DOMAIN DECOMPOSITION ANALYSIS OF AERODYNAMIC COUPLING

by

Özkan Aydın

B.S. in Ch.E., Boğaziçi University, 2005

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Mechanical Engineering

Boğaziçi University

2008

DOMAIN DECOMPOSITION ANALYSIS OF AERODYNAMIC COUPLING

APPROVED BY:

Assist. Prof. Ali Ecder .....
(Thesis Supervisor)

Assist. Prof. Kunt Atalık .....

Assoc. Prof. Can Özturan .....

DATE OF APPROVAL: 11.06.2008

ACKNOWLEDGEMENTS

This work would not have been possible without the support and encouragement of Assist. Prof. Dr. Ali Ecder, under whose supervision I conducted my research and wrote my thesis. I want to express my gratitude to him giving me the opportunity to work with him.

I would like to thank my examiners Assoc. Prof. Dr. Can Özturan and Assist. Prof. Dr. Kunt Atalık for their fruitful criticism and comments.

I also thank to each of my friends, especially FMS laboratory members Erhan Turan, Yalın Kaptan, Altuğ Melik Başol, and Bülent Düz for their support and guidance at critical stages of this study.

Finally, I am indebted to my family for their continuous support and encouragement during my entire education.

ABSTRACT

DOMAIN DECOMPOSITION ANALYSIS OF AERODYNAMIC COUPLING

The main objective of this research is the computational modeling and analysis of aerodynamic-coupling effects of multiple bodies in motion in a fluid using domain decomposition techniques. Simulation of such a flow field requires the solving of the nonlinear equations of the fluid motion. The computational model and techniques are first tested in the analysis of the internal flow problem. However, the objective of this thesis work is to analyze the aerodynamic coupling observed in the external flow problem. Computational domains is separated into simple cartesian subdomains. The flow at these subdomains is solved separately using stream function, vorticity formulation and these solutions are transferred into each other using the alternating multiplicative Schwarz technique. Newton's method combined with two different Krylov sub-space solvers is applied to solve the flow problem in rectangular region. In order to have global convergence, backtracking algorithm is used within Newton method. In order to decrease computational work, local mesh refinement strategy is used in critical areas of domains. In the internal flow problem, the effects of Reynolds number on the flow are investigated for three different Reynolds numbers, namely, Re=50, Re=100, Re=150. In the external flow problem, influences of the angle-of-attack, Reynolds number, and longitudinal distance between the objects on the flow properties are investigated.

Keywords: Newton's method, backtracking, Krylov solvers, domain decomposition, internal flow, external flow

ÖZET

AERODİNAMİK AKIŞLARIN ALAN AYRIŞTIRMA İLE ANALİZİ

Bu çalışmanın temel amacı akışkan içerisinde hareket eden birden fazla cismin, birbirine yaptığı aerodinamik etkilerin Alan Ayrıştırma Teknikleri ile bilgisayar modellemesinin yapılmasıdır. Bu çeşit akış alanlarının similasyonu akış hareketinin lineer olmayan denklemlerinin çözümünü gerektirir. Hesaplama teknikleri ve modelleri ilk önce iç akış problemlerinde test edilmiştir. Bununla birlikte bu tez çalışmasının ana hedefi dış akış problemlerindeki aerodinamik etkilerin incelenmesidir. Hesaplama alanı basit kartezyen alt alanlara ayrılmıştır. Bu alt alanlardaki akış, akım fonksyonu-vortisite formülasyonu kullanılarak çözülmüş ve bu çözümler birbirine Almaşık Çarpımsal Schwarz tekniği kullanılarak aktarılmıştır. İki adet Krylov alt uzay tekniği ile birleştirilen Newton yöntemi, dikdörtgensel alandaki akım probleminin çözümüne uygulanmıştır. Global yakınsamayı sağlamak için Newton metoduna, Geri Döndürme Algoritması eklenmiştir. Hesaplama yükünü azaltmak için akım alanının kritik bölgelerinde yerel ağ sayısı artırılmıştır. İç akış probleminde, Reynolds sayısının akım özellikleri üzerindeki etkisi üç farklı Reynolds sayısı (Re=50, Re=100, Re=150) için araştırılmıştır. Dış akış promleminde ise hücum açısının, Reynolds sayısının ve cisimler arası yatay uzaklığın akış özellikleri üzerindeki etkisi incelenmiştir.

Anahtar kelimeler: Newton metodu, geri döndürme, Krylov çözücüleri, alan ayrıştırma, iç akış, dış akış

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZET v
LIST OF FIGURES viii
LIST OF TABLES xiii
LIST OF SYMBOLS/ABBREVIATIONS xv
1. INTRODUCTION 1
2. NUMERICAL METHODS 7
2.1. Newton's Method 9
2.1.1. Inexact Newton Method 10
2.1.2. Global convergence and damping term 11
2.1.3. Termination of nonlinear iterations 13
2.2. Linear Solver 13
2.2.1. Newton-Krylov Methods 16
2.2.2. Krylov Methods for Solving Linear Equations 17
2.2.3. Directional Derivative 17
2.2.4. Compressed Storage Schemes 18
2.3. Preconditioning 18
2.3.1. Jacobi 21
2.3.2. Symmetric Gauss-Seidel (SGS) 21
2.3.3. Incomplete LU Decomposition (ILU) 22
2.4. Domain Decomposition 23
2.4.1. Overlapping Domain Decomposition 24
2.4.1.1. Multiplicative Schwarz Procedure 28
2.4.1.2. Additive Schwarz Procedure 30
2.4.2. Non-overlapping Domain Decomposition 31
3. COMPUTATIONAL MODELING 33
3.1. Internal Flow Problem 33
3.1.1. Problem Statement 33

3.1.2. Discretization of Governing Equations 37
3.1.3. Boundary Conditions 38
3.1.3.1. Inflow 38
3.1.3.2. Outflow 39
3.1.3.3. Solid Wall 39
3.1.3.4. Artificial Boundary 41
3.1.3.5. Re-entrant Corner 42
3.1.4. Solution Algorithm 42
3.1.5. Results of Internal Flow Problem 42
3.2. The External Flow Problem 62
3.2.1. Problem Statement 62
3.2.2. Discretization of the Governing Equations 63
3.2.3. The Boundary Conditions 66
3.2.3.1. Upstream 67
3.2.3.2. Downstream 68
3.2.3.3. Far-field 68
3.2.3.4. Artificial Boundary 69
3.2.3.5. Solid Wall 69
3.2.4. Solution Algorithm 69
3.2.5. Results of the External Flow Problems 70
4. DISCUSSION AND CONCLUSION 110
REFERENCES 115

LIST OF FIGURES

Figure 2.1. Initial partitioning of matrix A 21

Figure 2.2. Schematic representation of overlapping and non-overlapping domain decomposition participants of main domain (Ω) 25

Figure 2.3. Schematic representation of matching and non-matching grids . . 26

Figure 2.4. Algorithm of the Alternating Schwarz 27

Figure 2.5. Algorithm of the Multiplicative Schwarz 29

Figure 2.6. Algorithm of the Additive Schwarz 30

Figure 2.7. Solution algorithm of domain decomposition method 32

Figure 3.1. Geometry and boundary conditions of L-shaped domain 35

Figure 3.2. Geometry adaptively meshed L-shaped domain 36

Figure 3.3. Preconditioning performance of sample inner step with full matrix storage 44

Figure 3.4. Preconditioning performance of a sample inner step without full matrix storage 45

Figure 3.5. Comparison of different solution methods (exact, inexact, with / without damping) 47

Figure 3.6. L-shaped problem at Re=100, h=1/8 52

Figure 3.7. L-shaped problem at Re=100, h=1/16 52

Figure 3.8. L-shaped problem at Re=100, h=1/32 53

Figure 3.9. L-shaped problem at Re=100, h=1/64 53

Figure 3.10. L-shaped problem at Re=100, h=1/64 without mesh refinement . 54

Figure 3.11. Solution of L-shaped problem at Re=50 55

Figure 3.12. Solution of L-shaped problem at Re=100 56

Figure 3.13. Solution of L-shaped problem at Re=150 57

Figure 3.14. Residual norms of the inner and outer iterations 59

Figure 3.15. Reduction of difference vector norm with respect to number of visits at Re=100 60

Figure 3.16. Nonlinear residual norm for L-shaped problem at Re=100 61

Figure 3.17. Geometry and boundary conditions of external flow problems . . . 64

Figure 3.18. Adaptively meshed domains for the external flow 65

Figure 3.19. Infinity sensitivity test for the external flow problem at Re=100, h/L=1/20 73

Figure 3.20. Thin body with 45^0 angle-of-attack 75

Figure 3.21. Stream function values for the body with W/L=1/4 76

Figure 3.22. Stream function values for the body with W/L=1/2 76

Figure 3.23. Stream function and vorticity values for the body with W/L=1 . . 77

Figure 3.24. Original domain sizes for the body with W/L=1 78

Figure 3.25. Stream function values for the body with W/L=3/2 78

Figure 3.26. Stream function and vorticity values for Re=0.001 80

Figure 3.27. Stream function and vorticity values for Re=50 81

Figure 3.28. Stream function and vorticity values for Re=100 82

Figure 3.29. Stream function and vorticity values for Re=150 83

Figure 3.30. Reduction of difference vector norm for the single body external flow problem at Re=100 84

Figure 3.31. Residual norms of the inner and outer iterations 85

Figure 3.32. Nonlinear residual norm for single body external flow problem . . 86

Figure 3.33. Stream function and vorticity values for 0^0 angle-of-attack 88

Figure 3.34. Stream function and vorticity values for 15^0 angle-of-attack 89

Figure 3.35. Stream function and vorticity values for 45^0 angle-of-attack 90

Figure 3.36. Stream function and vorticity values for 75^0 angle-of-attack 91

Figure 3.37. Sub-domains for the two immersed body case 92

Figure 3.38. Stream function and vorticity values for two bodies and top object has 0^0 angle-of-attack at Re=100 94

Figure 3.39. Stream function and vorticity values for two bodies and top object has 15^0 angle-of-attack at Re=100 95

Figure 3.40. Stream function and vorticity values for two bodies and top object has 45^0 angle-of-attack at Re=100 96

Figure 3.41. Stream function and vorticity values for two bodies and top object has 75^0 angle-of-attack at Re=10^2 97

Figure 3.42. Stream function and vorticity values for two bodies and top object has 75^0 angle-of-attack at Re=10^0 98

Figure 3.43. Stream function and vorticity values for two bodies and top object has 75^0 angle-of-attack at Re=10^{-2} 99

Figure 3.44. Stream function and vorticity values for two bodies for X/L=0 at Re=100 101

Figure 3.45. Stream function and vorticity values for two bodies for X/L=0.5 at Re=100 102

Figure 3.46. Stream function and vorticity values for two bodies for X/L=1 at Re=100 103

Figure 3.47. Stream function and vorticity values for two bodies for X/L=1.5 at Re=100 104

Figure 3.48. Stream function and vorticity values for two bodies for X/L=2 at Re=100 105

Figure 3.49. Stream function and vorticity values for two bodies for X/L=2.5 at Re=100 106

Figure 3.50. Reduction of difference vector norm for the multibody body external flow problem and X/L=0 108

Figure 3.51. Residual norms of the inner and outer iterations for the sample step and X/L=0 109

LIST OF TABLES

Table 3.1. Exact Newton method ($tol = 10^{-4}$), without backtracking 48

Table 3.2. Inexact Newton method, constant forcing term ($\eta = 10^{-1}$), without backtracking 48

Table 3.3. Inexact Newton method,forcing term (η) varies as the nonlinear algorithm proceeds, without backtracking 49

Table 3.4. Exact Newton method ($tol = 10^{-4}$), with backtracking 49

Table 3.5. Inexact Newton method, constant forcing term($\eta = 10^{-1}$), with backtracking 50

Table 3.6. Inexact Newton method,forcing term (η) varies as the nonlinear algorithm proceeds, with backtracking 50

Table 3.7. Grid sensitivity test for L-shaped problem at Re=100 54

Table 3.8. Computational load for the L-shaped problem at different Re . . . 58

Table 3.9. Sample result without α term 72

Table 3.10. Sample result with α term 72

Table 3.11. Stream function values at sample points 73

Table 3.12. A sample nonlinear step for the single body external flow problem 87

Table 3.13. Summary of the computational loads for different longitudinal distances 107

LIST OF SYMBOLS/ABBREVIATIONS

$A(x)$	Coefficient matrix
$A(x)^{-1}$	Inverse of Coefficient matrix
$b(x)$	Right hand side of a non-linear system
D	Diagonal of A
E	The strict lower part of A
F	The strict upper part of A
F(x)	Non-linear system of equations
h	Grid size
I	Identity matrix
$J(x)$	Jacobian matrix
L	Body length
M	Preconditioner matrix
M_{left}	Left preconditioning matrix
M_{right}	Right preconditioning matrix
M^{-1}	Inverse of the preconditioner matrix
n	Iteration count
Re	Reynolds Number
x	Unknown vector
X	Longitudinal distance
x_0	Initial guess for unknown vector
u	Velocity vector in the x-direction
v	Velocity vector in the y-direction
W	Body width
λ	Newton step length
η	Forcing term
ρ	Fluid density
Ψ	Stream function
Ω	Vorticity

BICGSTAB	Bi-conjugate Gradient Stabilized
CGS	Conjugate Gradient Squared
DDM	Domain Decomposition Method
GMRES	Generalized Minimal Residual
ILU	Incomplete LU decomposition
LSSFDS	Least-Square Singular Finite Difference Scheme
PDE	Partial Differential Equation
SGS	Symmetric Gauss-Seidel

1. INTRODUCTION

The solutions to the governing equations of fluid mechanics represent one of the most challenging task in engineering. Generally, the fluid mechanics equations form a set of nonlinear second order partial differential equations (PDEs) which must be solved within an irregular domain, subject to various initial and boundary conditions.

Because of a singular point in domain or obstacles on grid generation, solution of a partial differential equations system on very complex geometries is very difficult. Such as in the situation of flow inside an L-shaped domain with re-entrant corner (inner corner of L-shaped domain) or moving multiple bodies mutually affect each other. The major difficulty in the L-shaped problem is that there is a singular point on the re-entrant corner. Singular points deteriorate the convergence rate of finite difference scheme and classical finite difference schemes become inaccurate or at least very costly due to need for highly refined mesh. Finite difference schemes have been modified in various ways in order to handle this difficulty, see, e.g., [37]. The suggested schemes require a large number of grid points in the neighborhood of the singularities. One of the most efficient solution to this pollution problem from the singular points is proposed by Yosibash et al. [34]. They suggest a semi analytic finite difference scheme named as, Least-Square Singular Finite Difference Scheme (LSSFDS). They apply LSSFDS in the vicinity of the singularity and a conventional finite difference scheme on the remaining domain. The other way of removing pollution problem is domain decomposition algorithm with adaptively or locally refine mesh strategy. In this work local mesh strategy is used. The optimal global and local refinement grid sizes for re-entrant corner problem is given in [16].

Some examples of the internal flow are the flow in pipe or the flow of blood in the arteries. Some examples of the flow over bodies that are immersed in a fluid are the flow of water around submarines and fish, the flow of air around airplanes, automobiles, or buildings. In order to have a correct design of a car, the fluid forces on the surface of the car have to be determined. The fuel consumption of the car can be decreased and

the handling characteristic can be improved by the minimization of these fluid forces. In order to have a proper building design, various wind effects must be considered especially for a tall skyscraper or a long suspension bridge.

Theoretical and experimental approaches are used to obtain information about external flow and internal flow. Because of the complexities of the governing equations and the complexities of the geometry of the obstacles for the external flow, the amount of information from analytical approaches is very limited. Because of this, much of the information comes from computational fluid dynamics and experimental work. In the experimental approach, experiments are carried out in a wind tunnel or water tunnel with a scaled model of objects. But generally these experiments are expensive and time-consuming and one experimental set up is used to determine one flow characteristic. The trend in the simulation of internal and external flow problems is to use computational fluid dynamic tools, and the validation of which is performed with experimental works.

Most of the difficulties in simulating external and internal flows numerically may be removed by using the domain decomposition method. The number of research on domain decomposition techniques has been increasing. The origin of this method was based on Schwarz's notes in 1890. Advances in parallel computing lead to the rebirth of this old technique. The book by Smith, Bjorstat, and Gropp [31] gives a thorough survey of domain decomposition methods. Chan and Mathew [5] established an important survey on this subject.

Schwarz originally derived the Alternating Method to prove the existence of a solution to a linear elliptic partial differential equation in a domain in which there is no known analytical solution. Lions [6, 7] gives much attention to overlapping Schwarz method in 1980s.

The interaction of fluid flow with the rigid bodies is often encountered in many areas of civil, mechanical, and aerospace engineering. One of the most prominent classes of such problems is the interaction of air flow with long, slender civil engineering struc-

tures, such as tall towers. In this case the rigid bodies are stationary. There are also moving rigid body examples such as the interaction between two cars or trains which are moving towards or passing by each other. There is a lot of experimental work to describe the convoy and parallel passing (overtaking). Flow visualization technique is used to clarify the flow behavior, especially wake structure behind interfering passenger cars. Drag is the most important factor from the viewpoint of fuel economy. From the control and stability point of view, however, the side force and yawing moment are the most crucial aerodynamic characteristics of a vehicle. Several studies measuring these quantities under steady conditions have been conducted. Experiments conducted by Zabat, et. al. [35, 36] determine the aerodynamic characteristics for a platoon of vehicles under steady conditions. They found that the effects of the interacting flow fields increased as the spacing between the vehicles decreased. For the overtaking vehicles, the transient forces acting on them are a function of the longitudinal and transverse spacings and of the relative velocity between the two vehicles. This transient aerodynamic phenomena associated with passing manoeuvres are investigated experimentally in [24].

A vast amount of studies has been performed to investigate different transition processes of the flow past a circular bodies-experimentally,numerically, and theoretically. By contrast, there are a few similar studies found on flow over rectangular bodies, at moderate Reynolds number. Results of some important numerical studies based on $\Psi - \Omega$ formulation and discrete vortex methods are briefly summarized in [27]

Flow around a single moving object can be efficiently simulated by using a single body-fitted grid in the non-inertial reference frame. Body-fitted curvilinear meshes are generally used in finite difference and finite volume computations for a variety of reasons. For example, it simplifies the application of boundary condition and allows clustering of grid points in high gradient regions. For a simple body shape, the use of a single body conforming curvilinear mesh leads to the most efficient solution procedure. This approach does not work for multiple objects moving relative to each other. A grid mapping varies with time as the object moves. Even for stationary multiple objects, it is quite difficult to generate a well-clustered curvilinear grid that is not overly

skewed. Currently, most of the research in treating complex configurations is being directed towards composite structured or unstructured grids. Both approaches require a more complex data handling program than that is required by simple structured grid. An efficient method for solving 2D incompressible viscous flow around multiple moving objects is presented by Russell [26]. This study employs an underlying regular cartesian grid to solve the system using a stream function vorticity formulation and discontinuities representing the embedded objects.

Mesh adaptation is another issue related to the numerical simulation of external flows. Vorticity is generated in the vicinity of solid boundary and concentrates further downstream in a wake, the outer flow is almost curl-free. Hence an adaptive method should concentrate grid points in the boundary layer and the wake. But if there are several obstacles immersed in the fluid and if these obstacles are in relative motion, the grid generation of this shape-changing flow is not an easy task.

Guermond, et. al. [13] uses a domain decomposition technique which combines the vortex method and the body-fitted finite differences method for the numerical approximations of the two dimensional external flows. The vortex method used in [13] was introduced by Chorin [8]. The vortex method is based on the Lagrangian formulation of Navier-Stokes equations. This class of method naturally concentrate discretization points in the region of interest. They use the vortex method in the flow region which is dominated by convective effects (far-field), whereas the finite difference method is used in the flow region where viscous diffusion effects are dominant (solid boundary). In order to guarantee the accuracy of the method, Biot-Savart technique is used instead of the vortex method in [14].

Chattot and Wang used the chimera approach for complex geometries [6]. It is a domain decomposition approach. The method is based on the crucial rule of domain decomposition, divide and conquer. In this approach, the complex shape is divided into several simpler ones and the grids are generated about these simple shapes, then they are overset to form a complete model. Information can be transmitted from one grid to another by means of interface boundaries. Because the grid for

each component is generated independently, the chimera approach allows for arbitrary adding or subtracting component grid, so refine mesh strategy can be used with domain decomposition method.

Another feature of the chimera approach is that each grid is updated in sequence, so it only requires enough memory to solve the largest grid and thus the problem can be handled with less memory requirement. Or, solving each grid independently creates the possibility of using a different processor for parallel computing. Furthermore, the possibility of using different schemes or different sets of equations and time steps for each grid also exist. For example, the Navier-Stokes equations can be solved on the finer inner grid subdomains and Euler equations can be solved on the coarser outer grid subdomains or primitive variables can be used on the inner grid to calculate pressure directly around the objects whereas stream function vorticity approach can be used for the outer grid.

Numerical simulation of incompressible flow in the primitive formulation have the same disadvantages. The major disadvantages are lack of dynamic equation and boundary condition for the pressure, need to satisfy incompressibility condition, implementation of no-penetration, no-slip boundary condition for the velocity. In the 2-D case, the first and second challenge can be overcome in the vorticity-stream function formulation. But at this time, vorticity boundary condition problem arises. For an unsteady flow at a large Reynolds number, a stable and efficient numerical method can be maintained by explicit treatment of the convection and diffusion terms. Some difficulties involved in the vorticity boundary condition can be circumvented. The no-slip boundary condition is enforced through a local vorticity boundary formula, such as Thom's or Wilkes' formula in the second order method. Besides, a difficulty arises in a multi-connected computation domain in determining the constants for the stream function on the boundary of the solid bodies. The derivation of these constants for multi-connected regions can be found in the literature [32, 23, 12, 22]. In this study, the method described in [22] is applied for the external flow problem.

When Navier-Stokes equations are discretized by using the finite difference method,

a nonlinear system of equations is obtained. The nonlinearity is mainly due to the convective terms in these equations. There are two main ways to solve these algebraic finite difference equations. The system can be solved by using well-known iterative techniques or this nonlinear system can be converted into a linear system by using Newton's method. After linearization, a linear solver can be used to find the unknowns at each Newton step. The iteration at each Newton step is called "inner iteration" and each Newton step is called "outer iteration".

The most important problem of the Newton method is that it has no global convergence property. Even in converged inner iterations, a problem may diverge due to the initial iterate that is far from the solution. Armijo rule and backtracking procedure may be used to remove the divergence of Newton method resulting from the bad initial iterate.

In the fluid flow problems mainly two types of mesh are used. They are structured and unstructured grids. In this study, structured grids are used. One of the crucial jobs in the numerical study is the determination of the grid size to which solution of the problem is not sensitive. If the problem is sensitive to moderate grid sizes, we should further decrease the grid sizes, which leads to extra computational load in terms of computational time and number of iterations. Refining mesh at critical regions and using coarse grids out of these critical regions will decrease the computational load.

The slow convergence performance and large storage requirements of direct and classical iterative solvers make them impractical to use in the large sparse system of equations. For these systems robust and fast iterative solvers must be used. Krylov methods such as GMRES, CGS, BI-CGSTAB, and QMR are among the fastest and the most robust iterative solvers. The basic process of each of the Krylov solvers is the projection of the original set of equations on to orthogonal sub-spaces.

2. NUMERICAL METHODS

The governing equations that define fluid flow are nonlinear, coupled PDEs. An analytical solution to these PDEs is generally not possible. For this reason, the governing equations of the fluid flow are discretized to a system of nonlinear algebraic coupled equations. Although PDEs which define the problem are valid for the whole solution domain, the system of algebraic equations obtained by discretization is valid only for discrete node. In other words, when the attention is focused on the values at the grid points, the continuous information from exact solution of PDEs is replaced with discrete values. The approximation of partial derivatives by algebraic equation is a fundamental concept of numerical schemes. The schemes by which the approximations to PDE can be developed are finite difference (FD) methods, finite volume (FV) methods, and finite element (FE) methods. The finite difference methods are used in conjunction with structured grids, whereas the finite volume and finite element methods are typically used in unstructured grids.

Finite difference is the simplest one among the three. In this thesis work finite difference discretization is used to obtain system of nonlinear algebraic coupled equations from the nonlinear, coupled PDEs. Problem geometries for this work are simple so body-fitted structured cartesian grid system is used and the application of the finite difference discretization for the structured grid system is easy. The derivation of finite difference formulations can be found in [15]. The finite difference method is based on the local approximations of the PDE with Taylor series expansions. The method is quite simple to define and implement especially for simple regions with uniform meshes. The resultant coefficient matrices from the discretization are well structured, which means they include only a few nonzero diagonal. The value at a grid point influences the distributed values of only in its immediate neighborhood. Finite difference has different schemes. In this work second order central difference formulations are used for the inner points and second order forward or backward formulations are applied for the solid boundaries. Second order spatial derivatives within the computational domain were approximated as follows;

$$\frac{\partial^2 f}{\partial x^2} = \frac{f_{i+1,j} - 2f_{i,j} + f_{i-1,j}}{(\Delta x)^2} \qquad central, \tag{2.1}$$

$$\frac{\partial^2 f}{\partial x^2} = \frac{2f_{i,j} - 5f_{i+1,j} + 4f_{i+2,j} - f_{i+3,j}}{(\Delta x)^2} \qquad forward, \tag{2.2}$$

$$\frac{\partial^2 f}{\partial x^2} = \frac{-f_{i-3,j} + 4f_{i-2,j} - 5f_{i-1,j} + 2f_{i,j}}{(\Delta x)^2} \qquad backward. \tag{2.3}$$

The system of the nonlinear algebraic equations is linearized with Picard or Newton methods. Picard's method is only a first-order method. For the stream function vorticity formulation, we assume solutions of the vorticity values and linear algebraic system is solved for the stream function values then vorticity values are solved with calculated stream function values. The procedure goes until the convergence criteria hold. Picard's algorithm has a reasonably large radius of convergence but the rate of convergence is generally fairly low. In order to enhance the rate of convergence, a second-order method, Newton's method, may be considered. In this thesis work, Newton's method is used to linearize the algebraic nonlinear equations.

A linear set of equations can be solved by either a direct or iterative method. Direct methods provide the solution after a fixed number of steps and they are less sensitive to the conditioning of the coefficient matrix. However, the main deficiency of the direct solvers is that they require the coefficient matrix to be stored in an ordered format. The limitations on CPU time and storage requirements make the use of direct solvers uneconomical. For large system, iterative methods are more efficient in that they require less storage and CPU time while giving comparable accuracy in solution.

2.1. Newton's Method

When the governing equations of the fluid flow are discretized to algebraic equations, the obtained system of equations is also nonlinear. This system can be solved with linear iterative solvers after linearization step. Newton's method is one of the most widely used methods in this manner.

The standard notation for system of N equation for N unknown;

$$F(x) = 0 \tag{2.4}$$

here F is called the nonlinear residual. If the components of F are differentiable, the Jacobian matrix $J(x)$ is defined by

$$J(x)_{i,j} = \frac{\partial F_i}{\partial x_j}(x). \tag{2.5}$$

The Newton sequence is

$$x_{n+1} = x_n - J(x_n)^{-1} F(x_n). \tag{2.6}$$

The interpretation of 2.6 is that at the current iterate x_n, F is modeled with a linear function

$$M_n(x) = F(x_n) + J(x_n)(x - x_n). \tag{2.7}$$

In equation 2.7 M_n is called the local linear model since its root is the next iteration [18]. If $J(x_n)$ is non-singular, then $M_n(x_{n+1}) = 0$ is equivalent to 2.6. The computation of a Newton iteration requires;

1. Evaluation of $F(x)$ and a test for termination

2. Approximate solution of the equation for Newton step s

$$J(x_n)s = -F(x_n) \tag{2.8}$$

3. Construction of $x_{n+1} = x_n + \lambda s$, where the step length λ is selected to guarantee decrease in $||F||$.

Item 2, how the Newton step is approximated, is the main reason for the variations in Newton's method. Computing the step may require evaluation and factorization of the Jacobian matrix or the solution of 2.8 by an iterative method.

The most efficient way in order to get rid of the difficulty in item 2 is to approximate it in some way. Approximating $J(x_n)$ is one of the these methods. Approximated $J(x_n)$ avoids computation of derivative and saves linear algebra work and matrix storage. The price for such an approximation is poor convergence of nonlinear steps [18] and difficulties on preconditioning.

2.1.1. Inexact Newton Method

Solving equation for Newton step approximately is the other option in order to overcome the difficulty in item 2. An inexact Newton method uses as a Newton step a vector s that satisfies the inexact Newton condition [9];

$$\|J(x_n)s + F(x_n)\| \leq \eta_n \|F(x_n)\| . \tag{2.9}$$

This formulation allows the use of an iterative linear algebra method. One first chooses η and then applies the iterative solver to 2.8 until s is determined for which the residual norm satisfies 2.9. In this context, η is often called a forcing term since its role is to force the residual of equation 2.8 to be suitably small.

The forcing terms η not only determine the asymptotic speed of convergence to a solution of equation 2.4 but also affect the efficiency and robustness of the algorithm

away from a solution. The problem named "oversolving" occurs when a very small η is chosen away from a solution. The choices for forcing term in [11] will tend to minimize oversolving while giving desirably fast asymptotic convergence.

Choosing a small value of η will make the iteration similar to Newton method, hence each linear step converge in fewer iterations. However computing Newton step with a small value of η can be very expensive.

Iterative methods for solving the equation for the Newton step would typically use equation 2.9 as a termination criterion. In this case overall nonlinear solvers are called Newton iterative solvers. Newton iterative methods are named by the particular iterative method used for the linear equation such as $Newton - Krylov$ methods.

2.1.2. Global convergence and damping term

The convergence theory of Newton's method is generally called as local convergence theory since this theory assumes that initial iterate x_0 is close to a solution.

In order to understand the importance of this assumption, let's try to find the root $x^* = 0$ of the function $F(x) = arctan(x)$ with an initial iterate $x_0 = 10$. With this initial iterate local convergence theory does not hold because it is too far from the root. After we apply Newton rule to the initial iterate the resultant four subsequent iterates are,

$$10,\ -138,\ 29 * 10^3,\ -1.5 * 10^8,\ 99 * 10^{16}.$$

At each step, Newton points in correct direction i,e., toward $x^* = 0$ but overshoots by larger and larger amounts. A simple solution of this problem is to reduce the step size by half until $\parallel F(x) \parallel$ has been reduced (Armijo rule).

In order to have more rigorous convergence, we first compute Newton direction

$$d = -J(x_n)^{-1}F(x_n). \tag{2.10}$$

Newton step will be a scaler multiple of Newton direction. To prevent the step from going too far, we should find the smallest integer $m \geq 0$ such that

$$\left\|F(x_n + 2^{-m}d\right\| \leq \left(1 - \alpha 2^{-m}\right) \|F(x_n)\|, \tag{2.11}$$

and step be $s = 2^{-m}d$ and $x_{n+1} = x_n + 2^{-m}d$. The condition in equation 2.11 is called the sufficient decrease of $\| F \|$. The parameter $\alpha \in (0, 1)$ should be selected in a way that sufficient decrease condition should hold as easy as possible. In this study $\alpha = 10^{-4}$ is chosen as parameter.

The line search method, also known as backtracking or damping, is similar to Armijo rule 2.11. Instead of taking half of the step size, steps are shortened as necessary until satisfactory steps are found. Some problems respond well to one or two reductions in the step length by a modest amount such as 1/2 and others require many such reductions but convergence may be better for a more aggressive step length reduction such as 1/10 [18]. In this study, if we do not have sufficient decrease after two reductions by halving, we build a quadratic polynomial model of

$$\phi(\lambda) = \|F(x_n + \lambda d)\|^2 \tag{2.12}$$

based on the interpolation of ϕ at the last three values of λ. The norm in the equation 2.12 is squared in order to make ϕ a smooth function that can be accurately modeled by a quadratic polynomial over a small range of damping term(λ). The next λ is chosen in a way that it will minimize the equation 2.12 with $safeguard$ limit;

$$\frac{1}{10} \leq \frac{\lambda_{m+1}}{\lambda_m} \leq \frac{1}{2}. \tag{2.13}$$

The line search terminates with the smallest $m \geq 0$ so that

$$\|F(x_n + \lambda_m d\| < (1 - \alpha\lambda_m) \|F(x_n)\| . \tag{2.14}$$

In this work, we use three-point parabolic model. In this approach, $\lambda_1 = 1/2$. To compute λ_m for $m > 1$, a parabola is fitted to the data from $\phi(0)$, $\phi(\lambda_m)$, and $\phi(\lambda_{m-1})$. The value of λ_m is chosen as the minimum of this parabola on the interval $[\lambda_{m-1}/10, \lambda_{m-1}/2]$.

We refer the reader to [17] for the details.

2.1.3. Termination of nonlinear iterations

The difference between the solution of a system and unknown values at a given iteration will give the error at that iteration. As we do not know the exact solution one can not know the real error at that iteration. However, in most cases, the norm of $F(x)$ can be used as a reliable indicator of the rate of decrease in error norm, as the iteration progresses.

2.2. Linear Solver

The direct and iterative methods can be used in order to solve the system of the linear algebraic system of equations with the aid of computer.

Cramer's Rule and Gaussian Elimination are the best known direct methods, enormous amount of arithmetic operation is required to produce a solution. This is the main disadvantage of the direct methods. The second obstacle to these methods is large storage requirement due to the full storage of coefficient matrix.

The term iterative methods refers to a wide range of techniques that use successive approximations to obtain more accurate solutions to a linear system at each iteration. Iterative methods used for solving the system of linear algebraic equations are simple and easy to program. The method works as follows;

- Guess an initial solution.
- Compute unknown values with guessed initial solution.
- Based on newly computed values, a newer solution is sought.
- The procedure is repeated until a specified convergence criterion has been reached.

The iterative methods may be categorized in to two groups, namely, the stationary and non-stationary methods. Stationary methods are older, simple to understand and implement, but usually not as effective. The same operation on the current iteration vector is applied in each iteration for the stationary method. Non-stationary methods are a relatively recent development. Their analysis are usually harder to understand and their implementation is difficult but they can be very effective. Except, Chebyshev iteration method, they are generally based on the idea of orthogonal vectors. Non-stationary methods have iteration dependent coefficients. The main difference between stationary and non-stationary methods is that in stationary methods coefficients are constant at each iteration but they change for non-stationary methods. Typically, constants are computed by taking inner product of residuals or other vectors arising from the iterative method.

There are many stationary iterative methods such as the Jacobi method, the Gauss-Seidel (GS) method, the Successive Overrelaxation (SOR) method, the Symmetric Successive Over Relaxation (SSOR) method. The main non-stationary iterative methods are Conjugate Gradient (CG) method, Generalized Minimal Residual (GMRES), Bi-Conjugate Gradient (BiCG), Quasi Minimal Residual (QMR), Conjugate Gradient Squared (CGS) Method, Bi-Conjugate Gradient Stabilized (BI-CGSTAB), and Chebyshev Iteration. These methods, except Chebyshev, are called Krylov sub-space methods, because they project the original set of equations on to a space called Krylov sub-space. Each of the Krylov sub-space solvers have their own characteristics. Choosing the most appropriate solver according to the type of the problem enables faster convergence. On the other hand, in some cases choosing an inappropriate method may even determine convergence or divergence property of the problem. For instance, CGS can be applied to non-symmetric matrices but CG can not.

There are many advantages of using Krylov methods instead of stationary iterative methods; they do not need a priori estimates of parameters. The GMRES method gives the residual as small as possible. One of the most important characteristics of the GMRES procedure for linear systems is that the matrix-free implementation can be used, therefore coefficient matrix does not need to be stored fully. Removing the storage restriction is one of the most vital improvements for the problems which have a coefficient matrix that is out of the computer memory. Generally, fluid flow problems require fine meshes and discretization gives large coefficient matrix so there is potential risk of storage restriction for the fluid flow problem.

In this work the discretized form of the physical system is large, sparse and most of the time non-symmetric. The slow asymptotic convergence performance and large storage requirements of direct and classical stationary iterative solver make them impractical. This work requires a fast and robust iterative solver. Krylov methods, GMRES and BI-CGSTAB are applicable to nonsymmetric systems so both of these methods are tested in the internal flow problem. For the external flow problems, only GMRES is used as iterative solver. GMRES must accumulate the history of the linear iteration as an orthonormal basis for the Krylov subspace. This is an important property of the method because for large problems it exhausts the available memory. Any implementation of GMRES must limit the size of the Krylov sub-space. GMRES(m) does this by restarting the iteration when the size of Krylov sub-space exceeds m. Both of these solvers are applicable to non-symmetric matrices which is the case in the thesis problems. A brief summary of the properties of these methods is given below [1].

Bi-conjugate Gradient Stabilized (BI-CGSTAB):

- Applicable to nonsymmetric matrices.
- Computational costs per iteration are similar to Bi-CG and CGS but the method doesn't require the transpose matrix.
- An alternative for CGS that avoids the irregular convergence patterns of CGS while maintaining approximately the same speed of convergence; as a result we often observe less loss of accuracy in the updated residual.

Generalized Minimal Residual (GMRES):

- Applicable to nonsymmetric matrices.
- GMRES leads to the smallest residual for a fixed number of iteration steps but these steps become increasingly expensive.
- In order to limit the increasing storage requirements and work per iteration step, it is necessary to restart. When to do so depends on coefficient matrix and on the right-hand side; it requires skill and experience.
- GMRES requires only matrix vector products with the coefficient matrix.
- The number of inner products grows linearly with the iteration number up to the restart point. In an implementation based on a simple Gram-Schmidt process, the inner products are independent so together they imply only one synchronization point. A more stable implementation based on modified Gram-Schmidt orthogonalization has one synchronization point per inner product.

2.2.1. Newton-Krylov Methods

Recall from section 2.1.1 that an inexact Newton method approximates the Newton direction with a vector d such that

$$\|J(x_n)d + F(x_n)\| \leq \eta \|F(x_n)\| \tag{2.15}$$

the parameter η is called the forcing term.

Newton iterative methods realize the inexact Newton condition 2.15 by applying a linear iterative method to the equation for the Newton step and terminating that iteration when 2.15 holds. This linear iteration is called as inner iteration and nonlinear iteration is called outer iteration. Krylov sub-space based linear solver is used in the Newton-Krylov methods, as the name suggests.

2.2.2. Krylov Methods for Solving Linear Equations

The solution of a linear system $Ax = b$ is obtained with Krylov methods as a sum of the form

$$x_n = x_0 + \sum_{j=0}^{j=n-1} \gamma_n A^n r_0 \tag{2.16}$$

where $r_0 = b - Ax_0$ is the initial residual and x_0 is the initial iterate. As we have no prior knowledge of the direction the most sensible initial iterate is $x_0 = 0$. We can express this in a compact form as $x_n \in \kappa_n$, where nth Krylov sub-space is

$$\kappa_n = span(r_0, Ar_0, ..., A^{n-1}r_0).$$

Krylov methods build an iterate in the appropriate Krylov sub-space by evaluating matrix vector products. Since Jacobian matrix is sparse, analytical Jacobian-vector products are highly time consuming and require huge memory. The directional derivative or compressed storage scheme is used to cope with these difficulties.

2.2.3. Directional Derivative

The Jacobian-vector product is easy to approximate with a forward difference directional derivative for nonlinear equations. The forward difference directional derivative at x for direction w is

$$D_h F(x:w) = \begin{cases} 0 & w = 0 \\ \|w\| \frac{F(x+\sigma(x,w)hw/\|w\|)-F(x)}{\sigma(x,w)h} & w \neq 0 \end{cases} \tag{2.17}$$

We first scale w to be a unit vector and take a numerical directional derivative in the direction $w/\parallel w \parallel$. If h is roughly the square root of the error in F, a difference increment is added in the forward difference to make sure that the appropriate low-order bits of x are perturbed. So we multiply h by $\sigma(x, w) = max(|x^T w|, ||w||)sgn(x^T w)/||w||$.

The applications of this method can be found in [18]

2.2.4. Compressed Storage Schemes

The Jacobian matrices arising from the discretization of partial differential equations have characteristic sparsity patterns. Storing only the non-zero elements of the matrix instead of storing each element of the matrix using one of the compressed storage schemes decreases the memory load of the computer considerably. Also defining the matrix vector multiplications is required for the linear solvers, and accordingly faster convergence can be achieved. There are various kinds of compressed storage schemes. One of them is compressed column storage (CCS). According to this scheme the nonzero elements of the matrix are stored column by column in a vector. The row number of each element is written at another vector. And a third vector indicates at which element a new column starts. In this work matrix-free algorithms are modified using the CCS scheme in order to apply SGS preconditioner in a fast way to the matrix-free methods.

2.3. Preconditioning

The convergence of an iterative method usually depends on the spectrum of the coefficient matrix. Even when the advanced and robust iterative methods are used, in some cases, because of the coefficient matrix's spectrum, these methods can be inefficient to solve the system. A preconditioner technique can be used to improve efficiency and robustness of iterative techniques. Preconditioning is simply a means of transforming the original linear equations system into one which has the same solution, but which is likely to be easier to solve with an iterative solver. A good preconditioner improves the convergence rate of the iterative method and overcomes the extra cost of constructing and applying the preconditioner.

Using a preconditioner in an iterative method incurs some extra work, both in the production of the preconditioner matrix and the application of the preconditioner matrix at each step. There is a trade-off between the cost of the construction and

application of the preconditioner and the gain in the convergence speed [1]. A good preconditioner provides a decrease in the computation time and they can make a solver converge whereas the solver could diverge without implementation of the preconditioner.

Accordingly, a preconditioner must be as close as possible to the coefficient matrix and the preconditioner system $Mx = c$ must also be easy to solve. Taking the preconditioner matrix equal to the coefficient matrix A would enable the solver to converge in a single iteration. However, the time required for solving the system $Mx = c$ would be equal to solving the real system $Ax = b$. On the other extreme, taking the preconditioner matrix equal to the identity matrix so that the system $Mx = c$ can easily be solved would not result in any reduction in the number of iteration steps. So the ideal preconditioner must be between the two extreme cases.

Preconditioning the matrix A means multiplying A from the right, left, or both sides by a preconditioner M. Of course, preconditioning can be done in a matrix-free manner. One needs only a function that performs a preconditioner-vector product.

In left preconditioning the preconditioner matrix M is applied to the original equation ($Ax = b$) from the left hand side. The Krylov methods are then applied to the new equation system;

$$M_{left}^{-1} A x = M_{left}^{-1} b. \tag{2.18}$$

The spectral properties of the preconditioned system ($M_{left}^{-1} A$) may be more favorable than the original one (A).

It is also possible to transform the system with right preconditioning, as given by the equation (2.19)

$$A M_{right}^{-1} y = b \quad where \quad x = M_{right}^{-1} y. \tag{2.19}$$

The transformed system is first solved for y, and then, the unknown vector x is computed by the relation given in the equation (2.19).

In left preconditioning, the preconditioner is applied directly to the residual vector so, it may cause the algorithm to stop prematurely or with delay [29]. For this reason right preconditioning is preferred in this work.

The other option, split preconditioning, can be used if the preconditioner matrix is of the form

$$M = LU. \tag{2.20}$$

According to this option

$$L^{-1}AU^{-1}y = L^{-1}b \quad where \quad x = U^{-1}y. \tag{2.21}$$

In this work three different kinds of preconditioners are used. These are Jacobi, symmetric Gauss-Seidel (SGS) and incomplete LU decomposition (ILU). Applying preconditioning generally requires explicitly matrix-vector products, so preconditioning in the matrix-free algorithm needs a new definition of the matrix vector multiplication routine. Inserting some preconditioner in to the matrix-free algorithm may be a very challenging task. The characteristics of the problem will determine the storage scheme. In the L-shaped flow problem, there are no storage restrictions and convergence problems. Due to this flexibility, full matrix storage, compressed column storage, and matrix-free procedure are used as the storage scheme in the solution of the problem and Jacobi, SGS, ILU are used as preconditioner. In the external flow problem, the main barrier to the solution is the memory restriction. Since the inexact Newton method with backtracking procedure removes the convergence problem almost completely, only matrix-free algorithm is used to handle memory restriction and only Jacobi preconditioner is used.

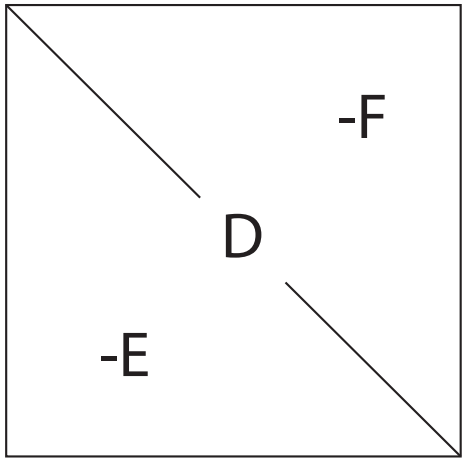


Figure 2.1. Initial partitioning of matrix A

[29]

2.3.1. Jacobi

This simplest preconditioner is also called diagonal scaling, since each row is scaled with respect to the entries in the diagonals

$$M = D. \tag{2.22}$$

2.3.2. Symmetric Gauss-Seidel (SGS)

In this method the preconditioner matrix is composed of the product of a lower triangular matrix and an upper triangular matrix

$$M = (D - E)D^{-1}(D - F). \tag{2.23}$$

In order to solve the system $Mx = p$ the factorized form of the preconditioner matrix is utilized instead of taking the inverse of the matrix. The solution procedure is given below.

- Solve $(D-E)D^{-1}y = p$ for y by forward substitution.
- Solve $(D-F)x = y$ for x by backward substitution.

2.3.3. Incomplete LU Decomposition (ILU)

A broad class of preconditioner is based on the incomplete factorization of the coefficient matrix. A general ILU factorization process computes a sparse lower triangular matrix L and a sparse upper triangular matrix U so that the residual matrix $R = LU - A$ which has zero entries in some locations. The exact LU factorization would require a similar amount of operation like Gaussian elimination and would therefore be an inefficient solution method. For this reason this factorization is carried out only at some locations in the matrix. ILU has also some versions based on the accuracy of the factorization. ILU(0) is a preconditioner with zero fill-in. The zero in parentheses indicates that the zero pattern of the decomposed matrices precisely fits in to the zero pattern of the Jacobian matrix. For ILU(0) the LU factorization is carried out only at the nonzero elements of the original matrix

$$M = LU. \tag{2.24}$$

In this method the system $Mx = c$ is solved using the factorized structure of the preconditioner matrix. The solution algorithm is given below.

- Solve $Ly = c$ for y by forward substitution.
- Solve $Ux = y$ for x by backward substitution.

The accuracy of the ILU(0) incomplete factorization may be in sufficient for an adequate rate of convergence. Allowing some fill-in increases accuracy of ILU factorizations. More accurate factorization yields more efficient preconditioner ILU(l) where l is the order of fill-in. The details and applications of ILU(l) can be found in [29].

2.4. Domain Decomposition

As multi processing technology has been developing, new types of numerical methods that use parallel processing are emerging. Among these methods domain decomposition methods are the best known since there is a lot of research on this subject. DDM methods which depend on the strategy to divide and conquer, combines the ideas from PDEs, linear algebra, mathematical analysis and techniques from graph theory [29].

The principle of DDM is to split a complex computational domain into smaller and simpler subdomain problems for which the efficient solver may be used to compute the local simplified solutions collaboratively. The most important advantage of DDM is the straight-forward applicability for parallel computing. The other major advantages are easy handling of the global solution domains of irregular shape, the possibility of using different numerical techniques in different subdomains, special treatment of singularities and allowing the modeling of the equations of the problem differently. The mathematical model can be different for different subdomains as in the problem of the flow over a flat plate, Navier-Stokes equations are used in the subdomains within the boundary layer thickness whereas Euler equations are used in the subdomains out of the boundary layer thickness.

Any DDM is based on the assumption that the given computational domain, say Ω is partitioned into subdomains $\Omega_i, i = 1, ..., P$, which may or may not overlap. Next the original problem can be reformulated upon each subdomain Ω_i, yielding a family of subproblems of reduced size that are coupled one to another through the values of the unknown solution at subdomain interfaces [28].

There are various choices for partitioning computational domain into subdomains. The total number of available processor for parallel computing is the main restriction on the partitioning of the domain. The number of subdomain can not exceed the number of available processors.

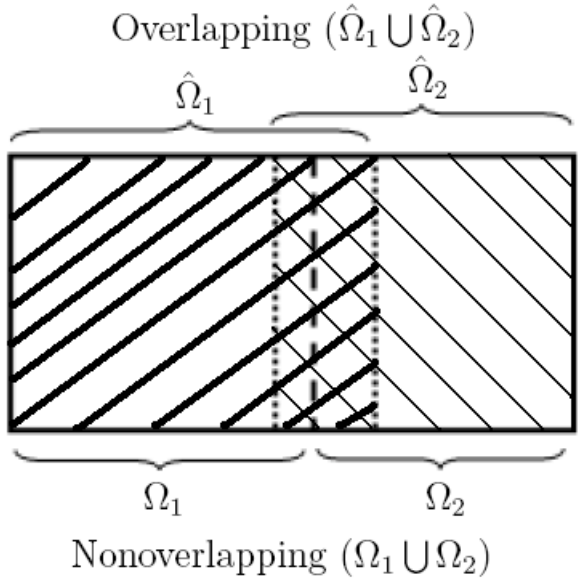
Generally, there are two kinds of approaches depending on whether the subdo-

mains overlap with one another (Schwarz methods) or are separated from one another by interfaces (Schur Complement, iterative substructuring) [1]. The overlapping or non-overlapping types of subdomains can be seen in the Figure 2.2. In this study overlapping DDM will be used. In numerical computation, the discretization of the subdomain will produce two different cases in terms of grids matching. In the special case, the grid points from two subdomains line up exactly in the overlap region and in the more general case, the grids do not match between the subdomains. Two types of grids are visualized in Figure 2.3. Interpolation and restriction processes used for solution of problems with non-matching grids are more complicated to be used for the matching case.

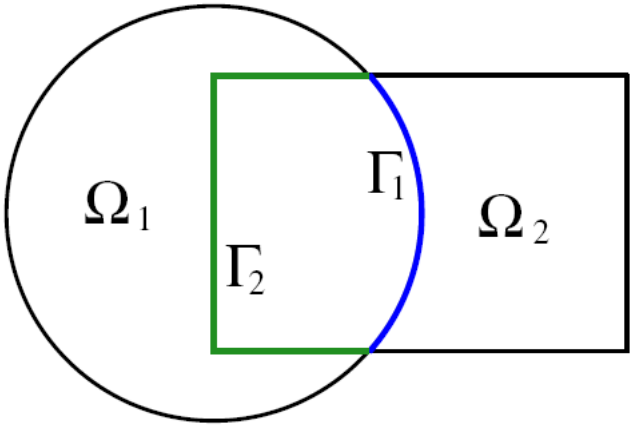
Subdomains can have different grid sizes. The general solution domain is discretized to coarse grid to obtain initial guesses for the subdomain in which effects of physical event can be seen more clearly. This critical subdomain is discretized to the fine grids to observe the changes in the physical properties more accurately (Figure 2.3-a). The correction of the physical properties in the coarse grids can be performed by using results of the subdomain with fine grids and restriction procedure. This feature gives the chance of obtaining more correct results about the critical region without solving the whole problem in the fine grids.

2.4.1. Overlapping Domain Decomposition

An overlapping domain decomposition technique is also known as the Schwarz Alternating method. In this method, there are overlapped regions between the neighboring subdomains. Schwarz used this idea to prove the existence for a solution of the Dirichlet problem on irregular regions. While solving the problem, Schwarz partitioned the complex domain (Ω) into overlapping subdomains $\Omega = \Omega_1 \cup \Omega_2$ (Figure 2.2-b) of smaller size and simpler geometries so as to reduce the original problem to the iterative solution of problems of the same type but on simpler subdomains.



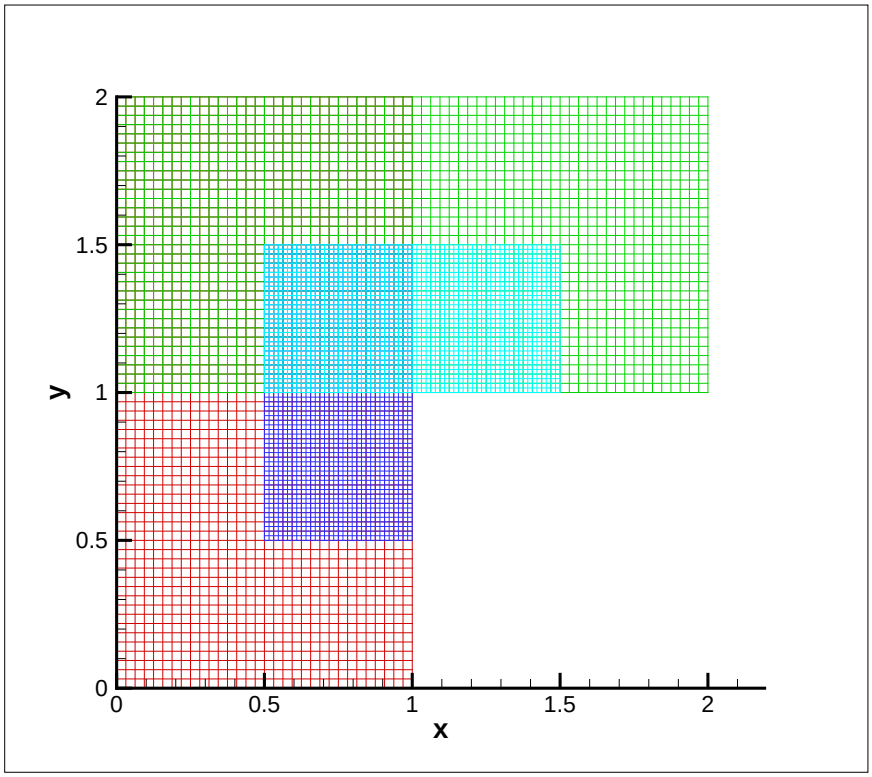
(a) Two rectangular domains



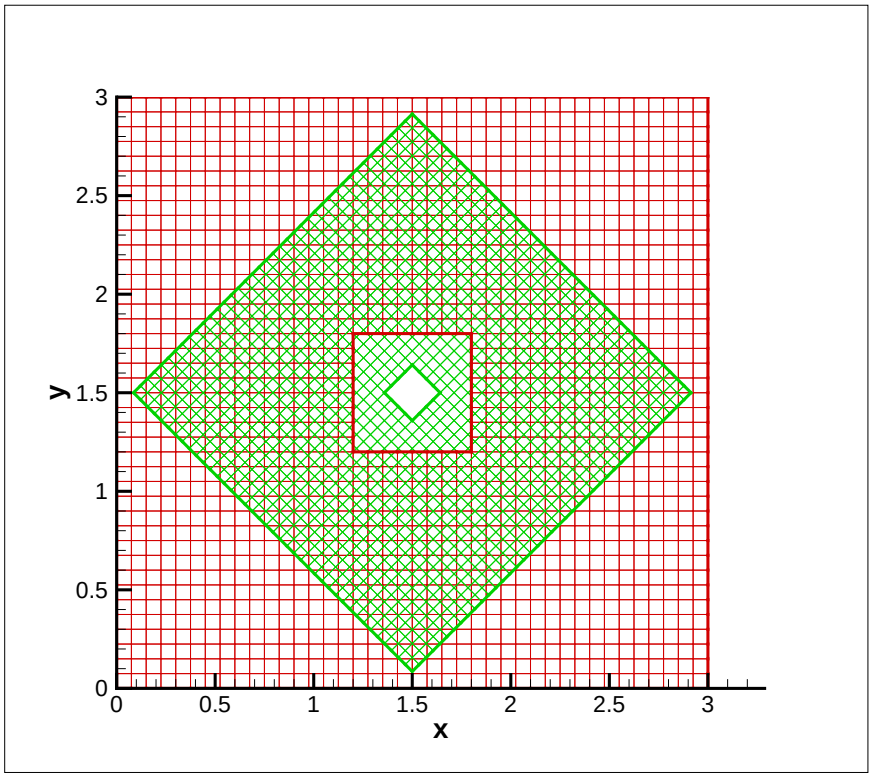
(b) A circle and a rectangular domain

Figure 2.2. Schematic representation of overlapping and non-overlapping domain decomposition participants of main domain (Ω)

[31]



(a) Matching grids



(b) Non-matching grids

Figure 2.3. Schematic representation of matching and non-matching grids

Let $\partial\Omega$ denote the boundary of Ω and note that Ω_is do not include their boundaries. Domain with its boundary means closure of the domain is denoted as $\bar{\Omega} = \Omega \cup \partial\Omega$. The rest of the subdomain boundaries are denoted by $\partial\Omega_i \backslash \Gamma_i$. In order to clarify alternating technique, we need to introduce some notations. Let u_i^n denote the approximate solution of $\bar{\Omega}$ after n iterations, and $u_1^n|_{\Gamma_2}$ be the restriction of u_1^n to Γ_2 and similarly $u_2^n|_{\Gamma_1}$ is the restriction of u_2^n to Γ_1. In order to have artificial boundary condition on Ω_1, we start with an initial guess for Ω_2 and for non-matching grid case we need to interpolate values from the nodes in the interior of Ω_j to the nodes on the artificial boundary Γ_i. This operation is denoted by $\mathrm{I}_{\Omega_j \to \Gamma_i}$. The algorithm of the Alternating Schwarz Method is as follows [31].

- $w_1^0 \leftarrow 0$
- For $n = 1, 2, ..$

  solve for u_1^n:

$$A_1 u_1^n = f_1 \qquad \text{in } \Omega_1$$
$$u^n_{\partial\Omega_1 \backslash \Gamma_1} = g_1 \qquad \text{on } \partial\Omega_1 \backslash \Gamma_1$$
$$u^n_{\Gamma_1} = w_1^{n-1} \qquad \text{on } \Gamma_1$$

  $w_2^n \leftarrow \mathrm{I}_{\Omega_1 \to \Gamma_2} u^n_{\Omega_1}$

  solve for u_2^n:

$$A_2 u_2^n = f_2 \qquad \text{in } \Omega_2$$
$$u^n_{\partial\Omega_2 \backslash \Gamma_2} = g_2 \qquad \text{on } \partial\Omega_2 \backslash \Gamma_2$$
$$u^n_{\Gamma_2} = w_2^{n-2} \qquad \text{on } \Gamma_2$$

  $w_1^n \leftarrow \mathrm{I}_{\Omega_2 \to \Gamma_1} u^n_{\Omega_2}$

  If $\left\| u_1^n - u_1^{n-1} \right\| \leq tol_{\Omega_1}$ and $\left\| u_2^n - u_2^{n-1} \right\| \leq tol_{\Omega_2}$ Stop
- end for

Figure 2.4. Algorithm of the Alternating Schwarz

When the program terminates, $u^n_{\Omega_1}$ contains the solution for Ω_1 and $u^n_{\Omega_2}$ contains the solution for Ω_2. For the overlapped region we can use either of the two solutions or some average, as the two solution will both converge to the same value when the mesh size is fine enough.

Termination of the alternating Schwarz method is controlled by the norm of the difference for successive solution vectors. One can control only artificial boundaries for termination also.

As a summary, overlapping DDM operates by an iterative procedure, where PDE is repeatedly solved within each subdomain. For each subdomain, the artificial internal boundary conditions is provided by its neighboring subdomains. The convergence of the solution on these internal boundaries ensures the convergence of the solution in the global solution domain. One further useful feature of the alternating Schwarz algorithm is that subdomains can have their own grids independently. If the mesh points over the artificial boundaries coincide, generally non-matching grids can work together easily. There are two ways of implementing the overlapping domain decomposition methods depending on the iteration logic.

2.4.1.1. Multiplicative Schwarz Procedure. Schwarz proposed the classical alternating DDM for the two subdomain case, namely, a circle and a rectangle. Multiplicative Schwarz procedure is the extended version of alternating method for more than two subdomains. We decompose the global solution domain into a set of p domains $\Omega = \Omega_1 + \Omega_2 + ... + \Omega_p$. $\aleph_i$ denotes the set of the indices of the subdomains that are neighbor to subdomain i, such that $j \in \aleph_i \Rightarrow \Omega_i \bigcup \Omega_j \neq \emptyset$

According to this procedure, for the first iteration, an initial guess for the boundary conditions of the first subproblem will be required. For the other subproblems a combination of initial guess and most recent solution will be used until the first iteration is completed. In the next iterations the boundary conditions will be updated using only the most recent solution. In other words, the artificial Dirichlet boundary conditions are updated by receiving the latest solutions from neighboring subdomains. Thus, a sequential solution algorithm is obtained.

Although it has a good rate of gaining the convergence, it is rarely used for parallel

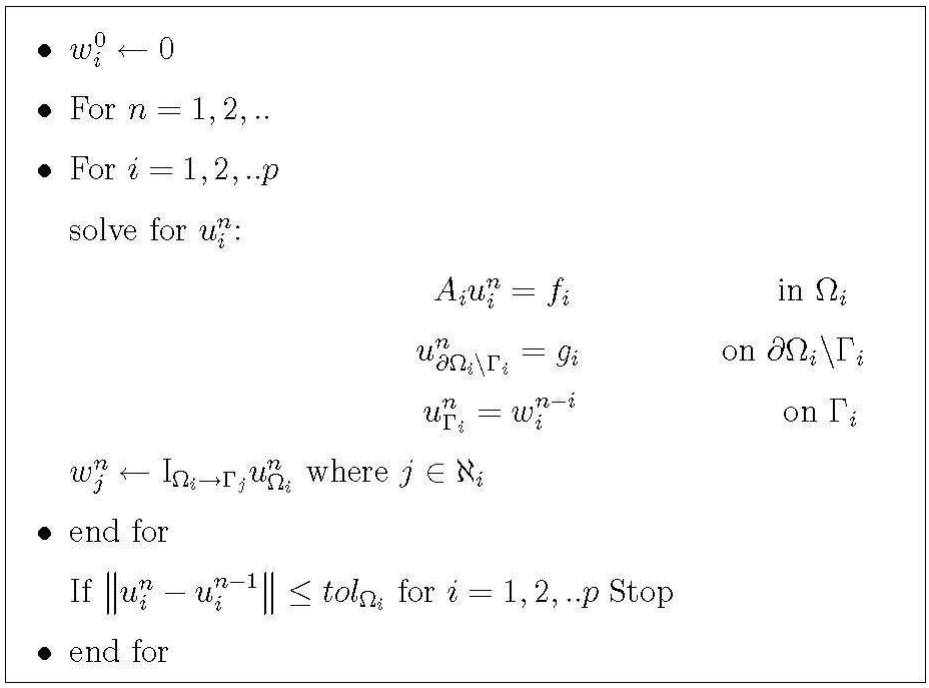


Figure 2.5. Algorithm of the Multiplicative Schwarz

computing because simultaneous solution of more than one domain is not allowed due to its sequential property. This is because each iteration for the sub-step is restricted to finding boundary conditions which depend on the recent solutions of the neighbors. Due to this property it reminds the block Gauss-Seidel iteration algorithm.

In order for the multiplicative Schwarz method to be able to run on a multi-processor platform, it is necessary to modify the method by using a multi-coloring scheme. More specifically, a small number of different colors and values are used for each subdomain with one color. The result of multi-coloring is that any pair of neighboring subdomains always has different colors. In a modified multiplicative Schwarz method, all the subdomains that have the same color can carry out the subdomain can be solved simultaneously. Therefore, multiple processors are allowed to work simultaneously in subdomains with the same color. It should be noted that the modified multiplicative Schwarz method is not so flexible as the additive Schwarz method when it comes to parallel implementation. The convergence speed of the modified multiplicative Schwarz method becomes slower compared to that of the standard multiplicative Schwarz method [3].

2.4.1.2. Additive Schwarz Procedure. Another type of overlapping domain decomposition method, which inherently promotes parallel computing, is called Additive Schwarz Method. The difference between the multiplicative Schwarz method and the additive counterpart lies in the way the artificial boundary condition is updated . The artificial Dirichlet condition for the n^{th} iteration is updated in a Jacobian fashion, using solutions from all the relevant neighboring subdomains of previous iteration $n-1$.

- $w_i^0 \leftarrow 0$
- For $n = 1, 2, ..$
- For $i = 1, 2, ..p$

  solve for u_i^n:

$$A_i u_i^n = f_i \qquad \text{in } \Omega_i$$
$$u^n_{\partial\Omega_i \backslash \Gamma_i} = g_i \qquad \text{on } \partial\Omega_i \backslash \Gamma_i$$
$$u^n_{\Gamma_i} = w_i^{n-i} \qquad \text{on } \Gamma_i$$

- end for

  $w_j^n \leftarrow \mathrm{I}_{\Omega_i \rightarrow \Gamma_j} u^{n-1}_{\Omega_i}$ where $j \in \aleph_i$

  If $\left\| u_i^n - u_i^{n-1} \right\| \leq tol_{\Omega_i}$ for $i = 1, 2, ..p$ Stop
- end for

Figure 2.6. Algorithm of the Additive Schwarz

In the first iteration, the boundary conditions are determined using an initial guess for all the subdomains. For the next iterations the solution from the previous iteration is used to update boundary conditions. The convergence speed of the additive Schwarz is less than the multiplicative one, since in this method we do not use advance of last solution for the artificial boundaries. However, the method is suitable for parallel implementation as each subproblem can be solved independently from the others [7]. Therefore, the subdomains which are solved with the additive Schwarz method can be carried out completely independently, thus this makes the method suitable for parallel processing. Although the additive Schwarz method suits well to parallel computing, it should be noted that its convergence property is inferior to that of the multiplicative

Schwarz method. In case of convergence, the additive Schwarz method uses roughly twice as many iterations as that of the standard multiplicative Schwarz method. This is not surprising when the Schwarz methods are compared to their linear system solver analogues; multiplicative Schwarz is a block Gauss-Seidel approach, whereas additive Schwarz is a block Jacobi approach [3].

Some problems, such as high Reynolds number flows, may be highly nonlinear. Even with global converging strategies such as linesearch, or trust region, the method often stagnate at local minima of the residual. Additive Schwarz Preconditioned Inexact Newton Algorithm (ASPIN) was introduced [19] to precondition the system of nonlinear equations.

2.4.2. Non-overlapping Domain Decomposition

A non-overlapping domain decomposition technique is also known as the Schur substracting method. In contrast to overlapping domain decomposition, there is no overlap region between neighboring subdomains. Non-overlapping partitioning of the global domain into two simple subdomains can be seen in Figure 2.2-a. Obviously, Γ which is the internal boundary condition for both subdomains divides the original domain into two subdomains. In other words, only nodes on the artificial boundary are common for both of the subdomains.

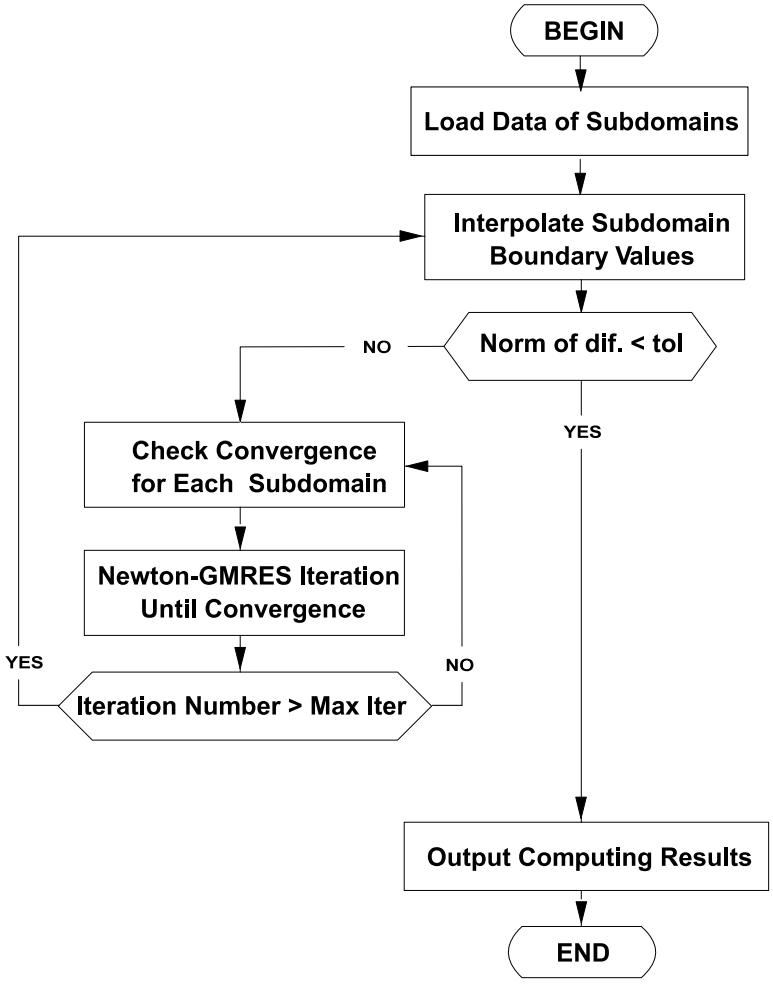


Figure 2.7. Schematic representation of solution algorithm of domain decomposition method for GMRES solver

3. COMPUTATIONAL MODELING

The governing equations for fluid flow are conservation of mass (continuity) and conservation of momentum (Navier-Stokes equations). They are used to determine the fluid velocity and pressure distribution in a domain. Navier-Stokes equations are well-known governing equations of the fluid flow. For applications for which the density remains uniform throughout the domain, the assumption of incompressible flow is invoked. No fluid is truly incompressible since even liquids can have density change through application of sufficient pressure. But density changes in a flow will be negligible if the Mach number (Ma) of the flow is small. Mach number is the ratio of the fluid velocity to speed of sound. The condition for incompressible flow is given by $Ma < 0.3$. If this condition hold, density is constant and no longer an unknown. Furthermore, variations in the coefficients of the viscosity are essentially negligible and it is assumed constant as well. The divergence of velocity is zero for the incompressible flow. Moreover, time derivatives in the equations can be dropped because a steady state solution of the problem is sought. Further simplification of two dimensional problems, like in this work, can be done by using stream function vorticity transport equations. When stream function vorticity transport equations are used, the continuity equation is automatically satisfied. After all, there are two governing equations and boundary conditions for each problem.

3.1. Internal Flow Problem

3.1.1. Problem Statement

In the first test case, domain decomposition method is applied to an internal flow example. The chosen flow geometry is an L-shaped domain with re-entrant corner. The re-entrant corner has mathematical singularities. Applying domain decomposition technique will remove this singularity to some extent. Our objective in solving the case flow in L-shaped domain is to adapt multiplicative Schwartz decomposition method for solving incompressible Navier-Stokes equations. In this problem there is a fluid

inflow into the domain in y direction and the inflow has a parabolic profile. The domain which has inflow boundary condition is called first domain and domain which has outflow boundary condition is called the second domain as in Figure 3.1. The convergence of the problem depends on the length of the second domain since outflow boundary condition forces the flow to uniform outflow and the flow needs enough length to develop.

Adaptivity is an important means to improve the efficiency of numerical methods. There are mainly two types of adaptive approaches, which are predefined (also called static) refinement and self-adaptive (also called dynamic) refinement. In the static approach the grid refinement is determined before the solution process is initiated whereas in dynamic approach grid refinements are carried out dynamically during the solution process. Adaptive approaches are particularly useful if the solution exhibits singular behavior. In many cases, these singularities occur locally so local adaptation of grid is appropriate. In the L-shaped problem, the peculiarities of the shape of the domain (re-entrant corner) create singularities so we used static adaptive approach.

Adaptive grid refinement technique starts with a global coarse grid covering the whole computational domain. Finer grids are introduced only in parts of the domain in order to improve the accuracy of the solution [33].

After solving the flow problem described by Figure 3.1, we realize that we need finer grids around the re-entrant corner and the upper left corner since the the changes around these sharp corners are much more than any other place in the flow field and vorticity generation occurs around these corners. The flow geometry with adaptive grids can be seen in Figure 3.2.

The boundaries of the local refined areas (artificial boundaries) are discussed in section 3.1.3

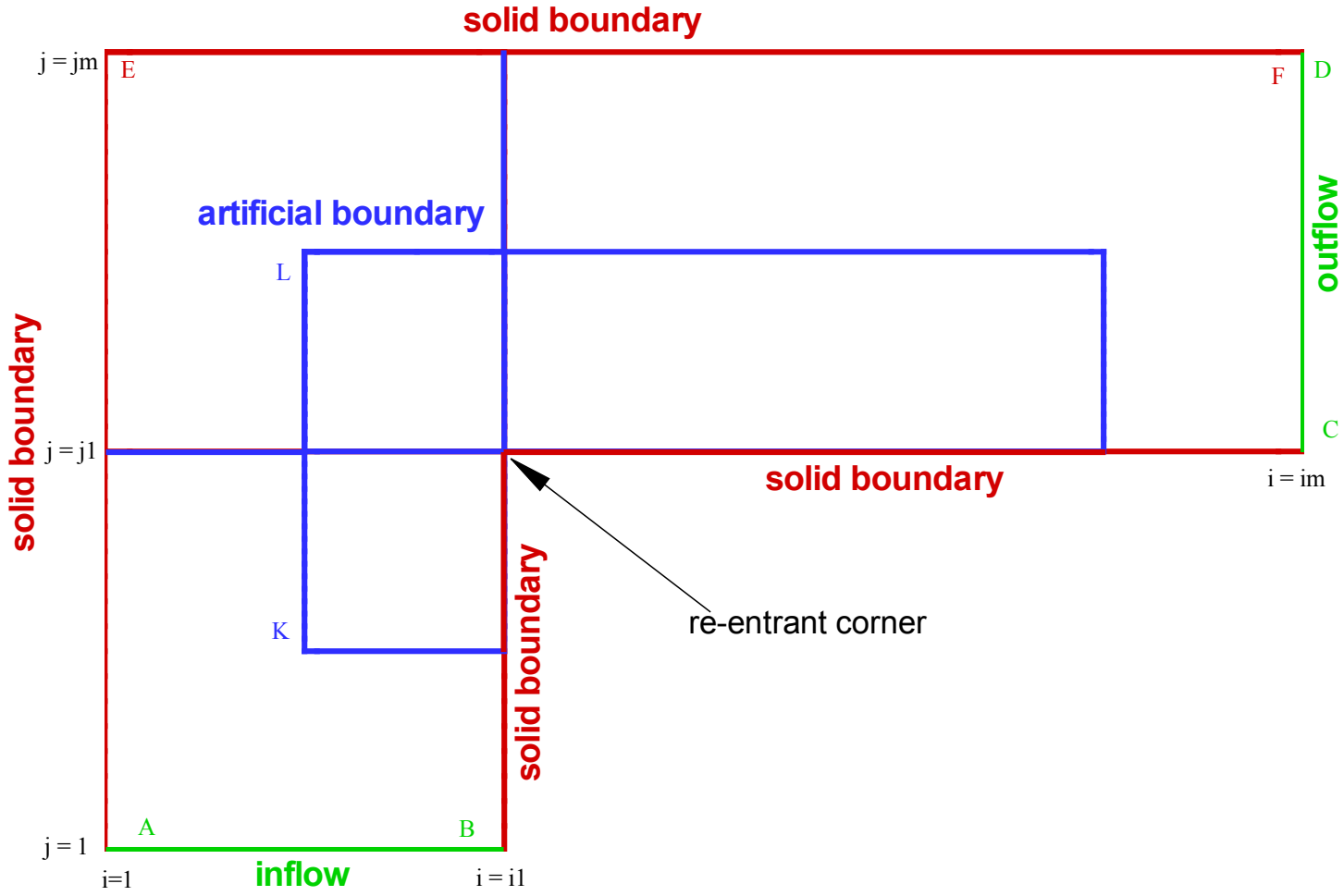


Figure 3.1. Geometry and boundary conditions of L-shaped domain

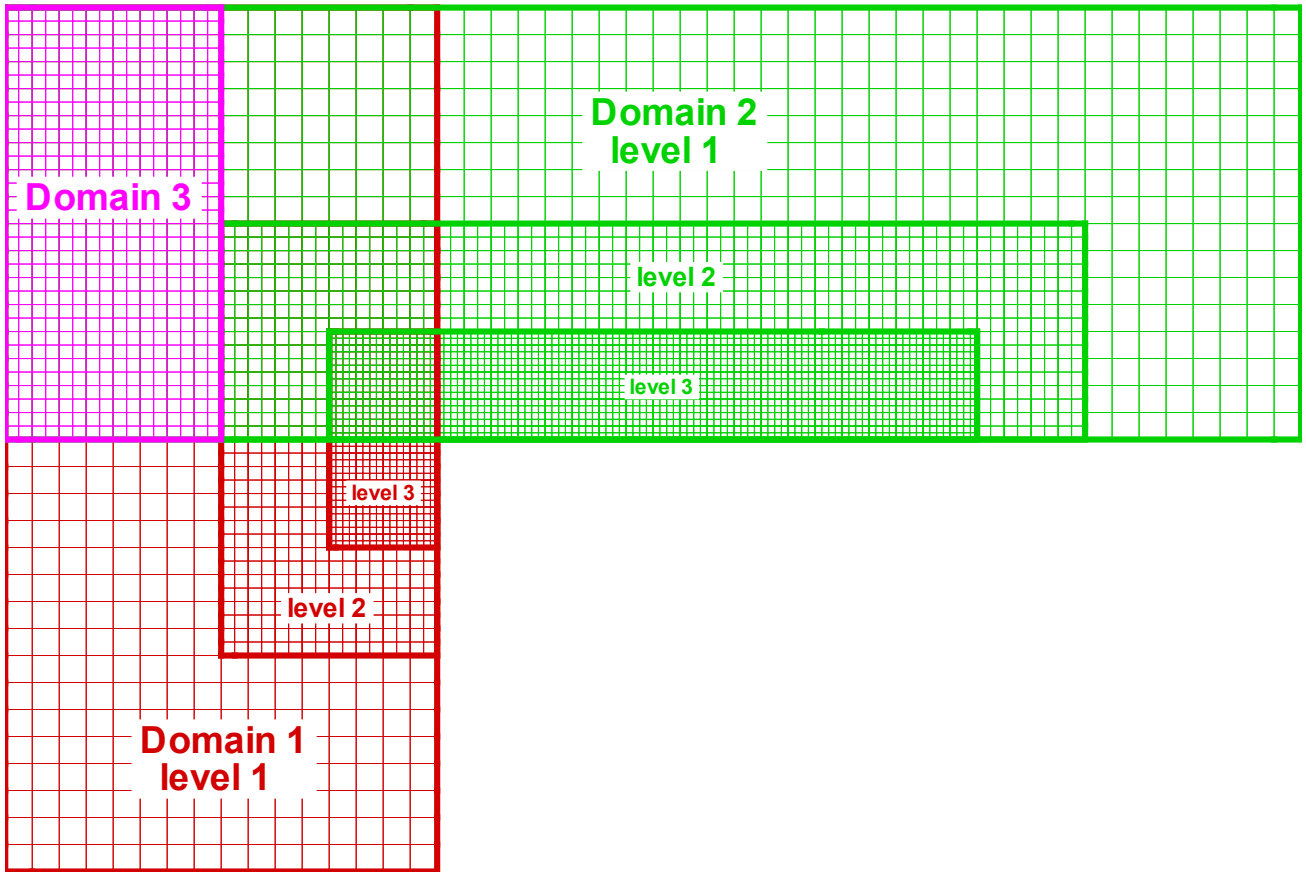


Figure 3.2. Geometry adaptively meshed L-shaped domain

3.1.2. Discretization of Governing Equations

There are two different formulations of these equations based on the dependent variables used. There are many difficulties in numerical simulation of the incompressible flow in primitive formulation, including the enforcement of the incompressibility constraint, lack of dynamic equation and boundary condition for the pressure. In this work stream function vorticity formulations are used. These are derived from the primitive variable formulations of the governing equations using the definitions of the vorticity and the stream function. The derivation of the equations for the cartesian coordinates can be found in [15].

The stream function and the vorticity transport equations for the cartesian system are shown respectively;

$$\nabla^2\Psi + \Omega = 0 \tag{3.1}$$

$$\nabla^2\Omega - Re\left[\frac{\partial\Psi}{\partial y}\frac{\partial\Omega}{\partial x} - \frac{\partial\Psi}{\partial x}\frac{\partial\Omega}{\partial y}\right] = 0. \tag{3.2}$$

Second order central finite difference formulations are used for the discretization of these equations except at the solid and computational domain boundaries;

$$\left[\frac{(\Psi_{i+1,j} - 2\Psi_{i,j} + \Psi_{i-1,j})}{\Delta x^2} + \frac{(\Psi_{i,j+1} - 2\Psi_{i,j} + \Psi_{i,j-1})}{\Delta y^2}\right] + \Omega_{i,j} = 0, \tag{3.3}$$

$$\left[\frac{(\Omega_{i+1,j} - 2\Omega_{i,j} + \Omega_{i-1,j})}{\Delta x^2} + \frac{(\Omega_{i,j+1} - 2\Omega_{i,j} + \Omega_{i,j-1})}{\Delta y^2}\right] - Re\left[\frac{\Psi_{i,j+1} - \Psi_{i,j-1}}{\Delta y}\frac{\Omega_{i+1,j} - \Omega_{i-1,j}}{\Delta x} - \frac{\Psi_{i+1,j} - \Psi_{i-1,j}}{\Delta x}\frac{\Omega_{i,j+1} - \Omega_{i,j-1}}{\Delta y}\right] = 0. \tag{3.4}$$

3.1.3. Boundary Conditions

There are five different boundary condition applied in the model problem. The Figure 3.1 shows these boundaries. At this section we will show how these conditions are inserted into numerical code.

3.1.3.1. Inflow. As we stated earlier, there is a parabolic velocity profile coming in domain 1. Inflow boundary is drawn as a green line (AB) in Figure 3.1. If we insert a cartesian coordinate system in Figure 3.1 so that point A and B have coordinates (0,0) and (0,L) respectively, parabolic velocity profile is formulated as;

$$v = \frac{-4v_{\max}x(x-L)}{L^2}. \tag{3.5}$$

In this equation $v_{\max}$ is the maximum velocity seen in $x = L/2$. The velocity and length scales to have non-dimensional equations are $v_{\max}$ and L respectively. From the definition of the stream function;

$$\begin{aligned} u &= \tfrac{\partial \Psi}{\partial y} \\ v &= -\tfrac{\partial \Psi}{\partial x} \\ \Psi &= -\int v dx \end{aligned} \tag{3.6}$$

Integrating equation 3.5 and setting integration constant such that $\Psi(L) = 0$ will give the stream function formula along AB as;

$$\Psi = \frac{4v_{\max}x^3}{3L^2} - \frac{2v_{\max}x^2}{L} + \frac{2Lv_{\max}}{3} \tag{3.7}$$

note that in non-dimensional form $v_{\max}$ and L have unity values. The reason why we set $\Psi(L) = 0$ along AB is that the solid wall is considered as a constant streamline.

For the vorticity boundary condition, equation 3.8 is used

$$\left(\frac{\partial^2\Psi}{\partial x^2}+\frac{\partial^2\Psi}{\partial y^2}\right)_{i,1}=-\Omega_{i,1}. \tag{3.8}$$

Applying a second-order central difference expression for x-derivative and Taylor series expansion for y-derivative will give a numeric inflow boundary condition;

$$\Omega_{i,1}=\frac{2\left(\Psi_{i,1}-\Psi_{i,2}\right)}{\Delta y^2}-\frac{\Psi_{i+1,1}-2\Psi_{i,1}+\Psi_{i-1,1}}{\Delta x^2}. \tag{3.9}$$

3.1.3.2. Outflow. Outflow boundary is drawn as a green line (CD) in Figure 3.1. At the outflow boundary, the values of the stream function and vorticity are usually extrapolated from the interior solution since $\frac{\partial\Psi}{\partial x}=0$ and $\frac{\partial\Omega}{\partial x}=0$ for the fully developed flow. The boundary conditions obtained by the second order backward difference formulas are;

$$\begin{aligned}\Psi_{im,j}&=\tfrac{1}{3}\left(4\Psi_{im-1,j}-\Psi_{im-2,j}\right)\\ \Omega_{im,j}&=\tfrac{1}{3}\left(4\Omega_{im-1,j}-\Omega_{im-2,j}\right)\end{aligned} \tag{3.10}$$

3.1.3.3. Solid Wall. All of the solid walls are drawn in red in Figure 3.1. The line AE is one of the solid walls, and the boundary conditions for the other solid wall will be similar to those of AE. A solid line can be considered as a streamline and, therefore, the stream function is constant along the wall. We set stream function at point B as $\Psi=0$ while obtaining inflow boundary condition, so the stream function is constant at the value zero along the solid wall BC. For the solid wall AD stream function is set to the value for the inflow at point A. Physical boundary condition for vorticity on solid wall does not exist. Therefore, a set of numerical boundary conditions must be constructed. In the model problem solid walls are non-porous and stationary.

Stream function is always the solution to an elliptic equation;

$$\frac{\partial^2 \Psi}{\partial x^2} + \frac{\partial^2 \Psi}{\partial y^2} = -\Omega. \tag{3.11}$$

Consider this equation for the left wall which has node coordinate as (1,j),

$$\left(\frac{\partial^2 \Psi}{\partial x^2} + \frac{\partial^2 \Psi}{\partial y^2}\right)_{1,j} = -\Omega_{1,j}. \tag{3.12}$$

Along this surface, the stream function is a constant which is consistent with continuity equation. In order to ensure consistency, the stream function value at the left of the inflow boundary is assigned as the left wall boundary stream function is constant. The equation 3.7 for $x = 0$ is

$$\Psi = \frac{2L\upsilon_{\max}}{3}. \tag{3.13}$$

For the right wall $x = L$, the equation 3.7 becomes

$$\Psi = 0. \tag{3.14}$$

As previously mentioned, there is no physical vorticity boundary condition. Therefore, a set of boundary conditions for the vorticity must be constructed. The procedure involves the stream function along with Taylor series expansion of the stream function. For the left solid wall;

$$\left.\frac{\partial^2 \Psi}{\partial y^2}\right|_{1,j} = 0 \tag{3.15}$$

and equation 3.12 is reduced to

$$\left.\frac{\partial^2 \Psi}{\partial x^2}\right|_{1,j} = -\Omega_{1,j}. \tag{3.16}$$

In order to obtain an expression for the second order derivative in the above equation,

consider Taylor series expansion;

$$\Psi_{2,j} = \Psi_{1,j} + \left.\frac{\partial \Psi}{\partial x}\right|_{1,j} \Delta x + \left.\frac{\partial^2 \Psi}{\partial x^2}\right|_{1,j} \frac{(\Delta x)^2}{2} + ... \tag{3.17}$$

Along the left wall

$$\upsilon_{1,j} = - \left.\frac{\partial \Psi}{\partial x}\right|_{1,j} = 0. \tag{3.18}$$

Therefore;

$$\Psi_{2,j} = \Psi_{1,j} + \left.\frac{\partial^2 \Psi}{\partial x^2}\right|_{1,j} \frac{(\Delta x)^2}{2} + \Theta(\Delta x)^3 \tag{3.19}$$

from which

$$\left.\frac{\partial^2 \Psi}{\partial x^2}\right|_{1,j} = \frac{2\left(\Psi_{2,j} - \Psi_{1,j}\right)}{\left(\Delta x\right)^2} + \Theta\left(\Delta x\right) \tag{3.20}$$

inserting equation 3.20 in to equation 3.19 yields vorticity boundary condition for the vorticity;

$$\Omega_{1,j} = \frac{2\left(\Psi_{1,j} - \Psi_{2,j}\right)}{\left(\Delta x\right)^2}. \tag{3.21}$$

A similar procedure is used for the right wall. Higher order implementations of the boundary conditions will generally increase the accuracy of the solution but this may cause instabilities in high Reynolds number flows.

3.1.3.4. Artificial Boundary. For the artificial boundary, Dirichlet boundary condition is applied. Bi-linear interpolation and restriction operations are used in order to transform information from a domain to boundary of other domain. Robin boundary value problem is another way of solving artificial boundaries [31].

3.1.3.5. Re-entrant Corner. When we apply solid wall boundary conditions to a concave sharp corner we probably face no problem but when solid boundary conditions are tried for sharp convex corner, there will be some problems. The computation conditions at the sharp convex corner require special considerations of boundary values and accuracy. Seven different methods of handling the corner vorticity in rectangular coordinates is listed in the book [25]. In this work boundary conditions for the wall corner with 45 degree are applied. The details of calculations are given in that book.

3.1.4. Solution Algorithm

For the solution at level 1, domain 1 is solved with zero initial guess. The solution of domain 1 in the overlapped region is interpolated to domain 2. The solution of domain 2 is interpolated to domain 1 as well. The visits between two domains are stopped when the changes of the two domain solutions are under tolerance. Boundary conditions of the overlapped region come from the last solution of the other domain.

The solution at level 1 is used to obtain boundary conditions and initial guess for level 2. This procedure goes on until the solution at level 3 is reached. After the solutions of all levels for domain 1 and 2 are obtained, the solution of domain 3 is sought in order to obtain a more accurate solution around the concave corner.

3.1.5. Results of Internal Flow Problem

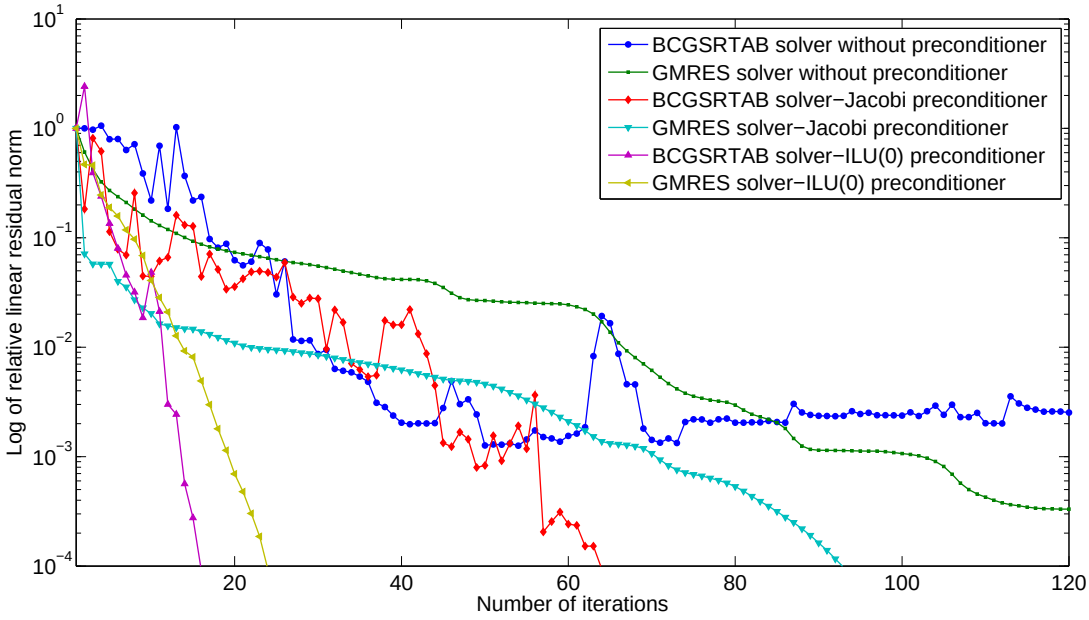
The performance of preconditioner for the solver GMRES and BCGSTAB is shown in Figure 3.3 and Figure 3.4. The first linear step of the second visit of domain 1 for level 1 is selected as sample inner iteration. The selection of first visit as sample visit will not be good because at the first visit artificial boundary conditions have not been updated from domain 2 yet and they include only the initial guess information. If we choose the sample step near the solution, the speed of convergence mainly results from good the initial guess, not from preconditioners. Two different solvers, namely BI-CGSTAB and GMRES, are applied for sample linear step. In order to compare preconditioner, exact Newton algorithm with tolerance of 1e-4 is used.

In Figure 3.3 Jacobi matrix, obtained from discretization, is fully stored. Figure 3.3-a gives a comparison of Jacobi and ILU preconditioners with non-preconditioned case and the next Figure 3.3-b gives fill-in effects for ILU preconditioner.

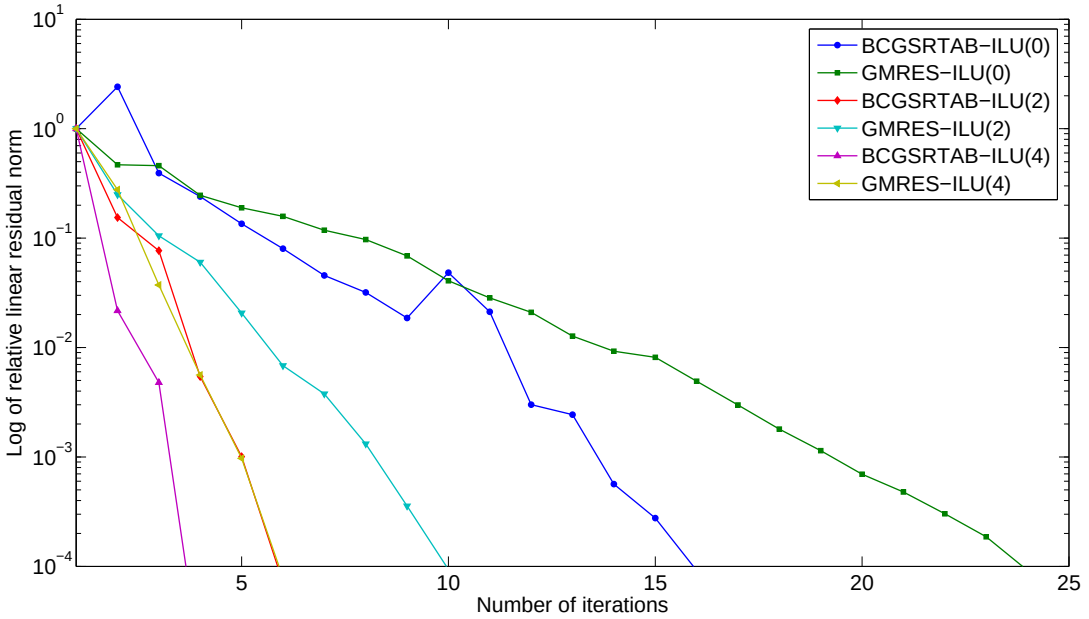
In Figure 3.4, instead of storing full Jacobi matrix, we use compressed column storage (Figure a) and matrix-free algorithm (Figure b). For the compressed column storage SGS and Jacobi preconditioner are applied, whereas for matrix-free scheme non-preconditioned and Jacobi preconditioned cases are given.

Newton's method has rapid convergence near a solution that is not prevented by a bad scaling of the variables. When Reynolds number is increased, the solution of the problem is impeded by bad scaling. Table 3.1 shows a diverged sample run for the L-shaped problem at Re=100. Iterative methods are preferred for linear steps in order to determine steps of Newtons method. Obtaining the exact solution of these linear steps is not doable with these iterative methods but residuals of the solution vectors will be under tolerance. The exact solution of a large linear system is not possible unless we have well conditioned and diagonally dominant matrices.

Inexact Newton method is an appropriate method for the solution of this bad scaling problem. The forcing term controls the local convergence behavior of an inexact Newton method. By choosing the forcing term as small as sufficient one can obtain local convergence speed that is similar to the convergence of Newton's method. But choosing a small forcing term at each linear step, especially when the iterate is far from the solution, causes to over-solve the problem. Backtracking procedure will remove bad effects of oversolving. The main drawback coming with oversolving is that, there is little or no descent in nonlinear residual norm. Even sometimes there is an acceptable decrease in nonlinear residual norm, the cost of obtaining such an accurate solution is relatively much more than a decrease in norm of the residual. A less accurate solution with less expense might yield more reduction in nonlinear residual norm and place less burden on the backtracking. Backtracking method improves the likelihood of convergence even when the initial iterate is not near a solution.

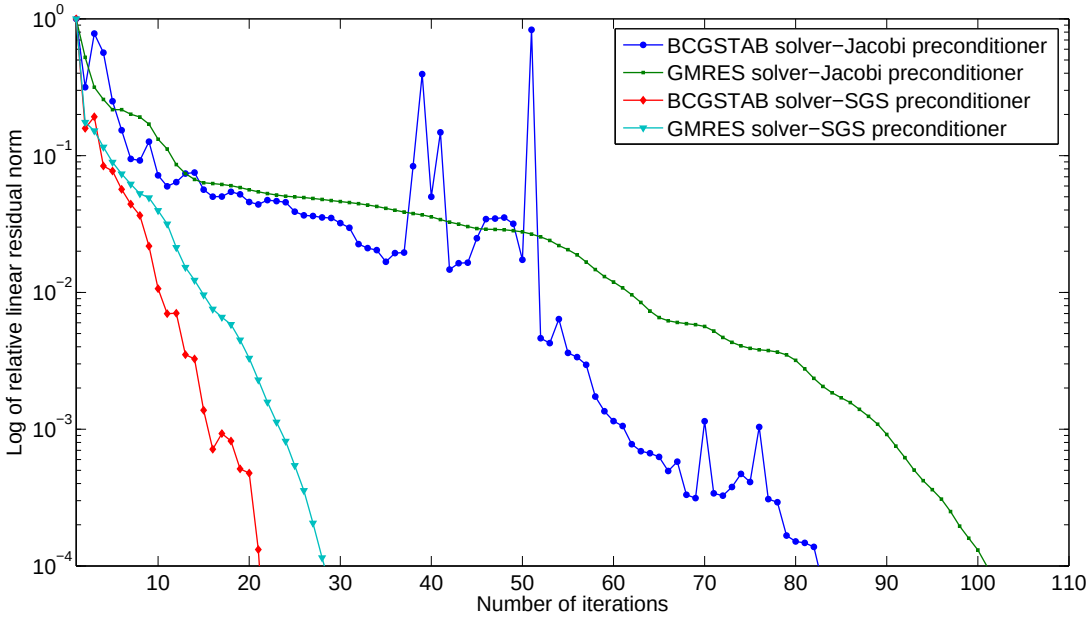


(a) Effects of different preconditioners

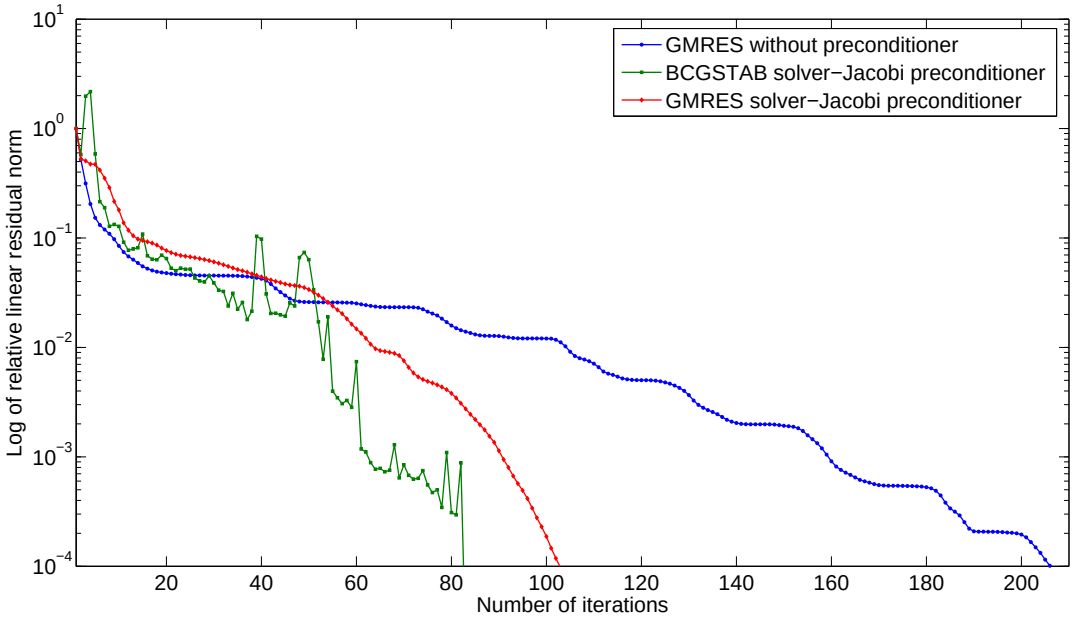


(b) Effects of fill-in for the ILU preconditioner

Figure 3.3. Preconditioning performance of sample inner step with full matrix storage



(a) Compressed column storage scheme

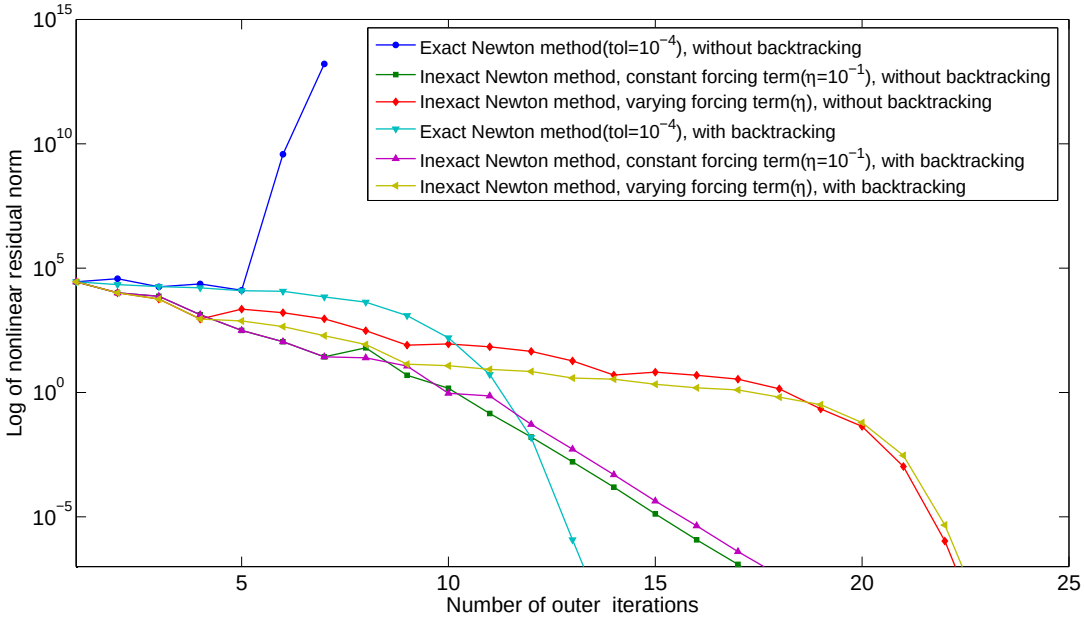


(b) Matrix-free scheme

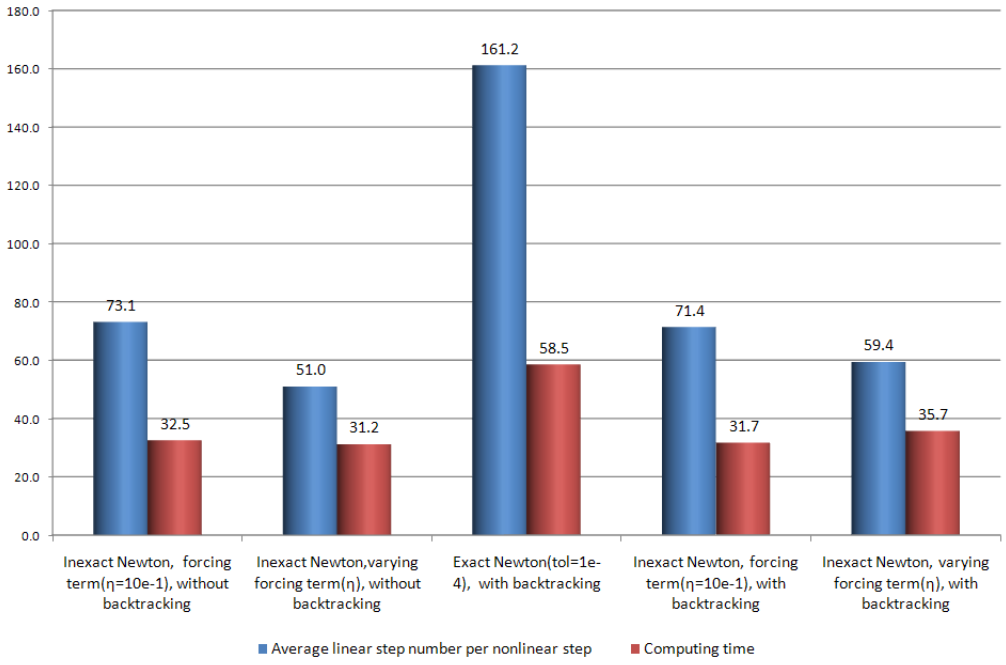
Figure 3.4. Preconditioning performance of a sample inner step without full matrix storage

Backtracking,also known as damping or line search technique, shortens Newton steps (decrease value of λ) as necessary to ensure an adequate decrease in the residual of the nonlinear system. The reduction parameter for linear step (η) varies as the nonlinear algorithm proceeds and the values of this parameter is based on how well the residual of the linear system reflects the behavior of the nonlinear residual.

In Tables 3.1-3.6, the outputs of sample outer iterations for L-shaped problem are given. All of these results are visualized in Figure 3.5. There is no increase in nonlinear residual norm for solution procedures with damping, whereas solution procedure without damping may diverge or have undesired increase in nonlinear residual norm. Exact Newton methods have less outer iteration but in that method the number of linear iterations per nonlinear step is higher so the computing time is bigger than the inexact Newton method. When we take into account the results of both Figure 3.5-a, and -b, we reach the best combination which is the inexact Newton with backtracking.



(a) Log of nonlinear iteration residual



(b) Computing performance

Figure 3.5. Comparison of different solution methods (exact, inexact, with / without damping)

Table 3.1. Exact Newton method ($tol = 10^{-4}$), without backtracking

VISIT	1					
DOMAIN	2					
LEVEL	1					
NONLINEAR INITIAL NORM	2.82E+04					
NONLINEAR TOLERANCE	1.00E-07					
Newton	Linear	nonlinear residual	linear resiual	lambda	eta	time
1	102	3.75E+04	9.87E-05	1.00E+00	1.00E-04	2.66
2	144	1.77E+04	9.81E-05	1.00E+00	1.00E-04	3.89
3	220	2.32E+04	9.69E-05	1.00E+00	1.00E-04	6.42
4	267	1.29E+04	8.91E-05	1.00E+00	1.00E-04	8.41
5	473	3.80E+09	9.86E-05	1.00E+00	1.00E-04	14.78
6	4800	1.63E+13	3.33E-03	1.00E+00	1.00E-04	*****

Table 3.2. Inexact Newton method, constant forcing term ($\eta = 10^{-1}$), without backtracking

VISIT	1					
DOMAIN	2					
LEVEL	1					
NONLINEAR INITIAL NORM	2.82E+04					
NONLINEAR TOLERANCE	1.00E-07					
Newton	Linear	nonlinear residual	linear resiual	lambda	eta	time
1	9	1.02E+04	9.97E-02	1.00E+00	1.00E-01	0.31
2	38	7.37E+03	9.88E-02	1.00E+00	1.00E-01	0.98
3	33	1.34E+03	9.83E-02	1.00E+00	1.00E-01	0.86
4	36	3.15E+02	9.59E-02	1.00E+00	1.00E-01	0.94
5	49	1.10E+02	9.75E-02	1.00E+00	1.00E-01	1.25
6	66	2.70E+01	9.96E-02	1.00E+00	1.00E-01	1.67
7	75	6.26E+01	9.72E-02	1.00E+00	1.00E-01	1.94
8	36	4.89E+00	9.67E-02	1.00E+00	1.00E-01	0.95
9	92	1.47E+00	9.86E-02	1.00E+00	1.00E-01	2.36
10	75	1.43E-01	9.76E-02	1.00E+00	1.00E-01	1.92
11	110	1.62E-02	9.75E-02	1.00E+00	1.00E-01	2.88
12	115	1.63E-03	9.66E-02	1.00E+00	1.00E-01	3.03
13	94	1.55E-04	9.72E-02	1.00E+00	1.00E-01	2.47
14	111	1.30E-05	8.73E-02	1.00E+00	1.00E-01	2.95
15	100	1.19E-06	9.85E-02	1.00E+00	1.00E-01	2.62
16	99	1.22E-07	9.72E-02	1.00E+00	1.00E-01	2.61
17	105	1.24E-08	9.77E-02	1.00E+00	1.00E-01	2.78
17	1243					32.53

Table 3.3. Inexact Newton method,forcing term (η) varies as the nonlinear algorithm proceeds, without backtracking

VISIT	1					
DOMAIN	2					
LEVEL	1					
NONLINEAR INITIAL NORM	2.82E+04					
NONLINEAR TOLERANCE	1.00E-07					
Newton	Linear	nonlinear residual	linear resiual	lambda	eta	time
1	9	1.02E+04	9.97E-02	1.00E+00	1.20E-01	0.31
2	34	5.69E+03	1.16E-01	1.00E+00	2.80E-01	0.89
3	16	8.92E+02	2.78E-01	1.00E+00	2.20E-02	0.45
4	120	2.24E+03	2.16E-02	1.00E+00	9.00E-01	3.17
5	1	1.61E+03	7.22E-01	1.00E+00	7.30E-01	0.11
6	2	9.19E+02	5.89E-01	1.00E+00	4.80E-01	0.14
7	8	3.05E+02	4.48E-01	1.00E+00	2.10E-01	0.27
8	20	7.99E+01	1.93E-01	1.00E+00	6.20E-02	0.56
9	90	8.93E+01	6.06E-02	1.00E+00	9.00E-01	2.33
10	2	6.85E+01	7.82E-01	1.00E+00	7.30E-01	0.14
11	5	4.49E+01	6.82E-01	1.00E+00	4.80E-01	0.2
12	13	1.86E+01	4.57E-01	1.00E+00	2.10E-01	0.41
13	42	5.03E+00	2.05E-01	1.00E+00	6.60E-02	1.09
14	93	6.55E+00	6.56E-02	1.00E+00	9.00E-01	2.41
15	1	4.94E+00	7.55E-01	1.00E+00	7.30E-01	0.12
16	2	3.43E+00	7.15E-01	1.00E+00	4.80E-01	0.12
17	8	1.40E+00	4.51E-01	1.00E+00	2.10E-01	0.28
18	55	2.18E-01	2.05E-01	1.00E+00	2.20E-02	1.42
19	120	4.32E-02	2.09E-02	1.00E+00	3.50E-02	3.17
20	92	1.05E-03	3.48E-02	1.00E+00	5.30E-04	2.41
21	185	1.06E-06	4.99E-04	1.00E+00	1.00E-04	5.27
22	203	9.22E-11	9.94E-05	1.00E+00	1.00E-04	5.88
22	1121					31.16

Table 3.4. Exact Newton method ($tol = 10^{-4}$), with backtracking

VISIT	1					
DOMAIN	2					
LEVEL	1					
NONLINEAR INITIAL NORM	2.82E+04					
NONLINEAR TOLERANCE	1.00E-07					
Newton	Linear	nonlinear residual	linear resiual	lambda	eta	time
1	102	2.17E+04	9.87E-05	5.00E-01	1.00E-04	2.67
2	118	1.81E+04	9.41E-05	1.00E+00	1.00E-04	3.12
3	174	1.61E+04	9.87E-05	1.00E+00	1.00E-04	4.86
4	171	1.23E+04	9.88E-05	5.00E-01	1.00E-04	4.83
5	196	1.17E+04	1.00E-04	9.70E-02	1.00E-04	5.64
6	168	6.94E+03	9.79E-05	5.00E-01	1.00E-04	4.7
7	161	4.30E+03	9.31E-05	1.00E+00	1.00E-04	4.48
8	176	1.23E+03	9.38E-05	1.00E+00	1.00E-04	4.97
9	138	1.57E+02	9.87E-05	1.00E+00	1.00E-04	3.75
10	140	5.26E+00	8.99E-05	1.00E+00	1.00E-04	3.81
11	162	1.54E-02	9.91E-05	1.00E+00	1.00E-04	4.5
12	180	1.19E-06	9.63E-05	1.00E+00	1.00E-04	5.08
13	209	1.15E-10	9.15E-05	1.00E+00	1.00E-04	6.09
13	2095					58.52

Table 3.5. Inexact Newton method, constant forcing term($\eta = 10^{-1}$), with backtracking

VISIT	1					
DOMAIN	2					
LEVEL	1					
NONLINEAR INITIAL NORM	2.82E+04					
NONLINEAR TOLERANCE	1.00E-07					
Newton	Linear	nonlinear residual	linear resiual	lambda	eta	time
1	9	1.02E+04	9.97E-02	1.00E+00	1.00E-01	0.34
2	38	7.37E+03	9.88E-02	1.00E+00	1.00E-01	0.95
3	33	1.34E+03	9.83E-02	1.00E+00	1.00E-01	0.84
4	36	3.15E+02	9.59E-02	1.00E+00	1.00E-01	0.94
5	49	1.10E+02	9.75E-02	1.00E+00	1.00E-01	1.27
6	66	2.70E+01	9.96E-02	1.00E+00	1.00E-01	1.69
7	75	2.46E+01	9.72E-02	5.00E-01	1.00E-01	1.91
8	71	1.16E+01	9.85E-02	1.00E+00	1.00E-01	1.83
9	61	9.27E-01	9.85E-02	1.00E+00	1.00E-01	1.56
10	101	7.28E-01	9.30E-02	1.00E+00	1.00E-01	2.62
11	67	5.19E-02	9.69E-02	1.00E+00	1.00E-01	1.73
12	90	5.24E-03	9.36E-02	1.00E+00	1.00E-01	2.34
13	112	4.92E-04	8.96E-02	1.00E+00	1.00E-01	2.94
14	106	4.30E-05	9.93E-02	1.00E+00	1.00E-01	2.8
15	107	4.34E-06	9.82E-02	1.00E+00	1.00E-01	2.84
16	89	3.97E-07	9.95E-02	1.00E+00	1.00E-01	2.33
17	103	4.41E-08	9.59E-02	1.00E+00	1.00E-01	2.72
17	1213					31.66

Table 3.6. Inexact Newton method,forcing term (η) varies as the nonlinear algorithm proceeds, with backtracking

VISIT	1					
DOMAIN	2					
LEVEL	1					
NONLINEAR INITIAL NORM	2.82E+04					
NONLINEAR TOLERANCE	1.00E-07					
Newton	Linear	nonlinear residual	linear resiual	lambda	eta	time
1	9	1.02E+04	9.97E-02	1.00E+00	1.20E-01	0.31
2	34	5.69E+03	1.16E-01	1.00E+00	2.80E-01	0.89
3	16	8.92E+02	2.78E-01	1.00E+00	2.20E-02	0.45
4	120	7.53E+02	2.16E-02	5.00E-01	6.40E-01	3.16
5	8	4.47E+02	6.26E-01	1.00E+00	3.70E-01	0.28
6	16	1.91E+02	3.54E-01	1.00E+00	1.60E-01	0.47
7	34	8.51E+01	1.57E-01	1.00E+00	1.80E-01	0.89
8	29	1.38E+01	1.79E-01	1.00E+00	2.40E-02	0.78
9	133	1.19E+01	2.23E-02	2.20E-01	6.70E-01	3.56
10	39	8.45E+00	6.64E-01	1.00E+00	4.50E-01	1.02
11	46	6.98E+00	4.44E-01	5.00E-01	6.10E-01	1.19
12	34	3.79E+00	6.08E-01	1.00E+00	3.40E-01	0.89
13	44	3.44E+00	3.26E-01	5.00E-01	7.40E-01	1.14
14	6	2.14E+00	7.19E-01	1.00E+00	4.90E-01	0.23
15	43	1.54E+00	4.79E-01	5.00E-01	4.60E-01	1.11
16	48	1.27E+00	4.53E-01	1.00E+00	6.10E-01	1.27
17	8	6.47E-01	6.07E-01	1.00E+00	3.40E-01	0.28
18	67	3.23E-01	3.33E-01	1.00E+00	2.20E-01	1.73
19	72	6.13E-02	2.23E-01	1.00E+00	3.20E-02	1.89
20	115	2.91E-03	3.08E-02	1.00E+00	2.00E-03	3.06
21	168	4.68E-06	1.91E-03	1.00E+00	1.00E-04	4.7
22	217	5.00E-10	9.90E-05	1.00E+00	1.00E-04	6.38
22	1306					35.69

In order to control grid sensitivity of L-shaped problem, the grid sizes of the computational domain are increased gradually as $h = 1/8$, $h = 1/16$, $h = 1/32$ and, $h = 1/64$. Four sample data are selected to decide at which grid size, the solution of the problem is not sensitive to that grid size. The locations of these data are selected in such a way that we comment easily on grid sensitivity and the advantages of the mesh refinement strategy. The coordinates of these sample points do not change with the change in the grid size (h), in other words they are constant. In Figures 3.6-3.9 solutions of the L-shaped problem with different mesh sizes at the Reynolds number 100 are given and the values of these sample points are given on these figures. These values are tabulated on Table 3.7 with number of grid points for the domain 1 and the computation time for the overall solution. The solution of this problem is not sensitive to mesh size up to three significant digits for the $h = 1/32$.

In Figure 3.10, the solution of the problem without mesh refinement is given. In this figure grid size is given as $h = 1/64$ for all domains, which means it has uniform grids. We compare this figure with the Figure 3.7 ,whose mesh size gradually increases from $h = 1/16$ to $h = 1/64$ next to the re-entrant corner by local mesh refinement. For $h = 1/64$, i.e. on the uniform grid with about 8500 grid points, the stream function value near to the re-entrant corner (sample point 4) is about 0.025. On an appropriate locally refined grid, however, it is possible to obtain a similar accuracy with less than 600 grid points and therefore with significantly less computational work. This comparison gives an impression of the potential of local refinement. When the values of the sample points 2 and 3 for the Figures 3.7 and 3.10 are compared, their similarity does not appear as good as sample point 4. In order to have better match, we may select boundary of the inner domains by taking into account the grid sensitivity. In other words, adaptive mesh refinement may become more useful than local refinement.

Up to now, the effects of the computational parameters on convergence for L-shaped flow problem have been studied. Reynolds number is a physical parameter that affects the convergence of the solution. In order to search this physical parameter's effects, three different Reynolds numbers are used. The solutions with these Reynolds numbers are given in Figures 3.11-3.13.

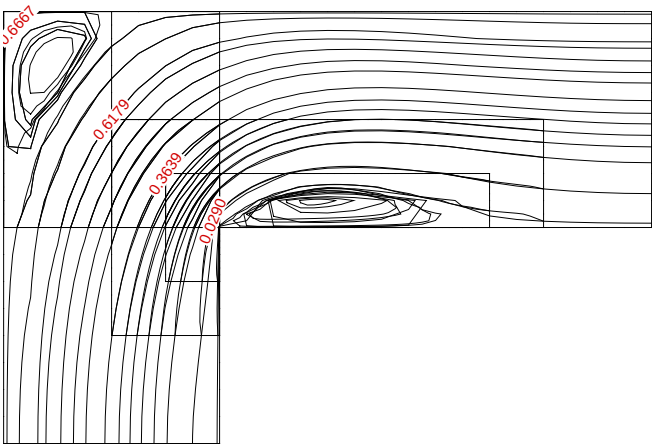


Figure 3.6. L-shaped problem at Re=100, h=1/8

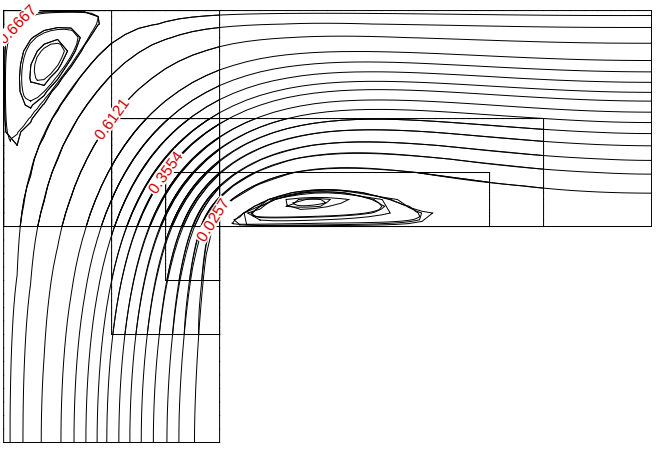


Figure 3.7. L-shaped problem at Re=100, h=1/16

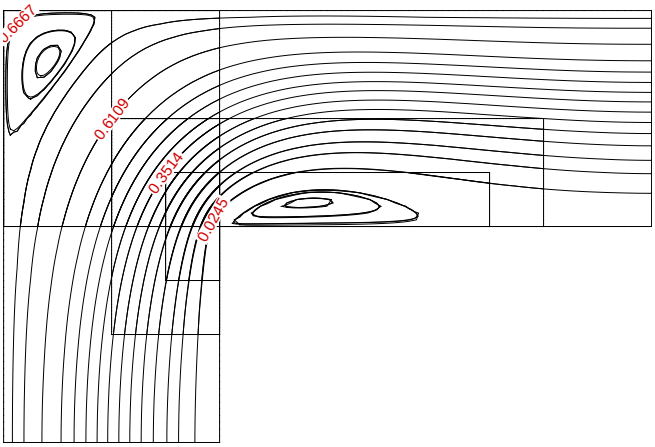


Figure 3.8. L-shaped problem at Re=100, h=1/32

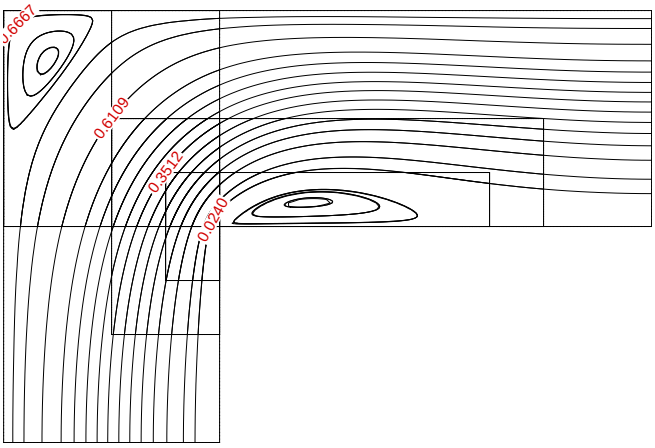


Figure 3.9. L-shaped problem at Re=100, h=1/64

Table 3.7. Grid sensitivity test for L-shaped problem at Re=100

Grid sizes	h=1/8	h=1/16	h=1/32	h=1/64
Sample point 1	0.6667	0.6667	0.6667	0.6667
Sample point 2	0.6179	0.6121	0.6109	0.6109
Sample point 3	0.3639	0.3554	0.3514	0.3512
Sample point 4	0.0290	0.0257	0.0245	0.0240
Computation time(sec)	4.0	71.6	991.4	16564.8
Number of grid points for domain 1	153	561	2145	8385

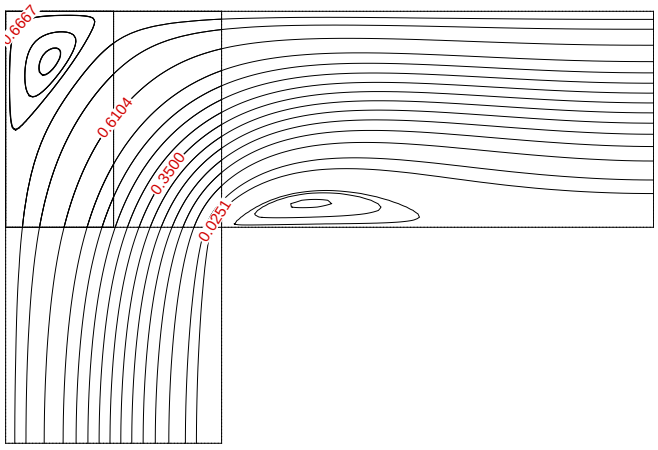
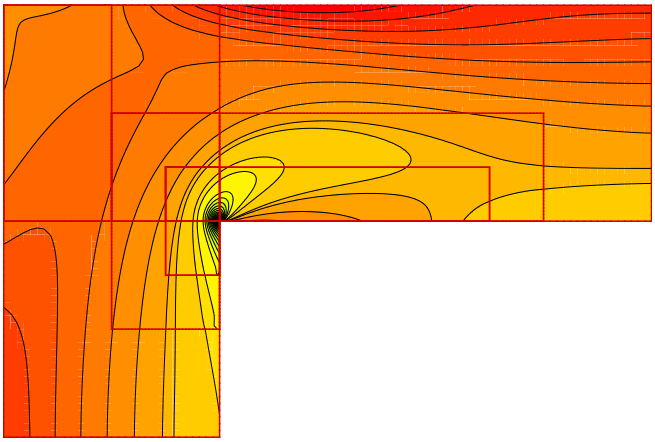
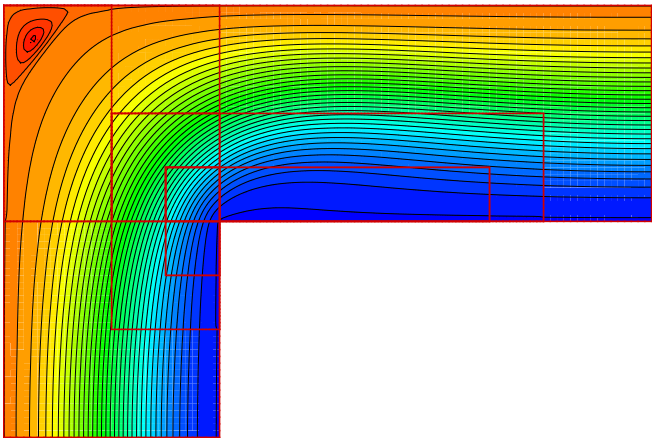


Figure 3.10. L-shaped problem at Re=100, h=1/64 without mesh refinement

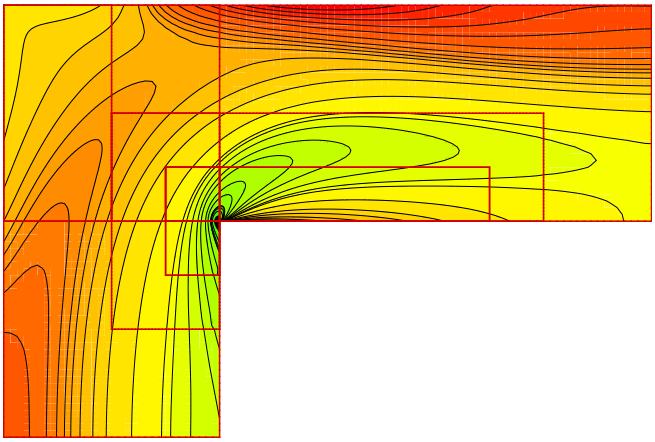


(a) Vorticity

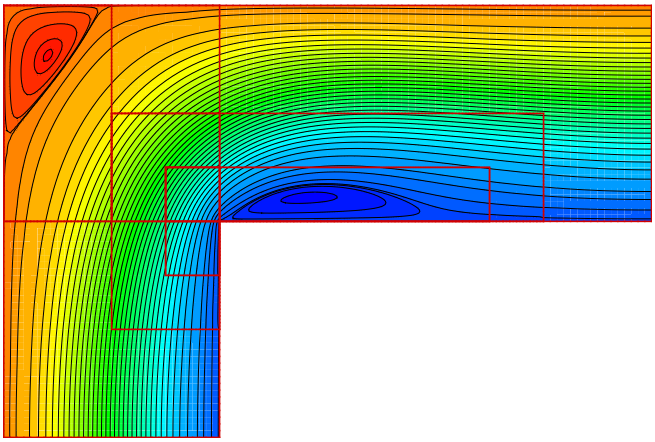


(b) Stream function

Figure 3.11. Solution of L-shaped problem at Re=50

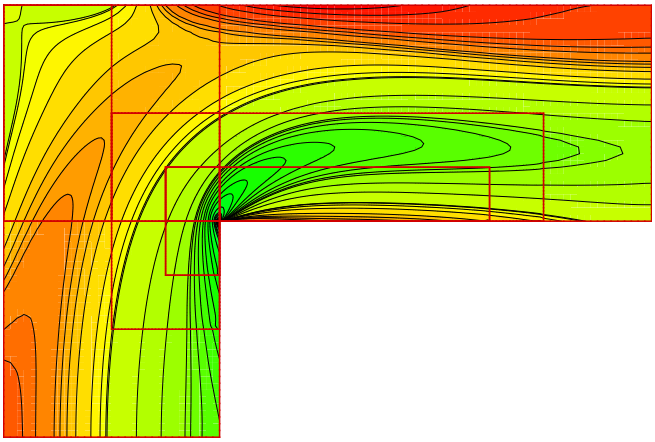


(a) Vorticity

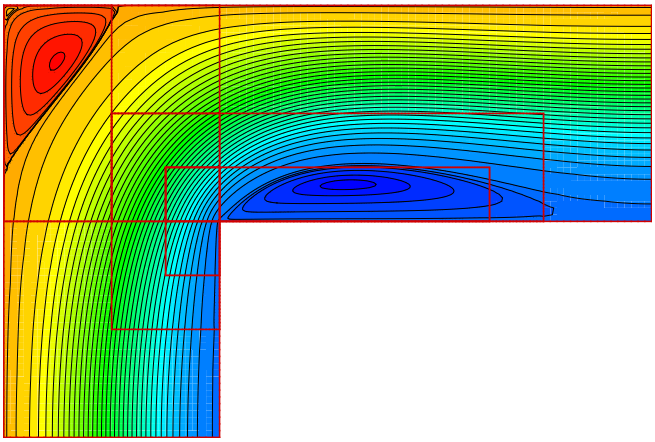


(b) Stream function

Figure 3.12. Solution of L-shaped problem at Re=100



(a) Vorticity



(b) Stream function

Figure 3.13. Solution of L-shaped problem at Re=150

Flows with different Reynolds numbers have different characteristics. The algebraic system that defines the flow becomes more nonsymmetric as Reynolds number increases. This makes the solution of the problem more difficult besides increase computational work. The number of linear and nonlinear iterations and computational times at Re=50, Re=100, Re=150 are showed in Table 3.8. Computational time and number of average linear step per nonlinear step increases with the Reynolds number.

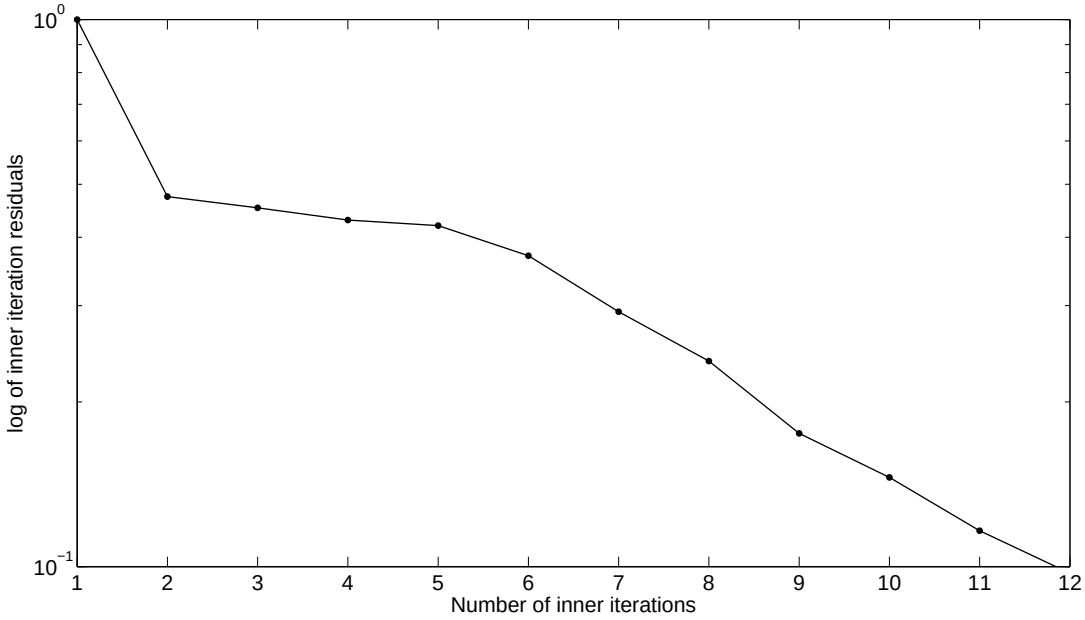
Table 3.8. Computational load for the L-shaped problem at different Re

Reynolds numbers	Re=50	Re=100	Re=150
Number of total nonlinear step	490	604	708
Number of total linear step	33531	58573	84329
Number of average linear step per nonlinear step	68.4	97.0	119.1
Computation time(sec)	407.3	991.6	1253.6

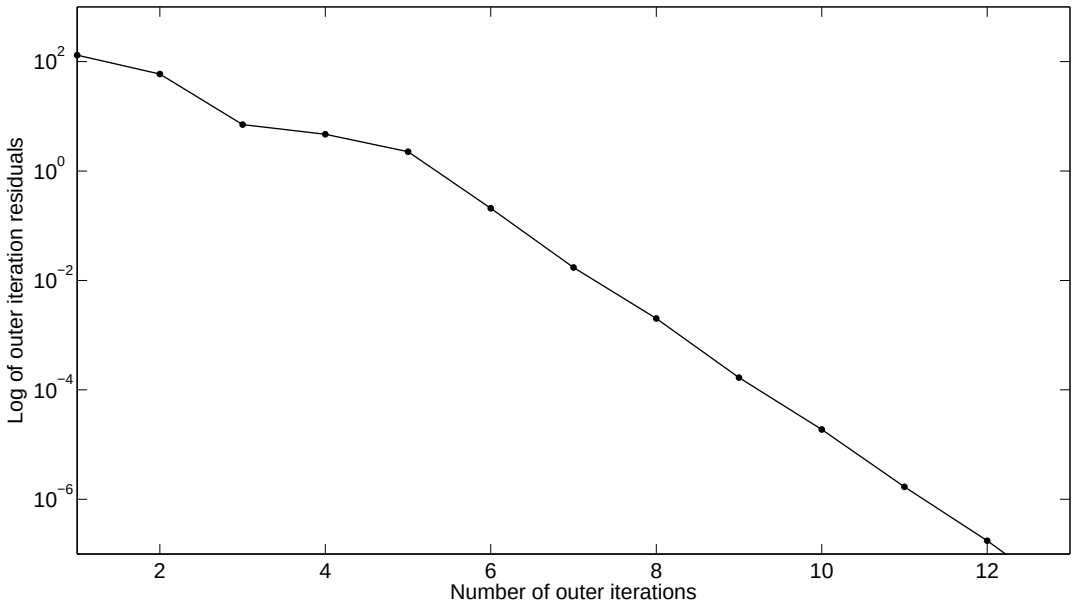
In the solution procedure of L-shaped problem the convergence criteria of outer iteration is constant at 1e-7 whereas the convergence criteria of the inner iteration is not constant because of altering the forcing term at each step. In Figure 3.14, the norm of the sample outer and inner iteration residuals are given at Re=100 with respect to the number of iterations.

The solutions of the domain 1 and domain 2 update each other in order to get the artificial boundary conditions. The choice of the number of updates is the convergence criteria for domain decomposition. In this work, the norms of the difference vectors for the following iterates are controlled in order to have converged domain decomposition algorithm. The limits of convergence is 1e-6 for all three levels. In Figure 3.15, reduction of this difference norm for domain 1 and domain 2 for all levels is given with respect to the number of visits between domain 1 and domain 2 at Re=100.

In Figure 3.14-b, norm of the nonlinear residual vector for a sample step is drawn with respect to the number of iterations. In order to look at the whole picture of convergence, we draw the nonlinear residual vector norms for both domain 1 and domain 2 in Figure 3.16. On the same figure log of residual norms for all levels is shown.

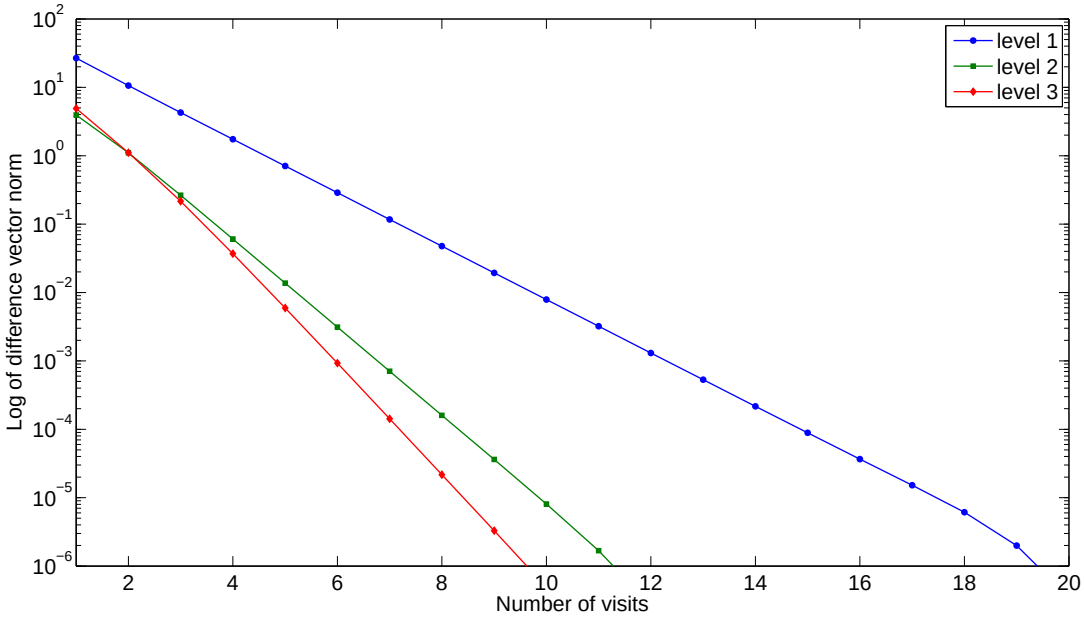


(a) Inner iterations

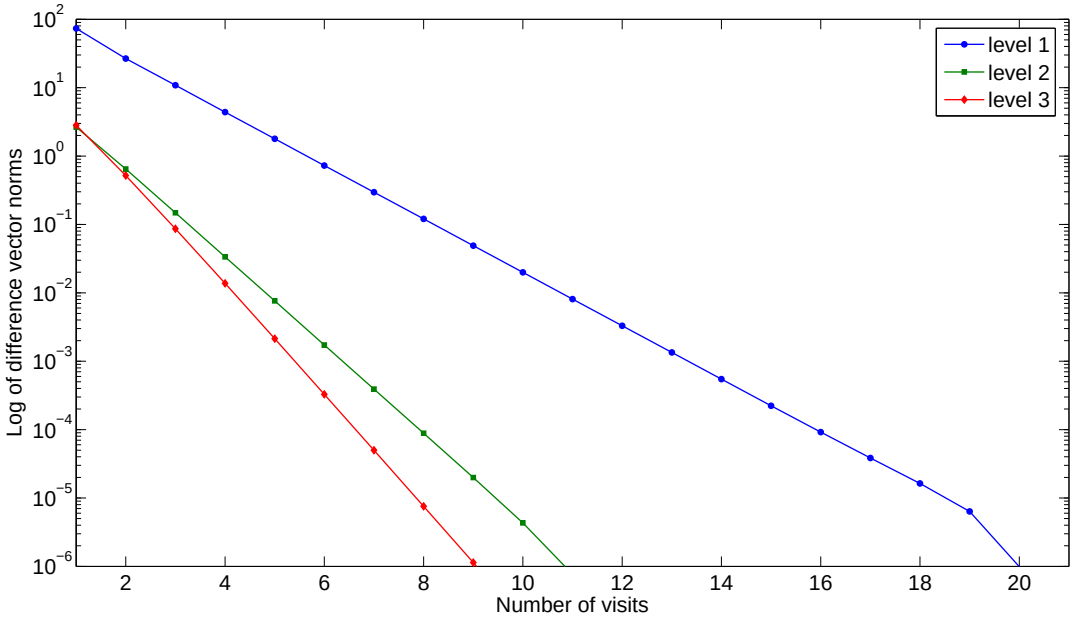


(b) Outer iterations

Figure 3.14. Residual norms of the inner and outer iterations



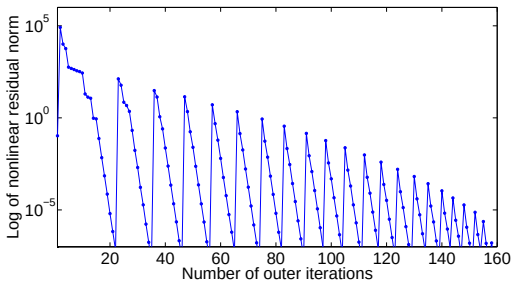
(a) Domain 1



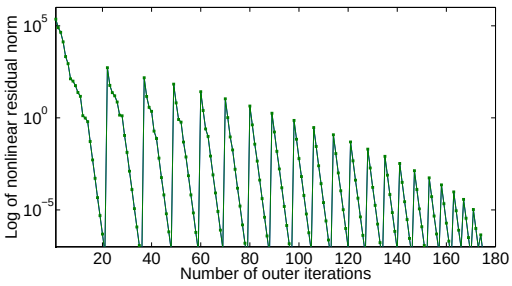
(b) Domain 2

Figure 3.15. Reduction of difference vector norm with respect to number of visits at Re=100

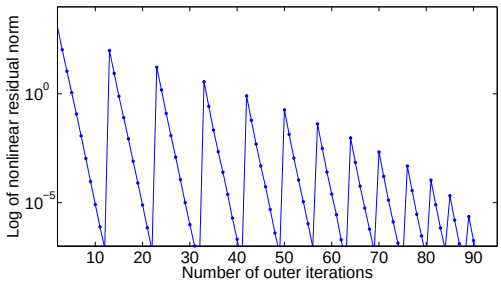
There are peaks in these figures since some parts of the boundary (artificial boundary) changes at each visit, which leads to a different problem. As the visits between domain increase, these problems resemble each other so the nonlinear initial norm decreases.



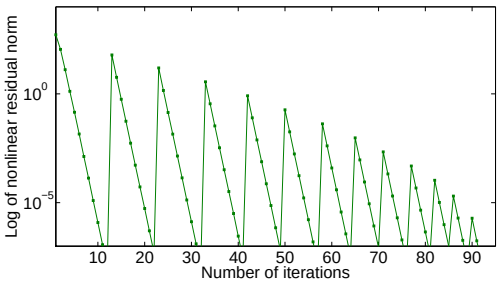
(a) Domain 1 level 1



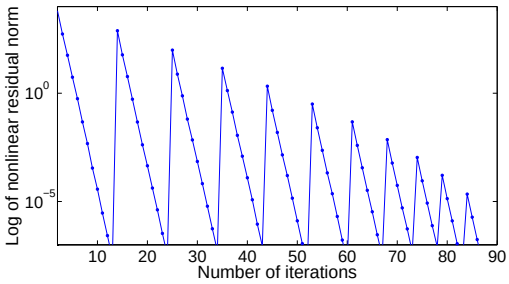
(b) Domain 2 level 1



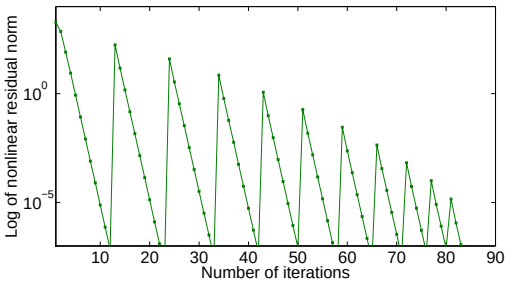
(c) Domain 1 level 2



(d) Domain 2 level 2



(e) Domain 1 level 3



(f) Domain 2 level 3

Figure 3.16. Nonlinear residual norm for L-shaped problem at Re=100

3.2. The External Flow Problem

3.2.1. Problem Statement

In this section domain decomposition technique is used for simulating 2D external, incompressible viscous flows. In this thesis work, two types of external flow problems are sought. In the first case one body immersed in the fluid resists to the flow (Figure 3.17a) and for the second case there are two bodies (Figure 3.17b). These figures also show the geometric decomposition of the sample problems. The fluid domains are decomposed into a set of subdomains. Small subdomains are composed of the immediate vicinity of the bodies. These objects have rectangular shapes so body-fitted cartesian grid system is used for the both of problems since for a simple body shape, the use of a single body conforming curvilinear mesh leads to the most efficient solution procedure.

For the external flow problem in which there is an immersed body, even the body or the fluid has velocity, we can fix the coordinate system in the objects and solve the problem of fluid flowing over a stationary body with upstream velocity. In the case of two immersed bodies case in which these bodies have no relative velocity with respect to each other, the upstream velocity usage will still be valid. But if there is a relative velocity between these objects, beside the upstream velocity, one of the objects should move with relative velocity. The major difficulty that comes with the movement of object is that remeshing of the computational domain is required for the simulation. The body motion and related unsteady flow field may be simulated as a number of sequential steady-state flow computations. However, freezing the body motion during each steady state computation may lead to erroneous predictions.

In these problems, the upstream velocity is in the x direction and the inflow has a linear profile. The domain which has upstream, downstream, and infinity boundary conditions is named as domain 1. Domain 2 is placed around the solid object. In order to transfer information from domain 1 to domain 2, artificial Dirichlet type boundary conditions are used inside domain 1 and outside domain 2. The two solutions from

domain 1 and domain 2 are coupled by means of a Schwarz-type strategy. For the two body in motion flow problems, we need one more subdomain (domain 3) for the new solid object. If the angle-of-attack to this new solid object is increased from zero to a critical value, some portion of the inner artificial boundary of the domain 1 may be non-overlapped with the other subdomains, hence this portion can not take information from the solid objects. In order to prevent this undesired situation, domain 4 is also used for the case of two obstacles

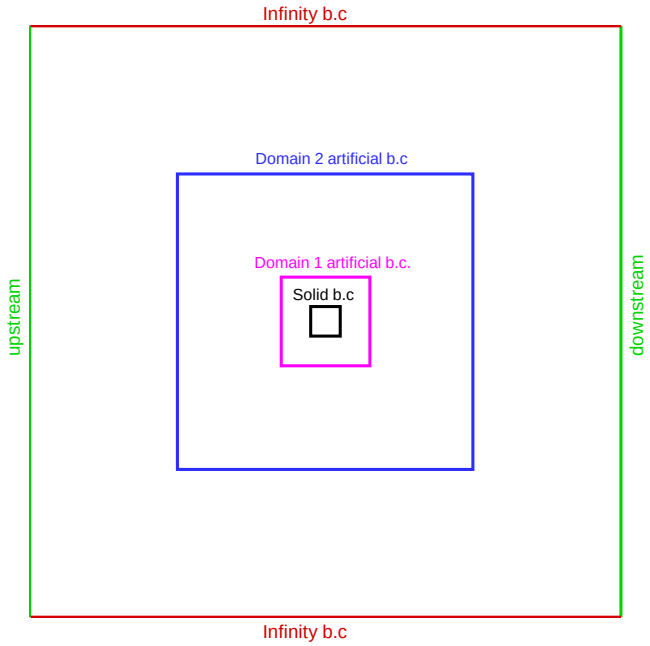
While simulating the external flow problem, one of the major difficulty is the application of the infinity boundary condition. This condition is often replaced by an artificial boundary condition which is applied at the outer limit of the computational domain. In order to guarantee accuracy, infinity boundary condition should be asymptotically far enough from the physical boundary. Because of this, the place of the outer boundary should be determined for a consistent solution.

In the external flow problems, vorticity is generated in a thin boundary layer located next to the solid boundary and concentrates further downstream in a wake, which means regions apart from the solid objects' surroundings and downstream are almost curl-free. Therefore, an adaptive grid strategy should be used in order to concentrate grid points in the boundary layer and in the wake in order to have better simulation. The adaptive grid system used for both of the external flow problem is shown in the Figure 3.18.

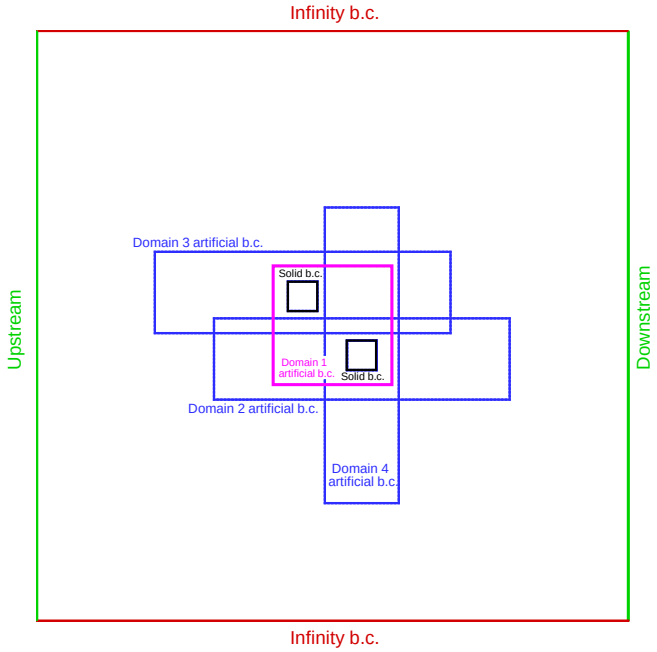
The convergence of the problems also depends on the length of the domain since the outflow boundary condition forces the flow to uniform outflow type and flow needs enough length to develop.

3.2.2. Discretization of the Governing Equations

In the external flow problems, both of the Lagrangian and Eulerian formulation of the Navier-Stokes equations should be considered. For the moderate and high Reynolds numbers, vorticity concentrates in the boundary layer and wakes as stated earlier. The

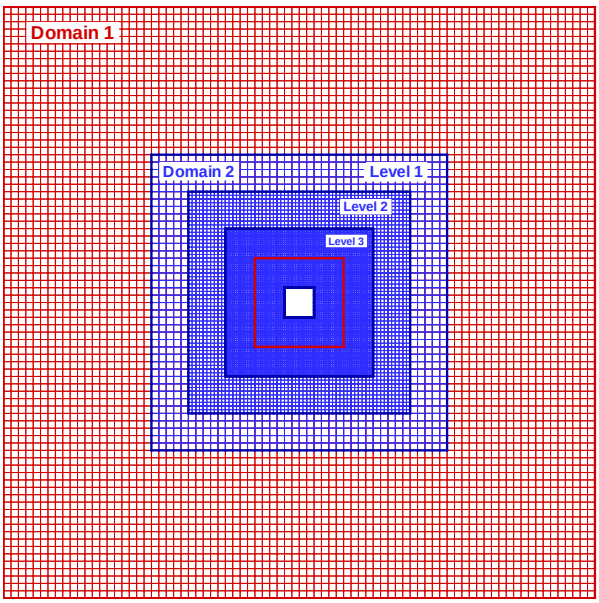


(a) One body in motion

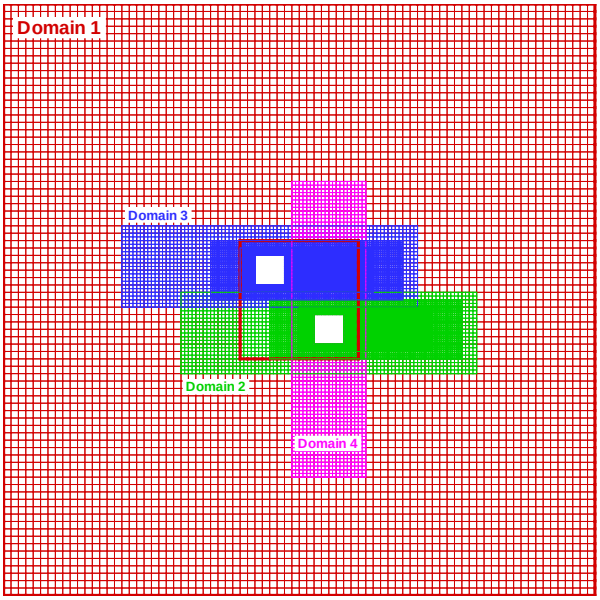


(b) Two bodies in motion

Figure 3.17. Geometry and boundary conditions of external flow problems



(a) One body in motion



(b) Two bodies in motion

Figure 3.18. Adaptively meshed domains for the external flow

discretization points in the critical region of the flow can be concentrated by using Lagrangian formulation of Navier-Stokes equations and this formulation may become better choice for the flow in which convection effects dominate the viscous effects. But boundary conditions can not easily be prescribed for this formulation and it may be inaccurate in the flow regions where viscous effects dominate convective effects. On the other hand, various forms of the boundary conditions can be easily handled with the methods based on Eulerian description of the governing equations. These methods are highly efficient for the bounded domains especially in the region where elliptic nature of the flow is dominant (viscous effects dominate convective effects) [13]. By taking all of these arguments into account, we decided to use finite difference method that is based on the Eulerian description of the Navier-Stokes equations.

Stream function vorticity formulation is used as governing equation and central difference formula is used as finite difference discretization method like in the case of the internal flow problem in the section 3.1.

3.2.3. The Boundary Conditions

At the boundaries of the physical domain, the values of certain physical quantities should be prescribed. Physical considerations usually provide some clues on boundary conditions. Some of them are relatively simple to implement. For example at the solid surface the condition on velocity since the natural boundaries of the physical domain are the walls which are exposed to the flow. However, the specification of boundary conditions for the velocity components at the far-field is not usually straightforward. The numerical treatment of boundary conditions requires particular care because the treatment of the boundary conditions effects accuracy, stability and convergence speed of the solution method.

There are five different kinds of the boundary conditions applied in these external flow problems. Figure 3.17 shows these boundary conditions. For the symmetric case of the external flow problem with one body in motion (in this case, the obstacle is located at the center of computational domain and have zero or 45^0 angel-of-attacks), the flow

is symmetric with respect to the line in the center of the domain with the direction of flow. As there is no flux across the boundary, symmetry boundary condition can be used on this line. The symmetry boundary will also decrease the computational load but it will be valid only for a few specific cases. Since there are also non-symmetric problems, symmetry boundary condition is not used in this study.

Interior solution of the computational domain has influence on the inflow, outflow, and far field boundary conditions, so updating the boundary conditions is required. These factors are due to the physical phenomena of the signal propagation i.e. for an incompressible flow, a disturbance is propagated in all directions.

Boundary conditions for the stream function and vorticity transport equation form can be obtained from the boundary conditions of the primitive variables by using the definition of stream function and vorticity with Taylor series expansion.

3.2.3.1. Upstream. There is a constant velocity profile in the x direction coming into the domain 1 and using the definition of the stream function;

$$\begin{aligned} u &= u_{\max} \\ \Psi &= \int u dy = \int u_{\max} dy \end{aligned} \tag{3.22}$$

$$\Psi = u_{\max} y + d \tag{3.23}$$

and setting integration constant $\Psi(L/2) = 0$, the stream function equation for the inflow is obtained as;

$$\Psi = u_{\max}(y - \frac{L}{2}). \tag{3.24}$$

The procedure for obtaining the vorticity boundary condition is similar to the internal flow problem.

3.2.3.2. Downstream. The velocity at the outflow is usually not known a priori and must be determined as the overall solution evolves. For a fully developed flow, there is no gradient in the x direction for the outflow boundary. The outflow boundary conditions of the internal flow problem and external flow problems are same. The reader is advised to refer to the outflow boundary condition section of the internal flow problem for the details of the numerical application.

3.2.3.3. Far-field. For the external flow problems like flows past airfoils, wings, cars, and other configurations, the numerical simulation needs a bounded domain. For this reason, an artificial far field boundary condition becomes necessary.

The physics and domain of the problem are the most important factors that determine the manner by which any boundary condition is specified. If the far-field boundary is located away from the objects (all of the flow activity take place very near to the solid boundary) one may impose free stream conditions along the boundary. However, if the far-field boundary is relatively close to the action, then the far-field boundary may be treated as inflow or outflow boundary depending on the sign of the normal components of the velocity.

In this study, far-field boundary represents a streamline. A constant value for the stream function along this line can be specified. The assignment of these constants along various boundaries must be consistent with respect to continuity equation. Because the differences between the values of stream function represent the flow rate. In order to ensure consistency, the stream function value at the top of the inflow boundary is assigned as the top outflow boundary stream function constant and the stream function value at the bottom of the inflow boundary is assigned as the bottom outflow boundary stream function constant.

The vorticity is assigned as zero at the far-field boundary since for the external flow problem, vorticity is generated in a thin boundary layer located near the solid boundary and concentrates further downstream in a wake.

3.2.3.4. Artificial Boundary. For the geometrically complex domain, it is usually not possible to generate a single grid system. Because of this, the physical domain is split in to subdomains. The boundary conditions of subdomains depend on the solution of the other domains,which means the information exchange between domains is performed by the boundary conditions. The transmission conditions for the artificial boundaries may be Dirchlet, Neumann, or Robin type. A Dirichlet type boundary condition is used for this problem, which means stream function and vorticity values on the artificial boundaries are evaluated from the solution of the other domains.

3.2.3.5. Solid Wall. The relative velocity between the wall surface and the fluid is assumed to be zero for a viscous fluid which passes a non-porous solid wall. This condition is called as no-slip boundary condition.

Using the stream function and vorticity variables is an effective way to simulate incompressible flows due to the fact that the continuity equation is automatically satisfied, the pressure variable is eliminated and high order scheme can be implemented. However, a difficulty arises in a multi-connected computational domain in determining the constants for the stream function on the boundary of the solid objects. The details of how these constants are evaluated can be found in the study of Liu [22].

Except the determination of these constants, implementation of the wall boundary condition is similar to the internal flow case so the reader is advised to refer to the internal flow wall boundary condition section.

3.2.4. Solution Algorithm

The solution procedure of the problem with one obstacle and two obstacles is different than each other.

For the one body in motion case, domain 1 is solved with zero initial guess so the inner artificial boundaries of the domain 1 take zero values for the first visit. In

the overlapped region solution of the domain 1 is assigned to domain 2. The outer artificial boundaries of the domain 2 take values from this interpolation. Before the second visit of domain 1, solution of domain 2 transfers the solid objects information to the inner artificial boundary and overlapped region of the domain 1 by the help of the bi-linear interpolation. This procedure continues until the changes of the two domain solutions are under tolerance. Solution of domain 2 and level 1 is interpolated to the level 2,which means the outer boundary condition and initial guess is taken from the level 1. For the level 3 solution of the level 2 is used.

For the two bodies in motion case, there is one extra domain for the new solid objects. If the angle-of-attack of this new domain is greater than a critical value, some portions of the inner artificial boundary of domain 1 can not take a value from the domain 2 and domain 3. In order to prevent this situation, domain 4 is also placed around the first object. The solution procedure starts with zero initial guess. The values from this solution is assigned to the domain 2, domain 3, and domain 4 with the help of the bi-linear interpolation. The inner visits between domain 2, domain 3, and domain 4 start with the solution of the domain 2. In order to improve convergence speed at each outer visit (visit between domain 1, domain 2, domain 3, and domain4), the inner visit iterations continue until the difference between subsequent solutions of these domains are under tolerance. After the convergence of the inner visit, the next outer visit is performed. The convergence criteria for the outer visits are the same with inner visits. In order to observe flow characteristics more clearly, adaptive mesh strategy is also used for this multi-block external flow problem. The outer boundaries and initial guesses of the inner levels are taken from the solutions of the outer levels.

3.2.5. Results of the External Flow Problems

In external flow problems, matrix free algorithm is used for the storage scheme. Because the solution of external flow problems may need more grid points than internal flow problems. Diagonal scaling (Jacobi preconditioning) is used to improve the convergence speed. First diagonal element is obtained by a matrix-free multiplication with a vector in which only nonzero element is the first element and its value is unity.

The second element is obtained by a multiplication with a vector in which the only nonzero element is the second element and its value is again unity. For the n unknown system, this procedure continues until nth diagonal element is obtained. This means n function evaluations and the most time consuming part of every nonlinear step is these function evaluations. In order to compute the diagonal of the Jacobian speedily, an α term is multiplied with the noncentral terms of the discrete form of the governing equations;

$$\left[\frac{(\alpha\Psi_{i+1,j} - 2\Psi_{i,j} + \alpha\Psi_{i-1,j})}{\Delta x^2} + \frac{(\alpha\Psi_{i,j+1} - 2\Psi_{i,j} + \alpha\Psi_{i,j-1})}{\Delta y^2}\right] + \alpha\Omega_{i,j} = 0, \qquad (3.25)$$

$$\left[\frac{(\alpha\Omega_{i+1,j} - 2\Omega_{i,j} + \alpha\Omega_{i-1,j})}{\Delta x^2} + \frac{(\alpha\Omega_{i,j+1} - 2\Omega_{i,j} + \alpha\Omega_{i,j-1})}{\Delta y^2}\right] - Re\left[\frac{\alpha\Psi_{i,j+1} - \alpha\Psi_{i,j-1}}{\Delta y}\frac{\alpha\Omega_{i+1,j} - \alpha\Omega_{i-1,j}}{\Delta x} - \frac{\alpha\Psi_{i+1,j} - \alpha\Psi_{i-1,j}}{\Delta x}\frac{\alpha\Omega_{i,j+1} - \alpha\Omega_{i,j-1}}{\Delta y}\right] = 0. \qquad (3.26)$$

If we set $\alpha = 0$ at the beginning of the each nonlinear step and perform a matrix-free multiplication with a unity vector, we can easily extract the diagonal of the Jacobian with just one function evaluation. After the diagonal terms in the Jacobian are extracted, the α parameter should be set to unity in order to return the original discrete form of the governing equations. The speeds of the two algorithms are compared for the one obstacle problem. The sample results for the first nonlinear step are tabulated in Tables 3.9 and 3.10

In external flow problems, infinity boundary should be placed far enough from the solid body in order to have a consistent solution. At infinity, existence of solid body has negligible effects on the flow properties. For the one immersed body with 45^0 angle-of-attack, infinity boundary is placed as follows; the width of the computational domain is taken as 18 times greater than the dimensions of the solid body and stream

Table 3.9. Sample result without α term

VISIT	1					
DOMAIN	1					
LEVEL	1					
NONLINEAR INITIAL NORM	2.39E-01					
NONLINEAR TOLERANCE	1.00E-07					
Newton	Linear	nonlinear residual	linear resiual	lambda	eta	time
1	96	3.70E-02	9.91E-05	1.00E+00	1.00E-04	3.94
2	135	3.11E-06	8.36E-05	1.00E+00	1.00E-07	7.86
3	235	4.03E-12	9.95E-08	1.00E+00	1.00E-07	12.38
3	466					24.17

function values at specific locations (corners of the adaptive subdomains) are noted. At each run the width of the computational domain is increased by the amount of object size and the change of stream function values at the corners of the subdomains are noted. The differences between the values of the stream functions at sample points become smaller at each run. When these differences are very close to zero, the width of the computational domain is enough for the modeling of the infinity. After this point, solution is not sensitive to the location of the infinity boundary.

Table 3.10. Sample result with α term

VISIT	1					
DOMAIN	1					
LEVEL	1					
NONLINEAR INITIAL NORM	2.39E-01					
NONLINEAR TOLERANCE	1.00E-07					
Newton	Linear	nonlinear residual	linear resiual	lambda	eta	time
1	96	3.70E-02	9.91E-05	1.00E+00	1.00E-04	1.11
2	135	3.37E-06	9.05E-05	1.00E+00	1.00E-07	2.59
3	228	5.24E-12	9.68E-08	1.00E+00	1.00E-07	7.48
3	459					11.19

Figure 3.19 shows result of sample run for different h/L ratios at the same Reynolds number. In this ratio h is the length of the square body and L is the width of the computational domain. In these runs, the number of grid points is increased with the width of the computational domain in order to have same grid sizes for all runs. The stream function values at sample points are tabulated in Table 3.11 and sample points' locations are shown in Figure 3.19. According to this table, the solution is not sensitive to the place of infinity boundary for aspect ratio 1/20, which means for this problem, twenty times the body length can be considered as the infinity length.

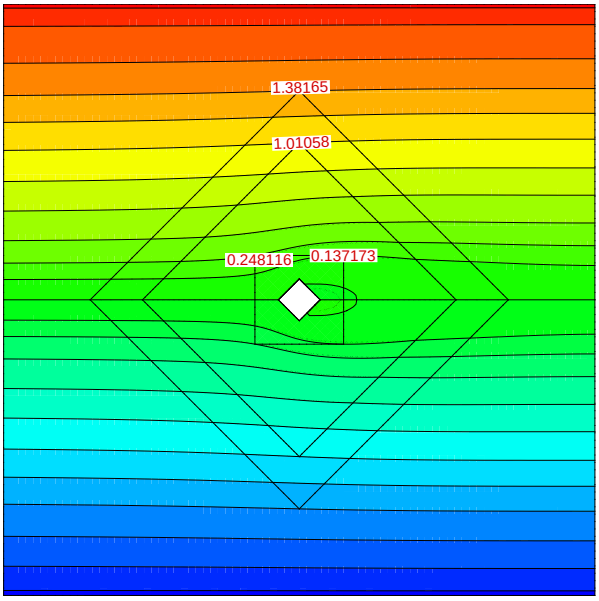


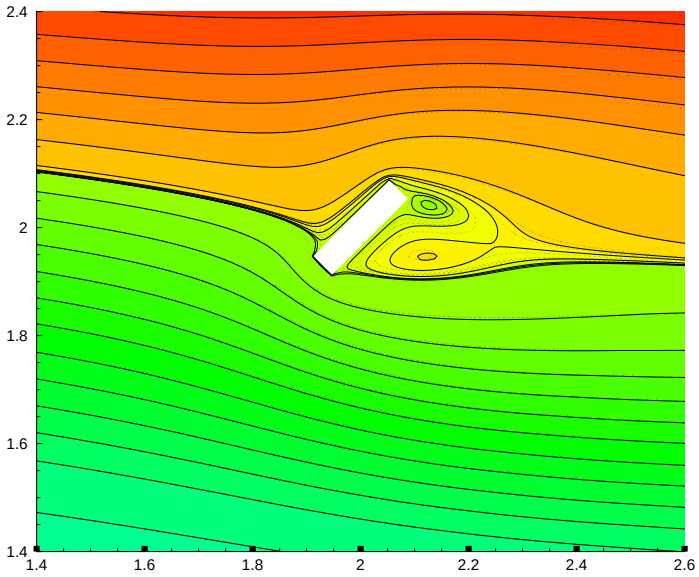
Figure 3.19. Infinity sensitivity test for the external flow problem at Re=100, h/L=1/20

Table 3.11. Stream function values at sample points

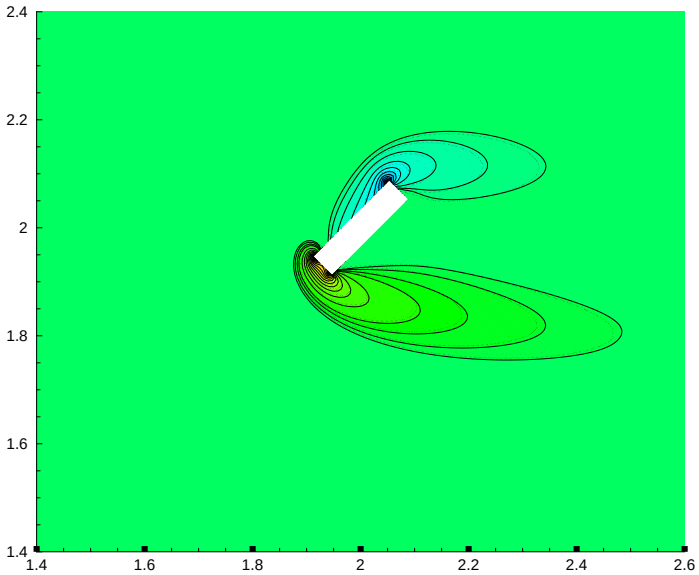
h/L	Sample point 1	Sample point 2	Sample point 3	Sample point 4
1/18	1.39098	1.01542	0.24941	0.13868
1/19	1.38679	1.01236	0.24874	0.13782
1/20	1.38165	1.01058	0.24812	0.13717
1/21	1.38147	1.01057	0.24810	0.13712

In external flow problems, the solid body shape has considerable effects on flow properties. In Figures 3.21-3.25, the solid bodies have different aspect ratios (width of the body over length of the body). The other parameter that determines body shape is the angle-of-attack (the angle between the upstream velocity and the x axis of the object) and this parameter is investigated for a thin solid body in Figure 3.20. To have a better visualization, we should focus on the region closely around the body except for the Figure 3.24. The computational domain size for other problems is same with this problem.

If the width of the body is less than the length or the angle-of-attack is not too large, the adverse pressure gradient is not too much. As a result, the boundary layer fluid can flow into the slightly increasing pressure region without separating from the surface. However, if the solid body is thick or the angle-of-attack is too large, the pressure gradient is too adverse so the boundary layer will separate from the surface. The bodies with the minimum separation from the surface are called streamlined bodies. These bodies are designed to reduce the effects of separation. Separation from the surface creates a large drag due to low pressure in the wake. Those bodies which have large separation are called blunt bodies. Although the boundary layer may be quite thin, it can appreciably alter the entire flow field because of the boundary separation for the blunt bodies.



(a) Stream function



(b) Vorticity

Figure 3.20. Thin body with 45^0 angle-of-attack

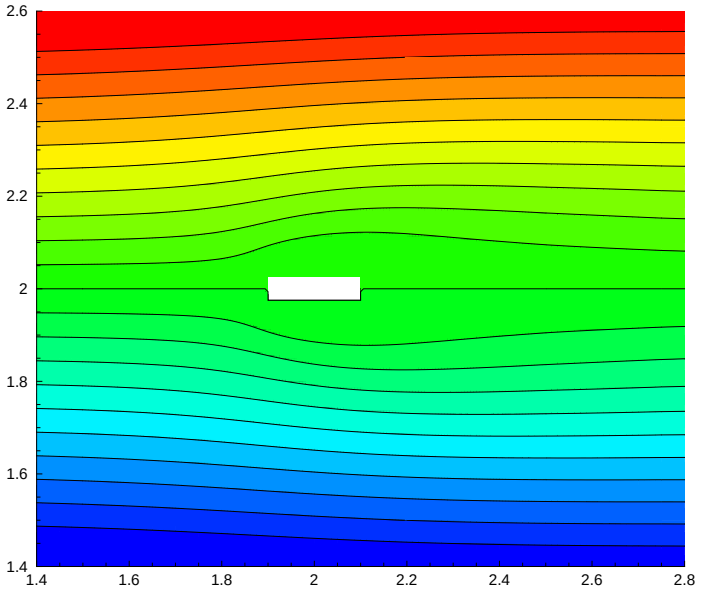


Figure 3.21. Stream function values for the body with W/L=1/4

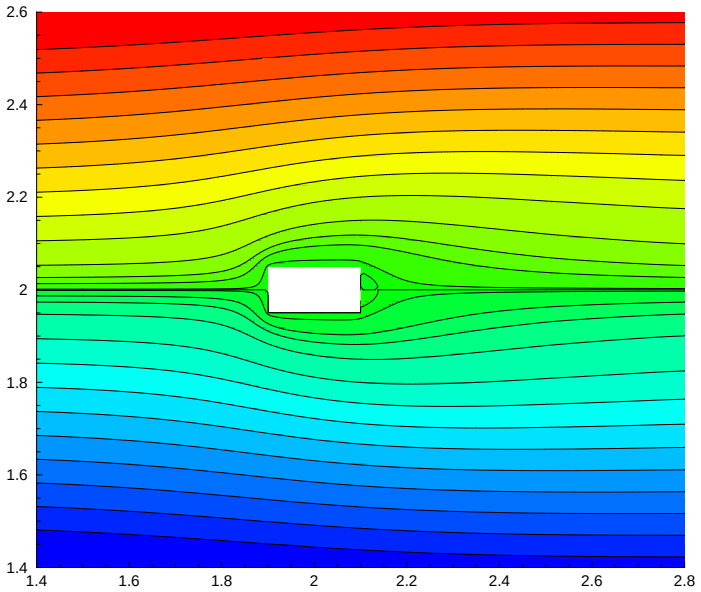
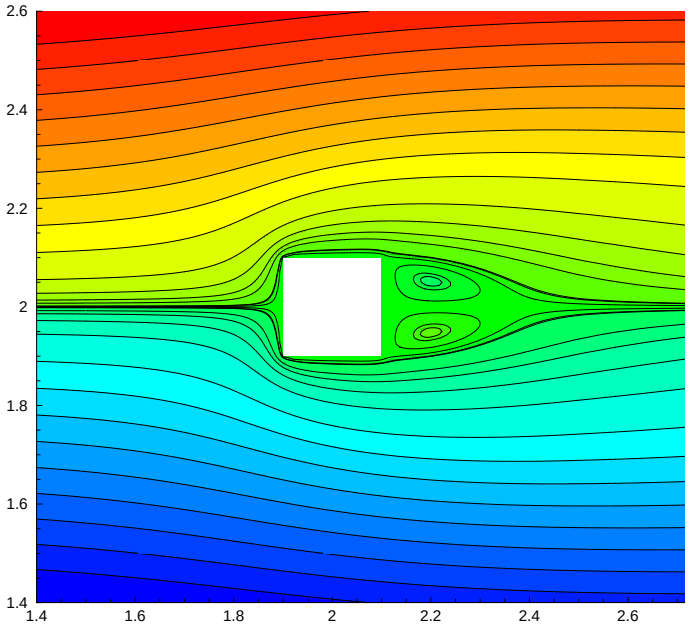
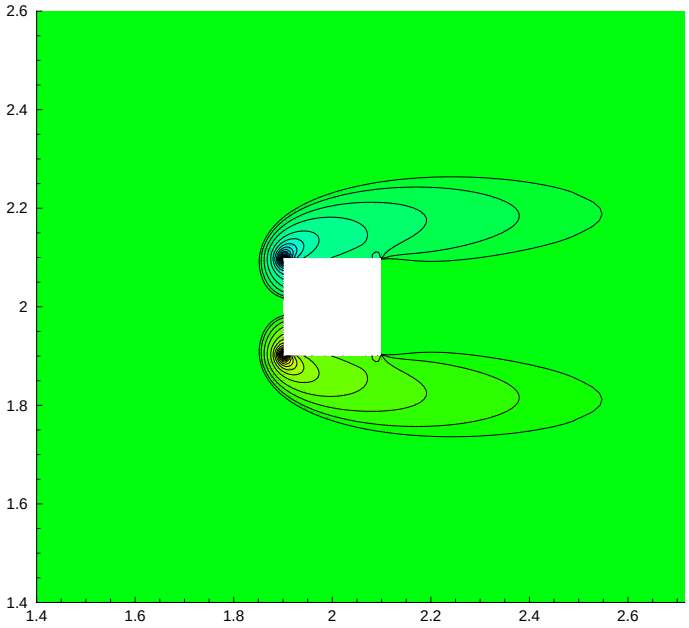


Figure 3.22. Stream function values for the body with W/L=1/2



(a) Stream function



(b) Vorticity

Figure 3.23. Stream function and vorticity values for the body with W/L=1

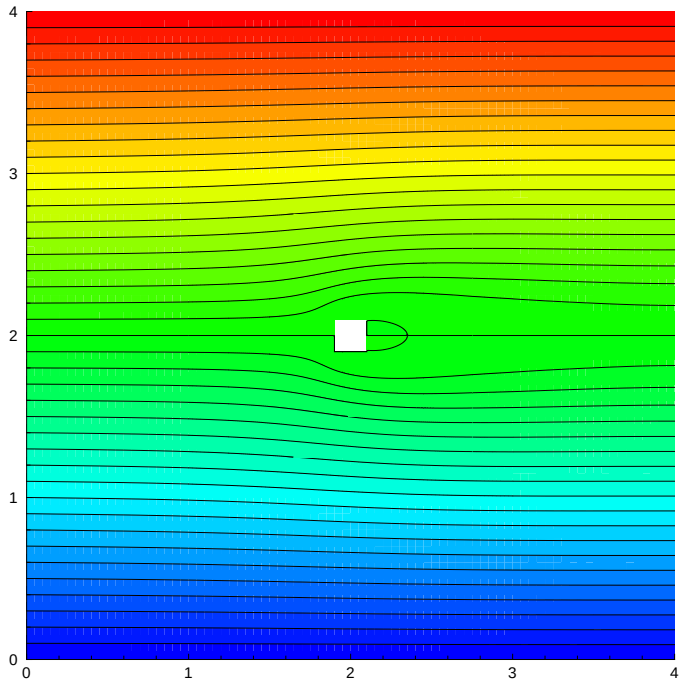


Figure 3.24. Original domain sizes for the body with W/L=1

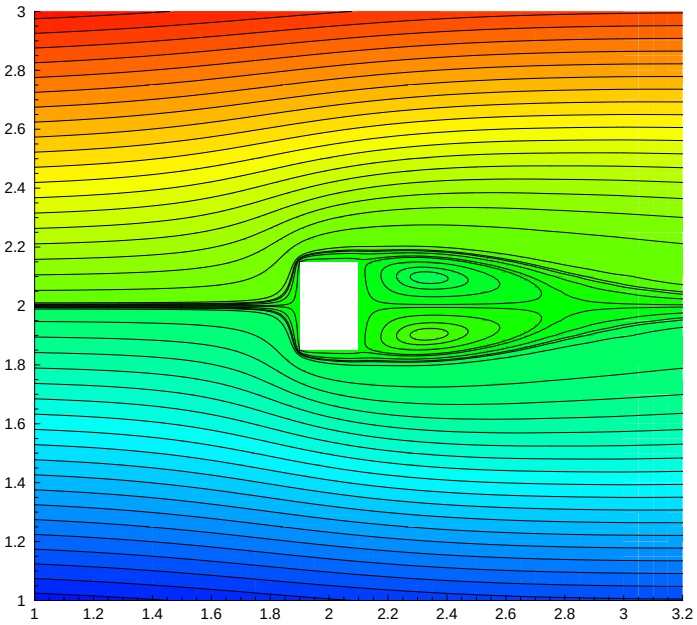
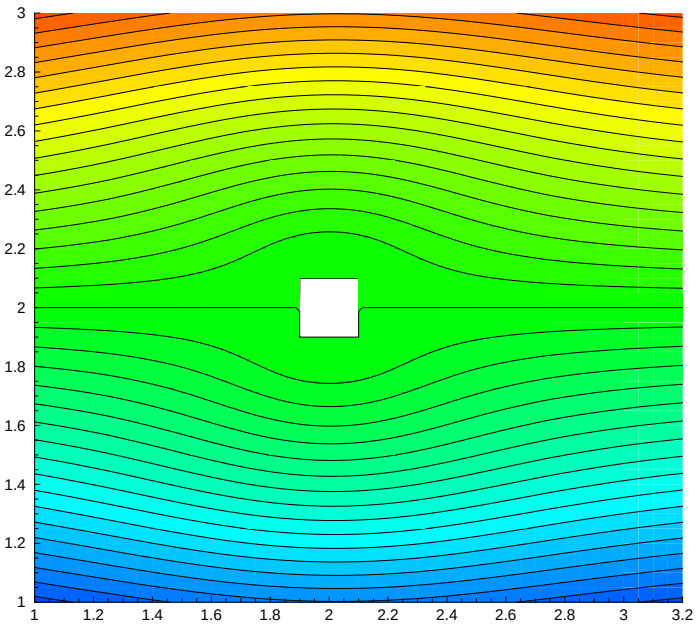


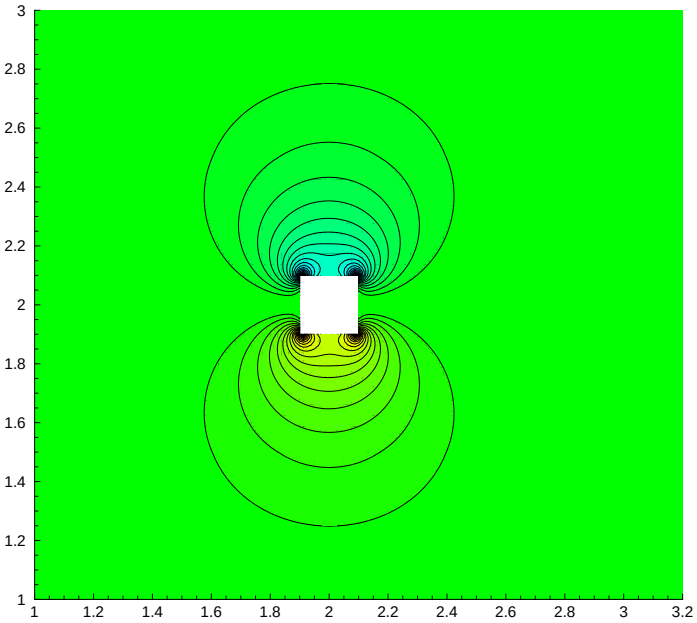
Figure 3.25. Stream function values for the body with W/L=3/2

In the external flow, the shape of the immersed body mainly determines the character of the flow field. Besides the body shape, size, orientation, and fluid properties, there are other effects in determining flow field character. Reynolds number and Mach number are the most important non-dimensional parameters for external flow problems. In this study, the effects of the Reynolds number are investigated. Reynolds number represents the ratio of inertial effects to viscous effects. The character of the flow changes greatly when Reynolds number changes from Re>>1 to Re<<1. This is because, when Reynolds number is less than unity, the viscous effects are dominant and when Reynolds number is greater than unity, inertia effects are dominant. If the Reynolds number is small, the viscous effects are relatively strong and solid body affects the uniform upstream flow. The viscous effects are felt far from the objects in all directions. As the Reynolds number increases, except for the downstream, the region in which viscous effects are dominant becomes smaller in all other directions.

In Figures 3.26-3.29, the external flow fields around a solid body are investigated for different Reynolds numbers. For all of the problems, computational domain has dimensions (0,4)×(0,4) and solid body has dimensions (0,0.2)×(0,0.2). This solid body is placed at the center of the computational domain. For better visualization, the region around the solid body is zoomed. In order to investigate Reynolds number effects, the other parameters are held same, only the Reynolds number changes Re=0.001, Re=50, Re=100, and Re=150.

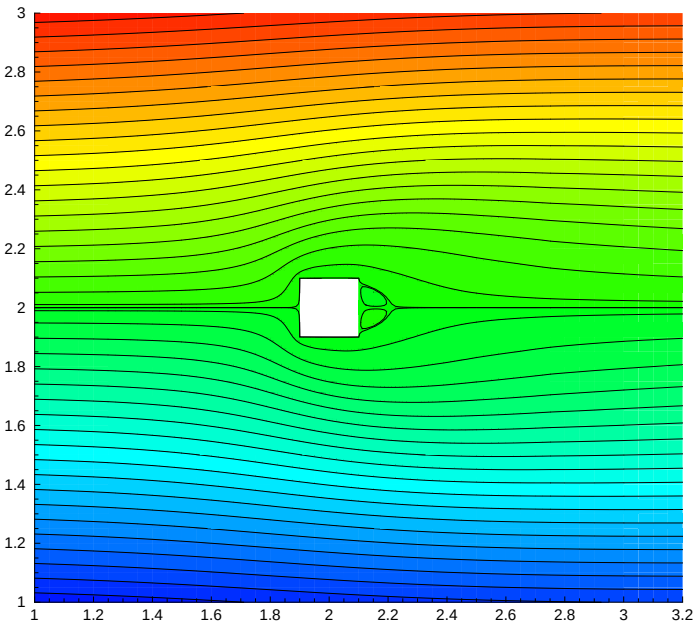


(a) Stream function

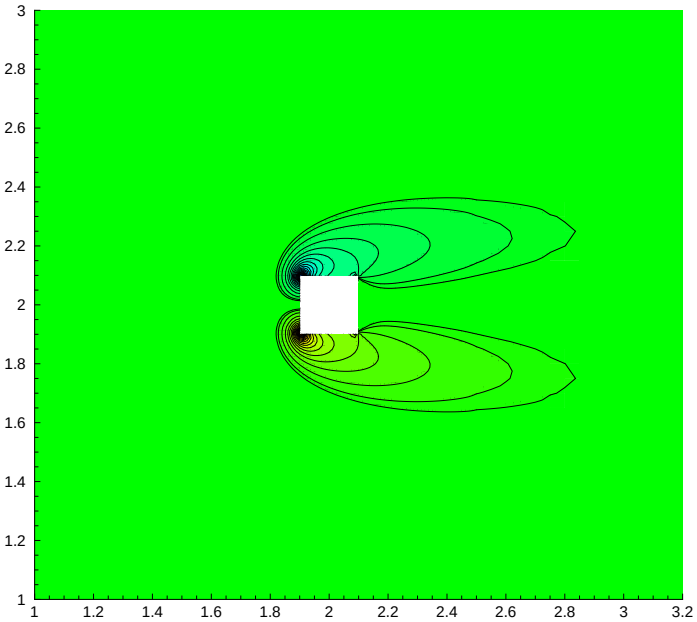


(b) Vorticity

Figure 3.26. Stream function and vorticity values for Re=0.001

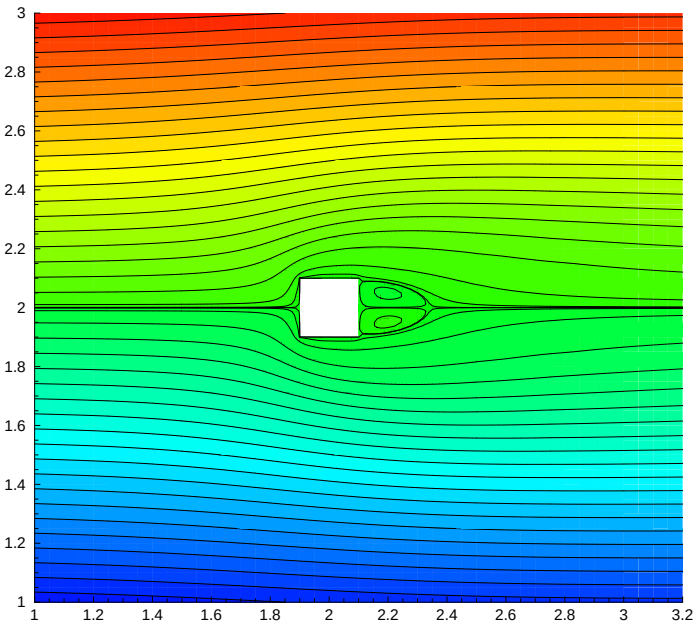


(a) Stream function

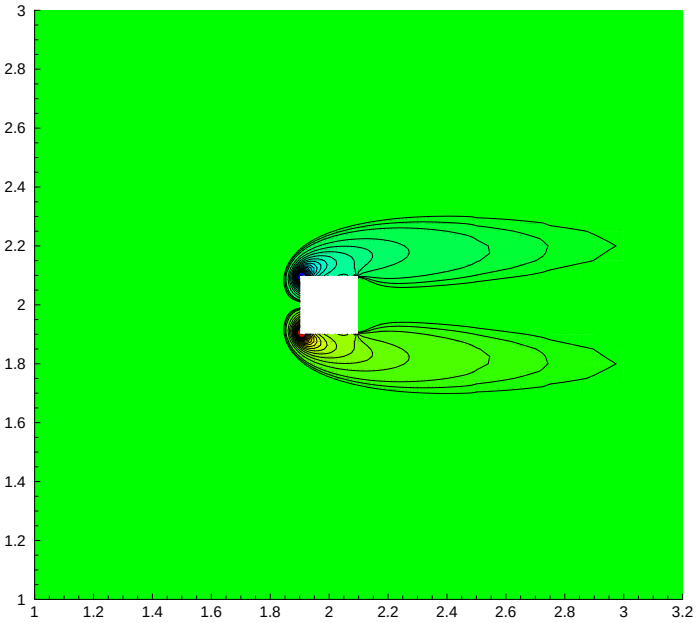


(b) Vorticity

Figure 3.27. Stream function and vorticity values for Re=50

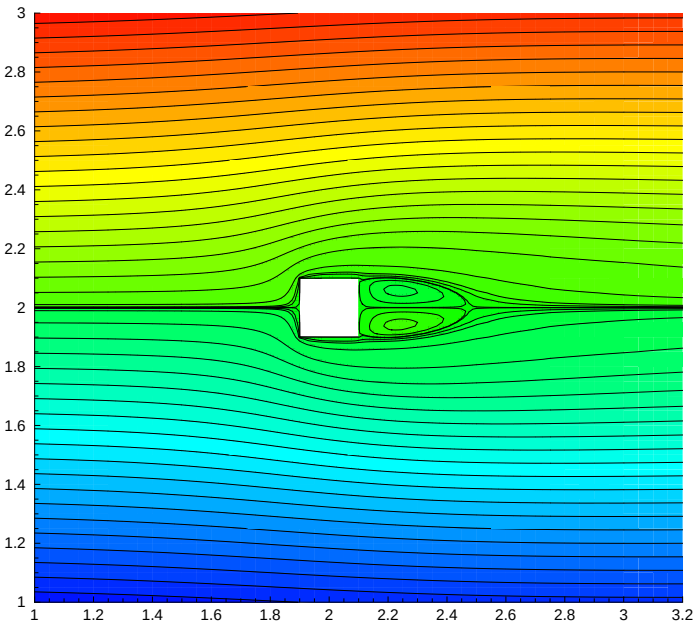


(a) Stream function

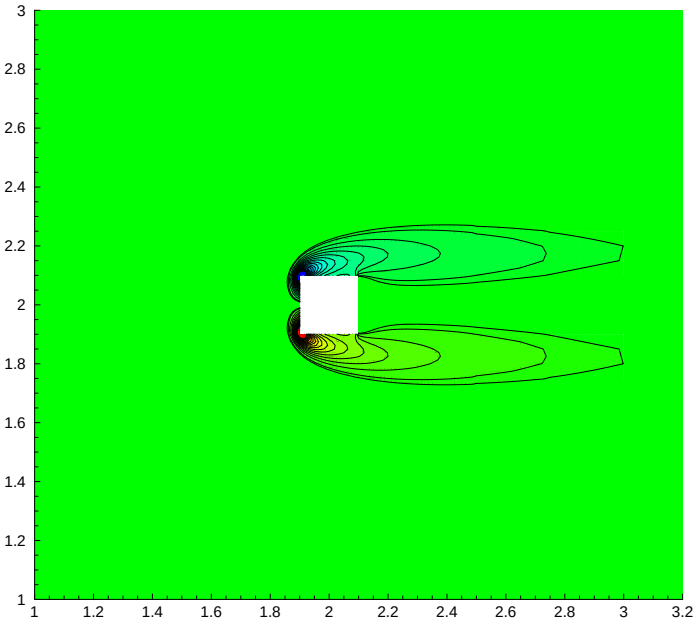


(b) Vorticity

Figure 3.28. Stream function and vorticity values for Re=100



(a) Stream function



(b) Vorticity

Figure 3.29. Stream function and vorticity values for Re=150

The solution of the domain 1 and domain 2 update each other in order to get artificial boundary conditions. The number of updates between domains can initially be given a fixed number. The coding of this type algorithm is very simple but there is no guarantee for convergence after previously determined fixed number of iteration. In this work, the norms of the difference vectors for the following iterates are controlled in order to have converged domain decomposition algorithm. The limits of convergence is chosen as 1e-6 for all domains. In Figure 3.30, the reduction of these difference norms for domain 1 and domain 2 are given with respect to the number of visits between domain 1 and domain 2 for Re=100 and zero angle-of-attack. For this problem, the reduction of the residual of the sample outer iteration and inner iteration is shown in the Figure 3.31. This sample outer iteration is the second visit of domain 1 and it is tabulated in Table 3.12. In order to look at the whole picture of convergence, the nonlinear residual vector norms for both domain 1 and domain 2 is drawn in the Figure 3.32

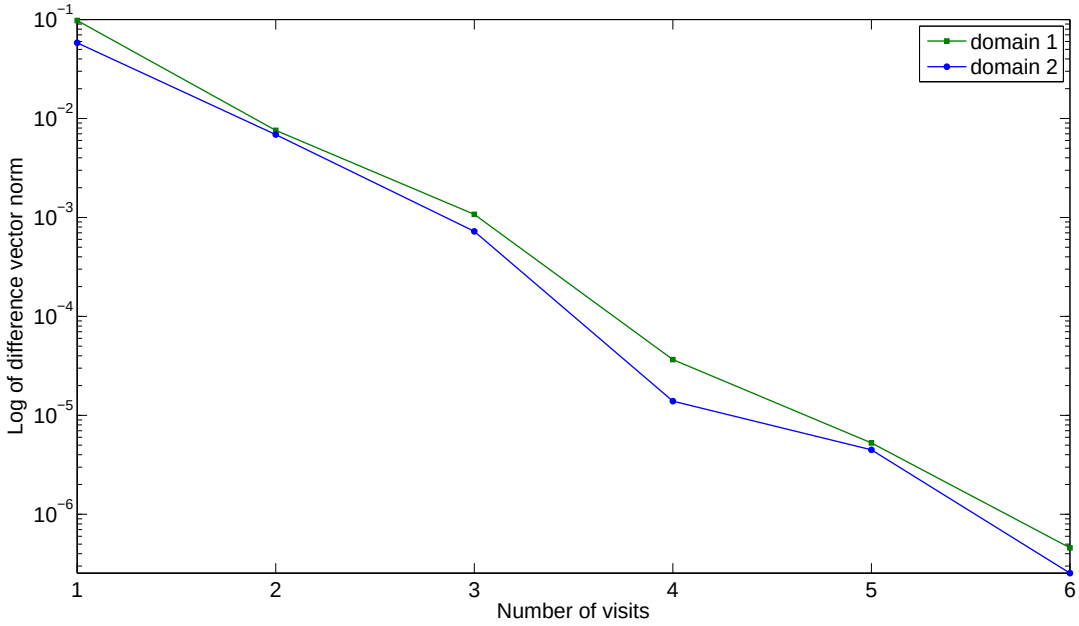
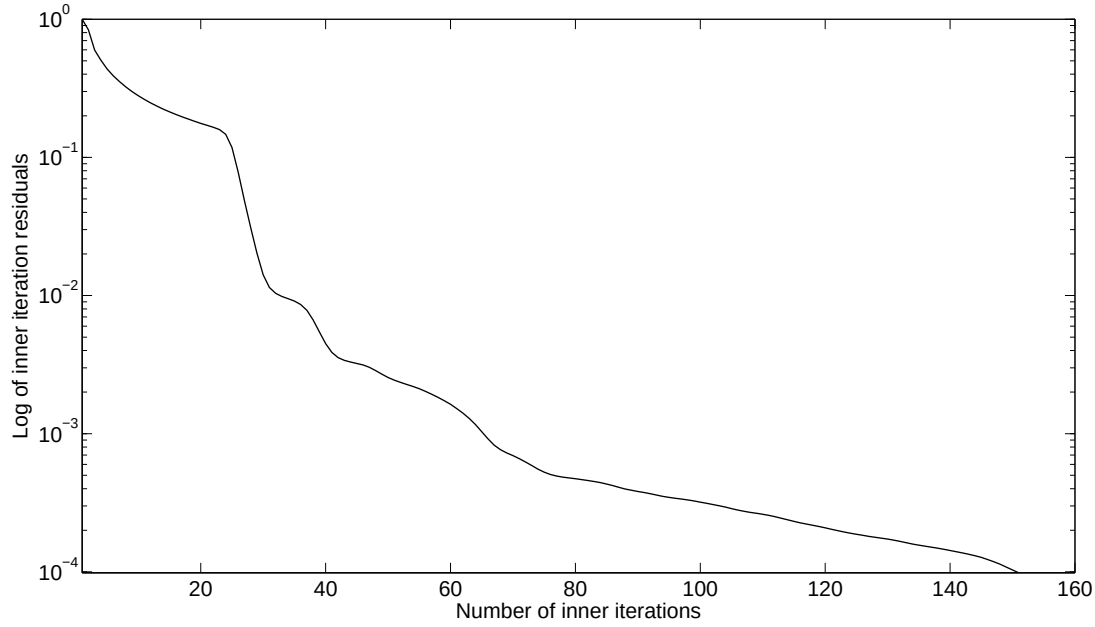
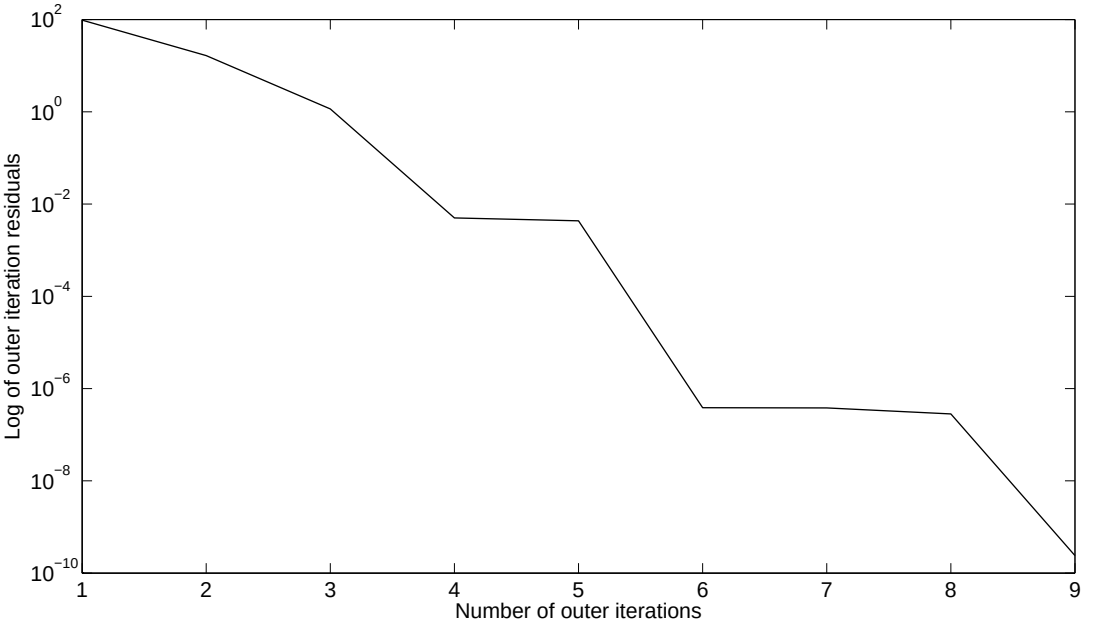


Figure 3.30. Reduction of difference vector norm for the single body external flow problem at Re=100

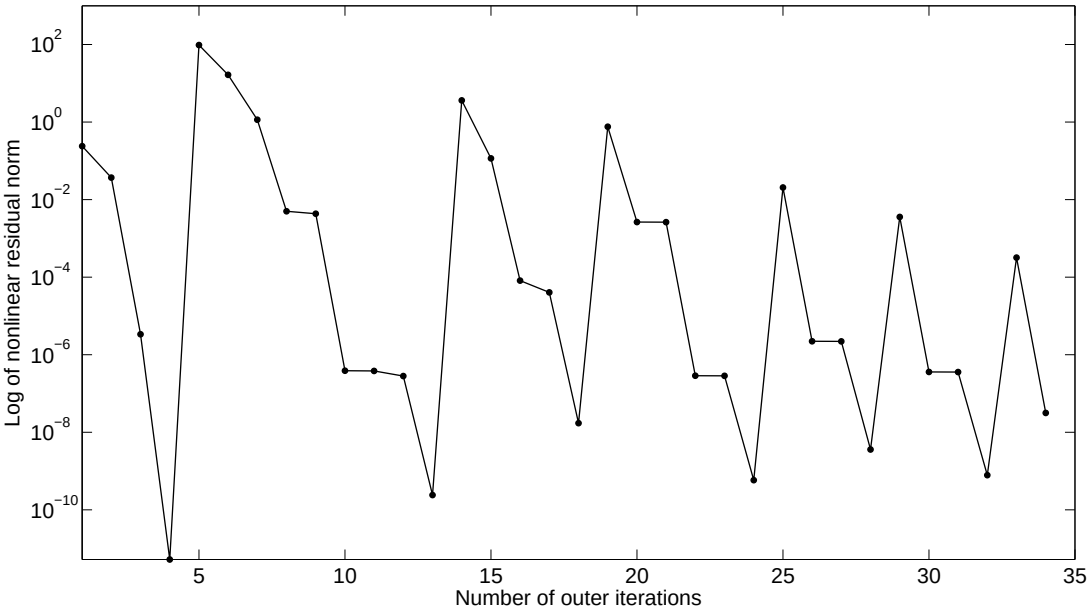


(a) Inner iterations

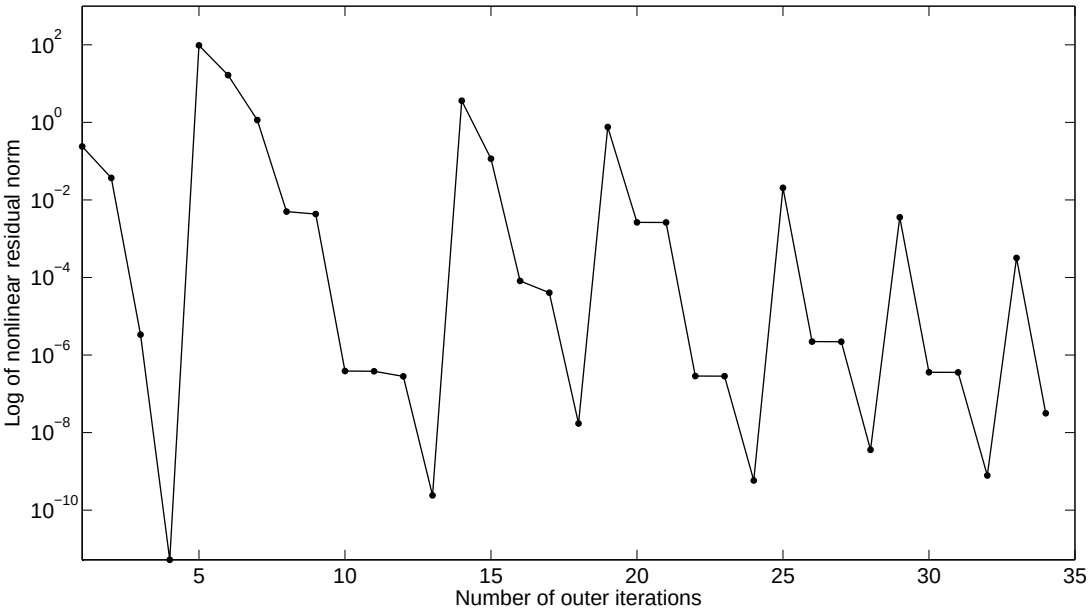


(b) Outer iterations

Figure 3.31. Residual norms of the inner and outer iterations



(a) Domain 1



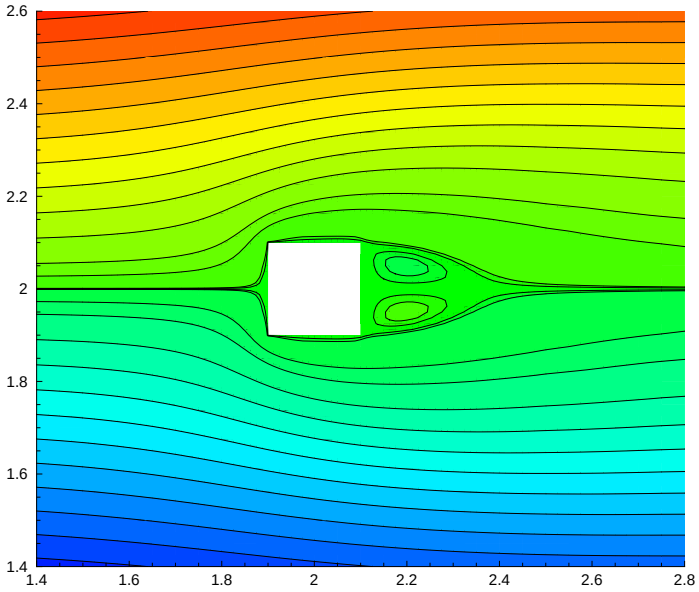
(b) Domain 2

Figure 3.32. Nonlinear residual norm for single body external flow problem

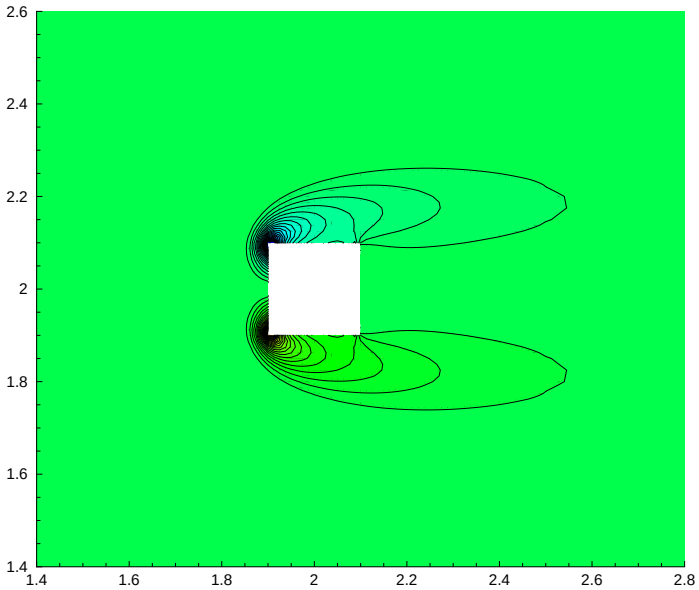
Table 3.12. A sample nonlinear step for the single body external flow problem

VISIT	2					
DOMAIN	1					
LEVEL	1					
NONLINEAR INITIAL NORM	0.9713E+02					
NONLINEAR TOLERANCE	0.1000E-06					
Newton	Linear	nonlinear residual	linear resiual	lambda	eta	time
1	150	1.66E+01	9.85E-05	1.00E+00	1.00E-04	4.31
2	168	1.15E+00	9.98E-05	1.00E+00	1.00E-04	7.73
3	197	5.01E-03	9.91E-05	1.00E+00	1.70E-05	9.03
4	302	4.33E-03	1.49E-05	1.00E+00	1.00E-04	18.27
5	108	3.85E-07	9.23E-05	1.00E+00	1.00E-07	1.98
6	1008	3.81E-07	9.83E-08	5.00E-01	1.00E-04	54.72
7	353	2.82E-07	9.86E-05	1.00E+00	1.00E-04	19.27
8	295	2.40E-10	9.95E-05	1.00E+00	6.50E-07	19.06
8	2581					134.38

As stated previously, the orientation of the body is an important factor that determine flow characteristic. The angle-of-attack, which is the angle between the upstream velocity and the x axis of the solid body, determines the orientation of a solid body in the center of the computational domain. In Figures 3.33-3.36, a rectangular shaped solid body (0,0.2)×(0,0.2) is placed in the center of the computational domain which has(0,4)×(0,4) dimensions. Reynolds number is chosen as 100 for these problems. The only changing parameter for this problems is the angle-of-attack and it changes as 0^0, 15^0, 45^0, and 75^0. In these figures, the physical domain rotates with the solid body whereas the inner artificial boundary of the domain 1 (Figure 3.17) remains stationary. The region around the solid body is zoomed again. The original domain is given for the 45^0 problem and also the rotation of the domains around the solid walls and inner subdomains is shown in Figure 3.35 for this problem.

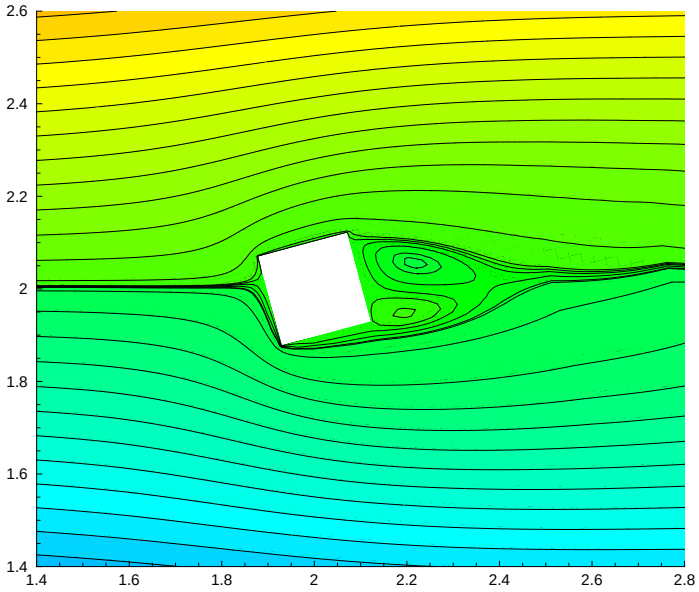


(a) Stream function

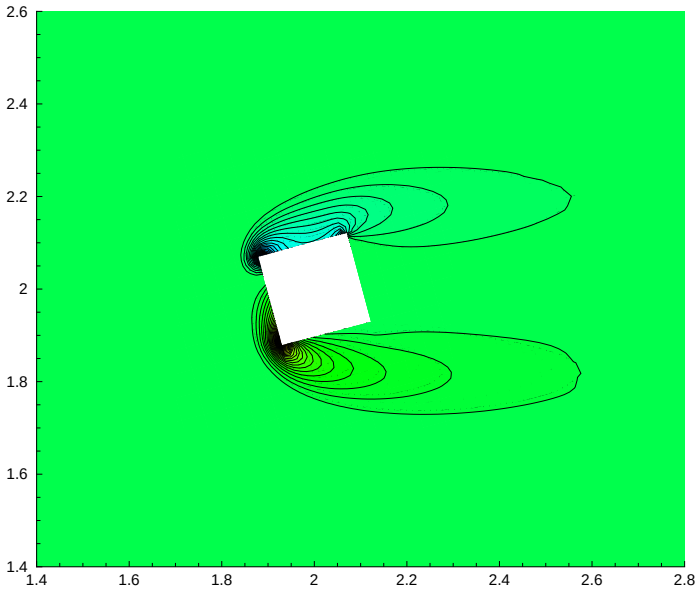


(b) Vorticity

Figure 3.33. Stream function and vorticity values for 0^0 angle-of-attack

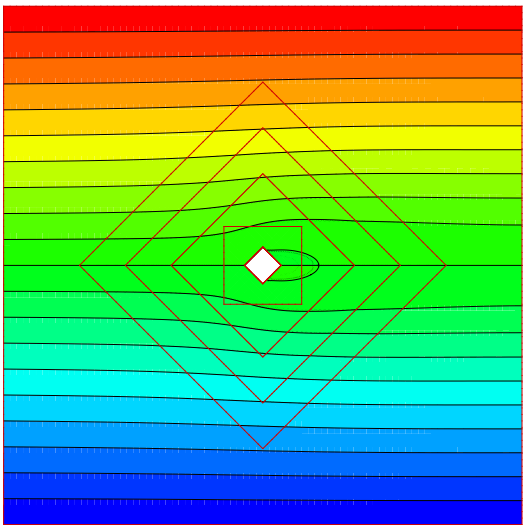


(a) Stream function

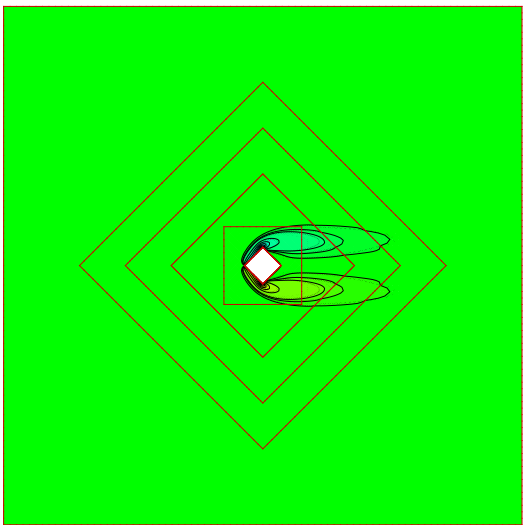


(b) Vorticity

Figure 3.34. Stream function and vorticity values for 15^0 angle-of-attack

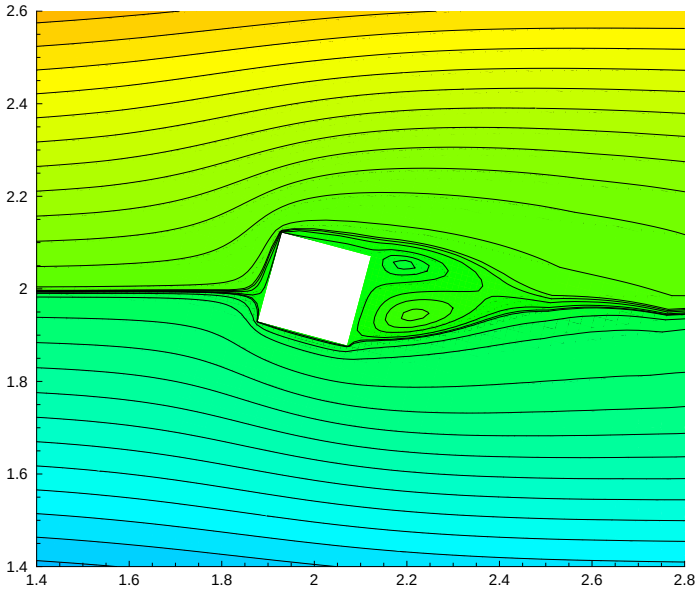


(a) Stream function

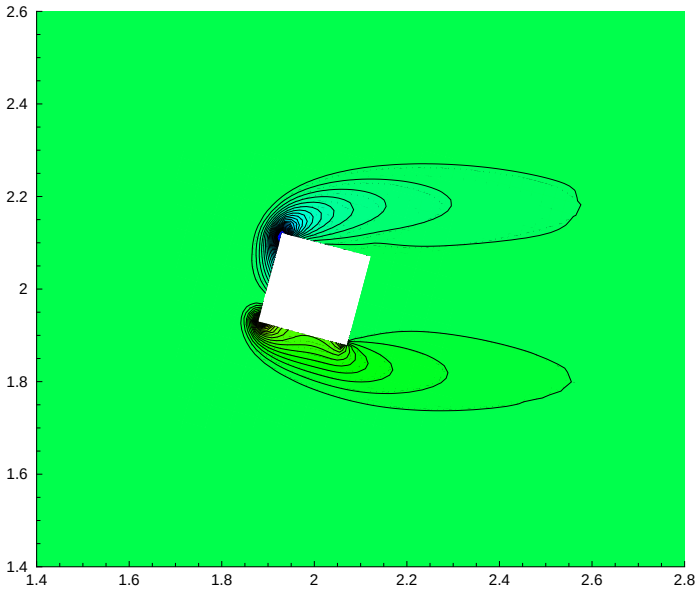


(b) Vorticity

Figure 3.35. Stream function and vorticity values for 45^0 angle-of-attack



(a) Stream function



(b) Vorticity

Figure 3.36. Stream function and vorticity values for 75^0 angle-of-attack

Like one-body case, difficulties in grid generation can be removed by using domain decomposition algorithm for the case of two immersed bodies. As stated previously, if one body rotates, there should be an extra subdomain. Otherwise, some parts of the inner artificial boundary of the domain 1 can not take value from the other subdomains. In Figure 3.37, sub-domains for a sample case can be seen. In this sample problem, top solid object has 75^0 angle-of-attack. If there is no angle-of-attack, there will be no need for the subdomain 4, but for this problem it is compulsory.

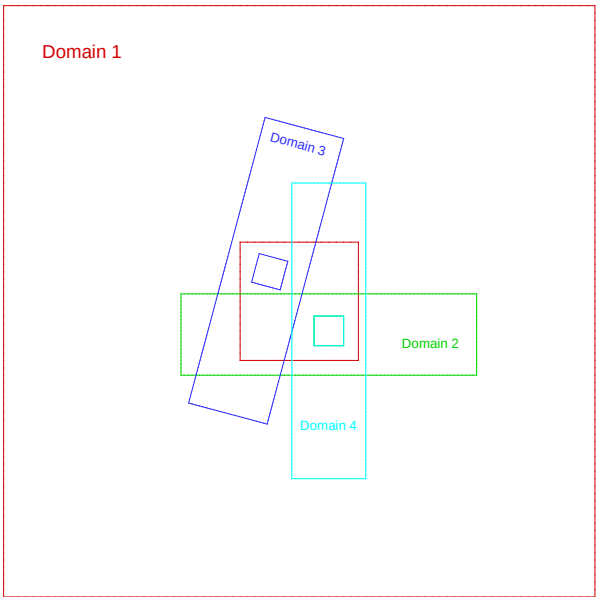


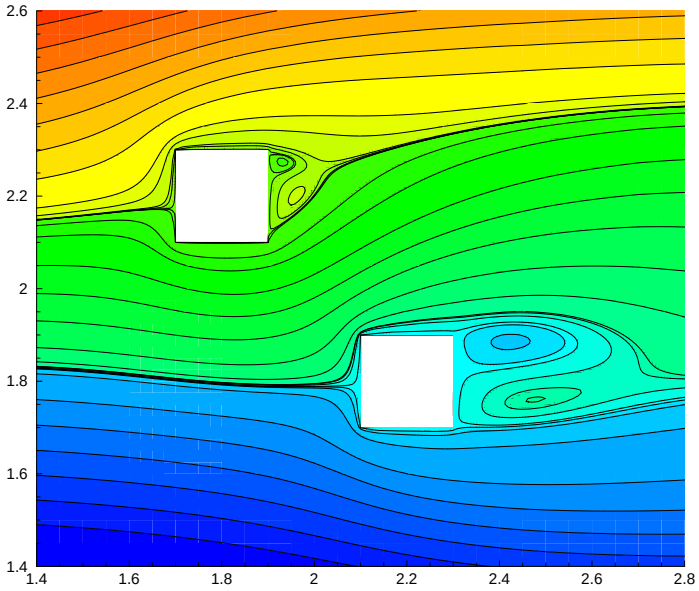
Figure 3.37. Sub-domains for the two immersed body case

For the two-body case, treatment of the boundary conditions associated with solid surfaces calls for particular care. This situation becomes especially interesting in the case of the non-symmetric multiply connected domains. The value of the stream function constant at the internal boundaries is unknown. In this study, an iterative procedure proposed in [22] is used to obtain these constants. The solution order of subdomains has important effects on these constants. For a symmetric problem, in order to have a good symmetric solution, a symmetric finite difference scheme (like central difference) should be used and the solution order of the subdomains should not distort the symmetry. In the two body problems, transmission between domain 1, 2, 3,

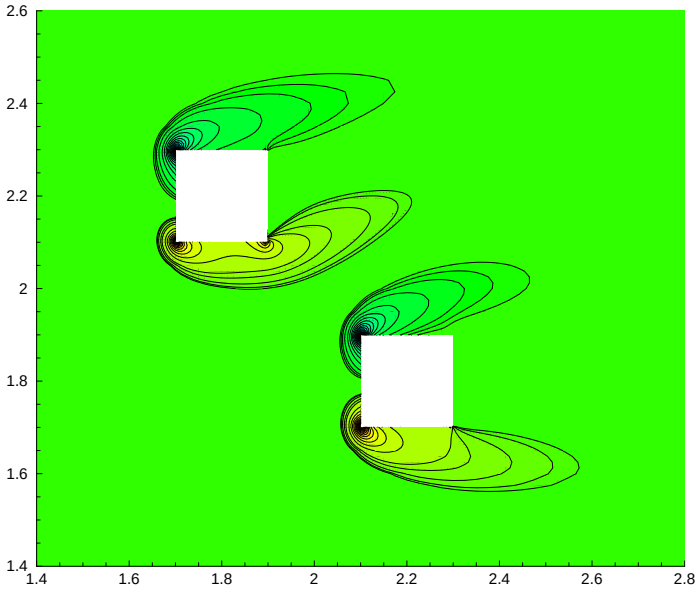
4 are called outer visit and at each outer visit there are inner visits between domain 2, 3, 4 in order to have good convergence and better symmetry for symmetric problems.

In Figures 3.38- 3.41, external flow over two immersed bodies is solved at Reynolds number 100. In these sample problems, the longitudinal spacing between the solid bodies is equal to the length of the one body and the top solid body has different angle-of-attacks at each problem. The effects of the angle-of-attack on the flow properties can be seen in these figures. As expected, these effects are more obvious for the rotated solid body.

Reynolds number is one of the most important non-dimensional parameters which has a big influence on flow properties. In figure 3.41, the two immersed body problem is solved at Reynolds 100. In order to see the effects of the Reynolds number, same problem is solved for Reynolds 0.01 and Reynolds 1. Solutions are given in Figures 3.42 and 3.43.

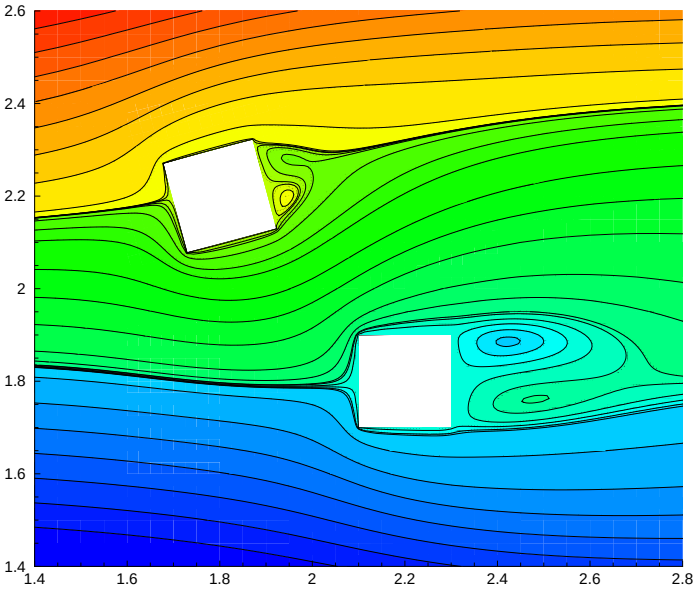


(a) Stream function

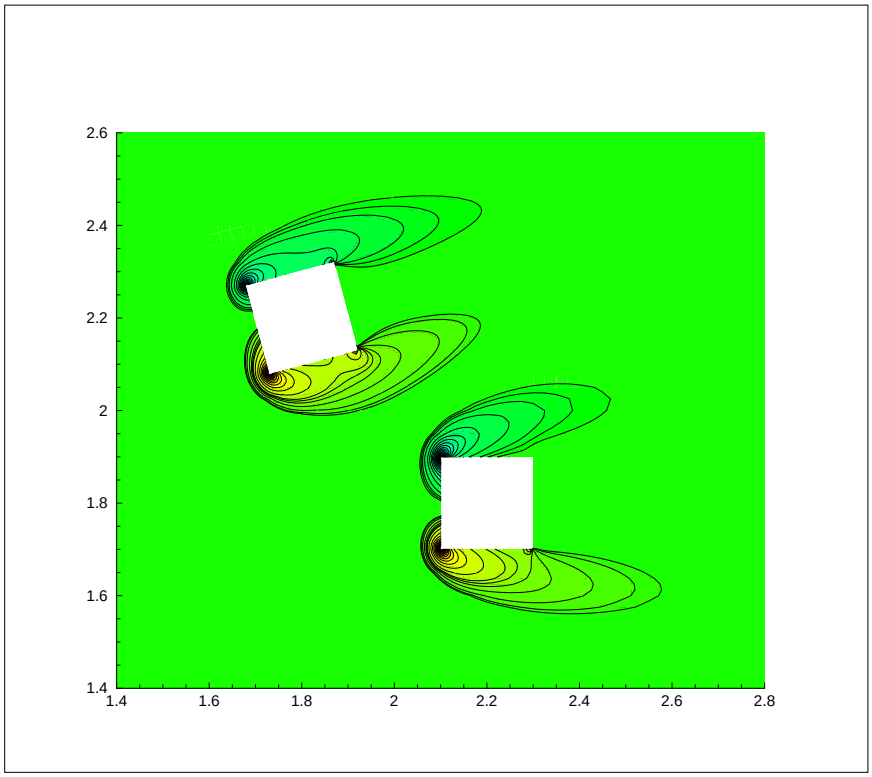


(b) Vorticity

Figure 3.38. Stream function and vorticity values for two bodies and top object has 0^0 angle-of-attack at Re=100

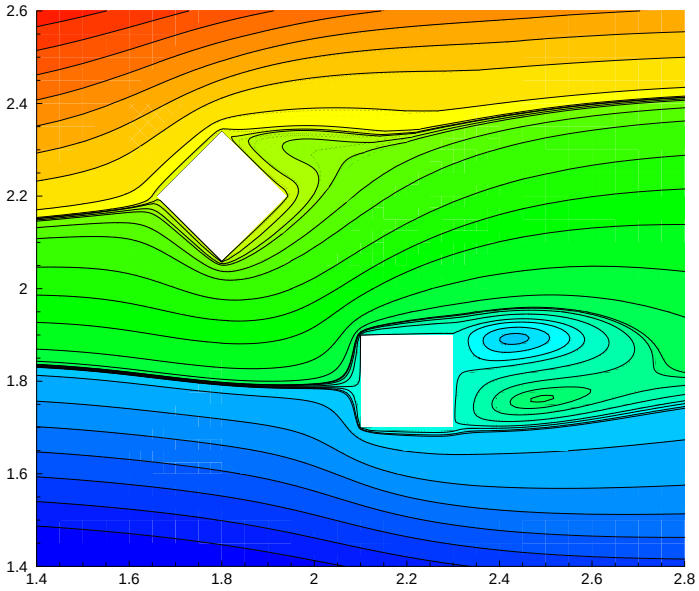


(a) Stream function

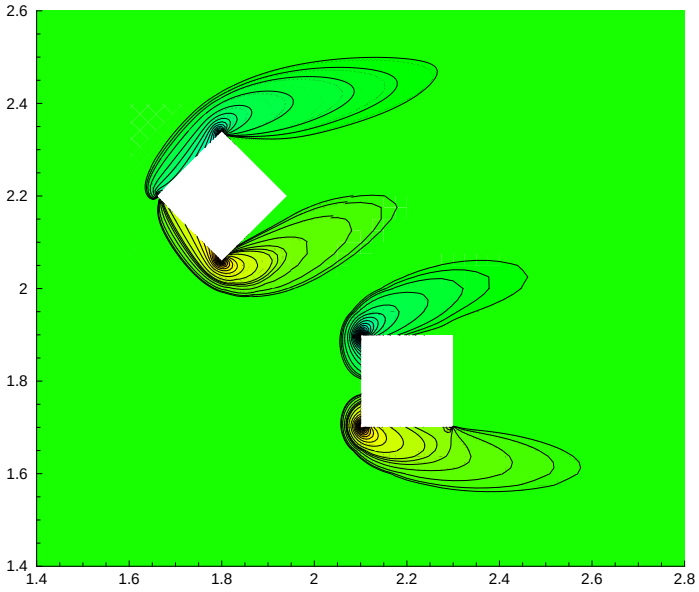


(b) Vorticity

Figure 3.39. Stream function and vorticity values for two bodies and top object has 15^0 angle-of-attack at Re=100

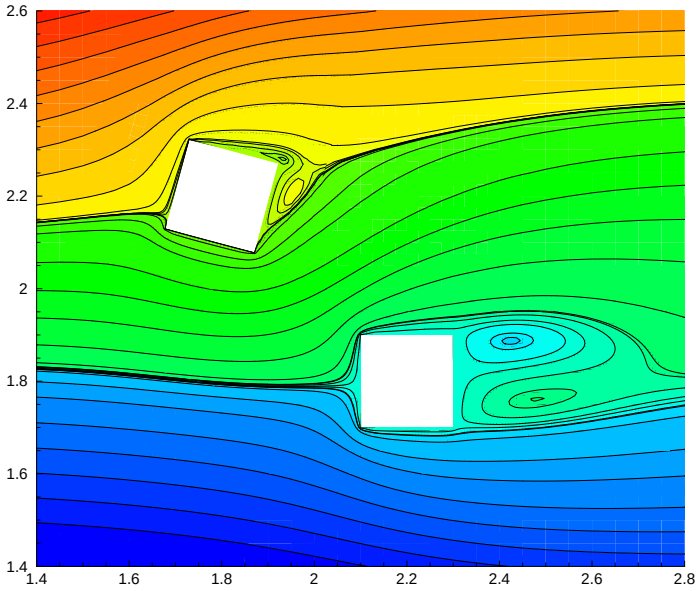


(a) Stream function

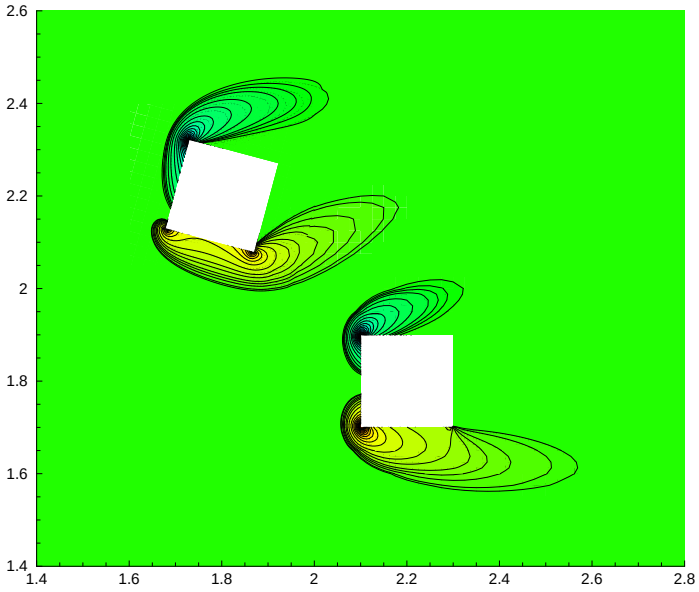


(b) Vorticity

Figure 3.40. Stream function and vorticity values for two bodies and top object has 45^0 angle-of-attack at Re=100

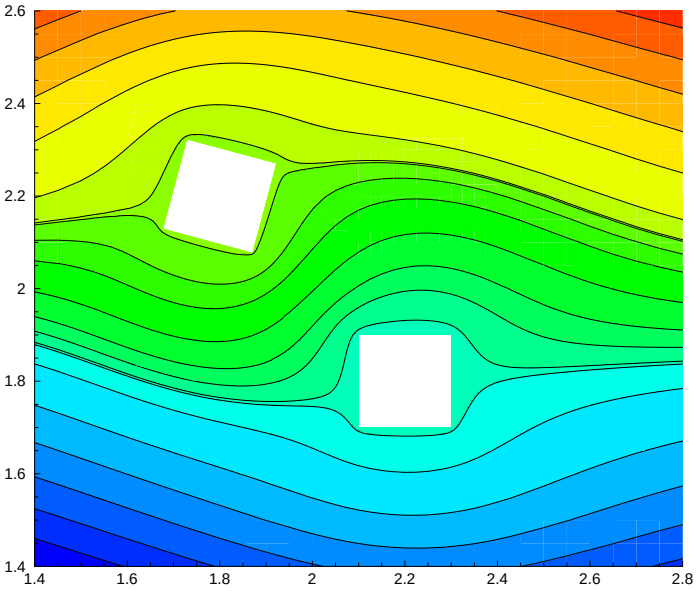


(a) Stream function

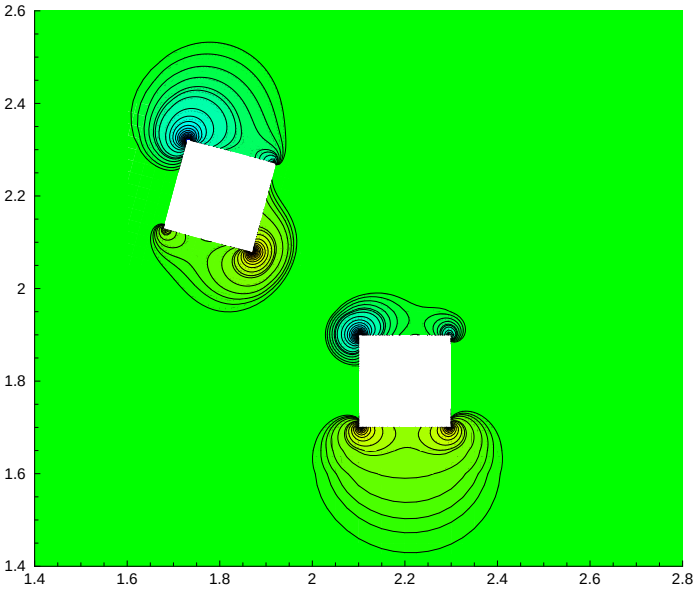


(b) Vorticity

Figure 3.41. Stream function and vorticity values for two bodies and top object has 75^0 angle-of-attack at Re=10^2

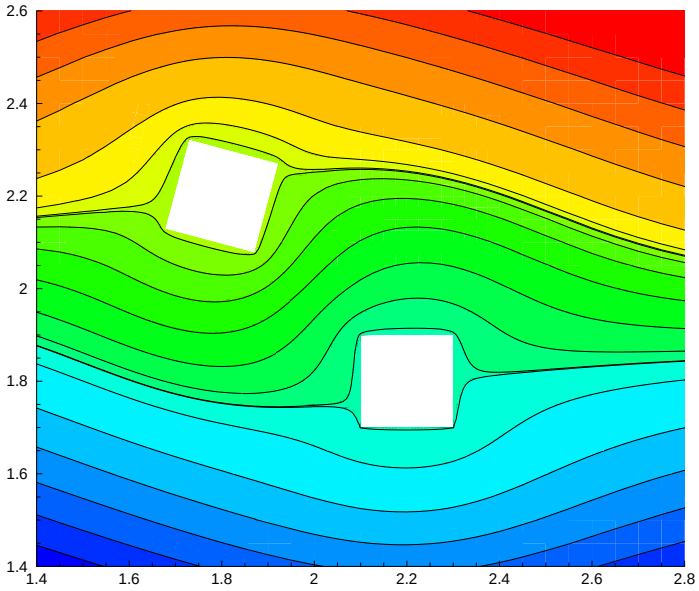


(a) Stream function

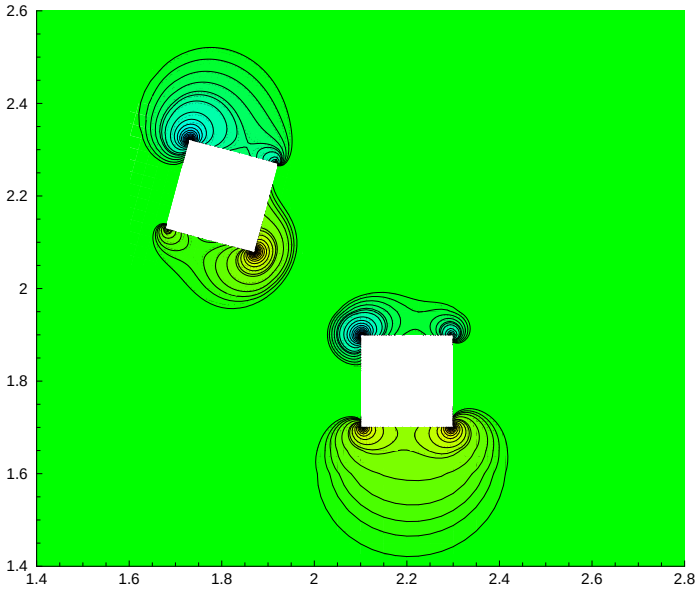


(b) Vorticity

Figure 3.42. Stream function and vorticity values for two bodies and top object has 75^0 angle-of-attack at Re=10^0



(a) Stream function

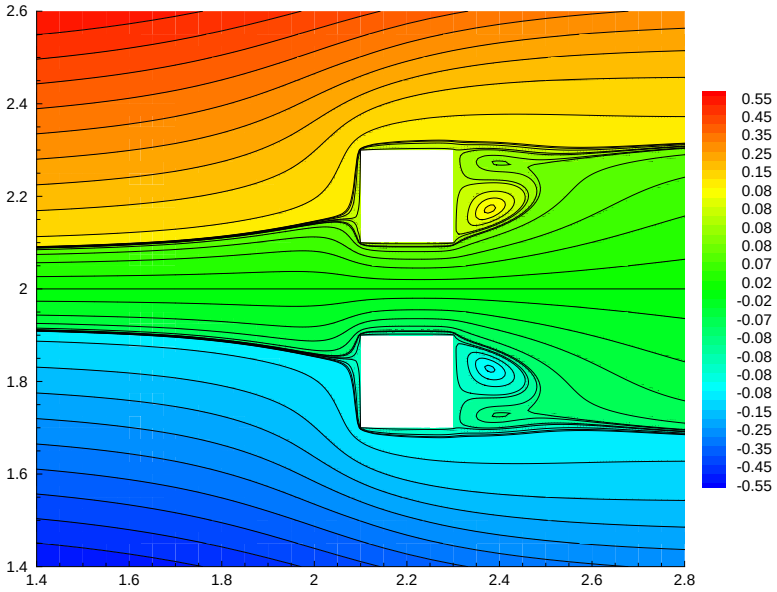


(b) Vorticity

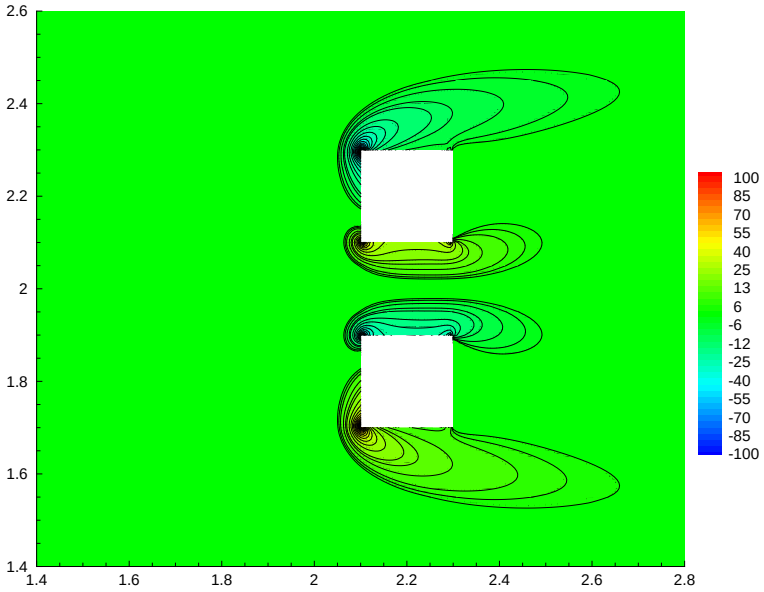
Figure 3.43. Stream function and vorticity values for two bodies and top object has 75^0 angle-of-attack at Re=10^{-2}

Passing manoeuvres may have important effects on the stability of road vehicles. When two vehicles are driven in close proximity, they mutually influence the flow field around each other. In this study, two rectangular solid bodies are used as passing vehicles for 2D viscous incompressible flow. Even passing manoeuvre is a totally unsteady phenomenon, time is frozen for specific locations of the solid bodies and we look for the steady-state solutions. The overtaken body is stationary and overtaking body passes it. The flow configuration is complex due to the interaction between the two wakes. It is clear that the longitudinal spacing between the two bodies greatly influences the flow structure. In this study, we restrict ourselves to Reynolds number equal to 100. We denote the distance between the centers of the rectangular bodies as X and the length of the object as L. In Figures 3.44-3.49, the steady state stream function and vorticity values are shown for 6 different spacings: X/L=0, X/L=0.5, X/L=1, X/L=1.5, X/L=2, X/L=2.5.

Solutions of the domain 1, domain 2, domain 3 update each other after the inner convergence between domain 2 and domain 3. The number of updates between domains can initially be given a fixed number. The norms of the difference vectors for the following iterates are controlled in order to have converged domain decomposition algorithm. The limits of convergence is chosen as 1e-8 for all the domains. In Figure 3.50, reduction of these difference norms for domain 1, domain 2, domain 3 are given with respect to number of outer visit. For this problem, the reduction of the residuals of the sample outer iteration and inner iteration are shown in Figure 3.51. In the table 3.13, the computational load is summarized for 6 different longitudinal distances.

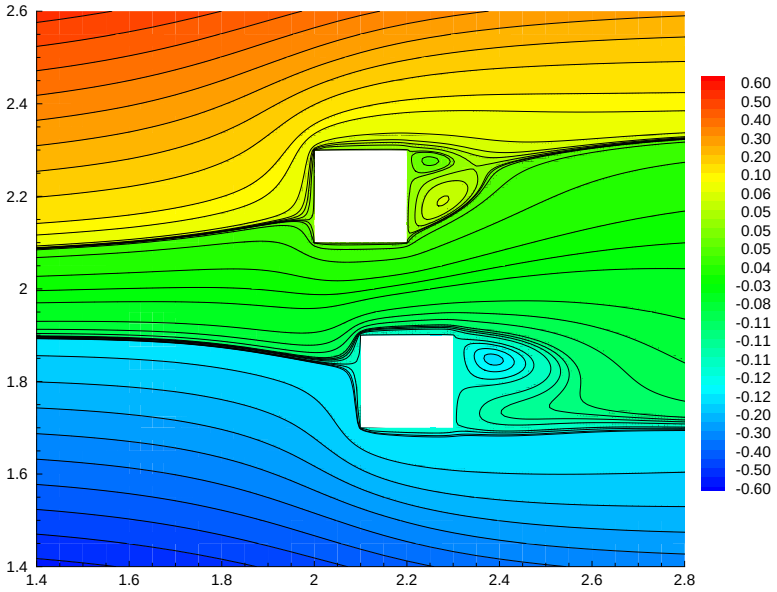


(a) Stream function

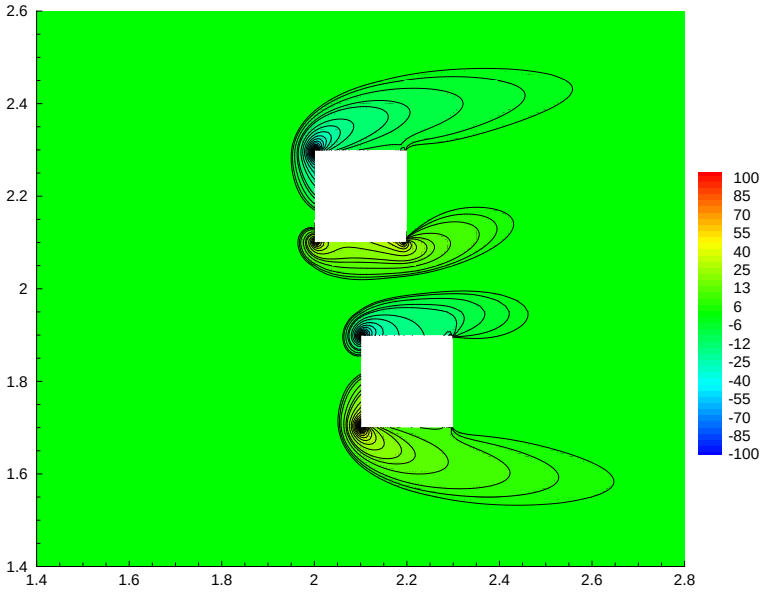


(b) Vorticity

Figure 3.44. Stream function and vorticity values for two bodies for X/L=0 at Re=100

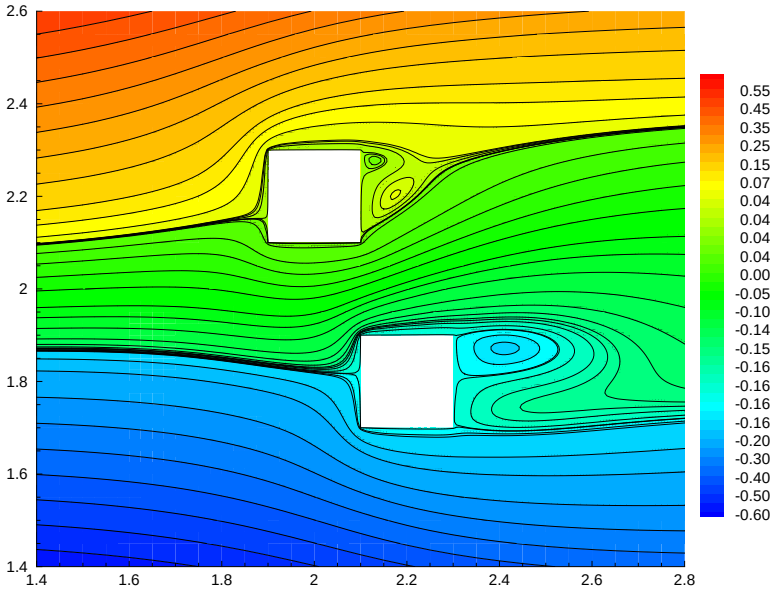


(a) Stream function

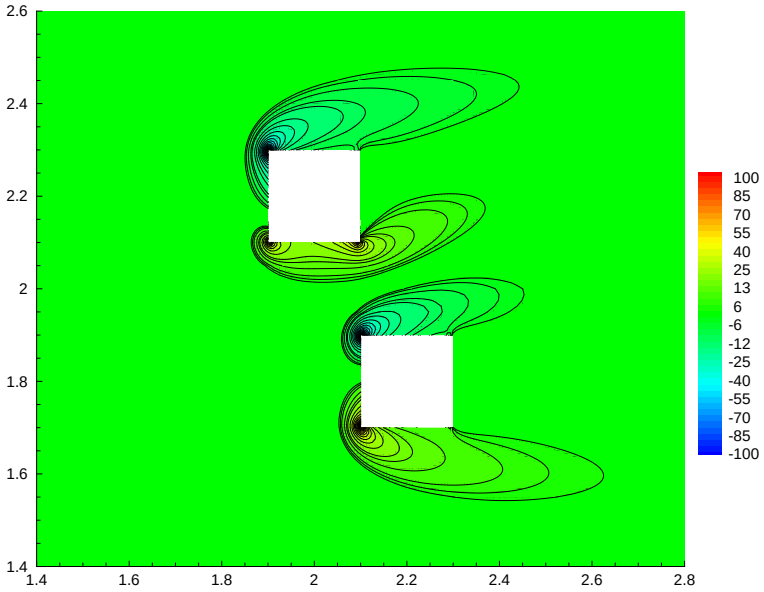


(b) Vorticity

Figure 3.45. Stream function and vorticity values for two bodies for X/L=0.5 at Re=100

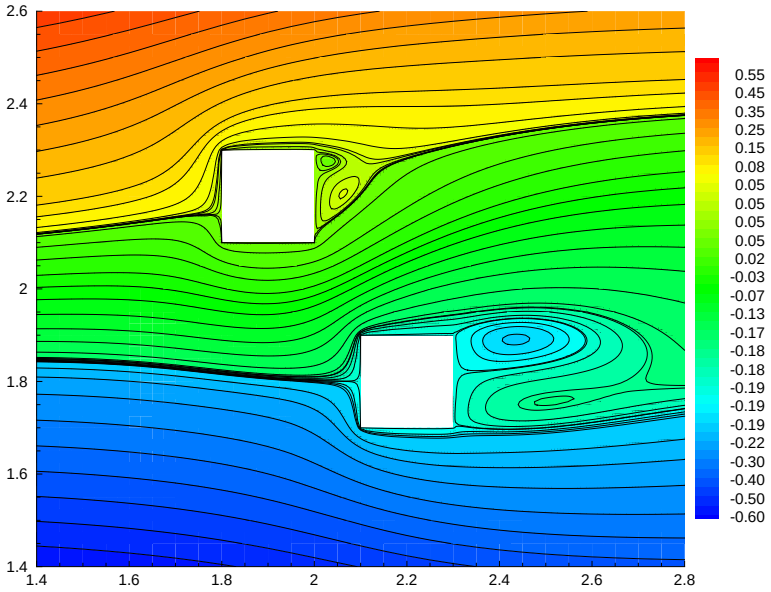


(a) Stream function

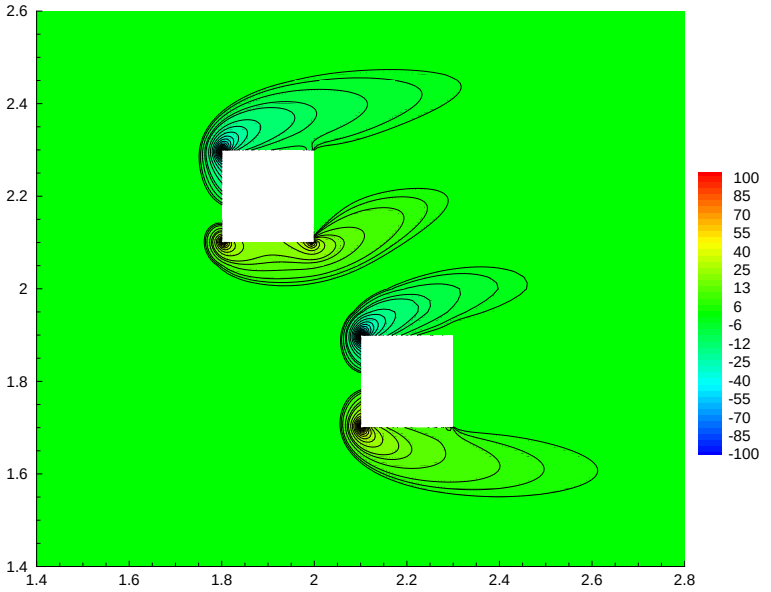


(b) Vorticity

Figure 3.46. Stream function and vorticity values for two bodies for X/L=1 at Re=100

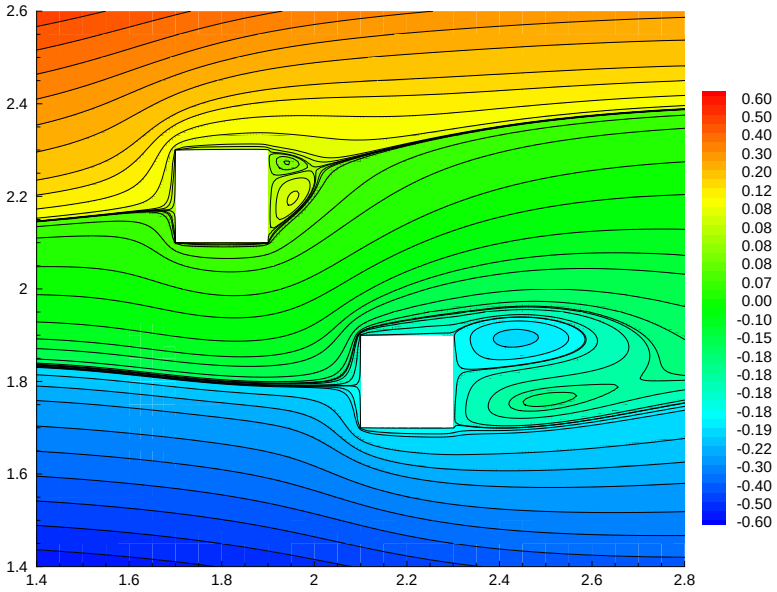


(a) Stream function

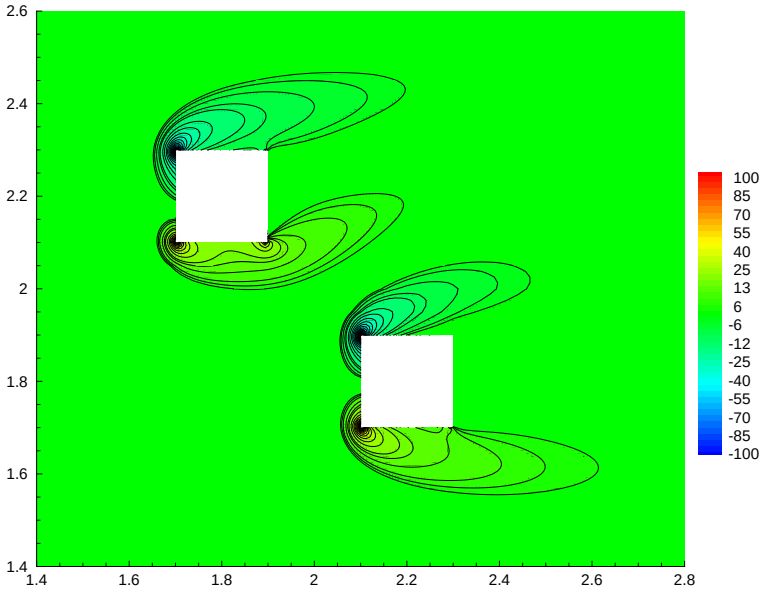


(b) Vorticity

Figure 3.47. Stream function and vorticity values for two bodies for X/L=1.5 at Re=100

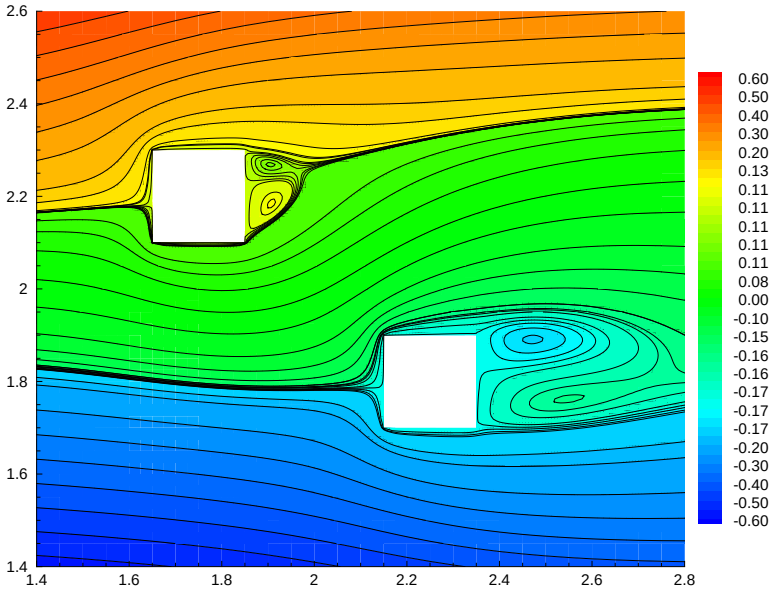


(a) Stream function

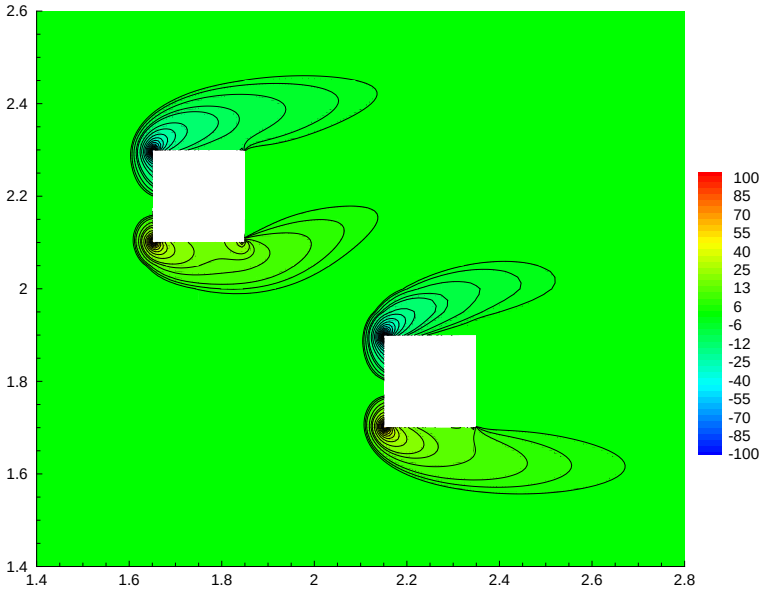


(b) Vorticity

Figure 3.48. Stream function and vorticity values for two bodies for X/L=2 at Re=100



(a) Stream function



(b) Vorticity

Figure 3.49. Stream function and vorticity values for two bodies for X/L=2.5 at Re=100

Table 3.13. Summary of the computational loads for different longitudinal distances

Longitudinal spacing	X/L=0	X/L=0.5	X/L=1.0	X/L=1.5	X/L=2.0	X/L=2.5
Number of non-linear step	1302	3893	3302	2949	3128	3537
Number of linear step	292105	903033	751336	715555	768782	957376
Number of visit	18	58	57	58	59	61
Aproximate comp. time(sec)	12118	29994	260085	26967	28711	33512

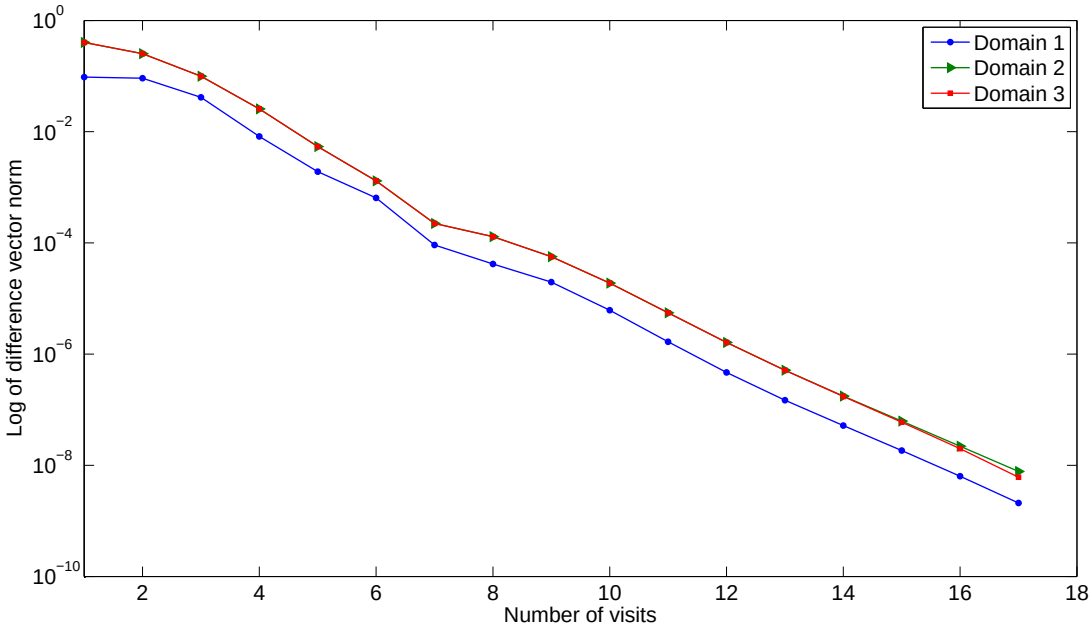
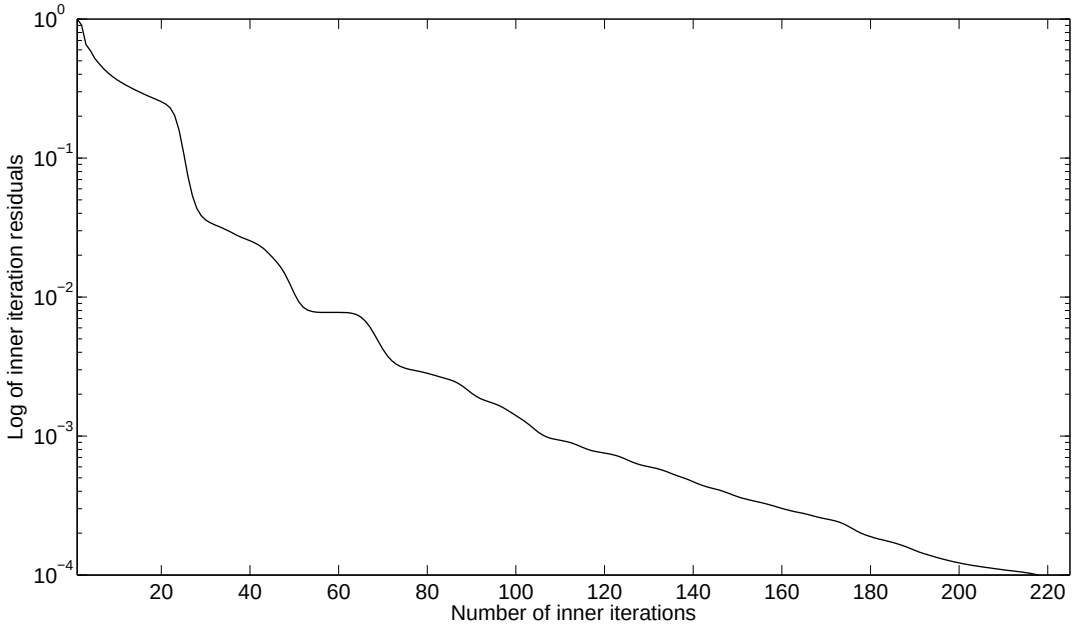
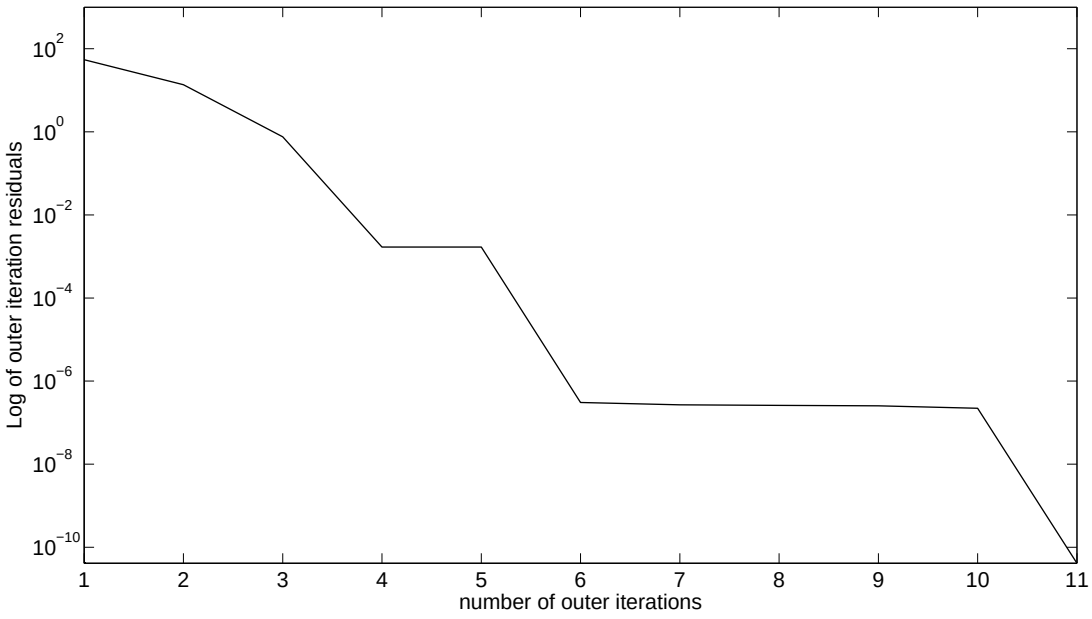


Figure 3.50. Reduction of difference vector norm for the multibody body external flow problem and X/L=0



(a) Inner iterations



(b) Outer iterations

Figure 3.51. Residual norms of the inner and outer iterations for the sample step and X/L=0

4. DISCUSSION AND CONCLUSION

Domain decomposition method is used for simulating two dimensional internal and external incompressible viscous flows. Formulations and numerical techniques are adapted to each subdomain. The subdomains overlap and are coupled by means of multiplicative Schwarz type strategy.

The computational model and techniques are first tested in the analysis of the internal flow problem. However, the main objective of this thesis work is to analyze the aerodynamic coupling observed in the external flow problem

Both of the exact and inexact Newton's methods are applied to linearize the discretized nonlinear PDEs and the updates are evaluated by Krylov solvers. BI-CGSTAB and GMRES(m) are combined with Jacobi, SGS, ILU(l) preconditioners. The performance of Krylov solvers with different preconditioners is compared in Figure 3.3-a and the effect of the fill-in strategy on the performance of the ILU preconditioner is shown in Figure 3.3-b. GMRES and BI-CGSTAB are powerful Krylov solvers especially for the nonsymmetric problems. BI-CGSTAB method can show a divergent behavior even at moderate Reynolds numbers if applied without a preconditioner, which means its performance is too much dependent on the performance of the preconditioner. On the other hand, GMRES is a robust and efficient technique of solving large nonlinear equation systems thanks to its stable convergence character. GMRES leads to the smallest residual for a fixed number of iteration steps but these steps become increasingly expensive. In order to limit the increasing storage and work per iteration step, GMRES(m) is used. GMRES requires only matrix-vector products so memory load ca be decreased when directional differencing is used.

Newton's method has rapid convergence near to exact solution. As the Reynolds number increases, dependence of the convergence speed on the initial guess increases. Inexact Newton is an appropriate method to decrease dependence of the convergence speed to the initial iterate. The forcing term controls the local convergence behavior of

an inexact Newton method. By choosing the forcing term as small as possible, one can obtain local convergence speed that is similar to convergence of the Newton method. But choosing a small forcing term at each linear step, especially when the iterate is far from the solution, cause to over-solve the problem. Inexact Newton method removes this oversolving problem. Backtracking, also known as damping technique shortens Newton step size as necessary to ensure an adequate decrease in the residual of the nonlinear system. In Tables 3.1-3.6, sample outer iterations are shown by using different combinations of the exact/inexact Newton methods with/without damping. All of the results are visualized in the Figure 3.5. The norm of the nonlinear residual vector decreases at each step for the algorithms with backtracking. Whereas the solution procedure without damping may diverge or have an undesired increase in the nonlinear residual vector norm. Exact Newton method has less outer iterations but in that method, the number of linear iterations per nonlinear step is higher. When the results of the Figure 3.5-a and 3.5-b are compared, we can conclude that the best combination is inexact Newton method with damping.

Mesh refinement strategy is an effective way to decrease the computational load. In Figure 3.10, the solution of the problem without mesh refinement around the critical region of the domain. In this figure the grid size is uniform and given as $h = 1/64$ for all domains. We compare this figure with the Figure 3.7, whose mesh size gradually increase from $h = 1/16$ to $h = 1/64$ near the re-entrant corner by local mesh refinement. For $h = 1/64$, i.e. on the uniform grid with about 8500 grid points, the stream function value near the re-entrant corner is about 0.025. On an appropriate locally refined grid, however, it is possible to obtain a similar accuracy with less than 600 grid points and therefore with significantly less computational work. This comparison gives an impression of the potential of local refinement.

The main objective of this thesis work is to analyze the aerodynamic coupling observed in external flow problems by using domain decomposition technique. The overall computational domain, which has a complex geometry, is divided into the simple subdomains. Each subdomain is solved sequentially so it requires enough memory to solve the largest grid and thus the problem can be handled with less memory re-

quirement. Furthermore, the possibility of using different schemes or different sets of equations and time steps for each grid also exist. For example, the Navier-Stokes equations can be solved on the finer inner grid subdomains and Euler equations can be solved on the coarser outer grid subdomains or primitive variables can be used on the inner grid to calculate pressure directly around the objects whereas stream function vorticity approach can be used for the outer grid.

In the external flow problems, matrix free algorithm is used for the storage scheme. Diagonal scaling is used as preconditioner to improve the convergence speed. In order to decrease the computational time further, the original stream function vorticity formulation is changed to equations 3.25 and 3.26. The improvement can be easily seen when we compare Tables 3.9 and 3.10. Computing time savage is about a half for the tabulated sample nonlinear steps.

Besides the angle-of-attack, the aspect ratio of the body sizes affects the flow characteristics. If the width of the body is not too large with respect to its length and the angle-of-attack is small, the adverse pressure gradient is not significant. As a result, the boundary layer fluid can flow in to the slightly increasing pressure region without separating from the surface. However if the width of the body is too large with respect to its length or the angle-of-attack is large, the pressure gradient is too adverse so the boundary layer will separate from the surface. Separation from the surface creates a large drag due to the low pressure in the wake. Although the boundary layer may be quite thin, it can appreciably alter the entire flow field because of boundary separation. It is easier to force a streamlined body through a fluid than it is to force a similar sized blunt body at the same velocity.

Reynolds number represents the ratio of inertial effects to viscous effects. The character of the flow changes greatly when Reynolds number changes from $Re>>1$ to $Re<<1$. This is because, when the Reynolds number is less than unity, the viscous effects are dominant and when the Reynolds number is greater than unity inertial effects are dominant. If the Reynolds number is small, the viscous effects are relatively strong and solid body affects the uniform upstream flow. The viscous effects are felt

far from the objects in all directions. As the Reynolds number increased, except for the downstream, the region in which viscous effects are dominant becomes smaller in all other directions. The displacement of the stream lines from their original uniform upstream case is not great as for the low Reynolds number case.

If the Reynolds number is large, the flow is dominated by inertial effects and viscous effects can be neglected except in a region very close to the solid wall and in relatively thin wake region. As in this thin boundary layer, there is a sharp velocity change. The fluid velocity on the object is zero since fluid must stick to the solid surface (no-slip boundary condition) and it reaches upstream velocity on the boundary layer. The thickness of this boundary layer increases in the direction of the flow. The streamlines outside the boundary layer are nearly parallel to the solid body, which means the effects of the body on streamlines outside of the boundary layer either a head, below, or above is very limited. In the wake region, the main effect is the viscous interactions between the fluid and solid body. If the Reynolds number is large for a blunt object, the region in which viscous effects are important will be small so flow separation will occurs.

For Low Reynolds number flow, viscous effects are felt throughout a large portion of the flow field. The streamline pattern is the same in front of the object as it is behind the object. As the Reynolds number increased, the region ahead of the object in which viscous effects importance becomes smaller. The viscous effects are convected to the wake region so flow loses its symmetry and flow may separate from the body at the separation point. In this point, the fluid's inertia is such that it can not follow the curved path around to the rear of the body. This results in inverse flow that means some of the fluid is actually flowing against the upstream velocity. A fluid particle, which travels from the front of the object to the top of the object, experiences the same pressure distribution as the particles in the free stream immediately outside the boundary layer. However the particle experiences a loss of energy as it flows due to viscous effects. If this loss is large, particle will not have enough energy to coast up the pressure hill to reach the rear side of the body. In other words, due to viscous effects, the particle does not have enough momentum to allow it to coast up to pressure hill.

The effects of the passing manoeuvres are investigated for the two simple rectangular body shape. As the transverse spacing between the bodies diminishes, the flow character considerably changes. The inner sides of the objects is effected by the passing manoeuvres while the outer sides remain little perturbed. Vorticity is generated in the vicinity of solid boundaries and concentrate further downstream in a wake, the outer flow is almost curl-free. Hence we used the adaptive mesh refinement strategy in the boundary layer and the wake.

Simulating a 3D "real-life" geometries of vehicles instead of 2D rectangular shapes can be considered as a next step of this thesis work. The number of unknowns will increase when the flow problem is extended from the 2D simple geometry to a 3D complex geometry. The low memory requirement of the matrix-free scheme enables the solution of the 3D problem with ease.

The domain decomposition method which is studied in this work will be more useful for the investigation of two vehicles passing manoeuvres. The fluid domain will be decomposed in to a set of subdomains. In the small subdomains that surround the solid boundaries, finite element method will be used to obtain a body-fitted grid system whereas in the large subdomain, the standard finite difference technique can be still used. It will be better to use a bi-cubic interpolation and restriction routine instead of a bi-linear routine to transform information between the subdomains. In order to measure the drag force coefficients as well as the static pressure distribution, the Navier-Stokes equations in the primitive form will be solved for the small subdomains. In the large subdomain, stream function and vorticity formulation will be used in order to decrease the number of unknowns.

The most important advantages of the domain decomposition method is the straight-forward applicability for the parallel computing. Since, a 3D problem will be highly nonlinear, the speed of convergence can be further improved by using ASPIN [19] in the solution algorithm with domain decomposition method.

REFERENCES

1. Barrett, R., M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine and H.Van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition*, SIAM, Philadelphia, PA, 1994.

2. Blazek, J., *Computational Fluid Dynamics:Principles and Applications*, Elsevier Science Ltd., UK, 2004

3. Cai, X., "Overlapping Domain Decomposition Methods" *Lecture Notes*, Department of Informatics, University of Oslo, 2004.

4. Cai, X. and Y. Saad, "Overlapping Domain Decomposition Algorithms for General Sparse Matrices" *Numerical Linear Algebra with Applications*, Vol. 3(3), pp. 221-237, 1996

5. Chan, T. F. and T. P. Mathew, "Domain Decomposition Algorithms." *Acta Numerica*, pp. 61-143, 1994

6. Chattot, J.J. and Y. Wang, "Improved treatment of the intersecting bodies with the chimera method and validation with a simple and fast solver" *Computers and Fluids*, Vol. 27, pp. 721-740, 1998

7. Chen, K. and K. Shanazari, "An Overlapping Domain Decomposition Dual Reciprocity Method." *Engineering Analysis with Boundary Elements*, Vol. 27, pp. 945-953, 2003

8. Chorin, A.J., "Numerical study of slightly viscous flow" *Journal of Fluid Mechanics*, Vol. 57, pp. 785-796, 1973

9. Dembo, R.S., S.C. Eisenstat and T. Steihaug, "Inexact Newton Methods" *Siam J.*

Numer. Anal., Vol. 19, pp. 400-408, 1982

10. Eisenstat, S.C. and H.F. Walker, "Globally convergent inexact Newton methods" *Siam J. Optim.*, Vol. 4, pp. 393-422, 1994

11. Eisenstat, S.C. and H.F. Walker, "Choosing the forcing terms in an inexact Newton method" *Siam J. Sci. Comput.*, Vol. 17, pp. 16-32, 1996

12. Gresho, P., "In compressible fluid dynamics: some fundamental formulation issues" *Annual Rev. Fluid Mech.*, Vol. 23, pp. 423-453, 1991

13. Guermod, J. L., S. Huberson and W.Z. Shen, "Simulation od 2D external viscous flows by means of a domain decomposition method" *Journal of Computational Physics*, Vol. 108, pp. 343-352, 1993

14. Guermod, J. L., and H. Z. Lu, "A domain decomposition method for simulating advection dominated, external incompressible viscous flows" *NComputers and Fluids*, Vol. 29, pp. 525-546, 1999

15. Hoffman, K.A., *Computational Fluid Dynamics for Enginners - Volume I*, Engineering Education System, Kansas, 1997.

16. Kaspar, W. and R. Remke, "Die numerische behandlung der Poisson-Gleichung auf einem gebiet mit einspringenden ecken" *Computing*, Vol. 22, pp. 141-151, 1979

17. Kelley, C.T., *Iterative Methods for Solving Linear and Nonlinear Equations*, Volume 16 in Frontiers in Applied Mathematics, Siam., Philadelphia, 1995

18. Kelley, C.T., *Solving Nonlinear Equations with Newton's Method*, Siam., Philadelphia, 2003

19. Keyes, D.E. and X.C. Cai, "Nonlinearly Preconditioned Inexact Newton Algortihm" *Siam J. SCI. Comput.*, Vol. 24, pp. 183-200, 2002

20. Lios, P.L., "Interpretation Stochastiwue de la Methode Altenee de Schwarz" *C. R. Acad. Sci*, Vol. 268, pp. 325-328, 1978

21. Lios, P.L., "On the Schwarz Alternating Method" *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pp. 1-42, SIAM, Philadelphia, 1988

22. Liu, J.G. and C. Wang, "High order finite difference methods for unsteady incompressible flows in multi-connected domains" *Computers and Fluids*, Vol. 33, pp. 223-255, 2004

23. Mızukamı, A., "A stream function-vorticity finite element formulation for Navier-Staokes equations in multi-connected domain" *International Journal for Numerical Methods for the Engineering*, Vol. 19, pp. 1403-1420, 1983

24. Noger, C., C. Regardin and E. Széchényi , "Investigation of the transient aerodynamic phenomena associated with passing manoeuvres" *Journal of Fluids and Structures*, Vol. 21, pp. 231-241, 2005

25. Roache, P.J., *Fundamentals of Computational Fluid Dynamics*, Hermosa Publishers, USA, 1998

26. Russell, D. and Z. J. Wang, "A cartesian grid method for modeling multiple moving objects in 2D incompressible viscous flow" *Journal of Computational Physics*, Vol. 191, pp. 177-205, 2003

27. Sohankar, A., C. Norberg and L. Davidson, "Low-Reynolds-number flow around a square cylinder at incidence: study of blockage, onset of vortex shedding and outlet boundary condition" *Int. J. Numer. Methods Fluids*, Vol. 26, pp. 39-56, 1998

28. Quarteroni, A. and A. Valli, *Domain decomposition methods for partial differential equations*, Oxford Universty Press, New York, 1999.

29. Saad, Y., *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company,

Boston, 1996.

30. Shadid, J.N., R. S. Tuminaro and H. F. Walker, "An inexact Newton method for fully coupled solution of the Navier-Stokes equations with heat and mass transport" *Joutnal of Comput. Physics*, Vol. 137, pp. 155-185, 1997

31. Smith, B., P. Bjorstad and W. Gropp, *Domain Decomposition:Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press,1996

32. Tezduyar, T.E., "Finite element formulation for the voticity-stream function form of the incompressible Euler equations on multiply-connected domains" *Computer Methods in Applied Mechanics and Engineering*, Vol. 73, pp. 331-339, 1989

33. Trottenberg, U., C. W. Oosterlee and A. Schüller, *Multigrid*, Academic Press, London, 2001

34. Yosibash, Z., M. Arad, A. Yakhot and G. Ben-Dor, "An accurate semi-analytic finite difference scheme for two dimentional eliptic problems with singularities" *Num. Meth. P.D.Es*, Vol. 14, pp. 281-296, 1998

35. Zabat, M., N. Stabile, S. Frascaroli and F. Browand, "The aerodynamic performance of platoons" *PATH report*,No UCB-ITS-PRR-95-35, 1995

36. Zabat, M., S. Frascaroli and F. Browand, "Drag measurements of a platoon of vehicles" *PATH(Partners for Advanced Transient and Highway) report*,No UCB-ITS-PRR-93-27, 1994

37. Zenger, C. and H. Gietl, "Improved difference schemes for the Dirichlet problems of the Poisson equation in the neighnorhood of corner" *Numerische Mathematik*, Vol. 30, pp. 315-332, 1978