Numerical method for simulation of unsteady flows with immersed moving/deforming elastic structures on unstructured grids

Lv Xin

2008

Lv X. (2008). Numerical method for simulation of unsteady flows with immersed moving/deforming elastic structures on unstructured grids. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/41529>

<https://doi.org/10.32657/10356/41529>

**A NUMERICAL METHOD FOR SIMULATION OF
UNSTEADY FLOWS WITH IMMERSED
MOVING/DEFORMING ELASTIC STRUCTURES ON
UNSTRUCTURED GRIDS**

lv Xin

School of Mechanical & Aerospace Engineering

A thesis submitted to the Nanyang Technological University
in fulfilment of the requirement for the degree of
Doctor of Philosophy

2008

ABSTRACT

ABSTRACT

This work aims to apply Computational Fluid Dynamics (CFD) and Computational Structural Dynamics (CSD) to simulate the interactions between unsteady compressible flow and immersed moving and/or deforming solid structures. The main challenge is due to the fact that fluid dynamics, structural dynamics and their interactions are highly nonlinear, multiscale and multiphysical phenomena. In this work, an efficient and accurate numerical simulation package, incorporating solution methods for both three-dimensional Navier-Stokes (NS) and structural dynamic equations, has been successfully developed and validated. The baseline method is a finite-volume scheme using unstructured grids. Such features ensure the convenient and accurate modeling of complex geometries. A 3rd-order high-resolution edge-based Roe approximate scheme is adopted in the NS solver to accurately capture the possible existence of shock in high-speed flow regions, while cell-based 2nd-order Galerkin-type formulation is used to calculate the variable gradients. For temporal integration, a novel matrix-free implicit dual time-stepping is adopted. To simulate turbulent flows, a novel mixed dynamic formulation of eddy-viscosity subgrid model based on Smagorinsky-Lilly method has been integrated into the Large-eddy simulation module of the package. To couple the NS and structural dynamics solvers more efficiently, the immersed membrane method (IMM) is adopted and enhanced to handle the fluid-structure interaction. In the IMM, a stationary Eulerian mesh is used for the computation of fluid domain, while the spatial domain occupied by the moving and/or deforming solid structures are treated as “vacuum”, whose domain is defined by the fluid-structure interface which may be

ABSTRACT

varying with time. To obtain accurate flow conditions in the vicinity of the interface, without changing the topology and geometry of the fluid mesh, the concept of ghost nodes with ghost values is introduced to account for discontinuities in pressures and fluid stresses across the interface. Velocity continuity at the fluid-structure interface is directly enforced by a newly developed 3rd-order MUSCL like one-sided extrapolation scheme for fluid velocity, with the ghost nodes and the extrapolated velocities being treated as boundary conditions for the fluid domain. The fluid-structure interaction (FSI) is calculated in a staggered and iterative manner. Firstly, the governing equations for the fluid domain are solved, after which fluid forces are calculated along the fluid-structure interface. Then the structure dynamics are calculated under the influence of fluid forces and the structure is moved to a new position with a new configuration. Lastly, the immersed membrane method is employed to obtain appropriate boundary conditions at the fluid-structure interface for the fluid domain with the newly updated structure configuration. The above procedure is repeated up to a desired time instant. The proposed IMM is suitable for various kinds of fluid-structure interaction problems. It can solve FSI phenomena involving both rigid and flexible bodies. It can also be extended to solve the FSI problems for objects with arbitrary shapes, where surfaces of the immersed objects are treated as immersed membranes and the insides of the objects do not have fluids. For the purpose of enhancing overall convergence rates, several state-of-the-arts technologies are implemented, which include implicit residual smoothing, low speed preconditioning and multigrid method. In order to further improve computational efficiency, the solver is parallelized using the Single Program Multiple Data (SPMD) programming paradigm and Message Passing Interface (MPI).

ABSTRACT

The efficiency, accuracy and robustness of the proposed numerical methods are tested as follows. The baseline NS solver with immersed rigid structure is firstly validated using several classical benchmark problems, such as a sphere rotating and oscillating in quiescent air, steady and unsteady flow over a stationary circular cylinder, and rigid thin membrane immersed in channel flow. Next, the spatial and temporal accuracy of the structural dynamic solver is validated using a spatial point-loaded fixed-free 2D and 3D cantilever problem. Finally, the capability of the developed package is demonstrated by applying it to two potential engineering applications: steady and unsteady air flow between two corotating disks in a fixed cylindrical enclosure, and the aeroelastic flutter of an ONERA M6 wing in transonic flow. All of the results obtained agree well with published analytical, numerical and experimental results wherever available. These show that the current package is capable of solving large displacement fluid-structure interaction problems. Error analysis is also carried out during the validation process and the results obtained indicate that the immersed membrane method is at least globally 2nd-order accurate.

Dedicated to my parents,

to my wife, Zhang Yali

and to my little baby, Jingyan

ACKNOWLEDGMENTS

The author would like to express his most sincere appreciation to his research supervisors, Associate Professor Zhao Yong and Associate Professor Huang Xiaoyang, for their immense unconditional support and encouragement throughout the research project. Most importantly, the author wishes to thank them for their patient guidance into the depth and breadth of my research project. He has learned a great deal from their invaluable insights into many of the numerical and physical problems. He would like to thank all the members of his supervisor's research group for all the help and suggestions they have given to him.

He also would like to express his gratitude to Prof. Wong Teck Neng for his kind help during the special period when the author's wife was delivering the baby; and to Prof. Yang Chun for his continuous care for the author's study and life.

His study and life at NTU has been enriched greatly by his friends and colleagues here. He would like to thank all of them for their help and support, especially, Dr. Xia Guohua and Dr. Su Xiaohui for many useful discussions and helps during the course of this work.

During his PhD study, the author's families have been very supportive. Their unconditional support has given him the strength to finish his study. He is very grateful to them. He is also much indebted to his wife *Yali* for all her love and support.

The author would like to acknowledge the Research Scholarship provided by both Nanyang Technological University and the Center for Mechanics and Micro-System (CMMS). He would also like to acknowledge the Centre for Advanced Numerical Engineering Simulations (CANES) and CMMS for their generous help and the computational facilities provided.

LIST OF KEYWORDS

Computational fluid dynamics; Computational structural dynamics; compressible flow; finite volume; MUSCL interpolation; high-resolution Roe scheme; low speed preconditioning; large-eddy simulation; Smagorinsky-Lilly model; unstructured tetrahedral grid; filtering; matrix-free implicit dual time stepping scheme; unsteady viscous flow; parallel-multigrid computing; immersed membrane method; fluid-structure interaction; ONERA M6 wing.

CONTENTS

CONTENTS

ABSTRACT.....	I
ACKNOWLEDGMENTS	IV
LIST OF KEYWORDS	VII
LIST OF FIGURES	XIV
LIST OF TABLES	XXII
NOMENCLATURE.....	XXIII
PUBLICATIONS.....	XXVII
CHAPTERS	
1 INTRODUCTION	1
1.1 BACKGROUND	2
1.2 POTENTIAL ENGINEERING APPLICATIONS	6
1.2.1 Hard Disk Drives (HDDs)	6
1.2.2 Offshore Wind Turbines	7
1.2.3 Summary.....	8
1.3 OBJECTIVE AND SCOPE OF RESEARCH.....	9
1.4 THESIS STRUCTURE.....	10
2 LITERATURE REVIEW	11
2.1 REVIEWS ON COMPUTATIONAL FLUID DYNAMICS	12
2.1.1 The Multigrid Method.....	12
2.1.1.1 Serial Multigrid Method	13

CONTENTS

2.1.1.2	Parallel Multigrid Method	15
2.1.2	Parallel Computation	18
2.1.3	Low Mach-Number Preconditioning	21
2.1.4	Finite Volume (FV)-Based Computational Structural Dynamics	24
2.1.5	Moving Boundary	25
2.1.5.1	Eulerian Method	26
2.1.5.2	Arbitrary Lagrangian-Eulerian (ALE) Method	30
2.1.5.3	Hybrid Method	34
2.1.6	Fluid-Structure Interaction (FSI)	35
2.2	SOME REMARKS	38
2.2.1	Serial/Parallel Multigrid Method	38
2.2.2	Parallel Computing	39
2.2.3	Low Mach-Number Preconditioning	39
2.2.4	FV Based Computational Structural Dynamics	40
2.2.5	Moving Boundary and Fluid-Structure Interaction	40
3	MATHEMATICAL FORMULATION FOR PRECONDITIONED COMPRESSIBLE FLOW SOLVER	42
3.1	BASIC GOVERNING EQUATIONS FOR THE NAVIER- STOKES FLUIDS	43
3.2	LOW SPEED PRECONDITIONING FORMULATION	48
3.3	DISCRETIZATION METHODS	50
3.4	BOUNDARY TREATMENT	63
3.4.1	Far Field Conditions	64
3.4.2	Solid Wall/Slip Conditions	69
3.4.3	Far Field Conditions for Preconditioned Systems	69
4	MATHEMATICAL FORMULATION FOR COMPUTATIONAL STRUCTURAL DYNAMICS	71
4.1	GOVERNING EQUATIONS	72

CONTENTS

4.2	CONSTITUTIVE RELATIONSHIP FOR STRESS AND STRAIN	73
4.3	DISPLACEMENT FORMULATION	74
4.4	DISCRETIZATION OF THE DISPLACEMENT EQUATIONS	76
4.5	CONVERGENCE ACCELERATION TECHNIQUES	83
4.5.1	Local Time Stepping in Pseudo Time	83
4.5.2	Implicit Residual Smoothing	85
4.5.3	The Multigrid Method	86
5	IMMERSED MEMBRANE METHOD AND FLUID-STRUCTURE INTERACTION	90
5.1	DEFINITION OF GHOST FLUID NODES	91
5.2	INTERACTIONS BETWEEN FLUID AND STRUCTURE MESHES	96
5.3	ONE-SIDED MUSCL LIKE EXTRAPOLATION	105
5.4	IMM BASED FLUID-STRUCTURE INTERACTION	111
6	THE MULTIGRID METHOD AND PARALLEL COMPUTATION	119
6.1	THE MULTIGRID METHOD	120
6.1.1	Mathematical Formulations	123
6.1.2	Data Structure for Different Multigrid Level	126
6.1.3	Zoning of Flow Field Domain	126
6.1.4	Inter-Connectivity Relationship between Meshes	128
6.1.5	Mesh-to-Mesh Transfer Operators	128
6.1.5.1	Flow Field Variables Transfer Operators	129
6.1.5.2	Residuals Transfer Operators	131
6.1.5.3	Correction Transfer Operators	134
6.2	PARALLEL COMPUTATION	137
6.2.1	One-level and Two-level Parallelization Strategies	138
6.2.2	One-Level Parallelization Strategy - Domain Decomposition	141
6.2.2.1	Meshes Partition algorithm	141

CONTENTS

6.2.2.2 Identification of Ghost Nodes and Overlapping Elements	142
6.2.2.3 Grid Files with Local Numbering.....	145
6.2.2.4 Data Structure for Intercommunications between Partitions	145
6.2.3 Multigrid Parallelization	146
6.2.4 Performance Measuring Techniques.....	148
6.2.4.1 Speed-up	149
6.2.4.2 Parallel Efficiency.....	150
7 LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW	151
7.1 FILTERED NAVIER-STOKES EQUATIONS FOR THE COMPRESSIBLE FLOW	152
7.2 DYNAMIC MIXED SUBGRID MODELLING FOR THE TURBULENCE STRESS TENSOR.....	158
7.3 DYNAMIC MIXED SUBGRID MODELLING FOR THE ENERGY EQUATION	167
7.4 FILTERED GOVERNING EQUATION IN VECTOR FORM.....	173
7.5 COMMUTATIVE DISCRETE FILTERING OPERATOR ON UNSTRUCTURED GRIDS.....	176
7.6 COOPERATING LES WITH IMM	182
8 NUMERICAL RESULTS AND DISCUSSION.....	185
8.1 VALIDATION OF BASELINE COMPRESSIBLE FLOW SOLVER WITH IMM	186
8.1.1 Steady Flow Induced By a Rotating Sphere in Quiescent Air – Order of Accuracy	186
8.1.2 Flow Induced by an Oscillating Sphere in a Closed Cavity	191
8.1.3 Steady and Unsteady Flow past a Circular Cylinder	194

CONTENTS

8.1.3.1	Steady flow	197
8.1.3.2	Unsteady flow	200
8.1.4	Flow over an Immersed Fixed Membrane	206
8.2	VALIDATION OF THE COMPUTATIONAL STRUCTURAL DYNAMICS SOLVER	212
8.2.1	Deformation of Point Loaded Fixed-Free Cantilever Structures	212
8.2.1.1	Two-Dimensional Cantilever	212
8.2.1.2	Three-Dimensional Cantilever	220
8.2.2	Three Dimensional Fixed-Free Cantilever Immersed in Fluid Flow	223
8.3	SUMMARY	237
8.4	STEADY AND UNSTEADY AIR FLOWS BETWEEN TWO RIGID DISKS COROTATING IN A FIXED CYLINDRICAL ENCLOSURE	239
8.4.1	Three-Dimensional Steady Flow	243
8.4.2	Three-Dimensional Unsteady Flow	249
8.4.2.1	Case One: $Re=22,200$	249
8.4.2.2	Case Two: $Re=44,400$	253
8.4.2.3	Effects of Disk-Enclosure Gap and Disk Thickness	258
8.4.2.4	Remarks	260
8.5	LARGE-EDDY SIMULATION OF AEROELASTIC FLUTTER FOR THE ONERA M6 WING IN THE TRANSONIC FLOW	260
8.5.1	Geometry and Calculation Parameters	261
8.5.2	LES of Steady Aerodynamic Computation of M6 Wing	264
8.5.3	LES of ONERA M6 Wing Flutter	275
9	CONCLUSIONS AND FUTURE WORK	283
9.1	CONCLUSIONS	284
9.1.1	Development and Validation of Numerical Schemes and Algorithms	285
9.1.2	Applications to Rotating-Disk Flow and Aeroelastic Wing Flutter	289
9.2	FUTURE WORKS	293

CONTENTS

APPENDIX (MATHEMATICAL DETAILS OF PRECONDITIONING) 296

REFERENCES..... 299

LIST OF FIGURES

LIST OF FIGURES

Figure 3.1	Construction of control volume within a tetrahedron for a node P	50
Figure 3.2	The inflow & outflow boundary conditions for a strong converged nozzle	67
Figure 4.1	Schematic of boundary node projection algorithm, is the computational domain.....	89
Figure 5.9	Introduction of higher-order MUSCL like interpolation.....	110
Figure 5.10	Schematic of our FSI solution	112
Figure 5.11	Relation between the structure domain and the fluid domain in fluid- structure interaction	114
Figure 6.1	Smoothing of errors by relaxation sweeps	121
Figure 6.2a	V multigrid cycles for three grid levels	122
Figure 6.2b	W multigrid cycles for three grid levels. $\textcircled{E}$ denotes flow field computation on a particular level and $\textcircled{I}$ denote interpolation of corrections of flow field variables.	123
Figure 6.3	Zoning of flow field domain [150].....	127
Figure 6.4a	Transfer of flow field values from the fine mesh to the coarse mesh using restriction transfer operator	130
Figure 6.4b	Transfer of variables from fine nodes to the coarse node at the boundary	130
Figure 6.5a	Transfer of residuals from the fine mesh to the coarse mesh using restriction transfer operator.....	132

LIST OF FIGURES

Figure 6.5b	Transfer of residuals from fine node to the coarse nodes at the boundary	133
Figure 6.6a	Transfer of corrections from the coarse mesh to the fine mesh using prolongation transfer operator.....	135
Figure 6.6b	Transfer of corrections from the coarse nodes to the fine node at the boundary	136
Figure 6.7	Flowchart depicting geometric domain decomposition parallelization strategy alone using MPI only [150].....	139
Figure 7.1	Fragment of 2D unstructured triangular grid illustrating gradient reconstruction stencil at vertex P.	178
Figure 7.2	schematic of the test-grid stencil on a 2D unstructured grid with an immersed boundary. Only 1th neighboring nodes are used to construct the filtering operator for node P. Extrapolated ghost values at i and j will be used to evaluate the filtering value of node P.	184
Figure 8.1	Convergence of the L_∞ , L_1 and L_2 error norms for the velocity field induced by a rotating sphere. Slope 1 and Slope 2 are the reference lines for 1-order and 2-order accuracy respectively.	188
Figure 8.2	Instantaneous snapshots of streamlines at the $y = 2.5$ plane depicting the early stages of flow evolution toward steady state for $Re = 100$. Time is measured from the start of the impulsive rotation and T is the rotation period of the sphere.	190
Figure 8.3	The angle between the z-axis and the radius that passes through the centre of the toroidal vortex ring.	191

LIST OF FIGURES

Figure 8.4	Instantaneous streamlines at $t = 1.0T$ for flow induced by an oscillating sphere in a closed cube filled with compressible and viscous fluid.	193
Figure 8.5	Convergence of the L_∞ , L_1 and L_2 error norms for the velocity field of oscillating sphere in a closed cube. Slope 1 and Slope 2 are the reference lines for 1-order and 2-order accuracy respectively.....	194
Figure 8.6	The computational domain and the immersed cylinder for viscous flow past a circular cylinder.....	195
Figure 8.7	Lift coefficients versus physical time using 3 different grids for flow over a circular cylinder (Parallel-MG, Preconditioned, $Re = 200$, inflow $Mach=0.2$).	196
Figure 8.8	Convergence histories for steady flow over a circular cylinder.	198
Figure 8.9	Streamlines plots ($z=0.25$) for flow over a three-dimensional stationary circular cylinder at $Re = 40$ and inflow $Mach=0.01$ using (a) Parallel-MG and preconditioning, (b) Parallel-MG without preconditioning.	199
Figure 8.10	Length of separation bubbles behind cylinder vs. Re [191].	200
Figure 8.11	Lift and drag coefficients versus physical time for flow over a circular cylinder using 3-level Parallel-MG and preconditioning (number of sub-iterations = 60 V-cycles, $Re = 200$, inflow $Mach = 0.2$).	201
Figure 8.12	Streamline patterns showing one cycle of vortex shedding using 3-level Parallel-MG and preconditioning ($z=0.25$, number of sub-iterations = 60 V-cycles, $Re = 200$, inflow $Mach=0.2$).	203

LIST OF FIGURES

Figure 8.13 Mach number contours showing one cycle of vortex shedding using 3-level Parallel-MG and preconditioning ($z=0.25$, number of sub-iterations = 60 V-cycles, $Re = 200$, inflow Mach=0.2)	205
Figure 8.14 Unsteady flow convergence history plot (MG used “V” cycle, number of sub-iterations = 60, $Re = 200$, inflow Mach=0.2).	206
Figure 8.15 Geometry of the computational domain for the flow over an immersed fixed membrane (1 unit length = 20 mm).	208
Figure 8.16 Convergence histories with immersed membrane and internal boundary.	209
Figure 8.17 (a. above) Streamlines on the channel central plane with the immersed membrane represented by the IMM; (b. below) flow field with an internal boundary computed by baseline solver.	210
Figure 8.18 (a. above) Comparison of velocity profiles at $X=0$, $Z=0.5$ (b. below) Comparison of velocity profiles at $X=1$, $Z=0.5$	211
Figure 8.19 Schematic of flexural (bending) deformation test of fixed-free cantilever.	213
Figure 8.20 Mesh refinements versus percentage error	216
Figure 8.21 Direct and shear stress distribution in the equilibrium state for the static point loaded free-fixed 2D cantilever	217
Figure 8.22 Displacement history of the tip in y direction for dynamic point loading on free-fixed 3D cantilever	219
Figure 8.23 Convergence rate acceleration given by 3-level MG method	221
Figure 8.24 Tip displacement history in z direction for the point loaded fixed-free 3D cantilever, 3-level MG method was employed	223
Figure 8.25 Schematic three-dimensional free-fixed cantilever in fluid flow	225

LIST OF FIGURES

Figure 8.26 Direct and shear stress distribution in the equilibrium state for the fixed-free cantilever immersed in fluid.....	226
Figure 8.27 Immersed cantilever tip vertical displacement history during the FSI process	227
Figure 8.28 V velocity contours in XY plane, $Re=18.7*10^5$	228
Figure 8.29 V velocity contours and streamlines in YZ plane, $Re=18.7*10^5$	229
Figure 8.30 Volume streamlines pattern depict the separation under the immersed cantilever, $Re=18.7*10^5$	230
Figure 8.31 Convergence of the L_∞ , L_1 and L_2 error norms for the velocity field with a immersed fixed-free cantilever. Slope 1 and Slope 2 are the reference lines for 1-order and 2-order accuracy respectively.....	234
Figure 8.32 Comparison of CPU times used by the two solvers	234
Figure 8.33 (a) Grid convergent comparison between the two solvers; (b), (c) and (d): First three fundamental mode shapes comparison between the two solvers. Note that in all pictures then beam is represented by its central line, and the displacements are normalized to unity	236
Figure 8.34 Schematic of the corotating disks flow configuration indicating the toroidal vortices that develop in the cross-stream plane near the fixed enclosure wall. The basic flow regions (I-V) induced in the inter-disk space are sketched in accordance with previous studies [194-198]: (I) enclosure wall boundary layer; (II) disk Ekman layers; (III) the region in solid-body rotation; (IV) core potential flow; (V) detached shear layer. In the bulk of the calculations, $R_1=56.4$ mm, $R_2=105.7$ mm, $H=9.53$ mm, and $b=0$. Selective calculations have been performed with $R_1=56.4$ mm, $R_2=105.0$ mm, $H=9.53$ mm, $a=2.7$ mm, and	

LIST OF FIGURES

- b=1.91 mm. The latter values correspond exactly to an experimental configuration used in other studies [194, 197, 199, 200 and 201].... 240
- Figure 8.35 Comparison between measurements and calculations of the circumferential velocity component along the inter-disk mid-plane for (a) $Re=4,440$ (60 rpm), (b) 7,400 (100 rpm), and (c) 14,400 (200 rpm). The results for the last two grids virtually coincide..... 244
- Figure 8.36 Iso-contours of the velocity components, Mach number and Temperature on the r-z plane. Obtained from three-dimensional calculations for $Re=7400$. In figure (b), an arrow marks the thickness of an Ekman layer, estimated according to 246
- Figure 8.37 Streamline patterns on the r-z plane. A pair of symmetrical, toroidally shaped, counter-rotating vortices can be easily identified. 247
- Figure 8.38 Contours of u and circumferential velocity components, pressure and temperature on the inter-disk mid-plane. 248
- Figure 8.39 (a) Circumferential velocity components; (b) Radial velocity components; (c) Axial velocity components and (d) Streamline patterns in the r-z plane. Obtained from three-dimensional calculations for $Re=22,200$ ($\Omega=300rpm$) and $t=103$ second..... 250
- Figure 8.40 Time history of the dimensional circumferential, radial and axial velocity components, as well as the nondimensional parameter, pressure coefficient (C_p), at a monitoring point located on the inter-disk mid-plane ($r/R_2=1.0$). Obtained from three-dimensional calculations for $Re=22,200$ ($\Omega=300rpm$)..... 251
- Figure 8.41 Comparison between measurements (circles) and calculations (line with triangles) of the nondimensional circumferential velocity

LIST OF FIGURES

component (a) mean and (b) rms along the inter-disk mid-plane for Re=22,200 ($\Omega=300\text{rpm}$).	253
Figure 8.42 Iso-contours of the axial vorticity component, ω_z , on the inter-disk mid-plane from three-dimensional calculations for Re=44,400 (600 rpm). 6 positive foci (colored by red) can be clearly identified.....	255
Figure 8.43 Iso-surface of the axial velocity component, u_z , from three- dimensional calculations for Re=44,400.	256
Figure 8.44 Instantaneous plot of the cross-stream flow field from three- dimensional calculation for Re=44,400 (600 rpm).....	257
Figure 8.45 (a) Circumferential velocity components; (b) Radial velocity components; (c) Axial velocity components and (d) Streamline patterns on the r-z plane. Obtained from three-dimensional calculations for Re=15,000.	259
Figure 8.46 Geometric layout of the ONERA M6 wing.....	262
Figure 8.47 Schematic of computational domain of the ONERA M6 wing.....	263
Figure 8.48 Computational grids near the tip of the ONERA M6 wing.....	264
Figure 8.49 Time history of the drag coefficient of the ONERA M6 wing.....	265
Figure 8.50 Time history of the lift coefficient of the ONERA M6 wing.	265
Figure 8.51 Time history of the moment coefficient of the ONERA M6 wing...	266
Figure 8.52 Time averaged Mach number distribution on the symmetric plane and upper surface of the ONERA M6 wing.	267
Figure 8.53 Time averaged pressure coefficients distributions on seven selected locations on ONERA M6 wing.....	271
Figure 8.54 Distributions of Time averaged pressure coefficient, temperature and density on symmetric plane and wing surface.	273

LIST OF FIGURES

Figure 8.55 Distributions of ratio between test filtering width and grid filtering width on symmetric plane.....	274
Figure 8.56 Time averaged streamline ribbons around the M6 wing surface. (Contoured by dimensional velocity magnitude).....	275
Figure 8.57 Flutter responses for ONERA M6 wing using intermediate grid at and 3.06o angle of attack (Displacement in y direction)	277
Figure 8.58 Time history of lift and drag coefficients for M6 wing fluttering using intermediate grid at and 3.06o angle of attack.....	278-279
Figure 8.59 Instantaneous Mach number distributions at three different moments in one cycle of the wing flutter	281-282
Figure 9.1 Demonstration of overlapping grid in IMM method dealing with the boundary layer around the structure; this method make it possible to simulate the FSI involving large deformation of structure.	295

LIST OF TABLES

LIST OF TABLES

Table 8.1 Rate of convergence calculated for different error norms	189
Table 8.2 Comparison of the measured [188, 189] and calculated angle	189
Table 8.3 Lift and drag coefficients and Strouhal number for unsteady flow over a three-dimensional circular cylinder ($Re=200$).	202
Table 8.4 Performance for parallel computation of unsteady flow past a circular cylinder ($Re = 200$, $M=0.3$ with preconditioning)	204
Table 8.5 Properties of vortices of flow field with immersed membrane and flow field with internal boundary.	212
Table 8.6 Computational parameters for the 2D cantilever	215
Table 8.7 Computational parameters for the 3D cantilever	220
Table 8.8 Rate of convergence calculated for different error norms	232
Table 8.9 Predicted mean Lift and Drag coefficients	266

Table 8.9 Predicted mean Lift and Drag coefficients

266

NOMENCLATURE

NOMENCLATURE

CFD	Computational Fluid Dynamics.
CSD	Computational Structural Dynamics.
CFL	Courant-Friedrichs-Lewy number.
dW_{h+1}	Correction from coarse grid $h+1$.
E_p	Efficiency for parallel computing.
FV	Finite Volume.
I_{h+1}^h	Correction transfer operator from the coarse grid $h+1$ to the fine grid h .
IMM	Immersed Membrane Method
$\vec{F}_i$	Convective flux vectors.
$\vec{F}_i^+$	Positive term of flux vector of convected quantities.
$\vec{F}_i^-$	Negative term of flux vector of convected quantities.
$\vec{F}_v$	Viscous flux vectors.
FSI	Fluid-Structure Interaction
L	Characteristic length.
LES	Large-Eddy Simulation.
M_∞	Free stream Mach number.
MG	Multigrid.
MPI	Message Passing Interface.
$\vec{n}$	Unit normal vector.
p	Pressure.
p_{out}	Outlet pressure.

NOMENCLATURE

Pr	Prandtl number.
Pr_t	Turbulent Prandtl number.
P_{h+1}	Forcing function in Runge-Kutta scheme for multigrid method.
Q_h^{h+1}	Residual transfer operator from fine grid h to the coarse grid $h+1$.
q	Heat flux.
Q	SGS heat flux.
$R(W_p)$	Flux residual.
$\tilde{R}(W_p^{n+1})$	Modified flux residual in implicit time stepping.
R	Gas constant.
Re	Reynolds number.
SG	Single Grid.
SGS	Subgrid-Scale.
S_p	Speed-up for parallel computing.
$SPMD$	Single Program Multiple Data.
S_t	Strouhal number.
T	Period for a cardiac cycle.
t	Time.
$\Delta\tau$	Pseudo-time step.
Δt	Real time step.
t_p	Total simulation wall-clock time.
t_{cpu}	CPU time.
t_{comm}	Communication time.
t_{comp}	Computation time.
t_{idle}	Idling time.

NOMENCLATURE

$t_{\text{comm}}^{\text{FFV}}$	Communication time for exchanging of flow field variables.
$t_{\text{comm}}^{\text{FT}}$	Communication time for exchanging of forcing terms.
$t_{\text{comm}}^{\text{NG}}$	Communication time for exchanging of nodal gradients.
T_h^{h+1}	Solution transfer operator from fine grid h to the coarse grid $h+1$.
U_∞	Free-stream velocity.
u, v, w	Velocity component in the x, y and z direction, respectively.
V-cycle	Multigrid V-cycle.
$\mathbf{W}_c$	Vector of conserved flow variables.
$\mathbf{W}_p$	Vector of primitive flow variables.
W-cycle	Multigrid W-cycle.
$\mathbf{W}_p$	Vector of conserved flow variables at node P.
$\mathbf{W}_h$	Solution from the fine grid.
$\mathbf{W}_h^+$	Updated variables of the solution on the fine grid h .
$\mathbf{W}_{h+1}^{(0)}$	Initial values transferred from the fine grid.
$\mathbf{W}_{h+1}^+$	Newly computed vaule on the coarse grid $h+1$.
x, y, z	Cartesian co-ordinates.

Greek Symbols

τ_{ij}	Viscous stress tensor.
σ_{ij}	SGS Reynolds stress tensor.
c	Speed of sound.
Γ_1	Preconditioning matrix.
ρ	Density.
ρ_∞	Free stream density.

NOMENCLATURE

μ	Molecule viscosity.
μ_t	Turbulent viscosity.
α_{RK}	Stage coefficients for Runge-Kutta time integration.
ε	Smoothing coefficient.
β	Preconditioning parameter.
β'	Preconditioning parameter.
κ	Weight used in MUSCL interpolation.
δ	Friction coefficient for structure dynamics equation.
γ	Ratio of specific heats.
Δ	Grid filtering length.
$\tilde{\Delta}$	Test filtering length.

PUBLICATIONS

The following papers are related to the present research:

- [1] X. Lv, Y. Zhao, X. Y. Huang, G. H. Xia and Z. J. Wang, "An Efficient Parallel/Unstructured-Multigrid Preconditioned Implicit Method for Simulating 3D Unsteady Compressible Flows with Arbitrarily Moving Objects," *Journal of Computational Physics*, Volume 215, Issue 2, 1 July 2006, Pages 661-690
- [2] X. Lv, Y. Zhao, X. Y. Huang, G. H. Xia, "A Matrix-Free Implicit Unstructured Multigrid Finite Volume Method for Simulating Structural Dynamics and Fluid-Structure Interaction," *Journal of Computational Physics*, Vol.225, Issue 1, p.120-144, 2007.
- [3] X. Lv, Y. Zhao, et al., An Efficient Parallel/Unstructured-Multigrid Implicit Method for Simulating 3D Fluid-Structure Interaction, Accepted by *Communication in Computational Physics*, 2008.
- [4] G. H. Xia, Y. Zhao, J. H. Yeo and X. Lv, "A 3D Implicit Unstructured-grid Finite Volume Method for Structural Dynamics," *Computational Mechanics*, available online, 2006.
- [5] X.H. Su, Y. Zhao, X.Y. Huang, X. Lv and C. Luo, "Parallel 3D Gas Dynamic of Nonlinear Resonant Oscillations in a Closed Resonator," submitted to *International Journal for Numerical Methods in Engineering*, 2006.

CHAPTER ONE

INTRODUCTION

In this chapter, the background and challenges of numerical studies on moving boundaries and fluid-structure interaction are briefly introduced. To begin with, the main difficulty is firstly described. And then, brief introduction to various types of typical numerical algorithms designed for this purpose is given. And, several representative potential engineering applications are discussed. This is followed by the description of research objective for this study-that is, developing a high performance computational package for numerical simulation of unsteady flow interacting with moving or deforming structures. Lastly, for clarity, the thesis structure is briefly outlined.

CHAPTER 1 – INTRODUCTION

1.1 BACKGROUND

Numerical simulations of unsteady flows interacting with dynamically moving or deforming structures are amongst the most challenging problems in computational fluid dynamics (CFD). The main challenge is due to the fact that fluid dynamic, structural dynamic and their interaction are highly nonlinear, multiscale and multiphysical phenomena. There are only a limited number of special cases where established boundary-conforming formulations can be directly applied to such problems with a relatively small overhead. Among which, the Arbitrary Lagrangian-Eulerian (ALE) method is no doubt the most popular one [1]. This method involves a continuous adaptation of the mesh without modifying the mesh topology in solving the fluid-structure interaction and moving boundary problem. The ALE methodology combines the best of features of both Eulerian and Lagrangian formulations. The essence of the ALE is that mesh motion can be chosen arbitrarily [2]. It includes three phases: an explicit Lagrangian update, an implicit iteration of the momentum equation with the equation of state, and a rezone/map phase [3]. The Lagrangian phase provides an explicit update of the equation of motion. When the material velocities are much smaller than the fluid sound speed, the optional implicit phase allows sound waves to move many cells per cycle, thereby significantly improving computational efficiency. Finally, a rezone algorithm may prescribe mesh velocities relative to the fluid, thus necessitating re-map phase in which the solution from the end of phase two is mapped onto the new mesh. A particularly important property of ALE is that it provides a means of minimizing advection errors [4]. ALE techniques have gained popularity for transient, high speed, small deformation problems in solid mechanics [5]. They are most commonly used in the aerospace field, where solid structures are subjected to complex

CHAPTER 1 – INTRODUCTION

airflows, e.g. see [6]. In some cases, however, due to the large deformation of structures within the computational domain, it is generally difficult to adapt the mesh in such a way that a proper mesh quality is maintained without changing the mesh topology. Alternatively, re-meshing can be performed, if the mesh quality is degenerated too much. This technique has been used by Horsten [7] to model the aortic valve in two dimensions. Re-meshing, however, not only introduces artificial diffusivity, it may also be difficult to perform with sufficient robustness and accuracy for three-dimensional problems. The use of moving reference frames [8], or coordinate transformations [9, 10] are some other examples. With more complex configurations moving and/or deforming grids that continuously adapt to the changing location of the body have to be adopted (see for example [11, 12]). For problems that involve multiple immersed bodies undergoing large motions and/or deformations, these algorithms are fairly complicated and have adverse impact on the accuracy and efficiency of the fluid solvers.

Although a variety of fluid-structure interaction algorithms has been developed over the years, relatively few applications in turbulent and transitional flows have been reported. In most cases, this is due to prohibitively high computational costs, or dissipative discretizations that limit the applicability of such methods to turbulence modeling. Further advancements in this field can be achieved by coupling state-of-the-art tools to model turbulence and transition (i.e. large-eddy simulations (LES) or hybrid formulations) with cost-effective numerical methods applicable to problems with large boundary motions and deformations.

An alternative class of methods that has the potential to overcome some of the above limitations is non-boundary-conforming formulations or the Eulerian Methods. In such

CHAPTER 1 – INTRODUCTION

methods the requirement that the grid conforms to a solid boundary is relaxed, and the effect of a complex object on the flow is introduced through proper treatment of the solution variables in the grid cells in the vicinity the body. The basic advantage of these formulations is the simplification of grid generation, especially in cases of moving or deforming boundaries where the need for regeneration or deformation of the grid is eliminated. Both the above features are particularly attractive for Direct Numerical Simulations (DNS) or LES of turbulent and transitional flows, where the use of highly efficient, energy conserving solvers is imperative for accurate computations. It is therefore conceivable that successful integration of non-boundary-conforming strategies with robust fluid solvers developed for DNS/LES will open new areas of applications for these tools. The immersed boundary (IB) method, proposed by Peskin *et al.* [13, 14], is a technique able to consider the moving boundaries and has been used to simulate the natural heart and heart valves combining Eulerian flow equations and a Lagrangian description of heart walls and valves. The philosophy of the IB method is to treat the elastic material as a part of the fluid in which additional forces (arising from the elastic stresses) are applied. The fluid equations are solved on a regular cubic lattice, the structure of which is not modified in any way by the presence of the immersed elastic bodies with complicated geometries. The elastic material is tracked in a Lagrangian fashion, by following a collection of representative material points. The spatial configuration of these points is used to compute elastic forces, which are applied to the nearby lattice points of the fluid. The fluid velocity is updated under the influence of these forces, and the new velocity is then interpolated at the elastic material points, which are moved at the interpolated velocity to complete the time step. The IB method has been applied to a wide range of problems, mostly in bio-fluid dynamics, including blood flow in the heart [14], platelet aggregation during blood clotting [15], fluid

CHAPTER 1 – INTRODUCTION

dynamics of the inner ear [16], flow in arterioles [17] and simulating the motion of flexible pulp fibers [18]. Later Peskin and co-investigators developed the method further and published an adaptive [19] and second-order accurate versions [20] of the IB method to enhance its accuracy. Generally speaking, the method is applicable to any problem in which a fluid interacts with an elastic material which may be undergoing large movement or deformation. The elastic material may be an active one, as in the case of cardiac muscle. A limitation of the IB method is that it is just suitable for considering the moving boundary as one-dimensional curves, thus it could only be used when the thickness of the moving object can be ignored. Following the methodology of Peskin, numerous formulations have been developed and reported. Majumdar *et al.* [21] proposed a ghost fluid cell method to treat the immersed boundary for structured grids. In his method, ghost cells are defined as cells in the solid domain that have at least one neighbour in the fluid domain. For each ghost cell, a complex interpolation scheme along wall normal direction that implicitly incorporates the boundary condition on the immersed boundary is then devised. Fedkiw [22] proposed another ghost fluid method to tackle the moving boundary problems. In his method, the fluid-structure interfaces are treated in Eulerian scheme that maintains a Heaviside profile of the density with no numerical smearing. A level set function is used to define two separate domains for the two separate fluids. Then a ghost cell at every point in the computational domain is imposed.

Up to the present, almost all of the published applications of the IB method are limited to incompressible Cartesian coordinate solvers, few of them involves compressible fluids or flows. No report on Fluid-Structure Interaction (FSI) using large-eddy simulation on unstructured meshes has been found.

CHAPTER 1 – INTRODUCTION

1.2 POTENTIAL ENGINEERING APPLICATIONS

Several potential applications and impacts of this study are listed in this section.

1.2.1 Hard Disk Drives (HDDs)

As a cornerstone of industries in Singapore, HDD manufacturing not only has offered a great number of employment opportunities, it also has stimulated the co-evolution of a wide range of supporting industrial activities in Singapore: printed circuit board (PCB) making and PCB assembly (PCBA) operations, die casting, metal stamping, precision machining and plating of various mechanical components such as base-plates, cover and actuator arms, connectors and automation and clean room design services. In order to achieve further technological breakthroughs, Singapore government established the Data Storage Institute (known earlier as the Magnetic Technology Centre) in 1992 to conduct research related to the data storage industry. In recent years, disk rotating speeds become higher and higher, resulting in a wide range of newly emerged challenges. The first problem must be addressed is the temperature control. It is well known that a large amount of thermal energy is generated inside hard disk drives as a result of high speed rotation of platters. In order to dissipate the generated heat out of the HDD casing efficiently, we must know the temperature distribution in the first place. But this information requires a full understanding of the details of fluid flow inside the casing. On the other hand, disks rotating at high speeds may become unstable due to their interaction with the surrounding air, which is called flutter. It leads to the vibration of actuator arms and positioning error signals (PES). In current HDD designs using high recording density and high-speed access, it is very important to estimate the loads exerted on both the disks and read/write arms to reduce undesirable vibration. It is

CHAPTER 1 – INTRODUCTION

estimated that to achieve 1Tb/in^2 areal density, the total airflow induced disturbance to PES should be less than 3 nm. Therefore, it is important to reduce the fluid flow induced vibration on the components of hard disk drives to increase their storage capacity and reliabilities. Currently there are no mature commercial toolkits which can fulfil the simulation requirements for new HDD designs. Furthermore accurate numerical simulation of three-dimensional (3D) large scale fluid-structure interaction is still a pacing item and technical barrier in computational fluid dynamics, which can only be overcome with real breakthroughs in both physical modelling and numerical simulation methods. Thus, the development of a comprehensive simulation and testing tool has significant application potentials in the HDD industry in Singapore and it also provide a basis for us to push back the frontier of basic research in related fields.

1.2.2 Offshore Wind Turbines

Offshore wind farms promise to become an important source of renewable energy in the near future: it is expected that by the end of this decade wind farms with a total capacity of thousands of megawatts will be installed in European seas [23]. This will be equivalent to several large traditional coal fired power plants. Plans are currently developed for such large-scale wind farms in Belgian, British, Dutch, German, Irish and Swedish waters and the first are now constructed in Danish seas. Apart from their advantages, such as more relaxed environmental restrictions and higher wind speeds, there is the disadvantage of high installation and maintenance costs. Weather restrictions will only allow limited access to offshore wind turbines for maintenance and repairs. Therefore robust and durable wind turbine designs that avoid excessive loads and vibrations are of utmost importance for offshore wind farms. The maximum size of

CHAPTER 1 – INTRODUCTION

wind turbine blades has been continuously increasing, from 15 meters diameter in 1985 up to the present size of 120m, and will continue to increase even further. Recently, the size of wind turbines has moved beyond a certain threshold where new issues have surfaced, with multi-stall operation and edgewise blade vibrations being the most important. An example is the structural damage in the form of serious longitudinal cracks observed on the trailing edge of the Glassfiber blades installed on wind turbines around the world [24]. Severe edgewise vibrations resulting from combined flap/lead-lag aero-elastic instability were the cause of this damage. In order to support the design of even larger new generation turbines a quantum leap in the understanding of the underlying unsteady flow and fluid-structure interaction mechanisms is also required. As wind tunnel experiments are costly, experimental work in this area has been minimal.

1.2.3 Summary

In summary, Continuous increase in rotating velocity of hard disk drives (HDDs) and wind turbines (WTs), as well as increase in sizes in WTs, requires a breakthrough in the understanding and simulation of unsteady flow and fluid-structure interactions (FSI) between their rotating parts and the surrounding air. Numerical simulations are expected to offer valuable insight into the physics of these problems. With the innovative simulation package to be developed within this project, 3D unsteady flow and FSI phenomena can be simulated efficiently, at the same time systematically providing estimates for uncertainties in computed loads. This package can be used to obtain greater insight into the physical phenomena that play a crucial role. In addition, comparing our Navier-Stokes results against the results of simpler models, their

CHAPTER 1 – INTRODUCTION

capabilities and limitations can be further assessed. Furthermore, our package can be used to construct a computational aero-elastic database for benchmarking engineering-type aero-elastic models when experimental data are not available to validate and improve simple design codes. Finally, with our package new technologies, such as smart rotor control and smart blades for the offshore wind turbine can be extensively tested.

1.3 OBJECTIVE AND SCOPE OF RESEARCH

This work concerns the development of a high performance computational package for numerical simulation of compressible unsteady flow interacting with moving or deforming structures. The objective of this research is to extend the three-dimensional finite volume (FV) Navier-Stokes solvers on unstructured tetrahedral grid [25] with a high-resolution method. The end products will include a preconditioned parallel-multigrid solver, a novel FV solver for computational structural dynamics, an efficient immersed membrane method (IMM) for the numerical study of compressible unsteady flow interacting with immersed dynamical moving or/and deforming solid bodies. In this work, all the developed numerical methods and algorithms are validated by comparing the results obtained with a wide range of well-documented published results as well as experimental measurements wherever available. This is to demonstrate the accuracy, effectiveness and robustness of the proposed method in order to testify its potential application fields including a variety of turbulent flows with complex geometries and moving/deforming immersed solid bodies.

CHAPTER 1 – INTRODUCTION

1.4 THESIS STRUCTURE

The thesis is structured as follows. Chapter 2 reviews some of the work relevant to the current study contributed by researchers of this field. Chapter 3 describes the mathematical models for preconditioned unsteady compressible Navier-Stokes equations and the numerical approaches employed in the solver. Chapter 4 describes mathematical formulation of the finite volume solver for the computational structural dynamics. Chapter 5 describes the implementation details of the immersed membrane method and how it can be used for coupling the two solvers. Chapter 6 describes the multigrid method and the parallel computing. The contents of this chapter include the algorithms for inter-connectivity relationship between meshes, the transfer operators for the multigrid method, the domain decomposition, data decomposition and the algorithms for identifying *ghost* nodes and overlapping elements. Chapter 7 describes the basic concept of Large-Eddy simulation and the mathematical formulations. Chapter 8 presents results from the validation of the developed numerical schemes and algorithms against well-documented published results. The computed results for various flow problems are showcased and discussed. Besides, the results and discussion for the computation of steady and unsteady flow between two co-rotating disks in fixed cylindrical enclosure, as well as the computation of aeroelastic flutter of an ONERA M6 wing are presented in this chapter. Finally, Chapter 9 draws conclusions based on the work performed and the results obtained. This chapter also provides suggestions for possible future work. Results from these chapters have also been published or submitted for publication (i.e. Refs. [141], [142], [143], [144]).

CHAPTER TWO

LITERATURE REVIEW

In this chapter, an in-depth review of the various numerical techniques leading to the present study on moving boundary and fluid-structure interaction (FSI) are conducted. These techniques include the serial multigrid (MG) method, both parallel single grid (SG) and MG computations, low speed preconditioning method, finite volume (FV)-based computational structural dynamics (CSD) techniques, moving boundary techniques, and lastly, FSI algorithms.

CHAPTER 2 – LITERATURE REVIEW

2.1 REVIEWS ON COMPUTATIONAL FLUID DYNAMICS

Computational Fluid Dynamics (CFD) has become an indispensable part of engineering research and design. In this section, various numerical techniques leading to the present study on moving boundary and FSI are reviewed. These techniques include the serial/parallel MG method, parallel computation, low Mach number preconditioning method, FV scheme based CSD, moving boundary and the FSI.

2.1.1 The Multigrid Method

Beside the use of parallel computers which allow numerical simulation to reduce computing time by increasing the number of processors, MG method has been established as a powerful tool for accelerating the numerical convergence and, thus also reducing the computing time. Fedorenko [26] was the first researcher to put forward the idea of MG method for elliptic equations in 1964 and the full potential of it was put forward by Brandt [27] in 1972. Jameson [28] applied the MG method to the Euler equations by solving the equations on the fine grid and then progressively transfers these flow field variables and residuals to a series of coarser grids.

It is worth mentioning that most of the developments and applications of MG method are based on structured grids. On the other hand, few studies in the literature have dealt with the development of the MG method in conjunction with the low speed preconditioning method.

CHAPTER 2 – LITERATURE REVIEW

2.1.1.1 Serial Multigrid Method

Farmer *et al.* [29] have developed a fast MG method for solving the incompressible Euler equations in conjunction with the artificial compressibility method (ACM) and applied it to free surface flows. They reported that 400 W-MG cycles were required to achieve convergence for the inviscid flow around a ship hull with free surface effects. Sheng *et al.* [30] have developed a MG algorithm for 3D incompressible turbulent flows in conjunction with the ACM and Newton relaxation methods. They proposed two different methods for constructing the coarse grid operator and investigated the influence of implicit correction smoothing on increasing the stability of the scheme. They reported fast convergence rates for the case of external flows, but deteriorate in the case of complex internal flows. Drikakis *et al.* [31] have implemented a nonlinear MG method in conjunction with ACM and with a third-order upwind characteristics-based scheme, and the fourth-order Runge-Kutta scheme. The MG method is based on the full multigrid (FMG)-full approximation storage (FAS) method. Based on the above-mentioned implementation, they further developed and investigated an adaptive-smoothing (AS) procedure to exploit the non-uniform convergence behavior of the numerical solution during iterations [32]. Iwamura *et al.* [33] have presented a robust and efficient algebraic multigrid (AMG) preconditioned conjugate gradient solver for systems of linear equations arising from the finite element discretization of a scalar elliptic partial differential equation of second order on unstructured tetrahedral meshes.

Tai and Zhao [34] have presented a two-dimensional unstructured non-nested MG method for efficient simulation of unsteady incompressible Navier-Stokes flows. The Navier-Stokes solver is based on the ACM and a higher-order characteristics-based finite-volume scheme on unstructured grids. Unsteady flow is calculated with a matrix-

CHAPTER 2 – LITERATURE REVIEW

free implicit dual time stepping scheme. Li Yuan [35] presented a common full approximation storage (FAS) MG algorithm implemented in conjunction with three different implicit schemes, which include a modified point Gauss relaxation, a standard Gauss-Seidel line relaxation, and the Beam-Warming alternating direction implicit (ADI) scheme. The flow solver used in his study is based on ACM and uses a third-order upwind difference for the convective terms and a second-order central difference for the viscous terms. The results of steady-state flow computations show that all the implicit MG schemes yield more than 50% computational time saving over their single grid counterparts. However, in unsteady flow computations, the computational time saving of the MG scheme is less than that in steady-state cases. Brandt *et al.* [36] presented recent advances in achieving textbook MG efficiency for fluid simulations. Textbook MG efficiency is defined as attaining the solution to the governing system of equations in a computational work, which is a small multiple of the operation counts associated with discretizing the system. Their code is based on the spatially factored scheme of Beam and Warming and uses MG methods to accelerate convergence to steady state. In their approach, they use a full multigrid (FMG) algorithm proceeding from the coarsest grid to finer grids.

Dennis Jespersen *et al.* [37] presented the development and testing of an unstructured “full multigrid” (FMG) algorithm for steady-state aerodynamic flows. In Dennis’s solver *OVERFLOW*, the compressible Navier-Stokes equations are discretized by a finite-volume technique on meshes of mixed element types, which may include tetrahedra, pyramids, prisms and hexahedra. The governing equations are discretized using a central difference finite-volume technique and three types of artificial dissipation are available. They are scalar dissipation, Roe third-order upwind-biased

CHAPTER 2 – LITERATURE REVIEW

differencing algorithm and the matrix artificial dissipation respectively. The first one is a basic dissipation model, which can produce undesirable velocity overshoots in boundary layer profiles and poor capturing of weak shocks; The Roe scheme usually produces better quality boundary layer profiles, but it is computationally expensive and its convergence rate is typically slow in *OVERFLOW* for unknown reasons; The matrix artificial dissipation is intended to give solutions which are of the same quality as Roe upwind solutions, but at a relatively lower computational expense, both in terms of work per step and in terms of number of steps to convergence. One important point of their implementation of multigrid in *OVERFLOW* is the sweeping method. There one begins on an initial coarse level and works up to the finest level, using the multigrid algorithm at any intermediate coarse levels. Their algorithm takes the coarse level as a subset of the fine level and takes the initial guess on the coarse level to adjust the restriction of the initial solution on the fine level to the coarse level. The other major point of their multigrid implementation is its interaction with the chimera algorithm. To ease the burden on the grid generation process and to separate the grid manipulations from the flow solver, a coarse level in one zone has no connection with another coarse level in another zone.

2.1.1.2 Parallel Multigrid Method

Mavriplis [38, 39] developed a parallel-unstructured agglomeration MG algorithm for steady-state aerodynamic flows. In Mavriplis's solver, the Reynolds averaged Navier-Stokes equations are discretized by a finite-volume technique on meshes of mixed element types, which may include tetrahedral, pyramids, prisms and hexahedra. The governing equations are discretized using a central difference finite-volume technique with added matrix-based artificial dissipation. The matrix dissipation approximates a

CHAPTER 2 – LITERATURE REVIEW

Roe Riemann-solver based upwind scheme [40], but relies on a bi-harmonic operator to achieve second-order accuracy, rather than on a gradient-based extrapolation strategy [41]. The viscous terms are discretized with second-order accuracy by the finite-difference approximation. The agglomeration MG strategy uses a graph algorithm to construct the coarse MG levels from the given fine grid, similar to an algebraic MG approach, but operates directly on the non-linear system using the Full Approximation Scheme (FAS) approach. The existing unstructured MG solver has been extended to support both cache-based and vector architectures as well as multi-level parallelism. The original Message-Passing Interface (MPI)-based parallel implementation has been extended to a two-level parallelization strategy, which employs MPI to communicate between groups of partitioned sub-domains, and OpenMP to perform vector-processing within each MPI process. The scalability and convergence rate of the MG algorithm are examined on the SGI Origin 2000 and the Cray T3E. For medium size problems involving several millions grid points, a slight drop-off in parallel efficiency is observed in his work for the MG V- and W-cycles, using up to 128 processors on the SGI Origin 2000 and up to 512 processors on the Cray T3E. For a large problem using 25 million grid points, good scalability is observed for the MG algorithm using up to 1450 processors on a Cray T3E, even when the coarsest grid level contains fewer points than the total number of processors.

Llorente *et al.* [42] presented, evaluated and analyzed an alternative highly parallel multigrid method for three-dimensional convection-dominated problems. The MG method employed semi-coarsening together with a well-balanced correction of discrete operators to maintain the same cross-characteristic interactions (CCI) on all the grids. The relaxation scheme employed in their algorithm is a four-color plane-implicit

CHAPTER 2 – LITERATURE REVIEW

scheme enabling a very efficient parallel implementation, which appears to be a good trade-off between parallel and convergence properties. The resulting MG solver exhibits a fast grid-independent convergence rate for solving the convection-diffusion operator on cell-centered grids with stretching. The load imbalance below the critical level is the main source of inefficiency in its parallel implementation. The parallel implementation of the MG algorithm was based on MPI. There are two different strategies proposed by them for the parallel implementation of the MG method, which are domain decomposition combined with multigrid (DD-MG) and grid partitioning (MG-DD). The DD-MG approach applies domain decomposition on the finest grid, and using a MG method inside each block. These kind of methods are often considered with the finite elements discretization since they are easier to implement and can be directly applied to general multi-block grids. From an architectural point of view, this method also implies fewer communications since they are only required on the finest grid. However, they lead to algorithms which are numerically different to the sequential version and have a negative impact on the convergence rate as reported by Smith et al. in [43]. In their research, they adopted the MG-DD technique. In this approach, MG is used to solve the problem in the whole grid, i.e. domain decomposition is applied on each level. This technique required more communication overheads since data exchange is required on every level. However, unlike DD-MG approach, MG-DD retains the convergence rate of the sequential algorithm [44]. The parallel MG algorithm is studied on the SGI Origin 2000 and the Cray T3E machine and satisfactory efficiencies (higher than 0.8) are obtained for up to 32 processors on a $128 \times 128 \times 128$ uniform grid.

CHAPTER 2 – LITERATURE REVIEW

2.1.2 Parallel Computation

Parallelization has been identified as the basis for future high performance computing. Computing speed and memory capacity have increased exponentially over the last several decades, which led naturally to work on parallel computational fluid dynamics.

Many techniques and tools for implementing parallel numerical methods are available to researchers. These include compiler directives (OpenMP [45]), message passing (MPI [46]), parallel languages (High Performance Fortran, HPF [47]), and parallel libraries (PETSc [48]). On machines with shared memory architectures, the usual parallelism model is Multiple Instruction Multiple Data (MIMD). The MIMD model is usually implemented at the loop level with compiler directives, such as the OpenMP set of directives. Although the MIMD model is easy to implement, scalable performance is usually limited to tens of processors [49]. Success with more processors is more difficult to produce [50]. On machines with distributed memory architectures, the popular parallelism model is Single Program Multiple Data (SPMD). The SPMD model is usually implemented by message passing, using libraries of message passing tools such as the MPI library [46] or PVM library [51]. The SPMD model has been shown to scale to hundreds and even to thousands of processors [49]. One important reason for this scalability is that good data locality can be maintained with the SPMD model. However, message-passing implementations generally require extensive changes to a baseline code because the domain decomposition and message passing must be explicitly programmed. It requires the programmer to create the data structures for communication and explicitly code the transfer of data between processors. This is a large and difficult hurdle most of the programmers encounter.

CHAPTER 2 – LITERATURE REVIEW

McManus [52] examined the method of parallelization by geometric domain decomposition using the SPMD programming paradigm with explicit message passing in his Ph.D. thesis. In the thesis some techniques and strategies to parallelize unstructured mesh codes were developed and tested using an integrated CFD and solid mechanics code to explore the computational advantages offered by distributed memory parallel processors. The performance of the parallel code was examined and some work was carried out to achieve automation of parallelization process. Yao *et al.* [53] presented the development and validation of the unsteady, three-dimensional, multiblock, parallel turbo machinery flow solver, TFLO. The Unsteady Reynolds Averaged Navier-Stokes (Unsteady RANS) equations are solved using a cell-centered discretization on arbitrary multiblock meshes. The solution procedure is based on efficient explicit Runge-Kutta methods with several convergence acceleration techniques such as multigrid, residual averaging, and local time stepping. The solver is parallelized using domain decomposition, an SPMD strategy, and the MPI Standard. The dual-time stepping technique is applied to advance unsteady computations in time. TFLO is routinely run on the CRAY T3E, SGI Origin 2000 and IBM-SP systems. Das *et al.* [54] implemented a three-dimensional unstructured grid Euler solver on massively parallel distributed-memory computer architectures. Their goal is to minimize solution time by achieving high computational rates with a numerically efficient algorithm. An unstructured MG algorithm with an edge-based data structure has been adopted, and a number of optimizations have been devised and implemented to accelerate the parallel computational rates. Large practical unstructured grid problems are solved on the Intel iPSC/860 hypercube and Intel Touchstone Delta machine. The quantitative effects of the various optimizations are demonstrated and the combined effect of these optimizations leads to roughly a factor of 3-performance improvement. The overall

CHAPTER 2 – LITERATURE REVIEW

solution efficiency is compared with that obtained on the Cray Y-MP vector supercomputer.

Professor Tezduyar's team in University of Minnesota have been using parallel computing extensively. With the support of powerful hardware facilities, most of their work was concerned with complex fluid-structure interaction problems in which mesh adaptation or regeneration took much of computing time. In 1995, they calculated 3D flow past a rectangular wing at Reynolds number 1,000, 2,500 and 10,000,000 in which the finite element mesh consisted of 753,168 nodes and 727,552 hexahedral elements, and flow past a flapping wing in which the finite element mesh consisted of 145,402 nodes and 137,280 elements [55]. These calculations were executed on a 512-node CM-5 supercomputer and good efficiency was achieved. In 1997 they calculated 3D flow past a sphere at Reynolds number of 400 [56]. The computational mesh consisted of 43,282 nodes and 258,569 elements and a mesh partition program, METIS, was used for domain decomposition. The calculation was carried on a 256-PN CM-5 supercomputer. In 1997 they calculated flow past an automobile and 5 spheres falling in a liquid-filled tube on Thinking Machines CM-5 and Cray T3D supercomputer [57]. Another problem of parallel finite element simulation of large ram-air parachutes was also carried out on the CM-5 supercomputer with a data-parallel programming paradigm and on a Cray T3D supercomputer with a message-passing paradigm [58]. The most difficult of their work might be the calculation of 1000 spheres falling in a liquid-filled tube at Reynolds number 10 [59]. The mesh size during the simulation reached 1.2 million tetrahedral elements in this calculation. The simulation involved calculation of spheres colliding with each other, forming groups, bumping on the tube wall, as well as the fluid-sphere interaction, mesh regeneration and mesh partition. In the simulation, the mesh partition,

CHAPTER 2 – LITERATURE REVIEW

flow computations and mesh movements were performed on a 512-node Thinking Machines CM-5, automatic mesh regeneration and projection of the solution were accomplished on a 2-processor SGI ONXY2 workstation.

2.1.3 Low Mach-Number Preconditioning

Time-marching algorithms are widely used for the computation of compressible flows. A major advantage of these techniques is that they apply to both inviscid and viscous flows and can be used in conjunction with virtually any spatial discretization in all Reynolds number regimes. In the past two or three decades, time-marching schemes have been widely accepted and applied as the method of choice for transonic, supersonic, and hypersonic flows. In the low subsonic Mach number regime, time-marching algorithms do not fare as well. When the magnitude of the flow velocity becomes small in comparison with the acoustic speed, the convective terms of the time-dependent equations become stiff and time-marching methods converge very slowly. These convergence difficulties are then further exacerbated by the magnitude of the diffusion terms, and corrective action can differ substantially when the diffusion terms are missing (the Euler equations), small (high Reynolds numbers) or dominant (low Reynolds numbers). Many works on preconditioning have been done by many researchers, for example, Turkel [60-64], van Leer *et al.* [65-67], Choi [68], Merkle [68-71], Venkateswaran [69-71], Weiss and Smith [72], etc. They have shown that the preconditioning method can eliminate low Mach number convergence difficulties in all Reynolds number regimes.

CHAPTER 2 – LITERATURE REVIEW

Merkle and Choi [68] were the first to show how to re-scale the convective propagation speeds to the diffusion rates. They stated that the reason for the slow convergence rates while the Reynolds numbers is low (below $Re = 50$) can be understood by considering the viscous time step parameter (they called it von Neumann number, VNN), which becomes important at low Reynolds numbers. Control of the CFL number alone at low Reynolds number makes the VNN number so large that the approximate factorization error in the diffusive terms slows convergence. For efficient convergence, one should control both the CFL number and the VNN number simultaneously. For this reason, they introduced a scaling parameter in the definition of the preconditioning matrix.

Radespiel [62] presented a generalization of the preconditioners given by Turkel and also Choi and Merkle. They pointed out that for a central difference scheme it is necessary to add artificial dissipative terms to the central difference approximation of the spatial derivatives. By using conservative or non-conservative variables, they introduced two different preconditioned artificial dissipation terms. Also the characteristic variables needed to compute flow conditions at inflow or outflow boundaries are derived. In their solvers, the boundary conditions in the far field are based on characteristic variables, even for viscous flow. Thus at inflow the incoming variables corresponding to positive eigenvalues are specified while the outgoing variables corresponding to negative eigenvalues are extrapolated. Although the preconditioning does not change the signs of the eigenvalues, it does change the characteristics of the system. Hence, it is necessary to also modify the boundary conditions for the preconditioned system.

CHAPTER 2 – LITERATURE REVIEW

Dennis Jespersen et al. [37] also successfully developed a novel preconditioning method into their unstructured finite-volume solver *OVERFLOW*. Their derivation starting point is the quasi-linear form of the flow equation in curvilinear coordinates and conservative form:

$$\frac{\partial W_c}{\partial t} + A \frac{\partial W_c}{\partial \xi} + B \frac{\partial W_c}{\partial \eta} + C \frac{\partial W_c}{\partial \zeta} = 0$$

Where $W_c = \{\rho, \rho u, \rho v, \rho w, \rho e_t\}$, A , B , and C are the convection Jacobians, transforming to primitive variables $W_p = \{P, u, v, w, T\}$ becomes:

$$M \frac{\partial W_p}{\partial t} + AM \frac{\partial W_p}{\partial \xi} + BM \frac{\partial W_p}{\partial \eta} + CM \frac{\partial W_p}{\partial \zeta} = 0$$

Where $M = \frac{\partial W_c}{\partial W_p}$, Their preconditioning step consists of replacing M by another matrix Γ_1 . By using this treatment, they changed the eigensystem and equilibrated the disparity of the original eigenvalues thus accelerated the convergence rates. They also pointed out the aspects of the scheme which are modified by the preconditioning. They used a MUSCL interpolation scheme (Koren limiter) to produce left and right flow field states, which are then employed in the definition of the Roe flux differencing scheme where the numerical flux at the center between two nodal points has also been modified. The modification lies in using the preconditioned eigenvalues and eigenvectors in the definition of the Roe's average matrix.

2.1.4 Finite Volume (FV)-Based Computational Structural Dynamics

Over the last five decades a wide variety of numerical methods have been proposed for the numerical solution of partial differential equations. Among them the finite element (FE) method has firmly established itself as the standard approach for problems in computational solid dynamics (CSD), especially with regard to deformation problems involving non-linear material analysis [73, 74]. As a contemporary, the FV method developed from early finite difference techniques and has similarly established itself within the field of CFD [75, 76]. Both classes of methods integrate governing equations over pre-defined control volumes [75, 77], which are associated with the elements making up the domain of interest. Furthermore, both approaches can be classified as weighted residual methods where they differ in the weighting functions adopted [78].

In many engineering applications, there is an emerging need to model multiphysics problems in a coupled manner. In principle, because of their local conservation properties the FV methods should be in a good position to solve such problems effectively. Over the last decade a number of researchers have applied FV methods to problems in CSD [79] and it is now possible to classify these methods into two approaches, cell-centered [80-84] and vertex-based ones [78, 79, 85, 86]. The first approach is based upon traditional FV methods [75] that have been widely applied in the context of CFD [76]. Subsequently, such techniques have been applied to CSD problems using structured [80, 81] and unstructured meshes [82-84, 87]. With regard to these techniques, it should be noted that when solid bodies undergo deformation the application of mechanical boundary conditions is the most effective if they can be imposed directly on the physical boundary. Obviously, the cell-centred approximation may have difficulty in prescribing the boundary conditions, when complex geometries

CHAPTER 2 – LITERATURE REVIEW

are considered and where displacements at the boundary are not prescribed directly and in a straight forward manner.

The second approach is based on some basic ideas of traditional FE methods, which employs shape functions to describe the variation of an independent variable, such as displacement, over an element and is therefore well suited to complex geometries [78, 79, 85, 86]. The approach can be roughly classified as a cell-vertex FV method [76, 78].

Both the above FV approaches apply strict conservation laws over a control volume and have demonstrated equivalency to traditional FE methods with regard to accuracy and efficiency [79, 82]. Some researchers have attributed this to the local conservation of independent variables as enforced by the control-volume methods employed [85, 86] and others to the enforced continuity of the derivatives of the independent variables across cell boundaries [82].

2.1.5 Moving Boundary

Moving boundary problems have important engineering applications in a variety of physical and engineering areas such as solid and fluid dynamics, combustion, heat transfer, material sciences, etc. Currently, there are many techniques devoted to solving the moving boundary problems, but they can be classified into three major groups according to the property of computational grid used in the calculation. These three groups are: Eulerian Method, Arbitrary Lagrangian-Eulerian (ALE) Method and Hybrid Method. The following sub-sections give a review of the methods used to tackle moving boundary problems that are relevant to this research.

CHAPTER 2 – LITERATURE REVIEW

2.1.5.1 Eulerian Method

In the Eulerian method, the mesh is stationary and does not move or deform. Therefore the fluid motion can be conveniently described with respect to this Eulerian reference frame. The Eulerian method includes the Fictitious Domain Method (FDM), Immersed Boundary Method (IBM), Ghost Cell Method (GCM), Ghost Fluid Method (GFM) and Volume of Fluid (VOF) method. The VOF method is mainly used in the two-phase flow computation, which it is not reviewed here.

2.1.5.1.1 Fictitious Domain Method

Glowinski et al. [88, 89] formulated a fictitious domain method for the numerical solutions of three-dimensional elliptic problems with Dirichlet boundary conditions and also of the Navier-Stokes equations modeling incompressible viscous flow. The solution method combines the finite element approximations, time discretization by operator splitting and conjugates gradient algorithms for the solution of the linearly constrained quadratic minimization problems. The main advantage of this method (sometimes called domain imbedding methods) is that they use fixed meshes and the boundary conditions of the moving bodies are imposed through a Lagrangian multiplier, therefore allowing the use of fast solvers. Thus, the time-consuming construction of a boundary-fitted mesh for each different position of moving rigid bodies can be avoided. And the method does not require a strong coupling between the actual boundary discretization and the grid

CHAPTER 2 – LITERATURE REVIEW

used in the auxiliary domain. However, this apparent simplification is accompanied by some complications such as the need to manage data structures pertaining to the actual geometry, an essential operation which for 3D problems is in general far from being trivial. They also require the interface conditions to be distributed to both the structural and fluid nodes for iterations in their solution processes, making them less efficient. They employed this method to simulate Stokes flow past moving rigid bodies [88], steady 3D external Stokes flow and vortex shedding behind a cylinder for unsteady incompressible viscous 2D flow [89].

2.1.5.1.2 Immersed Boundary Method

Peskin *et al.* [13, 14] proposed a technique known as the immersed boundary method to tackle the moving boundary to simulate the natural heart and heart valves combining Eulerian flow equations and a Lagrangian description of heart walls and valves. The philosophy of the immersed boundary method is to treat the elastic material as a part of the fluid in which additional forces (arising from the elastic stresses) are applied. The fluid equations are solved on a regular cubic lattice, the structure of which is not modified in any way by the presence of the immersed elastic bodies and the geometry of which may be quite complicated. The elastic material is tracked in Lagrangian fashion, by following a collection of representative material points. The spatial configuration of these points is used to compute elastic forces, which are applied to the nearby lattice points of the fluid. The fluid velocity is updated under the influence of these forces, and the new velocity is then interpolated at the elastic material points, which are moved at the interpolated velocity to complete the time step. The immersed boundary method has been applied to a wide range of problems, mostly in biofluid dynamics, including blood

CHAPTER 2 – LITERATURE REVIEW

flow in the heart [14], platelet aggregation during blood clotting [15], fluid dynamics of the inner ear [16], flow in arterioles [17] and simulating the motion of flexible pulp fibers [18]. Later Peskin and his co-investigators developed the method further and published an adaptive version [19] and a second-order-accurate [20] one to enhance its accuracy. Generally speaking, the method is applicable to any problem in which a fluid interacts with an elastic material. The elastic material may be an active one, as in the case of cardiac muscle. A limitation of the immersed boundary method is that it is just suitable for considering the moving boundary as one-dimensional curves, thus it could only be used when the thickness of the moving object can be ignored. It cannot consider inertial effect of the structure. Furthermore, it is not suitable for high-Reynolds-number flows, which have sharp differences in flow properties across the boundary and thin boundary layers.

2.1.5.1.3 Ghost Cell Finite Difference Method

To retain a sharp fluid-structure interface and achieve a higher local accuracy near the immersed moving boundary, the computational stencil near the interface is modified to directly impose the boundary conditions on the immersed boundary. Majumdar *et al.* [21] proposed a ghost fluid cell method to treat the immersed boundary for structured grids. Ghost cells are defined as cells in the solid domain that have at least one neighbour in the fluid domain. For each ghost cell, an interpolation scheme that implicitly incorporates the boundary condition on the immersed boundary is then devised. This method has been used for simulating a wide variety of flows including compressible flow past a circular cylinder and an airfoil [90] at Reynolds numbers up to $O(10^5)$, aquatic propulsion [91], flow through a rib-toughened serpentine passage [92]

CHAPTER 2 – LITERATURE REVIEW

and turbulent flow past a road vehicle [93]. It uses wall normal direction and its associate intersection points to perform interpolations to calculate flow properties of the ghost nodes, making it very complicated and difficult to implement in unstructured grid solvers.

2.1.5.1.4 Ghost Fluid Method

Fedkiw [94] proposed another ghost fluid method to tackle the moving boundary problems. In his method, the fluid-structure interfaces are treated in Eulerian scheme that maintains a Heaviside profile of the density with no numerical smearing. A level set function is used to define two separate domains for the two separate fluids. Then a ghost cell at every point in the computational domain is imposed. In this way, each grid point contains the mass, momentum and energy for the real fluid that exists at that point (according to the sign of the level set function) and a ghost mass, momentum and energy for the other fluid that dose not really exist at the point (it is on the other side of the interface). With the help of the ghost fluid, the integrity of the background Eulerian mesh can be maintained even though the fluid-structure interface cuts part of the mesh. So that the interface boundary conditions or jump conditions are properly captured. This method is designed to treat two-phase flow problems on Cartesian grids only.

The methods reviewed in the previous two sections introduce the boundary conditions directly into the discrete equations. They enable a sharp representation of the immersed boundary and this is desirable, especially for high-Reynolds-number flows. Furthermore, the direct enforcement of boundary conditions does not introduce any extra stability constraints in the representation of the solid bodies. Finally, this approach

CHAPTER 2 – LITERATURE REVIEW

decouples the equations for the fluid nodes from those for the nodes in the solid, thereby obviating the solutions of the governing equations for the solid grid nodes. However, one disadvantage of this approach is that inclusion of boundary motion can be more difficult than the FD and IB methods. Furthermore, the ghost fluid cell finite difference method and ghost fluid method are limited to structured grids so far, hence their applications to complex geometries are also limited.

2.1.5.2 Arbitrary Lagrangian-Eulerian (ALE) Method

The Arbitrary Lagrangian-Eulerian (ALE) method, first proposed by Donea *et al.* [1], involves a continuous adaptation of the mesh without modifying the mesh topology in solving the fluid-structure interaction and moving boundary problem. The mathematical formulation of the equation of motion for a fluid is most conveniently described with respect to an Eulerian reference frame. However, this is incompatible with the Lagrangian formulation, which is more appropriate to describe a solid phase. The ALE methodology combines the best of features of these two different formulations. The essence of the ALE is that the mesh motion can be chosen arbitrarily [2]. It includes three phases: an explicit Lagrangian update, an implicit iteration of the momentum equation with the equation of state, and a rezone/map phase [3]. The Lagrangian phase provides an explicit update of the equation of motion. When the material velocities are much smaller than the fluid sound speed, the optional implicit phase allows sound waves to move many cells per cycle, thereby significantly improving computational efficiency. Finally, a rezone algorithm may prescribe mesh velocities relative to the fluid, thus necessitating re-map phase in which the solution from the end of phase two is mapped onto the new mesh. A particularly important property of ALE is that it provides

CHAPTER 2 – LITERATURE REVIEW

a means of minimizing advection errors [4]. ALE techniques have gained popularity for transient, high speed, small deformation problems [5]. They are most commonly used in the aerospace field; where solid structures are subjected to complex airflows and flutter, e.g. see [6]. In heart valves, however, due to large movement/displacement of a thin leaflet within the computational domain, it is generally difficult to adapt the mesh in such a way that a proper mesh quality is maintained without changing the mesh topology. Alternatively, re-meshing can be performed, if the mesh quality is degenerated too much. This technique was used by Horsten [7] to model the aortic valve in 2D. Re-meshing, however, not only introduces artificial diffusivity, it may also be difficult to perform with sufficient robustness and accuracy for three-dimensional problems. Thus ALE method includes Mesh Regeneration Method and Deforming Mesh Method.

2.1.5.2.1 Deforming Mesh Method

The deforming mesh method is mainly used in systems with small displacements or small deformations. When the boundary of the flow field has certain small changes, the grid points lying on the moving boundary move with the boundary. Co-ordinates of internal grid nodes near the moving region are also adjusted to smooth the movement from the moving boundary to the internal region of the flow field, based on the distance from the moving boundary to the node to be moved. The computational expense of this method is relatively light. This method has been successfully used in CFD studies of aero-elasticity and flow in elastic tubes.

CHAPTER 2 – LITERATURE REVIEW

Heil [96] studied stokes flow in an elastic tube with deforming mesh method. In his calculation, a structured fluid mesh was generated in an un-deformed cylindrical tube. In this reference configuration, the fluid nodes on the tube wall coincide with certain material points in the tube. When the tube wall deformed, the fluid mesh was adjusted by keeping the fluid mesh ‘attached’ to the same material points in the tube. The interior mesh points changed their radial displacements with each other to suit the change in the boundary. In this implementation, the coordinates of grid points were changed, but their relative positions were kept. Hassan *et al.* [95] calculated pseudo-transient motion of a missile mounted initially underneath a wing with the deforming mesh method and the local mesh regeneration. Deforming mesh method was used when small displacement assumption was satisfied. In their deforming mesh method, the mesh was modeled as a spring network, with nodal points on the outer boundary held fixed while the instantaneous locations of points on the moving boundary were specified. The locations of the inner nodes of the mesh were then determined by solving static equilibrium equations that resulted from summing the forces in the spring system at each node. The mesh-smoothing algorithm thus altered the positions of the interior nodes without changing the topology of the mesh.

Farhat *et al.* [97] calculated a flat panel with infinite aspect ratio in supersonic airstreams. Arbitrary Lagrangian/Eulerian (ALE) method was used in their computation for moving boundary. In their work, differential equation was used to calculate the new positions of grid points, and the mathematical description of moving boundary was used as boundary condition to this differential equation of mesh movement. In their work the ALE method was just the deforming mesh method. Schulz *et al.* [98] calculated an elastically mounted cylinder vibrating freely in the flow field with deformable hybrid

CHAPTER 2 – LITERATURE REVIEW

grids. In their work, dynamic deformation of the hybrid mesh was achieved by matching the computed structural displacement with the mesh displacements immediately surrounding the body. The displacement of the remaining mesh was then linearly reduced based on the distance away from the structure such that the mesh near all far field boundaries remained undisturbed. In their recent work [206, 207], F. Liu *et al.* developed a parallel integrated CFD–CSD simulation for the simulation and prediction of flutter of aeroelastic system. This program consists of a three-dimensional, parallel, multiblock, multigrid, unsteady Navier–Stokes solver, a parallel dynamic grid deformation code, a CSD solver strongly coupled with the flow solver using dual time stepping, and a spline matrix method for interfacing the CFD and CSD grids and aerodynamic loading variables.

2.1.5.2.2 Mesh Regeneration Method

For relatively large deformation or displacement cases, the computational mesh needs to be regenerated. With different complexity of the flow field, mesh regeneration might be carried out locally or globally. When dealing with large deformation, this method could provide better mesh for the moving boundary calculation than any other methods, but its computational expense is out of the reach of most researchers.

With the help of a 512-node Thinking Machines CM-5, Tezduyar *et al.* simulated multiple spheres falling in a liquid-filled tube [57, 59], airflow past the parachute [57, 58], and flow past a flapping wing [55]. In fact their method was a combination of deforming mesh and mesh regeneration. In their work, the deforming mesh method was first used when boundary movement was still relatively small. When the mesh became

CHAPTER 2 – LITERATURE REVIEW

overly skewed, grid regeneration was deployed and the flow variables were interpolated from the old mesh to the new one. In some examples, such as the simulation of 1000 spheres falling in a liquid-filled tube [59], the total mesh was regenerated in a few steps. This may be the most expensive work done in moving boundary cases. Hassan *et al.* [95] calculated transient motion of a 3D body subjected to large displacement and pseudo transient motion of a missile configuration mounted initially underneath a wing using mesh regeneration combined with deforming mesh. They also used deforming mesh when the displacement was relatively small, and regenerated the mesh when it was severely skewed. As their configuration was relatively simple, they only regenerated the mesh in certain regions of the flow field.

2.1.5.3 Hybrid Method

Hybrid method combines the Lagrangian and Eulerian reference. The Chimera mesh method is a hybrid method.

2.1.5.3.1 Chimera Mesh Method

When the boundary moves without changing its geometry, Chimera mesh might be a suitable method for large displacement cases. In this method, there are two sets of meshes, an Eulerian mesh and a Lagrangian one. The Lagrangian mesh surrounds the moving boundary and moves with it. The Eulerian background mesh occupies other regions of the flow field. The two meshes have an overlapping region or a transition region. When the moving boundary moves to a new position, the transition region is regenerated, or the overlapping region is redefined. Flow variables on Eulerian mesh are

CHAPTER 2 – LITERATURE REVIEW

computed in an Eulerian manner, and those on the Lagrangian mesh are evaluated in a Lagrangian manner.

Kiris *et al.* [99] calculated blood flow through artificial heart devices with Chimera mesh method. In their work, structured Chimera mesh was used around the heart valves in mitral and aorta positions. The Chimera mesh and the background mesh exchanged flow information through an overlapping zone. Comparison between numerical results and experimental measurements showed that they correlated well. For moving boundary with fixed geometry, Chimera mesh method could be a good reference. It is less computational expensive than the mesh regeneration method, and could effectively deal with large displacements. The shortcoming is the introduction of errors during value exchanging between the two meshes.

2.1.6 Fluid-Structure Interaction (FSI)

Fluid-structure interaction (FSI) is a common phenomenon in nature, such as unsteady viscous flow in elastic (collapsible) tubes [96], flapping flags [100] and others. A typical example in biomechanics is the opening and closing behavior of aortic mechanical heart valves, which is a delicate interaction between blood flow and heart-valve leaflets (see, e.g. de Hart *et al.* [101]). In FSI phenomenon, the movement of the structure totally depends on the forces, especially the fluid forces, acting on the structure. Due to the action of these forces, the structure will have some kind of motion, including rotation, deformation or translation or some combinations or even all of these changes. Due to the corresponding changes of the structure in spatial positions and geometric shapes, the flow field around the structure is also changed. Fluid forces acting

CHAPTER 2 – LITERATURE REVIEW

on the structure evolve with the changing flow field, which brings the structure to a new state of movement. In this way, fluid and structure interact with each other in a loop fashion, and the system evolves with very complex nonlinear dynamics.

In numerical modeling of FSI, the fluid phase is most conveniently described with respect to an Eulerian reference frame, while a Lagrangian formulation is more appropriate for the solid phase. However, these formulations are incompatible. Many methods have been proposed to tackle this incompatibility (see the above reviews on moving boundaries). In most of research works, the flow field and the structure are computed separately, which begins by computing the flow field with Navier-Stokes equations or Euler equations. Fluid forces acting on the structure are then available and used to calculate the response of the structure. Depending on the complexity of the problem and the geometric change of the structure, the response of the structure is calculated either by the finite element method or simply solving an ordinary differential equation. Corresponding displacements or geometric changes of the structure are then fed into the flow field, and the computational mesh of the flow field is refreshed, using one or two of the techniques discussed in the moving boundary section. The flow field is then recalculated on the updated mesh and fluid forces are then passed to the structure again for calculating the new structural dynamics. In this way the calculation is carried out for several prescribed iterations or until the desired error limit is satisfied, then the interaction between fluid and structure is deemed to be fully coupled and field variables can satisfy the governing equations of both the fluid and the structure. The whole process can then be repeated until prescribed time instant.

Peskin [102] has calculated the blood-leaflet interaction in a natural heart valve with

CHAPTER 2 – LITERATURE REVIEW

immersed boundary method. The basic procedure of the calculation is the same as described above. As this method was specially designed to calculate those moving objects whose thickness could be ignored such as the natural heart valve and flexible pulp fibers, it can only be used in certain specified situations. In recent years, Baaijens [103] has been active in studying the FSI found in aortic heart valves. He proposed a new numerical method: the fictitious domain/mortar method for studying FSI. In his work, this method is used to study the motion of flexible thin beams in Stokes and Navier-Stokes flows. Two thin beams of different thickness are studied, and the results showed that the motions of these two beams are different. This illustrated the feasibility of the fictitious domain/mortar method to investigate the impact of structural changes on FSI. The key advantage of this method is that it does not require any updating of mesh in the fluid domain. This makes the fictitious domain/mortar method attractive for 3D calculations, where a robust updating of the mesh to follow the motion of the thin-shell-like structures poses major difficulties. De Hart *et al.* [104] studied a two-dimensional FSI model of aortic valve using the fictitious domain method and validated the numerical results by LDA measurements and high-speed video recordings. Here the 2D model is analyzed based on the finite element method, in which the fluid and structure are coupled by fictitious domain. A staggered approach can be used to solve the uncoupled systems [105]. De Hart *et al.* [101] extended this work to a three-dimensional model of aortic valves.

In George Mason University, Professor Löhner's group is another active group in research of FSI problems. The general idea of FSI calculation is the same as those described earlier. Besides the FSI calculation, they also considered the thermal effect in their calculations [106]. In their work, they combined commercial codes in their calculations. For CFD, they often used FEFLO98 for calculation, and for the structure

CHAPTER 2 – LITERATURE REVIEW

they used COSMIC-NASTRAN for linear computational structure dynamics and DYNA3D for non-linear computational structure dynamics. Several examples are given in Professor Löhner's homepage [107], including generic weapon fragmentation, supersonic flow over a deforming panel. Professor Tezduyar's research team in University of Minnesota is another active group. In their recent work, they studied a few cases with deforming spatial domain/stabilized space-time (DSD/SST) finite element method [18, 58, 108-110]. In these works, the fluid domain was solved with the finite element method, and the moving structure was calculated with either the finite element method for complex cases, or with dynamic equilibrium equation for simple cases. Refreshing of computational mesh was achieved by a combination of the deforming mesh method and the mesh regeneration method.

Many other researchers also reported their work in FSI, such as Elat *et al.* [111], Farhat *et al.* [112], Freitas *et al.* [113], Liang *et al.* [114], Piperno [115], Soria *et al.* [116], and Heil [96]. Their methods are almost the same as Tezduyar's method, except for using the finite volume method for computing fluid domain, and using the deforming mesh method for mesh refreshment.

2.2 SOME REMARKS

After reviewing the work done by researchers in these fields, there are a few important points to be noted.

CHAPTER 2 – LITERATURE REVIEW

2.2.1 Serial/Parallel Multigrid Method

The most important considerations of the multigrid method are the inter-connectivity relationship between meshes and the mesh-to-mesh transfer operators. Before flow field variables and residuals are transferred from the fine grid to the coarse grid or corrections are interpolated from the coarse grid back to the fine grid, it is necessary to determine in which coarse cell each fine node is located and vice-versa. The accuracy and efficiency of transferring the solutions and residuals from the fine grid to coarse grid, or interpolate the corrections from the coarse grid to fine grid strongly depends on the type of transfer operators used. The type of cycle strategies used in the multigrid method also affects the rate of convergence.

2.2.2 Parallel Computing

The important issues of parallel computation are mesh partitioning and domain decomposition. The decomposition of a mesh into a set of S sub-domains that may be allocated to a set of P processors involves finding partitions of the mesh so that the amount of computational time on each processor is almost equal. A good mesh partition is one which divides the computational load equally amongst the sub-domains (i.e. each sub-domain having the same number of nodes) and minimizes the amount of communication required between sub-domains. The parallel-MG is also adopted for better convergence as shown by Llorente et al. [42, 44].

2.2.3 Low Mach-Number Preconditioning

The basic point of the low speed preconditioning method is to equilibrate the eigenvalues of the original NS equations at low speed incompressible limits, scaling convective and acoustic speed to the same order of magnitude. The most important step in this process is the design of a preconditioning matrix. After preconditioning, the Roe upwind-biased differencing flux computation needs to be re-formulated. Boundary conditions also have to be changed too if they are based on the characteristics of the system. When conservative variables are used, one now has to keep track of how various forcing functions in the MG method have been stored on each grid. If the forcing terms are stored after they have been preconditioned then one should not precondition them again after they are restricted to the next coarser mesh. If the unpreconditioned forcing terms have been stored then one needs to precondition the forcing terms after the restriction. Thus, physical and multigrid forcing terms may be treated differently on the coarse grids.

2.2.4 FV Based Computational Structural Dynamics

The finite volume method, which is popular with computational fluid dynamics (CFD), is now widely used in computational structural dynamics (CSD). It provides an important tool to embed the structural analysis procedures with existing finite-volume based CFD solvers. Current available FV approaches rely heavily on the proper definition of the “shape function”, which demands intensive matrix operations and is the basis of traditional finite-element method (FEM). These approaches differ with the FV method utilized in current work in that the later employs iteration and matrix-free

CHAPTER 2 – LITERATURE REVIEW

scheme. Aiming to design a highly-efficient FSI simulation package, the development of a new FV-based matrix-free CSD solver is an important part of the current study.

2.2.5 Moving Boundary and Fluid-Structure Interaction

In the area of moving boundary, the mesh undergoes a large or small displacement with the development of system dynamics. When this happens, the computational mesh needs some form of adjustment or regeneration, and thus demanding extra computational resources. In order to avoid mesh regeneration, Eulerian methods remain attractive measures for the moving boundary problems. However, the immersed boundary method and fictitious domain method smooth the boundary conditions on the fluid-structure interfaces, which are not suitable for high Reynolds numbers problems or problems with strong emphasis on maintaining a sharp fluid-structure interface. The ghost fluid cell finite difference method can track a sharp interface, but it cannot handle complex geometries due to the structured mesh used. The ghost fluid method uses a level set function, and defines two separate fluid domains (one for real fluid, another for ghost fluid). The two sets of fluid domains can only be applied to two-phase flow problems on structured meshes.

The above remarks serve as a guide for the implementations of the serial MG method, the parallel single grid and MG schemes, the FV-based CSD solver, and the immersed membrane method into the general simulation package for the numerical simulation of moving boundary and fluid-structure interaction in three dimensional compressible flows.

CHAPTER THREE

MATHEMATICAL FORMULATION FOR PRECONDITIONED COMPRESSIBLE FLOW SOLVER

This chapter describes the mathematical formulation for the three-dimensional unsteady preconditioned compressible viscous flow solver developed in this work. The finite-volume Navier-Stokes solver employs high-resolution Roe's scheme on unstructured grids. The unsteady flow is calculated with a matrix-free implicit dual time stepping scheme. A five-stage Runge-Kutta time integration algorithm is used between each physical time step to iterate the numerical solution in pseudo time until convergence is reached. The methods used to accelerate the convergence rate to steady state in pseudo time, which are local time stepping and implicit residual smoothing, are briefly described. The Characteristics-based boundary treatments are also discussed in detail.

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

3.1 BASIC GOVERNING EQUATIONS FOR THE NAVIER-STOKES FLUIDS

The 3-D unsteady Navier-Stokes equations may be written in a number of forms. One common form of these equations is as follows:

Continuity:

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} + \frac{\partial(\rho w)}{\partial z} = 0 \quad (3.1)$$

u-Momentum:

$$\frac{\partial(\rho u)}{\partial t} + \frac{\partial(\rho u^2 + p)}{\partial x} + \frac{\partial(\rho uv)}{\partial y} + \frac{\partial(\rho uw)}{\partial z} = \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} \quad (3.2)$$

v-Momentum:

$$\frac{\partial(\rho v)}{\partial t} + \frac{\partial(\rho uv)}{\partial x} + \frac{\partial(\rho v^2 + p)}{\partial y} + \frac{\partial(\rho vw)}{\partial z} = \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} \quad (3.3)$$

w-Momentum:

$$\frac{\partial(\rho w)}{\partial t} + \frac{\partial(\rho uw)}{\partial x} + \frac{\partial(\rho vw)}{\partial y} + \frac{\partial(\rho w^2 + p)}{\partial z} = \frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial \tau_{zz}}{\partial z} \quad (3.4)$$

Energy Equation:

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$\begin{aligned}
& \frac{\partial(\rho e_t)}{\partial t} + \frac{\partial(\rho e_t u)}{\partial x} + \frac{\partial(\rho e_t v)}{\partial y} + \frac{\partial(\rho e_t w)}{\partial z} = \\
& \frac{\partial\left(\kappa \frac{\partial T}{\partial x}\right)}{\partial x} + \frac{\partial\left(\kappa \frac{\partial T}{\partial y}\right)}{\partial y} + \frac{\partial\left(\kappa \frac{\partial T}{\partial z}\right)}{\partial z} - \frac{\partial(up)}{\partial x} - \frac{\partial(vp)}{\partial y} - \frac{\partial(wp)}{\partial z} + \\
& \frac{\partial(u\tau_{xx} + v\tau_{xy} + w\tau_{xz})}{\partial x} + \frac{\partial(v\tau_{yy} + u\tau_{xy} + w\tau_{yz})}{\partial y} + \frac{\partial(w\tau_{zz} + v\tau_{zy} + u\tau_{xz})}{\partial z}
\end{aligned} \tag{3.5}$$

Here, the volumetric heating such as absorption or emission of radiation and body source terms such as gravity terms are not included. ρ is density; u , v , w are the Cartesian components of velocity along x , y and z axes; p is pressure and T temperature. They can be related by the state equation of perfect gas $p = \rho RT$, where R is the gas constant which is $287 \text{ m}^2/(\text{s}^2\text{K})$ for air at standard conditions. Furthermore, e_t is the total energy defined as $e_t = e + \frac{1}{2}(u^2 + v^2 + w^2)$, where e is internal energy per unit mass of the fluid and $e = C_v T$. Here C_v is the specific heat at constant volume and $C_v = \frac{R}{\gamma - 1}$. γ is the ratio of specific heats and $\gamma = 1.4$. If we assume that the fluid considered is a calorically perfect gas, then the Navier-Stokes equations are closed by the state equation for perfect gas:

$$p = (\gamma - 1) \rho e$$

$$T = \frac{(\gamma - 1)e}{R}$$

$$H = e + \frac{p}{\rho} + \frac{1}{2}(u^2 + v^2 + w^2)$$

Finally the viscous stresses are related to the velocity field by Stokes relations:

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$\tau_{xx} = 2\mu \frac{\partial u}{\partial x} - \frac{2}{3}\mu \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right), \quad \tau_{xy} = \tau_{yx} = \mu \left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \right),$$

$$\tau_{yy} = 2\mu \frac{\partial v}{\partial y} - \frac{2}{3}\mu \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right), \quad \tau_{xz} = \tau_{zx} = \mu \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right),$$

$$\tau_{zz} = 2\mu \frac{\partial w}{\partial z} - \frac{2}{3}\mu \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right), \quad \tau_{yz} = \tau_{zy} = \mu \left(\frac{\partial w}{\partial y} + \frac{\partial v}{\partial z} \right).$$

The molecular viscosity μ and conductivity κ are properties of the fluid and are functions of temperature. These two quantities are related by the Prandtl number:

$$Pr = \frac{\mu C_p}{\kappa}$$

For air, the Prandtl number is around 0.72 at room temperature.

Equations of fluid motion may be non-dimensionalized to achieve certain objectives. For one, it would provide conditions upon which dynamic and energetic similarity may be obtained for geometrical similar situations. Second, the solution of such equations would usually provide values within limits between zero and one.

The non-dimensional variables used in this study are defined as follows,

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$\begin{aligned}
(x^*, y^*, z^*) &= \left(\frac{x}{L}, \frac{y}{L}, \frac{z}{L} \right) & (u^*, v^*, w^*) &= \left(\frac{u}{U_\infty}, \frac{v}{U_\infty}, \frac{w}{U_\infty} \right); \\
t^* &= \frac{t}{L/U_\infty}; & p^* &= \frac{p - p_{in}}{\rho_\infty (U_\infty)^2}; & e_t^* &= \frac{e_t}{(U_\infty)^2}; & H^* &= \frac{H}{(U_\infty)^2}; \\
\rho^* &= \frac{\rho}{\rho_\infty}; & T^* &= \frac{T}{(U_\infty)^2 / C_v}; & \mu^* &= \frac{\mu}{\mu_\infty}; & Re_\infty &= \frac{\rho_\infty U_\infty L}{\mu_\infty}; \\
q^* &= -\frac{\gamma \mu^*}{Pr} \nabla^* T^*; & \gamma &= \frac{C_p}{C_v}.
\end{aligned}$$

where p_{in} is the inlet or reference pressure. L is the characteristic length of the computed model, U_∞ the inflow velocity and μ the dynamic viscosity of the flow. The variables with a superscript * here are non-dimensional parameters and the asterisk sign will be dropped in subsequent equations for sake of convenience. It should be noted that we subtract a constant value (the reference pressure) from the pressure term to control the round-off errors for low speed flows, which is found to be critical in controlling computational errors in the momentum equations for low-speed compressible flows. The use of gauge pressure is a common treatment for incompressible solvers because only pressure gradients are needed for all calculations. For compressible solvers, the absolute value of pressure must be used when dealing with the energy equation and state equation of gas. But when we use non-dimensional absolute pressure at low Mach numbers, it becomes extremely large although the pressure gradients in momentum equations are small. The use of gauge pressure can avoid performing the addition and subtraction operations between two extraordinarily large values of non-dimensional absolute pressures. In our experience, the result obtained with gauge pressure is more accurate than without using it. And the relative error can be up to 4%~5% of the result.

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

One thing should be noted is that the original absolute pressure (designated by p' hereafter) should be used instead of gauge pressure whenever the state equation of gas is involved.

The nondimensional equations of fluid motion may be expressed in vector form as

$$\frac{\partial W_c}{\partial t} + \frac{\partial E_i}{\partial x} + \frac{\partial F_i}{\partial y} + \frac{\partial G_i}{\partial z} = \frac{\partial E_v}{\partial x} + \frac{\partial F_v}{\partial y} + \frac{\partial G_v}{\partial z} \quad (3.6)$$

$$W_c = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e_t \end{bmatrix} \quad (3.7)$$

$$E_i = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ (\rho e_t + p)u \end{bmatrix} \quad (3.8) \quad E_v = \frac{1}{Re_\infty} \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ (u\tau_{xx} + v\tau_{xy} + w\tau_{xz} - q_x) \end{bmatrix} \quad (3.9)$$

$$F_i = \begin{bmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ \rho vw \\ (\rho e_t + p)v \end{bmatrix} \quad (3.10) \quad F_v = \frac{1}{Re_\infty} \begin{bmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ \tau_{yz} \\ (u\tau_{yx} + v\tau_{yy} + w\tau_{yz} - q_y) \end{bmatrix}$$

(3.11)

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$G_i = \begin{bmatrix} \rho w \\ \rho w u \\ \rho w v \\ \rho w^2 + p \\ (\rho e_i + p) w \end{bmatrix} \quad (3.12) \quad G_v = \frac{1}{Re_\infty} \begin{bmatrix} 0 \\ \tau_{zx} \\ \tau_{zy} \\ \tau_{zz} \\ u\tau_{zx} + v\tau_{zy} + w\tau_{zz} - q_z \end{bmatrix}$$

(3.13)

3.2 LOW SPEED PRECONDITIONING FORMULATION

One difficulty with compressible Navier-Stokes solvers is their slow convergence rates and even unstable solutions for low Mach number flows. This difficulty can be traced to a disparity between the acoustic and convective speeds [117-126], and can be addressed by a preconditioning algorithm. Previous work in this area has been reported by Venkateswaran and Merkle [120, 123], Turkel [124], Van Leer et al. [125], Weiss and Smith [126]. The applications of the preconditioning methods have been found in the computation of steady flows without considering arbitrarily moving objects in the flow field.

The preconditioned Navier-Stokes equations for three-dimensional compressible unsteady flows can be given in vector form explicitly expressing the conservation laws of mass, momentum and energy. We also introduce, in the equations, pseudo-time terms to provide pseudo-time marching for their numerical solutions:

$$\Gamma_1 \frac{\partial W_p}{\partial \tau} + \frac{\partial W_c}{\partial t} + \nabla \cdot \vec{F}_i = \nabla \cdot \vec{F}_v \quad (3.14)$$

where

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$W_c = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e_t \end{bmatrix} \quad (3.15) \quad W_p = \begin{bmatrix} p \\ u \\ v \\ w \\ T \end{bmatrix}$$

(3.16)

$$\vec{F}_i = \begin{bmatrix} \rho \vec{U} \\ \rho u \vec{U} + p \vec{i} \\ \rho v \vec{U} + p \vec{j} \\ \rho w \vec{U} + p \vec{k} \\ \rho H \vec{U} \end{bmatrix} \quad (3.17) \quad \vec{F}_v = \frac{1}{Re_\infty} \begin{bmatrix} 0 \\ \vec{\tau}_x \\ \vec{\tau}_y \\ \vec{\tau}_z \\ (\vec{\tau} \cdot \vec{U} - \vec{q}) \end{bmatrix} \quad (3.18)$$

τ is the pseudo time and Γ_1 is the preconditioning matrix in the pseudo-time terms for low-Mach-number flows which is defined in the appendix. W_c and W_p are the vectors of conservative and primitive dependent variables respectively; $\vec{F}_i$ and $\vec{F}_v$ are the inviscid convective flux and viscous flux vectors. Furthermore we have the following formulas:

$$\begin{aligned} \vec{U} &= u\vec{i} + v\vec{j} + w\vec{k}; \\ \vec{\tau} &= \tau_x\vec{i} + \tau_y\vec{j} + \tau_z\vec{k}; \\ \vec{\tau}_i &= \tau_{ix}\vec{i} + \tau_{iy}\vec{j} + \tau_{iz}\vec{k}; \\ \tau_{ij} &= \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3} \delta_{ij} \nabla \cdot \vec{U} \right) \\ \vec{q} &= q_x\vec{i} + q_y\vec{j} + q_z\vec{k} \end{aligned}$$

$$T' = p' / \rho = c^2 / \gamma$$

$\vec{i}, \vec{j}$ and $\vec{k}$ are the three unit vectors in three Cartesian directions, τ_{ix}, τ_{iy} and τ_{iz} the viscous stresses.

3.3 DISCRETIZATION METHODS

The 3D equations (3.14) are transformed into an integral form and discretized on an unstructured grid. A cell-vertex finite volume scheme is adopted here. For every vertex, as shown in Figure 3.1, a control volume is constructed using the median duals of the tetrahedral cells. The sub-volume with vertexes $1-c-2-b-3-a-P-O$ is the contribution of cell $ACBP$ to the control volume surrounding node P .

Figure 3.1 Construction of control volume within a tetrahedron for a node P

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

Spatial discretization is performed by using the integral form of the conservation equations over the control volume surrounding node P :

$$\iiint_{cv} \frac{\partial Q'_1}{\partial \tau} dV + \iiint_{cv} \frac{\partial W_c}{\partial t} dV + \iiint_{cv} \nabla \cdot \vec{F}_i dV - \iiint_{cv} \nabla \cdot \vec{F}_v dV = 0 \quad (3.19)$$

Please be noted that a new variable Q'_1 has arisen as $\frac{\partial Q'_1}{\partial \tau} = \Gamma_1 \frac{\partial W_p}{\partial \tau}$, and the Jacobian

$\Gamma_1 = \frac{\partial Q'_1}{\partial W_p}$. So that, we have:

$$\frac{\partial Q'_1}{\partial \tau} = \frac{\partial Q'_1}{\partial W_p} \frac{\partial W_p}{\partial \tau} = \Gamma_1 \frac{\partial W_p}{\partial \tau}$$

The convective term is transformed into a summation:

$$\iiint_{cv} \nabla \cdot \vec{F}_i dV = \int_{S_{cv}} \vec{F}_i \cdot \vec{n} dS = \sum_{k=1}^{nbseg} \left[\left(\vec{F}_i \right)_{ij} \cdot \vec{n} \Delta S \right]_k \quad (3.20)$$

where $nbseg$ is the number of the edges associated with node P , $(\vec{F}_i)_{ij}^k$ is the inviscid flux through the part of control volume surface associated with edge k , and $\vec{n}$ is the unit normal vector of the control volume surface. Finally, ΔS_k is a part of the control volume surface associated with edge k . Therefore, all the fluxes are calculated for the edges and then collected at the two end of each edge for updating of flow variables in time marching. The viscous term is calculated using a cell-based method:

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$\iiint_{cv} \nabla \cdot \vec{F}_v dV = \int_{S_{cv}} \vec{F}_v \cdot \vec{n} dS = \sum_{i=1}^{ncell} [\vec{F}_v \cdot \vec{n} \Delta S_c]_i \quad (3.21)$$

where $ncell$ is the number of elements associated with node P and ΔS_{ci} is the part of control volume surface in cell i . By using the following relation:

$$\int_{S_{cv}} \vec{n} dS = 0$$

the total vector surface of the control volume in a cell i becomes: $\vec{n} \Delta S_{ci} = \frac{1}{3} (\vec{n} \Delta S_{pi})$

Thus, the calculation of viscous terms can be simplified as

$$\sum_{i=1}^{ncell} [\vec{F}_v \cdot \vec{n} \Delta S_c]_i = \frac{1}{3} \sum_{i=1}^{ncell} [\vec{F}_v \cdot \vec{n} \Delta S_p]_i \quad (3.22)$$

where $\vec{n} \Delta S_{pi}$ is the surface vector of the face opposite node P of the tetrahedron under consideration. Here the $(\vec{F}_v)_i$ is calculated at the center of the tetrahedron with a node P , and can be obtained by using the Green's Theorem based on the variables at the four vertices of the tetrahedron. Similar to the Galerkin type of formulation, the gradient of a flow variable ϕ at the center of a tetrahedron is evaluated as follows:

$$grad \phi_c = - \frac{\sum_{i=1}^4 \phi_i 9S_i}{27V} = - \frac{1}{3} \frac{\sum_{i=1}^4 \phi_i S_i}{V} \quad (3.23)$$

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

where ϕ_i is the flow variable at a vertex i of the tetrahedron and S_i is the surface area that is opposite to node i , V is the volume of the tetrahedron. Gradients at the vertices are obtained by a volume averaging of the gradients at the center of cells associated with the vertex under consideration.

The Riemann Problem is an initial problem with piece-wise constant initial data. Its exact solution represents the real physical characteristics of a flow with several families of waves and their propagation. When the wave is a shock, the equations for this initial problem can not be explicitly solved and an iterative solution has to be used. In order to reduce the amount of calculation, some approximate treatments have been employed in numerical solutions. In this work, a high-order Roe's TVD scheme for compressible flow for arbitrary unstructured 3D grids has been adopted. In Roe's approach [127], cell surface values of density, velocity and enthalpy are calculated by using a special averaging procedure, called Roe's Averaging:

$$\begin{aligned}\bar{\rho} &= \sqrt{\rho^L} \sqrt{\rho^R} \\ (\bar{u}, \bar{v}, \bar{w}) &= \frac{(u, v, w)^L \sqrt{\rho^L} + (u, v, w)^R \sqrt{\rho^R}}{\sqrt{\rho^L} + \sqrt{\rho^R}}\end{aligned}\quad (3.24)$$

$$\bar{H} = \frac{H^L \sqrt{\rho^L} + H^R \sqrt{\rho^R}}{\sqrt{\rho^L} + \sqrt{\rho^R}} \quad (3.25)$$

where L and R represent the two neighbouring points on an edge, h denotes total enthalpy and it has the following relation:

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$H = e + \frac{p'}{\rho} + \frac{1}{2}(u^2 + v^2 + w^2) \quad (3.26)$$

The speed of sound can be derived as:

$$\bar{c} = \sqrt{\left(\bar{H} - \frac{1}{2}(u^2 + v^2 + w^2)\right)(\gamma - 1)} \quad (3.27)$$

The inviscid flux $(\vec{F}_i)_{ij}$ is evaluated based on Roe's approximate Riemann solver (or the flux difference splitting scheme):

$$\begin{aligned} (F_i)_{ij} &= \frac{1}{2} \left[F_i^L + F_i^R - |\bar{A}| (W^R - W^L) \right] \\ &= \frac{1}{2} \left[F_i^L + F_i^R - |\Delta F| \right] \\ &= F_i^L + \Delta F^- \\ &= F_i^R - \Delta F^+ \end{aligned} \quad (3.28)$$

Here

$$F_i^L = F_i(W_c^L)$$

$$F_i^R = F_i(W_c^R)$$

and W_c^L and W_c^R are the left and right conserved state vectors on the two neighbouring points of an edge. The flux difference is:

$$|\Delta F| = \Delta F^+ - \Delta F^- \quad (3.29)$$

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$\begin{aligned}
 \Delta F^\pm = & \lambda_1^\pm \left\{ \left(\Delta \rho - \frac{\Delta p}{(\bar{c})^2} \right) \begin{bmatrix} 1 \\ \bar{u} \\ \bar{v} \\ \bar{w} \\ (\bar{U}_n)^2 / 2 \end{bmatrix} + \bar{\rho} \begin{bmatrix} 0 \\ \Delta u - n_x \Delta U_n \\ \Delta v - n_y \Delta U_n \\ \Delta w - n_z \Delta U_n \\ \bar{u} \Delta u + \bar{v} \Delta v + \bar{w} \Delta w - \bar{U}_n \Delta U_n \end{bmatrix} \right\} \\
 & + \lambda_4^\pm \left(\frac{\Delta p + \bar{\rho} c \Delta U_n}{2(\bar{c})^2} \right) \begin{bmatrix} 1 \\ \bar{u} + n_x \bar{c} \\ \bar{v} + n_y \bar{c} \\ \bar{w} + n_z \bar{c} \\ \bar{H} + \bar{U}_n \bar{c} \end{bmatrix} \\
 & + \lambda_5^\pm \left(\frac{\Delta p - \bar{\rho} c \Delta U_n}{2(\bar{c})^2} \right) \begin{bmatrix} 1 \\ \bar{u} \pm n_x \bar{c} \\ \bar{v} \pm n_y \bar{c} \\ \bar{w} \pm n_z \bar{c} \\ \bar{H} \pm \bar{U}_n \bar{c} \end{bmatrix}
 \end{aligned} \tag{3.30}$$

Where

$$(\bar{U}_n)^2 = \bar{u}^2 + \bar{v}^2 + \bar{w}^2$$

$$\Delta U_n = \Delta u n_x + \Delta v n_y + \Delta w n_z$$

$$\bar{U}_n = \bar{u} n_x + \bar{v} n_y + \bar{w} n_z$$

Furthermore, the eigenvalues of $|\bar{A}|$ are $\lambda_1 = \bar{U}_n$, $\lambda_{4,5} = \bar{U}_n \pm \bar{c}$. Because of the

introduction of preconditioning matrix Γ_1 , the inviscid fluxes, $(\vec{F}_i)_{ij}^k$, through the face k

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

is now reformulated as

$$\begin{aligned} (\bar{F}_i)_{ij}^k &\equiv \frac{1}{2} \left((\bar{F}_i)_i + (\bar{F}_i)_j \right)_k - \frac{1}{2} \left| \frac{\partial \bar{F}_i}{\partial Q_1'} \right|_k (\delta Q_1')_k \\ &= \frac{1}{2} \left((\bar{F}_i)_i + (\bar{F}_i)_j \right)_k - \frac{1}{2} \left| \frac{\partial \bar{F}_i}{\partial W_p} \frac{\partial W_p}{\partial Q_1'} \right|_k \left(\frac{\partial Q_1'}{\partial W_p} \right)_k \left((W_p)_j - (W_p)_i \right)_k \end{aligned} \quad (3.31)$$

Note that we have retained the variable, Q_1' , in computing this flux. Defining the Jacobian in the normal direction as

$$(H_p)_k = \left(\frac{\partial \bar{F}_i}{\partial W_p} \right)_k$$

And using the previously defined Jacobian $\Gamma_1 = \frac{\partial Q_1'}{\partial W_p}$, then the above expression

becomes

$$(\bar{F}_i)_{ij}^k \equiv \frac{1}{2} \left((\bar{F}_i)_i + (\bar{F}_i)_j \right)_k - \frac{1}{2} |H_p \Gamma_1^{-1}|_k \Gamma_{1k} \left((W_p)_j - (W_p)_i \right)_k$$

Drop the subscript k in the flux vector and the Jacobian with the assumption that the fluxes and Jacobians all correspond to conditions in the normal direction on the given control volume surface. And after some simple algebraic derivations we have

$$(\bar{F}_i)_{ij} \equiv \frac{1}{2} \left((\bar{F}_i)_i + (\bar{F}_i)_j \right) - \frac{1}{2} \Gamma_1 |\Gamma_1^{-1} H_p| \left((W_p)_j - (W_p)_i \right) \quad (3.32)$$

Combined with the third-order MUSCL interpolation, it can produce accurate and stable solution on unstructured grids. The left and right state vectors W_L and W_R at a control

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

volume surface are evaluated using a nominally third-order upwind-biased interpolation scheme:

$$W_L = W_i + \frac{1}{4}[(1 - \kappa)\Delta_i^- + (1 + \kappa)\Delta_i^+] \quad (3.33a)$$

$$W_R = W_j - \frac{1}{4}[(1 - \kappa)\Delta_j^+ + (1 + \kappa)\Delta_j^-] \quad (3.33b)$$

where

$$\Delta_i^+ = \Delta_j^- = W_j - W_i$$

$$\Delta_i^- = W_i - W_{i-1} = 2\vec{ij} \cdot \nabla W_i - (W_j - W_i) = 2\vec{ij} \cdot \nabla W_i - \Delta_i^+$$

$$\Delta_j^+ = W_{j+1} - W_j = 2\vec{ij} \cdot \nabla W_j - (W_j - W_i) = 2\vec{ij} \cdot \nabla W_j - \Delta_j^-$$

Therefore, substituting the above equations into Equations (2.33a) and (2.33b), the final equations based on upwind-biased interpolation scheme is shown as follows:

$$W_L = W_i + \frac{1}{2}[(1 - \kappa)\vec{ij} \cdot \nabla W_i + \kappa\Delta_i^+] \quad (3.34a)$$

$$W_R = W_j - \frac{1}{2}[(1 - \kappa)\vec{ij} \cdot \nabla W_j + \kappa\Delta_j^-] \quad (3.34b)$$

where κ is set to 1/3, which corresponds to a nominally third-order accuracy. $\vec{ij}$ is the vector representing the edge, which points from node P to its neighbouring node under consideration. The gradients of W at i and j are calculated by volume-averaging the gradients of the cells that surround i and j .

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

Finally, for a given node P , the spatially discretized equation (3.19) form a system of coupled ordinary differential equations. This can be reformulated as:

$$\begin{aligned} & \frac{\partial \bar{Q}'_1}{\partial \tau} \Delta V_{cv} + \frac{\partial \bar{W}_c}{\partial t} \Delta V_{cv} \\ &= - \left\{ \frac{1}{2} \sum_{k=1}^{nbseg} \left[\left(\bar{F}_i \right)_i + \left(\bar{F}_i \right)_j \right] - \Gamma_1 \left| \Gamma_1^{-1} H_p \right| \left[\left(W_p \right)_j - \left(W_p \right)_i \right] \right]_k \cdot (\bar{n} \Delta S)_k - \frac{1}{3} \sum_{i=1}^{ncell} \left[\bar{F}_v \cdot \bar{n} \Delta S_p \right]_i \right\} \\ &= -R \end{aligned}$$

(3.35)

where R represents the residual which includes the convective and diffusive fluxes and ΔV_{cv} is the control volume of node P . the over-bar in Equations (3.35) denotes the cell-averaging value. An implicit scheme is adopted for Equations (3.35) and the time dependent term is discretized using a second-order-accurate backward differencing scheme:

$$\frac{\partial \bar{Q}'_1}{\partial \tau} \Delta V_{cv} = -R^{n+1} - \left(\frac{1.5 \Delta V_{cv}^{n+1} W_c^{n+1} - 2.0 \Delta V_{cv}^n W_c^n + 0.5 \Delta V_{cv}^{n-1} W_c^{n-1}}{\Delta t} \right) = \tilde{R}^{n+1} \quad (3.36)$$

where the superscript $(n+1)$ denotes the physical time level $(n+1) \Delta t$ and all the variables are evaluated at this time level, $\tilde{R}(W_p^{n+1})$ is the new modified residual which contains both the time derivative and flux vectors. The derivative with respect to a pseudo time τ is discretized as

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$\Delta V_{cv}^{n+1} \Gamma_1 \frac{W_p^{n+1,m+1} - W_p^{n+1,m}}{d\tau} = \tilde{R}^{n+1,m+1} \quad (3.37)$$

whose solution is sought by marching to a pseudo steady state in τ . Here m and $(m+1)$ denote the initial and final pseudo time levels. Once the artificial steady state is reached, the derivative of W_p with respect to τ becomes zero, and the solution satisfies $\tilde{R}^{n+1} = 0$. Hence, the original unsteady Navier-Stokes equations are fully recovered. Therefore, instead of solving the equations in each time step in the physical time domain (t), the problem is transformed into a sequence of steady-state computations in the artificial time domain (τ). Equation (3.37) can be integrated in pseudo time by an explicit five-stage Runge-Kutta scheme. However, the pseudo time step size may be severely restricted if the physical time step size is very small. Hence, a fully implicit dual time stepping is adopted here.

A Taylor series expansion is performed for the residual in Equations (3.37) with respect to the pseudo time for node i ,

$$\tilde{R}_i^{m+1} = \tilde{R}_i^m + \frac{\partial \tilde{R}_i}{\partial (W_p)_i} \Delta (W_p)_i + \sum_{j=1}^{nbseg} \frac{\partial \tilde{R}_i}{\partial (W_p)_j} \Delta (W_p)_j \quad (3.38)$$

Where $nbseg$ is the number of edges connected to i , which is also equal to the number of neighboring points connected to point i through the edges. An approximate flux function is introduced here to simplify the implicit time stepping calculation. The total flux (including both convective and viscous fluxes) across a control volume surface associated with a certain edge ij can be approximated as

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$F_{ij} \approx \frac{1}{2} \left[\left(\vec{F}_i \right)_i \cdot \vec{n} + \left(\vec{F}_i \right)_j \cdot \vec{n} - |\lambda_{ij}| \left((W_p)_j - (W_p)_i \right) \right]$$

where λ_{ij} is the spectral radius based on the preconditioned system which is associated with edge ij , and is given in the appendix. Then in all of the $R(W)$ terms (Equations (3.35)), we have

$$\frac{\partial R_i}{\partial (W_p)_i} = \sum_{j=1}^{nbseg} \frac{1}{2} \left[\frac{\partial (\vec{F}_i)_{ij}}{\partial (W_p)_i} + |\lambda_{ij}| \right] \quad \text{and} \quad \frac{\partial R_i}{\partial (W_p)_j} = \sum_{i=1}^{nbseg} \frac{1}{2} \left[\frac{\partial (\vec{F}_i)_{ij}}{\partial (W_p)_j} - |\lambda_{ij}| \right].$$

And for the physical time-dependent terms, we have

$$(W_c)_i^{m+1} = (W_c)_i^m + \frac{\partial W_c}{\partial W_p} \Delta (W_p)_i$$

After combining all the residuals terms at every node in the flow field into a vector and dropping the third term of the right-hand side of Equation (3.38) as an approximation, we have

$$\begin{aligned} \tilde{R}_i^{n+1,m+1} &= \tilde{R}_i^{n+1,m} - \frac{\partial R_i}{\partial (W_p)_i} \Delta (W_p)_i - 1.5 \frac{\Delta V_{cv}^{n+1}}{\Delta t} \frac{\partial W_c}{\partial W_p} \Delta (W_p)_i \\ &= \tilde{R}_i^{n+1,m} - \sum_{j=1}^{nbseg} (H_{p,j}) \Delta (W_p)_i - 1.5 \frac{\Delta V_{cv}^{n+1}}{\Delta t} M \Delta (W_p)_i \end{aligned}$$

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

where $H_{p,j} = \frac{1}{2} \left[\frac{\partial(\vec{F}_i)_{ij}}{\partial(W_p)_i} + |\lambda_{ij}| \right]$, $M = \frac{\partial W_c}{\partial W_p}$. And the whole-field equivalent of

Equations (3.36) can then be re-written as

$$\begin{aligned} & \left(I + \frac{1.5\Delta\tau}{\Delta t} \Gamma_1^{-1} M + \frac{\Delta\tau}{\Delta V_{cv}^{n+1}} \sum_{j=1}^{nbseg} (\Gamma_1^{-1} H_{p,j}) \right) \frac{W_p^{n+1,m+1} - W_p^{n+1,m}}{\Delta\tau} \Delta V_{cv}^{n+1} \\ & = \Gamma_1^{-1} \left(-\frac{1.5W_c^{n+1,m} \Delta V_{cv}^{n+1} - 2.0W_c^n \Delta V_{cv}^n + 0.5W_c^{n-1} \Delta V_{cv}^{n-1}}{\Delta t} - R^{n+1,m} \right) \end{aligned} \quad (3.39)$$

that is,

$$\tilde{A} \frac{W_p^{n+1,m+1} - W_p^{n+1,m}}{\Delta\tau} \Delta V_{cv}^{n+1} = \Gamma_1^{-1} \tilde{R}^{n+1,m}$$

thus,

$$\frac{W_p^{n+1,m+1} - W_p^{n+1,m}}{\Delta\tau} \Delta V_{cv}^{n+1} = \tilde{\tilde{R}}^{n+1,m}$$

where $\tilde{\tilde{R}}^{n+1,m} = \tilde{A}^{-1} \Gamma_1^{-1} \tilde{R}^{n+1,m}$ and $\tilde{A} = I + \frac{1.5\Delta\tau}{\Delta t} \Gamma_1^{-1} M + \frac{\Delta\tau}{\Delta V_{cv}^{n+1}} \sum_{j=1}^{nbseg} (\Gamma_1^{-1} H_{p,j})$

Therefore,

$$\tilde{\tilde{R}}^{n+1,m} = -R^{n+1,m} - K \left(\frac{1.5\Delta S_{cv}^{n+1} W^{n+1,m} - 2.0\Delta S_{cv}^n W^n + 0.5\Delta S_{cv}^{n-1} W^{n-1}}{\Delta t} \right)$$

Further approximation can be introduced in order to achieve matrix-free computation. If we employ point implicit treatment to the preceding equations, then only the diagonal terms in $\tilde{A}$ are used in the pseudo time stepping. As a result, the equation for every node can now be written as

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$\frac{W_p^{n+1,m+1} - W_p^{n+1,m}}{\Delta \tau} \Delta V_{cv}^{n+1} = \tilde{R}^{n+1,m} \quad (3.40)$$

where

$$\tilde{R}_i^{n+1,m} = \tilde{A}_{ii}^{-1} \Gamma_i^{-1} \tilde{R}_i^{n+1,m}$$

and

$$\tilde{A}_{ii}^{-1} = \text{diag} \left[\left(I + \frac{1.5 \Delta \tau}{\Delta t} \Gamma_i^{-1} M + \frac{\Delta \tau}{\Delta V_{cv}^{n+1}} \sum_{j=1}^{nbscg} \left(\Gamma_i^{-1} H_{p,j} \right) \right)^{-1} \right].$$

Pseudo time stepping is then performed on Equation (3.40). For a five-stage Runge-Kutta scheme, the stage coefficients are

$$\alpha_1 = \frac{1}{4}, \quad \alpha_2 = \frac{1}{6}, \quad \alpha_3 = \frac{3}{8}, \quad \alpha_4 = \frac{1}{2}, \quad \alpha_5 = 1$$

and we have

$$\begin{aligned} W_{c,p}^{(0)} &= W_{c,p}^m \\ W_{c,p}^{(1)} &= W_{c,p}^{(0)} - \alpha_1 C \frac{\Delta \tau}{\Delta V_{cv}} \tilde{R}(W_{c,p}^{(0)}) \\ W_{c,p}^{(2)} &= W_{c,p}^{(0)} - \alpha_2 C \frac{\Delta \tau}{\Delta V_{cv}} \tilde{R}(W_{c,p}^{(1)}) \\ W_{c,p}^{(3)} &= W_{c,p}^{(0)} - \alpha_3 C \frac{\Delta \tau}{\Delta V_{cv}} \tilde{R}(W_{c,p}^{(2)}) \\ W_{c,p}^{(4)} &= W_{c,p}^{(0)} - \alpha_4 C \frac{\Delta \tau}{\Delta V_{cv}} \tilde{R}(W_{c,p}^{(3)}) \\ W_{c,p}^{(5)} &= W_{c,p}^{(0)} - \alpha_5 C \frac{\Delta \tau}{\Delta V_{cv}} \tilde{R}(W_{c,p}^{(4)}) \\ W_{c,p}^{(m+1)} &= W_{c,p}^{(5)} \end{aligned} \quad (3.41)$$

To speed up the convergence rate, an implicit residual smoothing scheme developed for unstructured grids is employed. The smoothing equation for a vertex k can be expressed as follows:

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$\bar{R}_k = R_k + \varepsilon \nabla^2 \bar{R}_k \quad (3.42)$$

where R is the original residual, $\bar{R}$ is smoothed residual and ε is the smoothing coefficient, which can be defined as

$$\varepsilon = \max \left\{ \frac{1}{4} \left[\left(\frac{CFL}{CFL^*} \right)^2 - 1 \right], 0 \right\}$$

where CFL^* is the maximum CFL number of the basic scheme. The solution to the above equations can be obtained on an unstructured grid by using the Jacobi iterative method as follows,

$$\bar{R}_k^{(m)} = \frac{R_k^{(0)} + \varepsilon \sum_{i=1}^{numnod(k)} \bar{R}_i^{(m-1,m)}}{1 + \varepsilon \cdot numnod(k)} \quad (3.43)$$

where $numnod(k)$ is the number of neighboring nodes of vertex k .

3.4 BOUNDARY TREATMENT

Physically correct numerical boundary conditions are used to achieve time accurate solutions for both steady and unsteady flows. This kind of boundary conditions should be compatible with modern low-dissipation and low-dispersion numerical algorithms which have very low dispersion errors and require precise boundary conditions to avoid

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

numerical instabilities and to control spurious non-physical wave reflections at the computational boundaries. A high-order and high-resolution scheme as one of those numerical algorithms gives results sensitive to the boundary values in the computation, thus the quality of solutions even on interior nodes are dependent on the accuracy and correctness of boundary conditions [128].

Thompson [129, 130] decomposed hyperbolic equations into wave modes of definite velocity and then specified characteristic boundary conditions for incoming waves. The starting point of his analysis was nonlinear Euler equations. The idea of his approach was that one-dimensional characteristic analysis could be performed by consideration of the transverse terms as a constant source term. The amplitudes of outward propagating waves are defined entirely from the variables inside computational domain, while those of inward propagating waves are specified as the characteristic boundary conditions. In our studies, we choose to let the state variables of the far field boundary be floating rather than fixed, in other word, we solve the characteristic equations on the boundary nodes. This method can make the boundary treatment non-reflective to the maximum extent and our solver more stable [131].

3.4.1 Far Field Conditions

In a cell-centered scheme, the Roe TVD upwind scheme described above can be extended to the boundaries conveniently. Since the flux-formula automatically selects the proper information from either side of the boundary face. One may add in an extra cell across the boundary in which the full state is prescribed at any time, even for subsonic inflow and outflow; while in our cell-vertex scheme, the far field boundary

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

conditions for the flow equations are based on the flux vector splitting technique for the inviscid fluxes [132]. For a boundary node k with its control volume boundary face ΔS_k , the inviscid flux across ΔS_k is evaluated by using the Stegger and Warming splitting [132].

At the point upstream of the inflow boundary (denoted by “in”), $\rho_{in}, u_{in}, v_{in}, w_{in}$ and free stream Mach number M_∞ are prescribed. Then, the inflow pressure, temperature and total energy are determined by:

$$p_{in} = \frac{1}{\gamma (M_\infty)^2} \quad (3.44)$$

$$T_{in} = \frac{1}{\gamma (\gamma - 1) (M_\infty)^2} \quad (3.45)$$

$$(\rho e_t)_{in} = \frac{1}{2} \rho_{in} (u_{in}^2 + v_{in}^2 + w_{in}^2) + \frac{p_{in}}{\gamma - 1} \quad (3.46)$$

At the point downstream of the outflow boundary, the only one known state is the pressure. All of the other states must be extrapolated from interior flow field using characteristic variables [131, 25]. For an arbitrary boundary normal direction $\vec{n}$, we can write the variation of characteristic variables as

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$\partial W_n = \begin{pmatrix} \partial W_n^1 \\ \partial W_n^2 \\ \partial W_n^3 \\ \partial W_n^4 \\ \partial W_n^5 \end{pmatrix} = \begin{pmatrix} \partial \rho - \frac{1}{c^2} \partial p \\ \vec{s} \cdot \partial \vec{u} \\ \vec{t} \cdot \partial \vec{u} \\ + \vec{n} \partial \vec{u} + \frac{1}{\rho c} \partial p \\ - \vec{n} \partial \vec{u} + \frac{1}{\rho c} \partial p \end{pmatrix} \quad (3.47)$$

The first characteristic variation is proportional to entropy variations, the second and third are related to variations in shear and tangent velocity, and the last two represent acoustic disturbances. Thus, from ∂W_n^1

$$\rho_{out} = \rho_b \left(\frac{p_{out}}{p_b} \right)^{\frac{1}{\gamma}} \quad (3.48)$$

where subscript “*b*” denotes the variable from the boundary node and “*out*” denotes flow state at the point downstream of the outflow boundary. From ∂W_n^4 , ∂W_n^5 we have

$$U_{n,out} = \sqrt{u_{n,out}^2 + v_{n,out}^2 + w_{n,out}^2} = \sqrt{u_{n,b}^2 + v_{n,b}^2 + w_{n,b}^2} + \frac{2(c_b - \sqrt{\gamma p_{out}/\rho_{out}})}{\gamma - 1}$$

namely from ∂W_n^3

$$U_{t,out} = U_{t,b} = \vec{U}_b - U_{n,b} \vec{n}$$

Therefore, we get

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

$$\vec{U}_{out} = U_{t,out} + U_{n,out} \vec{n} = \vec{U}_b + (U_{n,out} - U_{n,b}) \vec{n} \quad (3.49)$$

$$u_{out} = u_b + (U_{n,out} - U_{n,b}) \vec{n}_x \quad (3.50)$$

$$v_{out} = v_b + (U_{n,out} - U_{n,b}) \vec{n}_y \quad (3.51)$$

$$w_{out} = w_b + (U_{n,out} - U_{n,b}) \vec{n}_z \quad (3.52)$$

$$(\rho e_t)_{out} = \frac{p_{out}}{\gamma - 1} + \frac{\rho_{out} |\vec{U}_{out}|^2}{2} \quad (3.53)$$

As an example, the far field boundary conditions treatment of a strong converged nozzle is shown in Figure 3.2.

Figure 3.2 The inflow & outflow boundary conditions for a strong converged nozzle

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

The boundary fluxes are calculated using Stegger and Warming's flux vector splitting formula [132, 133]:

$$\vec{F}_i \cdot \vec{n} S_k = \left(\vec{F}_i^+ + \vec{F}_i^- \right) \cdot \Delta S_k$$

$$F_i^\pm = \begin{bmatrix} \frac{\rho}{2\gamma} \alpha \\ \frac{\rho}{2\gamma} (\alpha u + \beta c n_x) \\ \frac{\rho}{2\gamma} (\alpha v + \beta c n_y) \\ \frac{\rho}{2\gamma} (\alpha w + \beta c n_z) \\ \frac{\rho}{2\gamma} \left(\alpha \frac{u^2 + v^2 + w^2}{2} + 2\beta c U_n + \frac{\varpi}{\gamma - 1} c^2 \right) \end{bmatrix} \quad (3.54)$$

Here

$$U_n = \vec{U} \cdot \vec{n}$$

$$\lambda_1 = \vec{U} \cdot \vec{n}$$

$$\lambda_4 = \vec{U} \cdot \vec{n} + c$$

$$\lambda_5 = \vec{U} \cdot \vec{n} - c$$

$$\alpha = 2(\gamma - 1) \lambda_1^\pm + \lambda_4^\pm + \lambda_5^\pm$$

$$\beta = \lambda_4^\pm - \lambda_5^\pm$$

$$\varpi = \lambda_4^\pm + \lambda_5^\pm$$

$$\lambda^+ = \max \{ \lambda, 0.0 \}$$

$$\lambda^- = \min \{ \lambda, 0.0 \}$$

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

Note, at the inflow, the boundary nodal states are used for calculating $\vec{F}_i^+$, the pre-imposed inflow states are used for $\vec{F}_i^-$; Similarly, at the outflow, the boundary nodal states are used while calculating $\vec{F}_i^+$, the pre-imposed and extrapolated outflow states are used for $\vec{F}_i^-$.

3.4.2 Solid Wall/Slip Conditions

At the solid wall, *slip* (for inviscid flows) or *no-slip* (for viscous flows) and no-injection boundary conditions are imposed, that is, the zero normal fluxes of mass, momentum, and energy are imposed. In addition, the solid surface is assumed to be adiabatic, that is:

$$\nabla \cdot \mathbf{T} = 0$$

3.4.3 Far Field Conditions for Preconditioned Systems

The far field calculations are based on characteristic variables (Reimann invariants). Thus at inflow the incoming variables corresponding to positive eigenvalues are specified while the outgoing variables corresponding to negative eigenvalues are extrapolated. Once we change the time dependent equations we also change the characteristics of the system (though not the signs of the eigenvalues). Hence, it is also necessary to modify the boundary conditions for the preconditioned system. Here the flux at a boundary is defined as

$$\vec{F}_i \cdot \vec{n} S_k = \left(\vec{F}_i^+ + \vec{F}_i^- \right) \cdot \Delta S_k$$

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

Here, $\vec{F}_i^\pm$ has been redefined as $\vec{F}_i^\pm = X_{H_p,R} \Lambda^\pm X_{H_p,L} W_p$. Where $\Lambda^\pm = \frac{\lambda_i \pm |\lambda_i|}{2}$, and

λ_i represents the eigenvalues of H_p (see Appendix for details of λ_i). $X_{H_p,R}$ and

$X_{H_p,L}$ are the right and left eigenvectors of H_p (see Appendix).

We first calculate

$$z = X_{H_p,L} W_p = \begin{pmatrix} (1-\gamma)n_x p / (\gamma\rho) + n_z v + n_x T' - n_y w \\ (1-\gamma)n_y p / (\gamma\rho) - n_z u + n_x w + n_y T' \\ (1-\gamma)n_z p / (\gamma\rho) + n_y u - n_x v + n_z T' \\ p + n_x \rho (\lambda_1 - \beta U) u + n_y \rho (\lambda_1 - \beta U) v + n_z \rho (\lambda_1 - \beta U) w \\ p + n_x \rho (\lambda_2 - \beta U) u + n_y \rho (\lambda_2 - \beta U) v + n_z \rho (\lambda_2 - \beta U) w \end{pmatrix} \quad (3.55)$$

And then:

$$\vec{F}_i^\pm = X_{H_p,R} \Lambda^\pm X_{H_p,L} W_p = (X_{H_p,R} \Lambda^\pm) z = \begin{pmatrix} ((\beta U - \lambda_2) \lambda_1^\pm z_4 + (\lambda_1 - \beta U) \lambda_2^\pm z_5) / S \\ n_y \lambda_0^\pm z_3 - n_z \lambda_0^\pm z_2 + (n_x \lambda_1^\pm z_4 - n_x \lambda_2^\pm z_5) / (\rho S) \\ n_z \lambda_0^\pm z_1 - n_x \lambda_0^\pm z_3 + (n_y \lambda_1^\pm z_4 - n_y \lambda_2^\pm z_5) / (\rho S) \\ n_x \lambda_0^\pm z_2 - n_y \lambda_0^\pm z_1 + (n_z \lambda_1^\pm z_4 - n_z \lambda_2^\pm z_5) / (\rho S) \\ n_x \lambda_0^\pm z_1 + n_y \lambda_0^\pm z_2 + n_z \lambda_0^\pm z_3 + \\ (\gamma - 1) ((\beta U - \lambda_2) \lambda_1^\pm z_4 + (\lambda_1 - \beta U) \lambda_2^\pm z_5) / (\gamma \rho S) \end{pmatrix} \quad (3.56)$$

where: $S = \sqrt{U^2 (\beta - 1)^2 + 4\beta c^2}$

CHAPTER 3 – MATHEMATICAL FORMULATION FOR COMPRESSIBLE FLOW SOLVER

CHAPTER FOUR

MATHEMATICAL FORMULATION FOR COMPUTATIONAL STRUCTURAL DYNAMICS

This chapter describes the mathematical formulation for the matrix-free structural dynamics implicit solver on unstructured grids developed in this work. The elastic deformation of the structure is calculated with a matrix-free implicit dual time stepping scheme. A five-stage Runge-Kutta time integration algorithm is used between each physical time step to iterate the numerical solution in pseudo time until convergence is reached. The methods used to accelerate the convergence rate to steady state in pseudo time, which are local time stepping, implicit residual smoothing and multigrid, are briefly described.

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

4.1 GOVERNING EQUATIONS

The governing equations for structural dynamics based on the continuum model are the Cauchy's equations:

$$\left. \begin{aligned} b_x + \frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + \frac{\partial \sigma_{xz}}{\partial z} &= \rho \frac{\partial^2 d_x}{\partial t^2} \\ b_y + \frac{\partial \sigma_{yx}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \sigma_{yz}}{\partial z} &= \rho \frac{\partial^2 d_y}{\partial t^2} \\ b_z + \frac{\partial \sigma_{zx}}{\partial x} + \frac{\partial \sigma_{zy}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} &= \rho \frac{\partial^2 d_z}{\partial t^2} \end{aligned} \right\} \quad (4.1)$$

where σ_{ij} is the stress tensor defined for either a fluid or solid medium, with the correlation $\sigma_{ij} = \sigma_{ji}$. b_x , b_y and b_z are the body force components in the three directions of Cartesian coordinates. ρ is the material density. d_x , d_y and d_z are the three components of displacement vector. This system of equations can be expressed in a more compact form:

$$\vec{b} + \nabla \cdot \sigma_{ij} = \rho \vec{a} \quad (4.2)$$

where $\vec{a}$ is the acceleration vector. In the theory of elasticity equation (4.2) is variously described as the stress equation of small motion [134], the equation of equilibrium [135] or the equation of motion [136]. The term equation of dynamic equilibrium will be employed in this paper to distinguish the dynamic problems considered in this research from static structural problems. Damping is the ability of a structure to dissipate energy and in structural mechanics the most common damping

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

device is the ideal linear viscous damper [137-139]. The ideal linear viscous damper opposes structural motion with a force proportional to velocity. Thus, for those cases where damping is required, the linear viscous damping term is incorporated into Equation (4.2) as follows:

$$\vec{b} + \nabla \cdot \sigma_{ij} = \rho \vec{a} + c \vec{U} \quad (4.3)$$

where $\vec{U}$ is velocity vector and c is the coefficient of viscous damping.

4.2 CONSTITUTIVE RELATIONSHIP FOR STRESS AND STRAIN

The generalized form of Hooke's law gives the following constitutive relationship between stress and strain for an isotropic homogeneous material undergoing small strains [140] in three dimensions:

$$\begin{pmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{xy} \\ \sigma_{yz} \\ \sigma_{zx} \end{pmatrix} = D \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \varepsilon_{xy} \\ \varepsilon_{yz} \\ \varepsilon_{zx} \end{pmatrix} = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \begin{pmatrix} 1 & \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\ \frac{\nu}{1-\nu} & 1 & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\ \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} \end{pmatrix} \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \varepsilon_{xy} \\ \varepsilon_{yz} \\ \varepsilon_{zx} \end{pmatrix} \quad (4.4)$$

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

where E and ν are respectively Young's modulus and Poisson's ratio, the stress vector is $\sigma^T = [\sigma_{xx}, \sigma_{yy}, \sigma_{zz}, \sigma_{xy}, \sigma_{yz}, \sigma_{zx}]$ and the strain vector $\varepsilon^T = [\varepsilon_{xx}, \varepsilon_{yy}, \varepsilon_{zz}, \varepsilon_{xy}, \varepsilon_{yz}, \varepsilon_{zx}]$.

The elastic strain at any instant in time may also be expressed in terms of the total and initial strains. Thus the constitutive relationship for an isotropic homogeneous material undergoing linear elastic strains is given by

$$\sigma - D(\varepsilon - \varepsilon^0) = 0 \quad \text{in } \Omega_s \quad (4.5)$$

where D is the constitutive property matrix, given by the term in the large square brackets in equation (4.4) and ε and ε^0 are the total and initial strains respectively. Ω_s is the structural domain.

4.3 DISPLACEMENT FORMULATION

This work is based on a linear strain-displacement formulation using the small strain assumption, which is valid for strains of the order of a few percent [134]. Thus the strains may be defined in a general displacement form as

$$\varepsilon = \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \varepsilon_{xy} \\ \varepsilon_{yz} \\ \varepsilon_{zx} \end{pmatrix} = L \vec{d} = \begin{pmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial x} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \end{pmatrix} \begin{pmatrix} d_x \\ d_y \\ d_z \end{pmatrix} = \begin{pmatrix} \frac{\partial d_x}{\partial x} \\ \frac{\partial d_y}{\partial y} \\ \frac{\partial d_z}{\partial z} \\ \frac{\partial d_x}{\partial y} + \frac{\partial d_y}{\partial x} \\ \frac{\partial d_y}{\partial z} + \frac{\partial d_z}{\partial y} \\ \frac{\partial d_x}{\partial z} + \frac{\partial d_z}{\partial x} \end{pmatrix} \quad (4.6)$$

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

where $\vec{d}$ is the vector of displacements and L is the matrix of differential operators. Applying the constitutive stress-strain equation (4.4) and the strain-displacement equation (4.6) to the dynamic equilibrium equation (4.3) yields the displacement formulation:

$$b + \nabla \cdot (DL\vec{d} - D\varepsilon^0) - \rho\vec{a} - c\vec{U} = 0 \quad \text{in } \Omega_s \quad (4.7)$$

where Ω_s represents the structural domain. Equation (4.7) is subject to the boundary conditions:

$$\left. \begin{aligned} \vec{d} - \vec{d}_p &= 0 & \text{for } \Gamma_d \\ T(DL\vec{d} - D\varepsilon^0) - \vec{t}_p &= 0 & \text{for } \Gamma_t \end{aligned} \right\} \quad (4.8)$$

where the structural boundary is a combination of prescribed displacement $\vec{d}_p$ and traction $\vec{t}_p$ boundaries, i.e. $\Gamma_s = \Gamma_d \cup \Gamma_t$ and T is the matrix of outward normal operators such that

$$T = \begin{pmatrix} n_x & 0 & 0 & n_y & 0 & n_z \\ 0 & n_y & 0 & n_x & n_z & 0 \\ 0 & 0 & n_z & 0 & n_y & n_x \end{pmatrix} \quad (4.9)$$

where n is the outward unit normal vector to the domain boundary with components n_x , n_y and n_z .

4.4 DISCRETIZATION OF THE DISPLACEMENT EQUATIONS

Equation (4.7) is discretized on an unstructured tetrahedral grid and a cell-vertex scheme is adopted here, i.e., all computed variables in vector $\vec{d}$ are stored at vertices of the tetrahedral cells. For every vertex, as shown in Figure 3.1 (which is repeated here for convenience), a control volume is constructed using the median dual of the tetrahedral grid [141]. In the Figure, nodes A , P , B and C form the vertices of the tetrahedral cell and O is the centre of the element $APBC$. Points a , b and c are the centres of the edges AP , BP and CP . 1, 2 and 3 are the centroids of triangles APC , CBP and ABP , respectively. In the cell-vertex scheme, the computed variables are stored at vertices A , P , B and C . Triangles $O1a$, $O3a$, $O3b$, $O2b$, $O1c$ and $O2c$ form part of the control volume surface for node P within the tetrahedral cell. Likewise, the control volume surfaces for different nodes A , B and C are constructed based on this idea.

Figure 3.1 Construction of control volume within a tetrahedron for a node P

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

The finite-volume discretization is based on the governing equations in integral form, on which spatial discretization is performed over the control volume surrounding a node (P for example):

$$\iiint_{CV} \left(\nabla \cdot (DL\vec{d} - D\epsilon^0) \right) \cdot dV + \iiint_{CV} (b - c\vec{U}) \cdot dV - \iiint_{CV} \rho \vec{a} \cdot dV = 0 \quad (4.10)$$

The first term on the left hand side is calculated using a cell-based method:

$$\iiint_{CV} \left(\nabla \cdot (DL\vec{d} - D\epsilon^0) \right) \cdot dV = \oint_{S_{CV}} (DL\vec{d} - D\epsilon^0) \cdot \vec{n} dS = \sum_{i=1}^{ncell} \left[(DL\vec{d} - D\epsilon^0) \cdot \vec{n} \Delta S_c \right]_i$$

where $ncell$ is the number of cells associated with node P and ΔS_{ci} is the part of control volume surface in cell i . By using the following relation for a given cell

$$\oint_{S_{CV}} d\vec{S} = 0$$

the total vector surface of the control volume in a cell i becomes

$$\vec{n} \Delta S_{ci} = \frac{1}{3} (\vec{n} \Delta S_{pi})$$

where $\vec{n} \Delta S_{pi}$ is the surface vector of the face opposite node P of the tetrahedron under consideration. Thus, the calculation of the first term on the left hand side of equation (4.10) can be simplified as

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

$$\sum_{i=1}^{ncell} \left[(DL\vec{d} - D\vec{\varepsilon}^0) \cdot \vec{n} \Delta S_c \right]_i = \frac{1}{3} \sum_{i=1}^{ncell} \left[(DL\vec{d} - D\vec{\varepsilon}^0) \cdot \vec{n} \Delta S_p \right]_i \quad (4.11)$$

Here the value $(DL\vec{d} - D\vec{\varepsilon}^0)_i$ is calculated at the center of the tetrahedron with a node P , and can be obtained by using Green's Theorem based on the variables at the four vertices of the tetrahedron. Similar to the Galerkin type of formulation, the gradient of variable ϕ at the center of a tetrahedron is evaluated as follows:

$$grad\phi_c = -\frac{\sum_{i=1}^4 \phi_i 9\vec{S}_i}{27V} = -\frac{1}{3} \frac{\sum_{i=1}^4 \phi_i \vec{S}_i}{V} \quad (4.12)$$

where ϕ_i is the variable at a vertex i of the tetrahedron and $\vec{S}_i$ is the surface area vector that is opposite to node i , V is the volume of the tetrahedron. Gradients at the vertices are obtained by volume averaging of the gradients at the centers of cells associated with the vertex under consideration.

If we use the spatial averages of density, acceleration and velocity, then equation (4.10) can be written as

$$\frac{1}{3} \sum_{i=1}^{ncell} \left[(DL\vec{d} - D\vec{\varepsilon}^0) \cdot \vec{n} \Delta S_p \right]_i + (b - c\vec{U}) \cdot V - \rho \vec{a} \cdot V = 0 \quad (4.13)$$

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

where V is the volume of the tetrahedron. Knowing that $\vec{a} = \frac{\partial \vec{U}}{\partial t}$, after arrangement we have

$$\frac{1}{3\rho V} \sum_{i=1}^{ncell} \left[(DL\vec{d} - D\epsilon^0) \cdot \vec{n} \Delta S_p \right]_i + \frac{(b - c\vec{U})}{\rho} = \frac{\partial \vec{U}}{\partial t} \quad (4.14)$$

An implicit scheme is adopted for Equations (4.14) and the time dependent term is discretized using a second-order-accurate backward differencing scheme. In order to obtain time-accurate solutions, we add a pseudo time term to this equation:

$$\frac{1}{3\rho V} \sum_{i=1}^{ncell} \left[(DL\vec{d} - D\epsilon^0) \cdot \vec{n} \Delta S_p \right]_i + \frac{(b - c\vec{U}^{n+1})}{\rho} = \left(\frac{1.5\vec{U}^{n+1} - 2.0\vec{U}^n + 0.5\vec{U}^{n-1}}{\Delta t} \right) + \frac{\partial \vec{U}}{\partial \tau} \quad (4.15)$$

where the superscript $(n+1)$ denotes the physical time level $(n+1)\Delta t$ and all the variables are evaluated at this time level. The derivative with respect to a pseudo time τ is discretized using a first-order-accurate forward differencing scheme. After moving the rest of the terms to another side of the equation, we have

$$\frac{\vec{U}^{n+1,m+1} - \vec{U}^{n+1,m}}{\Delta \tau} = \tilde{R}^{n+1,m} = \frac{1}{3\rho V} \sum_{i=1}^{ncell} \left[(DL\vec{d} - D\epsilon^0) \cdot \vec{n} \Delta S_p \right]_i + \frac{(b - c\vec{U}^{n+1,m})}{\rho} - \left(\frac{1.5\vec{U}^{n+1,m} - 2.0\vec{U}^n + 0.5\vec{U}^{n-1}}{\Delta t} \right) \quad (4.16)$$

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

The solution of the velocity vector is sought by marching the solution to a pseudo steady state in τ . Here m and $m+1$ denote the initial and next pseudo time levels. Once the artificial steady state is reached, the derivative of $\vec{U}$ with respect to τ becomes zero, and the solution satisfies $\tilde{R}^{n+1,m} = 0$. Hence, the original equation (4.14) is fully recovered. Therefore, instead of solving the equations in each time step in the physical time domain (t), the problem is transformed into a sequence of steady-state computations in the artificial time domain (τ). Equations (4.16) can be integrated in pseudo time by an explicit five-stage Runge-Kutta scheme. However, the pseudo time step size may be restricted if the physical time step size is very small. Hence, a fully implicit dual time stepping scheme is adopted here. A Taylor series expansion is performed for the residual in Equations (4.16) with respect to the pseudo time for node P :

$$\tilde{R}_p^{n+1,m+1} = \tilde{R}_p^{n+1,m} + \frac{\partial \tilde{R}_p}{\partial (\vec{U})_p} \Delta(\vec{U})_p + \sum_{j=1}^{nbseg} \frac{\partial \tilde{R}_p}{\partial (\vec{U})_j} \Delta(\vec{U})_j \quad (4.17)$$

Where $nbseg$ is the number of edges connected to P , which is also equal to the number of neighbouring points connected to point P through the edges. And for the viscous damping and physical time-dependent terms, we have

$$(\vec{U})_p^{n+1,m+1} = (\vec{U})_p^{n+1,m} + \Delta(\vec{U})_p \quad (4.18)$$

After combining all the residual terms at every node in the structural domain into a vector and dropping the third term of the right-hand side of equations (4.17), we have

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

$$\tilde{R}^{n+1,m+1} = \tilde{R}^{n+1,m} + A\Delta(\vec{U})_p - \frac{1.5}{\Delta t}\Delta(\vec{U})_p - \frac{c}{\rho}\Delta(\vec{U})_p. \quad (4.19)$$

where

$$A = \frac{1}{3M_p} \left(\frac{\partial \left\{ \sum_{i=1}^{ncell} \left[(DL\vec{d} - D\epsilon^0) \cdot \vec{n} \Delta S_p \right]_i \right\}}{\partial(\vec{U})_p} \right) = \frac{1}{3M_p} \left(\frac{\partial \left\{ \sum_{i=1}^{ncell} \left[(DL\vec{d}) \cdot \vec{n} \Delta S_p \right]_i \right\}}{\partial(\vec{U})_p} \right) \quad (4.20)$$

And the whole-field equivalent Equation (4.16) can then be re-written as,

$$\left(\frac{\Delta t + 1.5\Delta\tau}{\Delta t} + \frac{c\Delta\tau}{\rho} - A\Delta\tau \right) \frac{\Delta(\vec{U})_p}{\Delta\tau} = \tilde{R}^{n+1,m} = \frac{1}{3M_p} \sum_{i=1}^{ncell} \left[(DL\vec{d} - D\epsilon^0) \cdot \vec{n} \Delta S_p \right]_i + \frac{(b - c\vec{U}^{n+1,m})}{\rho} - \left(\frac{1.5\vec{U}^{n+1,m} - 2.0\vec{U}^n + 0.5\vec{U}^{n-1}}{\Delta t} \right) \quad (4.21)$$

where $\Delta(\vec{U})_p = \vec{U}^{n+1,m+1} - \vec{U}^{n+1,m}$ and $M_p = \rho V$ is the mass of the control volume

around the current node P . We need to evaluate the derivative

$\frac{\partial(\partial\vec{d}/\partial X_i)}{\partial\vec{U}}(X_i = x, y, z)$ in order to determine the Jacobian A . It can be calculated as

follows:

$$\frac{\partial(\partial\vec{d}/\partial X_i)}{\partial\vec{U}} = \frac{\partial(\partial\vec{d}/\partial X_i)/\partial X_i}{\partial\vec{U}/\partial X_i} = \frac{\partial^2\vec{d}/\partial X_i^2}{\partial\vec{U}/\partial X_i} \quad (4.22)$$

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

where $\vec{d}$ is the displacement vector at node P .

That is
$$\tilde{A} \frac{\Delta(\vec{U})}{\Delta\tau} = \tilde{R}^{n+1,m} \quad (4.23)$$

thus,
$$\frac{\Delta(\vec{U})}{\Delta\tau} = \tilde{\tilde{R}}^{n+1,m} \quad (4.24)$$

where $\tilde{\tilde{R}}^{n+1,m} = \tilde{A}^{-1} \tilde{R}^{n+1,m}$ and $\tilde{A} = \frac{\Delta t + 1.5\Delta\tau}{\Delta t} + \frac{c\Delta\tau}{\rho} - A\Delta\tau$.

Further approximation can be introduced in order to achieve matrix-free computation. If we employ point implicit treatment to the preceding equations, then only the diagonal term in $\tilde{A}$ is used in the pseudo time stepping. As a result, the equation for every node can now be written as

$$\frac{\Delta(\vec{U})}{\Delta\tau} = \tilde{\tilde{R}}^{n+1,m} \quad (4.25)$$

where $\tilde{\tilde{R}}_p^{n+1,m} = \tilde{A}_{pp}^{-1} \tilde{R}_p^{n+1,m}$ and $\tilde{A}_{pp}^{-1} = \text{diag} \left[\left(\frac{\Delta t + 1.5\Delta\tau}{\Delta t} + \frac{c\Delta\tau}{\rho} - A\Delta\tau \right)^{-1} \right]$.

Pseudo time stepping is then performed on Equation (4.24). In this work, a five-stage Runge-Kutta time integration algorithm is used to march the numerical solution in pseudo time τ until convergence is reached [141]. Therefore, the converged solution from the pseudo time steady-state equations becomes the time accurate solution at current physical time. To advance the solution in pseudo time from m to $m+1$, the

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

formulation of a five-stage Runge-Kutta scheme is as follows:

$$\begin{aligned}
 (\vec{U})_p^{(0)} &= (\vec{U})_p^m \\
 (\vec{U})_p^{(1)} &= (\vec{U})_p^{(0)} - \alpha_1 \Delta \tau \tilde{R} \left[(\vec{U})_p^{(0)} \right] \\
 (\vec{U})_p^{(2)} &= (\vec{U})_p^{(0)} - \alpha_2 \Delta \tau \tilde{R} \left[(\vec{U})_p^{(1)} \right] \\
 (\vec{U})_p^{(3)} &= (\vec{U})_p^{(0)} - \alpha_3 \Delta \tau \tilde{R} \left[(\vec{U})_p^{(2)} \right] \\
 (\vec{U})_p^{(4)} &= (\vec{U})_p^{(0)} - \alpha_4 \Delta \tau \tilde{R} \left[(\vec{U})_p^{(3)} \right] \\
 (\vec{U})_p^{(5)} &= (\vec{U})_p^{(0)} - \alpha_5 \Delta \tau \tilde{R} \left[(\vec{U})_p^{(4)} \right] \\
 (\vec{U})_p^{(m+1)} &= (\vec{U})_p^{(5)}
 \end{aligned} \tag{4.26}$$

where the stage coefficients for a five-stage Runge-Kutta time integration is as follows,

$$\alpha_1 = \frac{1}{4}, \quad \alpha_2 = \frac{1}{6}, \quad \alpha_3 = \frac{3}{8}, \quad \alpha_4 = \frac{1}{2}, \quad \alpha_5 = 1$$

After the velocity vector at physical time levels $n+1$ has been solved, we calculate the delta displacement during this time step as

$$\Delta \vec{d} = \frac{(\vec{U}^{n+1} + \vec{U}^n)}{2} \cdot \Delta t \tag{4.27}$$

4.5 CONVERGENCE ACCELERATION TECHNIQUES

4.5.1 Local Time Stepping in Pseudo Time

Due to the disparity in cell sizes in unstructured grid calculations, the chosen time step

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

size for the entire mesh will be the minimum of the local time steps of all the control volumes for time accurate calculations. In this work, the large variation in grid size for the unstructured mesh will restrict the time step used and the smallest control volume dictates the maximum time step size. In order to overcome the above problems, each control volume can be advanced in pseudo time by its own maximum local time step, which greatly enhances the convergence rate. In this study, the maximum permissible local time step size is determined by the critical time step size:

$$\Delta\tau \leq \Delta\tau^{crit} = \frac{2}{\omega_{max}} \quad (4.28)$$

If ω_{max} is the largest natural circular frequency, for the case of cantilever, the natural frequency is given by [136]

$$\omega_{max} = \frac{2c}{\Delta l}$$

where $c = \sqrt{E/\rho(1-\nu^2)}$ is the wave propagation velocity. Δl is the characteristic length scale associated with a node under consideration. Normally, it is taken as the smallest height of all the tetrahedral cells sharing the node. The local time step size is thus estimated via CFL stability condition as:

$$\Delta\tau = CFL \cdot \frac{\Delta l}{c} = CFL \cdot \frac{\Delta l}{\sqrt{E/\rho(1-\nu^2)}} \quad (4.29)$$

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

The global physical time step size Δt for the entire mesh will be the minimum of the local time steps of all the control volumes for time accurate calculations. That is

$$\Delta t = \min \{ \Delta \tau_1, \Delta \tau_1, \dots, \Delta \tau_{ns} \} \quad (4.30)$$

where ns is the total number of nodes.

4.5.2 Implicit Residual Smoothing

In order to speed up the convergence rate, an implicit residual smoothing scheme developed for unstructured grids is employed. The idea behind this is to replace the residual at one point of the domain with a smoothed or weighted average of the residuals at the neighboring points [141]. The averaged residuals are calculated implicitly in order to increase the maximum CFL number, thus increasing the convergence rate. Normally this procedure allows the CFL number to be increased by a factor of 2 or 3. The smoothing equation for a vertex k can be expressed as follows:

$$\bar{R}_k = R_k + \varepsilon \nabla^2 \bar{R}_k \quad (4.31)$$

where $\bar{R}$ is the smoothed residual, R is the original residual, and ε is the smoothing coefficient, which can be defined as

$$\varepsilon = \max \left\{ \frac{1}{4} \left[\left(\frac{CFL}{CFL^*} \right)^2 - 1 \right], 0 \right\} \quad (4.32)$$

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

where CFL^* is the maximum CFL number of the basic scheme. The solution to the preceding equations can be obtained on an unstructured grid by using the Jacobi iterative method as follows:

$$\bar{R}_k^{(m)} = R_k^{(0)} + \varepsilon \sum_{i=1}^{numnod(k)} [\bar{R}_i^{(m)} - \bar{R}_k^{(m)}]$$

$$\text{i.e.} \quad \bar{R}_k^{(m)} = \frac{R_k^{(0)} + \varepsilon \sum_{i=1}^{numnod(k)} \bar{R}_i^{(m-1,m)}}{1 + \varepsilon \cdot numnod(k)} \quad (4.33)$$

where $numnod(k)$ is the number of neighboring nodes of vertex k .

4.5.3 The Multigrid Method

It is well known that multigrid methods can dramatically reduce the overall cost of CFD simulations. The basic idea of the multigrid method is to carry out early iterations on a fine grid and then progressively transfer these solutions and residuals to a series of coarser grids. On the coarser grids, the low frequency errors become high frequency ones and they can be easily eliminated by a time stepping scheme. The equations are then solved on the coarser grids and the corrections are then interpolated back to the fine grid. The process is repeated over a sufficient number of times until satisfactory convergence on the fine grid is achieved. In this study, based on the early works done by Zhao *et al.* [141, 34], an efficient unstructured multigrid scheme has been developed for the equation of dynamic equilibrium, allowing convergence to be achieved with much

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

less CPU time than the single-grid scheme. The multigrid algorithm is described as follows. The discretized dynamic equation can be expressed by equation (4.23), which is repeated here for convenience:

$$\frac{\Delta(\vec{U})}{\Delta\tau} = \tilde{\tilde{R}}^{n+1,m} \quad (4.34)$$

where $\vec{U}$ is the velocity vector. This equation is solved iteratively by a dual-time stepping scheme. An pseudo code is given below to illustrate the basic procedures of the multigrid scheme. The outer cycles are based on the physical time t , which is numbered from 1 to $ktmax$, whereas the inner cycles are based on the pseudo-time τ , which is numbered from 1 to max_itr_sub . The grid levels range from 1 to nmg , where 1 is the finest level and nmg is the coarsest level. P_{h+1}^h is the prolongation operator from level $h+1$ to h , and Q_h^{h+1} and T_h^{h+1} are the residual transfer and restriction operators from level h to $h+1$.

ALGORITHM START

```

call setupinterconnect !---> build up inter-connectivity relationship between levels
DO kt=1, ktmax      !---> start of physical time step
  DO mg=1, nmg
    call cvvol(mg)    !---> calculate the volumes for control volume (CV) and cell
    call clhaut(mg)   !---> calculate CV characteristic length for determination
                        !---> of  $\Delta\tau$  and  $\Delta t$  computation
  ENDDO
  call dsize(mg=1)    !---> computation of the time-step size for the finest level
  DO itersub=1, max_itr_sub !---> start of sub-iteration
    DO ialpha=1, 5 !---> 5-stage Runge-Kutta time integration process
      call fvsolver(mg=1) !---> solve equation (3.1) for  $\vec{U}_1^k$  following the way
                        !---> described in forgoing sections
    
```

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

```

    ENDDO
    DO mg=2, nmg
        call transfrsol(mg-1) !---> restrict solution from h to h+1
            !--->  $(\vec{U}_{h+1}^{(0)} = T_h^{h+1} \vec{U}_h).$ 
        call transfresd(mg-1) !---> restrict residual from h to h+1
            !--->  $(\tilde{\vec{R}}_{h+1}^{(0)} = Q_h^{h+1} \tilde{\vec{R}}(\vec{U}_h)).$ 
        IF (itersub.eq.1) THEN
            call dtsize(mg) !---> calculate the time-step size for the current level
        ENDIF
        DO ialpha=1, 5
            call fvsolver(mg) !---> solve equation (3.1) for  $\vec{U}_{mg}^{kt}$  based on the
            initial
            !---> solution  $\vec{U}_{h+1}^{(0)}$  and initial residual  $\tilde{\vec{R}}_{h+1}^{(0)}$ 
        ENDDO
    ENDDO
    DO mg=nmg, 2, -1
        call transfrcorct(mg) !---> prolongate correction from h+1 to h
            !--->  $(\vec{U}_h^+ = \vec{U}_h + I_{h+1}^h (\vec{U}_{h+1}^+ - \vec{U}_{h+1}^{(0)})).$  Here  $\vec{U}_h^+$  is
            !---> the updated solution for the finer grids
    ENDDO
    ENDDO !---> end of sub-iteration

    DO mg=2, nmg
        call transfrsol(mg-1) !---> restrict solution from h to h+1
    ENDDO
    DO mg=1, nmg
        call updgrid(mg) !---> finally update the solid mesh
    ENDDO
    ENDDO !---> end of physical time step
ALGORITHM END

```

For detailed discussion of the mesh-to-mesh transfer operators, please refer to references in [141, 34]. One should note that we always use the initial inter-mesh connectivity, which is built up before the first time step starts, to perform the

CHAPTER 4 – MATHEMATICAL FORMULATION FOR CSD

mesh-to-mesh transfer operations. In order to enable the solver to tackle more geometrically complex structures, the search for nodes within the boundary faces (edges for 2D) will continue after the search for nodes within the cells are performed if the user specifies that the physical wall of the geometry is curved. See Figure 4.1 for example, a normal procedure can not find a corresponding finer grid cell for the coarser grid node P . It means the node P can not get a proper solution vector from the finer grid, which can pose a serious problem while solving the dynamic equation (4.28). Following the boundary node projection algorithm proposed in [34], we resort to finding a projection node for node P in the nearest finer mesh cell face and then use this projected node to perform the necessary multigrid operations.

Figure 4.1 Schematic of boundary node projection algorithm, Ω is the computational domain

CHAPTER FIVE

IMMERSED MEMBRANE METHOD AND FLUID-STRUCTURE INTERACTION

This chapter presents detailed implementation of the immersed membrane method (IMM) and the fluid-structure interaction solution strategy on unstructured meshes. The core elements of this method are the definition of ghost nodes and one-sided extrapolations for simulating the discontinuities of the flow conditions across the fluid-structure interface. With the ghost fluid nodes, the mesh regeneration or deformation for fluid domain are avoided and the fluid-structure interfacial boundary conditions are imposed. Hence, the numerical cost for mesh treatment is greatly reduced as compared to the traditional arbitrary Lagrangian/Eulerian (ALE) method. With the help of our novel one-sided 3rd-order MUSCL like extrapolation scheme, sharp fluid-structure interfaces can be preserved and “jump” flow conditions across the fluid-structure interface can be accurately captured without smoothing, thus more accurate fluid solution can be obtained.

5.1 DEFINITION OF GHOST FLUID NODES

The immersed membrane method was initially developed to simulate the interaction between the fluids and immersed thin membranes. When a thin structure is immersed in a flow field, it causes discontinuous flow conditions across the structure. Although velocity is continuous, the gradient of velocity, pressure, gradient of pressure and fluid stresses are quite different, even having sharp changes over both sides of the thin structure. See Figure 5.1 for a 2D example of thin membrane immersed into a flow field. In order to account for the discontinuous fluid conditions across the membrane, a set of ghost fluid nodes are introduced. Specifically, in cell 267, edges 26 and 27 are cut by the membrane, $I1$ and $I2$ are the intersection points respectively. Now we use the velocities at node 2 and point $I1$ to extrapolate the velocity at node 6, and assign this extrapolated velocity to the ghost node $g26$, which locates at the same position as node 6. Furthermore, we can use the pressure/density and pressure/density gradients at node 2 to extrapolate the ghost node pressure/density at $g26$ as well. In a similar way, we can use the corresponding conditions at node 6 and $I1$ to extrapolate the ghost values at $g62$, which locates at the same location as node 2. While calculating the fluxes along edge 26, ghost values at $g26$ will be used in place of the real-node conditions of node 6 when the residual for node 2 is accumulated. Likewise, ghost values of $g62$ will be used in place of real-node conditions of node 2 when cumulating the residual for node 6. There is the possibility that a node may hold multiple ghost-node values, node 2 ($g62$ and $g72$) in this case. All of these possible ghost values should be stored and used accordingly, while calculating the respective fluxes along different edges. One should note that the ghost-node values should be only used while computing the fluxes along the edges cut by the membrane. For the normal edges, the real-node conditions should be used instead. For example, flow conditions at node 6 and 7 should be used while computing the

fluxes along edge 67.

Figure 5.1 The concept of Ghost Nodes and Ghost Values

The IMM is not only limited to the studies of immersed membrane structures; it can also be extended to study the fluid-structure interaction for arbitrary object shapes. For an arbitrary object immersed in a fluid as shown in Figure 5.2, it occupies a part of the fluid domain. Some fluid nodes will fall within the object domain, which are represented by hollow circles in the figure. For these fluid nodes, special treatment should be performed to ensure that they will not take part in the fluid calculation since physically there are no fluids inside the object domain for which a pre-processing subroutine is designed. It will identify those fluid nodes which fall in the domain of the object.

Figure 5.2 Treatment of the fluid nodes inside the immersed body; for ghost-node A, it can possess up to 3 different ghost values corresponding to fluid nodes A1, A2 and A3; for fluid node B, 3 ghost nodes B1, B2, B3 contribute to its flux computations.

Before the identifying process begins, it would be beneficial if we can narrow down the search region of the fluid mesh that might contain the immersed object mesh. A bounding box is defined around the object by the extrema of the object mesh coordinates. Only those fluid nodes fall into this box will be checked and they are stored into an array named “suspected array”. This simple treatment can greatly save the search time according to our experiences. Next, what we need to do is just performing a do-loop based on the number of the cells of the immersed object mesh. During each cycle of this loop, we test the “suspected array” nodes one by one to see if they fall into a certain cell of the object mesh. If yes, we can identify it as a ghost node. The concept of bounding box is also applied in this testing process to improve the efficiency of the algorithm. The “suspected array” nodes are first tested against this box before the full test algorithm is performed. Basically, the algorithm for inter-connectivity relationship

CHAPTER 5 – IMMERSSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

between the suspected nodes and object cell is based on the concept of dot product between two vectors: i) The unit normal vectors oriented inward from the triangular face, $\vec{n}$; ii) and the vector $\vec{p}$ pointing from the centre of the triangular face of the cell to the node to be tested. The dot product between these two vectors, $(p_n)^{nface}$, must be greater than or equal to zero if the angle, θ , subtended between them is less than 90° and less than zero if the angle is greater than 90° , as depicted in the following equation [150]:

$$(p_n)^{nface} = \{\vec{p}\} \cdot \{\vec{n}\} = \begin{Bmatrix} p_x & p_y & p_z \end{Bmatrix} \cdot \begin{Bmatrix} n_x \\ n_y \\ n_z \end{Bmatrix} = (p_x n_x + p_y n_y + p_z n_z)^{nface}, nface=1 \text{ to } 4, \quad (5.1)$$

where the superscript $nface$ is the triangular face number of the tetrahedral cell.

For ease of clarification and illustration, all the figures in this section will be in 2D. With reference to Figure 5.3a, the dot product between these two vectors must be positive for the three edges for 2D (four faces for 3D) if the suspected node is to fall within the structure cell. If one of the dot products between these two vectors is less than zero, then the node being tested is not within the cell, as depicted in Figure 5.3b. The following equation shows that the criterion for the node to be within the tetrahedral cell is

$$\left\{ (p_n)^1 .and. (p_n)^2 .and. (p_n)^3 .and. (p_n)^4 \right\} \geq 0. \quad (5.2)$$

CHAPTER 5 – IMMERSSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

In order to reduce the pre-processing CPU time further, those nodes that fall within a particular cell are marked with flag 1 and they will not be tested for the rest cells anymore.

Figure 5.3a Suspected node falls within a 2D immersed object triangle cell using dot product

Figure 5.3b Suspected node does not fall within a 2D immersed object triangle cell

CHAPTER 5 – IMMersed MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

After all the suspected nodes with a flag of 0 in the box are tested against this immersed object cell, then the next object cell in the list will be tested with those fluid nodes having a flag of 0 again and those having a flag of 1 are skipped. This process will stop if all the suspected nodes are tested and marked as 1 or if all the object cells are tested whichever one is completed earlier. Subsequently, those fluid nodes with a flag 1 will not be involved in flow calculations. Only those ghost nodes having a connecting edge, that passes through the fluid-structure interface will be assigned for ghost-node values and used to impose the immersed boundary conditions. For ease of clarification and illustration, we ignore those ghost-nodes which are not adjacent to the interface from now on. And when we mention ghost-nodes, we actually refer to those which contribute to flow computation.

5.2 INTERACTIONS BETWEEN FLUID AND STRUCTURE MESHES

As described in forgoing chapters, in the fluid solver the convection fluxes are computed based on edges of fluid mesh. As a result, every node can become multiple ghost nodes with their corresponding ghost values since a node can be connected by multiple edges, which is especially true for 3D unstructured grids. The selection of a particular ghost value depends on which edge and node the computation involves. This is one of the novel features of IMM. Hence, it is very crucial to find all of the edges of the fluid mesh cut by the fluid-structure interface in an accurate and efficient manner. Our solution algorithm will be described in the following paragraphs.

Because the object is fully immersed in the fluid, there should be no exception that

CHAPTER 5 – IMMersed MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

every node of the object mesh is contained by a certain cell of the fluid mesh. As the first step of our algorithm, the relationship between the surface nodes of the object mesh and their containing cells of the fluid mesh will be established. Note that the same searching procedure determining the ghost nodes in the former section may apply here.

After all of the surface nodes of the object mesh have been associated with their corresponding containing fluid cells, our algorithm starts by performing a do-loop over the number of the surface triangles of the object mesh. The basic operations involved in every iteration of the loop are outlined as follows:

- 1). If all of the three vertexes of the triangle locate in the same fluid cell, the iteration exit because there is definitely no edge of the fluid mesh intersecting with the current surface triangle;
- 2). If the two of the vertices locate in the same fluid cell while the third one does not, any one of the six edges of the fluid cell containing the third vertex might pass through the fluid-structure interface and the possible intersection points (if exists) will be searched using a separate subroutine, which will be given later;
- 3). If the three vertices reside in three different fluid cells, this becomes more complicated:
 - a. The three cells have two common nodes. Then it can be made sure that among all of the edges of these three fluid cells, there is only one which

CHAPTER 5 – IMMERSSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

passes through the fluid-structure interface, which is the edge connecting these two nodes.

- b. The three cells have one common node. Then all of the edges of these three cells originate from such a common node might pass through the fluid-structure interface, and they are checked one by one using a separate subroutine (explained in following pages).
- c. The three cells have no common nodes. Then all of the edges of these three cells are likely to pass through the fluid-structure interface. Hence they are also checked one by one using a separate subroutine (explained in following pages).

When the do-loop finishes its execution, most of the fluid edges cut by the interface are identified and marked with flag 1. But there could be some edges that are missed by the given scheme. As shown in Figure 5.4, the fluid cell 1234 contains no surface node of the object mesh at all, but three of its edges pass through the interface. Obviously, they will be omitted by the given searching scheme. Therefore, a remedy procedure is designed to rescue such edges in the end, in which all of the fluid cells cut by the interface are examined to see whether any of their edges are cut by the interface, while those edges with flag 1 will be skipped.

The following subroutine will find the intersection point (if it exists) between a line segment and a planar 3 vertex facet. The mathematics and solution can also be used to find the intersection between a plane and line, which is a simpler problem. The

CHAPTER 5 – IMMERSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

intersection between more complex polygons can be found by first triangulating them into multiple 3 vertex facets. Source code will be provided at the end, it illustrates the solution more than being written for efficiency.

Figure 5.4 Fluid cell that doesn't contain any of the surface nodes of the object mesh

The labeling and naming conventions for the line segment and the facet are shown in the following diagram

CHAPTER 5 – IMMERSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

Figure 5.5 Determination of the intersection between a line segment and a planar facet with three vertices

The procedure will be implemented given the line segment defined by its two end points and the facet bounded by its three vertices. The solution involves the following steps

- Check that the line and plane are not parallel;
- Find the intersection of the line with the plane containing the facet;
- Check that the intersection lies along the line segment;
- Check that the intersection lies within the facet;

CHAPTER 5 – IMMERSSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

The intersection P is found by substituting the equation for the line $\vec{P} = \vec{P}_1 + \mu(\vec{P}_2 - \vec{P}_1)$ into the equation for the plane $Ax + By + Cz + D = 0$.

Note that the values of A , B and C are the components of the normal unit vector to the plane which can be found by taking the cross product of any two normalized edge vectors, for example:

$$(A, B, C) = \frac{(\vec{P}_b - \vec{P}_a) \times (\vec{P}_c - \vec{P}_a)}{\left| (\vec{P}_b - \vec{P}_a) \times (\vec{P}_c - \vec{P}_a) \right|} \quad (5.3)$$

Then D is found by substituting one vertex into the equation for the plane:

$$AP_{ax} + BP_{ay} + CP_{az} = -D \quad (5.4)$$

This gives an expression for μ from which the point of intersection P can be found using the equation of the line.

$$\mu = (D + AP_{1x} + BP_{1y} + CP_{1z}) / (A(P_{1x} - P_{2x}) + B(P_{1y} - P_{2y}) + C(P_{1z} - P_{2z})) \quad (5.5)$$

CHAPTER 5 – IMMERSSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

If the denominator above is 0 then the line is parallel to the plane and they don't intersect. For the intersection point to lie on the line segment, μ must be between 0 and 1.

Lastly, we need to check whether or not the intersection point lies within the planar facet bounded by P_a , P_b and P_c . The method used here relies on the fact that the sum of the internal angles of a point on the interior of a triangle is 2π , points outside the triangular facet will have lower angle sums. As shown in Figure 5.6.

Figure 5.6 The sum of the internal angles of a point on the interior of a triangle is

$$2\pi$$

If we form the unit vectors $\vec{P}_{a1}$, $\vec{P}_{a2}$ and $\vec{P}_{a3}$ as follows (P is the point being tested to see if it is in the interior)

CHAPTER 5 – IMMERSSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

$$\begin{aligned}
\vec{P}_{a1} &= \frac{(\vec{P}_a - \vec{P})}{\left|(\vec{P}_a - \vec{P})\right|} \\
\vec{P}_{a2} &= \frac{(\vec{P}_b - \vec{P})}{\left|(\vec{P}_b - \vec{P})\right|} \\
\vec{P}_{a3} &= \frac{(\vec{P}_c - \vec{P})}{\left|(\vec{P}_c - \vec{P})\right|}
\end{aligned} \tag{5.6}$$

the angles are

$$\begin{aligned}
a_1 &= \arccos(\vec{P}_{a1} \cdot \vec{P}_{a2}) \\
a_2 &= \arccos(\vec{P}_{a2} \cdot \vec{P}_{a3}) \\
a_3 &= \arccos(\vec{P}_{a3} \cdot \vec{P}_{a1})
\end{aligned} \tag{5.7}$$

The source code is as follows:

```

/* Determine whether or not the line segment p1, p2
   Intersects the 3 vertex facet bounded by pa, pb, pc
   Return true/false and the intersection point P
   The equation of the line is p= p1+ μ (p2-p1)
   The equation of the plane is ax+by+cz+d=0
                                   nxx+nyy+nzz+d=0

*/
int LineFacet(p1,p2,pa,pb,pc,p)
XYZ p1,p2,pa,pb,pc,*p;
{
    double d;
    double a1,a2,a3;
    double total,denom,mu;

```

CHAPTER 5 – IMMERSSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

```

XYZ n,pa1,pa2,pa3;

/* Calculate the parameters for the plane */
n.x = (pb.y - pa.y)*(pc.z - pa.z) - (pb.z - pa.z)*(pc.y - pa.y);
n.y = (pb.z - pa.z)*(pc.x - pa.x) - (pb.x - pa.x)*(pc.z - pa.z);
n.z = (pb.x - pa.x)*(pc.y - pa.y) - (pb.y - pa.y)*(pc.x - pa.x);
Normalise(&n);
d = - n.x * pa.x - n.y * pa.y - n.z * pa.z;

/* Calculate the position on the line that intersects the plane */
denom = n.x * (p2.x - p1.x) + n.y * (p2.y - p1.y) + n.z * (p2.z - p1.z);
if (ABS(denom) < EPS)          /* Line and plane don't intersect */
    return(FALSE);
mu = -(d + n.x * p1.x + n.y * p1.y + n.z * p1.z) / denom;
p->x = p1.x + mu * (p2.x - p1.x);
p->y = p1.y + mu * (p2.y - p1.y);
p->z = p1.z + mu * (p2.z - p1.z);
if (mu < 0 || mu > 1)          /* Intersection not along line segment */
    return(FALSE);

/* Determine whether or not the intersection point is bounded by pa,pb,pc */
pa1.x = pa.x - p->x;
pa1.y = pa.y - p->y;
pa1.z = pa.z - p->z;
Normalise(&pa1);
pa2.x = pb.x - p->x;
pa2.y = pb.y - p->y;
pa2.z = pb.z - p->z;
Normalise(&pa2);
pa3.x = pc.x - p->x;
pa3.y = pc.y - p->y;
pa3.z = pc.z - p->z;
Normalise(&pa3);
al = pa1.x*pa2.x + pa1.y*pa2.y + pa1.z*pa2.z;

```

```

    a2 = pa2.x*pa3.x + pa2.y*pa3.y + pa2.z*pa3.z;
    a3 = pa3.x*pa1.x + pa3.y*pa1.y + pa3.z*pa1.z;
    total = (acos(a1) + acos(a2) + acos(a3)) * RTOD;
    if (ABS(total - 360) > EPS)
        return(FALSE);

    return(TRUE);
}

```

As it has been shown that the intersection will be returned if it exists. With the help of this information, the areas of the three sub-triangles (S_1 , S_2 and S_3 in Figure 5.6) with the common vertex P will be calculated and stored for late use.

5.3 ONE-SIDED MUSCL LIKE EXTRAPOLATION

In the beginning of every physical time step, the fluid solver (TETRAKE) is executed to solve the fluid domain. And during this stage, IMM is used to impose the boundary conditions across the fluid-structure interface. In this method, those fluid cells crossing the interface need special treatment. Take Figure 5.7 for an example, where a fluid cell 1234 is cut by the interface. Nodes 1 , 2 and 3 lie in the fluid domain while node 4 lies in the structural domain.

CHAPTER 5 – IMMERSSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

Figure 5.7 Schematic of Immersed Membrane Method and one-sided extrapolation

As described in former chapters, convection fluxes are computed based on mesh edges. In the computation of the convection flux along edge 14 , the flow conditions at node 1 and ghost conditions at node 4 ($g14$) will be involved. The ghost conditions of node 4 are determined as follows:

- 1). Identify the intersection point I_l between the edge $l4$ and the interface;
- 2). Identify the surface triangle in which intersection point I_l lies;
- 3). Determine the velocity $\vec{u}_l$ of intersection point I_l using a area weighted scheme with the knowledge of velocities of node a , b and c ;

CHAPTER 5 – IMMERSERD MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

4). Extrapolate the ghost velocity of node 4 using the flow condition of node 1 and the velocity of the intersection point I_l .

5). Extrapolate the ghost pressure and density of node 4 using the known pressure, density and gradient of node 1.

Extrapolations of ghost-node velocities are illustrated in Figure 5.8 (a) and (b). Taking node 1 as the real node, velocity at ghost node $g14$ is calculated as follows:

$$\begin{cases} \frac{\vec{u}_{g14} - \vec{u}_l}{\vec{u}_1 - \vec{u}_l} = -\frac{|r_{4l}|}{|r_{1l}|} & \text{for } 0 \leq |r_{4l}| \leq |r_{1l}| \\ \frac{\vec{u}_{g14} - \vec{u}_l}{\vec{u}_2 - \vec{u}_l} = -\frac{|r_{4l}|}{|r_{2l}|} & \text{for } |r_{1l}| < |r_{4l}| \end{cases} \quad (5.8)$$

Therefore,

$$\begin{cases} \vec{u}_{g14} = -\frac{|r_{4l}|}{|r_{1l}|} \cdot (\vec{u}_1 - \vec{u}_l) + \vec{u}_l & \text{for } 0 \leq |r_{4l}| \leq |r_{1l}| \\ \vec{u}_{g14} = -\frac{|r_{4l}|}{|r_{2l}|} \cdot (\vec{u}_2 - \vec{u}_l) + \vec{u}_l & \text{for } |r_{1l}| < |r_{4l}| \end{cases} \quad (5.9)$$

where $\vec{u}_l$, $\vec{u}_1$ and $\vec{u}_{g14}$ are the velocity vectors at intersection point I_l , node 1, ghost node $g14$. In Figure 5(b), we have $|r_{2l}| = |r_{4l}|$ and $|r_{1l}| = |r_{14}|$. $\vec{u}_2$ is evaluated as following:

$$\vec{u}_2 = \frac{(|r_{1l}| - |r_{12}|)\vec{u}_1 + (|r_{2l}| - |r_{1l}|)\vec{u}_l}{|r_{14}|}$$

CHAPTER 5 – IMMERSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

where $\vec{u}_I = \vec{u}_1 + \vec{r}_{41} \cdot \nabla \vec{u}_1$.

Figure 5.8 (a), (b) Velocity extrapolation for ghost node 4;
(c) Pressure and density extrapolation for ghost node 4.

In this work, the structure and fluid domains are coupled by enforcing the velocity continuity condition at the interface,

$$\vec{u}_s = \vec{u}_f \quad (5.10)$$

Where $\vec{u}_s$ is the velocity vector of a structure surface node, $\vec{u}_f$ is the neighbouring fluid velocity at the same location. And Equation (5.9) is used to extrapolate fluid velocity to its corresponding ghost nodes. The extrapolation of ghost-node pressure is

CHAPTER 5 – IMMERSSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

illustrated in Figure 5.8(c). Taking node 1 as real node, the ghost-node pressure and density of node 4 is obtained by

$$\begin{aligned} p_{g14} &= p_1 + \vec{r}_{14} \cdot \nabla p_1 \\ \rho_{g14} &= \rho_1 + \vec{r}_{14} \cdot \nabla \rho_1 \end{aligned} \quad (5.11)$$

where p_{g14} and ρ_{g14} are the ghost-node pressure and density at node 4 , ∇p_1 and $\nabla \rho_1$ are the pressure and density gradients at node 1 , $\vec{r}_{14}$ is the distance vector pointing from node 1 to node 4 . Now all of the necessary ghost conditions for determining the convection flux along edge 14 are computed. Suppose edge 24 is the next edge along which the convection flux will be computed. We need to determine the ghost conditions for node 4 ($g24$) again with the knowledges of node 2 and intersection node I_2 . As a result, every node can hold multiple ghost nodes and thus multiple ghost values since a node can be connected by multiple edges, which is especially true for 3D unstructured grids. The selection of a particular ghost value depends on which edge and node the computation is for. This is a novel feature of the current implementation of unstructured IMM.

This linear interpolation results in a second-order accurate scheme. It is noted that higher-order MUSCL interpolation can also be applied during the flux computations if higher order accuracy is required. As depicted in Figure 5.9, edge 12 is cut by fluid-structure interface and I is the intersection point. C is the centre of edge 12 . Local 3rd-order accuracy can be achieved by introducing $\vec{u}_L$ and $\vec{u}_R$ for the two sides of C :

CHAPTER 5 – IMMersed MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

$$\begin{aligned}\vec{u}_L &= \vec{u}_1 + \frac{1}{4}[(1-\kappa)\Delta_1^- + (1+\kappa)\Delta_1^+] \\ \vec{u}_R &= (\vec{u}_1 + \vec{u}_{g12})/2\end{aligned}\tag{5.12}$$

where

$$\begin{aligned}\Delta_1^+ &= \vec{u}_{g12} - \vec{u}_1 \\ \Delta_1^- &= 2 \cdot \vec{12} \cdot \nabla \vec{u}_1 - (\vec{u}_{g12} - \vec{u}_1) = 2 \cdot \vec{12} \cdot \nabla \vec{u}_1 - \Delta_1^+\end{aligned}\tag{5.13}$$

Figure 5.9 Introduction of higher-order MUSCL like interpolation

And $\vec{u}_{g12}$ is the ghost velocity vector of node 2, $\nabla \vec{u}_1$ then velocity gradient at node 1.

$\vec{12}$ is the vector representing the edge. When κ is set to 1/3, which corresponds to a nominally third-order accuracy. Pressure and density can be interpolated in a similar way. The viscous fluxes and gradients are computed based on fluid mesh cells. In the cell N in Figure 5.7, when the viscous flux is computed for node 1, the flow conditions at nodes 1, 2 and 3 will be needed, as well as the ghost-node conditions at node 4 (It is $g14$ in this case. But it would be $g24$ if the viscous flux for node 2 is computed).

5.4 IMM BASED FLUID-STRUCTURE INTERACTION

In previous chapters we have already developed a novel parallel unstructured multi-grid preconditioned compressible Navier-Stokes solver TETRAKE. Here we aim to extend it for fluid-structure interaction simulation. The biggest challenge here is how to couple the two modules and how to synchronize them. Our solution algorithm is shown in Figure 5.10, from which it may be seen that different time stepping size between the fluid and the structure solvers are allowed. In the beginning of every physical time step, the fluid solver (TETRAKE) is executed to solve the fluid domain. And during this stage, IMM is used to impose the boundary conditions across the fluid-structure interface using the techniques introduced in the foregoing sections. After the computation of fluid domain, fluid forces acting on the structure are calculated along the fluid-structure interface and they are applied to advance the movement of the structure. In the structural domain, equation (4.1) is solved by the techniques described in former sections with the fluid forces exerted by ambient fluids as boundary conditions. Boundary conditions on the surface Γ_s of the structural domain Ω_s are described in terms of prescribed traction $\vec{t}_p$ on the boundary Γ_t and prescribed displacement $\vec{d}_p$ on the boundary of the structure Γ_d .

CHAPTER 5 – IMMERSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

Figure 5.10 Schematic of our FSI solution

Just as described in Equation (4.8). At the fluid-structure interface the structure experiences a surface traction due to the fluid; hence the appropriate boundary condition

for the structure at the fluid-structure interface is expressed as:

$$t_p = -p\delta_{ij} + \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} \mu \nabla \cdot \vec{U}_f \delta_{ij} \quad \text{on } \Gamma_i, (i, j=1, 2, 3) \quad (5.14)$$

where μ is the dynamic viscosity, p the pressure, and u_i the Cartesian component of fluid velocity $\vec{U}_f$ corresponding to direction x_i . In IMM, the loosely coupling relationship between the fluid domain and the structure domain is depicted in Figure 5.11. The fluid forces include pressure p , shear stress σ_t and normal stress σ_n . Since the fluid mesh and structural are non-confirmatory, the fluid pressure needs to be extrapolated to the fluid-structure interface for the computation of fluid forces. And again this extrapolation is edge-based. In Figure 5.7, I_3 is the intersection point between edge 34 and the interface, and we need to extrapolate the pressure of this point to calculate the fluid force. This extrapolation procedure can be expressed as following:

$$p_{I_3} = p_3 + \vec{r}_{3I} \cdot \nabla p_3 \quad (5.15)$$

where p_{I_3} is the fluid pressure on the fluid-structure interface on the side of vertex 3, ∇p_3 are the pressure gradient at vertex 3, $\vec{r}_{3I}$ is the distance vector pointing from node 3 to point I_3 . This linear extrapolation leads to a second order accuracy. Note that the higher-order MUSCL scheme similar to equation (5.12 and 5.13) can also apply here.

CHAPTER 5 – IMMERSSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

Figure 5.11 Relation between the structure domain and the fluid domain in fluid-structure interaction

And then the resultant pressure will be distributed to the structure nodes d , e and f based on an area weighted scheme. For Newtonian fluids, the fluid stress tensor is given as follows:

$$\bar{\bar{t}} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix} \quad (5.16)$$

and

$$\begin{aligned} \sigma_{xx} &= 2\mu \frac{\partial u}{\partial x} - \frac{2}{3}\mu \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right), \sigma_{xy} = \sigma_{yx} = \mu \left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \right) \\ \sigma_{yy} &= 2\mu \frac{\partial v}{\partial y} - \frac{2}{3}\mu \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right), \sigma_{xz} = \sigma_{zx} = \mu \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right) \\ \sigma_{zz} &= 2\mu \frac{\partial w}{\partial z} - \frac{2}{3}\mu \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \right), \sigma_{yz} = \sigma_{zy} = \mu \left(\frac{\partial w}{\partial y} + \frac{\partial v}{\partial z} \right) \end{aligned} \quad (5.17)$$

where the molecular viscosity μ is property of the fluid and is function of temperature. So the fluid stress σ can be calculated as the following:

$$\sigma = \bar{\bar{t}} \cdot \vec{n} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix} \cdot \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} \quad (5.18)$$

where $\vec{n}$ is the unit normal vector of the structure element. In the fluid solver, the velocity gradients are assumed to be constant within one fluid cell. It means that on the fluid-structure interface the gradients of fluid velocities are equal to those at the cell centroids. But for a cell intersected by the immersed structure, the gradients are only uniform and continuous for centroids on the same side of the structure. So the computations of fluid stresses on the interface should use the right gradients of velocity. Use edge 34 in Figure 5.7 for example, the fluid stresses at intersection point I_3 should be computed using the gradients of vertex 3, i.e.:

$$\sigma_3 = \bar{\bar{t}}_3 \cdot \vec{n} = \begin{bmatrix} \sigma_{3,xx} & \sigma_{3,xy} & \sigma_{3,xz} \\ \sigma_{3,yx} & \sigma_{3,yy} & \sigma_{3,yz} \\ \sigma_{3,zx} & \sigma_{3,zy} & \sigma_{3,zz} \end{bmatrix} \cdot \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} \quad (5.19)$$

As can be seen from Figure 5.10, there exists a special step at the end of each time step of structure computation, or at the beginning of each time step of fluid computation, namely ‘treatment of freshly cleared fluid nodes’.

In some Immersed Boundary Methods where the immersed boundaries are treated as sharp interfaces, such as those presented by Bayyuk et al. [145], Jianming Yang [146] and Udaykumar et al. [147], the issue of change of material needs to be addressed. This arises when a computational point (as in Figure 5.12), which was in the solid at one time

CHAPTER 5 – IMMERSED MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

step, emerges into the fluid at the next time step. In [146], because the fractional time advancement scheme is used, the evaluation of the *RHS* of the momentum equation at step $k+1$ requires physical values of the velocity vector and pressure, as well as their derivatives from step k at all fluid points. Due to the fact that the interface changes locations, it is possible that some of the required values from step k are not physical. To avoid forbiddingly complex special treatments of the computation of these derivatives every time such a change of flag is detected during the time advancement procedure, they propose a field-extension procedure, in which the velocity and pressure fields are “extended” in the solid phase at the end of each step. Practically, with this procedure the velocity and pressure fields are extrapolated at the Eulerian nodes in the solid that have at least one neighbor in the fluid phase. In the cut-cell formulation of Udaykumar et al. [147] this issue of *freshly cleared cells* was addressed using a cell-merging formulation in conjunction with quadratic interpolation among neighboring grid nodes in the fluid. In the current method, an approach similar to the field-extension procedure used in the finite-difference method of Jianming Yang [146] is employed. It is comprised of several basic substeps as described as follows. In the first substep, all of the computational nodes which locate inside the solid domain are successively ‘*leveled*’. The level tagging process is depicted in Figure 5.12. All ghost nodes are tagged as ‘*level 1*’ nodes, which are grid points in the solid phase that have one or more neighboring points in the fluid phase (neighboring here means connected by an edge). ‘*Level 2*’ nodes are grid points in the solid that have one or more ghost nodes as their neighboring points. ‘*Level 3*’ nodes are grid points in the solid that have one or more ‘*Level 2*’ nodes as their neighboring points (except ‘*level 1*’ points). The rest may be deduced by analogy.

Figure 5.12 A two-dimensional example that demonstrates the possible changes of flags of the grid nodes near a moving interface. The classification of the various nodes is according to the notation introduced in the text and is based on the location of the body at time k . In this special case, all of the ‘level 1’ nodes (ghost nodes) will emerge into the fluid at time $k+1$, and all of the ‘level 2’ nodes will become ‘level 1’ nodes at time $k+1$ and so on.

In the second substep of our approach, for every ‘level 1’ point, the labels of the neighboring points in the fluid phase are stored, as well as the lengths of the edges connecting the current point and the neighboring points in the fluid phase. For every ‘level 2’ point, the labels of the neighboring ‘level 1’ points are stored, as well as the lengths of the edges connecting the current point and the neighboring ‘level 1’ points. The same treatment will be performed for every point of successive levels. Finally, the velocity and pressure fields are “extended” in the solid through the levels. The inverse distance weighted method proposed by [148] is adopted in this study to perform the

CHAPTER 5 – IMMersed MEMBRANE METHOD AND FLUID STRUCTURE INTERACTION

“field extension” task. This method has the property of preserving local maxima and producing smooth reconstruction. The interpolation at a certain location (x, y, z) is

$$\begin{aligned}\phi(x, y, z) &= \sum_{m=1}^n w_m \phi_m / q \\ w_m &= \left(\frac{R - h_m}{R h_m} \right)^p \\ q &= \sum_{l=1}^n \left(\frac{R - h_l}{R h_l} \right)^p\end{aligned}\tag{5.20}$$

where ϕ_m represents the solution at a certain location, w_m the weight, and h_m the distance between the location (x, y, z) and the location of ϕ_m . R the maximum of all h_m ; p is a constant and normally set to 2. The extension process can be described as follows. The velocity and pressure fields of the fluid nodes neighboring to the solid interface are interpolated to the “Level 1” points using Equations (5.20). And then the velocity and pressure fields of the “Level 1” points are interpolated to the “Level 2” points using Equations (5.20) again and so on for successively levels. Generally, 3-level extension is enough, provided that the *CFL* number has not been set to an extraordinarily large value. As a result, not only the velocity and pressure at the newly emerging points, but also their derivatives will have physical values, eliminating problems in the computation of the *RHS* in the next time step for these points.

CHAPTER SIX

THE MULTIGRID METHOD AND PARALLEL COMPUTATION

In this chapter, the basic concepts of the MG method and parallel computation employed in this work are briefly explained and the mathematical formulations are presented. In this work, the MG method is applied to the solution of the time-dependent compressible Navier-Stokes equations on unstructured grid. An efficient and robust data structure for different MG level is described. To enhance the efficiency of the flow solver and reduce pre-processing time, the algorithm for zoning of the flow field domain is explained. The implementation of parallel in this study using the geometric domain decomposition technique based on the Single Program Multiple Data (SPMD) programming paradigm.

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

6.1 THE MULTIGRID METHOD

The philosophy of the multigrid method is to carry out early iterations on a fine grid and then progressively transfer these flow field variables and residuals to a series of coarser grids. Thus, the solutions on the coarser grids are driven by the fine grid residuals. Since the coarser grids have fewer grid points, fewer computations are needed for a given sweep of the flow field, and hence computer time is saved and efficiency is increased. On the coarser grids, the low frequency errors become high frequency errors and these errors can be easily eliminated on the coarser grids. The flow equations are then solved on the coarser grids and the correction, the difference between the newly computed coarse grid values and the transferred values, is then interpolated back to the fine grid. The purpose of this correction is to correct the solution on the fine grid, thus eliminating its low frequency errors. The process is repeated over a sufficient number of times until satisfactory convergence on the fine grid is achieved.

The advantage of the multigrid method is associated with the enhanced damping of numerical errors in the flow field. In a one-dimensional space of length L along the x -axis, suppose it is divided into M cells, with $\Delta x = L/M$, the wavelength of an error can vary from the smallest value of $\lambda_{min} = 2\Delta x$ to the largest value of $\lambda_{max} = L$. Errors associated with wavelengths near λ_{min} are called *high-frequency errors*, and those associated with wavelengths near λ_{max} are called *low-frequency ones* [149]. For a stable solution, the errors of all frequencies will generally be damped out during the course of the iterative process. However, in most cases, the high-frequency errors are reduced much faster than the low-frequency errors. Therefore, the speed of convergence would be enhanced if something could be done to increase the damping of these low-frequency

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

errors. After performing a few iterations on a fine grid, a large part of the high-frequency error components will generally be damped as shown in Figure 6.1. As a consequence, an iterative method with these properties will act as a *smoother* of the errors. In this way, after one or more iteration sweeps through the fine grid, known as *relaxation sweeps*, the errors are sufficiently smoothed for them to be adequately represented on a coarse grid. Due to a larger Δx in the coarser grid and hence larger $\lambda_{min} = 2\Delta x$, low-frequency errors begin to be damped at a faster rate than it would have taken place in the fine grid and by going progressively to coarser grids, the low-frequency errors are more readily damped. Thus, the low-frequency errors of the intermediate coarse grid results are smaller than they would have been for an equal number of sweeps just on the fine grid itself.

Figure 6.1 Smoothing of errors by relaxation sweeps

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

Two different cycle strategies had been investigated in Tai's work [150], which are the V-cycle and W-cycle. Figure 6.2 depicts the differences between the two strategies. For one complete V-cycle, the flow is interpolated back up to the two finer grids once the coarsest grid is reached, with one new evaluation of the flow field variables on the fine grid. For one complete W-cycle, the flow is interpolated up one grid after performing the iteration on the coarsest grid. Then the flow field variables are evaluated on this grid and the variables are transferred down to the coarsest grid again. The flow is solved on the coarsest grid and the correction is then interpolated up to the fine grid.

The total computational work per multigrid cycle relative to the work on the fine grid for both V-cycle and W-cycle using quadrilateral cells are as follows [150, 151]:

$$\text{V-cycle: } W_{MG} < \frac{4}{3} W_{fine}$$

$$\text{W-cycle: } W_{MG} < 2 W_{fine}$$

Figure 6.2a V multigrid cycles for three grid levels

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

Figure 6.2b W multigrid cycles for three grid levels. $\textcircled{E}$ denotes flow field computation on a particular level and $\textcircled{I}$ denote interpolation of corrections of flow field variables.

6.1.1 Mathematical Formulations

In this study, a three level multigrid is used. But in this section, a two level multigrid is used to describe the formulations in this work. Fine grid, h , denotes the first level, whereas coarse grid, $h + 1$, denotes the second level of the multigrid. The solution on the coarse grid is initialized by transferring the flow field variables from the fine grid using the transfer operator T_h^{h+1} :

$$W_{h+1}^{(0)} = T_h^{h+1} W_h \quad (6.1)$$

where $W_{h+1}^{(0)}$ is the initial values transferred from the fine grid, T_h^{h+1} is a transfer operator which will be defined in Section 6.5 and W_h is the solution from the fine grid.

And the initial residual on the coarse grid is defined as follows:

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

$$\tilde{R}_{h+1}^{(0)} = Q_h^{h+1} \tilde{R}(W_h) \quad (6.2)$$

where $\tilde{R}_{h+1}^{(0)}$ is the initial residual transferred from the fine grid, Q_h^{h+1} is the residual transfer operator based on weighted average which will be defined in Section 6.5 and $\tilde{R}(W_h)$ is the residual from the fine grid. The transfer operators in Equations (6.1) and (6.2) are *restriction* operators that may not necessarily be the same.

In order to use fine grid residuals to drive the solution on the coarser grids, a forcing function is calculated at the first stage of the Runge-Kutta scheme and subsequently, this forcing term was added to the residual at the coarse grid. The forcing function on the coarse grid is defined as follows:

$$P_{h+1} = \tilde{R}_{h+1}^{(0)} - \tilde{R}(W_{h+1}^{(0)}) \quad (6.3)$$

It should be noted that on coarse grids, time-dependent terms in the residual containing W at n and $n-1$ time levels are not included for ease of calculation. Instead, they are only included in the fine-grid residual and directly transferred to the coarse grids. And every stage of the Runge-Kutta scheme in Equation (3.41) on the coarse grid is modified to include the forcing term, P_{h+1} as follows:

$$W_{h+1}^{(m)} = W_{h+1}^{(0)} - \alpha_m C \frac{\Delta \tau}{\Delta S_{cv}} \left[\tilde{R}(W_{h+1}^{(m-1)}) + P_{h+1} \right] \quad (6.4)$$

where m denotes each stage of the Runge-Kutta scheme. Substituting the forcing term

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

into the above equation, it can be seen that $\tilde{R}(W_{h+1}^{(0)})$ and $\tilde{R}(W_{h+1}^{(m-1)})$ almost cancel each other if the reduction of residual is small, and $Q_h^{h+1}\tilde{R}(W_h)$ is the only term remains. Thus, this will ensure that the residual on the fine grid drives the coarse grid solution.

The solution and residuals are transferred onto successive independent coarser grids until the coarsest grid is reached. And after calculating the flow field variables on the coarsest grid, the correction is evaluated and then has to be interpolated back to the fine grid. The correction is the difference between the newly computed value on the coarse grid, W_{h+1}^+ , and the initial value that was transferred from the finer grid, $W_{h+1}^{(0)}$, and this correction is added to the values on the fine grid as follows,

$$W_h^+ = W_h + I_{h+1}^h(W_{h+1}^+ - W_{h+1}^{(0)}) \quad (6.5)$$

where I_{h+1}^h is an *prolongation* operator from the coarse grid to the fine grid, which will be defined in Section 6.5. W_h^+ is the updated variables of the solution and W_h is the solution from the fine grid.

In this study, in order to improve the efficiency for the simulation of viscous flows, the viscous terms are only evaluated on the fine grid and not evaluated at the coarser grids. Since the coarser grids are used to cancel the dominating low frequency errors on the fine grid and the solution on the coarser grids are driven by the residual from the fine grid, hence, this treatment does not affect the accuracy of the solution. The upwind-biased discretization scheme is set to first-order in the coarser levels where the

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

left-right states of Equation (3.34a) and (3.34b) are taken as the values of the two nodes of the edge to calculate the flux associated with the edge.

6.1.2 Data Structure for Different Multigrid Level

An efficient and robust data structure and dynamic memory allocation for different number of multigrid levels is employed in this work [151]. A two level multigrid is used to describe the data structure implemented in the solver. The general idea of this data structure is that the data of fine grid is stacking on top of the data of coarse grids. This requires an indexing array pointer to point to the starting location of a particular multigrid level array and this indexing array is a one-dimensional array. The indexing array pointer is established by getting the number of data for each multigrid level, for instance, the number of nodes and elements for fine and coarse grids, and the starting location of fine grid is 1, whereas for coarse grid is the number of data for fine grid plus one.

With the total number of nodes and elements for all MG level known, those arrays related to it can be dynamically allocated to the memory in a more efficient and memory saving way. The indexing array pointer is used as the parameters of the counting loop; they control the values of the loop index during execution of the loop.

6.1.3 Zoning of Flow Field Domain

In order to enhance the efficiency of the flow solver and reduce pre-processing time, the flow field domain is virtually covered by grid of cubes, forming a background Cartesian

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

mesh, in which searching for a particular node is only done within the cubes, instead of searching the entire whole flow field [150, 152]. For illustration purposes, Figure 6.3 shows a 2D flow field domain being covered by a 4×4 grid [150]. Increasing the number of cells or zones will increase the efficiency of the solver, but as for odd-shaped geometry with a lot of curvatures, increasing the number of cells or zones will not enhance the efficiency of the solver further.

Figure 6.3 Zoning of flow field domain [150]

The idea of this algorithm is that if any node of the triangular (2D)/tetrahedral (3D) cell is mapped onto a particular zone as shown in Figure 6.3, then this cell is considered mapped to this zone. To exam the detailed design and efficiency of the algorithm,

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

readers can refer to reference [150].

6.1.4 Inter-Connectivity Relationship between Meshes

Before the flow field variables and residuals are transferred from the fine grid to the coarse grid or the corrections are interpolated from the coarse grid back to the fine grid, it is necessary to determine in which coarse cell each fine node is located and vice-versa. In this work, the multigrid method makes use of a sequence of independently generated unstructured grids and the new algorithm used to determine in which coarse cell each fine node is located and vice-versa was developed here for the purpose of this study. In the pre-processing stage, finding nodes in tetrahedral cells often benefits from having an imaginary bounding box. The node is first tested against this box and having a flag marked as 0 before the full algorithm is employed to detect the node within the cell. The purposes of enclosing the cell within the box and marked the nodes as 0 are to narrow down the search area for the nodes in the same zone and not to repeat the algorithm for a node marked as 1, thus enhance the efficiency of the flow solver. Check reference [150] for detailed strategy design and mathematical formulations.

6.1.5 Mesh-to-Mesh Transfer Operators

Following the two-dimensional approach presented in [153] for data transfer within the domain and incorporating the new technique developed here for a curved boundary, there are two classes of mesh-to-mesh transfer operators being implemented in the present study for three-dimensions, which are *restriction* operators and *prolongation*

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

operators.

6.1.5.1 Flow Field Variables Transfer Operators

Once the inter-connectivity relationship between meshes has been determined, the *restriction* transfer operator, T_h^{h+1} , that transfers the flow field values from the fine grid to the coarse grid will be as follows:

$$W_1 = \frac{V_a W_a + V_b W_b + V_c W_c + V_d W_d}{V_a + V_b + V_c + V_d} \quad (6.6)$$

Figure 6.4a depicts the operation of this transfer operator for those nodes within a cell. Lower case letters denote fine grid nodes and Arabic numbers denote coarse grid nodes for all the figures in this section. The flow field values at the coarse node 1, which is contained in the fine cell formed by nodes a , b , c and d , is a weighted average of the values at those nodes. V_a is the volume of the sub tetrahedron cell with vertices b , 1 , c , and d . V_b is the volume of the sub tetrahedron cell with vertices a , 1 , c and d . V_c is the sub tetrahedron cell with vertices a , 1 , d and b . V_d is the sub tetrahedron cell with vertices a , 1 , b and c . The volume of the corresponding tetrahedron cell is the one opposite to the node. According to Equation (6.6), if node 1 coincides with node a , then W_1 will be equal to W_a .

At the boundary with curvature, the flow field values are transferred from the fine nodes of the triangle formed by vertices a , b and c onto the coarse node 1 using an area-weighted contribution calculation, as illustrated in Figure 6.4b. The *restriction* transfer operator, T_h^{h+1} , that transfers the flow field values from the fine grid to the coarse grid is as follows:

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

$$W_1 = \frac{A_a W_a + A_b W_b + A_c W_c}{A_a + A_b + A_c} \quad (6.7)$$

Figure 6.4a Transfer of flow field values from the fine mesh to the coarse mesh using *restriction* transfer operator

Figure 6.4b Transfer of variables from fine nodes to the coarse node at the boundary

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

Figure 6.4b illustrates the operation of this transfer operator for those nodes within the boundary triangles. The flow field values at the coarse node I , which is projected downward onto the fine triangle face formed by nodes a , b and c , is a weighted average of the values at those nodes. A_a is the area of the triangle with vertices b , I , and c . A_b is the area of the triangle with vertices c , I , and a . A_c is the area of the triangle with vertices a , I , and b . The area of the corresponding triangle is the one opposite to the node. Similarly, according to Equation (6.7), if node I coincides with node a , then W_I will be equal to W_a .

6.1.5.2 Residuals Transfer Operators

In this work, the *restriction* transfer operator is used for transferring residuals so that the transferred residuals will be conserved. Tai's report [150] has illustrated the reason why using *prolongation* operators to transfer residuals will not be conserved, which will not be repeated here.

The transfer of residuals from the fine grid to the coarse grid is based on a weighted average of the residual from the fine node to the four coarse nodes that form the coarse tetrahedral cell. As depicted in Figure 6.5a, the residual from node a is transferred to coarse node 1, 2, 3, and 4 by the transfer operator, Q_h^{h+1} . The transferred residual to a particular coarse node is the summation of all the transferred weighted average residuals from other fine nodes in those coarse-mesh tetrahedra surrounding this coarse node, as shown in following equation:

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

$$\begin{aligned}\tilde{\tilde{R}}_1 &= \tilde{R}_1 + \frac{V_1 \tilde{\tilde{R}}_a}{V_1 + V_2 + V_3 + V_4}; & \tilde{\tilde{R}}_2 &= \tilde{R}_2 + \frac{V_2 \tilde{\tilde{R}}_a}{V_1 + V_2 + V_3 + V_4}; \\ \tilde{\tilde{R}}_3 &= \tilde{R}_3 + \frac{V_3 \tilde{\tilde{R}}_a}{V_1 + V_2 + V_3 + V_4}; & \tilde{\tilde{R}}_4 &= \tilde{R}_4 + \frac{V_4 \tilde{\tilde{R}}_a}{V_1 + V_2 + V_3 + V_4}\end{aligned}\quad (6.8)$$

where V_1 is the volume of the sub tetrahedron cell with vertices 2, a , 3 and 4. V_2 is the

Figure 6.5a Transfer of residuals from the fine mesh to the coarse mesh using
restriction transfer operator

volume of the sub tetrahedron cell with vertices 1, a , 3 and 4. V_3 is the volume of the sub tetrahedron cell with vertices 1, a , 2 and 4. V_4 is the volume of the sub tetrahedron cell with vertices 1, a , 2 and 3. The volume of the corresponding tetrahedron cell is the one opposite to the node. It is easy to show that this transfer is conservative in the sense that the total fine mesh residuals are equal to the coarse mesh ones.

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

The residuals at the boundary faces of the curvature geometry are transferred to the coarse nodes of the triangle formed by vertices 1, 2 and 3 from the fine node a using an area-weighted contribution calculation. The transfer operator, Q_h^{h+1} , that transfers the residuals from the fine grid to the coarse grid is as follows:

$$\begin{aligned}\tilde{\tilde{R}}_1 &= \tilde{R}_1 + \frac{A_1 \tilde{\tilde{R}}_a}{A_1 + A_2 + A_3}; \\ \tilde{\tilde{R}}_2 &= \tilde{R}_2 + \frac{A_2 \tilde{\tilde{R}}_a}{A_1 + A_2 + A_3}; \quad \tilde{\tilde{R}}_3 = \tilde{R}_3 + \frac{A_3 \tilde{\tilde{R}}_a}{A_1 + A_2 + A_3}\end{aligned}\quad (6.9)$$

Figure 6.5b Transfer of residuals from fine node to the coarse nodes at the boundary

Figure 6.5b depicts the operation of this transfer operator for those nodes within the boundary faces. The residuals at the fine node a , which is projected downward onto the coarse triangle formed by nodes 1, 2 and 3, is a weighted average of the value at this

CHAPTER 6 — THE MULTIGRID METHOD AND PARALLEL COMPUTATION

fine node. The transferred residual is the summation of all the transferred residuals from other fine nodes to these coarse nodes. This also ensures that the residual transfer is conservative on the boundary. In this work, it has been proven that the total fine-grid residuals are equal to the coarse grid ones and this is achieved by performing a summation of the residuals on both fine and coarse grids and the totals are equal to each other. This proved that this *restriction* transfer operator is conservative.

6.1.5.3 Correction Transfer Operators

Prolongation operators are used to transfer corrections of the flow field variables from the coarse mesh to the fine mesh, as depicted in Figure 6.6. As shown in Equation (6.10), the correction, dW_{h+1} , is the difference between the newly computed value on the coarse grid, W_{h+1}^+ , and the initial value that was transferred from the fine grid, $W_{h+1}^{(0)}$.

$$dW_{h+1} = W_{h+1}^+ - W_{h+1}^{(0)} \quad (6.10)$$

The transferred corrections are transferred to the fine mesh by the *prolongation* operator, I_{h+1}^h :

$$v_h = I_{h+1}^h dW_{h+1}$$

and

$$W_h^+ = W_h + v_h$$

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

According to Figure 6.6a, the correction of the flow field variables transferred from the coarse nodes 1, 2, 3 and 4 to the fine node a is a weighted average of the corrections at these nodes and the expression for the transferred correction is as follows:

$$(v_a)_h = \frac{V_1(dW_1)_{h+1} + V_2(dW_2)_{h+1} + V_3(dW_3)_{h+1} + V_4(dW_4)_{h+1}}{V_1 + V_2 + V_3 + V_4} \quad (6.11)$$

where V_1 is the volume of the sub tetrahedron cell with vertices 2, a , 3 and 4. V_2 is the volume of the sub tetrahedron cell with vertices 1, a , 3 and 4. V_3 is the volume of the sub tetrahedron cell with vertices 1, a , 2 and 4. V_4 is the volume of the sub tetrahedron cell with vertices 1, a , 2 and 3. The volume of the corresponding tetrahedron cell is the one opposite to the node.

Figure 6.6a Transfer of corrections from the coarse mesh to the fine mesh using
prolongation transfer operator

The corrections of the flow field variables at the boundary are transferred from the

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

coarse nodes of the triangle formed by vertices 1 , 2 and 3 onto the fine node a . The transfer operator, I_{h+1}^h , that transfers the corrections from the coarse grid to the fine grid is as follows:

$$(v_a)_h = \frac{A_1(dW_1)_{h+1} + A_2(dW_2)_{h+1} + A_3(dW_3)_{h+1}}{A_1 + A_2 + A_3} \quad (6.12)$$

Figure 6.6b shows the operation of this transfer operator for those nodes within the boundary faces. The correction at the fine node a , which is projected downward onto the coarse face, formed by nodes 1 , 2 and 3 , is a weighted average of the values at those nodes. A_1 is the area of the triangle with vertices 2 , a , and 3 . A_2 is the area of the triangle with vertices 1 , a , and 3 . A_3 is the area of the triangle with vertices 1 , a , and 2 . The area of the corresponding triangle is the one opposite to the node. Similarly, according to Equation (6.12), if node a coincides with node 1 , then $(v_a)_h$ will be equal to $(dW_1)_{h+1}$.

Figure 6.6b Transfer of corrections from the coarse nodes to the fine node at the boundary

6.2 PARALLEL COMPUTATION

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

The quest for greater computing power has driven the development of codes using parallel machines as well as the computing hardware itself. A parallel computation consists of one or more tasks that execute concurrently. Numerical simulation of fluid flow problems is one of the most computationally intensive engineering tasks. Hence, in order to obtain short turn-around times, it is vital to use and exploit both parallel shared and distributed memory computer architectures. From a development point of view, it is critical to select numerical algorithms that lend themselves to parallelization and to implement them in a *scalable* fashion. In this context, '*scalable*' means that computational speed increases linearly with the number of processors for a given problem size (speed-up), and remains constant if the problem size is increased by the same factor as the number of processors (scale-up).

The main objective of this part of the work is to extend the algorithms and strategies, which were developed in Zhao and Tai's early work [150], to our new 3D Navier-Stokes compressible solver. These techniques would enable the 3D Navier-Stokes solver based on *unstructured grids* to exploit the computational advantages offered by SGI Origin 2000 and 3400, based on Silicon Graphics Scalable Distributed-Shared-Memory (DSM) Multiprocessing architecture. In their early works [150], Zhao and Tai have successfully extended the capabilities of these structured grid codes to unstructured grid codes. This work focuses on examining the one-level method of parallelization strategies by geometric domain decomposition technique. Check reference [150] for a detailed discussion of two-level implementations.

This part is arranged as follow. Section one is devoted to the basic concepts of parallel computation using MPI employed in this work. Section two outlines parallel strategies

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

used to decompose the geometric domain for both single grid (SG) and multigrid (MG) methods. Section three describes the geometric domain decomposition technique, the developed new algorithm used to identify the *ghost* nodes and overlapping elements in order to build the control volume for the finite-volume formulation and the data structure that define what node variables that will be sent to and received from which processors. Section four outlines the algorithms and techniques used to parallelize the multigrid method. Finally, Section five defines the methods used to measure the performance of the parallel 3D code.

6.2.1 One-level and Two-level Parallelization Strategies

The parallel supercomputer architecture of SGI Origin 3400 has the characteristics of both distributed and shared memory. This machine has a single memory address space like shared memory machines, but the memory may be physically distributed across nodes, where a node consists of 4 processors in this machine. There are a number of possible strategies for implementing both MPI alone and MPI in conjunction with OpenMP. Exploiting the architectural advantages of this machine, the one-level parallel strategy is adopted in this work to achieve a scalable parallel method for the 3D Navier-Stokes solver.

In our strategy, the original grid is distributed across the memories of the nodes and each node performs the computations for its assigned grid nodes. This is known as geometric domain decomposition technique, the same program on all of the processors in a parallel run. However, each program is only working on its part of the original data. Communication between the individual processes based on MPI is used to solve the

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

complete problem. The underlying mesh is partitioned into several sub-meshes, each defining a sub-domain. An overlapping mesh-partitioning technique is employed in this work so as to build the control volume for finite-volume formulation. Typically, a fixed number of MPI processes are used, corresponding to the number of sub-domains.

We will use Figure 5.1 of reference [150] to depict the geometric domain decomposition parallelization strategy employing MPI communication, which is repeated as Figure 6.7 for convenience here, $ktmax$ and $tmax$ denote the maximum time steps and the maximum physical time running, respectively.

Figure 6.7 Flowchart depicting geometric domain decomposition parallelization strategy alone using MPI only [150]

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

6.2.2 One-Level Parallelization Strategy - Domain Decomposition

Domain Decomposition is a generic name given to a variety of computational activities, which involve the division of a problem space into two or more parts that can be operated on separately by the same program to some advantages. Domain decomposition of a mesh into a set S of sub-domains that maybe allocated to a set of P processors involves finding a partition of the mesh so that the amount of computational time on each processor is almost equal. This method has the advantage of flexibility to allow variations in the decomposition strategy to be used to minimize the communication and maximize processor utilization. This must be done with care so that the amount of computation and/or message passing associated with sub-domain boundary interfaces does not grow rapidly with the number of processors. For structured mesh, the mesh partitioning is quite straightforward. However, for unstructured mesh, the mesh partitioning becomes a non-trivial problem. Following the 2D approach presented in [150, 154 and 155], domain decomposition contributes the most important part of this parallel implementation, which are outlined in the following sub-sections.

6.2.2.1 Meshes Partition algorithm

METIS [156, 157] is the computer code used to produce the partitions used in this work. It is a software package developed by University of Minnesota, Department of Computer Science, for partitioning unstructured meshes, large irregular graphs, and computing fill-reducing orderings of sparse matrices.

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

In general, METIS produces a good node partition, but the output nodal partitioned file is yet to be suitable for the 3D Navier-Stoke solver used in this work. In view of this, algorithms are developed to identify the “*ghost nodes*” and overlapping elements [154, 155]. The developed algorithm is written in a program named “*Partition*” and is fully integrated into the 3D solver, so as to have little intervention from the user. Basically, the algorithm developed in the program consists of four essential steps:

- (1) Identify *ghost* nodes and overlapping elements between the neighbouring sub-domains.
- (2) Generate individual grid files with local numbering for every partition.
- (3) Establish data structures for communication between core and *ghost* nodes.
- (4) Generate a script file, “*combine*”, to integrate all the result files into one after the parallel simulation has been performed.

6.2.2.2 Identification of Ghost Nodes and Overlapping Elements

After having obtained a partition of the mesh into P parts using METIS, algorithms are developed in [154, 155] to identify the “*ghost nodes*” and “*overlapping elements*” and to write the individual grid files with local numbering for each partition. The program also includes establishing the data structure for communication and generating a script file, “*combine*”, to integrate all the result files into one as each processor writes its own

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

results based on local numbering. The nodes and elements that are allocated uniquely to a processor are referred to as core mesh components in this work and each processor calculates the flow field variables and nodal gradients for them. Each sub-domain is enclosed with a layer of *ghost* nodes and overlapping elements, which overlap the sub-domains along the inter-processor boundaries.

These *ghost* nodes contain flow field variables and nodal gradients from neighboring sub-domains that are required for the solution of variables within the current sub-domain, so as to maintain a solution consistent with the original serial code and no computation is being performed on them. Communication between these core and *ghost* nodes is based on MPI and the data flow direction is always from core nodes to *ghost* nodes. *The main concept of this algorithm is that those elements along the inter-processor boundaries and with nodes having different partition numbers are considered as overlapping elements which are cut through by partition lines. And those nodes that formed these elements are a mixture of core and ghost nodes.* Basically, the *ghost* node of this partition is the mirror image of the core node of the other neighboring partition. For example, as shown in Figure 6.8, the *ghost* node number 11 of partition 3 is the mirror image of core node number 10 of partition 1. Therefore, making use of this correlation between the core and *ghost* nodes, the rest of the *ghost* nodes can be identified so as to build the control volume for the core node.

Figure 6.8 Mesh decomposed into 4 partitions showing each sub-domain extended with a layer of *ghost* nodes and overlapping elements [150]

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

6.2.2.3 Grid Files with Local Numbering

Since SPMD programming paradigm is employed in this work for geometric domain decomposition, which distributes the problem domain over P processors and each processor runs the same program individually. It is a wise strategy to re-write the global grid file into the respective local partitioned grid files with local numbering. In this way, each processor will only read and write their respective grid file and output files individually, without having any *master* processor to control the I/O operations. This avoids any synchronization in the event of reading and writing and it behaves like a serial program, each runs its own computation. The communication is only used during the passing of nodal gradients and flow variables. Thus a substantial amount of computational time and memory allocation is saved. Another advantage is that the solver only reads the respective partitioned grid file for the next simulation run without both accessing the global grid file and generating the mesh decomposition algorithm again.

6.2.2.4 Data Structure for Intercommunications between Partitions

The communication between the core and *ghost* nodes requires a data structure for each sub-domain, which holds the nodes and processor number to be sent and received. Each processor reflects its re-numbered sub-domain as a complete mesh consisting of 1 to n_e elements and 1 to n_p nodes, where n_e and n_p are the local numbers of elements and nodes respectively. In this work, communication and writing of variables to files are based on local numbering rather than global numbering on most parallel computers, moving data from one processor to another takes more time than moving or manipulating data within a

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

single process. To keep a program from being slowed down, many parallel computers allow users to start sending or receiving several messages and proceed with other operations simultaneously. Message passing is done asynchronously using the Persistent Communication Requests in this work. This allows all of the partitions to start sending or receiving messages at the same time, and this capability can minimize the system's whole waiting time. In order to reduce the communication time and the latency, the flow field variables, nodal gradients and forcing terms are packed into a single array before sending it to the respective partitions.

6.2.3 Multigrid Parallelization

In Mavriplis's work [158, 159], an agglomeration-based multigrid algorithm was parallelized based on independent partitioning of each level of mesh in the multigrid hierarchy and this results in an unrelated coarse and fine grid partitions. The advantage of this strategy is that load balancing for each grid level is easier to control. The disadvantage is that the partitions of coarser grids may bear no relation to the partition of the corresponding finer grids and results in increase in inter-grid communication. In the present work, the need for communication between processors for different grid levels during transferring of variables and residuals and interpolation of corrections are eliminated. The multigrid domain decomposition approach (MG-DD) [150, 154] is adopted for the multigrid parallelization in this study. This means that the non-nested multigrids are independently generated first. Then domain decomposition of the finest grid is performed, which is followed by decomposition of the various coarse levels of

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

grids guided by the finer grid partitions. This is achieved by using the fine grid partitions to infer the coarse level partitions (i.e. the coarse grid is to inherit its partition from that of the corresponding fine grid) and load balancing in the coarse mesh is reasonably well ensured. The only communication needed to perform once on the coarse grids is after transferring of flow field variables and residuals and on the fine grid after transferring of corrections, as shown in Figure 6.7. This communication is to ensure that the *ghost* nodes will have updated solution from the respective partitions after the transferring of variables from the fine grid and transferring of corrections from the coarse grid.

In the multigrid process, the flow field variables and residuals of the coarse grid nodes are obtained directly from their corresponding fine grid nodes. A two level multigrid and two sub-domains are used as examples to describe the procedure of partitioning the coarse grid using the fine grid. The main idea about this multigrid parallelization algorithm is that the fine grid is partitioned into two sub-domains according to the algorithm developed for single grid depicted in Section 6.1.3. And both the maximum and minimum values in the x- and y-direction (x_{min} , x_{max} , y_{min} & y_{max}) of each partition for the fine grid are noted. With these dimensions, an imaginary bounding box enclosing the sub-domains is formed. The main purpose of this bounding box is to identify the coarse nodes that fall within this box according to the fine grid partitioned, including those nodes beyond the sub-domains boundary.

Next, a search is done for those actual coarse nodes that fall within the sub-domain and filter out those nodes that fall onto the shared portions of the sub-domains using the

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

algorithm depicted in [150]. After classifying the respective coarse nodes according to which partition they belong to, and then the *ghost* nodes and overlapping elements are identified using the algorithm depicted in Section 6.2.2.2. The individual grid file for the partitioned coarse grid and data structure for communication is written to the respective partition.

6.2.4 Performance Measuring Techniques

Both speed-up and parallel efficiency are commonly used to measure the performance of a parallel code [150]. The run time of the original serial code is used as a measure of the run time on one processor. In this context, run time can be defined as the time that has elapsed from the moment when the first process actually begins execution of the program to the moment when the last process executes its last statement [160]. In this study, the run time or total simulation time starts from the moment before mesh partitioning for either single grid or MG, identifying *ghost* nodes and writing local grid files for different partitions to the time after writing all the results to the respective files.

Here, both CPU time and wall-clock time are used to record the total simulation time. The main difference is that CPU time is the recorded time when only the processor performs a calculation whereas clock time is almost similar to CPU time but it includes idling time when the processor idles while waiting for other processors to communicate. The wall-clock time is used to represent the total simulation time in this work since it includes the idling time, computation time and communication time, which is the true

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

representation of the total simulation time.

6.2.4.1 Speed-up

Parallel speed-up S_p is the ratio of the run-time on one processor t_1 to the run-time on P processors, t_p .

$$S_p = \frac{t_1}{t_p} \quad (6.13)$$

If the parallelization is 100% efficient, then $S_p = P$, but this is rarely the case as both communication and load balancing adversely affect the performance of parallel computation. In this work, the total simulation wall-clock time, t_p , is used as the run-time to compute the speedup and it is the maximum wall-clock time among all the processors defined as follows:

Total simulation wall-clock time,

$$t_p = \max (t_{\text{cpu}}^j + t_{\text{idle}}^j) \quad (6.14)$$

where j is the processor number, t_{cpu}^j and t_{idle}^j is the CPU time and idling time for processor j , respectively. And the CPU time for processor j is defined as follows:

CHAPTER 6 – THE MULTIGRID METHOD AND PARALLEL COMPUTATION

$$\text{CPU time, } t_{\text{cpu}}^j = t_{\text{comp}}^j + t_{\text{comm}}^j \quad (6.15)$$

where t_{comp}^j and t_{comm}^j is the computation and communication time for processor j , respectively. The communication time is the maximum wall-clock time among all the processors defined as follows:

$$t_{\text{comm}} = \max(t_{\text{comm}}^j) = \max\left(\sum_j t_{\text{comm}}^{\text{FFV}} + t_{\text{comm}}^{\text{NG}} + t_{\text{comm}}^{\text{FT}}\right) \quad (6.16)$$

where $t_{\text{comm}}^{\text{FFV}}$, $t_{\text{comm}}^{\text{NG}}$ and $t_{\text{comm}}^{\text{FT}}$ is the communication time for exchanging of flow field variables, nodal gradients and forcing terms, respectively. And the computation time is obtained as follows:

$$t_{\text{comp}} = t_p - t_{\text{comm}} \quad (6.17)$$

6.2.4.2 Parallel Efficiency

Parallel efficiency is sometimes used as the performance parameter for a parallel code. Parallel efficiency E_p is simply the ratio of the parallel speed-up S_p to the number of processors P .

$$E_p = \frac{S_p}{P} \quad (6.18)$$

where $0 \leq E_p \leq 1$.

CHAPTER SEVEN

LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

The mathematical formulation and implementation details of large-eddy simulation for compressible flow are introduced in this chapter. The subgrid-scale (SGS) terms are modeled using an eddy-viscosity formula, the Smagorinsky model. The model constants are determined using a newly developed dynamic procedure derived from the classic Lilly procedure. The turbulent Prandtl number is also determined by a similar dynamic procedure.

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

7.1 FILTERED NAVIER-STOKES EQUATIONS FOR THE COMPRESSIBLE FLOW

For large eddy simulation, a filtering operation is applied to the compressible Navier-Stokes equations to separate the effects of the large-scale and small-scale motions. The filtering operation is applied to local, instantaneous quantities in terms of a convolution integral as

$$\bar{f}(\vec{x}, t) = \int_D G(\vec{x}, \vec{\xi}) \bar{f}(\vec{\xi}, t) d\vec{\xi} \quad (7.1)$$

where G is the filter function and the integration is performed over the entire domain. The quantity f is decomposed as $f = \bar{f} + f'$, where $\bar{f}$ is the large-scale, or resolved component, and f' is the small-scale, unresolved, or SGS component.

The top-hat filter, also known as the box filter, was exclusively used with the finite volume formulation of this study. In physical space, it is given by

$$G(\vec{x}, \vec{\xi}) = \begin{cases} \frac{1}{\Delta^3} & \text{if } x_i - \Delta/2 \leq \xi_i \leq x_i + \Delta/2 \\ 0 & \text{otherwise} \end{cases} \quad (7.2)$$

where Δ is the filter width and normally can be evaluated by the grid size. To filter a quantity in physical space, the convolution integral in Equation (7.1) must be approximated. This was done by applying a three-dimensional approximation in the

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

homogeneous directions. The filtering operation in Equation (7.1) is applied to the nondimensional continuity, momentum and thermal energy equations (Equation 3.14).

And it can be shown that if G is a function of $(\vec{x} - \vec{\xi})$ only, then differentiation and the filtering operation commute [161] as

$$\frac{\partial \bar{f}}{\partial t} = \frac{\partial \bar{f}}{\partial t} \quad \text{and} \quad \frac{\partial \bar{f}}{\partial x} = \frac{\partial \bar{f}}{\partial x} \quad (7.3)$$

This leads to the following set of filtered governing equations:

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial (\bar{\rho} u_i)}{\partial x_i} = 0 \quad (7.4)$$

$$\frac{\partial (\bar{\rho} u_i)}{\partial t} + \frac{\partial (\bar{\rho} u_i u_j)}{\partial x_j} = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial (\bar{\tau}_{ij})}{\partial x_j} \quad (7.5)$$

$$\frac{\partial (\bar{\rho} e_i)}{\partial t} + \frac{\partial (\bar{\rho} e_i u_j)}{\partial x_j} = -\frac{\partial (\bar{p} u_j)}{\partial x_j} - \frac{\partial \bar{q}_j}{\partial x_j} + \frac{\partial (\bar{\tau}_{ij} u_i)}{\partial x_j} \quad (7.6)$$

where

$$\bar{\tau}_{ij} = \frac{2\mu}{Re_\infty} \left(S_{ij} - \frac{1}{3} S_{kk} \delta_{ij} \right) \quad (7.7)$$

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

$$\bar{q}_j = -\frac{\gamma}{Re_\infty Pr} \frac{\partial T}{\partial x_j} \quad (7.8)$$

and the equation of state becomes

$$\bar{p} = R\bar{\rho}\bar{T} \quad (7.9)$$

As can be seen from above that, when the system of compressible Navier-Stokes equations are filtered, some additional problems which are absent in its incompressible counterpart will arise. This is so because now there is one more unknown, ρ , which appears in the convective nonlinear terms. As a result, the convective terms to be filtered contain a triple product of the unknown variables. Tremendous complexities would be introduced into the modeling of different SGS stress terms using the original filtered variables. To simplify the filtered equations, a Favre-averaging typically used in traditional Reynolds averaging equations for compressible modeling is further applied to give [162]

$$\tilde{f} = \frac{\overline{\rho f}}{\bar{\rho}} \quad (7.10)$$

which implies the decomposition $f = \tilde{f} + f''$, where $\tilde{f}$ is the resolved component, and f'' is the unresolved component. As a result of the Favre filter, we have

$$\widetilde{\rho u_i} = \bar{\rho} \tilde{u}_i, \quad \widetilde{\rho u_i u_j} = \bar{\rho} \tilde{u}_i \tilde{u}_j, \quad \widetilde{\rho e_i u_j} = \bar{\rho} \tilde{e}_i \tilde{u}_j \quad (7.11)$$

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

Consequently, the Favre filtered governing equations are

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial (\bar{\rho} \tilde{u}_i)}{\partial x_i} = 0 \quad (7.12)$$

$$\frac{\partial (\bar{\rho} \tilde{u}_i)}{\partial t} + \frac{\partial (\bar{\rho} \tilde{u}_i \tilde{u}_j)}{\partial x_j} = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial \bar{\tau}_{ij}}{\partial x_j} + \frac{\partial (\bar{\tau}_{ij} - \bar{\tau}_{ij})}{\partial x_j} - \underbrace{\frac{1}{Re_\infty} \frac{\partial \sigma_{ij}}{\partial x_j}}_{(1^*)} \quad (7.13)$$

$$\begin{aligned} \frac{\partial (\bar{\rho} \tilde{e}_i)}{\partial t} + \frac{\partial [(\bar{\rho} \tilde{e}_i + \bar{p}) \tilde{u}_j]}{\partial x_j} &= \frac{\partial (\bar{\tau}_{ij} \tilde{u}_i)}{\partial x_j} - \frac{\partial \bar{q}_j}{\partial x_j} - \frac{\partial (\bar{q}_j - \bar{q}_j)}{\partial x_j} \\ &+ \frac{1}{Re_\infty} \left(-\underbrace{\alpha_1}_{(2^*)} - \underbrace{\alpha_2}_{(3^*)} - \underbrace{\alpha_3}_{(4^*)} + \underbrace{\alpha_4}_{(5^*)} + \underbrace{\alpha_5}_{(6^*)} \right) \end{aligned} \quad (7.14)$$

and

$$\bar{p} = R \bar{\rho} \tilde{T} \quad (7.15)$$

$$\tilde{e}_i = \frac{\bar{p}}{(\gamma - 1) \bar{\rho}} + \frac{1}{2} \tilde{u}_i \tilde{u}_i + \frac{1}{2} (\widetilde{u_i u_i} - \tilde{u}_i \tilde{u}_i) \quad (7.16)$$

where

$$q_{SGS}^2 = \frac{1}{2} (\widetilde{u_i u_i} - \tilde{u}_i \tilde{u}_i) \quad (7.17)$$

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

can be defined as the SGS kinetic energy, and

$$q_{SGS}^2 = \frac{1}{2} \tau_{ii} \quad (7.18)$$

Unlike the ‘bar’ and the ‘title’, the ‘arch’-symbol does not denote a filter operation but indicates that the quantity is based on filtered variables. The resolved viscous stress tensor and heat flux vector in Equations (7.13) and (7.14) are approximated as

$$\hat{\tau}_{ij} = \frac{2\hat{\mu}}{Re_\infty} \left(\hat{S}_{ij} - \frac{1}{3} \hat{S}_{kk} \delta_{ij} \right) \quad (7.19)$$

$$\hat{q}_{ij} = -\frac{\gamma\hat{\mu}}{Re_\infty Pr} \frac{\partial \tilde{T}}{\partial x_j} \quad (7.20)$$

where the strain rate tensor is

$$\hat{S}_{ij} = \frac{1}{2} \left(\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i} \right) \quad (7.21)$$

and $\hat{\mu}$ is given by Sutherland’s law based on the Favre-filtered temperature $\tilde{T}$. The approximations in Equations (7.19) and (7.20) require the assumption that the correlation between the fluid properties and the derivatives of the velocity and temperature are weak [163]. The third term in right hand side of filtered momentum equation (7.13) results

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

from the nonlinearity of the viscous term and the fact that the Favre filter and partial derivatives do not commute. A priori tests confirm that it is an order of magnitude smaller than the first term [165]. The same conclusion can be made for the third term on the right hand side of filtered energy equation (7.14). These terms are throughout neglected during this study.

The effects of the small-scale motion are present in the above equations through the SGS stress tensor in the Favre-filtered momentum equation.

$$\sigma_{ij} = \bar{\rho} \left(\widetilde{u_i u_j} - \tilde{u}_i \tilde{u}_j \right) \quad (7.22)$$

and the SGS terms in the energy equation [164, 165, 166 and 169]

$$\alpha_1 = \tilde{u}_i \frac{\partial \sigma_{ij}}{\partial x_j} \quad (7.23)$$

$$\alpha_2 = \frac{\partial (\overline{p u_j} - \bar{p} \tilde{u}_j)}{\partial x_j} / (\gamma - 1) \quad (7.24)$$

$$\alpha_3 = \overline{p \frac{\partial u_j}{\partial x_j}} - \bar{p} \frac{\partial \tilde{u}_j}{\partial x_j} \quad (7.25)$$

$$\alpha_4 = \overline{\tau_{ij} \frac{\partial u_i}{\partial x_j}} - \bar{\tau}_{ij} \frac{\partial \tilde{u}_i}{\partial x_j} \quad (7.26)$$

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

$$\alpha_s = \frac{\partial (\bar{\tau}_{ij} \tilde{u}_i - \tau_{ij} \tilde{u}_i)}{\partial x_j} \quad (7.27)$$

The subgrid-terms (1*) to (6*) in Equations (7.12-7.13) contain information from the SGS field. Subgrid models have to be included for these terms in order to express the filtered Navier-Stokes equations in filtered variables only. The turbulent stress tensor σ_{ij} is the only subgrid-term in incompressible flow. For this reason, we expect that compressible LES at low Mach numbers primarily requires the modeling of σ_{ij} . We expect the subgrid-terms in the energy equation to become more important if the Mach number is increased [169].

7.2 DYNAMIC MIXED SUBGRID MODELLIGN FOR THE TURBUMENT STRESS TENSOR

The SGS turbulent stress tensor σ_{ij} (term (1*) in Equation (7.13)) is the most important subgrid-term in LES. Much effort has been put into the development of good subgrid models [167] and, consequently, a large number of subgrid models exist. In this study, the dynamic mixed eddy-viscosity model [168, 169 and 170] is adopted, which is a linear combination of the scale similarity model and the dynamic Smagorinsky model. The dynamic eddy-viscosity model [170] is the Smagorinsky model in which the model constant is replaced by a coefficient which depends on the local turbulent structure of the flow. The dynamic eddy-viscosity model overcomes several shortcomings of the

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

Smagorinsky model, e.g. the excessive dissipation in laminar regions. The local value of the coefficient is obtained by substitution of the Smagorinsky model into the Germano identity, which is a relation between the turbulent stress tensor at several filter levels. The dynamic eddy-viscosity model has been successfully applied to LES of transitional channel flow [171] and to a number of other flows as well [167].

The SGS turbulent stress tensor σ_{ij} can be reformulated as

$$\sigma_{ij} = \bar{\rho} (\widetilde{u_i u_j} - \tilde{u}_i \tilde{u}_j) = \bar{\rho} (\widetilde{\tilde{u}_i \tilde{u}_j} - \tilde{u}_i \tilde{u}_j + \widetilde{\tilde{u}_i u_j''} + \widetilde{\tilde{u}_j u_i''} + \widetilde{u_i'' u_j''}) = L_{ij} + C_{ij} + R_{ij} \quad (7.28)$$

where

$$L_{ij} = \bar{\rho} (\widetilde{\tilde{u}_i \tilde{u}_j} - \tilde{u}_i \tilde{u}_j) \quad (7.29)$$

$$C_{ij} = \bar{\rho} (\widetilde{\tilde{u}_i u_j''} + \widetilde{\tilde{u}_j u_i''}) \quad (7.30)$$

$$R_{ij} = \bar{\rho} \widetilde{u_i'' u_j''} \quad (7.31)$$

are respectively, the SGS Leonard, cross, and Reynolds stresses based on Favre averaging filtering. It is clear that the Leonard stress in Equation (7.29) can be calculated directly from the filtered variables and does not need to be modeled. The cross stress is modeled with the scale similarity model [168, 169] with a coefficient of unity to ensure Galilean invariance overall model.

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

$$C_{ij} = \bar{\rho} (\tilde{u}_i \tilde{u}_j - \tilde{\tilde{u}}_i \tilde{\tilde{u}}_j) \quad (7.32)$$

The SGS Reynolds stress tensor is separated into deviatoric and isotropic parts, respectively, as follows:

$$R_{ij} = R_D + R_I \quad (7.33)$$

$$R_D = \bar{\rho} \left(\widetilde{u_i'' u_j''} - \frac{1}{3} \widetilde{u_k'' u_k''} \delta_{ij} \right) \quad (7.34)$$

$$R_I = \frac{1}{3} \bar{\rho} \widetilde{u_k'' u_k''} \delta_{ij} \quad (7.35)$$

where $\delta_{ij} = 1$ if $i=j$ and zero otherwise.

Before we go further into the dynamic model for the Reynolds stress tensor R_{ij} , we need to introduce the Germano Identity briefly as follows. We define the subgrid-term corresponding to an arbitrary nonlinear function or operator $f(w)$, where w is a vector function of space and time, as follows:

$$\sigma_f = \overline{f(w)} - f(\bar{w}) \quad (7.36)$$

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

We call σ_f the subgrid-term on the F -level, where the ‘bar’ denotes the basic filter operation. Apart from the grid-filter level (F -level), denoted by the bar-filter corresponding to the filter width Δ , Germano [170] introduced a test-filter (at the G -level), which is denoted by the hat ($\hat{\cdot}$) and corresponds to the filter width $\hat{\Delta}=2\Delta$. The consecutive application of these two filters, resulting in e.g. $\hat{\bar{\rho}}$, defines a filter on the ‘ FG -level’ with which a filter width $\hat{\hat{\Delta}}=k\Delta$ can be associated. The value of k equals 2 for the spectral cut-off filter [171] and $\sqrt{5}$ for Gaussian filters [170]. For spectral cut-off and Gaussian filters, k can be determined exactly, since the consecutive application of two of these filters yields a filter function of the same type. However, the consecutive application of two top-hat filters does not yield a top-hat filter. For top-hat filters an optimal value $k=\sqrt{5}$ can be derived. The subgrid-term on the FG -level reads

$$T_f = \widehat{f(\bar{w})} - f(\hat{\bar{w}}) \quad (7.37)$$

The following identity can be derived between the subgrid-terms at the FG - and the F -level

$$T_f - \widehat{\sigma_f} = L_f \quad (7.38)$$

where the right-hand side L_f can be explicitly calculated from the variable $\bar{w}$ on the F -level,

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

$$L_f = \widehat{f(\bar{w})} - f(\hat{\bar{w}}) \quad (7.39)$$

The terms on the left-hand side of the generalized Germano identity (7.38) cannot be calculated from the variables on the F -level. This generalized identity reduces to the Germano identity for the turbulent stress tensor in the case

$$f(w) = \rho u_i u_j \quad \text{with} \quad w = (\rho, u) \quad (7.40)$$

In this case identity (7.38) is equivalent to

$$\hat{\bar{\rho}} T_{ij} - \widehat{\bar{\rho} \sigma_{ij}} = L_{ij} \quad (7.41)$$

where σ_{ij} is the turbulent stress tensor and the other terms are given by

$$\hat{\bar{\rho}} T_{ij} = \widehat{\rho u_i u_j} - \widehat{\rho u_i} \widehat{\rho u_j} / \hat{\bar{\rho}} \quad (7.42)$$

$$L_{ij} = \left(\widehat{\rho u_i \rho u_j} / \hat{\bar{\rho}} \right)^\wedge - \widehat{\rho u_i} \widehat{\rho u_j} / \hat{\bar{\rho}} \quad (7.43)$$

The notation $(.)^\wedge$ indicates that the hat-filter is applied to the expression between the brackets. It is used in conjunction with the identically defined notation $(\hat{\cdot})$ for convenience in the exposure. The terms at the left-hand side of the Germano identity

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

(7.41) are the turbulent stress tensor on the FG -level and the turbulent stress tensor on the F -level filtered with the test-filter, respectively. The tensor L_{ij} can explicitly be calculated from the variables on the F -level, $\bar{\rho}$ and $\overline{\rho u}$. The dynamic model to be adopted for the turbulent stress tensor is obtained by substituting the corresponding base model into the Germano identity.

The dynamic eddy-viscosity model [170] adopts the compressible generalization of the Smagorinsky's eddy viscosity formulation, but the square of the Smagorinsky constant C_s is replaced by a coefficient C_d :

$$R_D = -2\bar{\rho}C_d\Delta^2 |S(\tilde{u})| \left(S_{ij}(\tilde{u}) - \frac{1}{3} S_{kk}(\tilde{u}) \delta_{ij} \right) \quad (7.44)$$

where $\delta_{ij} = 1$ if $i = j$ and zero otherwise. Following the research works done by Yoshizawa [172], the isotropic part of the SGS Reynolds stress tensor is modeled by

$$R_I = \frac{2}{3} \bar{\rho} C_I \Delta^2 |S(\tilde{u})|^2 \delta_{ij} \quad (7.45)$$

where

$$S_{ij}(\tilde{u}) = \frac{1}{2} \left(\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i} \right) \quad (7.46)$$

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

and

$$|S(\tilde{u})| = \sqrt{2S_{ij}(\tilde{u})S_{ij}(\tilde{u})} \quad (7.47)$$

We will use Equations (7.41), (7.44) and (7.45) to compute the coefficient C_d and C_l dynamically. To determine C_l , Equation (7.43) is used in conjunction with the model of Equation (7.45) for T_{kk} and σ_{kk} :

$$(\overline{\rho \tilde{u}_k \tilde{u}_k})^\wedge - (\overline{\rho \tilde{u}_k} \overline{\rho \tilde{u}_k}) / \hat{\rho} = 2C_l \left(\hat{\rho} \hat{\Delta}^2 |S(\hat{\tilde{u}})|^2 - \Delta^2 (\overline{\rho} |S(\tilde{u})|^2)^\wedge \right) \quad (7.48)$$

After some simple arrangement Equation (7.48) leads to

$$C_l = \frac{\langle (\overline{\rho \tilde{u}_k \tilde{u}_k})^\wedge - (\overline{\rho \tilde{u}_k} \overline{\rho \tilde{u}_k}) / \hat{\rho} \rangle}{\langle 2\hat{\rho} \hat{\Delta}^2 |S(\hat{\tilde{u}})|^2 - 2\Delta^2 (\overline{\rho} |S(\tilde{u})|^2)^\wedge \rangle} \quad (7.49)$$

where $\langle \rangle$ indicates applying a local average which is needed to make the determination of C_l and the other SGS coefficients well conditioned. Equation (7.49) is a set of independent equations that cannot be solved explicitly for C_l , which appears inside a filtering operation. As proposed by Lilly [178], C_l can be solved by employing a least-squares technique to ensure the overall error of the result coefficient C_l is in minimum level. An important assumption made to derive Equation (7.49) is that C_l is a function of the time and varies smoothly with space, such that it can be safely extracted

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

out of the filtering operation. But in practice, the fluctuation of this coefficient in the fluid domain is quite large. The mathematical inconsistency of Equation (7.49) may lead to serious problem if the Reynolds number is big. Further improvement can be introduced to alleviate this problem. For details please resort to reference [179, 181]. Finally the model for R_I is:

$$R_I = \frac{\bar{\rho} \left\langle \left(\widehat{\bar{\rho} \tilde{u}_k \tilde{u}_k} \right) - \left(\widehat{\bar{\rho} \tilde{u}_k} \widehat{\bar{\rho} \tilde{u}_k} \right) / \hat{\bar{\rho}} \right\rangle}{\left\langle 3 \frac{\hat{\Delta}^2}{\Delta^2} \hat{\bar{\rho}} \left| S(\hat{\tilde{u}}) \right|^2 - 3 \left(\bar{\rho} \left| S(\tilde{u}) \right|^2 \right) \right\rangle} \left| S(\tilde{u}) \right|^2 \delta_{ij} \quad (7.50)$$

To obtain C_d , we first write

$$\begin{aligned} \sigma_{ij} - \frac{1}{3} \sigma_{kk} \delta_{ij} &= \bar{\rho} \left(\widehat{\tilde{u}_i \tilde{u}_j} - \tilde{u}_i \tilde{u}_j \right) - \frac{1}{3} \bar{\rho} \left(\widehat{\tilde{u}_k \tilde{u}_k} - \tilde{u}_k \tilde{u}_k \right) \delta_{ij} - \\ &2 \bar{\rho} C \Delta^2 \left| S(\tilde{u}) \right| \left(S_{ij}(\tilde{u}) - \frac{1}{3} S_{kk}(\tilde{u}) \delta_{ij} \right) \end{aligned} \quad (7.51)$$

and

$$\begin{aligned} T_{ij} - \frac{1}{3} T_{kk} \delta_{ij} &= \hat{\bar{\rho}} \left(\widehat{\hat{\tilde{u}}_i \hat{\tilde{u}}_j} - \hat{\tilde{u}}_i \hat{\tilde{u}}_j \right) - \frac{1}{3} \hat{\bar{\rho}} \left(\widehat{\hat{\tilde{u}}_k \hat{\tilde{u}}_k} - \hat{\tilde{u}}_k \hat{\tilde{u}}_k \right) \delta_{ij} - \\ &2 \hat{\bar{\rho}} C_d \hat{\Delta}^2 \left| S(\hat{\tilde{u}}) \right| \left(S_{ij}(\hat{\tilde{u}}) - \frac{1}{3} S_{kk}(\hat{\tilde{u}}) \delta_{ij} \right) \end{aligned} \quad (7.52)$$

Using (7.51) and (7.52) in (7.41), we obtain

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

$$\begin{aligned}
 (\bar{\rho} \tilde{u}_i \tilde{u}_j)^\wedge - (\widehat{\bar{\rho} \tilde{u}_i} \widehat{\bar{\rho} \tilde{u}_j}) / \hat{\rho} &= \left(\frac{1}{3} T_{kk} \delta_{ij} - \frac{1}{3} \hat{\sigma}_{kk} \delta_{ij} \right) \\
 &\quad - 2C_d \hat{\rho} \hat{\Delta}^2 |S(\hat{\tilde{u}})| \left(S_{ij}(\hat{\tilde{u}}) - \frac{1}{3} S_{kk}(\hat{\tilde{u}}) \delta_{ij} \right) \\
 &\quad + 2C_d \Delta^2 \left[\bar{\rho} |S(\tilde{u})| \left(S_{ij}(\tilde{u}) - \frac{1}{3} S_{kk}(\tilde{u}) \delta_{ij} \right) \right]^\wedge \\
 &\quad + \hat{\rho} \left(\widehat{\tilde{u}_i \tilde{u}_j} - \hat{\tilde{u}}_i \hat{\tilde{u}}_j \right) - \left(\bar{\rho} (\widetilde{\tilde{u}_i \tilde{u}_j} - \tilde{\tilde{u}}_i \tilde{\tilde{u}}_j) \right)^\wedge \\
 &\quad - \frac{1}{3} \left[\hat{\rho} \left(\widehat{\tilde{u}_k \tilde{u}_k} - \hat{\tilde{u}}_k \hat{\tilde{u}}_k \right) - \left(\bar{\rho} (\widetilde{\tilde{u}_k \tilde{u}_k} - \tilde{\tilde{u}}_k \tilde{\tilde{u}}_k) \right)^\wedge \right] \delta_{ij}
 \end{aligned} \tag{7.53}$$

To solve for C_d , we again introduce in a least square approach. It leads to

$$C_d = - \frac{ \left\langle \left[(\bar{\rho} \tilde{u}_i \tilde{u}_j)^\wedge - (\widehat{\bar{\rho} \tilde{u}_i} \widehat{\bar{\rho} \tilde{u}_j}) / \hat{\rho} \right] - \left[\hat{\rho} \left(\widehat{\tilde{u}_i \tilde{u}_j} - \hat{\tilde{u}}_i \hat{\tilde{u}}_j \right) - \left(\bar{\rho} (\widetilde{\tilde{u}_i \tilde{u}_j} - \tilde{\tilde{u}}_i \tilde{\tilde{u}}_j) \right)^\wedge \right] + \right. }{ \left. \left[\frac{1}{3} \left[\hat{\rho} \left(\widehat{\tilde{u}_k \tilde{u}_k} - \hat{\tilde{u}}_k \hat{\tilde{u}}_k \right) - \left(\bar{\rho} (\widetilde{\tilde{u}_k \tilde{u}_k} - \tilde{\tilde{u}}_k \tilde{\tilde{u}}_k) \right)^\wedge \right] \delta_{ij} - \frac{1}{3} (T_{mm} - \hat{\sigma}_{mm}) \delta_{ij} \right. \right. } \tag{7.54}$$

$$\left. \left. \begin{aligned} &- 2\hat{\rho} \hat{\Delta}^2 |S(\hat{\tilde{u}})| \left(S_{ij}(\hat{\tilde{u}}) - \frac{1}{3} S_{kk}(\hat{\tilde{u}}) \delta_{ij} \right) + \\ &2\Delta^2 \left[\left(\bar{\rho} |S(\tilde{u})| S_{ij}(\tilde{u}) \right)^\wedge - \frac{1}{3} \left(\bar{\rho} |S(\tilde{u})| S_{kk}(\tilde{u}) \delta_{ij} \right)^\wedge \right] \right. \end{aligned} \right\rangle$$

where $T_{mm} - \hat{\sigma}_{mm} = (\bar{\rho} \tilde{u}_m \tilde{u}_m)^\wedge - (\widehat{\bar{\rho} \tilde{u}_m} \widehat{\bar{\rho} \tilde{u}_m}) / \hat{\rho}$.

Hence, the overall SGS stress tensor model we propose takes the form

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

$$\sigma_{ij} = \bar{\rho} \left(\widetilde{\tilde{u}_i \tilde{u}_j} - \tilde{\tilde{u}}_i \tilde{\tilde{u}}_j \right) - \frac{1}{3} \bar{\rho} \left(\widetilde{\tilde{u}_k \tilde{u}_k} - \tilde{\tilde{u}}_k \tilde{\tilde{u}}_k \right) \delta_{ij} - 2 \bar{\rho} C_d \Delta^2 |S(\tilde{u})| \left(S_{ij}(\tilde{u}) - \frac{1}{3} S_{kk}(\tilde{u}) \delta_{ij} \right) + \frac{2}{3} \bar{\rho} C_l \Delta^2 |S(\tilde{u})|^2 \delta_{ij} \quad (7.55)$$

In the present work, after the model coefficient C_d is computed from (7.54), it is averaged locally in space within the test-filtering volume. The use of local averaging alone is, however, not enough to prevent solutions from blowing up because the eddy viscosity $\mu_t = \bar{\rho} C_d \Delta^2 |S(\tilde{u})|$ using the locally averaged C_d could be negative and have a magnitude larger than the molecular viscosity μ . The result is that the total viscosity or the total diffusivity is negative which, if this condition persists, will lead to an exponentially growing unstable solution. In order to guarantee that this does not happen, we set the total viscosity $\mu + \mu_t$ to be zero whenever it becomes negative.

When C_d and C_l are used in above equation, the only adjustable input to the model will be the ratio of the two filter widths, $\alpha = \hat{\Delta} / \Delta$. In the LES calculations reported here, α was chosen to be $\sqrt{5}$, which has been shown to be the optimal value [171].

7.3 DYNAMIC MIXED SUBGRID MODELLING FOR THE ENERGY EQUATION

The filtered equations for compressible flow have subgrid-terms in the momentum and energy equations. In the previous section we focused on the modeling of the dominant

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

subgrid-term in the momentum equation, the turbulent stress tensor. The subgrid-terms in the energy equation (energy subgrid-terms) are the subject of this section. Compared to incompressible LES, much less research has been conducted on compressible LES. The first compressible subgrid-model has been formulated by Yoshizawa [172], who generalized the Smagorinsky formulation by incorporating a model for the isotropic part of the turbulent stress tensor. Erlebacher et al. [162] have extended the standard mixed model to compressible isotropic turbulence. Moin et al. [173] formulated the dynamic model for compressible LES, with the dynamic eddy-viscosity model in the momentum equation and a dynamic eddy-diffusivity model in the energy equation. This model has been applied to compressible isotropic turbulence (Moin et al. [173]) and to high-speed transitional boundary layers (El-Hady et al. [174]). The filtered energy equation contains five subgrid-terms α_1 to α_5 (Equations (7.23-7.27)). In this study we use the evolution equation for the total energy, whereas Erlebacher et al. and Moin et al. employ the internal energy equation [162]. As indicated by Vreman [169] the differences between the two formulations are not large. The filtered internal energy equation contains the subgrid-terms $\alpha_2, \alpha_3, \alpha_4$. It does not contain α_1 , whereas α_5 has a different form. The question arises whether all energy subgrid-terms are equally important. Erlebacher et al. and Moin et al. use a model for α_2 only, but they do not give satisfactory arguments why the other subgrid-terms can be neglected. According to Vreman [169, 175 and 176], we will model α_1 to α_4 , and ignore α_5 .

The turbulent stress tensor σ_{ij} in previous section is modeled with the dynamic eddy viscosity model. The expression in Equation (7.53) substituted in the definition of α_1

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

provides the subgrid-model for α_1 . Moin et al. [173] introduced a dynamic eddy-diffusivity concept to model the pressure velocity subgrid-term α_2 . The dynamic eddy-diffusivity model employed in section 7.2 is slightly different. The dynamic procedure is applied such that the model represents the sum $\alpha_2 + \alpha_3$ instead of α_2 only, in order to model the effect of α_3 . Hence, the following expression is adopted for these subgrid terms [175, 176]:

$$\alpha_2 + \alpha_3 = \widetilde{f(\tilde{w})} - f(\tilde{\tilde{w}}) - \frac{\partial}{\partial x_j} \left[\frac{\gamma \bar{\rho} C \Delta^2 |S(\tilde{u})|}{Pr_t} \frac{\partial \tilde{T}}{\partial x_j} \right] \quad (7.56)$$

where

$$\begin{aligned} \widetilde{f(\tilde{w})} &= \left[\frac{1}{(\gamma - 1)} \frac{\partial (\bar{\rho} \tilde{u}_j)}{\partial x_j} + \bar{\rho} \frac{\partial \tilde{u}_j}{\partial x_j} \right] \\ f(\tilde{\tilde{w}}) &= \left[\frac{1}{(\gamma - 1)} \frac{\partial (\bar{\bar{\rho}} \tilde{\tilde{u}}_j)}{\partial x_j} + \bar{\bar{\rho}} \frac{\partial \tilde{\tilde{u}}_j}{\partial x_j} \right] \end{aligned} \quad (7.57)$$

In the RHS of (7.56), the numerator in the fraction of the third term denotes the eddy-viscosity on the F-level and Pr_t is the dynamic turbulent Prandtl number. This eddy-diffusivity model is similar to the molecular heat flux term defined in chapter 3, but the molecular viscosity and Prandtl number have been replaced by the dynamic eddy-viscosity in equation (7.44) and the turbulent Prandtl number. The non-dynamic formulation was first introduced by Eidson [177] in LES of the Rayleigh-Bénard problem. Here the turbulent Prandtl number is a dynamic coefficient. To expose the dynamic procedure for Pr_t , the model is substituted in the generalized Germano identity, which

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

results in

$$\left[\widehat{f(\hat{w})} - f(\hat{w}) - \frac{\partial}{\partial x_j} \left(\frac{\gamma \hat{p} C_d \Delta^2 |S(\hat{u})|}{Pr_t} \frac{\partial \hat{T}}{\partial x_j} \right) \right] - \left[\widehat{f(\tilde{w})} - f(\tilde{w}) - \frac{\partial}{\partial x_j} \left(\frac{\gamma C_d \Delta^2}{Pr_t} \left(\frac{\bar{\rho} |S(\tilde{u})| \partial \tilde{T}}{\partial x_j} \right)^\wedge \right) \right] =$$

$$\left[\frac{1}{(\gamma-1)} \frac{\partial (\bar{p} \tilde{u}_j)}{\partial x_j} + \bar{p} \frac{\partial \tilde{u}_j}{\partial x_j} \right] - \left[\frac{1}{(\gamma-1)} \frac{\partial (\hat{p} \hat{u}_j)}{\partial x_j} + \hat{p} \frac{\partial \hat{u}_j}{\partial x_j} \right]$$

(7.58)

where C_d is determined by (7.52). We assume that Pr_t is only a function of time and it can consistently be taken out of the spatial derivatives and filterings. Hence, it can be explicitly computed from above equation with the least square approach.

In order to make the closure of Equation (7.14), we incorporate a model for the turbulent dissipation rate α_4 . It is a positive quantity, provided a positive filter is used. The term also occurs in the k -equation model of incompressible LES studied by Horiuti [180] and Ghosal et al. [181, 182]. Following these references we model the term by

$$\alpha_4 = c_\epsilon \bar{\rho} \frac{k^{3/2}}{\Delta} \quad (7.59)$$

where $k = \frac{1}{2} \sigma_{kk}$ is the subgrid turbulent kinetic energy and c_ϵ is a dynamic coefficient

which is assumed to be a function of time only. Note that the filter function has to be positive in order to guarantee the positivity of k . The model is ill-defined if the spectral

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

cut-off filter is used. In the following we present the procedure to obtain the dynamic coefficient c_ϵ .

Ghosal et al. [181, 182] remark that for high Reynolds number flows the coefficient cannot be determined using the standard dynamic procedure, since the turbulent dissipation is essentially a small-scale phenomenon. Turbulent energy is generated by the large scales and after it has been transferred to small scales it is dissipated by molecular viscosity. The magnitude of the turbulent dissipation rate is therefore set by the large scales, but the dissipation takes place at the smallest scales. This implies that α_4 will not be small in fully developed high Reynolds number flows. It also implies that there is no dissipation left in the resolved scales if the Reynolds number is high. Consequently, the Germano identity for α_4 is zero in the limit of infinite Reynolds number and cannot be used to determine c_ϵ . Therefore, Ghosal et al. [181, 182] do not use the Germano identity for the turbulent dissipation rate, but determine c_ϵ with a complicated procedure, using a local balance of the terms in the k -equation. We present a much simpler procedure to determine c_ϵ , using a global balance of the terms in the integrated k -equation. In this approach c_ϵ is necessarily a function of time only, in contrast to the approach by Ghosal et al., where c_ϵ depends on time and space. The k -equation in compressible flow can be written as

$$\frac{\partial(\bar{\rho}k)}{\partial t} = \alpha_1 + \alpha_3 - \alpha_4 + \frac{\partial}{\partial x_j} \left(\frac{\partial R_j}{\partial x_j} \right) \quad (7.60)$$

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

where the last term of RHS represents all contributions which can be written in divergence form. Integrating this equation over the computational domain and assuming that R_2 is negligible at the free-slip walls gives

$$\int \frac{\partial(\bar{\rho}k)}{\partial t} = \int (\alpha_1 + \alpha_3 - \alpha_4) dx \quad (7.61)$$

Substitution of the dissipation model (7.59) in this equation provides an expression for the model coefficient:

$$c_\varepsilon = \frac{\int \left(\alpha_1 + \alpha_3 - \frac{\partial(\bar{\rho}k)}{\partial t} \right) dx}{\int (\bar{\rho}k^{3/2} / \Delta) dx} \quad (7.62)$$

The quantities α_1 and k can be calculated from the resolved variables and the model for the turbulent stress tensor σ_{ij} . The integral $\int \alpha_3 dx$ is obtained by integrating the model for $\alpha_2 + \alpha_3$, since $\int \alpha_2 dx = 0$. The model for $\alpha_2 + \alpha_3$ in Equation (7.56) is in divergence form. Hence, it does not contribute to the global integral. To incorporate a similarity model for the dissipation is not very useful, since such a model extracts its information from the resolved scales. As stated in [176], the resolved scales do not contribute to α_4 if the Reynolds number is high.

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

7.4 FILTERED GOVERNING EQUATION IN VECTOR FORM

Finally, the nondimensional governing equations (7.12-7.14) can be written in vector form (preconditioned):

$$\Gamma_1 \frac{\partial W_p}{\partial \tau} + \frac{\partial W_c}{\partial t} + \frac{\partial E_i}{\partial x} + \frac{\partial F_i}{\partial y} + \frac{\partial G_i}{\partial z} = \frac{\partial E_v}{\partial x} + \frac{\partial F_v}{\partial y} + \frac{\partial G_v}{\partial z} + S_{SGS} \quad (7.63)$$

$$W_p = \begin{bmatrix} \tilde{p} \\ \tilde{u} \\ \tilde{v} \\ \tilde{w} \\ \tilde{T} \end{bmatrix} \quad (7.64)$$

$$W_c = \begin{bmatrix} \bar{\rho} \\ \bar{\rho}\tilde{u} \\ \bar{\rho}\tilde{v} \\ \bar{\rho}\tilde{w} \\ \bar{\rho}\tilde{e}_t \end{bmatrix} \quad (7.65)$$

$$E_i = \begin{bmatrix} \bar{\rho}\tilde{u} \\ \bar{\rho}\tilde{u}^2 + \bar{p} \\ \bar{\rho}\tilde{u}\tilde{v} \\ \bar{\rho}\tilde{u}\tilde{w} \\ (\bar{\rho}\tilde{e}_t + \bar{p})\tilde{u} \end{bmatrix} \quad (7.66) \quad E_v = \frac{1}{Re_\infty} \begin{bmatrix} 0 \\ \hat{\tau}_{xx} - \sigma_{xx} \\ \hat{\tau}_{xy} - \sigma_{xy} \\ \hat{\tau}_{xz} - \sigma_{xz} \\ (\tilde{u}\hat{\tau}_{xx} + \tilde{v}\hat{\tau}_{xy} + \tilde{w}\hat{\tau}_{xz} - \hat{q}_x - \hat{Q}_x) \end{bmatrix} \quad (7.67)$$

$$F_i = \begin{bmatrix} \bar{\rho}\tilde{v} \\ \bar{\rho}\tilde{v}\tilde{u} \\ \bar{\rho}\tilde{v}^2 + \bar{p} \\ \bar{\rho}\tilde{v}\tilde{w} \\ (\bar{\rho}\tilde{e}_t + \bar{p})\tilde{v} \end{bmatrix} \quad (7.68) \quad F_v = \frac{1}{Re_\infty} \begin{bmatrix} 0 \\ \hat{\tau}_{yx} - \sigma_{yx} \\ \hat{\tau}_{yy} - \sigma_{yy} \\ \hat{\tau}_{yz} - \sigma_{yz} \\ (\tilde{u}\hat{\tau}_{yx} + \tilde{v}\hat{\tau}_{yy} + \tilde{w}\hat{\tau}_{yz} - \hat{q}_y - \hat{Q}_y) \end{bmatrix} \quad (7.69)$$

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

$$G_i = \begin{bmatrix} \bar{\rho} \tilde{w} \\ \bar{\rho} \tilde{w} \tilde{u} \\ \bar{\rho} \tilde{w} \tilde{v} \\ \bar{\rho} \tilde{w}^2 + \bar{p} \\ (\bar{\rho} \tilde{e}_i + \bar{p}) \tilde{w} \end{bmatrix} \quad (7.70) \quad G_v = \frac{1}{Re_\infty} \begin{bmatrix} 0 \\ \hat{\tau}_{zx} - \sigma_{zx} \\ \hat{\tau}_{zy} - \sigma_{zy} \\ \hat{\tau}_{zz} - \sigma_{zz} \\ \tilde{u} \hat{\tau}_{zx} + \tilde{v} \hat{\tau}_{zy} + \tilde{w} \hat{\tau}_{zz} - \hat{q}_z - \hat{Q}_z \end{bmatrix} \quad (7.71)$$

$$S_{SGS} = [0, 0, 0, 0, \Psi]^T \quad (7.72)$$

where the subgrid-scale stress tensors are

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

$$\sigma_{xx} = -\frac{2\mu_t}{3} \left(2 \frac{\partial \tilde{u}}{\partial x} - \frac{\partial \tilde{v}}{\partial y} - \frac{\partial \tilde{w}}{\partial z} \right) + \bar{\rho} \left(\tilde{u}\tilde{u} - \tilde{u}\tilde{u} \right) - \mathbb{Q} + \mathbb{R}$$

$$\sigma_{yy} = -\frac{2\mu_t}{3} \left(2 \frac{\partial \tilde{v}}{\partial y} - \frac{\partial \tilde{u}}{\partial x} - \frac{\partial \tilde{w}}{\partial z} \right) + \bar{\rho} \left(\tilde{v}\tilde{v} - \tilde{v}\tilde{v} \right) - \mathbb{Q} + \mathbb{R}$$

$$\sigma_{zz} = -\frac{2\mu_t}{3} \left(2 \frac{\partial \tilde{w}}{\partial z} - \frac{\partial \tilde{u}}{\partial x} - \frac{\partial \tilde{v}}{\partial y} \right) + \bar{\rho} \left(\tilde{w}\tilde{w} - \tilde{w}\tilde{w} \right) - \mathbb{Q} + \mathbb{R}$$

$$\sigma_{xy} = -\mu_t \left(\frac{\partial \tilde{u}}{\partial y} + \frac{\partial \tilde{v}}{\partial x} \right) + \bar{\rho} \left(\tilde{u}\tilde{v} - \tilde{u}\tilde{v} \right)$$

$$\sigma_{xz} = -\mu_t \left(\frac{\partial \tilde{u}}{\partial z} + \frac{\partial \tilde{w}}{\partial x} \right) + \bar{\rho} \left(\tilde{u}\tilde{w} - \tilde{u}\tilde{w} \right)$$

$$\sigma_{yz} = -\mu_t \left(\frac{\partial \tilde{v}}{\partial z} + \frac{\partial \tilde{w}}{\partial y} \right) + \bar{\rho} \left(\tilde{v}\tilde{w} - \tilde{v}\tilde{w} \right)$$

$$\mathbb{Q} = \frac{\bar{\rho}}{3} \left(\left(\tilde{u}\tilde{u} - \tilde{u}\tilde{u} \right) + \left(\tilde{v}\tilde{v} - \tilde{v}\tilde{v} \right) + \left(\tilde{w}\tilde{w} - \tilde{w}\tilde{w} \right) \right)$$

(7.73)

$$\mathbb{R} = \frac{2}{3} \bar{\rho} C_t \Delta^2 |S(\tilde{u})|^2$$

$$\mu_t = \bar{\rho} C_d \Delta^2 |S(\tilde{u})|$$

and the subgrid-scale heat flux tensors are

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

$$\begin{aligned}
\hat{Q}_x &= -\frac{\gamma\mu_t}{Pr_t} \frac{\partial \tilde{T}}{\partial x} + \frac{\widetilde{\bar{p}u} - \bar{\bar{p}}\tilde{u}}{(\gamma-1)} \\
\hat{Q}_y &= -\frac{\gamma\mu_t}{Pr_t} \frac{\partial \tilde{T}}{\partial y} + \frac{\widetilde{\bar{p}v} - \bar{\bar{p}}\tilde{v}}{(\gamma-1)} \\
\hat{Q}_z &= -\frac{\gamma\mu_t}{Pr_t} \frac{\partial \tilde{T}}{\partial z} + \frac{\widetilde{\bar{p}w} - \bar{\bar{p}}\tilde{w}}{(\gamma-1)}
\end{aligned} \tag{7.74}$$

Finally, the dynamic turbulent viscosity μ_t and Prandtl number Pr_t are determined by the method described above and the subgrid-scale source term in filtered energy equation is

$$\begin{aligned}
\Psi &= -\tilde{u} \left(\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y} + \frac{\partial \sigma_{xz}}{\partial z} \right) - \tilde{v} \left(\frac{\partial \sigma_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \sigma_{yz}}{\partial z} \right) - \tilde{w} \left(\frac{\partial \sigma_{xz}}{\partial x} + \frac{\partial \sigma_{zy}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} \right) \\
&+ \left(\bar{\bar{p}} \left(\frac{\partial \tilde{u}}{\partial x} + \frac{\partial \tilde{v}}{\partial y} + \frac{\partial \tilde{w}}{\partial z} \right) - \overline{\bar{p} \left(\frac{\partial \tilde{u}}{\partial x} + \frac{\partial \tilde{v}}{\partial y} + \frac{\partial \tilde{w}}{\partial z} \right)} \right) + c_\epsilon \bar{\bar{p}} \frac{\left[1/2 (\sigma_{xx} + \sigma_{yy} + \sigma_{zz}) \right]^{3/2}}{\Delta}
\end{aligned} \tag{7.75}$$

7.5 COMMUTATIVE DISCRETE FILTERING OPERATOR ON UNSTRUCTURED GRIDS

One of the pressing fundamental problems impeding the applications of LES in a reliable way to real engineering problems, where typically finite-volume codes on unstructured grids are used, is the construction of commutative discrete filters. Recent work by Marsden et al. [184] resulted in a framework for the construction of filters on unstructured grids which commute with differentiation operator to a potentially arbitrarily high order. The key concept in the construction of a commutative discrete filter is that the

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

weights of the filter components exhibit vanishing moments. Their analysis indicated that the larger the number of vanishing moments, the higher the order of the commutation error. They also demonstrated a scheme for constructing a filter operator with a second-order commutation error. However, because their method heavily relies on the geometric distribution of grid nodes, it is quite complicated for a 3D mesh, even impossible near boundaries.

In this study, the unstructured grid filtering was adapted from a new filtering approaches developed by A. Haslbacher [185]. This new filtering method is based on the following observation: The conditions derived by Marsden for a filtering a function to a given order of commutation error are formally identical to the conditions for reconstructing the gradient of a function to the same order of truncation error. Therefore, the task of constructing the commutative filter operators may be transferred to constructing and suitably reformulating gradient-reconstruction operators.

Firstly we describe the least-squares gradient reconstruction, which serves as the basis for the new filtering method. Suppose we wish to reconstruct the gradient of a dependent variable at vertex P in figure 7.1. The running index i donates the points which are included in the reconstruction stencil at vertex P . The reconstruction will include at least the nearest neighbors of P , namely those vertices which are linked by an edge to vertex P . The stencil may be extended recursively to take into account m th-nearest neighbors, which are those vertices which are linked by an edge to $(m-1)$ th-nearest neighbors of vertex P and are not already included in the $(m-r)$ th-nearest neighbors, where $0 \leq r \leq 2 \leq m$.

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

Figure 7.1 Fragment of 2D unstructured triangular grid illustrating gradient reconstruction stencil at vertex P .

In the following, we define by d_0 the degree of vertex P , i.e., the number of neighbors included in gradient reconstruction at vertex P .

The least-squares gradient reconstruction procedure is based on approximating the variation of a dependent variable Φ along an edge linking vertices P and i by a truncated Taylor series, e.g., for a linear reconstruction,

$$\Phi_i = \Phi_0 + (\nabla \Phi)_0 \cdot \Delta r_{0i}, \quad 1 \leq i \leq d_0 \quad (7.76)$$

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

where $\Delta r_{0i} = r_i - r_0$ and r is the position vector. The application of Equation (7.76), or corresponding higher-order approximations, to all edges incident to vertex P gives a system of linear equations for the derivatives at vertex P ,

$$Ax = b \quad (7.77)$$

where A is a $d_0 \times n_0$ matrix of geometrical terms, x is an n_0 vector containing derivatives to be reconstructed, b is a d_0 vector of dependent variables, and n_0 is the number of derivatives reconstructed at vertex P . Since the degree d_0 is usually larger than the number of unknown derivatives n_0 , above equation is solved for x in a least-squares fashion. And the detailed solution expressions can be found in [185].

The least-squares gradient-reconstruction method can be turned into a filtering method simply by modifying Equation (7.76), so that Φ_0 is no longer a point value, but represents an interpolated or a filtered value $\bar{\Phi}_0$,

$$\Phi_i = \bar{\Phi}_0 + (\nabla \Phi)_0 \cdot \Delta r_{0i}, \quad 1 \leq i \leq d_0 \quad (7.78)$$

Such that the filtered value $\bar{\Phi}_0$ is affixed to the vector of unknowns x . The resulting system of equations can be solved using the method described in former paragraphs. This leads to an expression for the filtered values in the form of a weighted sum,

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

$$\bar{\Phi}_0 = \sum_{i=1}^{d_0} \bar{\omega}_{0i} \Phi_i \quad (7.79)$$

The weight for linear filtering in three dimensions is given by [186]

$$\bar{\omega}_{0i} = \frac{1}{r_{44}^2} \left[1 - \left(\frac{r_{44}}{r_{11}} - \frac{r_{12}}{r_{11}} \frac{r_{24}}{r_{22}} - \frac{r_{34}}{r_{33}} \frac{r_{13}r_{22} - r_{12}r_{23}}{r_{11}r_{22}} \right) \Delta x_{0i} - \frac{r_{24}r_{33} - r_{23}r_{34}}{r_{22}r_{33}} \Delta y_{0i} - \frac{r_{34}}{r_{33}} \Delta z_{0i} \right] \quad (7.80)$$

where the geometric terms r_{ij} are given by the following expressions:

$$r_{11} = \sqrt{\sum_{i=1}^{d_0} \Delta x_{0i}^2} \quad (7.81)$$

$$r_{12} = \frac{1}{r_{11}} \sum_{i=1}^{d_0} \Delta x_{0i} \Delta y_{0i} \quad (7.82)$$

$$r_{13} = \frac{1}{r_{11}} \sum_{i=1}^{d_0} \Delta x_{0i} \Delta z_{0i} \quad (7.83)$$

$$r_{14} = \frac{1}{r_{11}} \sum_{i=1}^{d_0} \Delta x_{0i} \quad (7.84)$$

$$r_{22} = \sqrt{\sum_{i=1}^{d_0} \Delta y_{0i}^2 - r_{12}^2} \quad (7.85)$$

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

$$r_{23} = \frac{1}{r_{22}} \left(\sum_{i=1}^{d_0} \Delta y_{0i} \Delta z_{0i} - r_{12} r_{13} \right) \quad (7.86)$$

$$r_{24} = \frac{1}{r_{22}} \left(\sum_{i=1}^{d_0} \Delta y_{0i} - r_{12} r_{14} \right) \quad (7.87)$$

$$r_{33} = \sqrt{\sum_{i=1}^{d_0} \Delta z_{0i}^2 - (r_{13}^2 + r_{23}^2)} \quad (7.88)$$

$$r_{34} = \frac{1}{r_{33}} \left(\sum_{i=1}^{d_0} \Delta z_{0i} - (r_{13} r_{14} + r_{23} r_{24}) \right) \quad (7.89)$$

$$r_{34} = \sqrt{d_0 - (r_{14}^2 + r_{24}^2 + r_{34}^2)} \quad (7.90)$$

In the interests of improving the spectral behavior of Equation (7.79), it may be replaced by

$$\bar{\Phi}_0 = \bar{\omega}_{00} \Phi_0 + (1 - \bar{\omega}_{00}) \sum_{i=1}^{d_0} \bar{\omega}_{0i} \Phi_i \quad (7.91)$$

where $l \leq \bar{\omega}_{00} \leq l$ is a user-defined parameter, normally defined as 0.5.

The first issue that needs to be addressed in constructing an operational filtering using above method is the construction of stencil used for the filtering. For every vertex P , all of its l th neighboring vertexes are used for constructing the filter stencil. If any of

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

following rules is violated, its 2th neighboring vertexes will be further included to constructing the filter stencil.

1). The number of its l th neighbors does not give the required stencil size, which is given by

$$d_{0,\min} = \frac{1}{D!} \prod_{i=1}^D (q + i) \quad (7.92)$$

where q denotes the degree of the highest derivative included in x . D the space dimension.

Typically, for a second-order filtering operator in three dimensional space, $d_{0,\min}$ is 4.

2). r_{11} , r_{22} , r_{33} , or r_{44} is zero.

7.6 COOPERATING LES WITH IMM

For the simulation of laminar flows or direct numerical simulation (DNS) of turbulence, the IMM formulation discussed in Chapter 5 can satisfy the boundary conditions on the immersed interfaces “exactly” to the second-order accuracy of the spatial discretization scheme for the Navier-Stokes equations. Also, the pressure (as well as continuity) boundary conditions are implicitly satisfied as the results of linearization of the velocity near the interface. However, for the SGS models the treatment of the turbulent viscosity near the immersed boundary has to be considered. The eddy viscosity μ_t and turbulence Prandtl number Pr_t at the boundary points are required in the computation of diffusion fluxes near the interface, and their calculations are no longer straightforward due to the

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

presence of solid body on the fixed grid. In the case of a dynamic SGS model for example, the evaluation of all test-filtered quantities in the vicinity of the immersed boundary, involves points from the interior of the solid body. A schematic of the test-grid stencil on a 2D unstructured grid with an immersed boundary is shown in Fig. 7.2. A direct computation ignoring the interface can be problematic and the resulting eddy viscosity may be contaminated by the non-physical solution at the solid points [187]. Explicit modification of the filtering operator or redefining the test-grid cells as in a cut-cell formulation can be considered, but a large number of cases will be introduced and the algorithm becomes complicated. To avoid complex modifications of the filtering operator at these points, a linear reconstruction procedure similar to that used for the velocity field is also applied on μ_t and Pr_t , which follows the same method proposed in Chapter 5. The linearised eddy viscosity and Pr_t are only approximations of their respective realistic values, which usually do not follow a linear distribution near the solid boundary. In [187] extensive testing of this approach in LES over immersed boundaries has been performed and the error in the results due to the linearised eddy viscosity near the solid boundary has been found to be small.

CHAPTER 7 – LARGE-EDDY SIMULATION FOR COMPRESSIBLE FLOW

Figure 7.2 schematic of the test-grid stencil on a 2D unstructured grid with an immersed boundary. Only 1th neighboring nodes are used to construct the filtering operator for node P . Extrapolated ghost values at i and j will be used to evaluate the filtering value of node P .

CHAPTER EIGHT

NUMERICAL RESULTS AND DISCUSSION

This chapter presents the validation results for the serial and parallel single grid (SG) methods, the multigrid (MG) method, the CSD solver and the immersed membrane method (IMM). In addition, it also presents the numerical simulation of steady/unsteady flows between two rigid disks corotating in a fixed cylindrical enclosure. And finally results from the computation for aeroelastic flutter of an ONERA M6 wing are presented. In this work, all computations are performed on the SGI Origin 3400 machine with 32 processors based on the Silicon Graphics Scalable Distributed-Shared-Memory (DSM) Multiprocessing architecture. It has a system bandwidth of 44.8 GB/sec, 32 GB onboard memory, 800 MFlops/CPU, 8 nodes and each node has four MIPs 64 Bit R12000 400MHz/8MB RISC CPUs.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

8.1 VALIDATION OF BASELINE COMPRESSIBLE FLOW SOLVER WITH IMM

8.1.1 Steady Flow Induced By a Rotating Sphere in Quiescent Air – Order of Accuracy

First of all, to determine the overall accuracy of the proposed methods, we carry out a grid convergence study for a test problem, which is a three-dimensional analogue of the problem used by Gilmanov *et al.* [188]. In this case, we simulate flow induced by a sphere of radius R_0 , rotating at constant angular velocity Ω about the z-axis in a nearly incompressible, viscous fluid with kinematic viscosity ν . The Reynolds number for this flow is defined as $Re = \Omega R_0^2 / \nu$. For Reynolds numbers in the range $Re = 1 - 100$, benchmark solutions for this problem have been reported by Dennis *et al.* [189] who solved numerically the steady, axisymmetric Navier-Stokes equations in polar coordinates using a vorticity-stream function formulation. In our studies, the Reynolds number is set to $Re = 100$, and we solve the full three-dimensional and unsteady flow problem with the sphere starting to rotate impulsively from quiescence relative to the stationary Cartesian coordinate system.

The computational domain is a $(10R_0)^3$ cube with its center located at $x = 5.0R_0$, $y = 5.0R_0$ and $z = 5.0R_0$. Four uniformly spaced and successively finer mesh sizes with 20^3 , 40^3 , 80^3 , and 160^3 grid points respectively are used for error analysis, and the finest-mesh

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

solution is considered to be the ‘exact’ solution. The surface of the sphere is discretized with an unstructured triangular mesh consisting of 14,612 elements. The sphere is placed at the center of the cubical domain and starts to rotate impulsively at $t = 0$ with constant rotational velocity Ω about the z-axis. On all the grids the same physical time step ($\Delta t = 0.01T$) is employed in order to concentrate on the spatial resolution of the method, as done in [188]). For all the grids, the simulation is continued for one complete period, at the end of which the L_∞ and L_q norms of the u -velocity errors are calculated as follows:

$$\varepsilon_N^\infty = \max_{i=1, N^3} |u_i^{(N)} - u_i^e|, \quad \varepsilon_N^q = \left[\frac{1}{N^3} \sum_{i=1}^{N^3} |u_i^{(N)} - u_i^e|^q \right]^{1/q} \quad (8.1)$$

where ε_N^∞ and ε_N^q are the infinity and q th error norms, $u_i^{(N)}$ is the u -velocity component at the i th node of the N^3 mesh, and u_i^e is the ‘exact’ velocity field calculated on the 160^3 grid. The results of the grid convergence study are summarized in Figure 8.1, which shows the variation of the L_∞ , L_1 and L_2 norms of errors with grid spacing in logarithmic coordinates. The lines with slope one and two are also given as reference. It is evident from Figure 8.1 that the method is second order accurate. To further demonstrate the accuracy of our method, we also use the Richardson estimation procedure to study the accuracy of the solver as in [188]. Let f^N denote the numerical solution on the N^3 mesh. Assume that the discrete solution is a γ -order approximation to its value f^{exact} , and the flow field is continuous and has no singularity points, then we

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

have

$$\gamma = \frac{\log\left(\frac{\|f^N - f^{N/2}\|}{\|f^{N/2} - f^{N/4}\|}\right)}{\log 2} \quad (8.2)$$

Figure 8.1 Convergence of the L_∞ , L_1 and L_2 error norms for the velocity field induced by a rotating sphere. Slope 1 and Slope 2 are the reference lines for 1-order and 2-order accuracy respectively.

where $\|\cdot\|$ denotes an error norm (L_∞ , L_1 or L_2). If $\gamma \approx 2$ the solution is second-order accurate. We apply the above procedure for $N = 160$ (using solutions obtained on meshes 40^3 , 80^3 , and 160^3) to calculate γ for successively refined meshes. And we use all three

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

norms to compute the error and the results are summarized in Table 8.1, which strongly supports our assertion about the second-order accuracy of our method.

Table 8.1 Rate of convergence γ calculated for different error norms

Norm	Grids
	$40^3, 80^3, 160^3$
L_∞	1.82
L_1	2.84
L_2	2.15

Figure 8.2 shows several snapshots of instantaneous streamlines on the plane at $y = 2.5$. To compare the result quantitatively with those in [188, 189], we compute the angle θ_s between the z-axis and the line that originates from the center of the sphere and passes through the center of the toroidal vortex ring (see Figure 8.3 for definition). In Table 8.2, we compare our θ_s with those reported in [188, 189] and the agreement is very good.

Table 8.2 Comparison of the measured [188, 189] and calculated angle θ_s

Reynolds Number	θ_{Dennis}	$\theta_{Gilmanov}$	θ_{calc}	$\varepsilon(\%)$
100	73.8	72.1	74.3	0.68(with [189]), 3.05(with [188])

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.2 Instantaneous snapshots of streamlines at the $y = 2.5$ plane depicting the early stages of flow evolution toward steady state for $Re = 100$. Time is measured from the start of the impulsive rotation and T is the rotation period of the sphere.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.3 The angle θ_s between the z-axis and the radius that passes through the centre of the toroidal vortex ring.

8.1.2 Flow Induced by an Oscillating Sphere in a Closed Cavity

The grid convergence study is also performed for the case where a solid oscillating sphere is immersed in a fluid enclosed with a cube with solid walls. This is to demonstrate the capability of the method in handling objects moving with large displacements. The sphere of diameter $D = 1.0$ units is placed initially at the center of the cube of dimension 2×2 units and oscillates horizontally with a nondimensional time period of 1.0 and an amplitude of $0.25D$. The oscillation is effected by moving the sphere

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

as a rigid body with velocity given by

$$u = 0.25\pi \sin(2\pi t), v = w = 0 \quad (8.3)$$

The Reynolds number (based on the sphere diameter and maximum velocity) has been set to 20. The following sequence of grid sizes is employed in performing the error analysis: 20^3 , 40^3 , 80^3 and 160^3 . And the result on the 160^3 mesh is taken to be the “exact” solution for this case. A small time step of $\Delta t = 0.005$ is chosen for all these simulations in order to minimize the effect of temporal errors on the solution. The simulations are carried out for one oscillation period and the velocity components at each grid point are recorded for all the meshes under consideration at the end of the period. The instantaneous streamlines at the end of an oscillation cycle ($t=1.0T$) are shown in Figure 8.4. Figure 8.5 presents the error norms for the three meshes (20^3 , 40^3 and 80^3). As can be noted, the convergence rates of errors in the simulations are close to the Slope 2 reference line, indicating second-order-accuracy.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.4 Instantaneous streamlines at $t = 1.0T$ for flow induced by an oscillating sphere in a closed cube filled with compressible and viscous fluid.

Figure 8.4 Instantaneous streamlines at $t = 1.0T$ for flow induced by an oscillating sphere in a closed cube filled with compressible and viscous fluid.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.5 Convergence of the L_∞ , L_1 and L_2 error norms for the velocity field of oscillating sphere in a closed cube. Slope 1 and Slope 2 are the reference lines for 1-order and 2-order accuracy respectively.

8.1.3 Steady and Unsteady Flow past a Circular Cylinder

Flow past a circular cylinder is a classical benchmark problem, which has been the subject of many theoretical, experimental and computational investigations due to its

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

simple geometry and its representative behavior of general bluff body flows. Both low and high Reynolds numbers are used to demonstrate and examine the performance, accuracy and robustness of the parallel-MG compressible solver with the implementation of IMM algorithm for steady and unsteady flows. The computational domain and the immersed cylinder are shown in Figure 8.6. The parallel calculations are performed on the SGI Origin 3400 machine. Three different grids are generated, which have 84,054, 122,202 and 182,536 nodes respectively, and the immersed cylinder surface is discretized using 33,578 triangular elements. All the parameters for the three simulation runs are kept exactly the same and the simulation is run until non-dimensional time, $t = 150.0$ for $Re = 200$ with an inflow Mach number of 0.2. The computed value of lift coefficient, C_l , is used as a criterion for convergence as shown in Figure 8.7.

Figure 8.6 The computational domain and the immersed cylinder for viscous flow past a circular cylinder.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.7 Lift coefficients versus physical time using 3 different grids for flow over a circular cylinder (Parallel-MG, Preconditioned, $Re = 200$, inflow $Mach=0.2$).

Comparing the results shown in Figure 8.7, the peak value of C_l obtained using 84,054 nodes is 0.56 as compared to 0.64 obtained by both grids of 122,202 and 182,536 nodes. And similarly, comparing the results obtained by both 122,202 and 182,536 nodes, the value of C_l does not deviate from each other significantly. Since there is not much difference in the results when the nodes density is increased from 122,202 to 182,536 nodes and to minimize computational time, a grid size of 122,202 nodes is employed in the present work.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

8.1.3.1 Steady flow

The low Reynolds number specified in this computation is 40.0 and the inflow Mach number is set to be a very small value. Thus the flow can be considered as steady two-dimensional, which is started from quiescent initial condition in the simulation. It is run using the low speed preconditioning with the parallel-MG method as shown in the convergence history plot in Figure 8.8. It is noted that for the same CPU time there is significant improvement in residual reduction using the preconditioning, i.e., improvement in real convergence rate.

Figure 8.9 (a) and (b) show the streamline plots in the wake region obtained with and without the preconditioning method respectively. Based on qualitative comparison with the experimental result of Dyke *et al.* [190], the wake formed behind the cylinder predicted by the preconditioning method has better agreement with Dyke's data than the non-preconditioning one.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.8 Convergence histories for steady flow over a circular cylinder.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.9 Streamlines plots ($z=0.25$) for flow over a three-dimensional stationary circular cylinder at $Re = 40$ and inflow $Mach=0.01$ using (a) Parallel-MG and preconditioning, (b) Parallel-MG without preconditioning.

A quantitative comparison of the aspect ratio (separation bubble length, S / cylinder diameter, d) with the experimental results obtained by Nishioka and Sato [191] is also carried out. Figure 8.10 shows the aspect ratio versus Re . With a Re of 40.0, an aspect ratio of 2.35 is obtained, which agrees well with our preconditioned result (as shown in Figure 8.9 (a)). As can be observed, the low speed preconditioning method not only gives better convergence rate, it also helps to improve the quality of the numerical results when the flow speed is extremely low. This favorable characteristic is very crucial for

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

compressible unsteady flow computation, especially for the cases where high speed flow field is embedded in low speed flow region.

Figure 8.10 Length of separation bubbles behind cylinder vs. Re [191].

8.1.3.2 Unsteady flow

The purpose here is to validate and assess the capability of the current parallel Navier-Stokes solver utilizing multi-grid and low speed preconditioning method as the

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

basic convergence acceleration techniques. The computed results are compared with numerical ones obtained by other researchers, and with available experimental results. . The computed lift and drag coefficients on the cylinder versus non-dimensional time with parallel-MG for an 8-partition mesh are shown in Figure 8.11.

Figure 8.11 Lift and drag coefficients versus physical time for flow over a circular cylinder using 3-level Parallel-MG and preconditioning (number of sub-iterations = 60 V-cycles, $Re = 200$, inflow Mach = 0.2).

A pronounced asymmetric wake begins to appear at non-dimensional time of 30. And the flow becomes completely periodic at a time of 63. It is found that the number of sub-iterations for MG is much less than that required by single grid (SG) computation,

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

which signifies that the MG method takes a shorter time than the SG to produce the vortex shedding phenomenon, thus less CPU time is needed for the flow to become fully periodic. Figures 8.12 and 8.13 present the instant streamlines and Mach-number contours in one cycle of the von Kármán vortex shedding. Figure 8.14 shows the convergence history in terms of numbers of subiterations in each physical time step for single grid, MG and MG with preconditioning method for a Reynolds number equal to 200. It is obvious that the combination of preconditioning and MG contributes significantly to the improvement in convergence within each physical time step, which is very useful for efficiently computing 3D unsteady flows. The lift coefficient, C_l , drag coefficient, C_d , and Strouhal number, S_t , obtained in this work are ± 0.65 , 1.38 ± 0.046 and 0.196, respectively, and they agree well with numerical solutions obtained by other researchers as well as with experimental measurements [128, 161-163]. And these results are tabulated in Table 8.3.

Table 8.3 Lift and drag coefficients and Strouhal number for unsteady flow over a three-dimensional circular cylinder ($Re = 200$).

Researchers	C_l	C_d	S_t
Present (parallel-MG grid)	± 0.65	1.38 ± 0.046	0.196
Tai and Zhao (parallel-MG grid, incompressible solvers) [128]	± 0.64	1.31 ± 0.041	0.195
Liu et al. [161]	± 0.69	1.31 ± 0.049	0.192
Williamson (Expt.) [162]	--	--	0.197
Wille (Expt.) [163]	--	1.30	--

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.12 Streamline patterns showing one cycle of vortex shedding using 3-level Parallel-MG and preconditioning ($z=0.25$, number of sub-iterations = 60 V-cycles, $Re = 200$, inflow Mach=0.2).

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

The performance of the parallel solver for simulating unsteady flow over a cylinder with $Re = 200$ using both parallel SG and MG is estimated based on the speedup characteristics, efficiency of parallelization and comparison between percentage computation and communication time, as shown in Table 8.4. The speed-up and parallel efficiency of the proposed method is found to be reasonably good. Although parallel MG has slightly low speed-up compared with parallel SG, the MG computation is still much more efficient because it requires much less number of sub-iterations for every physical time step.

Table 8.4 Performance for parallel computation of unsteady flow past a circular cylinder ($Re = 200$, $M=0.3$ with preconditioning)

Performance measuring techniques	Single Grid				Multigrid			
	Number of CPU				Number of CPU			
	2	4	8	16	2	4	8	16
Speed-up	1.77	3.48	7.02	13.4	1.72	3.29	6.39	10.18
Efficiency	0.89	0.88	0.85	0.81	0.88	0.83	0.79	0.72
Computation time and communication time (% to total simulation wall-clock time)	99.3	95.2	87.7	79.3	97.2	91.1	82.4	70.1

Figure 8.13 Mach number contours showing one cycle of vortex shedding using 3-level Parallel-MG and preconditioning ($z=0.25$, number of sub-iterations = 60 V-cycles, $Re = 200$, inflow Mach=0.2)

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.14 Unsteady flow convergence history plot (MG used “V” cycle, number of sub-iterations = 60, $Re = 200$, inflow Mach=0.2).

8.1.4 Flow over an Immersed Fixed Membrane

This case is to validate and assess the capability and accuracy of the developed method for thin structure problems. The model examined in this case is a 3D channel flow with a sinus cavity in the middle of the bottom wall. A rigid membrane is attached to the rigid

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

channel just before the sinus cavity. The channel has a length of $10L$ and a width of $1L$ (L is equal to 20mm). The radius of the sinus cavity is $0.5L$. The immersed membrane has a length of $0.5L$ and thickness of $0.5L/100$, and it is attached to the bottom wall at an angle of $\alpha = 42.5^\circ$. In the region near the membrane the mesh is further refined in order to capture the fine details of the flow. The immersed membrane is discretized into triangular cells. A close-up view of the mesh and the immersed membrane is shown in Figure 8.15. The inflow Mach number is 0.3 and the Reynolds number is 100 . The results are then compared with those using an internal boundary calculated by the baseline preconditioned compressible parallel-MG solver. The internal boundary has the same geometry as the immersed membrane, and it is under the same flow conditions as used by the immersed membrane method.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.15 Geometry of the computational domain for the flow over an immersed fixed membrane (1 unit length = 20 mm).

The convergence history of the simulation given in Figure 8.16 shows that the new solver based on the IMM actually converges faster and better than the baseline solver. Figure 8.17 confirms that the two flow fields have the same number of vortices with similar shapes, and the u -velocity profiles at the same locations agree well with each other as found in Figure 8.18. Table 8.5 compares the two flow fields quantitatively, which

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

demonstrates that the results obtained by the IMM agree well with those obtained by the baseline Navier-Stokes solver using an internal boundary.

Figure 8.16 Convergence histories with immersed membrane and internal boundary.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.17 (a. above) Streamlines on the channel central plane with the immersed membrane represented by the IMM; (b. below) flow field with an internal boundary computed by baseline solver.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.18 (a. above) Comparison of velocity profiles at $X=0, Z=0.5$ (b. below)
Comparison of velocity profiles at $X=1, Z=0.5$.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Table 8.5 Properties of vortices of flow field with immersed membrane and flow field with internal boundary.

	Immersed membrane		Internal boundary	
	center of the primary vortex	center of the secondary vortex	center of the primary vortex	center of the secondary vortex
X (mm)	0.932 L	0.067 L	0.923 L	0.047 L
Y (mm)	0.272 L	-0.031 L	0.273 L	-0.027 L
Vorticity	-5.013	0.126	-4.997	0.117

8.2 VALIDATION OF THE COMPUTATIONAL STRUCTURAL DYNAMICS SOLVER

8.2.1 Deformation of Point Loaded Fixed-Free Cantilever Structures

8.2.1.1 Two-Dimensional Cantilever

A standard problem in structural mechanics is that of a fixed-free cantilever supporting an applied load at the free end [81, 192, 193]. The fixed-free cantilever is shown in Figure 8.19, where $b = 2.0$ is the breadth, $L = 20.0$ the length of the cantilever and F the applied load. It is assumed that the depth $d = 1.0$. The static solution to this problem given by Timoshenko and Goodier [135] allows a slight distortion at the fixed end of the cantilever, whereas the solution given by Fenner [134] allows no such phenomenon. This test case

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

requires no such displacement or distortion at the fixed end of the cantilever, hence at $x = L$ the y displacement at the free end of the cantilever according to Fenner [134] is given by

$$d_y = -\frac{4FL^3}{Edb^3} \quad (8.4)$$

where E is Young's modulus and d is the depth of the cantilever. The fixed-free cantilever has a load of 200 N at the free end, as depicted by Figure 8.19. Note that the gravity effect is not considered in this study.

Figure 8.19 Schematic of flexural (bending) deformation test of fixed-free cantilever

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

The static solution given in equation (8.4) is independent of Poisson's ratio and is applicable to a cantilever undergoing pure flexure, i.e. no axial loads are supported and the out of plane load on the cantilever is zero. Thus for comparison with the analytic solution a zero Poisson's ratio is assumed. With the parameters as given in Table 8.6, equation (8.4) gives the static displacement in y direction at the tip of the cantilever as -0.08 m. Before embarking on a dynamic problem, a grid convergence study for the static displacement problem is carried out. The domain is meshed using triangular elements. The following five meshes are studied for increasingly higher mesh density:

- Coarse mesh, 20×2 elements and 33 nodes.
- First refinement, 40×4 elements and 105 nodes.
- Second refinement, 80×8 elements and 369 nodes.
- Third refinement, 100×10 elements and 561 nodes.
- Final refinement, 200×20 elements and 2121 nodes.

For all of the grids the simulation is initiated by exerting the same load on the right tip of the cantilever until the solution is converged. The percentage errors in the y displacement are shown in Figure 8.20. This percentage error is defined by $perr = \Delta y / Y$, where Δy is the difference between computed displacement and analytical displacement under a given external force. Y is the analytical displacement. From this picture, the percentage errors of less than 1.5% for the second mesh and less than 0.1% for the final mesh are observed. Thus the second mesh is considered to be sufficiently accurate and is used for the rest of the analysis. Figure 8.21 shows the stress distribution in the equilibrium state.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Table 8.6 Computational parameters for the 2D cantilever

2D fixed-free cantilever computational parameters	Value
Load, F	200 N
Length, L	20.0 m
Breadth, b	2.0 m
Density, ρ	2600.0 kg/m ³
Young's modulus, E	10 MPa
Poisson's ratio, ν	0.0

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.20 Mesh refinements versus percentage error

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.21 Direct and shear stress distribution in the equilibrium state for the static point loaded free-fixed 2D cantilever

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

In order to test the capability of this method for predicting dynamics of solid structures, we also perform a dynamic bending simulation for the fixed-free cantilever. We use a dynamic load, which is a sinusoidal function of time t :

$$F = 200 \sin(0.05t) \quad (8.5)$$

With the other parameters as given in Table 8.6 and depth $d = 1.0 \text{ m}$, equation (8.4) also gives the maximum displacement in y direction at the tip of the cantilever as 0.08 m . The simulation is kept running for a total of 2315 seconds. The cantilever tip displacement-time history in y direction is then plotted in Figure 8.22, which has a maximum displacement of 0.081 m in good agreement with the analytic solution.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.22 Displacement history of the tip in y direction for dynamic point loading on free-fixed 3D cantilever

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

8.2.1.2 Three-Dimensional Cantilever

The 2D problem studied in previous section is then extended to three dimensions and all of the computational parameters are given in Table 8.7.

Table 8.7 Computational parameters for the 3D cantilever

3D fixed-free cantilever computational parameters	Value
Load, F	1.0 MN
Length, L	20.0 m
Breadth, b	2.0 m
Depth, d	2.0 m
Density, ρ	2600.0 kg/m ³
Young's modulus, E	17.8 GPa
Poisson's ratio, ν	0.3

Thus the static y displacement at the neutral axis of the tip given by equation (8.4) is about -0.1124 m, which provides an upper bound to the amplitude. Similar to the 2D simulation, the 3D statically loaded cantilever is first studied. The following three meshes are tested for grid convergence and multigrid studies:

- Coarse mesh, 2400 tetrahedral elements.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

- First refinement, 3200 tetrahedral elements.
- Final refinement, 10800 tetrahedral elements.

According to our results, the solutions for single grid and 3-level multigrid are identical. And from the numerical solution for the final mesh an amplitude error of 1.36% is observed. The tip displacement for this mesh turns out to be -0.11365 m . Figure 8.23 shows the residual history and the convergence acceleration due to the 3-level multigrid method, which makes the convergence at least 3 times faster than single-grid solver in terms of CPU time.

Figure 8.23 Convergence rate acceleration given by 3-level MG method

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Next we will study the performance of the scheme for simulating dynamic loading. The natural frequency of a fixed-free cantilever [28] is given by

$$f = \frac{3.516}{2\pi L^2} \sqrt{\frac{EI}{\rho db}} \quad (8.6)$$

where I is the moment of inertia in the plane or the second moment of area of the cantilever about the neutral axis [13] and is given by

$$I = \int_{-\frac{1}{2}b}^{\frac{1}{2}b} z^2 dA = \frac{db^3}{12} \quad (8.7)$$

Using the same parameters in Table 8.7, from equation (8.6) and (8.7), the 3D fixed-free cantilever has a fundamental natural frequency of 2.1134 Hz with a period of oscillation of 0.4732 s . To capture the sinusoidal motion of the structure accurately, the time step for the solver is set to 0.05 s . The load is exerted on the initially un-deformed 3D cantilever and then kept until the static equilibrium state of the cantilever is reached, after which the load is suddenly removed. The calculated tip displacement history is depicted in Figure 8.24. The computed period is 0.4893 s , which again agrees well with the theoretical one. As pointed out by Slone in [28], for accuracy considerations, the updating of the cantilever grid in this study is based on the initial position and the total displacement

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

method. In order to accelerate the convergence rate, the multigrid method is used for all of the 3D problems.

Figure 8.24 Tip displacement history in z direction for the point loaded fixed-free 3D cantilever, and 3-level MG method was employed. In the figure, the red curve represents the oscillation history of the beam undergoing an external force. The blue one represents the oscillation history of the beam when the external force is removed

8.2.2 Three Dimensional Fixed-Free Cantilever Immersed in Fluid Flow

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

The basic configurations of the problem are shown in Figure 8.25 and the boundary conditions include an inflow boundary on the top of the domain, outflow boundary at the bottom and a non-slip wall boundary to which the cantilever is clamped. A symmetry plane is also used to divide the field into two halves because of its geometric symmetry. The fluid domain is meshed using 1,307,075 tetrahedral elements and the overall dimensions are: depth $d = 2.0\text{ m}$, breadth $b = 2.0\text{ m}$ and length $l = 20.0\text{ m}$. The cantilever is meshed using 12,196 tetrahedral elements. The Reynolds number and inflow Mach number are set to 18.7×10^5 and 0.05 respectively for this case. The material properties of the cantilever are: Young's modulus 21.0 GPa, Poisson's ration 0.3 and density 2600 kgm^{-3} . According to equation (5.3), the cantilever has a period of oscillation of 0.4356 second. The simulation time step is taken as 0.025s. The gravity force is also ignored in the simulation. The stress distribution in the cantilever in its final equilibrium state is shown in Figure 8.26 and the tip vertical displacement history is depicted in Figure 8.27, while the flow field in the final steady state is shown in Figures 8.28, 8.29 and 8.30 respectively. They show that while the beam is oscillating, a complex flow field is also developing under the beam and the wake region fluctuates with the beam accordingly.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.25 Schematic three-dimensional free-fixed cantilever in fluid flow

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.26 Direct and shear stress distribution in the equilibrium state for the fixed-free cantilever immersed in fluid

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.27 Immersed cantilever tip vertical displacement history during the FSI process

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.28 V velocity contours in XY plane, $Re=18.7*10^5$

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.29 V velocity contours and streamlines in YZ plane, $Re=18.7 \times 10^5$

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.30 Volume streamlines pattern depict the separation under the immersed cantilever, $Re=18.7 \times 10^5$

A grid convergence study on this case was also performed. Four successively finer mesh sizes with 434,572, 921,586, 1,307,075, and 1,908,265 tetrahedron cells respectively are

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

used for error analysis, and the finest-mesh solution is considered to be the ‘exact’ solution. On all the grids the same physical time step ($\Delta t = 0.01T$) is employed in order to concentrate on the spatial resolution of the method. For all the grids, the L_∞ and L_q norms of the u -velocity errors are calculated as follows,

$$\varepsilon_j^\infty = \max_{i=1,N} |u_i - u_i^e|, \quad \varepsilon_j^q = \left[\frac{1}{N} \sum_{i=1}^N |u_i - u_i^e|^q \right]^{1/q} \quad (j=1,4 \text{ } j=1 \text{ is the coarsest grid}) \quad (8.7)$$

where ε_j^∞ and ε_j^q are the infinity and q th error norms on the j th grid, u_i is the u -velocity component at the i th node of the current mesh, and u_i^e is the interpolated ‘exact’ velocity field from the results calculated on the finest grid. N is the number of grid nodes of current grid. The results of the grid convergence study are summarized in Figure 8.31, which shows the variation of the L_∞ , L_1 and L_2 norms of errors with grid spacing in logarithmic coordinates. The lines with slope one and two are also given as reference. It is evident from the figure that the method is second order accurate. To further demonstrate the accuracy of our method, we also use the Richardson estimation procedure to study the accuracy of the solver. Let f^j denote the numerical solution on the j th mesh. Assume that the discrete solution is a γ -order approximation to its value f^{exact} , and the flow field is continuous and has no singularity points, then we have

$$\gamma = \frac{\log \left(\|f^{(j)} - f^{(j-1)}\| / \|f^{(j-1)} - f^{(j-2)}\| \right)}{\log 2} \quad (8.8)$$

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

where $\| \cdot \|$ denotes an error norm (L_∞ , L_1 or L_2). If $\gamma \approx 2$ the solution is second-order accurate. We apply the above procedure for $j = 4$ (involving interpolation of solutions from $j=2, 3$ to $j=4$) to calculate γ for successively refined meshes. And we use all three norms to compute the error and the results are summarized in following table, which strongly supports our assertion about the second-order accuracy of our method.

Table 8.8 Rate of convergence γ calculated for different error norms

Norm	Grids
	$j=2, 3, 4$
L_∞	1.93
L_1	2.18
L_2	2.15

To further demonstrate and evaluate the performance of the proposed basic structural solver, both the solver and a commercial solver (ANSYS 9.0) are used to simulate the equilibrium state of the point loaded 3D fixed-free cantilever as used in the above FSI calculation, with a load of 1.0 MN. The geometry and material properties of the cantilever remain the same. Using the given parameters, the theoretical value of the

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

vertical displacement of the tip is 0.0952381 m. Four meshes are tested here:

- Grid One, 3,950 tetrahedral elements;
- Grid Two, 6,412 tetrahedral elements;
- Grid Three, 12,196 tetrahedral elements;
- Grid Four, 19,183 tetrahedral elements.

Figure 8.32 shows a comparison of the computational times (CPU times on a SGI O3400 workstation) used by the proposed basic structural solver and ANSYS 9.0. It is found that the two solvers use almost the same amount of CPU time to make the residual drop to the same level ($R/R0 = 10^{-5}$), with the current solver being slightly more efficient when the mesh density is increased. Figure 8.33 shows a comparison of grid convergence for the proposed solver and ANSYS 9.0. It can be observed that the current solver can produce faster convergence toward the exact solution than ANSYS. From the above comparisons, we can conclude that the proposed basic solver is comparable to established solvers in terms of efficiency and accuracy. With the use of multigrid, the efficiency will be significantly improved, as demonstrated in Section 8.2.1.2. For completeness, the comparisons of the first three mode shapes for point loaded cantilever beam between the two solvers are given in Figure 8.33.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.31 Convergence of the L_∞ , L_1 and L_2 error norms for the velocity field with a immersed fixed-free cantilever. Slope 1 and Slope 2 are the reference lines for 1-order and 2-order accuracy respectively

Figure 8.32 Comparison of CPU times used by the two solvers

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

(a)

(b)

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.33 (a) Grid convergent comparison between the two solvers; (b), (c) and (d): First three fundamental mode shapes comparison between the two solvers. Note that in all pictures then beam is represented by its central line, and the displacements are normalized to unity.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

8.3 SUMMARY

As presented in the previous two sections, a new method combining preconditioning, parallel unstructured MG method and a novel treatment of moving objects in fluids for efficient simulation of 3D unsteady compressible flows has been successfully developed and validated. The convergence of numerical solutions is found to be significantly improved with a combination of the preconditioning and unstructured MG methods. And the use of gauge pressure in pressure gradient terms is found to be important to eliminate round-off errors while the flow speed is very low. The parallel speed-up and efficiency of the method for both steady and unsteady flows are found to be reasonably good. The newly developed immersed membrane method is shown to work well with very large displacements of immersed moving objects. Compared with the immersed boundary method, this method allows sharp changes of fluid conditions across immersed thin structures without complicated and time-consuming interpolation and extrapolation. Results from a flow over a circular cylinder computed by the proposed method are found to agree well with existing numerical and experimental ones. Finally flow over an immersed membrane is calculated and results are compared with those with the membrane erected as a wall, and satisfactory agreement is observed. These studies demonstrate that the method proposed is an effective tool to solve 3D unsteady low-Mach-number compressible flows with arbitrarily moving objects. A novel 3D matrix-free implicit unstructured multigrid structural dynamic finite-volume solver has been successfully developed and validated. The convergence of numerical solutions is found to be significantly improved with the help of the implicit unstructured multigrid method. The efficiency and accuracy of the solver is fully validated using a point loaded

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

fixed-free cantilever, for which both 2D and 3D static and dynamic cases are thoroughly tested. Through the case involving a fixed-free cantilever immersed in fluid flow it is found that the current FV structural dynamic solver works well with our unstructured grid FV compressible fluid solver TETRAKE as well as the Immersed Membrane method [142]. These studies demonstrate the potential capability of the proposed method for large-scale complex fluid-structure interaction simulation.

To determine the overall accuracy of the proposed methods, we carried out a grid convergence study for a classic test problem, which is a three-dimensional analogue of the problem used by Gilmanov et al. [188]. In this case, we simulate flow induced by a sphere rotating at constant angular in a nearly incompressible, viscous. Benchmark solutions for this problem have been reported by Dennis et al. [189] who solved numerically the steady, axisymmetric Navier-Stokes equations in polar coordinates using a vorticity-stream function formulation. In our studies, the Reynolds number is set to $Re=100$, and we solve the full three-dimensional and unsteady flow problem with the sphere starting to rotate impulsively from quiescence relative to the stationary Cartesian coordinate system. Four uniformly spaced and successively finer mesh sizes are used for error analysis, and the finest-mesh solution is considered to be the ‘exact’ solution. The results of the grid convergence study are summarized in Figure 8.1, which shows the variation of the L_∞ , L1 and L2 norms of errors with grid spacing in logarithmic coordinates. The lines with slope one and two are also given as reference. It is evident from Figure 8.1 that the method is second order accurate. To further demonstrate the accuracy of our method, we also use the Richardson estimation procedure to study the

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

accuracy of the solver as in [188]. The results are summarized in Table 8.1, which strongly supports our assertion about the second-order accuracy of our method. Further supporting results can be found in section 8.1.2 (Figure 8.5) and section 8.1.3 (Figure 8.7). As to the structural dynamic solver, grid-invariance studies were also performed in section 8.2, which also support our claim about its 2nd-order accuracy.

8.4 STEADY AND UNSTEADY AIR FLOWS BETWEEN TWO RIGID DISKS COROTATING IN A FIXED CYLINDRICAL ENCLOSURE

The flow field configuration of interest is shown in Figure 8.34. It consists of a pair of disks clamped co-axially on a central hub that rotates in a stationary cylindrical enclosure. Small aspect ratios are considered here since they are especially relevant to industrial and technical applications such as hard disk drives. The configuration of coaxial disks co-rotating in a cylindrical enclosure provides a useful model for investigating flows in the hard disk drives used as data storage devices in computers. A better understanding of the complex unsteady flows that arise in disk storage devices is essential for their improved design and repeatable operation. A disk storage system consists of a stack of equidistant, centrally clamped disks co-rotating in a non-axisymmetric enclosure. Electronic data are distributed along micron-sized circular tracks on the disk surfaces. Data transfer to and from the disks is accomplished by means of magnetic heads suspended at sub-micron distances from the rotating disk surfaces by rigid supports.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.34 Schematic of the corotating disks flow configuration indicating the toroidal vortices that develop in the cross-stream plane near the fixed enclosure wall. The basic flow regions (I-V) induced in the inter-disk space are sketched in accordance with previous studies [194-198]: (I) enclosure wall boundary layer; (II) disk Ekman layers; (III) the region in solid-body rotation; (IV) core potential flow; (V) detached shear layer. In the bulk of the calculations, $R_1=56.4$ mm, $R_2=105.7$ mm, $H=9.53$ mm, and $b=0$. Selective calculations have been performed with $R_1=56.4$ mm, $R_2=105.0$ mm, $H=9.53$ mm, $\alpha=2.7$ mm, and $b=1.91$ mm. The latter values correspond exactly to an experimental configuration used in other studies [194, 197, 199, 200 and 201].

For the configuration with $\Gamma = H / (R_2 - R_1 + a) \approx 0.186$, past studies show that the

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

transition from axisymmetric two-dimensional steady flow to non-axisymmetric three-dimensional unsteady flow occurs approximately at $Re=23,150$, where Re is the Reynolds number based on the disk radius, the tip speed of the disks, and the kinematic viscosity of the fluid. Below the critical Reynolds number, the steady flow solutions are characterized by a symmetrical pair of counter-rotating toroidal vortices in the cross-stream (r - z) plane. This secondary motion is driven by the radial imbalance between the outward-directed centrifugal force and the inward-directed pressure gradient force. Above this critical value the motion is unsteady and periodic, while the features of the cross-stream flow pattern are broadly preserved. The symmetry of the motion about the mid-plane is broken by alternating periodic crossings of the toroidal vortices. This instability is maintained through an interaction that arises between outward-directed fluid in the disk Ekman layers and inward directed fluid in the return core flow. Three-dimensional calculations presented in [199] at $Re=22,200$ and $44,400$ show that the toroidal vortices acquire a time-varying sinuous shape in the circumferential direction. These calculations reveal circumferentially periodic reversals of the axial velocity component in the cross-stream plane, including the detached shear layer separating the region of motion in solid-body rotation near the hub from the potential core, in agreement with the flow visualization observations of Humphrey and Gor [200]. The wavelength of this oscillation is shown to be twice that of the circumferential velocity component which is responsible for the nodal distribution of axial vorticity. When plotted on the inter-disk mid-plane, the axial component of vorticity manifests itself as an even integer number, $2n$ ($n = 1, 2 \dots$), of circumferentially periodic foci. In this study, the steady and unsteady unobstructed laminar flow of air between the two co-rotating disks is simulated using

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

proposed numerical procedure. The flow configurations with aspect ratio $\Gamma \approx 0.186$ and Reynolds number ranging from 4,400 to 44,400 are considered, corresponding to rotation rates ranging from 60 to 600 rpm. In the bulk of the calculations, the thickness of the disks is ignored ($b=0$). Nevertheless, a case corresponding to the experimental values of $b=1.91$ is also examined to evaluate the influence of such simplification. For the configurations of interest here, the grid nodes are distributed non-uniformly throughout the calculation domain in order to resolve the strongly sheared regions of the flow. These regions are: the Ekman layers along the respective disk surfaces, the shear layer along the fixed cylindrical enclosure wall, and the detached shear layer lying between the region of flow in solid body rotation and the fully three-dimensional potential core. Estimates of the location and size of these regions provided in References [194, 199 and 200] were used to construct the present grids. This approach permits an effective distribution of nodes while avoiding unnecessary refinement where velocity gradients are weak, e.g. in the solid body rotation region. Four meshes with different grid sizes are tested, with 802,546, 1,316,248, 1,798,149 and 2,348,149 tetrahedron cells respectively.

Due to limited computer time and storage, a grid convergence study was performed to determine the most appropriate mesh for the final test. Calculations are first performed in the configuration of Figure 8.34 for conditions of the experiment of Schuler et al. [197]. Radial profiles of the circumferential velocity component (tangent velocity component in the u, v plane) calculated on the inter-disk mid-plane for $Re=4,440$, 7,400, and 14,800 ($\Omega = 60, 100$, and 200 rpm), are compared in Figure 8.35 with the experimental data of Schuler. The calculations are started from a zero initial velocity condition, and allowed to

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

evolve in time until the steady-state solution is reached. As can be seen from the figure, the agreement between the measurements and the calculations is very good, with only very slight differences arising in the near wall region. The results of the last two meshes virtually coincide with each other. Notwithstanding, the remaining calculations reported here correspond to the third grid with 1,798,149 elements.

8.4.1 Three-Dimensional Steady Flow

Figure 8.36 shows contours of the velocity components, Mach number and temperature in the cross-stream (r - z) plane for the speeds of rotation of $Re=7400$ ($\Omega=100$ rpm). In the figure, the hub is the right boundary, the enclosure wall is the left boundary, and the disks are the top and bottom boundaries. In this simulation, we ignored the thickness of the disks. The structure of the flow is remarkably similar to that predicted analytically by Schuler et al [197].

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.35 Comparison between measurements and calculations of the circumferential velocity component along the inter-disk mid-plane for (a) $Re=4,440$ (60 rpm), (b) 7,400 (100 rpm), and (c) 14,400 (200 rpm). The results for the last two grids virtually coincide.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

An Ekman layer develops along the surface of each disk, starting at the radial location where the circumferential velocity profile first deviates from solid-body rotation (see Figure. 36). Analysis shows that the thickness of an Ekman layer, marked by an arrow in Figure. 36 (b), is of order $\delta_E = (\nu/\Omega)^{1/2}$ and uniform. The fluid forced radially outward in the Ekman layers is redirected in the axial direction along the enclosure wall, and then radially inward into the core of the flow about the inter-disk mid-plane, thus creating a pair of symmetrical, toroidally shaped, counter-rotating vortices. A thin shear layer with thickness also of order δ_E , develops very near the fixed enclosure wall. The numerically calculated Ekman layer thicknesses are nearly uniform along with r , and in agreement with the above estimate for δ_E , obtained by Schuler et al. The basic structure of the flow is relatively independent of the Reynolds number, also in agreement with the predictions of the theoretical analysis presented in that reference. Figure 8.37 shows the streamline patterns on the same plane. From which, a pair of symmetrical, toroidally shaped, counter-rotating vortices can be easily identified.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.36 Iso-contours of the velocity components, Mach number and Temperature on the r - z plane. Obtained from three-dimensional calculations for $Re=7400$. In figure (b), an arrow marks the thickness of an Ekman layer, estimated according to $\delta_E = (\nu/\Omega)^{1/2}$.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.37 Streamline patterns on the r - z plane. A pair of symmetrical, toroidally shaped, counter-rotating vortices can be easily identified.

Results of current simulation in the form of contours of various flow variables on the inter-disk mid-plane are also shown in Figure 8.38.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.38 Contours of u and circumferential velocity components, pressure and temperature on the inter-disk mid-plane.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

8.4.2 Three-Dimensional Unsteady Flow**8.4.2.1 Case One: $Re=22,200$**

In contrast to steady flow, the axisymmetric flow at $Re = 22,200$ oscillates periodically about the inter-disk mid-plane. Figure 8.39 shows the instantaneous flow field on the r - z plane at an instant in time. The oscillation develops without the imposition of a perturbation. Figure 8.40 shows the time variation of the dimensional circumferential, radial and axial velocity components at a monitoring point located on the inter-disk mid-plane. Due to the periodic change in sign of the axial velocity component with respect to the mid-plane, its frequency is half that of the other two velocity components and pressure which do not change sign. Power spectra of these time series (calculated using data for nondimensional time $t > 100$ only) reveal a pair of dominant frequencies in this flow, namely $2\pi f/\Omega = 10.42$ for radial and circumferential components, and $2\pi f/\Omega = 5.21$ for axial component.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.39 (a) Circumferential velocity components; (b) Radial velocity components; (c) Axial velocity components and (d) Streamline patterns in the r - z plane. Obtained from three-dimensional calculations for $Re=22,200$ ($\Omega=300rpm$) and $t=103$ second.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.40 Time history of the dimensional circumferential, radial and axial velocity components, as well as the nondimensional parameter, pressure coefficient (C_p), at a monitoring point located on the inter-disk mid-plane ($r/R_2=1.0$). Obtained from three-dimensional calculations for $Re=22,200$ ($\Omega=300\text{rpm}$).

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.41 compares unsteady three dimensional calculations of the mean and *rms* (root-mean-square) circumferential velocity components on the inter-disk mid-plane at $\theta=0$, with corresponding measurements for $Re=22,200$. The mean velocity profiles are in reasonable agreement. However, although qualitatively similar, the calculated *rms* profile is significantly smaller in magnitude than the experimental profile. The under-prediction of the velocity *rms* by the current numerical procedure, and the higher predicted value of the Reynolds number for the transition to unsteady flow, can be attributed to several factors as discussed in [199]. First, the thickness of the disks is ignored in the calculations. The tip of a disk represents an additional source of instability for the flow ejected radially outward in the Ekman layer. Second, small but unavoidable imperfections in the disks, in the curvature of the enclosure wall, and in the relative orientation of various parts of the test section can induce perturbations in the experiment that result in higher *rms* and lower critical speed of rotation for transition to unsteady flow than calculated numerically. Finally, numerical diffusion error tends to stabilize the calculated flow, making necessary the use of extraordinarily refined grids or higher-order low diffusion schemes to counteract this effect.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.41 Comparison between measurements (circles) and calculations (line with triangles) of the nondimensional circumferential velocity component (a) mean and (b) rms along the inter-disk mid-plane for $Re=22,200$ ($\Omega=300$ rpm).

8.4.2.2 Case Two: $Re=44,400$

Three-dimensional calculations are performed for a configuration very similar to that investigated numerically by Humphrey *et al.* [199], in which the effects of the disk

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

rim-enclosure wall gap are considered but the thickness of the disks is neglected. The configuration is defined by setting $R1=56.4$ mm, $R2=105$ mm, $a=2.7$ mm, $H=9.53$ mm and $b=0$ in Figure 8.34, and fixing a symmetry boundary condition in the gap. The configuration has been calculated in this study for the same aspect ratio but different rotational speed (600 rpm, in this case corresponding to $Re=44,400$). Humphrey *et al.* predicted six foci in a converged calculation with $Re=44,400$. The calculation of this flow was performed using the previously calculated flow field shown in Figure 8.39 as the initial condition. With no perturbation applied to this initial flow field, the result after 0.1 second (one-third of a disk revolution) show little difference with the solution provided initially. Thus, a non-axisymmetric perturbation (w_{disk}) was applied in the form of a simulated wobble of the disks. The perturbation was imposed for 0.1 second and was circumferentially sinusoidal, with amplitude 5% of the local disk speed: $w_{disk} = 0.05\Omega r \sin\theta$. This axial velocity was applied in phase to both disks in such a way that global continuity was preserved. The result was an irregular (non-periodic) oscillatory flow field. The calculation was repeated with a quiescent fluid as an initial condition, and a similar irregular oscillatory flow was obtained. This ensured that the initial condition was not a factor in determining the fully developed flow.

Results of the present converged calculation in the form of instantaneous contours of the z-vorticity component, ω_z , on the inter-disk mid-plane are shown in Figure 8.42. (Here by converged we mean that the present flow acquired a steady periodic state with primary oscillation frequencies corresponding to $2\pi f/\Omega = 10.42$ for the radial and circumferential velocity components and $2\pi f/\Omega = 5.21$ for the axial). The contour

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

levels plotted are specifically selected to display the circumferentially periodic foci of axial vorticity described by Humphrey *et al.* [199].

Figure 8.42 Iso-contours of the axial vorticity component, ω_z , on the inter-disk mid-plane from three-dimensional calculations for $Re=44,400$ (600 rpm). 6 positive foci (colored by red) can be clearly identified.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.43 shows the three dimensional iso-surface of axial velocity component, which is color-contoured by Mach number. In this figure, the non-axisymmetric 3D flow structure is fully revealed.

Figure 8.43 Iso-surface of the axial velocity component, u_z , from three-dimensional calculations for $Re=44,400$.

Plots of the r-z cut-plane mesh and cross-stream flow field for an instant of time are

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

shown in Figure 8.44 for the flow at $Re=44,400$.

Figure 8.44 Instantaneous plot of the cross-stream flow field from three-dimensional calculation for $Re=44,400$ (600 rpm). (b) Circumferential velocity components; (c) Radial velocity components; (d) Axial velocity components; (e) Mach number and (f) Pressure

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

8.4.2.3 Effects of Disk-Enclosure Gap and Disk Thickness

Numerical calculations were also performed for the configuration shown in Figure 8.34, with $R_1=56.4$ mm, $R_2=105$ mm, $H=9.53$ mm, $a=2.7$ mm and $b=1.91$ mm, in order to assess the effects of the disk thickness and rim-enclosure wall gap on the flow. These dimensions yield the same aspect ratio, $\Gamma \approx 0.186$, as the previous configuration with $a=0$ and $R_2=107.7$ mm. Figure 8.45 shows the instantaneous flow field on r - z plane at an instant in time. No significant discrepancies were found between these results and the former results shown above without considering the disk thickness. Except that the critical value of the Reynolds number for the flow transitioning to the unsteady region changes to about *15,000*.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.45 (a) Circumferential velocity components; (b) Radial velocity components; (c) Axial velocity components and (d) Streamline patterns on the r - z plane. Obtained from three-dimensional calculations for $Re=15,000$.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

8.4.2.4 Remarks

Three-dimensional calculations performed for the unsteady laminar flow between a pair of disks corotating in a fixed cylindrical enclosure provide new insight for the interpretation of experimental observations and the improved understanding of this class of flows. The calculations results show good overall qualitative agreement with the experimental results available and reveal the instability phenomenon in the region near the curved enclosure wall, where the disk Ekman layers turn to collide against each other.

8.5 LARGE-EDDY SIMULATION OF AEROELASTIC FLUTTER FOR THE ONERA M6 WING IN THE TRANSONIC FLOW

In this case we study the flow over an ONERA M6 wing. It was tested in a wind tunnel at transonic Mach numbers (0.7, 0.84, 0.88 and 0.92) and various angles-of-attack up to 6 degrees by Schmitt and Charpin [202]. The Reynolds numbers were about 12 million based on the mean aerodynamic chord. The wind tunnel tests are documented in the AGARD Report AR-138 published in 1979 [202]. The ONERA M6 wing is a classic CFD validation case for external flows because of its simple geometry combined with complexities of transonic flow (i.e. local supersonic flow, shocks, and turbulent boundary layers separation). It has almost become a standard for CFD codes because of its inclusion as a validation case in numerous CFD papers over the years. To evaluate the accuracy of the steady aerodynamic solution, calculations for this wing are firstly carried out and compared to available experimental data. In this step, the wing body will be considered as a rigid solid and the structural domain will be ignored during the

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

calculations. In reference [202], the ONERA M6 wing was tested throughoutly, and the pressures at various span-wise locations were recorded, which will be served as a main source of validation for the current study. Next, the former aerodynamic result is employed as the initial state in the full fluid-structure interaction simulation and the wing flutter will be predicted using the proposed method. The transonic flutter of the M6 wing has been extensively studied by many researchers [203-205] and numerous results are available for validation purpose.

8.5.1 Geometry and Calculation Parameters

The ONERA M6 wing, which is a swept, semi-span wing with no twist, has a leading-edge sweep angle of 30° , an aspect ration of 3.8, and a taper ratio of 0.562. It uses a symmetric airfoil using the ONERA D section, which is a 10% maximum thickness-to-chord ratio conventional section. Figure 8.46 shows the geometric layout and some of the geometric properties of the wing. The values of the parameters for the steady aerodynamic and static aeroelastic calculations are as follows:

Angle of Attack, α	$= 3.06^\circ$
Angle of Side-Slip, β	$= 3.06^\circ$
Freestream Mach Number, M_∞	$= 0.8395$
Density of Air, ρ_∞	$= 1.223 \text{ kg/m}^3$
Freestream Velocity, U_∞	$= 285 \text{ m/s}$
Root Chord, c_r	$= 0.8059 \text{ m}$

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Reynolds Number, R_e = 11.72×10^6

Freestream Temperature, T_∞ = 460.0° Rankine

Freestream Pressure, P_∞ = 45.82899 psia

Figure 8.46 Geometric layout of the ONERA M6 wing

The computational domain is depicted in Figure 8.47 below. Three successively refined

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

grids are used for the grid convergence study purpose:

Grid one	1,769,472 elements, 306,577 nodes
Grid two	3,566,552 elements, 604,583 nodes
Grid three	5,308,416 elements, 909,521 nodes

Figure 8.48 shows the grid about the tip of the wing. The grid is clustered about the wing surface to resolve the turbulent boundary layer.

Figure 8.47 Schematic of computational domain of the ONERA M6 wing

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

8.5.2 LES of Steady Aerodynamic Computation of M6 Wing

The computation is performed using the time-marching capabilities of TETRAKE to march to a convergent solution starting from a uniform initial solution tabulated above. Local time stepping is used at each sub-iteration to enhance the convergence rate. The flow is assumed to be in transitional regime. Figure 8.49, 8.50 and 8.51 show the convergence histories of the lift coefficients C_l and the drag coefficients C_d , as well as the moment coefficient C_m along z-axis, respectively. A mean lift coefficient C_l and a mean drag coefficient C_d are derived from the limited time samples. They are listed in Table 8.9, and show good agreements with other published results. Results for three different grids are tabulated for ease of comparison.

Figure 8.48 Computational grids near the tip of the ONERA M6 wing

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.49 Time history of the drag coefficient of the ONERA M6 wing.

Figure 8.50 Time history of the lift coefficient of the ONERA M6 wing.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.51 Time history of the moment coefficient of the ONERA M6 wing.

Table 8.9 Predicted mean Lift and Drag coefficients

	Mean Lift Coefficient	Mean Drag Coefficient
Finest Grid	0.1428	0.0084704
Intermediate Grid	0.1420	0.0084663
Coarse Grid	0.1347	0.0073866

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Due to the fact that the computation on the intermediate grid (with 3,566,552 elements and 604,583 nodes) already gives satisfactory result, and in order to minimize the necessary computational time, this grid will be exclusively use in the following computation. All of the shown results hereafter are all corresponding to this mesh unless specified. The time averaged Mach number distribution on the symmetric plane and upper wing surface is shown in Figure 8.52. It shows that on the inboard region of the wing, the main shock is located slightly behind mid-chord. A leading edge shock at around $0.1c$ (c is local chord length) is also observed.

Figure 8.52 Time averaged Mach number distribution on the symmetric plane and upper surface of the ONERA M6 wing.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

For a wing calculation, another piece of important information is the pressure coefficient on the wing surface. The chord-wise time averaged pressure coefficient distributions on the wing surface at seven span-wise locations (defined in Figure 8.46) are presented in Figure 8.53. The collected sample time mean pressure coefficients data at the same locations of the wing from referenced experiment [202] are also identified in the figures for ease of comparison. These figures show that the predicted pressure coefficient agrees well with the experimental result, which in turn exhibits the accuracy of proposed Immersed Membrane method as well as the LES modeling.

(a).

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

(b).

(c).

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

(d).

(e).

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

(f).

(g).

Figure 8.53 Time averaged pressure coefficients distributions on seven selected locations on ONERA M6 wing.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

The distributions of time averaged pressure coefficient, temperature and density on the symmetric plane and upper wing surface are shown in Figure 8.54.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.54 Distributions of Time averaged pressure coefficient, temperature and density on symmetric plane and wing surface.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.55 shows the contours of the ratio between the test filtering width and grid filtering width on the symmetric plane. Figure 8.56 shows the time averaged streamline ribbons around the wing surface, where the symmetric plane and the ribbons are contoured by velocity magnitude.

Figure 8.55 Distributions of ratio between test filtering width and grid filtering width on symmetric plane.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.56 Time averaged streamline ribbons around the M6 wing surface. (Contoured by dimensional velocity magnitude)

8.5.3 LES of ONERA M6 Wing Flutter

The M6 wing flutter computation is based on the same flow parameters listed in the previous section and the intermediate grid. The immersed structural domain (represented

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

by the ONERA M6 wing body) is meshed by 42,150 tetrahedron elements and 9,094 nodes. The wing surface is meshed with 7,868 boundary triangles. The structure material parameters are as follows:

$$\text{Young's Modulus, } E = 7.102 \times 10^{10} \text{ Pa}$$

$$\text{Material Density, } \rho = 2770.0 \text{ kg/m}^3$$

$$\text{Poisson's Ratio, } \nu = 0.32$$

The simulation begins by employing the converged solution from the former computation as the initial solution for the flow domain and the initial wing velocity is zero (an initial tip displacement has been given). A fixed time step ($\Delta t = 0.05s$) is employed in the current simulation. Figure 8.57 shows the flutter responses of the ONERA M6 wing at $M_\infty = 0.8395$ and 3.06° angle of attack.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.57 Flutter responses for ONERA M6 wing using intermediate grid at $M_{\infty} = 0.8395$ and 3.06° angle of attack (Displacement in y direction)

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

While Figure 8.58 shows the time variation of lift and drag coefficients for the ONERA M6 wing during the flutter process.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.58 Time history of lift and drag coefficients for M6 wing fluttering using intermediate grid at $M_\infty = 0.8395$ and 3.06° angle of attack

As can be seen from the figures, the constant energy difference ($E_{tot} - W_E$) is equal to the initial energy. This verifies that the energy exchange between the structure and the fluid satisfies the global conservation law for the total energy. We also find that the maximum tip displacement at leading edge is about 87.6% of that at trailing edge, which means that the flutter of M6 wing is a composition motion of the first bending and first torsion mode while dominated by the former.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

Figure 8.59 shows the Mach number distributions on the upper surface of the M6 wing at three different moments (t_1 , t_2 and t_3) during the flutter. At $t = t_3$, the tip y -displacement of the wing reaches the maximum value.

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

(a). $T = t_1$

(b). $T = t_2$

CHAPTER 8 – NUMERICAL RESULTS AND DISCUSSION

(c). $T = t_3$

Figure 8.59 Instantaneous Mach number distributions at three different moments in one cycle of the wing flutter

CHAPTER NINE

CONCLUSIONS AND FUTURE WORK

This chapter concludes the thesis with a summary of the achievements and contributions of the present research work, which aims to develop efficient and accurate numerical methods for moving boundary and fluid-structure interaction problems in compressible fluid dynamic applications. This chapter will summarize the works based on the two main parts, namely, (i) the development and validation of the three-dimensional parallel Navier-Stokes compressible turbulent flow solver with the immersed membrane method for fluid-structure interaction phenomena, and (ii) the application of the developed solver to compute the steady and unsteady flows induced by two co-rotating disks in a fixed enclosure, and simulate the fluttering of the ONERA M6 wing immersed in high speed transition air flow. Finally this chapter proposes possible directions for future researches.

CHAPTER 9 – CONCLUSIONS AND FUTURE WORK

9.1 CONCLUSIONS

In this work, a parallel unstructured multigrid preconditioned implicit solver for the simulation of fluid-structure interaction (FSI) between three-dimensional turbulent compressible flows and arbitrarily immersed elastic solid objects has been successfully developed. The novel feature of the solver is that it employs a unique combination of a parallel multi-grid scheme with low-Mach-number preconditioning and the immersed membrane method so that the FSI for very complex geometry can be efficiently simulated. The low Mach number preconditioning method has been used as a remedy to the ill-conditioned time-marching density-based algorithms to tackle the mixed flows with very low Mach-number regions embedded in regions of high Mach-number flows. A novel matrix-free finite-volume solver for structural dynamic has been successfully developed. It is designed to simply reuse the existing iteration algorithms and data structures available in the existing fluid solver for ease of integration. The fluid and structural solvers are loosely coupled through the introduction of IMM. IMM serves as an intermediary between the two models, which enables the fluid-structure interaction to be efficiently simulated. The numerical methods proposed are validated by a number of benchmark cases. The framework is then applied to analyze the air flow between two co-rotating disks in a fixed enclosure. And then, it is applied to simulate the aero-elastic fluttering of an ONERA M6 wing in transonic flow. The results show that the current numerical methods can efficiently simulate the moving boundary problems, as well as fluid-structure interaction problems with satisfactory accuracy. The main achievements are summarized in the following sections.

CHAPTER 9 – CONCLUSIONS AND FUTURE WORK

9.1.1 Development and Validation of Numerical Schemes and Algorithms

The 3D parallel-MG solver used in this work is developed from the finite volume scheme based on unstructured tetrahedral meshes for compressible flows. The inviscid convection flux is computed by employing 3rd order Roe's scheme in an edge-based manner, while the viscous flux is computed by employing a 2nd order Galerkin type formulation in a cell-based manner. The unsteady flow is calculated with a matrix-free implicit dual time stepping scheme. A five-stage Runge-Kutta time integration algorithm is used between each physical time step to march the numerical solution in pseudo time until convergence is reached.

The low Mach number preconditioning methods have been used to circumvent the convergence difficulties due to an inappropriate structure of the algorithms rather than the physics of the governing equations. At low Mach numbers a wide disparity in the magnitudes of the eigenvalues creates a stiff system, resulting in the need for an incompressible formulation. However, in many practical and important classes of engineering problems, mixed flows with very low Mach-number regions embedded in regions of high Mach-number ones exist. Therefore, incompressible formulation cannot be used and traditional time-marching density-based schemes suffer from convergence deterioration. Hence, preconditioning extends the applicability of the high speed flow algorithms into the very low Mach number flow regions by rescaling the eigenvalues of the governing equations. Unfortunately promising characteristic methods based on the continuous system of equations do not always perform very well when implemented in a practical flow solver, especially for realistic test cases involv-

CHAPTER 9 – CONCLUSIONS AND FUTURE WORK

stagnation points. It must be noted that, in general, the use of most low-speed preconditioners are plagued by robustness issues, mainly arising from the problem of stagnation points. We have found that in order to obtain high rates of convergence in a robust fashion one needs to pay attention to a number of specific issues, including the selection of the slope limiter for the MUSCL Interpolation, the values of coefficients and parameters, and most importantly, the form of the entropy fix (for shock capture problems) and the formulation of the Roe average matrix.

The computation of complex turbulent flows of practical interest to engineers continues to be a challenge. The Large-Eddy simulation has been chosen to calculate the possible arising turbulent phenomena in this study. Despite its inherent superiority compared to other techniques for turbulence modeling, several issues have impeded the application of LES to practical problems. Firstly, non-periodic inflow and outflow boundary conditions that can sustain a realistic turbulent simulation have been elusive. Simulations have had to rely on periodic boundary conditions, where turbulence exiting the domain is fed back into the inlet the domain. Secondly, there are many deficiencies in the subgrid-scale (SGS) modeling. The present available models tend to break down for complex geometries and do not work well with low-order numerical schemes. Furthermore, such models have not been developed or adequately evaluated for many SGS contributions to the compressible form of the filtered energy equation. In this work, a compressible dynamic form of Smagorinsky SGS model has been implemented to calculate the SGS stress tensor. This model relies upon the Germano identity, which has been generalized in order to be applied to other subgrid-terms arising in the filtered energy equation. Furthermore, an improved formulation of the dynamic mixed model has been proposed for better representing the backscatter of turbulence energy which

CHAPTER 9 – CONCLUSIONS AND FUTURE WORK

has been proven to be important for compressible flows. The unstructured grid filtering was adapted from a new filtering approach based on the least-squares technique. This approach can filter a function to any given level of commutation error on unstructured grids. When used together with IMM, to avoid complex modifications of the filtering operator at the vicinity of the fluid-structure interface, a linear reconstruction procedure similar to that used for the velocity field is also applied for turbulent viscosity μ_t and Prandtl number Pr_t . As shown in the test case of ONERA M6 wing reported in Chapter 8, the proposed dynamic SGS models give satisfactory answers.

In many engineering applications, there is an emerging need to model multiphysics problems in a coupled manner. As another end product of this study, a novel matrix-free structural dynamic finite-volume solver has been successfully developed and validated. It is designed to simply reuse the existing iteration algorithms and data structures available in our fluid solver for ease of integration. The convergence of numerical solutions is found to be significantly improved with the help of the unstructured multigrid method. The efficiency and accuracy of the solver is fully validated using a point loaded fixed-free cantilever, for which both 2D and 3D static and dynamic cases are tested. It should be noted that the approach proposed in this work belongs to a special class of cell-vertex methods that employ non-overlapping control volumes. It should also be noted that the approach is different from previous non-overlapping FV methods in that we do not utilize shape functions at all and we have developed an effective implicit unstructured multigrid method for fast solution convergence.

With the aim of coupling the fluid and structural solvers in an efficient manner, the immersed membrane method (IMM) is developed in this work. IMM is adopted so that

CHAPTER 9 – CONCLUSIONS AND FUTURE WORK

no mesh update is required and allowance of capturing accurate ‘jump’ conditions across the immersed structure surface is possible. Thus more accurate cross-structure fluid conditions can be captured without smoothing procedures. The influence of the structure on ambient fluids is conveyed by enforcing the velocity continuity condition along the fluid-structure interface. And the enforcement of velocity continuity is implemented by the one-sided linear extrapolations of flow variables for ghost nodes along cell edges only. With the ghost fluid nodes, the conventional mesh update strategies, such as re-meshing or Arbitrary Lagrange-Euler (ALE) techniques, are consequently avoided. Hence, the computational cost for mesh treatment is greatly reduced. This method is unique in two main aspects compared with existing methods using ghost nodes: (i) Extrapolation of ghost values along fluid mesh edges instead of surface-normal directions of the structural wall. The surface normal vector of the structural wall is difficult to find, besides one also has to find out the intersection points between the normal vector and the 3D unstructured fluid mesh. Consequently no complicated interpolations to distribute interface conditions to nearby fluid nodes are required; (ii) Multiple ghost nodes are used for the same fluid node. The use of a particular value depends on the edge being considered. The implementation of multiple ghost nodes increases the accuracy of the method. Many efforts have been made to enhance the numerical stability and accuracy of proposed schemes, while performing the moving boundary simulations. Special treatment has been introduced during the ghost value extrapolation process to avoid extremely big unrealistic ghost values. The 3rd-order one-side MUSCL like flux computation is also adopted to further increase the overall accuracy. Level-based treatment for freshly cleared fluid nodes is developed to avoid unphysical solutions in the moving boundary simulations.

CHAPTER 9 – CONCLUSIONS AND FUTURE WORK

A parallel-MG solver has been successfully developed to accelerate the convergence rate, and at the same time, to reduce the time for simulating turbulent flows with complex geometries and without very large single processor memory requirements in sequential processing. This is demonstrated by various test cases listed in chapter 8. The numerical solutions obtained for all these cases show good agreement with published numerical results, as well as others' experimental measurements. For steady flow computation, the number of physical time steps and CPU time using the MG method are significantly reduced compared with SG one.

9.1.2 Applications to Rotating-Disk Flow and Aeroelastic Wing

Flutter

To demonstrate the capability of the methods for simulating high-speed swirling flow, the developed solver is used to compute steady and unsteady flows between two rigid disks co-rotating in a fixed cylindrical enclosure. The computational configuration consists of a pair of disks clamped co-axially on a central hub that rotates in a stationary cylindrical enclosure. Small aspect ratios are considered here since they are especially relevant to industrial and technical applications such as hard disk drives. A better understanding of the complex unsteady flows that arise in disk storage devices is essential for their improved designs and repeatable operations. In this study, the air flow between the two co-rotating disks is simulated using proposed numerical procedure. The flow configurations with aspect ratio $\Gamma \approx 0.186$ and Reynolds number ranging from 4,400 to 44,400 are considered, corresponding to rotation rates ranging from 60 to 600 rpm. In the bulk of the calculations, the thickness of the disks is ignored ($b=0$). Nevertheless, a case corresponding to the experimental values of $b=1.91\text{mm}$ is also

CHAPTER 9 – CONCLUSIONS AND FUTURE WORK

examined to evaluate the influence of such simplification. For the configurations of interest here, the grid nodes are distributed non-uniformly throughout the calculation domain in order to resolve the strongly sheared regions of the flow. These regions are: the Ekman layers along the respective disk surfaces, the shear layer along the fixed cylindrical enclosure wall, and the detached shear layer lying between the region of flow in solid body rotation and the fully three-dimensional potential core. As exhibited in the low Reynolds number calculation ($Re=4,400$), an Ekman layer develops along the surface of each disk, starting at the radial location where the circumferential velocity profile first deviates from solid-body rotation. Analysis shows that the thickness of an Ekman layer, marked by an arrow in each insert, is of order $\delta_E = (\nu / \Omega)^{1/2}$ and uniform. The fluid forced radially outward in the Ekman layers is redirected in the axial direction along the enclosure wall, and then radially inward into the core of the flow about the interdisk midplane, thus creating a pair of symmetrical, toroidally shaped, counter-rotating vortices. A thin shear layer with thickness also of order δ_E , develops very near the fixed enclosure wall. The numerically calculated Ekman layer thicknesses are nearly uniform with radial relative position, and in agreement with the above estimate for δ_E , obtained by Schuler *et al* [197]. The basic structure of the flow is relatively independent of the Reynolds number, also in agreement with the predictions of the theoretical analysis presented in that reference. The calculations of higher Reynolds number ($Re=22,200$ and $Re=44,400$) reveal an instability in the region near the curve enclosure wall. The cross-stream flow oscillations calculated when $Re > 22,200$ can be explained as follows. Assume that at a particular instant in time the inward-directed radial jet is nearer the top disk. As a result, fluid with a smaller circumferential component of momentum is brought closer to the top Ekman layer, increasing the axial gradient of circumferential velocity there. At the same time, since the jet is farther away from the

CHAPTER 9 – CONCLUSIONS AND FUTURE WORK

lower disk, the gradient of circumferential velocity near the bottom Ekman layer decreases. Because the axial gradient of circumferential velocity determines the intensity of the radial flow in the Ekman layers, the radial flow in the top Ekman layer will increase, relative to that in the lower layer. This intensified radial flow is redirected axially along the enclosure wall to displace the stagnation point and the radial jet downward, toward the bottom disk. Once jet is located below the geometrical symmetry plane, the same mechanism will operate in reverse to raise the jet. These calculations show that the current proposed methods are accurate enough to simulate the high-speed swirling flow.

To further demonstrate the capability of simulating the FSI phenomena, the developed solver is then used to predict the aeroelastic fluttering of ONERA M6 wing. To evaluate the accuracy of the steady aerodynamic solution, calculations are firstly carried out and results obtained are compared to available experimental data. In this step, the wing body will be considered as a rigid solid and the structural domain is ignored during the aerodynamic calculations. Next, the steady aerodynamic result is employed as the initial condition and the full FSI simulation is then carried out and the wing flutter predicted using the proposed method. As demonstrated by the numerical results shown in the previous chapter, a mean lift coefficient $C_l=0.1428$ and a mean drag coefficient $C_d=0.0084704$ are derived from the limited time samples. The Mach number distribution on the symmetric plane and upper wing surface shows that on the inboard region of the wing, the main shock is located slightly behind mid-chord. A leading edge shock at around $0.1c$ (c is local chord length) is also observed. The collected sample time mean pressure coefficients data at the same locations of the wing as in the experiment [202] are also calculated. It is shown that the predicted pressure coefficient

CHAPTER 9 – CONCLUSIONS AND FUTURE WORK

agrees well with the experimental result, which in turn exhibits the accuracy of proposed Immersed membrane method as well as the LES modeling.

In summary, the developed Navier-Stokes solver as well as structural dynamic solver coupled by the proposed immersed membrane method is very efficient and robust in solving moving rigid and elastic body problems with FSI in 3D unsteady flows. In this work, the proposed numerical models are found to give a reasonably accurate representation of the dominant flow patterns between co-rotating disks in a fixed enclosure. And the solver is found to be an effective computational tool to predict aeroelastic wing fluttering. All of these achievements serve as the first step in an effort to develop a high performance computational tool that hopefully can be used in the rational design and test of new rotating machineries.

9.2 FUTURE WORKS

In the present work, a high performance computational tool based on a parallel-unstructured unsteady Multigrid Navier-Stokes compressible solver with the immersed membrane method for fluid-structure interaction problems has been developed and validated. However, in the course of the work, several problems regarding the developments of numerical methods and algorithms are identified, which deserve future attentions as listed below:

- (1) Complex geometrical configurations and existence of very small gaps, together with high-speed swirling turbulent flows pose a big challenge to the author while he simulate disk flutter in hard disk drives. Despite the fact that IMM is

CHAPTER 9 – CONCLUSIONS AND FUTURE WORK

designed to handle the complex configuration with the existence of non-symmetric geometry, further enhancements will still be needed to remove instabilities introduced by the small gaps as well as the thin rotating disks. In this study, the immersed disks are all treated as rigid bodies. The extension to simulate fully fluid-structure interaction is one of his future works.

- (2) Large Eddy Simulation (LES) has been implemented as a versatile simulation tool to treat all flow conditions including laminar, turbulent flows and flows with laminar-to-turbulence transition. The incorporation of LES in compressible solvers is currently an active research area, which is still in its early stage. The numerical stability is still a problem, especially for solvers employing unstructured meshes. Due to the limitation of time and computational hardware, the newly developed SGS models as well as the adopted explicit filtering operator are not validated in a systematic manner. The grid utilized in the LES computation of ONERA M6 wing case is far from enough fine to capture the turbulence detail accurately, such as the instantaneous spanwise and streamwise vorticity components. These works need further efforts.
- (3) The biggest challenge in this study while simulating the FSI problems is how to couple the two modules (fluid solver and structural solver) and synchronize them. Due to the differences in physics and scales, the time step limitation poses a big problem to us. Further improvements may be necessary for the structural solver to alleviate its time step restriction.

CHAPTER 9 – CONCLUSIONS AND FUTURE WORK

- (4) Currently what is proposed can be applied to simulate the FSI cases with relatively small structural displacements. We need to make sure that the structure will not move out of the priorly refined region. Failing to ensure this may lead potential problems, for example, the inaccurate capturing of flow patterns inside boundary layers. Incorporation of the capability to handle large structural displacements is currently in progress. Take the Figure 9.1 for example; the basic idea is to employ three separated grids to deal with the problem associated with the extremely fine grid requirement in the boundary layer regions. Besides the background fluid mesh and the structural mesh, we further introduce an overlapping mesh, which is moving along with the structural mesh. This overlapping mesh could be fully refined to resolve the boundary layer around the structure. In every sub-iteration, the flow condition is firstly transferred from the fluid mesh to the overlapping mesh using higher-order interpolation scheme. Then the IMM scheme is applied between the overlapping mesh and the structure mesh. Next, the flow governing equations are solved on the overlapping mesh. Finally, the new flow condition is transferred back to the background fluid mesh. This method is currently under development.

CHAPTER 9 – CONCLUSIONS AND FUTURE WORK

Figure 9.1 Demonstration of overlapping grid in IMM method dealing with the boundary layer around the structure; this method make it possible to simulate the FSI involving large deformation of structure.

APPENDIX

APPENDIX

Preconditioning Matrix Definition [61]

$$\Gamma_1 = \begin{pmatrix} 1/\beta' \gamma T' & 0 & 0 & 0 & -\rho/T' \\ u/\beta' \gamma T' & \rho & 0 & 0 & -\rho u/T' \\ v/\beta' \gamma T' & 0 & \rho & 0 & -\rho v/T' \\ w/\beta' \gamma T' & 0 & 0 & \rho & -\rho w/T' \\ H/\beta' \gamma T' - 1 & \rho u & \rho v & \rho w & \rho[\gamma/(\gamma-1) - H/T'] \end{pmatrix}$$

Where:

$$\gamma = C_p / C_v$$

$$c^2 = \gamma p / \rho$$

$$T' = p / \rho = c^2 / \gamma$$

$$H = (\rho e_t + p) / \rho$$

$$\beta' = \beta / [1 + (\gamma - 1)\beta]$$

Noted that Γ_1 is a rank one modification of M (which is defined in the main body)

allows one to easily compute the matrix products $\Gamma_1^{-1}M$ and $M^{-1}\Gamma_1$ (which will be needed) via the Sherman-Morrison-Woodbury formula.

The general preconditioned Jacobian matrix is:

APPENDIX

$$\tilde{H}_p = \Gamma_1^{-1} H_p = \Gamma_1^{-1} \frac{\partial F_c}{\partial W_p} =$$

$$\begin{bmatrix} \beta U & \rho \beta \gamma T' n_x & \rho \beta \gamma T' n_y & \rho \beta \gamma T' n_z & 0 \\ n_x / \rho & U & 0 & 0 & 0 \\ n_y / \rho & 0 & U & 0 & 0 \\ n_z / \rho & 0 & 0 & U & 0 \\ U(\gamma-1)(\beta-1)/(\rho \gamma) & (\gamma-1)\beta T' n_x & (\gamma-1)\beta T' n_y & (\gamma-1)\beta T' n_z & U \end{bmatrix}$$

Where $U = n_x u + n_y v + n_z w$, with n_x, n_y, n_z are the unit normalized vectors. The eigenvalues of $\tilde{H}_p$ are

$$\lambda(\tilde{H}_p) = \left(\lambda_0 = U, \lambda_0 = U, \lambda_0 = U, \lambda_{1,2} = \frac{(\beta+1)U \pm S}{2} \right)$$

where: $S = \sqrt{U^2(\beta-1)^2 + 4\beta c^2}$

The right eigenvectors of $\tilde{H}_p$ are:

$$X_{\tilde{H}_p, R} =$$

$$\begin{bmatrix} 0 & 0 & 0 & (\beta U - \lambda_2)/S & (\lambda_1 - \beta U)/S \\ 0 & -n_z & n_y & n_x/(\rho S) & -n_x/(\rho S) \\ n_z & 0 & -n_x & n_y/(\rho S) & -n_y/(\rho S) \\ -n_y & n_x & 0 & n_z/(\rho S) & -n_z/(\rho S) \\ n_x & n_y & n_z & (\gamma-1)(\beta U - \lambda_2)/(\gamma \rho S) & (\gamma-1)(\lambda_1 - \beta U)/(\gamma \rho S) \end{bmatrix}$$

And the corresponding left eigenvectors matrix is given by:

APPENDIX

$$X_{\tilde{H}_p, L} = \begin{bmatrix} -(\gamma-1)n_x/(\gamma\rho) & 0 & n_z & -n_y & n_x \\ -(\gamma-1)n_y/(\gamma\rho) & -n_z & 0 & n_x & n_y \\ -(\gamma-1)n_z/(\gamma\rho) & n_y & -n_x & 0 & n_z \\ 1 & \rho(\lambda_1 - \beta U)n_x & \rho(\lambda_1 - \beta U)n_y & \rho(\lambda_1 - \beta U)n_z & 0 \\ 1 & \rho(\lambda_2 - \beta U)n_x & \rho(\lambda_2 - \beta U)n_y & \rho(\lambda_2 - \beta U)n_z & 0 \end{bmatrix}$$

REFERENCE

REFERENCES

- [1] Donea J., Giuliani S., and Halleux J. P., An Arbitrary Lagrangian-Eulerian Finite Element Method for Transient Dynamic Fluid-Structure Interactions, *Computer Methods in Applied Mechanics and Engineering*, **33**, pp.689-723, 1982.
- [2] Margolin L. G., Introduction to An Arbitrary Lagrangian-Eulerian Computing Method for All Flow Speeds, *Journal of Computational Physics*, **135**, pp.198-202, 1997.
- [3] Hirt C. W., Amsden A. A., and Cook J. L., An Arbitrary Lagrangian-Eulerian Computing Method for All Flow Speeds, *Journal of Computational Physics*, **135**, pp.203-216, 1997.
- [4] Brackbill J U., and Pracht W. E., An Implicit, Almost-Lagrangian Algorithm for Magnetohydrodynamics, *Journal of Computational Physics*, **13**, pp.455, 1973.
- [5] Benson D. J., Computational Methods in Lagrangian and Eulerian Hydrocodes, *Computer Methods in Applied Mechanics and Engineering*, **99**, pp.235-394, 1992.
- [6] Farhat C., Lesoinne M., and LeTallec P., Load and Motion Transfer Algorithms for Fluid/Structure Interaction Problems with Non-Matching Discrete Interfaces: Momentum and Energy Conservation, Optimal Discretization and Application to Aeroelasticity, *Computer Methods in Applied Mechanics and Engineering*, **157**, pp.95-114, 1998.
- [7] Horsten J. B. A. M., On the Analysis of Moving Heart Vales: A Numerical Fluid-Structure Interaction Model, Ph.D. Thesis, Eindhoven University of Technology, 1990.

REFERENCE

- [8] L. Li, S.J. Sherwin, and P.W. Bearman, A moving frame of reference algorithm for fluid/structure interaction of rotating and translating bodies, *Int. J. Numer. Meth. Fluids*, **38**, pp.187-206, 2002.
- [9] D.J. Newman and G.E. Karniadakis, A direct numerical simulation study of flow past a freely vibrating cable, *J. Fluid Mech.*, **344**, pp.95-136, 1997.
- [10] L. Shen, X. Zhang, D.K.P. Yue, and M.S. Triantafyllou, Turbulent flow over a flexible wall undergoing a streamwise traveling wave motion, *Journal of Fluid and Mechanics.*, **484**, pp.197-221, 2003.
- [11] H. H. Hu, N. A. Patankar, and M. Y. Zhu, Direct numerical simulations of fluid-solid systems using the arbitrary Lagrangian-Eulerian Technique, *Journal of Computational Physics*, **169**, pp.427-462, 2001.
- [12] K.W. Schulz and Y. Kallinderis, Unsteady Flow Structure Interaction for Incompressible Flows Using Deformable Hybrid Grids, *Journal of Computational Physics*, **143**, pp.569-597, 1998.
- [13] Peskin S. C., and McQueen D. M., Modelling Prosthetic Heart Valves for Numerical Analysis of Blood Flow in the Heart, *Journal of Computational Physics*, **37**, pp.113-132, 1980.
- [14] Peskin S. C., and McQueen D. M., A Three-dimensional Computational Method for the Blood Flow in the Heart, *Journal of Computational Physics*, **81**, pp.372-405, 1989.
- [15] Fogelson A. L., A Mathematical Model and Numerical Method for Studying Platelet Adhesion and Aggregation During Blood Clotting, *Journal of Computational Physics*, **56**, pp.111-134, 1984.
- [16] Beyer R. P., A Computational Model of the Cochlea Using the Immersed Boundary Method, *Journal of Computational Physics*, **98**, pp.145-162, 1992.

REFERENCE

-
- [17] Arthurs K. M., Moore L. C., Peskin S. C., Pitman E. B., and Layton H. E Modeling Arteriolar Flow and Mass Transport Using the Immersed Boundary Method, *Journal of Computational Physics*, **147**, pp.402-440, 1998.
 - [18] Stockie J. M. and Green S. I., Simulating the Motion of Flexible Pulp Fibres Using the Immersed Boundary Method, *Journal of Computational Physics*, **147** pp.147-165, 1998.
 - [19] Roma A. M., Peskin S. C., and Berger M. J., An Adaptive Version of the Immersed Boundary Method, *Journal of Computational Physics*, **153**, pp.509-534, 1999.
 - [20] Lai M.-C., and Peskin S. C., An Immersed Boundary Method with Formal Second-Order Accuracy and Reduced Numerical Viscosity, *Journal of Computational Physics*, **160**, pp.705-719, 2000.
 - [21] Majumdar S., Iaccarino G., Durbin P.A., RANS Solver with Adaptive Structured Boundary Non-Conforming Grids, *Annu. Res. Briefs 2001, Cent. Turbul. Res.* pp.353-364, 2001.
 - [22] Fedkiw R., et al., A Non-Oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (the Ghost Fluid Method), *Journal of Computational Physics*, **152**, pp.457-492, 1999.
 - [23] E.A. Henderson, C. Morgan, B. Smith, H.C. Sørensen, R.J. Barthelmie, B. Boesmans, Offshore Wind Energy in Europe-A Review of the State-of-the-Art. *Wind Energy*, **6**, pp.35-52, 2003.
 - [24] T. Moeller, Blade Cracks Signal New Stress Problem, *Wind Power Monthly*, May 1997.

REFERENCE

- [25] Daniel Huang, Computation of Compressible Flows Using Euler Solvers on Unstructured and Adaptive Meshes, M. Eng. Thesis, Nanyang Technological University, School of Mechanical and Production Engineering, 1997.
- [26] Fedorenko R. P., The Speed of Convergence of One Iterative Process, U.S.S.R., *Computational Mathematics and Mathematical Physics*, **4**(3), pp.227-235, 1964.
- [27] Brandt A., Multi-Level Adaptive Solutions to Boundary-Value Problems, *Mathematics of Computation*, **31**, pp.333-390, 1977.
- [28] Jameson A., Solution of the Euler Equations for Two Dimensional Transonic Flow by a Multigrid Method, *Applied Mathematics and Computation*, **13**, pp.327-355, 1983.
- [29] Farmer J., Martinelli L., and Jameson A., A Fast Multigrid Method for Solving Incompressible Hydrodynamic Problems with Free Surfaces, AIAA-93-0767, 1993.
- [30] Sheng C., Taylor L. K., and Whitfield D. L., Multigrid Algorithm for Three-Dimensional Incompressible High-Reynolds Number Turbulent Flows, *AIAA Journal*, **33**(11), 1995.
- [31] Drikakis D., Iliev O. P., and Vassileva D. P., A Nonlinear Multigrid Method for the Three-Dimensional Incompressible Navier-Stokes Equations, *Journal of Computational Physics*, **146**, pp.301-321, 1998.
- [32] Drikakis D., Iliev O. P., and Vassileva D.P., Acceleration of Multigrid Flow Computations through Dynamic Adaptation of the Smoothing Procedure, *Journal of Computational Physics*, **165**, pp.566-591, 2000.
- [33] Iwamura C., Costa F. S., Sbarski I., Easton A., and Li N., An Efficient Algebraic Multigrid Preconditioned Conjugate Gradient Solver, *Computer Methods in Applied Mechanics and Engineering*, **192**, pp.2299-2318, 2003.

REFERENCE

- [34] Tai C. H., and Zhao Y., A Finite Volume Unstructured Multigrid Method for Efficient Computation of Unsteady Incompressible Viscous Flows, *International Journal for Numerical Methods in Fluids*, **46**(1), pp.59-84, 2004.
- [35] Li Yuan, Comparison of Implicit Multigrid Schemes for Three-Dimensional Incompressible Flows, *Journal of Computational Physics*, **177**, pp.134-155, 2002.
- [36] Brandt A., Diskin B., and Thomas J. L., Recent Advances in Achieving Textbook Multigrid Efficiency for Computational Fluid Dynamics Simulations, NASA/CR-2002-211656, ICASE Report No. 2002-16, 2002.
- [37] Dennis Jespersen, Thomas Pulliam and Pieter Buning, Recent Enhancements to OVERFLOW (Navier-Stokes code), AIAA Paper, **97-0644**
- [38] Mavriplis D. J., Large-scale Parallel Viscous Flow Computations Using An Unstructured Multigrid Algorithm, NASA/CR-1999-209724, ICASE Report No. 99-44, Nov 1999.
- [39] Mavriplis D. J., Parallel Unstructured Mesh Analysis Of High-Lift Configurations, AIAA Paper 2000-0923, 38th AIAA Aerospace Sciences Meeting, January 2000.
- [40] Roe P. L., Approximate Riemann Solvers, Parameter Vectors and Difference Schemes, *Journal of Computational Physics*, **43**, pp. 357-372, 1981.
- [41] Mavriplis D. J., Multigrid Strategies for Viscous Flow Solvers on Anisotropic Unstructured Meshes, Proceedings of the 13th AIAA CFD Conference, Snowmass, CO, pp. 659-675. AIAA Paper 97-1952-CP, June 1997.
- [42] Llorente I. M., Prieto-Matias M., and Diskin B., An Efficient Parallel Multigrid Solver for 3-D Convection-Dominated Problems, NASA/CR-2000-210319, ICASE Report No. 2000-29, August 2000.

REFERENCE

- [43] Smith B., Bjorstad P., and Gropp W., Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations, Cambridge University Press, 1996.
- [44] Llorente I. M., and Tirado F., Relationships between Efficiency and Execution Time of Full Multigrid Methods on Parallel Computers, *IEEE Transactions on Parallel and Distributed Systems*, **8**, pp. 562-573, 1997.
- [45] OpenMP home page, OpenMP: Simple, Portable, Scalable SMP Programming. <http://www.openmp.org>, 1999.
- [46] Gropp W., Lusk E., and Skjellum A., Using MPI: Portable Parallel Programming with the Message-Passing Interface, The MIT Press, Cambridge, Massachusetts, 1994.
- [47] Karypis G., and Kumar V., A Fast and High Quality Scheme for Partitioning Irregular Graphs, *SIAM Journal of Scientific Computing*, **20**, pp.359-392, 1999.
- [48] Balay S., Gropp W. D., McInnes L. C., and Smith B. F., The Portable Extensible Toolkit for Scientific computing (PETSc), <http://www.mcs.anl.gov/petsc.html>
- [49] Manke J. W., Parallel Computing in Aerospace, *Parallel Computing*, **27**, pp.329-336, 2001.
- [50] Hoeflinger J., Alavilli P., Jackson T., and Kuhn B., Producing Scalable Performance with OpenMP: Experiments with Two CFD Applications, *Parallel Computing*, **27**, pp.391-413, 2001.
- [51] Geist A., Beguelin A., Dongarra J., Jiang W. C., Manchek R., and Sunderam V., PVM : Parallel Virtual Machine: A Users' Guide and Tutorial for Networked Parallel Computing, Scientific and Engineering Computation, MIT Press, Cambridge, MA, USA, 1994.

REFERENCE

- [52] McManus K., A Strategy for Mapping Unstructured Mesh Computational Mechanics Programs onto Distributed Memory Parallel Architectures, Ph.D. Thesis, School of Computing and Mathematical Science, University of Greenwich, 1996.
- [53] Yao J., Jameson A., Alonso J. J., and Liu F., Development and Validation of a Massively Parallel Flow Solver for Turbomachinery Flows, *38th Aerospace Sciences Meeting and Exhibit*, AIAA Paper 00-0882, January 10-13, 2000.
- [54] Das R., Mavriplis D. J., Saltz J., Gupta S., and Ponnusamy R., The Design and Implementation of a Parallel Unstructured Euler Solver using Software Primitives, AIAA-92-0562, *In Proceedings of the 30th Aerospace Sciences Meeting*, January 1992.
- [55] Mittal S., and Tezduyar T. E., Parallel Finite Element Simulation of 3D Incompressible Flows: Fluid-Structure Interaction, *International Journal for Numerical Methods in Fluids*, **21**, pp.933-953, 1995.
- [56] Kalro V., and Tezduyar T. E., 3D Computation of Unsteady Flow Past a Sphere with a Parallel Finite Element Method, *Army High Performance Computing Research Centre Reprint 97-011*, University of Minnesota, 1997.
- [57] Johnson A. A., and Tezduyar T. E., Parallel Computation of Incompressible Flows with Complex Geometries, *International Journal for Numerical Methods in Fluids*, **24**, pp.1321-1340, 1997.
- [58] Kalro V., Aliabadi S., Garrard W., Tezduyar T. E., Mittal S., and Stein K., Parallel Finite Element Simulation of Large Ram-Air Parachutes, *International Journal for Numerical Methods in Fluids*, **24**, pp.1353-1369, 1997.
- [59] Johnson A. A., and Tezduyar T. E., Advanced Mesh Generation and Update Methods for 3D Flow Simulations, *Supercomputer Institute Research Report*, University of Minnesota, 1998.

REFERENCE

- [60] E. Turkel, Improving the Accuracy of Central Difference Schemes, 11th International Conference on Numerical methods and Fluid Dynamics, *Springer-Verlag Lecture Notes in Physics*, **323**, pp.586-591, 1988.
- [61] E. Turkel, Robust Low Speed Preconditioning for Viscous High Lift Flows, AIAA Paper 2002-0962, *40nd AIAA Aerospace Sciences Meeting and Exhibit*, 14-17 January 2002/Reno, NV
- [62] R. Radespiel, E. Turkel and N. Kroll, Assessment of Preconditioning Methods, *Computers and Fluids*, **26**, pp.613-634, 1997
- [63] Zhao Y., Computation of Complex Turbulent Flow Using Matrix-Free Implicit Dual-Time Stepping Scheme and LRN Turbulence Model on Unstructured Grids, *Computers & Fluids*, **33**, pp.119-136, 2004.
- [64] E. Turkel, Preconditioned Methods for Solving the Incompressible and Low Speed Compressible Equations, *Journal of Computational Physics*, **72**, 1987
- [65] Bram Van Leer, Towards the Ultimate Conservation Difference Scheme V, A Second-order Sequel to Godunov's Method, *Journal of Computational Physics*, **32**, pp.101-136, 1979.
- [66] B. Van Leer, W.T. Lee, and P.L. Roe, Characteristic Time-Stepping or Local Preconditioning of the Euler Equations, *AIAA Paper* 91-1552, 1991.
- [67] D.L. Darmofal, B. Van Leer, Local Preconditioning: Manipulating Mother Nature to fool Father Time, in *Computing the Future II: Advances and Prospects in Computational Aerodynamics*, edited by M. Hafez and D.A. Caughey, Wiley, New York, 1998.
- [68] Y-H Choi and Charles L. Merkle, The Application of Preconditioning in Viscous Flows, *Journal of Computational Physics*, **105**, pp.207-223, 1993.

REFERENCE

- [69] Venkateswaran, S. and Charles L. Merkle, Dual-Time stepping and Preconditioning for Unsteady Computations, *AIAA Paper 95-0078, 33rd Aerospace Sciences Meeting and Exhibit*, January, 9-12, 1995.
- [70] S. Venkateswaran, P.E.O. Buelow and C.L. Merkle, Development of Linearised Preconditioning Methods for Enhancing Robustness and Efficiency of Euler and Navier-Stokes Computations, *AIAA Paper 97-2030*, 1997.
- [71] Venkateswaran, S. and Charles L. Merkle, Analysis of Time-Derivative Preconditioning for the Navier-Stokes Equations, *6th International Symposium on Computational Fluid Dynamics*, pp.1323-1328, 1995.
- [72] J.M. Weiss, W.A. Smith, Preconditioning Applied to Variable and Constant Density Flows, *AIAA Journal*, **33**, 2050, 1995.
- [73] Owen DRJ, Hinton E, *Finite Elements in Plasticity: Theory and Practice*, Pineridge Press Ltd., Swansea, UK, 1980.
- [74] Zienkiewicz O.C, Taylor RL, *The Finite Element Method: Volume 1: Basic Formulation and Linear Problems*, McGraw-Hill: Maidenhead, UK, 1989.
- [75] Patanker S.V, *Numerical Heat Transfer and Fluid Flow*, Hemisphere: Washington DC, 1980.
- [76] Hirsch C, *Numerical Computation of Internal and External Flows: Fundamentals of Numerical Discretization*, 1, Wiley: New York, 1988.
- [77] Zienkiewicz O.C, Origins, Milestones and Directions of the Finite Element method-a Personal View, *Archives of Computational Methods in Engineering*, **2**, pp.1-48, 1995.
- [78] Onate E, Cervera M, Zienkiewicz O.C, A Finite Volume Format for Structural Mechanics, *International Journal for Numerical Methods in Engineering*, **37**,

REFERENCE

- pp.181-201, 1994.
- [79] Taylor G.A. A Vertex-Based Discretization Scheme Applied to Material Non-Linearity within a Multi-Physics Finite Volume Framework, Ph.D. Thesis, the University of Greenwich, 1996.
 - [80] Demirdzic I, Martinovic D, Finite Volume Method for Thermo-Elasto-Plastic Stress Analysis, *Computer Methods in Applied Mechanics and Engineering*, **109**, pp.331-349, 1992.
 - [81] Hattel J.H, Hansen P.N, A Control Volume-Based Finite Difference Method for Solving the Equilibrium Equations in terms of displacements, *Applied Mathematical Modeling*, **19**, pp.210-243, 1995.
 - [82] Wheel M.A, A Geometrically Versatile Finite Volume Formulation for Plane Elastostatic Stress Analysis, *Journal of Strain Analysis*, **31(2)**, pp.111-116, 1996.
 - [83] Wheel M.A, A Mixed Finite Volume Formulation for Determining the Small Strain Deformation of Incompressible Materials, *International Journal for Numerical Methods in Engineering*, **44**, pp.1843-1861, 1999.
 - [84] Jasak H, Weller H.G, Application of the Finite Volume Method and Unstructured Meshes to Linear Elasticity, *International Journal for Numerical Methods in Engineering*, **48**, pp.267-287, 2000.
 - [85] Fryer Y.D, Bailey C, Cross M, Lai C-H, A Control Volume Procedure for Solving the Elastic Stress-Strain Equations on an Unstructured Mesh, *Applied Mathematical Modeling*, **15**, pp.639-645, 1991.
 - [86] Bailey C, Cross M, A Finite Volume Procedure to Solve Elastic Solid Mechanics Problems in Three Dimensions on an Unstructured Mesh, *International Journal for Numerical Methods in Engineering*, **38**, pp.1757-1776, 1995.

REFERENCE

- [87] Demirdzic I, Muzaferija S, Finite Volume Method for Stress Analysis in Complex Domains, *International Journal for Numerical Methods in Engineering*, **37**, pp.3751-3766, 1994.
- [88] Glowinski R., Pan T. W., and Periaux J., A Fictitious Domain Method for Dirichlet Problems and Applications, *Computer Methods in Applied Mechanics and Engineering*, **111**, pp.283-303, 1994.
- [89] Glowinski R., Pan T. W., and Periaux J., A Fictitious Domain Method for External Incompressible Viscous Flow Modeled by Navier-Stokes Equations, *Computer Methods in Applied Mechanics and Engineering*, **112**, pp.133-148, 1994.
- [90] Ghias R., Mittal R., Lund T.S., A Non-Body Conformal Grid Method for Simulation of Compressible Flows with Complex Immersed Boundaries. *AIAA Paper*: 2004-0800.
- [91] Mittal R., Seshadri V., Udaykumar H.S., Flutter, Tumble and Vortex Induced Autorotation. *Theoretical Computational Fluid Dynamics*, **17**(3): 165-70.
- [92] Iaccarino G., Kalizin G., Elkins C.J., Numerical and Experimental Investigation of the Turbulent Flow in Ribbed Serpentine Passage, *Annu. Res. Briefs 2001, Cent. Turbul. Res.* 379-388, 2003.
- [93] Kalitzin G., Iaccarino G., Khalighi B., Towards an Immersed Boundary Solver for RANS Simulations. *AIAA Paper*: 2003-0770. Fung Y. C., *Biodynamics – Circulation*, Springer – Verlag, New York, 1984.
- [94] Fedkiw R., et al., A Non-Oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (the Ghost Fluid Method), *Journal of Computational Physics*, **152**, pp.457-492, 1999.

REFERENCE

- [95] Hassan O., Probert E. J. and Morgan K., Unstructured Mesh Procedures for the Simulation of Three Dimensional Transient Compressible Inviscid Flows with Moving Boundary Components, *International Journal for Numerical Methods in Fluids*, **27**, pp.41-55, 1998.
- [96] Heil M., Stokes Flow in an Elastic Tube – A Large Displacement Fluid-Structure Interaction Problem, *International Journal for Numerical Methods in Fluids*, **28**, pp.243-265, 1998.
- [97] Farhat C., Michel L., and Nathan M., Mixed Explicit/Implicit Time Integration of Coupled Aero-elastic Problems: Three-field Formulation, Geometric Conservation and Distributed Solution, *International Journal for Numerical Methods in Fluids*, **21**, pp.807-835, 1995.
- [98] Schulz K. W., and Kallinderis Y., Unsteady Flow Structure Interaction for Incompressible Flows Using Deformable Hybrid Grids, *Journal of Computational Physics*, **143**, pp.569-597, 1998.
- [99] Kiris C., Kwak D., Rogers S., and Chang I-D., Computational Approach for Probing the Flow Through Artificial Heart Devices, *Journal of Biomechanical Engineering*, **119**, pp.452-460, 1997.
- [100] Farnell D. J. J., David T., and Barton D. C., Coupled States of Flapping Flags, *Journal of Fluids and Structures*, **19**, pp.29-36, 2004.
- [101] De Hart J., Peters G.W.M., Schreurs P.J.G., Baaijens F.P.T., A Three-Dimensional Computational Analysis of Fluid–Structure Interaction in the Aortic Valve, *Journal of Biomechanics*, **36**, pp.103-112, 2003.
- [102] Peskin C. S., Flow Patterns Around Heart Valves: A Digital Computer Method for Solving the Equation of Motion, Ph.D. Thesis, Albert Einstein College of Medicine, Yeshiva University, 1972.

REFERENCE

- [103] Baaijens F. P. T., A Fictitious Domain/Mortar Element Method for Fluid-Structure Interaction, *International Journal of Numerical Methods in Fluids*, **35**, pp.743-761, 2001.
- [104] De Hart J., Peters G.W.M., Schreurs P.J.G., Baaijens F.P.T., A Two-Dimensional Fluid-Structure Interaction Model of the Aortic Valve, *Journal of Biomechanics*, **33**, pp.1079-1088, 2000.
- [105] Bluestein D., and Einav S., Techniques in the Analysis of Stability of Pulsatile Flow through Heart Valves, In: Leondes, C.T. (Ed.), *Biomechanics Systems Techniques and Applications: Cardiovascular Techniques*, Vol. II. CRC Press LLC, Boca Raton, FL, pp.4-1-4-39, 2001.
- [106] Löhner R., Chi Y., Cezral J. R., Baum J. D., Luo H., Mestreau E., Pelessone D., and Charman C., Fluid-Structure-Thermal Interaction Using a Loose Coupling Algorithm and Adaptive Unstructured Grids, Technical Report, Institute for Computational Sciences and Informatics, George Mason University, 1998.
- [107] Professor Rainald Löhner's Homepage – Computational Fluid Dynamics – GMU, <http://www.science.gmu.edu/~rlohner>.
- [108] Johnson A. A., and Tezduyar T. E., Three-dimensional of Fluid-Particle Interactions with the Number of Particles Reaching 100, *Computer Methods in Applied Mechanics and Engineering*, **145**, pp.301-321, 1997.
- [109] Wren G. P., Ray S. E., Aliabadi S. K., and Tezduyar T. E., Space-Time Finite Element Computation of Compressible Flows between Moving Components, *International Journal for Numerical Methods in Fluids*, **21**, pp.981-991, 1995.
- [110] Wren G. P., Ray S. E., Aliabadi S. K., and Tezduyar T. E., Simulation of Flow Problems with Moving Mechanical Components, Fluid-structure Interactions and Two-fluid Interfaces, *International Journal for Numerical Methods in Fluids*, **24**, pp.1433-1448, 1997.

REFERENCE

- [111] Elad D., Katz D., Kimmel E., and Einav S., Numerical Schemes for Unsteady Fluid Flow through Collapsible Tubes, *Journal of Biomechanical Engineering*, **13**, pp.10-18, 1991.
- [112] Farhat C., Lesginne M., Maman N., Mixed Explicit/Implicit Time Integration of Coupled Aeroelastic Problems: Three-field Formulation, Geometric Conservation and Distributed Solution, *International Journal for Numerical Methods in Fluids*, **21**, pp.807-835, 1995.
- [113] Freitas F. B. Rachid, and Costa H.S. Mattos, Modelling of Pipeline Integrity Taking into Account the Fluid-Structure Interaction, *International Journal for Numerical Methods in Fluids*, **28**, pp.337-355, 1998.
- [114] Liang S. J., Neitzel G. P., and Aidun C. K., Finite Element Computations for Unsteady Fluid and Elastic Membrane Interaction Problems, *International Journal for Numerical Methods in Fluids*, **24**, pp.1091-1110, 1997.
- [115] Piperno S., Explicit/Implicit Fluid/Structure Staggered Procedures with a Structural Predictor and Fluid Subcycling for 2D Inviscid Aeroelastic Simulation, *International Journal for Numerical Methods in Fluids*, **25**, pp.1207-1226, 1997.
- [116] Soria A., and Casadei F., Arbitrary Lagrangian-Eulerian Multi-component Compressible Flow with Fluid-Structure Interaction, *International Journal for Numerical Methods in Fluids*, **25**, pp.1263-1284, 1997.
- [117] N.A. Pierce and M.B. Giles, Preconditioning Compressible Flow Calculations on Stretched Meshes, *ALAA Paper* 96-0889, 1996.
- [118] P. Moinier and M.B. Giles. Stability Analysis of Preconditioned Approximations of the Euler Equations on Unstructured Meshes, *Journal of Computational Physics*, **178**, pp.498-519, 2002.

REFERENCE

- [119] P. Moinier and M.B. Giles. Preconditioned Euler and Navier-Stokes Calculations on Unstructured Grids, *6th ICFD Conference on Numerical Methods for Fluid Dynamics*, Oxford, UK, 1998.
- [120] S. Venkateswaran, D. Li and C.L. Merkle, Influence of Stagnation Regions on Preconditioned Solutions at Low Speeds, AIAA-2003-0435, *41st Aerospace Sciences Meeting and Exhibit*, Reno, NV, January 6-9, 2003.
- [121] S. Venkateswaran, and C.L. Merkle, Dual-Time stepping and Preconditioning for Unsteady Computations, *AIAA Paper 95-0078*, *33rd Aerospace Sciences Meeting and Exhibit*, January, 9-12, 1995.
- [122] S. Venkateswaran, P.E.O. Buelow and C.L. Merkle, Development of Linearized Preconditioning Methods for Enhancing Robustness and Efficiency of Euler and Navier-Stokes Computations, *AIAA Paper 97-2030*, 1997.
- [123] S. Venkateswaran, C.L. Merkle, Analysis of Time-Derivative Preconditioning for the Navier-Stokes Equations, *6th International Symposium on Computational Fluid Dynamics*, pp.1323-1328, 1995.
- [124] E. Turkel, Preconditioned Methods for Solving the Incompressible and Low Speed Compressible Equations, *Journal of Computational Physics*, **72**, 1987.
- [125] B. Van Leer, W.T. Lee, and P.L. Roe, Characteristic Time-Stepping or Local Preconditioning of the Euler Equations, *AIAA Paper 91-1552*, 1991.
- [126] J.M. Weiss, W.A. Smith, Preconditioning Applied to Variable and Constant Density Flows, *AIAA Journal* **33(11)**, 2050, 1995.
- [127] Roe P. L., Approximate Riemann Solvers, Parameter Vectors and Difference Schemes, *Journal of Computational Physics*, **43**, pp.357-372, 1981.

REFERENCE

- [128] Jae Wook Kim and Duck Joo Lee, Generalized Formulation and Application of Characteristic Boundary Conditions, *AIAA Paper*, 98-222.
- [129] Jae Wook Kim and Duck Joo Lee, Implementation of Boundary Conditions for Optimized High-Order Compact Schemes, *Journal of Computational Acoustics*, **5 (No.2)**, pp.177-191, 1997.
- [129] Thompson, K. W., Time Dependent Boundary Conditions for Hyperbolic systems, *Journal of Computational Physics*, **68**, pp.1-24, 1987.
- [130] Thompson, K. W., Time Dependent Boundary Conditions for Hyperbolic systems II, *Journal of Computational Physics*, **89**, pp.439-461, 1990.
- [131] Poinso, T. J. and Lele, S. K., Boundary Conditions for Direct Simulations of Compressible Viscous Flows, *Journal of Computational Physics*, **101**, pp.104-129, July 1992.
- [132] Steger, J. L., Warming, R. F., Flux Vector Splitting of the Inviscid Gasdynamic Equations with Application to Finite-Difference Methods, *Journal of Computational Physics*, **40**, pp.263-293, 1981.
- [133] Joseph L. Steger and R. F. Warming, Flux Vector Splitting of the Inviscid Gasdynamic Equations with Application to Finite Difference Methods, *NASA contractor report*, HC A04/MF A01, N79-28950.
- [134] R.T. Fenner, Engineering Elasticity: Applications of Numerical and Analytical Techniques, Ellis Horwood, 1986.
- [135] S.P. Timoshenko, J.N. Goodier, Theory of Elasticity, McGraw-Hill, 1982.
- [136] T.J.R. Hughes, The Finite Element Method, Linear Static and Dynamic Finite Element Analysis, Prentice-Hall, 1987.

REFERENCE

- [137] R.D. Blevins, Flow Induced Vibration, Van Nostrand Reinhold, 1977.
- [138] R.R. Craig Jr., Structural Dynamics, Wiley & Sons, 1981.
- [139] S.W. Key, Transient Response by Time Integration, in J. Donéa (Ed.) Advanced Structural Dynamics, *Applied Science Publishers*, pp.71-95, 1978.
- [140] O.C. Zienkiewicz, R.L. Taylor, The Finite Element Method, Volumes 1-3, Butterworth-Heinemann, 2000.
- [141] X. Lv, Y. Zhao, *et al.*, An Efficient Parallel/Unstructured-Multigrid Preconditioned Implicit Method for Simulating 3D Unsteady Compressible Flows with Moving Objects, *Journal of Computational Physics*, **215**, Issue 2, Pages 661-690, 1 July 2006.
- [142] X. Lv, Y. Zhao, X. Y. Huang, G. H. Xia, A Matrix-Free Implicit Unstructured Multigrid Finite Volume Method for Simulating Structural Dynamics and Fluid-Structure Interaction, *Journal of Computational Physics*, in press, 2006.
- [143] G. H. Xia, Y. Zhao, J. H. Yeo and X. Lv, A 3D Implicit Unstructured-grid Finite Volume Method for Structural Dynamics, *Computational Mechanics*, in press, 2006.
- [144] X.H. Su, Y. Zhao, X.Y. Huang, X. Lv and C. Luo, Parallel 3D Gas Dynamic of Nonlinear Resonant Oscillations in a Closed Resonator, *submitted to International Journal for Numerical Methods in Engineering*, 2006.
- [145] S. A. Bayyuk, K. G. Powell, and B. van Leer, A Simulation Technique for 2-D Unsteady Inviscid Flows around Arbitrarily Moving and Deforming Bodies of Arbitrary Geometry, *Technical Paper 93-3391-CP* (AIAA Press, Washington DC, 1993).

REFERENCE

- [146] Jianming Yang and Elias Balaras, An Embedded-Boundary Formulation for Large-Eddy Simulation of Turbulent Flows Interacting with Moving Boundaries, *Journal of Computational Physics*, **215** (1), pp.12-40, 2006.
- [147] H.S. Udaykumar, R. Mittal, P. Rampunggoon, A. Khanna, A sharp interface Cartesian grid method for simulating flows with complex moving boundaries, *Journal of Computational Physics*, **174**, pp.345-380, 2001.
- [148] Franke R., Scattered data interpolation: Tests of some methods, *Math. Computer*, **38**, pp.181-200, 1982.
- [149] John D. Anderson, JR., Computational Fluid Dynamics: The Basics with Applications, McGraw-Hill, Inc.
- [150] Tai C. H., Numerical Simulation of Prosthetic Heart Valve with Coupled Fluid Structure Interaction Under Pulsatile Flow, Ph.D. First Year Report, Mechanical and Production Engineering, Nanyang Technological University, 1996.
- [151] Luigi M., Calculations of Viscous Flows with a Multigrid Method, Ph.D. Thesis, Princeton University, 1987.
- [152] Mitty T. J., Development of a Delaunay-Based Adaptation Scheme with Applications to Complex Three Dimensional Rotational Flows, Ph.D. Thesis, Princeton University, 1993.
- [153] Tai C. H., and Zhao Y., A Finite Volume Unstructured Multigrid Method for Efficient Computation of Unsteady Incompressible Viscous Flows, *International Journal for Numerical Methods in Fluids*, John Wiley & Sons, 2003.
- [154] Zhao Y., and Tai C. H., Parallel Unsteady Incompressible Viscous Flow Computations using an Unstructured Multigrid Method, *Parallel CFD 2002: 13th International Conference*, JAERI-KRE, Kansai Science City, Japan, May 20-22, 2002.

REFERENCE

- [155] Zhao Y., and Tai C. H., Parallel Computation of Unsteady Incompressible Viscous Flows using an Unstructured Multigrid Method, *ASME Forum on Parallel Computing Methods*, Paper No. IMECE2002-FE-34388, Proceedings of IMECE'02, 2002 ASME International Mechanical Engineering Congress & Exposition New Orleans, Louisiana, November 17-22, 2002.
- [156] Karypis G., and Kumar V., A Parallel Algorithm for Multilevel Graph Partitioning and Sparse Matrix Ordering, *Journal of Parallel and Distributed Computing*, **48**, pp. 71-85, 1998.
- [157] Karypis G., and Kumar V., Metis: A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices, Version 4.0, University of Minnesota, Department of Computer Science, Sept 1998.
- [158] Mavriplis D. J., Large-scale Parallel Viscous Flow Computations Using An Unstructured Multigrid Algorithm, NASA/CR-1999-209724, ICASE Report No. 99-44, Nov 1999.
- [159] Mavriplis D. J., Parallel Unstructured Mesh Analysis Of High-Lift Configurations, AIAA Paper 2000-0923, *38th AIAA Aerospace Sciences Meeting*, January 2000.
- [160] Peter S. Pacheco., *Parallel Programming with MPI*, Morgan Kaufmann Publishers, Inc., San Francisco, California, 1997.
- [161] Leonard, A., Energy Cascade in Large-Eddy Simulations of Turbulent Fluid Flows, *Advances in Geophysics*, **18A**, pp.237-248, 1974.
- [162] Erlebacher, G., Hussaini, M. Y., Speziale, C. G., and Zang, T. A., Toward the Large-eddy Simulation of Compressible Turbulent Flows, *ICASE Report*, pp.90-76, NASA Langley Research Center, 1990.

REFERENCE

- [163] Cebeci, T. and Smith, A. M. O., Analysis of Turbulent Boundary Layers, Academic Press, New York, 1974.
- [164] Vreman, B., Geurts, B. and Kuerten H., Subgrid-Modeling in LES of Compressible Flow, *Applied Scientific Research*, Vol. 54, pp.191-203, 1995.
- [165] Vreman, B., Geurts, B., and Kuerten, H., A Priori Tests of Large Eddy Simulation of the Compressible Plane Mixing Layer, *J. Eng. Math.*, **29**, pp.299-327, 1995.
- [166] Christer, F., Fernando, F. and Grinstein, Large Eddy Simulation of High Reynolds Number Free and Wall-Bounded Flows, *Journal of Computational Physics*, **181**, pp.68-97, 2002.
- [167] Moin, P., and Jimenez, J., Large Eddy Simulation of Complex Turbulent Flows, *AIAA 24th Fluid Dynamics Conference*, Orlando, 1993.
- [168] Zang, Y., Street, R. L., and Koseff, J. R, A Dynamic Mixed Subgrid-Scale Model and its Application to Turbulent Recirculating Flows, *Phys. Fluids*, **5**, pp.3186-3196, 1993.
- [169] Vreman, B., Geurts, B., and Kuerten, H. (b), On the Formulation of the Dynamic Mixed Subgrid-Scale Model, *Phys. Fluids*, **6**, pp.4057-4059, 1994.
- [170] Germano, M., Turbulence: the Filtering Approach, *J. Fluid Mech.*, **238**, pp.325-336, 1992.
- [171] Germano, M., Piomelli, U., Moin, P., and Cabot, W.H., A Dynamic Subgrid-Scale Eddy Viscosity Model, *Phys. Fluids A*, **3**, pp.1760-1765, 1991.

REFERENCE

- [172] Yoshizawa, A., Statistical Theory for Compressible Turbulent Shear Flows, With the Application to Subgrid Modeling, *Phys. Fluids*, **29**, pp.2152-2164, 1986.
- [173] Moin, P., Squires, K., Cabot, W., and Lee, S., A Dynamic Subgrid-Scale Model for Compressible Turbulence and Scalar Transport, *Phys. Fluids A*, **3**, pp.2746-2757, 1991.
- [174] El-Hady, N. M., Zang, T. A. and Piomelli, U., Dynamic Subgrid-Scale Modeling for High-Speed Transitional Boundary Layers, *ASME FED*, **162**, pp.103-112, 1993.
- [175] Vreman, A. W., Geurts, B. J., and Kuerten, J. G. M., Subgrid-Modeling in LES of Compressible Flow, *Direct and Large-Eddy Simulation I*, pp.133-144, 1994.
- [176] Vreman, B., Geurts, B., and Kuerten, H., A Priori Tests of Large Eddy Simulation of the Compressible Plane Mixing Layer, *J. Eng. Math.*, **29**, pp.299-327, 1995.
- [177] Eidson, T. M., Numerical Simulation of the Turbulent Rayleigh-Bénard Problem Using Subgrid Modeling, *J. Fluid Mech.*, **158**, pp.245-268, 1985.
- [178] Lilly, D. K., A Proposed Modification of the Germano Subgrid-Scale Closure Method, *Phys. Fluids A*, **4**, pp.633-635, 1992.
- [179] Piomelli, U., and Liu, J., Large-Eddy Simulation of rotating channel flows using a localized dynamic model: Application of Direct and Large Eddy Simulation to Transition and Turbulence, *AGARD Conference Proceedings*, **551**, 3.1-9, 1994.
- [180] Horiuti, K., Large Eddy Simulation of Turbulent Channel Flow by One-Equation Modeling, *J. Phys. Soc. Japan*, **54**, pp.2855-2865, 1985.

REFERENCE

- [181] Ghosal, S., Lund, T.S., Moin, P., and Akselvoll, K., A Dynamic Localization Model for Large-Eddy Simulation of Turbulent Flows, *J. Fluid Mech.*, **286**, pp.229-255, 1995.
- [182] Ghosal, S., and Moin, P., The Basic Equations for the Large Eddy Simulation of Turbulent Flows in Complex Geometry, *J. Comp. Phys.*, **118**, pp.24-37, 1995.
- [183] Todd Allen Simons, Large Eddy Simulation of turbulent flows using finite volume methods with structured, unstructured, and zonal embedded grids, Ph.D Thesis, Iowa State University, 1998.
- [184] Alison L. Marsden, Oleg V. Vasilyev, and Parviz Moin, Construction of Commutative Filters for LES on Unstructured Meshes, *J. Comp. Phys.*, **175**, pp.584-603, 2003.
- [185] Andreas Haslbacher, Oleg V. Vasilyev, Commutative Discrete Filtering on Unstructured Grids Based on Least-Square Techniques, *J. Comp. Phys.*, **187**, pp.197-211, 2003.
- [186] Andreas Haslbacher, Private Communication, 2006.
- [187] E. Balaras, Modeling Complex Boundaries Using an External Force Field on Fixed Cartesian Grids in Large-Eddy Simulations, *Comput. Fluids*, **33**, pp.375-404, 2004.
- [188] Anvar Gilmanov, Fotis Sotiropoulos, A Hybrid Cartesian/Immersed Boundary Method for Simulating Flows with 3D, Geometrically Complex, moving bodies, *Journal of Computational Physics*, **207**, Issue 2, pp.457-492, 2005.
- [189] S.C.R. Dennis, S.N. Singh, D.B. Ingham, The steady flow due to a rotating sphere at low and moderate Reynolds numbers, *Phys. Fluids*, **101**, pp.257-279, 1980.

REFERENCE

- [190] Van Dyke and Milton D., An Album of Fluid Motion, Parabolic Press, Stanford, California, USA, 1982.
- [191] M. Nishioka, and H. Sato, Mechanism of Determination of the Shedding Frequency of Vortices behind a Cylinder at Low Reynolds Numbers, *Journal of Fluid Mechanics*, **89**, pp.49-60, 1978.
- [192] A.K. Slone, C. Bailey, M. Cross, Dynamic Solid Mechanics Using Finite Volume Methods, *Applied Mathematical Modeling*, **27**, pp.69-87, 2003.
- [193] A.K. Slone, K. Pericleous, C. Bailey, M. Cross, C. Bennett, A Finite Volume Unstructured Mesh Approach to Dynamic Fluid-Structure Interaction: an Assessment of the Challenge of Predicting the Onset of Flutter, *Applied Mathematical Modeling*, **28**, pp.211-239, 2004.
- [194] C. A. Schuler, W. R. Usry, B. Weber, J. A. C. Humphrey, and R. Greif, On the Flow in the Unobstructed Space between Shrouded Corotating Disks, *Phys. Fluids A*, **2**, pp.1760, 1990.
- [195] C. J. Chang, C. A. Schuler, J. A. C. Humphrey, and R. Greif, Flow and Heat Transfer in the Space between two Corotating Disks in an Axisymmetric Enclosure, *J. Heat Transfer*, **111**, pp.625, 1989.
- [196] C. J. Chang, J. A. C. Humphrey, and R. Greif, Calculation of turbulent convection between corotating disks in axisymmetric enclosures, *Int. J. Heat Mass Transfer*, **33**, pp.2701, 1990.
- [197] C. A. Schuler, Investigation of the Flow between Rotating Disks in an Enclosure, Ph.D. thesis, University of California, Berkeley, 1990.
- [198] Immaculada Iglesias, and J. A. C. Humphrey, Two- and Three-Dimensional Laminar Flows Between Disks Co-Rotating in a Fixed Cylindrical Enclosure, *Int. J. Numer. Meth. Fluids*, **26**, pp.581-603, 1998.

REFERENCE

- [199] J. A. C. Humphrey, C. A. Schuler and D. R. Webster, Unsteady Laminar Flow between a Pair of Disks Corotating in a Fixed Cylindrical Enclosure, *Phys. Fluids A*, **7**, pp.1225-1240, 1995.
- [200] J. A. C. Humphrey and D. Gor, Experimental Observations of an Unsteady Detached Shear Layer in Enclosed Corotating Disk Flow, *Phys. Fluids A*, **5**, pp.2438-2442, 1993.
- [201] A. Z. Szeri, S. J. Schneider, F. Labbe and H. N. Kaulman, Flow between Rotating Disks; Part I, basic flow, *J. Fluid Mech.*, **134**, pp.103-131, 1983.
- [202] Schmitt, V. and F. Charpin, Pressure Distributions on the ONERA-M6-Wing at Transonic Mach Numbers, Experimental Data Base for Computer Program Assessment, *Report of the Fluid Dynamics Panel Working Group 04*, AGARD AR 138, May 1979.
- [203] Hwang, G., Parallel Finite Element Solutions of Nonlinear Aeroelastic and Aero-servoelastic Problems in Three-Dimensional Transonic Flows, Ph.D. Dissertation, Mechanical and Aerospace Engineering Department, University of California, Los Angeles, CA, 1997.
- [204] Oddvar O. Bendiksen, Hwang, G. Y., Transonic flutter suppression using dynamic twist control, AIAA/ASME/ASCE/AHS/ASC Structures, *Structural Dynamics and Materials Conference and Exhibit*, 37th, Salt Lake City, UT, Apr. 15-17, 1996, Technical Papers. Pt. 4 (A96-26801 06-39), Reston, VA, American Institute of Aeronautics and Astronautics, pp.2670-2684, 1996.
- [205] Guclu Seber and Oddvar O. Bendiksen, Nonlinear Flutter Calculations Using Finite Elements in a Large Deformation Direct Eulerian-Lagrangian Formulation, 46th AIAA/ASME/ASCE/AHS/ASC Structures, *Structural Dynamics & Materials Conference Austin, Texas*, 2005.

REFERENCE

- [206] Liu, F., Cai, J., Zhu, Y., Wong, A.S.F., and Tsai, H.M., Calculation of Wing Flutter by a Coupled Fluid-Structure Method, *Journal of Aircraft*, Col. 38 No. 2, pp.334-342, March-April, 2001.

- [207] Tsai, H.M., Wong, A.S.F., Cai, J., Zhu, Y., and Liu, F., Unsteady Flow Calculations with a Parallel Multiblock Moving Mesh Algorithm, *AIAA Journal*, Vol. 39, No. 6, pp. 1021-1029, Jun., 2001.