A Thesis

entitled

Improved Helicopter Rotor Performance Prediction through Loose and Tight CFD/CSD Coupling

by

Jacob C. Ickes

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Masters of Science Degree in Mechanical Engineering

________________________
Dr. Chunhua Sheng, Committee Chair

________________________
Dr. Ray Hixon, Committee Member

________________________
Dr. Abdeh Afjeh, Committee Member

________________________
Dr. Glenn Lipscomb, Committee Member

________________________
Dr. Patricia R. Komuniecki, Dean
College of Graduate Studies

The University of Toledo
December 2014
An Abstract of

Improved Helicopter Rotor Performance Prediction through Loose and Tight CFD/CSD Coupling

by

Jacob C. Ickes

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Masters of Science Degree in Mechanical Engineering

The University of Toledo
December 2014

Helicopters and other Vertical Take-Off or Landing (VTOL) vehicles exhibit an interesting combination of structural dynamic and aerodynamic phenomena which together drive the rotor performance. The combination of factors involved make simulating the rotor a challenging and multidisciplinary effort, and one which is still an active area of interest in the industry because of the money and time it could save during design. Modern tools allow the prediction of rotorcraft physics from first principles. Analysis of the rotor system with this level of accuracy provides the understanding necessary to improve its performance. There has historically been a divide between the comprehensive codes which perform aeroelastic rotor simulations using simplified aerodynamic models, and the very computationally intensive Navier-Stokes Computational Fluid Dynamics (CFD) solvers. As computer resources become more available, efforts have been made to replace the simplified aerodynamics of the comprehensive codes with the more accurate results from a CFD code.

The objective of this work is to perform aeroelastic rotorcraft analysis using first-principles simulations for both fluids and structural predictions using tools available at the University of Toledo. Two separate codes are coupled together in both loose coupling (data exchange on a periodic interval) and tight coupling (data exchange each time step) schemes. To allow the coupling to be carried out in a reliable and
efficient way, a Fluid-Structure Interaction code was developed which automatically performs primary functions of loose and tight coupling procedures. Flow phenomena such as transonics, dynamic stall, locally reversed flow on a blade, and Blade-Vortex Interaction (BVI) were simulated in this work. Results of the analysis show aero-dynamic load improvement due to the inclusion of the CFD-based airloads in the structural dynamics analysis of the Computational Structural Dynamics (CSD) code. Improvements came in the form of improved peak/trough magnitude prediction, better phase prediction of these locations, and a predicted signal with a frequency content more like the flight test data than the CSD code acting alone. Additionally, a tight coupling analysis was performed as a demonstration of the capability and unique aspects of such an analysis.

This work shows that away from the center of the flight envelope, the aerodynamic modeling of the CSD code can be replaced with a more accurate set of predictions from a CFD code with an improvement in the aerodynamic results. The better predictions come at substantially increased computational costs between 1,000 and 10,000 processor-hours.
Acknowledgments

Special acknowledgement is due to those who have contributed to this work directly and indirectly. My research associates, Jingyu Wang and Dr. Qiuying Zhao, lent their skill and knowledge in setting up and running some of the CFD cases. My advisor, Dr. Chunhua Sheng, guided my path and held me to the proper level of rigor in carrying out this work. The members of the thesis committee, Drs. Abdeh Afjeh, Ray Hixon, and Glenn Lipscomb, have taken time out of their lives to review this thesis and sit for the defense and their effort is greatly appreciated.

Dr. Marilyn Smith of the Georgia Institute of Technology provided the UH-60A DYMORE model, and Dr. Robert Kufeld of NASA Ames provided the flight test data for the UH-60A. Mr. Mark Dreier clarified numerous rotorcraft phenomena, and his book was used as a general reference. Mr. Alan Egolf of Sikorsky, Dr. Nischint Rajmohan of Advanced Rotorcraft Technology, and Dr. Jennifer Abras of NAVAIR all contributed their understanding of various parts of the computational methods or codes involved.

My parents and sister morally supported me throughout this period of my life, when it was much needed. And finally, I owe deepest thanks to Karyn, whose good humor is as perfect as her proofreading.
Contents

Abstract iii

Acknowledgments v

Contents vi

List of Tables x

List of Figures xi

List of Abbreviations xv

List of Symbols xvi

1 Introduction 1

1.1 Research Motivation and Objectives ......................... 2

1.2 The Physical Problem ...................................... 2

1.2.1 Flow Phenomena ........................................ 3

1.2.2 Flight Mechanics and Trim Conditions .................. 5

1.3 Previous Work ............................................. 9

1.4 Layout of this Document ................................... 12

2 Computational Methods 13

2.1 The U^2NCLE Flow Solver .................................. 13

2.1.1 Governing Equations .................................... 14
2.1.2 Discretization and Solution Procedure 17
2.1.3 Free Wake Model 23
2.1.4 Grid Deformation 25

2.2 The DYMORE Structural Solver 28
2.2.1 Multibody Dynamics Features 29
2.2.2 Aerodynamic Models 29
2.2.3 Autopilot Feature 30
2.2.4 Virtual Sensors and Post-Processing 31

2.3 The Fluid-Structure Interface 32
2.3.1 Design Requirements 33
2.3.2 Loose Coupling Procedure 36
2.3.3 Tight Coupling Procedure 40
2.3.4 Other Code Features 42
2.3.4.1 Restart Capability 42
2.3.4.2 Convergence Metric Compilation 42
2.3.4.3 Deflection Evaluation 43

3 Model Setup Description 44
3.1 Coordinate Systems 44
3.2 U^2NCLE Models 47
3.2.1 U^2NCLE Inputs 47
3.2.2 U^2NCLE Outputs 51
3.3 DYMORE 2 Models 52
3.3.1 DYMORE Inputs 53
3.3.1.1 Structural Specification 55
3.3.1.2 Aerodynamic Specification 55
3.3.1.3 Control System Specification 56
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.3.2</td>
<td>DYMORE Outputs</td>
<td>56</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Setup Procedure</td>
<td>57</td>
</tr>
<tr>
<td>3.4</td>
<td>Model Setup for Coupling</td>
<td>58</td>
</tr>
<tr>
<td>3.5</td>
<td>The Fluid-Structure Interface</td>
<td>59</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Loose Coupling Implementation</td>
<td>59</td>
</tr>
<tr>
<td>3.5.2</td>
<td>Tight Coupling Implementation</td>
<td>62</td>
</tr>
<tr>
<td>4</td>
<td>Results</td>
<td>65</td>
</tr>
<tr>
<td>4.1</td>
<td>UH-60A Loose Coupling Results</td>
<td>65</td>
</tr>
<tr>
<td>4.1.1</td>
<td>UH-60A Problem Setup</td>
<td>66</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Flight Conditions</td>
<td>69</td>
</tr>
<tr>
<td>4.1.3</td>
<td>Flight Counter c8534 Results</td>
<td>70</td>
</tr>
<tr>
<td>4.1.3.1</td>
<td>Convergence</td>
<td>70</td>
</tr>
<tr>
<td>4.1.3.2</td>
<td>Airloads</td>
<td>72</td>
</tr>
<tr>
<td>4.1.3.3</td>
<td>Structural Loads</td>
<td>81</td>
</tr>
<tr>
<td>4.1.4</td>
<td>Flight Counter c8513 Results</td>
<td>84</td>
</tr>
<tr>
<td>4.1.4.1</td>
<td>Convergence</td>
<td>84</td>
</tr>
<tr>
<td>4.1.4.2</td>
<td>Airloads</td>
<td>85</td>
</tr>
<tr>
<td>4.1.4.3</td>
<td>Structural Loads</td>
<td>93</td>
</tr>
<tr>
<td>4.1.5</td>
<td>Flight Counter c9017 Results</td>
<td>96</td>
</tr>
<tr>
<td>4.1.5.1</td>
<td>Convergence</td>
<td>96</td>
</tr>
<tr>
<td>4.1.5.2</td>
<td>Airloads</td>
<td>97</td>
</tr>
<tr>
<td>4.1.5.3</td>
<td>Structural Loads</td>
<td>105</td>
</tr>
<tr>
<td>4.1.6</td>
<td>Flight Counter c9020 Results</td>
<td>108</td>
</tr>
<tr>
<td>4.1.6.1</td>
<td>Convergence</td>
<td>108</td>
</tr>
<tr>
<td>4.1.6.2</td>
<td>Airloads</td>
<td>110</td>
</tr>
<tr>
<td>4.2</td>
<td>Bell 427 Loose Coupling Results</td>
<td>118</td>
</tr>
</tbody>
</table>
# List of Tables

2.1 U²NCLE internal variable normalization .......................... 16  
2.2 FSI Requirements .................................................. 34  
2.3 FSI Requirements, continued ..................................... 35  
2.4 Design Requirements of the Fluid-Structure Interface .......... 35  
4.1 UH-60A flight conditions ........................................... 70  
4.2 Bell 427 Main Rotor Forward Flight Conditions ................. 119
## List of Figures

1-1 Rotor layout showing transonic (T) and reverse flow (R) regions . . . . . 4
1-2 Blade motion conventions ............................................. 5
1-3 Helicopter profile and some important reference frames for analysis . . 6
1-4 Flight envelope for level flight of the UH-60A ............................ 8
1-5 Operating conditions of “high-thrust” flight conditions (UH-60A c9017 and c9020) . ............................................. 9

2-1 Features of the free wake model ........................................ 25
2-2 Grid deformation, with the support radius visible ........................ 28
2-3 The Loose Coupling Process ............................................ 38
2-4 Coupling data exchange process ........................................ 39
2-5 Tight Coupling process flowchart ...................................... 41

3-1 U²NCLE Body coordinate system and the Component Frame .......... 46
3-2 Black box U²NCLE functionality ....................................... 49
3-3 DYMOR E black-box behavior ........................................ 53
3-4 Example DYMOR E hierarchy .......................................... 54
3-5 Directory Configuration Created by the FSI to organize coupling process data .................................................. 61

4-1 The topology of a single blade of the 4-bladed UH-60A DYMOR E model 66
4-2 Computational domain of the UH-60A single blade ..................... 68
4-27 UH-60A flight counter c9017 airloads, r/R = 0.400 . . . . . . . . . . . . 100
4-28 UH-60A flight counter c9017 airloads, r/R = 0.775 . . . . . . . . . . . . 101
4-29 UH-60A flight counter c9017 airloads, r/R = 0.875 . . . . . . . . . . . . 102
4-30 UH-60A flight counter c9017 airloads, r/R = 0.925 . . . . . . . . . . . . 103
4-31 UH-60A flight counter c9017 airloads, r/R = 0.975 . . . . . . . . . . . . 104
4-32 UH-60A flight counter c9017 flapping moments, r/R = 0.3, 0.7 . . . . . 106
4-33 UH-60A flight counter c9017 torsional moments, r/R = 0.3, 0.7 . . . . 107
4-34 Trim setting convergence during CFD/CSD coupling, UH-60A flight counter
c9020 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 109
4-35 Control output divergence for c9020 based on DYMORE aerodynamics . 109
4-36 Wake model results for the UH-60A flight counter c9020 . . . . . . . . 111
4-37 UH-60A flight counter c9020 airloads, r/R = 0.225 . . . . . . . . . . . . 112
4-38 UH-60A flight counter c9020 airloads, r/R = 0.400 . . . . . . . . . . . . 113
4-39 UH-60A flight counter c9020 airloads, r/R = 0.775 . . . . . . . . . . . . 114
4-40 UH-60A flight counter c9020 airloads, r/R = 0.875 . . . . . . . . . . . . 115
4-41 UH-60A flight counter c9020 airloads, r/R = 0.925 . . . . . . . . . . . . 116
4-42 UH-60A flight counter c9020 airloads, r/R = 0.975 . . . . . . . . . . . . 117
4-43 Bell 427 DYMORE model layout, from Morillo et al. [19] . . . . . . . . . 118
4-44 Control System Convergence for the Bell 427 . . . . . . . . . . . . . . . 120
4-45 Bell 427 Free Wake Structure . . . . . . . . . . . . . . . . . . . . . . . 121
4-46 Bell 427 beam bending moment, r/R = 0.43 . . . . . . . . . . . . . . . 122
4-47 Bell 427 chord bending moment, r/R = 0.42 . . . . . . . . . . . . . . . 122
4-48 Bell 427 torsional moment, r/R = 0.43 . . . . . . . . . . . . . . . . . . 123
4-49 Bell 427 pitch link load . . . . . . . . . . . . . . . . . . . . . . . . . . . 123
4-50 Bell 427 beam bending moment, r/R = 0.43, LC vs. TC . . . . . . . . 125
4-51 Bell 427 chord bending moment, r/R = 0.43, LC vs. TC . . . . . . . . 125
4-52 Bell 427 torsional moment, r/R = 0.43, LC vs. TC . . . . . . . . . . . . 126
### List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BVI</td>
<td>Blade-Vortex Interaction</td>
</tr>
<tr>
<td>CFL</td>
<td>Courant-Friedrichs-Lewy number</td>
</tr>
<tr>
<td>FVL</td>
<td>Future Vertical Lift</td>
</tr>
<tr>
<td>FSI</td>
<td>Fluid-Structure Interface</td>
</tr>
<tr>
<td>JMR</td>
<td>Joint Multi-Role</td>
</tr>
<tr>
<td>LE</td>
<td>Leading Edge</td>
</tr>
<tr>
<td>MPI</td>
<td>Message-Passing Interface</td>
</tr>
<tr>
<td>U²NCLE</td>
<td>Unsteady Unstructured Computation of Field Equations</td>
</tr>
<tr>
<td>RANS</td>
<td>Reynolds-Averaged Navier-Stokes</td>
</tr>
<tr>
<td>TE</td>
<td>Trailing Edge</td>
</tr>
<tr>
<td>VTOL</td>
<td>Vertical Take-off and Landing</td>
</tr>
</tbody>
</table>
List of Symbols

a ............ Speed of Sound
A ............ Rotor disk area, Control surface area
A₀ ........... Collective Angle
A₁ ........... Longitudinal Cyclic Angle
A₂ ........... Lateral Cyclic Angle
A_{CFD} ...... Airloads (forces and moments for each airstation at each azimuth position) from CFD
A_{CSD} ...... Airloads (forces and moments for each airstation at each azimuth position) from CSD
ΔA .......... Delta Airloads
Cₜ .......... Thrust Coefficient, US customary definition, $C_t = T/\rho A (\Omega R)^2$
cₚ .......... Specific heat at constant pressure
cᵥ .......... Specific heat at constant volume
CₙM² ........ Sectional normal force coefficient, $C_n M^2 = L/0.5 \rho a^2 L^2$
CₘM² ........ Sectional pitching moment coefficient, $C_m M^2 = M/0.5 \rho a^2 L^3$
d .......... Derivative operator
e .......... Specific energy
E .......... Total energy
Eₐ .......... Eckert number, $(\gamma - 1) M_r^2$
g .......... Generalized viscous term
G .......... Gain Matrix, Flux vector in y direction
h .......... Specific enthalpy
H .......... Total enthalpy, Flux vector in z direction
J .......... Jacobian Matrix
k .......... Thermal conductivity
L .......... Length
M_r .......... Reference Mach number
M₀ .......... Rotating Mach number
n .......... Vector normal to surface
p .......... Pressure
P .......... Viscous or inviscid flux
R .......... Gas constant for air
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Re</td>
<td>Reynolds number</td>
</tr>
<tr>
<td>t</td>
<td>Time</td>
</tr>
<tr>
<td>T</td>
<td>Temperature, Rotor thrust</td>
</tr>
<tr>
<td>u</td>
<td>Velocity in x direction</td>
</tr>
<tr>
<td>u</td>
<td>Vector of control system outputs</td>
</tr>
<tr>
<td>v</td>
<td>Velocity in y direction</td>
</tr>
<tr>
<td>\vec{V}</td>
<td>Velocity Vector</td>
</tr>
<tr>
<td>w</td>
<td>Velocity in z direction, Characteristic variable</td>
</tr>
<tr>
<td>W</td>
<td>Preconditioned system characteristic variable vector</td>
</tr>
<tr>
<td>y</td>
<td>Vector of control system inputs</td>
</tr>
<tr>
<td>α</td>
<td>Local blade section angle of attack or rotor shaft angle of attack</td>
</tr>
<tr>
<td>β</td>
<td>Rotor blade flapping angle or preconditioning coefficient</td>
</tr>
<tr>
<td>Δ</td>
<td>Change or difference operator</td>
</tr>
<tr>
<td>γ</td>
<td>Rotor blade lead-lag angle, Ratio of specific heats</td>
</tr>
<tr>
<td>Γ</td>
<td>Preconditioning matrix, Flow circulation</td>
</tr>
<tr>
<td>Λ</td>
<td>Diagonal matrix of eigenvalues</td>
</tr>
<tr>
<td>µ</td>
<td>Advance ratio, Molecular viscosity</td>
</tr>
<tr>
<td>ρ</td>
<td>Density</td>
</tr>
<tr>
<td>Ω</td>
<td>Rotor rotational speed</td>
</tr>
<tr>
<td>σ</td>
<td>Rotor Solidity</td>
</tr>
<tr>
<td>τ_{ij}</td>
<td>Shear stress on the i face, in the j direction</td>
</tr>
<tr>
<td>Ψ</td>
<td>Rotor blade azimuthal position, Roll Euler angle</td>
</tr>
<tr>
<td>Θ</td>
<td>Euler angle for pitch</td>
</tr>
<tr>
<td>Φ</td>
<td>Euler angle for yaw</td>
</tr>
<tr>
<td>\nabla</td>
<td>Gradient operator</td>
</tr>
<tr>
<td>(b)</td>
<td>Boundary value</td>
</tr>
<tr>
<td>(f)</td>
<td>Final State</td>
</tr>
<tr>
<td>(fm)</td>
<td>Pertaining to the flight mechanics reference frame</td>
</tr>
<tr>
<td>(helo)</td>
<td>Pertaining to the helicopter rotor frame</td>
</tr>
<tr>
<td>(i)</td>
<td>Initial state, Index in x direction</td>
</tr>
<tr>
<td>(I)</td>
<td>Pertaining to the Inertial reference frame</td>
</tr>
<tr>
<td>(j)</td>
<td>Index in y direction</td>
</tr>
<tr>
<td>(k)</td>
<td>Index in z direction</td>
</tr>
<tr>
<td>(L)</td>
<td>Left side of face</td>
</tr>
<tr>
<td>(r)</td>
<td>Reference Quantity</td>
</tr>
<tr>
<td>(R)</td>
<td>Right side of face</td>
</tr>
<tr>
<td>^</td>
<td>Quantity normalized for U^2NCLE</td>
</tr>
<tr>
<td>(t)</td>
<td>Total aerodynamic/thermodynamic quantity</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

The ability to transport people and materials is a critical capability in modern society; the military needs to deploy troops and supplies, rescue operations must quickly move patients, and producers need to transport their products. Rotorcraft have carved out a niche in the transportation market by virtue of their vertical take off and landing (VTOL) capability. Of the vehicles available, helicopters are very efficient in hovering and VTOL operations [1], but are limited in forward flight speed due to aerodynamic phenomena occurring on the rotor.

The ongoing Army Joint Multi-Role (JMR) project will lead into the Future Vertical Lift (FVL) program, which aims to replace the fleet of military helicopters which are currently in service with newly designed vehicles [2]. Advanced simulation tools enable more informed design decisions for vehicles such as those in the JMR program. Although helicopters and helicopter design methods go back many decades, improvements are still being made in the field of helicopter performance prediction. The progress is centered on the use of high performance computers for flow and structural analysis.

A rotorcraft engineer’s ability to design a vehicle is dependent on the performance information which is available. It is at this point where advanced computer simulation is at its most valuable in terms of reduced time and money invested. The high value
placed on computer simulation in the role of providing performance information is evident in the great amount of resources applied to this area over the past 40 years. And indeed, the state of the art has evolved during that time from simplified fluid and structural representations [3], to full three dimensional viscous Navier-Stokes solvers on the fluids side and exact finite element solvers on the structural side [4].

1.1 Research Motivation and Objectives

The primary objective of this work is to improve the fidelity of predictive aeroelastic models of helicopter rotors based on tools available at the University of Toledo. This improvement will come through the coupling of a high fidelity RANS CFD solver and a rotorcraft comprehensive code which has historically used simplified aerodynamic models to predict airloads. Code coupling will take the form of the delta airloads loose coupling procedure of Potsdam [4] and an unstaggered tight coupling procedure. To enable the efficient and repeatable execution of the coupling process, the objective heavily involved the development of tools to handle the data exchange process between the two codes. Completed coupling simulations will be analyzed for convergence behavior of the trim settings and CFD-based airloads. The predicted airloads and structural loads will be compared to flight test data where it is available.

1.2 The Physical Problem

Industry and analysis-specific conventions are introduced in this section along with a description of the flow phenomena which are typically present in helicopter rotor flow. The effect of these different phenomena will be seen in the flight test data and simulation results. More information can be found on general analysis considerations in the reference books by Johnson [5] and Leishman [6].
1.2.1 Flow Phenomena

In the cases considered in this work, the effects listed below are present and impact the airload results. Different flight conditions have differing amounts of each effect.

1. Incoming flow velocity: The blades of a hovering rotorcraft see aerodynamic loads which are constant throughout the revolution, but in a forward-moving vehicle, blade kinematics and airloads are periodic functions of the azimuth position. Aerodynamic effects which are exacerbated by the asymmetry ultimately limit the vehicle’s top speed. Transonic flow or reverse flow over the blade can result from the superposition of the local blade rotation speed \((r\Omega)\) and the freestream velocity. All American helicopters have a rotor which rotates counter-clockwise when viewed from above as shown in Figure 1-1. Transonic flow first occurs at the tip of the advancing blade, near \(\Psi = 90\) degrees, and reverse flow occurs on the retreating blade near \(\Psi = 270\) degrees. Figure 1-1 highlights the affected regions. Forward flight cases usually have a minimum blade pitch at 90 degrees and a maximum near 270 degrees. On the retreating blade (270 degrees) the combination of active pitching motion, low relative velocity, and large pitch can lead to dynamic stall events.

2. Blade-Vortex Interaction (BVI): Rotor blades can encounter trailing vortices shed from leading blades under certain flight conditions. Upon approaching a blade, vortices can induce large velocity in the flow and change the local angle of attack seen by the blade. This can lead to impulsive loads on the blade and vehicle vibration under certain flight conditions. Incoming flow from nearly straight-on or from slightly below the rotor encourages BVI. Low and moderate forward flight speeds also contribute to a flow environment which allows BVI.

3. Aeroelasticity: Helicopter rotors are aeroelastic systems because blade deflection is a function of the aerodynamic loading on the blade, and aerodynamic
loading on the blade depends on the blade deflection. The blade vertical translation and pitching which results from elastic deflection can change the local angle of attack or contribute to dynamic stall. Because of the inter-relation, these two effects are coupled and must be considered together.

4. Spinning Effects: Two important effects result from the high rate of rotor spin. Blades feel a center-seeking force as they turn, and the blade inertia which resists this force creates a centrifugal force which tends to stiffen the blade beyond the stiffness it has due to its construction. The second effect is gyroscopic and has the primary outcome of creating a lag in the effect of the applied moments by 90 degrees azimuth.

These phenomena are important and any computational method used for design purposes requires substantial capability to successfully capture all of them. Figure 1-1 also shows a convention of the rotorcraft industry which will be used throughout this work: blade azimuthal position is reported as angle Ψ which is zero when the blade is over the tail boom of the vehicle. This direction coincides with the flow direction in forward flight.

Figure 1-1: Rotor layout showing transonic (T) and reverse flow (R) regions
Figure 1-2 shows the blade deflection conventions and some features of rotor dynamics. The top view shows the in-plane angular motion of the blade, called lead-lag motion and written as $\gamma$. The convention of the lead-lag angle is that it is positive in lag as shown. Accordingly, a positive lead-lag moment on the blade is one which tends to increase this angle. The most significant deflection angle is the flap angle $\beta$ which is the angle measured from the nominal rotor plane, positive above the plane. Positive flapping moments tend to increase the angle $\beta$. The third angle is the pitching angle of the blade denoted by $\alpha$, which is shown in Section A-A. As is typical, positive pitching moments are those which tend to nose the airfoil up (a clockwise rotation according to Figure 1-2).

1.2.2 Flight Mechanics and Trim Conditions

Flight mechanics and the control mechanisms of the helicopter play an important role in the coupling analyses which follow, so a brief description of the way these features work is provided in this section. Figure 1-3 is a right side view of a helicopter and three coordinate systems. For flight mechanics considerations the body-attached frame is typically used, with $x_{fm}$ and $z_{fm}$ representing the frame in this figure.
accepted layout of this frame is to point $x_{fm}$ out the front of the vehicle, $y_{fm}$ out the right side of the body, and $z_{fm}$ out of the bottom of the vehicle. With this orientation and the frame being located at the vehicle’s center of mass, it is ideal for representing vehicle aerodynamic loads as well as linear and angular rates. The traditional helicopter reference frame is also shown, with $x_{helo}$ pointing out the rear of the vehicle, $y_{helo}$ out the left side, and $z_{helo}$ pointing downwards. This setup is consistent with the convention that zero azimuth is the blade pointing to the rear of the vehicle as described in Section 1.2.1. Finally, Figure 1-3 shows a frame called the inertial frame $(x_I, y_I, z_I)$ which is the frame used to represent many important quantities in the numerical analysis. It is allowed to have any orientation but is usually chosen to have the orientation shown. This frame and its use is described in more detail in Chapter 3.

![Figure 1-3: Helicopter profile and some important reference frames for analysis](image)

Some background information on the setup of typical helicopter control systems is required to understand the implementation of the control systems described in later sections. The complex state of a helicopter main rotor is governed by just three
control settings. These are referred to as the collective, longitudinal cyclic, and lateral cyclic control settings.

The collective setting is a control system output which is linked to the overall rotor thrust. This means, for example, that when the control system finds that the thrust is below the target value, it prescribes a change in the collective setting to try and correct it. In an analogous fashion the lateral cyclic control system output is linked to the rolling moment produced by the rotor, and the forward cyclic setting is linked to the pitching moment of the rotor. The prescribed pitch of each blade as a function of azimuth position $\Psi$ is:

$$\text{pitch}(\Psi) = A_0 + A_1 \sin(\Psi) + A_2 \cos(\Psi)$$

(1.1)

where $A_0$ is the collective angle setting, $A_1$ is the longitudinal cyclic setting, and $A_2$ is the lateral cyclic setting. Because of the 90 degree force-displacement phase lag in the rotor, the association of a sine term to longitudinal (i.e. vehicle pitch) control and a cosine term to lateral (i.e. vehicle rolling) control is non-intuitive, but correct. The phase lag means that a high force must be applied at 0 degrees azimuth to rotate the rotor disk up at 90 degrees, for example. It is really the rotor disk orientation which prevails in the rotor load generation. The values of each coefficient is driven by the pilot’s input to the yoke or the vehicle autopilot. Helicopter main rotors typically rotate at a constant angular speed, so changes in the forces and moments produced by the rotor are purely due to changes in these three settings.

Figure 1-4 shows qualitatively the envelope for level flight of the UH-60A vehicle. A very significant feature of the flight envelope is that it is bounded on the “top” by the McHugh Lifting Boundary [7]. This is the boundary between flight conditions which agree with the assumptions of the control system and those that do not. In effect, when the vehicle is above the McHugh boundary gross aerodynamic stall is
occurring in the rotor disk and actions such as increasing collective angle will not lead to an increase in thrust. By simulating flight cases situated close to this line, the ability of the computational tools to predict this significant feature can be assessed. Additionally, the aerodynamic phenomena occurring at flight conditions near the boundary lend themselves to improvement through CFD/CSD coupling because it is in these regions where blade aerodynamics begin to encounter phenomena not captured in the look-up tables, such as dynamic stall.

A more immediate illustration of the McHugh Lifting Boundary and what it means for the helicopter control system is shown in Figure 1-5. For the two high-thrust UH-60A flight counters, the rotor must operate very near the maximum thrust location in order to provide the required thrust. When the rotor is operating in this condition, increases in the collective angle can actually decrease the thrust produced. The assumption of increasing collective leading to increasing thrust is broken, and the control system will diverge as it tries to match the thrust target.

Figure 1-4: Flight envelope for level flight of the UH-60A
1.3 Previous Work

The analysis and prediction of rotorcraft performance has been an area of study for many decades. Problems in this field are multidisciplinary, requiring a working knowledge of controls, aerodynamics, structural and multibody dynamics, and numerical analysis. Simulating any one of the phenomena listed above requires years of work to create an analysis code substantially free from bugs, and gain enough experience with the code to be called proficient at it. It is because of these factors that no single code has been created to accurately handle all effects from first principles. This section describes the progression from high level control volume analysis to today’s advanced computational procedures.

This multi-faceted simulation solution to the rotorcraft analysis problem consti-
tutes the union of two lines of analysis: rotorcraft comprehensive simulation and rotorcraft flow simulation. In the modern loose coupling process the two lines of work have joined, but limited computing resources have historically forced more of a distance between them.

Comprehensive codes are numerous and are named as such because each captures, in one piece of software, the aerodynamic environment of the helicopter as well as the important structural aspect. Part of the functionality of these codes is also the ability to determine trim conditions through a simulated control system. Serious computer-based simulation efforts using comprehensive codes began in the 1960s with a code called C81 at Bell Helicopter. Other notable codes are RODYNE (Sikorsky) [8], CAMRAD II [9], RCAS (U.S. Army) [10], and DYMORE (Georgia Institute of Technology) [11]. The evolution of comprehensive code development and use is presented by Johnson [3].

With a set of reasonably mature comprehensive codes which were effective in simulating dynamics and control system behavior, further improvements in the field correspond to improved accuracy of aerodynamic predictions. Because limits on computational resources prevented the inclusion of first-principles flow solver results over the entire flow domain, coupling efforts began relatively modestly. In 1984, Tung and Carradonna computed transonic effects on an advancing blade tip with a Transonic Small-Disturbance (TSD) code coupled to CAMRAD [12]. Potsdam et al. [4] introduced the delta airloads coupling method for allowing two-way data flow between the RANS code OVERFLOW [13] and CAMRAD II. The high quality of his results led others to investigate this promising approach, including coupled analysis of the UH-60A by Biedron and Lee-Rausch [14] using FUN3D [15, 16] and CAMRAD II. Abras used DYMORE coupled rotor structural models with the aerodynamic loads computed by two different codes [17]. The first code is called Maryland Free Wake (MWF) and is from the University of Maryland. Secondly, Abras coupled DYMORE
and the unstructured government code FUN3D. Rajmohan coupled DYMORE with the
finite volume CFD code GT-Hybrid to study four different steady flight cases with
loose coupling, and also used tight coupling with account made for vehicle flight me-
chanics to simulate a maneuver of the UH-60A [18]. The inclusion of flight mechanics
in the model means that the simulation includes high fidelity structural response and
first-principles aerodynamics along with changing inflow direction associated with the
maneuver. The Bell 427 main rotor has been analyzed using loose coupling by Morillo
[19].

Using a CFD solver is not without its challenges when it comes to simulating
rotorcraft flow features, the wake structure shed from the blade tips being a critical
one. Potsdam and Strawn described the issue of under-resolved wake [20], noting the
deleterious effects it has on rotor performance prediction. Wake models have been
developed to overcome the artificial dissipation that comes with representing vortices
solely within the CFD grid. Rigid wake [21] and free wake [22, 23] models allow the
wake feature to be retained in the flowfield for as long as the user chooses. The rigid
kind is quicker to run but does not account for the self-distorting effect the wake has
on itself.

Important work has been done to provide test data which serves as the standard
for gauging the accuracy of predictive tools. In the U.S., the UH-60A “Blackhawk”
helicopter was instrumented and tested as a part of the UH-60A Airloads Program,
a collaborative effort between the Army and NASA from 1984 to 1994 [24, 25]. The
Higher-harmonic Aeroacoustics Rotor Test (HART) [26, 27] is another data set which
has been used to compare to the predictions of CFD/CSD coupling. An early aca-
demic CFD/CSD coupling work by Hill [28] used the HART II data set.
1.4 Layout of this Document

This thesis describes the development and verification of a loose coupling and a tight coupling procedure at the University of Toledo. In this first chapter, pertinent flow physics and the motivation for this work has been described. A history of helicopter rotor analysis is given to provide context for this work. The second chapter describes the tools which were used, and goes into some detail about the computational methods used by U²NCLE, DYMORE, and the Fluid-Structure Interface (FSI). Chapter 3 describes the model setup requirements which must be satisfied in order to couple the CFD and CSD codes, and describes how one can go about setting up coupling-compatible models. Chapter 4 contains model setup details and results for the configurations analyzed: loose coupling of the UH-60A main rotor, and both loose coupling and tight coupling analysis of the Bell 427 main rotor. Where available, the results are compared with flight test data. In the final chapter, conclusions drawn from the analysis are presented and discussed. Finally, some recommendations for future work are presented.
Chapter 2

Computational Methods

This chapter describes the methods which have been developed for the coupled fluid-structure analysis of rotorcraft. First, the individual equation-solving codes and the critical functionalities of each are described. Following this is a description of the two kinds of coupling procedures which were implemented to study the two kinds of problems of interest: periodic and non-periodic problems. Finally, the Fluid-Structure Interface which was developed for the purpose of automating the CFD/CSD coupling process is detailed.

2.1 The U^2NCLE Flow Solver

U^2NCLE is an unstructured grid Reynolds-Averaged Navier-Stokes (RANS) solver using the finite volume formulation currently developed and maintained at the University of Toledo [29]. It includes arbitrary Mach number preconditioning, as well as a free wake model and the Spalart-Allmaras, 2 equation k-epsilon turbulence models. The code has the capability to deform the computational grid during run time, and includes periodic boundary conditions and a wake model which make it well suited to perform efficient helicopter flow simulations. The code is written in Fortran [30] and C [31], and utilizes the Message-Passing Interface (MPI) to facilitate communication among distributed-memory computer nodes [32]. The detailed workings of the solver
and more are described in this section, beginning with the governing equations.

### 2.1.1 Governing Equations

The governing equations used by U$^2$NCLE are the Navier-Stokes equations in integral form which express the conservation of mass, momentum in three directions, and energy. These relations are shown below.

\[
\frac{\partial}{\partial t} \int_V \rho \, dV + \oint_S \rho \vec{V} \cdot \vec{n} \, dA = 0 \tag{2.1}
\]

\[
\frac{\partial}{\partial t} \int_V \rho \vec{V} \, dV + \oint_S (\rho \vec{V} \vec{V} + pI - \tau) \cdot \vec{n} \, dA = 0 \tag{2.2}
\]

\[
\frac{\partial}{\partial t} \int_V \rho E \, dV + \oint_S (\rho \vec{V} H - k \nabla T - \tau \cdot \vec{V}) \cdot \vec{n} \, dA = 0 \tag{2.3}
\]

Above, $\rho$ is density, $\vec{V}$ is the velocity vector, $p$ is pressure, temperature is $T$, stagnation enthalpy is $H$, the 3 x 3 identity matrix is $I$, total energy is $E$, fluid thermal conductivity is $k$, the shear stress tensor on the $i$ face in the $j$ direction is $\tau_{ij}$, $S$ represents the surface enclosing a control volume $V$, $\vec{n}$ is the unit vector normal to a surface, and time is $t$. Many of these values are vectors or matrices and are enumerated in the following equations.

\[
\vec{V} = \begin{pmatrix} u \\ v \\ w \end{pmatrix} \tag{2.4}
\]
\[ \tau = \begin{pmatrix} \tau_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \tau_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \tau_{zz} \end{pmatrix} \] (2.5)

\[ \tau_{xx} = \mu \frac{2}{3} \left( 2 \frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} - \frac{\partial w}{\partial z} \right) \] (2.6)

\[ \tau_{yy} = \mu \frac{2}{3} \left( 2 \frac{\partial v}{\partial y} - \frac{\partial u}{\partial x} - \frac{\partial w}{\partial z} \right) \] (2.7)

\[ \tau_{zz} = \mu \frac{2}{3} \left( 2 \frac{\partial w}{\partial z} - \frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} \right) \] (2.8)

\[ \tau_{xy} = \tau_{yx} = \mu \left( \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) \] (2.9)

\[ \tau_{xz} = \tau_{zx} = \mu \left( \frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right) \] (2.10)

\[ \tau_{yz} = \tau_{zy} = \mu \left( \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \right) \] (2.11)

\[ H = h + \frac{|\vec{V}|^2}{2} \] (2.12)

\[ E = e + \frac{|\vec{V}|^2}{2} \] (2.13)

\[ h = c_p T = \frac{\gamma R}{\gamma - 1} T \] (2.14)
\[ e = c_v T = \frac{R}{\gamma - 1} T \] (2.15)

Above, energy and enthalpy per unit mass are \( e \) and \( h \), respectively. Specific heat at constant pressure is \( c_p \), specific heat at constant volume is \( c_v \), the gas constant is \( R \), the ratio of specific heats is \( \gamma = c_p/c_v \), \( \mu \) is molecular viscosity, and \( x \), \( y \), and \( z \) denote directions in the Cartesian coordinate system. Velocity components in the \( x \), \( y \), and \( z \) directions are \( u \), \( v \), and \( w \).

The variables which the code actually uses in the solution procedure are normalized in order to reduce the accuracy lost due to including very large values and very small ones in the same calculation. In the U^2NCLE code, the method by which the variables in the governing equations are normalized as shown in Table 2.1. Typical values chosen for normalizing rotorcraft flows are also listed.

### Table 2.1: U^2NCLE internal variable normalization

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Variable</th>
<th>Normalization</th>
<th>Typical Reference Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length</td>
<td>L</td>
<td>( \bar{L} = \frac{L}{L_r} )</td>
<td>( L_r ) = Rotor Radius</td>
</tr>
<tr>
<td>Density</td>
<td>( \rho )</td>
<td>( \bar{\rho} = \frac{\rho}{\rho_r} )</td>
<td>( \rho_r ) = Nominal Density</td>
</tr>
<tr>
<td>Velocity</td>
<td>( u )</td>
<td>( \bar{u} = \frac{u}{u_r} )</td>
<td>( u_r ) = Incoming Flow</td>
</tr>
<tr>
<td>Temperature</td>
<td>T</td>
<td>( \bar{T} = \frac{T}{T_r} )</td>
<td>( T_r ) = Nominal Temperature</td>
</tr>
<tr>
<td>Time</td>
<td>t</td>
<td>( \bar{t} = \frac{t}{t_r} )</td>
<td>—</td>
</tr>
<tr>
<td>Pressure</td>
<td>p</td>
<td>( \bar{p} = \frac{p}{\rho_r u_r^2} )</td>
<td>—</td>
</tr>
<tr>
<td>Energy</td>
<td>e</td>
<td>( \bar{e} = \frac{e}{\rho_r u_r^2} )</td>
<td>—</td>
</tr>
<tr>
<td>Enthalpy</td>
<td>h</td>
<td>( \bar{h} = \frac{h}{c_p T_r} )</td>
<td>—</td>
</tr>
</tbody>
</table>
Finally, in this discussion of the governing equations, we turn attention to identifying the relationships which can be used to close out the system of equations. The relations of ideal gas dynamics are used for this purpose, and are shown below after having been normalized according to the table above.

\[ p = \frac{\rho a^2}{\gamma} \]  
\[ a^2 = \frac{T}{M_r^2} \]  
\[ E = M_r^2 \left( \frac{a^2}{\gamma} + \frac{\gamma - 1}{2} V^2 \right) \]  
\[ H = M_r^2 \left( a^2 + \frac{\gamma - 1}{2} V^2 \right) = E + \frac{E_c p}{\gamma} \]

In the above relations the Eckert number is \( E_c \), \( \gamma = c_p/c_v \) is the ratio of specific heats, E is the total energy, H the total enthalpy, V is the vector sum of velocity, \( M_r \) is the reference Mach number, and the speed of sound is written as a.

### 2.1.2 Discretization and Solution Procedure

As is the typical purview of CFD, the governing equations are discretized and applied to a grid in order to be solved numerically, since no closed-form solution can be found. U^2NCLE uses a finite volume method which means the discretization takes the form of splitting the bulk computational domain volume up into many smaller volumes, with mass, energy, and momentum conservation holding true for each. There are two common methods for determining the configuration of the many little control volumes, or *cells*, from a set of points as are output from a pre-processing software. One way is the node-centered scheme, which derives cell boundaries from the set of
faces normal to lines drawn between immediately adjacent node points. The second way is the cell-centered scheme, which constructs the discrete volume partitions based on lines drawn between adjacent node points. This section describes the development of the equations being solved by U^2NCLE.

**Preconditioning**

Preconditioning is achieved by multiplying by the preconditioning matrix described by Sheng [29], shown below.

\[
\Gamma_q^{-1} = \begin{pmatrix}
1 & & & \\
1 & 1 & & \\
& 1 & 1 & \\
& & & \frac{1}{\beta}
\end{pmatrix}
\]  \hspace{1cm} (2.20)

In this matrix, \( \beta \) is equal to the minimum of 1 or \( M_r^2 + M_\Omega^2 \), where \( M_r \) is the reference Mach number and \( M_\Omega \) is the rotating Mach number. The effect of using this preconditioning is improved numerical behavior in domains with large ranges of Mach numbers, as compared to non-preconditioned equations.

**Convective and Diffusive Fluxes**

A method attributable to Briley [33], which is an extension of Roe’s scheme [34], is used to compute convective flux across cell faces. The formulation considers a control volume surface \( j \), with flow properties on the left and right side of it denoted by subscript L and subscript R, respectively. Flux of the flow properties across surface \( j \) is written as:

\[
\hat{P}_r \cdot \hat{n} = \frac{1}{2} (F(q_L) + F(q_R)) - \frac{1}{2} \hat{M}_\Gamma^{-1} |k_\Gamma| (q_R - q_L)
\]  \hspace{1cm} (2.21)
where $q_R$ and $q_L$ represent the primitive variables on the right and left sides of the interface, respectively. $\hat{k}_T$ is the preconditioning matrix.

$U^2$NCLE can carry out either a first order accurate scheme or a second order accurate scheme. The first order scheme uses values of $q_R$ and $q_L$ which are simply the flow quantities in the cells on the right and left sides of the interface. The second order accurate scheme is implemented by using a Taylor series about the node at the center of the control volume according to the relation:

$$ q_j = q_i + \nabla q_i \cdot \Delta r $$

Above, $r$ is a position vector starting at the center node $i$ and ending at the midpoint of the cell edges. $\Delta q_i$ is the matrix of primitive variable derivatives at the central node, which is evaluated using an un-weighted least-squares procedure [35]. Zhao [36] implemented a higher-order scheme in the $U^2$NCLE solver.

In addition to the convective flux, the Navier-Stokes equations contain a provision for flux contributions from diffusion effects, written as:

$$ P_v \cdot n = -\frac{1}{Re} \cdot \nabla g $$

Here $g$ is a generalized viscosity term, $Re$ is the Reynolds number, and $P_v$ is the viscous flux.

**Time Integration**

$U^2$NCLE uses an implicit time marching scheme which allows large CFL numbers and two different orders of accuracy. The general formulation for the time marching relation is written as:

$$ S_i^{n+1} = M \Gamma^{-1} \frac{\Delta q_i^n - \frac{\theta}{1+\theta} \Delta q_i^{n-1}}{\Delta t} + \frac{1}{\Delta V_i} \sum_{j=1}^{n} (\hat{P} \cdot \vec{n})_j^{n+1} \Delta A_j $$

(2.24)
Above, $\Delta t$ is the time increment between the current and next time steps. Changes in the primitive variables between the previous time step (n-1), current step (n), and next step (n+1), are captured in $\Delta q_{n+1}^i = q_{n+1}^i - q_n^i$ and $\Delta q_{n-1}^i = q_n^i - q_{n-1}^i$. $\hat{P}$ is the contribution of both viscous and inviscid fluxes. The index of the current control volume is i, and j is the index of the control surface which bounds the volume.

One result of using this time scheme is that $U^2$NCLE is able to take larger time steps than explicit codes. So, while the run time for a single step is longer than for a structured explicit code, fewer time steps are needed overall. Typically, time steps are set to correspond to 1 degree azimuth of blade rotation.

**Boundary Conditions**

Four kinds of boundary conditions are able to be applied in $U^2$NCLE: farfield, inflow (based on mass flow), outflow (based on back pressure), and impermeable wall. Versions of these conditions consistent with the arbitrary Mach number preconditioning method of $U^2$NCLE have been developed by Sheng and Wang [37].

The equation governing the flow at the various boundary types is:

$$\frac{\partial W}{\partial t} + \Lambda_{\Gamma_x} \frac{\partial W}{\partial x} + \Lambda_{\Gamma_y} \frac{\partial W}{\partial y} + \Lambda_{\Gamma_z} \frac{\partial W}{\partial z} = R^{-1} \Gamma_q M^{-1} S \quad (2.25)$$

where $W = R_0^{-1} q$. $R_0^{-1}$ is a matrix with the left eigenvectors of the preconditioning matrix $k_\Gamma$. Subscript 0 indicates that the matrix is evaluated at a reference condition. Eigenvalues of $k_\Gamma$ are contained in the diagonal matrices $\Lambda_{\Gamma_x}$, $\Lambda_{\Gamma_y}$, and $\Lambda_{\Gamma_z}$. One lambda matrix corresponds to wave propagation in each of the three directions.

$W$ is a vector of preconditioned system characteristic variables, written as shown below. Based on the wave propagation direction, points interior or exterior to the domain can be used in computing these values.
\[ w^1 = \left( \rho - \frac{p}{\beta c_0^2} \right)n_x - (\Theta - n_t)\psi_0 n_x - wn_y + vn_z \]  
(2.26)

\[ w^2 = \left( \rho - \frac{p}{\beta c_0^2} \right)n_y - (\Theta - n_t)\psi_0 n_y - un_z + wn_x \]  
(2.27)

\[ w^3 = \left( \rho - \frac{p}{\beta c_0^2} \right)n_z - (\Theta - n_t)\psi_0 n_z - vn_x + un_y \]  
(2.28)

\[ w^4 = \frac{p}{2\rho_0 \sigma_0} + (\Theta - n_t)c_0^+ \]  
(2.29)

\[ w^5 = \frac{p}{2\rho_0 \sigma_0} + (\Theta - n_t)c_0^- \]  
(2.30)

The characteristic far field boundary condition is intended to allow waves to enter and freely leave the flow domain. The eigenvalues at the boundary are used to determine whether a wave is entering or leaving; a negative eigenvalue means the associated characteristic wave is propagating into the flow domain from the far field because the surface normal vectors point outward from the outer surface of the domain. Likewise, a positive eigenvalue signifies that a wave is leaving the flow domain. The equation along a characteristic line:

\[ \frac{dl}{dt} = \lambda_\Gamma \]  
(2.31)

where \( dl \) is a small distance in a specific direction and \( \lambda_\Gamma \) is the eigenvalue in the direction of \( dl \). These variables can be further broken down as:

\[ dl = n_x dx + n_y dy + n_z dz \]  
(2.32)
\[ \lambda_r = n_x \lambda_{rx} + n_y \lambda_{ry} + n_z \lambda_{rz} \] (2.33)

Assigning characteristic inflow to a boundary

\[ \rho_b = \frac{\gamma M_r^2 p_t}{T_t} \left( 1 - \frac{\gamma - 1}{2 T_t} M_r^2 U_b^2 \right) \frac{1}{\gamma - 1} \] (2.34)

\[ p_b = p_t \left( 1 - \frac{\gamma - 1}{2 T_t} M_r^2 U_b^2 \right) \frac{\gamma}{\gamma - 1} \] (2.35)

\[ u_b = U_b \cos \phi_x \] (2.36)

\[ v_b = U_b \cos \phi_y \] (2.37)

\[ w_b = U_b \cos \phi_z \] (2.38)

where \( U_b \) is the boundary velocity, \( p_b \) pressure at the boundary, \( \rho_b \) the boundary density. At an inflow where the mass flow rate is specified, total temperature \( T_t \), mass flow rate \( \dot{m} \), and inflow angles \( \phi_x \), \( \phi_y \), and \( \phi_z \) are known, with the following relations applying between them:

\[ \dot{m} = \rho_b U_b \] (2.39)

\[ T_t = T_b + \frac{\gamma - 1}{2} M_r^2 U_b^2 \] (2.40)

\[ p_b = \frac{\rho_b T_b}{\gamma M_r^2} \] (2.41)
Characteristic variable outflow conditions can be set, with the first four eigenvalues being positive and the fifth one negative. The pressure at the boundary is specified, $p_b = p_{exit}$.

The impermeable wall boundary condition implements a zero flow-through condition at a surface, meaning that $\Theta_b = 0$:

$$u_b n_x + v_b n_y + w_b n_z = -n_t$$

(2.42)

**Turbulence Modeling**

To close the system of RANS equations, several turbulence models are implemented in U$^2$NCLE, including Spalart-Allmaras, $k - \omega$, and $k - \epsilon$ models [38, 39, 40]. This work used only the Spalart-Allmaras model, thus it is the only one described here. An equation for turbulence viscosity transport can be written:

\[
\frac{\partial \tilde{\nu}}{\partial t} + \vec{U} \cdot \nabla \tilde{\nu} = c_b (f_{v1} - ft2) \tilde{S} \tilde{v} - \frac{1}{Re} (c_{w1} f_w - \frac{c_b}{k^2} f_{t2}) (\frac{\tilde{v}}{d})^2
\]

\[
+ \frac{1}{\sigma Re} \{ \nabla \cdot [\tilde{v} + (1 + c_{b2})\tilde{v}] \nabla \tilde{v} ] - c_{b2} \tilde{v} \nabla \cdot (\nabla \tilde{v}) \} = 0
\]

Above, the first two terms on the right hand side are the production terms, and the third term accounts for turbulent dissipation. The coefficients in this model are numerous and the development of these in the U$^2$NCLE flow solver is described by Zhao [36].

### 2.1.3 Free Wake Model

Rajmohan [18] describes a free wake model, a version of which is implemented in the U$^2$NCLE code [41]. This model tracks the motion of vortices shed from the rotor blades as they turn and accounts for the induced flow velocities from each element of
the discretized wake structure. To accomplish this, the model uses the lift computed by the Navier-Stokes solver to determine circulation as a function of blade radius according to the Kutta-Juokowski Theorem:

\[ L' = \rho U \Gamma \]  

(2.43)

where \( \Gamma \) is the circulation in the flow at a given radial location, \( L' \) is the lift per unit span, \( \rho \) is density, and \( U \) is the freestream flow velocity. The code uses changes in the circulation to determine the strength of the shed vortex, with the circulation centroid of each section of blade serving as the location from which the trailer is shed. Once free from the blade, the vortex will convect according to the combination of freestream velocity and induced velocity from other elements of the free wake structure. A box enclosing the CFD domain is defined as input to the code; the wake model elements outside the box are included in the induced velocity computations in the CFD domain, whereas wake model elements inside the box are not included. Instead, these are included in the Navier-Stokes solution through induced velocity on the leading periodic surface. The purpose of excluding wake model elements which are inside the bounding box is to avoid including the wake effects twice.

Trailing vortices are salient flow for some time after they are shed from the blade, and simulation efforts typically try to retain them for up to 4 or 5 rotor revolutions. CFD has a tendency to be more dissipative than nature, and as a result the trailing vortices can dissipate “early,” meaning that the CFD code has lost an important flow feature. Numerical models of wake behavior have been developed which improve upon the ability of CFD to retain trailing vortices in certain cases. These wake models allow a single-blade analysis to be performed instead of a full rotor analysis in the case of a periodic flight condition. The wake model keeps track of vortex elements which are shed from the blade. The user can choose to use only a single or multiple trailers; a
single trailer is shed from the circulation centroid location, whereas multiple trailers are spread evenly along the length of the blade.

Figure 2-1 illustrates the objects represented in the wake model. Wake trailers leave the blade parallel to the chord direction, while shed vortices come off parallel to the blade radius direction. Each of these carries with them a certain amount of vorticity.

![Figure 2-1: Features of the free wake model](image)

2.1.4 Grid Deformation

$U^2$NCLE has the ability to accept and implement deformation data from an external source. The deformation data refers to the airstation locations, and quantifies linear and angular deformations of the airstations from their initial undeformed configuration. Linear deformations are not normalized when exchanged with a CFD code,
and the angular deformations are presented as 3-2-1 Euler angles. Equation 2.44 shows the relation between a vector in two different reference frames which relate to each other through a 3-2-1 Euler angle rotation [42]. In this equation C and S are short for cosine and sine, respectively. The 3-2-1 Euler angles are a triplet (Φ, Θ, Ψ). Variables x, y, and z are the coordinates of the initial surface point in the local coordinate system. The new location of the surface point is given by the variables x’, y’, and z’. The two are related through the matrix equation:

\[
\begin{pmatrix}
  x \\
  y \\
  z
\end{pmatrix}
= 
\begin{pmatrix}
  C_\phi C_\psi & S_\phi S_\psi C_\psi - C_\phi S_\psi & C_\phi S_\psi C_\psi + S_\phi S_\psi \\
  C_\phi S_\psi & S_\phi S_\psi C_\psi + C_\phi C_\psi & C_\phi S_\psi S_\psi - S_\phi C_\psi \\
  -S_\phi & S_\phi C_\phi & C_\phi C_\phi
\end{pmatrix}
\begin{pmatrix}
  x' \\
  y' \\
  z'
\end{pmatrix}
\] (2.44)

According to this method of orientation, an initial axis system C will be rotated about the z axis to end up with the coordinate system \(C' = \{x' y' z'\}\). Then \(C'\) is rotated about the \(y'\) axis to end up with \(C'' = \{x'' y'' z''\}\). Finally, \(C''\) is rotated about \(x''\) by the specified amount to end up with the coordinate system being specified by the angular deflection output.

The proper airstation orientation is achieved by representing each local airstation frame in a frame that we will refer to as the blade frame. The blade frame rotates with the hub, but does not otherwise change orientation. Each local airstation frame serves as the frame of resolution for the position of points on the surface of the blade, which is always the same because the code does not account for deformation of the blade shape. Now, when \(U^2\text{NCLE}\) is given a set of angles from DYMORE, the surface point positions are implemented in the computational grid according to the blade frame, but the blade frame coordinates are given by taking the local airstation point position vector and transforming it according to the transformation tensor.

The user must provide a “support radius” input parameter to \(U^2\text{NCLE}\), and any
nodes nearer to the deformed body than this radius will be included in the deformation and those outside will be undeformed. An example of the grid which results from one such deformation is shown in Figure 2-2. Abras [17] noted that abrupt changes in adjacent cell size can reduce the order of accuracy of the computational method, and to rectify this an interpolation was used. Support radius values are expressed in terms of multiples of the characteristic length. Values from 20 to 50 are typical; smaller values are generally detrimental to code robustness, as they do not allow enough space for the deformation to spread throughout the grid surrounding the deformed surface.

The user must also provide two other inputs to the deformation routine of U2NCLE: Greedy frequency and number of Greedy points. The number input for Greedy frequency is the number of steps the code will take between a re-choosing of the surface points (also referred to as Greedy points). This frequency can influence the robustness of the code. Typical values range from 15 to 120; too small or large of values will cause the code to fail in the re-choosing of surface points. Periodic changes in this value has been found to improve code robustness. The number of Greedy points will determine how finely the deformation routine resolves the surface which is being deformed. This parameter is usually set to 3 to 4 times the number of airstations present in the CFD domain, and has also been found to impact code robustness.
2.2 The DYMORE Structural Solver

DYMORE is the name of a multibody dynamics analysis code which was developed under the guidance of Dr. Olivier Bauchau at the Georgia Institute of Technology [11]. It was developed throughout the 1990s as an alternative to other comprehensive codes mentioned in Section 1.3 which may include assumptions of small deflections or have other restrictions to their general application [3]. This work uses DYMORE 2, which is consistent with the performance described in the User’s Manual released in 2006 [11], although there are more current versions of the code. The method these codes use for solving the system of equations which results from the system definition is the energy-decaying scheme [43, 44] which is unconditionally stable in part due to numerical dissipation of high frequency vibration modes. These modes
can become excited from nonlinearities in the system, and this excitation can prevent convergence, as noted by Abras [17]. This section describes features of DYMORE structural dynamics, aerodynamics, control system simulator, and virtual sensors. The description of DYMORE in this section follows that of Abras [17].

2.2.1 Multibody Dynamics Features

Models are constructed in DYMORE 2 based on first defining a hybrid topology/geometry configuration (a set of points and curves) which describe the layout of the body in physical space and connections between elements within the body. In addition, structural and aerodynamic properties are assigned to the different curves, and the different joints are specified as connecting bodies at the various points. In order to acquire and use certain model data, it is usual to specify which joints have a displacement or rotation which is prescribed by a time function.

2.2.2 Aerodynamic Models

DYMORE has its own built-in aerodynamics model which runs rather quickly but is not based directly on physical principles. Rather, the simplified aerodynamics model uses look-up tables for the lift, drag, and moment coefficients at different angles of attack and Mach numbers which are input by the user. The code discretizes the blade aerodynamically by splitting it up into airstations, each of which can have a unique built-in angle (i.e. built-in blade twist) and necessarily has a unique radial position. For each airstation, DYMORE evaluates the local velocity magnitude and direction based on rotor rotation, incoming flow, and blade flapping/pitching rates. The local angle of attack is used to extract the lifting, drag, and moment coefficients from the look-up tables, and the forces and moments are applied to each airstation to compute the kinematics for the subsequent time step. The airloads from this
simplified model are referred to as the lifting line airloads throughout this document.

The inputs to DYMORE are the flight parameters (air speed, etc). When the loose coupling method is specified in the DYMORE input file, DYMORE treats the input values as delta airloads, meaning that the sum of the input loads and the internally-computed lifting line loads are applied to the structure. In contrast, when the coupling method specified is tight coupling, DYMORE treats the input values as the airloads; no lifting line loads are computed internally, and the input load values are the only airloads applied to the structure.

2.2.3 Autopilot Feature

In addition to its structural solver functionality, DYMORE has built-in another critical function that enables loose coupling: an autopilot control system. To use this system, the user specifies a set of target values ($\hat{y}$) which are typically the thrust, roll moment, and pitch moment generated by the rotor. A virtual sensor associated with the rotor must be designated to determine the measured values ($y_f$) of each of these parameters. The user must link each measured value with a unique control output by way of a control system definition. The three control outputs are included in the vector $u$. As the simulation marches along in time, the control system module compares the measured parameters against the target values, and adjusts the outputs according to:

$$u_f = u_i + \Delta t J^{-1} G \left( \hat{y} - y_f \right)$$

(2.45)

where $(\cdot)_i$ and $(\cdot)_f$ denote conditions taken at the beginning and end of a time step, respectively. $G$ is the user-specified gain matrix which impacts the responsiveness and stability of the autopilot routine, and $\Delta t$ is the time step. The inverse Jacobian matrix $J^{-1}$ is the derivative of the inputs with respect to the outputs and can be
user-specified if known, or it can be numerically figured by a perturbation method in
DYMORE.

The most typical setup is for thrust, roll moment, and pitch moment of the rotor
to be linked to collective, longitudinal, and lateral cyclic control system outputs,
respectively. Whereas the control system is nominally trying to converge to steady
values, any of the control system input signals could have periodic fluctuations in
them and thus cause unsteadiness in the output. Periodicity in these signals can
be removed by running them through a discrete Fourier decomposition function in
DYMORE. The outputs of this module are the magnitudes of the signal components
at different frequencies so the steady-state part of the signal is fed into the control
system and a steady output signal which meets the target value can be determined
from it.

The user inputs a time history for the control signals by default. For the control
system to work, there must be at least three user-input values for each control system
setting. When the control system is commented out of the input file or otherwise
turned off, the control signals revert to these user-specified time histories. With the
control system on, the user inputs will be used for the initial setting value but the
control system will override the user input during the simulation. The initial control
setting has a substantial impact on the control system convergence behavior in certain
situations.

2.2.4 Virtual Sensors and Post-Processing

For outputting results, DYMORE includes a battery of virtual sensors for struc-
tural members as well as aerodynamic aspects of the model. Some examples of typical
sensor data are:

1. Airloads at an airstation, in the local frame
2. Integrated aerodynamic loads on a rotor object

3. Deflection at a position along a beam member

4. Forces and moments in a structural member

5. Rotational position of a joint

The user is able to create any number of sensors to output wide varieties of data based on the needs of the simulation being performed. In addition to the sensors, DYMORE can perform data post-processing to meet analysis needs. After the simulation has run, DYMORE will parse the archive file (.rcv) to extract and then process data according to the post-processing directives given in an input file. The user can even direct DYMORE to perform just the post-processing task by specifying it in the 

@PROCESS_CONTROL_DEFINTION.

2.3 The Fluid-Structure Interface

A Fluid-Structure Interface (FSI) code developed as a part of this work automatically performs coupling data exchange tasks between the CFD and CSD codes. Separate loose coupling and tight coupling modules are contained within the code. One outcome of the FSI code is that it reduces analyst labor while reliably carrying out either coupling process. While the core of the FSI is the main loose and tight coupling codes, there are a number of tools associated with the FSI which can be used to manually perform the steps of the loose coupling process or perform other data handling tasks. The FSI is described in detail along with the procedures it executes in this section. Chapter 3 has step-by-step details of the FSI operation.
2.3.1 Design Requirements

Table 2.2 enumerates the requirements which served as inputs to the FSI design process. The leftmost column lists a high level, intuitive requirement which requires interpretation to be implemented. The next column provides a more specific subrequirement which contributes to the goal of the higher level one, but leaves little or no room for interpretation. Column 3 reports the means by which the sub-requirement was implemented in the FSI tool, and the method of verification is listed in the rightmost column.
<table>
<thead>
<tr>
<th>No.</th>
<th>High-Level Requirement</th>
<th>Sub-Requirement</th>
<th>Implementation</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Executes Loose Coupling Process</td>
<td>Executes loose coupling process flow per procedure in section 2.3.3</td>
<td>Function LC()</td>
<td>Performed correctly during test runs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Computes Delta Airloads per Eqn. 2.22</td>
<td>Function computeDeltaAirloads()</td>
<td>Hand calculations at randomly chosen locations agree with code output</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Creates $U^2\text{NCLE}$ Deflection files from DYMORE output</td>
<td>Function generateDeflectionFile()</td>
<td>Number-by-number comparison shows agreement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Compiles coupling process convergence metrics</td>
<td>Function compileConvergenceMetrics()</td>
<td>Verified during test runs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Leaves traceable data trail</td>
<td>File structure separates iterations and code runs</td>
<td>Performed correctly during test runs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Detects code failure and restarts</td>
<td>Function powerThruU2ncl() performs according to section 2.3.6.1</td>
<td>Verified during test runs</td>
</tr>
</tbody>
</table>
Table 2.3: FSI Requirements, continued

<table>
<thead>
<tr>
<th>No.</th>
<th>High-Level Requirement</th>
<th>Sub-Requirement</th>
<th>Implementation</th>
<th>Verification</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Executes Tight Coupling Process</td>
<td>Executes tight coupling process flow per section 2.3.5</td>
<td>Function fsi_4p2.c</td>
<td>Verified during test runs</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Program is safe to run</td>
<td>FSI never goes above the directory level in which it was invoked</td>
<td>No directory changes occur during tight coupling process</td>
<td>Verified during test runs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Inadvertent file or directory removal does not occur</td>
<td>File or directory names are included in every “remove” command</td>
<td>Code check performed</td>
</tr>
</tbody>
</table>

Table 2.4: Design Requirements of the Fluid-Structure Interface


The separation of the FSI into loose and tight coupling modules is unavoidable given the different requirements of each procedure. Details of each procedure and a description of the implementation are given below.

2.3.2 Loose Coupling Procedure

Loose coupling for rotorcraft is a method for computing structural response and aerodynamic loads based on a periodic data exchange between two codes. A control system can be involved in the process to adjust control system inputs such that the trim targets are met. The end goal of the analysis is to determine the rotor performance while using the higher fidelity CFD airloads in place of the simplified airloads native to the comprehensive code.

The most straightforward method of code coupling is the direct exchange of the codes. This approach, while valid, will result in large computation times for the problem at hand because the CSD solution requires a few tens of rotor revolutions to damp out spurious excitations associated with model start-up. It is desirable to leave the CFD out of this part and let the faster aerodynamic computations of the CSD code be used to determine the proper trim settings. The march-together method is referred to as tight coupling and has benefits over, but is not as efficient for steady computations as the method described here, called loose coupling.

In steady forward flight, helicopter blade deflections and airloads are periodic and vary only with blade azimuth position. So, loose coupling presupposes airloads and deflections which can be fully described by giving values for each airstation at each azimuth position over one revolution. If the straight airloads are input to DYMORE, DYMORE will not be able to use its control system to perform the required task of meeting the trim targets. In order to allow the control system to play any part in this analysis, lifting line airloads will have to be used somewhere in the process. Potsdam [4] devised the solution to this problem in the form of delta airloads coupling. Instead
of having the CSD code read in the pure CFD airloads, it will read in the difference between the CFD airloads and CSD airloads according to Equation 2.46.

\[
\Delta A^i = \begin{cases} 
A_{CFD}^{i-1} - A_{CSD}^{i-1} & \text{if } i = 1 \\
\Delta A^{i-1} + (A_{CFD}^{i-1} - A_{CSD}^{i-1}) & \text{if } i > 1 
\end{cases}
\]  

(2.46)

In this equation, \(\Delta A^n\) denotes the delta airloads, \(i\) or \(i-1\) indicates the set from the current or previous coupling iteration, respectively. \(A_{CSD}\) is the set of airloads (dimensional forces and moments) for all airstations at all azimuth locations which the CSD code applies to the structure. \(A_{CFD}\) is the set of dimensionalized airloads computed by the CFD code. The sources of these different data sets is important to understand in order to carry out the process. The CSD code outputs an airload file over a user-specified interval; it is important to note that this output is not the lifting line loads, but rather is the set of loads that gets applied to the structure. This means that in cases when DYMORE reads in and applies the delta airloads to the structure, the airload output will be the lifting line loads plus the delta airloads. During the first step of the analysis there is no delta airload data, so the loads applied to the structure are just the lifting line airloads. This explains why the delta airloads calculation is different for iteration 1. A schematic of the loose coupling process as implemented in the FSI is shown in Figure 2-3.
Figure 2-3: The Loose Coupling Process

First in the loose coupling process is a run of the CSD code using its own internal aerodynamics. The deflections determined are then passed to the CFD code, which implements the deflections and computes high fidelity airloads. Using the airloads from both the CFD code and CSD code, the delta airloads calculation for iteration 0 is performed to create a delta airloads file, which serves as input to the next run of the CSD code. This inclusion of the delta airloads is the way that the CFD airloads are included in the CSD computations. After the “boot strapping” just described, the process proceeds until the process is converged as measured by the CSD-computed trim settings.
Figure 2-4 shows the data exchange between the structural and fluids models during the loose coupling process. The series of coordinate systems represent the matching airstation locations and orientations in the two models. Deflections from the CSD model are functions of the delta airloads, and the delta airloads are functions of the deflections. This further illustrates the coupled nature of the problem and the fact that the aim of the process is to find a consistent set of deflections and airloads.

As noted by Potsdam [4], the aerodynamic results of the CSD code are important only to the extent that they provide the control system with correct trends (i.e. increasing collective corresponds to an increase in thrust) because at the end of the coupling process, the airloads on the rotor are the CFD airloads only. This fact can be exploited to improve process robustness, as will be described in Section 4.1.5.
2.3.3 Tight Coupling Procedure

Tight coupling is a code coupling method whereby data is exchanged between two specialized codes at every time step. The difference in data exchange interval means that non-periodic behavior which is impossible to capture in loose coupling can be captured with the tight coupling method. Maneuvering flight with prescribed control inputs is easily incorporated into the analysis, provided that the codes (and geometry deformation routine in particular) are robust enough to handle the problem. Although tight coupling could be used to achieve the goal of determining trim settings, it would be much more computationally intensive. This task is best left to the loose coupling process which exploits the periodicity of the steady problem in order to save effort. Figure 2-5 is a flowchart of the tight coupling process.

The first step in tight coupling is actually to run a loose coupling case. Trim conditions and the developed flow solution from the loose coupling run are used as the starting point for a tight coupling run. During the tight coupling run, the codes take turns making single time steps. An external code senses when each code has completed a time step, performs a data conversion so the other code can read the output, and then flags the code to take the next step. The values passed between the codes are the dimensional values of force or deflection; no delta airloads computation is performed for tight coupling. This process continues until the time limit for the simulation has been reached.
At the start of the tight coupling process, one code must take the first time step to initiate the process. Several types of time-stepping schemes have been proposed, including a staggered scheme [45]. The scheme used in this work is the simple time-lag scheme in which the codes are evaluated at the same time step, and the data from the last code to run in the current time step is passed to the other code in order to perform the next time step. The CSD code is “primary” in the tight coupling procedure used, and the CFD code is simply a follower in the time domain.
2.3.4 Other Code Features

The balance of the FSI design requirements are not critical functions in terms of performing either coupling process, but are critical in terms of efficient execution and minimizing the tedious work required by the analyst. These capabilities are included in the FSI and are described here.

2.3.4.1 Restart Capability

Periodically during operation of the CFD code while using the deformation capability, the code can quit running due to poorly-formed cells yielding an invalid flow solution. The remedy to this problem is often to simply re-start the code from the most recently saved solution; the re-start forces the code to pick a new set of points on the surface of the deformed body to use to track the deformation. When these new points were used, the code would typically have no trouble running through the step that previously caused a failure.

In order to avoid a code blow-up resulting in the computer sitting idle for long periods of time, the FSI includes a function which watches the CFD code and detects a failure. After detection, the code determines how much longer the CFD code has to run to finish the 360 time steps which need to be run in this coupling iteration. The FSI will then re-write the \textit{U}^2\textsc{NCLE} input file so that the run restarts and finishes the revolution it is currently trying to run. Then, \textit{U}^2\textsc{NCLE} will be restarted. When the CFD run is over, another FSI function will remove duplicate time steps which were written to the sectional force file.

2.3.4.2 Convergence Metric Compilation

The FSI conveniently compiles the data which corresponds to convergence of the coupling procedure. The first data set in this category is the set of control setting determined by DYMORE. A function in the FSI finds the appropriate output files
where this data is recorded, computes the averaged value of control setting over the
last revolution, and then writes each of these three values to the output file in the
master directory.

2.3.4.3 Deflection Evaluation

In monitoring the coupling procedure, the analyst would like to know the details
of how the deflection profile is changing from iteration to iteration. The deflection
data is large, and the analyst would like a snapshot of the overall deflections being
computed by the CSD code. To provide this snapshot, the FSI includes a function to
compute the average value over all airstations as a function of blade azimuth position.
This data can provide valuable troubleshooting information in the case that the CFD
code is quitting due to the deformation routine.
Chapter 3

Model Setup Description

This chapter details the process of developing CFD and CSD models in a way that allows the coupling process to be performed. Inputs unique to the coupling setup will be described, along with the conditions that must be met in order to perform a valid coupling analysis. The entire setup must be based on a consistent system of coordinate systems, and that is where the setup description begins.

3.1 Coordinate Systems

To correctly interpret the results of the analyses which will be presented, a firm grasp of the variety of coordinate systems in which the results are resolved is important. The rules given by Dreier [46] are good guidelines to understanding the many reference frames and quantities involved in rotorcraft analysis. This work uses the following coordinate systems to describe different results:

1. Inertial Frame: The inertial frame is a non-accelerating coordinate system. It is present in every DYMORE model by default, and is the frame into which the freestream flow and trim settings are resolved. The INERTIAL frame serves as the root frame for the entire model setup. Airloads read-in by DYMORE are assumed to be in the INERTIAL frame.
2. Blade Frame: Each blade has one blade frame which sits at the center of rotation and rotates with the blade. At all times the x axis points down the length of the blade, taken to be in the undeformed state. The y axis points towards the leading edge of the blade, but is not used to define the leading edge. Then, the z axis points in the direction of the rotation vector for the rotor. Deflections output by DYMORE are in this frame.

3. Airstation Frame: A local coordinate system is present at each airstation location along the length of a blade. By definition, the x direction points along the length of the blade, the y direction points along the chord line to the leading edge, and then the z direction is perpendicular to the chord line and is the direction of the normal force coefficient, which is a primary output of the coupled analysis. Sectional pitching moment is the other primary output and is the aerodynamic moment about the x direction in this frame.

Additionally, there are two coordinate systems in U^2NCLE that describe forces and moments based on Component and Body groups:

1. Component Frame: This is the CFD grid frame equivalent of the INERTIAL frame in DYMORE, and its axes are denoted as $X_{com}$, $Y_{com}$, and $Z_{com}$ in Figure 3-1. All integrated and sectional forces and moments reported in the Component frame have the same moment center, which is defined by the user.

2. Body Frame: Similar to the airstation frame in DYMORE, the Body frame is defined in U^2NCLE to report integrated and sectional forces, and moments in a given Body group (a user-defined set of surfaces in the CFD grid). This is the frame used to output airloads which are then reported to DYMORE. In the $rotor\_deflection.N.dat$ files, a pitching axis, denoted pp, is defined which runs along the blade length, flapping axis pf points toward the blade leading
edge, and pt is parallel to the rotation direction of the rotor. A beam axis pb is also defined along the pitching axis. Figure 3-1 shows an example of this reference frame which is rotated 90 degrees compared to the Component Frame $(X_{com}, Y_{com}, Z_{com})$.

Figure 3-1: U$^{2}$NCLE Body coordinate system and the Component Frame

Part of the deflection functionality included in U$^{2}$NCLE is angular deflection of the airstations, which must agree with the method DYMORE uses to report the deflections. DYMORE 2 has several ways of outputting rotations, including various Euler angle methods and the Weiner-Milenkovic parameters. Here the Euler Angles 3-2-1 rotation process is described because it is the one used for the CFD/CSD coupled analysis. Refer to section 2.1.4 for details on that process.
3.2 U²NCLE Models

Inputs and outputs of the U²NCLE code are described in this section, with an emphasis placed on the parts of the setup which facilitate coupling with DYMORE.

3.2.1 U²NCLE Inputs

At a high level, U²NCLE functions as shown in Figure 3-2. Flow and operational inputs are read in from the u2ncle.input. The u2ncle files hold the computational domain; before the code has been run these files exist but hold no useful flowfield data. After U²NCLE has been run the flow information is updated in the u2ncle files and they can serve as inputs to a restart case. Deformed bodies are designated alongside the boundary conditions in the .bc file. After these inputs, the wake model file stores locations of the airstation, airfoil leading edge, and trailing edge locations. The airstation locations must match those in the DYMORE input.

File(s) rotor_deflection.N.dat define the Body Frame and store deformation data for the N deformed bodies in the analysis. The initial azimuth position of the blade in the CFD grid, pre-coning, and pre-pitching angles are input in this file and are used to relate the initial blade orientation to the Body Frame. All blades of the same rotor should have the same Body Frame and beam axis definition, but different blade azimuth angles, and pre-coning and pre-pitching angles, if any. The file deformation_N.rst is an output of U²NCLE but is used for restarting a run only when the blade(s) are in a position other than the initial position of the analysis.

Wake model parameters are input to U²NCLE through the file(s) wake_model.N.dat where there is a file from 1 to N, where N is the number of blades in the CFD model. Wake model input files describe the flow and rotor characteristics (advance ratio, number of blades in rotor) as well as user parameters such as the number of rotor revolutions to retain the trailing vortices in the solution and the number of trailing
vortices which the model will compute. The bulk of these files are rows of ten columns which list for each airstation the airstation number (column 1), the normalized x, y, and z position of the airstation (cols. 2-4), the x, y, z location of the blade leading edge at the airstation’s radial location (cols. 5-7), and the x, y, and z position of the trailing edge at that radial location (cols. 8-10). The airstation coordinates are used as the locations for reporting sectional forces. Lastly, the normalized corner coordinates of a bounding box are listed. This bounding box should envelope the CFD computational domain and have two of its walls coincide with the two periodic faces of the domain. As described in Section 2.1.3, wake model features inside this box will not have their induced velocity included in the CFD domain because it would be double-counted. The induced velocity of the wake structure inside this box is included in the solution as a boundary condition on the leading periodic surface.
Figure 3-2: Black box U²NCLE functionality
U^2NCLE takes inputs which are normalized according to a common set of reference values, which are first introduced in section 2.1.1. The inputs are captured in the input files `u2ncle.input`, `wake_model.1.dat`, `rotor_deflection.1.dat`, `.bc`. Unfortunately, there is some redundancy in the input parameter definitions between these files. Flow initial conditions are set in `u2ncle.input`, while the boundary conditions are given in the `.bc` file. Some normalizations presented here are typical but have not been previously documented, and some are inputs required by newer features of the code. The hat over the variables below denote a value normalized for U^2NCLE. A normalized value for rotational speed \( \hat{\Omega} \) is computed from dimensional values according to the relation:

\[
\hat{\Omega} = \Omega \frac{L_r}{V_r}
\]

(3.1)

where \( \Omega \) is rotation speed in radians per second, and \( L_r \) and \( V_r \) are the familiar reference quantities given in Table 2.1. The boundary condition file for a rotating grid will require this rotation speed and the normalized pressure as well. The normalized pressure \( \hat{p} \) is:

\[
\hat{p} = \frac{p}{\rho_r V_r^2}
\]

(3.2)

with \( p \) being freestream static pressure, \( \rho_r \) being reference density, and \( V_r \) being reference velocity. A time step corresponding to one degree azimuth position change is typically chosen for rotor analysis. More generally, the user will want to advance the simulation by \( N \) degrees per time step. The normalization for the time step \( \hat{dt} \) which produces an \( N \) degree azimuth change is computed as:

\[
\hat{dt} = \left( \frac{2\pi}{\Omega} \right) \left( \frac{N}{360} \right) \frac{V_r}{L_r}
\]

(3.3)

where as before \( \Omega \) is the dimensional rotation speed in radians per second, \( V_r \) is
reference velocity, and $L_r$ is reference length. As stated in section 2.1.1, the reference length $L_r$ is typically the rotor blade radius and the reference velocity $V_r$ is usually the freestream flow speed. Lastly, the *wake_model.1.dat* file requires the advance ratio of the rotor which is:

$$
\mu = \frac{V_r \cos(\alpha)}{\Omega R}
$$

(3.4)

The FSI is set up to allow the analyst to establish a master deflection file containing the airstation positions in the baseline configuration. Then after the CFD grid is rotated to the collective angle after iteration 0, the airstation locations have changed and need to be updated in the U²NCLE input files. If the user inputs the collective rotation angle, the FSI will rotate the airstation positions in the wake model file by that number of degrees before writing the file.

U²NCLE outputs normalized sectional forces and moments at each airstation along the blade. The FSI requires the factors which will transfer these values back into the dimensional domain. To convert the U²NCLE output values to dimensional values, the following relations are used:

$$
\text{force} = 1 \frac{1}{2} \rho_r V_r^2 L_r^2 \quad \text{and} \quad \text{moment} = 1 \frac{1}{2} \rho_r V_r^2 L_r^3
$$

(3.5)

$\text{force}$ has units of force, and $\text{moment}$ has units of a moment in whatever unit system was used to normalize U²NCLE. When multiplied by the U²NCLE output sectional forces, the result is analogous to the sectional load outputs of DYMORE.

### 3.2.2 U²NCLE Outputs

U²NCLE writes out force and process data to various files during run time. This section describes the files listed in Figure 3-2, their contents and utility in the coupling process. Files global_resid.out and global_turb.out hold the residuals for the Navier-
Stokes solver and the turbulence solver, respectively. Either will have the current process step number and several other values. Detecting a residual of NaN in either one of these files signifies a blow-up in that particular part of the code and the user can use this information to go about adjusting parameters so as to avoid future blow-ups during unsteady CFD runs.

Two different reference frames are used by U²NCLE to express forces and moments: the Component frame and the Body frame. Component airloads are resolved in the Component Frame or Grid Frame described earlier, whereas integrated and sectional body airloads are reported in the Body Frame. On top of the component or body distinction, two different moment centers can be chosen for computing the moments. Forces in \textit{forces.\text{Comp}.\text{comp}_N} and \textit{secforces.\text{Comp}.\text{comp}_N} use the moment center defined in the Component Frame (defined in the \textit{.bc} file), whereas forces in \textit{forces.\text{Body}.\text{body}_N} and \textit{secforces.\text{Body}.\text{body}_N} files use the local beam coordinates in \textit{rotor.deflection.N.dat} defined in the Body Frame. Component sectional forces are written for each airstation defined in the \textit{wake.model.N.dat} file(s), while Body-frame sectional forces are written for each local beam coordinate in \textit{rotor.deflection.N.dat} file(s).

Continuity in the grid deformation is retained through data stored in the deformation.N.rst file(s). Each of the N deformable bodies in the flow domain has its current deformation parameters written to these files periodically. Upon restart, this data is called upon to determine the configuration of the blades as implemented in the \textit{.u2ncle} flow solution/restart files.

\subsection*{3.3 DYMORE 2 Models}

DYMORE 2 inputs and outputs are described in this section as they pertain to performing a coupled analysis with U²NCLE. A procedure for setting up the models
is provided as well.

### 3.3.1 DYMORE Inputs

A simplified input/output chart for DYMORE is shown in Figure 3-3. The case_name.dym file is fed to the DYMORE program on the command line and can include any number of other files which contain the model definition.

**INPUTS**
- caseName.dym
- deltaAirloadsInertial.dat
- initialConditions.rcv

**OUTPUTS**
- caseName.rcv
- caseName.out
- DymoreAirloadsInertial.dat
- Deflections.dat
- sensor output

**Figure 3-3: DYMORE black-box behavior**

Purposes of the various input files are given below. Numerical inputs are provided to DYMORE as physical values (i.e. values which have units), as opposed to normalized values as in U²NCLE.

*caseName.dym* is the main simulation file which is given to DYMORE as a command line argument. In addition to containing the main directives and simulation parameters (time step size, etc.), it typically contains the list of files which themselves contain the model specification. *aeroIntInput.dat* is a file containing delta airloads which are read in by DYMORE and applied to an aerodynamic body in the INERTIAL frame. *caseName.rcv* is an archive file containing partial kinematics data output by a previous run. The archive frequency is user-specified and an archived time step result can serve as the initial conditions for a simulation. Alternatively, the
initial conditions can be chosen as being at rest.

A DYMORE case can be effectively restarted from a previously saved run step by specifying an initial condition as an input. The initial condition is read from the archive \textit{.rcv} file and requires that the previous run saved the step that becomes the restart step. Blade deformation is included in the restart, but the velocities are not. This is among the reasons for starting a maneuvering case off with a steady control signal.

Model specification inputs to DYMORE can be separated according to the aspect of the model they address. Multibody dynamics, aerodynamics, and the autopilot simulator are the three features of DYMORE which facilitate the coupling process. As a reference for the following discussion, Figure 3-4 shows the hierarchy of these features, their interrelated elements, and the unifying main file of the model.

![Figure 3-4: Example DYMORE hierarchy](image)

At the top level is the main simulation file which is fed into DYMORE as a command-line argument and ends in \textit{.dym}. This file has certain declarations it must
contain, as outlined in the DYMORE manual [11]. Typically, this file also includes a series of @INCLUDE commands which allow it to read in data in other input files. In order to compartmentalize the model and make understanding it easier, different elements of the system will be contained in these other files whose names end in .dat.

3.3.1.1 Structural Specification

Topology begins with the specification of points. The points are used to define any number of elements in the DYMORE library, and the commonly used elements are beams, rigid bodies, and joints. As shown in Figure 3-4, a beam element is based on a curve element, which is in turn defined by specifying two points. The curve has associated with it an orientation distribution, which is defined by using the word @ORIENTATION\_DISTRIBUTION in a DYMORE input file. The orientation distribution specifies the angular rotation (in Euler Angles 3-2-1, for example) at two or more points along a geometric element. Locations along the element can be specified by normalized or non-normalized coordinates. While this distribution seems meaningless for a curve, the beam inherits its orientation from the curve. That is, the orientation of the beam cross-section principle axes will be computed from the orientation distribution associated with the curve. Rigid bodies are connectors between two coincidental or non-coincidental points, which prohibit any motion between the two points. Joints are the third important class of objects which are used in the models of this work. By specifying two points and a triad, the user can specify a revolute joint in the system.

3.3.1.2 Aerodynamic Specification

Aerodynamic lift, drag, and pitching moment are computed using 2-D airfoil lookup tables. Incoming flow properties are defined under the @AIR\_PROPERTY\_DEFINITION keyword.
3.3.1.3 Control System Specification

The control system performs a critical role in the coupling process, and its setup has some nuances covered in this section. DYMORE keyword @CONTROLLER_DEF INITION declares the presence of a control system in the model. There are a variety of control system types, with only the autopilot kind described here. The control system object links the input signals to the output signals, and links to the object that describes how the inputs relate to the outputs.

Inputs to the control system are typically the total rotor thrust, roll moment, and pitching moments for a helicopter system, as described in section 1.2.2. Control system outputs can be sent to a joint which directly controls the blade pitch or, alternatively, a set of push rods to control their position.

3.3.2 DYMORE Outputs

In addition to creating code-monitoring output files automatically, DYMORE can output any simulation result which corresponds to one of the sensor types in its library. Considering a main file named caseName, the following files will be written by DYMORE.

During run time, main DYMORE output is directed to the file caseName.out. It reports the inputs as they are read, the result of various validity checks performed on the model, the iteration-by-iteration status of the run while it is in progress, and the post-processing stage of the analysis. This is the first place to check in the case of unexpected behavior. The second standard output from DYMORE is to an archive file called caseName.rcv. Data on the kinematic state of the model is stored in this file at user-specified intervals.

In addition to these automatic outputs, DYMORE outputs a number of files which are used in the coupling processes. Details on the content of these files is not
well documented in the 2006 version of the DYMORE User’s Manual [11], and the
information presented here is intended to make up for this dearth.

The TotalAirloadsInertialConcentrated.dat file contains the aerodynamic loads
which DYMORE applied to each airstation of each blade in the rotor, reported in
the INERTIAL coordinate system. The deflection file name is also specified in the
aerodynamic interface. When the airloads type is set to OVERFLOW, deflections
in this file are reported in the local blade frame. Sensor outputs as described in
Section 2.2.4 are also written according the user input. The DYMORE Manual [11]
has complete information on sensor specification.

3.3.3 Setup Procedure

The information above is summarized in the recommended procedure for setting
up DYMORE 2 models:

1. Setup Structural Model

   (a) Set up model topology/geometry

      i. Define coordinate systems

      ii. Define model geometry points

      iii. Define curves

      iv. Define beams and rigid bodies

   (b) Define aerodynamic features

      i. Create lifting line features (including airstation locations)

      ii. Establish rotor object, which is a collection of lifting lines

      iii. Declare the aerodynamic interface by assigning one rotor object to it

   (c) Set up control system
3.4 Model Setup for Coupling

The following process is recommended for setting up models which will be used in the coupling process. To carry out this process, the user will need to identify the built-in collective angle (at r/R = 0.75) in the DYMORE model blade. The outcome of this process is that the grid deformation implemented by U²NCLE will be minimized and through this the code robustness will be improved. The procedure is:

1. Run the DYMORE case of interest using the internal lifting line airloads and obtain the collective control setting. This will serve as collective angle for the baseline geometry for this test case; the next steps are to implement this collective angle in both the CFD and CSD geometries.

2. Set the collective angle in the DYMORE geometry.
   
   (a) Identify the pitching axis of the blade, define a new coordinate system to coincide with it if one does not already exist.
   
   (b) Redefine the blade and other equipment (according to the user’s judgment) in the coordinate system corresponding to the pitch axis.
   
   (c) Set the angle of the pitch axis such that the built-in collective of the DYMORE configuration becomes the collective value determined in Step 1.

3. Set the collective angle in the U²NCLE geometry.
   
   (a) Rotate the blade geometry to match the collective value determined during the first DYMORE run.
   
   (b) Construct wake model file. Use airstation locations from the DYMORE input file and leading edge/trailing edge values picked from the U²NCLE
geometry file (NOTE: Currently, the FSI does not automatically adjust the LE/TE locations when a new collective angle is set).

4. With the new collective-adjusted CFD and CSD models, the coupling run proceeds with a second running of DYMORE using lifting line airloads only, followed by a CFD run.

3.5 The Fluid-Structure Interface

This section describes the procedure used by the FSI code to perform loose coupling and tight coupling.

3.5.1 Loose Coupling Implementation

According to the design requirements listed in Table 2.2, the coupling procedure was implemented in a Python code [47]. The enumeration of the procedure is presented below, where X is the user-specified iteration whose results are the seed for starting the automated process.

1. Create the main data object.

2. Parse input file fsi.input located in the directory where the run was initiated.

3. Check for the presence of the directories required to start and run the loose coupling process: csd.X, cfd.X, data.X, routines.

4. Loop:

   (a) Run DYMORE 2.

   (b) Verify the control system converged to a steady value.

   (c) Compile DYMORE 2 convergence data: collective, lateral, and forward cyclic control settings.
(d) Translate DYMORE 2 deflection output to the format U^2NCLE can accept.

(e) Run U^2NCLE for 1 revolution using the blade deflection computed in the 
    DYMORE run of step a.

(f) Compile U^2NCLE convergence data: thrust, pitching moment, and rolling 
    moment.

(g) Evaluate convergence metrics.
    i. If converged to within user-specified tolerance, quit process.
    ii. If not converged, continue looping.

(h) Compute delta airloads according to Equation 2.46.

(i) Prepare for next iteration if directed by the user: create directories for the 
    next iteration.

(j) Advance iteration number and begin at top of loop.

5. DONE.

The FSI converts data into the appropriate read/write format, and operates such 
that the data is in a sensible structure. One advantage of using the FSI to perform 
loose coupling is that it leaves an intuitive “directory trail” of the process. Figure 3-5 
shows the directories used or created by the coupling process, and the order in which 
the directories are stepped through during the procedure.
The *master* directory contains convergence data which is updated as the code runs. The code starts by running with internal aerodynamics-only in the directory csd.0. To be sure that internal aerodynamics are being used, the user is encouraged to verify that the `@FILE_NAME_READ_AIRLOADS` field is commented out of the appropriate aerodynamic interface definition. Deflection data from this run is copied to data.0, where it is converted to a form which can be read by U²NCLE. Then, the new *rotor_deflection.1.dat* file is copied into the cfd.0 directory, where the U²NCLE
is then run, generating a set of CFD-based sectional airloads. The CFD sectional airloads are copied into data.0 along with the CSD sectional airloads, and a delta airload file is computed and written in DYMORE format; this ends the coupling start-up process (denoted by the “0” in the name).

With the delta airloads file from iteration 0 in hand, the process moves on to the first actual data-exchange coupling iteration, iteration 1. The FSI creates the folders for the new iteration (csd.1, data.1, and cfd.1). The aerodynamic interface for csd.1 must be set to read in data, so the read-in command must be un-commented. The delta airloads file from data.0 is copied into the folder csd.1 and DYMORE is run again, generating a new deflection file. The process repeats like this until convergence.

### 3.5.2 Tight Coupling Implementation

The Fluid-Structure Interface has a tight coupling module separate from the loose coupling one described above, coded in C. The code requires effort on the part of the user to start both the U^2NCLE and DYMORE processes which must run simultaneously during the tight coupling process. Once started, the tight coupling routine will perform data exchange tasks without user intervention until the process has run its course. The complete code process as implemented in the FSI is listed below. These cases are typically restarted from a previous loose coupling DYMORE run, so the DYMORE input file must reflect that fact by having a restart file and time step specified in the .dym file. As is observed from the listing below, the tight coupling routine manages the process by acting as a go-between for U^2NCLE and DYMORE 2. Specifically, the tight coupling code is looking for simple one-number flag files to be output by each respective code indicating that the computation for the current time step is complete and airloads or deflections have been written to an output file. The tight coupling process executed by the code is:
1. The user starts U$^2$NCLE in the main terminal with tight coupling option on.

2. User starts DYMORE in a separate terminal, with tight coupling selected as the @COUPLING_TYPE.

3. Allow U$^2$NCLE a start-up time to generate the airloads at the initial azimuthal position. After the start-up period, write a 1 to the flag file cfd_csd.flag.

4. Loop:

   (a) Read cfd_csd.flag file. If content is 1, proceed with:
      i. Advance the tight coupling iteration number by 1.
      ii. Convert U$^2$NCLE force output into DYMORE format.
      iii. Write the new time step to cfdFlag.dat.
      iv. DYMORE 2 reads this and will compute the next time step.

   (b) Wait for 2 seconds.

   (c) Read csdFlag.dat. If its content has been updated since the last file read, proceed with:
      i. Convert DYMORE 2 deflection output into U$^2$NCLE format.
      ii. Write a 1 to the file csd_cfd.flag.
      iii. U$^2$NCLE reads this and will compute the next time step.

   (d) Wait for 2 seconds.

   (e) If the timestep is greater than or equal to the intended ending time step number, quit. Else, continue with the next loop.

5. DONE.

Between the two file checks in steps (a) and (c), there are short waiting periods to avoid any interference in the input/output operations. The DYMORE computations
are rather quick (less than 10 seconds), but the U\textsuperscript{2}NCLE computations can take a minute or longer. Then this code is hyperactive in checking for the next flag file from either code - it will check multiple times whether a new flag file has been written out or not. The inputs and outputs from each code contain information for each blade at the given time step. Unlike in the loose coupling procedure where the data for a single blade is used (with the appropriate phase change) for each other blade in the rotor, this tight coupling routine must use the data from each blade individually, because each could be undergoing a unique deflection.

In the listing above, the control system in DYMORE must be turned off, so the code is using user-defined control system inputs. By manipulating these control inputs, the user can specify any profile of interest. For example, to simulate a +/- 2 degree collective pitch doublet as was done in this work, the user gives this profile as the time history for the control system outputs.
Chapter 4

Results

This chapter reports the setup and coupled analysis results for two helicopter rotors. First, the UH-60A (Blackhawk) main rotor results are presented at four level flight conditions in separate CFD/CSD loose coupling runs. Each condition posed a unique challenge to the coupling process in terms of flow physics and simulation convergence. The second configuration is that of the Bell 427 vehicle main rotor. Both loose coupling and tight coupling were performed for this configuration, as a demonstration of capability. Coupled results are compared to flight test data where it is available.

4.1 UH-60A Loose Coupling Results

Each of the four flight conditions had its own slightly different setup in order to optimize the code robustness. Results for flight counters c8534, c8513, c9017, and c9020 are presented, each flight counter presenting a different combination of flow phenomena. These results were initially presented in separate conference papers [48, 49].

Loose CFD/CSD coupling analysis typically shows process convergence by way of converged control signals (i.e. control signals which don’t change with more iterations of the coupling process). This condition means that the CFD model will have achieved
the target trim settings. Once convergence is reached, the results of interest are the airloads from the CFD, structural loads from the CSD analysis, and the final trim settings computed by the CSD code. The UH-60A pitching moment airload has an erroneous offset in the computed values due to sensor malfunction. In order to avoid misrepresentation of the results, the mean has been removed from the flight test data and the computed airload moments in all the following results. Also, values and units are not included on the structural load results to protect the data. This omission does not hinder comparison between predicted results and flight test data.

4.1.1 UH-60A Problem Setup

The DYMORE model of the UH-60A rotor is fairly simple in that the swashplate and mast components are not included in the model. Whereas the control system of the real hardware applies its outputs to swashplate displacements corresponding to each control mode, the UH-60A model control system controls a node on the pitch horn, which means the prescribed pitch of the blade is limited to the form shown in Equation 1.1, which is periodic and unable to capture any sort of non-periodic control inputs as would be seen during a maneuver, for example.

![Figure 4-1: The topology of a single blade of the 4-bladed UH-60A DYMORE model](image)

Figure 4-1 shows the topology of the UH-60A model used. The full model featured four of these blade/damper systems, all connecting to a common hub element.
The setup used 81 airstation locations. Twist was built in to the blade in DYMORE by specifying an orientation distribution for the airstations along the length of the blade. Blade geometry data was taken from a report by Bousman [50]. Aerodynamic properties are associated with a lifting line object, named as such because it implements the lifting line theory of Prandtl in the calculation of local circulation and its effects. Accordingly, there is a lifting line definition feature in DYMORE where the user specifies the number of airstations in the lifting line, the orientation distribution associated with the airstations in the line, and the location of the airstations themselves.

To this point, the data for just a single blade has been specified. DYMORE 2 has a copy command which allows the user to specify the base item which is being copied and the frame of the copy. The phase difference between the blades is also specified as a percentage of $2\pi$ radians, and serves to correctly implement the control inputs which can vary with the azimuth position of the blade. For example, the UH-60A has four blades separated by 90 degrees; the copy command for this case will specify the base item name and frame, the frame of the blade leading the base blade by 90 degrees, and then a phase difference of 0.25 is specified. From this, DYMORE constructs a copy of the base item, but does so in the new frame and then associates a 90 degree phase difference between the copy and base item.

The assembly of lifting lines which have been specified and copied are collectivized as a rotor object. In addition to having one or more lifting lines associated with it, a rotor has a specified rotation speed which is redundant with the speed of rotation of the rotor’s main revolute joint, and a reference length. The final layer in the hierarchy is the aerodynamic interface, which is a required item whether data is exchanged with an external code or not. The aerodynamic interface is where the aerodynamic scheme of the model is specified. Users can tell the code that all aerodynamics will be from internal aerodynamic models, or they can specify numerous external codes.
OVERFLOW, COPTER, and RDYNE are three external codes which exchange airloads with DYMORE, each in a different format and reference frame. For this model, the airloads scheme was set to OVERFLOW, so DYMORE outputs and reads in airloads in the inertial reference frame, and reports deflections in the local frame. This deflection-reporting local frame is specified in the lifting line definition and is in the moving frame name field.

All UH-60A models were set up according to the procedure in Section 3.4 which minimizes the deformation that has to be implemented by the CFD grid deformation routine. To accomplish this, all CFD grids had the blade pitched to the collective angle determined by the first run of DYMORE, which used only DYMORE internal airloads. With this and the moving frame name frame set to the same angle, deflections were minimized and the CFD code robustness was improved, as compared to runs which did not use this method. Figure 4-2 shows the UH-60A grid refinement zones, with the blade visible in the center of the innermost volume.

Figure 4-2: Computational domain of the UH-60A single blade
Wake model settings used for the UH-60A cases used a single wake trailer and updated every 5 time steps. The airstation locations were normalized and listed in an U$^2$NCLE input file called wake_model.1.dat. Also listed in this file were the Cartesian coordinates of the leading edge and trailing edge points at the same blade radii as the airstations. The LE and TE point locations are required so that the wake model algorithm knows to compute the circulation around the blade at a location further from the airstation than either of these points. Finally, the wake model input file contains the corner coordinates of a bounding box which contains the entire computational domain. Values were chosen for the bounding box so that it was just large enough to completely envelope the domain.

The user-specified support radius had a great effect on the stability of the code. In order for the code to run smoothly, the support radius was set to 50.0 R, meaning that the entire CFD computational domain was included in the deformation. This increased run time, but was required to get a solution. The number of Greedy points were changed in a non-systematic manner through the various CFD runs; values ranged from 240 to 300.

### 4.1.2 Flight Conditions

UH-60A analysis considers four flight conditions which have become the standard cases for studying CFD/CSD coupling (summarized in Table 4.1). Velocity is expressed in the INERTIAL frame of DYMORE, with the x direction pointing down the nose of the vehicle and the z direction pointing upwards, in the thrust direction. Table 4.1 lists the names of the flight conditions (“counters”) and the associated values of some flow variables.
Table 4.1: UH-60A flight conditions

<table>
<thead>
<tr>
<th>Flight Counter</th>
<th>Flow Velocity (ft/s)</th>
<th>μ</th>
<th>$C_t/\sigma$ (m/s)</th>
<th>Thrust (lbf)</th>
<th>Pitch Mom. (ft-lbf)</th>
<th>Roll Mom. (ft-lbf)</th>
</tr>
</thead>
<tbody>
<tr>
<td>c8534</td>
<td>266.5</td>
<td>0.368</td>
<td>0.084</td>
<td>17,665</td>
<td>-2,583</td>
<td>6,884</td>
</tr>
<tr>
<td>c8513</td>
<td>110.4</td>
<td>0.153</td>
<td>0.076</td>
<td>16,302</td>
<td>-5,470</td>
<td>958</td>
</tr>
<tr>
<td>c9017</td>
<td>170.2</td>
<td>0.237</td>
<td>0.129</td>
<td>16,688</td>
<td>112</td>
<td>-320</td>
</tr>
<tr>
<td>c9020</td>
<td>177.6</td>
<td>0.245</td>
<td>0.120</td>
<td>16,535</td>
<td>-2,176</td>
<td>616</td>
</tr>
</tbody>
</table>

4.1.3 Flight Counter c8534 Results

UH-60A flight counter c8534 is a high speed forward flight case. Transonic flow occurs on the advancing blade tip, and there is a substantial region of reverse flow on the retreating blade near the hub.

4.1.3.1 Convergence

Convergence metrics for the c8534 flight counter are shown in Figure 4-3. The CFD/CSD coupling process converged in 15 iterations, which occupied 127 hours of wall clock time (7 hours U²NCLE run time on 64 computers, 1.5 hours of DYMORE run time on one computer), or about 6,700 CPU-hours. Convergence was achieved while using the exact flight conditions in both the CFD and CSD models. This flight condition is near the upper limit of the vehicle’s forward flight speed; the rotor is not producing a high percentage of its maximum thrust, and coupling process convergence is not expected to pose a challenge.
Figure 4-3: Trim setting convergence during CFD/CSD coupling, UH-60A flight counter c8534

Figure 4-4 shows the average pitch profile of all the airstations on the UH-60A blade as a function of azimuth position. The result shown includes both the pitch which is driven by the control system and the pitch which results from the elastic deflection of the blade. A negative sine wave prevails in this profile, and this is consistent with the control system prescribed profile. It is worth noting this profile and how it compares and contrasts with the airloads presented below.
4.1.3.2 Airloads

The wake model part of the flow solution is shown in Figure 4-5. The high speed flow of this flight condition causes transonic flow near 90 degrees azimuth, and the wake model shows signs of changing lift profile along the blade radius in this region. This change is reflected in changing wake trailer location as each blade moves beyond the 90 degree mark. A set of trailer “horns” are visible in the wake trailer path, which testifies to the under-resolution of the changing lift profile and the under-resolution of the wake structure. Predicted airloads near the tip of the blade could be erratic and unrealistic due to the induced flow of the wake trailer not being captured in the correct location. An argument for a more well-resolved wake can be made based on these observations, but in this case it was time prohibitive to achieve higher resolution.
Considering Figures 4-6 to 4-11, the airload trends change markedly between \( r/R = 0.40 \) and stations outboard of this. At the inboard stations the normal force follows a trend like the local velocity, making a cosine wave. On the more outboard locations, the normal force trends go like the driven pitch profile, a negative sine wave. The effects of reverse flow at radii less than about \( r/R = 0.40 \) can be seen in Figures 4-6 and 4-7, as a rise in the sectional pitching moment. Notably, the coupled results capture the reverse flow effect while the DYMORE aerodynamic model fails to. This simulation does not include the helicopter fuselage, but Potsdam [4] demonstrated that the pitching moment peak at 270 degrees and \( r/R = 0.400 \) seen in Figure 4-6 is more true to flight test data when the fuselage is included.

Normal force coefficients are captured quite well at all radial locations. Coupled results obtain a high rate of change with azimuth position, which matches the quality of the flight test data. The pitching moment coefficient of Figure 4-11 shows a trend opposite of the flight test data. This change in trend is explained by observing the changes in the wake trailer location shown in Figure 4-5. \( r/R = 0.975 \) is near the tip
of the blade and outboard of the wake trailer shed location in this flight condition. Thus the induced velocity from the wake is opposite of the physical value in this location and an incorrect trend results.

Airload results for this case have been improved due to the inclusion of the CFD-based airloads. Improvement takes the form of better peak/trough prediction, better prediction of the phases of airload trends, and the inclusion of the correct signal frequencies in the end signal. Evidence of the first two elements is present in any of the airload results. The point of the third element of improvement is that the airloads change in a manner similar to what is shown in the data. The clearest evidence of this is seen in the pitching moment at $r/R = 0.925$, near 90 degrees azimuth; a sharp jump is seen, possibly associated with tip flow becoming transonic. The coupled airloads capture this trend but DYMORE results do not.
Figure 4-6: UH-60A flight counter c8534 airloads, r/R = 0.225
Figure 4-7: UH-60A flight counter c8534 airloads, r/R = 0.400
Figure 4-8: UH-60A flight counter c8534 airloads, r/R = 0.775
Figure 4-9: UH-60A flight counter c8534 airloads, r/R = 0.875
Figure 4-10: UH-60A flight counter c8534 airloads, r/R = 0.925
Figure 4-11: UH-60A flight counter c8534 airloads, r/R = 0.975
4.1.3.3 Structural Loads

Loads experienced by the blade structure are important outcomes of rotorcraft analysis in that it provides the mechanical designer with information which leads to a structure which is neither over- nor under-designed. Torsional and flapping moments shown in Figures 4-12 and 4-13 are felt in the structure as a result of the airloads and system kinematics.

The torsional moments of Figure 4-13 do show enhancement as a result of the coupled analysis. Enhanced airloads have resolved a sharp change in twisting moment at \( r/R = 0.700 \), but peak load prediction was not improved in the coupled results as compared to the DYMORE airload results. Structural loads are generated through the kinematic interpretation of the airloads and certain erroneous features of the airloads could be exaggerated during this transformation; as a result, the requirements of a coupling process which generates quality structural load predictions are higher than that which produces quality aerodynamic predictions. Specifically, the coupling process is required to include some dampening or data processing to remove problematic aspects of the CFD airloads, such as non-physical oscillations or perturbations. Structural loads throughout this work demonstrate the need for such a function in the loose coupling process.
Figure 4-12: UH-60A flight counter c8534 flapping moments, r/R = 0.3, 0.7
c8534 Torsional Moment at $r/R = 0.300$

Figure 4-13: UH-60A flight counter c8534 torsional moments, $r/R = 0.3, 0.7$
4.1.4 Flight Counter c8513 Results

In the UH-60A flight counter c8513, the moderate forward speed and almost zero vertical speed of the incoming flow leads to a significant amount of blade-vortex interaction. Structural vibration is known to occur at this flight condition, driven by the impulsive vortex interactions with blades.

4.1.4.1 Convergence

Convergence was achieved within 14 coupling iterations, occupying 119 hours of wall clock time or 6,700 CPU-hours. The time to get one result from the CFD or CSD code did not change from the c8534 case (7 hours and 1.5 hours, respectively). Figure 4-14 shows how the trim settings changed as the coupling iterations progressed.

![Figure 4-14: Trim setting convergence during CFD/CSD coupling, UH-60A flight counter c8513](image-url)
4.1.4.2 Airloads

The wake structure predicted by the free wake model is shown in Figure 4-15. The forward speed of this flight condition is low enough to keep wake feature in the vicinity of the rotor for the next blade to interact with. There is a slightly positive angle of attack of the freestream flow, which also contributes to the wake features staying near the plane of the rotor. The resulting wake structure is quite warped and curled up on itself as compared to the more ordered wake structures of the other cases seen here. This warping is indicative of a higher level of interaction with the rotor system. The absence of transonic flow in this case means that the wake trailer position is more steady, and the “horns” of the c8534 wake structure are not observed in Figure 4-15.

![Wake structure of c8513 case](image)

Figure 4-15: Wake structure of c8513 case

Normal force and pitching moment airloads are shown in Figures 4-16 to 4-21. A negative sine wave shape is apparent in the normal force coefficient near the rotor hub, as was observed for counter c8534. Moving outwards from r/R = 0.400, however,
the flight test data shows a different trend and sharp changes near 90 and 270 degrees azimuth. These locations of airload discontinuity are most visible in the pitching moments of Figures 4-19, 4-20, and 4-21, and correspond to blade interaction with the wake structure. The velocity profile of the vortex gets imposed on the bulk flow of the blade and strongly affects the flow moving normal to the chordline. The resulting induced velocity is a stark change in the aerodynamic environment of the blade, and the large change in $C_nM^2$ or $C_mA^2$ results.

The CFD/CSD coupling method was unsuccessful in capturing the BVI events which are so clear in the pitching moment flight test data. Of all the UH-60A results, this flight condition showed the least improvement in the capture of flow phenomena, with the prevailing wake phenomena not being properly captured. As noted elsewhere, the use of just one wake trailer in the wake model is responsible for the poor wake resolution and subsequent failure of the process to capture BVI. Increasing the number of trailers to 10 or 15 would likely improve results. Because the wake model module in U2NCLE is serial, increasing the number of trailers quickly increases computation time. In a preliminary check, 4 more wake trailers to a simulation doubled the run time from 8 hours to 16 hours.
Figure 4-16: UH-60A flight counter c8513 airloads, r/R = 0.225
Figure 4-17: UH-60A flight counter c8513 airloads, r/R = 0.400
Figure 4-18: UH-60A flight counter c8513 airloads, r/R = 0.775
Figure 4-19: UH-60A flight counter c8513 airloads, \( r/R = 0.875 \)
Figure 4-20: UH-60A flight counter c8513 airloads, r/R = 0.925
Figure 4-21: UH-60A flight counter c8513 airloads, r/R = 0.975
4.1.4.3 Structural Loads

Loads in the rotor structure are shown in the Figures 4-22 and 4-23. It is notable that the coupled predictions show an amount of high frequency change which is not seen in the flight test data. For the blade flapping moment, the CFD/CSD coupling process predicted mean trends which match the flight test data, whereas the trends of the two do not match well for the torsional moment at either blade radius. The differing levels of agreement between the coupling results and flight test data in airloads (where an improved prediction is achieved) and the structural loads (where the coupling prediction frequently disagrees with the flight test) is thought to be linked to spurious discontinuities in the CFD airloads.
Figure 4-22: UH-60A flight counter c8513 flapping moments, r/R = 0.3, 0.7
Figure 4-23: UH-60A flight counter c8513 torsional moments, r/R = 0.3, 0.7
4.1.5 Flight Counter c9017 Results

While this flight condition lies in the forward flight envelope, it is right next to the McHugh Lifting Boundary [7]. When this case is run in DYMORE with the exact flight condition parameters, the control system diverges, showing an important way in which the DYMORE aerodynamics are not true to the real aerodynamics (it cannot detect the extents of the flight envelope properly). Fortunately, in implementing the CFD/CSD coupling process, the DYMORE airloads are just variables in the process and are only required to capture the trends assumed by the control system, as noted earlier.

4.1.5.1 Convergence

Figure 4-24 shows the trim settings as a function of the coupling iteration. After iteration 12 changes in the values become very marginal, but the process was run to iteration 19. One hundred fifty-two hours of wall clock time were occupied by the simulation. This flight condition is, on paper, the most difficult one to resolve due to the very high thrust coefficient to solidity ratio. Even for iteration 1 DYMORE required an artificially high density to achieve stable control system behavior. Artificially increasing the air density in the CSD model is allowed during the coupling process because the end result contains only the CFD-based airloads.

Increasing the density value slightly allowed DYMORE to determine converged control settings using its internal aerodynamics. However, with this same density setting, DYMORE could not converge in iteration 2, which included delta airloads. While continually increasing the density used in DYMORE calculations is a viable option, the preferred method of improving the robustness of the DYMORE control system is to artificially extend the range of the airfoils positive slope of the $C_L$ vs. $\alpha$ curve.
4.1.5.2 Airloads

Below, the wake model results at iteration 15 are shown for flight counter c9017. From viewing the comparatively well-formed wake structure of this case, one is led to understand that there is minimal BVI occurring. Airloads results bear this out; whereas the c8513 case had a high level of BVI and was thus highly dependent upon a well-resolved wake structure (i.e. use of many wake trailers), this case is less dependent upon the wake model accuracy. With the use of just a single wake trailer, the wake in all cases can be considered under-resolved. Fortunately for this case, final results will not be appreciably effected by this condition.
Airloads predicted by DYMORE internal modeling are presented alongside the CFD/CSD coupling results and flight test data in the figures below. Airloads inboard of \( r/R = 0.400 \) trend like the local chordwise velocity magnitude, making a sine wave just as in all flight counters. The flow briefly reverses flow direction relative to the blade at \( r/R = 0.225 \), resulting in a rise in the pitching moment which can be seen in Figure 4-26. The mean-value offset seen in the normal force of Figure 4-27 was observed also by Rajmohan [18].

Because the flow condition requires a high collective, dynamic stall is prominent on the retreating blade. The first occurs near 270 degrees azimuth, and the second at about 315 degrees.
Figure 4-26: UH-60A flight counter c9017 airloads, r/R = 0.225
Figure 4-27: UH-60A flight counter c9017 airloads, r/R = 0.400
Figure 4-28: UH-60A flight counter c9017 airloads, r/R = 0.775
Figure 4-29: UH-60A flight counter c9017 airloads, r/R = 0.875
Figure 4-30: UH-60A flight counter c9017 airloads, r/R = 0.925
Figure 4-31: UH-60A flight counter c9017 airloads, r/R = 0.975
A very good agreement between flight test data and the coupled results is shown near the blade tip, Figures 4-30 and 4-31. The two dynamic stall events feature prominently in the results outboard of $r/R = 0.4$, and the coupled prediction does well to capture the presence of the effects as well as the magnitude. No gross errors in any signal phase is observed other than in the pitching moment at $r/R = 0.975$. The effect causing this inaccuracy at the tip is consistently applied across all flight conditions considered. As stated previously, the wake trailer shed location is inboard of this location and causes an upwash which occurs at a radius less than $r/R$ of 0.975. The end result is a reversed trend at this location.

4.1.5.3 Structural Loads

Structural loads for this case follow the trend of the flight test data, but show some higher frequency components which do not agree with the measured values. A signal of approximately 14/rev is clearly present in the flapping moment at both radial locations. The torsional moment at $r/R = 0.300$ is more like the flight test data in its frequency content, but the higher frequency is again present at the $r/R = 0.700$ in Figure 4-33. The blade lacks sufficient stiffness in the flapping direction and the pitching axis at $r/R = 0.700$ to mitigate the structural excitation.
Figure 4-32: UH-60A flight counter c9017 flapping moments, r/R = 0.3, 0.7
Figure 4-33: UH-60A flight counter c9017 torsional moments, r/R = 0.3, 0.7
4.1.6 Flight Counter c9020 Results

Counter c9020 is the most difficult to converge of the flight conditions simulated in this work. It is at a high altitude but has a lower thrust trim target than c9017. As with c9017, the condition is characterized by two dynamic stall events which occur on the retreating blade, although these are less severe in the present case.

4.1.6.1 Convergence

Convergence of the control settings for the c9020 flight case is shown in Figure 4-34. A non-monotonic evolution of these signals differentiates this case from the others which were considered. In order to achieve control system stability, the density in the CSD code had to be increased at iterations 4, 8, and 12. The flight condition was so near the peak of the lift curve for the rotor that the CFD-based airloads caused the CSD control system to seek collective angles higher than that which provided the maximum lift. As noted in the c9017 case, changes to the aerodynamics of the CSD model are allowed on the grounds that the final, converged solution contains only the CFD airloads. The density changes served in this case only to increase the maximum lift which could be achieved by the DYMORE control system.
Figure 4-34: Trim setting convergence during CFD/CSD coupling, UH-60A flight counter c9020

Figure 4-35: Control output divergence for c9020 based on DYMORE aerodynamics
The straight c9020 flight conditions would result in divergence of the control system outputs, based on DYMOROE internal aerodynamics. Figure 4-35 shows the behavior of the control outputs at a variety of different density values. As it happens, c9020 is near the McHugh Lifting Boundary [7], which is where the aerodynamics begin to betray the control system due to airfoil stall. When the control system tries to increase collective to get more thrust, the aerodynamics can’t provide it and instead the thrust goes down (according to the lift curve of the airfoils in the blade). Luckily the DYMOROE aerodynamics only serve as a variable set as described in section 2.3.2. They will be replaced with the U²NCLE airloads so the exact values of the airloads are not important. Therefore, any conditions which allow convergence on the DYMOROE end can be used to arrive at a nominally coupled result. Potsdam pointed this out in his description of the coupling process [4].

4.1.6.2 Airloads

Figure 4-36 displays the wake model results for the c9020 flight counter. Trailing vortices are convected downstream and away from the rotor before having much interaction with the rotor. This case is very similar to c9017 and BVI is expected to have a small or negligible effect.
Figures 4-37 to 4-42 present the aerodynamic normal force and pitching moments computed by the CFD code alongside the same quantity computed by DYMORE’s aerodynamics and flight test data. Flight test data for this case has been digitized from the results of Rajmohan [18]. Figure 4-37 and 4-38 show typical waveforms for inboard radial stations.

Prediction of the dynamic stall effect on pitching moment in Figure 4-40 matched flight test data quite well, although the event was predicted 10 to 15 degrees prematurely in the simulation. The normal force at this radial station matches the trend of the data well but has a slight offset. Overall, the simulation succeeded in matching the flight test data.
Figure 4-37: UH-60A flight counter c9020 airloads, r/R = 0.225
Figure 4-38: UH-60A flight counter c9020 airloads, r/R = 0.400
Figure 4-39: UH-60A flight counter c9020 airloads, r/R = 0.775
Figure 4-40: UH-60A flight counter c9020 airloads, $r/R = 0.875$
Figure 4-41: UH-60A flight counter c9020 airloads, r/R = 0.925
Figure 4-42: UH-60A flight counter c9020 airloads, r/R = 0.975
No structural load data is available from flight test data.

### 4.2 Bell 427 Loose Coupling Results

Bell Helicopter provided a DYMORE model of the Bell 427 helicopter rotor, which included a swashplate and pitch horn. This model was previously used by Morillo et al. to simulate the same case as is considered here [19] and is shown in Figure 4-43. The work reported here was initially presented by Sheng [51].

![Figure 4-43: Bell 427 DYMORE model layout, from Morillo et al. [19]](image)

Figure 4-43 shows the topology of a single blade in the DYMORE model, although four were present during the simulations. This model implements the controls through linear displacements of pushrods attached to the swashplate. There is more complexity to this model than is absolutely necessary for either loose or tight coupling analysis, but the model is higher fidelity and provides more information as well. Eighty-three airstations are present in each blade, concentrated towards the tip. Ten third order finite elements discretize the blade for the kinematic analysis.
4.2.1 Flight Condition

The Bell 427 main rotor performance is predicted for forward flight at 3,000 ft altitude. Table 4.2 lists the numbers input to the DYMORE model, in English units. Based on the rotational speed and incoming flow velocities, one can compute that reverse flow will occur on the retreating blade inboard of about 5.6 feet.

<table>
<thead>
<tr>
<th>Flight Condition</th>
<th>Flow Velocity (ft/s)</th>
<th>µ</th>
<th>$C_l/\sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>High speed forward flight</td>
<td>235</td>
<td>0.306</td>
<td>0.066</td>
</tr>
</tbody>
</table>

Loose coupling was performed manually for this case because it was done prior to the verification of the FSI code. Convergence was achieved in 15 iterations; one run of the CSD code took 1.5 hours of wall clock time, and one run of the CFD code occupied 5.5 hours of wall clock time running on 64 computer cores. Altogether, the coupling process took 105 hours of wall clock time.

4.2.2 Convergence

Figure 4-44 shows the convergence behavior of the loose coupling process for this case. There is some amount of unsteadiness in the control signals once they have flattened out. Periodic discontinuities in the CFD sectional airloads create aerodynamics will be slightly non-repetitive once the simulation has converged. This unsteadiness causes the minor changes with each iteration which is visible in this and other loose coupling simulations.
4.2.3 Wake Model Results

Figure 4-45 displays the wake structure for the Bell 427 forward flight case. Freestream flow is coming from right to left (from the pilot’s view) across the vehicle, and the convection of the wake structure due to this is evident. This wake shows only a slight amount of tip roll-up beginning to occur when the wake is about 4 rotor revolutions old. Based on viewing this wake structure, no BVI effects are expected to be seen in the coupling results.
4.2.4 Structural Loads

Structural loads at different radial positions along the blade length are shown in Figures 4-46 to 4-49. The data shown here has been filtered through a Fourier transform, limiting signal frequencies in the final solution to the first 10/rev. The same effect which introduced error in the UH-60A structural load results is present in this simulation, and the structural loads of the coupled analysis show no general improvement as compared to the DYMORE lifting line values.
Figure 4-46: Bell 427 beam bending moment, r/R = 0.43

Figure 4-47: Bell 427 chord bending moment, r/R = 0.42
Figure 4-48: Bell 427 torsional moment, \( r/R = 0.43 \)

Figure 4-49: Bell 427 pitch link load
4.3 Bell 427 Tight Coupling Results

The forward flight case was analyzed using tight coupling so that the results could be compared with loose coupling results. These two sets of data should be nearly identical if the tight coupling process was carried out correctly. All results in this tight coupling section were obtained with an error in the reference frame, and as such are incorrect and should not be compared to the results in Section 4.2.4, which use the correct reference frame. The intent of this section can still be achieved by observing that the loose and tight coupling processes produce very similar results, which is possible only when the loose coupling process is performing as intended. Secondly, the added value of a tight coupling analysis is observed through the generation of results for an unsteady flight case.

4.3.1 Forward Flight

Prior to considering the maneuvering flight case, the forward flight case first analyzed using loose coupling was simulated using the tight coupling technique. Having both of these results serves two important functions. First, the quality of the loose coupling process can be assessed by how closely the tight coupling results match those of the loose. Second, the tight coupling forward flight results provide a sensible baseline for comparison against the doublet flight.

The forward flight loose coupling and tight coupling results are shown in Figure 4-50 to 4-53. The structural loads show only slight magnitude changes between the two coupling results.
Figure 4-50: Bell 427 beam bending moment, r/R = 0.43, LC vs. TC

Figure 4-51: Bell 427 chord bending moment, r/R = 0.43, LC vs. TC
Figure 4-52: Bell 427 torsional moment, r/R = 0.43, LC vs. TC

Figure 4-53: Bell 427 pitch link load, forward flight LC vs. TC
4.3.2 Maneuver

Nygaard [45] considered the UH-60A with a +/- 2 degree pitch doublet imposed on top of the forward flight trim settings. This procedure was replicated here for the Bell 427 rotor and was first presented by Sheng [52]. A user-prescribed pitch profile was input to DYMORE as a deviation from the nominal trim settings computed by loose coupling. Figures 4-54 to 4-56 show the transient pitch responses of all four blades over two rotor revolutions; these results are obtained by computing the difference between the predicted pitch during a maneuvering case and a steady case. The maneuver case was initiated by first restarting the simulation from the end point of the converged loose coupling analysis. Ninety-five time steps were run, and then the doublet was implemented starting when blade 1 was at 96 degrees azimuth. The first change was a -2 degree delta to the collective. The pitch change occurred over the course of 2 time steps (1 degree collective per degree azimuth) and was held for a duration of 83 steps before undergoing another 1 degree pitch per degree azimuth change which brought it to a +2 degree collective.

A key feature of tight coupling is its ability to provide insights to the non-periodic, transient phenomena occurring during a maneuver. As the blades are at different azimuth positions, each one experiences a unique flow environment. Differences in flow speed and angle, as well as blade-vortex interactions, naturally lead each blade to have a unique transient response following the doublet. For example, as shown in Figure 4-54, a certain amount of waveform lag and warping occurs even at blade positions close to the root, so that the actual pitch profile is different from what was intended. This difference is attributable to joint stiffness as well as the blade inertia and airloads. As the position moves away from the blade root to tip, one notices a delay in the onset of the doublet profile. It is most pronounced at the blade tip, as shown in Figure 4-56, and this is due to blade flexibility. As expected, the
further out on the blade, the greater the deviation from the applied doublet profile. Figure 4-57 compares the first blade deflections at the three different radial positions. The pitch deviations for a given blade share a waveform and reduce in magnitude as the simulation proceeds, with a light damping shown. The simulation time was not sufficient to see the excitation damped completely.

Figure 4-54: Blade Pitch Deflection at the pitch horn station
Figure 4-55: Blade Pitch Deflection at r/R = 0.56

Figure 4-56: Blade Pitch Deflection at r/R = 0.99
Passing through the 90 degree azimuth position tended to excite the blades as can be observed in Figure 4-56. Following the doublet at $r/R = 0.99$, blade 3, then 2, then 1, then 4 experience deflection perturbations which correspond to the blades moving through the 90 degree position. The effect increased the pitch deflection as compared to the non-doublet case for all blades.

Figures 4-59 to 4-61 compare forward flight test data with the tight coupling structural loads which were established in the previous section to be quite similar to the loose coupling structural loads. The third line shown in each graph is the structural loading experienced by the blade during and after the application of the pitching doublet. It should be noted that the x axis in these figures is the azimuth position of blade 1, and it covers two full revolutions in order to observe the blade transient behavior. Flight test data and the forward flight tight coupling analysis are the same between the first and second revolutions; they are only present during the
second revolution for comparison purposes.

As expected, the tight coupling structural load results are identical up to 90 degrees, which is when the doublet begins. The chord bending moment of Figure 4-59 shows a higher degree of load variation about roughly the same mean value as a result of the doublet. That is to say, the moment after the doublet begins is higher in magnitude than that of the forward flight case, but then swings to being lower than it after about 1 revolution. This indicates that the doublet has excited a long-period oscillation corresponding to the chord bending moment. Figure 4-58 shows a beam bending moment which is little changed due to the doublet.

Blade torsional moment and pitch link load, Figures 4-60 and 4-61, respectively, characterize the twisting felt in the blade due to the doublet. The axis about which the doublet acts is the primary axis affecting these loads, and the results bear this out. In the pitch link load and the torsional moment to a lesser extent, notable peaks in the loads occur when blade 1 is at 90 degrees, 180 degrees, and 270 degrees azimuth: the three locations when the control system is applying doublet-related changes to the rotor collective.

Figure 4-58: Bell 427 tight coupling beam bending moment, steady flight vs. doublet
Figure 4-59: Bell 427 tight coupling chord bending moment, steady flight vs. doublet

Figure 4-60: Bell 427 tight coupling torsional moment, steady vs. doublet
Figure 4-61: Bell 427 tight coupling pitch link load, steady vs. doublet

The differences in rotor behavior between forward flight and doublet has been seen in this section to persist for two rotor revolutions or longer. Results of this section demonstrate the unique capabilities of tight coupling to include non-periodic behavior in the analysis. Peak load magnitudes occur in the pitch link load during the change in collective angle from the doublet. Predicting these load magnitudes accurately provides the design engineer a basis for sizing the components correctly without having to perform expensive prototype testing.
Chapter 5

Conclusion

Rotorcraft performance prediction is a challenging task which will benefit from continued reduction in computing costs and further study of the computational methods used. A variety of flight conditions were studied in this work, each with slightly different physics and/or coupling process requirements. The loose coupling analysis of the four UH-60A flight conditions included transonic flow, high levels of Blade-Vortex Interaction (BVI), dynamic stall, and locally reversed flow. Two of the flight conditions tested the ability of the loose coupling procedure to be convergent. The Bell 427 tight coupling analysis demonstrated the unique benefits of that kind of non-periodic analysis to simulate transient phenomena.

The development of the Fluid-Structure Interface (FSI) code has enabled both the loose coupling and tight coupling processes. Through the use of this code, analyst workload is greatly reduced and the processes are carried out reliably. This tool will continue to be an asset to the analysis team as more analyses are performed and features are added.

The following sections list conclusions which result from this work, as well as recommendations for the direction of future work.
5.1 Conclusions

A summary of the findings of this work is given below.

1. DYMORE internal aerodynamic models, while sufficient for purposes early in the design process, fail to capture salient flow features such as Blade-Vortex Interaction, dynamic stall, reverse flow over blades, transonic effects, and multi-structure flow interference as demonstrated in a recent paper by Zhao [53].

2. Coupling DYMORE with a CFD code is an effective technique to improve aerodynamic results, particularly in cases of reverse flow, transonic flow, or dynamic stall events.

3. An automated CFD/CSD loose coupling script was demonstrated which reliably executes the coupling process while improving the efficiency of computer resource use and reducing the workload on the analyst.

4. Using a single-trailer free wake model, simulations with less BVI (c8534) produced results that matched flight test data better than the case with more BVI (c8513), suggesting improvements could be made with different wake model settings or possibly with in-grid wake capturing.

5. Flight counter c9017 and c9020 were on the edge of the flight envelope, and DYMORE was unable to reach a converged control system result for these two cases due to inaccuracies in its own internal aerodynamic models. Using aerodynamics from U²NCLE allowed these cases to be properly resolved as being within the level flight envelope.

6. The use of an artificially high density in the CSD code to improve the robustness of the control system was demonstrated as a valid method for adding stability to the CFD/CSD loose coupling process with no adverse effect on results.
7. An automated CFD/CSD tight coupling code was demonstrated.

8. Tight coupling was demonstrated as a simulation technique with the ability to resolve non-periodic rotor behavior.

9. The CFD/CSD coupling process is between 1,000 and 10,000 times more computationally expensive than the use of CSD alone. While it was clearly shown to add value in the cases considered, its place in the overall design scheme is to be weighed against the difference in cost.

5.2 Future Work

Work to advance the maturity of the CFD/CSD coupling process at the University of Toledo can continue to be performed with two main thrusts. Firstly, the results presented here have room for improvement, and work can still be done to tailor individual CFD and CSD results to the coupling process and produce improved results. Secondly, the automated Fluid-Structure Interface and associated toolkit can continue to be used and expanded to eventually provide a mature set of generic tools which add value to the CFD and CSD analysis processes. Specific goals are enumerated below:

1. Improve the implementation of airstation deflection to match the Euler 3-2-1 orientation scheme.

2. Study loose coupling with CFD sectional forces which are free from spurious excitations. The CFD code can be modified to achieve this, or a data processing step can be added to the coupling process.

3. Add capability to implement higher harmonic control simulation.
4. Extending the code coupling to include vehicle flight mechanics. Separately, Bhagwat [54] and Bauchau [44] have extended the coupling capabilities of RCAS and DYMORE, respectively, to include flight conditions in addition to the load and deflection coupling. This amounts to the vehicle flight mechanics as computed by the comprehensive code being shared with the CFD code, and the farfield conditions (i.e. flow direction) being changed accordingly.
References


141


Appendix A

Fluid-Structure Interface main script

# This routine performs loose coupling
#
# Data about the coupling process (convergence, etc.) is stored in ./master/convergence.dat
#
# As a rule, never use os.system("rm -f *") or os.system("rm -rf *")). This
# is dangerous. Call out each file separately
#
import os
import os.path
import sys
import time
import datetime
import lc
import lc_run
import lc_check
import fsiSettings
import IOops
import fsiParse
import dataTypes as dt

# Initialize the main object
recFile = IOops.safeOpen("lc.record","w")
S = dt.simMain(recFile)
# Parse fsi.input
fsiParse.parseMainInput(S)

# Make sure to start in the correct folder: If it1 == 0, start working in # folders with index 0. Else, start working in folders with index it1 + 1
if (S.it1 != 0): S.setAdvance(1)
else: S.setAdvance(0)

# CHECK FOR THE NECESSARY DIRECTORIES AND ROUTINES
lc_check.dir_check(S)
# This was off before I commented everything out
#lc_check.exe_check()

# PRINT THE BANNER, DO SOME MISCELLANEOUS CHECKS/FOLDER MAKING
linecount = lc_run.initializer(S)
fsiParse.writeMainObjectValues(S)
S.recFile.close()

# THE MAIN LOOP
for i in range( S.it1+S.advance, S.it1 + S.its + S.advance):
    # STEP 1A: RUN DYMORE
    lc_run.printLoopLabel(i)
lc_run.runDymore(i)
    lc_run.checkAbort()
# STEP 1B: SAVE RESULTS TO master/ AND UPDATE CONVERGENCE DATA FILES

lc_run.compileConvergenceData(i)

# STEP 2: Translate Deflections.dat into rotor_deflection.X.dat

lc_run.translateDeflections(i)

# STEP 3: RUN U2NCLE

lc_run.runU2ncle(i,linecount)

lc_run.checkAbort()

# STEP 4: COMPUTE DELTA AIRLOADS

lc_run.computeDeltaAirloads(i)

# STEP 5: PREPARE FOR THE NEXT ITERATION

if(i == (fsiSettings.its-1)):
    break # leave this for loop, don’t create new files so there is no confusion by the user
else:
    lc_run.prepareForNextIteration(i)

print("END")
Appendix B

Example Input File: fsi.input

# FSI SETTINGS:
#================================
# Type of coupling. "LOOSE" or "TIGHT" are the options. If loose is selected, then all the parameters under
# the corresponding heading are required. Else, the headings under TIGHT COUPLING are required

cpl_type = LOOSE

dymoreStaticFileName = uh60bls.dym
dymoreDynamicFileName = uh60bl.dym

# The command to execute or submit the U2NCLE job:
#u2nclineFileName = qsub UH60.sh
u2nclineFileName = UH60.sh
runtpe = qsub

# The angular velocity of the rotor, normalized for U2NCLE
omega = 4.260660907

built_in_angle = 8.77

# LOOSE COUPLING SETTINGS
# The number of the seed iteration, 0 will start from the beginning with a special
first run of the cfd code
it1 = 1

# The number of coupling iterations to run
its = 2

# The number of the first iteration to use the dAP in the dAI calculation
firstDAP = 2

# Flag to overwrite master/convergence: 1 = Yes, 0 = No
owflag = 1

# The number of CFD steps to run in the start up process (iteration 0, if applicable)
cs = 720

# The number of blades, and airstations on each blade
blades = 1
airstations = 81

# Normalization factors used by U2NCLE to convert nondimensional force and mo-
moment to dimensional:
ffac = 22588.0
mfac = 416753.0

# The trim target values dymore uses:
thrust_trim = 5507.56
roll_trim = -195.96
pitch_trim = -3062.94

# Time to wait for u2ncle start up (in minutes):
start_up_time = 5

# Time to wait before checking global_resid.out (in minutes):
check_resid_every = 15
# Memory Cap, the percentage of memory filled which will cause the code to abort
mem_cap = 80

# TIGHT COUPLING SETTINGS
#================================
# The number of the first iteration, 0 will start from the beginning with a special
first run of the cfd code