolson scheme, also Crank-Nicolson AF ACDI gives faster solutions than Runge Kutta Order 4 Method if the CPU time per iteration is taken into account, since it handles only the half of the operations through the cell direction solutions. The stability character is better than the Runge Kutta order 4 method because of having more implicit terms. The order of accuracy of Crank-Nicolson AF ACDI is almost similar to Runge Kutta Order 4 method for both of the structured and unstructured quadrilateral grids.
- Both of the simple AF ACDI and Crank-Nicolson AF ACDI methods give better time accurate results than previous ACDI study [3], Point Gauss Seidel Method and Line Gauss Seidel Method.
- Simple AF ACDI method converges with less number of iterations than Point Gauss Seidel method if the optimum time step size is used. Full Implicit Laasonen method can converge with larger time step sizes and less number of iterations, but if the CPU times of the convergence character plots are inspected, it will be seen that Full Implicit Laasonen method loses its advantage of having better convergence character.
- Point Gauss Seidel and Full Implicit Laasonen methods remain stable with relatively larger time step sizes, but it is a known fact that usage of very large time steps should be avoided with implicit schemes [13] since the accuracy of the solution decreases in a serious manner. The order of accuracy plots for simple AF ACDI shows that the case is the same for simple AF ACDI similarly to implicit or fast implicit schemes.
- It is possible to solve convective problems with relatively larger time step sizes while protecting stability if the simple AF ACDI is used, but using high Courant numbers result with less accurate results. Besides, the solution behaves less diffusive than the exact analytic solution.
- The approximate factorization that used with the ACDI method is easy to parallelize for shared memory applications. The type of the grid does not change the performance of the parallelization seriously. If the grid has enough cell directions, the overhead of having cell directions in different lengths is compensated.
- It is shown that it is possible to increase the order of spatial discretization via using appropriate flux calculation methods [34]. Two different flux calculation approaches applied for the method and it is seen that the stability characteristics of the approach depends on the flux

calculation method. The inverse distance flux calculation ends up in unconditionally stable scheme diffusion problem whereas the stability of the higher order flux calculation depends on the time step size and the cell area, also dependency to grid size is lower for the low order flux calculation but it has a higher order of error.

- In case of having even number of edges, the current solution procedure includes no randomness. It was previously shown by Çete [3] that the ‘cell directions’ were unique for quadrilateral grids, but sequence selection of cell direction solutions were creating randomness for previous ACIDI studies. Solution procedure also became unique as well as the cell directions with the current study.
- Usage of triangular elements disturbs the implicitness of the scheme and the proposed scheme is no more an approximation of a fully implicit schemes. The solution bands are broken inside the domain and poorer time accurate results are obtained, since the broken cell directions block the flow of information. Besides, the solution method loses its uniqueness if triangular cells exist inside the domain. The solution band that passes through a triangular element becomes programmer defined, but the numerical formulation is still valid since one of the nodes of the cell is assumed as a zero length edge.
- The advection-diffusion equation is solved using the advective term explicitly and diffusive terms are approximately factorized for the current study. The problem of continuous point source release is solved using Fully Implicit, Runge Kutta order 4 and Simple AF ACIDI methods for different Courant numbers and the resulting mass concentration contours are given in Appendix D. All the solutions are obtained using the unsteady form of the equation and the contours have been plotted for converged results. It can be seen that using the advective term explicitly results in decrease of accuracy for Courant numbers greater than about 1 and stability problems for Courant numbers greater than about 4. The advective term is not included into the factorized form of the equation since the order of the additional term that arises from the factorization would be relatively high for the integrated advection-diffusion equation. Two different options could be considered for the solution of the problem. First of them is the usage of a semi-Lagrangian approach [38] where factorized form of the diffusion terms are used. The second choice might be an approach that a hybrid of proposed approximate factorization and Point Gauss Seidel method. It would be possible to obtain an increased stability character if the cell center values are updated before the calculation of another cell center value inside the domain, but such an approach would harm the uniqueness of the numerical solution.

CHAPTER 5

FUTURE WORK

The current study on Alternating Cell Directions Implicit methods show that the performance of the approach is highly promising for numerical fluid dynamics applications. One of the most important properties of the proposed iteration scheme is its capacity to unify the solution algorithm for different type of grids. The future work on AF ACDI study can be summarized as below using the guidance of the current thesis study.

- The cell directions concept might also be defined for quad-tree grids and grid adaptive solution approaches may be generated.
- Cell directions are easy to define for 3-D structured grids, but it is not that easy for tetragonal elements. An updated definition for tetragonal elements may be required for a well defined solution on 3-D unstructured grids.
- The method also should be tested on the solution of more advanced fluid dynamics applications like Euler or Navier-Stokes solutions. As previously stated previous ACDI method [3] has been tried on Incompressible Navier-Stokes Solver by Bas [12] and it is shown that the approach is superior to Point Gauss Seidel Iteration Method as a fast implicit method.
- Both of the simple AF ACDI and Crank-Nicolson AF ACDI methods should be tried with higher order flux calculation methods without using explicit terms in the flux calculation operations. Such application most probably would highlight the success of the numerical approach in a better way.
- The usage of approximate factorization is very widespread with full potential and transonic small disturbance equations [27][35][36][37]. The proposed method is used to solve the full potential equation for the transonic cases. Usage of Simple AF gives very convergent and highly stable solutions. The solution of full potential equation should be repeated with simple AF ACDI but relatively more advanced flux calculation approach should be used for the solution procedure, or Crank Nicolson AF ACDI method solution should be repeated with another flux calculation method without explicit node terms. These solution choices might be compared with the AF methods for full potential equation in the literature [27][35][37] and then the convergence behavior might be tested on unstructured quadrilateral grids.
- It is shown that the method is very suitable for shared memory parallelization applications. The numerical tests that performed to show the parallelization performance of the method are carried on without any parallelization optimization. At the very end of the sweep solutions some of the solution threads wait the others since they can not find a sweep to solve and this situation results with an overhead. Parallelization optimization study may be carried on as a future work.
- For the cases where shared memory parallelization is not possible in whole domain, it is possible to combine block decomposition parallelization and sweep parallelization of the current study. It will be more advantageous to use less decomposed blocks since the data sharing boundaries will be shorter for such an hybrid parallelization.

REFERENCES

- [1] Caughey, D.A. and Hafez, M.M., “Frontiers of Computational Fluid Dynamics”, John Willey & Sons, Newyork, 1994.
- [2] Dimitri, J.M., “Unstructured Mesh Discretization and Solvers for Computational Aerodynamics”, AIAA Computational Fluid Dynamics Conference, Vol. 18, pp. 1-28, 2007.
- [3] Çete, A.R., “Alternating Cell Directions Implicit Method”, PhD Thesis, Istanbul Technical University, 2006.
- [4] Venkatakrishnan, V., “Implicit Schemes and Parallel Computing in Unstructured Grid CFD”, NASA ICASE, Lecture Notes Prepared for 26th Computational Fluid Dynamics Lecture Series Program of VKI, 1995.
- [5] Gibbons, A., “Algorithmic graph theory”, Cambridge University Press, New York, NY, 1985.
- [6] Hassan, O., Morgan, K. and Peraire, J. “An adaptive implicit/explicit finite element scheme for compressible high speed flows” AIAA Paper 89-0363, 1989.
- [7] Meyer, G.H., “An Alternating Directions Method for Multi-Dimensional Parabolic Free Surface Problems”, International Journal for Numerical Methods in Engineering, Vol. 11, pp. 741-752, 1977.
- [8] Kellog, R.B., “A Nonlinear Alternating Direction Method”, Mathematics of Computation, Vol. 9, pp. 23-27, 1969.
- [9] Vries, H.B.D., “A Comparative Study of ADI Splitting Methods Methods for Parabolic Equations in Two Space Dimensions”, Journal of Computational and Applied Mathematics, Vol. 10, pp. 179-193, 1984.
- [10] Çete, A.R. and Kaynak, U., “A New Approximate Factorization Method Suitable for Structured and Unstructured Grids”, AIAA Paper 2006-3789, 9th AIAA/ASME Joint Thermo physics and Heat Tranfer Conference, 2006.
- [11] Abrashin, V.N., Dzyuba I.A., “An Alternating Direction Method for Solving Multidimensional Problems of Mathematical Physics in Domain with Curvilinear Boundary”, Differential Equations, Vol. 30, pp. 1082-1087, 1994.
- [12] Bas, O., “Development of an Incompressible Navier-Stokes Solver with Alternating Cell Direction Implicit Method on Structured and Unstructured Quadrilateral Grids”, MSc Thesis, Middle East Technical University, 2007.
- [13] Hoffman, K.A., Chiang, S.,J., “ Computational Fluid Dynamics Volume 1”, A Publication of Engineering Education, Kansas, USA, August 2000.
- [14] Stone, C.P., Duque, E.P.N., Zhang, Y., Car, D., Owens, J.D. and Davis, R.L., “GPGPU Parallel Algorithms for Structured Grid CFD Codes”, 20th Computational Fluid Dynamics Conference, AIAA 2011-3221, 2011.
- [15] Ramos, J.I., “Linearly Implicit Approximate Factorization Exponential Methods for Multidimensional Reaction-Diffusion Equations”, Applied Mathematics and Computations, Vol. 174, pp. 1609-1633, 2006.
- [16] Sasena, J.M., “Optimization of Computer Simulations via Smoothing Splines and Krigging Metamodels”, MSc Thesis, University of Michigan, 1998.
- [17] Pacheo, P.S., “A User’s Guide to MPI”, Department of Mathematics, University of San Fransisco, 1998.
- [18] Barney, B., Lawrence Livermore National Laboratory
<https://computing.llnl.gov/tutorials/OpenMP>
- [19] Mathey, F. Kloos, P. and Blaise, P., “Openmp optimisation of a parallel mpi cfd code”, 2nd European Workshop on OpenMP (EWOMP2000), September 2000.
- [20] Decker, H.N., Naik, V.K. and Nicoules, M., “Parallelization of Implicit Finite Difference Schemes in Computational Fluid Dynamics”, Institute for Computer Applications in Science and Engineering, NASA Langley Research Center, 1990.
- [21] Koren, B., “A Robust Upwind Discretization Method for Advection, Diffusion and Source Terms”, CWI, 1993.
- [22] Gooch, C.O., Altena, M.V., “A High-Order-Accurate Unstructured Mesh Finite-Volume Scheme for the Advection– Diffusion Equation”, University of British Columbia, Department of Mechanical Engineering, 2002.

- [23] Ivan, L., Groth, C.P.T., “High-Order Solution-Adaptive Central Essentially Non-Oscillatory (CENO) Method for Viscous Flows”, 49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, January 2011.
- [24] Neel, R.E., “Advances in Computational Fluid Dynamics: Turbulent Separated Flows and Transonic Potential Flows”, PhD Thesis, Virginia Polytechnic Institute and State University, 1997.
- [25] Socolofsky, S.A., Jirka, G.H., “Environmental Fluid Mechanics Part I: Mass Transfer and Diffusion Engineering-Lectures”, Institut für Hydromechanik, Universität Karlsruhe, 2002.
- [26] Boghrati, M., Moghiman, M., “New methods for calculating the inlet hydrodynamic and thermal length in a laminar nanofluid flow by applying entropy generation theory”, ECOS, Navi Sad, July 2011.
- [27] Holst, T.L., Ballhaus, W.F., “Conservative Implicit Schemes for the Full Potential Equation Applied to Transonic Flows”, NASA/Ames Research Center, March 1978.
- [28] Soulis, J.V., “A finite volume method for two dimensional transonic potential flow through turbomachinery blade rows”, Whittle Laboratory, Department of Engineering, University of Cambridge, September 1983.
- [29] Saied, B.S., Alireza, D.D., “An Unstructured Grid Generation Approach for Inviscid Flow Solutions”, WSEAS TRANSACTIONS on FLUID MECHANICS, January 2009.
- [30] Djojodihardjo, H., Widodo, A.F., “Development of a Simple and Fast Computational Routine to Solve the Full Potential Equation of the Transonic Axi-symmetric Flow”, 24th International Congress of the Aeronautical Sciences, 2004.
- [31] Marty, M.R., “Amdahl’s Law in the Multicore Era”, IEEE Computer, 2008.
- [32] Jaswinder, P.S., Weber, W.D. and Gupta, A., “SPLASH: Stanford Parallel Applications for Shared Memory”, Computer Systems Laboratory, Technical Report, 1991.
- [33] Smith, L. and Bull, M., “Development of Mixed Mode MPI / OpenMP Applications”, Workshop on OpenMP Applications and Tools (WOMPAT 2000), 2000.
- [34] Cai, Z., Douglas, J. and Park, M., “Development and Analysis of Higher Order Finite Volume Methods over Rectangles for Elliptic Equations”, Advances in Computational Mathematics, Vol. 19, pp. 3-33, 2003.
- [35] Ballhaus, W.F., Jameson, A., and Albert, J. “Implicit Approximate-Factorization Schemes for the Efficient Solution of Steady Transonic Flow Problems”, NASA/Ames Research Center, January 1977.
- [36] Batina, J.T., “A Finite-Difference Approximate-Factorization Algorithm for Solution of the Unsteady Transonic Small-Disturbance Equation”, Langley Research Center, January 1992.
- [37] Vadyak, J., Atta, E.H., “Approximate Factorization Algorithm for Three-Dimensional Transonic Nacelle/Inlet Flowfield Computations”, 19th Joint Propulsion Conference, June 1983
- [38] Spiegelman, M., Katz, R.F., “A Semi-Lagrangian Crank-Nicolson Algorithm for the Numerical Solution of Advection-Diffusion Problems”, Geochemistry, Geophysics, Geosystems, Vol. 7, Issue 4, April 2006.

APPENDIX A

LIST OF THE TEST CASES

1. Mean Error Comparison for Validation of Proposed Approximate Factorization:

- Unsteady diffusion equation - 20x20 structured grid - Finite difference - dt 0.0001 - simple AF
- Unsteady diffusion equation - 20x20 structured grid - Finite difference - dt 0.0001 - Crank-Nicolson AF
- Unsteady diffusion equation - 20x20 structured grid - Finite difference - dt 0.0001 - Crank-Nicolson like AF
- Unsteady diffusion equation - 40x40 structured grid - Finite difference - dt 0.0001 - simple AF
- Unsteady diffusion equation - 40x40 structured grid - Finite difference - dt 0.0001 - Crank-Nicolson AF
- Unsteady diffusion equation - 40x40 structured grid - Finite difference - dt 0.0001 - Crank-Nicolson like AF

2. Comparison of Time Accurate Results:

- Unsteady diffusion equation - 20x20 structured grid - Trapezoidal rule flux calculation - dt 0.0001 - Runge Kutta Order 4 Method
- Unsteady diffusion equation - 20x20 structured grid - Trapezoidal rule flux calculation - dt 0.0001 - Full Implicit Laasonen Method (iterative)
- Unsteady diffusion equation - 20x20 structured grid - Trapezoidal rule flux calculation - dt 0.0001 - Point Gauss Seidel Iteration Method
- Unsteady diffusion equation - 20x20 structured grid - Trapezoidal rule flux calculation - dt 0.0001 - Line Gauss Seidel Iteration Method
- Unsteady diffusion equation - 20x20 structured grid - Trapezoidal rule flux calculation - dt 0.0001 - ACDI Method
- Unsteady diffusion equation - 20x20 structured grid - Trapezoidal rule flux calculation - dt 0.0001 - Simple AF ACDI Method
- Unsteady diffusion equation - 20x20 structured grid - Trapezoidal rule flux calculation - dt 0.0001 - Crank-Nicolson AF ACDI Method

- Unsteady diffusion equation - 297 element unstructured grid - Trapezoidal rule flux calculation - dt 0.0001 - Runge Kutta Order 4 Method
- Unsteady diffusion equation - 297 element unstructured grid - Trapezoidal rule flux calculation - dt 0.0001 - Full Implicit Laasonen Method (iterative)
- Unsteady diffusion equation - 297 element unstructured grid - Trapezoidal rule flux calculation - dt 0.0001 - Point Gauss Seidel Iteration Method
- Unsteady diffusion equation - 297 element unstructured grid - Trapezoidal rule flux calculation - dt 0.0001 - ACDI Method
- Unsteady diffusion equation - 297 element unstructured grid - Trapezoidal rule flux calculation - dt 0.0001 - Simple AF ACDI Method
- Unsteady diffusion equation - 297 element unstructured grid - Trapezoidal rule flux calculation - dt 0.0001 - Crank-Nicolson AF ACDI Method

3. Comparison of Mean Errors Using Different Grid Types:

- Unsteady diffusion equation - 45 element structured grid - Trapezoidal rule flux calculation - dt 0.0001 - Simple AF ACDI Method

- Unsteady diffusion equation - 46 element triangular grid - Trapezoidal rule flux calculation - dt 0.0001 - Simple AF ACADI Method
- Unsteady diffusion equation - 43 element quadrilateral grid - Trapezoidal rule flux calculation - dt 0.0001 - Simple AF ACADI Method
- Unsteady diffusion equation - 44 element hybrid polygonal grid - Trapezoidal rule flux calculation - dt 0.0001 - Simple AF ACADI Method

4. Comparison of Mean Errors for Different Flux Calculation Methods:

- Unsteady diffusion equation - 297 element unstructured grid - Trapezoidal rule flux calculation - dt 0.0001 - Simple AF ACADI Method
- Unsteady diffusion equation - 297 element unstructured grid - Inverse distance weighting flux calculation - dt 0.0001 - Simple AF ACADI Method
- Unsteady diffusion equation - 1283 element unstructured grid - Trapezoidal rule flux calculation - dt 0.0001 - Simple AF ACADI Method
- Unsteady diffusion equation - 1283 element unstructured grid - Inverse distance weighting flux calculation - dt 0.0001 - Simple AF ACADI Method

5. Comparison of Convergence Characters:

- Laplace equation - 500 element structured grid - Trapezoidal rule flux calculation - Various time steps - Simple AF ACADI Method
- Laplace equation - 500 element structured grid - Trapezoidal rule flux calculation - Various time steps - Point Gauss Seidel Method
- Laplace equation - 500 element structured grid - Trapezoidal rule flux calculation - Various time steps - Full Implicit Laasonen Method (iterative)
- Laplace equation - 500 element structured grid - Trapezoidal rule flux calculation - Various time steps - Runge Kutta Order 4 Method
- Laplace equation - 789 element unstructured grid - Trapezoidal rule flux calculation - Various time steps - Simple AF ACADI Method
- Laplace equation - 789 element unstructured grid - Trapezoidal rule flux calculation - Various time steps - Point Gauss Seidel Method
- Laplace equation - 789 element unstructured grid - Trapezoidal rule flux calculation - Various time steps - Full Implicit Laasonen Method (iterative)
- Laplace equation - 789 element unstructured grid - Trapezoidal rule flux calculation - Various time steps - Runge Kutta Order 4 Method

6. Comparison of Time Accurate Results:

- Advection-Diffusion equation - 512 element structured grid - Trapezoidal rule flux calculation - dt 0.01 - Simple AF ACADI Method
- Advection-Diffusion equation - 512 element structured grid - Trapezoidal rule flux calculation - dt 0.01 - Runge Kutta Order 4 Method
- Advection-Diffusion equation - 512 element structured grid - Trapezoidal rule flux calculation - dt 0.01 - Full Implicit Laasonen Method (iterative)
- Advection-Diffusion equation - 512 element structured grid - Trapezoidal rule flux calculation - dt 0.25 - Simple AF ACADI Method
- Advection-Diffusion equation - 512 element structured grid - Trapezoidal rule flux calculation - dt 0.25 - Runge Kutta Order 4 Method
- Advection-Diffusion equation - 512 element structured grid - Trapezoidal rule flux calculation - dt 0.25 - Full Implicit Laasonen Method (iterative)
- Advection-Diffusion equation - 624 element unstructured grid - Trapezoidal rule flux calculation - dt 0.01 - Simple AF ACADI Method

- Advection-Diffusion equation - 624 element unstructured grid - Trapezoidal rule flux calculation - dt 0.01 - Runge Kutta Order 4 Method
- Advection-Diffusion equation - 624 element unstructured grid - Trapezoidal rule flux calculation - dt 0.01 - Full Implicit Laasonen Method (iterative)
- Advection-Diffusion equation - 624 element unstructured grid - Trapezoidal rule flux calculation - dt 0.25 - Simple AF ACADI Method
- Advection-Diffusion equation - 624 element unstructured grid - Trapezoidal rule flux calculation - dt 0.25 - Runge Kutta Order 4 Method
- Advection-Diffusion equation - 624 element unstructured grid - Trapezoidal rule flux calculation - dt 0.25 - Full Implicit Laasonen Method (iterative)

7. Solution of Full Potential Equation:

- Full Potential Equation - 1500 element structured grid - Trapezoidal rule flux calculation - Ma 0.25 - Crank-Nicolson AF ACADI
- Full Potential Equation - 720 element unstructured grid - Trapezoidal rule flux calculation - Ma 0.25 - Crank-Nicolson AF ACADI
- Full Potential Equation - 1500 element structured grid - Trapezoidal rule flux calculation - Ma 0.7 - Crank-Nicolson AF ACADI
- Full Potential Equation - 720 element unstructured grid - Trapezoidal rule flux calculation - Ma 0.7 - Crank-Nicolson AF ACADI
- Full Potential Equation - 1500 element structured grid - Trapezoidal rule flux calculation - Ma 0.25 - Simple AF ACADI
- Full Potential Equation - 720 element unstructured grid - Trapezoidal rule flux calculation - Ma 0.25 - Simple AF ACADI
- Full Potential Equation - 1500 element structured grid - Trapezoidal rule flux calculation - Ma 0.7 - Simple AF ACADI
- Full Potential Equation - 720 element unstructured grid - Trapezoidal rule flux calculation - Ma 0.7 - Simple AF ACADI
- Full Potential Equation - 1800 element structured grid - Trapezoidal rule flux calculation - Ma 0.7 - Simple AF ACADI

8. Performance of Parallelization:

- Unsteady Diffusion equation - 100x100x100 element structured grid - Finite difference - dt 0.01 - Simple AF
- Unsteady Diffusion equation - 100x100 element structured grid - Trapezoidal rule flux calculation - dt 0.00001 - Simple AF ACADI
- Unsteady Diffusion equation - 8823 element unstructured grid - Trapezoidal rule flux calculation - dt 0.00001 - Simple AF ACADI

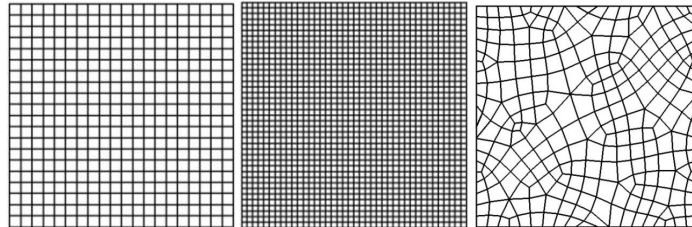
9. Numerical Stability Tests and Comparisons with Full Implicit and Full Explicit Schemes:

- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 2 - Simple AF ACADI
- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 3 - Simple AF ACADI
- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 4 - Simple AF ACADI
- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 5 - Simple AF ACADI
- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 0.25 - Simple AF ACADI
- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 0.5 - Simple AF ACADI

- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 0.25 - Runge Kutta Order 4 Method
- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 0.5 - Runge Kutta Order 4 Method
- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 2 - Full Implicit Laasonen Method
- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 3 - Full Implicit Laasonen Method
- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 4 - Full Implicit Laasonen Method
- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 5 - Full Implicit Laasonen Method
- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 0.25 - Full Implicit Laasonen Method
- Advection-Diffusion Equation - 654 element unstructured grid - Trapezoidal rule Flux Calculation - Co 0.5 - Full Implicit Laasonen Method

APPENDIX B

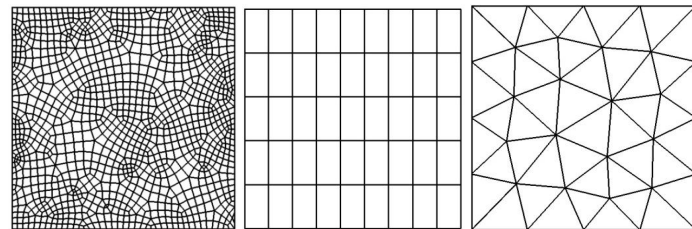
GRIDS USED FOR THE TEST CASES



(a)

(b)

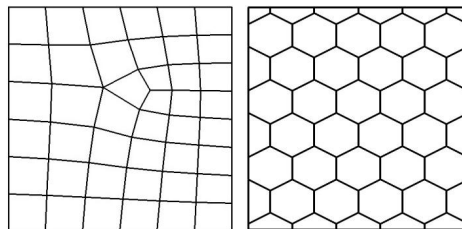
(c)



(d)

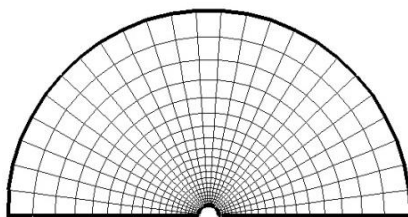
(e)

(f)

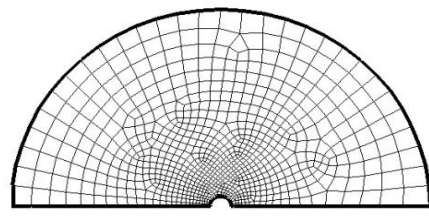


(g)

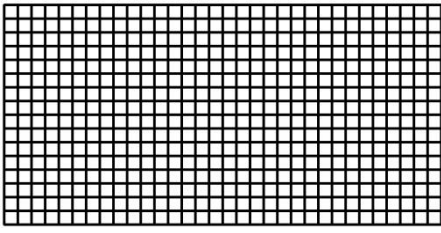
(h)



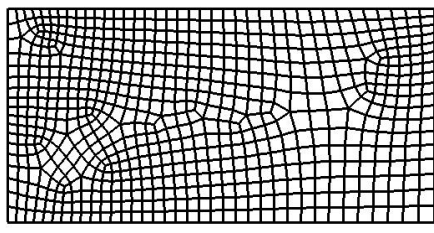
(i)



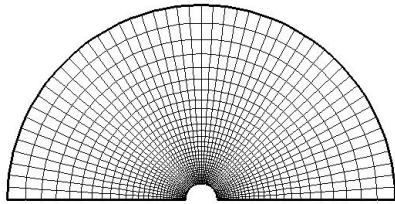
(j)



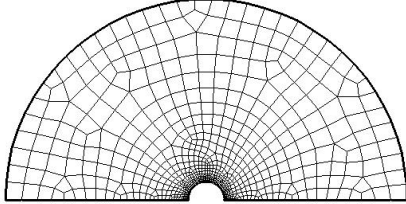
(k)



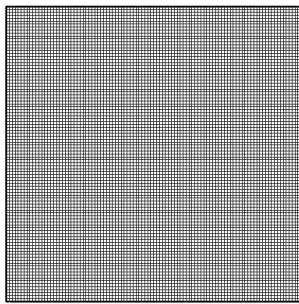
(l)



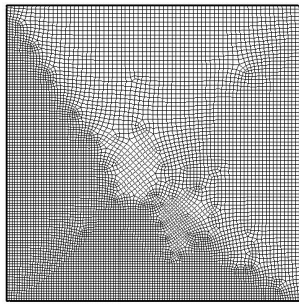
(m)



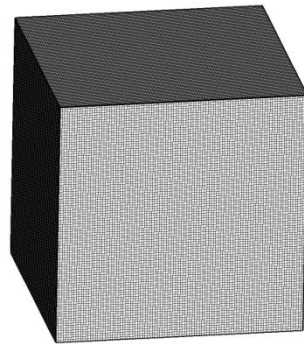
(n)



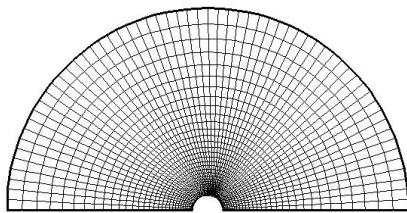
(o)



(p)



(q)



(r)

a	20x20 structured grid	j	789 element quadrilateral uns. grid
b	40x40 structured grid	k	16x32 structured grid
c	297 element quadrilateral uns. grid	l	624 element quadrilateral uns. grid
d	1283 element quadrilateral uns. grid	m	30x50 structured grid
e	9x5 structured grid	n	720 element quadrilateral uns. grid
f	46 element triangular uns. grid	o	100x100 structured grid
g	43 element quadrilateral uns. grid	p	4589 element quadrilateral uns. grid
h	44 element hybrid polygonal grid	q	100x100x100 structured grid
i	25x20 structured grid	r	30x60 structured grid

APPENDIX C

EQUATIONS USED FOR THE NUMERICAL TESTS AND FACTORIZATIONS

All factorizations shown for quadrilateral elements.

UNSTEADY DIFFUSION EQUATION

$$\frac{\partial T}{\partial t} - \frac{\partial^2 T}{\partial x^2} - \frac{\partial^2 T}{\partial y^2} = 0$$

Simple Approximate Factorization

$$\left(1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1 - \frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3\right) \left(1 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n}\right)_2 - \frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n}\right)_4\right) T = T^n$$

$$\left(1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1 - \frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3\right) T_1 = T^n$$

$$\left(1 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n}\right)_2 - \frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n}\right)_4\right) T_2 = T^n$$

Crank Nicolson like Approximate Factorization ($\theta=0.5$)

$$\left[1 - \left(\frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1\right) - \left(\frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3\right)\right] \left[1 - \left(\frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n}\right)_2\right) - \left(\frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n}\right)_4\right)\right] \Delta T$$

$$= (1 - \theta) \left(\sum_{i=1}^N \frac{\Delta t}{A} \Delta s_i \left(\frac{\partial T}{\partial n}\right)_i\right)^n$$

$$\left[1 - \left(\frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1\right) - \left(\frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3\right)\right] \Delta T_1 = (1 - \theta) \left(\sum_{i=1}^N \frac{\Delta t}{A} \Delta s_i \left(\frac{\partial T}{\partial n}\right)_i\right)^n$$

$$\left[1 - \left(\frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n}\right)_2\right) - \left(\frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n}\right)_4\right)\right] \Delta T_2 = (1 - \theta) \left(\sum_{i=1}^N \frac{\Delta t}{A} \Delta s_i \left(\frac{\partial T}{\partial n}\right)_i\right)^n$$

LAPLACE EQUATION

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = 0$$

Simple Approximate Factorization

$$\left(1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1 - \frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3\right)$$

$$\left(1 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n}\right)_2 - \frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n}\right)_4\right) \psi = \psi^n$$

$$\left(1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1 - \frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3\right) \psi_1 = \psi^n$$

$$\left(1 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n}\right)_2 - \frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n}\right)_4\right) \psi_2 = \psi^n$$

ADVECTION-DIFFUSION EQUATION

$$\frac{\partial c}{\partial t} - D_x \frac{\partial^2 c}{\partial x^2} - D_y \frac{\partial^2 c}{\partial y^2} + u \frac{\partial c}{\partial x} = 0$$

Simple Approximate Factorization

$$\left(1 - D \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1 - D \frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3\right)$$

$$\left(1 - D \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n}\right)_2 - D \frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n}\right)_4\right) c^{n+1}$$

$$= c^n - \left(\frac{\Delta t}{A} \sum_{i=1}^N u c n_x \Delta s_i\right)^n$$

$$\left(1 - \frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1 - \frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3\right) c_1 = c^n - \left(\frac{\Delta t}{A} \sum_{i=1}^N u c n_x \Delta s_i\right)^n$$

$$\left(1 - \frac{\Delta t}{A} \Delta s_2 \left(\frac{\partial}{\partial n}\right)_2 - \frac{\Delta t}{A} \Delta s_4 \left(\frac{\partial}{\partial n}\right)_4\right) c_2 = c^n - \left(\frac{\Delta t}{A} \sum_{i=1}^N u c n_x \Delta s_i\right)^n$$

FULL POTENTIAL EQUATION

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial x} = 0$$

Simple Approximate Factorization

$$\left(1 - \frac{\Delta t}{A} \Delta s_1 \rho_1 \left(\frac{\partial}{\partial n}\right)_1 - \frac{\Delta t}{A} \Delta s_3 \rho_3 \left(\frac{\partial}{\partial n}\right)_3\right) \left(1 - \frac{\Delta t}{A} \Delta s_2 \rho_2 \left(\frac{\partial}{\partial n}\right)_2 - \frac{\Delta t}{A} \Delta s_4 \rho_4 \left(\frac{\partial}{\partial n}\right)_4\right) \varphi = \varphi^n$$

$$\left(1 - \frac{\Delta t}{A} \Delta s_1 \rho_1 \left(\frac{\partial}{\partial n}\right)_1 - \frac{\Delta t}{A} \Delta s_3 \rho_3 \left(\frac{\partial}{\partial n}\right)_3\right) \varphi_1 = \varphi^n$$

$$\left(1 - \frac{\Delta t}{A} \Delta s_2 \rho_2 \left(\frac{\partial}{\partial n}\right)_2 - \frac{\Delta t}{A} \Delta s_4 \rho_4 \left(\frac{\partial}{\partial n}\right)_4\right) \varphi_2 = \varphi^n$$

Crank Nicolson like Approximate Factorization ($\theta=0.5$)

$$\left[1 - \left(\frac{\Delta t}{A} \Delta s_1 \rho_1 \left(\frac{\partial}{\partial n}\right)_1\right) - \left(\frac{\Delta t}{A} \Delta s_3 \rho_3 \left(\frac{\partial}{\partial n}\right)_3\right)\right] \left[1 - \left(\frac{\Delta t}{A} \Delta s_2 \rho_2 \left(\frac{\partial}{\partial n}\right)_2\right) - \left(\frac{\Delta t}{A} \Delta s_4 \rho_4 \left(\frac{\partial}{\partial n}\right)_4\right)\right] \Delta \varphi$$

$$= (1 - \theta) \left(\sum_{i=1}^N \frac{\Delta t}{A} \Delta s_i \rho_i \left(\frac{\partial \varphi}{\partial n}\right)_i\right)^n$$

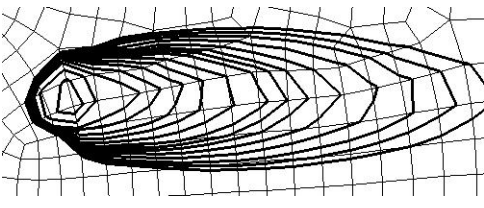
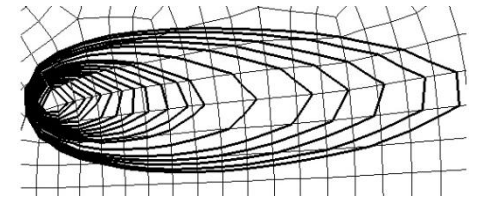
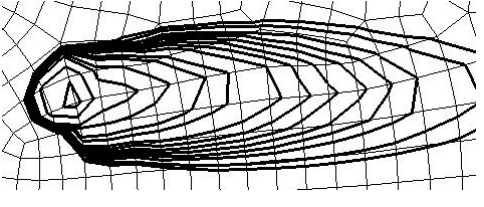
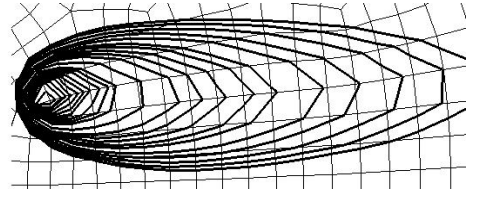
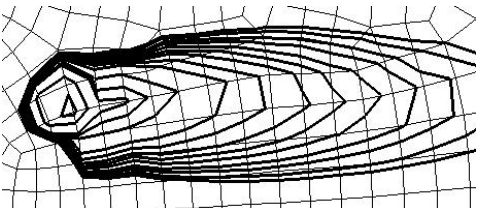
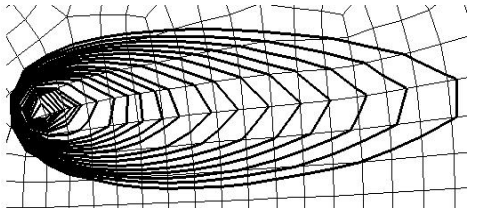
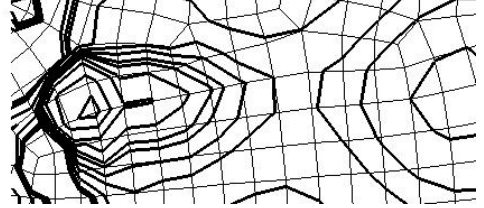
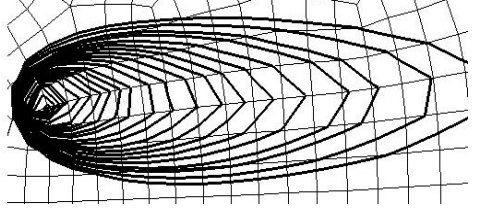
$$\left[1 - \rho_1 \left(\frac{\Delta t}{A} \Delta s_1 \left(\frac{\partial}{\partial n}\right)_1\right) - \rho_3 \left(\frac{\Delta t}{A} \Delta s_3 \left(\frac{\partial}{\partial n}\right)_3\right)\right] \Delta \varphi_1 = (1 - \theta) \left(\sum_{i=1}^N \frac{\Delta t}{A} \Delta s_i \rho_i \left(\frac{\partial \varphi}{\partial n}\right)_i\right)^n$$

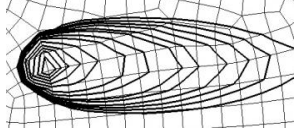
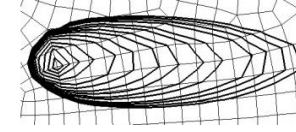
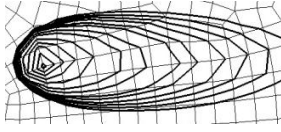
$$\left[1 - \left(\frac{\Delta t}{A} \Delta s_2 \rho_2 \left(\frac{\partial}{\partial n}\right)_2\right) - \left(\frac{\Delta t}{A} \Delta s_4 \rho_4 \left(\frac{\partial}{\partial n}\right)_4\right)\right] \Delta \varphi_2 = (1 - \theta) \left(\sum_{i=1}^N \frac{\Delta t}{A} \Delta s_i \rho_i \left(\frac{\partial \varphi}{\partial n}\right)_i\right)^n$$

APPENDIX D

NUMERICAL STABILITY TESTS USING ADVECTION-DIFFUSION EQUATION FOR CONTINUOUS POINT SOURCE PROBLEM

Source cell concentration – 0.01 kg/m^3
 Maximum local Pecklet number-1.714
 Diffusion coefficient-0.07
 Advection coefficient-0.1
 Solutions are obtained for RMS residual value of 10^{-5}

Courant Number	ACDI SIMPLE AF	FULL IMPLICIT (iterative)
2		
3		
4		
5		

Courant Number	ACDI SIMPLE AF	FULL IMPLICIT (iterative)	RK4
0.25			
0.5	