A Comparative Study of the SIMPLE and Fractional Step Time Integration Methods for Transient Incompressible Flows

by

Jonathan Hines

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Mechanical Engineering

Waterloo, Ontario, Canada, 2008

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Time integration methods are necessary for the solution of transient flow problems. In recent years, interest in transient flow problems has increased, leading to a need for better understanding of the costs and benefits of various time integration schemes. The present work investigates two common time integration schemes, namely the Semi-Implicit Method for Pressure-Linked Equations (SIMPLE) and the Fractional Step (FS) method.

Three two-dimensional, transient, incompressible flow problems are solved using a cell centered, finite volume code. The three test cases are laminar flow in a lid-driven skewed cavity, laminar flow over a square cylinder, and turbulent flow over a square cylinder. Turbulence is modeled using wall functions and the $k-\varepsilon$ turbulence model with the modifications suggested by Kato and Launder. Solution efficiency as measured by the effort carried out by the flow equation solver and CPU time is examined. Accuracy of the results, generated using the SIMPLE and FS time integration schemes, is analyzed through a comparison of the results with existing experimental and/or numerical solutions.

Both the SIMPLE and FS algorithms are shown to be capable of solving benchmark flow problems with reasonable accuracy. The two schemes differ slightly in their prediction of flow evolution over time, especially when simulating very slowly changing flows. As the time step size decreases, the SIMPLE algorithm computational cost (CPU time) per time step remains approximately constant, while the FS method experiences a reduction in cost per time step. Also, the SIMPLE algorithm is numerically stable for time steps approaching infinity, while the FS scheme suffers from numerical instability if the time step size is too large. As a result, the SIMPLE algorithm is recommended to be used for transient simulations with large time steps or steady state problems while the FS scheme is better suited for small time step solutions, although both time-stepping schemes are found to be most efficient when their time steps are at their maximum stable value.

Acknowledgements

I would like to thank my supervisor, Dr. Fue-Sang Lien, for his valuable guidance and support over the past five years.

I would like to thank Dr. Cécile Devaud and Dr. Sanjeev Bedi for their helpful comments on this work.

I would like to thank Cameron McCarteny and Kun-Jung Hsieh who were great sources of advice and inspiration throughout the construction of this work.

I would like to thank Cameron, K.J., Jonathan, Jee-Whan and Kristen for their help in proofreading this work.

I would like to acknowledge financial support of my research from MITACS, NSERC, the University of Waterloo and Dr. Fue-Sang Lien.

I would also like to acknowledge the support of SHARCNET which provided much of the computational resources needed to complete this research.

I would like to thank my parents who fed the curiosity they saw growing in me and who encouraged me to seek out answers for myself.

Finally, I would like to thank my wife, Kristen Hines, who allowed me to pursue this higher education and who has provided loving support along the way.

Dedication

For Kristen.

Contents

List of Tables

List of Figures

Nomenclature

Roman Characters

$\widehat{A_P}$	modified present cell matrix coefficient
A	matrix equation coefficient
A,B,C,D	wall mid-points of skewed cavity
C_μ	eddy viscosity coefficient
$C_{\varepsilon 1}$	constant from the turbulence dissipation rate equation
$C_{\varepsilon 2}$	constant from the turbulence dissipation rate equation
$C_{D,P}$	drag coefficient due to pressure
C_D	drag coefficient
$C_{L,P}$	lift coefficient due to pressure
C_L	lift coefficient
D	reference length or the diameter of the square cylinder
E	integration constant for smooth walls
e	error
f	frequency
h	flow domain height
$\mathbf{J}$	Jacobian matrix
J	Jacobian
k	turbulent kinetic energy

L	skewed cavity wall length
L_P	lift due to pressure
$\dot{m}$	mass imbalance
n	time step index
n_i	normal vector component in the x_i direction
P	pressure
P_k	turbulence production
Q	lagged source term in discretized transport equation
q	volume flux across a face
Re	Reynolds number
$\widetilde{S_{u_1}}$	u_1 source term modified for wall boundaries
S	turbulent strain rate invariant
S_ϕ	source term for ϕ transport equation
S_P	implicit part of S_ϕ
S_u	explicit part of S_ϕ
St	Strouhal number
t	time
$\widehat{u_i}$	pseudo-velocity component
u	velocity component
u^+	dimensionless velocity for smooth walls
U_{lid}	skewed cavity lid velocity
u_τ	shear velocity
x	Cartesian direction component
y^+	dimensionless wall distance

Greek Symbols

α_ϕ	relaxation coefficient
Δ	discrete difference operator
δ_{ij}	Kronecker delta
Γ_ϕ	diffusion coefficient for ϕ transport equation
κ	von Kármán constant
ν	kinematic viscosity
ν_t	eddy viscosity
Ω	vorticity invariant
Φ	benchmark solution variable
ϕ	unknown variable
σ_ε	constant from the turbulence dissipation rate equation
σ_k	constant from the turbulence kinetic energy equation
τ_w	wall shear stress
ε	turbulet dissipation rate
ξ	curvilinear coordinate

Superscripts and Mathematical Symbols

$\bar{\phi}$	mean value
$\frac{\partial}{\partial n}$	partial derivative in the face normal direction
$\langle\phi\rangle$	ensemble averaged
ϕ'	fluctuating component or correction
$\phi^{\tan}$	tangential component wall boundaries
ϕ^{ε}	term from ε equation

ϕ^k	term from k equation
ϕ^*	non-dimensional form or unconverged solution
ϕ^{n+1}	value taken at the $(n+1)^{\text{th}}$ time step
ϕ^n	value taken at the n^{th} time step
$\widehat{\phi}$	unit vector
CD	cross diffusion
DC	differed correction

Subscripts

∞	free stream condition
ξ_i	partial derivative with respect to ξ_i
i, j	indices
max	maximum absolute value
nb	neighbour cells
rms	root mean square
t	partial derivative with respect to t
x_i	partial derivative with respect to x_i
n	control volume face, north
s	control volume face, south
e	control volume face, east
w	control volume face, west
ne	control volume face, north-east
nw	control volume face, north-west
se	control volume face, south-east

sw	control volume face, south-west
P	control volume index, present
N	control volume index, north
S	control volume index, south
E	control volume index, east
W	control volume index, west
NN	control volume index, north-north
SS	control volume index, south-south
EE	control volume index, east-east
WW	control volume index, west-west
NE	control volume index, north-east
NW	control volume index, north-west
SE	control volume index, south-east
SW	control volume index, south-west

Acronyms

CFD	Computational Fluid Dynamics
DNS	Direct Numerical Simulation
FS	Fractional Step
IMEX	Implicit Explicit
KL	Kato Launder (modified $k - \varepsilon$ turbulence model)
LES	Large Eddy Simulation
QUICK	Quadratic Upstream Interpolation for Convective Kinematics
RANS	Reynolds Averaged Navier-Stokes

SIMPLE	Semi-Implicit Method for Pressure-Linked Equations
SIMPLEC	Semi-Implicit Method for Pressure-Linked Equations, Consistent
SS	Steady State
TDMA	Tri-Diagonal Matrix Algorithm
URANS	Unsteady Reynolds Averaged Navier-Stokes

Chapter 1

Introduction

Computational Fluid Dynamics (CFD) is an important tool in modern engineering design. Until recent years it was thought that it was completely infeasible to determine an entire complicated flow field as part of a design process. This was due in part to the incredible cost of iterative calculation. As an example, one of the first CFD solutions of a fluid flow took 20 hours a week for a year and a half to compute on a mechanical calculator [17]. Although the equations governing Newtonian fluid motion have been known for over a hundred years, for the majority of the time since their discovery, their solutions were limited to a small set of simple systems. With the creation and widespread use of computers came the ability to do iterative calculations with relative ease, making CFD a viable tool for the study of fluids and practical engineering design.

1.1 Motivation

Today, the rising demand for energy, coupled with concerns relating to global climate change have increased interest in alternative energy sources like wind turbines. Wind turbines are mechanical devices designed to extract energy from moving air to allow it to be used for other purposes such as adding electrical energy to the grid. One of the major concerns related to wind turbines is the noise the rotating blades generate and the effect of the noise on the surrounding environment. In fact, noise is one of the primary limiting factors in the creation of new wind farms. Design of quieter wind turbines will help to allow more wind farms to be created and allow the energy demands of the coming years to be met without increasing greenhouse gas emissions.

Prediction through numerical means of noise produced by fluid flow, or aero-acoustic noise, is of interest due to the high cost of producing prototypes and conducting experiments. In order to do this efficiently, CFD code is needed which can accurately predict transient flow fields around turbine blades in a reasonable amount of time. In the past, Reynolds Averaged Navier-Stokes (RANS) turbulent flow solvers have produced steady state flow fields which take into account the effects of turbulence by modeling it statistically. When used for transient simulations, Unsteady RANS (URANS) can be used to resolve only the very largest eddies while approximating the rest of the turbulence with a turbulence model, for example, Ahn et al. [1]. Increasing the amount of resolved turbulence so that the small eddies are modeled and the medium to large eddies are resolved is another approach which provides added solution accuracy. This is referred to as Large Eddy Simulation (LES), ie. Kim and Moin [18]. In an attempt to produce solutions with the efficiency of URANS and the accuracy of LES, hybrid approaches to turbulence modeling have been suggested. Hybrid URANS/LES combines features of both schemes to achieve faster solutions and better accuracy, for example Ahn et al. [1].

In order to predict a transient flow field, the flow solver must divide the solution time into steps and march through time to produce a time dependent solution. Traditionally, RANS codes (and in turn, URANS codes) have used implicit methods which solve all the flow equations simultaneously for each time step which are variants of the Semi-Implicit Method for Pressure-Linked Equations (SIMPLE) algorithm, proposed by Patankar and Spalding [31]. Many commercial codes also depend on (implicit) SIMPLE based time stepping, for example, FLUENT [15], STAR-CD [6], and ANSYS-CFX [14]. LES codes on the other hand have traditionally used explicit Fractional Step (FS) time-stepping schemes which rely heavily the information from the previous time step to generate the solution for the next time step. For example, in 1985, Kim and Moin [18] presented a 2 step Adams-Bashforth FS, LES code which simulated the flow over a backwards facing step. In 1991 Le and Moin [19] presented a 4th order Runge-Kutta FS code which allowed larger time steps to be used. Both of these studies were performed on Cartesian, staggard grids. In 1994, Zang et al. [42] studied a LES code which used FS time stepping on a collocated, non-orthogonal grid and in 2004, Mahesh et al. [27] proposed a LES code which operated on an unstructured grid.

Hybrid URANS-LES simulation methodology has recently become of interest in an effort to combine the efficiency of URANS simulations and the accuracy of LES. For example, in 2003, Davidson and Peng [8] separated the flow domain of a hill in a channel into near-wall and off-wall regions which were solved with URANS

and LES approaches respectively. In 2006, Liu and Shih [24] proposed a method of applying a continuous model to a flow domain which could be moved from URANS to LES modes through the use of a resolution control parameter. In 2008, Lien et al. [23] showed that hybrid URANS-LES was more accurate for simulating flow in an urban environment than pure URANS when using the $k - \varepsilon$ turbulence model.

With growing interest in hybrid URANS-LES CFD solutions, the question of which time-stepping scheme is best to use for these simulations has become an important one which appears to still be unanswered.

1.2 Objectives

This study will investigate the SIMPLE and the Fractional Step time-stepping schemes. The SIMPLE time-stepping scheme is a semi-implicit scheme which solves a pressure equation, the velocity equation and any turbulence model equations implicitly. The FS time-stepping scheme solves the pressure and turbulence model equations implicitly and solves the velocity equations explicitly.

The relative costs and benefits of the two time-stepping schemes are of interest, especially their relative efficiencies. Accuracy of the two time-stepping schemes will be analyzed as far as is possible using existing transient benchmark solutions, however few reliable transient test cases exist. The comparison will be accomplished by conducting simulations using codes that are nearly identical, with the exception of the time-stepping scheme. For each of the test cases, existing SIMPLE code will be modified to use the FS time-stepping scheme. Simulations under the same parameters will be conducted to compare the variations between the two time-stepping schemes.

Three test cases of increasing complexity will be completed and analyzed to determine the relative benefits of the two time-stepping schemes.

1. Laminar Flow in a Lid Driven Skewed Cavity
2. Laminar Flow over a Square Cylinder
3. Turbulent Flow over a Square Cylinder

All test cases are incompressible and are two-dimensional due to cost and time restrictions.

The final goal of this study is to determine which time-stepping scheme is best suited for transient CFD simulation. Specifically, a time-stepping scheme must be recommended for use in small time step transient flow simulations. This will lead towards the goal of efficiently using hybrid URANS-LES CFD as a tool for aero-acoustic rotating blade problems.

1.3 Literature Survey

This study investigates three test cases which have been studied both experimentally and numerically in the past. Previous comparisons of CFD time-stepping schemes in the literature are also of interest.

1.3.1 Lid-Driven Skewed Cavity

Lid driven cavities have been studied as a numerical test case ever since 1966 when Burggraf [5] published a numerical study of a square, lid driven cavity. The square cavity is a convenient test case for structured orthogonal mesh codes as it has very simple geometry but can exhibit complex flow structures which are highly Reynolds number dependent.

The skewed cavity exhibits similar geometrical simplicity to the square cavity but it requires a non-orthogonal mesh. Skewed cavity numerical studies were the natural extension of the square cavity test case when non-orthogonal and unstructured meshes began to be studied in depth [11]. The first published study to use the skewed cavity as a test case was in 1992 when Demirdžiü et al. [10] published a benchmark solution of a steady state skewed cavity at skew angles of 45° and 30° on a 320×320 structured grid at Reynolds numbers of 100 and 1000. Many studies since then have used this work as a benchmark, for example, Louaked et al. [25], Shklyar and Arbel [33] and Xu and Zhang [41].

In 2007, a high resolution (512×512) steady state numerical simulation of a lid driven skewed cavity at a variety of skew angles was carried out by Erturk and Dursun [11]. This work provides a tabulated solution for the center-line velocity profiles for each of the skew angles studied. The results reported by Erturk and Dursun [11] for 45° and 30° skew angles and Reynolds numbers of 100 and 1000 are consistent with the results first reported by Demirdžiü et al. [10].

1.3.2 Laminar Flow over a Square Cylinder

This test case has been studied many times both experimentally and numerically. In this study flows of a Reynolds number of 100 are to be studied in line with the Masters Thesis by Thompson [38]. In fact much of the SIMPLE time stepping base code used by Thompson was used for this study, although some of the boundary conditions have been modified. Thompson was interested in studying vortex induced vibrations and used this as a first test case to (successfully) validate his code. Thompson carried out tests on a variety of meshes to produce a range of summary properties.

In 1982, Davis and Moore [9] reported numerical results for this test case and in 1990 Franke et al. [13] published numerical results generated by their SIMPLEC code (which is similar to SIMPLE) for this test case for Reynolds numbers less than or equal to 300.

Experiments were performed by Okajima [30] in 1982 using a water tank. Although he did not run tests at exactly Re = 100, he did report results very close to the target Reynolds number (80 and 150). Okajima provided results from multiple experiments at each tested Reynolds number which provided a better picture of the spread in experimental results for unsteady flow. Interpolating between the results of Okajima provides an approximate range of values for experimental Strouhal numbers.

In 1996 Sohankar et al. [34] published an internal report at Chalmers University in Sweden reporting the results of a numerical study of a rectangular cylinder in cross flow at a variety of angles of attack. This was followed by a paper in 1997 which expanded the results and provided more discussion on previously unpublished experimental work on this test case as well as a compilation of numerical results from between 1982 and 1995 [35]. Simulations for this work were carried out using a transient SIMPLEC based code. The effects of blockage ratio and entrance length were explored producing a range of summary properties for this numerical simulation. In general, Sohankar showed that the results of his simulation were quite sensitive to mesh density and the flow domain dimensions. The results published in the 2007 paper included - results from experiments performed, but never before published, by Norberg in 2006 [35].

The results discussed above have been summarized in Table 1.1. This test case continues to be studied as a benchmark case for new algorithms, for example, Lee et al. [20].

Study		St	$\overline{C_L}$	C_{Lrms}	$\overline{C_{D,P}}$	$C_{D,P_{rms}}$
Thompson [38]	Min	0.150	-8.44×10^{-5}	0.116	1.57	3.03×10^{-3}
	Max	0.150	7.27×10^{-4}	0.132	1.61	3.87×10^{-3}
Sohankar et al. [35]	Min	0.120	–	0.138	1.45	–
	Max	0.155	–	0.156	1.76	–
Sohankar et al. [35]	Exp.	0.145	–	–	–	–
Okajima [30]	Min	0.132	–	–	–	–
	Max	0.144	–	–	–	–
Sohankar et al. [34]	Min	0.135	–	0.082	1.348	1.4×10^{-3}
	Max	0.151	–	0.160	1.462	5.9×10^{-3}
Franke et al. [13]	–	0.154	–	0.191	1.55	–
Davis and Moore [9]	Min	0.148	–	–	1.64	–
	Max	0.153	–	–	1.64	–

Table 1.1: Summary of results presented in literature for laminar flow over a square cylinder.

1.3.3 Turbulent Flow over a Square Cylinder

In 1995 Lyn et al. [26] reported laser-Doppler velocimetry (LDV) experimental results for flow over a square cylinder with a Reynolds number of 21400. These results were used as a benchmark for several workshops and additional numerical studies of unsteady, turbulent flow around a square cylinder [3, 4, 40]. Lyn's results are listed in Table 1.2

St	$\overline{C_D}$
0.132 ± 0.004	2.1

Table 1.2: Experimental results reported by Lyn et al. [26] for turbulent flow over a square cylinder at Re = 21400.

A large number of numerical solutions were generated for this case for the second ERCOFTAC workshop on Direct and Large-Eddy Simulation. In total, 7 groups provided 20 different numerical solutions using a variety of Direct Numerical Simulation (DNS) and LES codes [40]. Table 1.3 was assembled by Voke [40] as a summary of the workshop results.

Further LES results were reported by Murakami and Mochida [28] in 1995 and Bouris and Bergeles [4] in 1999. Their results are listed in Table 1.4.

Study	St	$\overline{C_L}$	C_{Lrms}	$\overline{C_D}$	C_{Drms}
UK1	0.13	-0.02	1.01	2.2	0.14
UK2	0.13	-0.04	1.15	2.3	0.14
UK3	0.13	-0.05	1.02	2.23	0.13
GRO	0.133	0.005	1.45	2.09	0.18
NT7	0.131	-0.05	1.39	2.05	0.12
UOI	0.13	0.04	1.29	2.03	0.18
IS1	0.13	-0.29	1.31	2.041	0.26
IS2	0.13	-0.0066	1.235	2.067	0.15
IS3	0.133	-0.125	1.68	2.79	0.36
TIT	0.131	0.0093	1.39	2.62	0.23
ST2	0.16	0.01	1.26	2.72	0.28
ST3	0.15	-0.005	1.33	2.66	0.27
ST4	0.139	0.012	1.36	2.74	0.29
ST5	0.161	0.009	1.38	2.78	0.28

Table 1.3: Summary of results reported from the second ERCOFTAC workshop on Direct and Large-Eddy Simulation, test case LES2 [40], turbulent flow over a square cylinder at Re = 21400.

Study	St	$\overline{C_D}$
Murakami and Mochida [28]	0.132	2.09
Bouris and Bergeles [4]	0.134	2.18

Table 1.4: Results of the LES studies of Murakami and Mochida [28] and Bouris and Bergeles [4] for turbulent flow over a square cylinder at Re = 21400.

URANS simulations of this test case were carried out by Thompson [38] using the same code as the current study. This study uses wall functions and a version of the $k-\varepsilon$ model with the modifications proposed by Kato and Launder [16] in their study of bluff bodies (referred to as the KL turbulence model). By varying the mesh density Thompson [38] generated a range of summary properties which could be compared to those arising from the ERCOFTAC results. It should be noted that Thompson's coarse mesh has the same wall geometry and grid density as the mesh used in this study. Thompson's results are summarized in Table 1.5.

	St	$\overline{C_L}$	C_{Lrms}	$\overline{C_D}$	C_{Drms}
Coarse Mesh	0.141	4.21×10^{-4}	1.09	2.15	0.048
Fine Mesh	0.143	1.71×10^{-3}	1.45	2.29	0.1188

Table 1.5: Summary of results presented by Thompson [38] for turbulent flow over a square cylinder at Re = 21400.

In order to quantify the effects of using different the turbulence models, Bosch and Rodi [3], Kato and Launder [16] and Franke and Rodi [12] simulated this test case geometry at a similar Reynolds number of 22000, using a variety of turbulence models in conjunction with different near wall treatments. Turbulence models studied included the KL model [16, 3], the standard $k-\varepsilon$ model [16, 3, 12] and the Reynolds Stress Equation (RSE) model [12]. Solid wall conditions were applied through either wall functions (WF) [16, 3, 12] or the two-layer approach (TL) [3, 12]. Their results are summarized in Table 1.6.

1.3.4 Comparisons of Different Time Stepping Methods

Very few studies have attempted to compare the SIMPLE and FS time-stepping schemes using a finite volume code. A study of different time stepping methods for finite element solutions of the incompressible Navier-Stokes equations was presented by Turek [39] in 1996 which compared Backwards Euler, Crank-Nicolson and Fractional Step (second order) methods for discretizing the time derivative. Turek [39] also compared a variety of options for solving the non-linear advection terms in the momentum equations, namely, in his terms, fully non-linear implicit, linear extrapolated semi-implicit, constant extrapolated semi-implicit and fully explicit. The study also explored the role of the equation solver and made use of adaptive time step adjustment methods.

Study	Turb. Mdl.	Walls	St	C_{Lrms}	$\overline{C_D}$	C_{Drms}
Bosch and Rodi [3]	$k-\varepsilon$	WF	0.126	0.050	1.62	0.0003
Kato and Launder [16]	$k-\varepsilon$	WF	0.127	0.100	1.66	-
Franke and Rodi [12]	$k-\varepsilon$	TL	0.124	0.228	1.79	0.0
Bosch and Rodi [3]	$k-\varepsilon$	TL	0.122	0.178	1.75	0.0012
Bosch and Rodi [3]	KL	WF	0.146	1.012	2.11	0.0325
Kato and Launder [16]	KL	WF	0.145	0.820	2.05	0.0212
Franke and Rodi [12]	RSE	WF	0.136	1.49	2.15	0.270
Franke and Rodi [12]	RSE	TL	0.159	1.30	2.43	0.055

Table 1.6: Summary of URANS results presented by Bosch and Rodi [3] and Kato and Launder [16] and Franke and Rodi [12] for turbulent flow over a square cylinder at Re = 22000.

All combinations of discretization schemes and non-linear term approaches were tested in numerical simulations of flow through a venturi pump and vortex shedding from an inclined plate in cross flow. Care was taken to insure that the implementations of the various schemes were optimized to the same degree and all simulations were implemented in Fortran 77 and computed on Silicon Graphics and SUN workstations of "similar performance ratings" [39]. Results including the pressure difference across the inclined plate and total CPU time allowed detailed discussion and analysis of the various schemes.

Turek [39] recommended that second order time-stepping schemes should be used for transient flows because Backwards Euler discretization, fully explicit and semi-implicit advection term approaches required much smaller time step sizes to maintain accuracy and stability, reducing their efficiency to the same level as second order schemes at a larger time step. CPU times were not reported for first order schemes. Of the second order schemes, Fractional Step was found to be the most efficient as the Crank-Nicolson scheme was more prone to instability as the time step increased in size. In general, the transient solutions reported varied widely in frequency and amplitude, depending on the time-stepping scheme. In some cases, solutions were completely non-physical, which speaks to the importance of selecting an appropriate time-stepping scheme and ensuring that it is operating within reasonable operating parameters. The study also strongly advocated adaptive time step control, citing that correctly implemented adaptive time step control could lead to much greater accuracy while at the same time ensuring numerical stability [39]. It should be noted that the recommendations of Turek exclude cases in which

very small time steps are necessary such as in the case of aero-acoustic problems.

1.4 Outline

The remainder of this work will focus on formulating the code used in this work and then examining the results obtained from test cases as a method of comparing the two time-stepping schemes. Finally, conclusions and recommendations will be made concerning the relative benefits of the SIMPLE and FS time stepping methods. The discussion will be arranged as follows:

- Chapter 2: Numerical Methods
- Chapter 3: Laminar Flow in a Lid Driven Skewed Cavity
- Chapter 4: Laminar Flow over a Square Cylinder
- Chapter 5: Turbulent Flow over a Square Cylinder
- Chapter 6: Conclusions

Chapter 2

Numerical Methods

The code used for this study was a modified version of the finite volume RANS code, STREAM, developed by Lien and Leschziner [22]. The original STREAM code employed the SIMPLE algorithm first proposed by Patankar and Spalding [31]. A second version of the STREAM employs the Fractional Step time-stepping method following the framework described by Zang et al. [42] who extended the work of Kim and Moin [18] to work on a collocated grid. The work of Kim and Moin [18] was based on earlier work by Chorin [7] and Teman [37]. STREAM solves the governing equations on a curvilinear, non-orthogonal, co-located grid, using the interpolation scheme first proposed by Rhie and Chow [32].

Initial modifications to the SIMPLE version of the STREAM code were carried out as part of the study detailed in the Masters thesis by Thompson [38] in order to predict vortex shedding caused by an infinite square cylinder in cross flow. Parallel modifications to the Fractional Step version of the STREAM code have been carried out as part of this study in order to compare the FS and SIMPLE algorithms.

2.1 Governing Equations

The flows of interest in this study are assumed to be incompressible Newtonian fluids which is a standard assumption for most gases with a Mach number of less than 0.3. All studied flows are time dependent. Two of them exhibit laminar flow while the third is known to be turbulent. Incompressible transient flows can be fully described by a momentum equation and a continuity equation. For this study, the momentum equation is

$$\frac{\partial u_i}{\partial t} + \frac{\partial u_i u_j}{\partial x_j} = -\frac{1}{\rho}\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\nu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right] \tag{2.1}$$

and the continuity equation is

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{2.2}$$

2.1.1 Non-Dimensionalization of Governing Equations

Variables in the STREAM code were non-dimensionalized as follows.

$$t^* = \frac{tu_\infty}{D}, x^* = \frac{x_i}{D}, u_i^* = \frac{u}{u_\infty}, \tag{2.3}$$
$$P^* = \frac{P}{\rho u_\infty^2}, \nu^* = \frac{\nu}{u_\infty D} = \frac{1}{\mathrm{Re}},$$

Upon the substitution of these non-dimensional parameters into the governing equations, Equations 2.1 and 2.2 become

$$\frac{\partial u_i^*}{\partial t^*} + \frac{\partial}{\partial x_j^*}\left(u_i^* u_j^*\right) = -\frac{\partial P^*}{\partial x_i^*} + \frac{\partial}{\partial x_j^*}\left[\frac{1}{\mathrm{Re}}\left(\frac{\partial u_i^*}{\partial x_j^*} + \frac{\partial u_j^*}{\partial x_i^*}\right)\right] \tag{2.4}$$

and

$$\frac{\partial u_i^*}{\partial x_i^*} = 0 \tag{2.5}$$

From this point on the superscript * will be dropped for the sake of simplicity but all following values can be assumed to be dimensionless unless otherwise stated.

2.2 Turbulence

In the case of turbulent flow, a choice must be made of whether or not the computer resources available are adequate to resolve all of the turbulent fluctuations present in the flow. As the time scale and length scale of the smallest turbulent fluctuations are typically much smaller than the time and length scales for the flow in question, it is often advantageous to decompose the quantities of interest into ensemble averaged and fluctuating components. That is,

$$\phi = \langle\phi\rangle + \phi' \tag{2.6}$$

where $\langle\phi\rangle$ is the ensemble averaged value of ϕ and ϕ' denotes the random fluctuating component of ϕ. The ensemble average is defined as

$$\langle\phi\rangle = \frac{1}{N}\sum_{i=1}^{N}\phi_i \tag{2.7}$$

where N is a large number, which represents the number of physical tests sampled, and ϕ represents a sampled value at one particular time and space coordinate in each of the sampled tests. Substituting Equation 2.6 into Equations 2.1 and 2.2 yields

$$\frac{\partial \langle u_i \rangle}{\partial t} + \frac{\partial}{\partial x_j}\left(\langle u_i \rangle \langle u_j \rangle + \left\langle u_i' u_j' \right\rangle\right) = -\frac{\partial \langle P \rangle}{\partial x_j} + \frac{\partial}{\partial x_j}\left[\nu\left(\frac{\partial \langle u_i \rangle}{\partial x_j} + \frac{\partial \langle u_j \rangle}{\partial x_i}\right)\right] \tag{2.8}$$

and

$$\frac{\partial \langle u_i \rangle}{\partial x_i} = 0 \tag{2.9}$$

respectively.

The primary difference between Equations 2.1 and 2.8, besides the change from actual values to ensemble averaged values in most of the terms, is the addition of the Reynolds stress tensor $\left\langle u_i' u_j' \right\rangle$, a new term which must be modeled for the system of equations to be closed.

2.2.1 Turbulence Models

One of the most common ways of modeling the Reynolds stress is using the Boussinesq approximation

$$-\left\langle u_i' u_j' \right\rangle = \nu_t\left(\frac{\partial \langle u_i \rangle}{\partial x_j} + \frac{\partial \langle u_j \rangle}{\partial x_i}\right) - \frac{2}{3}\delta_{ij}k \tag{2.10}$$

where k is the turbulent kinetic energy, that is

$$k = \frac{1}{2}\langle u_i' u_i' \rangle \tag{2.11}$$

This allows Equation 2.8 to be written in the convenient form,

$$\frac{\partial \langle u_i \rangle}{\partial t} + \frac{\partial}{\partial x_j}\left(\langle u_i \rangle \langle u_j \rangle\right) = -\frac{\partial}{\partial x_i}\left(\langle P \rangle + \frac{2}{3}k\right) + \frac{\partial}{\partial x_j}\left[(\nu + \nu_t)\left(\frac{\partial \langle u_i \rangle}{\partial x_j} + \frac{\partial \langle u_j \rangle}{\partial x_i}\right)\right] \tag{2.12}$$

The k transport equation is

$$\frac{\partial \langle k \rangle}{\partial t} + \frac{\partial \langle u_j \rangle \langle k \rangle}{\partial x_j} = P_k - \langle \varepsilon \rangle + \frac{\partial}{\partial x_j}\left(\frac{\langle \nu_t \rangle}{\sigma_k}\frac{\partial \langle k \rangle}{\partial x_j}\right) \tag{2.13}$$

and the turbulence dissipation rate, ε, is obtained by solving

$$\frac{\partial \langle \varepsilon \rangle}{\partial t} + \frac{\partial \langle u_j \rangle \langle \varepsilon \rangle}{\partial x_j} = \frac{\langle \varepsilon \rangle}{\langle k \rangle}\left(C_{\varepsilon 1} P_k - C_{\varepsilon 2}\langle \varepsilon \rangle\right) + \frac{\partial}{\partial x_j}\left(\frac{\langle \nu_t \rangle}{\sigma_\varepsilon}\frac{\partial \langle \varepsilon \rangle}{\partial x_j}\right) \tag{2.14}$$

where the eddy viscosity is modeled as

$$\langle \nu_t \rangle = C_\mu \frac{\langle \varepsilon \rangle^2}{\langle k \rangle} \tag{2.15}$$

Turbulence production takes the form

$$P_k = -\left\langle u_i' u_j' \right\rangle \frac{\partial \langle u_i \rangle}{\partial x_j} = C_\mu \langle \varepsilon \rangle S^2 \tag{2.16}$$

where the turbulent strain rate invariant is

$$S = \frac{\langle k \rangle}{\langle \varepsilon \rangle} \sqrt{\frac{1}{2} \left[\frac{\partial \langle u_i \rangle}{\partial x_j} + \frac{\partial \langle u_j \rangle}{\partial x_i} \right]^2} \tag{2.17}$$

So Equation 2.16 becomes

$$P_k = C_\mu \frac{\langle k \rangle^2}{\langle \varepsilon \rangle} \left[\frac{1}{2} \left[\frac{\partial \langle u_i \rangle}{\partial x_j} + \frac{\partial \langle u_j \rangle}{\partial x_i} \right]^2 \right] \tag{2.18}$$

Finally to provide better results in bluff body flow, the Kato-Launder [16] modification to the $k - \varepsilon$ equations is used. That is,

$$P_k = C_\mu \langle \varepsilon \rangle S \Omega \tag{2.19}$$

where

$$\Omega = \frac{\langle k \rangle}{\langle \varepsilon \rangle} \sqrt{\frac{1}{2} \left[\frac{\partial \langle u_i \rangle}{\partial x_j} - \frac{\partial \langle u_j \rangle}{\partial x_i} \right]^2} \tag{2.20}$$

Thus, Equation 2.19 becomes

$$P_k = C_\mu \frac{\langle k \rangle^2}{\langle \varepsilon \rangle} \sqrt{\frac{1}{2} \left[\frac{\partial \langle u_i \rangle}{\partial x_j} + \frac{\partial \langle u_j \rangle}{\partial x_i} \right]^2 \left[\frac{\partial \langle u_i \rangle}{\partial x_j} - \frac{\partial \langle u_j \rangle}{\partial x_i} \right]^2} \tag{2.21}$$

The addition of the term Ω improves the accuracy of the $k - \varepsilon$ model for bluff body flows because the $k - \varepsilon$ model typically over predicts turbulence production in stagnation regions for which the $\Omega \approx 0$. In regions of simple shear, $\Omega \approx S$, allowing the model to act like the standard $k - \varepsilon$ model. Table 2.1 lists the standard values of the constants in the k and ε equations. From now on the ensemble averaged notation, <>, will be dropped for simplicity but all values can be assumed to be ensemble averaged unless otherwise stated.

C_μ	$C_{\varepsilon 1}$	$C_{\varepsilon 2}$	σ_k	σ_ε
0.09	1.44	1.92	1.0	1.3

Table 2.1: Empirical turbulence model constants [22].

2.2.2 Wall Functions

Turbulence close to a solid wall is difficult to model as the gradients of the various flow properties tends to be quite steep and thus requires a large number of control volumes to resolve them. To increase the efficiency of the solution, near wall turbulent effects are modeled with the use of wall functions. For high Reynolds number flows, the boundary layer can be separated into the inner and outer regions. Non-dimensionalizing the distance from the wall into the flow, denoted by y^+, the inner region is defined as $0 < y^+ < 11.6$ and the outer (or log law) region is defined as $y^+ > 11.6$. The definition of y^+ is

$$y^+ = \frac{y u_\tau}{\nu} = \frac{y C_\mu^{1/4} k^{1/2}}{\nu} \tag{2.22}$$

where the shear velocity is

$$\begin{aligned} u_\tau &= \sqrt{\frac{\tau_w}{\rho}} \text{ by definition} \\ u_\tau &= C_\mu^{1/4} k^{1/2} \text{ from log – law} \end{aligned} \tag{2.23}$$

and the shear stress at the wall is

$$\begin{aligned} \tau_w &= \frac{\mu u}{y} \text{ for } y^+ < 11.6 \\ \tau_w &= \frac{\kappa C_\mu^{1/4} k^{1/2} \rho u}{\ln E y^+} \text{ for } y^+ > 11.6 \end{aligned} \tag{2.24}$$

where $\kappa = 0.42$ is the von Kármán constant and $E = 9.8$ is the integration constant for smooth walls. Thus the velocity profile will be accurately approximated for a turbulent boundary layer as

$$u^+ = \frac{u}{u_\tau} = \frac{\ln E y^+}{\kappa} \tag{2.25}$$

as well as turbulent kinetic energy,

$$k = C_\mu^{-1/4} u_\tau^2 \tag{2.26}$$

and the dissipation rate,

$$\varepsilon = \frac{u_\tau^3}{\kappa y} = \frac{C_\mu^{3/4} k^{3/2}}{\kappa y} \ . \tag{2.27}$$

2.3 Transformation to Curvilinear Coordinates

In most practical numerical simulations of fluid flows, bounding walls and other such surfaces are not shaped in such a way that they can mapped by an orthogonal grid so that the solid surfaces and boundaries line up with the cell boundaries. On the other hand the most convenient way to discretize the governing equations is in using a uniform, orthogonal grid. The governing equations described in the previous sections have been written in Cartesian space, denoted by x_i i.e. (x_1, x_2, x_3). This difference between numerical and physical convenience can be bridged by mapping the governing equations from the Cartesian coordinate system to an arbitrary, curvilinear coordinate system, ξ_i i.e. (ξ_1, ξ_2, ξ_3) which will follow the geometry of the physical system to be modeled as suggested by, for example, Anderson [2].

First it is important to note that the momentum, k and ε equations can be written in the form

$$\frac{\partial \phi}{\partial t} + \frac{\partial u_j \phi}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\Gamma_\phi \frac{\partial \phi}{\partial x_j}\right] + S_\phi \tag{2.28}$$

The definition of the source term S_ϕ and the diffusion coefficient Γ_ϕ is determined by the variable represented by ϕ. Table 2.2 lists the possible variables represented by ϕ and the source terms related to them.

Equation	ϕ	Γ_ϕ	S_ϕ
2.12	u_i	$\nu + \nu_t$	$-\frac{\partial}{\partial x_i}\left(P + \frac{2}{3}k\right) + \frac{\partial}{\partial x_j}\left[\left(\nu + \nu_t\right)\frac{\partial u_j}{\partial x_i}\right]$
2.13	k	$\frac{\nu_t}{\sigma_k}$	$P_k - \varepsilon$
2.14	ε	$\frac{\nu_t}{\sigma_\varepsilon}$	$\frac{\varepsilon}{k}\left(C_{\varepsilon 1} P_k - C_{\varepsilon 2}\varepsilon\right)$

Table 2.2: Definitions of source terms and diffusion coefficients of governing equations

Knowing that $\phi = \phi(x_i, t)$ and that $x_i = x_i(\xi_j)$ the chain rule can be applied to transform the differentials from the Cartesian to the curvilinear system. i.e.

$$dx_i = \frac{\partial x_i}{\partial \xi_j} d\xi_j = \mathbf{J}_{ij} d\xi_j \tag{2.29}$$

or

$$d\xi_i = \frac{\partial \xi_i}{\partial x_j} dx_j \tag{2.30}$$

where $\mathbf{J}$ is the Jacobian matrix, with its ij^{th} component being

$$\mathbf{J}_{ij} = \left(\frac{\partial x_i}{\partial \xi_j} \right) \tag{2.31}$$

From this point on, a subscript spatial or time coordinate variable will denote a partial derivative. For example

$$\frac{\partial x_i}{\partial \xi_j} = x_{i\xi_j} \tag{2.32}$$

Applying the chain rule expansion to Equation 2.28 yields

$$\phi_t + (u_j \phi)_{\xi_i} \xi_{i x_j} = \left(\Gamma_\phi \phi_{x_j} \right)_{\xi_i} \xi_{i x_j} + S_\phi \tag{2.33}$$

By comparing Equations 2.29 and 2.30 it can be shown that

$$\xi_{i x_j} = \left(\mathbf{J}^{-1} \right)_{ij} \tag{2.34}$$

Substituting Equation 2.34 into Equation 2.33 provides

$$\phi_t + (u_j \phi)_{\xi_i} \left(\mathbf{J}^{-1} \right)_{ij} = \left(\Gamma_\phi \phi_{x_j} \right)_{\xi_i} \left(\mathbf{J}^{-1} \right)_{ij} + S_\phi \tag{2.35}$$

As we are interested in two-dimensional simulation of flows in this study, Equation 2.35 will be expanded for the two dimensional case of $i = 1, 2$ and $j = 1, 2$. In this case, the Jacobian matrix can be written as

$$\mathbf{J} = \begin{bmatrix} x_{1\xi_1} & x_{1\xi_2} \\ x_{2\xi_1} & x_{2\xi_2} \end{bmatrix} \tag{2.36}$$

This Jacobian matrix can be easily inverted by finding its adjugate and dividing by its determinant. That is,

$$\mathbf{J}^{-1} = \frac{1}{J} \begin{bmatrix} x_{2\xi_2} & -x_{1\xi_2} \\ -x_{2\xi_1} & x_{1\xi_1} \end{bmatrix} = \begin{bmatrix} \xi_{1x_1} & \xi_{1x_2} \\ \xi_{2x_1} & \xi_{2x_2} \end{bmatrix} \tag{2.37}$$

where the Jacobian, J, (not to be confused with the Jacobian matrix, $\mathbf{J}$) is

$$J = \det(\mathbf{J}) = x_{1\xi_1} x_{2\xi_2} - x_{1\xi_2} x_{2\xi_1} \tag{2.38}$$

Expanding Equation 2.33 into 2 dimensions provides

$$\begin{aligned} \phi_t &+ \\ (u_1\phi)_{\xi_1}\left(\mathbf{J}^{-1}\right)_{11} &+ (u_2\phi)_{\xi_1}\left(\mathbf{J}^{-1}\right)_{12} + \\ (u_1\phi)_{\xi_2}\left(\mathbf{J}^{-1}\right)_{21} &+ (u_2\phi)_{\xi_2}\left(\mathbf{J}^{-1}\right)_{22} = \\ (\Gamma_\phi\phi_{x_1})_{\xi_1}\left(\mathbf{J}^{-1}\right)_{11} &+ (\Gamma_\phi\phi_{x_2})_{\xi_1}\left(\mathbf{J}^{-1}\right)_{12} + \\ (\Gamma_\phi\phi_{x_1})_{\xi_2}\left(\mathbf{J}^{-1}\right)_{21} &+ (\Gamma_\phi\phi_{x_2})_{\xi_2}\left(\mathbf{J}^{-1}\right)_{22} + \\ S_\phi & \end{aligned} \tag{2.39}$$

Applying the chain rule to the remaining x_i partial derivatives provides

$$\begin{aligned} \phi_{x_1} &= \phi_{\xi_1}\xi_{1_{x_1}} + \phi_{\xi_2}\xi_{2_{x_1}} = \phi_{\xi_1}\left(\mathbf{J}^{-1}\right)_{11} + \phi_{\xi_2}\left(\mathbf{J}^{-1}\right)_{21} \\ \phi_{x_2} &= \phi_{\xi_1}\xi_{1_{x_2}} + \phi_{\xi_2}\xi_{2_{x_2}} = \phi_{\xi_1}\left(\mathbf{J}^{-1}\right)_{12} + \phi_{\xi_2}\left(\mathbf{J}^{-1}\right)_{22} \end{aligned} \tag{2.40}$$

Substituting Equation 2.40 into Equation 2.39 provides

$$\begin{aligned} \phi_t &+ \\ (u_1\phi)_{\xi_1}\left(\mathbf{J}^{-1}\right)_{11} &+ (u_2\phi)_{\xi_1}\left(\mathbf{J}^{-1}\right)_{12} + \\ (u_1\phi)_{\xi_2}\left(\mathbf{J}^{-1}\right)_{21} &+ (u_2\phi)_{\xi_2}\left(\mathbf{J}^{-1}\right)_{22} = \\ \left(\Gamma_\phi\phi_{\xi_1}\left(\mathbf{J}^{-1}\right)_{11} + \Gamma_\phi\phi_{\xi_2}\left(\mathbf{J}^{-1}\right)_{21}\right)_{\xi_1}\left(\mathbf{J}^{-1}\right)_{11} &+ \left(\Gamma_\phi\phi_{\xi_1}\left(\mathbf{J}^{-1}\right)_{12} + \Gamma_\phi\phi_{\xi_2}\left(\mathbf{J}^{-1}\right)_{22}\right)_{\xi_1}\left(\mathbf{J}^{-1}\right)_{12} + \\ \left(\Gamma_\phi\phi_{\xi_1}\left(\mathbf{J}^{-1}\right)_{11} + \Gamma_\phi\phi_{\xi_2}\left(\mathbf{J}^{-1}\right)_{21}\right)_{\xi_2}\left(\mathbf{J}^{-1}\right)_{21} &+ \left(\Gamma_\phi\phi_{\xi_1}\left(\mathbf{J}^{-1}\right)_{12} + \Gamma_\phi\phi_{\xi_2}\left(\mathbf{J}^{-1}\right)_{22}\right)_{\xi_2}\left(\mathbf{J}^{-1}\right)_{22} + \\ S_\phi & \end{aligned} \tag{2.41}$$

Combining terms provides

$$\begin{aligned} &\phi_t + \\ &\left(u_1\phi\left(\mathbf{J}^{-1}\right)_{11} + u_2\phi\left(\mathbf{J}^{-1}\right)_{12}\right)_{\xi_1} + \\ &\left(u_1\phi\left(\mathbf{J}^{-1}\right)_{21} + u_2\phi\left(\mathbf{J}^{-1}\right)_{22}\right)_{\xi_2} = \\ &\left(\Gamma_\phi\phi_{\xi_1}\left(\mathbf{J}^{-1}\right)_{11}^2 + \Gamma_\phi\phi_{\xi_2}\left(\mathbf{J}^{-1}\right)_{21}\left(\mathbf{J}^{-1}\right)_{11} + \Gamma_\phi\phi_{\xi_1}\left(\mathbf{J}^{-1}\right)_{12}^2 + \Gamma_\phi\phi_{\xi_2}\left(\mathbf{J}^{-1}\right)_{22}\left(\mathbf{J}^{-1}\right)_{12}\right)_{\xi_1} + \\ &\left(\Gamma_\phi\phi_{\xi_1}\left(\mathbf{J}^{-1}\right)_{11}\left(\mathbf{J}^{-1}\right)_{21} + \Gamma_\phi\phi_{\xi_2}\left(\mathbf{J}^{-1}\right)_{21}^2 + \Gamma_\phi\phi_{\xi_1}\left(\mathbf{J}^{-1}\right)_{12}\left(\mathbf{J}^{-1}\right)_{22} + \Gamma_\phi\phi_{\xi_2}\left(\mathbf{J}^{-1}\right)_{22}^2\right)_{\xi_2} + \\ &S_\phi \end{aligned} \tag{2.42}$$

Substituting Equation 2.37 into Equation 2.42 and rearranging the terms provides

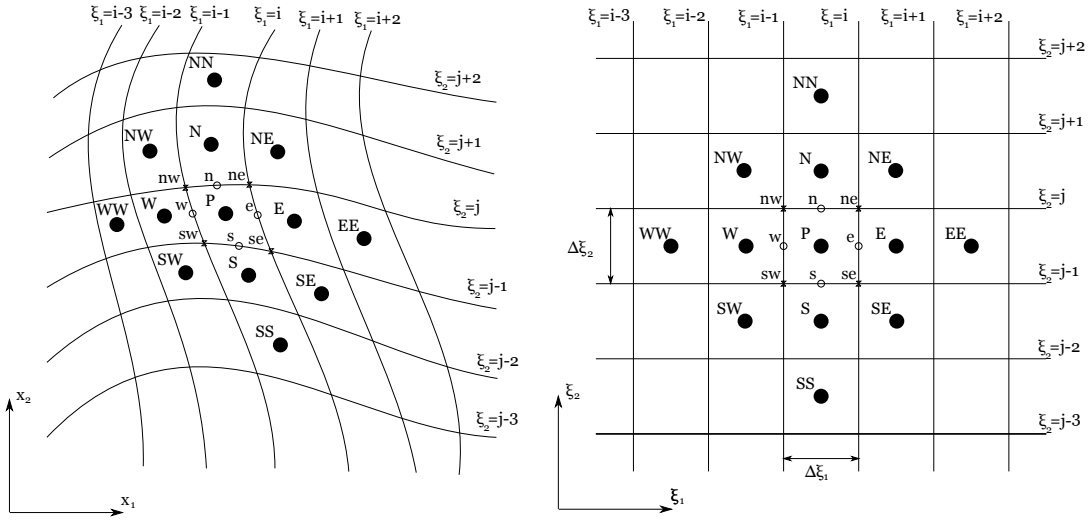
the generic transport equation for two dimensional curvilinear space.

$$\begin{aligned} J\phi_t &+ \\ \left(u_1\phi x_{2\xi_2} - u_2\phi x_{1\xi_2}\right)_{\xi_1} &+ \left(u_2\phi x_{1\xi_1} - u_1\phi x_{2\xi_1}\right)_{\xi_2} \\ -\frac{1}{J}\left(\Gamma_\phi\phi_{\xi_1}x_{2\xi_2}^2 + \Gamma_\phi\phi_{\xi_1}x_{1\xi_2}^2\right)_{\xi_1} &- \frac{1}{J}\left(\Gamma_\phi\phi_{\xi_2}x_{2\xi_1}^2 + \Gamma_\phi\phi_{\xi_2}x_{1\xi_1}^2\right)_{\xi_2} = \\ -\frac{1}{J}\left(\Gamma_\phi\phi_{\xi_2}x_{2\xi_1}x_{2\xi_2} + \Gamma_\phi\phi_{\xi_2}x_{1\xi_1}x_{1\xi_2}\right)_{\xi_1} &- \frac{1}{J}\left(\Gamma_\phi\phi_{\xi_1}x_{2\xi_2}x_{2\xi_1} + \Gamma_\phi\phi_{\xi_1}x_{1\xi_2}x_{1\xi_1}\right)_{\xi_2} \\ &+ JS_\phi \end{aligned} \tag{2.43}$$

The conversion of JS_ϕ to curvilinear coordinates is dependent on the variable assigned to ϕ and is exemplified in Appendix A for $\phi = u_1$ and $\phi = u_2$.

2.4 Finite Volume Method

In order to solve practical flow problems, a curvilinear coordinate system will be chosen such that each control volume or cell in the coordinate system is a square with unit length sides and such that the cell faces are defined by a constant value of either ξ_1 or ξ_2. This is shown in Figure 2.1.



(a) x_i space

(b) ξ_i space

Figure 2.1: Finite volume grid

Integrating Equation 2.43 over the P control volume shown in Figure 2.1 provides

$$\begin{gathered} J\Delta\xi_1\Delta\xi_2\phi_t + \\ \Delta\xi_2\left[\left(u_1x_{2\xi_2} - u_2x_{1\xi_2}\right)_e\phi_e - \left(u_1x_{2\xi_2} - u_2x_{1\xi_2}\right)_w\phi_w\right] \\ \Delta\xi_1\left[\left(u_2x_{1\xi_1} - u_1x_{2\xi_1}\right)_e\phi_n - \left(u_2x_{1\xi_1} - u_1x_{2\xi_1}\right)_w\phi_s\right] \\ -\frac{\Delta\xi_2}{J}\left[\left(\Gamma_\phi\phi_{\xi_1}x_{2\xi_2}^2 + \Gamma_\phi\phi_{\xi_1}x_{1\xi_2}^2\right)_e - \left(\Gamma_\phi\phi_{\xi_1}x_{2\xi_2}^2 + \Gamma_\phi\phi_{\xi_1}x_{1\xi_2}^2\right)_w\right] \\ -\frac{\Delta\xi_1}{J}\left[\left(\Gamma_\phi\phi_{\xi_2}x_{2\xi_1}^2 + \Gamma_\phi\phi_{\xi_2}x_{1\xi_1}^2\right)_n - \left(\Gamma_\phi\phi_{\xi_2}x_{2\xi_1}^2 + \Gamma_\phi\phi_{\xi_2}x_{1\xi_1}^2\right)_s\right] = \\ -\frac{\Delta\xi_2}{J}\left[\left(\Gamma_\phi\phi_{\xi_2}x_{2\xi_1}x_{2\xi_2} + \Gamma_\phi\phi_{\xi_2}x_{1\xi_1}x_{1\xi_2}\right)_e - \left(\Gamma_\phi\phi_{\xi_2}x_{2\xi_1}x_{2\xi_2} + \Gamma_\phi\phi_{\xi_2}x_{1\xi_1}x_{1\xi_2}\right)_w\right] \\ -\frac{\Delta\xi_1}{J}\left[\left(\Gamma_\phi\phi_{\xi_1}x_{2\xi_2}x_{2\xi_1} + \Gamma_\phi\phi_{\xi_1}x_{1\xi_2}x_{1\xi_1}\right)_n - \left(\Gamma_\phi\phi_{\xi_1}x_{2\xi_2}x_{2\xi_1} + \Gamma_\phi\phi_{\xi_1}x_{1\xi_2}x_{1\xi_1}\right)_s\right] \\ +\Delta\xi_1\Delta\xi_2JS_\phi \end{gathered} \tag{2.44}$$

Due to the arrangement of the ξ_i grid, both $\Delta\xi_1$ and $\Delta\xi_2$ are unity. The face values in the convective terms (terms 2 and 3 of Equation 2.44) are approximated using the Quadratic Upstream Interpolation for Convective Kinematics (QUICK) scheme [21]. The QUICK convection scheme dictates that face values are calculated using an upwind quadratic interpolation; for the east face this becomes

$$\phi_e = \begin{cases} \frac{3}{8}\phi_E + \frac{3}{4}\phi_P - \frac{1}{8}\phi_W & \text{if } q_e > 0 \\ \frac{3}{8}\phi_P + \frac{3}{4}\phi_E - \frac{1}{8}\phi_{EE} & \text{if } q_e < 0 \end{cases} \tag{2.45}$$

where the volume flux across the east face, q_e is found by

$$q_e \approx \left(u_1x_{2\xi_2}\right)_e - \left(u_2x_{1\xi_2}\right)_e \tag{2.46}$$

for future reference,

$$q_n \approx \left(-u_1x_{2\xi_1}\right)_n + \left(u_2x_{1\xi_1}\right)_n \tag{2.47}$$

Applying Equation 2.46 at the west face and Equation 2.47 at the south face provides the values of q_w and q_s respectively. The advecting velocities u_1 and u_2 in Equations 2.46 and 2.47 are calculated at the cell faces using the Pressure Weighted Interpolation Method (PWIM) proposed by Rhie and Chow [32]. For example,

$$u_{1e} = \frac{1}{2}\left[\widehat{u_1}_P + \widehat{u_1}_E\right] + \frac{\Delta t}{J}\left(-\left[P_E^{n+1} - P_P^{n+1}\right]\left(x_{2\xi_2}\right)_e + \left[P_{ne}^{n+1} - P_{se}^{n+1}\right]\left(x_{2\xi_1}\right)_e\right) \tag{2.48}$$

Noting that the first part of each of the convection terms is the face volume flux, this term can be used to convert the convection terms into a more convenient

form. For example, for the east face

$$\begin{gathered}\left(u_1 x_{2\xi_2} - u_2 x_{1\xi_2}\right)_e \phi_e \approx q_e \phi_e \approx \\ \max\left[q_e, 0\right]\left(\tfrac{3}{8}\phi_E + \tfrac{3}{4}\phi_P - \tfrac{1}{8}\phi_W\right) - \max\left[-q_e, 0\right]\left(\tfrac{3}{8}\phi_P + \tfrac{3}{4}\phi_E - \tfrac{1}{8}\phi_{EE}\right)\end{gathered} \tag{2.49}$$

The value of ϕ for the diffusive flux terms (terms 4 and 5 in Equation 2.44) are determined using the second order central difference scheme, for example,

$$\left(\Gamma_\phi \phi_{\xi_1} x_{2\xi_2}^2 + \Gamma_\phi \phi_{\xi_1} x_{1\xi_2}^2\right)_e = \left[\Gamma_\phi x_{2\xi_2}^2 + \Gamma_\phi x_{1\xi_2}^2\right]\left(\phi_E - \phi_P\right) \tag{2.50}$$

The cross diffusive terms (terms 6 and 7 in Equation 2.44) are approximated in a similar way, for example,

$$\left(\Gamma_\phi \phi_{\xi_2} x_{2\xi_1} x_{2\xi_2} + \Gamma_\phi \phi_{\xi_2} x_{1\xi_1} x_{1\xi_2}\right)_e = \left[\Gamma_\phi x_{2\xi_1} x_{2\xi_2} + \Gamma_\phi x_{1\xi_1} x_{1\xi_2}\right]\left(\phi_{ne} - \phi_{se}\right) \tag{2.51}$$

where

$$\phi_{ne} = \frac{\phi_N + \phi_P + \phi_P + \phi_{NE}}{4} \tag{2.52}$$

and similar approximations are made at the other corners.

Making the substitutions described by Equations 2.49, 2.50 and 2.51 into Equation 2.44 provides

$$\begin{gathered}J\phi_{P_t} \\ + \left(\max\left[q_e, 0\right]\left(\tfrac{3}{8}\phi_E + \tfrac{3}{4}\phi_P - \tfrac{1}{8}\phi_W\right) - \max\left[-q_e, 0\right]\left(\tfrac{3}{8}\phi_P + \tfrac{3}{4}\phi_E - \tfrac{1}{8}\phi_{EE}\right)\right) \\ - \left(\max\left[q_w, 0\right]\left(\tfrac{3}{8}\phi_P + \tfrac{3}{4}\phi_W - \tfrac{1}{8}\phi_{WW}\right) - \max\left[-q_w, 0\right]\left(\tfrac{3}{8}\phi_W + \tfrac{3}{4}\phi_P - \tfrac{1}{8}\phi_E\right)\right) \\ + \left(\max\left[q_n, 0\right]\left(\tfrac{3}{8}\phi_N + \tfrac{3}{4}\phi_P - \tfrac{1}{8}\phi_S\right) - \max\left[-q_n, 0\right]\left(\tfrac{3}{8}\phi_P + \tfrac{3}{4}\phi_N - \tfrac{1}{8}\phi_{NN}\right)\right) \\ - \left(\max\left[q_s, 0\right]\left(\tfrac{3}{8}\phi_P + \tfrac{3}{4}\phi_S - \tfrac{1}{8}\phi_{SS}\right) - \max\left[-q_s, 0\right]\left(\tfrac{3}{8}\phi_S + \tfrac{3}{4}\phi_P - \tfrac{1}{8}\phi_N\right)\right) \\ - \left(\left[\tfrac{\Gamma_\phi}{J} x_{2\xi_2}^2 + \tfrac{\Gamma_\phi}{J} x_{1\xi_2}^2\right]_e \left(\phi_E - \phi_P\right) - \left[\tfrac{\Gamma_\phi}{J} x_{2\xi_2}^2 + \tfrac{\Gamma_\phi}{J} x_{1\xi_2}^2\right]_w \left(\phi_P - \phi_W\right)\right) \\ - \left(\left[\tfrac{\Gamma_\phi}{J} x_{2\xi_1}^2 + \tfrac{\Gamma_\phi}{J} x_{1\xi_1}^2\right]_n \left(\phi_N - \phi_P\right) - \left[\tfrac{\Gamma_\phi}{J} x_{2\xi_1}^2 + \tfrac{\Gamma_\phi}{J} x_{1\xi_1}^2\right]_s \left(\phi_P - \phi_S\right)\right) = \\ - \left(\left[\tfrac{\Gamma_\phi}{J} x_{2\xi_1} x_{2\xi_2} + \tfrac{\Gamma_\phi}{J} x_{1\xi_1} x_{1\xi_2}\right]_e \left(\phi_{ne} - \phi_{se}\right) - \left[\tfrac{\Gamma_\phi}{J} x_{2\xi_1} x_{2\xi_2} + \tfrac{\Gamma_\phi}{J} x_{1\xi_1} x_{1\xi_2}\right]_w \left(\phi_{nw} - \phi_{sw}\right)\right) \\ - \left(\left[\tfrac{\Gamma_\phi}{J} x_{2\xi_2} x_{2\xi_1} + \tfrac{\Gamma_\phi}{J} x_{1\xi_2} x_{1\xi_1}\right]_n \left(\phi_{ne} - \phi_{nw}\right) - \left[\tfrac{\Gamma_\phi}{J} x_{2\xi_2} x_{2\xi_1} + \tfrac{\Gamma_\phi}{J} x_{1\xi_2} x_{1\xi_1}\right]_s \left(\phi_{se} - \phi_{sw}\right)\right) \\ + JS_\phi\end{gathered} \tag{2.53}$$

Collecting like terms allows the equation to be written as

$$J\phi_{P_t} + A_P\phi_P - A_E\phi_E - A_W\phi_W - A_N\phi_N - A_S\phi_S = JS_\phi^{DC} + JS_\phi^{CD} + JS_\phi \tag{2.54}$$

where

$$A_E = \max\left[-q_e, 0\right] + \left[\frac{\Gamma_\phi}{J}\left(x_{2\xi_2}^2 + x_{1\xi_2}^2\right)\right]_e \tag{2.55}$$

$$A_W = \max\left[q_w, 0\right] + \left[\frac{\Gamma_\phi}{J}\left(x_{2\xi_2}^2 + x_{1\xi_2}^2\right)\right]_w \tag{2.56}$$

$$A_N = \max\left[-q_n, 0\right] + \left[\frac{\Gamma_\phi}{J}\left(x_{2\xi_1}^2 + x_{1\xi_1}^2\right)\right]_n \tag{2.57}$$

$$A_S = \max\left[q_s, 0\right] + \left[\frac{\Gamma_\phi}{J}\left(x_{2\xi_1}^2 + x_{1\xi_1}^2\right)\right]_s \tag{2.58}$$

$$A_P = A_E + A_W + A_S + A_N + q_e - q_w + q_n - q_s \tag{2.59}$$

The values of the convection terms in Equations 2.55 through 2.58 correspond to a first order upwind convection scheme. This is returned to the QUICK scheme through the deferred correction source term, JS_ϕ^{DC}, which takes the form

$$\begin{aligned} JS_\phi^{DC} = & \\ -\left(\max\left[q_e, 0\right]\left(\tfrac{3}{8}\phi_E - \tfrac{1}{4}\phi_P - \tfrac{1}{8}\phi_W\right) - \max\left[-q_e, 0\right]\left(\tfrac{3}{8}\phi_P - \tfrac{1}{4}\phi_E - \tfrac{1}{8}\phi_{EE}\right)\right) & \\ +\left(\max\left[q_w, 0\right]\left(\tfrac{3}{8}\phi_P - \tfrac{1}{4}\phi_W - \tfrac{1}{8}\phi_{WW}\right) - \max\left[-q_w, 0\right]\left(\tfrac{3}{8}\phi_W - \tfrac{1}{4}\phi_P - \tfrac{1}{8}\phi_E\right)\right) & \\ \left(-\max\left[q_n, 0\right]\left(\tfrac{3}{8}\phi_N - \tfrac{1}{4}\phi_P - \tfrac{1}{8}\phi_S\right) - \max\left[-q_n, 0\right]\left(\tfrac{3}{8}\phi_P - \tfrac{1}{4}\phi_N - \tfrac{1}{8}\phi_{NN}\right)\right) & \\ +\left(\max\left[q_s, 0\right]\left(\tfrac{3}{8}\phi_P - \tfrac{1}{4}\phi_S - \tfrac{1}{8}\phi_{SS}\right) - \max\left[-q_s, 0\right]\left(\tfrac{3}{8}\phi_S - \tfrac{1}{4}\phi_P - \tfrac{1}{8}\phi_N\right)\right) & \end{aligned} \tag{2.60}$$

Finally, the cross diffusion source term is written as

$$\begin{aligned} JS_\phi^{CD} = & \\ -\left(\left[\tfrac{\Gamma_\phi}{J}\left(x_{2\xi_1}x_{2\xi_2} + x_{1\xi_1}x_{1\xi_2}\right)\right]_e + \left[\tfrac{\Gamma_\phi}{J}\left(x_{2\xi_2}x_{2\xi_1} + x_{1\xi_2}x_{1\xi_1}\right)\right]_n\right)\phi_{ne} & \\ +\left(\left[\tfrac{\Gamma_\phi}{J}\left(x_{2\xi_1}x_{2\xi_2} + x_{1\xi_1}x_{1\xi_2}\right)\right]_e + \left[\tfrac{\Gamma_\phi}{J}\left(x_{2\xi_2}x_{2\xi_1} + x_{1\xi_2}x_{1\xi_1}\right)\right]_s\right)\phi_{se} & \\ +\left(\left[\tfrac{\Gamma_\phi}{J}\left(x_{2\xi_1}x_{2\xi_2} + x_{1\xi_1}x_{1\xi_2}\right)\right]_w + \left[\tfrac{\Gamma_\phi}{J}\left(x_{2\xi_2}x_{2\xi_1} + x_{1\xi_2}x_{1\xi_1}\right)\right]_n\right)\phi_{nw} & \\ -\left(\left[\tfrac{\Gamma_\phi}{J}\left(x_{2\xi_1}x_{2\xi_2} + x_{1\xi_1}x_{1\xi_2}\right)\right]_w + \left[\tfrac{\Gamma_\phi}{J}\left(x_{2\xi_2}x_{2\xi_1} + x_{1\xi_2}x_{1\xi_1}\right)\right]_s\right)\phi_{sw} & \end{aligned} \tag{2.61}$$

The continuity equation, Equation 2.5, can also be integrated over the standard control volume and written in curvilinear terms. It becomes,

$$q_e - q_w + q_n - q_s = 0 \tag{2.62}$$

which allows Equation 2.59 to be rewritten as

$$A_P = A_E + A_W + A_S + A_N \ . \tag{2.63}$$

2.5 Time-Stepping Schemes

The time derivative (transient) term of the transport equations (term 1 in Equation 2.54) is often set to zero, which causes a steady state solution to be generated. In cases that it is included in the transport equations it is typically included for one of two reasons. The first is to observe some transient flow phenomena such as acoustic waves or vortex shedding. The second is included to improve convergence of difficult steady state problems by allowing the solution to reach steady state in a manner which mimics natural flow physics. In the first case, temporal accuracy is important while in the second case, commonly referred to as "pseudo time-stepping," temporal accuracy is not important as only the final solution is of interest.

A variety of methods can be used to deal with the transient term in the generic transport equation. These methods range from fully explicit schemes to fully implicit time-stepping schemes. Fully explicit schemes, such as the forward Euler or Runge-Kutta methods, depend only on information from the previous time step. As a result, explicit schemes tend to march through time steps with a relatively low computational cost as the equations governing flow through each control volume can be solved individually. The primary disadvantage of explicit schemes is that they tend to become unstable when time step sizes get too large. This restriction on step size can be prohibitively limiting especially in cases where only the final steady state solution is of interest.

In contrast to explicit time-stepping schemes, implicit schemes such as Backwards Euler or Crank-Nicolson use both information from the previous time step as well as information from the current time step. These schemes allow the use of larger maximum time step sizes than explicit schemes which can make them more efficient for reaching a steady state solution. The need for information at the current time step, however, requires that the flow values in all control volumes in the domain be solved simultaneously which corresponds to a large computational cost at each time step.

Combinations of implicit and explicit time-stepping schemes are typically called semi-implicit or implicit explicit (IMEX) methods. These time-stepping schemes take flow values from the previous time step for some terms and the current time step for the rest of the terms.

In addition to IMEX methods for solving individual transport equations, semi-implicit time-stepping schemes can also be employed in which some of the flow equations are solved implicitly and some are solved explicitly. Two different schemes

will be evaluated in this work; one is the SIMPLE algorithm which involves solving for pressure correction and velocity implicitly. The second time-stepping scheme evaluated here is the Fractional Step method which solves for pressure implicitly but solves for the velocity components explicitly. In both cases turbulence quantities are solved implicitly to ensure stability.

The implementations of these time-stepping schemes in the STREAM code approximate the time derivative as

$$\phi_{Pt} \simeq \frac{\phi_P^{n+1} - \phi_P^n}{\Delta t} \tag{2.64}$$

where a superscript containing the index n indicates a value taken at the n^{th}, or previous, time step.

The velocity transport equations (ie $\phi = u_1$ and $\phi = u_2$) are treated differently from the other transport equations as they contain pressure as well as velocity. These equations are coupled with the continuity equation, Equation 2.43, to determine the velocity and pressure fields. The momentum equation for u_1 is rewritten as

$$\begin{aligned} \tfrac{J}{\Delta t} {u_1}_P^{n+1} + A_P u_{1P} - A_E u_{1E} - A_W u_{1W} - A_N u_{1N} - A_S u_{1S} = \\ \tfrac{J}{\Delta t} {u_1}_P^{n} + Q_{u_1} + \left(-P_{\xi_1} x_{2\xi_2} + P_{\xi_2} x_{2\xi_1}\right)_P \end{aligned} \tag{2.65}$$

where

$$Q_{u_1} = JS_{u_1}^{QUICK} + JS_{u_1}^{CD} + JS_{u_1} - \left(-P_{\xi_1} x_{2\xi_2} + P_{\xi_2} x_{2\xi_1}\right)_P \tag{2.66}$$

It should be noted that the time coordinate has not been set for the majority of the terms in Equation 2.65. A decision must be made for where in time, relative to the current step $(n+1)$, each of the flow variables will sampled at in order to solve the equation. The choice of position in time for each of the terms depends on the time-stepping scheme. In general, a fully explicit scheme would have all remaining terms sampled at the n^{th} time step while an implicit scheme will take them all at the $(n+1)^{\text{th}}$ step.

2.5.1 SIMPLE

To implement an implicit time-stepping scheme, all terms with unassigned time steps are assigned to be sampled at the $(n+1)^{\text{th}}$ time step. Equation 2.65 is solved for the current cell velocity providing

$$u_{1P}^{n+1} = \left[\frac{1}{\widehat{A_P}} \left(\sum_{nb=E,W,N,S} A_{nb} u_{1nb} + Q_{u_1} + \frac{J}{\Delta t} u_{1P}^{n} + \left(-P_{\xi_1} x_{2\xi_2} + P_{\xi_2} x_{2\xi_1} \right)_P \right) \right]^{n+1} \tag{2.67}$$

where

$$\widehat{A_P} = \left(\frac{J}{\Delta t} + A_P \right)$$

In this section all flow values will be assumed to be at the $(n+1)^{\text{th}}$ time step unless otherwise stated.

In order to solve this equation, velocity and pressure are cast as an unconverged or guessed solutions and corrections (denoted by a superscript * and ' respectively.) In other words,

$$\begin{aligned} u_1 &= u_1^* + u_1' \\ u_2 &= u_2^* + u_2' \\ P &= P^* + P' \end{aligned} \tag{2.68}$$

Substituting this into Equation 2.67 provides

$$\begin{aligned} (u_1^* + u_1')_P = \tfrac{1}{\widehat{A_P}} \left(\textstyle\sum_{nb=E,W,N,S} A_{nb} \left(u_1^* + u_1' \right)_{nb} + Q_{u_1} + \tfrac{J}{\Delta t} u_{1P}^{n} \right) \\ + \tfrac{1}{\widehat{A_P}} \left(- \left(P^* + P' \right)_{\xi_1} x_{2\xi_2} + \left(P^* + P' \right)_{\xi_2} x_{2\xi_1} \right)_P \end{aligned} \tag{2.69}$$

For a given guessed pressure field, a corresponding unconverged velocity field can be calculated, which satisfies the momentum equation, or

$$(u_1^*)_P = \frac{1}{\widehat{A_P}} \left(\sum_{nb=E,W,N,S} A_{nb} u_{1nb}^* + Q_{u_1} + \frac{J}{\Delta t} u_{1P}^{n} + \left(-P_{\xi_1}^* x_{2\xi_2} + P_{\xi_2}^* x_{2\xi_1} \right)_P \right) \tag{2.70}$$

Subtracting Equation 2.70 from Equation 2.69 provides

$$u_{1P}' = \frac{1}{\widehat{A_P}} \sum_{nb=E,W,N,S} A_{nb} u_{1nb}' + \frac{1}{\widehat{A_P}} \left(-P_{\xi_1}' x_{2\xi_2} + P_{\xi_2}' x_{2\xi_1} \right)_P \tag{2.71}$$

When the solution has converged all of the correction terms will become zero, thus the SIMPLE algorithm assumes that the u_{1nb}' terms in Equation 2.71 can all be neglected, which provides the velocity correction equation,

$$u'_{1P} = \frac{1}{\widehat{A_P}} \left(-P'_{\xi_1} x_{2\xi_2} + P'_{\xi_2} x_{2\xi_1} \right)_P \tag{2.72}$$

Similarly, u'_2 can be found to be

$$u'_{2P} = \frac{1}{\widehat{A_P}} \left(P'_{\xi_1} x_{1\xi_2} - P'_{\xi_2} x_{1\xi_1} \right)_P \tag{2.73}$$

The continuity equation, Equation 2.62, can be written in the unconverged-corrected form

$$q'_e - q'_w + q'_n - q'_s + q^*_e - q^*_w + q^*_n - q^*_s = 0 \tag{2.74}$$

or

$$q'_e - q'_w + q'_n - q'_s = -\dot{m}^* \tag{2.75}$$

where the mass imbalance is defined as

$$\dot{m}^* = q^*_e - q^*_w + q^*_n - q^*_s \tag{2.76}$$

To find the volume flux corrections, Equations 2.72 and 2.73 are substituted into Equations 2.46 and 2.47. This provides

$$q'_e = \left(\frac{1}{\widehat{A_P}} \left(-P'_{\xi_1} x_{2\xi_2} + P'_{\xi_2} x_{2\xi_1} \right) x_{2\xi_2} \right)_e - \left(\frac{1}{\widehat{A_P}} \left(P'_{\xi_1} x_{1\xi_2} - P'_{\xi_2} x_{1\xi_1} \right) x_{1\xi_2} \right)_e \tag{2.77}$$

Equation 2.77 can be simplified for the sake of the matrix solver used in the present study by neglecting $P'_{\xi_2} x_{2\xi_1}$ from the first group of terms and $P'_{\xi_1} x_{1\xi_2}$ from the second set of terms. This provides the result,

$$q'_e = -\left(\frac{1}{\widehat{A_P}} \left[P'_{\xi_1} \left(x^2_{2\xi_2} + x^2_{1\xi_2} \right) \right] \right)_e \tag{2.78}$$

Similarly,

$$q'_n = -\left(\frac{1}{\widehat{A_P}} \left[P'_{\xi_2} \left(x^2_{1\xi_1} + x^2_{2\xi_1} \right) \right] \right)_n \tag{2.79}$$

Substituting this back into Equation 2.75 provides

$$\begin{aligned} &-\left(\frac{1}{\widehat{A_P}} \left[P'_{\xi_1} \left(x^2_{2\xi_2} + x^2_{1\xi_2} \right) \right] \right)_e \\ &+\left(\frac{1}{\widehat{A_P}} \left[P'_{\xi_1} \left(x^2_{2\xi_2} + x^2_{1\xi_2} \right) \right] \right)_w \\ &-\left(\frac{1}{\widehat{A_P}} \left[P'_{\xi_2} \left(x^2_{1\xi_1} + x^2_{2\xi_1} \right) \right] \right)_n \\ &+\left(\frac{1}{\widehat{A_P}} \left[P'_{\xi_2} \left(x^2_{1\xi_1} + x^2_{2\xi_1} \right) \right] \right)_s \\ &= -\dot{m}^* \end{aligned} \tag{2.80}$$

The pressure derivatives at the faces can be approximated as, for example,

$$\left(P'_{\xi_1}\right)_e = P'_E - P'_P \tag{2.81}$$

Equation 2.80 then becomes

$$\begin{aligned} &- [P'_E - P'_P] \left(\frac{x_{2\xi_2}^2 + x_{1\xi_2}^2}{\widehat{A_P}} \right)_e + [P'_P - P'_W] \left(\frac{x_{2\xi_2}^2 + x_{1\xi_2}^2}{\widehat{A_P}} \right)_w \\ &- [P'_N - P'_P] \left(\frac{x_{1\xi_1}^2 + x_{2\xi_1}^2}{\widehat{A_P}} \right)_n + [P'_P - P'_S] \left(\frac{x_{1\xi_1}^2 + x_{2\xi_1}^2}{\widehat{A_P}} \right)_s \\ &= -\dot{m}^* \end{aligned} \tag{2.82}$$

This equation can be rewritten in a form which is convenient to solve using an iterative matrix solver, that is

$$A_P^{P'} P'_P = \sum_{nb=E,W,N,S} A_{nb}^{P'} P'_{nb} - \dot{m}^* \tag{2.83}$$

where

$$\begin{aligned} A_E^{P'} &= \left(\frac{x_{2\xi_2}^2 + x_{1\xi_2}^2}{\widehat{A_P}} \right)_e \quad A_W^{P'} = \left(\frac{x_{2\xi_2}^2 + x_{1\xi_2}^2}{\widehat{A_P}} \right)_w \\ A_N^{P'} &= \left(\frac{x_{1\xi_1}^2 + x_{2\xi_1}^2}{\widehat{A_P}} \right)_n \quad A_S^{P'} = \left(\frac{x_{1\xi_1}^2 + x_{2\xi_1}^2}{\widehat{A_P}} \right)_s \\ A_P^{P'} &= \sum_{nb=E,W,N,S} A_{nb}^{P'} \end{aligned}$$

Solution of this equation provides the pressure corrections for the current time step. Given the pressure corrections, Equations 2.72 and 2.73 can be used to find velocity corrections to the velocity field produced by solving Equation 2.70 and its equivalent for the u_2^* field. The final step is to calculate the corrected pressure field and velocity fields. To provide stable convergence, under relaxation factors are used to scale the correction factors so that

$$\begin{aligned} P_P &= P_P^* + \alpha_P P'_P \\ u_{1P} &= u_{1P}^* + \alpha_{u_1} u_{1P}' \\ u_{2P} &= u_{2P}^* + \alpha_{u_2} u_{2P}' \end{aligned} \tag{2.84}$$

For this study, constant values of $\alpha_P = 0.4$ and $\alpha_{u_1} = \alpha_{u_2} = 0.6$ were used.

2.5.2 Fractional Step

The fractional step method uses a more explicit approach than SIMPLE. Equation 2.65 is solved for ${u_1}_P^{n+1}$ while all the right hand terms are evaluated at the n^{th} time step with the exception of the pressure terms which are evaluated at the $(n+1)^{\text{th}}$.

$${u_1}_P^{n+1} = {u_1}_P^{n} + \frac{\Delta t}{J}\left[-A_P u_{1P} + \sum_{nb=E,W,N,S} A_{nb} u_{1nb} + Q_{u_1}\right]^n + \left(-P_{\xi_1} x_{2\xi_2} + P_{\xi_2} x_{2\xi_1}\right)_P^{n+1} \tag{2.85}$$

or

$${u_1}_P^{n+1} = \widehat{u_1}_P + \frac{\Delta t}{J}\left(-P_{\xi_1} x_{2\xi_2} + P_{\xi_2} x_{2\xi_1}\right)_P^{n+1} \tag{2.86}$$

where

$$\widehat{u_1}_P = {u_1}_P^{n} + \frac{\Delta t}{J}\left[-A_P u_{1P} + \sum_{nb=E,W,N,S} A_{nb} u_{1nb} + Q_{u_1}\right]^n \tag{2.87}$$

A pressure equation is derived from the continuity equation, Equation 2.62. The face volume flows can be found using PWIM [32] to find the face velocities as described earlier, i.e. Equation 2.48. The face velocities are then substituted into Equations 2.46 and 2.47 to provide the face volume flows. Substituting the volume

flows into Equation 2.62 provides

$$\begin{aligned}
&\left[\tfrac{1}{2}\left[\widehat{u}_{1P}+\widehat{u}_{1E}\right]+\left(\tfrac{\Delta t}{J_e}\left[P_{ne}^{n+1}-P_{se}^{n+1}\right]\left(x_{2\xi_1}\right)_e\right)\right]\left[x_{2\xi_2}\right]_e \\
&+\left[-\tfrac{1}{2}\left[\widehat{u}_{2P}+\widehat{u}_{2E}\right]+\left(\tfrac{\Delta t}{J_e}\left[P_{ne}^{n+1}-P_{se}^{n+1}\right]\left(x_{1\xi_1}\right)_e\right)\right]\left[x_{1\xi_2}\right]_e \\
&+\left[P_P^{n+1}-P_E^{n+1}\right]\tfrac{\Delta t}{J_e}\left(\left[x_{2\xi_2}\right]_e^2+\left[x_{1\xi_2}\right]_e^2\right) \\
&+\left[-\tfrac{1}{2}\left[\widehat{u}_{1P}+\widehat{u}_{1W}\right]+\left(-\tfrac{\Delta t}{J_w}\left[P_{nw}^{n+1}-P_{sw}^{n+1}\right]\left(x_{2\xi_1}\right)_w\right)\right]\left[x_{2\xi_2}\right]_w \\
&+\left[\tfrac{1}{2}\left[\widehat{u}_{2W}+\widehat{u}_{2P}\right]+\left(-\tfrac{\Delta t}{J_w}\left[P_{nw}^{n+1}-P_{sw}^{n+1}\right]\left(x_{1\xi_1}\right)_w\right)\right]\left[x_{1\xi_2}\right]_w \\
&+\left[P_P^{n+1}-P_W^{n+1}\right]\tfrac{\Delta t}{J_w}\left(\left[x_{2\xi_2}\right]_w^2+\left[x_{1\xi_2}\right]_w^2\right) \\
&+\left[-\tfrac{1}{2}\left[\widehat{u}_{1P}+\widehat{u}_{1N}\right]+\left(\tfrac{\Delta t}{J_n}\left[P_{ne}^{n+1}-P_{nw}^{n+1}\right]\left(x_{1\xi_1}\right)_n\right)\right]\left[x_{2\xi_1}\right]_n \\
&+\left[\tfrac{1}{2}\left[\widehat{u}_{2P}+\widehat{u}_{2N}\right]+\left(\tfrac{\Delta t}{J_n}\left[P_{ne}^{n+1}-P_{nw}^{n+1}\right]\left(x_{1\xi_2}\right)_n\right)\right]\left[x_{1\xi_1}\right]_n \\
&+\left[P_P^{n+1}-P_N^{n+1}\right]\tfrac{\Delta t}{J_n}\left(\left[x_{2\xi_1}\right]_n^2+\left[x_{1\xi_1}\right]_n^2\right) \\
&+\left[\tfrac{1}{2}\left[\widehat{u}_{1S}+\widehat{u}_{1P}\right]+\left(\tfrac{\Delta t}{J_s}-\left[P_{se}^{n+1}-P_{sw}^{n+1}\right]\left(x_{1\xi_1}\right)_s\right)\right]\left[x_{2\xi_1}\right]_s \\
&+\left[-\tfrac{1}{2}\left[\widehat{u}_{2S}+\widehat{u}_{2P}\right]+\left(-\tfrac{\Delta t}{J_s}\left[P_{se}^{n+1}-P_{sw}^{n+1}\right]\left(x_{1\xi_2}\right)_s\right)\right]\left[x_{1\xi_1}\right]_s \\
&+\left[P_P^{n+1}-P_S^{n+1}\right]\tfrac{\Delta t}{J_s}\left(\left[x_{2\xi_1}\right]_s^2+\left[x_{1\xi_1}\right]_s^2\right) \\
&=0
\end{aligned} \tag{2.88}$$

which can be written in a form which allows a pressure field to be calculated. The final form of the pressure equation is

$$A_P^P P_P^{n+1}=\sum_{nb=E,W,N,S} A_{nb}^P P_{nb}^{n+1}+S_u^P \tag{2.89}$$

where

$$
\begin{aligned}
A_E^P &= \frac{\Delta t}{J_e}\left(\left[x_{2\xi_2}\right]_e^2 + \left[x_{1\xi_2}\right]_e^2\right) \\
A_N^P &= \frac{\Delta t}{J_n}\left(\left[x_{2\xi_1}\right]_n^2 + \left[x_{1\xi_1}\right]_n^2\right) \\
A_P^P &= \sum_{nb=E,W,N,S} A_{nb}^P \\
S_u^P &= -\left[\tilde{q}_e - \tilde{q}_w + \tilde{q}_n - \tilde{q}_s\right] \\
\tilde{q}_e &= \left(\tilde{u}_1 x_{2\xi_2}\right)_e - \left(\tilde{u}_2 x_{1\xi_2}\right)_e \\
\tilde{q}_n &= \left(-\tilde{u}_1 x_{2\xi_1}\right)_n + \left(\tilde{u}_2 x_{1\xi_1}\right)_n \\
\tilde{u_{1e}} &= \left[\frac{1}{2}\left[\widehat{u_{1P}} + \widehat{u_{1E}}\right] - \left(\frac{\Delta t}{J_e}\left[P_{ne}^{n+1} - P_{se}^{n+1}\right]\left(-x_{2\xi_1}\right)_e\right)\right] \\
\tilde{u_{2e}} &= \left[\frac{1}{2}\left[\widehat{u_{2P}} + \widehat{u_{2E}}\right] - \left(\frac{\Delta t}{J_e}\left[P_{ne}^{n+1} - P_{se}^{n+1}\right]\left(x_{1\xi_1}\right)_e\right)\right] \\
\tilde{u_{1n}} &= \left[\frac{1}{2}\left[\widehat{u_{1P}} + \widehat{u_{1N}}\right] - \left(\frac{\Delta t}{J_n}\left[P_{ne}^{n+1} - P_{nw}^{n+1}\right]\left(x_{1\xi_1}\right)_n\right)\right] \\
\tilde{u_{2n}} &= \left[\frac{1}{2}\left[\widehat{u_{2P}} + \widehat{u_{2N}}\right] - \left(\frac{\Delta t}{J_n}\left[P_{ne}^{n+1} - P_{nw}^{n+1}\right]\left(-x_{1\xi_2}\right)_n\right)\right]
\end{aligned}
$$

The corner pressures (e.g. P_{ne}) are found using the same bilinear interpolation described by Equation 2.52. The pressure equation can now be solved using a variety of matrix equation solvers. For this study, the standard Tri-Diagonal Matrix Algorithm (TDMA), a special case of Gaussian elimination, is used to solve the required matrix equations.

2.6 Linearization of Source Terms

The k and ε equations arising from the modeling of turbulence are more difficult to solve as the values of both k and ε must remain positive to be physically relevant. In addition, these equations are highly non-linear and tend to cause instability issues when solving them with a linear solver. The stability of the solution can be increased by making the diagonal term in the coefficient matrix larger via source term linearization. That is

$$S_\phi = S_u + S_P\phi \text{ where } S_P < 0 \tag{2.90}$$

The source term for the k equation, Equation 2.13, is linearized as

$$S_u^k = P_k \tag{2.91}$$

$$S_P^k = -\frac{\varepsilon}{k} = -\frac{\varepsilon}{k} C_\mu \frac{k^2}{\varepsilon \nu_t} = -\frac{C_\mu k}{\nu_t} \tag{2.92}$$

The source term for the ε equation, Equation 2.14, is linearized as

$$S_u^\varepsilon = C_{\varepsilon 1} P_k \frac{\varepsilon}{k} \tag{2.93}$$

$$S_P^\varepsilon = -C_{\varepsilon 2} \frac{\varepsilon}{k} = -C_{\varepsilon 2} \frac{\varepsilon}{k} C_\mu \frac{k^2}{\varepsilon \nu_t} = -\frac{C_\mu C_{\varepsilon 2} k}{\nu_t} \tag{2.94}$$

2.7 Solution Generation

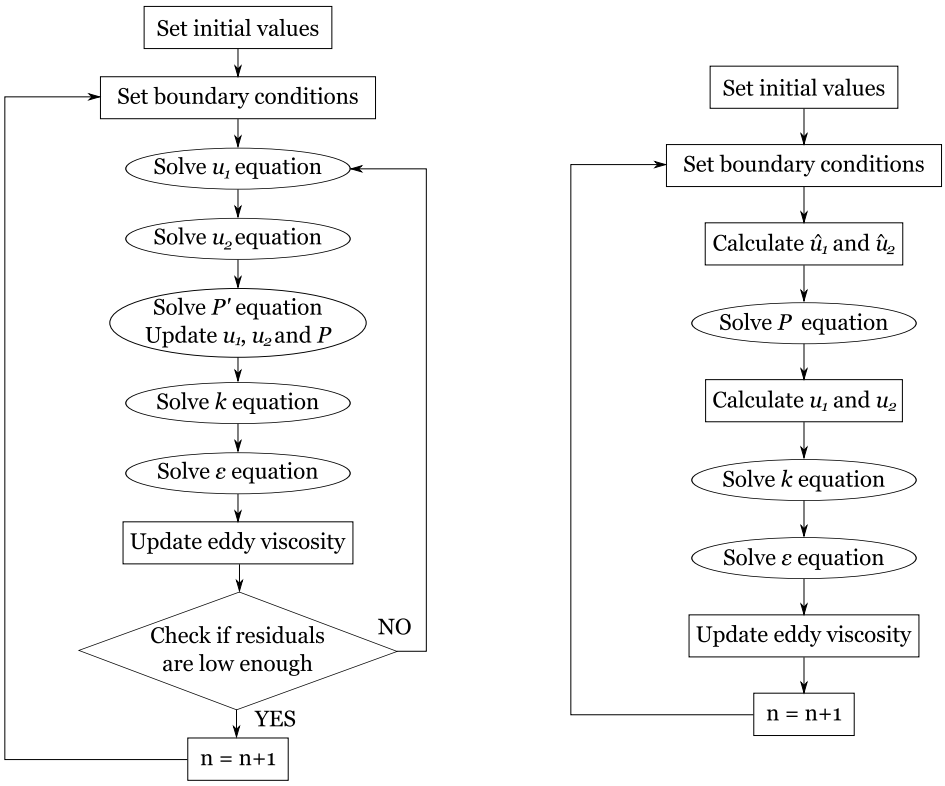
Once the time-stepping scheme has been selected, a computational domain and grid must be generated depending on the geometry and predicted flow physics of the problem. Problem dependent boundary and initial conditions are set to provide STREAM with values of the flow variables at $t = 0$ (or $n = 0$) and at the boundaries. Table 2.3 lists the initial conditions for cells not bordering a boundary for the cases studied here (turbulence quantities are only relevant to turbulent flows.)

u_1	u_2	P	k	ε
0	0	0	$0.005 u_\infty^2$	$\frac{C_\mu k^2}{20\nu}$

Table 2.3: Initial conditions of flow variables for cells not bordering a boundary.

Given initial conditions the code can then march through time in steps of Δt, increasing the value of n with each step. Flow charts are shown in Figure 2.2 for the SIMPLE and Fractional Step time-stepping schemes implemented in the STREAM code.

It should be noted that any block which calls for the solution of an equation requires an iterative solution of a matrix equation which tends to be more time consuming than a block which explicitly calculates the solution. Based on this, Figure 2.2 suggests that Fractional Step should solve each time step more quickly than SIMPLE as it only needs to iteratively solve for the pressure, k and ϵ fields, while the u_1 and u_2 fields are calculated based on previously determined values. In addition, the SIMPLE flow chart requires an internal loop to ensure that all of the flow variable fields are consistent with one another, adding additional calculation cost to the SIMPLE solution procedure.



(a) SIMPLE (b) Fractional Step

Figure 2.2: Flow charts for the (a) SIMPLE and (b) FS algorithms

2.8 Boundary Conditions

In order to close a solution domain, the conditions at the boundary must be specified. In the cases studied here four different types of boundary conditions were employed. In most cases these conditions are applied through the use of boundary cells which are grid of zero thickness cells that lie on the boundary of the solution domain which allow boundary conditions to be applied explicitly. Alternatively, in some cases, boundary conditions are applied through modification of the governing equation source terms.

2.8.1 Inlet

The inlet conditions are defined using boundary cells. The inlets have specified velocities, pressures, and turbulence quantities. Table 2.4 lists the typical inlet values for an inlet face normal to the x_1 direction on the west side of the domain.

u_{1W}	u_{2W}	P_W	k_W	ε_W
$U(x_2)$	0	P_P	$0.005u_\infty^2$	$\frac{C_\mu k^2}{20\nu}$

Table 2.4: Inlet boundary condition values

Typically the value of $U(x_2)$ is set to unity for all values of x_2, but occasionally it is necessary to define a non-symmetrical inlet flow to allow quasi-steady phenomena to develop as will be discussed later. The pressure inlet condition is equivalent to $\frac{\partial P}{\partial n} = 0$ at the inlet where n denotes the direction normal to the face (not to be confused with the n^{th} time step.)

2.8.2 Outlet

Outlet boundary conditions are the same as the inlet conditions for pressure. For velocity, the convective boundary condition followed by a global mass flow correction as described below is employed. The convective boundary condition for an east face outlet is

$$\frac{\partial u_1}{\partial t} + \frac{\partial u_1}{\partial x_1} = 0 \tag{2.95}$$

After this condition is applied, the velocity profiles at the inlet and outlet are used to determine the volume flows. The difference of the volume flows is then divided by

the exit area and added to the exit velocity profile to ensure net mass conservation in the system is always satisfied during each iteration. Boundary conditions for u_2, k and ε are taken as $\frac{\partial u_2}{\partial n} = 0$, $\frac{\partial k}{\partial n} = 0$ and $\frac{\partial \varepsilon}{\partial n} = 0$ respectively. These conditions are implemented through the use of boundary cells, with the boundary cell values for an east face outlet listed in Table 2.5.

u_{1E}	u_{2E}	P_E	k_E	ε_E
${u_1}_E^n - \Delta t \frac{{u_1}_E^n - {u_1}_P^n}{\Delta x_1}$	u_{2P}	P_P	k_P	ε_P

Table 2.5: Outlet boundary condition values

2.8.3 No-Slip Wall Boundaries

No-slip walls are applied through both boundary cells or through source term manipulation. Walls at the edge of the mesh use boundary cells to introduce boundary conditions while walls immersed in the mesh, like those seen in bluff body flow simulations, have their boundary conditions introduced through the use of source term manipulations applied to cells neighbouring the boundary. Table 2.6 lists the boundary cell values used for wall boundaries for a wall on the south side of a domain. Only laminar boundary values are reported here because the only test case in this study which has no-slip walls at the boundaries is laminar flow in a lid-driven skewed cavity.

u_{1S}	u_{2S}	P_S
0	0	P_P

Table 2.6: Wall boundary cell values for walls at the edge of the solution domain.

No-slip wall boundary conditions for turbulent flows are only applied to walls contained inside the flow domain in this study. These boundary conditions are applied through source term manipulation. In this case, the velocity and turbulence values close to the wall are determined using wall functions as described in Section 2.2.2. Once the values are known, the terms they occupy are moved into the source terms. For example, for a wall running along the south side of a control volume and the case where $\phi = u_1$, Equation 2.54 is modified by setting $A_S^{u_1} = 0$, and replacing the source term, S_{u_1} with a modified source term, $\widetilde{S_{u_1}}$ where

$$\widetilde{S_{u_1}} = S_{u_1} - \frac{k_P^{1/2} C_\mu^{1/4} \kappa}{\ln(Ey^+)} u_{1P}^{\tan} \sqrt{\left(x_{1_{\xi_1}}\right)_s^2 + \left(x_{2_{\xi_1}}\right)_s^2} \tag{2.96}$$

$$u_{1P}^{\tan} = (1 - n_1^2)u_{1P} - n_1 n_2 u_{2P} \tag{2.97}$$

and

$$[n_1, n_2] = \frac{1}{\sqrt{\left(x_{1_{\xi_1}}\right)_s^2 + \left(x_{2_{\xi_1}}\right)_s^2}} \left[\left(-x_{2_{\xi_1}}\right)_s, \left(x_{1_{\xi_1}}\right)_s\right] \tag{2.98}$$

In the k equation, Equation 2.13, the production term P_k is modified as follows in order to be consistent with the log law described in Section 2.2.2

$$P_k = \frac{k_P^{1/2} C_\mu^{1/4} \kappa}{y \left[\ln\left(Ey^+\right)\right]^2} \left(u_{1P} n_2 - u_{2P} n_1\right)^2 \tag{2.99}$$

and ε is modified in accordance with Equation 2.27.

2.8.4 Slip Wall Boundaries

Slip wall boundary conditions assume that $\frac{\partial \phi}{\partial n} = 0$ along the face. Boundary cell values for this boundary condition when applied along the north face of a domain are listed in Table 2.7.

u_{1N}	u_{2N}	P_N	k_N	ε_N
u_{1P}	0	P_P	k_P	ε_P

Table 2.7: Slip wall boundary condition values.

2.9 Summary

A numerical solution method for the transient, incompressible Navier-Stokes equations has been presented here. The largest turbulent eddies are to be resolved while the rest of the turbulent fluctuations are modeled through the use of $k - \varepsilon$ model with the modifications recommended by Kato and Launder [16] for bluff body flow. Time integration, or time stepping, is carried out using one of two approaches. The first approach, SIMPLE, is to solve two velocity equations, a pressure correction equation and two turbulence model equations (for turbulent flow cases only) implicitly for each time step. The second approach, FS, is to solve a pressure equation and two turbulence equations (in the turbulent flow cases) implicitly while treating the velocity terms explicitly.

It is expected that the selection of time stepping scheme will have a major impact on the efficiency of the code, however, this may be offset by numerical stability issues which are common in explicit time integration methods. This will be explored in the coming chapters.

Chapter 3

Lid-Driven Skewed Cavity

3.1 Problem Definition

The first and simplest test case selected to compare the SIMPLE and Fractional Step time-stepping schemes is the lid driven skewed cavity. The problem consists of a cavity with one unit long walls, top (lid) and floor. The sides of the cavity are skewed at 45°. The lid moves at a non-dimensional speed of 1 in the left to right (positive) direction. A schematic of the problem geometry is shown in Figure 3.1.

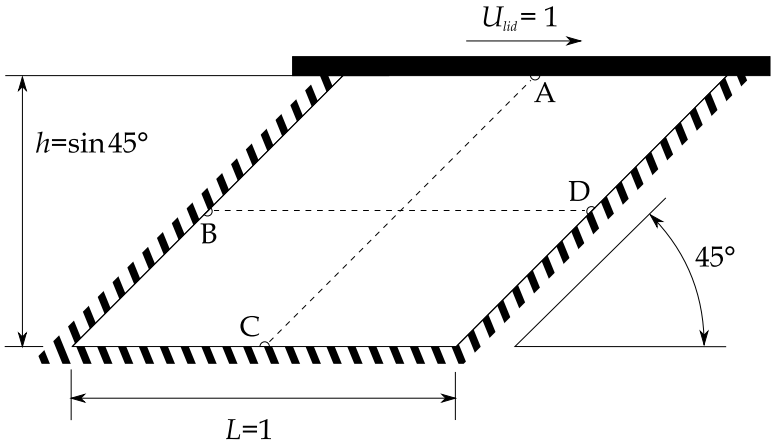


Figure 3.1: Schematic of the lid-driven skewed cavity. The dashed lines and labeled points indicate paths along which a benchmark solution is available. Points A through D are labeled in line with Erturk and Dursun [11].

Dimensional analysis yields that the flow physics of this problem are a function of the Reynolds number, which, for this system, is defined as

$$Re = \frac{U_{lid} L}{\nu} \tag{3.1}$$

The problem begins with quiescent fluid inside the cavity, i.e. $u_1 = u_2 = 0$ for $t < 0$. At $t = 0$ the cavity lid starts to move generating flow inside the cavity. Eventually steady state is achieved and the flow ceases to evolve.

3.2 Benchmark

This problem has been investigated by Erturk and Dursun [11] using a fine mesh (512×512) to provide as accurate as possible results. Erturk and Dursun [11] provide the steady state profiles of the velocity in the horizontal direction along a line reaching from the midpoint of the lid to the midpoint of the floor (AC in Figure 3.1) and the velocity in the vertical direction across a horizontal line at exactly half the height of the cavity (BD in Figure 3.1). This data will be used to verify that the FS and SIMPLE codes converge to a correct steady state solution for Re = 100 and Re = 1000.

3.3 Computational Mesh

The mesh for this case is a uniformly distributed skewed mesh. Each mesh of this type is constrained by the number of cells per side. Figure 3.2 shows a 40 cell mesh for this case.

3.4 Definition of Summary Properties

The error in the solutions will be judged based on the difference between the approximate flow variable values, ϕ and the benchmark solution, Φ. That is,

$$e_i = \Phi_i - \phi_i \tag{3.2}$$

The average error is found as

$$\bar{e} = \frac{1}{N} \sum_{i=1}^{N} |e_i| \tag{3.3}$$

where N is the number of error values available. The maximum error is found as

$$e_{max} = \max\left(|e_i|\right) \tag{3.4}$$

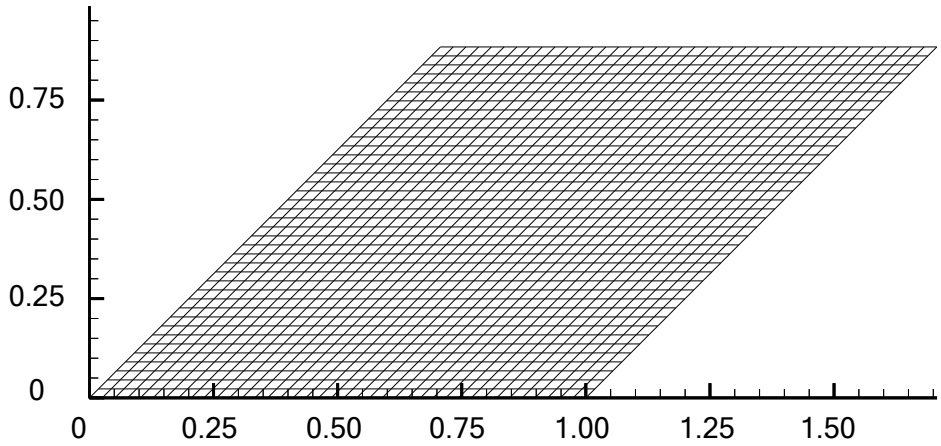


Figure 3.2: Lid-driven skewed cavity on a mesh of 40×40 cells.

In addition to the average of the local error, $\bar{e}$, and the maximum error, e_{max}; the root mean square (rms) of the error is a useful way of categorizing error as it is more sensitive to large error values. The rms error is defined as

$$e_{rms} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} e_i^2} \tag{3.5}$$

3.5 Test Matrix

A variety of CFD simulations were completed on this test arrangement. The simulations were stopped when the $\max(\Delta u_1, \Delta u_2) < 10^{-8}$ indicating that steady state has been achieved. Simulations were performed at $Re = 100$ and $Re = 1000$ using both time-stepping schemes. Mesh sizes of 20, 40 and 80 cells per side were used to study the effects of grid size. For the case of Re = 100, the maximum value of Δt that would remain stable was 0.001; at $\Delta t = 0.0011$ the simulation would diverge and eventually eventually causing overflow errors and halting the simulation. For comparison sake, the initial SIMPLE and FS simulations were conducted at $\Delta t = 0.001$. For the Re = 1000 case, a time step of 0.005 was stable for both FS and SIMPLE solvers, and this value was used for initial simulations. Unlike the FS code, the SIMPLE code was stable for large values of Δt. In order to

take advantage of this feature, the SIMPLE code was run at $\Delta t = 10^{30}$, effectively removing the transient terms from the flow equations and allowing the code to work in steady state (SS) mode. Results from these tests are listed and discussed in Section 3.6. A complete list of the simulations completed are listed in Table 3.1.

Grid	Δt	Re	Time-stepping scheme
80^2	0.001	100	SIMPLE
40^2	0.001	100	SIMPLE
20^2	0.001	100	SIMPLE
40^2	10^{30}	100	SIMPLE
80^2	0.001	100	FS
40^2	0.001	100	FS
20^2	0.001	100	FS
80^2	0.005	1000	SIMPLE
40^2	0.005	1000	SIMPLE
20^2	0.005	1000	SIMPLE
40^2	10^{30}	1000	SIMPLE
80^2	0.005	1000	FS
40^2	0.005	1000	FS
20^2	0.005	1000	FS

Table 3.1: Parameters and time-stepping schemes used for lid-driven skewed cavity tests.

3.6 Results

The flow patterns generated by the SIMPLE and FS cases were nearly identical at steady state as would be expected. The streamlines of the velocity profiles as determined using the SIMPLE code are plotted in Figures 3.3 and 3.4 for Re = 100 and Re = 1000 respectively.

Both the Re = 100 and Re = 1000 streamline plots agree fairly well with what would be expected for lid-driven skewed cavity flow. They are qualitatively similar to the results reported by Erturk and Dursun [11]. In order to gain a more quantitative grasp on the error in the current study, the centerline profiles at steady state are compared for Reynolds numbers of 100 and 1000 are shown in Figures 3.5 and 3.6 respectively along with the benchmark data [11].

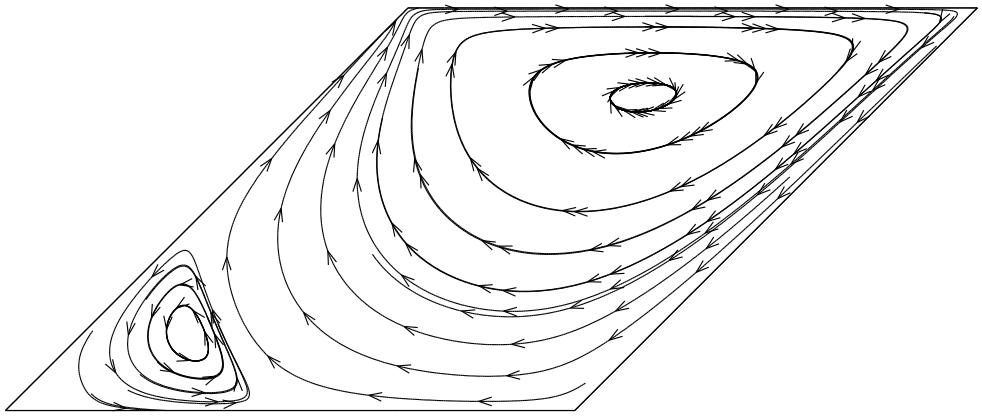


Figure 3.3: Lid-driven skewed cavity streamlines for Re = 100, calculated using SIMPLE.

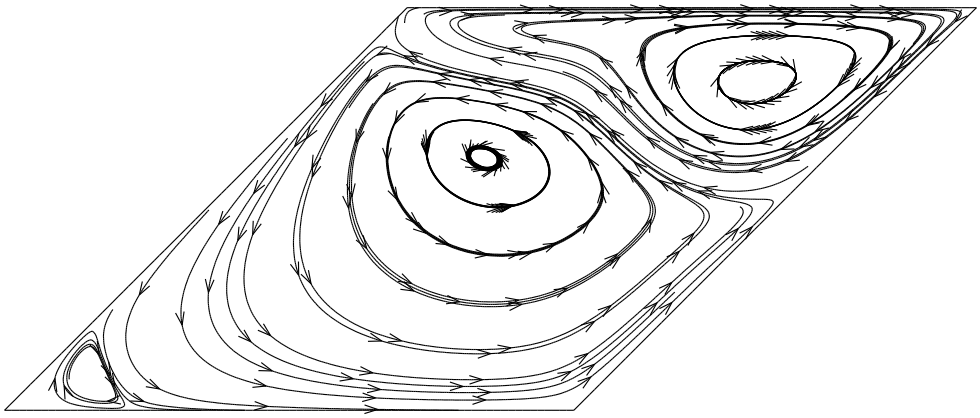


Figure 3.4: Lid-driven skewed cavity streamlines for Re = 1000, calculated using SIMPLE.

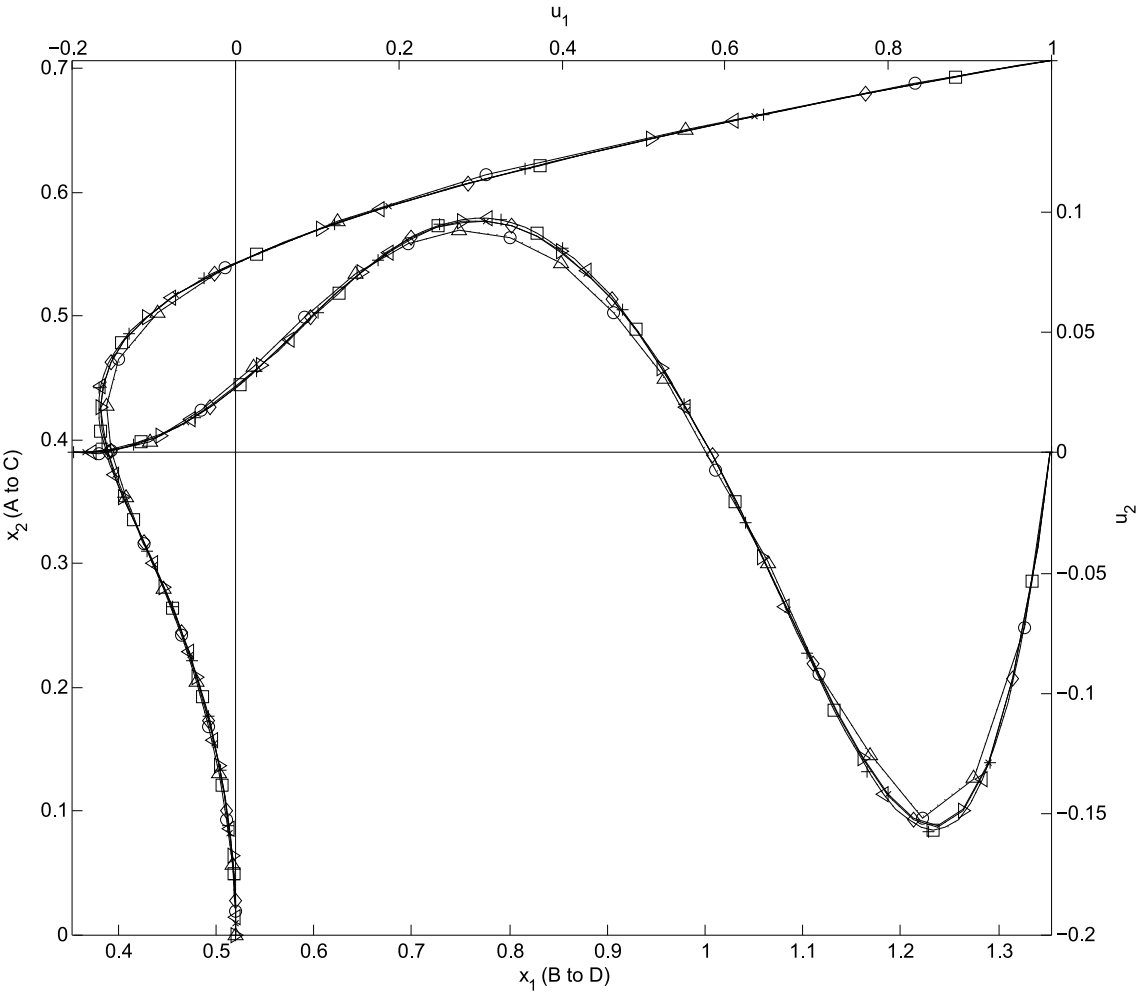


Figure 3.5: Comparison of centerline velocity profiles for Re = 100 at $\Delta t = 0.001$ with benchmark values [11] marked as +; SIMPLE runs, marked with solid lines, using 80, 40 and 20 cell meshes marked as $\square$, $\diamond$ and $\circ$ respectively; Fractional Step runs marked with dotted lines using 80, 40 and 20 cell meshes marked as $\triangleleft$, $\triangleright$ and $\triangle$ respectively. Finally a SIMPLE run with $\Delta t = 10^{30}$ for one time step using a 40 cell mesh is marked as $\times$.

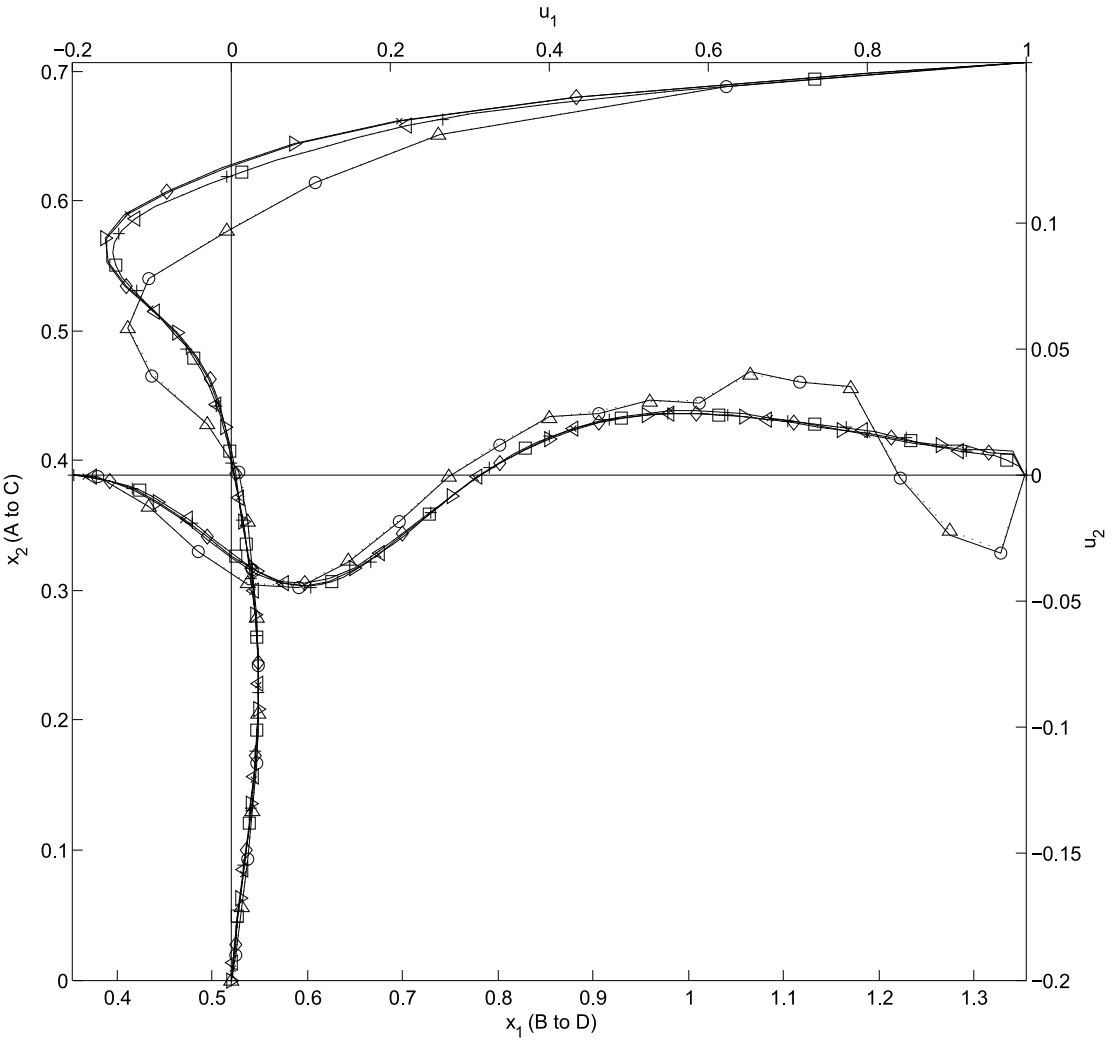


Figure 3.6: Comparison of centerline velocity profiles for Re = 1000 at $\Delta t = 0.005$ with benchmark values [11] marked as +; SIMPLE runs, marked with solid lines, using 80, 40 and 20 cell meshes marked as $\square$, $\diamond$ and $\circ$ respectively; Fractional Step runs marked with dotted lines using 80, 40 and 20 cell meshes marked as $\triangleleft$, $\triangleright$ and $\triangle$ respectively. Finally a SIMPLE run with $\Delta t = 10^{30}$ for 1 time step using a 40 cell mesh is marked as $\times$.

For the Re = 100 plots shown in Figure 3.5, results obtained with all three mesh densities seem to agree well with the benchmark data. As is expected, the closest match to the benchmark is provided by the 80 cell mesh while the most error is present in the 20 cell mesh. In addition, at steady state, the FS and SIMPLE codes provide nearly identical results, increasing confidence in the results of both codes. The SIMPLE SS run is also nearly identical to the other 40 cell mesh solutions.

For the Re = 1000 plots shown in Figure 3.6, the 40 and 80 cell mesh solutions seem to match the benchmark solution while the two 20 cell solutions differ a significant amount from the benchmark solution, especially in the high gradient region in the upper half of the u_1 profile and in the right hand third of the u_2 plot. For all the profiles, solutions depend much more on the grid density than on the time-stepping scheme used to generate them, suggesting that the steady state accuracy is the same for both FS and SIMPLE codes, even compared with SIMPLE in SS and transient modes.

Using cubic spline interpolation between the solution profile points, the error from the benchmark solution is calculated according to Equation 3.2. It should be noted that the benchmark data is reported by Erturk and Dursun [11] to 4 significant digits limiting the accuracy of error calculations as the error becomes smaller. Average, maximum and rms error values for the profiles shown in Figures 3.5 and 3.6 are listed in Tables 3.2 and 3.3 respectively.

Solver	Grid	$\bar{e}$		e_{max}		e_{rms}	
		u_1	u_2	u_1	u_2	u_1	u_2
FS	20^2	5.71E-3	3.71E-3	1.85E-2	8.98E-3	7.91E-3	4.39E-3
	40^2	1.50E-3	9.21E-4	4.19E-3	1.93E-3	2.07E-3	1.10E-3
	80^2	3.95E-4	2.39E-4	1.13E-3	5.48E-4	5.62E-4	2.99E-4
SIMPLE	20^2	5.66E-3	3.68E-3	1.94E-2	9.28E-3	7.97E-3	4.43E-3
	40^2	1.51E-3	9.22E-4	4.22E-3	1.95E-3	2.08E-3	1.11E-3
	80^2	4.04E-4	2.42E-4	1.15E-3	5.62E-4	5.74E-4	3.05E-4
SIMPLE SS	40^2	1.48E-3	9.20E-4	4.15E-3	1.90E-3	2.04E-3	1.09E-3

Table 3.2: Error values for lid-driven skewed cavity flow compared against the benchmark solution at Re = 100 [11].

The error values listed in Tables 3.2 and 3.3 confirm that the calculated steady state solutions do not vary appreciably between FS, transient SIMPLE and SS SIMPLE. For each mesh density, the error values are almost identical across the three

Solver	Grid	$\bar{e}$		e_{max}		e_{rms}	
		u_1	u_2	u_1	u_2	u_1	u_2
FS	20^2	3.05E-2	1.08E-2	1.29E-1	3.84E-2	5.38E-2	1.40E-2
	40^2	7.88E-3	8.27E-4	4.80E-2	1.79E-3	1.58E-2	9.82E-4
	80^2	9.26E-4	2.09E-4	4.15E-3	6.48E-4	1.41E-3	2.87E-4
SIMPLE	20^2	3.09E-2	1.09E-2	1.32E-1	3.99E-2	5.47E-2	1.44E-2
	40^2	7.88E-3	8.29E-4	4.79E-2	1.83E-3	1.58E-2	9.94E-4
	80^2	9.30E-4	2.03E-4	3.81E-3	6.35E-4	1.37E-3	2.83E-4
SIMPLE SS	40^2	7.61E-3	8.28E-4	4.71E-2	1.76E-3	1.53E-2	9.53E-4

Table 3.3: Error values for lid-driven skewed cavity flow compared against the benchmark solution at Re = 1000 [11].

solution methods with most solutions differing by less than 2% and a maximum difference in error values of 8% occurring in the maximum error of the u_1 curve for Re = 1000 with an 80×80 cell grid. Those error values are on the order of 10^{-3}, which is close to the benchmark solution accuracy. Across the board, errors decrease with an increase in mesh density as expected. As well, magnitudes of average error are the lowest, max error are the largest and RMS error falls between the two.

While the solution error seems to be much more dependent on grid density than on the time-stepping scheme used to calculate the solution, the time to solution generation is drastically different between FS, SIMPLE transient and SIMPLE SS. To generate a solution, the largest portion of CPU time is generally taken up by the iterative simultaneous solution of flow variables. For this study, a linear TDMA solver was used to solve the large matrix equations generated by the FS and SIMPLE codes. The solver computes a solution by iteratively sweeping the rows of matrix equation until a converged solution is reached. In general, the fewer sweeps the solver needs to make, the more efficiently a solution can be generated. Along these lines, the count of solver sweeps can be related to the amount of CPU time required to generate a solution, and is in turn a measure of the computational (and monetary) cost of the simulation.

The solver sweeps for the transient SIMPLE and FS solutions are plotted versus time step in Figures 3.7 and 3.8 for Re = 100 and Re = 1000 respectively. Based on Figures 3.7 and 3.8, the FS solutions tend to take much more solver sweeps to complete the initial time steps than the transient SIMPLE solutions. This is likely

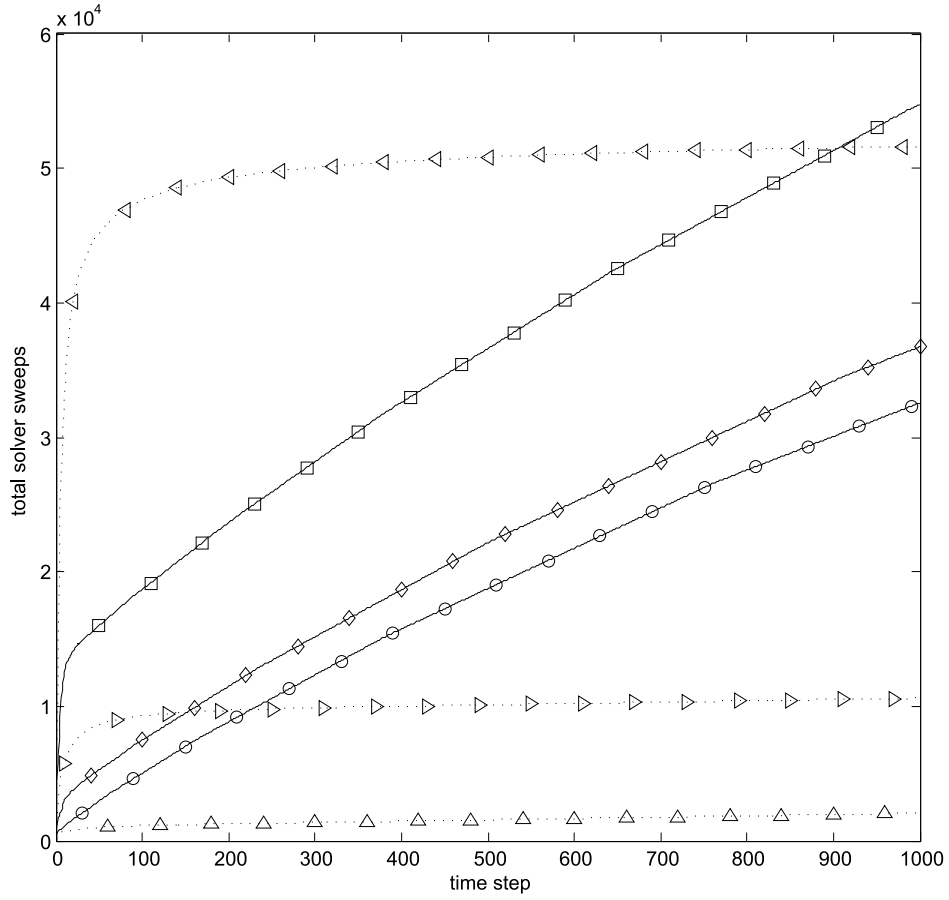


Figure 3.7: Comparison of the cumulative total solver sweeps versus time step for laminar flow in a lid-driven skewed cavity with Re = 100 and $\Delta t = 0.001$. SIMPLE runs are marked with solid lines, using 80, 40 and 20 cell meshes marked as $\square$, $\diamond$ and $\circ$ respectively; Fractional Step runs marked with dotted lines using 80, 40 and 20 cell meshes marked as $\triangleleft$, $\triangleright$ and $\triangle$ respectively.

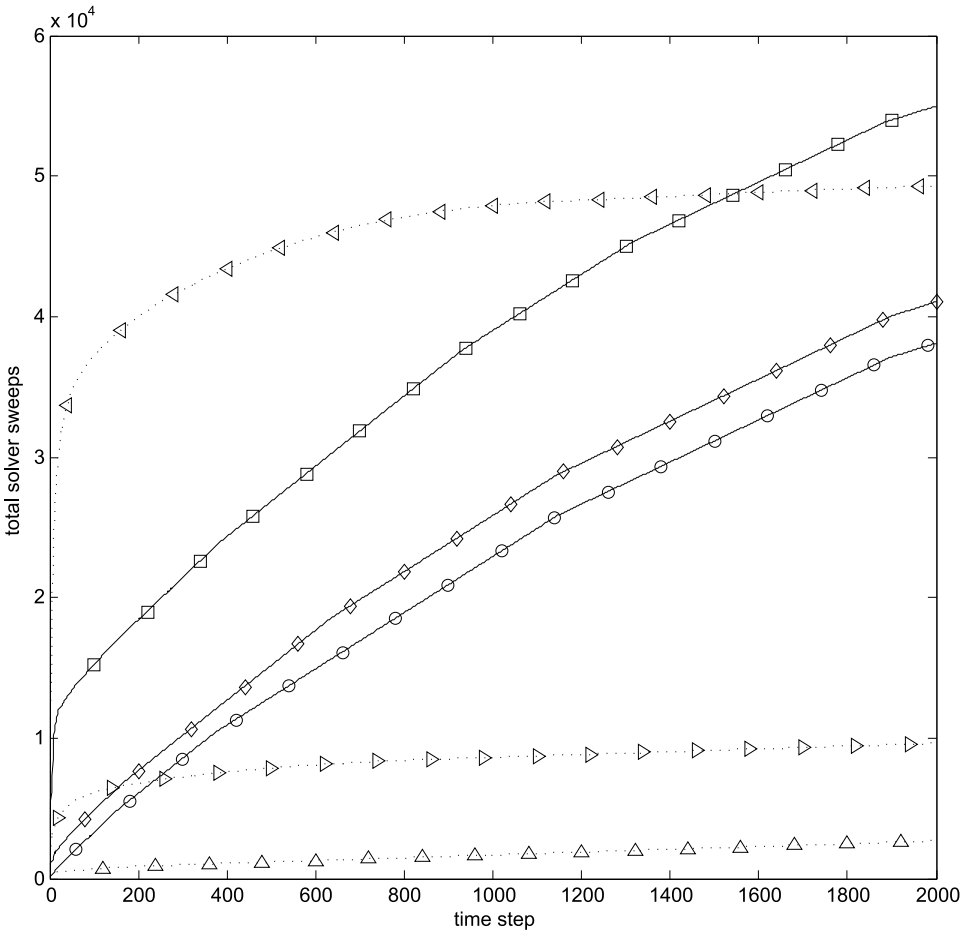


Figure 3.8: Comparison of the cumulative total solver sweeps versus time step for laminar flow in a lid-driven skewed cavity with Re = 1000 and $\Delta t = 0.005$. SIMPLE runs are marked with solid lines, using 80, 40 and 20 cell meshes marked as $\square$, $\diamond$ and $\circ$ respectively; Fractional Step runs marked with dotted lines using 80, 40 and 20 cell meshes marked as $\triangleleft$, $\triangleright$ and $\triangle$ respectively.

due to very strong non-physical gradients, in the u_1 velocity field close to the lid for the first few time steps. Especially for the first time step, the lid velocity is unity while the fluid inside the cavity has a uniform velocity of 0. It is up to the solver to fit a pressure field which is consistent to the previous velocity field before the solution can move on to the next time step. SIMPLE does not have the same problem at the beginning as it can adjust both the current velocity and pressure fields at the new time step to be consistent with the non-physical initial condition. This causes a much shallower slope of the solver efficiency trends for the SIMPLE simulations, shown on the left side of Figures 3.7 and 3.8. The poor efficiency of the FS solver in the first few time steps is most pronounced in the densest grids, which supports the speculation that initial large gradients close to the lid are to blame for this poor efficiency as Δx_2 decreases as grid density increases, increasing the approximated gradient at the lid.

After the initial costly time steps have passed, the FS solver behaves more efficiently than the SIMPLE solver, requiring much fewer sweeps per time step than the SIMPLE solutions. This is likely due to the need for the SIMPLE solver to solve 3 equations (u_1, u_2 and P') for each time step while the FS solver only needs to solve a P equation. Analysis of the raw data (not shown here) confirms that after some initial number of time steps both the SIMPLE and FS solvers do the minimum number of sweeps per time step, giving the FS solver a huge advantage in efficiency after the initial time steps have passed.

Based on the above discussion it can be concluded that the choice of solver for a transient solution should take into account the length of time that needs to be simulated. If it is a small number of time steps, likely SIMPLE will be more efficient, whereas if it is a large number of time steps, FS will likely be more efficient. For all cases tested in this section, FS is always more efficient than transient SIMPLE overall because the number of time steps needed to achieve steady state is fairly large.

The total solver sweeps to steady state are listed in Tables 3.4 and 3.5. Listed along with the transient solutions are the results from the single time step SIMPLE SS run. The results in Tables 3.4 and 3.5 confirm that the FS solver is indeed more efficient in reaching steady state than the SIMPLE solver in transient mode. For both Reynolds numbers, FS ranges from 10 to 2.5 times more efficient than the SIMPLE solver with the greatest gains made in the coarse 20 cell meshes and the smallest gains in the 80 cell mesh. It should be noted that the SIMPLE solutions take more time steps to reach steady state in addition to requiring more solver sweeps. The time to reach steady state is more consistent with FS than with

Run Parameters		Run Length		Solver Sweeps			
Solver	Grid	n	t	P	u_1	u_2	Total
FS	20^2	9124	9.124	10178	0	0	10178
	40^2	9217	9.217	18822	0	0	18822
	80^2	9238	9.238	59857	0	0	59857
SIMPLE	20^2	13599	13.599	74421	24807	24807	124040
	40^2	14226	14.226	79947	26649	26649	133250
	80^2	16096	16.096	101070	33689	33689	168450
SIMPLE SS	40^2	1	10^{30}	8103	2701	2701	13505

Table 3.4: Summary for Re = 100 of time steps (n), simulated time (t), solver sweeps in the pressure or pressure correction (P) and velocity (u_1 and u_2) equations, and total solver sweeps carried out before steady state solution is reached. Criteria for steady state is $\max(\Delta u_1, \Delta u_2) < 10^{-8}$. The time step size was $\Delta t = 0.001$ for all cases except the SIMPLE SS case for which $\Delta t = 10^{30}$.

Run Parameters		Run Length		Solver Sweeps			
Solver	Grid	n	t	P	u_1	u_2	Total
FS	20^2	11544	57.72	12225	0	0	12225
	40^2	11891	59.455	19520	0	0	19520
	80^2	12428	62.14	59724	0	0	59724
SIMPLE	20^2	18380	91.9	74049	24683	24683	123420
	40^2	19396	96.98	78348	26116	26116	130580
	80^2	21995	109.97	94323	31441	31441	157210
SIMPLE SS	40^2	1	10^{30}	6948	2316	2316	11580

Table 3.5: Summary for Re = 1000 of time steps (n), simulated time (t), solver sweeps in the pressure or pressure correction (P) and velocity (u_1 and u_2) equations, and total solver sweeps carried out before steady state solution is reached. Criteria for steady state is $\max(\Delta u_1, \Delta u_2) < 10^{-8}$. The time step size was $\Delta t = 0.005$ for all cases except the SIMPLE SS case for which $\Delta t = 10^{30}$.

SIMPLE. Changing the grid density from 80^2 cells to 20^2 cells changed the SIMPLE results 27% for Re = 100 and 20% for Re = 1000 compared to 1% for Re = 100 and 8% for Re = 1000 for the FS results. This variance suggests that the time accuracy of SIMPLE may not be as good as that of FS, or that there are small oscillations that keep SIMPLE from converging to steady state. Plotting the maximum change in velocity over the simulation time indicates that oscillations are not present however, as shown in Figures 3.9 and 3.10.

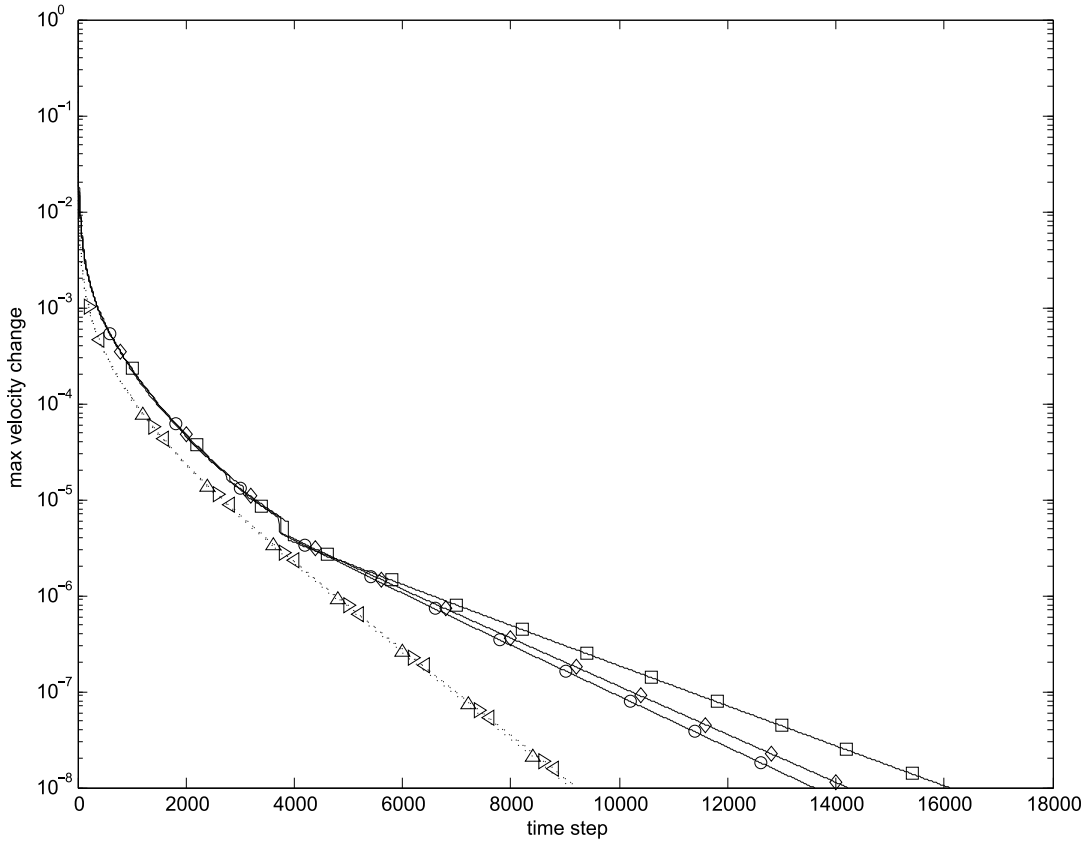


Figure 3.9: Comparison of the maximum change in velocity versus time step for laminar flow in a lid-driven skewed cavity with Re = 100 and $\Delta t = 0.005$. SIMPLE runs are marked with solid lines, using 80, 40 and 20 cell meshes marked as $\square$, $\diamond$ and $\circ$ respectively; Fractional Step runs marked with dotted lines using 80, 40 and 20 cell meshes marked as $\triangleleft$, $\triangleright$ and $\triangle$ respectively.

Another point of interest raised by Tables 3.4 and 3.5 is that SIMPLE running in steady state mode is more efficient for finding the steady state solution than FS for both Reynolds numbers. At Re = 100 SIMPLE SS finishes faster on a 40 cell grid than the fastest transient solver on a 20 cell grid. This suggests that steady

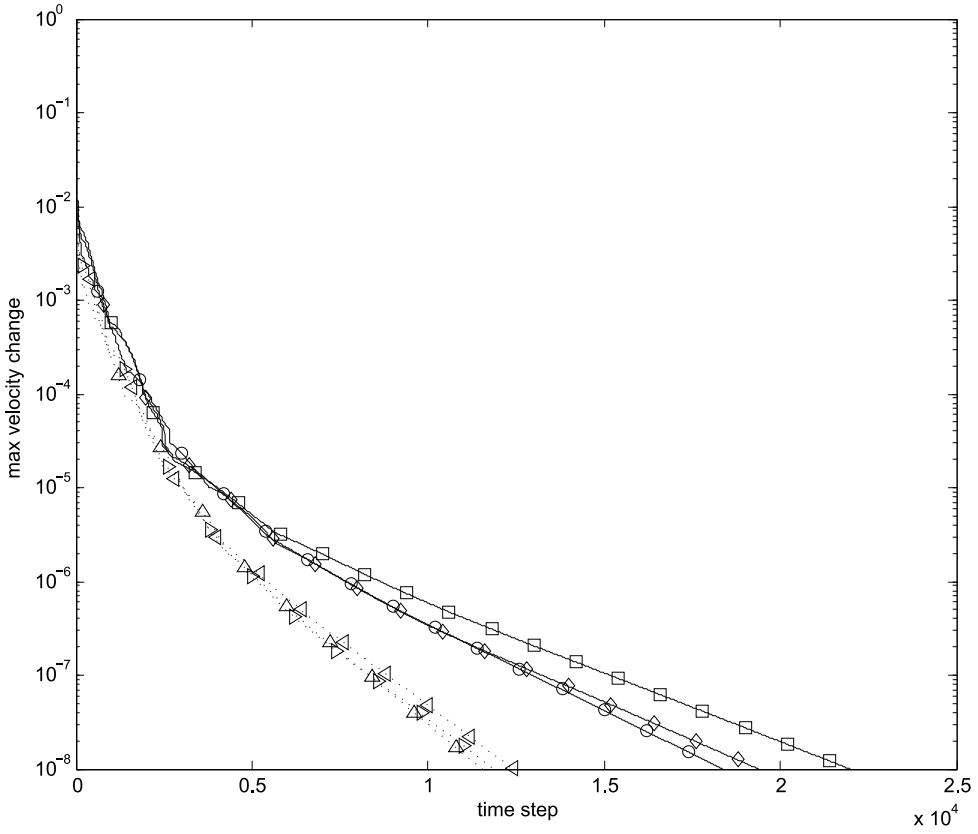


Figure 3.10: Comparison of the maximum change in velocity versus time step for Re = 1000 at $\Delta t = 0.005$. SIMPLE runs are marked with solid lines, using 80, 40 and 20 cell meshes marked as $\square$, $\diamond$ and $\circ$ respectively; Fractional Step runs marked with dotted lines using 80, 40 and 20 cell meshes marked as $\triangleleft$, $\triangleright$ and $\triangle$ respectively.

state solvers with the time terms removed are more efficient for solving steady state problems, assuming that they are capable of solving the problem. The down side of the steady state solver is that it gives no indication of how long it will take to reach steady state.

3.7 Conclusions

The results in this section indicate that the accuracy of the SIMPLE and FS time-stepping schemes are equivalent when used to generate a steady state solution. Even when the grid is not fine enough to resolve major features in the flow, the solution is still nearly identical for either scheme on the same grid. Both codes also agree with the steady state SIMPLE code. In addition to being self consistent, the codes also agree with the benchmark solution of Erturk and Dursun [11] with levels of error dependent on the mesh size.

For steady state solution efficiency, SIMPLE running in steady state mode was found to be the most efficient followed closely by FS and then by transient SIMPLE which required between 2.5 to 10 times more solver sweeps than FS. This efficiency of the time-stepping scheme is somewhat dependent on the flow being simulated as the FS code requires more solver sweeps per time step at the beginning of this case. The beginning of this case is characterized by large changes in the flow structure over time and sharp velocity gradients near the lid. FS becomes more efficient than SIMPLE after the initial large changes in flow have ceased and the velocity gradients are smaller. The exact point where FS becomes more efficient is problem dependent. As a rule of thumb, SIMPLE is likely to be more efficient for short quickly changing flows or large time step transient simulations and FS is likely to be more efficient for long transient simulations and/or small time step simulations.

Due to numerical stability issues associated with the explicit FS time-stepping scheme, FS cannot produce results for time steps which are too large, putting it at a major disadvantage when steady state results are needed. It may be more accurate in time than SIMPLE as it showed less variance in its prediction of time to steady state. This issue is explored further in the next chapter of this work.

Chapter 4

Laminar Flow Over a Square Cylinder

4.1 Problem Definition

The second test case is a simulation of laminar flow over an infinitely long (2D) square cylinder. Flow enters from the left of the domain at U_∞ and exits from the right. No fluid escapes from the top or bottom of the test domain. A schematic of the flow domain is shown in Figure 4.1.

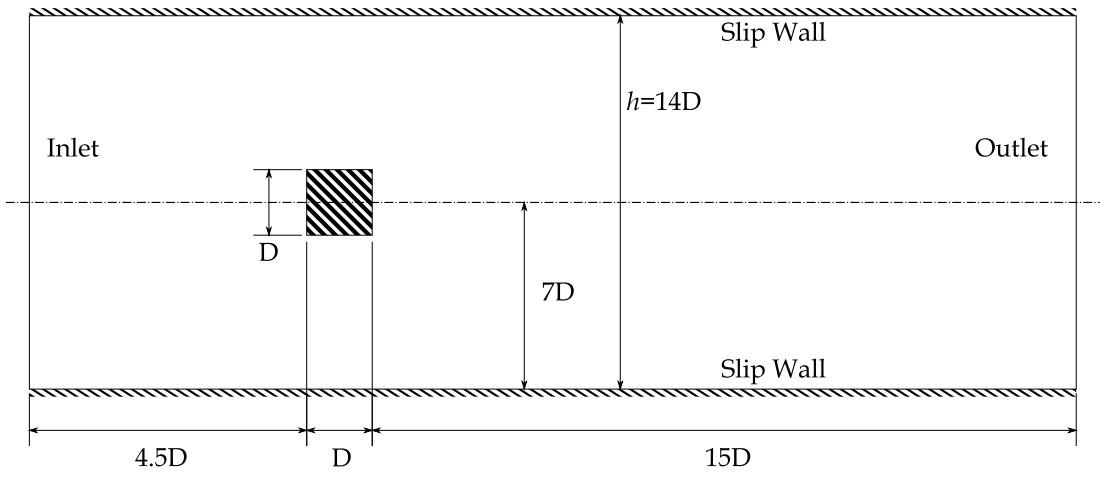


Figure 4.1: Schematic of square cylinder flow domain (not to scale.)

At $t < 0$ flow is quiescent everywhere in the flow domain, but at $t = 0$ fluid enters from the inlet at $u_1 = 1$. Dimensional analysis yields that the flow physics of

this system are a function of the Reynolds number as was the case for the lid-driven skewed cavity case. In this case, the Reynolds number is defined as

$$Re = \frac{U_\infty D}{\nu} \tag{4.1}$$

From experiments it is known that flow through this domain can be laminar at low Reynolds numbers and turbulent at high Reynolds numbers. At extremely low flow rates the flow achieves a steady, non-fluctuating state after some initial transient flow features have passed. At higher Reynolds numbers the flow never reaches a true steady state but continues to shed vortices in a repeating pattern. This flow is unsteady and exhibits large eddies but not small eddies, allowing for all flow features to be resolved without the use of turbulence models. A Reynolds number of Re = 100 fits within the laminar unsteady range, making it a suitable candidate for the present study. It is all the more suitable as it has been well studied in the literature, i.e. Sohankar et al. [35].

4.2 Computational Mesh

The mesh for this study is made up of 70×100 rectangular cells with a concentration of cells around the square cylinder in order to resolve the interaction of the flow with the cylinder. This mesh is very similar to the mesh used by Thompson [38] to study vortex shedding from a square cylinder in cross flow. Using the same base code as the SIMPLE code used here, Thompson determined that this mesh was capable of producing reasonably accurate results. In addition, doubling the grid density did not appreciably change the results suggesting a grid independent solution was reached. The mesh is shown in Figure 4.2.

4.3 Definition of Summary Properties

The instantaneous lift coefficient, C_L is a non-dimensional measure of the lift force applied to the cylinder by the fluid. For this case, the lift coefficient per unit depth due to pressure is defined as

$$C_{L,P} = \frac{L_P}{\frac{1}{2}\rho U_\infty^2 D} \tag{4.2}$$

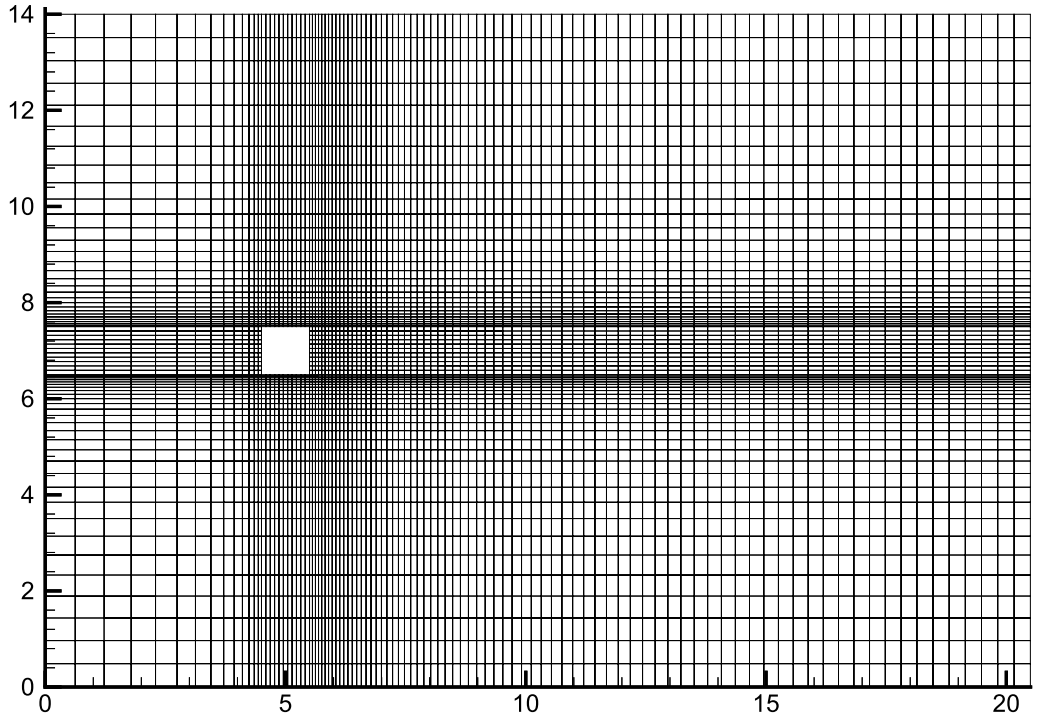


Figure 4.2: Mesh used for square cylinder study.

where the lift due to pressure is

$$L_P = \int_{cylinder} P\,(\hat{n} \cdot \widehat{x_2})\,dx \tag{4.3}$$

where $\hat{n}$ is the unit normal pointing into the surface of the cylinder and $\widehat{x_2}$ is a unit vector pointing in the x_2 direction. This value can be determined per unit depth of cylinder by summing the product of the face pressure and length along the bottom of the cylinder and subtracting the product of the face pressure and length along the top of the cylinder.

The time average of the lift coefficient is found as

$$\overline{C_{L,P}} = \frac{\sum C_{L,P}}{N} \tag{4.4}$$

where N is the total number of time steps in the averaged range. The rms of the lift coefficient is found as

$$C_{L,P_{rms}} = \sqrt{\frac{\sum \left(C_{L,P} - \overline{C_{L,P}}\right)^2}{N}} \tag{4.5}$$

Adding the lift due to wall shear stress to the lift due to pressure provides the overall lift coefficient, C_L.

The drag coefficient is found in a similar way as the lift coefficient; Equation 4.2 is modified by replacing the lift, L_P with drag by substituting $\widehat{x_1}$ into Equation 4.3 for $\widehat{x_2}$. Mean and rms drag coefficient values are determined using similar equations to Equations 4.4 and 4.5. The overall lift and drag coefficients can be found by including the wall shear forces, however, much of the literature reports only the pressure component, and the contribution of the wall shear forces is small enough to be negligible in most cases.

The Strouhal number is a non-dimensional frequency of vortex shedding. It is defined as

$$St = \frac{fD}{U_\infty} \tag{4.6}$$

where f is the vortex shedding frequency as determined by the frequency of the C_L signal.

4.4 Test Matrix

For this problem, two simulations were of primary interest. One with the SIMPLE code and the other with the FS code. The simulations were run from quiescent conditions at $t = 0, n = 0$ until $t = 312.5, n = 25000$ at which point the solutions could be assumed to have reached a quasi-steady state for a majority of this solution. The parameters for the two original simulations of interest are listed in Table 4.1.

Δt	Re	Time-stepping scheme
0.0125	100	SIMPLE
0.0125	100	FS

Table 4.1: Parameters and time-stepping schemes used for laminar flow over a square cylinder simulations.

The computational domain is symmetrical and at this low of a Reynolds number the flow will not oscillate unless some non-symmetry is present in the solution for a few time steps. This was introduced through a skewed inlet velocity profile, applied for the first 40 time steps. The skewed profile takes the form

$$U(x_2) = U_\infty \left(\frac{5}{4} - \frac{x_2}{2h} \right) \tag{4.7}$$

4.5 Results

The solutions generated by both the SIMPLE and FS code are quite similar when assessed based on the instantaneous streamline plots and pressure contours. Both solutions reach a quasi-steady state when presented with a non-symmetrical initial flow condition. The solution as computed by the FS code is shown in Figure 4.3 at a point in time after the flow has developed into its quasi-steady state.

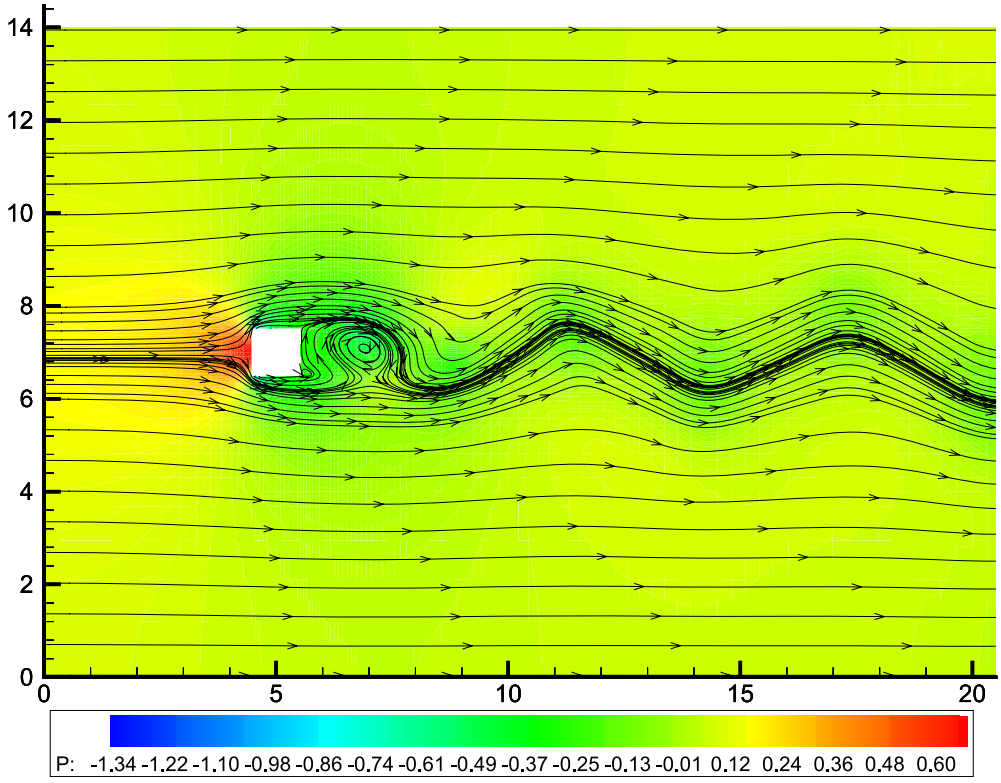


Figure 4.3: Instantaneous pressure contours and streamlines for a square cylinder in cross flow at Re = 100 calculated using FS time-stepping. Contours and streamlines are shown at a point in time after quasi-steady flow has been achieved.

The unstable vortex shedding can be clearly seen in Figure 4.3 with a new vortex being shed from the bottom of the cylinder and the previous vortex, which was shed from the top, still visible downstream of the cylinder. The shedding of vortices corresponds to oscillations in the cylinder's lift and drag coefficients which are shown in Figures 4.4 and 4.5 respectively for a small portion of their time histories. The summary properties of quasi-steady portions of the C_L and $C_{D,P}$ histories are shown in Table 4.2.

Table 4.2 demonstrates that although there are some minor differences between

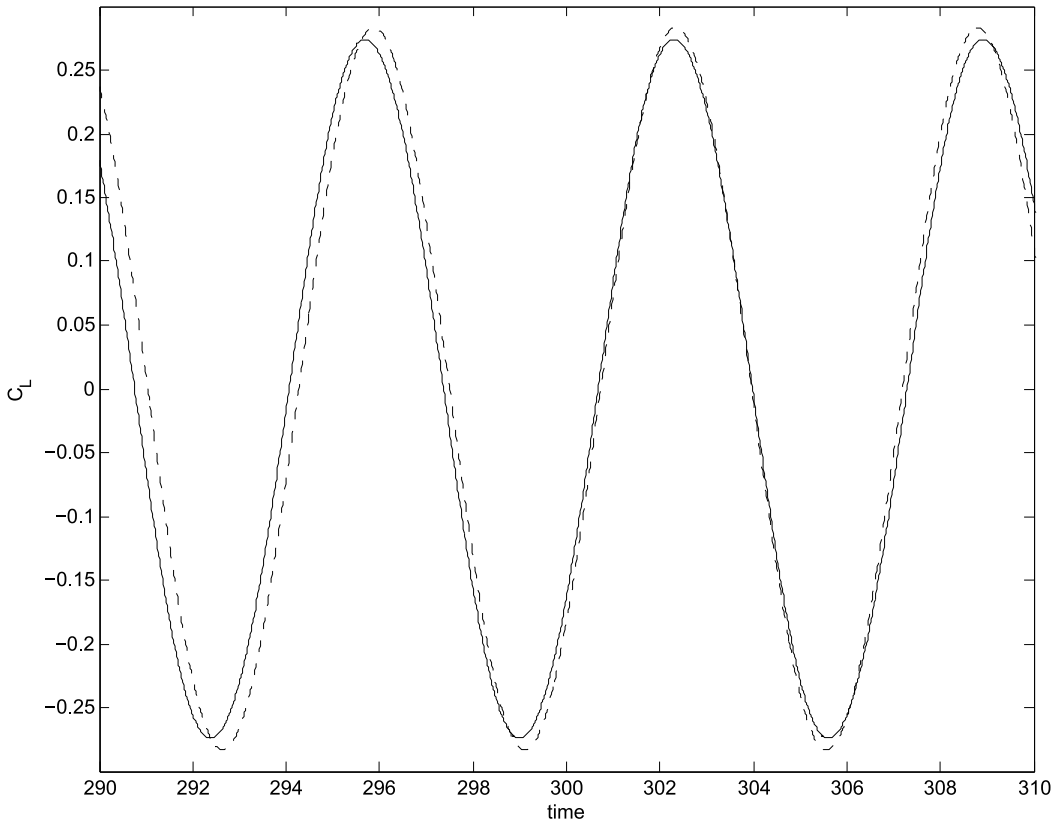


Figure 4.4: Lift coefficient versus time for the laminar square cylinder test case. The SIMPLE solution is shown by a solid line while the FS solution is shown by a dashed line.

Time-stepping scheme	St	$\overline{C_L}$	C_{Lrms}	$\overline{C_{D,P}}$	$C_{D,P_{rms}}$
SIMPLE	0.156	-7.67×10^{-7}	0.194	1.652	5.80×10^{-3}
FS	0.160	4.26×10^{-5}	0.199	1.634	6.69×10^{-3}

Table 4.2: Summary properties results for laminar flow over a square cylinder at $\mathrm{Re} = 100$.

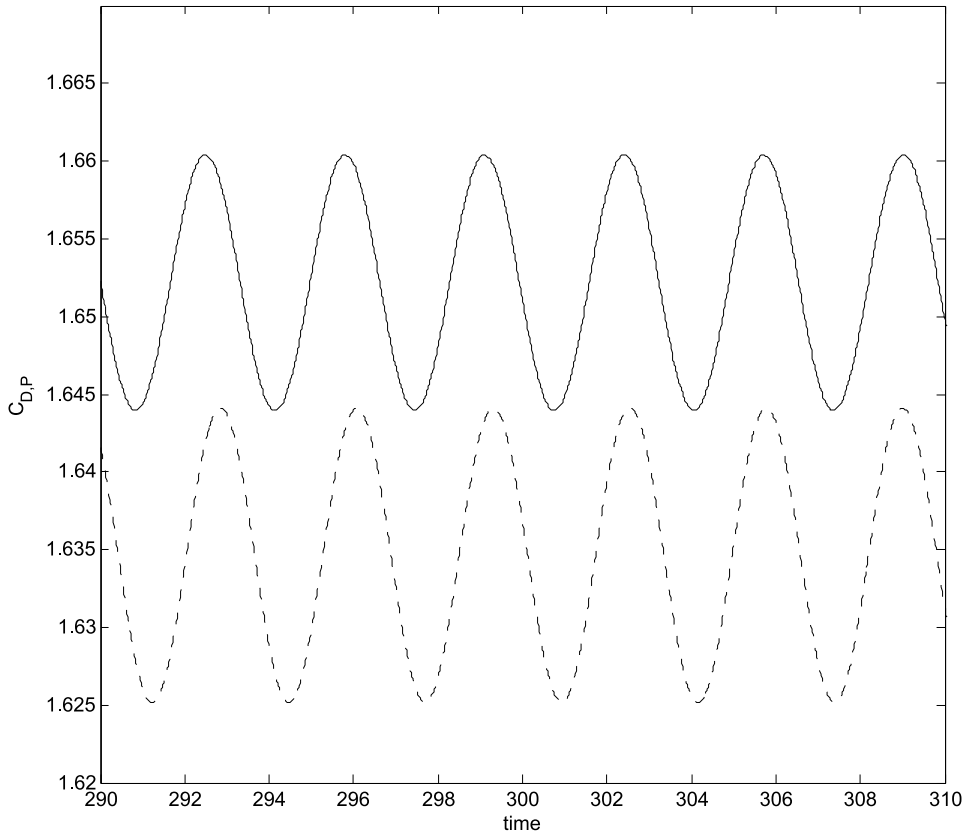


Figure 4.5: Drag coefficient versus time for the laminar square cylinder test case. The SIMPLE solution is shown by a solid line while the FS solution is shown by a dashed line.

the two solutions they are quite similar. The value of the Strouhal number for the FS code is larger indicating that the predicted vortex shedding rate is slightly higher, although this difference is slight as the Strouhal number only varies by 2.5%. This faster flow evolution seems to correspond to the faster convergence to steady state of the FS code compared to the SIMPLE code observed in Chapter 3. Both solvers predict that $\overline{C_L} \approx 0$ as is expected for a symmetrical body in cross flow at zero angle of attack. The values of C_{Lrms} are also nearly identical, varying by 2.5%. The values of $C_{D,P}$ shown in Figure 4.5 appear to be quite different in mean value due to the scale of the figure. However, the values of $\overline{C_{D,P}}$ only vary 1% between the two solvers. The largest percent variation is found in the values of $C_{D,P_{rms}}$ which vary by 15%. The amplitude of the fluctuations is much smaller compared to the total value of the drag coefficient, and the variance of these values is also smaller than the variance seen in the literature, as reported by Sohankar et al. [35].

Comparing the results in Table 4.2 to results in the surveyed literature (Table 1.1) suggests that most of the values are on the high end of the spectrum recorded in the literature. Both the Strouhal numbers and the C_{Lrms} values are slightly above the range reported previously. The St value predicted by the SIMPLE code was 0.6% greater than the maximum value presented by Sohankar et al. [35] while the FS code was higher, with a St value 3.5% greater than the maximum reported by Sohankar et al. [35]. Compared to the experimental St value of 0.145 presented by Sohankar et al. [35], the SIMPLE and FS codes over-predict by 7.6% and 10.3% respectively. This study's variation from the experimental values is less than the variation of the minimum value presented by Sohankar et al. which is 17.2% below the experimental value. The variation from experimental results of this study is on the same order as Okajima [30], 9.0% and Sohankar et al. [34], 6.9%. The values of Thompson [38] are 4.0% lower than the SIMPLE code results in the present study. Thompson completed his study using very similar code and identical flow domain geometry. Differences between the two codes suggest that the difference in St is due to differences in the mesh distribution or differences in the boundary conditions. In contrast to the present study, Thompson [38] used no-slip walls on the top and bottom flow boundaries and a zero velocity partial derivative in the flow direction at the outlet.

The values of C_{Lrms} presented in Table 4.2 are also above the values presented in the literature, however, in this case, no experimental values are available. In addition, of the studies surveyed, 4 of 7 reported C_{Lrms} values. Of the studies surveyed that did report C_{Lrms} values the variation was quite large. The difference

between values of C_{Lrms} presented by Sohankar et al. [34] and Franke et al. [13] is 133%, suggesting that this quantity is fairly sensitive and difficult to predict. The value reported by Thompson [38] is 32% smaller than the value generated by the SIMPLE results in Table 4.2, which is relatively low compared to the variation shown in the literature.

The values of $\overline{C_{D,P}}$ in Table 4.2 are on the high end of the range of values reported in the literature but are lower than the maximum value reported by Sohankar et al. [35]. The results $\overline{C_{D,P}}$ match best with the results of Davis and Moore [9] and are slightly higher than the results reported by Thompson [38].

$C_{D,P_{rms}}$ is rarely reported in the literature. Of the works surveyed, 2 of 7 reported results for this quantity. The present values of $C_{D,P_{rms}}$ for the SIMPLE case falls within the range of values presented by Sohankar et al. [34], while the FS value is slightly above the results of Sohankar et al. The rarity of this quantity in the literature and the large range reported by Sohankar et al. [34] suggests that, as was the case with C_{Lrms}, this quantity is difficult to compute and may be quite sensitive to minor changes in the grid or flow domain dimensions.

The efficiency of the two codes is initially measured by the number of solver sweeps performed by the TDMA solver to compute the solution. The cumulative solver sweeps for both the FS and SIMPLE runs up to $t = 150$ is shown in Figure 4.6. As shown in Figure 4.6, for $\Delta t = 0.0125$ the FS code takes more solver sweeps to solve the system than the SIMPLE code for the entire span of time simulated. For $\Delta t = 0.0125$ the FS code is also divergent from the SIMPLE code, increasing the gap between the two as time goes on. It should also be noted that the curves are steeper once the flow has developed into a quasi-steady fluctuating flow than they are in the initial transient portion before a quasi-steady state has been attained, indicating that the quasi-steady flow is more costly than the initial development period. The same is true for the smaller time step, $\Delta t = 0.00125$ except that in this case, SIMPLE is by far the most costly. For the smaller time step, the FS code is by far the most efficient, even passing the larger time step runs. Even though the code does 10 time steps for every time step the large time step code does, it has much fewer cumulative sweeps than both SIMPLE runs and the large time step FS run. To further examine the computational cost, the solver sweeps for each time step are shown along with the coefficient of lift for a portion of the quasi-steady flow history in Figure 4.7 for $\Delta t = 0.0125$ and Figure 4.8 for $\Delta t = 0.00125$.

Figure 4.7 indicates that the periods of high computational cost correspond to regions of large change to the lift coefficient. The peaks and troughs on the C_L curve

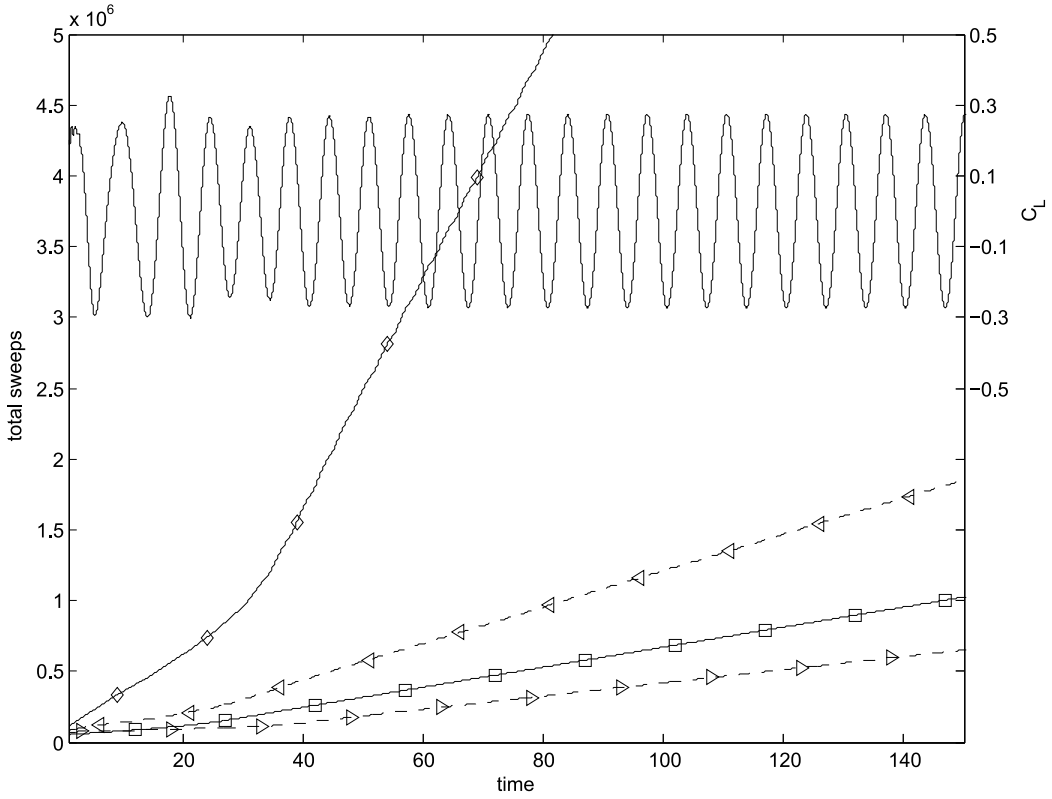


Figure 4.6: Cumulative solver sweeps for the laminar flow over a square cylinder case. The SIMPLE run is shown with a solid line and the FS run is shown with a dotted line. Markings are SIMPLE, $\Delta t = 0.0125$, $\square$; SIMPLE, $\Delta t = 0.00125$, $\diamond$; FS, $\Delta t = 0.0125$, $\triangleleft$; and FS, $\Delta t = 0.00125$, $\triangleright$. The C_L value is shown with a solid line in the upper portion of the figure and was calculated using the SIMPLE code at $\Delta t = 0.0125$.

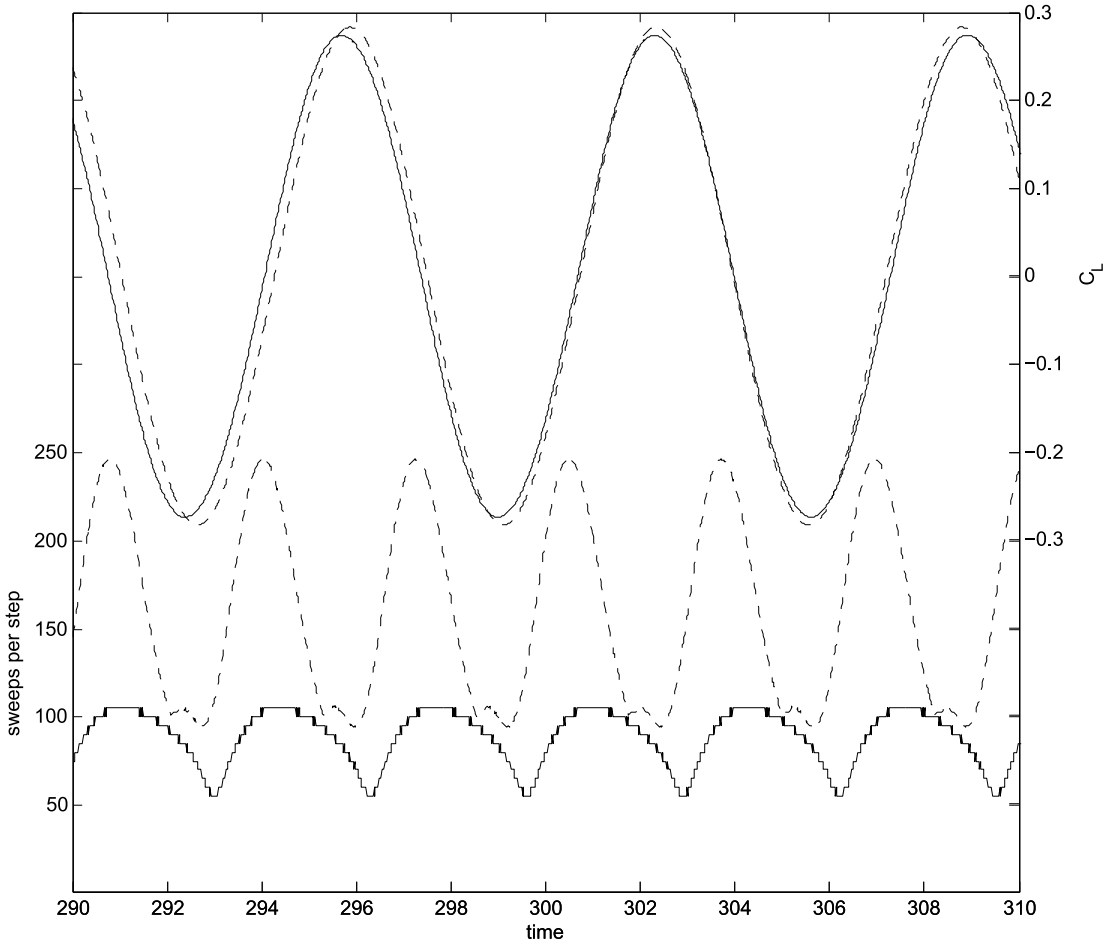


Figure 4.7: Solver sweeps per step for the laminar flow over a square cylinder case with $\Delta t = 0.0125$. The SIMPLE run is shown with a solid line and the FS run is shown with a dashed line. The upper curves correspond to C_L and the lower curves correspond to solver sweeps per time step.

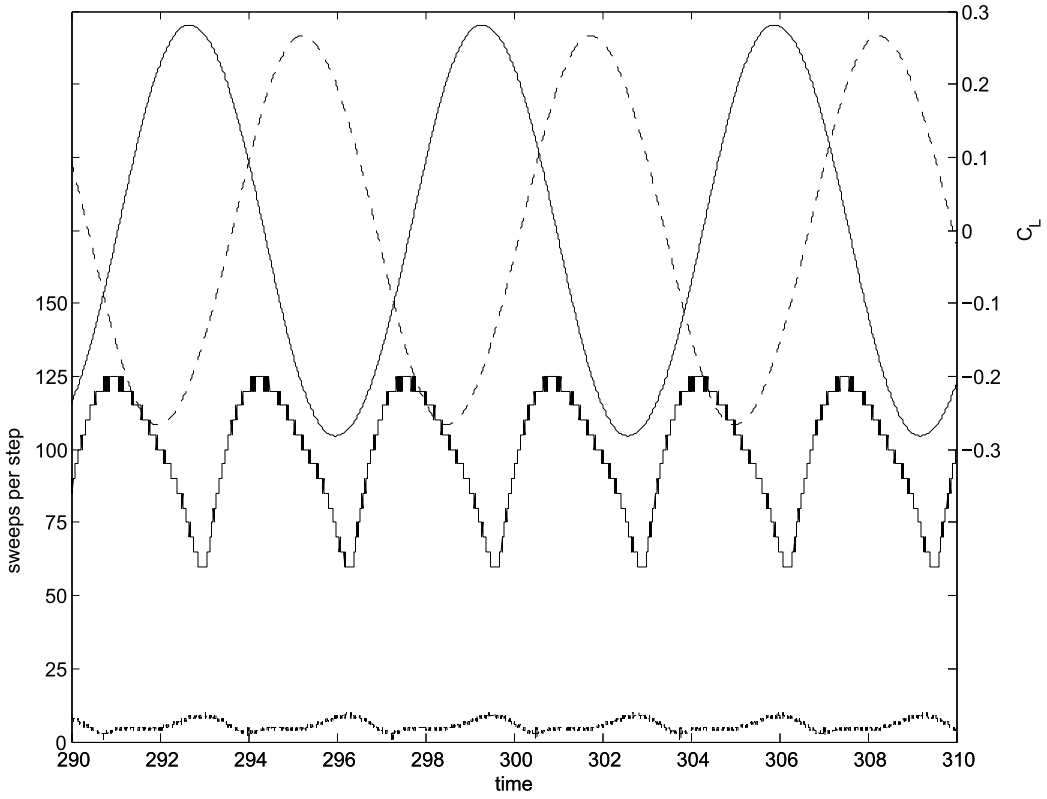


Figure 4.8: Solver sweeps per step for the laminar flow over a square cylinder case with $\Delta t = 0.00125$. The SIMPLE run is shown with a solid line and the FS run is shown with a dashed line. The upper curves correspond to $C_{L,P}$ and the lower curves correspond to solver sweeps per time step.

correspond to vortices traveling across the top of the cylinder and bottom of the cylinder respectively. The portions in between the peaks and troughs represent large vortices traveling downstream of the cylinder. These large vortices are likely the cause of the high computational cost per step as there are large changes occurring in a large number of cells from time step to time step. It should be noted that the peak solver sweeps per time step do not occur at precisely the same points in time for the two time-stepping schemes, relative to both simulated time and phase of the corresponding C_L signal. The SIMPLE solver reaches its peak load at a greater phase angle of its C_L signal than the FS code, further illustrating the difference between the two time-stepping schemes. FS time stepping is most costly when large changes are occurring in the solution domain, increasing computational cost in this case by approximately 2.5 times from best to worst performance, while SIMPLE increases approximately 2 times from best to worst performance. This is similar to the trend discussed in Chapter 3.

Comparing Figure 4.8 to Figure 4.7 illustrates the difference seen between the two time step sizes shown in Figure 4.6. For the $\Delta t = 0.00125$ case, the number of SIMPLE sweeps are much greater than that of FS, and are even greater than the number of sweeps recorded for the larger time step. More drastic than the increase in the SIMPLE sweeps is the decrease in FS sweeps with peak sweeps per time step dropping to 9 at $\Delta t = 0.00125$ from a value of just under 250 at $\Delta t = 0.0125$. This indicates that the efficiency of the FS time-stepping scheme is much more sensitive to variations in Δt. For this time step, again, the solver peaks seem to correspond to the portions of the C_L curve with the steepest slopes for both the SIMPLE and FS solutions.

Up to this point in this study, all simulations were completed using scientific clusters provided by SHARCNET [29], a consortium of computational clusters available at minimal charge to researchers. The side effect of using this resource is that the computational power of the processor that code is running on is difficult to quantify as the specific CPU and its loading is assigned by the cluster scheduling software. Through the process of generating summary property and solver load results, it seemed that the SIMPLE runs were taking longer than the FS runs even though the number of sweeps was typically much higher for the FS runs. This suggested that the actual CPU time was not accurately reflected by the number of sweeps performed by the matrix solver. An in-depth analysis of the time spent on the various parts of the solution procedure suggests a much different division of the CPU time than was assumed earlier in this work. The SIMPLE and FS codes were divided into sections and the time spent in each section was logged and then added

up at the end of the run. The results of this study are summarized in Figures 4.9 and 4.10 for the FS and SIMPLE codes respectively. It should be noted that the results reported here are quite rough as the system clock only reports events up to the millisecond and one solver sweep tends to take less than a millisecond. On average however, this study should give a good indication of the trends in CPU time.

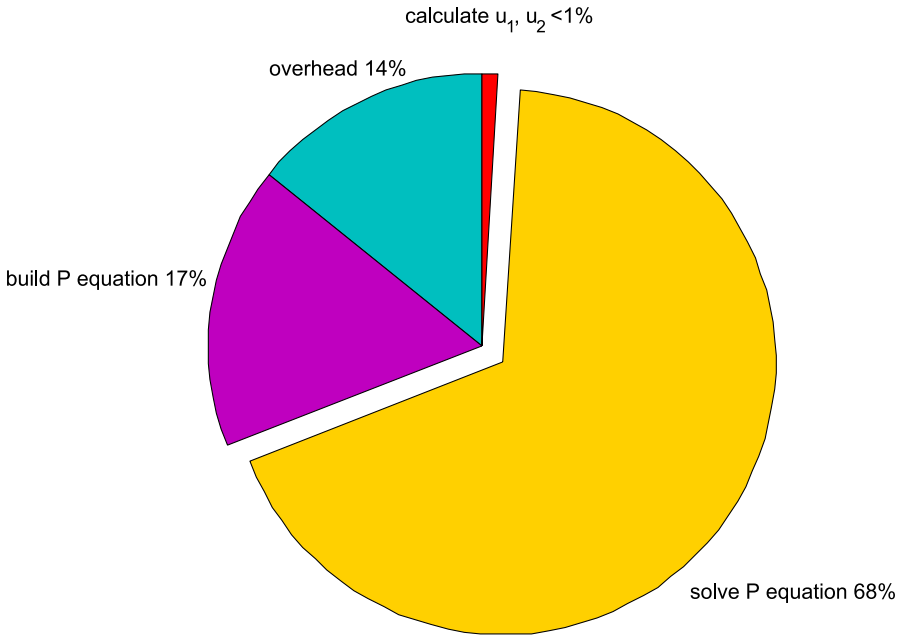


Figure 4.9: Division of CPU time for the FS code. Study was conducted for 8000 time steps at $\Delta t = 0.0125$.

The FS code time division is much closer to what was expected than the SIMPLE division of time. That is, the FS code spends the majority of its time, 68%, solving the P equation while the SIMPLE code spends the majority of its time setting up the equations with only 8% of the CPU time devoted to solving equations. This is in part due to the efficiency of the solver used in this case. If the grid were much more dense, the TDMA solver used here would not be as efficient and would likely dominate a larger portion of the CPU time. Another potential cause for this small cost for solving compared to the cost of setup is that the cost of setup is quite high for the SIMPLE scheme in general. In the current SIMPLE configuration, the solver does 1 sweep of the u_1 and u_2 equations and 3 sweeps of the P' equation before recalculating the equation coefficients. A new configuration was tested to determine if overall speed of solution could be increased for the SIMPLE code by taking advantage of the efficiency of the solver. The number of sweeps per internal

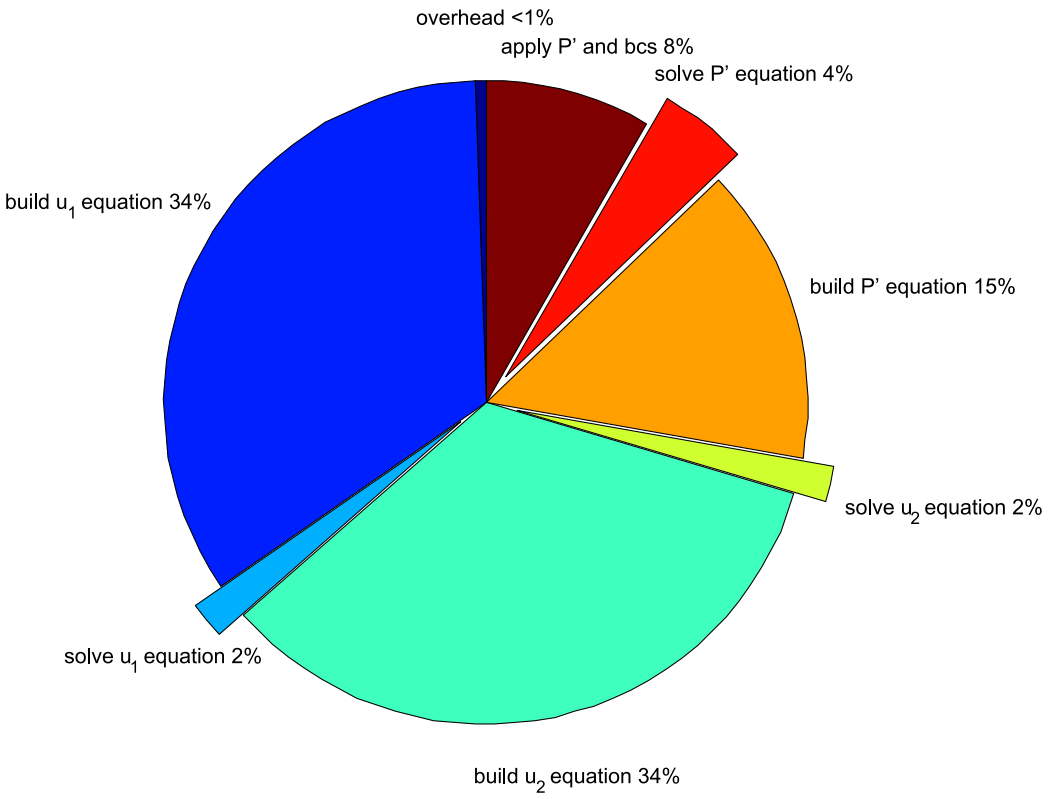


Figure 4.10: Division of CPU time for the SIMPLE code with the standard solver sweep configuration of 1 sweep for the u_1 equation, 1 sweep for the u_2 equation, and 3 sweeps for the P' equation. Study was conducted for 8000 time steps at $\Delta t = 0.0125$.

iteration was increased to 3 for the u_1 and u_2 equations and 12 for the P' equation. The results from this test are shown in Figure 4.11.

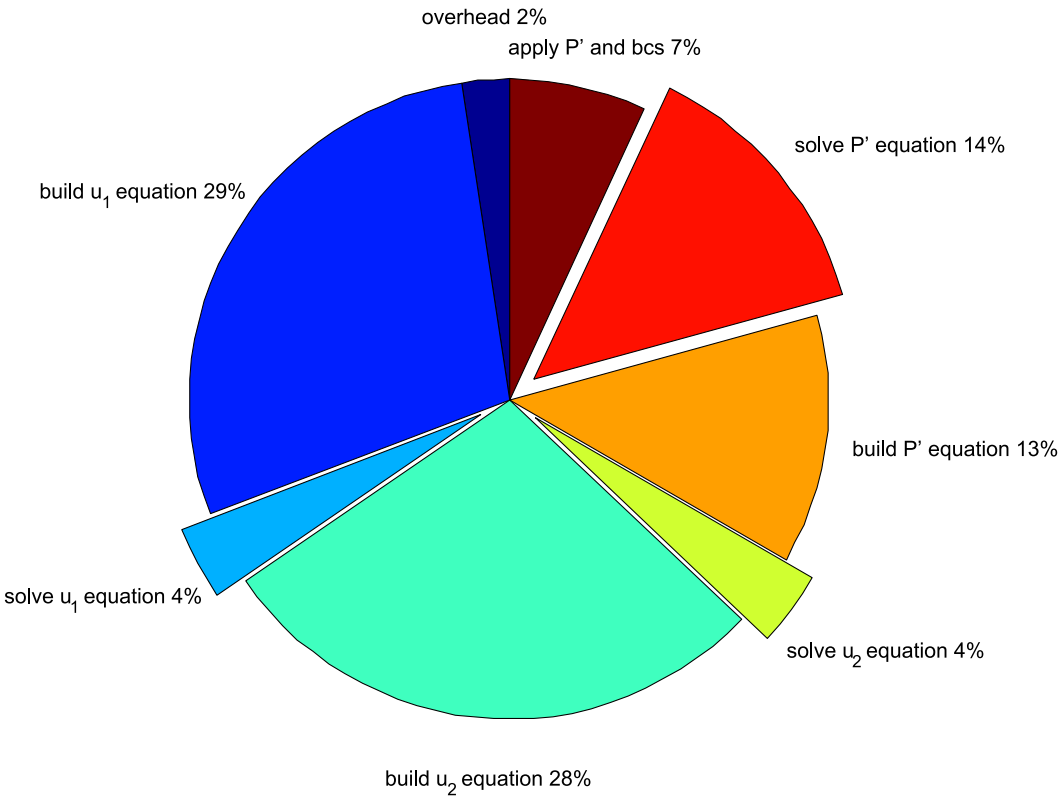


Figure 4.11: Division of CPU time for the SIMPLE code with a modified solver sweep configuration of 3 sweeps for the u_1 equation, 3 sweeps for the u_2 equation, and 12 sweeps for the P' equation. Study was conducted for 8000 time steps at $\Delta t = 0.0125$.

As seen in Figure 4.11, the overall cost of the solver did increase with the number of sweeps per internal loop, however it is difficult to determine from this chart if the overall efficiency is improved due to a lack of knowledge about what resources available for each run. To overcome this problem, a dedicated, single CPU machine was acquired and the SIMPLE and FS codes were run on it one configuration at a time. The total number of sweeps and wall clock times for each run are shown in Table 4.3. The FS solver and the SIMPLE solver in both sweep configurations were tested at $\Delta t = 0.00125$ and $\Delta t = 0.0125$ in runs up to $t = 200$. An additional run was done for the three solver configurations at $\Delta t = 0.0125$ up to $t = 250$ to gauge the effect of the length of the run on the time values and the effect of the initial unsteady transient portion. The values in Table 4.3 have also been normalized by dividing them by the values for the FS run in the second row of Table 4.3, these

values are shown in Table 4.4.

Solver	Δt	n	CPU Time	Sweeps	Loops	$\frac{\text{Sweeps}}{n}$	$\frac{\text{CPU time}}{\text{Sweeps}}$
FS	0.00125	160000	3239773	883983	160000	5.52	3.66
FS	0.0125	16000	1044451	2493258	16000	155.83	0.42
FS	0.0125	20000	1262714	3140403	20000	157.02	0.40
SIMPLE	0.00125	160000	46902803	14415560	2883112	90.10	3.25
SIMPLE	0.0125	16000	4712064	1379590	275918	86.22	3.42
SIMPLE	0.0125	20000	5784186	1732895	346579	86.64	3.34
SIMPLE2	0.00125	160000	60897304	58402548	3244586	365.02	1.04
SIMPLE2	0.0125	16000	5510142	5188806	288267	324.30	1.06
SIMPLE2	0.0125	20000	6983177	6557238	364291	327.86	1.06

Table 4.3: Summary of time study using identical resources for solution. CPU time is reported in ms and refers to actual wall clock CPU time. SIMPLE2 refers to the new SIMPLE TDMA solver sweep configuration of 3 sweeps for the u_1 equation, 3 sweeps for the u_2 equation, and 12 sweeps for the P' equation. Loops refer to the number of inner loop iterations performed for the test.

The primary observation to be made from Table 4.3 is that the efficiency results based on solver sweeps are not a conclusive indicator of the computational cost of a job. The fastest, least expensive simulation was the FS, $\Delta t = 0.0125$ simulation even though it had many more sweeps than the smaller time step FS run and the standard sweep (1 sweep each for the u_1 and u_2 equations, and 3 sweeps for the P' equation) SIMPLE runs. The reduction in the number of sweeps achieved by lowering the time step size of the FS code did not result in a reduction of CPU time. This is likely due to the increase in the number of time steps and, in turn, the number of times the equations must be assembled which was shown in Figure 4.9 to be a costly procedure.

The SIMPLE solver exhibits very different behavior from the FS solver. A reduction by 10 times in time step size did not improve the time convergence. Instead, it increased the solution CPU time by nearly 10 times, providing a CPU time per time step which is nearly identical for both large and small Δt sizes. Increasing the sweeps per step does decrease the amount of time per sweep required, however, the total number of sweeps increases as well. It was hypothesized earlier in this section that increasing the number of solver sweeps per inner loop iteration would decrease convergence time. This assumed that increasing the number of

Solver	Δt	n	CPU Time	Sweeps	Loops	$\frac{\text{Sweeps}}{n}$	$\frac{\text{CPU time}}{\text{Sweeps}}$	$\frac{\text{CPU Time}}{n}$
FS	0.1	10	3.10	0.35	10.00	0.04	8.75	0.31
FS	1	1	1.00	1.00	1.00	1.00	1.00	1.00
FS	1	1.25	1.21	1.26	1.25	1.01	0.96	0.97
SIMPLE	0.1	10	44.91	5.78	180.19	0.58	7.77	4.49
SIMPLE	1	1	4.51	0.55	17.24	0.55	8.15	4.51
SIMPLE	1	1.25	5.54	0.70	21.66	0.56	7.97	4.43
SIMPLE2	0.1	10	58.31	23.42	202.79	2.34	2.49	5.83
SIMPLE2	1	1	5.28	2.08	18.02	2.08	2.53	5.28
SIMPLE2	1	1.25	6.69	2.63	22.77	2.10	2.54	5.35

Table 4.4: Normalized summary of time study using identical resources for solution. Values are normalized by dividing by the second row values. SIMPLE2 refers to the new SIMPLE TDMA solver sweep configuration of 3 sweeps for the u_1 equation, 3 sweeps for the u_2 equation, and 12 sweeps for the P' equation. Loops refer to the number of inner loop iterations performed for the test.

sweeps per inner loop would reduce the number of inner loops needed to solve the u_1, u_2 and P' equations simultaneously. Table 4.3 indicates that the inner loops were not reduced by increasing the number of sweeps per loop. In fact, they were slightly increased. As a result, all of the modified sweep cycle SIMPLE runs were less efficient than the standard sweep cycle used in the rest of this study.

The degree of similarity in the efficiencies of various runs is evident in Table 4.4. For runs that shared the same value of Δt and solver, sweeps per time step, CPU time per sweep and CPU time per time step are always within 5% of each other and are often much more similar. For the SIMPLE runs, even reducing the time step size by 10 times had little effect on the value of CPU time per sweep. As noted above, the sweeps per time step did change with time step size, although not nearly as much for SIMPLE as for FS. A 10% increase in time per time step was observed with a reduction in time step size for the modified SIMPLE run.

All of the results in Tables 4.3 and 4.4 were generated by simulations which start from initial conditions and experience a non-periodic development period of time before quasi-steady flow is achieved. This occurs in the simulations from $t = 0$ up to around $t = 40$. To examine the quasi-steady state performance of the tested solvers, 6 more runs were completed up to $t = 50$ on the same machine as the previous tests. The CPU time and sweeps performed by these solutions was

subtracted from the corresponding result in Table 4.3 to provide a more accurate look at the quasi-steady performance of the studied solvers. These results are shown in Table 4.5; the normalized results are listed in Table 4.6.

Solver	Δt	n	CPU time	Sweeps	Loops	$\frac{\text{Sweeps}}{n}$	$\frac{\text{CPU time}}{\text{Sweeps}}$
FS	0.00125	120000	2458652	698169	120000	5.82	3.52
FS	0.0125	12000	805954	1925775	12000	160.48	0.42
FS	0.0125	16000	1024217	2572920	16000	160.81	0.40
SIMPLE	0.00125	160000	46902803	14415560	2883112	90.10	3.25
SIMPLE	0.0125	12000	3661181	1062875	212575	88.57	3.44
SIMPLE	0.0125	16000	4733303	1416180	283236	88.51	3.34
SIMPLE2	0.00125	160000	60897304	58402548	3244586	365.02	1.04
SIMPLE2	0.0125	12000	4394545	4117050	228725	343.09	1.07
SIMPLE2	0.0125	16000	5867580	5485482	304749	342.84	1.07

Table 4.5: Quasi-steady time study summary. Original results were corrected by subtracting the initial transient portion of the solution from the results to provide results only from the quasi-steady portion of the solution. CPU time is reported in ms and refers to actual wall clock CPU time. SIMPLE2 refers to the new SIMPLE TDMA solver sweep configuration of 3 sweeps for the u_1 equation, 3 sweeps for the u_2 equation, and 12 sweeps for the P' equation. Loops refer to the number of inner loop iterations performed for the test.

The changes in the summary quantities between Tables 4.3 and 4.5 are slight. The FS values of sweeps per time step, CPU time per sweep and CPU time per time step vary 5% or lower after the development period contributions were removed. For the SIMPLE cases, the CPU time per sweep changed less than 1%. The SIMPLE cases at $\Delta t = 0.00125$ experienced changes of approximately 10% for solver sweeps per time step and CPU time per time step. The effect of the development period removal was smaller for the $\Delta t = 0.0125$ SIMPLE cases, with 3.5% or lower for the standard sweep configured solver and 6% or less for the modified sweep configured solver in both solver sweeps per time step and CPU time per time step. The change between solutions with the same solver and time step size is of the same order of magnitude as the variation between the values in Table 4.3 ($\sim 5\%$). This suggests that the error involved in this study is of the same order as the correction, and suggests that the results are likely correct within 10%, with or without the initial development period removed. Using the normalized values of CPU time per sweep (averaged for the $\Delta t = 0.0125$ FS and SIMPLE cases), Figure 4.6 can be adjusted

Solver	Δt	n	CPU time	Sweeps	Loops	$\frac{\text{Sweeps}}{n}$	$\frac{\text{CPU time}}{\text{Sweeps}}$	$\frac{\text{CPU time}}{n}$
FS	0.1	10	3.05	0.36	10	0.04	8.41	0.31
FS	1	1	1	1	1	1.00	1.00	1.00
FS	1	1.33	1.27	1.34	1.33	1.00	0.95	0.95
SIMPLE	0.1	10	48.05	6.19	198.71	0.62	7.76	4.81
SIMPLE	1	1	4.54	0.55	17.71	0.55	8.23	4.54
SIMPLE	1	1.33	5.87	0.74	23.60	0.55	7.99	4.40
SIMPLE2	0.1	10	62.72	25.31	225.63	2.53	2.48	6.27
SIMPLE2	1	1	5.45	2.14	19.06	2.14	2.55	5.45
SIMPLE2	1	1.33	7.28	2.85	25.40	2.14	2.56	5.46

Table 4.6: Normalized quasi-steady time study summary. Original results were corrected by subtracting the initial transient portion of the solution from the results to provide results only from the quasi-steady portion of the solution. Values are normalized by dividing by the second row values. SIMPLE2 refers to the new SIMPLE TDMA solver sweep configuration of 3 sweeps for the u_1 equation, 3 sweeps for the u_2 equation, and 12 sweeps for the P' equation. Loops refer to the number of inner loop iterations performed for the test.

to indicate the actual relative CPU time of the run as a function of simulation time by multiplying the values shown in Figure 4.6 by the corresponding values of normalized CPU time per time step in column 8 of Table 4.6. This is shown in Figure 4.12.

As seen in Figure 4.12 the cost is least for the FS solver with $\Delta t = 0.0125$ and is most for the SIMPLE solver with $\Delta t = 0.00125$. FS is more efficient for all configurations of SIMPLE studied up to this point. It should be noted that ten times more data is available from the FS run with $\Delta t = 0.00125$ than is available from the SIMPLE run with the time step 10 times larger and it still solves in less time.

As a final point of interest, an attempt was made to determine the most efficient time step size for both solvers. SIMPLE is known to be most efficient at its maximum time step which produces the least information (a single steady state solution) but still provides some insights into the flow physics. Up to this point it is difficult to say where the optimum time step lies for the FS solver. In order to find the optimum time step for this case, runs at different time steps were completed using the FS code on a dedicated machine. Each run simulated the same total

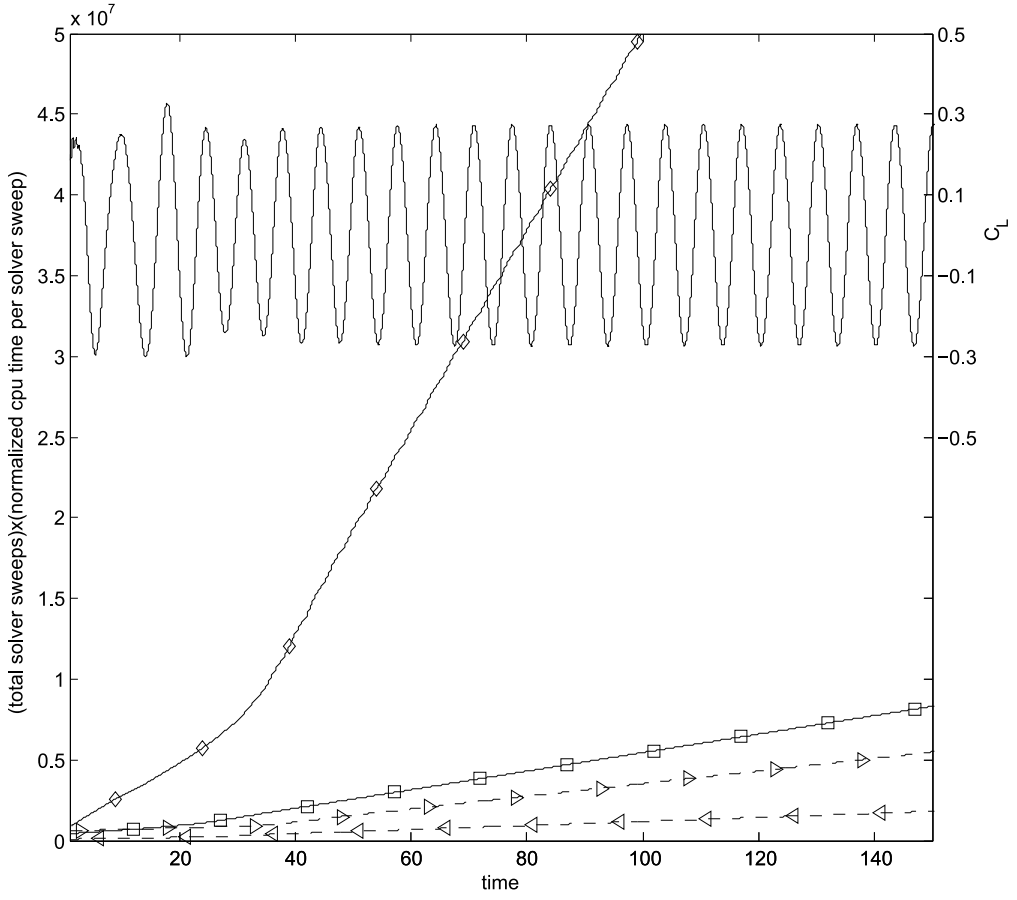


Figure 4.12: Relative cumulative wall clock time versus simulation time for the laminar flow over a square cylinder case. The SIMPLE run is shown with a solid line and the FS run is shown with a dotted line. Markings are SIMPLE, $\Delta t = 0.0125$, $\square$; SIMPLE, $\Delta t = 0.00125$, $\diamond$; FS, $\Delta t = 0.0125$, $\triangleleft$; and FS, $\Delta t = 0.00125$, $\triangleright$. The C_L value is shown with a solid line in the upper portion of the figure and was calculated using the SIMPLE code at $\Delta t = 0.0125$.

length of time (from $t = 0$ to $t = 200$) but had a different time step size and number of time steps. Solutions diverged for $\Delta t \geq 0.03$ so the value of Δt was varied from 0.00125 to 0.029. The results of this study are shown in Figure 4.13.

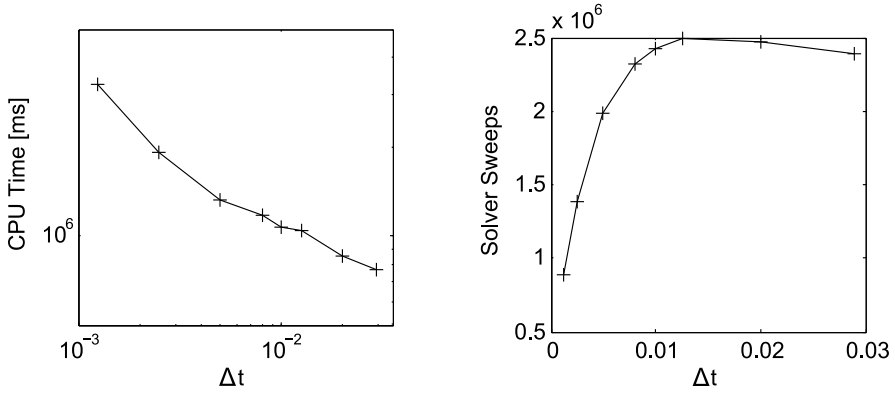
Figure 4.13a indicates that the least expensive time step size is the largest possible size before the code becomes divergent. The decrease in time appears to be nearly exponential (decay). Figure 4.13b indicates that as the time step size increases, the number of sweeps per time step increases until it reaches a peak and then decreases slightly beyond that point. The cost of the solution when divided by the number of sweeps seems to increase with the log of the time step size as shown in Figure 4.13c, while Figure 4.13d indicates total sweeps divided by the number of time steps increases as a nearly linear function of Δt. The CPU time divided by number of time steps also seems to increase as a linear function of Δt as shown by Figure 4.13d.

4.6 Conclusions

In this chapter it was shown that both the FS and SIMPLE codes provide reasonable solutions to a square cylinder in laminar cross flow problem. Both codes predict a flow field that exhibits quasi-steady vortex shedding after the initial transients have passed. The value of $\overline{C_L}$ for both simulations was approximately zero as would be expected for a symmetrical body at zero angle of attack. The summary properties of both solutions are of the same order as the experimental and previous numerical results presented in the literature, and for the most part fall within the range of previously reported results.

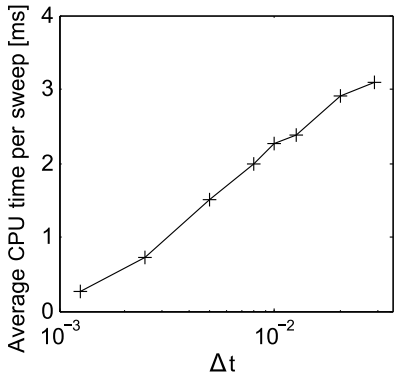
The FS code was found to predict a flow field which evolved slightly faster than the solution provided by the SIMPLE code, as indicated by a larger St. FS also predicted drag which was 2.5% lower than the prediction of the SIMPLE code.

Computational efficiency as measured by solver sweeps was found to be better for the SIMPLE code than for the FS code at the original solution time step size, with the FS code consistently needing more solver sweeps per time step leading to a greater number of sweeps overall. When the time step was reduced, the SIMPLE code showed a marked decrease in solver efficiency, requiring the same or more sweeps per time step than SIMPLE code at the larger original time step. On the other hand, for the FS code, a 10 times reduction in time step size reduced the number of sweeps needed per time step from a maximum value of 250 to a maximum value of 9. Both SIMPLE and FS codes were found to exhibit the worst

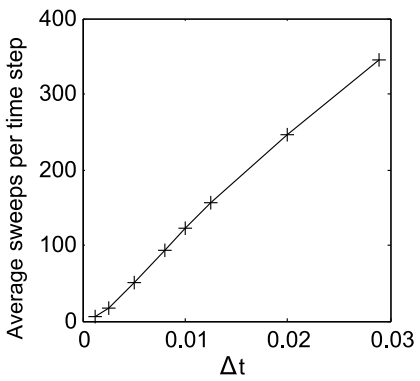


(a)

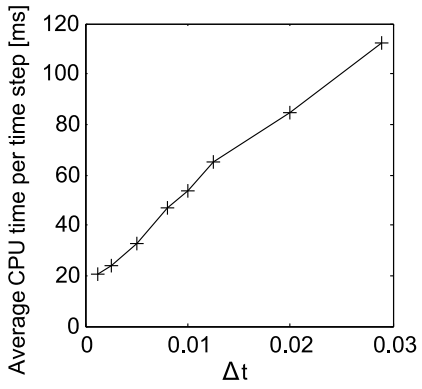
(b)



(c)



(d)



(e)

Figure 4.13: FS solution behavior versus time step. Plotted quantities are (a) total wall clock CPU time, (b) total solver sweeps, (c) average CPU time per solver sweep, (d) average number of solver sweeps per time step and (e) average CPU time per time step. Results were generated on a dedicated machine with FS up to $t = 200$.

efficiency when the flow was exhibiting large changes in a large number of cells and better efficiency when less change was occurring from one time step to the next. For the larger time step, FS seemed to have more trouble with fluctuating flow than SIMPLE as indicated by a greater increase in the number of solver sweeps for the FS code when the time step increased in size.

An analysis of the division of CPU time amongst the solution procedure components in the SIMPLE and FS code indicated that the matrix equation solution portion of both codes required a much smaller percentage of CPU time than was expected, indicating that solver sweeps are not a conclusive measure of computational cost. The FS code operating at $\Delta t = 0.0125$ required a large number of solver sweeps and only spent 68% of its total CPU time solving matrix equations while. The SIMPLE code operating at $\Delta t = 0.0125$ only spent 8% of its CPU time solving matrix equations. Increasing the number of sweeps per internal loop in the SIMPLE code did increase the solver cost to 24% of the total cost but did not decrease the overall number of sweeps needed as the number of internal loops required for each time step did not decrease.

Running the various configurations on a dedicated machine allowed their relative time costs to be evaluated. FS was found to be the most efficient at the largest time step while the SIMPLE code running at the smallest time step was found to be the least efficient. Increasing the number of time steps by 10 times and decreasing the step size by the same amount increased the cost of the SIMPLE solution by approximately 10 times but only increased the cost of the FS solution 3.1 times. When cost per time step is considered, SIMPLE seems to be the best at a larger time step while FS is better at a smaller time step.

The differences in efficiency with time step size suggests that both time-stepping schemes could be useful, but for different applications. For very large time step applications, such as large amplitude vortex induced vibration problems, SIMPLE is likely to be more cost efficient than FS. Alternatively, for very small time step simulations such as aero-acoustic simulations, FS is likely to be vastly superior in cost efficiency. FS does increase in cost, however, as time step decreases, suggesting that the time step should not be decreased in size beyond the time resolution required for an accurate solution or as is needed for transient results.

When CFD is used as a tool for design, computational efficiency becomes very important because an efficient code allows more design iterations than a less efficient code. Based on the case studied here, FS solutions of transient laminar flow systems are much more cost effective than SIMPLE within the range of time steps FS is

able to operate. Above some critical time step size, SIMPLE should be used as it is stable for all large time steps and increases in cost efficiency as time step increases.

Chapter 5

Turbulent Flow Over a Square Cylinder

5.1 Problem Definition

The third test case considered in this study is turbulent flow over a square cylinder. This case uses the same domain geometry as the laminar flow over a square cylinder case described by Figure 4.1. As in the laminar case, flow enters from the left and exits to the right. For this geometry, Re = 21400 is known to be within the turbulent flow regime. This case has been studied extensively both experimentally and numerically in the literature making it a good candidate for comparison with existing results. The computational mesh used is the same 70×100 cell mesh described in Section 4.2. It can be seen in Figure 4.2. The summary properties for this case are the same as described in Section 4.3.

5.2 Test Matrix

For this problem, two simulations were completed, one with FS and the other using SIMPLE. The simulations were run from quiescent conditions at $t = 0, n = 0$ until $t = 625, n = 50000$ at which point the solutions could be assumed to have reached a quasi-steady state for a majority of this solution. The parameters for the two original runs of interest are listed in Table 5.1.

In this case, it was necessary to use the skewed initial inlet condition described by Equation 4.7 for only the FS code. The SIMPLE code exhibited oscillations

Δt	Re	Time-stepping scheme
0.0125	21400	SIMPLE
0.0125	21400	FS

Table 5.1: Parameters and time-stepping schemes used for initial turbulent flow over a square cylinder tests.

without the use of a skewed inlet condition. The FS code however did not achieve a quasi-steady oscillating state without the skewed inlet for the first 40 times steps. Instead, it formed a non-oscillating symmetrical flow field. This solution was quite unstable numerically and would eventually produce overflow errors if allowed to run up to around $n = 10000$. Introduction of a non-symmetrical inlet condition for a short portion of this beginning of the run provided a quasi-steady oscillating solution which was also numerically stable.

5.3 Results

As was the case for the laminar case, the SIMPLE and FS codes produced qualitatively very similar results. The streamlines and turbulent kinetic energy contours for a single instant in time are shown in Figure 5.1 as calculated using the FS code.

At the point shown in Figure 5.1 a large vortex has been shed from the upper half of the cylinder and a new vortex is forming on the bottom side of the cylinder. Half a period later the flow will be the same as shown in Figure 5.1 only mirrored about the central axis. This process of vortex shedding continues for the duration of the simulation.

As was discussed in Section 4, a vortex passing by the top of the cylinder corresponds to a positive C_L while a vortex passing by the bottom of the cylinder corresponds to a negative C_L. The C_L signal for this case is shown for both time-stepping schemes in Figure 5.2.

As seen in Figure 5.2, the lift coefficients are similar in magnitude and frequency but not identical. The FS frequency appears to be slightly lower and its magnitude is greater than that of the SIMPLE C_L signal. These differences will be discussed more quantitatively below in terms of the values of St and C_{Lrms}. The mean value of the C_L signal does appear to be zero for both cases as expected.

The $C_{D,P}$ oscillates as well, but its peaks correspond to a vortex being present

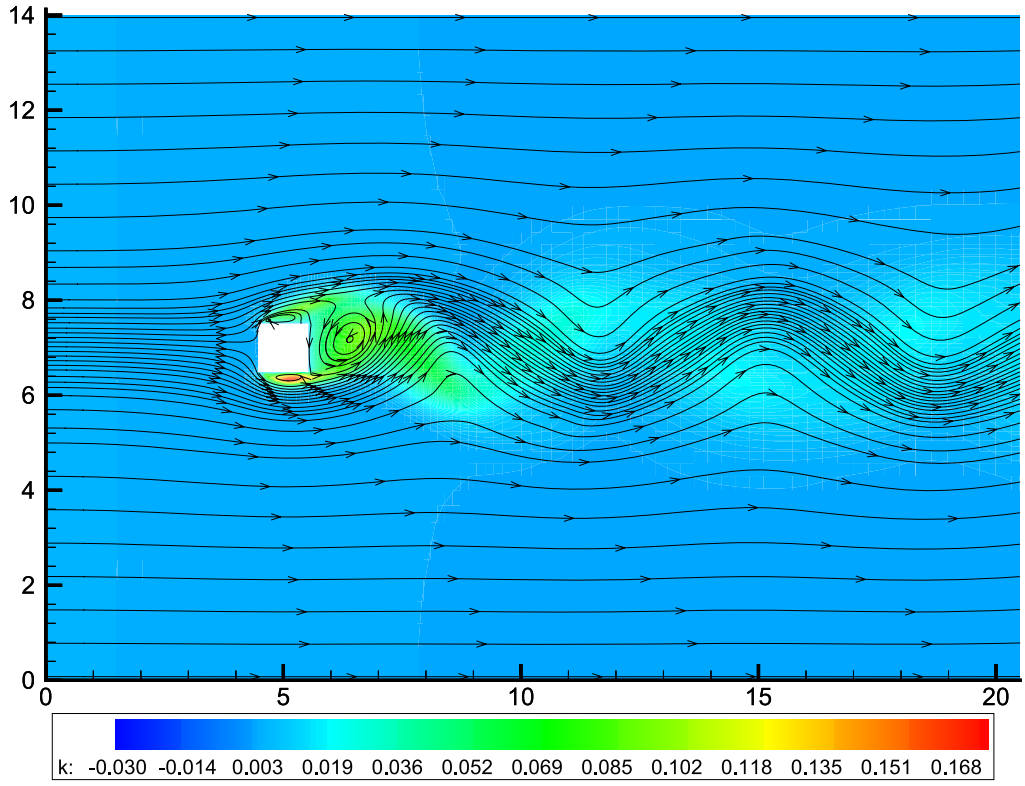


Figure 5.1: Instantaneous turbulent kinetic energy contours and streamlines for a square cylinder in cross flow with Re = 21400 calculated using FS time stepping. Contours and streamlines are for at a point in time after quasi-steady flow has been achieved.

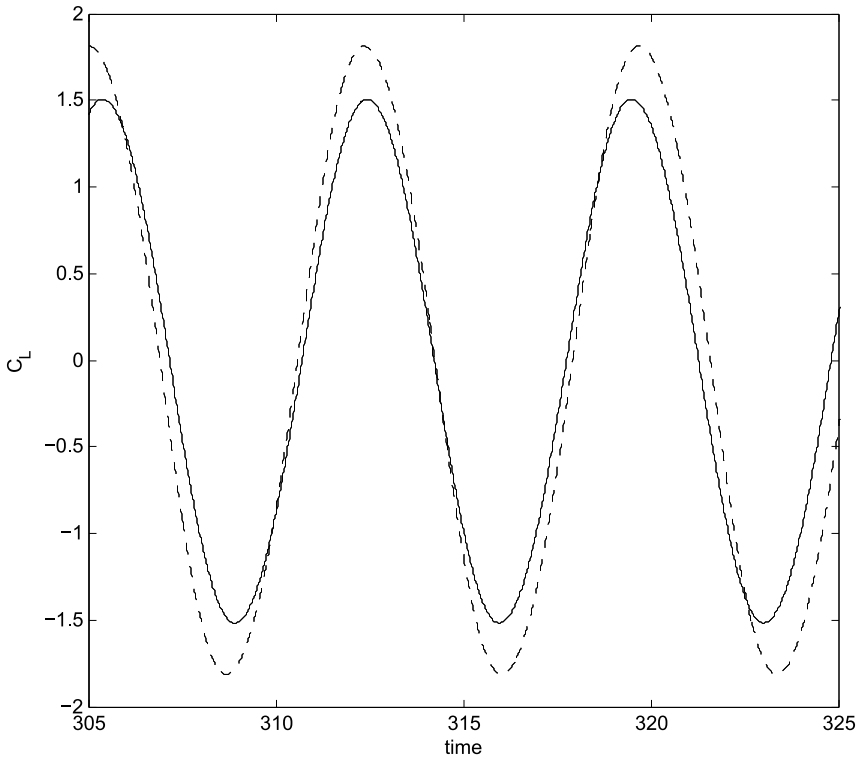


Figure 5.2: Lift coefficient vs. time for the turbulent square cylinder test case. The SIMPLE solution is shown by the solid line while the FS solution is shown by a dashed line.

downstream of the cylinder, producing a frequency that is double that of the C_L. The $C_{D,P}$ signal is shown for a portion of the quasi-steady solutions in Figure 5.3 for both time-stepping schemes.

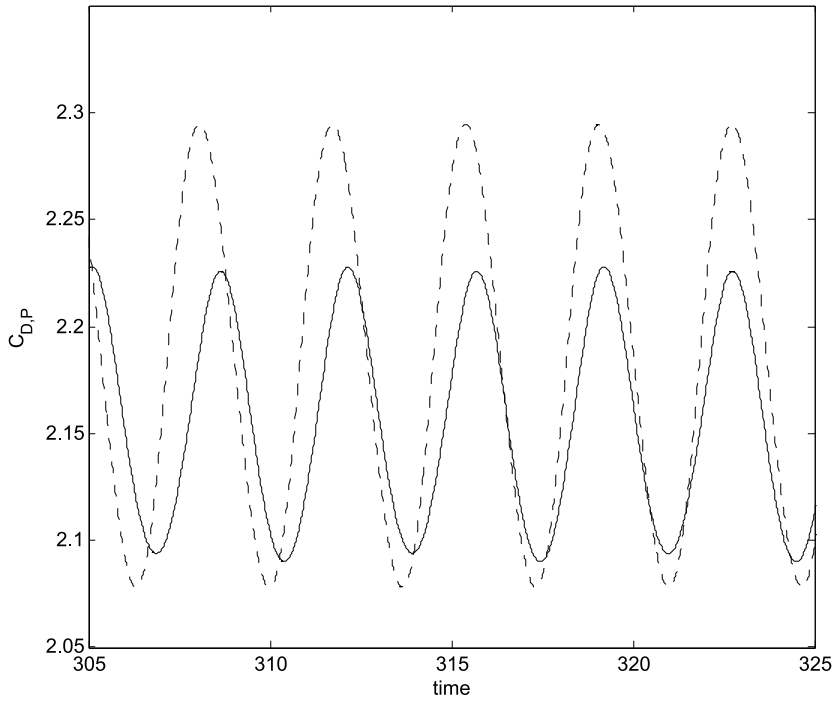


Figure 5.3: Drag coefficient vs. time for the turbulent square cylinder test case. The SIMPLE solution is shown by a solid line while the FS solution is shown by a dashed line.

As shown in Figure 5.3, the mean values and amplitudes of the drag coefficients are somewhat different but are in the same range. The frequencies are also slightly different as would be expected given the different frequencies shown in Figure 5.2. The summary properties for the two simulations are listed in Table 5.2.

Time-stepping scheme	St	$\overline{C_L}$	$C_{L_{rms}}$	$\overline{C_{D,P}}$	$C_{D,P_{rms}}$
Simple	0.144	-5.10×10^{-3}	1.070	2.15	4.79×10^{-2}
FS	0.138	6.01×10^{-6}	1.288	2.19	7.67×10^{-2}

Table 5.2: Summary properties for turbulent flow over a square cylinder with Re = 21400.

Table 5.2 demonstrates that the two solutions are fairly similar while still being distinct. Both $\overline{C_L}$ are essentially zero as expected. The Strouhal numbers vary

by about 4% and in this case FS is lower suggesting that flow physics for this case evolve less quickly using FS compared to SIMPLE. This goes against the hypothesis raised in Chapters 3 and 4 that FS simulated flows evolve more quickly than SIMPLE simulated flows. The values of $\overline{C_{D,P}}$ are quite close and vary by only 1.6% while the rms values, which are a measure of the amplitude of the signal, vary by 20% for the C_L signals and 60% for the $C_{D,P}$ signals. This may seem significant, however, the actual magnitude of the difference is 0.281 for C_{Lrms} values (less than 15% of the amplitude), and 0.0287 for $C_{D,P_{rms}}$ (approximately 1.5% of the value of $C_{D,P}$). All of the values reported fall within the range reported in literature for numerical studies and are within 10% of the experimental values reported by Lyn et al. [26].

The St results for this study are well within the range of LES results presented in Tables 1.3 and 1.4. The FS St value of 0.138 is closer to the maximum experimental value of 0.136 reported by Lyn et al. [26] than is the SIMPLE value. The SIMPLE results seem to agree best with the URANS results for the KL model using wall functions reported by Bosch and Rodi [3] and Kato and Launder [16] (Table 1.6) as expected. The FS St value agrees best with the URANS RSE model using wall functions as presented by Franke and Rodi [12].

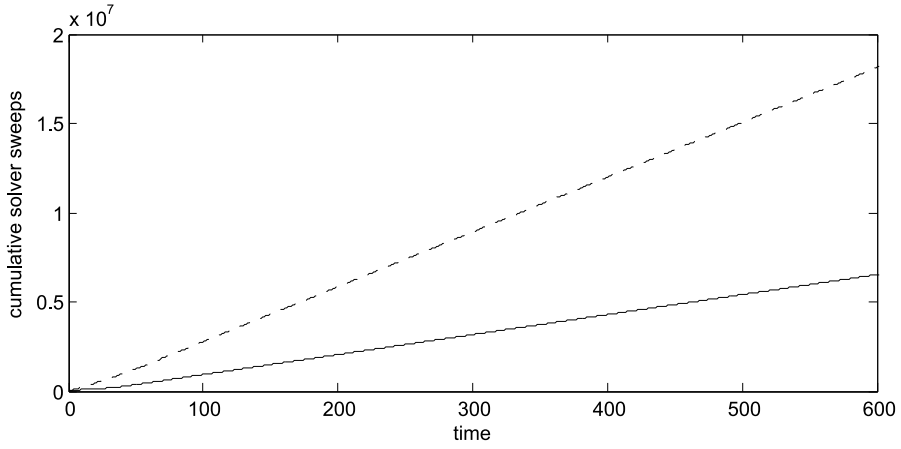
Values of C_{Lrms} in Table 5.2 agree well with the LES results in Table 1.3 and with the results of Thompson [38]. When compared with other URANS studies, the SIMPLE value agrees best with the KL model with wall functions results of Bosch and Rodi [3] and FS seems to agree better with the RSE two level results of Franke and Rodi [12].

The $\overline{C_{D,P}}$ values are very close to the experimental value [26] and the LES results in Table 1.3. The SIMPLE value is identical to the value reported by Thompson [38]. Of the other URANS studies, the $\overline{C_{D,P}}$ SIMPLE and FS values agree best with the KL model with wall functions results of Bosch and Rodi [3].

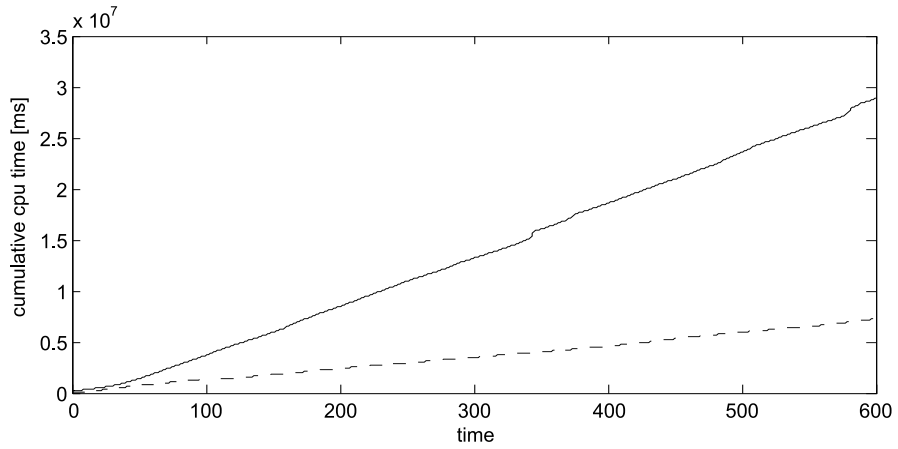
Compared to the LES results, the values of $C_{D,P_{rms}}$ in Table 5.2 are very low. An examination of the URANS results in the literature, however, indicates that this is a common difference between URANS and LES studies of turbulent flow over square cylinders. The FS $C_{D,P_{rms}}$ value is slightly above the range reported in the surveyed literature for URANS simulations (Tables 1.5 and 1.6), but it is closer to the LES results which are likely to be more accurate. The SIMPLE $C_{D,P_{rms}}$ value is nearly identical to the value reported by Thompson [38] for this case.

Both runs for this case were completed on the same dedicated machine allowing the computation time to be monitored and compared in addition to the number of

solver sweeps carried out by the TDMA solver. The cumulative solver sweeps and time to completion is shown for both time-stepping schemes in Figure 5.4.



(a)



(b)

Figure 5.4: Comparison of (a) cumulative solver sweeps and (b) cumulative CPU time for the turbulent flow over a square cylinder case. The SIMPLE run is shown with a solid line and the FS run is shown with a dashed line.

As was demonstrated in the laminar case, the FS code requires more sweeps per time step but takes less time per time step providing a faster solution than the SIMPLE scheme. Both solutions exhibit an initial low number of solver sweeps before the quasi-steady flow is achieved. The CPU time curve is not as smooth as the solver sweeps curve. There are several regions of the CPU time curves which suddenly increase, a behavior which is not reflected in the solver sweeps curves. The most obvious example of these discontinuities is the SIMPLE CPU time curve in the region of $t = 300$. These sudden increases in time are likely caused by low

cost background processes running on the host machine which slightly slow down the solution. Even with these small discontinuities, the trend is clear that the FS code solves in a much shorter time than the SIMPLE code, even with the addition of 2 extra equations which are solved implicitly. An examination of the cost of the two additional equations in the turbulent FS code indicates that the k and ε equations solve much more easily than the P equation. Typically the k and ε equations require 2 sweeps per time step each while the P equation may require between 100 and 600 sweeps per time step. In these conditions, implicitly solving the k and ε equations is nearly as cost efficient as solving them explicitly without the added danger of numerical instability. SIMPLE on the other hand requires all 5 equations to be solved simultaneously. In the current solver configuration, the P' equation receives 3 sweeps per internal loop and all others receive 1, resulting in a total of 7 solver sweeps per internal iteration. In the quasi-steady region of the SIMPLE solution, the solver requires between 12 and 25 internal loops which explains the much larger overall cost of the SIMPLE solution.

In order to compare the turbulent and laminar cases, the solver sweeps per time step are plotted in Figure 5.5 along with the C_L signals for a portion of the quasi-steady solution.

As seen for the laminar case, Figure 5.5 demonstrates that the maximum cost to the solver occurs at regions of high slope in the C_L curves. In addition, the FS code seems to react more strongly than SIMPLE to this increase with its solver sweeps increasing nearly 6 times in high slope regions. This is the same trend as was observed in Chapter 4. However, it is more pronounced when the turbulence model is included in the solution.

To compare the correlation between solver sweeps and CPU time for this case, the solver sweeps per time step was divided by the CPU time per time step to produce the cost per solver sweep as a function of time step. These results are plotted for a portion of the quasi-steady solution in Figures 5.6 and 5.7.

The trends shown in Figures 5.6 and 5.7 are as expected for the most part. The FS results indicate that the CPU time is not directly correlated to the number of solver sweeps for the FS solver. As observed for the laminar flow over a square cylinder problem, the cost of assembling the equations is fairly substantial and is the same for each time step in the FS code. The value that does change with the position of the solution is the number of sweeps needed to solve the P equation. At peaks and troughs in the C_L curve, the number of sweeps is at its lowest for both codes, which corresponds to a higher relative cost of each sweep as the assembly

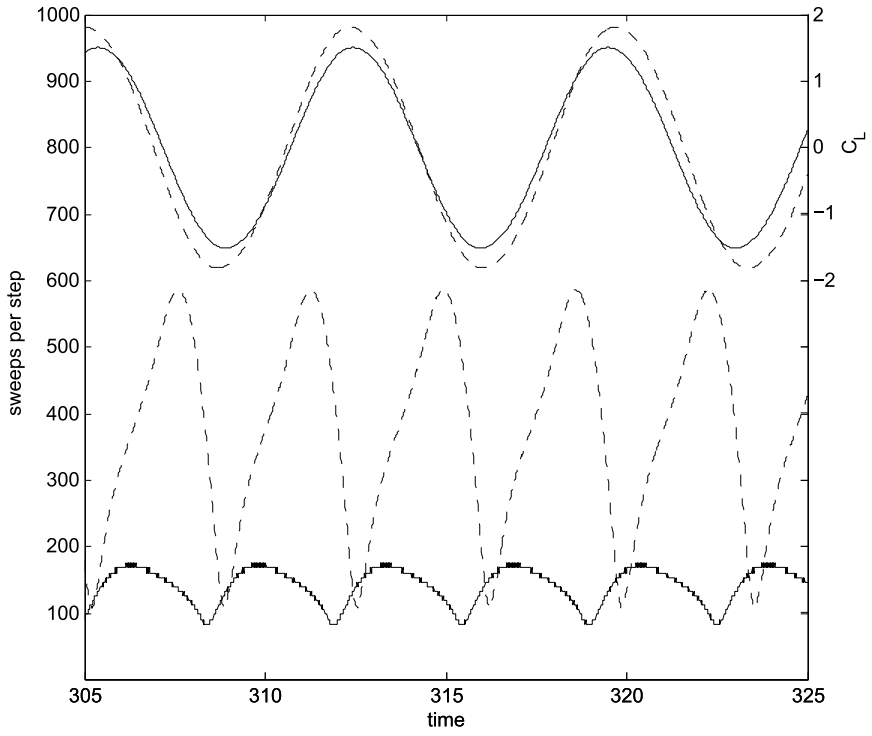


Figure 5.5: Solver sweeps per time step for turbulent flow over a square cylinder with Re = 21400. The SIMPLE run is shown with a solid line and the FS run is shown with a dashed line.

cost is spread amongst fewer sweeps. In high slope regions of the C_L signal, the number of sweeps is the highest, driving down the cost per sweep. In the case of the SIMPLE results shown in Figure 5.7, the time cost per sweep remains approximately constant compared to the C_L value because the number of times the equations are assembled is proportional to the number of sweeps. As a result, the time cost per sweep is approximately constant throughout the solution.

There are two anomalies which appear in Figures 5.6 and 5.7 which need to be addressed. First of all, both figures show sudden spikes in the CPU time per sweep. Figure 5.7 suggests that this phenomena may be connected to the solution physics as it seems to occur at the same phase on every other cycle. A careful look at the full range of the data (not shown here) indicates that these spikes occur at different points along the C_L cycle and apparent solution dependance is purely due to chance and the portion of solution time chosen for the plot. Further investigation into the cause of the spikes indicated that the wall clock time between the spikes was almost exactly 300 seconds or 5 minutes apart, suggesting that a background process on the computer used for the solution is the cause. Each of these spikes corresponds

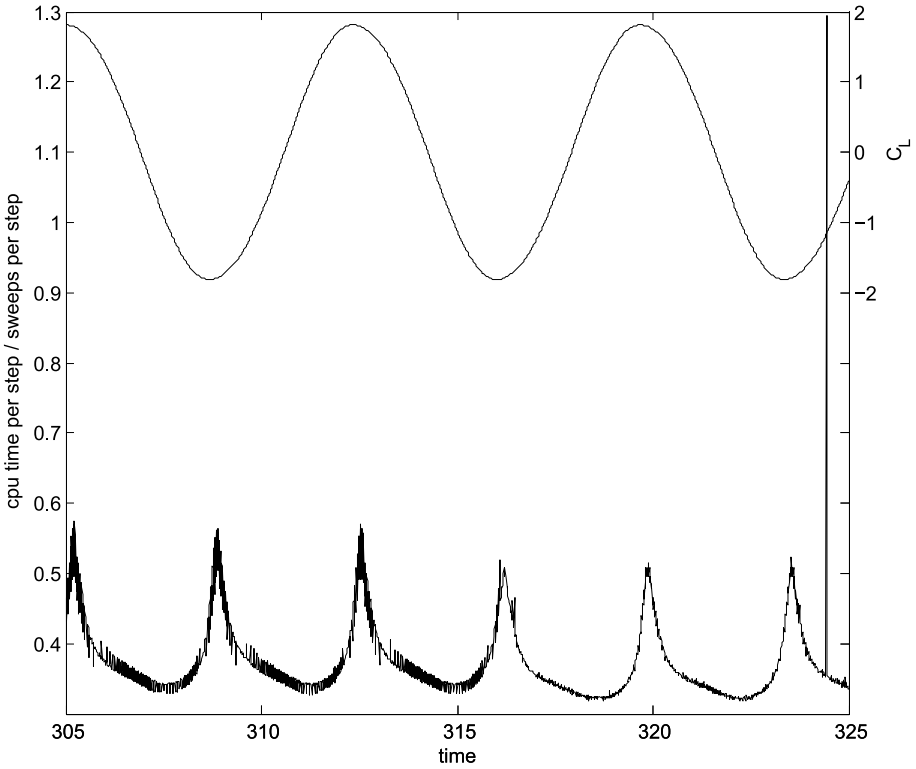


Figure 5.6: FS CPU time per sweep as a function of time step. The C_L signal is included for reference.

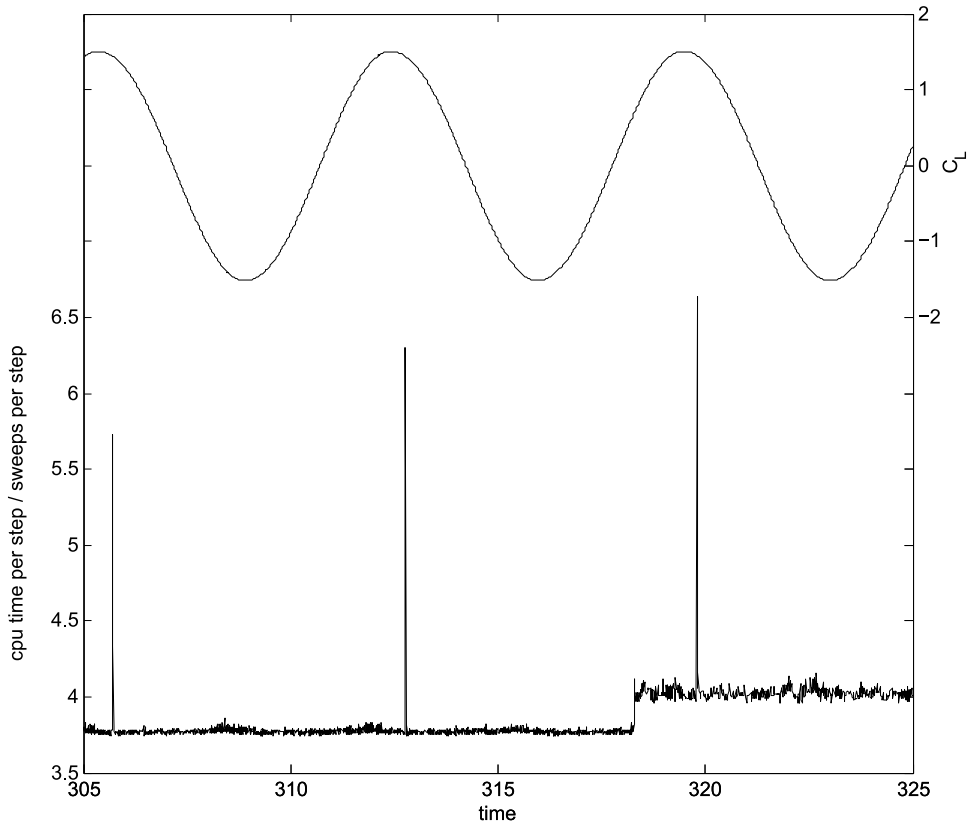


Figure 5.7: SIMPLE CPU time per sweep as a function of time step. The C_L signal is included for reference.

to an increase of approximately 0.5 seconds for every 300 seconds of CPU time, or approximately 0.16% of the total time recorded in this study, suggesting the error introduced by this anomaly is negligible.

The second anomaly is the change in average value of CPU time per sweep observed most clearly around $t = 318$ in Figure 5.7 but also visible in Figure 5.6 at $t = 316$ to a lesser degree. As with the previous issue, a look at the CPU time per solver sweep results for the entire solution indicates that several of these changes occur throughout the solution. These changes are likely due to background processes running on the host machine which require very few resources but still have some effect on the overall wall clock time of the solution. These changes in mean value vary up to 30% of the global mean and will introduce some error into the efficiency results reported in this study. However, the mean value of the CPU cost per sweep is on the order of 10 times greater for SIMPLE than it is for FS which is well outside the order of error introduced by background processes on the host machine. This trend is seen in the summary quantities which span the entire length of the runs and were completed up to $n = 50000$, or $t = 625$. These results are tabulated in Table 5.3.

Solver	CPU time	Sweeps	$\frac{\text{Sweeps}}{n}$	$\frac{\text{CPU time}}{\text{Sweeps}}$	$\frac{\text{CPU time}}{n}$
FS	7567353	18880131	377.60	0.40	151.35
SIMPLE	30078459	6808536	136.17	4.42	601.57

Table 5.3: CPU time and solver sweep results for the turbulent flow over a square cylinder test case. CPU time is in ms. Sweeps refer to TDMA solver sweeps.

To better appreciate the differences between the two codes, Table 5.3 is normalized by the FS run results shown in Table 5.4. The normalized SIMPLE laminar flow results for the same time step size are copied from Table 4.3 for the sake of comparison.

As shown in Table 5.4, the percent difference in CPU time for the turbulent flow case is slightly less than the percent difference for the laminar case at this time step. This is a result of the increase in the number of sweeps needed to solve the P equation for the FS code and to a minor degree, the increase in the number of equations that must be solved by the FS code from 1 to 3 compared to 3 to 5 for the SIMPLE code. The percent increase in sweeps per time step is greater for the turbulent code by 19%. The normalized values of time per sweep increase from laminar to turbulent runs largely due to the large increase in sweeps needed by the

FS code in the turbulent simulation. A solver sweep itself is not very expensive for either code, and the FS code only needs to build the equations once which drives down the cost per sweep when the number of sweeps is high. This increase in the required solver sweeps is best illustrated by the average sweeps per time step. The FS code increased 2.4 times from 156 to 378 average sweeps per time step over the change from laminar to turbulent solutions while SIMPLE increased 1.6 times from 86 to 136. Although individual sweeps are not particularly costly, the large increase in solver sweeps per time step required by the FS code from laminar to turbulent flow does somewhat close the gap in efficiency between FS and SIMPLE algorithms used for turbulent flows.

Solution	CPU time	Sweeps	$\frac{\text{Sweeps}}{n}$	$\frac{\text{CPU time}}{\text{Sweeps}}$	$\frac{\text{CPU time}}{n}$
SIMPLE laminar	4.51	0.55	0.55	8.15	4.51
SIMPLE turbulent	3.97	0.36	0.36	11.02	3.97

Table 5.4: Comparison of normalized SIMPLE solver results. Results are normalized by dividing by the results of the FS run of the same Re, Δt and n. Sweeps refer to TDMA solver sweeps.

5.4 Conclusions

Both FS and SIMPLE codes have been shown to produce results which fall within the results previously presented in the literature. In order to ensure that the FS code was numerically stable, the turbulence equations are solved implicitly. This addition of two extra equations has a minimal effect on the efficiency of the FS code because the solver is able to solve the equations in two sweeps for each turbulence equation in most cases. The pressure equation in the FS code takes much more effort to solve at this higher Reynolds number. It is this increase in solver effort that reduces the increase in efficiency from SIMPLE to FS codes that was seen in Chapter 4.

The results indicate that the improvement in efficiency seen in FS time stepping is more pronounced when the flow is laminar, but there is still a definite advantage in efficiency when using FS to solve quasi-steady turbulent flow problems. This confirms the conclusions of Chapter 4 which suggest that FS is a better choice in terms of efficiency when a quasi-steady small time step simulation is required; indeed, it extends the conclusions to turbulent flows.

Chapter 6

Conclusions

6.1 Summary

The framework for a numerical solution of the transient, incompressible Navier-Stokes equations was laid out with the option of two different time-stepping schemes, referred to here as the SIMPLE and the Fractional Step time-stepping schemes. The SIMPLE time-stepping scheme implemented here iteratively solves two velocity equations, a pressure correction equation and any necessary turbulence equations. The FS scheme is a semi-implicit method which solves a pressure equation and any turbulence equations iteratively, but solves the velocity equations explicitly. Both time-stepping schemes are first order accurate in time.

Turbulence is approached using URANS turbulence modeling. The Kato-Lauder turbulence model is used in this study instead of the standard $k - \varepsilon$ model as the former has been shown to be more accurate for flow over bluff bodies [16, 3].

Three test cases were explored using the code described above:

1. Laminar Flow in a Lid-Driven Skewed Cavity
2. Laminar Flow over a Square Cylinder
3. Turbulent Flow over a Square Cylinder

The first test case demonstrated that both codes are able to predict the same steady state flow in a skewed cavity on the same mesh. Both solvers agreed with the benchmark solution, suggesting that they are reasonably spatially accurate and could be used in cases where steady state solvers have difficulty generating a

converged solution. The rate of convergence to a steady state solution was found to be somewhat dependent on the time-stepping scheme used in addition to grid size and time step size. SIMPLE was found to vary more with grid size than FS, and it had a longer predicted time to steady state. The number of TDMA solver sweeps required to reach a steady state solution was used as a measure of the overall efficiency of the SIMPLE and FS numerical solutions. It was determined that the most efficient way to solve the steady state problem was with the SIMPLE code using a very large time step (essentially running in steady state mode.) This is not possible with the FS scheme as it becomes unsteady if the time step gets too big. For example, for the FS test with $Re = 100$, on a 40×40 grid, the solution was stable at $\Delta t = 0.001$ but not at $\Delta t = 0.0011$.

The second test case confirmed that flow features evolve at somewhat different rates depending on the time-stepping scheme. This was indicated by the difference in Strouhal number (2.5% difference). In addition, the amplitudes and mean values of the lift and drag coefficients were different depending on the time-stepping scheme. All summary quantities explored were similar and were close to the range reported in the literature indicating that both the SIMPLE and FS codes were functioning correctly. The computational cost as measured by the number of solver sweeps was found to increase for those periods of time in the solutions characterized by quickly changing lift coefficient. Regions of change in the lift coefficient correspond to periods of time featuring large vortices just downstream of the cylinder. This suggests that increased solver effort corresponds to portions of the solution time during which large changes occur in a large number of cells.

An investigation of the division of the computational cost in the two laminar flow codes demonstrated that, in contradiction to the assumption made in the first part of this study, the solver was not the primary contributor to the computational cost of the code. To better understand the computational demands of the test codes, a series of simulations were completed on a dedicated, single CPU machine. The results of the CPU time study indicated that the FS code was more efficient than the SIMPLE code in terms of total computational cost for an equivalent solution. This is largely because the FS code only needs to assemble the governing equations once per time step while the SIMPLE code requires a new equation assembly per internal iteration. The results of the CPU time study indicate that the FS code is much more efficient for small time step simulations, while the SIMPLE code is preferable for very large time steps. Both codes are least costly overall when the time step is largest. As the time step size drops, the cost per time step remains fairly constant for SIMPLE while it reduces in magnitude for FS. A reduction of 10

times in the time step size resulted in a 3.2 times reduction of CPU time per time step, largely due to an increase in matrix solver efficiency (solver sweeps per time step at the smaller time step were 25 times lower).

The third test case demonstrated results which were consistent with the literature, partially due to the large spread in results reported in the literature, both from experimental and numerical studies. The summary properties of the solutions investigated here all fell within the range of results reported in the literature. The resultant values of $C_{D,P_{rms}}$ were much lower than the LES/DNS results in the literature, but were larger than most other URANS results. As demonstrated in the laminar square cylinder test case, the number of solver sweeps was not a good judge of computational efficiency for this case. Although the SIMPLE code required 1.8 times more solver sweeps, it generated an equation of equivalent accuracy to the FS solution in 4.51 times longer. An analysis of the CPU time per time step indicated that some error was present in the CPU time study due to background processes on the dedicated machine. The effect of these errors was found to be less than 30% of the mean for either case, and do not effect the general trend, that the FS time cost was on the order of 10 times less than the SIMPLE time cost per time step. Finally, for this case, as with the previous one, FS was shown to be more efficient for small time step problems involving quasi-steady flow than the SIMPLE algorithm due to the enormous cost of reassembling the equations inside the inner loop of the SIMPLE algorithm.

6.2 Recommendations

The basic findings of this study apply to transient, quasi-steady, incompressible laminar and URANS cell centered finite volume codes. They can be summarized as follows.

1. Both of the time-stepping schemes analyzed were most efficient at their largest time steps. Time steps should be used at the largest value for which reasonable time resolution, accuracy and stability are achieved.

2. For steady state flow, the SIMPLE algorithm should be used in preference to the FS time-stepping scheme.

3. For studies which require small time steps below the maximum stable time step size for numerical stability of the FS scheme, the FS scheme should be

used in preference to the SIMPLE algorithm. In fact, the FS scheme is likely to be more efficient than the SIMPLE scheme even if, to ensure numerical stability, the FS time step is on the order of 10 times smaller than would be needed for and acceptably accurate solution using SIMPLE.

In the case of the compressible URANS codes for aero-acoustics, such as the code used by Ahn et al. [1], it is likely that FS will be more efficient than SIMPLE but further work is needed to conclusively demonstrate this.

6.3 Future Work

The recommendations of this study are based on the low computational cost of the matrix solver used here. As the dimensionality or the grid resolution increases, the TDMA solver used here may become less efficient. It is recommended that other solvers, such as the Strongly Implicit Procedure as suggested by Stone [36], should be investigated to determine if the relative efficiency of the SIMPLE and FS methods remains the same as was found here.

The effects of compressibility are not accounted for in this work and should be examined in the context of time integration methods.

As suggested by Turek [39], second or higher order time-stepping schemes are likely to yield further benefits in terms of efficiency and accuracy. It is recommended that higher order time-stepping schemes be investigated for cell centered finite volume codes.

APPENDICES

Appendix A

Source Term Transformation

ϕ	JS_ϕ
u_1	$-\left(P+\frac{2}{3}k\right)_{\xi_1} x_{2\xi_2} + \left(P+\frac{2}{3}k\right)_{\xi_2} x_{2\xi_1}$ $+\frac{\nu_{eff}}{J}\left[u_{1\xi_1}x_{2\xi_2}^2 - u_{1\xi_2}x_{2\xi_1}x_{2\xi_2}\right]_{\xi_1} - \frac{\nu_{eff}}{J}\left[u_{1\xi_1}x_{2\xi_2}x_{2\xi_1} - u_{1\xi_2}x_{2\xi_1}^2\right]_{\xi_2}$ $-\frac{\nu_{eff}}{J}\left[u_{2\xi_1}x_{2\xi_2}x_{1\xi_2} - u_{2\xi_2}x_{2\xi_1}x_{1\xi_2}\right]_{\xi_1} + \frac{\nu_{eff}}{J}\left[u_{2\xi_1}x_{2\xi_2}x_{1\xi_1} - u_{2\xi_2}x_{2\xi_1}x_{1\xi_1}\right]_{\xi_2}$
u_2	$\left(P+\frac{2}{3}k\right)_{\xi_1} x_{1\xi_2} - \left(P+\frac{2}{3}k\right)_{\xi_2} x_{2\xi_2}$ $-\frac{\nu_{eff}}{J}\left[u_{1\xi_1}x_{1\xi_2}x_{2\xi_2} - u_{1\xi_2}x_{1\xi_1}x_{2\xi_2}\right]_{\xi_1} + \frac{\nu_{eff}}{J}\left[u_{1\xi_1}x_{1\xi_2}x_{2\xi_1} - u_{1\xi_2}x_{2\xi_2}x_{2\xi_1}\right]_{\xi_2}$ $+\frac{\nu_{eff}}{J}\left[u_{2\xi_1}x_{1\xi_2}^2 - u_{2\xi_2}x_{1x_1}x_{1\xi_2}\right]_{\xi_1} - \frac{\nu_{eff}}{J}\left[u_{2\xi_1}x_{1\xi_2} - u_{2\xi_2}x_{1\xi_1}^2\right]_{\xi_2}$

Table A.1: Source terms of the momentum equations in 2D curvilinear coordinates.

References

[1] H. Ahn, F. S. Lien, J. Larsson, and J. Hines. Simulation of aeroacoustic resonance in a deep cavity with grazing flow using a pressure-based solver. *International Journal of Computational Fluid Dynamics*, 22(1):39–47, 2008. 2, 94

[2] J. D. Anderson. *Computational Fluid Dynamics, The Basics with Applications*. McGraw-Hill, Inc., New York, 1st edition, 1995. 16

[3] G. Bosch and W. Rodi. Simulation of vortex shedding past a square cylinder with different turbulence models. *International Journal for Numerical Methods in Fluids*, 28(4):601–616, 1998. ix, 6, 8, 9, 83, 91

[4] D. Bouris and G. Bergeles. 2D LES of vortex shedding from a square cylinder. *Journal of Wind Engineering and Industrial Aerodynamics*, 80(1-2):31–46, 1999. ix, 6, 7

[5] O. R. Burggraf. Analytical and numerical studies of the structure of steady separated flows. *Journal of Fluid Mechanics*, 24:113–151, 1966. 4

[6] CD-adapco. CFD Solutions | CFD Software | Fluid Mechanics | Computational Fluid Dynamics. Web Site, 2008. http://www.cd-adapco.com/products/STAR-CD/index.html. 2

[7] A. J. Chorin. Numerical solution of the Navier-Stokes Equations. *Mathematics of Computation*, 22(104):745–762, 1968. 11

[8] L. Davidson and S. H. Peng. Hybrid LES-RANS modelling: a one-equation SGS model combined with a $k - \omega$ model for predicting recirculating flows. *International Journal for Numerical Methods in Fluids*, 43(9):1003–1018, 2003. 2

[9] R. W. Davis and E. F. Moore. A numerical study of vortex shedding from rectangles. *Joural of Fluid Mechanics*, 116:475–506, 1982. 5, 6, 61

[10] I. Demirdžiü, Z. Lileký, and M. Periü. Fluid flow and heat transfer test problems for non-orthogonal grids: Bench-mark solutions. *International Journal for Numerical Methods in Fluids*, 15:329–354, 1992. 4

[11] E. Erturk and B. Dursun. Numerical solutions of 2-D steady incompressible flow in a driven skewed cavity. *ZAMM - Journal of Applied Mathematics and Mechanics*, 87:377–392, 2007. x, xii, 4, 37, 38, 40, 42, 43, 44, 45, 52

[12] R. Franke and W. Rodi. Calculation of vortex shedding past a square cylinder with various turbulence models. In *Turbulent Shear Flows 8: Selected Papers from the Eigth Symposium on Turbulent Shear Flows*. Springer Verlag, 1993. ix, 8, 9, 83

[13] R. Franke, W. Rodi, and B. Schnung. Numerical calculation of laminar vortex-shedding flow past cylinders. *Journal of Wind Engineering and Industrial Aerodynamics*, 35(1-3):237–257, 1990. 5, 6, 61

[14] Ansys Inc. Ansys cfx fast, reliable, robust, accurate numerics. Web Site, 2006. www.ansys.com/assets/tech-briefs/cfx-numerics.pdf. 2

[15] Fluent Inc. CFD Flow Modeling Software & Solutions from Fluent. Web Site, 2008. http://www.fluent.com. 2

[16] M. Kato and B. E. Launder. The modeling of turbulent flow around stationary and vibrating square cylinders. In *Proceedings of the 9th Symposium on Turbulent Shear Flows*, Kyoto, 1993. ix, 8, 9, 14, 35, 83, 91

[17] M. Kawaguti. Numerical solution of the Navier-Stokes equations for the flow around a circular cylinder at Reynolds number 40. *Journal of the Physical Society of Japan*, 8:747–757, 1953. 1

[18] J. Kim and P. Moin. Application of a fractional-step method to incompressible Navier-Stokes equations. *Journal of Computational Physics*, 59:308–323, 1985. 2, 11

[19] H. Le and P. Moin. An improvement of fractional step methods for the incompressible Navier-Stokes equations. *Journal of Computational Physics*, 92(2):369–379, 1991. 2

[20] E. S. Lee, C. Moulinec, R. Xu, D. Violeau, D. Laurence, and P. Stansby. Comparisons of weakly compressible and truly incompressible algorithms for

the SPH mesh free particle method. *Journal of Computational Physics*, 227(18):8417–8436, 2008. 5

[21] B. P. Leonard. A stable and accurate convective modelling procedure based on quadratic upstream interpolation. *Computer Methods in Applied Mechanics and Engineering*, 19(1):59–98, 1979. 20

[22] F. S. Lien and M. A. Leschziner. A general non-orthogonal finite volume algorithm for turbulent flow at all speeds incorporating second-moment closure, part 1: numerical implementation. *Computer Methods in Applied Mechanics and Engineering*, 114:123–148, 1994. ix, 11, 15

[23] F. S. Lien, E. Yee, H. Ji, and K. J. Hsieh. Partially resolved numerical simulation and RANS modeling of flow and passive scalar transport in an urban environment. *Journal of Wind Engineering and Industrial Aerodynamics*, 96(10-11):1832–1842, 2008. 3

[24] N. Liu and T. Shih. Turbulence modeling for very large-eddy simulation. *AIAA Journal*, 44(4):687–697, 2006. 3

[25] M. Louaked, L. Hanich, and K. D. Nguyen. An efficient finite difference technique for computing incompressible viscous flows. *International Journal for Numerical Methods in Fluids*, 25(9):1057–1082, 1997. 4

[26] D. A. Lyn, S. Einav, W. Rodi, and J. H. Park. A laser-Doppler velocoimetry study of ensemble averaged charateristics of the turbulent near wake of a square cylinder. *Journal of Fluid Mechanics*, 304:285–319, 1995. ix, 6, 83

[27] K. Mahesh, G. Constantinescu, and P. Moin. A numerical method for large-eddy simulation in complex geometries. *Journal of Computational Physics*, 197(1):215–240, 2004. 2

[28] S. Murakami and A. Mochida. On turbulent vortex shedding flow past 2D square cylinder predicted by CFD. *Journal of Wind Engineering and Industrial Aerodynamics*, 54-55:191–211, 1995. ix, 6, 7

[29] The Shared Hierarchical Academic Research Computing Network. SHARCNET - shared hierarchical academic research computing network. Website, 2008. www.sharcnet.ca. 65

[30] A. Okajima. Strouhal numbers of rectangular cylinders. *Journal of Fluid Mechanics*, 123:379–398, 1982. 5, 6, 60

[31] S. V. Patankar and D. B. Spalding. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *International Journal of Heat and Mass Transfer*, 15:1787–1806, 1972. 2, 11

[32] C. M. Rhie and W. L. Chow. Numerical study of the turbulent flow past an airfoil with trailing edge separation. *AIAA Journal*, 21(11):1525–1532, 1983. 11, 20, 28

[33] A. Shklyar and A. Arbel. Numerical method for calculation of the incompressible flow in general curvilinear co-ordinates with double staggered grid. *International Journal for Numerical Methods in Fluids*, 41(12):1273–1294, 2003. 4

[34] A. Sohankar, C. Norberg, and L. Davidson. A numerical study of unsteady two-dimensional flow around rectangular cylinders at incidence. Internal Report 96/25, Chalmers University of Technology, Gothenburg, Sweden, 1996. 5, 6, 60, 61

[35] A. Sohankar, C. Norberg, and L. Davidson. Numerical simulation of unsteady low-Reynolds number flow around rectangular cylinders at incidence. *Journal of Wind Engineering and Industrial Aerodynamics*, 69-71:189–201, 1997. 5, 6, 54, 60, 61

[36] H. L. Stone. Iterative solution of implicit approximations of multidimensional partial differential equations. *SIAM Journal on Numerical Analysis*, 5(3):530–558, 1968. 94

[37] R. Témam. Sur l'approximation de la solution des quations de navier-stokes par la mthode des pas fractionnaires (ii). *Archive for Rational Mechanics and Analysis*, 33(5):377–385, 1969. 11

[38] G. P. Thompson. A numerical study of the vortex shedding from a square cylinder in a uniform cross-flow. Master's thesis, University of Waterloo, 2003. ix, 5, 6, 8, 11, 54, 60, 61, 83

[39] S. Turek. A comparative study of time-stepping techniques for the incompressible Navier-Stokes equations: from fully implicit non-linear schemes to semi-implicit projection methods. *International Journal for Numerical Methods in Fluids*, 22(10):987–1011, 1996. 8, 9, 94

[40] P. Voke. Second ERCOFTAC workshop on Direct and Large-Eddy Simulation: Flow Past a Square Cylinder: Test Case LES2. Website, 1998. http://ercoftac.mech.surrey.ac.uk/LESig/dles2/Proceedings/uos.ps. ix, 6, 7

[41] H. Xu and C. Zhang. Numerical calculations of laminar flows using contravariant velocity fluxes. *Computers and Fluids*, 29(2):149–177, 2000. 4

[42] Y. Zang, R. L. Street, and J. R. Koseff. A non-staggered grid, fractional step method for time-dependent incompressible Navier-Stokes equations in curvilinear coordinates. *Journal of Computational Physics*, 114(1):18–33, 1994. 2, 11