CONVERGENCE PERFORMANCE OF THE APPROXIMATE FACTORIZATION METHODS WITH MULTI-BLOCK IMPLICIT BOUNDARY CONDITIONS AT HYPERSONIC SPEEDS

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

BY

MELİKŞAH KOCA

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

SEPTEMBER 2022

Approval of the thesis:

CONVERGENCE PERFORMANCE OF THE APPROXIMATE FACTORIZATION METHODS WITH MULTI-BLOCK IMPLICIT BOUNDARY CONDITIONS AT HYPERSONIC SPEEDS

submitted by **MELİKŞAH KOCA** in partial fulfillment of the requirements for the degree of **Master of Science** i**n Aerospace Engineering, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Serkan Özgen
Head of the Department, **Aerospace Engineering** _____

Prof. Dr. Sinan Eyi
Supervisor, **Aerospace Engineering, METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Nilay Sezer Uzol
Aerospace Engineering, METU _____

Prof. Dr. Sinan Eyi
Aerospace Engineering, METU _____

Asst. Prof. Dr. Özge Başkan Perçin
Aerospace Engineering, METU _____

Asst. Prof. Dr. Özgür Uğraş Baran
Mechanical Engineering, METU _____

Asst. Prof. Dr. Sıtkı Uslu
Mechanical Engineering, TOBB ETU _____

Date: 02.09.2022

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: Melikşah Koca

Signature:

ABSTRACT

CONVERGENCE PERFORMANCE OF THE APPROXIMATE FACTORIZATION METHODS WITH MULTI-BLOCK IMPLICIT BOUNDARY CONDITIONS AT HYPERSONIC SPEEDS

Koca, Melikşah
Master of Science, Aerospace Engineering
Supervisor: Prof. Dr. Sinan Eyi
September 2022, 105 pages

This thesis study presents convergence characteristics of the implicit approximate factorization methods at hypersonic flow conditions and with 2-dimensional and 3-dimensional geometries. The efficiency of the implicit boundary conditions at block interfaces for the multi-block grids is investigated for different approximate factorization methods. Standard Alternating Direction Implicit (ADI) method, Diagonal Dominant Alternating Direction Implicit method (DDADI) with and without Huang's sub-iteration correction, Spatially Factored with Diagonal Time terms (SFDT) method which CFL3D open-source code uses as an approximate factorization method and Diagonal Dominant Spatialy Factored with Diagonal Time terms (DDSFDT) method which is developed for this study with and without Huang's sub-iteration correction are used to obtain tridiagonal matrices on the left-hand side while solving compressible Reynolds-averaged Navier-Stokes equations. Residual histories and total run time of the analyses are compared for compression ramp, Sajben Transonic Diffuser, double wedge, and 2 and 3-dimensional Apollo Command Module geometries. DDADI and DDSFDT with Huang's sub-iteration correction methods showed the best convergence characteristics in terms of residual history for 2 dimensional simple cases. However, the total run time is higher

compared to the SFDT method. ADI method has the slowest convergence rates in general. For 3 dimensional Apollo Command Module, SFDT methods gave the best-converged solution. For diagonal dominant relaxation schemes maximum allowable CFL numbers decreased with implicit boundary conditions on block interfaces. However, at equal CFL numbers convergency improved. Implicit boundary conditions improved the convergence performance of the Standard ADI method dramatically, maximum allowable CFL numbers are increased for multi-block grids. SFDT method with implicit boundary conditions eliminates the errors caused by the multi-block treatment and the maximum allowable CFL number is increased for the 2-dimensional Apollo Command Module. However, for the other cases, maximum allowable CFL numbers remained the same despite the elimination of the errors caused by the multi-block explicit boundary conditions treatment. It is shown that the diagonal dominance of the approximate factorization method directly affects the efficiency of the implicit boundary condition treatment for the block interfaces of the multi-block grids. It is concluded that convergence characteristics of the approximate factorization methods and efficiency of the multi-block implicit boundary conditions are related and case-dependent. Hence, it is concluded that having multiple solution algorithm options in the solver is a great advantage.

Keywords: Approximate Factorization Methods, Hypersonic Flow, Multi-block Grids, Computational Fluid Dynamics

ÖZ

YAKLAŞIK ÇARPANLARA AYIRMA YÖNTEMLERİNİN ÇOK BLOKLU KAPALI SINIR KOŞULLARI İLE BİRLİKTE HİPERSONİK HIZLARDA YAKINSAMA PERFORMANSLARI

Koca, Melikşah
Yüksek Lisans, Havacılık ve Uzay Mühendisliği
Tez Yöneticisi: Prof. Dr. Sinan Eyi
Eylül 2022, 105 sayfa

Bu tez çalışması, farklı akış koşullarında, özellikle hipersonik akış koşullarında ve 2 boyutlu ve 3 boyutlu geometrilerde örtük yaklaşık çarpanlara ayırma yöntemlerinin yakınsama özelliklerini sunmaktadır. Çok bloklu çözüm ağları için blok ara yüzlerindeki kapalı sınır koşullarının etkinliği, farklı yaklaşık çarpanlara ayırma yöntemleri için araştırılmıştır. Standart Değişken Yön Kapalı (ADI) yöntemi, Huang'ın alt adımlama düzeltmesi olan ve olmayan Köşegen Baskın Değişen Yön Kapalı yöntemi (DDADI), CFL3D açık kaynak akış çözücüsünde kullanılan Köşegen Zaman terimleri ile Uzamsal olarak faktöre Alınmış (SFDT) yöntemiyle ve Huang'ın alt adımlama düzeltmesi olan ve olmayan bu çalışma için geliştirilmiş Köşegen Baskın Köşegen Zaman terimleri ile Uzaysal olarak faktöre Alınmış (DDSFDT) yöntem, sıkıştırılabilir Reynolds-ortalamalı Navier-Stokes denklemlerini çözerken sol tarafta üç köşegenli matrisler elde etmek için kullanılmıştır. Sıkıştırma rampası, Sajben Transonic Difüzörü, çift kama, 2 ve 3 boyutlu Apollo Komuta Modülü geometrileri için analizlerin hata geçmişleri ve toplam çalışma süreleri karşılaştırılmıştır. Huang'ın alt iterasyon düzeltme yöntemleri ile DDADI ve DDSFDT, 2 boyutlu basit durumlar için hata geçmişi açısından en iyi yakınsama özelliklerini gösterir. Ancak toplam çalışma süresi SFDT yöntemine göre daha yüksektir. ADI yöntemleri genel olarak en yavaş yakınsama

oranlarına sahiptir. 3 boyutlu Apollo Komuta Modülü için SFDT yöntemleri en iyi yaklaşım göstermiş çözümü verir. Köşegen baskın yaklaşık çarpanlara ayırma şemaları için izin verilen maksimum CFL sayıları, blok ara yüzleri üzerindeki kapalı sınır koşulları ile azalmıştır. Ancak, eşit CFL sayılarında yakınsama iyileşmiştir. Kapalı sınır koşulları, Standart ADI yönteminin yakınsama performansını önemli ölçüde iyileştirmiştir, çok bloklu çözüm ağları için izin verilen maksimum CFL sayıları artmıştır. Kapalı sınır koşullarına sahip SFDT yöntemi, çoklu blok çözüm ağı uygulamasının neden olduğu hataları ortadan kaldırmıştır ve 2 boyutlu Apollo Komut Modülü için izin verilen maksimum CFL sayısı artırmıştır. Ancak, diğer durumlarda, çoklu blok çözüm ağı uygulamasının neden olduğu hataların ortadan kaldırılmasına rağmen, izin verilen maksimum CFL sayıları aynı kalmıştır. Yaklaşık çarpanlara ayırma yönteminin köşegen baskınlığının, çok bloklu çözüm ağları blok ara yüzleri için örtük sınır koşulu uygulamasının etkinliğini doğrudan etkilediği gösterilmiştir. Yaklaşık çarpanlara ayırma yöntemlerinin yakınsama özellikleri ile çok bloklu kapalı sınır koşullarının etkinliğinin ilişkili ve duruma bağlı olduğu sonucuna varılmıştır. Dolayısıyla, akış çözücüde farklı çözüm algoritmalarının kullanılması farklı akış/geometriler için gerekmektedir.

Anahtar Kelimeler: Yaklaşık Çarpanlara Ayırma Yöntemleri, Hipersonik Akış,
Çok Bloklu Çözüm Ağları, Hesaplamalı Akışkanlar Mekaniği.

Dedication to my family

ACKNOWLEDGMENTS

I would like to thank my supervisor Prof. Dr. Sinan Eyi for his guidance, advice, criticism, encouragement, and insight throughout the research.

I would like to thank Hilmi Berk Gür for his supports, guidance and advices that he has given at the beginning of my thesis study about hypersonic flow regimes and CFL3D open-source code.

My thanks are extended to my dear family, my father İlhan Koca, my mother Kadriye Özdemir, my brother Buğrahan Koca and my fiancé Fatma Nur Kuloğlu for their endless support and useful advices.

I would like to thank my colleagues especially Buğrahan Öztürk and my friends Berk Sarıkaya, Serkan Gemalmaz, Onur Gençoğlu, Tolga Kayabaşı, Hüseyin Coşgun, Şule Nur Altunay, Enes Coşkun (and others) From the department of Aerospace Engineering for their support and enjoyable discussions throughout the years.

Lastly I would like to thank my childhood friends, Alpay Şahin Okur, Burak Yüksel, Erkin Girgin, Şerif Ercan Özkan, Nazmi Can Çalık, Muhammed Emin Karataş, Hakan Alkan, (and others) for their valuable friendship throughout the years.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS

ADI	Alternating Direction Implicit
AF	Approximate Factorization
CFL3D	Computational Fluid Laboratory 3-Dimensional
DDADI	Diagonally Dominant Alternating Direction Implicit
DDSFDT	Diagonally Dominant Spatially Factored with Diagonal Time terms
EBC	Explicit Boundary Conditions
IBC	Implicit Boundary Conditions
ICASE	International Council of Associations for Science Education
NASA	National Aeronautics and Space Administration
SFDT	Spatially Factored with Diagonal Time terms

LIST OF SYMBOLS

LATIN SYMBOLS

A	Jacobian matrix of F flux vector
a,b,c	Coefficients of the left-hand side Jacobian Matrix
B	Jacobian matrix of G flux vector
C	Jacobian matrix of H flux vector
e	Energy
F	Inviscid flux vector in x-direction
G	Inviscid flux vector in y-direction
H	Inviscid flux vector in z-direction
J	Jacobian of coordinate transformation from Cartesian to generalized coordinates
M	Mach number
p	Pressure
Q	Flow variable vector in conservative form
q	Flow variable vector in primitive form
R	Residual vector of the system
Re	Reynolds number
u,v,w	Velocity components
U,V,W	Contravariant velocity components
x,y,z	Components of Cartesian coordinates

C_p	Pressure coefficient
Δt	Time step

GREEK SYMBOLS

$\dot{q}$	Heat flux
γ	Ratio of the specific heats
δ	Difference operator
ξ, η, ζ	Components of generalized coordinates
ρ	Density
τ	Shear stress
∂	Partial differentiation operator

SUBSCRIPTS

∞	Free-stream value
i,j,k	Cell centered grid indices
x,y,z	Differentiation with respect to x,y,z
ν	Viscous terms
ξ, η, ζ	Differentiation with respect to ξ, η, ζ

SUPERSCRIPTS

0	coefficient of the previous block
1	coefficient of the current block
2	coefficient of the next block
m,k	Iteration counter

CHAPTER 1

INTRODUCTION

1.1 Motivation

Over the years, as the level of technology increased, the interest and need for hypersonic speeds increased. International space and arms races have made this flow regime much more popular. Intercontinental rockets and military high-speed flight vehicles need to reach hypersonic speeds to become undetectable and uncatchable. Also, for space missions constructing research on hypersonic flow regimes has significant importance. For instance, in order to have an orbit around the earth, satellites need to reach high speeds. In addition, coming back to earth from a space mission is a great problem. During atmospheric reentry, manned or unmanned capsules reach extremely high speeds. Therefore, designing such reentry capsules, rockets, and flight vehicles requires substantial knowledge about hypersonic regimes.

Designing any kind of flight and space vehicle require aerodynamic simulations and analysis both experimental and computational. However, achieving hypersonic regimes with an experimental setup is quite costly and difficult to construct. In order to simulate reentry conditions, a fairly high-pressure tank with a quite small scaled model of the reentry capsule must be provided. Therefore, the most preferable way to simulate hypersonic flows is by performing computational fluid dynamics analysis (CFD).

To achieve a successful design process, CFD analysis should be adequate in terms of accuracy and robustness. Since design is an iterative process, obtaining quick and accurate results from CFD analysis becomes significantly important in terms of time-

saving. Hence, accurate, fast, and robust numerical methods and algorithms must be implemented into the flow solver. Moreover, the flow solver should have the capability to solve different geometries with a variety of flow conditions. However, solutions with numerical methods of the governing equations of the CFD solvers are not always stable. Some of the numerical methods solve the required equations for low Mach numbers, but convergency slows down for high-speed flows. Therefore, choosing a proper numerical method for hypersonic flow regimes would be an appropriate start for a design process of a hypersonic vehicle.

1.2 Background

CFD solvers perform the simulations via solving partial differential equations in the computation domain in order to construct aerodynamic analysis. The necessary calculations required to operate in the domain are carried out by the nodes within it. These nodes can be arranged in a structured or unstructured manner. Both structured and unstructured grids have advantages and disadvantages. Structured grids have distributed nodes in an organized pattern around the geometry, while the unstructured grid has nodes that are more spread out. Structured grids require less storage and work well for simple configurations. For structured grids converting the Cartesian coordinates to generalized coordinates and obtaining the solution domain is much simpler in terms of coding compared to the unstructured solvers. However, it is not always possible to create single-block grids with structured cells on the flow domain for complex geometries. Therefore, having multi-block capabilities in the solver is a great advantage for generating a structured grid around complex geometry. Moreover, multi-block grids with parallelized solver codes give the solution quicker compared to the single-block grids.

Coding a numerical algorithm that solves Navier-Stokes equations has two main aspects: time marching procedure and spatial discretization. The spatial accuracy of

the flow domain and resolution of the solution is directly associated with the accuracy of spatial discretization. On the other hand, the time-marching procedure is directly related to the computational efficiency and robustness. Time-marching algorithms can also be divided into two main categories explicit and implicit time marching algorithm. Explicit algorithm limits the time steps. It is not possible to obtain the solution with large time steps for a high Reynolds number wall-bounded with boundary layers simulation compared to the implicit methods. By contrast, large time steps can be significantly efficient with implicit time-marching algorithms. Implicit methods are more stable and has higher iterative convergence rates in general.

1.3 Objectives

There exist three main objectives of this thesis study. Firstly, comparing the convergence performance of the approximate factorization methods for different geometries and flow conditions. Secondly, detecting which method is stable, has the best convergence rates and best option in terms of accuracy and robustness for hypersonic flow regimes. Lastly, analyzing the efficiency of the implicit boundary conditions on block interfaces with different approximate factorization methods.

1.4 Outline

In chapter 2; a literature review of the thesis is presented. Articles written about convergence acceleration methods such as implicit boundary conditions on block interfaces of multi-block grids, Huang's sub-iteration correction, and approximate factorization methods are explained. Mesh topology for reentry capsules is discussed.

In chapter 3; governing equations of the solver code are presented. The exact form of the left-hand side Jacobian matrix is obtained and converted to the tridiagonal matrices with approximate factorization methods. Error terms introduced by the factorizations are calculated. The implicit boundary conditions algorithm on block interfaces is explained with equations. Huang's sub-iteration procedure is presented.

In chapter 4; results are presented with necessary figures and tables. Convergence characteristics of the approximate factorization methods are compared. The efficiency of the implicit boundary condition is discussed with 4 different approximate factorization methods. The convergence performances of the methods are compared.

CHAPTER 2

LITERATURE REVIEW

Design projects may require hundreds of flow analyses. Therefore, before starting a design project it is crucial to choose an appropriate solver for rapid convergence in terms of time-saving. The solver should have accurate, robust, and efficient time-marching algorithms. Time-marching algorithms can be classified into two main categories which are implicit time-marching and explicit time-marching methods. Explicit methods which are represented by the multistage Runge-Kutta [1] method are not efficient for wall-bounded with high Reynolds numbers and viscous boundary layers. For explicit time-marching algorithms, allowable time steps are quite small. Therefore implicit methods are developed because implicit methods provide adequate stability features and convergence rates, and allow larger time steps that help the solution to converge rapidly. However, implicit time-marching algorithms requires solution of a large matrix equations which is quite computationaly costly. Hence, in order to increase the computational efficiencies of the implicit methods, approximation algorithms are developed for these large matrices such as Alternating Direction Implicit methods, also known as approximate factorization methods.

One of the first studies for solving PDEs with a stable implicit method is Crank-Nicolson [2] algorithm which is developed in 1947 for diffusion equations. Lower-upper symmetric Gauss-Seidel(LU-SGS) [3], alternating direction implicit or approximate factorization methods [4]–[27], and Newton-Krylov methods [28]–[30] are developed and become attractive alternatives among implicit methods due to the

superlinear convergence rates, unconditional stability, and allowable large CFL numbers.

Accurate residual linearization in order to obtain a set of large-scale linearized equations, efficient and accurate solver of the linearized equations, and multi-block implicit boundary conditions on block interfaces are the major steps of successful implementation of the implicit time advancement algorithms. Matrix-free methods [29], automatic differentation [31] and analytical approximation [32] are the methods that commonly used for linearization procedure of the equations. Matrix-free methods and automatic differentiation methods are costly in terms of computing and memory storage [29], [33]. Although the accuracy of the analytical approximation methods highly relies on the accuracy of spatial discretization, it is the most popular technique because of its higher efficiency and easy applicability and is used in this thesis study. In or the solve the linearized equations direct Gauss elimination, iterative methods with relaxation schemes, Newton-Krylov subspace methods, and alternating direction implicit or approximate factorization schemes can be used. For large-scale matrices, Gauss elimination is not practical. Iterative methods demand large memory requirements and have low convergence rates. Implementation of the preconditioners is required for the Newton-Krylov methods. Therefore, the most popular way to solve the linearized equations is using approximate factorization methods despite the factorization error.

2.1 Approximate Factorization Methods

A major development occurred in 1955 for noniterative implicit methods when Peaceman and Rachford [26] introduced the first version of the ADI method. Douglas [24], Pearson [23], Wilkes and Chuchill [22], Briley, and Walls [21] made further analysis on ADI method. Douglas and Gunn, Briley and Mcdonald [19], [20], [27] have constructed further studies for three-dimensional compressible Navier stokes equations. ADI method has taken its baseline form with studies of R. Beam

and R.F Warming [5] and was introduced as the Standard ADI method in 1978. Using Standart ADI method as a baseline method in 1987 J. Bardina and C.K. Lombard [11] developed the Diagonal Dominant ADI(DDADI) method which is resurrected by R.W. MacCormark [25] later in 1997. With the same procedure that Pulliam and Chaussee [7] developed in 1981 for diagonalization of the ADI method G.H. Klopfer [12] developed Diagonalized Diagonal Dominant ADI method in 1998. Efficiencies and convergence characteristics of these approximate factorization methods are discussed by Pulliam, MacCormack, and Venkateswaran [9] and a comprehensive review of the approximate factorization methods is published by Briley, and McDonald [17]. In the early 1980s, NASA and ICASE brought together the leading scientist in numerical methods and introduced CFL3D open-source code with another approximate factorization method [34]. In 1992, James L. Thomas [13] introduced an implicit multigrid spatially-factored scheme that is used in CFL3D open-source code. In his study, in contrast to the Standard ADI method, time terms remained as diagonal elements. In fact, in 1997 approximate factorization method that was developed for CFL3D open-source code was claimed as one of the best approximate factorization methods in terms of convergence rate and stability in Ref. [34]. However, it is not proven with any published paper which compares the approximate factorization method used in CFL3D with other approximate factorization methods. Pulliam, MacCormack, and Venkateswaran [9] claim that approximate factorization methods are case-dependent and convergence characteristics of the approximate factorization methods might change from case to case. CFL3D open-source code developed for transonic cases and there is only one study in the literature that investigates the convergence characteristics of the CFL3D open-source at hypersonic regimes at Mach number 6 [35]. Therefore, in this study, in order to fill these gaps in the literature converge characteristics of the approximate factorization methods are compared at hypersonic speeds. According to mentioned previous studies after linearizing the equations in order to reduce the computational cost, Jacobian matrices can be converted to tridiagonal matrices using approximate

factorization methods. These tridiagonal matrices can be solved with Thomas algorithms easily.

2.2 Implicit Boundary Conditions on Block Interfaces

Multi-block grids can be very helpful for CFD analysis in terms of time-saving. Splitting the grids to a certain number of blocks to have equally distributed cell numbers and allocating each block to different processors with the use of 1-1 blocking and MPI features reduces the computational cost for processors. Hence, the same number of iterations can be completed in a shorter time period. However, as mentioned in the introduction chapter splitting blocks and solving tridiagonal matrices separately introduce errors in the block interfaces. In order to overcome this issue in 2011, Satbir Singh and Donghyun You [18] used implicit boundary conditions. Recently, in 2020, Hojun moon, Seungyo Hong, Donghyun You [14] implement implicit boundary conditions into their flow solver and observed a reduction in the convergence rates for different flow cases. Their suggestion was to use the parallel diagonal dominant algorithm instead of thomas algorithm to accelerate the convergency. In 2020, Yao Jin, Fei Liao, and Jinsheng Cai [16] coupled the Navier-stokes equations with Spalart-Almaras turbulence model and introduced implicit boundary conditions coupled with Huang's sub-iteration correction for DDADI and Diagonalized DDADI methods. In addition, they claimed that the importance of keeping the implicit treatment of the turbulence model to be consistent with Navier-Stokes equations is important, otherwise, the convergence characteristics of the Navier-Stokes equations will be adversely affected. They also showed that DDADI and Diagonalized DDADI method has better convergence rates compared to the LU-SGS scheme.

2.3 Huang's Sub-iteration Correction

As mentioned in the introduction chapter approximate factorization methods introduce a factorization error because, in order to obtain tridiagonal matrices instead of full Jacobian, the left-hand side summation term is written as multiplication. This situation causes factorization errors. To eliminate factorization errors MacCormack [36]–[38] and Huang [39] introduced a sub-iteration procedure. G.H. Klopfer [12] used this sub-iteration procedure with the Diagonalized Diagonal Dominant ADI method and observed better convergence characteristics.

2.4 Mesh Topology For Reentry Capsules

Reentry vehicles fly at hypersonic speeds during atmospheric reentry. In order to obtain a proper simulation of atmospheric reentry of a capsule, grids must be adjusted according to capsule geometries. Most of the atmospheric reentry capsules have axisymmetic geometries. While forming the grid for 3-dimensional axisymmetric geometry with structured type cells unstructured type cells might be observed in both poles of the geometries. Therefore, if hybrid grids are not suitable for the flow solver, the solution may converge but there might be bumps caused by the mathematical singularity on the pole which is not realistic. In order to obtain the grids with only structured cells, O-H(butterfly)-type grids must be used. Prabhakar Subrahmanyam [40] developed a grid generation tool for axisymmetric reentry capsules and suggested O-H-type grids for such vehicles. In this thesis study mesh topology shown in figure 2-1 is used for the 3-dimensional Apollo Command Module.

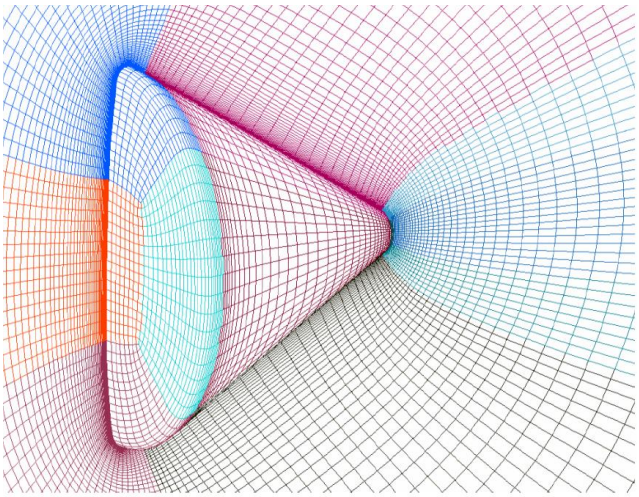


Figure 2-1 Mesh Topology for Apollo Command Module[40]

CHAPTER 3

SOLUTION METHODS

CFL3D is an open-source code developed by NASA and it is a compressible Reynolds-averaged Navier-Stokes solver for structured grids. CFL3D solves the time-dependent conservation law form of the equations in generalized coordinates. Spatial discretization involves a semi-discrete finite-volume approach. Upwind-biasing is used for the convective and pressure terms, while central differencing is used for the shear stress and heat transfer terms. Time advancement is implicit and has the ability to solve steady or unsteady flows. Numerous turbulence models are provided, including 0-equation, 1-equation, and 2-equation models and Spalart-Almaras turbulence model. Multiple-block topologies are possible with the use of 1-1 blocking[41]. In this study, in order to be consistent, in all analyses, Spalart-Almaras turbulence model is used as a turbulence model, and Upwind-biased third order spatial differencing and flux-difference splitting of Roe's scheme is used for the flux calculations. CFL3D open-source code has a multigrid algorithm but CFL3D was developed for transonic flow regimes and for hypersonic flow regimes convergence ability is quite poor with multigrid. Therefore, in this study multigrid capability of the CFL3D is not used.

3.1 Governing Equations

Three-dimensional time-dependent compressible Navier-Stokes equations in generalized coordinates can be written as follows:

$$\frac{\partial \hat{Q}}{\partial t} + \frac{\partial\left(\hat{F} - \hat{F}_v\right)}{\partial \xi} + \frac{\partial\left(\hat{G} - \hat{G}_v\right)}{\partial \eta} + \frac{\partial\left(\hat{H} - \hat{H}_v\right)}{\partial \zeta} = 0 \qquad (3.1)$$

Where $\hat{F} - \hat{F}_v$, $\hat{G} - \hat{G}_v$, $\hat{H} - \hat{H}_v$ are inviscid and viscous flux terms with respect to the generalized directions. (See Appendix A for details.) Jacobian of the transformation matrix J and conservative variables vector can be written as follows:

$$J = \frac{\partial(x, y, z, t)}{\partial(\xi, \eta, \zeta, t)} \quad , \quad \hat{Q} = JQ = J \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{pmatrix} \qquad (3.2)$$

Where J is the transformation matrix and Q is the vector form of the conservative variables such as: density, momentum in all three dimensions, and total energy per unit volume. After simplification and applying implicit time-marching technique we have:

$$J\frac{\partial Q}{\partial t} + R(Q) = 0 \ , \quad J\frac{\partial Q^{m+1}}{\partial t} + R(Q^{m+1}) = 0 \qquad (3.3)$$

Where:

$$R(Q) = \frac{\partial\left(\hat{F} - \hat{F}_v\right)}{\partial \xi} + \frac{\partial\left(\hat{G} - \hat{G}_v\right)}{\partial \eta} + \frac{\partial\left(\hat{H} - \hat{H}_v\right)}{\partial \zeta} \qquad (3.4)$$

Superscript $m + 1$ correspond to Newton iteration index and steady state solution can be obtained by iterating equation 3.3. With the help of Taylor series expansion residual terms can be linearized by discarding the high-order truncation terms.

$$R(Q^{m+1}) = R(Q^m) + \frac{\partial R}{\partial Q}\Big|^m \Delta Q^{m+1} + 0(\|Q^{m+1}\|^2) \qquad (3.5)$$

Linearized and reorganized form of the equation 3.3 becomes:

$$\left(\frac{J}{\Delta t}I + \frac{\partial R}{\partial Q}\Big|^{m}\right)\Delta Q^{m+1} = -R(Q^{m}) \tag{3.6}$$

Where $\frac{J}{\Delta t}I$ is the diagonal pseudo time stepping term and $\frac{\partial R}{\partial Q}\Big|^{m}$ is flux jacobian matrices. Substituting equation 3.4 into equation 3.6 and rearranging:

$$\left\{I + J^{-1}\Delta t\left(\delta_{\xi}A_{\xi} + \delta_{\eta}B_{\eta} + \delta_{\zeta}C_{\zeta}\right)\right\}\Delta Q^{m+1} = -J^{-1}\Delta t R(Q^{m}) \tag{3.7}$$

Where:

$$A_{\xi} = \frac{\partial\left(\hat{F} - \hat{F}_{v}\right)}{\partial Q}\ , \qquad B_{\eta} = \frac{\partial\left(\hat{G} - \hat{G}_{v}\right)}{\partial Q}\ , \qquad C_{\zeta} = \frac{\partial\left(\hat{H} - \hat{H}_{v}\right)}{\partial Q} \tag{3.8}$$

δ_{ξ}, δ_{η} and δ_{ζ} are difference operators along generalized coordinates ξ, η and ζ. In the CFL3D open-source code viscous fluxes are only added in k direction for the left-hand side and manual suggest that while generating the grid k direction should be the viscous direction [41]. That means k direction must be perpendicular to the wall boundary in the orientation of the coordinates of grid. However, in this thesis study, in all directions, left-hand side viscous fluxes added to the inviscid fluxes and this situation is fixed for all methods.

3.2 Approximate Factorization Methods

Solving equation 3.7 without approximate factorization methods is costly in terms of coding and time. Obtaining a full implicit Jacobian matrix and finding the solution of the system of equations with direct gauss elimination is impracticable because of the reasons mentioned in chapter 2. In full implicit methods, solution process requires additional large-scale matrice solvers and more storage compared to the factored schemes. By contrast, approximate factorization methods require less storage and Thomas algorithm is sufficient for solving the tridiagonal matrices. However, approximate factorization methods introduce factorization errors. This error differs according to the algorithm of the factorization method and can be reduced to lower levels with specific methods such as sub-iterations. In the following

subsections, approximate factorization methods and introduced factorization errors are discussed.

3.2.1 Standard Alternating Direction Implicit Method

In order to have tridiagonal matrices on the left-hand side summation term of equation 3.7 can be written as multiplication.

$$\begin{aligned}\left\{I + J^{-1}\Delta t\delta_\xi A_\xi\right\} * \left\{I + J^{-1}\Delta t\delta_\eta B_\eta\right\} * \left\{I + J^{-1}\Delta t\delta_\zeta C_\zeta\right\}\Delta Q^{m+1} \\ = -J^{-1}\Delta t R(Q^m)\end{aligned} \tag{3.9}$$

After this approximation equation 3.9 can be solved in 3 different direction as shown below:

$$\left\{I + J^{-1}\Delta t\delta_\xi A_\xi\right\}q_\xi = -J^{-1}\Delta t R(Q^m) \tag{3.10}$$

$$\left\{I + J^{-1}\Delta t\delta_\eta B_\eta\right\}q_\eta = q_\xi \tag{3.11}$$

$$\left\{I + J^{-1}\Delta t\delta_\zeta C_\zeta\right\}q_\zeta = q_\eta \tag{3.12}$$

$$q_\zeta = \Delta Q^{m+1} \tag{3.13}$$

After ΔQ^{m+1} is obtained flow field can be updated as:

$$Q^{m+1} = Q^m + \Delta Q^{m+1} \tag{3.14}$$

Aproximate factorization of the equation 3.7 introduce factorization error. Factorization error can be calculated after substracting equation 3.7 from equation 3.9.

$$\begin{aligned}\varepsilon = J^{-1}\Delta t\delta_\xi A_\xi * J^{-1}\Delta t\delta_\eta B_\eta * J^{-1}\Delta t\delta_\zeta C_\zeta + J^{-1}\Delta t\delta_\xi A_\xi * J^{-1}\Delta t\delta_\eta B_\eta \\ + J^{-1}\Delta t\delta_\xi A_\xi * J^{-1}\Delta t\delta_\zeta C_\zeta + J^{-1}\Delta t\delta_\eta B_\eta * J^{-1}\Delta t\delta_\zeta C_\zeta\end{aligned} \tag{3.15}$$

Coordinate transformation matris J corresponds to volume of a cell. Therefore, inverse of pseudo time stepping term becomes a large number for high CFL numbers. Factorization error of the standart ADI approximate factorization method can be expressed as $(J^{-1}\Delta t)^3$.

3.2.2 Diagonal Dominant Alternating Direction Implicit Method

After extracting all the diagonal elements in the implicit operator in the equation 3.7 into a separate matrix D implementation of the diagonal dominant factorization algorithm for ADI method can be written as:

$$\begin{aligned}\left\{I + J^{-1}\Delta t\delta_\xi A_\xi + D_\eta + D_\zeta\right\} * D^{-1}\left\{I + J^{-1}\Delta t\delta_\eta B_\eta + D_\xi + D_\zeta\right\} \\ * D^{-1}\left\{I + J^{-1}\Delta t\delta_\zeta C_\zeta + D_\xi + D_\eta\right\}\Delta Q^{m+1} \\ = -J^{-1}\Delta t R(Q^m)\end{aligned} \tag{3.16}$$

Where:

$$D = \left\{I + D_\xi + D_\eta + D_\zeta\right\} \tag{3.17}$$

$D_\xi, D_\eta\ ve\ D_\zeta$ are the diagonal elements of the $\left\{J^{-1}\Delta t\delta_\xi A_\xi\right\}, \left\{J^{-1}\Delta t\delta_\eta B_\eta\right\}$ and $\left\{J^{-1}\Delta t\delta_\zeta C_\zeta\right\}$ matrices. Equation 3.16 can be solved in 3 generalized directions.

$$\left\{I + J^{-1}\Delta t\delta_\xi A_\xi + D_\eta + D_\zeta\right\}q_\xi = -J^{-1}\Delta t R(Q^m) \tag{3.18}$$

$$\left\{I + J^{-1}\Delta t\delta_\eta B_\eta + D_\xi + D_\zeta\right\}q_\eta = Dq_\xi \tag{3.19}$$

$$\left\{I + J^{-1}\Delta t\delta_\zeta C_\zeta + D_\xi + D_\eta\right\}q_\zeta = Dq_\eta \tag{3.20}$$

$$q_\zeta = \Delta Q^{m+1} \tag{3.21}$$

After ΔQ^{m+1} is obtained flow field can be updated as:

$$Q^{m+1} = Q^m + \Delta Q^{m+1} \tag{3.22}$$

Factorization error can be calculated after substracting equation 3.7 from equation 3.16.

$$\begin{aligned}\varepsilon = J^{-1}\Delta t\delta_\xi A_\xi * D^{-1}J^{-1}\Delta t\delta_\eta B_\eta * D^{-1}J^{-1}\Delta t\delta_\zeta C_\zeta + J^{-1}\Delta t\delta_\xi A_\xi \\ * D^{-1}J^{-1}\Delta t\delta_\eta B_\eta + J^{-1}\Delta t\delta_\xi A_\xi * D^{-1}J^{-1}\Delta t\delta_\zeta C_\zeta \\ + D^{-1}J^{-1}\Delta t\delta_\eta B_\eta * D^{-1}J^{-1}\Delta t\delta_\zeta C_\zeta \\ + \cdots 23\ term(\Delta t)\end{aligned} \tag{3.23}$$

Factorization error is in the order of $(J^{-1}\Delta t)^1$ since the term D has $J^{-1}\Delta t$ inside.

3.2.3 SFDT Approximate Factorization

The difference of the factorization method of the CFL3D from the standard ADI method is the pseudo time terms are remains as diagonal element, not multiplied by the coefficients of the Jacobian matrices.

$$\left\{\frac{J}{\Delta t}I + \left(\delta_\xi A_\xi + \delta_\eta B_\eta + \delta_\zeta C_\zeta\right)\right\}\Delta Q^{m+1} = -R(Q^m) \tag{3.24}$$

Equation 3.24 is spatially factored and below equation is obtained.

$$\begin{aligned}\left\{\frac{J}{\Delta t}I + \delta_\xi A_\xi\right\} * \frac{\Delta t}{J}\left\{\frac{J}{\Delta t}I + \delta_\eta B_\eta\right\} * \frac{\Delta t}{J}\left\{\frac{J}{\Delta t}I + \delta_\zeta C_\zeta\right\}\Delta Q^{m+1} \\ = -R(Q^m)\end{aligned} \tag{3.25}$$

Equation 3.25 is solved in all 3 generalized coordinates.

$$\left\{\frac{J}{\Delta t}I + \delta_\xi A_\xi\right\} q_\xi = -R(Q^m) \tag{3.26}$$

$$\left\{\frac{J}{\Delta t}I + \delta_\eta B_\eta\right\} q_\eta = \frac{J}{\Delta t} q_\xi \tag{3.27}$$

$$\left\{\frac{J}{\Delta t}I + \delta_\zeta C_\zeta\right\} q_\zeta = \frac{J}{\Delta t} q_\eta \tag{3.28}$$

$$q_\zeta = \Delta Q^{m+1} \tag{3.29}$$

After ΔQ^{m+1} is obtained flow field can be updated as:

$$Q^{m+1} = Q^m + \Delta Q^{m+1} \tag{3.30}$$

Factorization error can be calculated after substracting equation 3.24 from equation 3.25.

$$\begin{aligned}\varepsilon = \delta_\xi A_\xi \frac{\Delta t}{J}\delta_\eta B_\eta \frac{\Delta t}{J}\delta_\zeta C_\zeta + \delta_\xi A_\xi \frac{\Delta t}{J}\delta_\eta B_\eta \frac{J}{\Delta t} + \delta_\xi A_\xi \frac{\Delta t}{J}\delta_\zeta C_\zeta \frac{J}{\Delta t} \\ + \frac{J}{\Delta t}\frac{\Delta t}{J}\delta_\eta B_\eta \frac{\Delta t}{J}\delta_\zeta C_\zeta\end{aligned} \tag{3.31}$$

Factorization error is in the order of $(J^{-1}\Delta t)^2$.

3.2.4 Diagonal Dominant SFDT Approximate Factorization Method

SFDT algorithm is already diagonal dominant because of the diagonal time terms. In order to detect whether this method can become more diagonal dominant or not same procedure for DDADI algorithm is applied to the SFDT method.

$$\left\{\frac{J}{\Delta t}I + \delta_\xi A_\xi + D_\eta + D_\zeta\right\} * D^{-1}\left\{\frac{J}{\Delta t}I + \delta_\eta B_\eta + D_\xi + D_\zeta\right\} * D^{-1}\left\{\frac{J}{\Delta t}I + \delta_\zeta D_\zeta + D_\xi + D_\eta\right\}\Delta Q^{m+1} = -R(Q^m) \tag{3.32}$$

$$D = \left\{\frac{J}{\Delta t}I + D_\xi + D_\eta + D_\zeta\right\} \tag{3.33}$$

D_ξ, D_η and D_ζ are the diagonal elements of the $\{\delta_\xi A_\xi\}, \{\delta_\eta B_\eta\}$ and $\{\delta_\zeta C_\zeta\}$ matrices.

$$\left\{\frac{J}{\Delta t}I + \delta_\xi A_\xi + D_\eta + D_\zeta\right\} q_\xi = -J^{-1}\Delta t R(Q^m) \tag{3.34}$$

$$\left\{\frac{J}{\Delta t}I + \delta_\eta B_\eta + D_\xi + D_\zeta\right\} q_\eta = D q_\xi \tag{3.35}$$

$$\left\{\frac{J}{\Delta t}I + \delta_\zeta C_\zeta + D_\xi + D_\eta\right\} q_\zeta = D q_\eta \tag{3.36}$$

$$q_\zeta = \Delta Q^{m+1} \tag{3.37}$$

After ΔQ^{m+1} is obtained flow field can be updated as:

$$Q^{m+1} = Q^m + \Delta Q^{m+1} \tag{3.38}$$

Factorization error can be calculated after substracting equation 3.24 from equation 3.32.

$$\varepsilon = \delta_\xi A_\xi * D^{-1}\delta_\eta B_\eta * D^{-1}\delta_\zeta C_\zeta + \delta_\xi A_\xi * D^{-1}\delta_\eta B_\eta \frac{J}{\Delta t} + \delta_\xi A_\xi * D^{-1}\delta_\zeta C_\zeta \frac{J}{\Delta t} + D^{-1}\delta_\eta B_\eta * D^{-1}\delta_\zeta C_\zeta \frac{J}{\Delta t} + \cdots 23\ term\left(\Delta t^0\right) \tag{3.39}$$

Factorization error is in the order of $(J^{-1}\Delta t)^2$ since the term D has $J^{-1}\Delta t$ inside.

3.3 Huang's Sub-iteration Correction

As mentioned before, approximate factorization methods introduce factorization error. One of the favorable elimination method of this error is the sub-iteration procedure. In order to remove this factorization error Huang proposed the below sub-iteration procedure diagonal dominant relaxation schemes:

$$\left\{I + J^{-1}\Delta t\delta_\xi A_\xi + D_\eta + D_\zeta\right\}\left(q_\xi^m - q_\xi^{m-1}\right) = D\left(q_\zeta^{m-1} - q_\xi^{m-1}\right) \qquad (3.40)$$

$$\left\{I + J^{-1}\Delta t\delta_\eta B_\eta + D_\xi + D_\zeta\right\}\left(q_\eta^m - q_\eta^{m-1}\right) = D\left(q_\xi^{m-1} - q_\eta^{m-1}\right) \qquad (3.41)$$

$$\left\{I + J^{-1}\Delta t\delta_\zeta C_\zeta + D_\xi + D_\eta\right\}\left(q_\zeta^m - q_\zeta^{m-1}\right) = D\left(q_\eta^{m-1} - q_\zeta^{m-1}\right) \qquad (3.42)$$

For each newton iteration after the first solution sweeps are completed for all directions, above equations are solved for sub-iterations. This procedure eliminates the recalculation of the right-hand side fluxes and q_ξ^m, q_η^m, and q_ζ^m are convergence to more accurate values. This procedure eliminates the factorization errors and allows higher CFL numbers. However, required time for each newton iteration increases because of the additional calculations that sub-iteration brings.

3.4 Implicit Boundary Conditions on Block Boundaries

Multi-block grids can be obtained by dividing the flow domain into separate flow domains. After proper block to node mapping, each processor solves the tridiagonal matrix equations separately. This procedure speeds up the CFD analysis considerably. However, during the dividing processes, some of the coefficients of the tridiagonal matrices are omitted if approximate factorization methods are used in the solver. Number of the omitted coefficients increases for direct methods. Hence explicit boundary condition on block interfaces cause error. In order to overcome this issue implicit boundary condition can be implemented. For Standard ADI method equations 3.10, 3.11, and 3.12, for DDADI method equations 3.18, 3.19, and 3.20, for SFDT method equations 3.26, 3.27, and 3.28, and for DDSFDT method equations 3.34, 3.35, and 3.36 can be written in the matrix form as shown below.

$$\begin{bmatrix} b_1^0 & c_1^0 & & & \vdots & & & & \\ a_2^0 & b_2^0 & c_2^0 & & \vdots & & & & \\ & \ddots & \ddots & \ddots & \vdots & & & & \\ & & a_N^0 & b_N^0 & \vdots & c_N^0 & & & \\ \cdots & \cdots & \cdots & \cdots & \vdots & \cdots & \cdots & \cdots & \cdots \\ & & & a_1^1 & \vdots & b_1^1 & c_1^1 & & \\ & & & & \vdots & a_2^1 & b_2^1 & c_2^1 & \\ & & & & \vdots & & \ddots & \ddots & \ddots \\ & & & & \vdots & & & a_N^1 & b_N^1 \end{bmatrix} \begin{pmatrix} \Delta q_1^{0,k} \\ \Delta q_2^{0,k} \\ \vdots \\ \Delta q_N^{0,k} \\ \cdots \\ \Delta q_1^{1,k} \\ \Delta q_2^{1,k} \\ \cdots \\ \Delta q_N^{1,k} \end{pmatrix} = \begin{pmatrix} R_1^{0,k-1} \\ R_2^{0,k-1} \\ \vdots \\ R_N^{0,k-1} \\ \cdots \\ R_1^{1,k-1} \\ R_2^{1,k-1} \\ \vdots \\ R_N^{1,k-1} \end{pmatrix} \tag{3.43}$$

In equation 3.43 a_1^0, b_1^0, and c_1^0 represents the coefficients of the Jacobian matrices for the block number 0. $\Delta q_1^{0,k}$ and $R_1^{0,k-1}$ represents the unknown variable and right-hand side variables of the equations 3.10, 3.11, 3.12, 3.18, 3.19, 3.20, 3.26, 3.27, 3.28, 3.34, 3.35, and 3.36. Dashed lines represent the block interfaces. Explicit boundary conditions ignore the multiplications of the a_1^1 and c_N^0 coefficients with Δq_0^{k-1} and Δq_{N+1}^{k-1} which are unknown variables of the neighboring blocks. By contrast, with single-block grid these variables are not neglected. Therefore, for

multi-block grids omitting these multiplications of the coefficients with right-hand side variables introduce an error and this situation jeopardizes the implicit structure of the methods.

$$\begin{bmatrix} (a_1^1) & \vdots & b_1^1 & c_1^1 & & & & \vdots & \\ & \vdots & a_2^1 & b_2^1 & c_2^1 & & & \vdots & \\ & \vdots & & \ddots & \ddots & \ddots & & \vdots & \\ & \vdots & & & a_{N-1}^1 & b_{N-1}^1 & c_{N-1}^1 & \vdots & \\ & \vdots & & & & a_N^1 & b_N^1 & \vdots & (c_N^2) \end{bmatrix} \begin{pmatrix} \Delta q_0^{k-1} \\ \dots \\ \Delta q_1^k \\ \Delta q_2^k \\ \vdots \\ \Delta q_{N-1}^k \\ \Delta q_N^k \\ \dots \\ \Delta q_{N+1}^{k-1} \end{pmatrix} = \begin{pmatrix} R_1^{k-1} \\ R_2^{k-1} \\ \vdots \\ R_{N-1}^{k-1} \\ R_N^{k-1} \end{pmatrix} \tag{3.44}$$

a_1^1 and c_N^2 represent the left-hand side coefficients of the Jacobian matrix of the current block while Δq_0^{k-1} and Δq_{N+1}^{k-1} represents the left-hand side unknown flow variables of the neighboring blocks for previous iteration. After first iteration is completed, Δq_0^{k-1} and Δq_{N+1}^{k-1} flow variables are founded for all blocks. For the second iterations, these variables should be communicated between blocks in order to implement implicit boundary conditions algorithm. CFL3D code establishes communication for updated flow variables for right-hand side residual calculations. However, in order to implement the multi-block implicit boundary conditions treatment on block boundaries for equations 3.10, 3.11, 3.12, 3.18, 3.19, 3.20, 3.26, 3.27, 3.28, 3.34, 3.35, and 3.36, q_ξ^{k-1}, q_η^{k-1}, and q_ζ^{k-1} flow variables of the previous iteration of the neighboring blocks should be communicated between blocks.

In order to eliminate the error on block boundaries, omitted coefficients can be calculated for the current block and founded flow variables of the neighboring blocks can be extracted from the previous iteration. After the calculation of the

multiplication of these variables, these terms can be added to the right-hand side for the current iteration and the current block as shown in below equation 3.45. This procedure must be implemented into all directions if there exist block interfaces in all directions.

$$\begin{bmatrix} b_1^1 & c_1^1 & & & & \\ a_2^1 & b_2^1 & c_2^1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & a_{N-1}^1 & b_{N-1}^1 & c_{N-1}^1 & \\ & & & a_N^1 & b_N^1 \end{bmatrix} \begin{pmatrix} \Delta q_1^k \\ \Delta q_2^k \\ \vdots \\ \Delta q_{N-1}^k \\ \Delta q_{N+1}^k \end{pmatrix} = \begin{pmatrix} R_1^{k-1} - (a_1^1)\Delta q_0^{k-1} \\ R_2^{k-1} \\ \vdots \\ R_{N-1}^{k-1} \\ R_N^{k-1} - (c_N^2)\Delta q_{N+1}^{k-1} \end{pmatrix} \quad (3.45)$$

Stability features and robustness directly related with the diagonal dominancy of the Jacobian matrices of the linear system of equations. In fact, in diagonal dominant algorithms, diagonal elements of the Jacobian matrices drive the solutions because a_1^1, and c_1^1 are quite small compared to the b_1^1 coefficients. This means omitted variables on block interfaces are negligible for sufficiently diagonal dominant solution algorithms. However, it should be noted that these coefficients are case-dependent and it is not proper to say omitted coefficients are always negligible for diagonal dominant algorithms. For Standard ADI method which is not diagonal dominant algorithm, these omitted terms cause accumulation of the error introduced on block interfaces and lead divergence of the solution.

CHAPTER 4

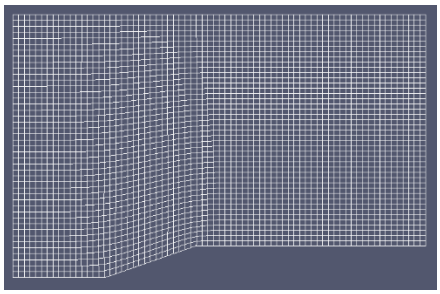
RESULTS AND DISCUSSION

Convergence characteristic of the approximate factorization methods are investigated on different geometries. Upwind-biased third order spatial differencing and flux-difference splitting of Roe's scheme is used for the flux calculations. In all analyses Spalart-Almaras turbulance model is used for consistency. Efficiency of the implicit boundary conditions with all of the approximate factorization methods are compared. Flow inside the Transonic Sajben Transonic Diffuser and flow over Compression Ramp, Double Wedge, 2 and 3 dimensional Apollo Command Module are simulated. It is observed that approximate factorization methods and multi-block implicit boundary conditions methods are case-dependent. Compression Ramp case selected as simplest study in order to show the full capability of the approximate factorization methods. Sajben Transonic Diffusers case is studied in the purpose of expressing the case dependency of the multi-block implicit boundary condition and approximate factorization methods. In order to prove the fact that maximum allowable CFL numbers are used in the analyses, divergent solutions are also added into residual history charts. In addition it should be noted that in the residual history charts, symbol "D" means solution diverges for given CFL number.

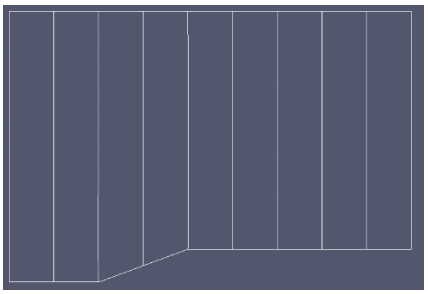
4.1 Compression Ramp (2D)

Compression Ramp is one of the simplest case at hypersonic speeds. Geometry and grid are provided in the website and manual of the CFL3D [41]. 20 degree ramp is used in this compression ramp case. Mach number of the free stream settled as 10. 2D inviscid Euler equations are solved with implicit approximate factorization methods. Effect of the implicit boundary conditions on block interfaces are

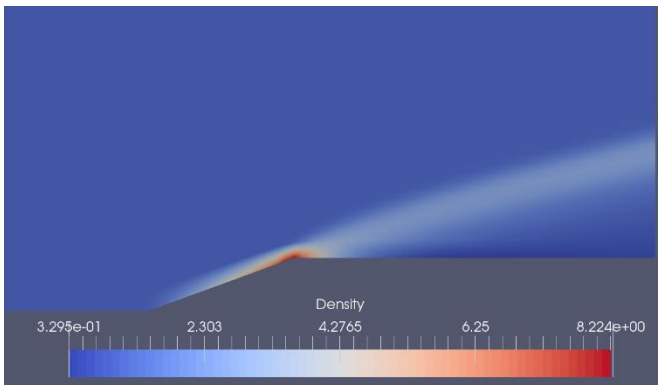
investigated after splitting the single-block grid to 9 blocks. 9-block grids are solved with 10 processors.



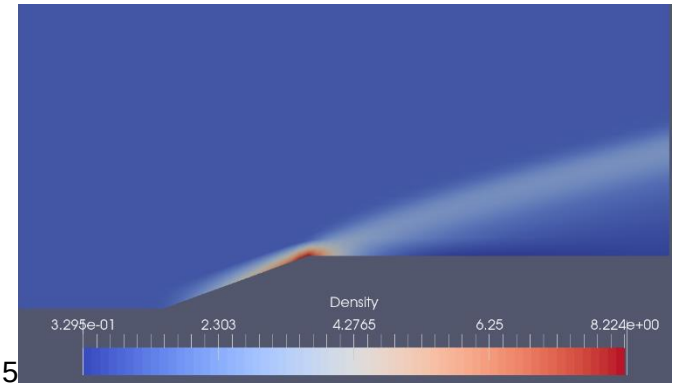
a) Single-block Grid



b) 9 Block Grid

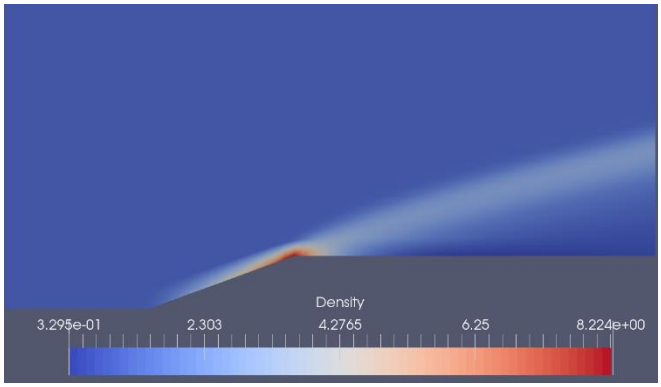


c) Density Distribution (ADI)

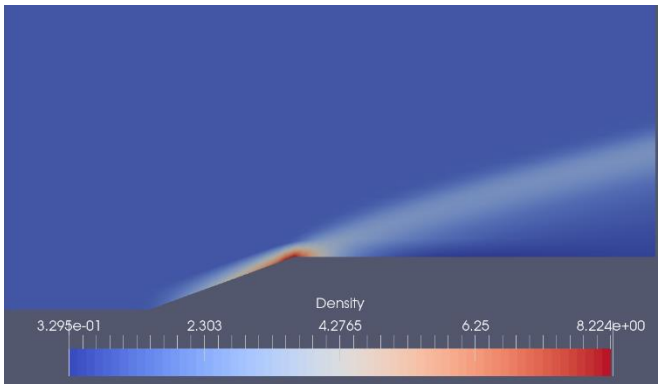


5

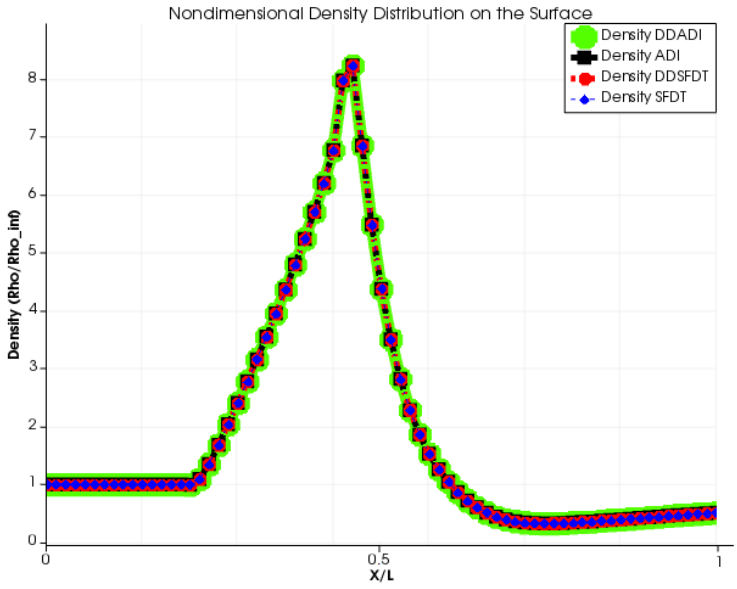
d) Density Distribution (DDADI)



e) Density Distribution (DDSFDT)



f) Density Distribution (SFDT)



g) Density Distribution on the Surface Along the Flow Field

Figure 4-1 Generated Mesh, Contour Plots, and Pressure Coefficient Comparison Plot of the AF Methods for the Compression Ramp Geometry

Figure 4-1 shows generated mesh, contour plots of the analyses of the Compression Ramp, and comparison of the density distribution on the surface obtained by different approximate factorization methods. Single-block grid is obtained as shown in figure 4-1 a) and split into 9 blocks as shown in figure 4-1 b). By comparing figure 4-1 c)-f) and considering the figure 4-1 g) it can be said that all of the methods converged to a similar solution. It should be noted that densities are nondimensionalized with free stream density which is settled as 1. (See Appendix A for details.)

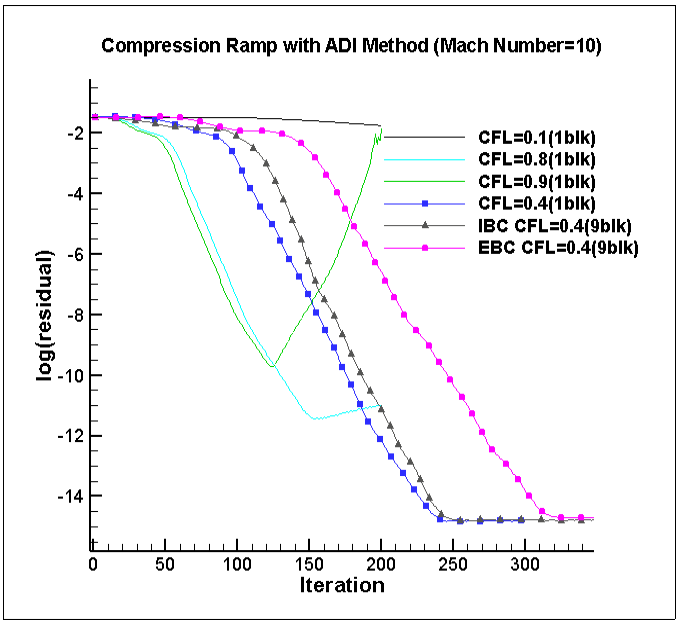


Figure 4-2 Residual History of the Compression Ramp Analysis with ADI Method

Figure 4-2 shows the residual history of the analyses for the Compression Ramp with ADI method. From the above figure, it can be said that the optimum CFL number is 0.4 for compression ramp with ADI method. Thanks to the implementation of the multi-block implicit boundary conditions, errors accumulated on block interfaces caused by multi-block topology is eliminated. Therefore, single-block grid and 9-block grid with implicit boundary condition on block interfaces shows similar

convergence characteristics. On the other hand, for 9-block grid with explicit boundary conditions convergency suffers because of the error introduced by omitting the coefficients on block interfaces.

Table 4-1 Total Run time for Convergence with ADI Method for Compression Ramp

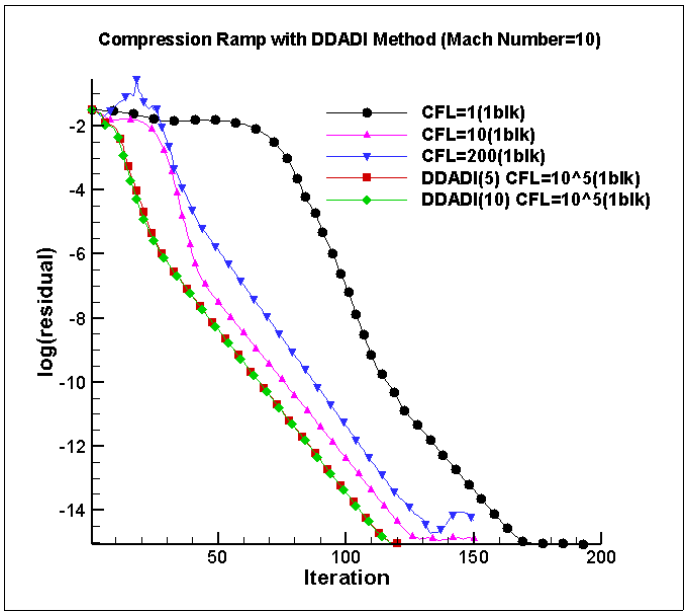
AF Methods	Total run time(s) for convergence	Total run time(s)	μs per cell per iteration	Cell count	Total iteration	Redundant iterations
ADI (1 blk, CFL=0,4)	4,20	5,04	5,29	3168	300	50
EBC ADI(9blk, CFL=0,4)	0,81	1	0,86	3168	400	70
IBC ADI(9blk, CFL=0,4)	0,71	1,09	0,79	3168	400	150

Table 4-2 Achieved Speed-up Compared to the Single-block Analysis with ADI Method

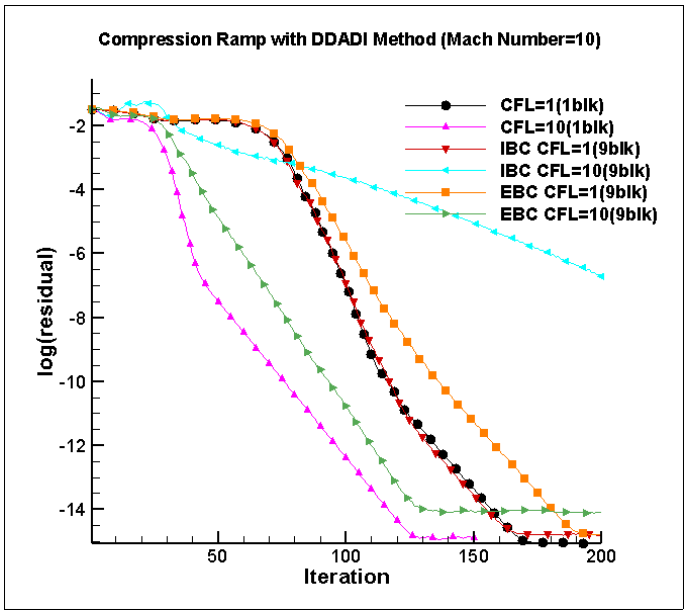
AF Methods	Speed-up
ADI (1 blk, CFL=0,4)	0%
EBC ADI(9blk, CFL=0,4)	81%
IBC ADI(9blk, CFL=0,4)	83%

In Table 4-1 total run times can be calculated by subtracting time wasted by redundant iterations from the total run time since the calculation time per cell per iteration is an output of the CFL3D open-source code. Single-block grid gives the slowest results because only one processor computes necessary calculations.

Splitting the single-block into 9 block and distributing the necessary calculations of the 9 blocks to 9 processors reduces the total run time significantly. In fact, implementing implicit boundary condition on block interfaces reduces the total run time further. Quickest solution can be obtained for ADI method with 9 block grid with implicit boundary condition on block interfaces. In Table 4-2, it is shown that by splitting the single block grid into 9-block grid %81 speed-up can be achieved. In fact, thanks to the implementation of the multiblock implicit boundary conditions, achieved speed-up can be increased up to %83.



a) Single-block



b) Multi-block

Figure 4-3 Residual History of the Compression Ramp Analysis with DDADI Method

In Figure 4-3 a) and b) residual history of the Compression Ramp with the DDADI method. Figure 4-3 a) shows the convergence characteristics of single-block grid while Figure 4-3 b) shows the convergence characteristics of multi-block grid, and also compares the residual history of the single-block grid with the residual history of the multi-block grids. For single-block grid, DDADI algorithm allows higher CFL numbers compared to the ADI and SFDT methods. Without Huang's sub-iteration correction maximum allowable CFL number is 200. However, the optimum CFL number for this case is 10. Since the sub-iterations eliminate the factorization error, the maximum CFL number increases with the use of sub-iterations up to 10^5. Therefore, the DDADI method with Huang's sub-iteration correction shows the best convergence characteristics. 9-block grid with implicit boundary conditions on block interfaces shows the same convergence characteristic as the single-block grid at the same CFL number while convergence slows down for 9 block grid with explicit boundary conditions. However, the maximum allowable CFL number decreases with multi-block implicit boundary conditions. In addition, there is not much difference

between the convergence characteristic of the single-block grid and 9 block grid, because omitted coefficients on the block interfaces are small compared to the diagonal coefficients for diagonal dominant relaxation schemes.

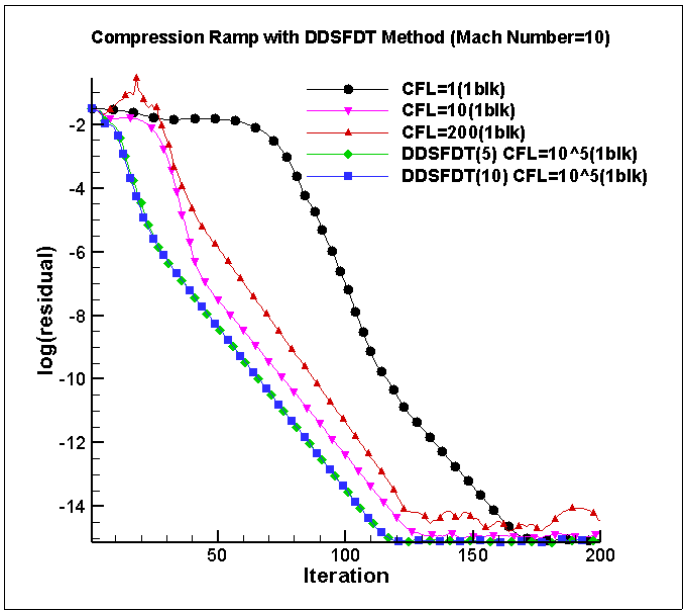
Table 4-3 Total Run time for Convergence with DDADI Method for Compression Ramp

AF Methods	Total run time(s) for convergence	Total run time (s)	μs per cell per iteration	Cell count	Total iteration	Redundant iterations
DDADI (1blk,CFL=10)	4,60	5,53	11,63	3168	150	25
DDADI(5) (1blk,CFL=10)	12,59	13,14	34,54	3168	120	5
DDADI(10) (1blk,CFL=10)	14,95	15,62	42,09	3168	120	5
EBC DDADI (9blk,CFL=10)	0,72	1,16	1,83	3168	200	75
DDADI (1blk,CFL=1)	8,09	9,59	15,78	3168	200	30
EBC DDADI (9blk,CFL=1)	1,26	1,26	1,98	3168	200	0
IBC DDADI (9blk,CFL=1)	1,76	2,21	4,73	3168	200	30

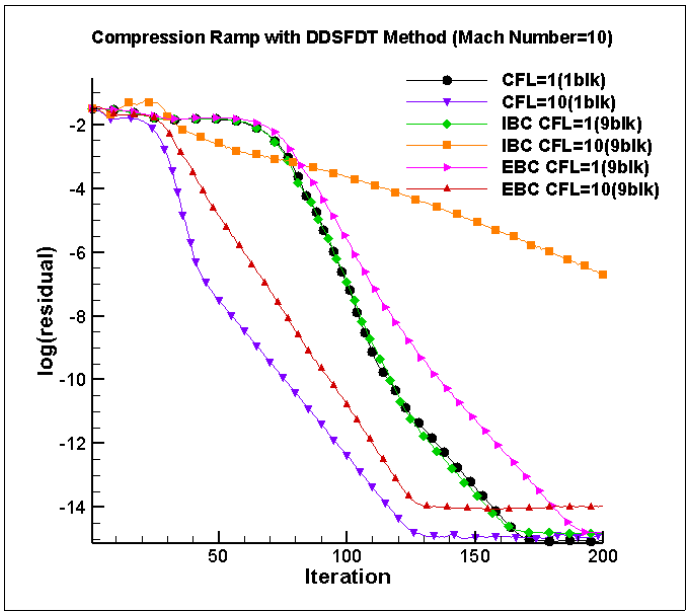
Table 4-4 Achieved Speed-up Compared to the Single-block Analysis with DDADI Method

AF Methods	Speed-up
DDADI (1blk,CFL=10)	0%
DDADI(5) (1blk,CFL=10)	-173%
DDADI(10) (1blk,CFL=10)	-224%
EBC DDADI (9blk,CFL=10)	84%
DDADI (1blk,CFL=1)	-76%
EBC DDADI (9blk,CFL=1)	73%
IBC DDADI (9blk,CFL=1)	62%

Table 4-3 shows the total run time for convergence of DDADI method for Compression Ramp geometry. Single-block grids have the highest run time as expected. Total run time increase even furher with sub-iteration for single-block grids even though sub-iteration reduces the required iteration number for convergencent solution by eliminating the factorization error. Multi-block implicit boundary conditions slows down the convergency for DDADI method. 9 block grid with explicit boundary conditions has the highest convergence rates because convergency of the diagonal dominant algorithm does not suffer from omitting the a and c coefficient of the Jacobian on block interfaces since they are small compared to the diagonal elements. In Table 4-4, it is shown that by splitting the single block grid into 9-block grid %84 speed-up can be achieved. However, sub-iteration correction and multiblock implicit boundary condition could not increase the convergence rate. Hence speed-up percentages are negative for DDADI with sub-iterations and multi-block implicit boundary conditions.



a) Single-block



b) Multi-block

Figure 4-4 Residual History of the Compression Ramp Analysis with DDSFDT Method

Figure 4-4 shows the residual history of the DDSFDT method for the Compression Ramp. Convergence characteristic of the DDSFDT method is very similar to the

DDADI method. DDSFDT with Huang's sub-iteration correction has the lowest iteration number for convergency. 9-block grid with multi-block implicit boundary conditions shows similar convergence characteristic to single-block grid at equal CFL numbers. However, the maximum allowable CFL number drops in general.

Table 4-5 Total Run time for Convergence with DDSFDT Method for Compression Ramp

AF Methods	Total run time(s) for convergence	Total run time (s)	μs per cell per iteration	Cell count	Total iteration	Redundant iterations
DDSFDT (1blk,CFL=10)	6,48	9,98	15,78	3168	200	70
DDSFDT(5) (1blk,CFL=10)	13,42	22,62	36,3	3168	200	80
DDSFDT(10) (1blk,CFL=10)	24,74	41,55	66,29	3168	200	80
EBC DDSFDT (9blk,CFL=10)	0,94	1,65	3,16	3168	200	70
DDSFDT (1blk,CFL=1)	8,12	9,62	15,78	3168	200	30
EBC DDSFDT (9blk,CFL=1)	1,61	1,61	3,16	3168	200	0
IBC DDSFDT (9blk,CFL=1)	2,71	3,19	5,04	3168	200	30

Table 4-6 Achieved Speed-up Compared to the Single-block Analysis with DDSFDT Method

AF Methods	Speed-up
DDSFDT (1blk,CFL=10)	0%
DDSFDT(5) (1blk,CFL=10)	-107%
DDSFDT(10) (1blk,CFL=10)	-282%
EBC DDSFDT (9blk,CFL=10)	85%
DDSFDT (1blk,CFL=1)	-25%
EBC DDSFDT (9blk,CFL=1)	75%
IBC DDSFDT (9blk,CFL=1)	58%

In Table 4-5 total run time for convergency with DDSFDT method for Compression Ramp is tabulated. The total run time for convergency values are similar to DDADI since the same diagonal dominant relaxation scheme is used. The quickest solution can be obtained from the 9-block grid with explicit boundary conditions on block interfaces by setting the optimum CFL number as 10. In Table 4-6, it is shown that by splitting the single block grid into 9-block grid %85 speed-up can be achieved. However, sub-iteration correction and multiblock implicit boundary condition could not increase the convergence rate. Hence speed-up percentages are negative for DDSFDT with sub-iterations and multi-block implicit boundary conditions.

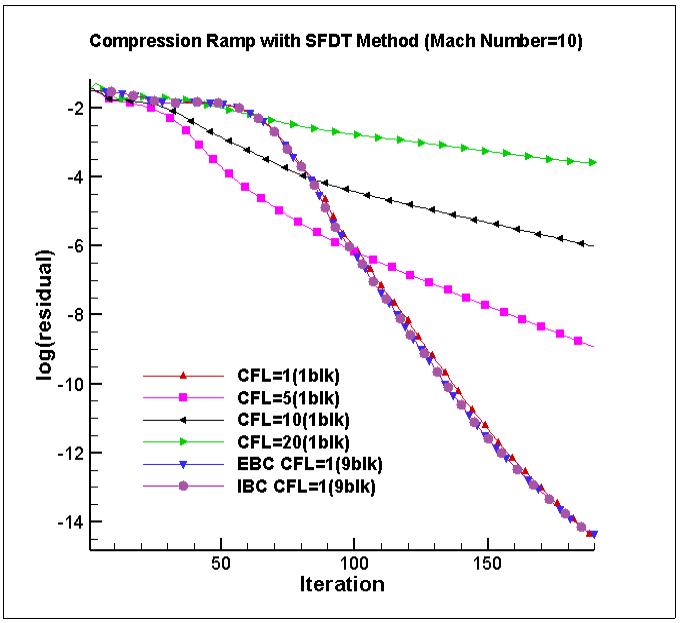


Figure 4-5 Residual History of the Compression Ramp Analysis with SFDT Method

Figure 4-5 shows the residual history of the SFDT method for the Compression Ramp. From the figure, it can be said that the optimum CFL number is 1. A higher CFL number slows down the convergency for the SFDT method. Single-block grid, 9-block grid with explicit boundary condition and implicit boundary condition shows the same convergence characteristics. Since the SFDT method adds the time terms to the diagonals of the tridiagonal matrices, the SFDT method is already sufficiently diagonal dominant. Because of this situation, omitted coefficients on block interfaces do not cause a big difference between single-block grids and multi-block grids. Hence, multi-block implicit boundary conditions can not make a big difference and not necessary for compression ramp cases.

Table 4-7 Total Run time for Convergence with SFDT Method for Compression Ramp

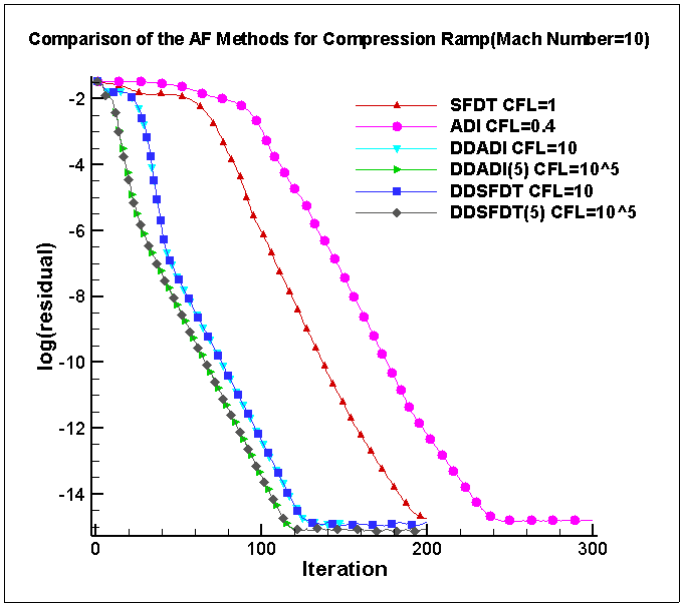
AF Methods	Total run time(s) for convergence	Total run time (s)	µs per cell per iteration	Cell count	Total iteration	Redundant iterations
SFDT (1blk,CFL=1)	2,22	2,22	3,49	3168	200	0
EBC SFDT (9blk,CFL=1)	0,72	0,72	1,13	3168	200	0
IBC SFDT (9blk,CFL=1)	0,73	0,73	1,58	3168	200	0

Table 4-8 Achieved Speed-up Compared to the Single-block Analysis with SFDT Method

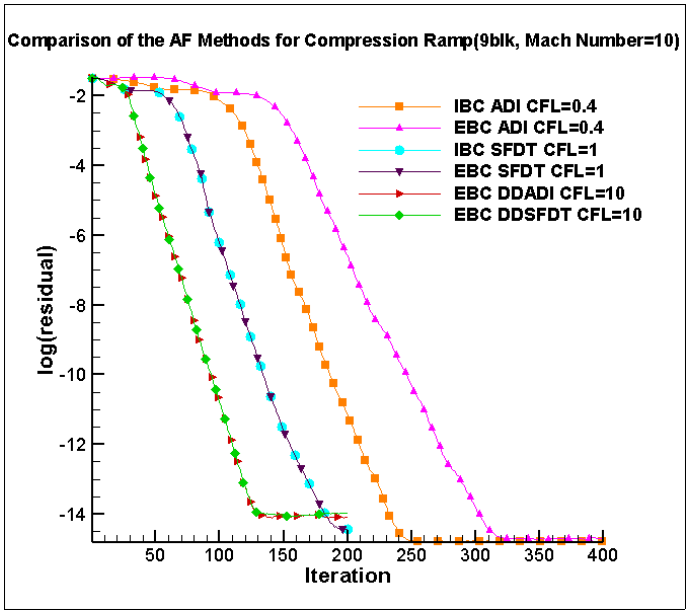
AF Methods	Speed-up
SFDT (1blk,CFL=1)	0%
EBC SFDT (9blk,CFL=1)	68%
IBC SFDT (9blk,CFL=1)	67%

The total run time for convergency with the SFDT method is tabulated in Table 4-4. Single-block grid has the slowest convergence rate as expected. Multi-block grids have similar convergence rate. Because of the additional communication between processors in order to implement implicit boundary conditions on block interfaces, required time per iteration is a little higher with multi-block implicit boundary conditions, but the difference is 0,01 seconds for 200 iteration. The quickest result can be obtained from 9 block grid with explicit boundary conditions. In Table 4-6, it is shown that by splitting the single block grid into 9-block grid %68 speed-up can be achieved. Multi-block implicit boundary conditions with 9-block grid speed-up

the solution %67 which is lower than the speed-up of the explicit boundary conditions.



a) Single-block



b) Multi-block

Figure 4-6 Comparison of the Residual Histories of the AF Methods for Compression Ramp

Figure 4-6 shows comparisons of the residual histories of the approximate factorization methods for both single-block grids and 9-block grids. From the figure, 4-6 a) it can be understood that for single-block grids residuals drops to sufficient levels within fewer iteration number with DDADI and DDSFDT methods compared to the ADI and SFDT methods. ADI methods require the highest iteration number for convergency. The iteration number drops further with Huang's sub-iteration correction DDADI and DDSFDT methods. From the figure, 4-6 b) it can be said that for 9 block grids DDADI and DDSFDT have better convergence characteristics compared to the ADI and SFDT methods as it is observed for single-block grids. It is observed that the ADI method poses with multi-block implicit boundary condition. The optimum and allowable CFL numbers of the DDADI and DDSFDT methods are equal and 10 and 200 respectively. Allowable CFL numbers of DDSFDT and DDADI methods are quite high compared to the SFDT and ADI methods which are 1 and 0,4 respectively. Since for ADI method error introduced on block interfaces for multi-block grids is higher compared to the other approximate factorization methods, implementation of the multi-block implicit boundary condition algorithm is quite efficient for the ADI method.

Table 4-9 Comparisons of Total Run Time for Convergence of AF Methods for Compression Ramp

AF Methods	Total run time(s) for convergence	Total run time (s)	µs per cell per iteration	Cell count	Total iteration	Redundant iterations
IBC ADI (9blk,CFL=0,4)	0,71	1,09	0,79	3168	400	150
EBC DDADI (9blk,CFL=10)	0,72	1,16	1,83	3168	200	75
EBC DDSFDT (9blk,CFL=10)	0,95	1,65	3,16	3168	200	70
EBC SFDT (9blk,CFL=1)	0,72	0,72	1,13	3168	200	0

Table 4-10 Achieved Speed-up Compared to the 9-block Grid Analysis with ADI Method

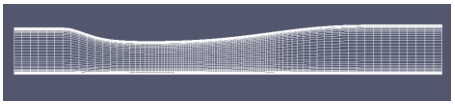
AF Methods	Speed-up
IBC ADI (9blk,CFL=0,4)	0%
EBC DDADI (9blk,CFL=10)	-1%
EBC DDSFDT (9blk,CFL=10)	-33%
EBC SFDT (9blk,CFL=1)	-1%

The best configurations of the approximate factorization methods in terms of total run time are tabulated in the above table 4-5. Since multi-block grids have better convergence rates compared to single-block grids, only 9-block grids results are tabulated. The quickest result can be obtained from the ADI method thanks to the multi-block implicit boundary conditions within 0,71 seconds. SFDT and DDADI methods have equal required run time for convergence and it is 0,72 seconds. DDSFDT has the slowest convergence rate for this case. Table 4-10 shows the speed-

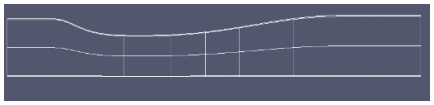
up percentages of the approximate factorization methods compared to the 9-block grid analysis with ADI method. It can be said that quickest solution can be obtained by ADI method with 9-block grid.

4.2 Sajben Transonic Diffuser (2D)

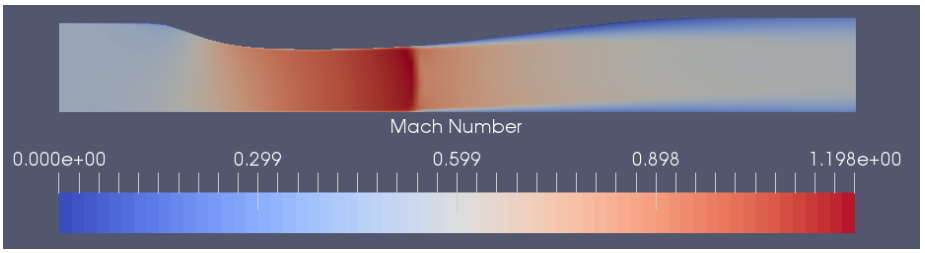
In the flow analysis of the Sajben Transonic Diffuser geometry [41], [42], unlike the compression ramp, the flow inside the Sajben Transonic Diffuser is determined to be viscous at wall boundaries. Spalart-Almaras turbulence model is used. Upwind-biased third order spatial differencing and flux-difference splitting of Roe's scheme is used for the flux calculations. The ratio of the entrance area to the throat area is settled as 1.4 and the ratio of the exit area to the throat area is 1.5. The total stagnation pressure at the inlet is determined as 19.5 psia, and the static pressure at the outlet is determined as 16.05 in order to obtain a weak shock in the throat. After the throat separation occurs and expansion of the flow cannot be obtained. Results are obtained using 24-block grid with 25 processors.



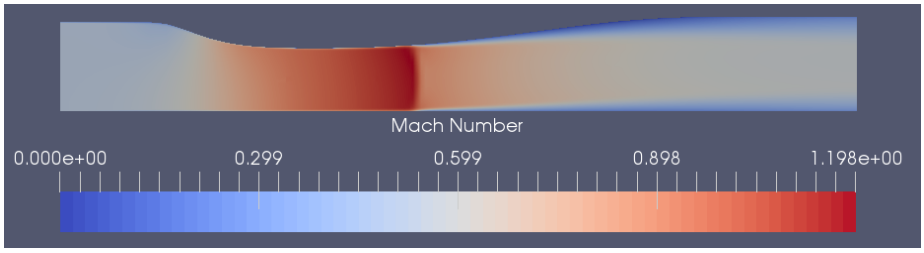
a) Single-block Grid



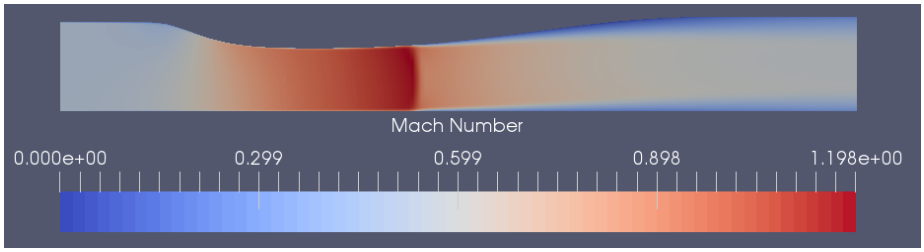
b) 24 Block Grid



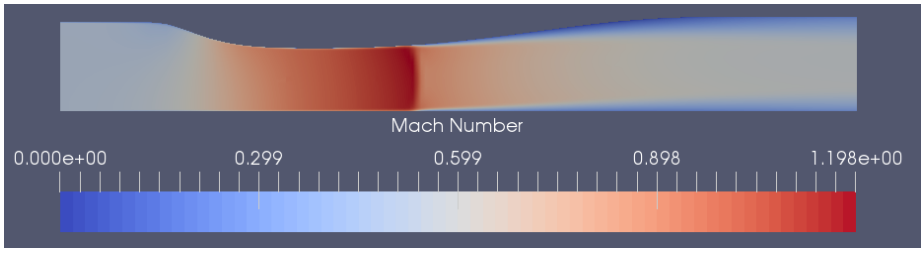
c) Mach Number (ADI)



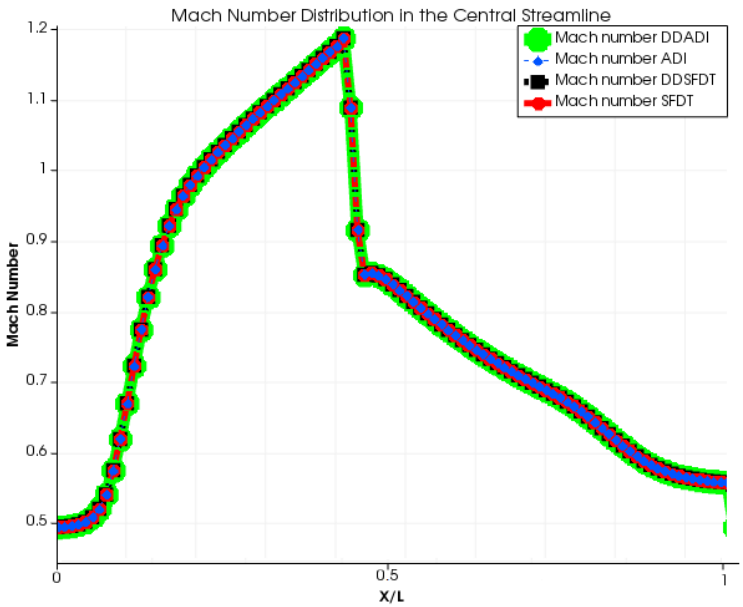
d) Mach Number (DDADI)



e) Mach Number (DDSFDT)



f) Mach Number (SFDT)



g) Mach Number Distribution in the Central Streamline

Figure 4-7 Generated Mesh, Contour Plots, and Pressure Coefficient Comparison Plot of the AF Methods for the Sajben Transonic Diffuser Geometry

Figure 4-7 a) and b) shows the generated mesh and 24 block grid of Sajben Transonic Diffuser geometry. Figure 4-7 c)-f) shows the Mach number distribution. From this figure 4-7 g) it can be said that all of the methods converge to a common solution. Weak shock is visible at the throat in all of the solutions obtained by different approximate factorization methods.

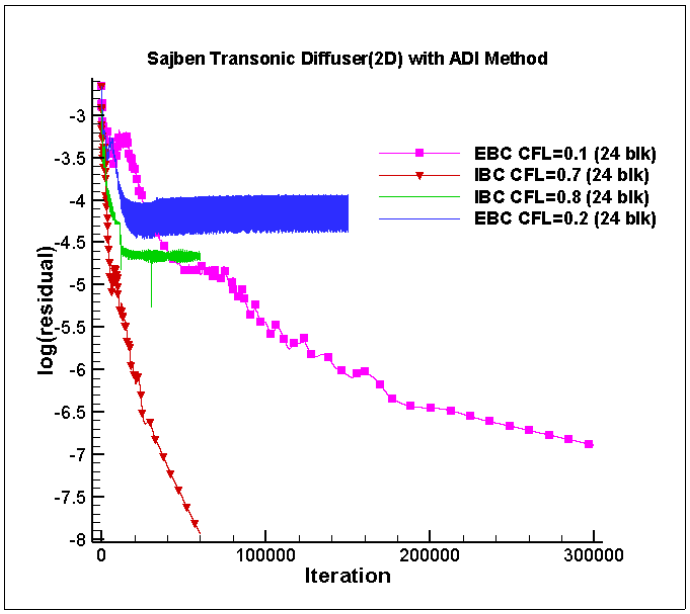


Figure 4-8 Residual History of the Sajben Transonic Diffuser Analysis with ADI Method

Figure 4-8 shows the convergence characteristics of the ADI method. Importance of the implementation of the multi-block implicit boundary conditions algorithm to the ADI solver can be clearly seen from the above figure. ADI with multi-block implicit boundary condition algorithm shows way better convergence characteristics compared to the ADI with multi-block explicit boundary conditions. Since the accumulated error between block interfaces quite a lot, convergence suffers with explicit boundary conditions. It is observed that ADI with explicit boundary conditions cannot converge at CFL number higher than 0.1. However, thanks to implicit boundary conditions on block interfaces maximum allowable CFL number increases to 0.7. Therefore, convergence rates considerable increases and solution can be obtained much more rapidly. Furthermore, lower residual levels can be achieved with the help of multi-block implicit boundary condition algorithm within a 60000 iteration. In contrast, ADI with explicit boundary conditions require 300000 iterations to converge to acceptable residual levels.

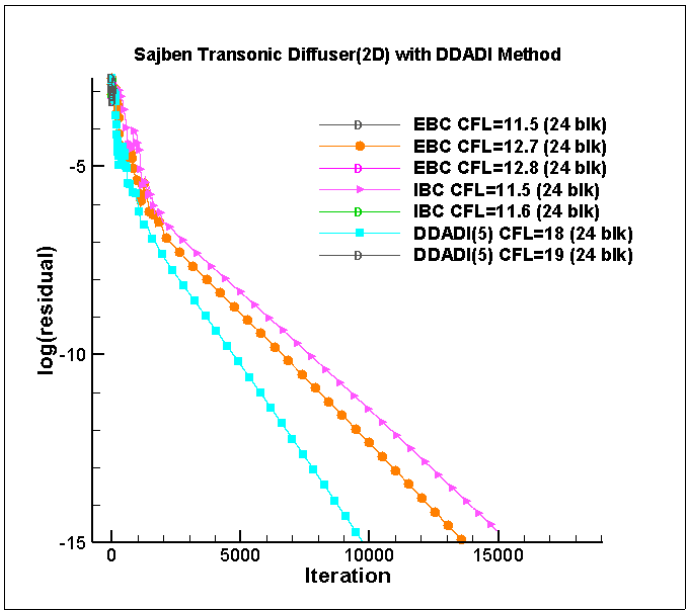


Figure 4-9 Residual History of the Sajben Transonic Diffuser Analysis with DDADI Method

Above figure 4-9 shows the residual histories of the analysis which are obtained from DDADI method. As expected DDADI(5) method which has Huang's sub-iteration correction converges to acceptable residual levels within least number of iteration. CFL number increases with the help of the elimination of the factorization error thanks to the sub-iterations. Maximum allowable CFL number is 18 for DDADI(5) while maximum allowable CFL number is 12.7 for DDADI method. With the implementation of the multi-block implicit boundary conditions algorithm convergency slows down for the Sajben Transonic Diffuser analysis and maximum allowable CFL number drops from 12.7 to 11.5. As it is mentioned in Ref. before implementation of the multi-block implicit boundary conditions only to Navier-Stokes equations has adverse effect. Therefore, turbulence model equations should be coupled with Navier-Stokes equations and should be included in the implicit boundary conditions in order to achieve convergence acceleration.

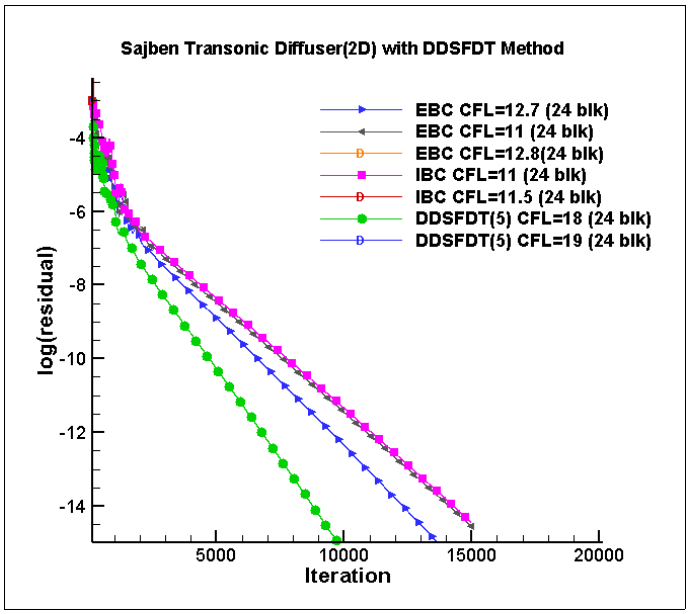


Figure 4-10 Residual History of the Sajben Transonic Diffuser Analysis with DDSFDT Method

Figure 4-10 demonstrate the convergence characteristics of the DDSFDT method for Sajben Transonic Diffuser analysis. From the above figure, it is observed that DDSFDT shows quite similar characteristics with DDADI method. Similarly, DDSFDT methods shows the best convergence characteristics in terms of iteration number thanks to the sub-iterations. DDSFDT(5) method converges to acceptable residual levels within 11500 while the other methods converges within 15000 iteration. Maximum allowable CFL numbers of the DDSFDT with explicit boundary condition, DDSFDT with implicit boundary condition and DDSFDT(5) with sub-iterations methods are 12.7, 11.0 and 18 respectively. Multi-block implicit boundary conditions has negative effect on convergency because maximum CFL number decreases. With the same CFL number DDSFDT with implicit and explicit boundary conditions shows similar convergence characteristic. Hence, it can be said that implementation of the multi-block implicit boundary conditions is unnecessary for DDSFDT method if Navier-stokes equation is not coupled with turbulence model equations [16].

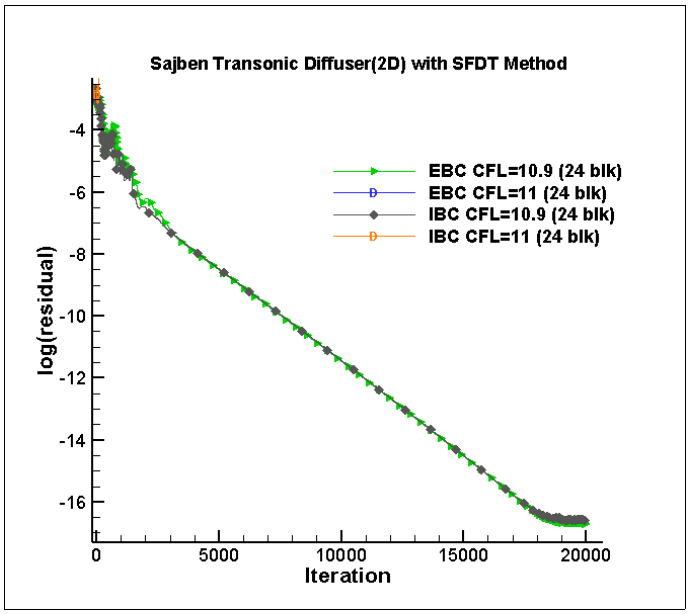


Figure 4-11 Residual History of the Sajben Transonic Diffuser Analysis with SFDT Method

Figure 4-11 shows the residual history of the SFDT method for Sajben Transonic Diffuser analysis. From the figure it can be understood that SFDT method with explicit and implicit boundary conditions show the exactly the same convergence characteristics. Since the method is sufficiently diagonal dominant, omitted coefficients cannot affect the convergency of the solution for this case. Maximum allowable CFL number does not change with the implementation of the multi-block implicit boundary conditions. Each method converges within 18000 iterations with CFL number 10.9. Because of the additional communication and calculation that multi-block implicit boundary conditions algorithm cause, explicit boundary conditions works fine and it is unnecessary to implement multi-block implicit boundary conditions algorithm.

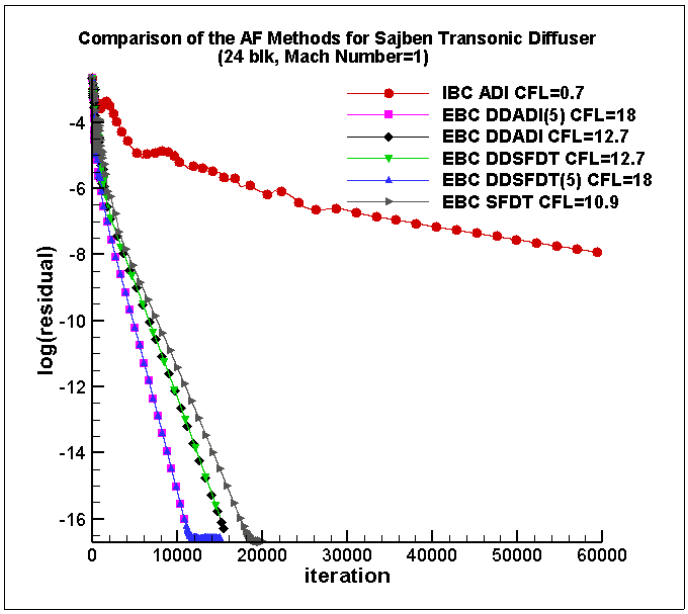


Figure 4-12 Comparison of the Residual Histories of the AF Methods for Sajben Transonic Diffuser

Convergence characteristics of the approximate factorization methods compared in figure 4-12. It is obvious that ADI method has the slowest convergence rate despite the implementation of the multi-block implicit boundary conditions. SFDT method has way better convergence characteristics compared to the ADI method, but DDADI and DDSFDT converges to the reasonable residual level within a fewer number of iteration. Sub-iterations of DDADI(5) and DDSFDT(5) methods reduces the number of iteration even further. Highest CFL number which is 18 can be applicable for DDADI(5) and DDSFDT(5) methods. As mentioned before iteration number is not the only value that gives information about convergence rates and efficiency of the approximate factorization method. Total run time of the analyses has also significant importance. In the following table total run time of the approximate factorization methods will be discussed.

Table 4-11 Comparisons of Total Run Time for Convergence of AF Methods for Sajben Transonic Diffuser

AF Methods	Total run time(s) for convergence	Total run time (s)	μs per cell per iteration	Cell count	Total iteration	Redundant iterations
EBC ADI (24 blk, CFL=0,1)	721	74	0,39	6144	300000	0
IBC ADI (24 blk, CFL=0,7)	358,8	180	0,97	6144	60000	0
EBC DDADI (24blk, CFL=12,7)	60	60	0,63	6144	15500	0
DDADI(5) (24blk,CFL=18)	99	99	1,49	6144	10800	0
EBC DDSFDT (24blk, CFL=12,7)	59	59	0,64	6144	15000	0
DDSFDT(5) (24blk,CFL=18)	95,8	141	1,53	6144	15000	4800
EBC SFDT (24 blk, CFL=10,9)	43,7	46	0,37	6144	20000	1000
IBC SFDT (24 blk, CFL=10,9)	44,6	47	0,38	6144	20000	1000

Table 4-12 Achieved Speed-up Compared to the 24-block Multi-block Explicit Boundary Conditions Analysis with ADI Method

AF Methods	Speed-up
EBC ADI (24 blk, CFL=0,1)	0%
IBC ADI (24 blk, CFL=0,7)	50,2%
EBC DDADI (24blk, CFL=12,7)	91,7%
DDADI(5) (24blk,CFL=18)	86,3%
EBC DDSFDT (24blk, CFL=12,7)	91,8%
DDSFDT(5) (24blk,CFL=18)	86,7%
EBC SFDT (24 blk, CFL=10,9)	93,9%
IBC SFDT (24 blk, CFL=10,9)	93,8%

Total run time for convergence of the approximate factorization methods are tabulated in above table 4-11. From the table it can be said that ADI method with explicit boundary conditions on block interfaces is considerably slow compared to the other methods. However, multi-block implicit boundary conditions significantly accelerate the convergence rate of the ADI method. DDADI method and DDSFDT methods have similar convergence rates and gives the solution within 60 seconds. DDADI(5) and DDSFDT(5) methods has better convergence characteristics in terms of iteration, but required time for convergence is higher compared to the DDADI and DDSFDT methods because of the sub-iterations. Although SFDT methods converges within higher iteration number compared to the DDADI and DDSFDT methods, quickest results can be obtained from the SFDT method. Since the diagonal dominancy is sufficient for Sajben Transonic Diffuser case, there is no need to use multi-block implicit boundary conditions, because implicit boundary conditions algorithm requires additional calculation and communications. From the table 4-12 it is observed that all of the methods speed-up the solution compared to the ADI method with 24-block grid. Highest speed-up can be achieved with multi-block explicit boundary condition and SFDT method which is %93,9.

4.3 Double Wedge (2D)

Hypersonic flow over a Double Wedge with a sharp edge is studied. The angle of the first ramp and second ramp is adjusted as 9 and 20.5 degrees respectively as shown in figure 4-13. Mach number, Reynolds number, and free stream temperature is settled as $M = 8.3, Re = 3.76 * 10^6, T = 102\ K$ respectively. Spalart-Almaras turbulent model is used. Upwind-biased third order spatial differencing and flux-difference splitting of Roe's scheme are used for the flux calculations. Single-block grid is generated and split into 24 blocks and solved with 25 processors. In addition, it should be noted that with each method, after 100 iteration flow solver ramps the CFL number to 4 times of it.

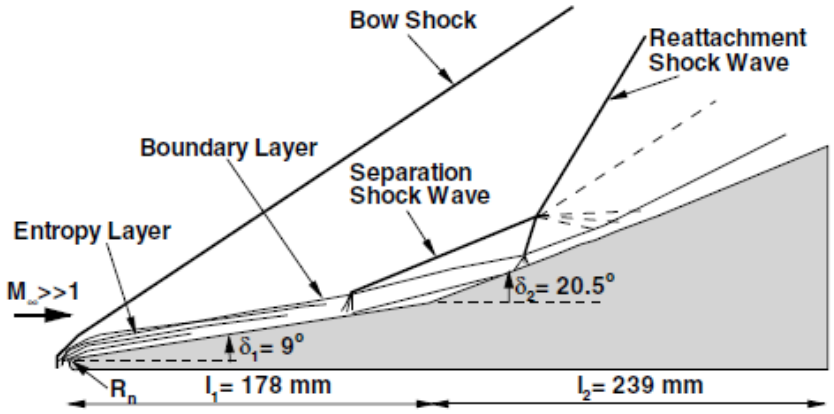
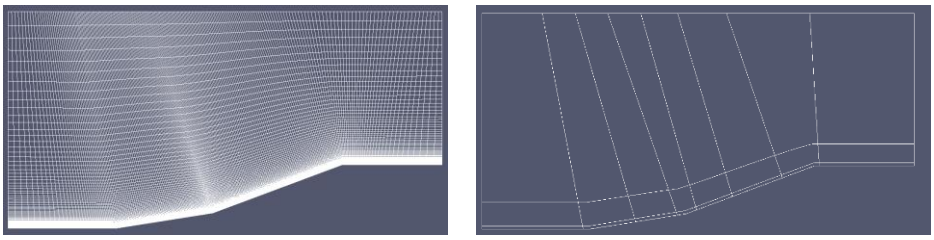
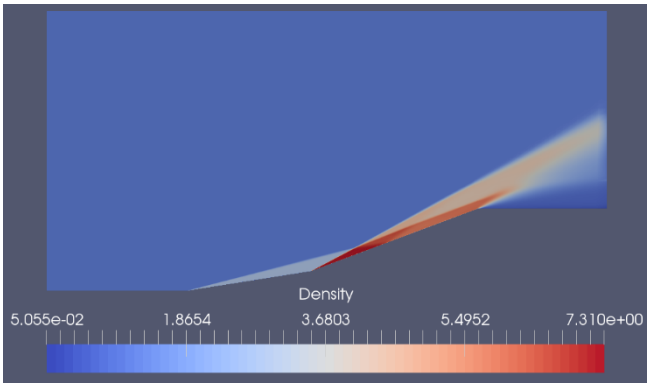


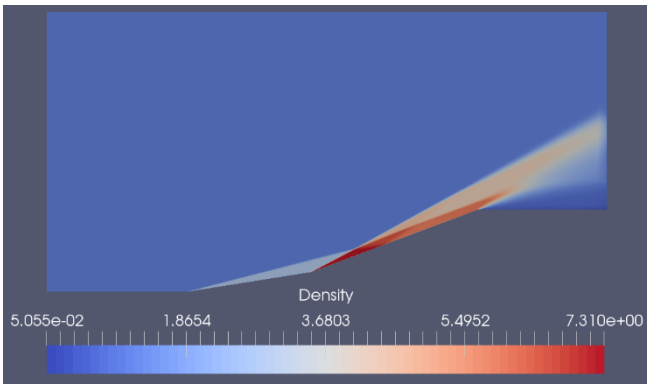
Figure 4-13 Double Wedge Configuration with Sharp Edge[43], [44]



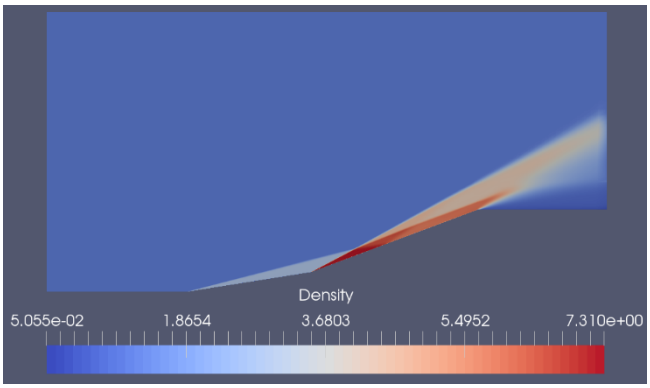
a) Single-block Grid b) 24 Block Grid



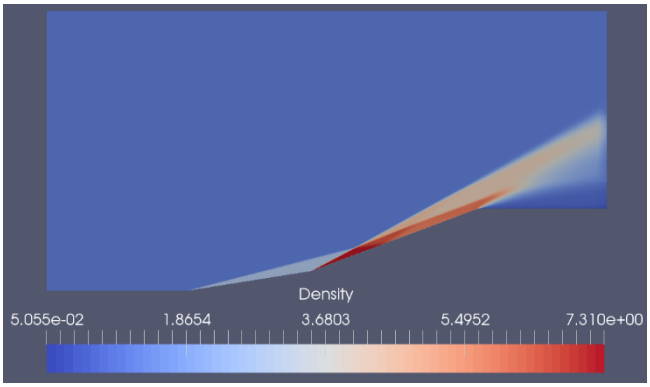
c) Density Distribution (ADI,IBC)



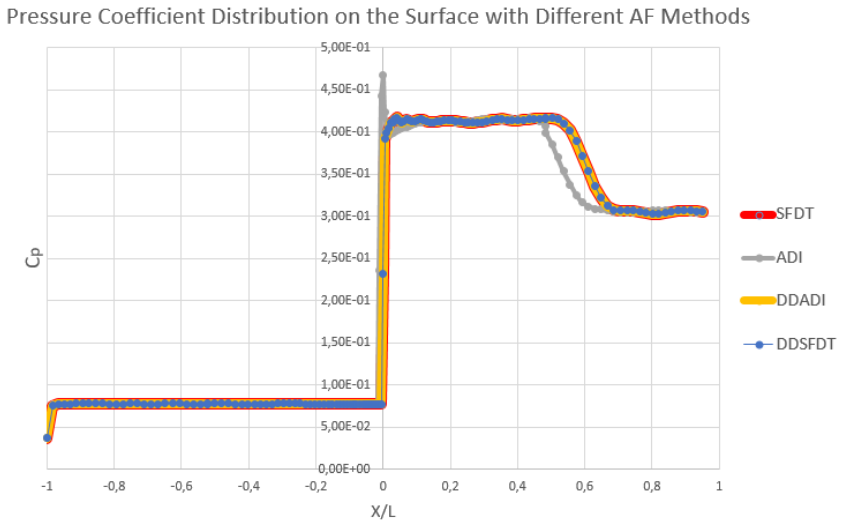
d) Density Distribution (DDADI)



e) Density Distribution (DDSFDT)



f) Density Distribution (SFDT)



g) Pressure Coefficient Distribution with Different AF Methods

Figure 4-14 Generated Mesh, Contour Plots, and Pressure Coefficient Comparison Plot of the AF Methods for the Double Wedge Geometry

Single-block grid is generated and split in to 24 blocks as shown in figure 4-14 a) and b). Density distribution contours are compared in figure 4-14 e)-f). From the pressure contour plots and figure 4-14 g) it can be understood that all of the methods converge to a common solution. Shock and boundary layer can be seen between two ramp and it seems this interaction cause quite high pressures. Densities are nondimensionalized with free stream density which is settled as 1.

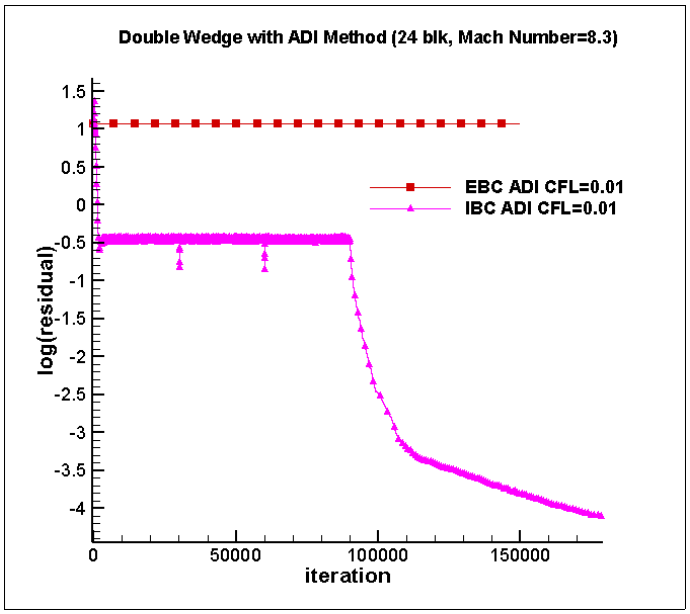


Figure 4-15 Residual History of the Double Wedge Analysis with ADI Method

Figure 4-15 illustrates the convergence characteristics of the ADI method for Double wedge geometry. It is observed that the ADI method with explicit boundary conditions on block interfaces cannot converge to a solution even after 150000 iterations with CFL number 0.01. ADI method with multi-block implicit boundary conditions poorly converges a solution with CFL number 0.01. Residual levels drop to order of -4 and this requires 180000 iterations. Therefore, the obtained solution may not be reliable.

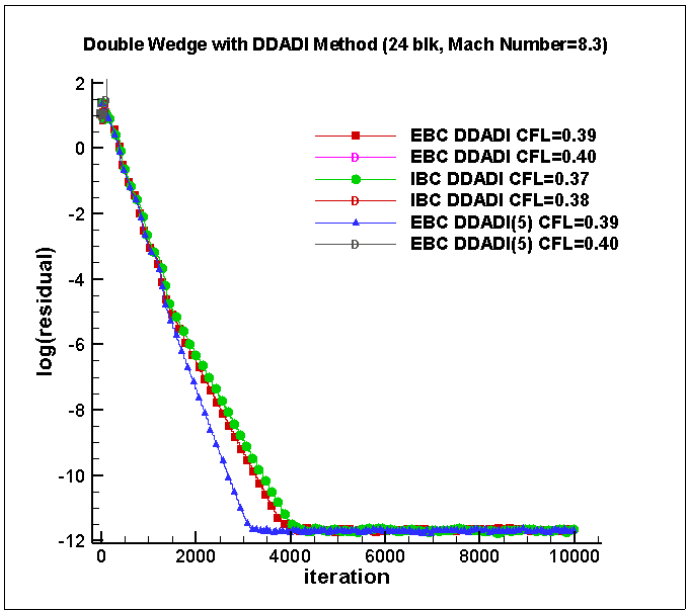


Figure 4-16 Residual History of the Double Wedge Analysis with DDADI Method

Above figure 4-16 shows the residual history of the DDADI method for Double wedge Geometry. From the figure it can be understood that DDADI(5) with sub-iteration requires the lowest iteration number for convergence. Maximum allowable CFL numbers are equal for DDADI and DDADI(5) and it is 0.39. Implementation of the multi-block implicit boundary conditions reduces the maximum allowable CFL number from 0.39 to 0.37. However, for the Double Wedge case it can be said that multi-block explicit boundary condition and implicit boundary conditions show very similar convergence characteristics.

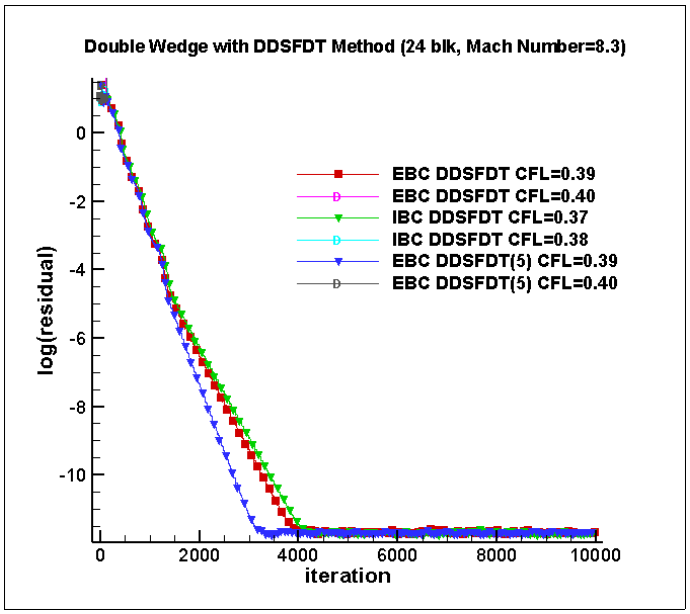


Figure 4-17 Residual History of the Double Wedge Analysis with DDSFDT Method

Figure 4-17 demonstrates the convergence characteristics of the DDSFDT method for Double Wedge geometry. Maximum allowable CFL numbers and convergence characteristics are very similar to the DDADI method in this case. The maximum allowable CFL number of DDSFDT and DDSFDT methods is 0.39. The DDADI method converges within 4000 iterations while DDSFDT(5) converges within 3000 iterations. The DDSFDT method with multi-block implicit boundary conditions has maximum allowable CFL number as 0.37 and shows similar convergence characteristics to DDSFDT with multi-block explicit boundary conditions.

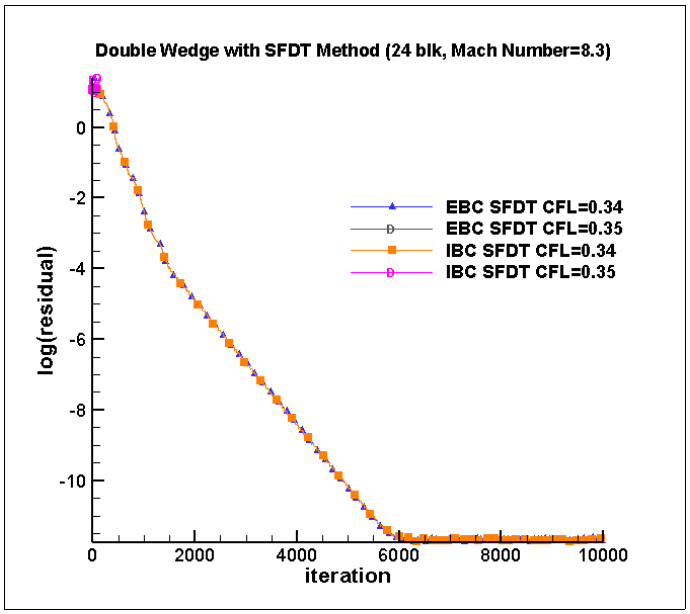


Figure 4-18 Residual History of the Double Wedge Analysis with SFDT Method

Above figure 4-18 shows convergence characteristics of the SFDT method for Double Wedge geometry. Maximum allowable CFL number is determined that 0.34 for both SFDT with multi-block explicit and implicit boundary conditions. Both methods converge at the same iteration number and shows exactly the same convergence characteristics. This means omitted coefficients at block interfaces are too small to affect the convergency. Therefore, because of the additional calculation and communication that comes with the multi-block implicit boundary condition treatment, using multi-block explicit boundary condition is more reasonable in order not to waste time with unnecessary calculations and communications. This situation shows that SFDT method is sufficiently diagonal dominant and stable.

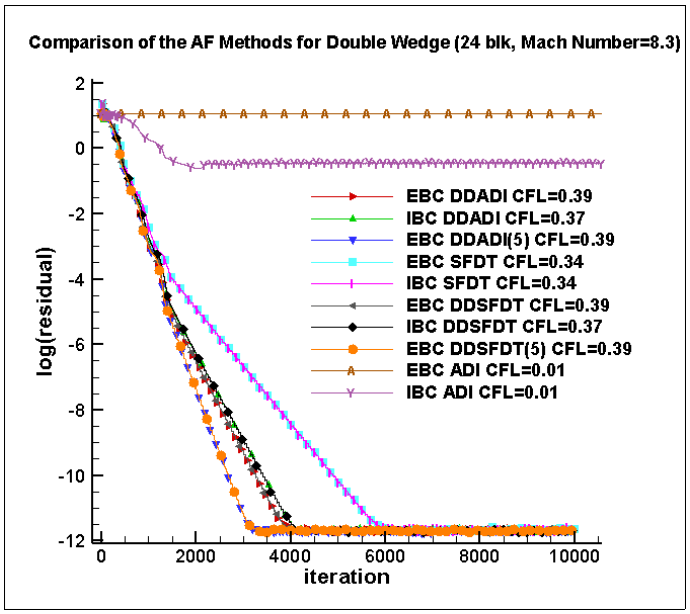


Figure 4-19 Comparison of the Residual History of the AF Methods for Double Wedge

Figure 4-19 demonstrates the comparisons of the residual histories of the approximate factorization methods. DDSFDT(5) and DDADI(5) shows the similar convergence characteristics and converge within 3000 iteration. DDADI, DDSFDT and multi-block implicit boundary conditions versions of each method has approximate convergence rates in terms of iteration number. SFDT method shows slower convergence characteristics compared to the DDADI and DDSFDT methods. ADI with multi-block explicit boundary conditions cannot converge and ADI with multi-block implicit boundary conditions requires the highest iteration number compared to the other approximate factorizations methods. However, residual histories do not tell the complete story about convergence rates of the methods. Therefore, it is crucial to check total run times for convergence of the methods.

Table 4-13 Comparisons of Total Run Time for Convergence of AF Methods for Double Wedge

AF Methods	Total run time(s) for convergence	Total run time (s)	μs per cell per iteration	Cell count	Total iteration	Redundant iterations
EBC DDADI (24 blk, CFL=0,39)	89	223	1,04	21504	10000	6000
IBC DDADI (24 blk, CFL=0,37)	136	340	1,58	21504	10000	6000
DDADI(5) (24 blk,CFL=0,39)	154	512	2,38	21504	10000	7000
EBC DDSFDT (24 blk, CFL=0,39)	96	239	1,11	21504	10000	6000
IBC DDSFDT (24 blk, CFL=0,37)	134	337	1,57	21504	10000	6000
DDSFDT(5) (24 blk, CFL=0,39)	153	508	2,36	21504	10000	7000
IBC SFDT (24blk, CFL=0,34)	75	126	0,59	21504	10000	4000
EBC SFDT (24blk, CFL=0,34)	74	123	0,57	21504	10000	4000

Table 4-14 Achieved Speed-up Compared to the 24-block Multi-block Explicit Boundary Conditions Analysis with DDADI Method

AF Methods	Speed-up
EBC DDADI (24 blk, CFL=0,39)	0%
IBC DDADI (24 blk, CFL=0,37)	-53%
DDADI(5) (24 blk,CFL=0,39)	-73%
EBC DDSFDT (24 blk, CFL=0,39)	-8%
IBC DDSFDT (24 blk, CFL=0,37)	-51%
DDSFDT(5) (24 blk, CFL=0,39)	-72%
IBC SFDT (24blk, CFL=0,34)	15%
EBC SFDT (24blk, CFL=0,34)	17%

Total run time for convergence of the approximate factorization methods are tabulated in above table 4-13. ADI method with multi-block explicit boundary conditions cannot converge and multi-block implicit boundary conditions converges poorly and may not be reliable. DDADI(5) and DDSFDT(5) methods with sub-iteration gives the solution within a 154 seconds which is slower than the DDADI, DDSFDT and SFDT methods. Total run time for convergence of DDADI and DDSFDT methods is around 90 seconds while SFDT methods converges within 75 seconds. Therefore, it can be said that quickest solution can be obtained with SFDT method for Double Wedge geometry and multi-block implicit boundary conditions is not necessary for this case. For the table 4-14, since the ADI method cannot give convergent solution, multi-block explicit boundary conditions with DDADI method is chosen as baseline method for calculation of the speed-up percentages. It is observed that only SFDT method can achieve positive speed-up which is %17.

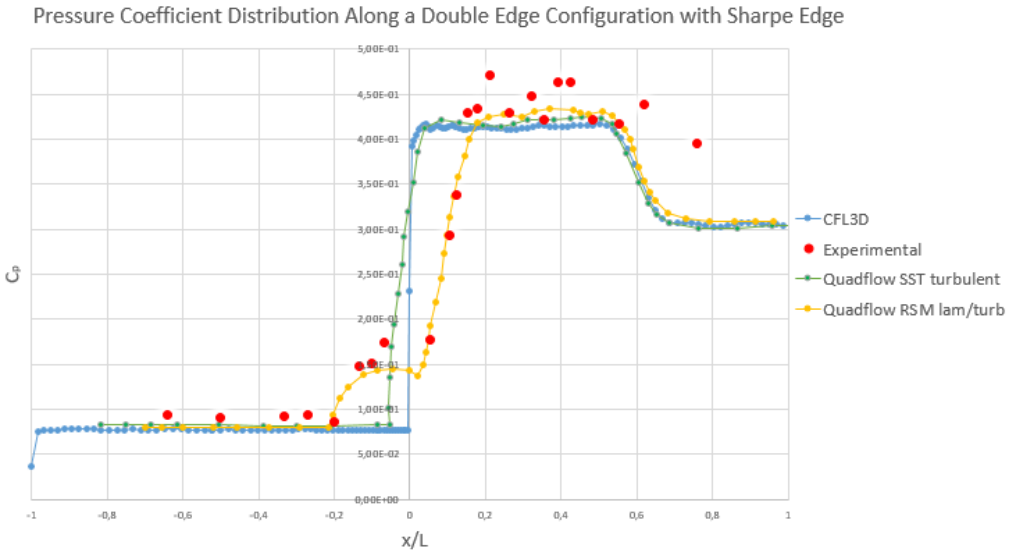


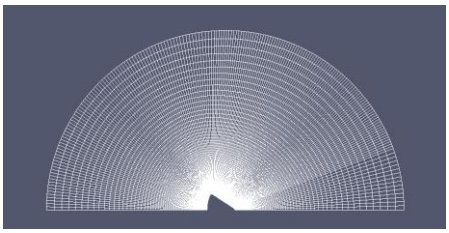
Figure 4-20 Comparison of the Pressure Coefficient Distribution of CFD and Experimental Results

Above Figure 4-20 shows the pressure coefficient distribution obtained from experiment and flow solvers. It is observed that for Double Wedge case it is crucial to simulate the laminar to turbulent transition region as accurate as possible. Reinartz and Ballmann[43], [44] have shown that in order to match the CFD solver results with the experimental results[45] certain changes must be done in the flow solver. They have seen that Reynolds Stress Model simulates the flow over Double Wedge more accurately compared to the other turbulence models. Therefore, it is expected to obtain such results from CFL3D that not perfectly matched with experimental results. However, in this study main goal is to examine the convergence characteristics of the approximate factorization methods and multi-block implicit boundary conditions, spatial accuracy is not the first concern.

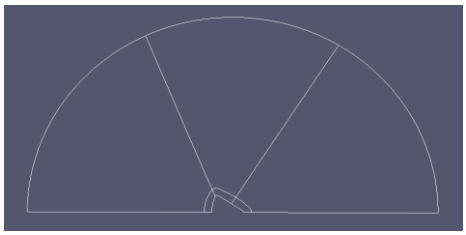
4.4 Apollo Command Module (2D)

Reentry conditions of the Apollo Command Module are simulated with 2-dimensional viscous grid. Flow conditions are matched with the experimental study

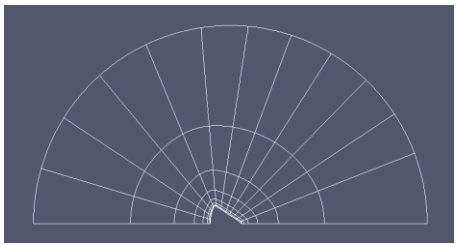
of NASA [46] in order to validate the flow solver. The angle of attack is 0 degree. Mach number is 10,18, and Reynold number is settled as $1 * 10^6$. Spalart-Almaras turbulent model is used. Upwind-biased third order spatial differencing and flux-difference splitting of Roe's scheme are used for the flux calculations. Single-block grid is split into 6 and 96 blocks in order to observe the effect of the increasing block number. 6-block grids are solved with 7 thread which are distributed to 7 processors while 96-block grids are solved with 97 threads which are distributed to 64 processors. In addition, it should be noted that with each method after 100 iteration flow solver ramps the CFL number to 4 times of it.



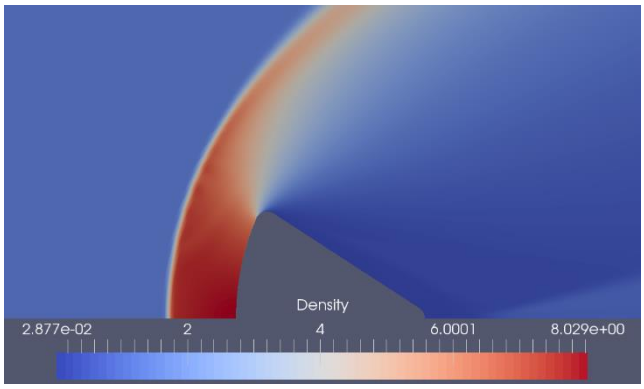
a) Single-block Grid



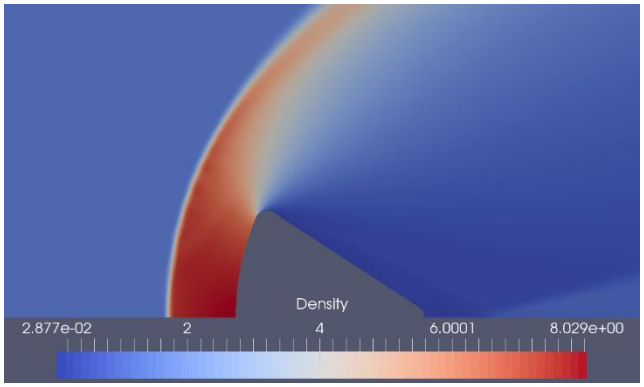
b) 6 Block Grid



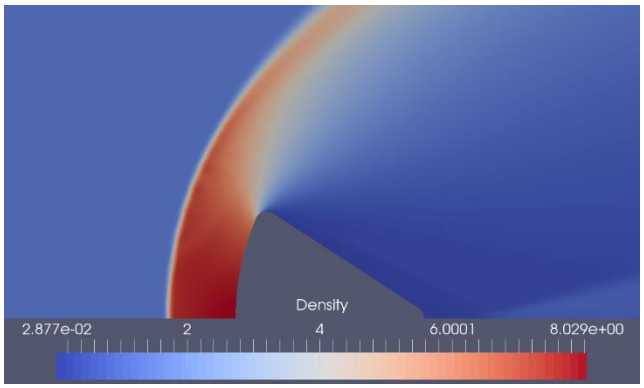
c) 96 Block Grid



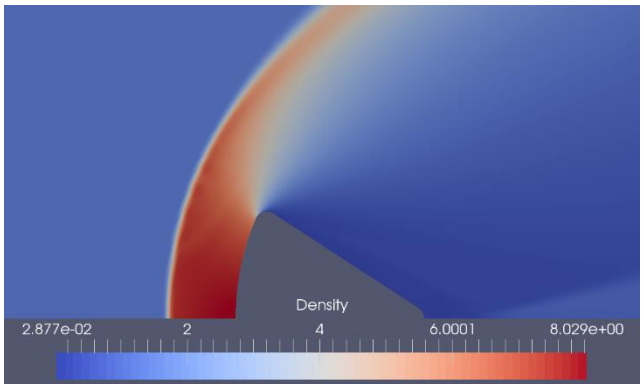
d) Density Distribution (ADI)



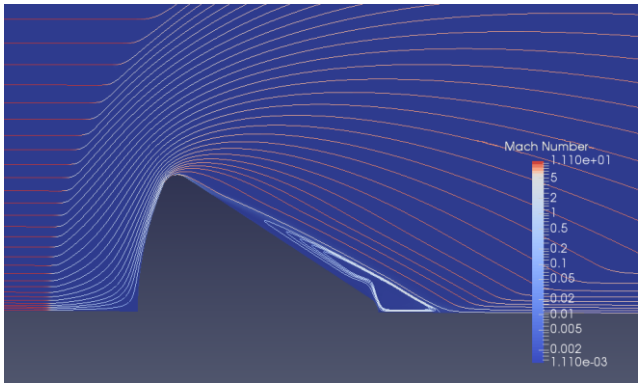
d) Density Distribution (DDADI)



d) Density Distribution (DDSFDT)



d) Density Distribution (SFDT)



i) Stream lines

Figure 4-21 Generated Mesh and Contour Plots of the Apollo Command Module (2D)

Above figure 4-21 includes generated 2D mesh for Apollo Command Module, split grids and contour plots of the analyses. Single-block grid shown in the figure 4-21 a) is split into 6 and 96 block as shown in the figure 4-21 b) and c). Figure 4-21 d) shows the temperature distribution which is obtained from SFDT method. Pressure distributions of the approximate factorization methods are compared in figure 4-21 e), f), g) and h). It is observed that all of the methods give similar solution. Figure 4-21 i) shows the stream lines and flow separations and reverse flow is visible at the shadow region of the capsule. Densities are nondimensionalized with free stream density which is settled as 1.

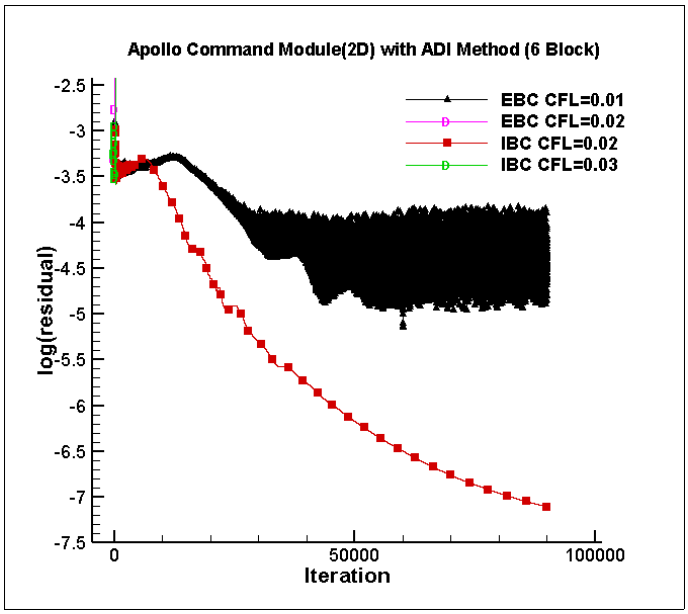


Figure 4-22 Residual History of the Apollo Command Module (2D) Analysis with ADI Method

Above figure 4-22 shows the residual history of the ADI method with explicit and implicit boundary conditions on block interfaces for 6-block grid. It can be observed from the figure introduced error of explicit boundary conditions on block interfaces cause oscillations on solution with CFL number 0.01. Higher CFL number such as 0.02 cause the divergence of the solution. Therefore, sufficiently convergent solution cannot be obtained from ADI method if explicit boundary condition is used on block interfaces. On the other hand, multi-block implicit boundary conditions algorithm eliminates the error on block interfaces and convergent solution can be obtained with optimum CFL number which is 0.02. Residual level of the solution can be reach to order of -7 , but this requires 90000 iterations.

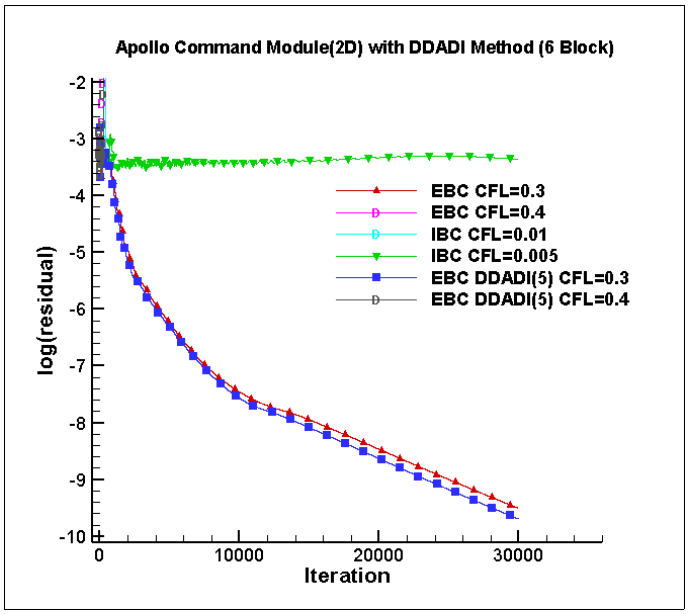


Figure 4-23 Residual History of the Apollo Command Module (2D) Analysis with DDADI Method

Figure 4-23 illustrates the residual history of the DDADI method for the Apollo Command Module with 6-block grid. The DDADI method with explicit boundary conditions converges to a sufficient residual level within 30000 iterations with CFL number 0.3 and diverges with higher CFL numbers. The sub-iteration correction could not increase the CFL number as it is observed from the compression ramp solutions and it can be said that the optimum CFL number for the DDADI method with 5 sub-iteration is also 0.3. In contrast to the ADI method, multi-block implicit boundary condition has an adverse effect on solutions with the DDADI method. It can be observed from the above figure that explicit boundary conditions on block interfaces converged to reasonable residual levels while the multi-block implicit boundary conditions cannot.

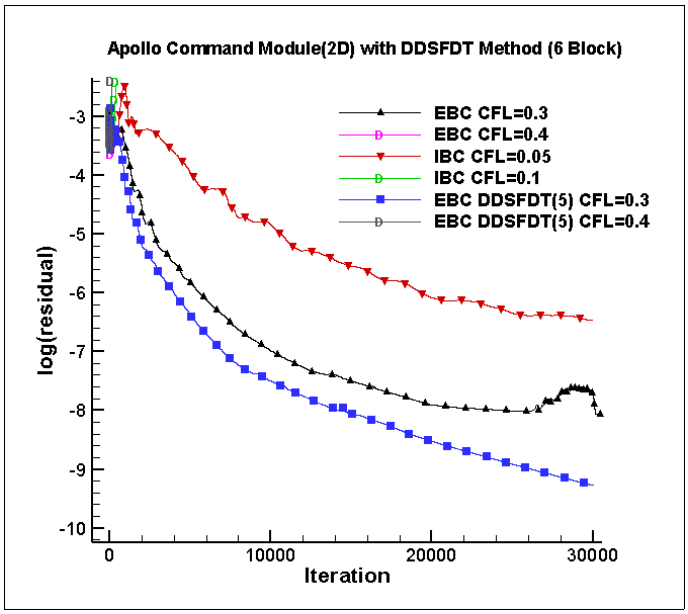


Figure 4-24 Residual History of the Apollo Command Module (2D) Analysis with DDSFDT Method

Above figure 4-24 demonstrate the residual history of the DDSFDT method for Apollo Command Capsule. As observed in the DDADI method, multi-block implicit boundary conditions algorithm slows down the convergence for the DDSFDT as well. However, in this case DDSFDT method does not shows the exactly the same convergence characteristics with the DDADI method. DDSFDT method with multi-block implicit boundary condition can converge up to order of -6 in contrast to the DDADI with multi-block implicit boundary conditions. In addition, sub-iteration correction has better effect on the DDSFDT method compared to the DDADI method. DDSFDT with sub-iteration correction can converge approximately 1 order lower residual levels compared DDSFDT. Optimum CFL numbers DDSFDT and DDSFDT with sub-iteration correction are 0.3 and equal to the optimum CFL numbers of the DDADI and DDADI with sub-iteration correction for Apollo Command Module.

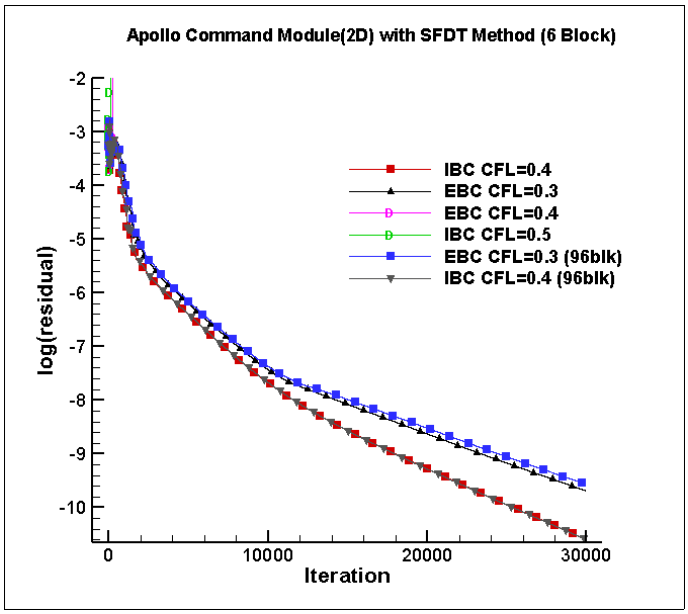


Figure 4-25 Residual History of the Apollo Command Module (2D) Analysis with SFDT Method

Figure 4-25 shows the convergence characteristics of the SFDT method for Apollo Command Module. It is found out that optimum CFL number for SFDT method with explicit boundary conditions is 0.3 while the optimum CFL number for SFDT methot with multi-block implicit boundary conditions is 0.4. Since increase in CFL number is achieved better convergence characteristics can be accomplished. With the help of the elimination of the error occurred in the block boundaries thank to the multi block boundary conditions, residual level can drop to the lower level within a fewer iteration number. Residual level of SFDT method with multi-block implicit boundary condition is lower than order of -10 while the residual level of the explicit boundary condition is order of -9 within 30000 iterations. Furthermore, while convergency slows down slightly with the increase in the number of blocks for SFDT with explicit boundary condition, convergence characteristics of the SFDT method remains unchanged thanks to the implementation of the implicit boundary conditions. Residual history lines of the 6 blocks grid and 96 blocks grid overlaps perfectly for multi-block implicit boundary conditions. In contrast, residual history

line of 96 blocks grid is slightly higher compared to the 6 blocks grid for explicit boundary conditions.

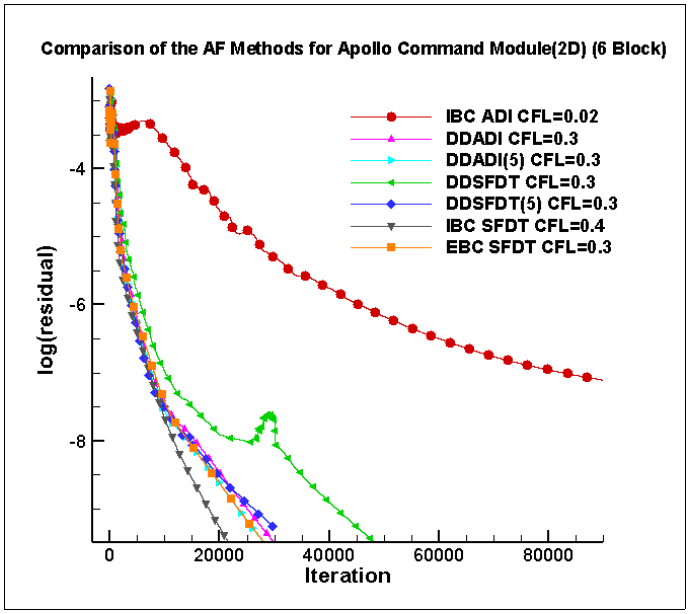


Figure 4-26 Comparison of the Residual Histories of the AF Methods for Apollo Command Module (2D)

Figure 4-26 shows the comparison of the convergence characteristics of the approximate factorization methods. As mentioned before ADI method with explicit boundary condition cannot converge sufficiently, and in order to obtain a convergent solution with ADI method with multi-block implicit boundary conditions, it requires 90000 iterations. Hence, from the figure it can be understood that ADI method with multi-block implicit boundary condition has the slowest convergence rates compared to the other approximate factorization methods. DDSFDT methods shows better convergence characteristics compared to the ADI methods, but has lower converge rates than DDSFDT with sub-iteration correction, DDADI and SFDT methods. DDSFDT with sub-iteration correction, DDADI with and without sub-iteration correction and SFDT method with explicit boundary condition methods shows approximate convergence characteristics. From the figure it is observed that SFDT

method with multi-block implicit boundary conditions has the highest allowable CFL number and shows the best convergence characteristics in terms of residual level and iteration number.

Table 4-15 Comparisons of Total Run Time for Convergence of AF Methods for Apollo Command Module (2D)

AF Methods	Total run time(s) for convergence	Total run time (s)	μs per cell per iteration	Cell count	Total iteration	Redundant iterations
IBC ADI (6 blk, CFL=0,02)	6039	6039	2,73	24576	90000	0
EBC DDADI (6blk,CFL=0,3)	3280	3280	4,45	24576	30000	0
DDADI(5) (6 blk, CFL=0,3)	8041	8041	10,91	24576	30000	0
EBC DDSFDT (6blk,CFL=0,3)	3875	3875	3,5	24576	45000	0
DDSFDT(5) (6 blk, CFL=0,3)	8515	8515	11,55	24576	30000	0
EBC SFDT (6 blk, CFL=0,3)	1604	1604	2,18	24576	30000	0
IBC SFDT (6 blk, CFL=0,4)	1103	1470	1,99	24576	30000	7500
EBC SFDT (96 blk, CFL=0,3)	192	192	0,26	24576	30000	0
IBC SFDT (96 blk, CFL=0,4)	159	212	0,29	24576	30000	7500

Table 4-16 Achieved Speed-up Compared to the 6-block Multi-block Implicit Boundary Conditions Analysis with ADI Method

AF Methods	Speed-up
IBC ADI (6 blk, CFL=0,02)	0%
EBC DDADI (6blk,CFL=0,3)	45,7%
DDADI(5) (6 blk, CFL=0,3)	-33,2%
EBC DDSFDT (6blk,CFL=0,3)	35,8%
DDSFDT(5) (6 blk, CFL=0,3)	-41,0%
EBC SFDT (6 blk, CFL=0,3)	73,4%
IBC SFDT (6 blk, CFL=0,4)	81,7%
EBC SFDT (96 blk, CFL=0,3)	96,8%
IBC SFDT (96 blk, CFL=0,4)	97,4%

The required total run time of the approximate factorization methods can be found in table 4-15. Although DDADI(5) and DDSFDT(5) with sub-iteration correction show better convergence characteristics in terms of iteration number compared to the DDADI and DDSFDT without sub-iteration correction, the required time for convergence is considerably higher because of the additional sub-iterations. ADI with multi-block implicit boundary conditions gives the converged solution quicker than DDADI(5) and DDSFDT(5) while ADI with explicit boundary conditions could not converge. The DDADI method has a lower required time compared to the DDSFDT for Apollo Command Module analysis. For this case, best convergence rates can be accomplished by using the SFDT method. In fact, the required time for convergence can be reduced further by implementing implicit boundary conditions on block interfaces. The quickest solution can be obtained with the SFDT method with multi-block implicit boundary conditions. Table 4-16 shows the speed-up percentages of the methods compared to the 6-block grid analysis with multi-block

implicit boundary condition with ADI method because multi-block explicit boundary conditions cannot give a convergent solution for ADI method in this case. For 6-block grids maximum speed-up can be obtained by SFDT method with multi-block implicit boundary conditions which is %81,7. The convergence rate of the solution can be increased further by increasing the block number from 6 to 96 blocks.

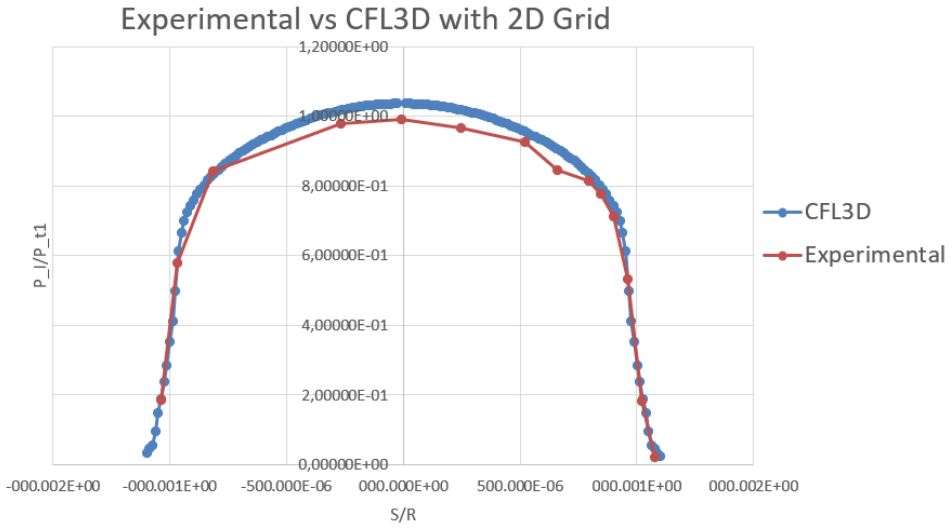
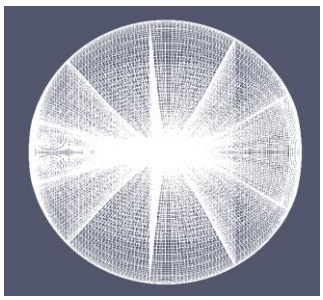


Figure 4-27 Comparison of the Experimental Results versus Solver Results for 2D Grid

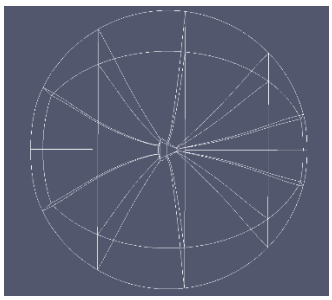
Figure 4-27 compares the pressure ratio distribution over the surface of the Apollo Command Module between experimental results[46] and solver results. From the figure it can be said that in front of the Apollo Command Module pressure is slightly over estimated. This situation is expected and can be overcome by using 3-dimensional grids.

4.5 Apollo Command Module (3D)

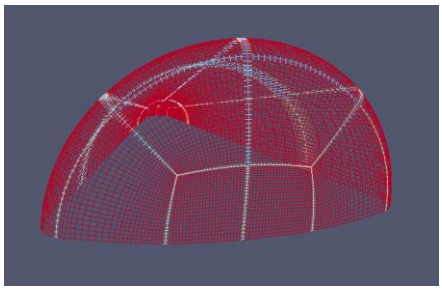
Reentry conditions of the Apollo Command Module are simulated with 3-dimensional viscous grid. Flow conditions are matched with the experimental study of NASA[46] in order to validate the flow solver. The angle of attack, Mach number, and Reynold number are settled as 0, 10,18, and $1 * 10^6$ respectively. Spalart-Almaras turbulent model is used. Upwind-biased third order spatial differencing and flux-difference splitting of Roe's scheme are used for the flux calculations. Single-block grid is split into 20 blocks and solved with 21 processors.



a) 20 Block Grid



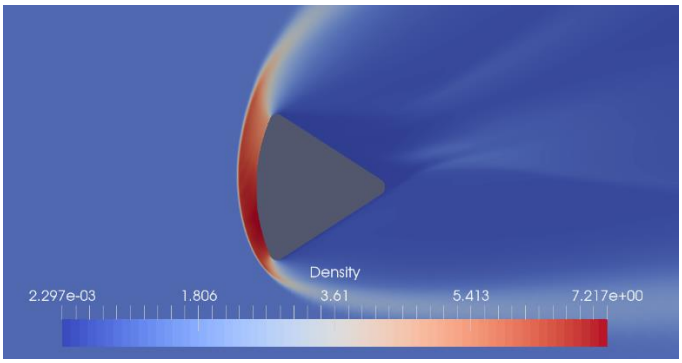
b) 20 Block Grid



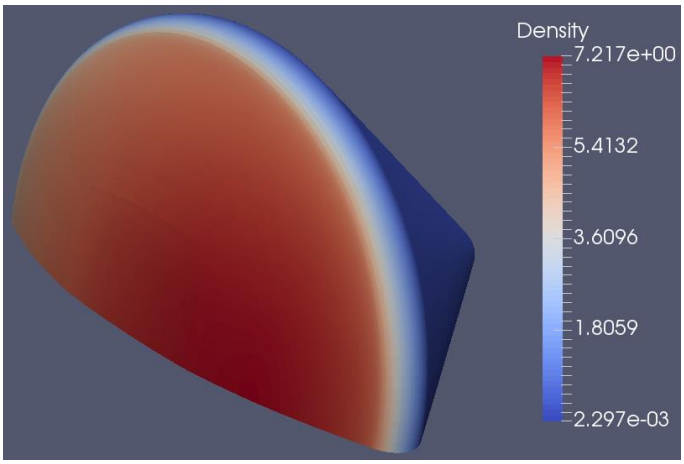
c) Surface Grid

Figure 4-28 Generated Mesh for the Apollo Command Module (3D)

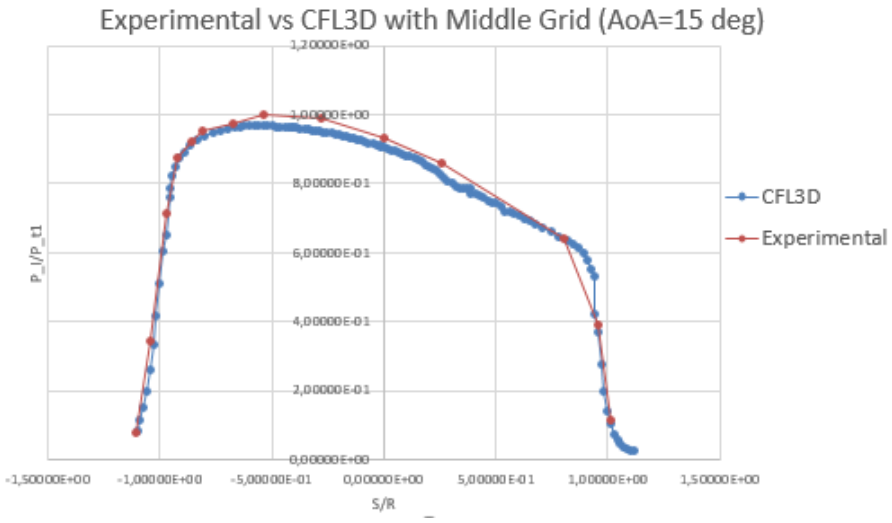
Figure 4-28 a) and b) show the 20 block grids of the Apollo Command Module and figure 4-28 c) shows the surface mesh. In order to obtain accurate results on the poles of the geometry, O-H type mesh topology which is mentioned in chapter 2 is used.



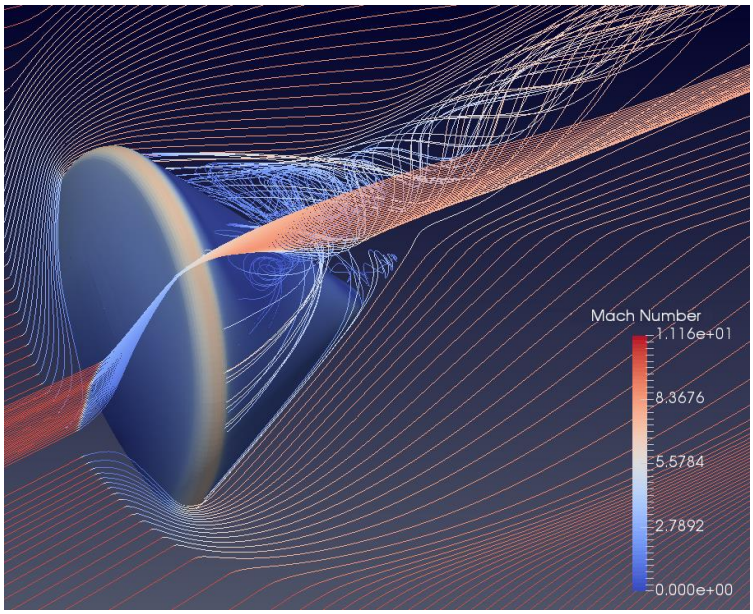
a) Density Distribution



b) Density Distribution on the Surface



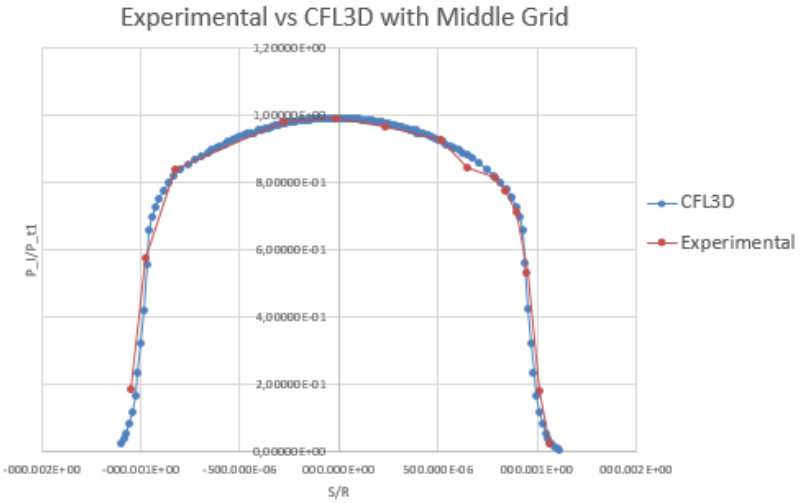
c) Comparisons of the Solver Results with Experimental Results



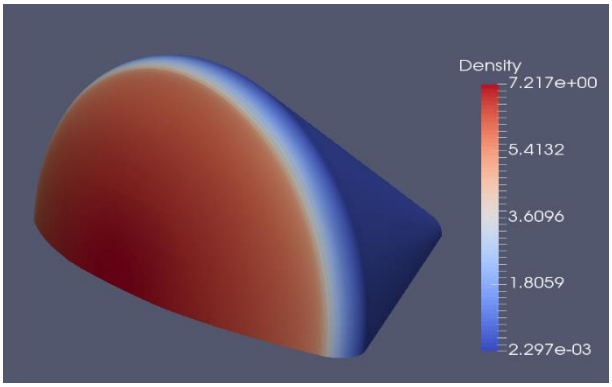
d) Mach Number Stream Lines

Figure 4-29 Contour Plots of the Apollo Command Module (3D) with Angle of Attack = 15^0

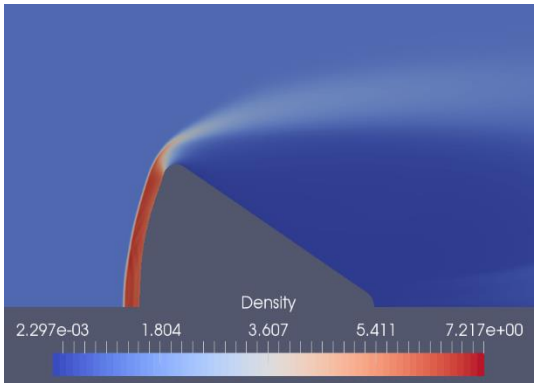
Figure 4-29 shows a) density distribution, b) surface density distribution of the flow field of the Apollo Command Module with angle of attack = 15^0. Densities are nondimensionalized with free stream density which is settled as 1. Figure 4-29 c) compares the solver results and experimental results for pressure ratios on the surface of the capsule. It can be said that on the high-pressure region solver lower estimates the pressure ratios, but in general solver results matches with the experimental result. Figure 4-29 d) shows the three-dimensional stream lines over the apollo geometry and flow separation is visible in the shadow region of the capsule.



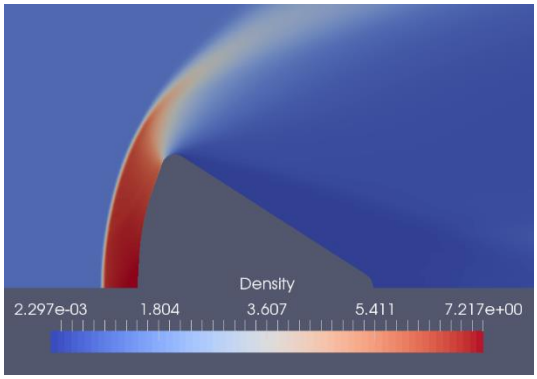
a) Comparisons of the Solver Results with Experimental Results at AoA=0 degree



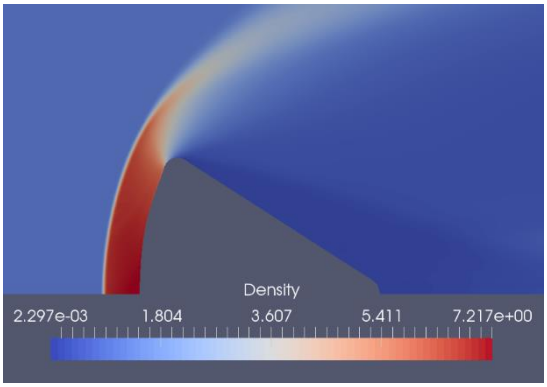
b) Surface Density Distribution



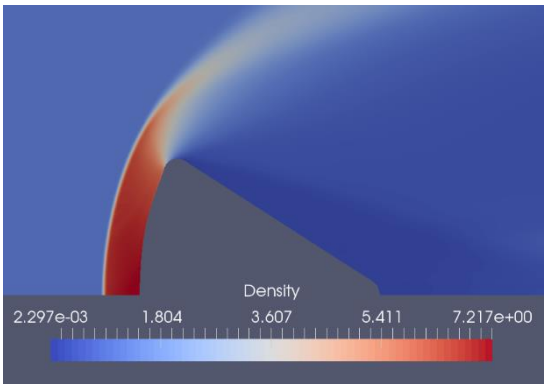
c) Density Distribution (ADI)



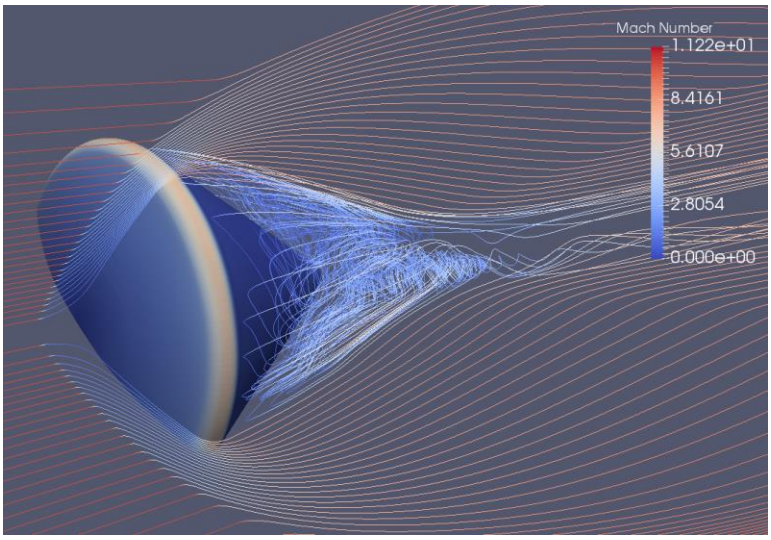
d) Density Distribution (DDADI)



e) Density Distribution (DDSFDT)



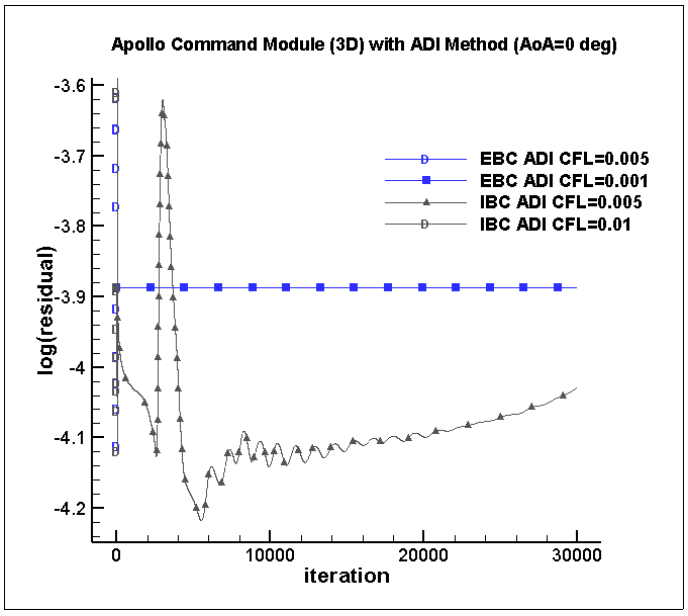
f) Density Distribution (SFDT)



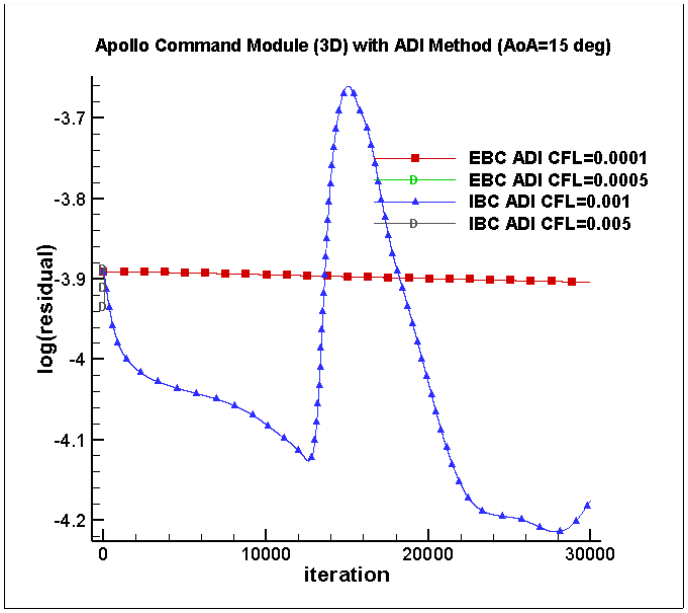
g) Mach Number Stream Lines

Figure 4-30 Contour Plots of the Apollo Command Module (3D) with Angle of Attack = 0^0

Figure 4-30 includes contour plots of the flow over Apollo Command Module analyses. Figure 4-30 a) shows the comparisons of the pressure ratios obtained from the solver and experimental results. It can be said that solver and experimental results are perfectly matched. Figure 4-30 b) shows density distribution on the surface, Density distributions of the approximate factorization methods are compared in figure 4-30 c)-f). It is observed that all of the methods give similar solution except ADI method. It should be noted that densities are nondimensionalized with free stream density which is settled as 1. Figure 4-30 g) shows the three-dimensional stream lines over the apollo geometry and flow separation is visible in the shadow region of the capsule.



a) Angle of Attack = 0^0

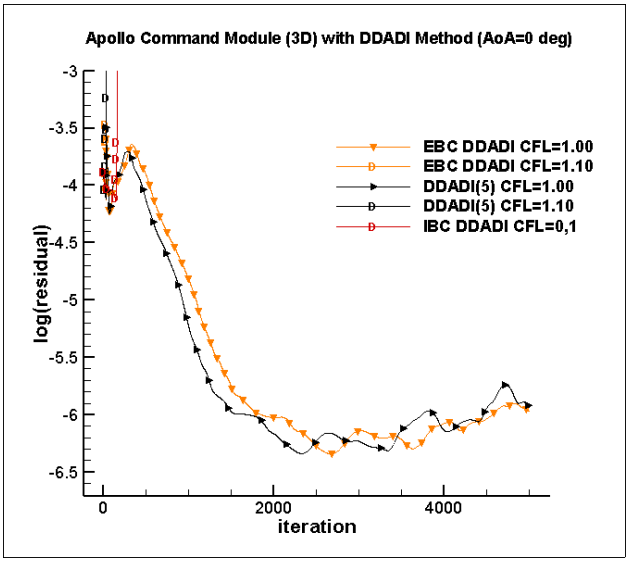


b) Angle of Attack = 15^0

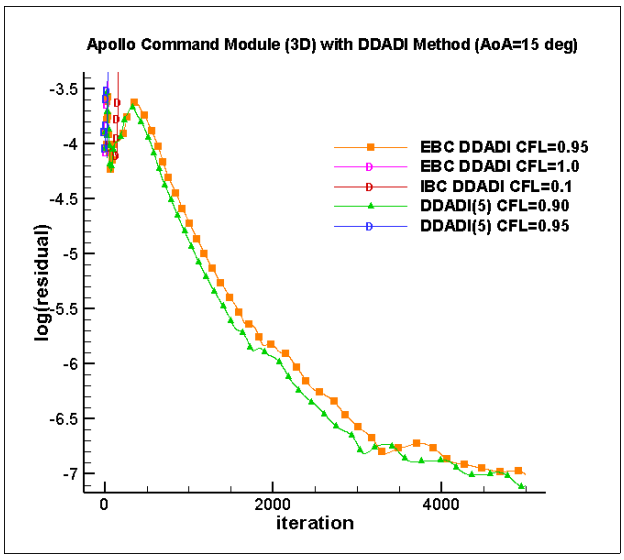
Figure 4-31 Residual History of the Apollo Command Module Analysis with ADI Method for Different Angle of Attacks

From the figure 4-31, it can be said that ADI method with or without multi-block implicit boundary condition allows quite low CFL numbers such as 0.001 for explicit boundary conditions and 0.05 for implicit boundary conditions with 0 degrees of angle of attack and CFL numbers decreases even further for 15 degrees of angle of

attack. Therefore, required time and iteration for convergence increases significantly. Factorization error of ADI methods increases for 3D cases compared to the 2D cases, because error terms mentioned in chapter 3 becomes quite large. Hence, convergency suffers a lot for Standard ADI method and convergent solution cannot be obtained within a reasonable iteration number for both different angle of attack cases.



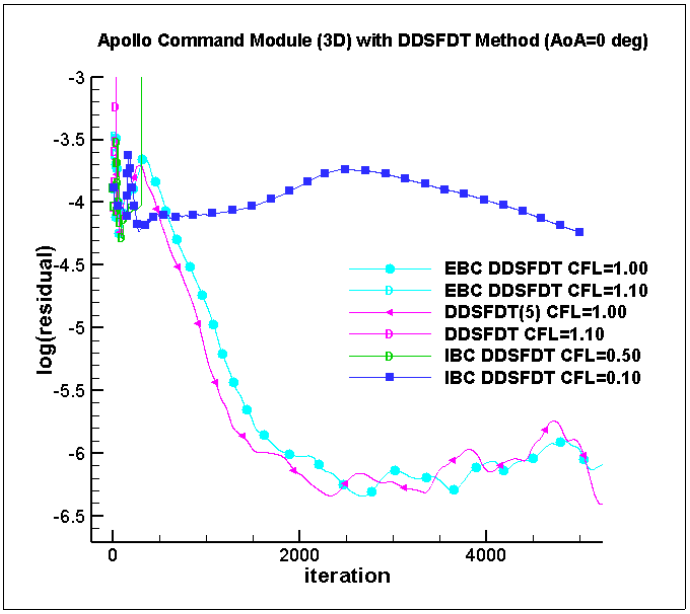
a) Angle of Attack = 0^0



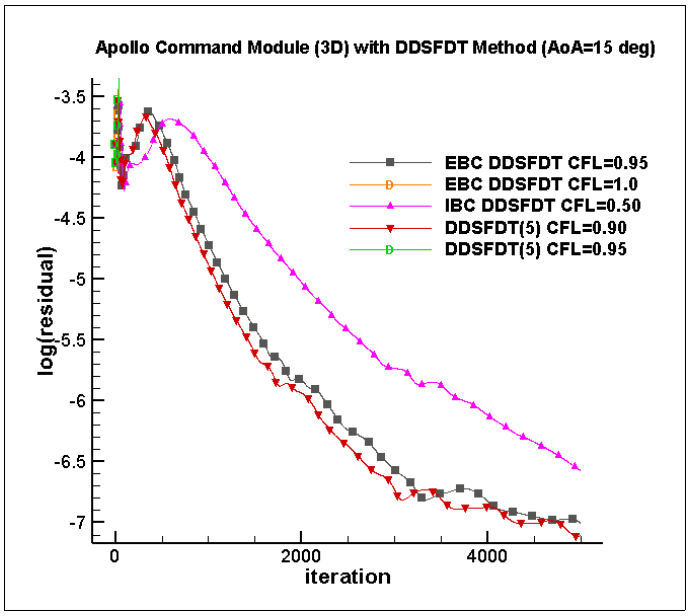
b) Angle of Attack = 15^0

Figure 4-32 Residual History of the Apollo Command Module with DDADI Method for Different Angle of Attacks

Figure 4-32 shows the residual histories of the analyses of the flow over 3-dimensional Apollo Command Module with DDADI method for 0 and 15 degrees of angle of attack. From the figure it can be said that change in angle of attack affects convergence characteristics of the methods. This situation is expected because especially in hypersonic speeds, position, angle and shape of the shock wave might be change for different angle of attacks. It is observed that convergence characteristics of the DDADI method for 15 degrees of angle of attack looks more linear compared to the 0 degree of angle of attack. For both angle of attack values, multi-block implicit boundary condition slows down the convergence and cannot give for even CFL number 0,1. Sub-iterations might help to speed up the convergence at the beginning iterations. However, after 5000 iteration DDADI and DDADI(5) reaches the same residual levels. Maximum allowable CFL number for DDADI method drops from 1,00 to 0,95 when the angle of attack is increased from 0 to 15 degrees. In fact, this reduction on CFL number caused by the increase in the angle of attack can be observed more clearly for DDADI(5) method, CFL numbers drops from 1,00 to 0,90 with sub-iterations.



a) Angle of Attack = 0^0

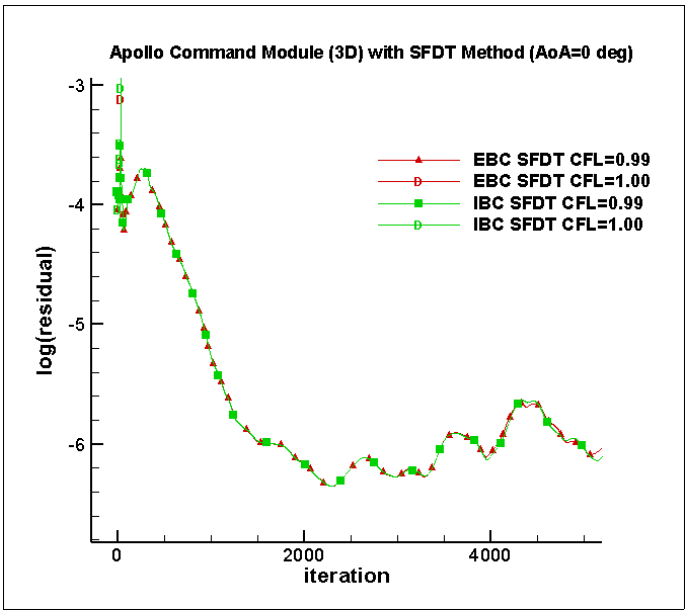


b) Angle of Attack = 15^0

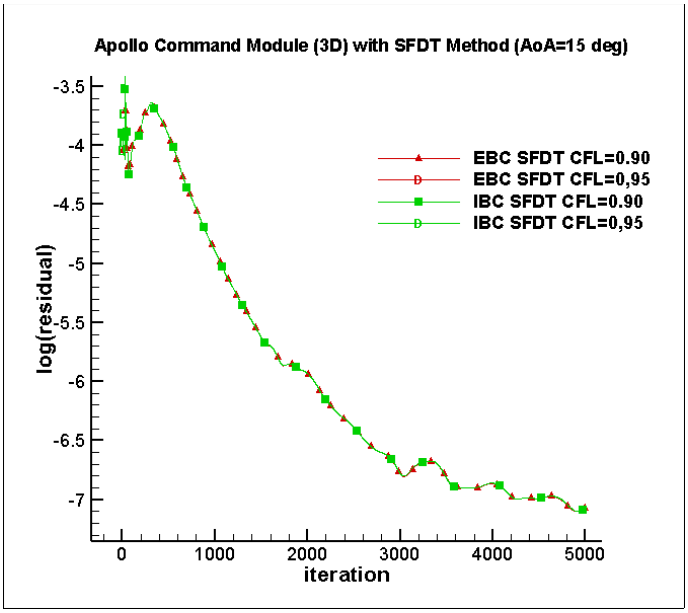
Figure 4-33 Residual History of the Apollo Command Module with DDSFDT Method for Different Angle of Attacks

Figure 4-33 shows the residual histories of the analyses of the flow over 3-dimensional Apollo Command Module with DDSFDT method for 0 and 15 degrees of angle of attack. Convergence characteristics of the DDSFDT method are very

similar to the convergence characteristics of the DDADI method for Apollo geometry. As it is observed from the convergence characteristics of the DDADI method for Apollo Command Module, from the figure 4-27 it can be said that change in angle of attack affects convergence characteristics of the methods. As mentioned before this situation is expected. It is observed that convergence characteristics of the DDSFDT method for 15 degrees of angle of attack looks more linear compared to the 0 degree of angle of attack. In contrast to the DDADI method for 15 degrees of angle of attack DDADI method with multi-block implicit boundary conditions can give a convergent solution. However, as observed on previous cases implicit boundary conditions slows down the convergence. Sub-iterations might help to speed up the convergence at the beginning of iterations. However, after 5000 iteration DDSFDT and DDSFDT(5) reaches the same residual levels. Maximum allowable CFL number for DDSFDT method drops from 1,00 to 0,95 when the angle of attack is increased from 0 to 15 degrees. In fact, this reduction on CFL number caused by the increase in the angle of attack can be observed more clearly for DDSFDT(5) method, CFL numbers drops from 1,00 to 0,90 with sub-iterations.



a) Angle of Attack = 0^0

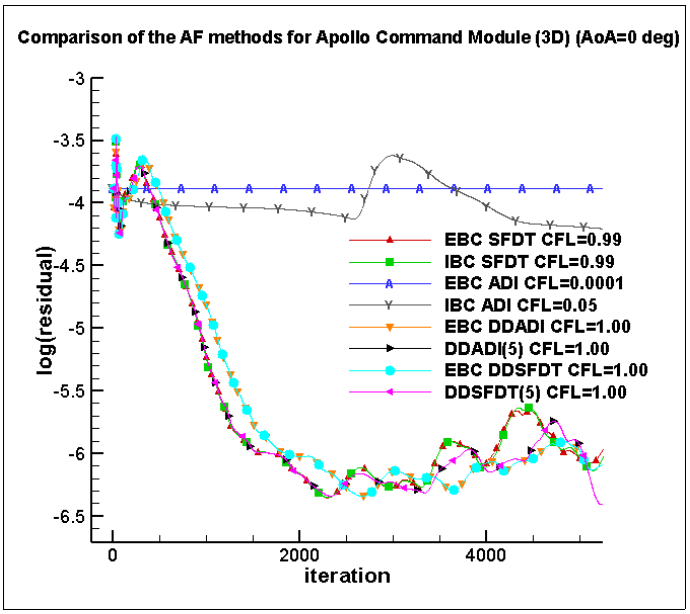


b) Angle of Attack = 15^0

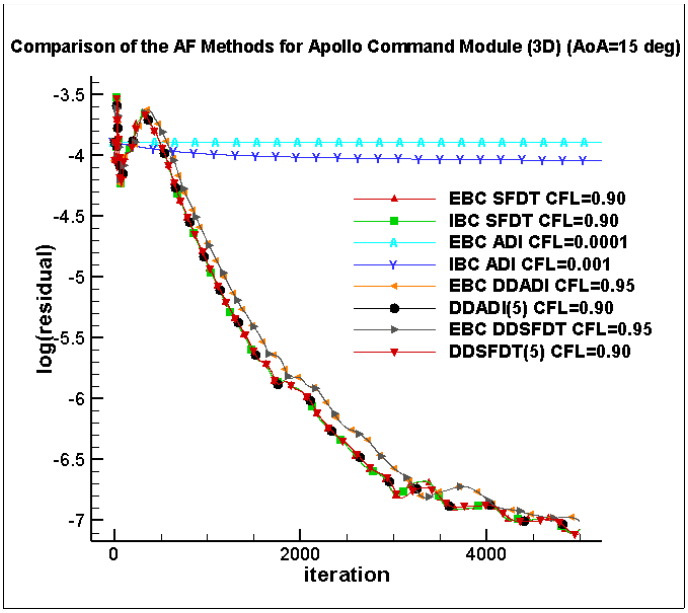
Figure 4-34 Residual History of the Apollo Command Module with SFDT Method for Different Angle of Attacks

Figure 4-34 shows the residual history of the flow analysis of the Apollo Command Module at angle of attack a) 0 degree and b) 15 degrees with the SFDT method with implicit and explicit boundary conditions on block interfaces. For 3-dimensional Apollo geometry at the angle of attack 0-degree residuals start oscillating after 2000

iterations, but the sufficient residual drop is achieved which can be proven with the comparison between the solver and experimental result in figure 4-30. Residual levels drop up to -6,5 within a 5000 iteration and at 0-degree angle of attack these residual levels seem sufficient. At 15 degrees of angle of attack convergence characteristics of the SFDT method looks more linear compared to the 0-degree angle of attack. The maximum allowable CFL number is 0,90 at 15 degrees of angle of attack. CFL number higher than 0.90 cause divergence of the solution as shown in figure 4-34 b). At 0-degree angle of attack maximum allowable CFL number is 0,99 and with CFL numbers that are higher than this value, solution diverges as shown in figure 4-34. In both angles of attack cases, it is observed that multi-block implicit boundary conditions cannot accelerate the convergency. This situation shows that omitted coefficients are too small to affect the convergence of the solution.



a) Angle of Attack = 0^0



b) Angle of Attack = 15^0

Figure 4-35 Comparison of the Residual Histories of the AF Methods for Apollo Command Module with Different Angle of Attacks

Figure 4-35 shows the comparison of the convergence characteristics of the approximate factorization methods. From the figure 4-35 a) and b) it can be said that the ADI method with or without multi-block implicit boundary conditions algorithm allows quite low CFL numbers such as 0.001 for explicit boundary conditions and

0.05 for implicit boundary conditions at 0-degree angle of attack. Therefore, the required time and iteration for convergence increases significantly compared to the other methods. Factorization error of ADI methods increases for 3-dimensional cases compared to the 2-dimensional cases, because the error terms mentioned in chapter 3 become quite large. Hence, as expected convergency suffers significantly for the Standard ADI method in 3-dimensional cases. However, for the other methods, it is possible to obtain convergent solutions.

Maximum allowable CFL numbers are indicated on the label of the charts. DDADI, The DDADI(5), DDSFDT, and DDSFDT(5) allow the highest CFL number as 1 while the maximum allowable CFL number is 0.99 for the SFDT method at 0-degree angle of attack. Maximum allowable CFL numbers drop for all approximate factorization methods at 15 degrees of angle of attack. From the figure, it can be said that except the ADI method all of the methods show similar characteristics and start oscillating after a point at 0-degree angle of attack while convergence characteristics looks more linear at 15 degrees of angle of attack. DDADI and DDSFDT show exactly the same characteristics and it seems, the convergence rates of these methods are slightly slow at the first 2000 iteration compared to the DDADI(5), DDSFDT(5) and SFDT methods. SFDT, DDSFDT(5), and DDADI(5) methods show very similar convergence characteristics for the 3-dimensional Apollo Command Module case for different angle of attack cases. SFDT method with multi-block implicit boundary conditions shows similar convergence characteristics with explicit boundary conditions. However, implicit boundary conditions can reach a slightly lower residual level after 5000 iterations, but since the residual history of both methods oscillates after a point, it cannot be said that implicit boundary conditions speed up the convergence for this case.

Table 4-17 Comparisons of Total Run Time for Convergence of AF Methods for Apollo Command Module (3D) at 0-degree Angle of Attack

AF Methods	Total run time(s) for convergence	Total run time (s)	μs per cell per iteration	Cell count	Total iteration	Redundant iterations
EBC DDADI (20blk, CFL=1,0)	5349	5349	0,97	1105920	5000	0
DDADI(5) (20blk, CFL=1,0)	19418	19418	3,51	1105920	5000	0
EBC DDSFDT (20blk, CFL=1,0)	5473	10947	0,99	1105920	10000	5000
DDSFDT(5) (20blk, CFL=1,0)	15098	30249	2,74	1105920	10000	5000
EBC SFDT (20blk, CFL=0,99)	3375	20793	0,63	1105920	30000	25000
IBC SFDT (20blk, CFL=0,99)	3866	23496	0,71	1105920	30000	25000

Table 4-18 Comparisons of Total Run Time for Convergence of AF Methods for Apollo Command Module (3D) at 15 degrees Angle of Attack

AF Methods	Total run time(s) for convergence	Total run time (s)	μs per cell per iteration	Cell count	Total iteration	Redundant iterations
EBC DDADI (20blk, CFL=1,0)	5838	5838	1,06	1105920	5000	0
DDADI(5) (20blk, CFL=1,0)	19851	19851	3,59	1105920	5000	0
EBC DDSFDT (20blk, CFL=1,0)	6864	6864	1,24	1105920	5000	0
DDSFDT(5) (20blk, CFL=1,0)	17842	17842	3,23	1105920	5000	0
EBC SFDT (20blk, CFL=0,99)	3197	3197	0,58	1105920	5000	0
IBC SFDT (20blk, CFL=0,99)	4180	4180	0,76	1105920	5000	0

Total run times for convergence of the approximate factorization methods for 0-degree angle of attack and 15 degrees of angle of attack are tabulated in the table 4-17 and table 4-18. It is assumed that after 5000 iteration all methods are converged

sufficiently. Required run times are compared for 5000 iterations because residual levels of the approximate factorization methods drop to similar levels which is order of $-6{,}5$ for 0-degree of angle of attack and $-7{,}0$ for 15 degrees of angle of attack. Within the equal number of iteration quickest solution can be obtained from the SFDT method with explicit boundary conditions on block interfaces for both angle of attack cases. SFDT method with multi-block explicit boundary conditions has very similar convergence characteristics with implicit boundary conditions in terms of required iteration number for convergence. However, since multi-block implicit boundary conditions algorithm has additional calculation and communication, required time for equal iteration number is higher compared to the explicit boundary conditions. DDADI and DDSFDT methods gives the solution within a similar time interval and quicker than the DDADI(5) and DDSFDT(5) with sub-iteration correction methods. Required time for convergence of the DDADI(5) and DDSFDT(5) are higher than the other methods because of the additional required time caused by the sub-iteration procedure.

Table 4-19 Achieved Speed-up Compared to the 20-block Multi-block Explicit Boundary Conditions Analysis with DDADI Method

AF Methods	Speed-up
EBC DDADI (20blk, CFL=1,0)	0%
DDADI(5) (20blk, CFL=1,0)	-263%
EBC DDSFDT (20blk, CFL=1,0)	-2%
DDSFDT(5) (20blk, CFL=1,0)	-182%
EBC SFDT (20blk, CFL=0,99)	37%
IBC SFDT (20blk, CFL=0,99)	28%

Table 4-20 Achieved Speed-up Compared to the 20-block Multi-block Explicit Boundary Conditions Analysis with DDADI Method

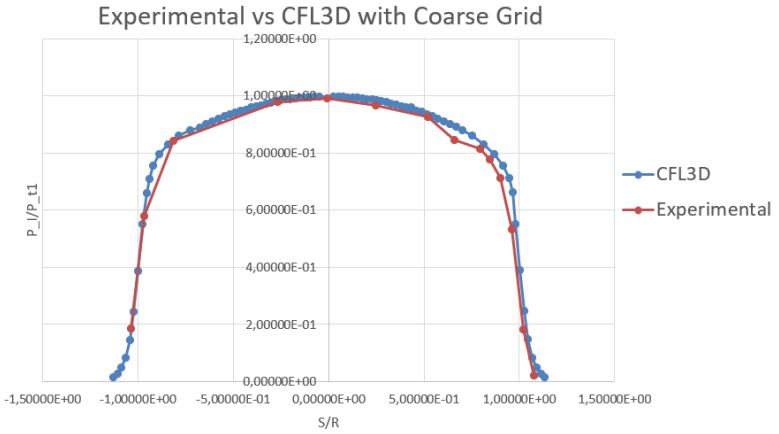
AF Methods	Speed-up
EBC DDADI (20blk, CFL=1,0)	0%
DDADI(5) (20blk, CFL=1,0)	-240%
EBC DDSFDT (20blk, CFL=1,0)	-18%
DDSFDT(5) (20blk, CFL=1,0)	-206%
EBC SFDT (20blk, CFL=0,99)	45%
IBC SFDT (20blk, CFL=0,99)	28%

Table 4-19 and 4-20 shows the achieved speed-up of the methods by taking the 20-block explicit boundary conditions with DDADI method as a baseline for speed-up calculations. It is observed that for the both angle of attack case SFDT method with explicit boundary conditions has the highest speed-up. Since there is no benefit of the implementation of the multiblock implicit boundary conditions for 3-dimensional apollo command module case in terms of convergence acceleration, it is wise to use explicit boundary conditions on block interfaces not to waste time for additional calculation and communication. In addition, since the convergence characteristics of the DDADI, DDSFDT, DDADI(5) and DDSFDT(5) methods are similar, sub-iteration calculations cause higher required time for convergence. Hence, it is concluded that sub-iterations are unnecessary for the 3-dimensional apollo command module analysis.

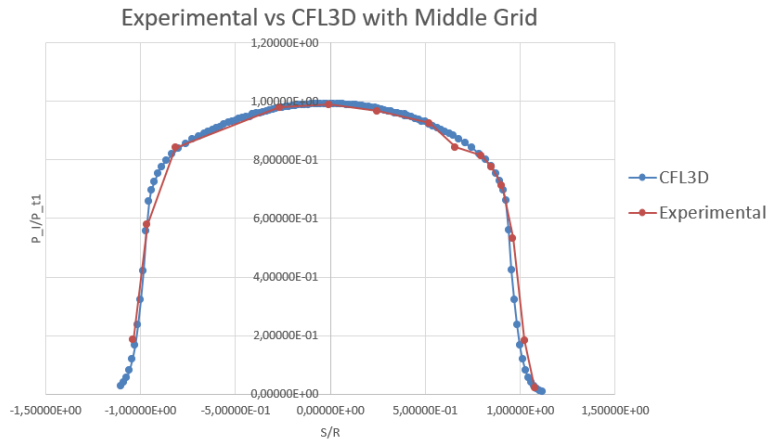
4.5.1 Grid Refinement for Apollo Command Module (3D)

With 3D grid for Apollo Command Module possibility of obtaining results from the solver that perfectly matched with experimental results is quite high compared to 2D cases. However, grid convergence must be construct in order to obtain the optimum grid. In addition, grid with optimum sizes prevents wasting time with finer grids while giving the solution that matches with experimental result. Coarse, middle and

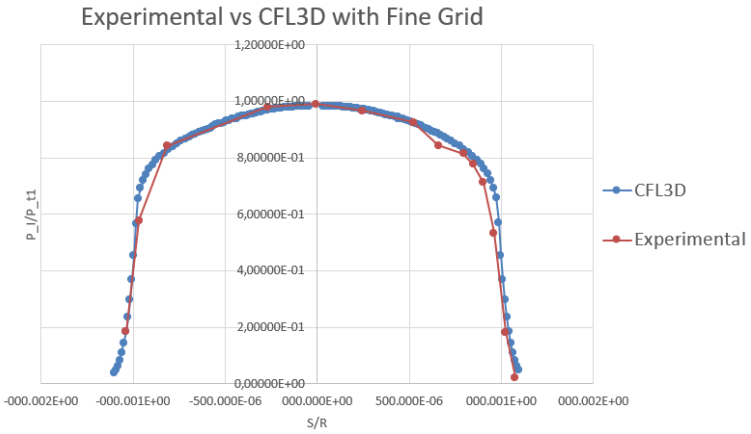
fine grids are generated. Course grid contains 364650 cells while middle grid and fine grids contains 1188250 and 2767050 cells respectively. Solutions with these 3 grids are obtained and compared with the experimental results.



a) Coarse Grid



b) Middle Grid



c) Fine Grid

Figure 4-36 Comparison of the Experimental Results versus Solver Results with Different Grids

Figure 3-36 shows the comparisons of the pressure ratio distribution of the a) coarse, b) middle and c) fine grids with experimental results. It is observed that coarse mesh over estimates the pressure ratios in front of the capsules and fine mesh over estimates the pressure ratio on the sides of the capsule where the flow passing through from high pressure region to shadow region of the capsule. From the figure it can be claimed that solution which is obtained by using middle grid perfectly matches with the experimental results. Therefore, it is concluded that middle grid with 1188250 cells is the optimum grid.

CHAPTER 5

CONCLUSION

5.1 Conclusion Remarks

In this thesis study convergence performance of the implicit approximate factorization methods hypersonic flow conditions for 2-dimensional and 3-dimensional geometries is presented. The efficiency of the implicit boundary conditions at block interfaces for the multi-block grids is investigated for different approximate factorization methods. The Standart Alternating Direction Implicit (ADI) method, Diagonal Dominant Alternating Direction Implicit method (DDADI) with and without Huang's sub-iteration correction, Spatially Factored with Diagonal Time terms (SFDT) method and Diagonal Dominant Spatially Factored with Diagonal Time terms (DDSFDT) method with and without Huang's sub-iteration correction are implemented into CFL3D open-source code.

Residual histories and total run time of the analyses are compared for flow inside a transonic Sajben Transonic Diffuser and flow over a compression ramp, double wedge, 2 and 3-dimensional Apollo Command Module geometries. For compression ramp case, DDADI and DDSFDT methods without Huang's sub iteration correction maximum allowable CFL number is 200. Since the sub iterations eliminates the factorization error, maximum CFL number increases with the use of sub iterations up to 10^5 and gives the convergent solution within 115 iterations. For the Sajben Transonic diffuser, sub-iteration correction increases the maximum allowable CFL number from 12,7 to 18. In the analysis of the double wedge case maximum allowable CFL number remained same which is 0,39. However, thanks to implementation of the sub-iterations required iteration for convergence drops from

4000 to 3000 for DDADI and DDSFDT methods. To sum up, DDADI and DDSFDT with Huang's sub-iteration correction methods showed the best convergence characteristics in terms of number of iterations for 2-dimensional Sajben Transonic Diffuser, flow over a compression ramp, and double wedge cases. However, the total run time is higher compared to the SFDT method. ADI method has the slowest convergence rates in general. For the 3-dimensional Apollo Command Module, the SFDT method has the best convergence rate.

For diagonal dominant relaxation schemes, maximum allowable CFL number decreases with implicit boundary conditions on block interfaces. However, at equal CFL numbers convergency improved. Implicit boundary conditions algorithm improves the convergence performance of the Standart ADI method dramatically, maximum allowable CFL numbers are increased for multi-block grids. SFDT method with implicit boundary conditions eliminates the errors caused by the multi-block treatment and the maximum allowable CFL number is increased for 2 dimensional Apollo Command Module from 0,3 to 0,4 and convergence rate increases. However, for the other cases, maximum allowable CFL numbers remained the same despite the elimination of the errors caused by the multi-block treatment. It is shown that the diagonal dominance of the approximate factorization method directly affects the efficiency of the implicit boundary condition treatment for the block interfaces of the multi-block grids.

To conclude, in the literature, there is no source that compares the SFDT method with other approximate factorization methods. Hence, to fill this gap in the literature, the SFDT method is compared with ADI and DDADI methods. There is no official source about the implementation of the diagonal dominant relaxation scheme to SFDT method. Therefore, the DDSFDT method is developed and the convergence characteristic of the new method is compared with ADI, DDADI, and SFDT methods. It is observed that convergence characteristics of the approximate

factorization methods are case-dependent. However, it can be said that the ADI method shows the slowest convergence characteristics in generals. For simple 2-dimensional cases, DDADI and DDSFDT methods give the convergent solution within a fewer number of iterations compared to the SFDT methods. However, SFDT method has the lowest required time for convergence in general and it can be concluded that the SFDT method has the best convergence rate in hypersonic flow regimes.

In addition, in the literature, there is no source that investigate and compare the efficiency of the multi-block implicit boundary conditions with different approximate factorization methods for different flow cases. Therefore, in order to contribute to the literature, in this study efficiency of the multi-block implicit boundary conditions on block interfaces with different approximate factorization methods is investigated. Similar to the approximate factorization methods, the efficiency of the multi-block implicit boundary conditions is also case-dependent. For the ADI method it is obvious that with multi-block grids multi-block implicit boundary conditions algorithm should be definitely implemented into ADI method because the convergence performance of the ADI method increases dramatically with multi-block implicit boundary conditions. For diagonal dominant factorization schemes implementation of the multi-block implicit boundary conditions reduces the allowable CFL number. However, at equal CFL number DDADI and DDSFDF methods with implicit boundary conditions shows better convergence characteristics compared to the explicit boundary conditions. Implementation of the multi-block implicit boundary conditions algorithm into the SFDT method accelerates the convergence dramatically for the 2-dimensional Apollo Command Module. However, in general, convergence characteristics of the SFDT method remain the same with the implementation of the multi-block implicit boundary conditions. Yet, since the efficiency of the implicit boundary conditions with the SFDT method is case-dependent, it is reasonable to have an option to use this algorithm for the cases

that the accumulated errors on the block boundaries are significantly high to affect the solution.

5.2 Future Work Recommendations

Since the diagonalization of the approximate factorization methods increases the computational efficiency, diagonalized versions of the approximate factorization methods can be added to the comparisons as a future work. In order to increase the efficiency of the implicit boundary conditions for DDADI and DDSFDT methods turbulence model can be coupled with Navier-Stokes equation.

After obtaining proper time-advancement algorithm for hypersonic flow regimes, accuracy of the spatial discretization can be increase by adding RSM turbulence model. In addition, since chemical reactions are occur in the flow at hypersonic speeds, conservation mass of the species of the chemical reactions can be added to the Naviesr-Stokes equations to obtain more accurate solutions.

REFERENCES

[1] A. Jameson, “Numerical solution of the Euler equation for compressible inviscid in: Numerical Methods for Euler Equations of Fluid Dynamics,” pp. 199–245, 1983.

[2] J. Crank and P. Nicolson, “A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type,” *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 43, no. 1, pp. 50–67, 1947, doi: 10.1017/S0305004100023197.

[3] S. Yoon and A. Jameson, “Lower-upper symmetric-gauss-seidel method for the euler and navier-stokes equations,” *AIAA Journal*, vol. 26, no. 9, pp. 1025–1026, 1988, doi: 10.2514/3.10007.

[4] R. M. Beam and R. F. Warming, “An Implicit Finite-Difference Algorithm for Hyperbolic Systems in Conservation-Law Form,” 1976.

[5] R. M. Beam and R. F. Warmingt, “An implicit factored scheme for the compressible navier-stokes equations,” *AIAA Journal*, vol. 16, no. 4, pp. 393–402, 1978, doi: 10.2514/3.60901.

[6] T. H. Pulliam and J. L. Steger, “Implicit Finite-Difference Simulations of Three-Dimensional Compressible Flow,” *AIAA Journal*, vol. 18, no. 2, pp. 159–167, 1980, doi: 10.2514/3.50745.

[7] T. H. Pulliam and D. S. Chaussee, “A Diagonal Form of an Implicit Approximate-Factorization Algorithm,” 1981.

[8] D. S. Chaussee and T. H. Pulliam, “Two-dimensional inlet simulation using a diagonal implicit algorithm,” *AIAA Journal*, vol. 19, no. 2, pp. 153–159, 1981, doi: 10.2514/3.50936.

[9] T. H. Pulliam, R. W. Maccormack, and S. Venkateswaran, “Convergence Characteristics of Approximate Factorization Methods.”

[10] C.-H. Sung, S.-H. Park, and J. H. Kwon, “Multigrid Diagonalized-ADI Method for Compressible Flows.”

[11] J. Bardina and C. K. Lombard, “Three Dımensıonal Hypersonıc Flow Sımulatıons Wıth The Cscm Implıcıt Upwınd Navıer-Stokes Method,” 1987.

[12] G. H. Klopfer, C. Mao Hung, and J. T. Onufer, “A Diagonalized Diagonal Dominant Alternating Direction Implicit (D3ADI) Scheme and Subiteration Correction,” 1998.

[13] J. L. Thomas, “An Implıcıt Multıgrıd Scheme For Hypersonıc Strong-Interactıon Flowfıelds,” 1992.

[14] H. Moon, S. Hong, and D. You, “Application of the parallel diagonal dominant algorithm for the incompressible Navier-Stokes equations,” *J Comput Phys*, vol. 423, Dec. 2020, doi: 10.1016/j.jcp.2020.109795.

[15] X.-H. Sun, “Parallel Computing Application and accuracy of the parallel diagonal dominant algorithm *,” 1995.

[16] Y. Jin, F. Liao, and J. Cai, “Convergence acceleration for subiterative DDADI/D3ADI using multiblock implicit boundary condition,” *J Comput Phys*, vol. 429, Mar. 2021, doi: 10.1016/j.jcp.2020.110009.

[17] W. R. Briley and H. Mcdonald, “An overview and generalization of implicit Navier-Stokes algorithms and approximate factorization,” 2001. [Online]. Available: www.elsevier.com/locate/comp¯uid

[18] S. Singh and D. You, “A multi-block ADI finite-volume method for incompressible Navier-Stokes equations in complex geometries,” *J Comput Phys*, vol. 230, no. 19, pp. 7400–7417, 2011, doi: 10.1016/j.jcp.2011.06.006.

[19] W. R. Briley and H. Mcdonald, “Solutıon of the Three-Dımensıonal Compressıble Navıer-Stokes Equatıons By an Implıcıt Technıque.”

[20] W. R. Briley and H. Mcdonald, “Solution of the Multidimensional Compressible Navier-Stokes Equations by a Generalized Implicit Method,” 1977.

[21] W. R. Briley and H. A. Walls, “A Numerical Study of Time-Dependent Rotating Flow in a Cylindrical Container at Low and Moderate Reynolds Numbers,” 1971.

[22] J. Wilkes and S. Churchill, “Finite difference computation of natural convection in an enclosed cavity(rectengular enclosure),” *AIChE*, 1966.

[23] C. E. Pearson, “A computational method for viscous flow problems,” 1966.

[24] J. Douglas, “On The Numerıcal Integratıon of by Implicit Methods,” 1955. [Online]. Available: http://www.siam.org/journals/ojsa.php

[25] R. W. Maccormack, “A New Implicit Algorithm For Fluid Flow,” 1997.

[26] D. W. Peaceman and H. H. Rachford, “The Numerical Solution Of Parabolic And Elliptic Differential Equations*,” 1955. [Online]. Available: http://www.siam.org/journals/ojsa.php

[27] J. Douglas and J. E. Gonn, “A General Formulation of Ahernating Direction Methods Part I. Parabolic and Hyperbolic Problems.”

[28] Y. Saad and M. H. Schultz, “Gmres: A Generalized Minimal Residual Algorithm For Solving Nonsymmetric Linear Systems*,” 1986. [Online]. Available: http://www.siam.org/journals/ojsa.php

[29] A. Yildirim, G. K. W. Kenway, C. A. Mader, and J. R. R. A. Martins, “A Jacobian-free approximate Newton–Krylov startup strategy for RANS simulations,” *J Comput Phys*, vol. 397, Nov. 2019, doi: 10.1016/j.jcp.2019.06.018.

[30] D. A. Knoll and D. E. Keyes, “Jacobian-free Newton-Krylov methods: A survey of approaches and applications,” *Journal of Computational Physics*, vol. 193, no. 2. Academic Press Inc., pp. 357–397, Jan. 20, 2004. doi: 10.1016/j.jcp.2003.08.010.

[31] G. K. W. Kenway, C. A. Mader, P. He, and J. R. R. A. Martins, “Effective adjoint approaches for computational fluid dynamics,” *Progress in Aerospace Sciences*, vol. 110. Elsevier Ltd, Oct. 01, 2019. doi: 10.1016/j.paerosci.2019.05.002.

[32] T. BARTH, “Analysis of implicit local linearization techniques for upwind and TVD algorithms,” Mar. 1987. doi: 10.2514/6.1987-595.

[33] H. Asgharzadeh and I. Borazjani, “A Newton–Krylov method with an approximate analytical Jacobian for implicit solution of Navier–Stokes equations on staggered overset-curvilinear grids with immersed boundaries,” *J Comput Phys*, vol. 331, pp. 227–256, Feb. 2017, doi: 10.1016/j.jcp.2016.11.033.

[34] C. L. Rumsey, R. T. Biedron, and J. L. Thomas, “CFL3D: Its History and Some Recent Applications,” 1997.

[35] P. F. Richardson *et al.*, “Hypersonic CFD Applications for the National Aero-Space Plane,” 1989.

[36] R. W. MacCormack, “Implicit methods for fluid dynamics,” *Comput Fluids*, vol. 41, no. 1, pp. 72–81, Feb. 2011, doi: 10.1016/j.compfluid.2010.09.017.

[37] R. W. MacCormack and T. H. Pulliam, “Assessment of a new numerical procedure for fluid dynamics,” in *29th AIAA Fluid Dynamics Conference*, 1998, pp. 1–9. doi: 10.2514/6.1998-2821.

[38] R. W. Maccormack, “A New Implıcıt Algorıthm For Fluıd Flow,” 1997.

[39] P. Huang, “Implementation and validation of turbulence models in owerflow code,” *29th AIAA Fluid Dynamics Conference*, 1998.

[40] P. Subrahmanyam, “AIAA Atmospheric Flight Mechanics Conference Database Driven Computational Investigation of Aerothermodynamics And Automatic Grid Generation of Atmospheric Entry Vehicles,” 2005.

[41] S. L. Krist, R. T. Biedron, and C. L. Rumsey, “NASA CFL3D User’s Manual (Version 5.0),” 1998. [Online]. Available: http://www.sti.nasa.

[42] T. J. Bogar, M. Sajben, and J. C. Kroutil, “Characteristic frequencies of transonic diffuser flow oscillations,” *AIAA Journal*, vol. 21, no. 9, pp. 1232–1240, 1983, doi: 10.2514/3.8234.

[43] A. B. Siemens, B. Reinartz, A. Bosco, and S. Müller, “Computation of hypersonic shock boundary layer interaction on a double wedge using a differential Reynolds Stress Model Perfluortensidbeseitigung aus wasser mit nichtthermischem Plasma View project Piston-Ring-Package View project Computation of hypersonic shock boundary layer interaction on a double wedge using a differential Reynolds Stress Model,” 2009. [Online]. Available: https://www.researchgate.net/publication/238709752

[44] B. Reinartz and J. Ballmann, “Computation of hypersonic double wedge shock / boundary layer interaction.”

[45] T. Neuenhahn and H. Olivier, “Influence of the wall temperature and the entropy layer effects on double wedge shock boundary layer interactions,” 2006.

[46] J. J. Bertin, “Nasa Tm X-1243 The Effect Of Protuberances, Cavities, And Angle O F Attack On The Wind-Tunnel Pressure And Heat-Transfer Distribution For The Apollo Command Module The Effect Of Protuberances, Cavities, Andangleofattackonthewind-Tunnelpressure And Heat-Transfer Distribution For The Apollo Command Module.”

APPENDICES

A. Supplementation of Governing Equations

The inviscid flux terms are written as:

$$\hat{F} = JF = J\begin{pmatrix} \rho U \\ \rho Uu + \xi_x p \\ \rho Uv + \xi_y p \\ \rho Uw + \xi_z p \\ (e+p)U - \xi_t p \end{pmatrix}, \hat{G} = JG = J\begin{pmatrix} \rho V \\ \rho Vu + \eta_x p \\ \rho Vv + \eta_y p \\ \rho Vw + \eta_z p \\ (e+p)V - \eta_t p \end{pmatrix} \quad \text{(A.1)}$$

$$\hat{H} = JH = J\begin{pmatrix} \rho W \\ \rho Wu + \zeta_x p \\ \rho Wv + \zeta_y p \\ \rho Ww + \zeta_z p \\ (e+p)W - \zeta_t p \end{pmatrix} \quad \text{(A.2)}$$

Where the contravariant velocities are:

$$U = \xi_x u + \xi_y v + \xi_z w + \xi_t \quad \text{(A.3)}$$

$$V = \eta_x u + \eta_y v + \eta_z w + \eta_t \quad \text{(A.4)}$$

$$W = \zeta_x u + \zeta_y v + \zeta_z w + \zeta_t \quad \text{(A.5)}$$

The viscous flux terms are written as:

$$\hat{F_v} = JF_v = J\begin{pmatrix} 0 \\ \xi_x \tau_{xx} + \xi_y \tau_{xy} + \xi_z \tau_{xz} \\ \xi_x \tau_{xy} + \xi_y \tau_{yy} + \xi_z \tau_{yz} \\ \xi_x \tau_{xz} + \xi_y \tau_{yz} + \xi_z \tau_{zz} \\ \xi_x b_x + \xi_y b_y + \xi_z b_z \end{pmatrix} \quad \text{(A.6)}$$

$$\widehat{G_v} = JG_v = J\begin{pmatrix} 0 \\ \eta_x \tau_{xx} + \eta_y \tau_{xy} + \eta_z \tau_{xz} \\ \eta_x \tau_{xy} + \eta_y \tau_{yy} + \eta_z \tau_{yz} \\ \eta_x \tau_{xz} + \eta_y \tau_{yz} + \eta_z \tau_{zz} \\ \eta_x b_x + \eta_y b_y + \eta_z b_z \end{pmatrix} \quad \text{(A.7)}$$

$$\widehat{H_v} = JH_v = J\begin{pmatrix} 0 \\ \zeta_x\tau_{xx} + \zeta_y\tau_{xy} + \zeta_z\tau_{xz} \\ \zeta_x\tau_{xy} + \zeta_y\tau_{yy} + \zeta_z\tau_{yz} \\ \zeta_x\tau_{xz} + \zeta_y\tau_{yz} + \zeta_z\tau_{zz} \\ \zeta_x b_x + \zeta_y b_y + \zeta_z b_z \end{pmatrix} \tag{A.8}$$

The shear stress and hear flux terms are defined in tensor notations (summation convention implied) as:

$$\tau_{x_i x_j} = \frac{M_\infty}{Re_{\widetilde{L_R}}}\left[\mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + \lambda\frac{\partial u_k}{\partial x_k}\delta_{ij}\right] \tag{A.9}$$

$$b_{x_i} = u_j\tau_{x_i x_j} - \dot{q_{x_\iota}} \quad and \quad \dot{q_{x_\iota}} = \left[\frac{M_\infty\mu}{Re_{\widetilde{L_R}}\Pr(\gamma-1)}\right]\frac{\partial a^2}{\partial x_i} \tag{A.10}$$

The pressure is obtained by the equation of state for a perfect gas

$$p = (\gamma-1)\left[e - \frac{\rho}{2}(u^2 + v^2 + w^2)\right] \tag{A.11}$$

The above equations are nondimensionalized in terms of free stream properties which are density, speed of sound and molecular viscosity. To evaluate the derivatives with respect to cartesian coordinates x, y and z in terms of generalized coordinates ξ, η and ζ chain rule is used.

The following parameters are used to nondimensionalize the flow-field variables:

$\tilde{L}$ = characteristic length, feet

L_{ref} = corresponding length in the grid (nondimensional)

$\tilde{L}_R = \tilde{L}/L_{ref}$ = reference length used by code, feet (dimensional)

$\tilde{\rho}_\infty$ = free-stream density, slug/feet

$\tilde{a}_\infty$ = free-stream speed of sound, feet/second

$|\widetilde{V}|_\infty$ = free-stream velocity, feet/second

$\tilde{\mu}_\infty =$ free-stream molecular viscosity, slug/feet-second

$\tilde{t} =$ time, seconds

In CFL3D, the nondimensionalizations are performed as follows:

$$\begin{aligned} \rho &= \frac{\tilde{\rho}}{\tilde{\rho}_\infty} \quad , \quad \rho_\infty = 1 \\ p &= \frac{\tilde{p}}{\tilde{\rho}_\infty \tilde{a}_\infty{}^2} \quad , \quad p_\infty = \frac{1}{\gamma} \\ a &= \frac{\tilde{a}}{\tilde{a}_\infty} \quad , \quad a_\infty = 1 \\ T &= \frac{\tilde{T}}{\tilde{T}_\infty} = \frac{\gamma p}{\rho} \quad , \quad T_\infty = 1 \end{aligned} \tag{A.12}$$

The numerical algorithm uses a semi-discrete finite-volume formulation, resulting in a consistent approximation to conservation laws in integral form

$$\frac{\partial}{\partial t}\iiint_V Q dV + \iint_S \vec{f}.\vec{n} dS = 0 \tag{A.12}$$

where $\vec{f}$ denotes the net flux through a surface S with unit normal $\vec{n}$ containing the (time-invariant) volume V. Integration of Equation (A-12) over a control volume bounded by lines of constant ξ, η, and ζ gives the semi-discrete form.

$$\begin{aligned} \left(\frac{\partial \hat{Q}}{\partial t}\right)_{i,j,k} &+ \left(\hat{F} - \hat{F}_v\right)_{i+\frac{1}{2},j,k} - \left(\hat{F} - \hat{F}_v\right)_{i-\frac{1}{2},j,k} + \left(\hat{G} - \hat{G}_v\right)_{i,j+\frac{1}{2},k} \\ &- \left(\hat{G} - \hat{G}_v\right)_{i,j-\frac{1}{2},k} + \left(\hat{H} - \hat{H}_v\right)_{i,j,k+\frac{1}{2}} \\ &- \left(\hat{H} - \hat{H}_v\right)_{i,j,k-\frac{1}{2}} = 0 \end{aligned} \tag{A.13}$$

where, for convenience,

$$\begin{aligned} \Delta\xi &= \xi_{i+\frac{1}{2},j,k} - \xi_{i-\frac{1}{2},j,k} = 1 \\ \Delta\eta &= \eta_{i,j+\frac{1}{2},k} - \eta_{i,j-\frac{1}{2},k} = 1 \\ \Delta\zeta &= \zeta_{i,j,k+\frac{1}{2}} - \zeta_{i,j,k-\frac{1}{2}} = 1 \end{aligned} \tag{A.14}$$

The discrete values $\hat{Q}_{i,j,k}$ are regarded as average values taken over a unit computational cell; similarly, discrete values of $\hat{F}$, $\hat{G}$, and $\hat{H}$ are regarded as face-average values. The convective and pressure terms are differenced using either the upwind flux-difference splitting technique of Roe or the flux-vector-splitting technique of van Leer. The MUSCL (Monotone Upstream-centered Scheme for Conservation Laws) approach of van Leer is used to determine state-variable interpolations at the cell interfaces. The shear stress and heat transfer terms are centrally differenced [41].