AERODYNAMIC ANALYSIS OF A PARAGLIDER WING USING DOMAIN DECOMPOSITION TECHNIQUES

by

Altuğ Melik Başol

B.S. in Ch.E., Boğaziçi University, 2004

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

2007

ACKNOWLEDGEMENTS

Firstly I would like to express my gratitude to my advisor Assist. Prof. Ali Ecder for giving the opportunity of working with him, for his guidance and continuous support throughout my study.

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

I also thank Hatice Mercan, FMS laboratory members Dr. Mehmet Orhan, Hüseyin Saygın, Özkan Aydın and Bülent Düz. Besides, I would especially like to thank Erhan Turan and Yalın Kaptan, also members of the FMS laboratory, for their help and guidance at the critical stages of this study.

Finally, I would like to thank my family for their continuous support and encouragement during my entire education.

ABSTRACT

AERODYNAMIC ANALYSIS OF A PARAGLIDER WING USING DOMAIN DECOMPOSITION TECHNIQUES

The flow around a curvature tube is investigated using it as a simple 2D model of a paraglider wing based on the similarity between their top views. The solutions are obtained for five different angle of attacks varying from 0^0 up to 42^0, for two different radius of curvatures and for two different Reynolds number namely for Re=80 and for Re=160.

Two non-matching overlapping domains are used. The flow at these domains is solved separately and the solutions are transferred into each other using the alternating multiplicative Schwarz technique. With the aid of this technique a solver is developed which is capable of solving the flow around various forms of the tube and at different angle of attacks without making any modifications in the outer boundaries and without using grid generation.

Newton's methods combined with three different Krylov sub-space solvers are applied. Implementing also Jacobi, symmetric Gauss-Seidel (SGS) and incomplete LU decomposition (ILU(0)) preconditioners into the solvers their effects on the convergence behavior are investigated. It is observed that the ILU(0) has a superior effect on the convergence behavior than the others have. However, its unavailability for the matrix free algorithms makes SGS preconditioned inexact Newton's method a better option as a solver because of its low storage load and low code development period.

Keywords: Newton's method, Krylov solvers, overlapping domain, preconditioners

ÖZET

YAMAÇ PARAŞÜTÜ KANADININ ALAN AYRIŞTIRMA TEKNİKLERİ KULLANILARAK AERODİNAMİK ANALİZİ

Yamaç paraşütüyle eğik tüpün üstten görünüşlerindeki benzerliğe dayanarak basit iki boyutlu yamaç paraşütü modeli olarak eğik tüp üzerindeki akış incelenmiştir. Sonuçlar 0^0 dan 42^0 ye kadar beş farklı hücum aşısı, iki değişik yarıçap ve iki farklı Reynolds sayısı için elde edilmiştir.

İki tane üst üste binen çözüm alanı kullanılmıştır. Bu alanlardaki akış birbirlerinden bağımsız olarak çözülmüş ve sonuçları birbirlerine değişken çarpımsal Schwarz tekniği kullanılarak aktarılmıştır. Bu tekniğin yardımıyla dış sınır koşullarını değiştirmeden ve yeni ağ yaratmadan değişik formlardaki tüplerin üzerindeki akış farklı hücum açılarında incelenebilmektedir.

Newton yöntemi üç farklı Krylov çözücüsüyle birleştirilerek kullanılmıştır. Ayrıca Jacobi, simetrik Gauss-Seidel (SGS) ve tamamlanmamış matris ayrıştırma (ILU(0)) iyileştiricileri kullanılarak bunların Krylov yöntemlerininin çözüme ulaşma performansları üzerindeki etkileri incelenmişir. ILU(0) iyileştiricisinin diğerlerine göre daha etkili olduğu gözlenmiştir. Ancak bu iyileştiricinin matrissiz methodlara uygulanamamasından SGS iyileştiricili yaklaşık Newton yöntemi hem düşük hafıza kullanımıyla hem de kod yazım süresinin kısalığıyla daha uygun bir çözücü olarak gözükmektedir.

Anahtar kelimeler: Newton yöntemi, Krylov yöntemleri, üst üste binen çözüm alanları, iyileştiriciler

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

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 residual
I	Identity matrix
$J(x)$	Jacobian matrix
k	Iteration count
M	Preconditioner matrix
M_{left}	Left preconditioning matrix
M_{right}	Right preconditioning matrix
M^{-1}	Inverse of the preconditioner matrix
M_{in}	The length of the solid in the first cartesian domain in y-direction given in terms of number of grids
M_{in2}	The length of the inner boundary in the second cartesian domain in y-direction given in terms of number of grids
M_{out}	The length of the first cartesian domain in y-direction given in terms of number of grids
M_{out2}	The length of the second cartesian domain in y-direction given in terms of number of grids
N_{in}	The length of the solid in the first cartesian domain in x-direction given in terms of number of grids
N_{in2}	The length of the inner boundary in the second cartesian domain in x-direction given in terms of number of grids
N_{out}	The length of the first cartesian domain in x-direction given in terms of number of grids
N_{out2}	The length of the second cartesian domain in x-direction given in terms of number of grids

N_{r-in}	The length of the curvature tube in the cylindrical domain in r-direction given in terms of number of grids
N_{r-out}	The length of the cylindrical domain in r-direction given in terms of number of grids
$N_{\theta-in}$	The length of the curvature tube in the cylindrical domain in θ-direction given in terms of number of grids
$N_{\theta-out}$	The length of the cylindrical domain in θ direction given in terms of number of grids
Re	Reynolds Number
r_{mean}	Radius of the mean arc passing through the span of the tube
r_{in}	Radius of the inner arc of the cylindrical computational domain
s	Bratu parameter
$Space_x$	The difference between the x coordinate of the left boundary of the object in the first cartesian domain and the x coordinate of the inlet boundary
$Space_y$	The difference between the y coordinate of the bottom boundary of the object in the first cartesian domain and the y coordinate of the bottom boundary
$Space_{x2}$	The difference between the x coordinate of the left boundary of the rectangle in the second cartesian domain and the x coordinate of the inlet boundary
$Space_{y2}$	The difference between the y coordinate of the bottom boundary of the rectangle in the second cartesian domain and the y coordinate of the bottom boundary
$Space_r$	The difference between the r coordinate of the bottom boundary of the object in the cylindrical domain and the r coordinate of the bottom boundary of the cylindrical computational domain
$Space_\theta$	The difference between the θ coordinate of the right boundary of the object in the cylindrical domain and the θ coordinate of the right boundary of the cylindrical computational domain
x	Unknown vector
x_0	Initial guess for unknown vector

U_0	Inlet velocity
u	Velocity vector in the x-direction
v	Velocity vector in the y-direction
α	Angle of attack
Γ_i	Boundaries of the sub-domains
Δ_x	Grid spacing in the x-direction in the cartesian domain
Δ_y	Grid spacing in the y-direction in the cartesian domain
Δ_r	Grid spacing in the r-direction in the cylindrical domain
Δ_θ	Grid spacing in the θ-direction in the cylindrical domain
μ	Dynamic viscosity
ν	Kinematic viscosity
ρ	Fluid density
ϕ	Unknown vector for Bratu equation
Ψ	Stream function
Ω	Vorticity
DDM	Domain Decomposition Method
PDE	Partial Differential Equation
CGS	Conjugate Gradient Squared
BiCGSTAB	Bi-conjugate Gradient Stabilized
GMRES	Generalized Minimal Residual
SGS	Symmetric Gauss-Seidel
ILU	Incomplete LU decomposition

1. INTRODUCTION

Today paragliding is one of the most widespread aviation sports all over the world. Only in Europe there is an estimated 400,000 pilots. This sport began to emerge at the beginning of the 1980's. The first paragliders had a very poor performance and were potentially more dangerous. They had a gliding speed around 20kph and a glide ratio around three. Since then the performance and the safety of the gliders are being improved continuously. Today the gliders in the competition class have a glide ratio around ten and a cruising speed of 40kph.

Paragliders are composed of the wing called canopy, the suspension lines and the harness. Beside the wing also the harness and the line geometry affect the performance and the stability of the aircraft. But the wing has the dominant effect on the overall performance of the glider. Paraglider wings basically have similar forms with the aircraft wings. The main difference is that they do not have any rigid structure at all. They are made of a special cloth which takes the form of a wing by the pressure of the air entering into the canopy through the openings at the leading edge. This structure brings it a weight advantage (around 7 kg) over the conventional aircrafts with the rigid structures and makes it easily transportable and cheap. Due to these advantages paragliders become widespread in a short time as super lightweight aircrafts. The wing is composed of two layers of cloth and of cell walls which attach these. Cell walls give the wing rigidity, helps it to preserve its shape better. So the wing is divided into similar compartments called cells. At the leading edge these compartments are open two allow air flow in and at the rear edge they are closed to prevent air from escaping. Taking the cross section of the wing we will see an airfoil shape. Moving the airfoil at the allowed angle of attack range through the air it creates lift and drag which counterbalance the gravity force.

In order to survive in the growing paraglider market companies should produce better performing and more reliable gliders. This competition in the market leads the designer to use the most recent technologies in order to build better performing

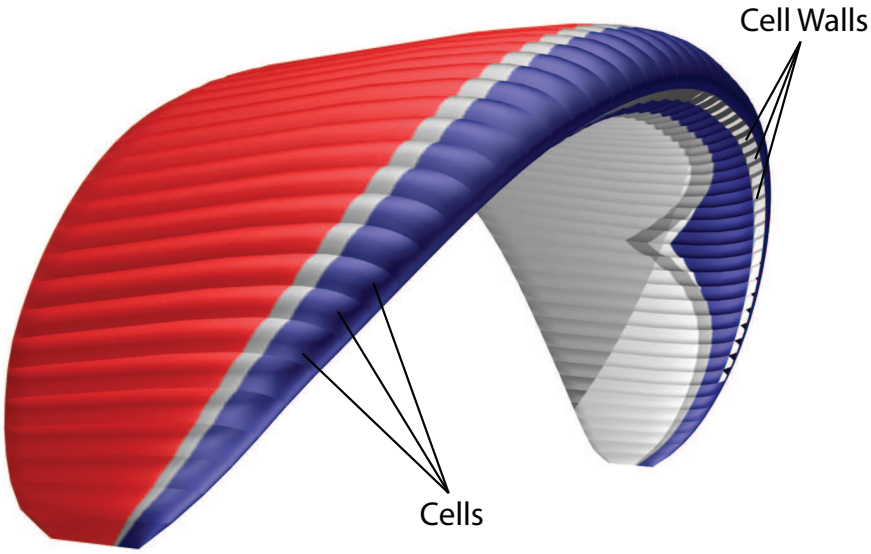


Figure 1.1. Paraglider wing [2]

gliders, to lower the costs in the development process and also to make them less time consuming. Formerly, the design of the wing was mainly based on the try and error method. According to this procedure first the designer decides on his new creation's planform and aerodynamic profile. Making the 3D model of the wing on the computer and constructing it the design procedure continues with the test flight carried out by the professional test pilots. Improvements on the design are made based on the feedback of the pilots. This process was clearly a time consuming and costly method of design. Therefore producers started to shift from the conventional design methods and began to benefit from CFD. Nowadays commercial flow modeling programs are started to be used by the companies. CFD specialists are begun to be incorporated to the research and development teams.

On the other hand, in designing a paraglider a designer should optimize the wing with respect to many parameters. The paragliders should not only perform well but also they have to be safe to fly also in demanding air conditions. Due to the non-rigid structure of the paraglider wings they can collapse if subjected to turbulent conditions. The characteristics of the gliders suffering from a collapse can only be monitored and evaluated by the test pilots. The present CFD techniques are far from modeling of these behaviors of the wings and incorporating the collapsed behavior of the gliders

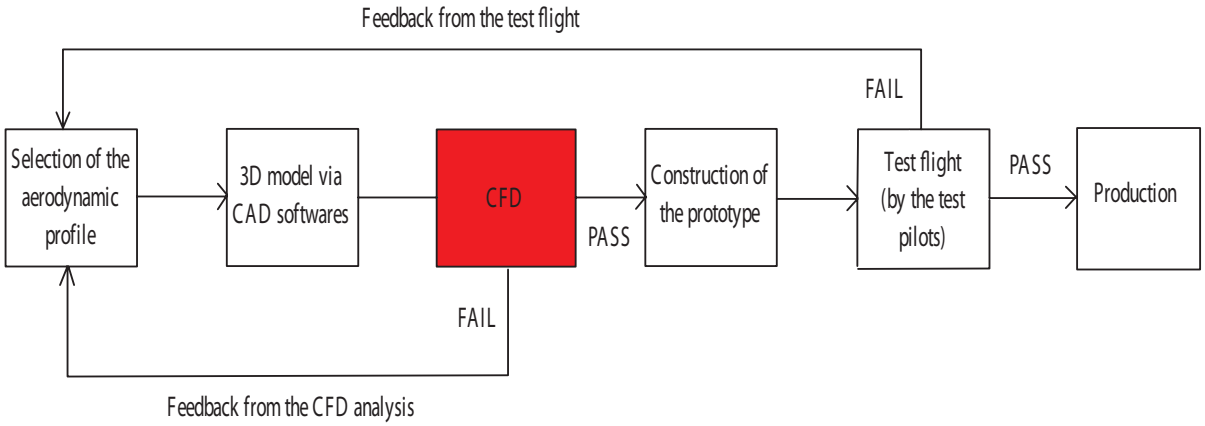


Figure 1.2. Design procedure

will be another milestone in the design process [10]. Paragliders have a very diverse usage. Beside their popular usage as a sport equipment their modified versions are also used in the landing of the space shuttles to the earth. Because of their various usage paragliders are extensively studied in the literature. Although most of these studies have an experimental character recently with the advent in the computer technology CFD also began to enter into this field.

A related study has been done by Babinsky [1] on the performance of the paragliders. Building a two dimensional model of the paraglider wing the drag and the lift coefficient of the model is measured in wind tunnel. In this study a number of modifications to a flexible paraglider section is investigated with the aim of reducing the detrimental effects. The devices studied are: vortex generators; leading edge stiffeners and trailing edge slots. According to the measurements it is shown that these modifications are capable of increasing the maximum lift coefficient of the wing as well as giving up to 25 percent improvement in the glide ratio.

In this thesis a computational study of the flow around a simple model of the paraglider wing is carried out. Curvature tube is used as a model in this work. It is a 2D regular shape. Because of its regularity it can be embedded into a cylindrical domain without generating grids around it. In this study the flow around the object is analyzed using the domain decomposition method. Using the simplicity it brings to the problem the flow at various angle of attacks are solved with ease. The effect of

the physical parameters such as angle of attack or Reynolds number are investigated. Moreover, a great importance is also given to the convergence behavior of the numerical methods used in the thesis. Memory efficient and flexible algorithms are developed and their performances are compared both by means of iteration number and computation time.

2. NUMERICAL METHODS

The governing equations of fluid motion are non-linear differential equations and generally do not have an analytical solution except some special cases such as Couette or Poiseuille flow in which some simplifications due to the geometry and boundary conditions can be done so that they can be solved analytically. However, for the more general cases application of the numerical methods are needed. CFD is the field of fluid mechanics which deals with these techniques in order to obtain approximate but sufficiently accurate results to all kinds of flow problems. CFD is a very broad field of research therefore only a brief introduction of the key concepts and the methods used in this work will be presented in this chapter. Detailed information about the concepts mentioned here can be found in the materials written in the reference section.

2.1. Finite Difference Discretization

Since the governing equations are not analytically solvable they have to be transformed into approximate expressions so that they can be solved using numerical solution techniques. In order to convert the derivatives in the equations to algebraic expressions the computational domain has to be divided into nodes and the derivatives have to be redefined with respect to these nodes using one of the discretization techniques. There are mainly three different discretization methods. These are finite difference, finite volume and finite element. Finite difference is the simplest one among the three. In this work finite difference discretization formulations are used. The derivation of the finite difference formulations can be found in [5].

Although the governing differential equations are valid throughout the whole domain the discretized form of these equations are valid only at the respective node. Writing these equations for each node an equation system is obtained which can be solved using iterative methods.

Finite difference has also different schemes and one of them is selected according

to the location of the node in the computational domain. 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 computational and solid boundaries.

Second order spatial derivatives within the computational domain were approximated as follows.

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

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

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

2.2. Newton's Method

The discretization of the non-linear differential equations results in nonlinear system of equations. In order to solve them using linear system solvers they have to be linearized. Newton's method is one of the most widely used methods in this manner.

Newton's algorithm is given below. Here f is a vector composed of the nonlinear equations aroused from the discretization of the governing equations with respect to the nodes in the computational domain and J is the Jacobian matrix composed of the partial derivatives of the equations with respect to all unknowns. And k indicates the number of the Newton step.

1. Guess a solution vector
2. Solve the linearized equation system below for Δx^{k+1}

$$J(x^k)\Delta x^{k+1} = -f(x^k) \tag{2.4}$$

3. $x^{k+1} = x^k + \Delta x^{k+1}$
4. Continue until norm of the f vector drops below the tolerance

$$J_{i,j} = \frac{\partial f_i^k}{\partial x_j^k} \tag{2.5}$$

To solve the linearized system of equations linear system solvers must be used. A broad range of solvers of this kind can be found in the literature. In this work only Krylov sub-space solvers are used. Brief information about them will be given in the next section. All these solvers require Jacobian matrix vector multiplications. And Newton's method can be grouped into two main categories with respect to the computation of this multiplication. In the exact Newton's method the Jacobian matrix is written analytically in the code, stored in the memory and multiplied with the desired vector when necessary. The main drawback of the exact Newton approach is the large memory requirements for storage of the Jacobian matrix. This generally limits the use of such methods to small-scale problems unless a large memory machine is available. Moreover, multiplications with vectors are also time consuming due to the large sizes of the Jacobian matrices. However, preconditioners can be easily applied in this method due to the storage of the Jacobian matrix in the memory.

Second approach is called inexact Newton's method. In this method the matrix vector multiplications are carried out using the f vector and the directional differencing technique without computing the Jacobian matrix analytically and storing it in the memory. This approach reduces the memory requirement considerably and also speeds up the matrix vector multiplications by a great extent. However, preconditioners which require the Jacobian matrix explicitly cannot be applied in this method.

$$Jv = \frac{f(x + \epsilon v) - f(x)}{\epsilon} \tag{2.6}$$

The selection of the value of ϵ has a crucial importance. There are several formula for the computation of ϵ. In the present double-precision computations, an effective choice

for ϵ was found to be

$$\epsilon = \sigma^{1/2}/\|x\| \tag{2.7}$$

when $\|x\| \neq 0$, and if $\|x\| = 0$, the result of the matrix vector product is set identically to zero. Here, σ is taken to be 10^{-14} [7]

There are also other methods for dealing the non-linear equation systems with. Although Newton's method is one of the most widely used methods it has also some drawbacks beside the advantages it has.

Advantages:

- Quadratically convergent from good starting guesses if J is non-singular.
- Exact solution in one iteration for an affine f (exact at each iteration for affine component functions of f)

Disadvantages:

- Not globally convergent for many problems.
- Requires J at each iteration.
- Each iteration requires the solution of a system of linear equations that may be singular or ill conditioned.

2.3. Solution Methods of the Linear Algebraic Equation Systems and Krylov sub-space solvers

Frequently iterative methods are preferred in CFD applications due to the surplus of the arithmetic operations in the direct methods such as Gaussian elimination, which makes them too costly by means of computation time. The term iterative method refers to a wide range of techniques that use successive approximations to obtain more accurate solutions to a linear system at each step. There are two types of iterative

methods. Stationary methods such as Jacobi, Gauss-Seidel, Successive over-relaxation (SOR) are older, simpler to understand and to implement but usually not as effective due to their slow convergence behavior[8].

On the other hand, non-stationary methods are a relatively recent development, their analysis is usually harder to understand but they can be highly effective. The non-stationary methods presented here are based on the idea of sequences of orthogonal vectors. Some of the well-known methods are Conjugate Gradient (CG), Generalized Minimal Residual (GMRES), Bi-Conjugate Gradient (BiCG), Bi-Conjugate Gradient Stabilized (BiCGSTAB), Quasi Minimal Residual (QMR), Conjugate Gradient Squared (CGS) Method. These methods are called Krylov sub-space methods, because they project the original set of equations onto a so 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 lead to divergence. For instance, CGS can be applied to non-symmetric matrices but CG can not.

In this work CGS, BiCGSTAB and GMRES methods are used. These are among the fastest and robust iterative solvers for a wide variety of applications. All of them are applicable to the non-symmetric matrices which is the case in the thesis problem. A brief summary of the properties of these methods is given below [8].

Conjugate Gradient Squared (CGS):

- Applicable to nonsymmetric matrices.
- Converges (diverges) typically about twice as fast as BiCG.
- Convergence behavior is often quite irregular which may lead to a loss of accuracy in the updated residual.
- Computational costs per iteration are similar to BiCG but the method doesn't require the transpose matrix.
- Unlike BiCG the two matrix vector products are not independent so the number of synchronization points in a parallel environment is larger.

Bi-conjugate Gradient Stabilized (BiCGSTAB):

- Applicable to nonsymmetric matrices.
- Computational costs per iteration are similar to BiCG 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 about 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 restarting is necessary. When to do so depends on A and 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.

2.4. Preconditioning

The convergence rate of iterative methods depends on the spectral properties of the coefficient matrices. Preconditioners transform the linear system into one that has the same solution but into which the linear solvers can be more efficiently applied. The need for preconditioners arouses because problems in the fluid mechanics usually give rise to matrices with undesired spectral properties especially at high Reynolds numbers.

Preconditioners increase the amount of operations per iteration. However, the additional time spent to produce the preconditioner matrix and to carry out the additional operations at each iteration is compensated by the reduction in the number of iteration steps. So they provide a decrease in the computation time on overall. Moreover, they can make a solver converge whereas a solver could diverge without the implementation of a preconditioner [8].

Accordingly, a preconditioner must be similar to the coefficient matrix as close as possible and also the preconditioner system $Mx = p$ 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 $Mx = p$ would be equal of solving the real system $Ax = b$. On the other extreme, taking the preconditioner matrix equal to the identity matrix so that the system $Mx = p$ can be easily solved would not result in any reduction in the number of iteration steps. So the ideal preconditioner must be between the two extreme cases.

The preconditioners are divided into three groups according to the application of the preconditioner matrix. These are left, right and split preconditioning.

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

$$M_{left}^{-1} A x = M_{left}^{-1} b \tag{2.8}$$

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 (2.9),

$$A M_{right}^{-1} y = b \quad where \quad x = M_{right}^{-1} y \tag{2.9}$$

The transformed system is first solved for y, and then, the unknown vector x is computed by the relation given in the equation (2.9).

In the left preconditioning the preconditioner is applied directly to the residual vector so it may cause the algorithm stop prematurely or with delay [3]. Because of 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

$$M = LU \tag{2.10}$$

According to this option

$$L^{-1}AU^{-1}y = L^{-1}b \quad where \quad x = U^{-1}y \tag{2.11}$$

In this work three different kinds of preconditioners are used. These are Jacobi, symmetric Gauss-Seidel (SGS) and incomplete LU decomposition (ILU). The preconditioner matrices of each of these methods are given below.

2.4.1. Jacobi

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

$$M = D \tag{2.12}$$

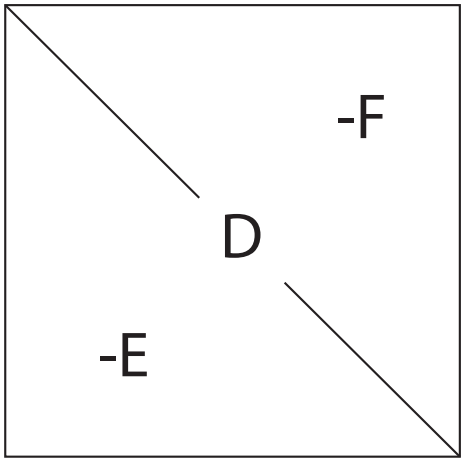


Figure 2.1. Initial partitioning of matrix A [3]

2.4.2. Symmetric Gauss-Seidel(SGS)

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

$$M = (D - E)D^{-1}(D - F) \tag{2.13}$$

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

2.4.3. Incomplete LU decomposition(ILU)

As it can be understood from its name the Jacobian matrix is partially decomposed into upper and lower triangular matrices in this method. Inaccurate factorization improves the spectral properties of the matrix and provides faster convergence. The exact LU factorization would require similar amount of operation like Gaussian elimination and would be therefore an inefficient solution method. Because of 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. In this work only ILU(0) is applied. The zero in the parenthesis indicate that the zero pattern of the decomposed matrices precisely fit the zero pattern of the Jacobian matrix. For ILU(0) the LU factorization is carried only at the non-zero elements of the original matrix.

$$M = LU \tag{2.14}$$

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

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

2.5. Compressed Storage Schemes

The Jacobian matrices arousing from the discretization of the partial differential equations have characteristic sparsity patterns. Instead of storing each element of the matrix storing only the non-zero elements of the matrix using one of the compressed storage schemes decreases the memory load of the computer considerably. Also defining the matrix vector multiplications required for the linear solvers 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 non-zero 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. Another scheme used in this work is the compressed row storage scheme abbreviated as CRS. Adapting the exact Newton method with respect to this scheme an efficient algorithm is created. However, the analytical computation of the Jacobian matrix and entering it with respect to the CRS scheme makes the code development process time consuming.

2.6. Application of the Preconditioners to the Matrix-Free Algorithms

As mentioned above preconditioners can have an important effect on the convergence behavior of the Krylov sub-space solvers especially when the forming matrices are ill-conditioned, which is usually the case for the flow problems at high Reynolds numbers. However, some of the preconditioners require the coefficient matrix explicitly such as ILU. Therefore these type of preconditioners cannot be adapted into a fully matrix free algorithm. On the other hand, preconditioners such as Jacobi or symmetric Gauss-Seidel (SGS) can be used in the matrix-free methods [6]. In order to apply SGS the forward and backward algorithms must be modified so that they can be carried out in a column by column manner instead of row by row. The algorithm of the matrix-free version of the backward substitution is given below.

```
for i=1:n
    v(n+1-i,1)=1
    Av=(f(x+eps*v)-f(x))/eps
    v(n+1-i,1)=0
    x(n+1-i,1)=p(n+1-i,1)/Av(n+1-i,1)
    for j=i:n
        p(n+1-j,1)=p(n+1-j,1)-Av(n+1-j,1)*x(n+1-i,1)
    end
end
```

And the forward substitution algorithm is shown below.

```
for i=1:n
    v(i,1)=1;
    Av=(f(x+eps*v)-f(x))/eps
    v(i,1)=0;
    x(i,1)=p(i,1);
    for j=i:n
```

```
        p(j,1)=p(j,1)-Av(j,1)*x(i,1)/Av(i,1);
    end
end
```

The application of the SGS preconditioner to the matrix free algorithms increases the operation load considerably. However, this load can be decreased combining the matrix free algorithm with the compressed column storage (CCS) scheme. In this method the Jacobian matrix is computed column by column using the directional differencing technique and stored via the CCS storage scheme. For example, using the vector with 1 only at the first entry and zeros at the remaining ones the first column of the Jacobian matrix is obtained. Continuing this process and storing every time the non-zero elements in the columns of the Jacobian matrix using CCS the matrix can be stored in a compressed manner. Using this matrix and adapting backward and forward substitutions for SGS according to the storage scheme the operation load of the algorithm is decreased considerably. This modification in the matrix free algorithm only affects the operations related with the preconditioner. The Jacobian matrix vector multiplications in the solvers are carried out matrix free. Although this modification increases the memory load it is far below than the load the exact Newton method brings.

2.7. Domain Decomposition

Dividing a problem into subproblems and combining the solution of each one in order to build the puzzle is the idea behind the domain decomposition methods. Instead of solving the problem as a whole partitioning it into regions can contribute to simplicity and speed.

The simplicity may depend on the geometrical advantages of the sub-domains or on the difference of the modeling equations that are used in each sub-system. For example restricting the uequations to the proximity of the object and using Euler equation in the other regions could be one of the practical usages of this idea. On the other hand, this method could also enable decrease in computation time using

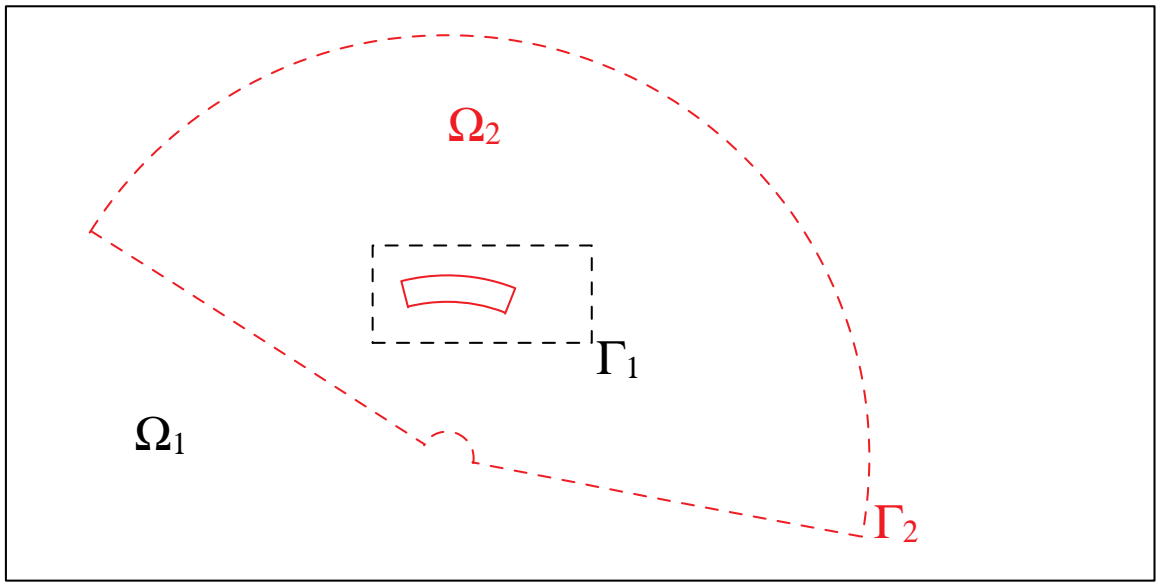


Figure 2.2. Partitioning of the domain in the thesis problem

the advantages of the parallel computing. Using multiple machines each sub-problem can be solved simultaneously and combined afterwards. Although this technique has begun to be more widely used in the recent years the roots of this idea are in the 19th century. In 1870 Schwarz introduced an alternating algorithm which provided a parallel, potentially fast, and robust method for the solution of linear or non-linear systems of equations resulted usually from discretization of PDEs [11]. According to this method a domain Ω is partitioned into the sub-regions $\Omega_i, i = 1, ..., M$. Using this method called Multiplicative Schwarz Procedure the problem is solved on a sub-domain at each iteration and the most recent solution of this domain is used as a boundary condition for the other sub-regions. As shown in the figure 2.2 there are two overlapping domains (cartesian and cylindrical) labeled as Ω_1 and Ω_2 respectively. Γ_1 and Γ_2 show the two artificial boundaries between the domains shown with the dotted lines. These are the boundaries where the solution in one domain is transferred into the other domain.

Domain decomposition is a very wide field of research. Here only a brief summary of the general ideas and the solution algorithm of the thesis problem is mentioned. For more information please refer to [3], [9].

2.8. Convergence Criterion

Setting a criterion for convergence is a crucial aspect of the numerical methods. They give an approximate solution and the accuracy of the solution depends on the criterion set.

There are different ways of deciding for convergence. In this study the two norm of the residual vector is monitored which is a frequently used way of concluding the convergence. There are two types of residual vectors in this thesis. These are the residual of the linear system $J\Delta x = -f$ and the non-linear residual f.

As mentioned above the discretized form of the governing equations have a non-linear character and they are linearized by the Newton's method in order to be solved by the linear solvers. So in each Newton step a linear system has to be solved. At each step the reduction of the two norm of the linear residual vector is monitored with respect to the norm of the initial residual vector.

$$\frac{\|b - Ax^k\|}{\|b - Ax^0\|} < tol \tag{2.15}$$

Moreover, a maximum number iteration steps is also set at each Newton step. If the linear system does not converge in the given number of iteration steps then a different solver is used or the same solver combined with a more efficient preconditioner is tried.

After deciding for convergence at a Newton step and making the necessary corrections in the solution vector the procedure continues with the next Newton step. After each Newton step the two norm of the f vector composed of the non-linear system equations is checked.

$$\|f\| < tol \tag{2.16}$$

In the thesis problem two different domains, cartesian and cylindrical, are used and the solutions of each domain are transferred to the other until the change in the solutions

of the domains drops below the tolerance given. Accordingly, the difference of the successive solution vectors of the cylindrical domain is taken and the two norm of the resulting vector is monitored to decide for convergence.

$$\|x^{k}_{cylindrical} - x^{k-1}_{cylindrical}\| < tol \tag{2.17}$$

3. COMPUTATIONAL MODELING

3.1. Problem Statement

Paraglider wings have a characteristic airfoil cross-section. Therefore the simulation of the flow around the wing requires the use of grid generation techniques. In this work curvature tube is used as a 2D simple model of the wing based on the similarity between the top views of the paraglider wing and the curvature tube. A curvature tube is a 2D shape composed of two arcs which belong to two concentric circles and also cover the same angle $\Delta\theta$. In this shape there are also two lines connecting these arcs at their ends. Using domain decomposition methods the flow around the tube is determined. Defining the physical and the computational parameters their effects on the flow and on the convergence of the iterative methods respectively are investigated.

3.2. Solution Algorithm

Embedding the curvature tube in a cylindrical domain and defining the outer boundary conditions at that domain could be one way of solving this problem. However, there have to be also a second cylindrical grid around the object in order to simulate the flow around the vicinity of the object more accurate. Instead of that, it is preferred to use a rectangular base grid in which it is easier to define the boundary conditions and to overlap a cylindrical smaller domain around the vicinity of the object to model the flow around the solid more precise.

Accordingly, first the flow around a rectangle similar to the curvature tube in size is solved. After that, passing to the overlapped cylindrical domain the flow around the curvature tube is solved taking the outer boundary conditions from the solution of the cartesian system by bi-linear interpolation. However, the values at the outer boundaries of the cylindrical domain do not reflect the effect of the curvature tube because of the difference of the shapes between the tube and the rectangle. Therefore the effect of the curvature tube has to be reflected back to the cartesian domain by taking the inner

Figure 3.1. Paraglider wing-Top view [12]

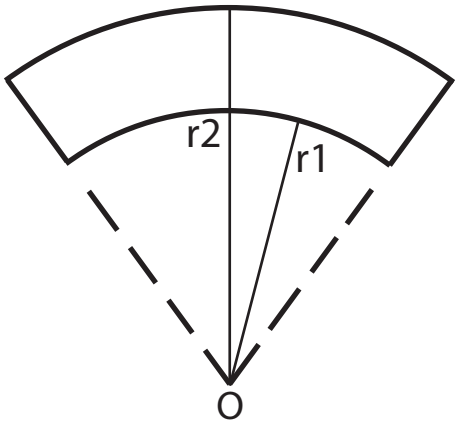


Figure 3.2. Curvature tube

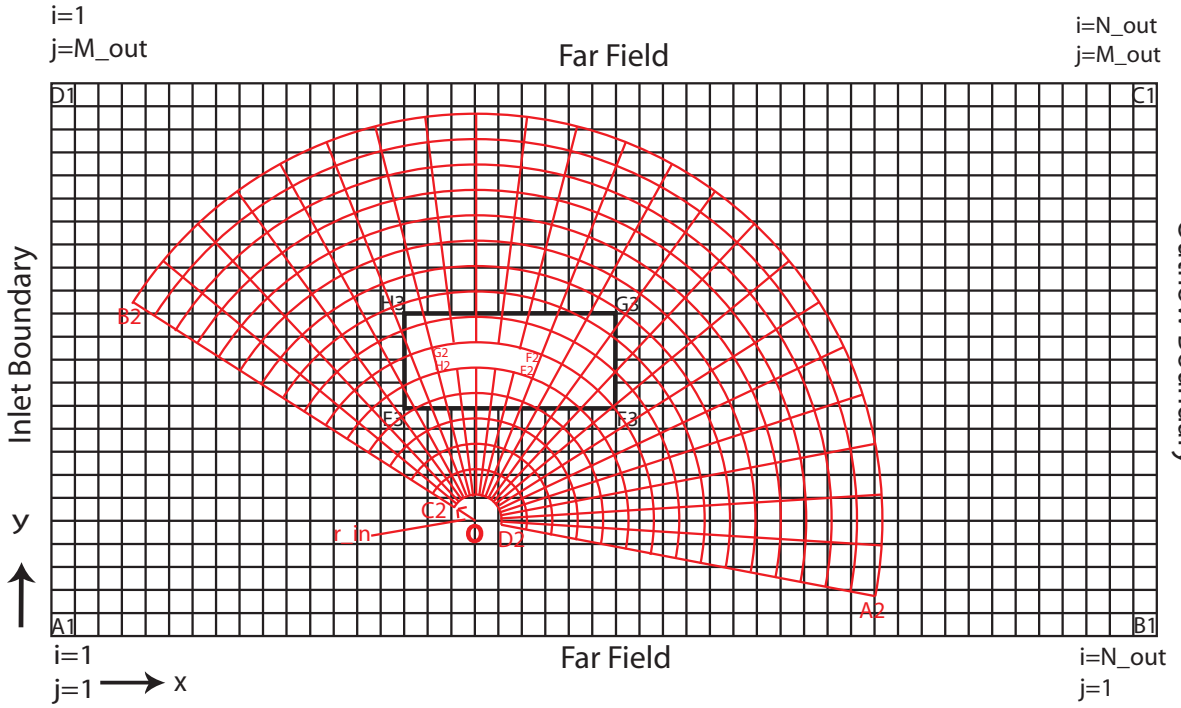


Figure 3.3. Overlapping of the domains

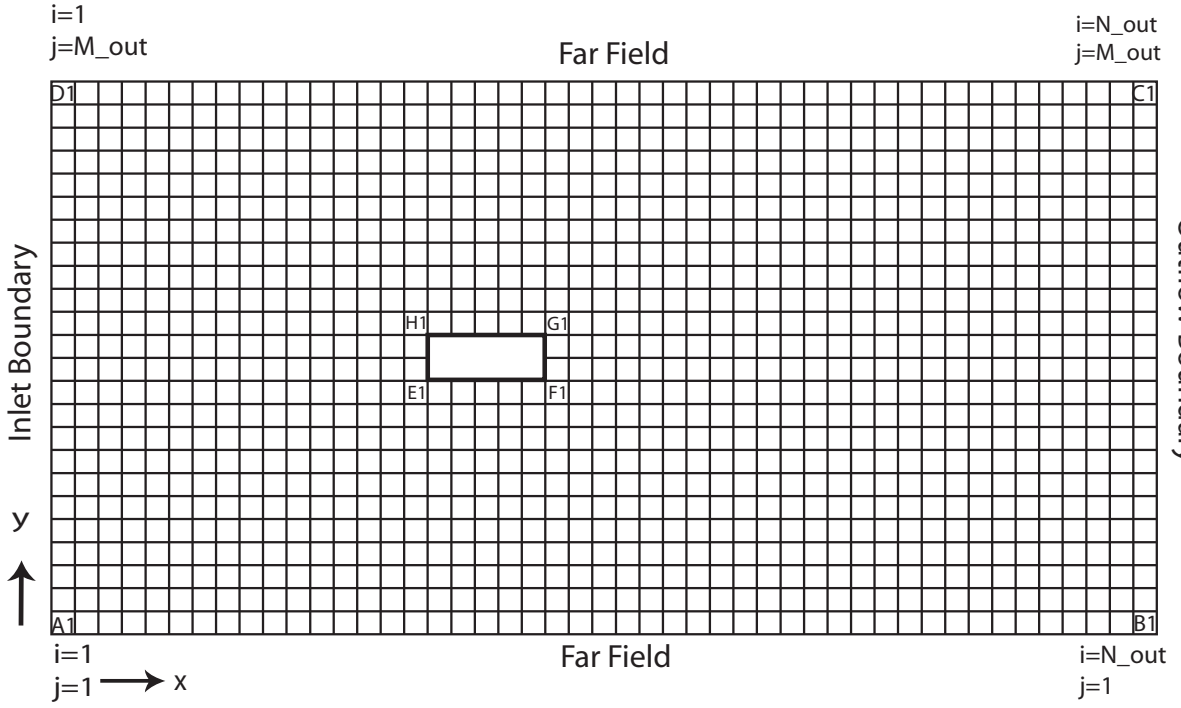


Figure 3.4. First cartesian domain

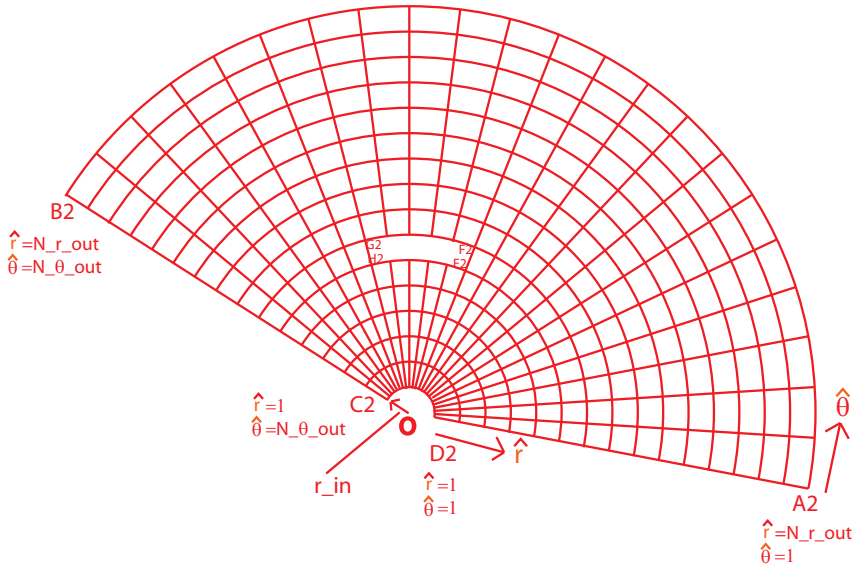


Figure 3.5. Cylindrical domain with the curvature tube

boundary conditions from the solution of the cylindrical system using interpolation. So solving each system separately and transferring each time the boundary conditions to the other domain the flow field around the tube can be determined. To decide for convergence the difference between the adjacent solutions of the cylindrical domain are checked. This difference is in form of a vector therefore it is checked whether the greatest value is below the tolerance or not.

3.3. Interpolation

For the interpolations both from cartesian to cylindrical and from cylindrical to cartesian bi-linear interpolation algorithm is used. Their formulations are given in the equations. 3.1, 3.2, 3.3 ,3.4.

$$\Psi_E = \left(1 - \frac{(x_E - x_A)}{\Delta x}\right)\left(1 - \frac{(y_E - y_A)}{\Delta y}\right)\Psi_A + \left(\frac{(x_E - x_A)}{\Delta x}\right)\left(1 - \frac{(y_E - y_A)}{\Delta y}\right)\Psi_B + \left(\frac{(x_E - x_A)}{\Delta x}\right)\left(\frac{(y_E - y_A)}{\Delta y}\right)\Psi_C + \left(1 - \frac{(x_E - x_A)}{\Delta x}\right)\left(\frac{(y_E - y_A)}{\Delta y}\right)\Psi_D \quad (3.1)$$

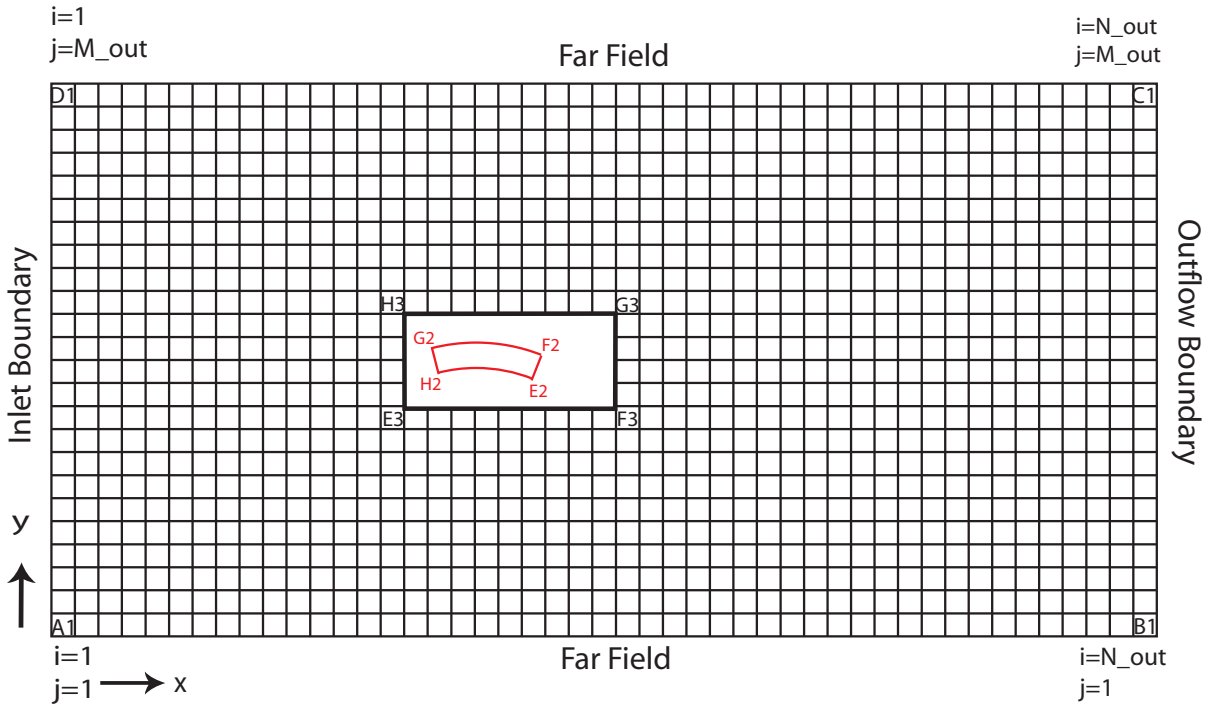


Figure 3.6. Second cartesian domain

$$\Omega_E = \left(1 - \frac{(x_E - x_A)}{\Delta x}\right)\left(1 - \frac{(y_E - y_A)}{\Delta y}\right)\Omega_A + \left(\frac{(x_E - x_A)}{\Delta x}\right)\left(1 - \frac{(y_E - y_A)}{\Delta y}\right)\Omega_B + \left(\frac{(x_E - x_A)}{\Delta x}\right)\left(\frac{(y_E - y_A)}{\Delta y}\right)\Omega_C + \left(1 - \frac{(x_E - x_A)}{\Delta x}\right)\left(\frac{(y_E - y_A)}{\Delta y}\right)\Omega_D \quad (3.2)$$

$$\Psi_E = \left(1 - \frac{(r_E - r_A)}{\Delta r}\right)\left(1 - \frac{(\theta_E - \theta_A)}{\Delta \theta}\right)\Psi_A + \left(\frac{(r_E - r_A)}{\Delta r}\right)\left(1 - \frac{(\theta_E - \theta_A)}{\Delta \theta}\right)\Psi_B + \left(\frac{(r_E - r_A)}{\Delta r}\right)\left(\frac{(\theta_E - \theta_A)}{\Delta \theta}\right)\Psi_C + \left(1 - \frac{(r_E - r_A)}{\Delta r}\right)\left(\frac{(\theta_E - \theta_A)}{\Delta \theta}\right)\Psi_D \quad (3.3)$$

$$\Omega_E = \left(1 - \frac{(r_E - r_A)}{\Delta r}\right)\left(1 - \frac{(\theta_E - \theta_A)}{\Delta \theta}\right)\Omega_A + \left(\frac{(r_E - r_A)}{\Delta r}\right)\left(1 - \frac{(\theta_E - \theta_A)}{\Delta \theta}\right)\Omega_B + \left(\frac{(r_E - r_A)}{\Delta r}\right)\left(\frac{(\theta_E - \theta_A)}{\Delta \theta}\right)\Omega_C + \left(1 - \frac{(r_E - r_A)}{\Delta r}\right)\left(\frac{(\theta_E - \theta_A)}{\Delta \theta}\right)\Omega_D \quad (3.4)$$

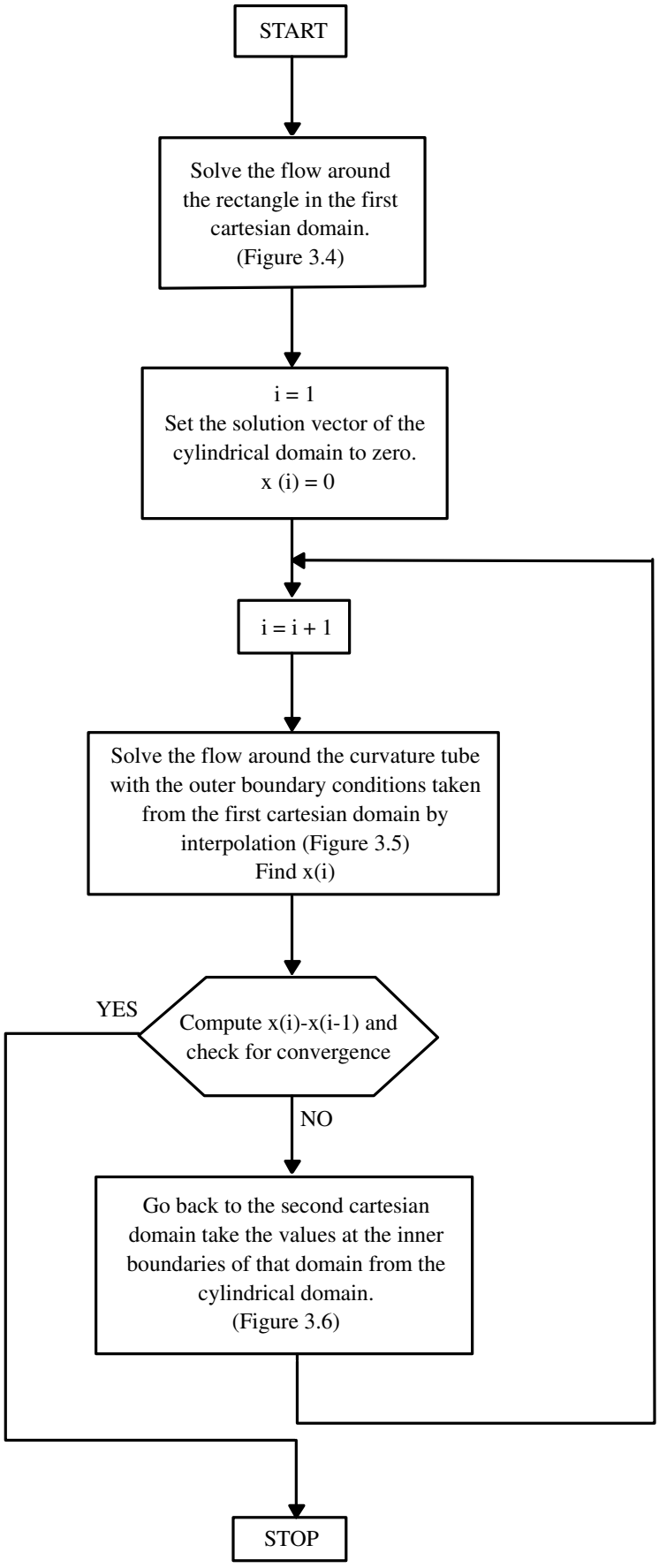


Figure 3.7. Flowchart of the solution algorithm

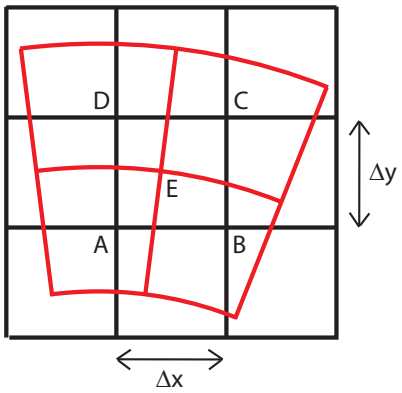


Figure 3.8. Interpolation from cartesian to cylindrical domain

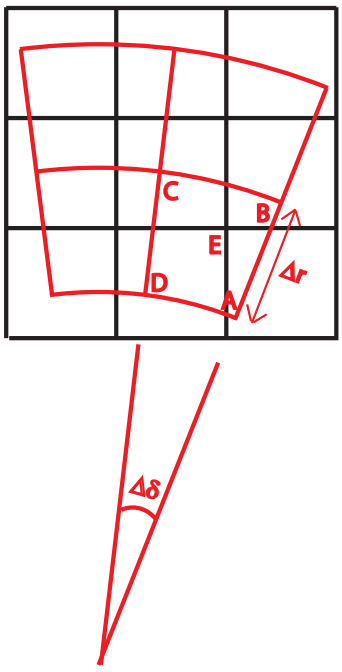


Figure 3.9. Interpolation from cylindrical to cartesian domain

3.4. Discretization of the Governing Equations and Domains

Navier-Stokes equations are the well-known governing equations of the fluid flow. Since the velocities subjected in this problem are far below the compressibility limit the fluid can be assumed as incompressible and incompressible forms of these equations can be used. Moreover time derivatives in the equations can be dropped because steady state solution of the problem is sought.

There are two different formulations of these equations based on the dependent variables used. 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. Since the problem consists of both cartesian and cylindrical domains the equations both for cartesian and cylindrical coordinates are required. The derivation of the formulation for the cylindrical coordinate system can be found in appendix A. The derivation of the equations for the cartesian coordinates can be found in [5].

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

$$\nabla^2 \Psi + \Omega = 0 \tag{3.5}$$

$$\nabla^2 \Omega - Re \left[\frac{\partial \Psi}{\partial y} \frac{\partial \Omega}{\partial x} - \frac{\partial \Psi}{\partial x} \frac{\partial \Omega}{\partial y} \right] = 0 \tag{3.6}$$

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

$$\left[\frac{(\Psi_{i+1,j} - 2\Psi_{i,j} + \Psi_{i-1,j})}{\Delta x^2} + \frac{(\Psi_{i,j+1} - 2\Psi_{i,j} + \Psi_{i,j-1})}{\Delta y^2} \right] + \Omega_{i,j} = 0 \tag{3.7}$$

$$\left[\frac{(\Omega_{i+1,j}-2\Omega_{i,j}+\Omega_{i-1,j})}{\Delta x^2}+\frac{(\Omega_{i,j+1}-2\Omega_{i,j}+\Omega_{i,j-1})}{\Delta y^2}\right]- Re\left[\frac{\Psi_{i,j+1}-\Psi_{i,j-1}}{\Delta y}\frac{\Omega_{i+1,j}-\Omega_{i-1,j}}{\Delta x}-\frac{\Psi_{i+1,j}-\Psi_{i-1,j}}{\Delta x}\frac{\Omega_{i,j+1}-\Omega_{i,j-1}}{\Delta y}\right]=0 \quad (3.8)$$

The stream function and the vorticity transport equation for the cylindrical system are respectively.

$$\frac{\partial^2\Psi}{\partial r^2}+\frac{1}{r}\frac{\partial\Psi}{\partial r}+\frac{1}{r^2}\frac{\partial^2\Psi}{\partial\theta^2}+\Omega=0 \quad (3.9)$$

$$\left[\frac{\partial^2\Omega}{\partial r^2}+\frac{1}{r}\frac{\partial\Omega}{\partial r}+\frac{1}{r^2}\frac{\partial^2\Omega}{\partial\theta^2}\right]-\frac{Re}{r}\left[\frac{\partial\Psi}{\partial\theta}\frac{\partial\Omega}{\partial r}-\frac{\partial\Omega}{\partial\theta}\frac{\partial\Psi}{\partial r}\right]=0 \quad (3.10)$$

Second order central finite difference formulation of the equations in cylindrical coordinates are:

$$\frac{\Psi_{\hat{r}+1,\hat{\theta}}+\Psi_{\hat{r}-1,\hat{\theta}}-2\Psi_{\hat{r},\hat{\theta}}}{(\Delta r)^2}+\frac{1}{r_{in}+(\hat{r}-1)\Delta r}\frac{\Psi_{\hat{r}+1,\hat{\theta}}-\Psi_{\hat{r}-1,\hat{\theta}}}{2\Delta r}+\frac{1}{(r_{in}+(\hat{r}-1)\Delta r)^2}\frac{\Psi_{\hat{r},\hat{\theta}+1}+\Psi_{\hat{r},\hat{\theta}-1}-2\Psi_{\hat{r},\hat{\theta}}}{(\Delta\theta)^2}+\Omega_{\hat{r},\hat{\theta}}=0 \quad (3.11)$$

$$\frac{\Omega_{\hat{\mathrm{r}}+1,\hat{\theta}}+\Omega_{\hat{\mathrm{r}}-1,\hat{\theta}}-2\Omega_{\hat{\mathrm{r}},\hat{\theta}}}{(\Delta r)^2}+\frac{1}{r_{in}+(\hat{\mathrm{r}}-1)\,\Delta r}\frac{\Omega_{\hat{\mathrm{r}}+1,\hat{\theta}}-\Omega_{\hat{\mathrm{r}}-1,\hat{\theta}}}{2\Delta r}+\frac{1}{(r_{in}+(\hat{\mathrm{r}}-1)\,\Delta r)^2}\frac{\Omega_{\hat{\mathrm{r}},\hat{\theta}+1}+\Omega_{\hat{\mathrm{r}},\hat{\theta}-1}-2\Omega_{\hat{\mathrm{r}},\hat{\theta}}}{(\Delta\theta)^2}-\frac{\mathrm{Re}}{r_{in}+(\hat{\mathrm{r}}-1)\,\Delta r}\left[\left(\frac{\Psi_{\hat{\mathrm{r}},\hat{\theta}+1}-\Psi_{\hat{\mathrm{r}},\hat{\theta}-1}}{2\Delta\theta}\right)\left(\frac{\Omega_{\hat{\mathrm{r}}+1,\hat{\theta}}-\Omega_{\hat{\mathrm{r}}-1,\hat{\theta}}}{2\Delta r}\right)-\left(\frac{\Psi_{\hat{\mathrm{r}}+1,\hat{\theta}}-\Psi_{\hat{\mathrm{r}}-1,\hat{\theta}}}{2\Delta r}\right)\left(\frac{\Omega_{\hat{\mathrm{r}},\hat{\theta}+1}-\Omega_{\hat{\mathrm{r}},\hat{\theta}-1}}{2\Delta\theta}\right)\right]=0 \quad (3.12)$$

3.5. The Boundary Conditions of the First Cartesian Domain

The derivation of the boundary conditions in this domain are given below.

3.5.1. Solid Boundaries

Since the object is non-porous and stationary the velocity at the solid wall can only be parallel to it. This means that $v = 0$ along the E1-F1 line shown in figure 3.4. Using the definition of the stream function ($v = -\frac{\partial \Psi}{\partial x} = 0$) it is found that the stream function is constant along that boundary. Applying the same principle to the other three solid boundaries it can be seen that the value of the stream function is constant along the boundaries of the solid and can be chosen arbitrarily.

$$\Psi = 0 \tag{3.13}$$

Boundary conditions for the vorticity does not exist. Therefore they have to be constructed. Derivation of the conditions for the E1-F1 line is given whereas the conditions for the other three boundaries are only specified.

Vorticity boundary conditions are derived using the stream function equation.

$$\frac{\partial^2 \Psi}{\partial x^2} + \frac{\partial^2 \Psi}{\partial y^2} = -\Omega \tag{3.14}$$

Along the E1-F1 line in figure 3.4 the value of the stream function is constant and therefore the equation 3.14 simplifies to this form for this boundary.

$$\frac{\partial^2 \Psi}{\partial y^2}|_{i,j} = -\Omega|_{i,j} \tag{3.15}$$

Writing the second order Taylor series expansion of the equation;

$$\Psi_{i,j-1} = \Psi_{i,j} + \frac{\partial \Psi}{\partial y}|_{i,j}\Delta y + \frac{\partial^2 \Psi}{\partial y^2}|_{i,j}\frac{(\Delta y)^2}{2} + O(\Delta y)^3 \tag{3.16}$$

Applying the no-slip boundary condition for that line by taking $\frac{\partial \Psi}{\partial y} = 0$ the following

equation appears.

$$\Psi_{i,j-1} = \Psi_{i,j} + \frac{\partial^2 \Psi}{\partial y^2}|_{i,j} \frac{(\Delta y)^2}{2} + O(\Delta y)^3 \tag{3.17}$$

Solving the equation for $\frac{\partial^2 \Psi}{\partial y^2}$ and using the equation (3.33) the vorticity equation for that boundary are formed.

The vorticity boundary conditions of the inner boundaries are given with respect to the coordinates of the corners and dimensions of the inner rectangle. The coordinates of the corners of the rectangle are:

$E1(Space_x + 1, Space_y + 1)$; $F1(Space_x + N_{in}, Space_y + 1)$;
$G1(Space_x + N_{in}, Space_y + M_{in})$; $H1(Space_x + 1, Space_y + 1)$

For the E1-F1 boundary,

$$\Omega_{i,j} = \frac{2(\Psi_{i,j} - \Psi_{i,j-1})}{(\Delta y)^2} \tag{3.18}$$

where $Space_x + 1 \leqslant i \leqslant Space_x + N_{in}$ and $j = Space_y + 1$

For the F1-G1 boundary,

$$\Omega_{i,j} = \frac{2(\Psi_{i,j} - \Psi_{i+1,j})}{(\Delta x)^2} \tag{3.19}$$

where $i = Space_x + N_{in}$ $Space_y + 2 \leqslant j \leqslant Space_y + M_{in} - 1$

For the G1-H1 boundary,

$$\Omega_{i,j} = \frac{2(\Psi_{i,j} - \Psi_{i,j+1})}{(\Delta y)^2} \tag{3.20}$$

where $Space_x + 1 \leqslant i \leqslant Space_x + N_{in}$ $j = Space_y + M_{in}$

For the H1-E1 boundary,

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

where $i = Space_x + 1$ $Space_y + 2 \leqslant j \leqslant Space_y + M_{in} - 1$

3.5.2. Computational Domain Boundaries

At the inlet,

$$\frac{\partial \Psi}{\partial y} = u = U_0 \qquad -\frac{\partial \Psi}{\partial x} = v = 0 \tag{3.22}$$

Using first order finite difference equations and specifying the value of the stream function at one of the grid points the values at the other points of the inlet boundary can be found with respect to this point. For instance, the value of the node matching with the stagnation point of the object is taken zero in this problem since the values of the stream function were specified as zero at the object boundaries.

$$\frac{\Psi_{i,j+1} - \Psi_{i,j}}{\Delta y} = U_0 \tag{3.23}$$

where $1 \leqslant j \leqslant M_{out} - 1$ and $i = 1$

$$\Omega_{i,j} = 0 \tag{3.24}$$

where $1 \leqslant j \leqslant M_{out}$ and $i = 1$

At the far field,

$$\Psi_{i,j} = \Psi_{1,1} \tag{3.25}$$

where $j = 1$ and $2 \leqslant i \leqslant N_{out}$

$$\Psi_{i,j} = \Psi_{1,M_{out}} \tag{3.26}$$

where $j = M_{out}$ and $2 \leqslant i \leqslant N_{out}$

$$\Omega_{i,j} = 0 \tag{3.27}$$

where $j = 1$ or $j = M_{out}$ and $2 \leqslant i \leqslant N_{out}$

At the outflow,

$$\frac{\partial \Psi}{\partial x} = 0 \qquad\qquad \frac{\partial \Omega}{\partial x} = 0 \tag{3.28}$$

Applying second order backward differencing,

$$\frac{-3\Psi_{i,j} + 4\Psi_{i-1,j} - \Psi_{i-2,j}}{2\Delta x} = 0 \tag{3.29}$$

$$\frac{-3\Omega_{i,j} + 4\Omega_{i-1,j} - \Omega_{i-2,j}}{2\Delta x} = 0 \tag{3.30}$$

where $i = N_{out}$ and $2 \leqslant j \leqslant M_{out} - 1$

3.6. The Boundary Conditions of the Cylindrical Domain

The derivations of the boundary conditions of this domain are given below.

3.6.1. Solid Boundaries

The same principles valid for the solid in the first cartesian domain are also valid for the object in the cylindrical domain shown in the figure 4.2. So the value of the

stream function along the solid boundaries is:

$$\Psi = 0 \tag{3.31}$$

The derivation of the vorticity boundary conditions for the cylindrical domain is similar to the way they are derived for the cartesian domain.

Vorticity boundary conditions for this domain are derived using the stream function equation valid for the cylindrical domain.

$$\frac{\partial^2\Psi}{\partial r^2} + \frac{1}{r}\frac{\partial\Psi}{\partial r} + \frac{1}{r^2}\frac{\partial^2\Psi}{\partial\theta^2} = -\Omega \tag{3.32}$$

For instance, along the E2-F2 line shown in the figure 4.2 the value of the stream function is constant and therefore the equation 3.32 simplifies to this form for this boundary.

$$\frac{1}{r^2}\frac{\partial^2\Psi}{\partial\theta^2}|_{\widehat{r},\widehat{\theta}} = -\Omega|_{\widehat{r},\widehat{\theta}} \tag{3.33}$$

Writing the second order Taylor series expansion of the equation

$$\Psi_{\widehat{r},\widehat{\theta}-1} = \Psi_{\widehat{r},\widehat{\theta}} + \frac{\partial\Psi}{\partial\theta}|_{\widehat{r},\widehat{\theta}}\Delta\theta + \frac{\partial^2\Psi}{\partial\theta^2}|_{\widehat{r},\widehat{\theta}}\frac{(\Delta\theta)^2}{2} + O(\Delta\theta)^3 \tag{3.34}$$

Applying the no-slip boundary condition for that line $\frac{\partial\Psi}{\partial\theta}|_{r,\theta} = 0$ and solving the equation for $\frac{\partial^2\Psi}{\partial\theta^2}$ and using the equation (3.33) the vorticity equation for that boundary will be formed.

The boundary conditions are given according to the coordinates of the corners and lengths of the inner curvature tube. The coordinates of the corners of the tube are:

$E2(Space_r + 1, Space_\theta + 1)$; $\quad F2(Space_r + N_{r-in}, Space_\theta + 1)$;
$G2(Space_r + N_{r-in}, Space_\theta + N_{\theta-in})$; $\quad H2(Space_r + 1, Space_\theta + N_{\theta-in})$

For the E2-F2 boundary

$$\Omega_{\widehat{r},\widehat{\theta}} = \frac{1}{(r_{in} + (\widehat{r} - 1)\Delta r)^2} \frac{2(\Psi_{\widehat{r},\widehat{\theta}} - \Psi_{\widehat{r},\widehat{\theta}-1})}{(\Delta\theta)^2} \tag{3.35}$$

where $Space_r + 2 \leqslant \widehat{r} \leqslant Space_r + N_{r-in} - 1$ and $\widehat{\theta} = Space_\theta + 1$

For the F2-G2 boundary,

$$\Omega_{\widehat{r},\widehat{\theta}} = \frac{2(\Psi_{\widehat{r},\widehat{\theta}} - \Psi_{\widehat{r}+1,\widehat{\theta}})}{(\Delta r)^2} \tag{3.36}$$

where $\widehat{r} = Space_r + N_{r-in}$ and $Space_\theta + 1 \leqslant \widehat{\theta} \leqslant Space_\theta + N_{\theta-in}$

For the G2-H2 boundary,

$$\Omega_{\widehat{r},\widehat{\theta}} = \frac{1}{(r_{in} + (\widehat{r} - 1)\Delta r)^2} \frac{2(\Psi_{\widehat{r},\widehat{\theta}} - \Psi_{\widehat{r},\widehat{\theta}+1})}{(\Delta\theta)^2} \tag{3.37}$$

where $Space_r + 2 \leqslant \widehat{r} \leqslant Space_r + N_{r-in} - 1$ and $\widehat{\theta} = Space_\theta + N_{\theta-in}$

For the H2-E2 boundary,

$$\Omega_{\widehat{r},\widehat{\theta}} = \frac{2(\Psi_{\widehat{r},\widehat{\theta}} - \Psi_{\widehat{r}-1,\widehat{\theta}})}{(\Delta r)^2} \tag{3.38}$$

where $\widehat{r} = Space_r + 1$ and $Space_\theta + 1 \leqslant \widehat{\theta} \leqslant Space_\theta + N_{\theta-in}$

3.6.2. Computational Domain Boundaries

The boundary conditions for stream function and vorticity are taken from the solutions of the first and second cartesian domains. According to the solution algorithm shown in the figure 3.7 the iteration begins by solving the flow in the first cartesian domain and continues with the flow in the cylindrical domain using the boundary conditions taken from the first cartesian domain. However, for the remaining iterations

second cartesian domain is used and the values for the computational boundaries of the cylindrical domain are taken from the solution of that domain. So,

$$\Psi_{\widehat{r},\widehat{\theta}} = \Psi_{interpolation} \tag{3.39}$$

$$\Omega_{\widehat{r},\widehat{\theta}} = \Omega_{interpolation} \tag{3.40}$$

where $\widehat{r} = 1$ *or* $\widehat{r} = N_{r-out}$ and $1 \leqslant \widehat{\theta} \leqslant N_{\theta-out}$

3.7. The Boundary Conditions of the Second Cartesian Domain

The aim of the first cartesian domain was to provide a good initial guess for the iterations between the domains as mentioned in the solution algorithm. Moreover, there is a second cartesian domain whose function is to reflect the effect of the flow field of the curvature tube to the far fields which the cylindrical domain cannot cover.

The boundary conditions of this domain and the boundary conditions of the first cartesian domain have the same inlet, far field and outflow conditions. And the values of the inner boundaries are taken from the solution of the cylindrical domain by bi-linear interpolation.

3.7.1. Inner Boundaries

The boundary conditions are given according to the coordinates of the corners and lengths of the inner rectangle. The coordinates of the corners of the rectangle are:

$E3(Space_{x2}+1, Space_{y2}+1)$; $F3(Space_{x2}+N_{in2}, Space_{y2}+1)$;
$G3(Space_{x2}+N_{in2}, Space_{y2}+M_{in2})$; $H3(Space_{x2}+1, Space_{y2}+M_{in2})$

$$\Psi_{i,j} = \Psi_{interpolation} \tag{3.41}$$

$$\Omega_{i,j} = \Omega_{interpolation} \tag{3.42}$$

where $Space_{y2} + 1 \leqslant j \leqslant Space_{y2} + M_{in2}$ and $Space_{x2} + 1 \leqslant i \leqslant Space_{x2} + N_{in2}$

3.8. Computational and Physical Parameters

Parameters affecting the flow field around the object are grouped into the physical parameters. These are the angle of attack, Reynolds number and the radius of curvature. The effect of the velocity, viscosity and the dimensions of the object are combined into the dimensionless Reynolds number. On the other hand, there are also computational parameters affecting the convergence behavior of the problem. The convergence pattern depends also on the physical parameters because changes in these parameters also affect the character of the equations or the boundary conditions. For example, increasing Reynolds number increases the non-linear character of the equations and convergence slows down. Other than these, preconditioners, the solver and the overlapping area between the domains also influence the convergence of the problem considerably. These parameters are grouped into computational parameters.

3.8.1. Physical Parameters

3.8.1.1. Reynolds Number. Reynolds number is one of the most important flow parameters. It is the ratio of the inertial forces to the viscous forces. Solving the problem at different Reynolds numbers while holding the other parameters constant the effect of this parameter on the flow field can be analyzed.

3.8.1.2. Angle of Attack. This is a term used to describe the angle between the airfoil's chord line and the direction of airflow. In case of curvature tube the chord line is also the line joining the leading edge with the trailing edge. Angle of attack is also a very crucial parameter. It greatly affects the flow and also the forces acting on the object. Taking the solution at different angle of attacks the effect of this parameter on the flow field is investigated.

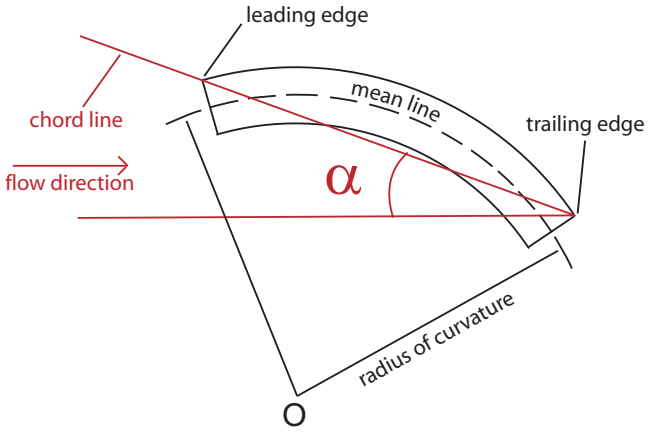


Figure 3.10. Radius of curvature and angle of attack α of the curvature tube

3.8.1.3. Radius of Curvature. By this parameter the radius of the mean curve passing through the span of the tube is meant. Like the other flow parameters it also affects the flow field considerably. Solving the problem for curvature tubes with different radius of curvatures but with the same width conclusions about the effect of this parameter can be made.

3.8.2. Computational Parameters

3.8.2.1. Solvers. Different Krylov sub-space solvers are used in this work. These are CGS, BiCGSTAB and GMRES(m). Different from the other two solvers the amount of work and storage in GMRES linearly rises with the iteration count. To overcome this limitation the iteration is restarted after a given limit m, the accumulated data are cleared and the intermediate results are used as the initial data for the next m iterations. This procedure is repeated until convergence is achieved.

These solvers have different characteristics and residual reduction behaviors. Their speed of convergence are also compared by means of iteration number and computation time.

3.8.2.2. Preconditioners. Preconditioners improve both the robustness and the efficiency of the iterative techniques. The convergence of the problem is investigated using different preconditioners. Their effect on the convergence behavior is analyzed both by means of iteration number and computation time.

3.8.2.3. Overlapping of the Domains. Both the overlapping area and the layout of this area on the base domain affect the convergence of the problem. Unsuitable selection of these parameters can lead to slow convergence and even in some cases the problem may diverge.

3.9. Code Development and Validation

The validity of the code developed is tested using an example problem. The test case should be similar to the actual problem in form and also its results must be found in the literature so that they can be compared with the solutions of the code.

3.9.1. Sample Problem:Bratu Equation

Bratu equation is one of the well known non-linear partial differential equations. It is basically Poisson equation with a non-linear source term and resembles the form of the fluid equations. Both have a non-linear character and they have to be linearized using Newton's method in order to use linear system solvers. This problem has been studied extensively and results are found in the literature.

The Bratu equation:

$$\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} = -se^{\phi} \quad where \quad 0 \leqslant s \leqslant 6.808 \tag{3.43}$$

The equation is discretized using second order central finite difference formulation.

$$\frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{(\Delta x)^2} + \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{(\Delta y)^2} = -se^{\phi_{i,j}} \tag{3.44}$$

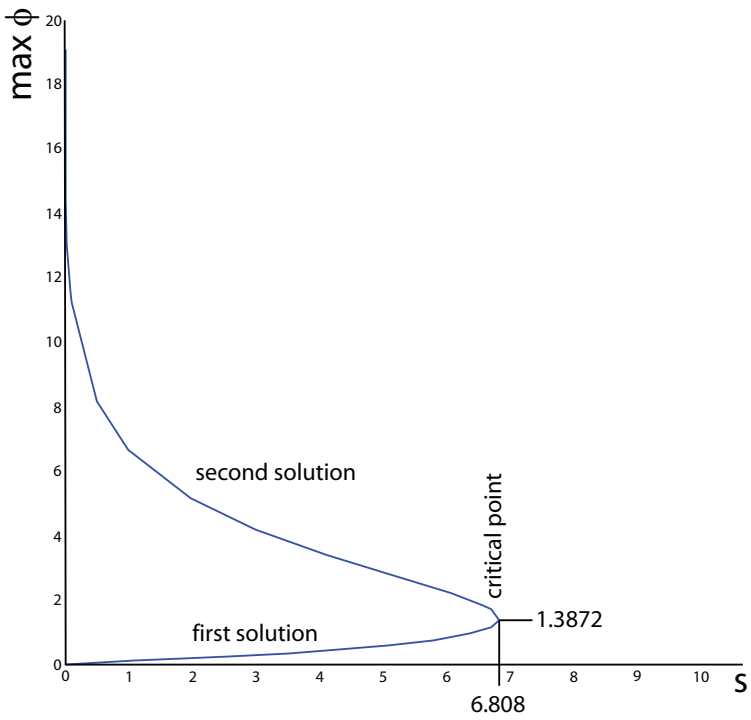


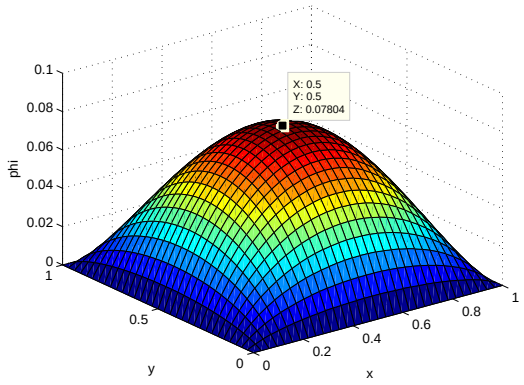
Figure 3.11. Maximum ϕ values vs. the parameter (s)

As shown in the figures the equation has two solution for values less than $s^* = 6.808$. As s gets greater the two solutions approach each other and at the limit they become equal. This point is called the bifurcation point. Going beyond this point no solutions can be obtained.

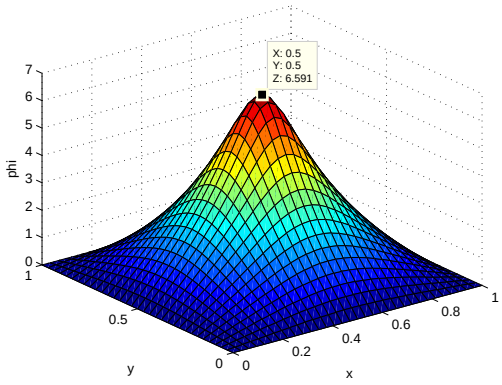
3.9.2. Results of the Bratu Equation

Bratu equation is solved on a unit 33x33 square grid with the zero Dirichlet boundary conditions for six different s numbers varying from 1 up to 6.8.

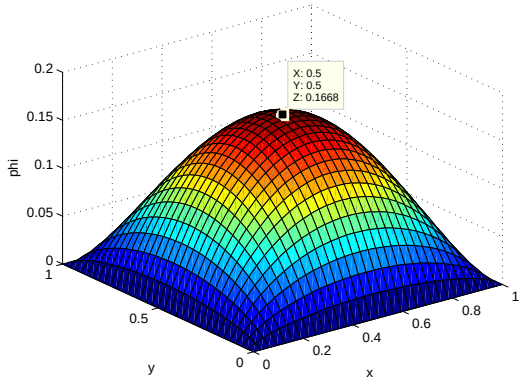
The solution on the lower branch of the graph is obtained giving the initial solution vector in the Newton's method equal to zero. Second solution can be attained changing the initial vector. The convergence of the second solution for values of s higher than three is easy so the selection of the initial guess is not so crucial. However, in order to obtain the second solution for the lower s values the second solution of the Bratu equation for the higher s values have to be used as initial guess.



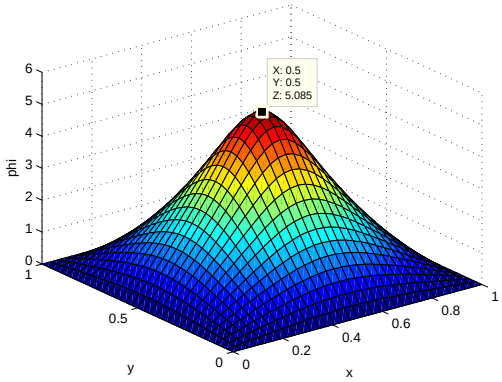
(a) $s = 1.0$ First solution



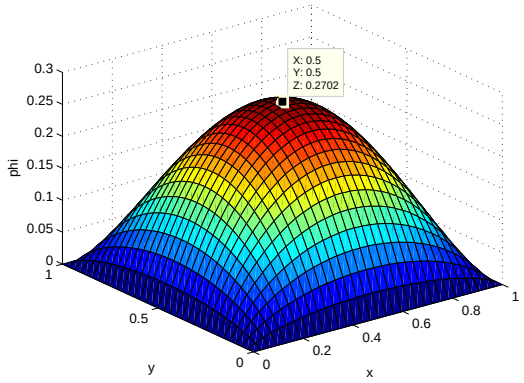
(b) $s = 1.0$ Second solution



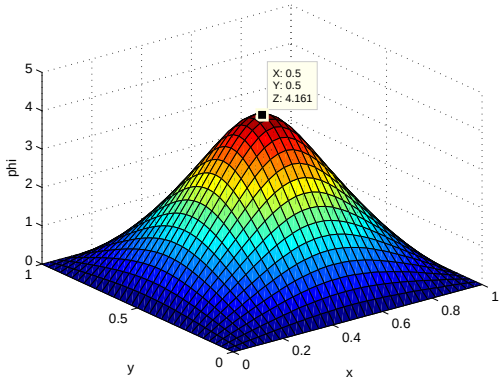
(c) $s = 2.0$ First solution



(d) $s = 2.0$ Second solution

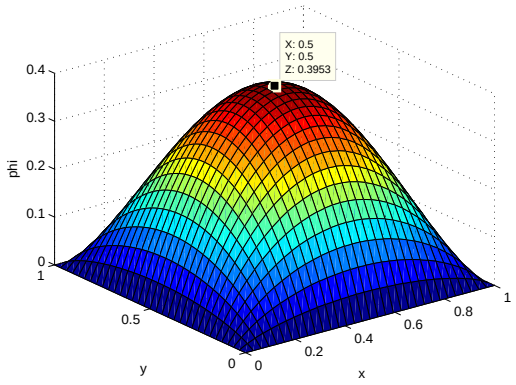


(e) $s = 3.0$ First solution

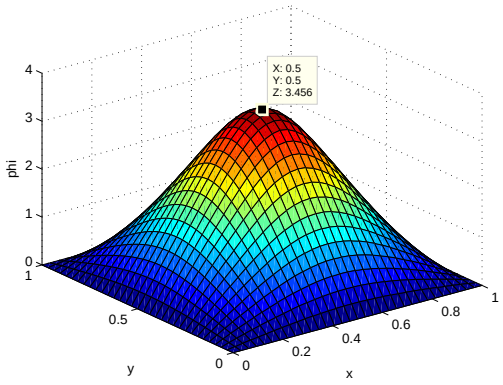


(f) $s = 3.0$ Second solution

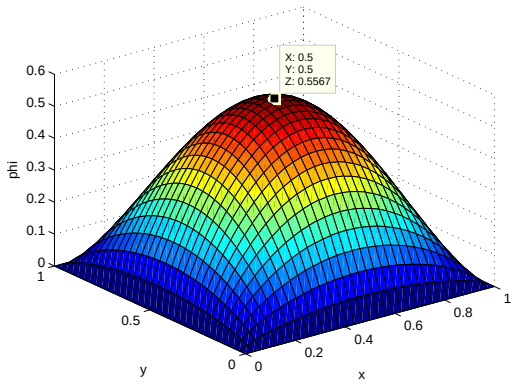
Figure 3.12. Surface plots of the Bratu equation at $s = 1.0, s = 2.0, s = 3.0$



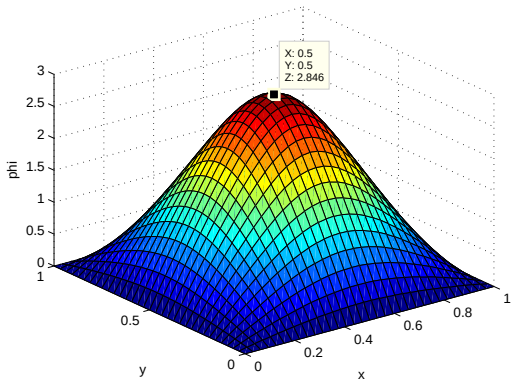
(a) $s = 4.0$ First solution



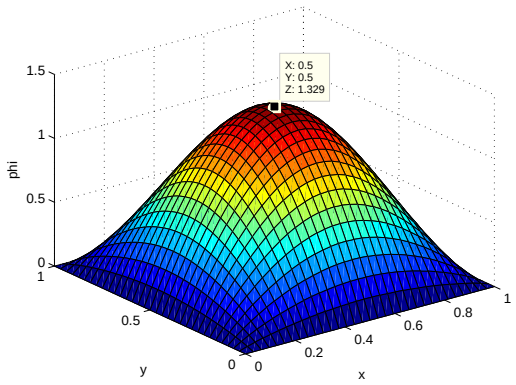
(b) $s = 4.0$ Second solution



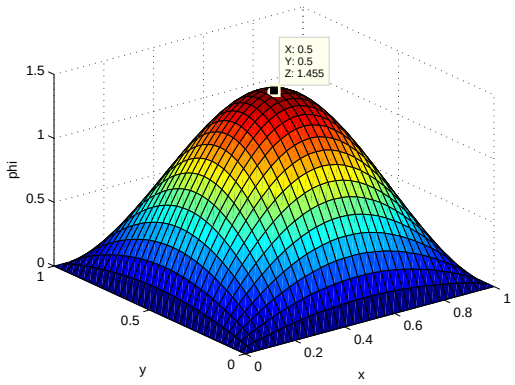
(c) $s = 5.0$ First solution



(d) $s = 5.0$ Second solution



(e) $s = 6.8$ First solution



(f) $s = 6.8$ Second solution

Figure 3.13. Surface plots of the Bratu equation at $s = 4.0, s = 5.0, s = 6.8$

On the figures above the two solutions of the Bratu equation for alternating values of s are given. As shown in the figures the maximum ϕ values of the two solutions approach each other as s gets greater. And at s=6.8 max ϕ value of the two solutions are nearly the same. They are not exactly the same because this value is not exactly the value of the bifurcation point which is $s^* = 6.808$.

4. RESULTS AND DISCUSSION

4.1. Flow Around the Curvature Tube

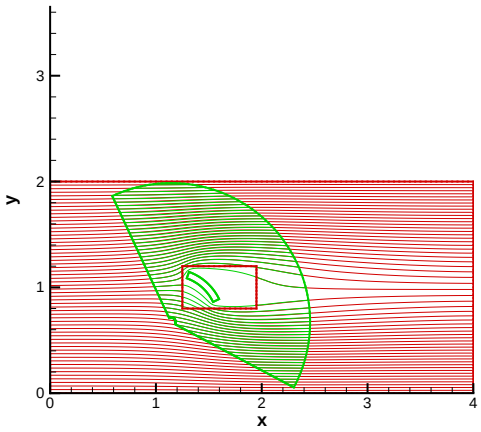
The flow around the curvature tube submitted to free flow boundary conditions are studied numerically. The results cover both the non-symmetric and symmetric cases. The non-symmetric cases cover the solutions from zero angle of attack up to 42^0. Flow at each of these angle of attacks are solved for Re=80 and Re=160. The cylindrical and cartesian domain consist of 81x81 and 61x81 grids respectively.

In the figures below the solution of both the cartesian domain and cylindrical domain are shown. The contours in the cartesian domain are plotted in red whereas the contours in the cylindrical domain are drawn in green. These are given in order to compare the solution of both two domains. The similarity between the solutions at the overlapping region indicates the convergence of the problem.

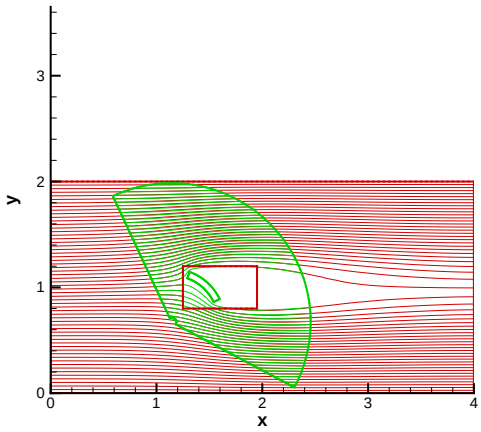
4.1.1. Effect of the Physical Parameters on the Flow Field

4.1.1.1. Angle of Attack. The field around the curvature tube is determined at five different angle of attacks from zero up to 42^0. As shown in the figures an increase in angle of attack enlarges the wake behind the solid. And after a certain value specific for that tube separation of the flow behind the object is observable and this shows us the maximum attainable angle of attack for the tube without entering into a stall which is a sharp drop in the lift force and a dramatic increase in the drag force due to the vortex formation.

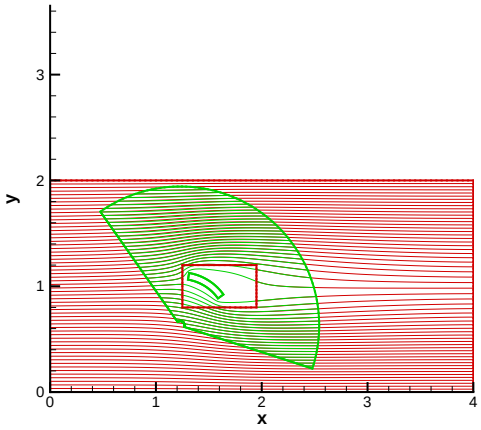
The vortex formation occurs at the angle of attack $\alpha = 28^0$ and at Re=80 as shown in the figure 4.5 and at the angle of attack $\alpha = 19^0$ and at Re=160 as shown in the figure 4.6. Increasing the Re number the separation occurs at a lower angle of attack.



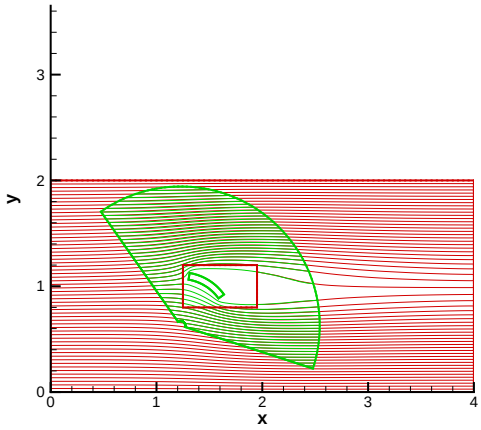
(a) $\alpha = 42^0$ Re=80



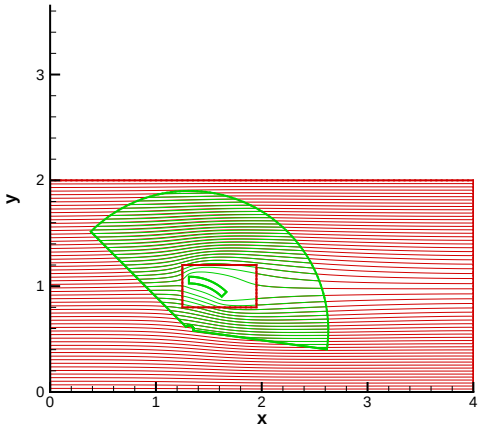
(b) $\alpha = 42^0$ Re=160



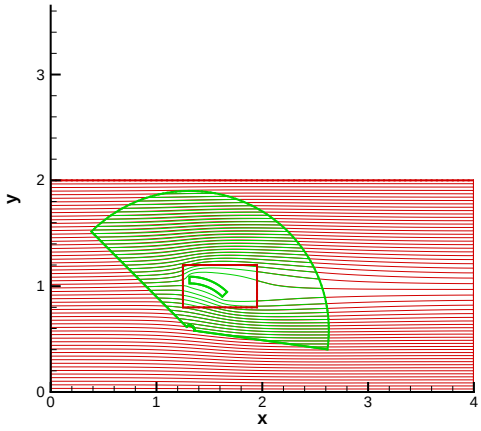
(c) $\alpha = 33^0$ Re=80



(d) $\alpha = 33^0$ Re=160

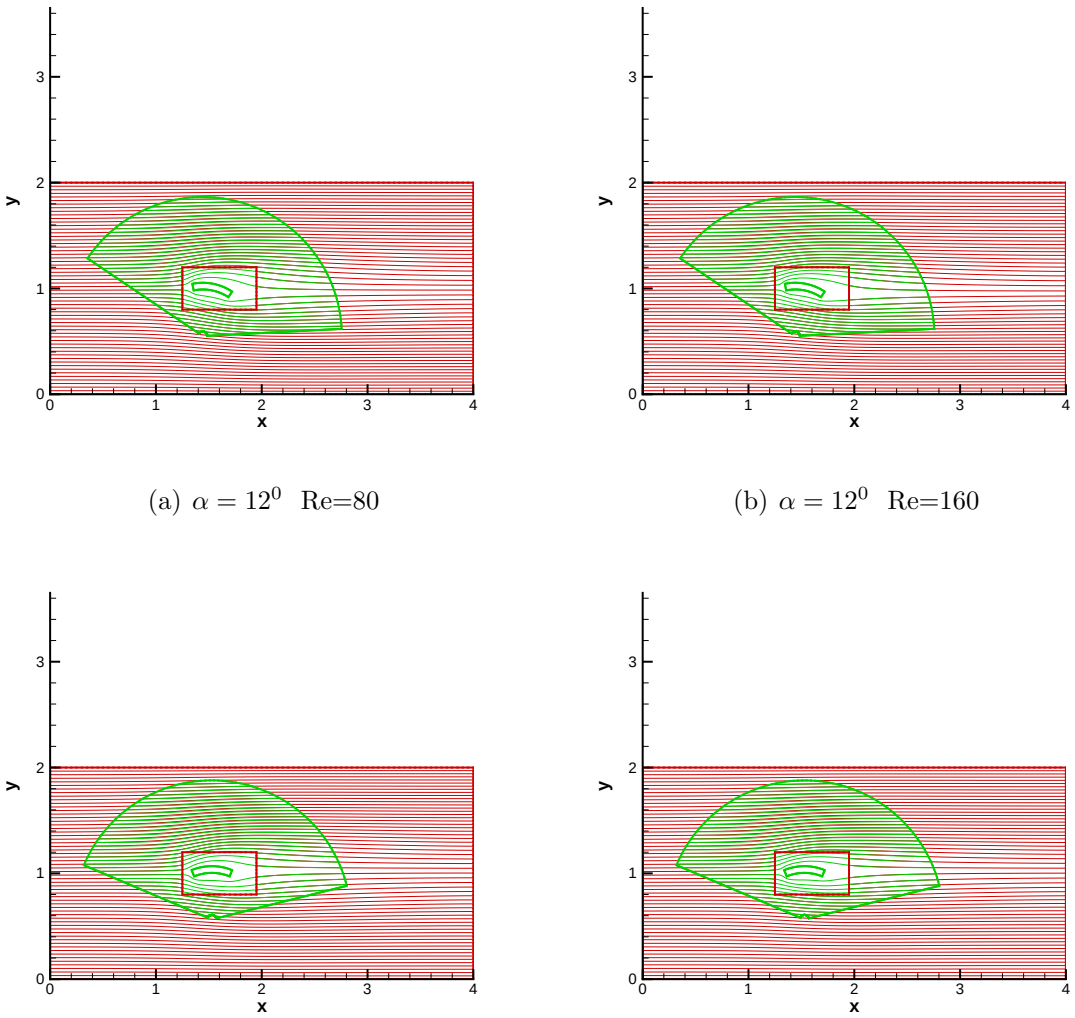


(e) $\alpha = 23^0$ Re=80



(f) $\alpha = 23^0$ Re=160

Figure 4.1. Stream function contours at the angle of attacks $\alpha = 42^0$, $\alpha = 33^0$, $\alpha = 23^0$ and at the Reynolds numbers Re=80, Re=160-Overlapping of the domains



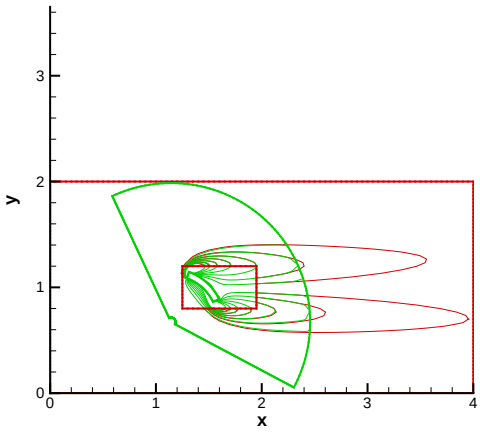
(a) $\alpha = 12^0$ Re=80

(b) $\alpha = 12^0$ Re=160

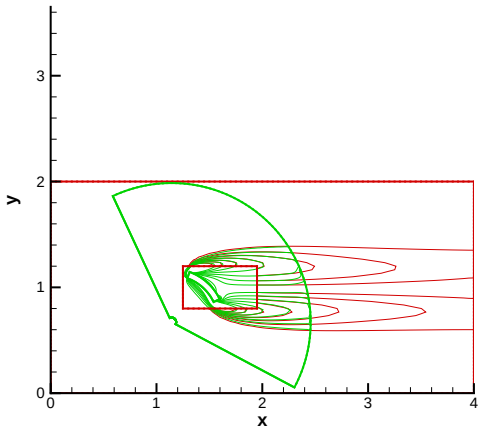
(c) $\alpha = 0^0$ Re=80

(d) $\alpha = 0^0$ Re=160

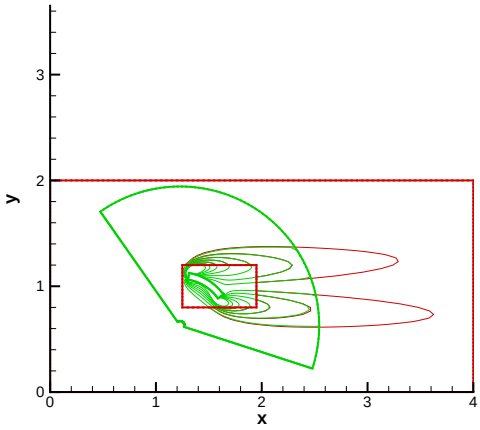
Figure 4.2. Stream function contours at the angle of attacks $\alpha = 12^0, \alpha = 0^0$ and at the Reynolds numbers Re=80, Re=160-Overlapping of the domains



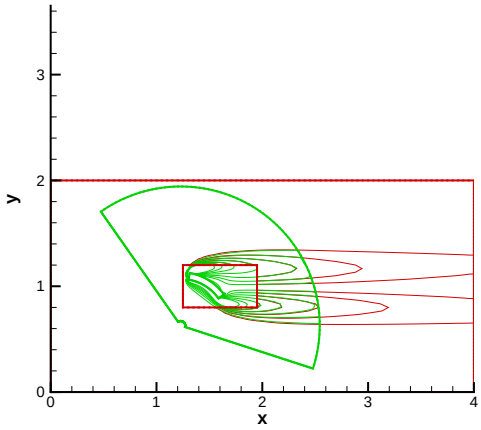
(a) $\alpha = 42^0$ Re=80



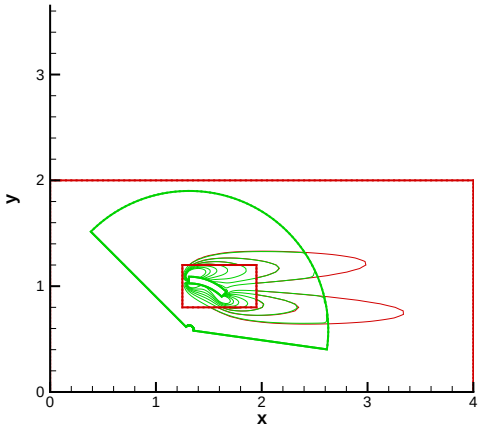
(b) $\alpha = 42^0$ Re=160



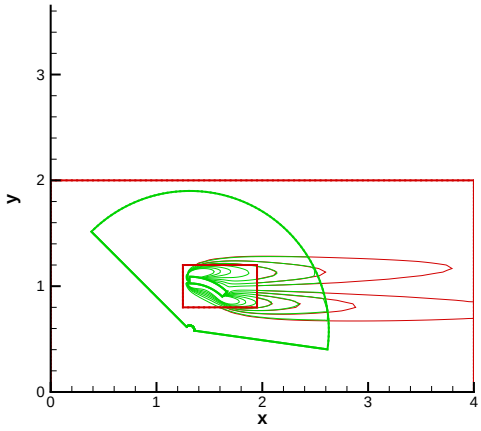
(c) $\alpha = 33^0$ Re=80



(d) $\alpha = 33^0$ Re=160

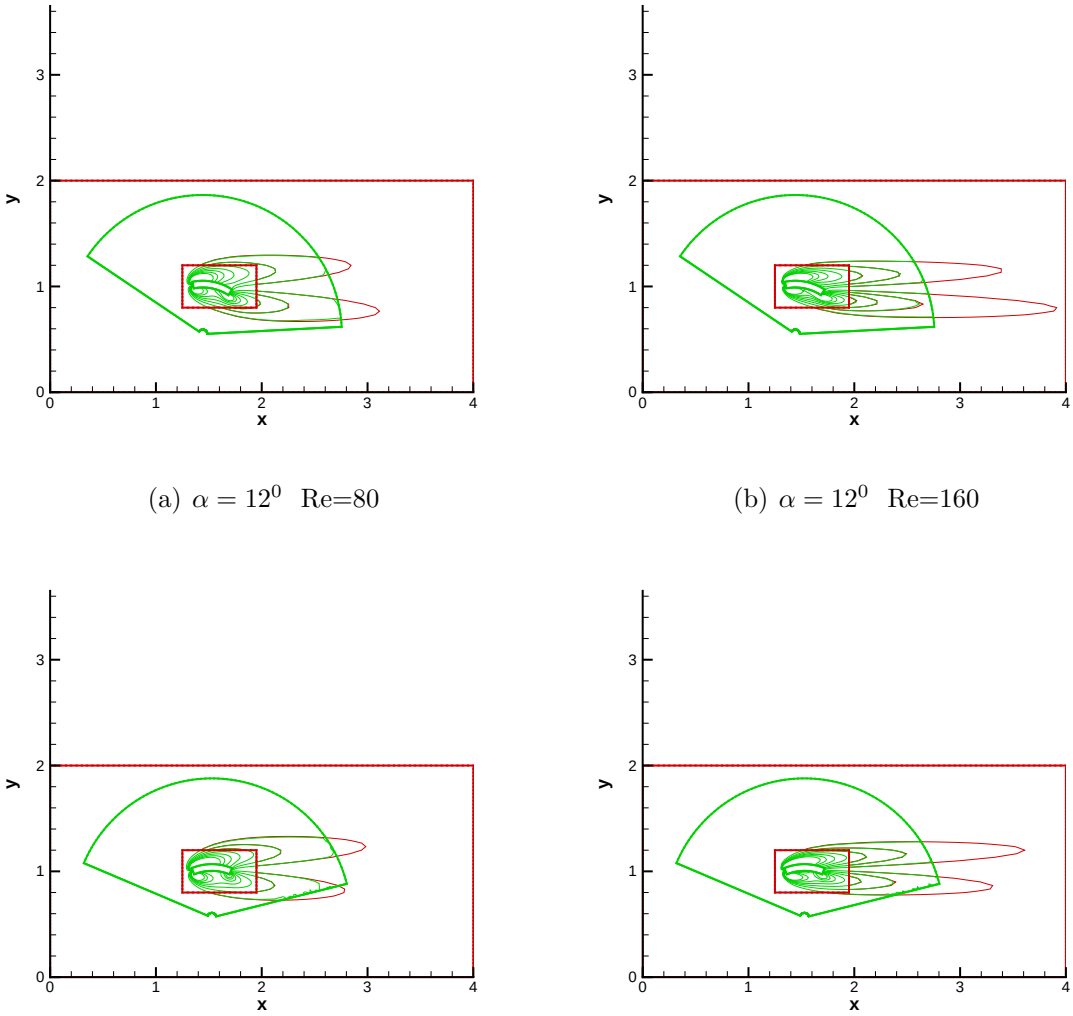


(e) $\alpha = 23^0$ Re=80



(f) $\alpha = 23^0$ Re=160

Figure 4.3. Vorticity contours at the angle of attacks $\alpha = 42^0, \alpha = 33^0, \alpha = 23^0$ and at the Reynolds numbers Re=80, Re=160-Overlapping of the domains



(a) $\alpha = 12^0$ Re=80

(b) $\alpha = 12^0$ Re=160

(c) $\alpha = 0^0$ Re=80

(d) $\alpha = 0^0$ Re=160

Figure 4.4. Vorticity contours at the angle of attacks $\alpha = 12^0, \alpha = 0^0$ and at the Reynolds numbers Re=80, Re=160-Overlapping of the domains

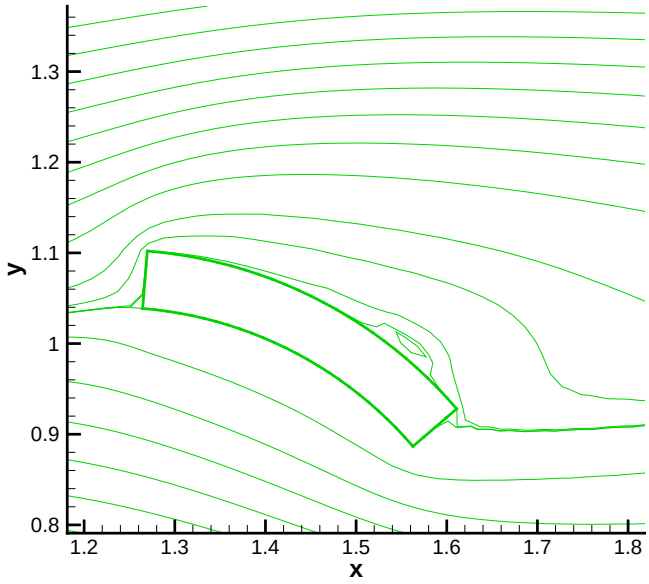


Figure 4.5. Vortex formation behind the solid at the angle of attack $\alpha = 28^0$ and at Re=80

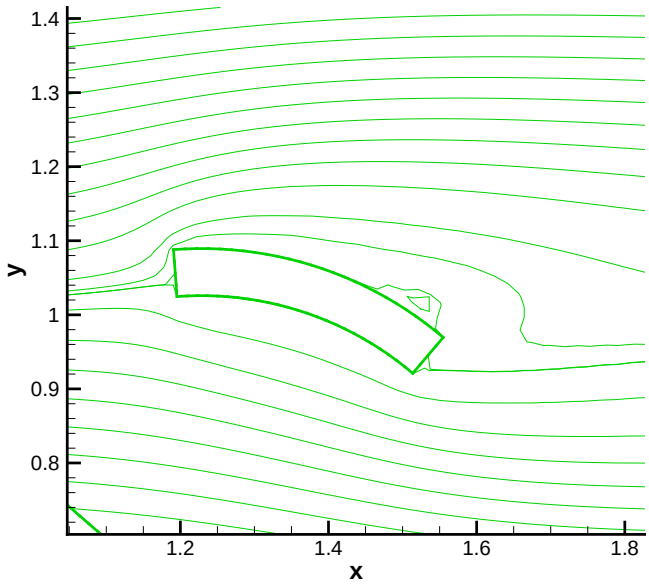


Figure 4.6. Vortex formation behind the solid at the angle of attack $\alpha = 19^0$ and at Re=160

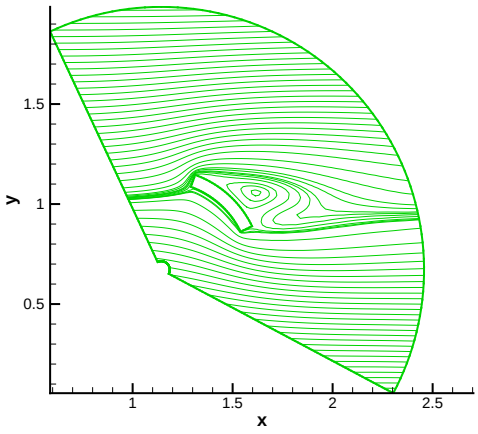
4.1.1.2. Reynolds Number. Reynolds number is one of the most important non dimensional numbers determining the flow regime. As known the flow regime shifts first from laminar to transition and then to turbulence with increasing Reynolds number. Actually the flow around the real paraglider wing at its cruising speed of $10\frac{m}{sec}$ is around 6×10^6. Therefore the flow around the wing is quite turbulent and usage of turbulent modeling is required. However, this study investigates the low Reynolds number flows so that the flow stays in the laminar region.

Looking to the stream function contours at two different Re numbers it is observed that the wake area behind the solid increases with increasing Re number. And this indicates a higher pressure force acting on the solid for high Re number flows. Also increasing the Re number the sizes of the vortices grow and they become observable. For instance, vortex formation could not be observed in the flow at the angle of attack $\alpha = 23^0$ and at Re=80. However, they form at Re=160 at the same angle of attack.

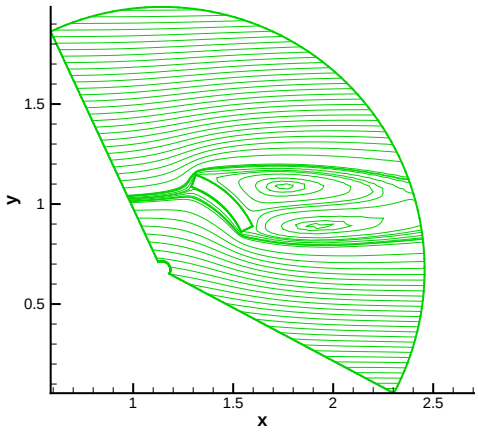
4.1.1.3. Radius of Curvature. In order to clarify the effect of the radius of curvature the flow field of two tubes having the same width but not the same radius curvatures are compared. The results are given below. As shown in the figures the wake behind the tube with higher radius curvature is larger at the same Reynolds number. Larger wake means larger pressure drag because wakes are low pressure areas. So the tube with greater radius of curvature or the tube which is more flat has a larger drag of coefficient than the one with smaller radius of curvature. This result is also in harmony with the experimental results. For instance, at the extreme case the drag coefficient of a vertical flat plate is higher than the one of a circle. [13].

4.1.2. Effect of the Computational Parameters on the Convergence Behavior

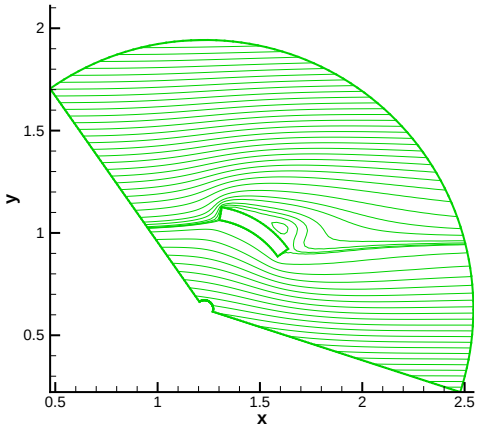
In the figures 4.16, 4.17, 4.18, 4.20, 4.21, 4.22 the convergence behavior of the solvers and the effect of the preconditioners on this behavior are shown. Three different preconditioners namely Jacobi, SGS and ILU(0) are combined with three different



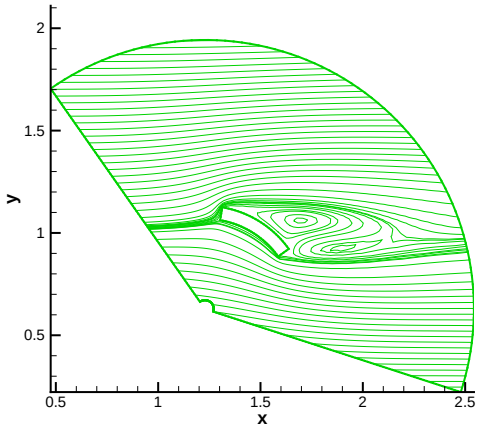
(a) $\alpha = 42^0$ Re=80



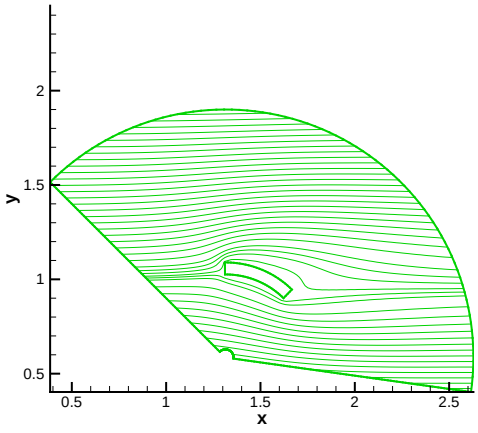
(b) $\alpha = 42^0$ Re=160



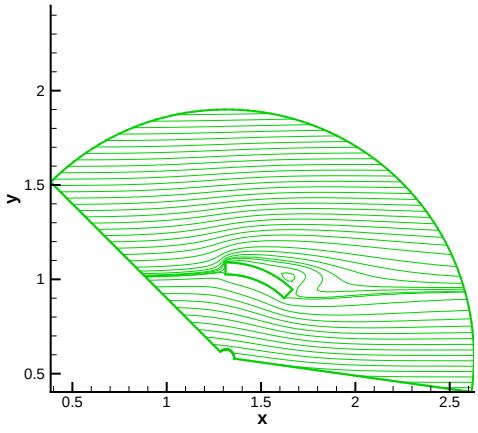
(c) $\alpha = 33^0$ Re=80



(d) $\alpha = 33^0$ Re=160

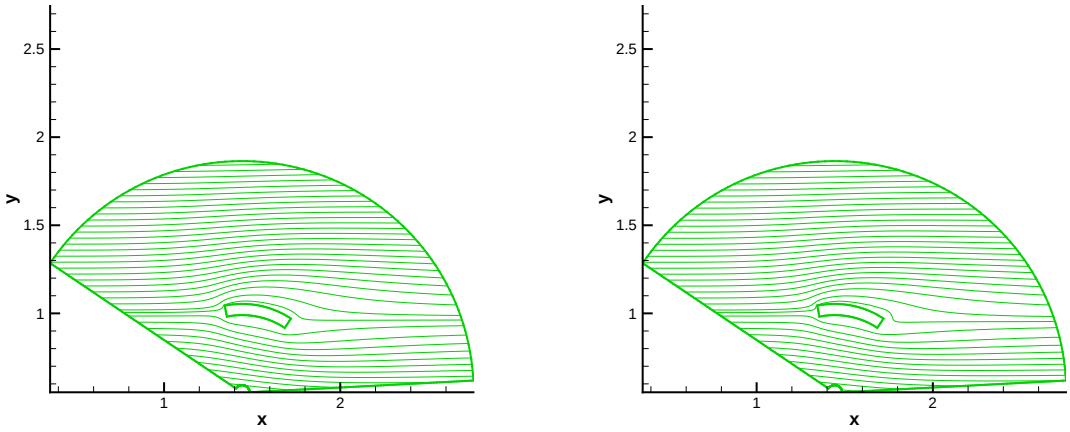


(e) $\alpha = 23^0$ Re=80



(f) $\alpha = 23^0$ Re=160

Figure 4.7. Stream function contours at the angle of attacks $\alpha = 42^0$, $\alpha = 33^0$, $\alpha = 23^0$ and at the Reynolds numbers Re=80, Re=160-Cylindrical domain



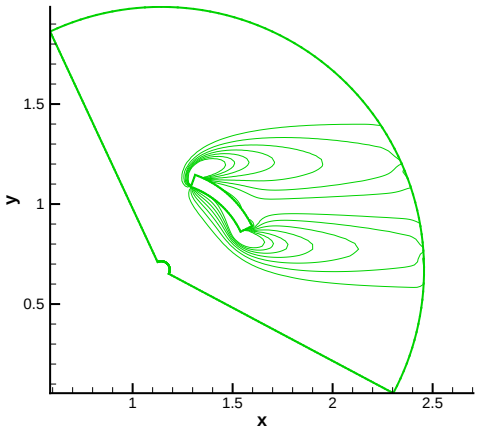
(a) $\alpha = 12^0$ Re=80

(b) $\alpha = 12^0$ Re=160

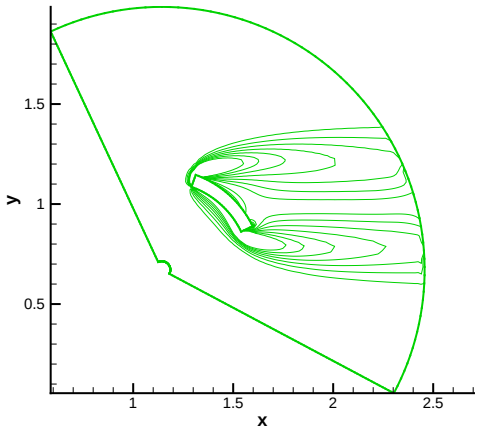
(c) $\alpha = 0^0$ Re=80

(d) $\alpha = 0^0$ Re=160

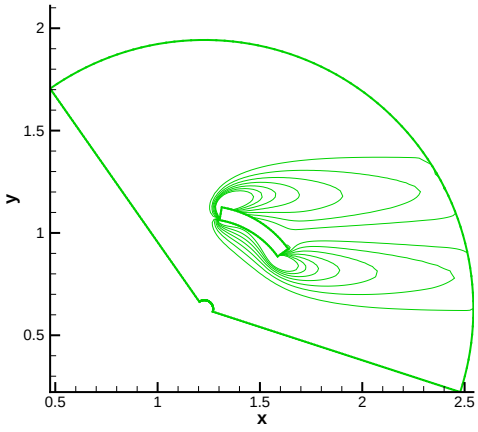
Figure 4.8. Stream function contours at the angle of attacks $\alpha = 12^0, \alpha = 0^0$ and at the Reynolds numbers Re=80, Re=160-Cylindrical domain



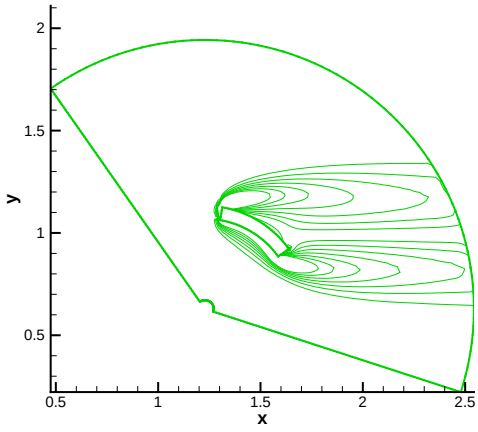
(a) $\alpha = 42^0$ Re=80



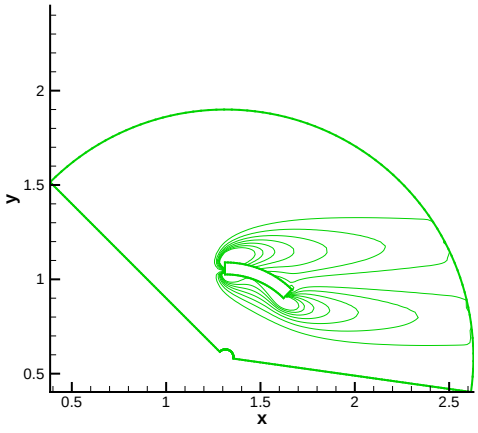
(b) $\alpha = 42^0$ Re=160



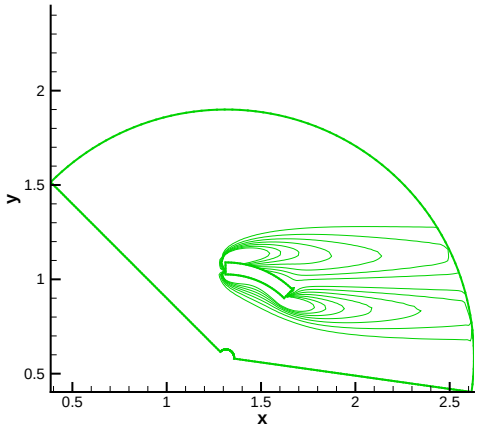
(c) $\alpha = 33^0$ Re=80



(d) $\alpha = 33^0$ Re=160

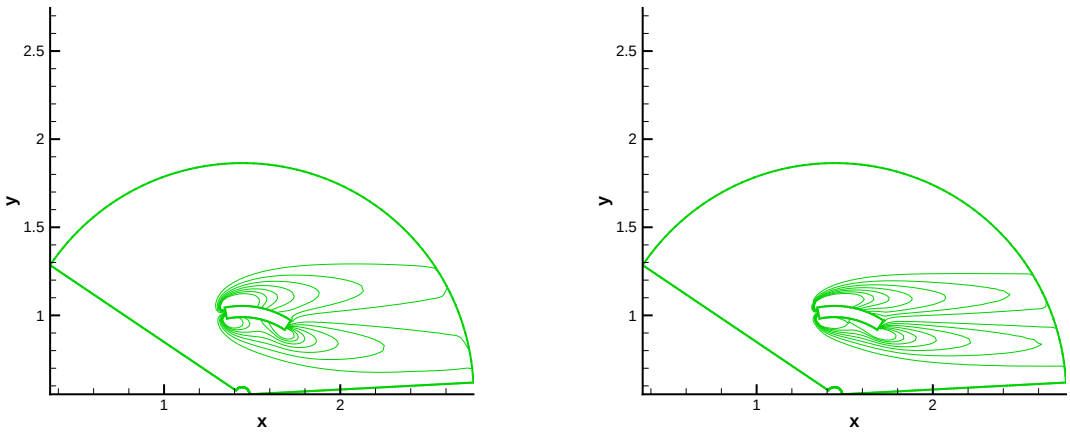


(e) $\alpha = 23^0$ Re=80



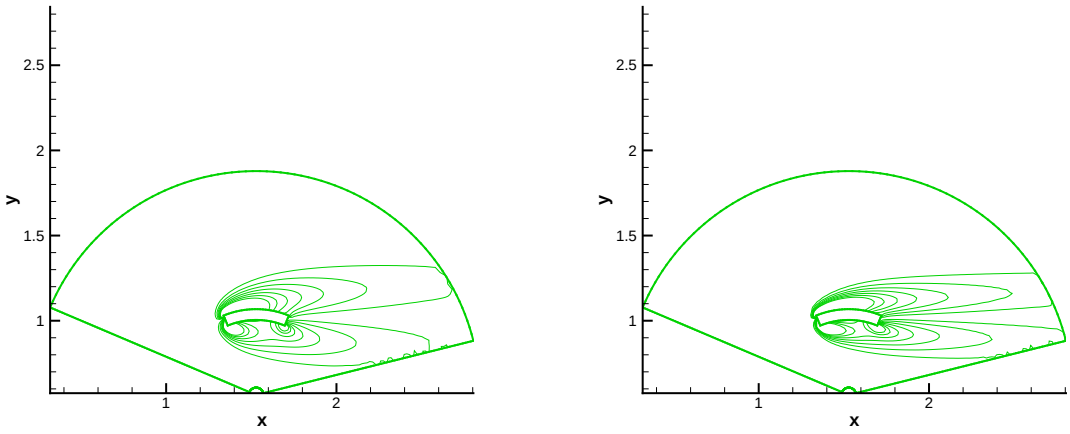
(f) $\alpha = 23^0$ Re=160

Figure 4.9. Vorticity contours at the angle of attacks $\alpha = 42^0, \alpha = 33^0, \alpha = 23^0$ and at the Reynolds numbers Re=80, Re=160-Cylindrical domain



(a) $\alpha = 12^0$ Re=80

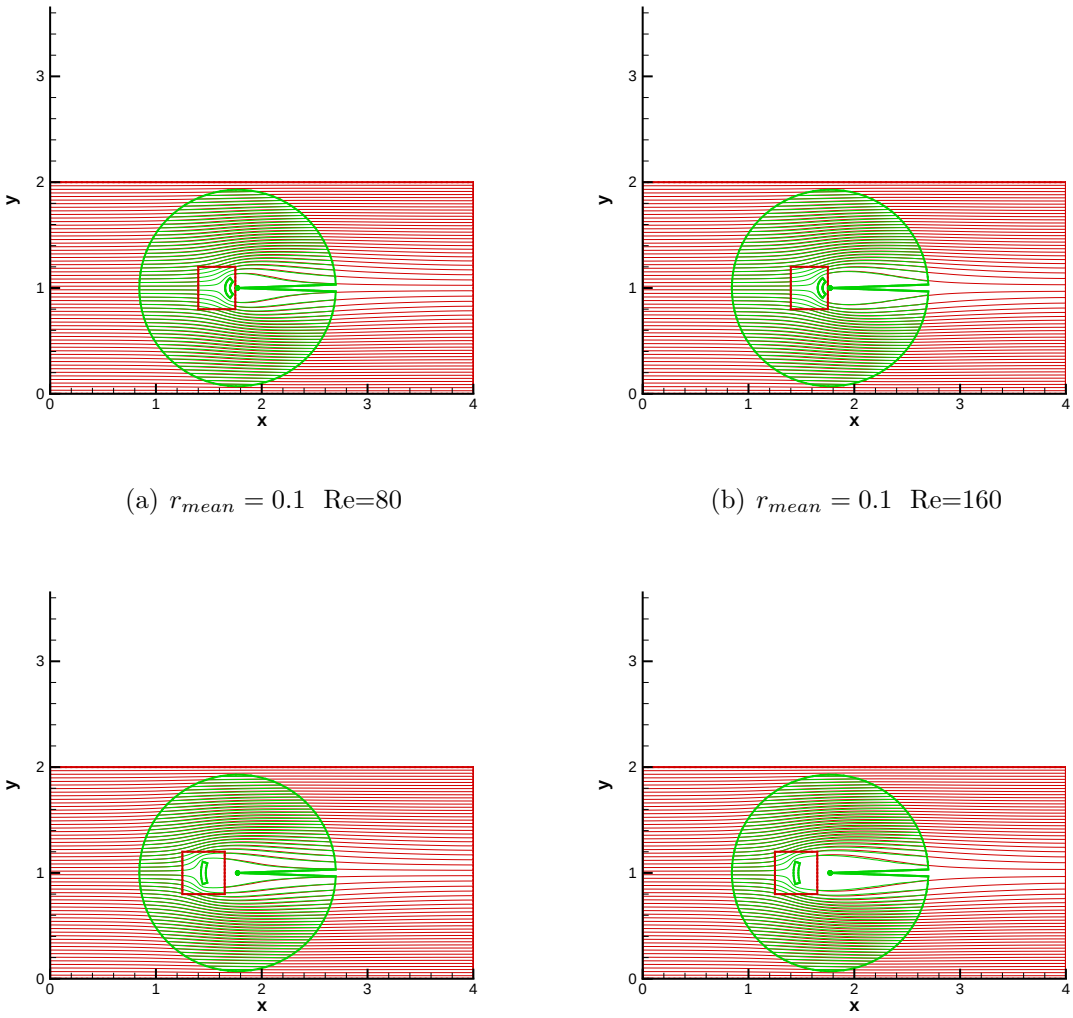
(b) $\alpha = 12^0$ Re=160



(c) $\alpha = 0^0$ Re=80

(d) $\alpha = 0^0$ Re=160

Figure 4.10. Vorticity contours at the angle of attacks $\alpha = 12^0, \alpha = 0^0$ and at the Reynolds numbers Re=80, Re=160-Cylindrical domain



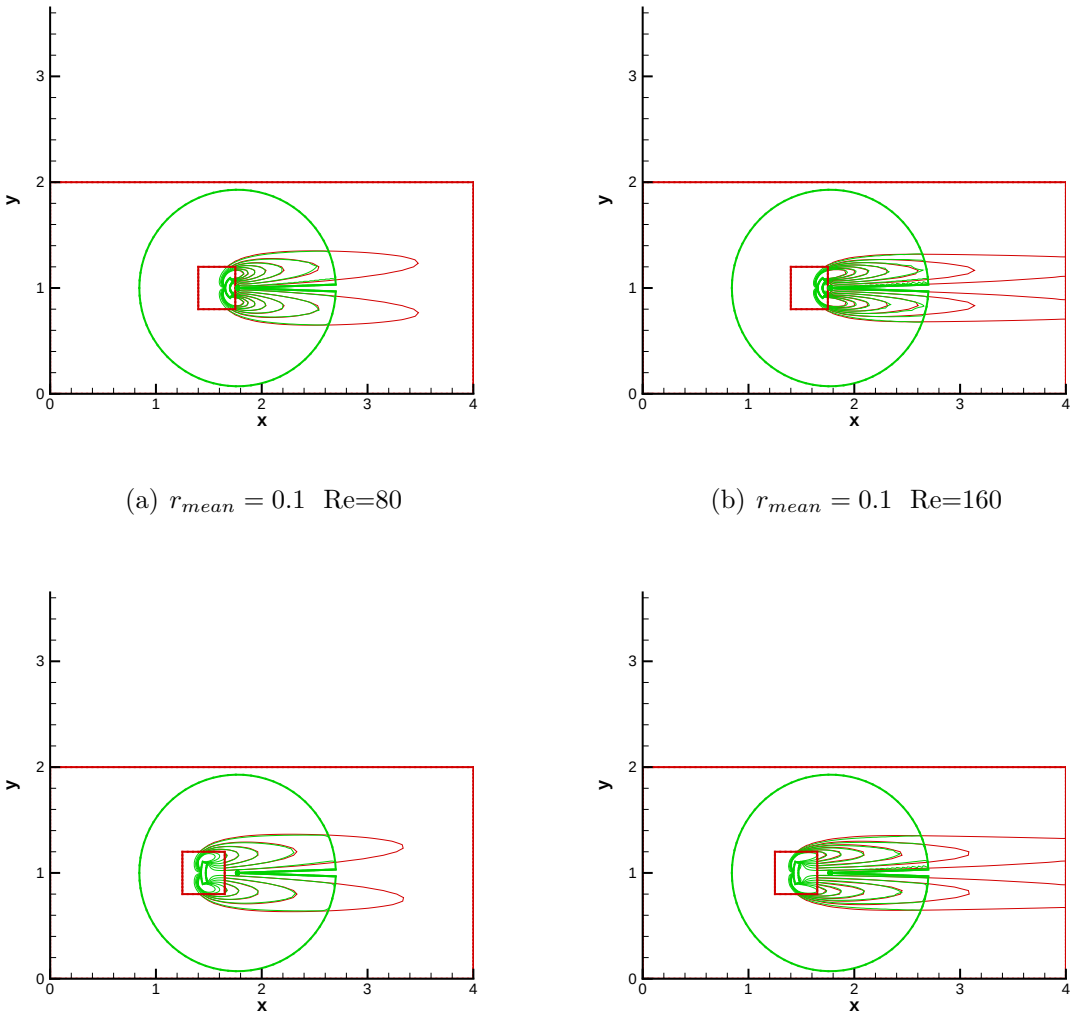
(a) $r_{mean} = 0.1$ Re=80

(b) $r_{mean} = 0.1$ Re=160

(c) $r_{mean} = 0.3$ Re=80

(d) $r_{mean} = 0.3$ Re=160

Figure 4.11. Stream function contours at the radius of curvatures $r = 0.1$, $r = 0.3$ and at the Reynolds numbers Re=80, Re=160-Overlapping of the domains



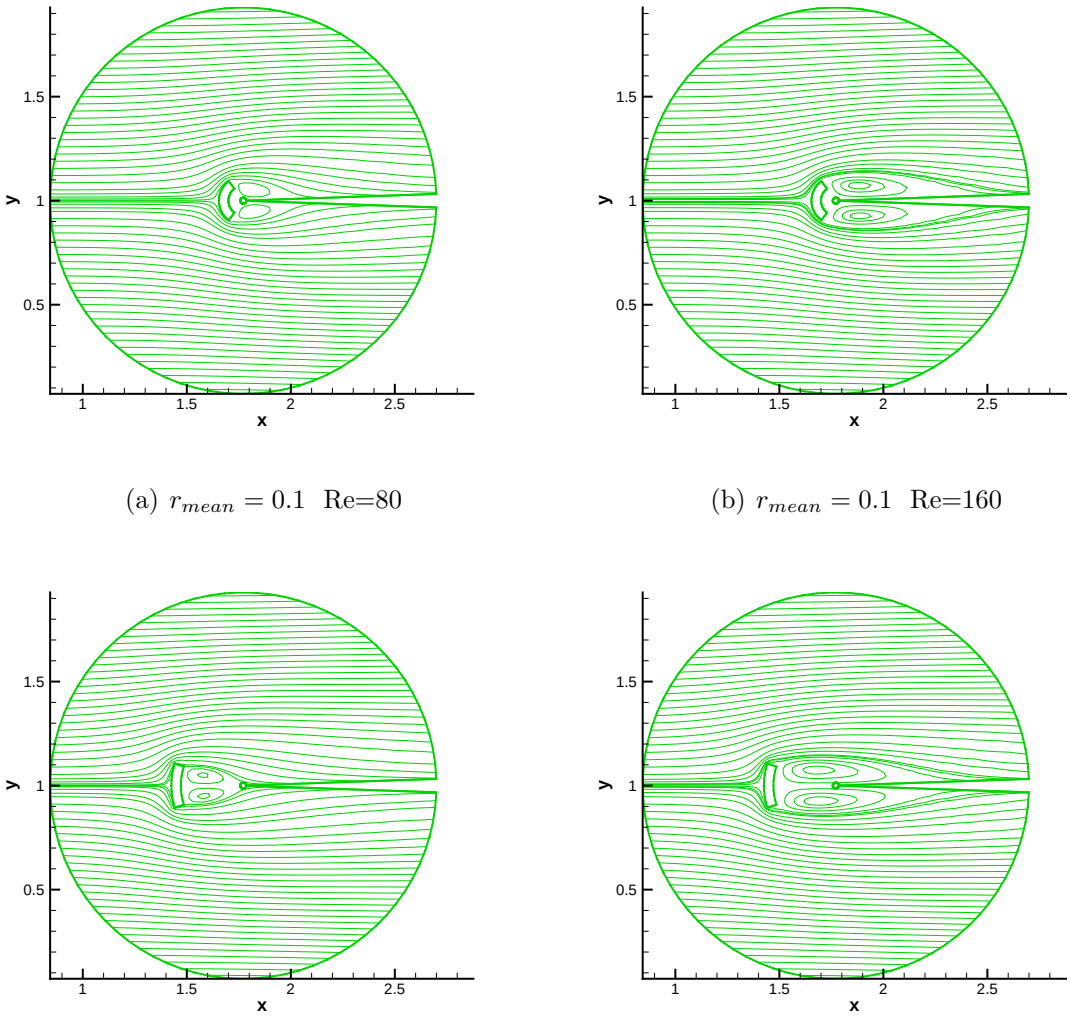
(a) $r_{mean} = 0.1$ Re=80

(b) $r_{mean} = 0.1$ Re=160

(c) $r_{mean} = 0.3$ Re=80

(d) $r_{mean} = 0.3$ Re=160

Figure 4.12. Vorticity contours at the radius of curvatures $r = 0.1, r = 0.3$ and at the Reynolds numbers Re=80, Re=160-Overlapping of the domains



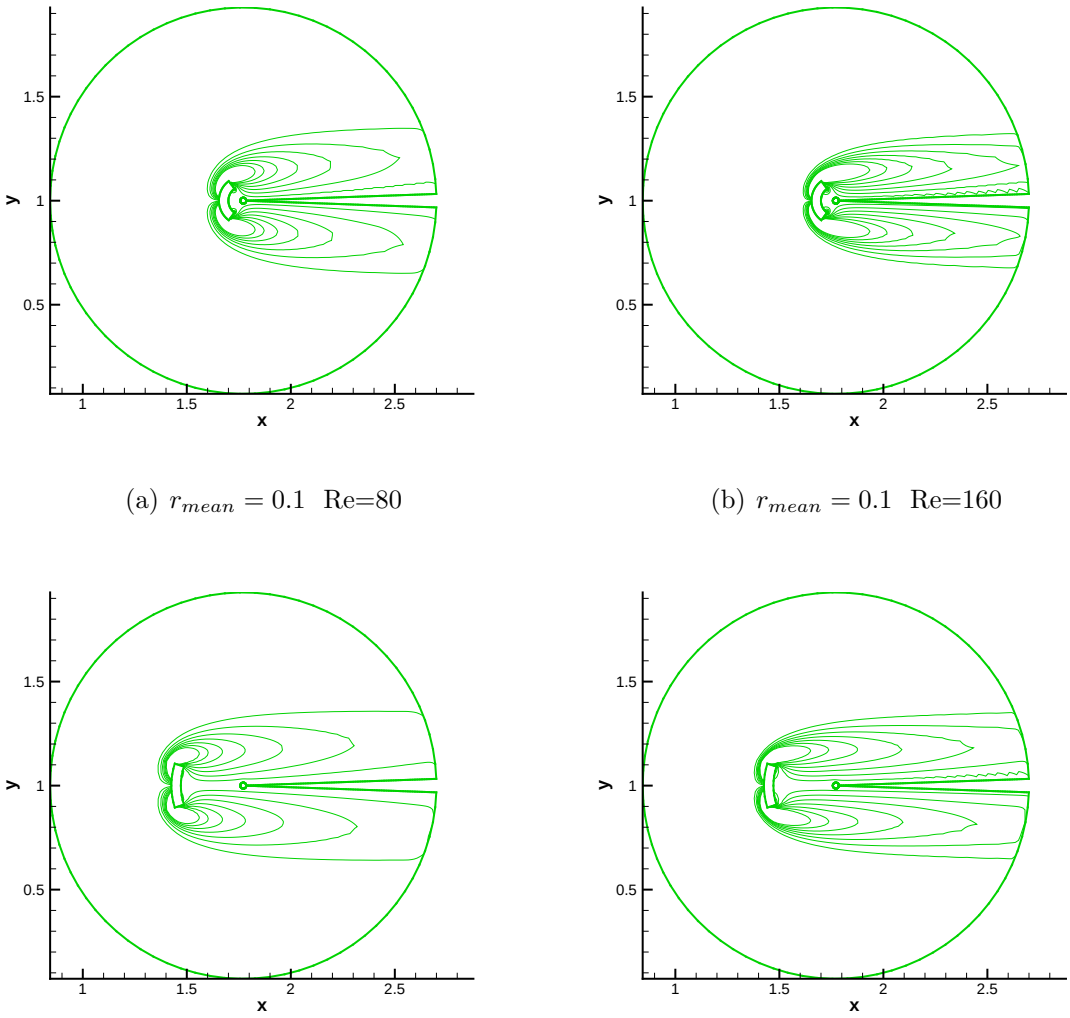
(a) $r_{mean} = 0.1$ Re=80

(b) $r_{mean} = 0.1$ Re=160

(c) $r_{mean} = 0.3$ Re=80

(d) $r_{mean} = 0.3$ Re=160

Figure 4.13. Stream function contours at the radius of curvatures $r = 0.1$, $r = 0.3$ and at the Reynolds numbers Re=80, Re=160-Cylindrical domain



(a) $r_{mean} = 0.1$ Re=80

(b) $r_{mean} = 0.1$ Re=160

(c) $r_{mean} = 0.3$ Re=80

(d) $r_{mean} = 0.3$ Re=160

Figure 4.14. Vorticity contours at the radius of curvatures $r = 0.1, r = 0.3$ and at the Reynolds numbers Re=80, Re=160-Cylindrical domain

Krylov sub-space methods such as CGS, BiCGSTAB and GMRES. The performance of these solvers are compared using the thesis problem but on a lower grid resolution. For instance, 53×82 grids for the cartesian domain and 41×51 grids for the cylindrical domain. Reynolds number is fixed to 80 at the test problem. Relative residual is checked to decide for convergence and is taken as 1×10^{-5}.

The variation of the non-linear residual with respect to the number of Newton steps are also given in the figures 4.15, 4.19. Damping factor is not needed in the test problem and zero vector is used as the initial guess both for the cartesian and the cylindrical domains. Different from the linear residual absolute non-linear residual is monitored and the tolerance is set to 1×10^{-6}.

Using the domain decomposition method the problems are solved separately in each domain using the boundary conditions of the other domain as an input. This iteration wise solution technique continues until the two norm of the difference of the solution vector of the cylindrical domain drop below the tolerance which is taken 1×10^{-5} in the test case.

As mentioned before ILU can not be applied to the matrix-free algorithms. In order to compare its effect with the effects of Jacobi and SGS exact Newton algorithm is used in the test problem. This algorithm is adapted to the compressed row storage(CRS) scheme. Otherwise the grid density in the test problem would lead to a high memory load and also the computation time would be very high.

4.1.2.1. Solvers. The residual of the linear system is evaluated relative to the initial residual of the system. That's why the residuals equal to unity at the beginning of the iterations. The variation of the relative residual is shown with respect to the iteration number in the figures 4.16, 4.17, 4.18, 4.20, 4.21, 4.22. According to the figures CGS has the most irregular convergence pattern among the solvers. BiCGSTAB shows a smoother behavior and converges in less iteration than any other solvers do. Although BiCGSTAB has to complete two more inner products than CGS does per iteration its

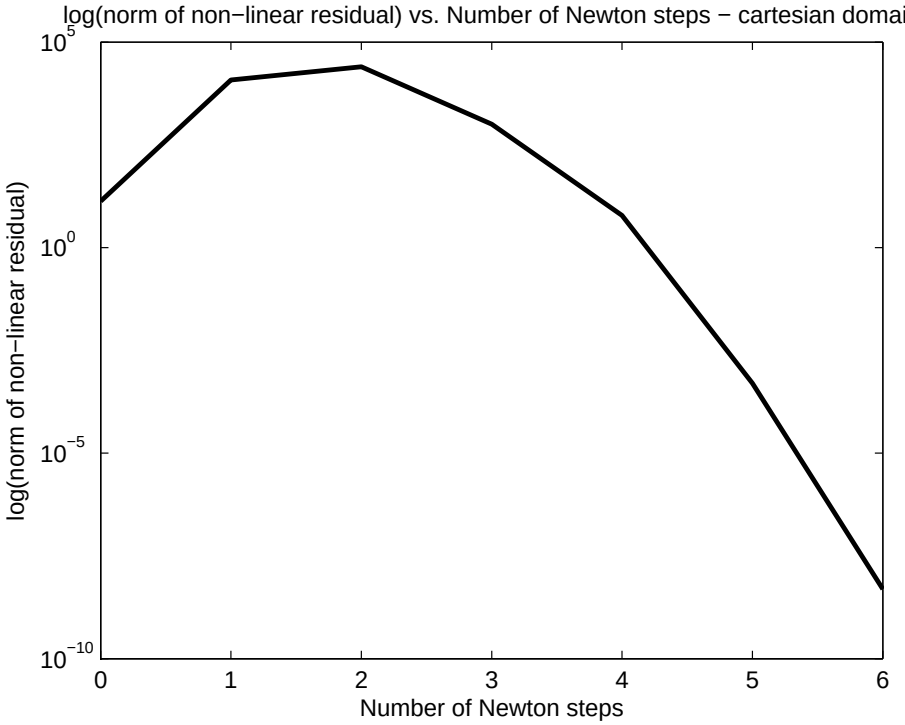


Figure 4.15. 2-norm of the non-linear residual vs. Newton steps-First cartesian domain

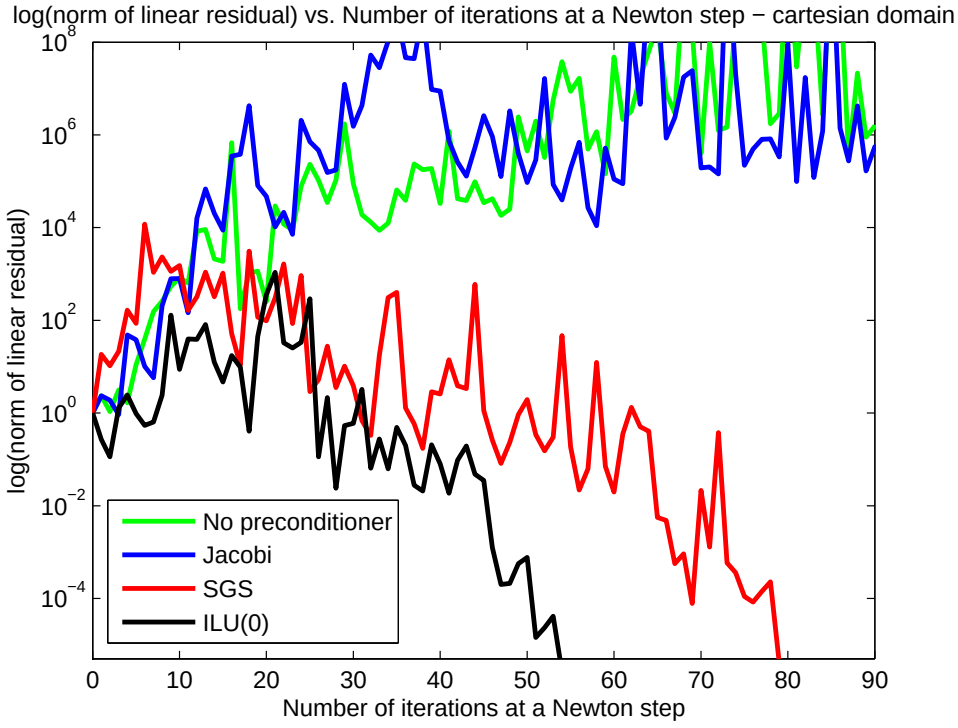


Figure 4.16. Comparison of the preconditioners/Solver:CGS-First cartesian domain

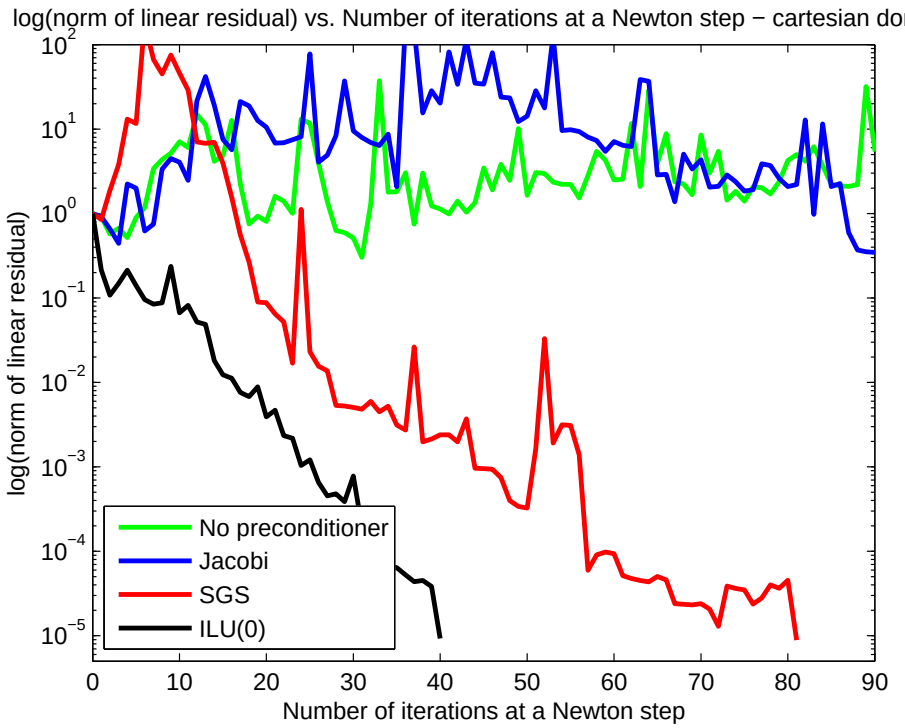


Figure 4.17. Comparison of the preconditioners/Solver:BiCGSTAB-First cartesian domain

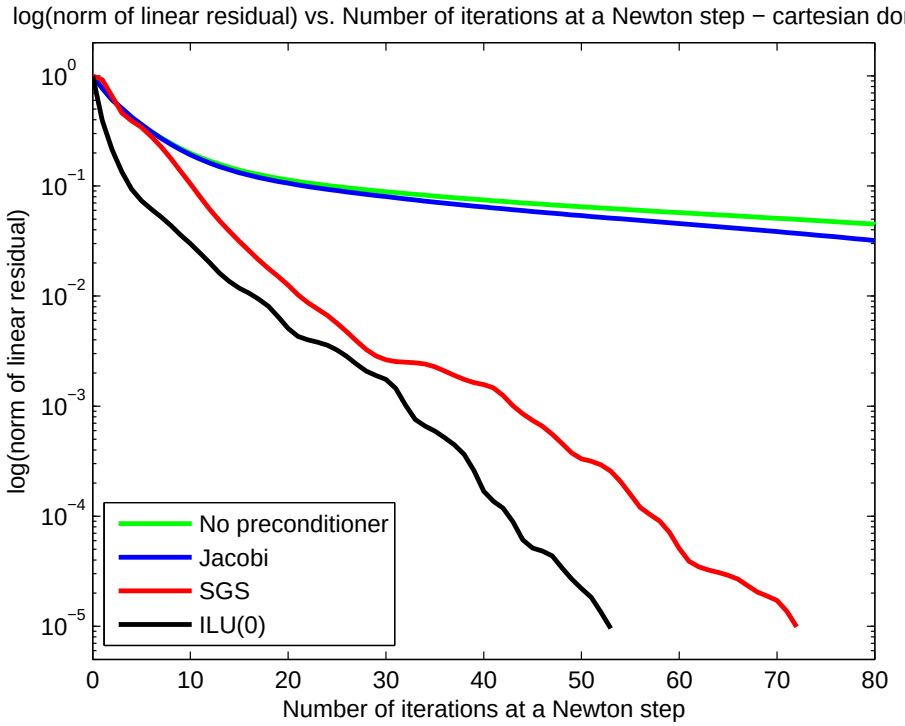


Figure 4.18. Comparison of the preconditioners/Solver:GMRES-First cartesian domain

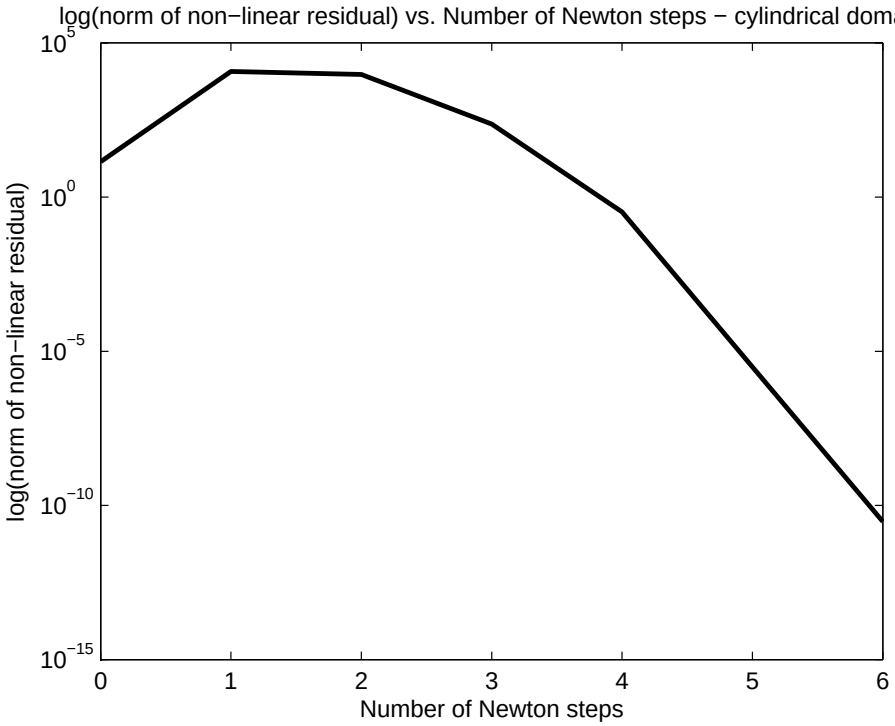


Figure 4.19. 2 norm of the non-linear residual vs. Newton steps-Cylindrical domain

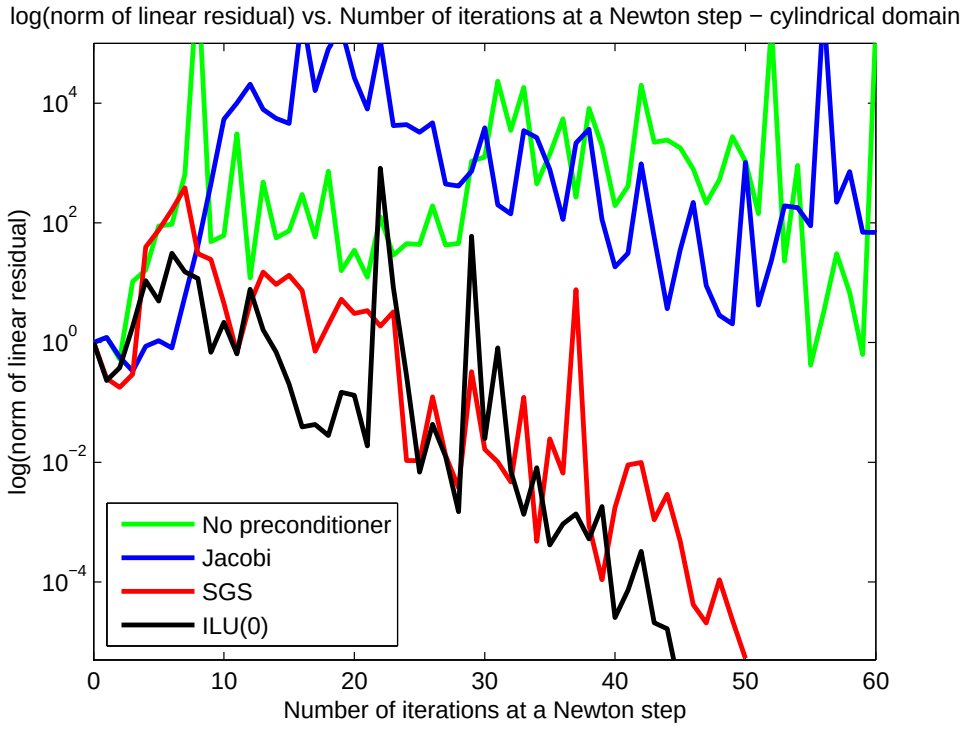


Figure 4.20. Comparison of the preconditioners/Solver:CGS-Cylindrical domain

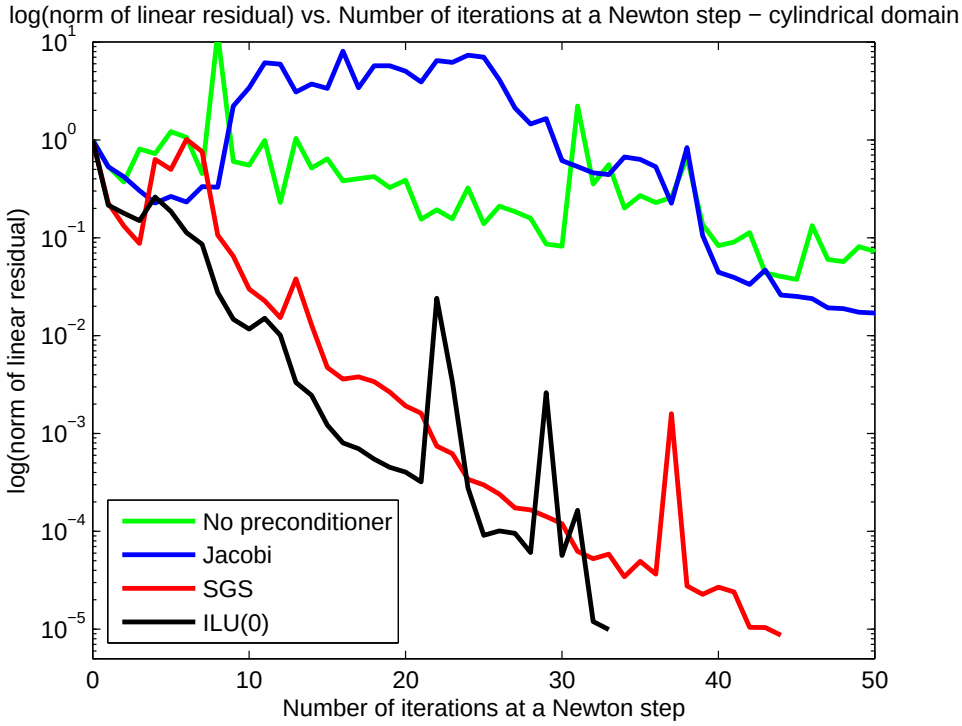


Figure 4.21. Comparison of the preconditioners/Solver:BiCGSTAB-Cylindrical domain

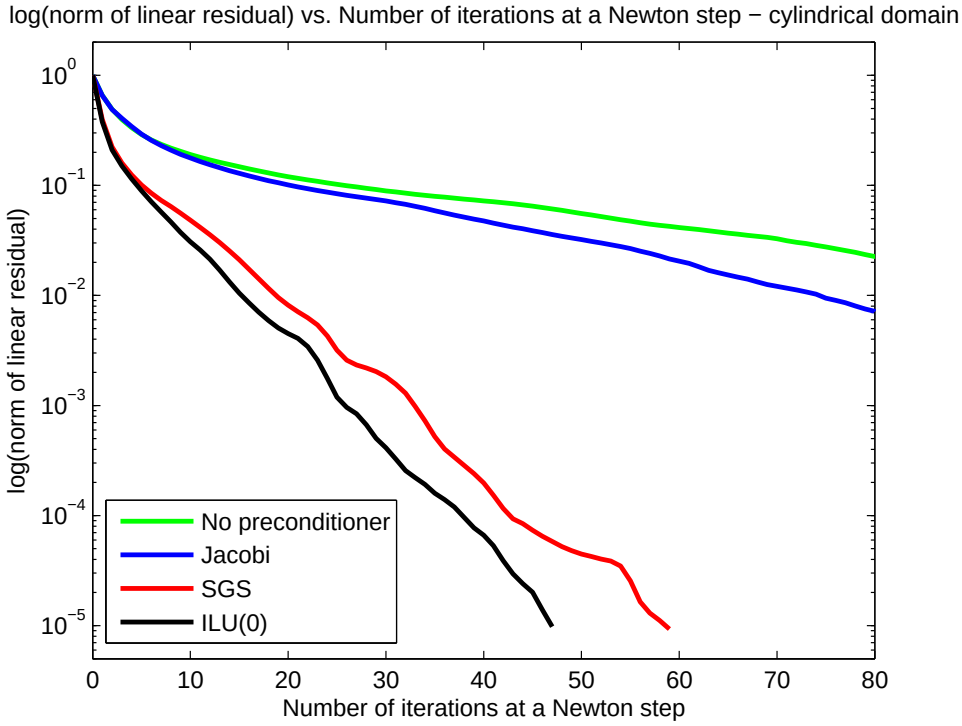


Figure 4.22. Comparison of the preconditioners/Solver:GMRES-Cylindrical domain

advantage in the iteration steps brings it also the advantage in computation time.

GMRES is the one with the most regular convergence pattern. Iterations in GMRES are restarted after 100 iterations. The accumulated data are cleared and the intermediate results are used as the initial data for the next 100 iterations. In GMRES the residual of the linear system always decreases as iteration continues which is not the case at CGS and BiCGSTAB. However, it requires more iteration steps in the sample problem than the others require to converge. On the other hand, the robust and stable character of GMRES can be used at the ill-conditioned matrices where the other solvers may diverge. The detailed comparison of these solvers are given in the tables. 2.6

In the figures 4.15, 4.19 the variation of the non-linear residual of the first cartesian domain and of the cylindrical domain are given respectively. This variation does not change with respect to the selected linear solver because the same tolerance value is valid for all of them. Changing the linear solver only affects the variation of the linear residual as far as it reaches the desired tolerance.

4.1.2.2. Preconditioners. Because of the unsuitable spectral properties of the forming Jacobian matrix solvers without preconditioners could not make the solvers converge in the given iteration limit. BiCGSTAB and CGS methods even diverge when used without preconditioner.

Jacobi as a preconditioner does not contribute to the convergence performance much. It can not ensure the convergence of the solver. CGS and BiCGSTAB methods converge with the aid of Jacobi in the cylindrical domain however they can not in the cartesian domain because of the higher grid resolution. It is one of the most easily applicable preconditioners however it does not have an important affect to the convergence behavior of the solvers.

On the other hand, SGS and ILU(0) methods greatly enhance the convergence

behavior of all three methods not only in the number of iteration steps but also in computation time. Looking to the graphs it is understood that ILU(0) preconditioned methods have a superior convergence character to the SGS preconditioned ones.

4.1.2.3. Overlapping of the Domains. The variation of the two norm of the difference between the successive stream function vectors of the cylindrical domain is given in the figure 4.23. The iterations between the domains continue until the norm of the difference vector drops below the tolerance 1×10^{-5}. In the test case nine iterations were required to achieve the tolerance set. Changing the relative sizes of the overlapping domains and the layout of these domains the iteration number between the domains would be different.

As mentioned the relative sizes of the domains as well as their layouts are important parameters affecting the convergence of the solution.

The iterations between the domains begin taking the solution of the flow around the rectangle in the cartesian domain as initial guess. Adjusting the size of this rectangle is one of the key issues in this problem. Choosing the size of the rectangle close to the size of the tube enables faster convergence. On the other hand, beginning the iterations with a rectangle with great differences in size from the curvature tube would slow down the convergence and in some cases especially at high Reynolds numbers a solution may even not converge. Shortly, it should serve as a good initial guess. Secondly, the size of the overlapping area is also a very important parameter affecting the convergence of the problem. At the first iteration the cartesian domain has the solution of the flow around the rectangle. Passing to the cylindrical domain the flow around the curvature tube is solved with respect to the outer boundary conditions taken from the cartesian domain. Although the solution at the first iteration is not correct the effect of the curvature tube is being reflected to the cartesian domain as iteration continues. Correction in the field also serves for more correct boundary conditions for the cylindrical domain. So, choosing the cylindrical domain small with respect to the tube the reflection of the solution between the domains is hindered. The

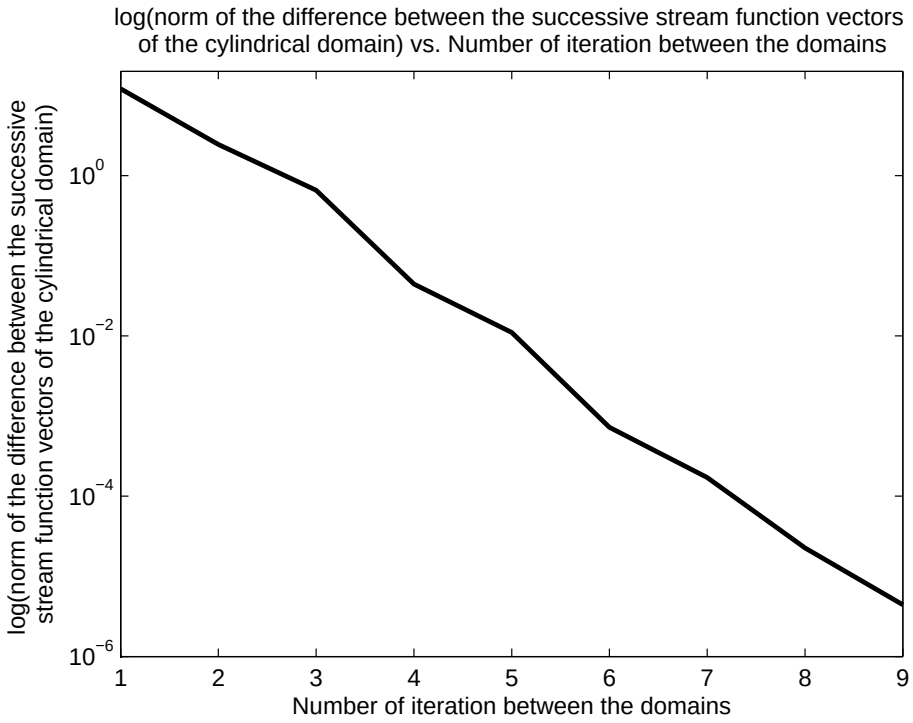


Figure 4.23. 2-norm of the difference between the successive stream function vectors in the cylindrical domain vs. Iterations between the domains

reasons for this is the formation of the vortexes behind the object especially at high Reynolds numbers. These are the signs of non-uniform flows and there the change in velocity with respect to the location is high. Transferring the values belonging to the wake of the object can lead to boundary conditions for the cylindrical domain which can be difficult to impose. Therefore it can increase the number of iterations between the domains required for convergence or the problem may even diverge. So, it is better to select the cylindrical domain as large as possible.

Another important point about the domains is the size of the rectangle which should serve the second cartesian domain as inner boundary conditions. This rectangle and the rectangle at the first iteration are different in size. The function of the rectangle at the first iteration was to produce a flow field which is similar to the field of the curvature tube. However, this rectangle serves as the inner boundary conditions for the second cartesian domain at the passage back from the cylindrical domain. Taking the rectangle too close to the solid will slow down the convergence since the solution of

the flow field near the tube is complex and therefore transferring these values back to the cartesian system and forcing the flow field in that domain to converge can lead to non-physical results. Then either a solution does not converge or even if it converges it can be non-physical. On the other hand taking the rectangle too large will increase the convergence speed but than the solution of the cylindrical domain could not be adequately reflected back to the cartesian domain. So the solution converged may also not be physical.

The overlapping area at the front of the object is not that crucial, since vorticity contours stretch parallel to the flow direction to the back of the object. Unless the overlapping area is too near to the solid non-uniform flow formation does not occur in that area. Therefore it is adequate to cover only a small part of this area and slide the overlapping area to the back of the object as far as possible so that it covers the wake behind the object.

Table 4.1. Comparison of the solvers and preconditioners by means of iteration number and computation time-cartesian domain

Method	# of Newton Steps	# of total iterations	# of iterations per Newton Step (average)	Max number of iterations among Newton Steps	Time (sec.)
CGS	-	-	-	> 500	-
Jacobi-CGS	-	-	-	> 500	-
SGS-CGS	6	519	86.5	101	25.9
ILU(0)-CGS	6	372	62	101	19.8
BiCGSTAB	-	-	-	> 500	-
Jacobi-BiCGSTAB	-	-	-	> 500	-
SGS-BiCGSTAB	6	481	80.2	101	24.2
ILU(0)-BiCGSTAB	6	295	49.2	75	16.7
GMRES(100)	-	-	-	$> 5 \times 100$	-
Jacobi-GMRES(100)	-	-	-	$> 5 \times 100$	-
SGS-GMRES(100)	6	560	93.3	147	26.7
ILU(0)-GMRES(100)	6	412	68.7	118	17.9

Table 4.2. Comparison of the solvers and preconditioners by means of iteration number and computation time-cylindrical domain

Method	# of Newton Steps	# of total iterations	# of iterations per Newton Step (average)	Max number of iterations among Newton Steps	Time (sec.)
CGS	-	-	-	> 500	-
Jacobi-CGS	6	1290	215	264	17.9
SGS-CGS	6	316	52.7	67	8.0
ILU(0)-CGS	6	257	42.8	53	7.1
BiCGSTAB	-	-	-	> 500	16
Jacobi-BiCGSTAB	6	1122	187	215	16
SGS-BiCGSTAB	6	276	46	60	7.4
ILU(0)-BiCGSTAB	6	221	36.8	52	6.5
GMRES(100)	-	-	-	$> 5 \times 100$	-
Jacobi-GMRES(100)	-	-	-	$> 5 \times 100$	-
SGS-GMRES(100)	6	390	65	87	8.3
ILU(0)-GMRES(100)	6	319	53.2	71	7.45

5. Conclusions

In the present study, the flow around the curvature tube is investigated based on the similarity between the paraglider wing and the curvature tube. The flow field is investigated at two different Reynolds numbers combined with five different angle of attacks and two different radius of curvatures. Moreover parameters related with the convergence of the problem are also studied in this work.

In this work it is shown that the flow around the curvature tube can be solved using the domain decomposition technique. Overlapping of the non-matching domains are used namely cartesian and cylindrical. The cartesian domain serves as the base system and serves to reflect the effect of the solid to the outer boundaries. On the other hand, the overlapping cylindrical domain models the flow around the vicinity of the solid using the outer boundary conditions taken from the base domain. Iterating between the domains and correcting each time the values in the boundaries the solution is attained.

Usage of the overlapping domains enables to specify the inlet, far field and outflow boundary conditions in the cartesian domain. Holding these conditions fixed and turning the overlapping cylindrical domain the flow at various angle of attacks is obtained.

Based on the solutions it is concluded that increase in angle of attacks brings an increase in the wake area behind the object. This in turn indicates to a rise in the pressure drag. Also Reynolds number has a similar effect on the flow field. Higher Reynolds number engenders an increased pressure drag as a result of the larger wake area behind the solid.

Both exact and inexact Newton's methods are used to linearize the discretized non-linear equation system and the corrections are evaluated by the Krylov sub-space solvers (CGS, BiCGSTAB and GMRES)combined with the preconditioners Jacobi,

SGS and ILU(0).

Inexact Newton's method combined with the GMRES is a robust and efficient technique of solving large non-linear equation systems because of the stable convergence character of the GMRES and the low memory load due to the directional differencing. Moreover the performance of this method is increased with the implementation of the preconditioners such as Jacobi and SGS. The importance of the implementation of the preconditioners are shown in the figures 4.16, 4.17, 4.18, 4.20, 4.21, 4.22 comparing the convergence behavior of the linear solvers with and without preconditioners. Increasing the Reynolds number the non-linear character of the equations is increased so the gap in performance between the solvers with and without preconditioners gets bigger. Moreover, CGS and BiCGSTAB methods can show a divergent behavior even at low Reynolds numbers if applied without a preconditioner.

Implementation of SGS to the inexact Newton's method requires the modification of the backward and forward substitutions so that they can be applied also to the matrix free algorithms. Adapting the substitution methods accordingly slows down the iterations considerably. However, they can be used in ill-conditioned systems where the Jacobi preconditioner can not make GMRES converge. On the other hand, the backward and forward substitutions in the SGS preconditioned matrix free GMRES algorithm can be speeded up storing the Jacobian matrix using the compressed column storage (CCS) scheme. Writing the substitution algorithms accordingly SGS preconditioning can be applied in a faster way. Although this will not be a fully matrix free algorithm using this scheme the memory load on the computer is not increased dramatically since only the non-zero elements of the matrix are stored in the memory which make up only a small percentage of the full Jacobian matrix.

Exact Newton method is also applied in this work. Adapting the exact Newton algorithm to the compressed row storage (CRS) scheme by storing only the non-zero elements of the Jacobian matrix and defining the matrix vector multiplications accordingly a fast non-linear system solver is obtained. Moreover, the storage of the Jacobian matrix in the memory enables the implementation of the preconditioners which cannot

be used in the matrix free methods. For example, writing the ILU(0) preconditioner algorithm according to the CRS scheme also it could be applied. However, the code development phase of this method is much longer than the one of the inexact Newton since the Jacobian matrix has to be computed analytically and written in the code according to the CRS scheme.

The flow around the curvature tube is solved at two different Reynolds numbers combined with five different angle of attacks. Also the effect of the radius of curvature is also investigated by comparing the flow fields of the tubes with different radius of curvatures.

In this study a regular curvature tube is used in order to embed it into a single cylindrical domain. It composed of the arcs having the same radius of curvatures. However, using a tube with two different radius of curvatures would be a more realistic model of a paraglider wing and could be dealt with as a future project.

Using a 3D curvature tube instead of 2D can be considered as a next step of the present study. Modeling the flow in three dimensions the effect of the induced drag can also be incorporated which is due to the tip vortexes at the end of the wings. This kind of drag significantly contributes to the overall drag especially at the low gliding speeds. Therefore this kind of modeling can help us to figure out the relation between the induced drag and the form of the curvature tube.

Extending the 2D problem to 3D will result in an increase in the number of unknowns. However, the low memory load of the matrix free approach enables also to deal with the 3D flow problems with ease. Also the code development stage will not take long since only the f vector has to be modified according to the new problem statement. Using also the preconditioners adapted to the matrix free environment the problem can be solved in a memory efficient and fast way.

Today the performance of the real paraglider wings are also modeled on the computer using commercial fluid flow programs. However, they do not take the elasticity

of the cloth and also the lines into account. Moreover, modeling of the flight characteristics of collapsed gliders is also another crucial aspect. Such considerations combined with aero-elastic modeling is perhaps the future of paraglider design and will enable more realistic airflow modeling of the paraglider wings.

APPENDIX A: DERIVATION OF THE STREAM FUNCTION VORTICITY FORMULATION IN CYLINDRICAL COORDINATES

Continuity equation

$$v_r + r\frac{\partial v_r}{\partial r} + \frac{\partial v_\theta}{\partial \theta} = 0 \tag{A.1}$$

Definition of vorticity

$$\Omega = \frac{\partial v_\theta}{\partial r} + \frac{v_\theta}{r} - \frac{1}{r}\frac{\partial v_r}{\partial \theta} \tag{A.2}$$

Momentum equations

$$r - dir \qquad \rho\left(\frac{\partial v_r}{\partial t} + v_r\frac{\partial v_r}{\partial r} + \frac{v_\theta}{r}\frac{\partial v_r}{\partial \theta} - \frac{v_\theta^2}{r}\right) = -\frac{\partial p}{\partial r} + \rho g_r + \mu\left[\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial v_r}{\partial r}\right) - \frac{v_r}{r^2} + \frac{1}{r^2}\left(\frac{\partial^2 v_r}{\partial \theta^2}\right) - \frac{2}{r^2}\left(\frac{\partial v_\theta}{\partial \theta}\right)\right] \tag{A.3}$$

$$\theta - dir \qquad \rho\left(\frac{\partial v_\theta}{\partial t} + v_r\frac{\partial v_\theta}{\partial r} + \frac{v_\theta}{r}\frac{\partial v_\theta}{\partial \theta} + \frac{v_r v_\theta}{r}\right) = -\frac{1}{r}\frac{\partial p}{\partial \theta} + \rho g_\theta + \mu\left[\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial v_\theta}{\partial r}\right) - \frac{v_\theta}{r^2} + \frac{1}{r^2}\left(\frac{\partial^2 v_\theta}{\partial \theta^2}\right) + \frac{2}{r^2}\left(\frac{\partial v_r}{\partial \theta}\right)\right] \tag{A.4}$$

Taking the derivative of the momentum equation in r direction with respect to θ,

$$\rho\left(\frac{\partial^2 v_r}{\partial \theta \partial t} + \frac{\partial v_r}{\partial \theta}\frac{\partial v_r}{\partial r} + v_r\frac{\partial^2 v_r}{\partial \theta \partial r} + \frac{1}{r}\left(\frac{\partial v_\theta}{\partial \theta}\frac{\partial v_r}{\partial \theta} + v_\theta\frac{\partial^2 v_r}{\partial \theta^2}\right) - \frac{2v_\theta}{r}\frac{\partial v_\theta}{\partial \theta}\right) = -\frac{\partial^2 p}{\partial \theta \partial r} + \rho\frac{\partial g_r}{\partial \theta} + \mu\left[\frac{1}{r}\frac{\partial^2 v_r}{\partial \theta \partial r} + \frac{\partial^3 v_r}{\partial \theta \partial r^2} - \frac{1}{r^2}\frac{\partial v_r}{\partial \theta} + \frac{1}{r^2}\frac{\partial^3 v_r}{\partial \theta^3} - \frac{2}{r^2}\frac{\partial^2 v_\theta}{\partial \theta^2}\right] \tag{A.5}$$

Multiplying the momentum equation in θ direction with r and taking the derivative with respect to r,

$$\rho\left(\frac{\partial v_\theta}{\partial t}+\frac{\partial^2 v_\theta}{\partial r\partial t}+v_r\frac{\partial v_\theta}{\partial r}+r\left(\frac{\partial v_r}{\partial r}\frac{\partial v_\theta}{\partial r}+v_r\frac{\partial v_\theta}{\partial r^2}\right)+\frac{\partial v_\theta}{\partial r}\frac{\partial v_\theta}{\partial \theta}+v_\theta\frac{\partial^2 v_\theta}{\partial r\partial\theta}+v_\theta\frac{\partial v_r}{\partial r}+v_r\frac{\partial v_\theta}{\partial r}\right)$$
$$=-\frac{\partial^2 p}{\partial r\partial\theta}+\rho\left(g_\theta+r\frac{\partial g_\theta}{\partial r}\right)+$$
$$\mu\left[\frac{\partial^2}{\partial r^2}\left(r\frac{\partial v_\theta}{\partial r}\right)-\frac{1}{r}\frac{\partial v_\theta}{\partial r}+\frac{v_\theta}{r^2}-\frac{1}{r^2}\frac{\partial^2 v_\theta}{\partial\theta^2}+\frac{1}{r}\frac{\partial^3 v_\theta}{\partial r\partial\theta^2}-\frac{2}{r^2}\frac{\partial v_r}{\partial\theta}+\frac{2}{r}\frac{\partial^2 v_r}{\partial r\partial\theta}\right] \tag{A.6}$$

Subtracting A.6 from A.5

LHS:

$$\rho\left[\begin{array}{l}\left(\frac{\partial^2 v_r}{\partial\theta\partial t}-\frac{\partial v_\theta}{\partial t}-r\frac{\partial^2 v_\theta}{\partial r\partial t}\right)+v_r\left(\frac{\partial^2 v_r}{\partial\theta\partial r}-r\frac{\partial^2 v_\theta}{\partial r^2}-2\frac{\partial v_\theta}{\partial r}\right)+v_\theta\left(\frac{1}{r}\frac{\partial^2 v_r}{\partial\theta^2}-\frac{1}{r}\frac{\partial v_\theta}{\partial\theta}-\frac{\partial^2 v_\theta}{\partial r\partial\theta}\right)+\\ \left(\frac{1}{r}\frac{\partial v_\theta}{\partial\theta}\frac{\partial v_r}{\partial\theta}+\frac{\partial v_r}{\partial\theta}\frac{\partial v_r}{\partial r}-r\frac{\partial v_\theta}{\partial r}\frac{\partial v_r}{\partial r}-\frac{\partial v_\theta}{\partial r}\frac{\partial v_\theta}{\partial\theta}-\frac{v_\theta}{r}\frac{\partial v_\theta}{\partial\theta}-v_\theta\frac{\partial v_r}{\partial r}\right)\end{array}\right]=$$

$$\rho\left[\begin{array}{l}\frac{\partial}{\partial t}\left(\frac{\partial v_r}{\partial\theta}-r\frac{\partial v_\theta}{\partial r}-v_\theta\right)+v_r\frac{\partial}{\partial r}\left(\frac{\partial v_r}{\partial\theta}-r\frac{\partial v_\theta}{\partial r}-v_\theta\right)+v_\theta\frac{\partial}{\partial\theta}\left(\frac{1}{r}\frac{\partial v_r}{\partial\theta}-\frac{\partial v_\theta}{\partial r}-\frac{v_\theta}{r}\right)+\\ \left(\frac{1}{r}\frac{\partial v_\theta}{\partial\theta}\frac{\partial v_r}{\partial\theta}-\frac{\partial v_\theta}{\partial r}\frac{\partial v_\theta}{\partial\theta}-\frac{v_\theta}{r}\frac{\partial v_\theta}{\partial\theta}\right)+\left(\frac{\partial v_r}{\partial\theta}\frac{\partial v_r}{\partial r}-r\frac{\partial v_\theta}{\partial r}\frac{\partial v_r}{\partial r}-v_\theta\frac{\partial v_r}{\partial r}\right)\end{array}\right]=$$

$$\rho\left[\frac{\partial}{\partial t}\left(-\Omega r\right)+v_r\frac{\partial}{\partial r}\left(-\Omega r\right)+v_\theta\frac{\partial}{\partial\theta}\left(-\Omega\right)+\left(-\Omega\frac{\partial v_\theta}{\partial\theta}\right)+\left(-\Omega r\frac{\partial v_r}{\partial r}\right)\right]= \tag{A.7}$$

RHS:

$$=\mu\left[\begin{array}{l}\left(\frac{1}{r^2}\frac{\partial^3 v_r}{\partial\theta^3}-\frac{2}{r^2}\frac{\partial^2 v_\theta}{\partial\theta^2}-\frac{1}{r}\frac{\partial^3 v_\theta}{\partial r\partial\theta^2}+\frac{1}{r^2}\frac{\partial^2 v_\theta}{\partial\theta^2}\right)+\left(\frac{\partial^3 v_r}{\partial\theta\partial r^2}-\frac{\partial^2}{\partial r^2}\left(r\frac{\partial v_\theta}{\partial r}\right)\right)\\ +\left(\frac{1}{r}\frac{\partial^2 v_r}{\partial\theta\partial r}-\frac{1}{r^2}\frac{\partial v_r}{\partial\theta}+\frac{2}{r^2}\frac{\partial v_r}{\partial\theta}-\frac{2}{r}\frac{\partial^2 v_r}{\partial r\partial\theta}+\frac{1}{r}\frac{\partial v_\theta}{\partial r}-\frac{v_\theta}{r^2}\right)\end{array}\right]$$

$$=\mu\left[\frac{\partial^2}{\partial\theta^2}\left(\frac{1}{r^2}\frac{\partial v_r}{\partial\theta}-\frac{1}{r}\frac{\partial v_\theta}{\partial r}-\frac{v_\theta}{r^2}\right)+\frac{\partial^2}{\partial r^2}\left(\frac{\partial v_r}{\partial\theta}-r\frac{\partial v_\theta}{\partial r}\right)+\frac{\partial}{\partial r}\left(\frac{1}{r}\frac{\partial v_r}{\partial\theta}-\frac{2}{r}\frac{\partial v_r}{\partial\theta}+\frac{v_\theta}{r}\right)\right]$$

$$= \mu \left[\frac{\partial^2}{\partial \theta^2} \left(-\frac{\Omega}{r} \right) + \frac{\partial^2}{\partial r^2} \left(-r \left(-\frac{1}{r} \frac{\partial v_r}{\partial \theta} + \frac{\partial v_\theta}{\partial r} + \frac{v_\theta}{r} - \frac{v_\theta}{r} \right) \right) + \frac{\partial}{\partial r} \left(-\frac{1}{r} \frac{\partial v_r}{\partial \theta} + \frac{v_\theta}{r} + \frac{\partial v_\theta}{\partial r} - \frac{\partial v_\theta}{\partial r} \right) \right]$$

$$= \mu \left[\frac{\partial^2}{\partial \theta^2} \left(-\frac{\Omega}{r} \right) + \frac{\partial^2}{\partial r^2} \left(-r \left(\Omega - \frac{v_\theta}{r} \right) \right) + \frac{\partial}{\partial r} \left(\Omega - \frac{\partial v_\theta}{\partial r} \right) \right]$$

$$= \mu \left[\frac{\partial^2}{\partial \theta^2} \left(-\frac{\Omega}{r} \right) + \frac{\partial^2}{\partial r^2} \left(-r\Omega + v_\theta \right) + \frac{\partial \Omega}{\partial r} - \frac{\partial^2 v_\theta}{\partial r^2} \right]$$

$$= \mu \left[\frac{\partial^2}{\partial \theta^2} \left(-\frac{\Omega}{r} \right) + \frac{\partial^2 \left(-r\Omega \right)}{\partial r^2} + \frac{\partial^2 v_\theta}{\partial r^2} + \frac{\partial \Omega}{\partial r} - \frac{\partial^2 v_\theta}{\partial r^2} \right]$$

$$= \mu \left[\frac{\partial^2}{\partial \theta^2} \left(-\frac{\Omega}{r} \right) + \frac{\partial^2 \left(-r\Omega \right)}{\partial r^2} + \frac{\partial \Omega}{\partial r} \right]$$

$$= \mu \left[-\frac{\partial^2}{\partial \theta^2} \left(\frac{\Omega}{r} \right) - \frac{\partial^2 \left(r\Omega \right)}{\partial r^2} + \frac{\partial \Omega}{\partial r} \right] \tag{A.8}$$

Final Equation:

$$\rho \left[-\frac{\partial \left(\Omega r \right)}{\partial t} - v_r \frac{\partial \left(\Omega r \right)}{\partial r} - v_\theta \frac{\partial \Omega}{\partial \theta} - \Omega \frac{\partial v_\theta}{\partial \theta} - \Omega r \frac{\partial v_r}{\partial r} \right] =$$
$$\rho \left(\frac{\partial g_r}{\partial \theta} - g_\theta - r \frac{\partial g_\theta}{\partial r} \right) + \mu \left[-\frac{\partial^2 \left(\Omega / r \right)}{\partial \theta^2} - \frac{\partial^2 \left(r\Omega \right)}{\partial r^2} + \frac{\partial \Omega}{\partial r} \right] \tag{A.9}$$

Stream Function:

$$v_r = \frac{1}{r} \frac{\partial \Psi}{\partial \theta} \qquad\qquad v_\theta = -\frac{\partial \Psi}{\partial r} \tag{A.10}$$

Stream Function Equation:

$$\frac{\partial^2\Psi}{\partial r^2}+\frac{1}{r}\frac{\partial\Psi}{\partial r}+\frac{1}{r^2}\frac{\partial^2\Psi}{\partial\theta^2}+\Omega=0 \tag{A.11}$$

Vorticity Transport Equation:

$$\rho\left[-\frac{\partial\left(\Omega r\right)}{\partial t}-\frac{1}{r}\frac{\partial\Psi}{\partial\theta}\frac{\partial\left(\Omega r\right)}{\partial r}+\frac{\partial\Psi}{\partial r}\frac{\partial\Omega}{\partial\theta}+\Omega\frac{\partial^2\Psi}{\partial\theta\partial r}+\frac{\Omega}{r}\frac{\partial\Psi}{\partial\theta}-\Omega\frac{\partial^2\Psi}{\partial r\partial\theta}\right]=$$
$$\rho\left(\frac{\partial g_r}{\partial\theta}-g_\theta-r\frac{\partial g_\theta}{\partial r}\right)+\mu\left[-\frac{\partial^2\left(\Omega/r\right)}{\partial\theta^2}-\frac{\partial^2\left(r\Omega\right)}{\partial r^2}+\frac{\partial\Omega}{\partial r}\right]$$

$$\rho\left[-\frac{\partial\left(\Omega r\right)}{\partial t}+\frac{\partial\Psi}{\partial r}\frac{\partial\Omega}{\partial\theta}-\frac{\partial\Psi}{\partial\theta}\frac{\partial\Omega}{\partial r}\right]=\rho\left(\frac{\partial g_r}{\partial\theta}-g_\theta-r\frac{\partial g_\theta}{\partial r}\right)-$$
$$\mu r\left[\frac{\partial^2\Omega}{\partial r^2}+\frac{1}{r}\frac{\partial\Omega}{\partial r}+\frac{1}{r^2}\frac{\partial^2\Omega}{\partial\theta^2}\right] \tag{A.12}$$

Neglecting gravity and assuming steady state:

$$\rho\left[\frac{\partial\Psi}{\partial r}\frac{\partial\Omega}{\partial\theta}-\frac{\partial\Psi}{\partial\theta}\frac{\partial\Omega}{\partial r}\right]=-\mu r\left[\frac{\partial^2\Omega}{\partial r^2}+\frac{1}{r}\frac{\partial\Omega}{\partial r}+\frac{1}{r^2}\frac{\partial^2\Omega}{\partial\theta^2}\right] \tag{A.13}$$

$$\left[\frac{\partial^2\Omega}{\partial r^2}+\frac{1}{r}\frac{\partial\Omega}{\partial r}+\frac{1}{r^2}\frac{\partial^2\Omega}{\partial\theta^2}\right]-\frac{\rho}{\mu r}\left[\frac{\partial\Psi}{\partial\theta}\frac{\partial\Omega}{\partial r}-\frac{\partial\Psi}{\partial r}\frac{\partial\Omega}{\partial\theta}\right]=0 \tag{A.14}$$

Non-dimensionalizing the equation

$$\left[\frac{\partial^2\Omega}{\partial r^2}+\frac{1}{r}\frac{\partial\Omega}{\partial r}+\frac{1}{r^2}\frac{\partial^2\Omega}{\partial\theta^2}\right]-\frac{\mathrm{Re}}{r}\left[\frac{\partial\Psi}{\partial\theta}\frac{\partial\Omega}{\partial r}-\frac{\partial\Psi}{\partial r}\frac{\partial\Omega}{\partial\theta}\right]=0 \tag{A.15}$$

REFERENCES

1. Babinsky, H., "Aerodynamic Improvements of Paraglider Performance" *American Institute of Aeronautics and Astronautics*, AIAA-99-3148

2. www.nova-wings.com

3. Saad, Y., *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, Boston, 1996.

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

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

7. Ning Qin, D. K. Ludlow and S. T. Shaw, "A matrix-free preconditioned Newton:GMRES method for unsteady Navier–Stokes solutions" *International Journal for Numerical Methods in Fluids*, Vol. 33, pp. 223248, 2000

8. 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.

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

10. http://www.skyfly.cz/zajimavo_e/software03_e.htm

11. Schwarz, H.A., "Ueber einen Grenzbergang durch alternirendes Verfahren" *Viertel-jahrsschrift der Naturforschenden Gesellschaft in Zrich*, 15, pp. 272-286, 1870

12. http://skysports-turkey.com/gallery.html

13. Bertin, J. J., *Aerodynamics for Engineers-Fourth Edition*, Prentice Hall, Upper Saddle River, New Jersey, 2002