Innovative Strategies for Rotary-Wing Coupled Aeroelastic Simulations

Final Technical Report
August 15, 2007 – February 14, 2011

Submitted by:

Marilyn J. Smith and Olivier A. Bauchau
School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0150

Grant Number: NNX07AP43A
Georgia Tech Research Corporation

Contract Monitor:

Ms. Elizabeth Lee-Rausch
Computational AeroSciences Branch
NASA Langley Research Center
Hampton, VA

May 14, 2011

Table of Contents

1. Technical Objectives 6
2. Background 7
   - 2.1. CFD-CSD Rotorcraft Aeroelasticity 7
   - 2.2. Aerodynamics 7
   - 2.3. Structural Dynamics 7
   - 2.4. Multi-body Dynamics 8
   - 2.5. CFD-CSD Coupling 10
   - 2.6. Conventional Trim Procedures 12
   - 2.7. Control Theory 13
3. Integrated Framework for Loose and Tight CFD-CSD Coupling 16
4. Steady-State Trim Algorithm 23
5. Kriging Control Algorithm 29
   - 5.1. Kriging meta-model with optimized free parameters 30
   - 5.2. Test and optimization of the kriging implementation 31
6. Optimization of the Tight Coupling CFD/CSD Process 37
7. CFD/CSD Tight Coupling Demonstrations 42
   - 7.1. Grid Descriptions 42
   - 7.2. Level Flight Trim Demonstrations 45
   - 7.3. Maneuvering Flight Demonstration 70
   - 7.4. Tight Coupled Application: Stability Analysis 74
8. Parallelization and Portability of Codes 81
9. Documentation and Technology Transfer 83
10. Conclusions 85
11. References 86

List of Figures

Figure 1: Detailed Multi-Body Representation Of A Rotor System. At Right, A Typical Articulated Blade. At Left A Bearingless Blade Design. 9
Figure 2: Trim Of Hart1 Baseline Case With Loosely-Coupled Overflow-Dymore With An Autopilot Trim Model. 11
Figure 3: CFD/CSD Loose And Tight Coupling flowcharts Demonstrating Fun3d And Dymore Coupling. 16
Figure 4 Python CFD/CSDCoupling Framework 19
Figure 5: Trimmed Thrust And Moments With Errors For The Jacobian Computed With Predetermined Control Settings. The Horizontal Lines In The Figures On The Left Are The Target Values. 25
Figure 6: Trimmed Swash Plate Displacement And Tilt With Errors For The Jacobian Computed With Predetermined Control Settings. The Horizontal Lines In The Figures On The Left Are The Autopilot Solutions. 26
Figure 7: Trimmed Thrust And Moments, And Errors With Errors For The Jacobian Computed With Control Settings At Zero. The Horizontal Lines In The Figures On The Left Are The Target Values. 27
Figure 8: Trimmed Swash Plate Displacement And Tilt With Errors For The Jacobian Computed With Control Settings At Zero. The Horizontal Lines In The Figures On The Left Are The Autopilot Solutions. 28
Figure 9: CFD/CSD-Based Kriging Algorithm Training Using Thrust As The Independent Parameter 32
Figure 10: CFD/CSD-Based Kriging Algorithm Training Using Rolling Moment As The Independent Parameter 32
Figure 11: CFD/CSD -Based Kriging Algorithm Training Using Pitching Moment As The Independent Parameter 33
Figure 12: Comparison Of Kriging Meta-Models Predictions With Different Free Parameters. 36
Figure 14: Summary Of Thrust Errors During Different Tight Coupling Starts From Loose Coupling. 37
Figure 15: Revolutions Needed To Reach Thrust Convergence (1% Error) Based On Loose Coupling Initialization. 38
Figure 16: Example Of The Large Transients That Occur When Kriging Is Applied Immediately Upon Transfer From Loose To Tight Coupling (At Revolution 1). 39
Figure 17. Tight Coupling Trim May Be Achieved For Some Cases With A Training Database Provided From A Comprehensive Code. 40
Figure 18: Pitch Control History, Varying Length Of Transition Between Low-Order And CFD Loads. (From Reference 60) 40
Figure 19: Hub Loads History, Varying Length Of Transition Between Low-Order And CFD Loads. (From Reference 60) 41
Figure 20. Uh60-A Cfd Structured Coarse Mesh. 43
Figure 21: Illustrations Of The 13.6 Million Node Composite Mesh Used For Hart-Ii Simulations. 44
Figure 22: Comparison Of The Influence Of The Turbulence Model Predictions On Pitching Moment At A Typical Radial Section. 46
Figure 23. Schematic Of The Uh60-A Rotor System. 47
Figure 24: UH60-A Airloads Computed With Overflow-Dymore Using The Coarse CFD Grid. 48
Figure 25: Inboard UH60-A Structural Moments Computed With Overflow-Dymore Using The Coarse CFD Grid. 51

Figure 26: Outboard Uh60-A Structural Moments Computed With Overflow-Dymore Using The Coarse Cfd Grid 52
Figure 27: Uh60-A Tip Deflections Computed With Overflow-Dymore Using The Coarse CFD Grid. 53
Figure 28: Uh60-A Airloads Computed With Overflow-Dymore Using Refined Cfd Grids. 54
Figure 29: C9017 Convergence With Loose And Tight Coupling Using Overflow-Dymore On The Coarse Grid 57
Figure 30: C9017 Airloads Predictions For Loose And Tight Coupling Using Overflow-Dymore On The Coarse Grid With The HRLES Turbulence Model. 58
Figure 31: Hub Of The Hart Dymore Model. 61
Figure 32: Convergence Of The 1st Torsion Frequency With The Number Of Finite Elements In The Blade Beam 61
Figure 33: Fan Plot For The Hart Dymore Model 62
Figure 34: First Six Mode Shapes For The Hart Dymore Model 63
Figure 35: Convergence Of Blade Pitch Controls With Loose Coupling Iterations 64
Figure 36: Convergence Of Airloads At 87% Span For The Loosely-Coupled Trim Phase. For Measured Pitching Moment, The Mean Value Is -0.00258; For Iteration 5, The Mean Pitching Moment Is -0.00513. 65
Figure 37: Comparison Of Structural Moments Computed Using Fun3d/Dymore And Fun3d/Camrad-Ii In Reference 67. Mean Values Removed In Second Column Are Listed In Table 10. Moment Is Negative For Fun3d Results. 66
Figure 38: Comparison Of Airloads At 87% Span Computed Using Fun3d/Dymore And Fun3d/Camrad-Ii In Reference 67. Mean Pitching Moments Are -0.00258, -0.00424, And -0.00513 For Measured, Fun3d/Camrad-Ii, And Fun3d/Dymore, Respectively 67
Figure 39: Tip Deflections Predicted In Tight Coupling And The Final Iteration Of Loose Coupling. Bars On The Measured Data Indicate Blade-To-Blade Variation 68
Figure 40: Sectional Normal Force And Pitching Moment 87% Span, Comparing Loose And Tight Coupling. 68
Figure 41: Q-Criterion Iso-Surfaces. Q=0.0075 And Is Non-Dimensionalized 69
Figure 42: UH60-A C11029 Tightly-Coupled Airloads Using Overflow-Dymore On The Coarse Grid 71
Figure 43: Uh60-A C11029 Tightly-Coupled Structural Moments Using Overflow-Dymore On The Coarse Grid 73
Figure 44: Singular Values Arranged In Decreasing Order For The 3.5 Revolution Analysis. 77
Figure 45. Comparison Of The Original And Reconstructed Signals For The Four Signal Locations. 78
Figure 46: Predicted Frequencies For The Four Rotor Modes. 78
Figure 47: Predicted Damping Ratios For The Four Rotor Modes 79
Figure 48: Blade Response At Maximum Perturbation Location. 79
Figure 49: UH60-A CFD Mach Number Variation During Blade Perturbation 80

List of Tables

Table 1: Metrics Of Success 6
Table 2: Comparison Of The Controls Estimated From Kriging To The Converged (Iteration 5) Loose Coupling Of UH60-A High Speed Flight Case, 8534 33
Table 3: Comparison Of The First Iteration Trim Characteristics Estimated From Kriging To The Converged (Iteration 5) Loose Coupling Of UH60-A High Speed Flight Case, 8534 34
Table 4: Comparison Of Converged Loads For Csd Alone And CFD/CSD Loose Coupling Using Controls Estimated With And Without The Kriging Algorithm. 34
Table 5: Tight Coupling Requirements To Reach Convergence Using Kriging To Trim During Tight Coupling. 38
Table 6: UH60-A Test Cases 42
Table 7: Dimensions Of Each Component Of The Composite Mesh. All Cells Are Tetrahedra. 43
Table 8: Geometric Parameters For The Hart-Ii Rotor. 60
Table 9: Converged Blade Pitch Controls, In Degrees. 64
Table 10: Mean Values Subtracted From Structural Moments In Second Column Of Figure 37. 67
Table 11: Characteristics Of The Stability Approaches. 77
Table 12: Damping Ratios For The Four Rotor Modes. 77

1. Technical Objectives

This effort was to develop an innovative framework to couple Computational Fluid Dynamics (CFD) and Computational Structural Dynamics (CSD) codes to replace current ad hoc coupling and trim approaches. A computational control (CC) module will be developed using system identification techniques that updates trim concurrently to the more expensive CFD parallel simulations. A kriging-based controller has been developed to create a computational simulator that more accurately approximates true flight and to perform rapid trim during CFD/CSD simulations. The process included further enhancements in the guise of a python-based framework and optimization of CFD analyses for accurate blade loads. The modifications were to be demonstrated on isolated rotors for steady and maneuvering flight. This report presents a subset of these results that demonstrate the efficacy of the approaches developed. Detailed development and analysis of the methods can be found in the papers and theses listed in Section 9 of this report.

There were eight metrics to determine the technical success of the project, as denoted in Table 1. All eight metrics were met or exceeded during the project.

.

Table 1: Metrics of Success

Metric	Measure of Success	Project Outcome
1 - New Python-based Framework allows plug and play of CFD-CSD-CAA-CC modules	Demonstration of framework on 2+ machines using a combination of codes such as OF, FUN3D, Dymore, RCAS, PSU-WOPWOP	The Python framework has been successfully demonstrated on local Georgia Tech and NAS HPC computers.
2 – Enable automated OF+Dymore tight coupling simulations.	Show identical results for loose and tight coupling for simple cases, such as UH60-A, C8534.	Results for the UH60-A 8534 and 9017 cases, and the HART-II baseline tests are comparable using loose and tight coupling.
3 – Implement simple, autopilot based, trimming algorithm for tight coupling.	Demonstrate the ability of tight coupling to deal with more difficult cases (e.g., UH60-A C9017).	The tight coupling has been successfully demonstrated for cases with dynamic stall and BVI, as well as for stability analysis applications
4 - New system identification based trim algorithm provides physics-based control of level flight & steady turns	Demonstration of 40-50% clock time reduction in tightly-coupled simulations when using the new trimming strategy.	Optimization tests indicate that tightly-coupled simulations with kriging-based trim run time requirement is roughly comparable to loosely-coupled simulations, which significantly exceeds the estimated savings.
5 - Neural-Network-based CC module provides additional physics-based trim and motion for CFD-CSD simulations	Comparison with baseline and Task 2 results show more robust trimming performance or pilot-like response of simulations or both	The new steady-state trimmer and kriging controller are robust and respond within 1-3 revolutions to control changes.
6 - Optimization of CFD methodologies to provide accurate airloads while minimizing CPU time	Improve UH60-A loads compared with flight test and baseline runs. Discrepancies will be quantified. Increase computation efficiency by 20-30%.	Optimization tests indicate that a tightly-coupled run started from a short (1/4-1/2 rev) loose coupled simulation.
7 - Portability and parallelization of codes from Beowulf cluster to HPC machines	Demonstrate success of all modules to parallelize and run on a minimum of 4 different compilers and/or computational platforms.	The parallelized framework has been ported to 4 and 6 different platforms, for FUN3D and OVERFLOW, respectively.
8 - Successful transfer of research to NASA and R/C industry	Incorporation of mods into the FUN3D and OF distributions; at least 2 conference & journal papers	Modifications provided to NASA; new tightly-coupled code being used by Bell and provided to Sikorsky; 4 journal papers; contributed to 2 PhD theses.

2. Background

2.1. CFD-CSD Rotorcraft Aeroelasticity

Accurate simulations of rotorcraft flow fields are typically much more difficult to obtain due to the strong nonlinearities in the tightly-coupled fluid mechanics, structural dynamics and controls of the problem. As the rotor blades complete each revolution, strong variations in Mach regime are encountered, inducing complex elastic blade deformations in addition to their rigid motions. The strong wake system may also remain in the region of the vehicle, adding complexity to the blade loading as well as unsteady loads on the fuselage. These unsteady phenomena impact the intricate control system used to trim the rotor in level flight or to control maneuvers.

These three components of the aeroelastic system: aerodynamics, structural dynamics and controls, require dramatically different methods of numerical simulation. A computational method that resolves all three components using the same numerical scheme is not truly practical, not only due to the different methodologies, but also because of the large base of excellent codes that exist within each separate field. Instead, modeling techniques have relied on the coupling of each individual analysis method to create multidisciplinary analyses.

2.2. Aerodynamics

Computational fluid dynamics (CFD) methods, based on the solution of the Reynolds-averaged Navier-Stokes (RANS) equations are utilized to provide the aerodynamic response of the rotor. Strawn et al[1] provides a review of time-accurate RANS-based CFD analysis for rotary-wing applications. Wake resolution still relied solely on the CFD wake capturing; the resulting rotor loads resolution was poor due in part to computational memory limitations that precluded grid refinement in the wake region. Some hybrid methods attempt closure of the wake problem by applying the CFD method to capture the near-field wake and utilizing free- or prescribed-wake methods to resolve the far-wake, reducing the computational grid requirements, but at the expense of the physics. The recent DARPA Quiet Helicopter project[2] has resulted in efforts to utilize an adaptive Cartesian-based CFD method in the wake. Other efforts include the coupling CFD solvers with the vortex transport equations[3,4].

These advanced CFD methods are based on two general grid topologies, either structured or unstructured. The structured grid topology was the initial choice, since the natural ordering of the grid allows optimization of computational memory and time. However, creating a structured grid can be very time consuming, especially for very complex configurations. Since the most physically accurate analysis requires the time-accurate RANS solution of individual rotor blades, both rotating and stationary grids must be used for rotor-fuselage interactions, since the rotor is in a dynamic frame and the fuselage is in an Eulerian (static) frame. Overset methodologies have evolved to permit the solution of these frames simultaneously. The primary overset structured solver utilized in the U.S. rotorcraft industry is OVERFLOW[5].

2.3. Structural Dynamics

A set of specialized structural dynamics methodologies has been developed for rotor applications. These methods, known as comprehensive codes, were designed primarily to model the rotor system dynamics, but they have incorporated simplified aerodynamic and inflow models to make the analysis more complete. Current comprehensive codes are characterized by

the use of time-accurate analysis and finite element methods, as described in Kunz[6]. Among these are RCAS[7], UMARC[8], CAMRAD II[9], RDYNE[10], HOST[11], and Dymore[12]. Some of these methods apply finite elements based on the assumption of moderate blade deflections (UMARC, CAMRADII), while others are based on the geometrically exact formulation. The UMARC trim model solves 6 force balance equations for a target setting, while HOST, developed through the combined effort of ONERA, Eurocopter and DLR, incorporates trim analysis based on computations done on a harmonic representation of movement and internal state. Sikorsky's RDYNE trims using a minimum variance controller that uses the computed hub loads as the target values. Dymore, a multi-body code developed at Georgia Tech, uses geometrically exact, nonlinear beam and shell finite elements and uses an autopilot-based trimmer. For the elastic analysis, Dymore uses geometrically exact finite elements along the blades. Dymore is also capable of modeling composite elements based on the work of Berdichevsky[13], as extended to beams[14] and shells[15], which are very important for composite rotors and fuselage components.

2.4. Multi-body Dynamics

Multi-body dynamics analysis was originally developed as a tool for modeling mechanisms with simple tree-like topologies composed of rigid bodies, but has considerably evolved to the point where it can handle nonlinear flexible systems with arbitrary topologies. It is now widely used as a fundamental design tool in many areas of mechanical engineering. In the automotive industry, for instance, multi-body dynamics analysis is routinely used for optimizing vehicle ride qualities, a complex multidisciplinary task that involves the simulation of many different sub-components. Modern multi-body codes can deal with complex mechanisms of arbitrary topologies including sensors, actuators and controls, are interfaced with CAD solid modeling programs that allow one to directly import the problem geometry, and have sophisticated graphics, animation and post-processing features[16]. The success of multi-body dynamics analysis tools stems from their flexibility: a given mechanism can be modeled by an idealization process that identifies the mechanism components from within a large library of elements implemented in the code. Each element provides a basic functional building block, e.g., a rigid or flexible member, a hinge, a motor, etc. Assembling these elements, it is possible to construct a mathematical description of the mechanism at the required level of accuracy.

Figure 1 depicts the conceptual representation of a rotorcraft system as a flexible multi-body system[17]. The various mechanical components of the system are associated with the elements found in the library of typical multi-body analysis tools. This familiar control linkage configuration can be modeled using the elements: rigid bodies, used to model the lower and upper swash-plate components and scissors links, and beams for modeling the flexible shaft and pitch-link. These bodies are connected through standard mechanical joints: *revolute, universal, and spherical*.

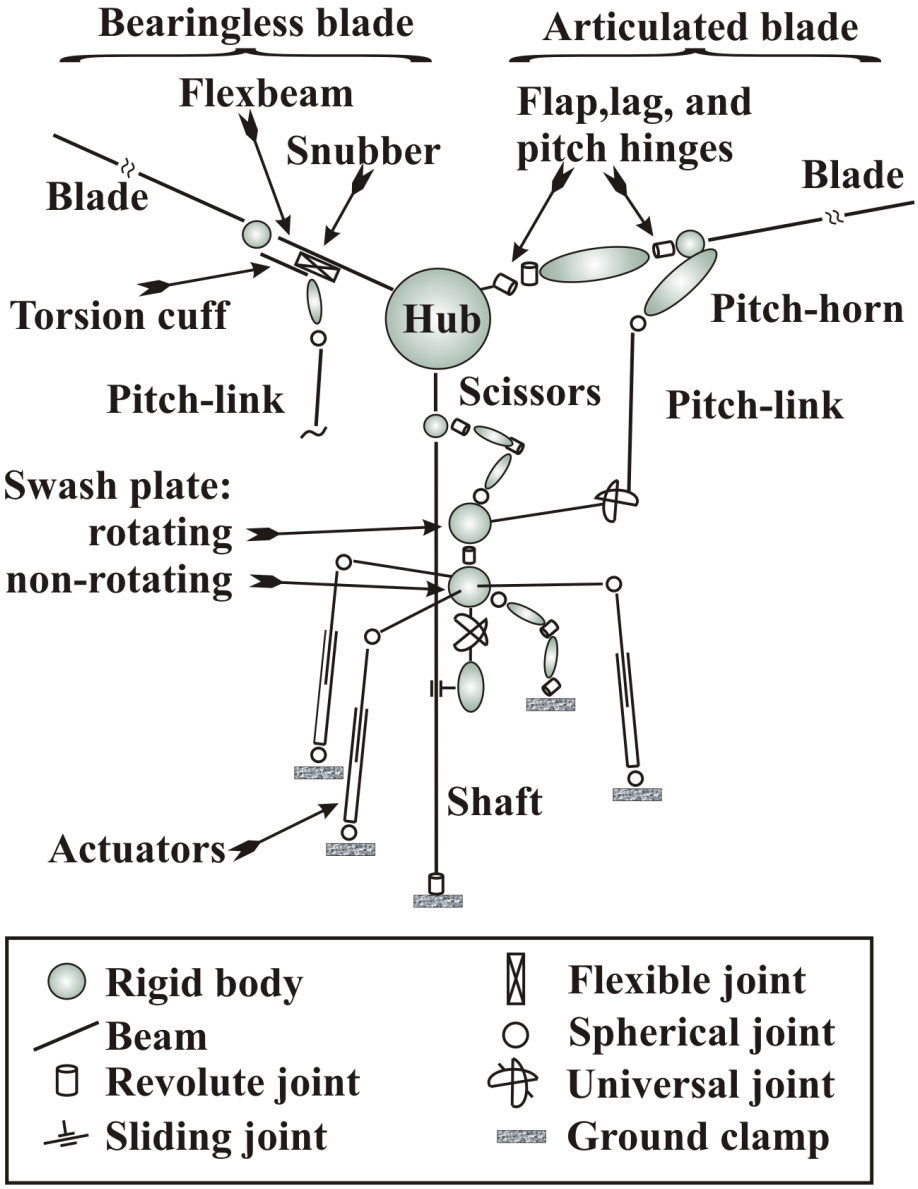


Figure 1: Detailed multi-body representation of a rotor system. At right, a typical articulated blade. At left a bearingless blade design.

In this approach, the formulations of structural elements are geometrically exact, *i.e.* they account for arbitrarily large displacements and finite rotations, but are limited to small strains. The equations of equilibrium are written in a Cartesian inertial frame. Constraints are modeled using the Lagrange multiplier technique. This leads to systems of equations that are highly sparse, although not of minimal size. This approach can treat arbitrarily complex topologies. Furthermore, because it is an extension of the finite element method to multi-body systems, the algorithms such as sparse solvers, and data structures developed for FEM analyses are directly applicable to the present approach. In contrast with most other rotorcraft dynamics codes, no modal reduction is performed; the analysis directly operates on the finite element model providing improved accuracy, particularly in the high frequency range.

Figure 1 also shows two different rotor configurations: a classical, fully articulated design on the right and a bearingless design on the left. The articulated blade is connected to the hub

through three revolute joints that model the flap, lag and pitch hinges. Possible offsets between these joints could be modeled by means of rigid or flexible bodies. The blade itself is modeled by an appropriate beam element that should account for the inertial and elastic couplings that arise from the use of composite materials[18]. The bearingless design is a multiple load path configuration, involving a flex-beam and a torsion cuff assembled in parallel and connected by a snubber. It is important to note that the two designs, fully articulated or bearingless, can be modeled by assembling different sets of elements from the multi-body library of elements. There is no need to derive and validate two different sets of equations for the two configurations.

2.5. CFD-CSD Coupling

In the 1980's, full-potential aerodynamic methods were coupled with comprehensive codes[19], however poor moment correlations have to date limited the usefulness of these coupled methods. Early rotor aeroelasticity simulations with RANS occurred in the 1990's when Smith[20], as well as Bauchau and Ahmed[21] coupled CFD methods with nonlinear computational structural dynamics (CSD) methods. During the early 2000's, interest in CFD-CSD coupling revived and a number of more advanced CFD methods have been coupled with comprehensive codes, as discussed by Datta et al.[22]. NASA rotary-wing codes have also been coupled with these comprehensive codes, specifically OVERFLOW has been coupled with several comprehensive solvers, including CAMRADII[23], RCAS[24] and with Dymore by the co-PIs[25]. An overset methodology has been recently implemented and demonstrated in FUN3D[26,27] at Georgia Tech, and overset CFD-CSD coupling collaborations between Prof. Smith's group (with Dymore) and the NASA FUN3D development group (with CAMRAD II) are underway.

The de facto standard for coupling in the aforementioned instances has to date focused what is known as "loose coupling." In loose coupling, data is exchanged between the CFD and CSD codes between each rotor revolution, as soon as the blade loads have become periodic. This data exchange is termed "an iteration." To accelerate the process, a rotor revolution may consist of a partial azimuthal sweep of each rotor blade, appended to form an entire revolution if the flow remains "fairly periodic."[1] For example, with a four-bladed rotor, a revolution can be formed from a quarter-revolution simulation by appending each blade's motion to form the full 360^{o} azimuthal sweep. Blade loads are provided to the comprehensive code to compute the blade deflections and any control changes needed to meet user preset trim convergence criteria (e.g., thrust and hub moments). These deflections and control changes are provided back to the RANS code to compute a new set of periodic blade loads. Data are passed in this fashion until a converged (trimmed) solution has been obtained. Trimming is achieved via the algorithm resident within the comprehensive code, using what has become known as the delta airloads approach[29]. This approach determines the difference between the CFD and comprehensive code airloads, which are typically resolved using lower-fidelity aerodynamics. This delta is frozen and added to the changing airloads of the comprehensive code as part of the trimming process.

[1] "fairly periodic" in this sense is interpreted by the user based on their past experience with the coupled code combinations. Users have reported different levels of success using this approach.

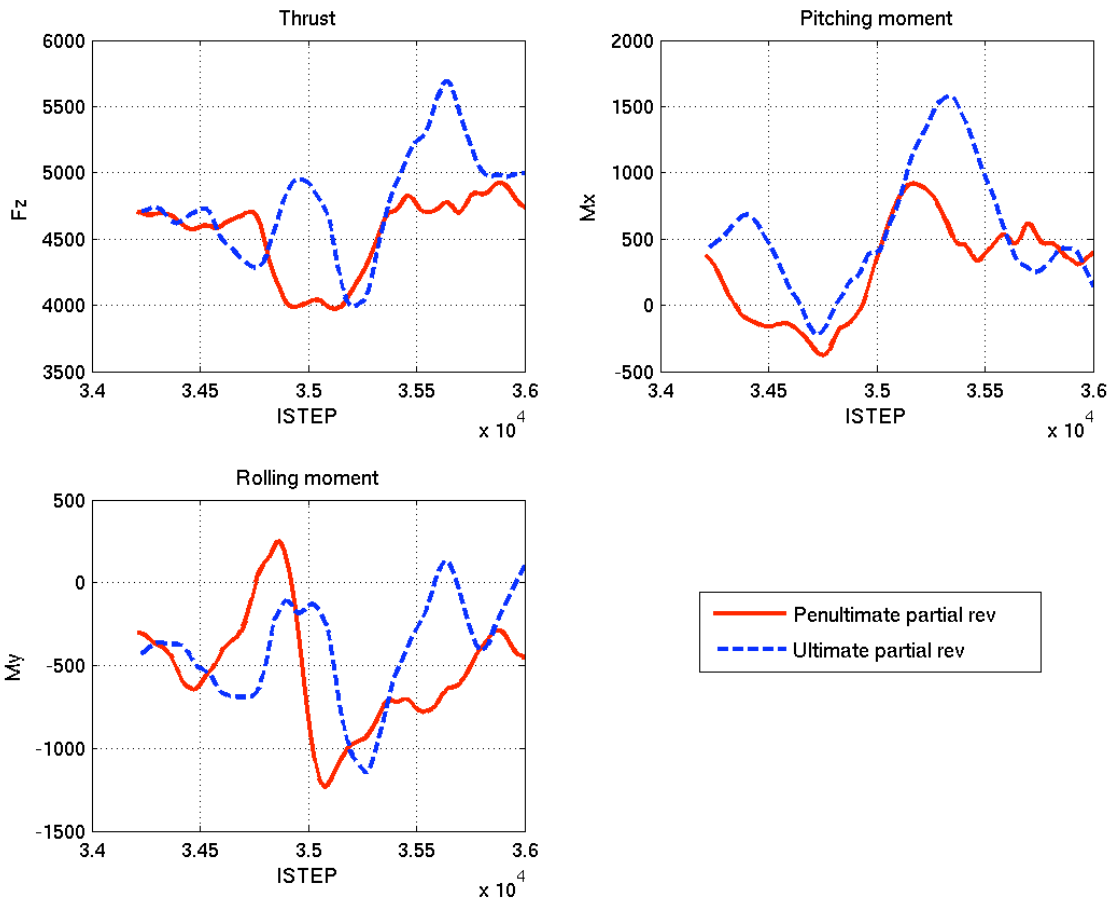


Figure 2: Trim of HART1 baseline case with loosely-coupled OVERFLOW-Dymore with an autopilot trim model.

Loose coupling has been improvised as a technique to improve the computational time required to obtain a trimmed solution for level flight (or steady turns). While in theory, for a quasi-steady simulation, the results from a loose coupling simulation should be identical to a tight coupling simulation, in practice several limitations of this approach have been encountered with various combinations of CFD-CSD solvers[28]. Problems are typically encountered in the loose coupling approach when runs near the edge of the flight envelope (e.g., Case 9017 for the UH60) or when there is significant wake interaction (e.g., Case 8513 for the UH60 and the HARTI and II baseline cases). If the initial deflections computed by the lower-fidelity aerodynamics solver and the comprehensive code are not close to what results in trim, then convergence problems may also be encountered. As seen in Figure 2, for the HARTI baseline case, the deflections predicted after several loose-coupled iterations indicate that the deflections are suddenly tracking away from the conditions needed for trim. Convergence for a loose coupling approach is achieved when the CFD-CSD iterations no longer significantly change the parameters used to track trim (e.g., thrust and hub moments). At this point, the simulation is halted and performance parameters are computed. In some instances, this criterion for trim convergence may not be appropriate. During the DARPA Quiet Helicopter program, Smith and Bauchau discovered that when using higher-fidelity turbulence models (LES-based), further

coupling iterations resulted in solutions driven away from what appeared to be convergence as the physical flow field was not truly periodic.

Tight coupling is the correct way to achieve a physics-based aeroelastic rotor simulation, no matter what the flight condition. Traditionally, tight coupling has been designated as the approach when data between the CFD and CSD codes are passed at each CFD or CSD time step. This approach has been designated for static aeroelastic analyses in fixed-wing simulations, such as level flight, as no control feedback is used. However, for rotary-wing applications, trim is an integral part of even level flight or hover to achieve the correct thrust and zero the hub moments. The integration of a trim control in tightly-coupled analyses has to date been somewhat ad hoc, based on modification of the loosely-coupled approach and very specific test cases. Nygaard et al.[29] first demonstrated an OVERFLOW-2+RCAS coupling for transient response. They apply loose coupling to reach a trimmed state before the maneuver, use tight coupling for about a ¼ revolution, and then drive the code using user-defined control changes. Tight coupling is achieved by sending loads and deflections between the CFD and CSD codes at each time step; no control feedback appears to be utilized during the last ¼ revolution, and there is no feedback to the controller during the maneuver. Others[30] simply update the trim after each revolution once periodicity is obtained (or each ¼ for a four-bladed rotor), resulting in the requirement that the rotor be run for 15-20 revolutions to obtain a steady, level flight convergence.

2.6. Conventional Trim Procedures

The numerical trimming of a rotor system is a complex problem. Although many trim targets can be selected, a typical problem statement is as follows: find the collective and cyclic inputs to the rotor that will generate a given thrust and moments on the rotor. Peters and Barwey[31] have discussed a general theory of rotorcraft trim and reviewed the many algorithms that have been used for this purpose. However, when using very complex models such as finite-element-based multi-body dynamics for CSD modeling coupled to CFD tools for the computation of airloads, far fewer approaches to trim are practical, some of which have been mentioned in a previous section on CSD methods. The recommended approach[31] is the autopilot procedure, which is a very simple control algorithm. The first step of the procedure is to identify the trim matrix, a linearized relationship between the inputs and output of the system, i.e., between the collective and cyclic control inputs and the resulting rotor thrust and moments. In the second step, this trim matrix is used as a simplified plant model to drive the actual rotor thrust and moments to their target values.

The autopilot procedure presents the following advantages. It can be used with any comprehensive rotorcraft model, no matter how complex, because it *only deals with the inputs and outputs of the system*, no knowledge of the comprehensive model is required. Unfortunately, it also suffers serious drawbacks. First, the identification of the trim matrix is an expensive, inefficient operation. Typically, perturbations are made to a reference, periodic equilibrium configuration of the rotor; each trim state must be perturbed separately to obtain the sensitivity of the output, thrust and moments, to each of the input, collective and cyclic controls. Each perturbation provides one column of the trim matrix. To obtain accurate sensitivities, the rotor must be simulated for many revolutions after a control input has been perturbed; indeed, the new periodic equilibrium configuration of the rotor resulting from the perturbation must be obtained to assess sensitivity to the control input through finite difference. Simulation lengths of as much as 6 to 12 revolutions are often required to reach the new periodic equilibrium state of the rotor. It must be noted that this process is very inefficient: the rotor is simulated for many revolutions,

but only the beginning and end points of the simulation are used to assess sensitivities. Second, once the trim matrix has been identified, the performance of the autopilot depends on the selection of a gain matrix. If low gains are selected, it will take a long time to reach trim; if high gains are selected, trim could be reached much faster, but system could become unstable. The autopilot approach does not provide guidelines for the selection of the gain matrix.

Hence, while the autopilot approach can be applied to very complex rotorcraft models, including coupled CFD-CSD simulations, it is also computationally very expensive. Assuming a trim problem with three states, the collective and two cyclic control inputs, the complete identification of the trim matrix will require four separate runs. The first establishes a periodic reference configuration of the rotor, and the next three provide the periodic solutions under three distinct perturbations of the control inputs. With this information, the trim matrix is constructed and the actual autopilot trimming can start. An optimum estimate is that each step of the procedure requires the simulation of about four to six revolutions of the rotor (two for the initial startup, and four ¼ revolutions for a four-bladed rotor, minimum). The total process will then require the simulation of a total of five times four, or twenty rotor revolutions. While this computational effort is reasonable when using simplified aerodynamic models, it is clearly unacceptable if CFD is used to predict unsteady aerodynamic loads. Neural networks have been applied to improve the non-linear capability of these autopilot-based trim models[32,33,34] and they, appear to hold promise in relieving some of the coupling issues previously discussed.

2.7. Control Theory

The problem of system realization or system identification for linear time-invariant models has received considerable attention in numerous engineering applications such as dynamic simulation and control of flight vehicle, identification of vibration modes of large-scale flexible structures, the health monitoring and damage detection of civil engineering structures, or electrical circuits and imaging processes. In general, system identification aims at creating a mathematical model of a dynamical system from measurements of its input and output. Past decades have witnessed the development of identification tools for the construction of state space representation of linear systems. Kim and Arora[35] have reviewed the subject of system identification, focusing on linear and nonlinear dynamical systems; Viberg[36] has reviewed subspace-based identification methods.

The seminal work Kalman introduced the concepts of controllability and observability, which are important prerequisites to identification. The Ho-Kalman algorithm[37] subsequently provided a state space approach, and a minimum realization was obtained from Markov parameters. This algorithm is widely used as an identification algorithm, but it also contributed to the development of state space models presenting balanced properties[38]. When used in conjunction with numerically stable tool such as the singular value decomposition, the Ho-Kalman algorithm has been extended to the eigensystem realization algorithm[39]. To decrease the effects of noise and nonlinearities, the eigensystem realization algorithm with data correlation was developed. Furthermore, eigensystem realization algorithm combined with observer/Kalman filter identification became an optimal procedure to construct a minimum order plant and compute Kalman filter gain matrix from input-output data. However, the computation of Markov parameters by observer/Kalman filter identification remains complex and determines the accuracy of the system realization. If a poor approximation of Markov parameters is obtained, system identification might be meaningless, prompting the development of methods aimed at

improving the accuracy of these parameters. System realization methods based on Ho-Kalman algorithms and its extensions are known as minimum realization procedures.

Another system identification approach is based on subspace identification methods[40]. In these methods, a state space representation of linear system is found by matrix projection operations, and elimination of the effects of noise is a major concern. For stochastic system, Peeters and de Roeck[41] have used Kalman filters to eliminate the effect of the white noise with zero mean. For more general cases, Overschee and de Moor[42] have reviewed subspace methods and algorithms for the identification of linear time-invariant systems from given input-output data. Robust identification procedures have been developed for deterministic, stochastic, and combined deterministic-stochastic systems. Because matrix projection operations are computationally expensive, these methods are most suitable for solving small size problems.

Artificial Neural Networks, or simply Neural Networks (NN) present one of the most popular means to approximate complex non-linear relationships $y = f(x)$, where $x \in R^n$ and $y \in R^m$ are vectors of inputs and outputs, respectively. This is accomplished by the process of statistical learning (training of NN) based on a known set of data points $y_i = f(x_i), i \in \Sigma$ by tuning "free" parameters, vector *p*, that generally include the number of hidden layers and neurons (both are integers) and the weights assigned to neutrons assuming real values. Here the data set Σ can be obtained via a deliberate effort (design of experiments), or can be naturally available as a part of the problem set up. The training of NN usually involves a minimization of a mismatch between the "true" function and its NN approximation on the available data set: $\min_p \| y_i - \tilde{y}_i \|, i \in \Sigma$ with respect to an appropriate metric, e.g., a Euclidian norm on $R^{\Sigma \times m}$. The resulting approximation, often called a surrogate- or metamodel, $\tilde{y} = \tilde{f}(x)$ can be then used instead of the original relationship practically without any computational cost. NN are very flexible, so that an even single-layer configuration can approximate a function of arbitrary complexity with a desired precision provided it has a sufficiently large number of neurons[43]. This flexibility, while explaining popularity of the NN, comes at a price: it can lead to the problem of "overfitting" when the resulting metamodel captures not only the desired functional relationship (i.e., "trend") but also the "noise" present in the training data set that can be amplified in an unpredictable fashion for inputs outside of this training set, leading to poor predictive capabilities of the resulting metamodel. This problem must be addressed by balancing the complexity of the NN (i.e., making it flexible enough to capture the trend, but simple enough to being capable of filtering out the noise). A related (often more critical) problem of NN is the lack of transparency; the resulting metamodel is effectively a black-box devoid of any physical meaning.

Both of these issues are significantly mitigated by using a metamodel as a bridge between a low-fidelity model that captures some fundamental behavior of the system at hand and the higher fidelity model that is being approximated. This idea is effectively a foundation of a recent "variable fidelity" approach to global optimization, where kriging, which has generally comparable to NN properties, is a more popular choice of a metamodel. This also largely explains the success of NN applications in adaptive control where simple, usually linear, low-fidelity models are readily available. For example[44], where controller feedback linearization resulting in so-called pseudo-control signal is used to construct a low-fidelity model, and the neural net is used to model the error of this model. As an added benefit this error (called the disturbance signal) not only accounts for the unmodeled dynamics of the system, but captures the uncertainty associated with the parameters of the system as well. Further study by the same

group of researchers at the Georgia Institute of Technology (GIT) experimentally confirmed[45] the effectiveness of this approach to adaptive control of a model helicopter. Similar concepts were used where so-called "reference model" was augmented by an adaptive neural net in order to identify and compensate for the mismatch between the reference model, and enable solution of an optimal control and improving performance of an auto-pilot-based controller. Therein NN provided a static mapping between the state vector that does not take an advantage of the inherent structure of this vector that are imposed by the dynamic of the system. While it is possible to consider so-called dynamic/recurrent networks that do take this structure into consideration explicitly during the training procedure[46,47], the added benefits of this approach to the rotor control problem are not clear.

3. Integrated Framework for Loose and Tight CFD-CSD Coupling

The traditional delta-airloads loose coupling framework applied by most CFD-CSD coupling is currently some variation of the flowchart shown in Figure 3a). In these methods, the trim is performed by the CSD method, typically applying an auto-pilot controller and exchanging data at some fraction of a revolution via external files (file I/O). This process works well for level flight simulations, but transient or maneuvering aeroelastic predictions require a tight coupling process wherein the loads and deflections are exchanged at each time step, as illustrated in Figure 3b). The amount of data that is exchanged during each time step becomes prohibitive when applying file I/O, so that this data exchange needs to be performed via arrays in memory between the two methods.

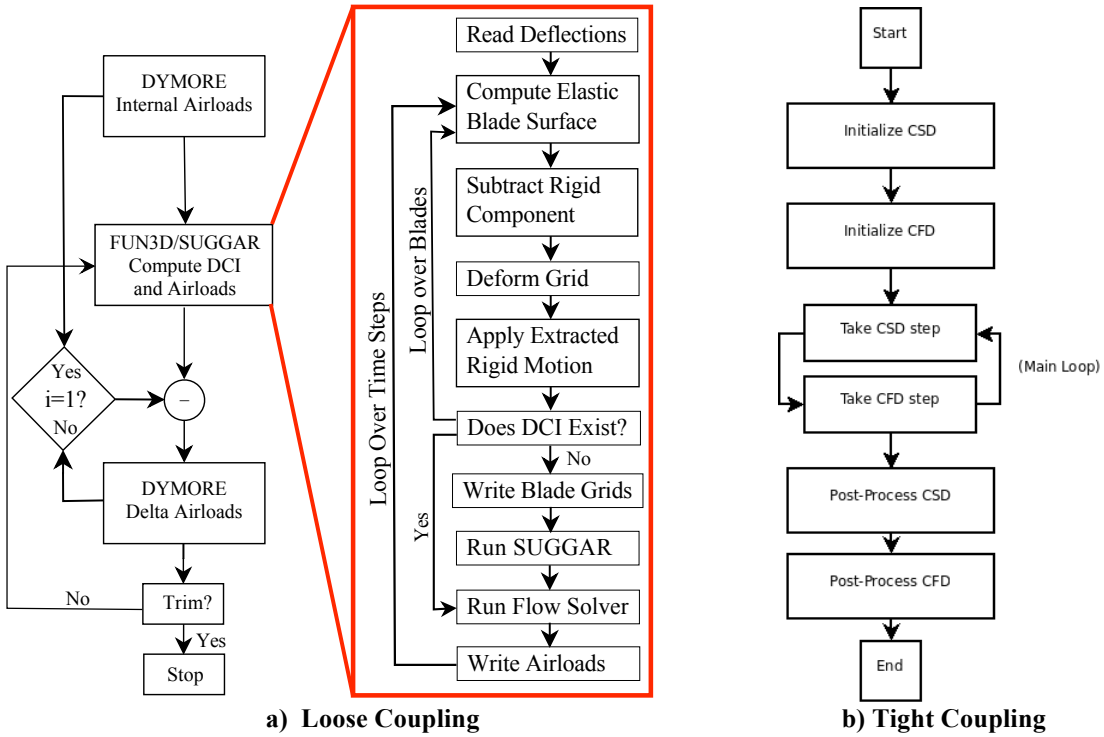


Figure 3: CFD-CSD loose and tight coupling flowcharts demonstrating FUN3D and Dymore coupling.

To achieve the ability to interchange various codes within a multidisciplinary framework ("plug and play"), an integrated PYTHON[48]-based framework that permits the CFD-CSD and computational controllers (CC) to interact via direct access module data without the need for external files has been developed. It should be noted that this is only necessary for the CFD-CSD-CC data; MPI and file I/O for post-processing are not affected.

The data structures to accomplish the transfers were originally planned to use the same data structures developed for the U.S. Army's Hi-Arms project, so that the NASA OVERFLOW and FUN3D codes, as well as the Georgia Tech Dymore code could be utilized interchangeably within the resulting Hi-Arms methodology. Delays in obtaining the protocols for the Hi-Arms

project precluded the availability of the framework, but similar protocols have been developed and demonstrated.

The overall Python framework has the format shown in Figure 4. For any CFD-CSD code that is being coupled, the codes should first be initialized, as denoted by the section labeled "Python Initialize". At this point, the framework is ready to begin the integration of the CFD and CSD equations of motion, and moves to the section "Python Solution". The total number of coupling iterations is controlled via an input to the Python script driver, denoted in Figure 4 as N_total. This would be for example, 10, to perform 10 loose-coupling cycles. Within each coupling iteration, the CFD and CSD codes are advanced by CFD_int and CSD_int time steps, respectively, so that each code may be optimized to the required time step needed to advance the simulation. For example, to exchange information every 1^o of azimuth, CFD_int could be set to 20 iterations (0.05^o azimuth per time step), while the CSD_int may be set to 1 (1^o azimuth per time step). For "true" tight coupling, both CFD_int and CSD_int would be set to 1 and advance the same azimuthal amount during the time step. When the CFD-CSD coupled solution has completed the required number of coupling iterations, the framework proceeds to the "Python Finalize" section. Here, post-processing and orderly shut down of the individual codes is completed.

In order to accomplish the Python coupling, the main routine of each code was rewritten to conform to the framework illustrated in Figure 4. For example, FUN3D's main routine was decomposed into several routines, f3d_init, f3d_iter, f3d_fmg_pre, f3d_fmg_post, f3d_loop, and f3d_post for performing initialization, iterations, full multi-grid pre-processing, full multi-grid post-processing, looping through all iterations, and post-processing, respectively. These are accessible to python as illustrated by the coding:

```
subroutine fun3D_init
  use main, only : f3d_init
  implicit none

  call f3d_init
end subroutine fun3D_init

subroutine fun3D_loop
  use main, only : f3d_loop
  implicit none

  call f3d_loop
end subroutine fun3D_loop

subroutine fun3D_iter(bcontinue)
  use main, only : f3d_iter, iter
  implicit none

  logical, intent(out) :: bcontinue

  call f3d_iter(bcontinue)
  iter = iter + 1
end subroutine fun3D_iter

subroutine fun3D_fmg_pre
  use main, only : f3d_fmg_pre, fl, iter, fmg_levels_request
  implicit none
```

```
     call f3d_fmg_pre

!....Setup variables that we need to call iter
     fl = fmg_levels_request
     iter = 1
   end subroutine fun3D_fmg_pre

   subroutine fun3D_fmg_post
     use main, only : f3d_fmg_post
     implicit none

     call f3d_fmg_post
   end subroutine fun3D_fmg_post

   subroutine fun3D_fully_loop
     use main, only : f3d_init, f3d_fmg_pre, f3d_iter, f3d_fmg_post,
&                     f3d_post, fl, iter, fmg_levels_request

     implicit none
     logical :: bcontinue

     call f3d_init
     call f3d_fmg_pre
     fl = fmg_levels_request
     iter = 1
     call f3d_iter(bcontinue)
     call f3d_fmg_post
     call f3d_post
   end subroutine fun3D_fully_loop
```

This code is wrapped within Python so that it can be called from the main Python script. Alternatively, these routines can be utilized outside of Python by a new main control routine, illustrated by:

```
program pymain
  use main, only : f3d_init, f3d_loop, f3d_post
  call f3d_init
  call f3d_loop
  call f3d_post
end program pymain
```

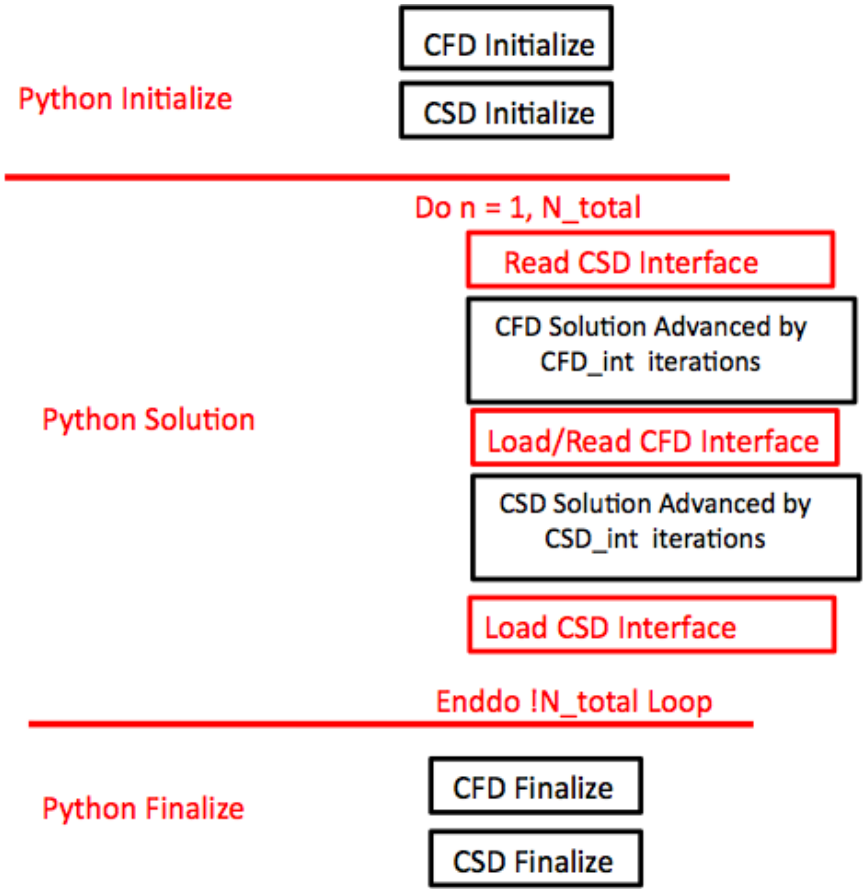


Figure 4 Python CFD-CSD Coupling Framework

These new subroutines created from the original main routine are the only routines, outside of the routines that read and write to the Python data interfaces that need to be "wrapped" within Python. The remainder of CFD code, including any libraries needed such as Suggar or Dirtlib, is compiled as a set of shared libraries that are accessed by the code when needed. Thus, the intrusion into the CFD code is minimized.

To write to or read from the Python interface, existing coupling routines in CFD also were modified. Rather than writing to a file or calling a CSD code directly, the routines instead write or read the data into the predefined kinematic interface protocols. These are accomplished via a set of ifdef conditional statements that are activated or deactivated during the compile time so that either the Python arrays are filled or the data is output to the original file format. An example of this process is illustrated by the code from fsi_coupling:

```
#ifdef HAVE_Dymore
!     Get linear deflections, non-dim by R
      write(*,'(a)',advance='no') 'Getting linear defs from Dymore ... '
      call Dymore_get_pos(motbuff, buffsize)
      write(*,'(a)') 'done'
      call se_flush
#else
!     Copy into the motion buffer deflections coming from python
      motbuff = py_deflections
#endif
```

The same coding process has been performed for the CSD code, Dymore. Its main routine has been decomposed into three sections: Dym_Initialize, Dym_Time_Step, and Dym_Post_Processing. Dymore.c, the main controlling routine of the code, contains the main functions that are called (and wrapped) by the Python script: CallInt(int, char** argv), CallSetFEM(), CallTimeStep(), CallTimeLoop(), CallEndFEM(), and CallFin(). The remainder of the Dymore code is not wrapped, but becomes a shared library as previously noted for FUN3D. A swig interface was written to wrap Dymore.c under Python, and the appropriate typemap module, argv.i, was included in order to transform Python list into char ** as required for the initialization function of Dymore CallInt(int,char*argv[]):

```
%module argv

// This tells SWIG to treat char ** as a special case
%typemap(in) char ** {
  /* Check if is a list */
  if (PyList_Check( input)) {
    int size = PyList_Size( input);
    int i = 0;
     1 = (char **) malloc((size+1)*sizeof(char *));
    for (i = 0; i < size; i++) {
      PyObject *o = PyList_GetItem( input,i);
      if (PyString_Check(o))
       1[i] = PyString_AsString(PyList_GetItem( input,i));
      else {
      PyErr_SetString(PyExc_TypeError,"list must contain strings");
      free( 1);
      return NULL;
      }
    }
     1[i] = 0;
  } else {
    PyErr_SetString(PyExc_TypeError,"not a list");
    return NULL;
  }
   }
```

The data structure interface formats (SIF) were defined to pass the loads and deflections. The interface format is defined in Dymore as:

```
/*=======================================================
/*  Data Structure KinInt
/*=======================================================

typedef struct KinInt{
      int NbOfLfnLines;       /*number of LfnLine
      int *NbOfLfnLineAst;    /* Ast number per LfnLine
      int *LfnLinePsi;        /* LfnLine Psi Angle
      double *LfnLineAstPos;  /* Ast Position for Ast LfnLine
      double *LfnLineAstOrient;     /* Ast Orientation for Ast LfnLine
      double *LfnLineAstLinVel;     /* Ast Linear Velocity for Ast LfnLine
      double *LfnLineAstAngVel;     /* Ast Angular Velocity for Ast LfnLine
}*KinInt, OKinInt;
/*=======================================================
```

These data are retrieved from the data structure by FUN3D (in FORTRAN) by code such as the following:

```
subroutine pass_deflections(deflections, rotations, buffsize)
!
! Inputs:
! f2py integer check (0<buffsize) :: buffsize
! f2py integer :: buffsize
! f2py real*8 dimension(buffsize) intent(in) :: deflections
! f2py real*8 dimension(buffsize) intent(in) :: rotations
!
  use pydata
  implicit none

  integer, intent(in) :: buffsize
  real*8, dimension(buffsize), intent(in) :: deflections, rotations

  ! Store the current deflections and rotations
  py_deflections = deflections
  py_rotations   = rotations
end subroutine
```

The load data have a similar protocol, defined by:

```
/*===================================================================
/* Data structure LoadInt
/*===================================================================
typedef struct loadInt{
      int NbOfLfnLines;             /* number of LfnLine
      int *NbOfLfnLineAst:          /*ast number per LfnLine
      double *LfnLineAstForces;     /* ast forces for Ast LifnLine
      double *LfnLineAstMoments;    /* ast moment for Ast LifnLine
}*LoadInt, OLoadInt;
/*===================================================================
```

These data are placed into the data structure in FUN3D via subroutines, such as the following:

```
subroutine get_loads(forces, moments, buffsize)
!
! Inputs:
! f2py integer check (0<buffsize) :: buffsize
! Outputs:
! f2py real*8 dimension(buffsize) intent(out) :: forces
! f2py real*8 dimension(buffsize) intent(out) :: moments
!

  use pydata
  implicit none

  integer, intent(in) :: buffsize
  real*8, dimension(buffsize), intent(out) :: forces, moments

  forces = py_forces
```

```
  moments = py_moments
end subroutine get_loads
```

An example of the comparable c code implemented into Dymore is

```
%include "numpy.i"
     %init
     %{
      import_array();
     %}
  %apply (double* ARGOUT_ARRAY1, int DIM1) {(double* array, int npts)};
  %include "get_pos.h"
```

where get_pos.h is the header file containing the definitions of these funtions:

```
void get_Pos(double* array,int npts);…
```

In compiling the new CFD-CSD Python protocol, it is necessary to have not only the Python (for Fortran code) and swig (for c code) resident on the computer, but also numpy and f2py. There can be issues associated with compiling and porting these codes to different machines, which are discussed in a later section. Porting problems can occur with the libraries that are needed by the core FUN3D and Dymore codes, such as SUGGAR.

In order to ensure that the codes were properly rewritten and working under the Python framework, both codes were run separately for a given problem in both stand-alone and python mode. The results were meticulously compared and it was found that the outputs were identical no matter which way the codes were run. Next, a small case – which was not a true CFD-CSD coupling – was contrived to ensure that the data were passed correctly between the CFD and CSD codes while using the Python SIFs. Only at this juncture was the CFD-CSD coupling under Python attempted.

The modified main routines, as well as the SIFs, have been provided to both the NASA FUN3D and OVERFLOW developers during the course of this project.

4. Steady-State Trim Algorithm

The classical autopilot control law[49,50] is extensively used for trimming rotor models in comprehensive analysis codes. It consists of a simple control law that constructs a map relating the inputs and outputs of the system, based on a static approximation to its behavior. It is then easy to compute suitable filter time constants and control gains such that a closed loop controller will steer the system to its trimmed configuration with a desired performance. However, when this control law is used to steer complex rotorcraft models, such as those used in comprehensive analysis codes, stable behavior is only observed for judiciously chosen values of the controller parameters. Three major sources of error are responsible for the observed discrepancy, as identified by Peters et al.[1]: in the design of the controller, 1) the dynamic characteristics of the plant are ignored, 2) the non-linear behavior of the plant is not taken into account, and 3) the Jacobian of the system is assumed to be known exactly.

The first part of this effort has focused on the implications of these three assumptions on the behavior of the classical autopilot, by studying their effect through both numerical closed-loop experiments on a realistic UH60-A multi-body rotor model (the plant), and eigenvalue analysis of the closed-loop characteristics of different reduced order models of the full plant[51].

Briefly, it is known that based on a rigid blade assumption and momentum theory, the trim Jacobian matrix can be approximated by as follows

$$J^{-1} = \left[A_{\mu} \right]^{-1} = \begin{bmatrix} \frac{1}{6}\left(1+\frac{3}{2}\mu^2\right) & \frac{2}{9}\mu & 0 \\ \frac{-C}{6}\left(\frac{\mu}{1+C^2}\right)\left(C-\frac{\gamma}{16p^2}\right) & \frac{-C}{16\left(1+C^2\right)}\left\{C\left(1+\frac{3\mu^2}{2}\right)-\frac{2}{9}\mu^2\frac{\gamma}{p^2}\right\} & \frac{-C}{16\left(1+C^2\right)} \\ \frac{C}{6}\left(\frac{\mu}{1+C^2}\right)\left(1+C\frac{\gamma}{16p^2}\right) & \frac{C}{16\left(1+C^2\right)}\left\{1+2\mu^2-\frac{2}{9}\mu^2 C\frac{\gamma}{p^2}\right\} & \frac{-C^2}{16\left(1+C^2\right)}\left(1+\frac{\mu^2}{2}\right) \end{bmatrix}^{-1}$$

A quasi-steady trimmer has two phases, identified as the reference and adjustment phases. The trim module first computes the adjustment to the control settings at the beginning of the adjustment phase using the expression $\Delta\underline{\theta} = J^{-1}G(\underline{L} - \underline{L}_{target})$ where J is the Jacobian matrix, $J = \frac{\partial \underline{L}}{\partial \underline{\theta}}$, as described above. The behavior and convergence characteristics of the quasi-steady trimmer are strongly affected by the diagonal gain matrix, G, and the individual gains should be adjusted to obtain the best convergence characteristics. It was observed that high gain values may render the closed-loop system unstable. In addition, even moderate gain values can lead to instabilities because the quasi-steady trimmer does not account for system dynamics. To overcome this problem, an adaptive gain selection strategy was introduced such that when the difference between the input and its target value becomes small, the gain is gradually decreased to zero using a hyperbolic tangent function. Results have been obtained that imply that the inaccurate determination of the Jacobian matrix is responsible for the observed lack of stability

of the autopilot algorithm at high gains. This work has been summarized in a paper that was presented at the AHS Forum in 2008[52].

The analysis of the classic autopilot algorithm underlined the fact that the dynamics of the system are largely ignored in that approach. This observation led to the elaboration of a new strategy for trimming, called the quasi-steady trim algorithm. The procedure for the proposed trim strategy is as follows:

1. Identify the Jacobian $\boldsymbol{J}$ and its inverse $\boldsymbol{J}^{-1}$;
2. Obtain the initial guess for the control settings as $\underline{\boldsymbol{\nu}}_0 = \boldsymbol{J}^{-1}\,\underline{\boldsymbol{T}}_0$, where $\underline{\boldsymbol{T}}_0$ is the target values for the trim;
3. Run a *static analysis* with $\underline{\boldsymbol{\nu}}_0$ as control settings;
4. Run a *dynamic analysis* with $\underline{\boldsymbol{\nu}}_0$ as control settings and the configuration from the static analysis of step 3 as initial conditions; the converged trim variables are obtained as $\underline{\boldsymbol{T}}_1$;
5. If $\|\underline{\boldsymbol{T}}_k - \underline{\boldsymbol{T}}_0\|$ $(k = 1, 2, \ldots)$ is less than the expected error criteria, stop; otherwise, update the control settings as follows:
   a. $\underline{\boldsymbol{\nu}}_k = \boldsymbol{J}^{-1}\,[\,\underline{\boldsymbol{T}}_0 - \underline{\boldsymbol{T}}_k] + \underline{\boldsymbol{\nu}}_{k-1}$;
   b. Run a static analysis with $\underline{\boldsymbol{\nu}}_k$ as control settings;
   c. Run a dynamic analysis with $\underline{\boldsymbol{\nu}}_k$ as control settings and the configuration from the static analysis (in step b) as initial conditions; the converged trim variables are obtained as $\underline{\boldsymbol{T}}_{k+1}$;
   d. Go to step 5.

Using the above procedure, studies to determine the sensitivity of the Jacobian were performed with the UH60 model. As an example, consider two cases were the rotor speed is 27.02 rad/sec, and the target values for the trim are 17,944 lb, 6,884 lb-ft, and -2,583 lb-ft for the thrust, roll and pitch moments, respectively. Figure 5 and Figure 6 illustrate the results for the first case, where the Jacobian was computed with predetermined control settings as 0.178 ft, -0.0183 rad, and -0.077 rad for swash plate displacement, lateral and longitudinal tilt, respectively. Figure 5 shows the resulting thrust and moments obtained at each coupling iteration, along with the errors when compared with the target values. Figure 6 provides the converged swash plate displacement and tilt, and the errors comparing with the results from autopilot run for each iteration. It is readily observed that the trim convergence is achieved in 5 to 6 coupling iterations.

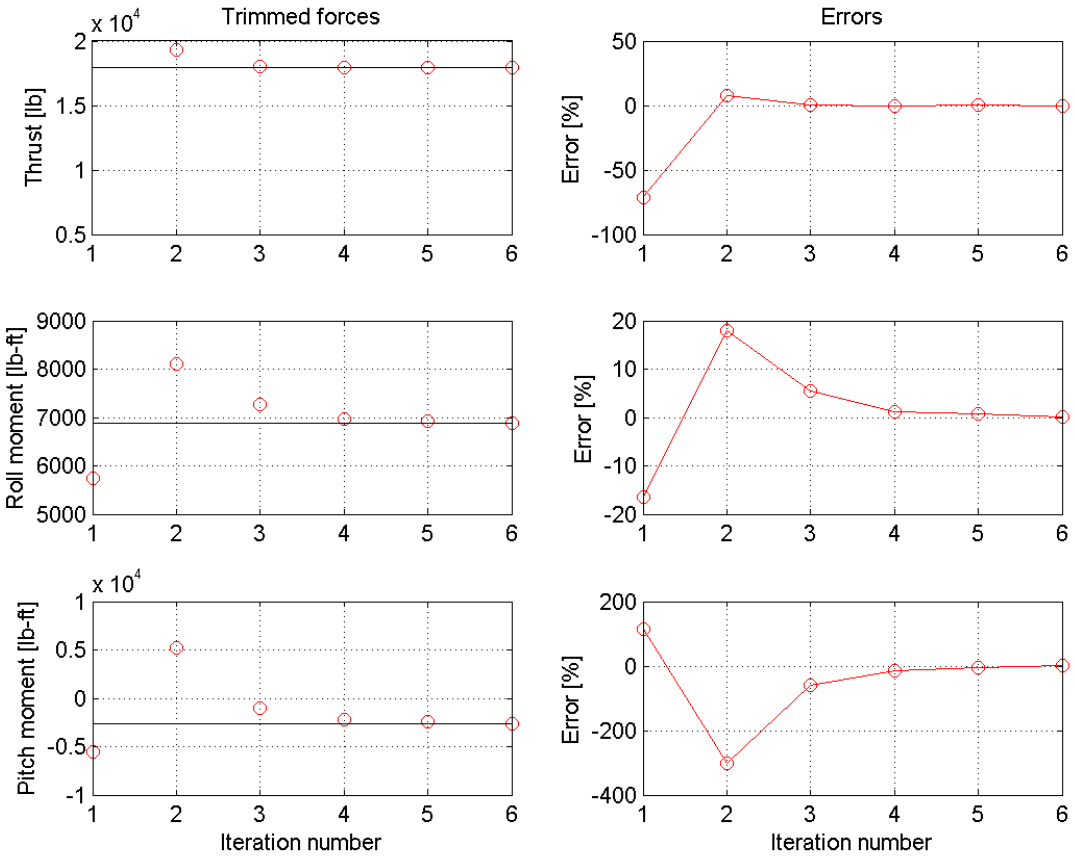


Figure 5: Trimmed thrust and moments with errors for the Jacobian computed with predetermined control settings. The horizontal lines in the figures on the left are the target values.

The next two figures illustrate the typical results for the second example where the Jacobian was computed at zero settings for the swash plate displacement, lateral and longitudinal tilt. Similar to the first method, Figure 7 shows the converged thrust and moments, while Figure 8 illustrates the resulting converged swash plate displacement and tilt. Clearly, the trim variables have not converged by the 6th iteration. These typical examples illustrate the sensitivity of Jacobian quality with convergence when using this approach. Thus, a more accurate approximation of the system Jacobian should accelerate the trimming process.

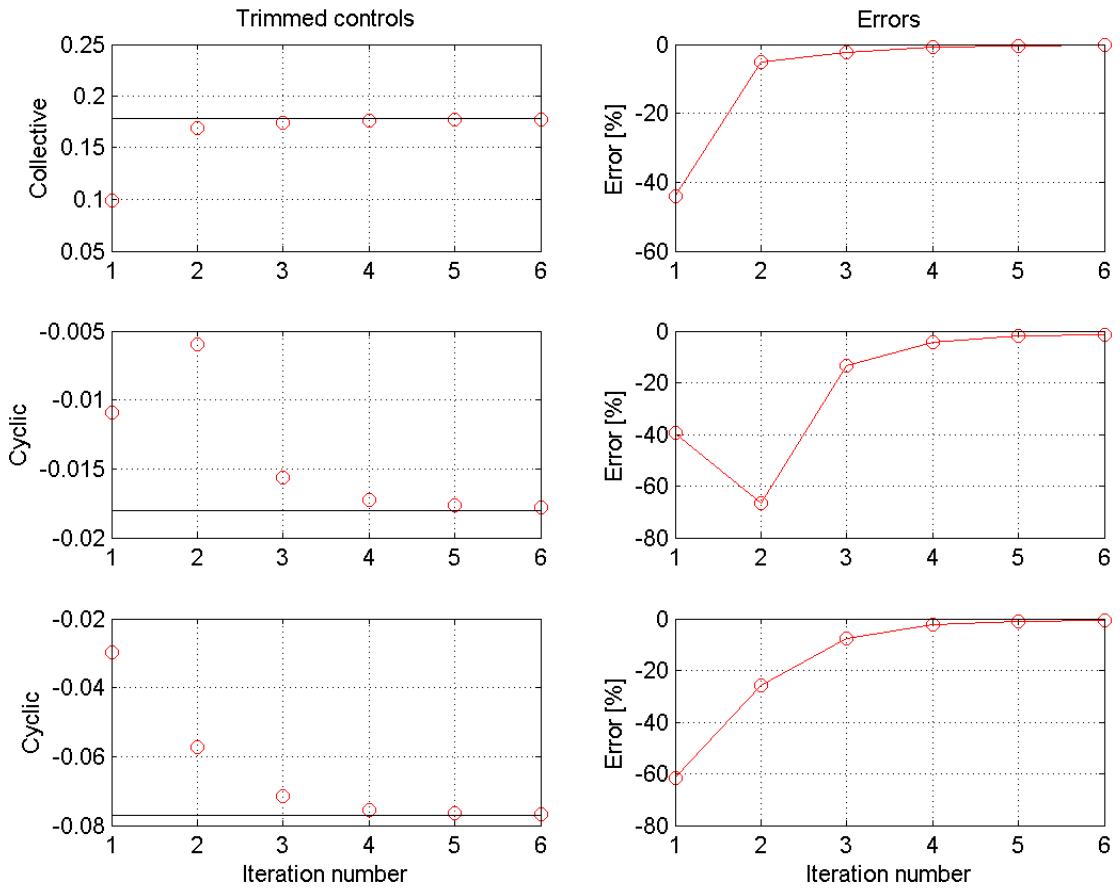


Figure 6: Trimmed swash plate displacement and tilt with errors for the Jacobian computed with predetermined control settings. The horizontal lines in the figures on the left are the autopilot solutions.

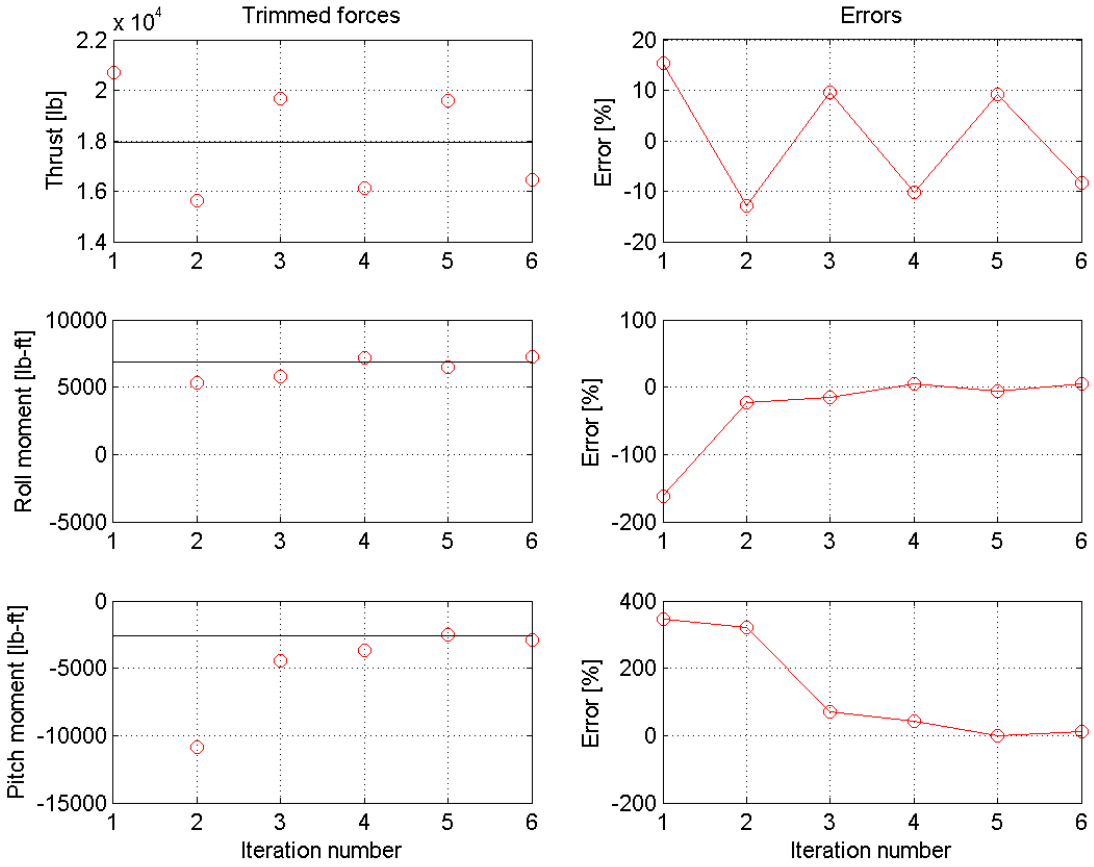


Figure 7: Trimmed thrust and moments, and errors with errors for the Jacobian computed with control settings at zero. The horizontal lines in the figures on the left are the target values.

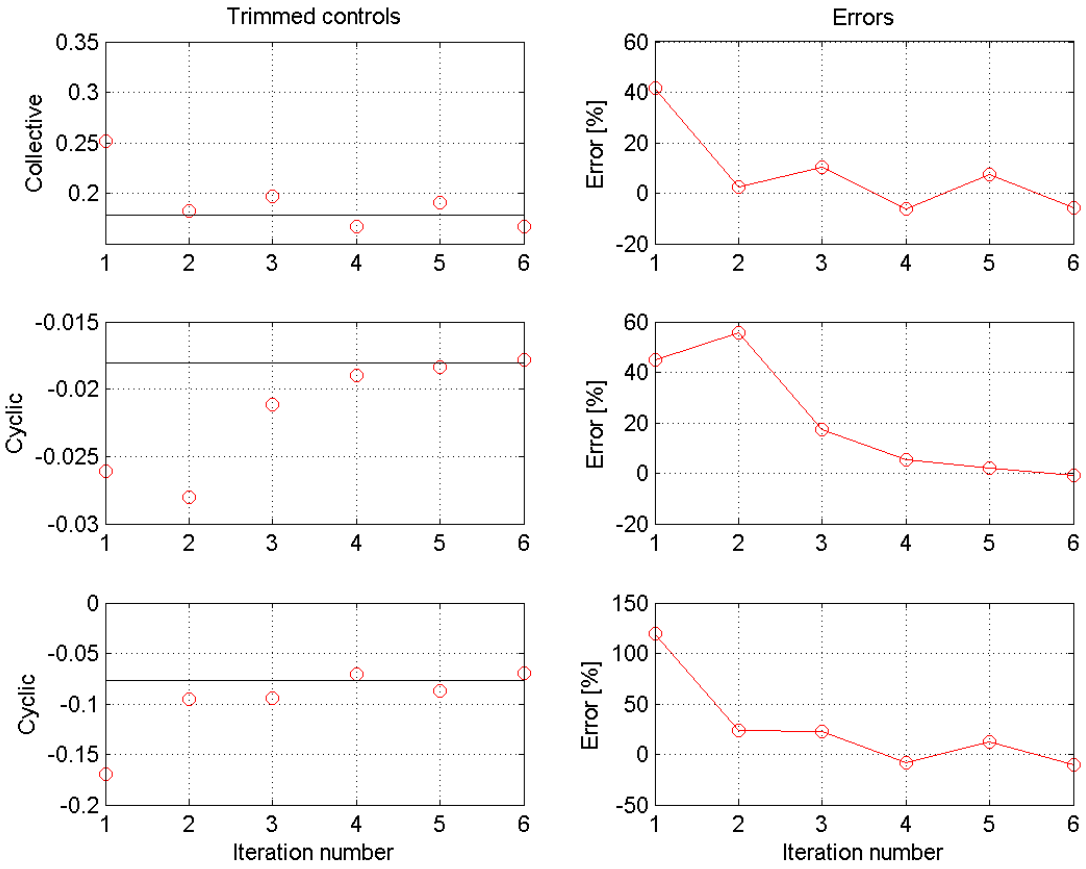


Figure 8: Trimmed swash plate displacement and tilt with errors for the Jacobian computed with control settings at zero. The horizontal lines in the figures on the left are the autopilot solutions.

5. Kriging Control Algorithm

With an efficient trimmer in place, development focused on the identification and creation of a neural network approach to accelerate and enhance the trimming process for quasi-static (e.g., level flight, steady turns) and dynamic (maneuvers, stability) simulations. The quasi-steady trimmer was used to generate a set of trimmed flight conditions for various advance ratio and thrusts to construct a parametric study that facilitated the selection parameters and type of the meta-model for the system identification process.

A meta-model is an approximation of the input/output (I/O) function that is implied by the underlying simulation model that can be either deterministic or random (stochastic). Meta-models may be used for validation and verification of the simulation model, sensitivity of the model, and optimization of the simulated system. A deterministic simulation is a simulation that gives same output for the same input. The linear regression methods fit the data points to linear or non-linear curve functions by minimizing the distances between these samples points and the curve to fit (the error).

While in the case of the non-linear regression techniques like Artificial Neural Networks (ANN), the training points are fitted to a complicated function with no easy way to predetermine the character of the curve. That is, in the case of the ANN, the curve function is determined implicitly and not as a set of unknown coefficients as is the case for linear regression techniques. A Gaussian process (GP) does not require curve fitting; instead GP uses the information in this data set to create a statistical prediction or estimate by incorporating Bayesian regression. Not all the predictions obtained by a Gaussian process will have same measure of goodness (accuracy); the closer the point is to one of the training points, the smaller the variance of the prediction will be, until it reaches zero at a known training point.

Kriging as a meta-modeling technique is similar to Gaussian processes in the local component, however does not typically make use of the Bayesian step in the way that Gaussian processes do. Kriging provides a flexible means to construct meta-models that accurately approximate highly non-linear behavior. Kriging meta-models are typically applied in prediction processes, sensitivity analysis and optimization. Its behavior entirely depends on the covariance function and the training data. It is a necessary and sufficient condition that a covariance function of a Gaussian random function be a positive semi-definite function. For application in this effort, an isotropic stationary covariance function that has a set of *(n+2)* free parameters, $\Theta = (\theta_1, \theta_3, \vec{r})$, for a function of *m* input directions and is infinitely differentiable (and thereby very smooth) has been used. That is,

$$C(\vec{x}_i, \vec{x}_j) = \frac{1}{\theta_1} \exp\left[-\frac{1}{2} \left(\vec{x}_i - \vec{x}_j \right)^2 \vec{r} \right] + \delta_{i,j} \theta_3 \qquad \text{i, j} = 1, 2, \ldots \text{n}$$

where *n* is the number of training points, θ_1 controls the overall scale variation of the function, and θ_3 controls the scale of the input independent noise. The vector of free parameters $\vec{r} = (r_1, r_2, \ldots r_m)$ represents the measure of length scale of variation in each of the *m* input directions. When the Gaussian process meta-model is used to determine a deterministic computer experiment, there is no noise in the measurement. These hyper-parameters are determined by maximization of the likelihood of the observed training data to find the Gaussian random function that best approximates the training data. Once the meta-model data are available, the

covariance matrix, ***C,*** is formed by the evaluation of the covariance function between the *n* training points. The covariance vector **c** is formed by the evaluation of the covariance function between the test point $\vec{x}^*$ and the *n* training points. The prediction of the response *y** at a specified test point $\vec{x}^*$ is then computed by

$$y^* = \mathbf{c}^T C^{-1} \vec{y}$$

where $\vec{y}$ is the response vector corresponding to the *n* training input points, $\vec{x}$. Since the prediction is based on a Gaussian random function, it is then in a probabilistic form and its estimated value is its mean value at the test point $\vec{x}^*$.

The kriging method originated in the field of geostatics and was popularized by G. Matheron[53]. Simultaneously to the development of the approach in geostatics, Gandin simultaneously developed the approach for use in the field of meteorology under the name "optimum interpolation"[54].

This ordinary kriging meta-model was implemented in Dymore with two options to either specify the free parameters as a user input or to apply the implemented optimization technique to compute the free parameter values. Additional details of the kriging development can be found in Reference 55.

5.1. Kriging meta-model with optimized free parameters

The most efficient set of free parameters of kriging model for a set of training points is one that maximizes the maximum likelihood estimate that will results in the most accurate estimates of the unknown points. This method is known as the maximum likelihood estimate (MLE) method. However this method is based on the assumption that the observed data will behave as a Gaussian process, which can be a very restrictive constraint. When this constraint precludes the use of the MLE method, then the Cross Validation method (CV) results in better estimates of the model parameters. The disadvantages of the CV method are that it is significantly more computational expensive than the MLE method, and it does not provide an estimate of the variance, which is directly computed with the MLE technique[56].

Based primarily on the cost savings and error prediction, the MLE technique was selected and implemented in Dymore. Both steepest descent and Polak-Robiere algorithms were evaluated to minimize the Gaussian constraint associated with the MLE technique. The steepest descent algorithm was the primary method used to minimize the constraint. When difficulties arose during the convergence of the MLE optimization, the code applied the Polak-Robiere technique to improve the quality of the convergence. During investigations of these two methods, it was determined that poor convergence in many instances was not due to the efficiency of the optimizer, but was highly dependent on the test case. The Polak-Robiere method's main attraction is its simple formula that updates the direction vector, requiring only the computation of the first order partial derivatives. This simplification then conceptually decreases the computation effort associated with the computation of the second-order partial derivatives. The cost of the second-order partial derivative increases with the increase of the number of independent variables. While this method is slightly more complicated than steepest descent, it exhibits more efficient convergence properties. To further optimize convergence, the method was designed so that when it restarted a new loop of optimizer iterations, the free parameters were initialized with the free parameter values computed during the last iteration of the previous loop. The algorithm was also optimized to compute the maximum log likelihood function at random grid points of the free parameters. Here the initial guess of the first optimizer loop is based on the free parameter set corresponding to the maximum log likelihood function.

5.2. Test and optimization of the kriging implementation

To test the kriging implementation, three dependent variables (thrust, rolling moment, pitching moment) were selected, along with three associated control angles (collective, longitudinal and lateral cyclics). This mimics the current method used in CFD/CSD trim for level flight and steady turns. One cost function is maximized and applied in the computation of the free parameters, which is then applied to determine the values of three dependent variables. This means that each of the dependent kriging meta-models could require a different set of free parameters to achieve an optimum fit. For example, to determine an a priori value for the thrust, a particular set of controls will be determined that may not result in the requested values for the rolling and pitching moments. To address this problem, the controls were redefined as functions of the rotor performance rather than the original formulation where the rotor performance parameters are determined from the controls. This inverse formulation permits the target rotor performance to be defined by the control estimates.

The MLE method depends on an unconstrained optimization problem, which is usually solved through a numerical iteration algorithm, such as the method of steepest descent or Newton's method. In these algorithms, the iterative loop initialization begins from a set of initial guesses for the free parameters. For the trim test problem, it was observed that it was difficult to determine a unique set of initial guesses for the free parameters, and that the cost function could include several local minima that were dependent on the value of the initial guess. Thus, during the minimum error search, it was quite likely that a local minimum rather than the global minimum was identified.

The first optimization is based on the free parameters of the collective as function of the thrust and two moments, then the same free optimized set of parameters was used for the two cyclic controls meta-model as a function of the loading. Figure 9, Figure 10, and Figure 11 summarize the results obtained from the kriging meta-model using thrust, longitudinal and lateral cyclics, respectively, as the kriging free (independent) parameters. The meta-models were obtained by optimization of the collective-airloads relations and the data sample points consisted of 9 points obtained by CFD/CSD loose coupling for the UH60 c8534 (high speed) flight case. In order to test the model, first the values of the independent variables for the 9 sample points were passed to the model, where the model then estimated values for the dependent variables. These estimates (red crosses) were compared to the data training points (blue squares) in Figure 9 to Figure 11, where it is observed that the estimated values are almost identical to the trained values. Next, a range of random values for the independent variables were generated and evaluated in the kriging meta-model to test the interpolation accuracy. These interpolated values are plotted as continuous lines in these three aforementioned figures. These two-dimensional interpolated values detect the general behavior of the dependent variables, although a large variance is observed. This variance is attributed primarily to the low number of training points (9) that are used to map the variation of three control variables, which are functions of three independent variables. However, as the primary variable behavior was captured, which suggests that the Jacobian resulting from this meta-model may be accurate enough to obtain a converged solution. It is clear that the use of this model predicted control values that were very close to those of the 5th iteration loose coupling converged results, as shown in Table 2 and Table 3.

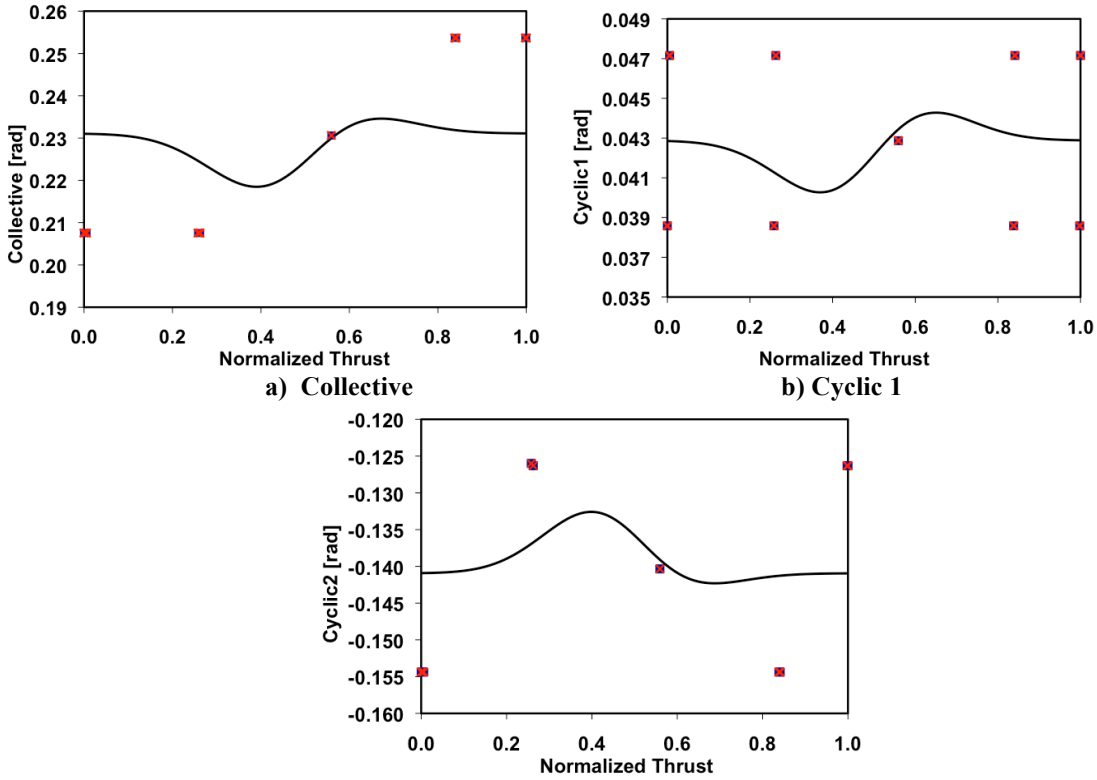


Figure 9: CFD/CSD-based kriging algorithm training using thrust as the independent parameter

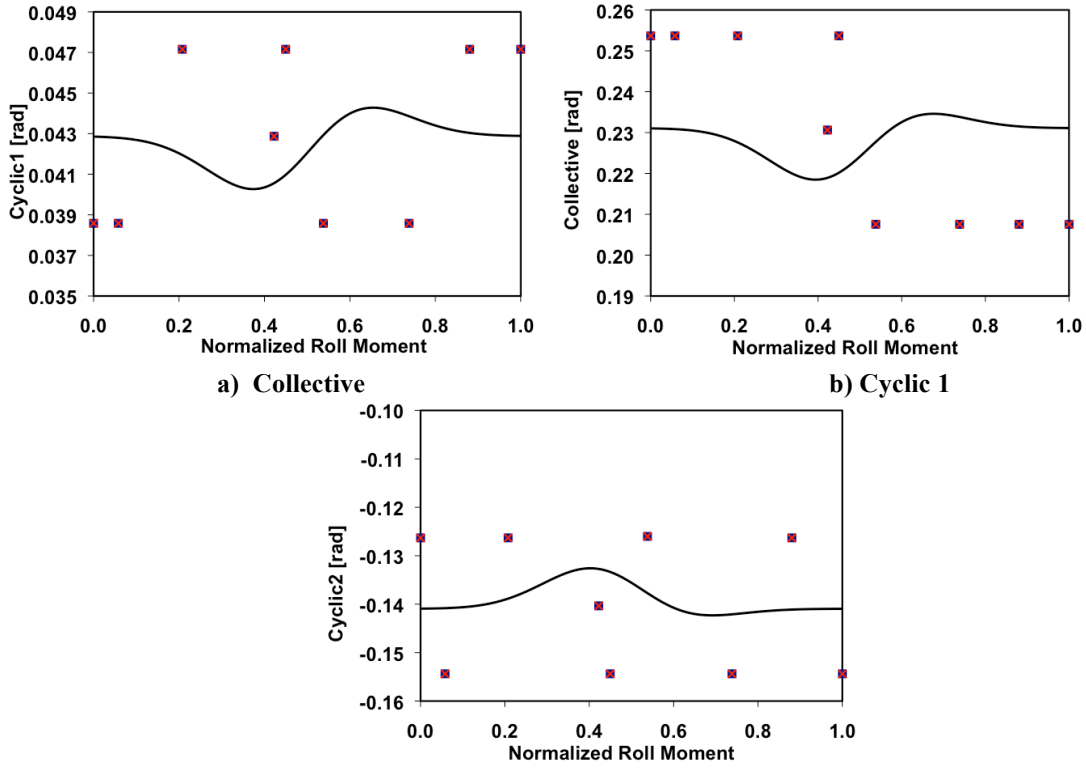
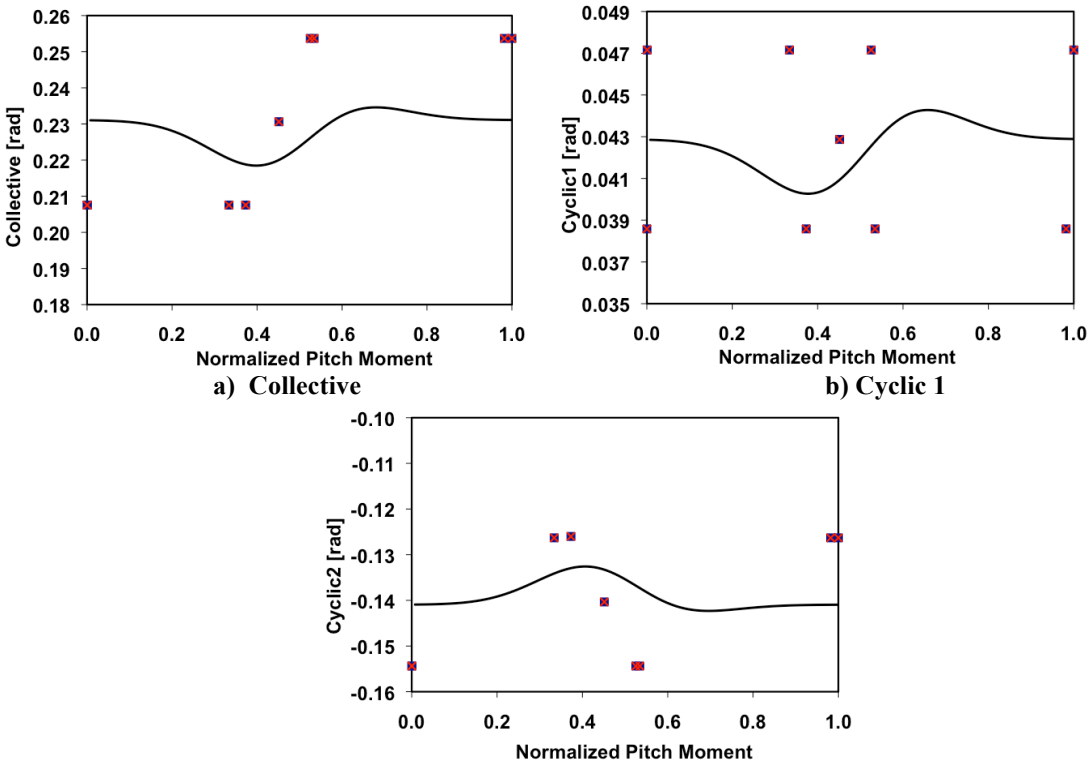


Figure 10: CFD/CSD-based kriging algorithm training using rolling moment as the independent parameter



a) Collective b) Cyclic 1

c) Cyclic 2

Figure 11: CFD/CSD-based kriging algorithm training using pitching moment as the independent parameter

Table 2: Comparison of the controls estimated from kriging to the converged (iteration 5) loose coupling of UH60-AA high speed flight case, 8534

	Collective [rad]	Cyclic1 [rad]	Cyclic2 [rad]	Estimation Variance
9 CFD/CSD training points, 1 free parameter – Case 1	0.228295	0.0423120	-0.139071	0.203
9 CFD/CSD training points, 3 free parameters – Case 2	0.228369	0.0423925	-0.139435	0.199, 0.156, 0.169
5th iteration loose coupling	0.230615	0.0428699	-0.14034	
9 CSD training points, 1 free parameter – Case 1	0.232475	0.0482541	-0.146125	0.657
9 CSD training points, 3 free parameters – Case 2	0.232414	0.0482736	-0.146489	0.643, 0.611, 0.622
16 CSD training points, 1 free parameter – Case 3	0.232997	0.0484298	-0.147962	0.3528
16 CSD training points, 3 free parameters - Case 4	0.232997	0.0484417	-0.148384	0.3527, 0.32, 0.321
CSD trimmed controls	0.238321	0.0496658	-0.154414	

Table 3: Comparison of the first iteration trim characteristics estimated from kriging to the converged (iteration 5) loose coupling of UH60-AA high speed flight case, 8534

	Thrust [lb]	Rolling Moment [lb.ft]	Pitching Moment [lb.ft]
Target loads	16602	6042	-4169
CSD trimmed loads	16602	6032	-4179
CFD-CSD converged loads	16609	6211	-2253
CSD Loads - Case 1	16912	5929	-3330
CSD Loads - Case 3	16842	5983	-3667

The application of the kriging control estimates to start the CFD simulation (CFD/CSD iteration 0) resulted in trimmed loads that were less accurate than the initial estimates with the known flight test control settings. However, the use of the kriging to estimate the controls during the loosely-coupled simulation resulted in more rapid convergence to the target loads than the original autopilot CFD/CSD loose coupling using the known flight test controls, as observed in Table 4. The larger error observed at iteration 0, which is the first CFD simulation using the initial estimates of the CSD trim without the feedback from the delta airloads, appears to be subsequently rapidly adjusted in subsequent coupling iterations.

Table 4: Comparison of converged loads for CSD alone and CFD/CSD loose coupling using controls estimated with and without the kriging algorithm.

	Thrust [lb]	Rolling Moment [lb.ft]	Pitching Moment [lb.ft]
Target loads	16602	6042	-4169
Iteration 0 (Flight test controls)	16380	-3311	-6050
Converged at Iteration 5 (Flight test controls)	16609	6211	-2253
Iteration 0 (Kriging controls)	16050	21140	2857
Converged at Iteration 3 (Kriging controls)	16630	7495	-4235

These initial comparisons were determined using kriging training points that were regularly spaced about the fifth iteration loosely-coupled converged solution. In order to test the accuracy of the kriging prediction, the next numerical experiment applied the same number of training points, but which were randomly spaced. In order to accomplish this test, the simplified aerodynamics was used within the CSD stand-alone method was applied to reduce the run costs. This simplification is justified based on the results (CSD training points), which are included in Table 1. It is clear that for the same number of training points, a larger standard deviation was associated with the cases where the random training points were utilized.

The variance of the estimated values of the dependent values, as compared to the locations of the independent variables sample points, shows that when the estimated points near the training points the variance decreases (a known characteristic of the kriging) with a minimum value of 0.4 for this implementation.

The variance of the estimates was observed in some cases to be unacceptably large for the estimates based on both the CFD/CSD and the CSD training. While an optimally small number of training points is sought, too few may result in these large variances. In examine this aspect of the training, additional training points were added to the CSD training model. Table 2 confirms

that the variance of the CSD model with 16 training points was smaller (about half) than the 9 training points. However, the differences in the estimated controls were insignificant.

During these tests, it was discovered that as the value of the free parameter(s) passed to the optimizer loop could significantly influence the resulting control estimates and the standard deviation between the meta-model equation and training points. For some instances, it was also difficult to achieve a converged set of free parameters using the MLE method.

Several other researchers have also encountered this situation and proposed different solutions. Reference 56 suggested that this might indicate that the constraint of the MLE method used to optimize the free parameters is questionable, since the method is observed to not always behave as a Gaussian process. Reference 57 noted that, given some data distributions, the likelihood profile function has a long very flat ridge with a number of local maxima of almost identical height for the Gaussian correlation function is used. This is similar to some of the data that are used for the controller in this application. Another similarity to the kriging process here can be found in Mardia and Watkins[58] who noted difficulties with the method of maximum likelihood estimation for a spatial linear model. In these instances, even with a regular covariance scheme such as the exponential function used here, the likelihood can be multimodal. One of their solutions proposes multiple optimization cycles, and then selecting the most accurate among them. The free parameters magnitude is very important and can vary widely, while their exact value is not so important, and hence repeating the optimization multiple times with different initial values for the free parameters, which vary with large steps, could help in resolving this issue..

Runze and Sudjianto[59] addressed the problem of the computed cost for a multi-varying function based on a small number of training points. They noted that these functions encounter difficulty in obtaining the minimum value. They suggest the use of a penalized likelihood function instead of the profile likelihood in these situations. While the penalized likelihood function requires additional computational effort and cost, it addresses the problem that was faced in this study, where a minimal number of training points are sought because of the cost of the obtaining each point. Runze and Sudjianto's solution, with respect to the current trim algorithm, was explored using a simple sine function ($y = sin(x)$) using the original and new methods in Dymore. A sample of 6 points (x = 0, 2, 4... 10) was employed. The prediction based on the Gaussian kriging model shows that the prediction becomes very erratic when x is not equal to the sample data, as shown in Figure 12, with a prescribed free parameter equal to 3, a value suggested by reference 59. When Dymore was run to detect the optimized free parameter for this data set, a different value of the free parameter (1.05) was obtained. When the independent variables in the training data set were normalized to the range [0,1], a value of 30 was obtained for the free parameter. The corresponding predictions for these three models are compared to the exact function in reference 59. It is clear that the most accurate results are obtained when the free parameter is set to the value obtained using the implemented MLE in Dymore and normalizing the independent variables. As the profile function has a very large constant region near its optimum value, where the variance will increase. To minimize the variance, as well as to halt the optimizer before it reaches this constant region, a tolerance constraint of 1% on the normalized maximum independent variable of the training data was imposed on the Euclidian norm of the first partial derivatives of the profile function. This constraint, along with normalization of the independent variables, was shown to resolve this issue.

Various applications of the training model were examined for the UH60-A high thrust flight case (C8534). The use of 16 versus 9 training points was evaluated, as well as the use of one dependent variable (thrust) or all three dependent variables as free parameters (Θ) to optimize the kriging algorithm. The efficacy of applying CSD training points, which are much less expensive than CFD/CSD training points was also examined. The reduction in the number of training points from 16 to 9 does not adversely impact the overall quality of the control predictions for CSD-based training, but it does increase the estimated variance (which is discussed in the next paragraph). This variance can be reduced by the use of normalization of the training data, implementation of a tolerance during training, and use of the MLE algorithm. The kriging algorithm based on the CFD/CSD training provides controls that are within 1% of the predicted controls from a converged CFD/CSD loose coupling simulation. ***Thus, when the controls are not known prior to a CFD/CSD simulation, kriging can provide a close estimate of the controls.***

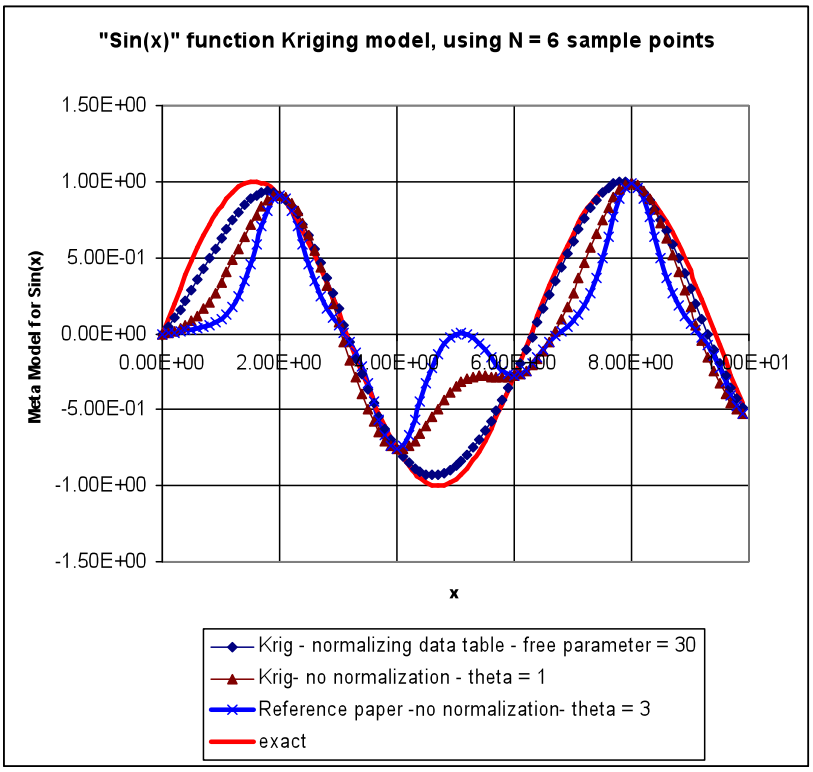


Figure 12: Comparison of Kriging meta-models predictions with different free parameters.

6. Optimization of the Tight Coupling CFD/CSD Process

The tightly-coupled CFD/CSD process was first verified using the 8534 test case for the UH60-A via OVERFLOW and Dymore. This was accomplished by reproducing the last iteration of the loosely-coupled simulation with the slower file I/O process. The results were shown in most instances to be identical, and they are discussed in a section to follow. The cost of the coupling process wherein the converged loose coupling is used to start the tight coupling process is not efficient. Therefore, a series of runs were examined to determine the optimal start-up of the tightly-coupled process.

The initial CFD simulation whether or not it includes component motion is characterized by large flow field transients. The addition of component motion during this transient period can compromise the stability of the simulation for some flight conditions. Therefore, the question arises as with regard to the most efficient method to start up the tight coupling. To examine this, the C8534 UH60-A flight case was restarted from various solutions between 0 and 1 revolution. The thrust was used as the parameter to determine the efficiency of the transfer between methods. Figure 13 pictorially illustrates the behavior of the simulation at each of these initializations. Convergence for the thrust parameter is defined as the first instance in which the error drops below 1% and remains for a full revolution. "Cold starting" tight coupling is not efficient, as it requires a number of revolutions before the error remains below the 1% threshold. Immediately upon restarting from some fractional portion of a loose coupling revolution (which used the initial kriging estimate for the controls), the errors rapidly drop below 5% towards the 1% demarcation. Thus, some level of loose coupling is recommended prior to starting a tightly-coupled simulation. Various loose coupling initializations were examined for their convergence characteristics, the details of which are provided in Table 5. When examined and visually inspected as in Figure 14, the optimal loose coupling start appears to be ¼ to ½ revolution.

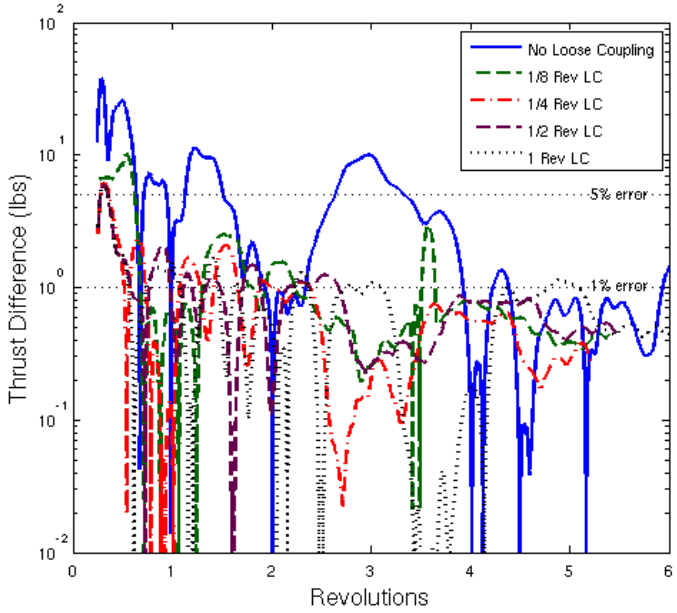


Figure 13: Summary of thrust errors during different tight coupling starts from loose coupling.

Table 5: Tight coupling requirements to reach convergence using kriging to trim during tight coupling.

Total Time Required (revs)	Loose Coupling (revs)	Tight Coupling (revs)	Kriging Trim On (Part of Tight Coupling) (revs)	CPU hours
>6	0	6.0	5.0	2814
3.6	1/8	3.5	2.5	1728
2.4	1/4	2.2	1.2	1166
2.7	1/2	2.2	1.2	1306
5.1	1	4.1	3.1	2432
2.5	5/2	0.0	0.0	1453

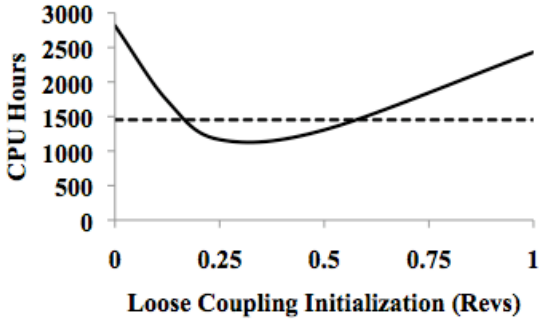


Figure 14: Revolutions needed to reach thrust convergence (1% error) based on loose coupling initialization.

It must be noted that the controls are held for one revolution (which has not been optimized) after switching from loose to tight coupling. This approach was found to be necessary as an additional transient between the loose and tight coupling loads transfer appears. If the kriging is used immediately upon the transfer to tight coupling, then large transients, such as the example in Figure 15 will appear. At the end of this time, however, an additional training point can be added to the kriging database for use in the current as well as future simulations.

Due to the large cost of developing CFD-CSD training points, it was investigated as to whether or not efficient trimming could be achieved via kriging using a far less costly CSD training point database. The CSD training databases examined for 8534 had very good correspondence (less than 5% error) with the controls predicted by OVERFLOW loose coupling. C8534 tightly-coupled simulations were performed using both a twelve point CSD and the nine point CFD-CSD database (Figure 16). As it was previously noted that initialization from a quarter revolution of loose coupling speeds up convergence, both cases used this optimized start. The results indicate that, at least for some conditions, comparable convergence behavior can be obtained with the significantly reduced training cost using a CSD training database, provided directly from a comprehensive code.

A similar tight coupling process was applied in a different effort via funding from the Vertical Lift/Rotorcraft Center of Excellence with FUN3D and UM/NLABS for a rotor undergoing camber changes. This effort is discussed in a paper[60] presented at the 2009 AHS Forum in Grapevine, Texas in May 2009. In addition to verifying that the process is stable, additional testing was accomplished to determine the most efficient and robust method to transition from the initial loading used for trim as defined by the lower-fidelity CSD model to the loads predicted by the CFD method. As shown in Figure 17 and Figure 18, a step function defining the switch between the methods appears to be robust, as well as efficient for the test

cases explored. These simulations highlight the effect that the kriging has on the rapid convergence of the trim, observed in Figure 13. Here, using an auto-pilot trim with the CSD method, comparable to the process applied for delta airloads loose coupling, the process requires 4 to 5 revolutions to attain the 1% thrust error, even with loose coupling initialization.

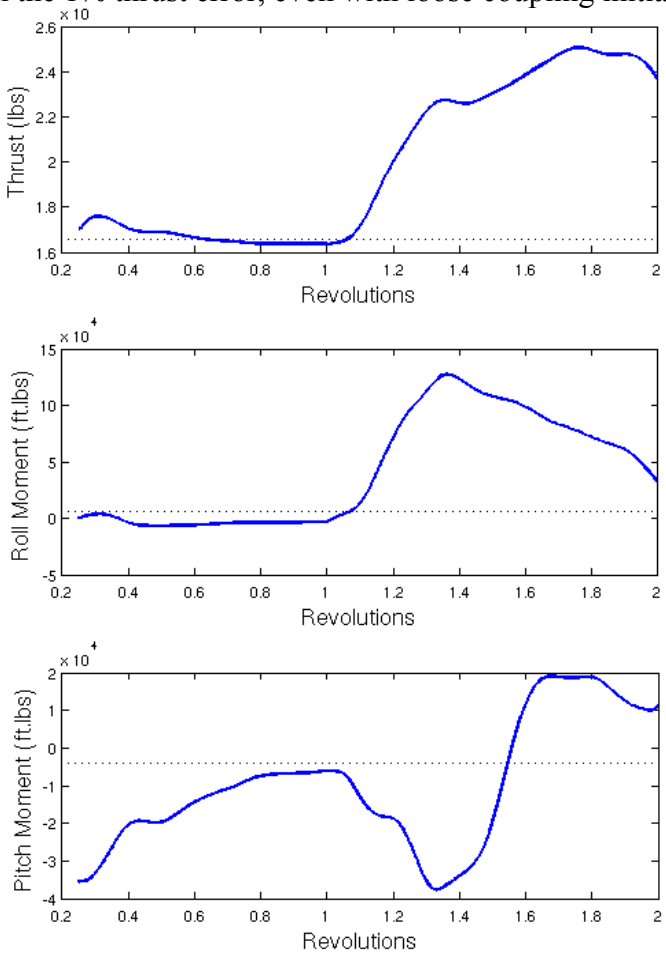


Figure 15: Example of the large transients that occur when kriging is applied immediately upon transfer from loose to tight coupling (at revolution 1).

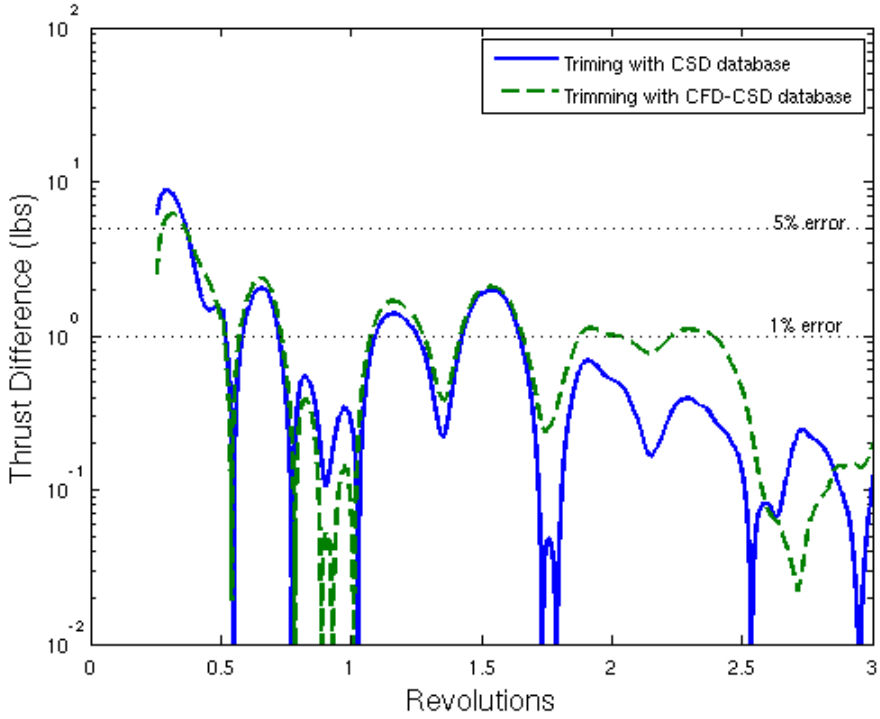


Figure 16. Tight coupling trim may be achieved for some cases with a training database provided from a comprehensive code.

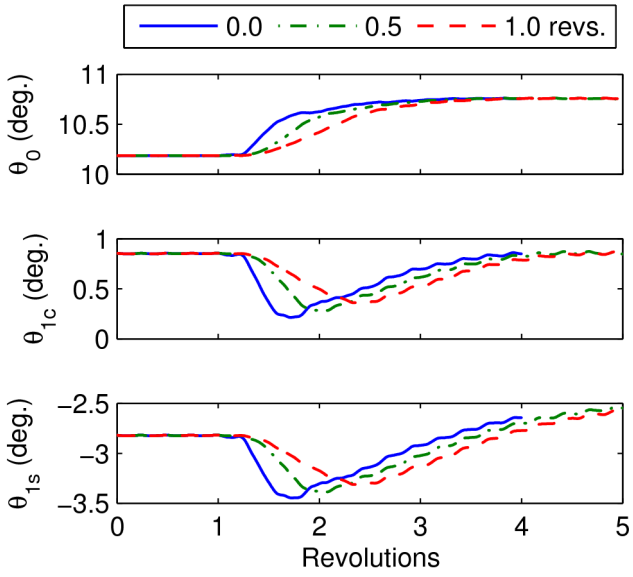


Figure 17: Pitch control history, varying length of transition between low-order and CFD loads. (From Reference 60)

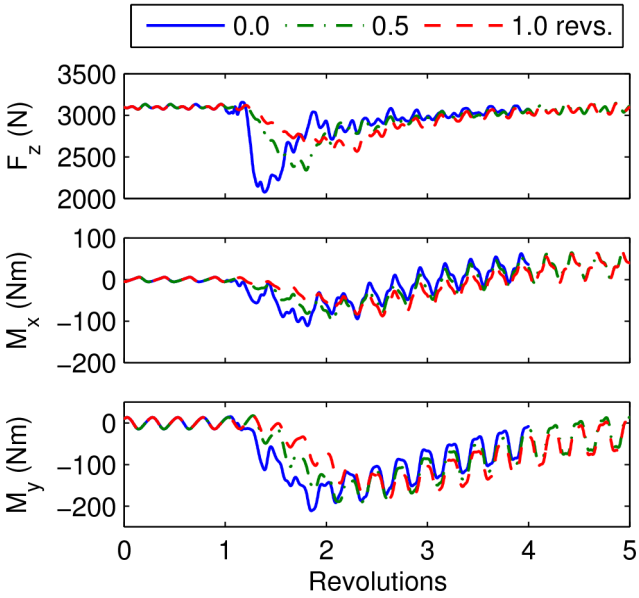


Figure 18: Hub loads history, varying length of transition between low-order and CFD loads. (From Reference 60)

7. CFD/CSD Tight Coupling Demonstrations

The CFD solvers, OVERFLOW and FUN3D, have been evaluated using the new tight coupling implementation and kriging controller. Several test cases involving different physics (UH60-A cases, Table 6, and HARTI baseline) have been evaluated. Application of the algorithm to obtain stability characteristics has also been demonstrated. Optimization of the tight coupling has also been accomplished to minimize the required computational time to obtain tightly-coupled results.

Table 6: UH60-A Test Cases

Case	Rotor Speed (RPM)	Density (slug/ft^3)	Temperature (F)	Airspeed (ft/s)	Pitch (deg)	Sideslip (deg)
c8534	258.1	0.0020823	71.814	266.5	-4.31	1.27
c9017	255.8	0.0013242	24.761	170.2	2.85	-1.59
c11029	257.3	0.0020066	35.43	258.5	-3.84	-1.69

Case	Thrust (lb)	Roll Moment (ft.lb)	Pitch Moment (ft.lb)
c8534	16602	6042	-4169
c9017	16452	379	-138
c11029	15727	5188	-4617

7.1. Grid Descriptions

UH60-A Coarse Mesh

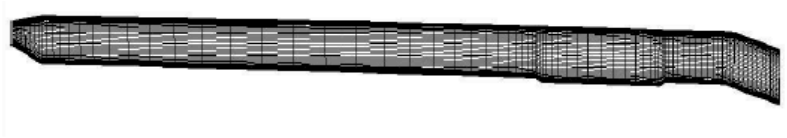
The structured coarse mesh contains 44 grids for 5.2 million cells, approximately 1.9 million of which are in the near field. The fine mesh contains 118 grids and has 80.9 million points with 14.3 million cells in the near field. Both were run with 4^{th} order spatial accuracy with time steps corresponding to $1/20^{th}$ of a degree blade motion in azimuth. Airstations were taken between radial nodes, as is usual in a structured grid.

Each blade, shown in Figure 19a), has 81 spanwise stations, which are clustered at the root and tip (Figure 19b)) with 105 surface nodes in the chordwise direction at each station. Each blade has a tip and root cap grid at each of the ends, however these grids were not considered for calculating forces and moments due to their small size and effect. The complete overset mesh contains all four blades (Figure 19c)), and has 44 grids for a total of 5.165 million points. The spacing normal to the blade surfaces yield a y+ of approximately one over the blade. While this is not the optimal grid for determining in particular the drag of the system, it is sufficient to demonstrate the ability of the CFD-CSD tight-coupling procedure to perform a rotor stability analysis.

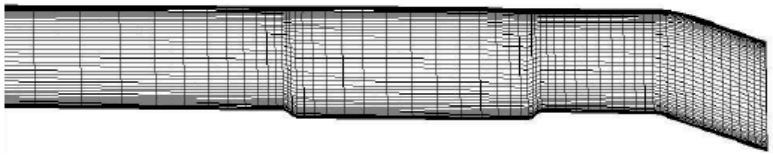
A refined grid was also examined for a limited number of runs. This grid is over fifteen times finer than the coarse grid, having a total of 80.9 million grid points while the coarse grid has just 5.2 million. The fine grid contains twice as many spanwise stations as the coarse grid (162 for the fine grid, and 81 for the coarse grid), and nearly twice as many chordwise grid points as the coarse grid (249 for the fine grid and 125 for the coarse grid). The fine grid also contained a denser near field than the coarse grid did, with 14.3 million cells compared to just 1.9 million for the coarse grid, while maintaining a proportionally finer far field. The fine grid

had 82% of the grid cells located in the far field versus just 63% for the coarse grid, allowing the fine grid to better preserve flow features.

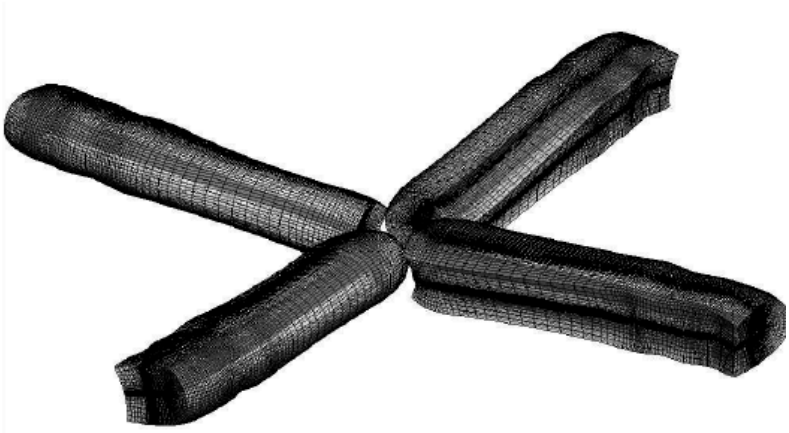
The unstructured grid used by FUN3D contains 40.3 million tetrahedrals. Time steps of 1-degree azimuth blade motion were taken using second order accuracy with 10 subiterations. The airstations used in coupling were taken to correspond to the coarse structured grid airstations, except at the tip where they were coarsened so that the FUN3D slicing algorithm would not skip a station when the tight coupling began.



(a) Blade surface



(b) Blade tip surface



(c) 4-bladed near-body rotor grid

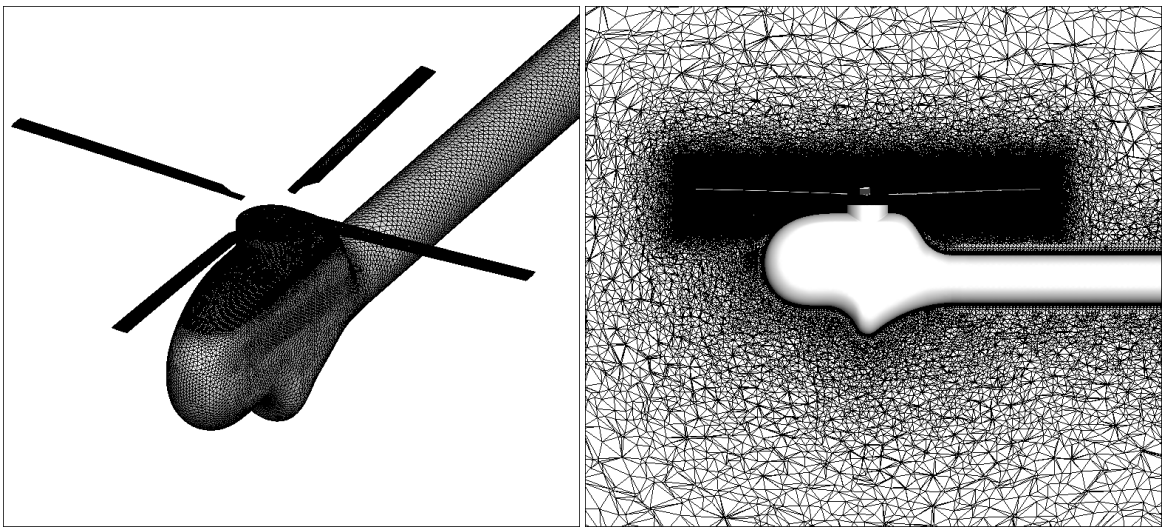
Figure 19. UH60-A CFD structured coarse mesh.

HART-II Mesh

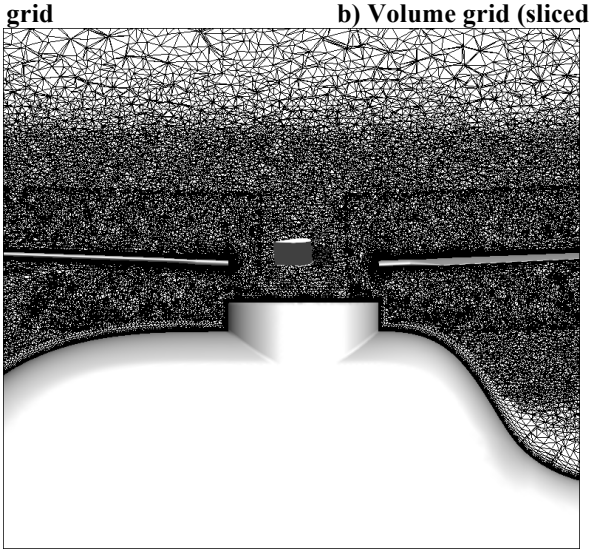
The mesh used for these simulations (Figure 20) included a total of 13.6 million nodes. The relevant mesh statistics are summarized in **Table 7**. This mesh is identical to that used by Biedron and Lee-Rausch in 2008, and as described in Reference 67, this mesh was the finest of a set of meshes used in a grid convergence study.

Table 7: Dimensions of each component of the composite mesh. All cells are tetrahedra.

Component	Nodes	Boundary faces	Cells	Grid Generator
Blades (each of 4)	1,156,735	114,302	6,731,961	VGRID
Fuselage/background	8,971,420	63,026	52,733,053	VGRID
Total	13,598,360	520,234	79,660,897	



a) Surface grid **b) Volume grid (sliced through centerline)**



c) Volume grid (close-up of hub region)

Figure 20: Illustrations of the 13.6 million node composite mesh used for HART-II simulations.

7.2. Level Flight Trim Demonstrations

There are two current methods of accomplishing the coupling within the rotorcraft community. The first method is the one applied by Potsdam et al. (Ref. 9), which updates the solution between coupling updates at every *1/n* revolutions, where *n* is the number of blades in the rotor. The other method monitors the periodicity of the CFD solution, and a coupling iteration is only performed when the rotor loads at a given reference radial station are periodic. The *1/n* update required eight coupling iterations to converge, while the periodic update required five coupling iterations. Although the latter method only needed five coupling iterations, the total time to achieve comparable convergence was a ½ revolution more than the *1/n* update method.

In tight coupling, the same trim procedure described above would be *extremely* expensive since the CFD solver would have perform as many revolutions as the CSD solver. For this effort, trim is computed with a conventional loose delta coupling method until convergence has been reached, after which the simulation is switched to a tight coupling algorithm. The initial loose coupling procedure (for trimming) follows that used by Abras[61] and Biedron and Lee-Rausch[67]. The only difference is that the new quasi-steady trimmer (Section 4) was used in place of the auto-pilot trimmer.

The baseline case applied a Spalart-Allmaras turbulence model. Menter's kω-SST two-equation model was compared, as well as two hybrid LES methods. The first of these was the Kinetic Eddy Simulation (KES), which is a Large Eddy Simulation/Very Large Eddy Simulation (LES/VLES). The second is a hybrid RANS-LES method (HRLES), that employs the Menter kω-SST RANS turbulence model near a wall and solve the k-equation, along with a subgrid scale model away from the way in the zones of separated flows. Both the HRLES and KES methods converged quickly to the nominal thrust coefficient, but did not converge on the moment coefficients, but instead fluctuated periodically about the nominal trim value. The Menter kω-SST model had trouble with the coarse grid used in this simulation, as has been observed in prior applications on various grids for the UH60-AA flight test cases. A typical radial section of the pitching moment is shown in Figure 21. The Spalart-Allmaras turbulence model over predicts the pitch down in the third quadrant, as well as anticipating the pitch down by almost 40^o. Overall, the HRLES simulation best predicts the phase of the pitch down moment, although it still over predicts the magnitude of the pitching moment.

OVERFLOW has four dissipation schemes, which smooth different variable combinations to achieve numerical dissipation. The default scheme is the TLNS3D, which was used in the baseline case, and it smoothes the total enthalpy, $\rho h_0 = \rho\varepsilon_0 + p$. Another option is the F3D scheme, which smoothes the flow parameter $\rho\varepsilon_0$ by mixing 2^{nd} and 4^{th} order smoothing. An organized attempt has been made to reduce the floating-point operation count in F3D by scaling the pressure sensor by the local rather than free stream Mach number. The ARC3D scheme is very similar to the F3D scheme, but it has slightly different coefficients in its 2^{nd} and 4^{th} order smoothing functions. Matrix dissipation is the final dissipation scheme. The other schemes are scalar based schemes, and matrix dissipation is intended to improve the quality of the solution. It is more computationally intensive, but typically more accurately resolves shocks and boundary layers.

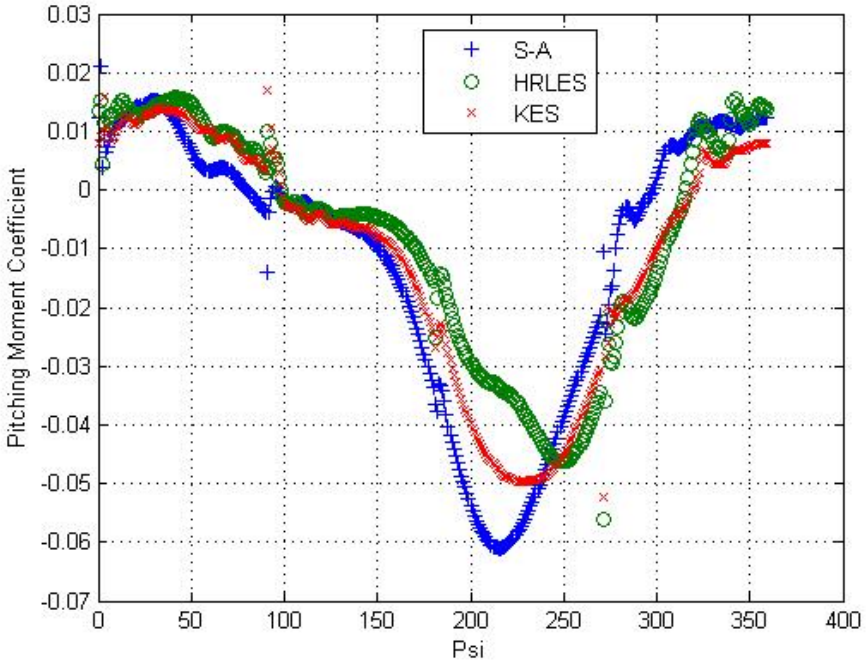


Figure 21: Comparison of the influence of the turbulence model predictions on pitching moment at a typical radial section.

During the simulations, it was found that the F3D and matrix dissipation schemes lacked robustness for this grid and flight conditions, and they did not achieve convergence. The ARC3D dissipation scheme converged quickly on the thrust coefficient, but its prediction of the moment coefficients fluctuated periodically about the nominal trim values. Although this periodicity in moments were observed, overall, it appears that the ARC3D dissipation scheme may be a better option to apply rather than the original TLNS3D scheme. The normal force and pitching moment predictions with the TLNS3D scheme missed several salient features of the simulation at various radial stations in the fourth quadrant, and typically produced the third quadrant pitching moment dip with a 40-degree phase lead over the flight test data. The ARC3D dissipation scheme on the other hand did not show as large of a phase lead (0 – 10 degrees), but the magnitudes of the features remained the same.

UH60-A Structural Model

The Dymore multi-body finite element analysis code, developed at Georgia Tech provides the structural dynamics module for this effort. Dymore can be applied to arbitrary nonlinear elastic systems, and has been previously utilized using OVERFLOW for CFD-CSD loose-coupling. Dymore includes an extensive library of multi-body components to model the mechanical components of a rotor system so that it can be applied to new topological designs with existing library elements or by the addition of new elements. For modeling flexible rotors, Dymore uses geometrically exact finite elements based on formulations developed by Simo[62], which have been extended to beams and shells.

The UH60-A rotor is a fully articulated system that exhibits all possible motions and thus requires that all the hinge motions be modeled. The articulated motion consists of pitch, flap, and lead-lag components with higher harmonic content greater than zero, as well as hinge offsets and shaft tilt. This example deals with a detailed aeroelastic model of the UH60-A rotor system shown in Figure 22. This problem involves both structural and aerodynamic states. The structural model involves four blades connected to the hub through blade root retention structures and lead-lag dampers. Each blade was discretized by means of ten cubic finite elements. The root retention, connecting the hub to the blade, was separated into three segments and modeled by one, two and two beam elements, respectively, labeled segment 1, 2, and 3, respectively. Three revolute joints connecting the first two segments of the root retention structures described the flap, lead-lag and pitch hinges of the blade. Prismatic joints were used to model the lead-lag dampers, assumed to be dashpots with linear properties. The complete structural model involved 5,656 states.

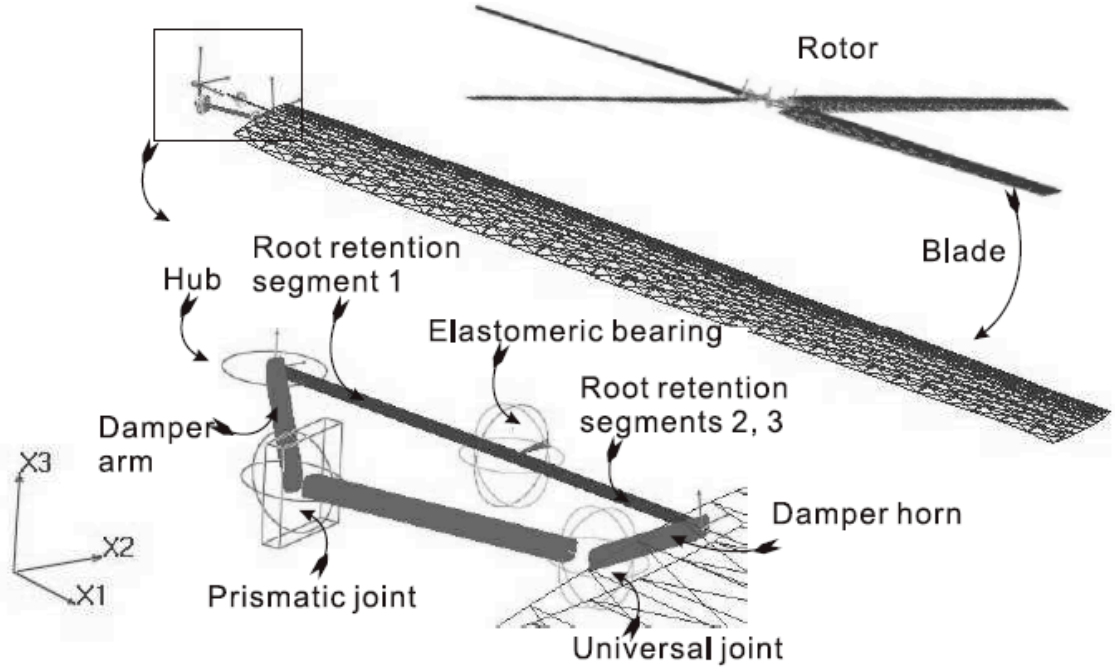
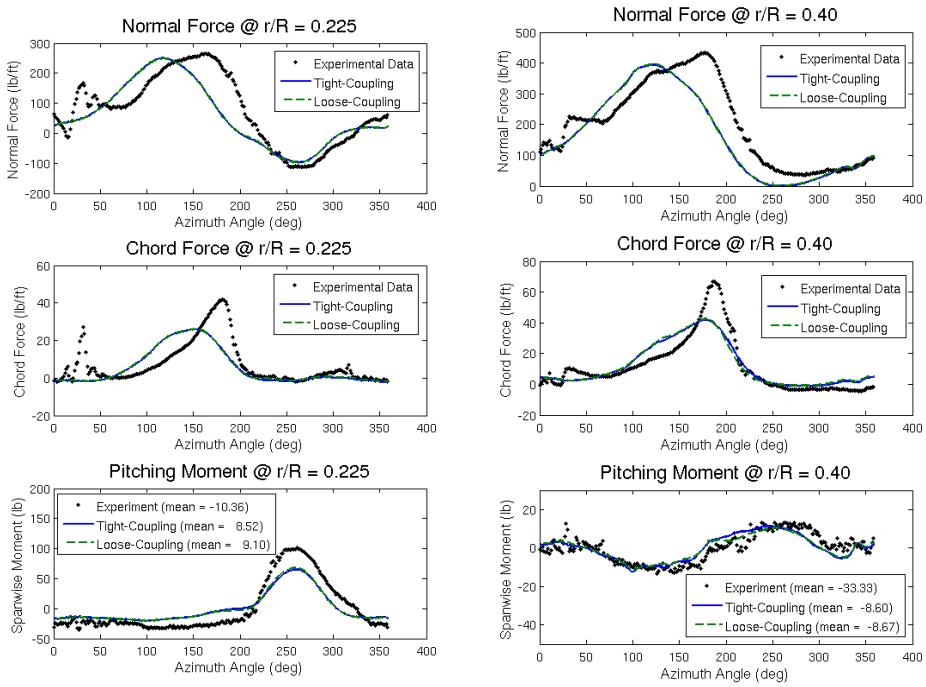


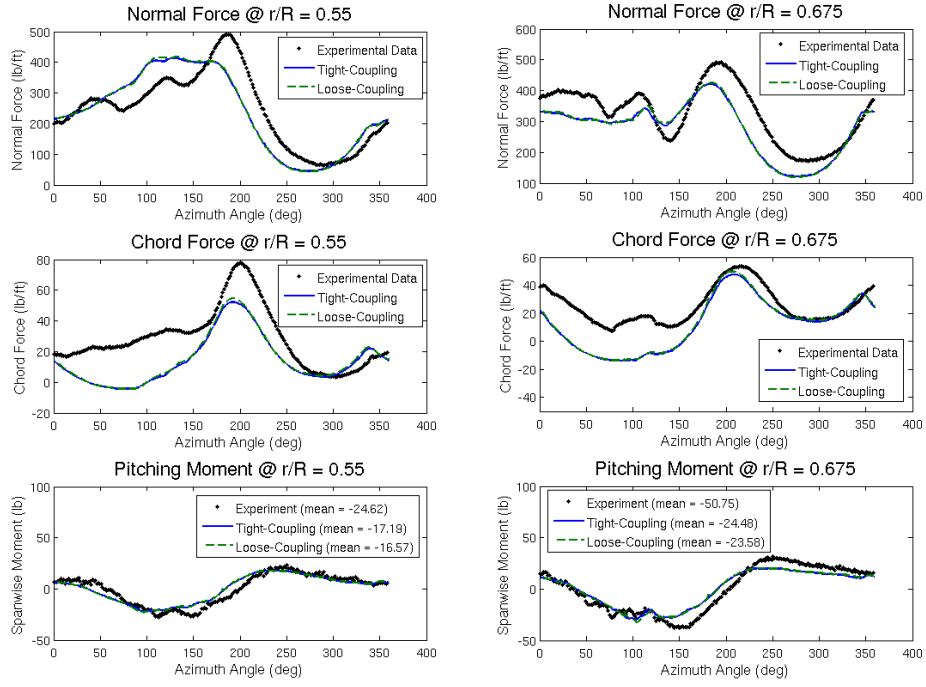
Figure 22. Schematic of the UH60-A rotor system.

UH60-A C8534

This case was utilized to examine whether the tight and loose coupling would result in identical airloads. To examine this, first the coarse structured grid was resolved using the OVERFLOW-Dymore coupling. The predicted airloads for both coupling approaches are compared with experiment in Figure 23. These figures indicate that the difference in the loose and tight coupling, run to convergence in both methods do not yield any discernable differences in the predicted results. This is also true for the structural moments (Figure 24 and Figure 25) and the predicted tip deflections (Figure 26). The differences in the simulations and experimental data are comparable to those predicted by loose coupling from other researchers, such as Potsdam et al.[63], within the context of the different CSD solver.

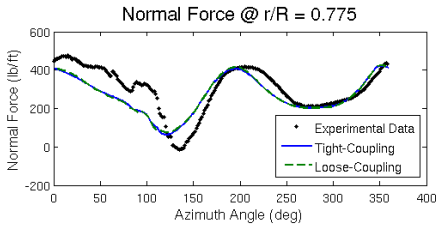


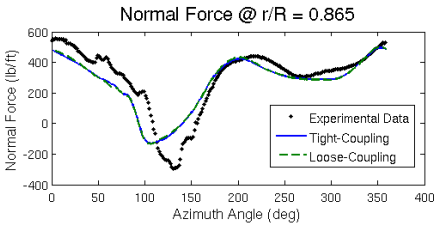
a) r/R = 0.225 **b) r/R = 0.40**

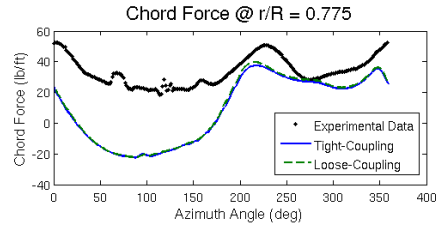


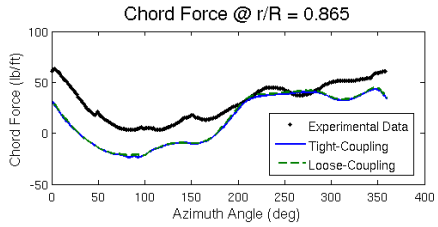
c) r/R = 0.55 **d) r/R = 0.675**

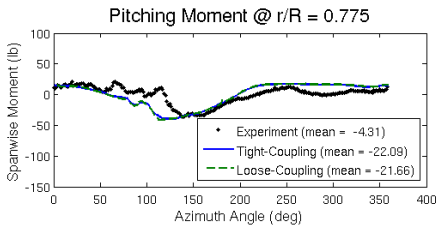
Figure 23: UH60-A airloads computed with OVERFLOW-Dymore using the coarse CFD grid.

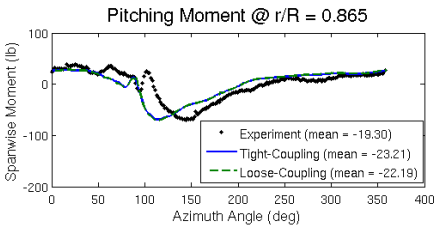




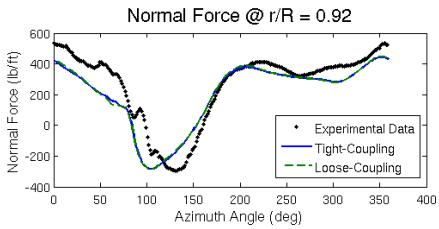


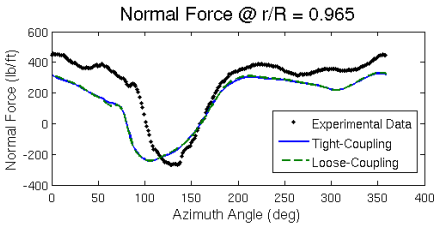


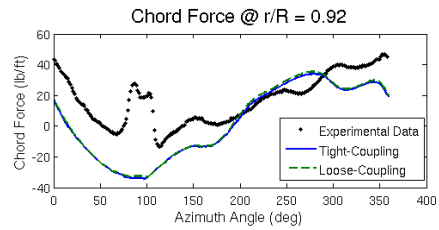


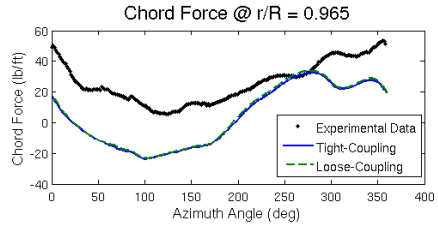


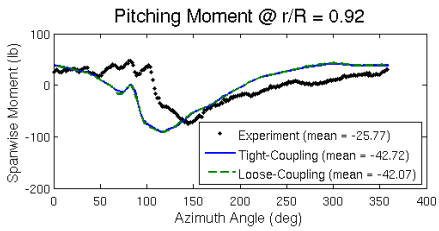
e) r/R = 0.775 **f) r/R = 0.865**

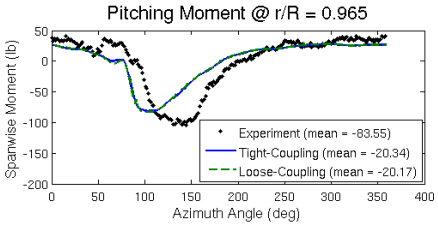






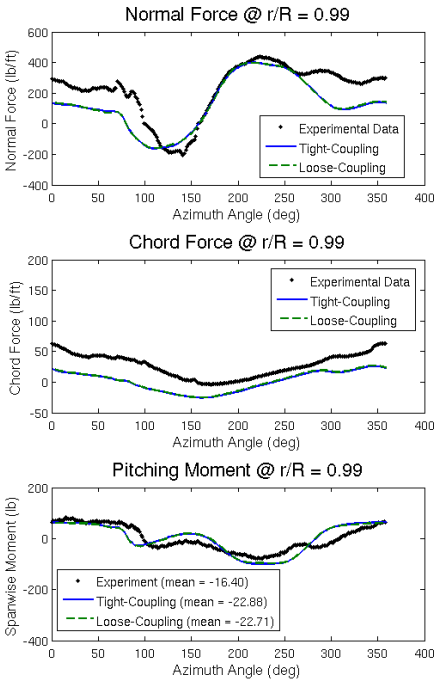






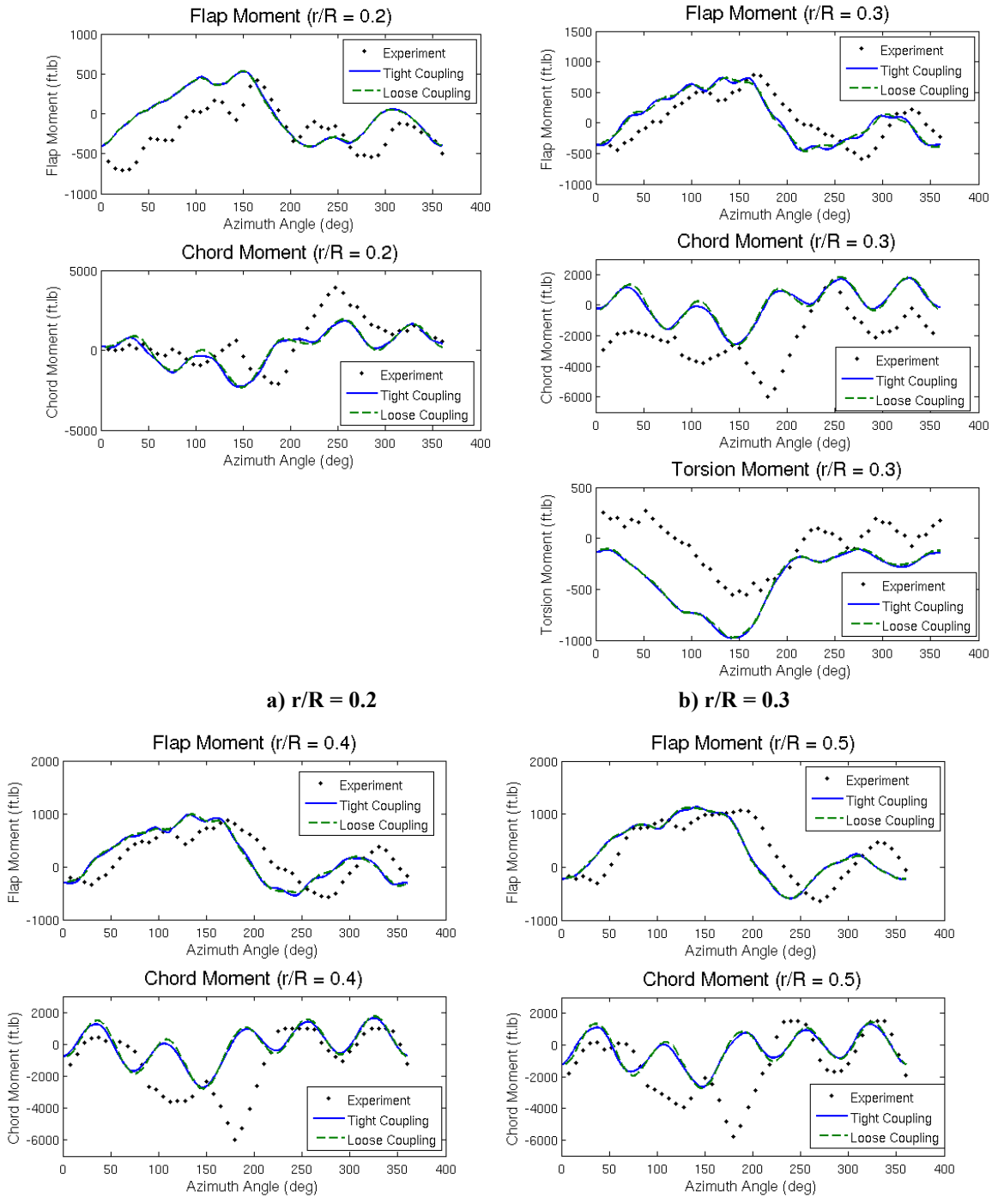
g) r/R = 0.92 **h) r/R = 0.965**

Figure 23 **(cont.): UH60-A airloads computed with OVERFLOW-Dymore using the coarse CFD grid.**



i) r/R = 0.99

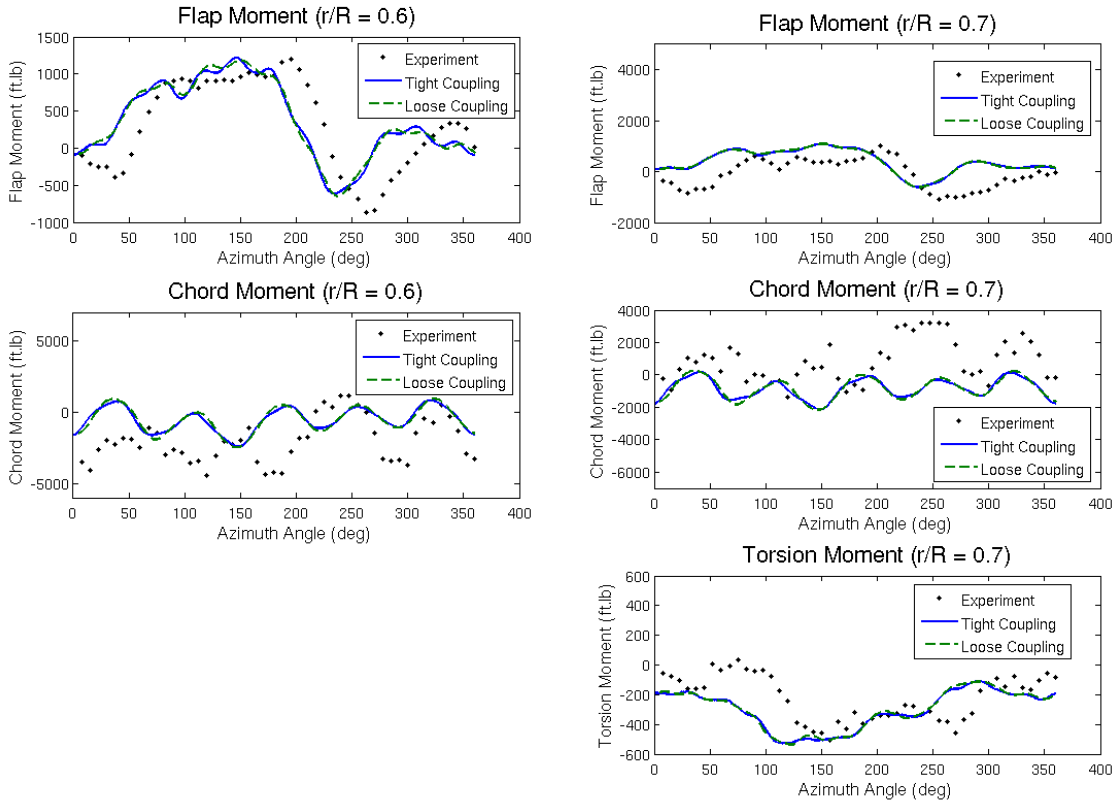
Figure 23 **(cont.): UH60-A airloads computed with OVERFLOW-Dymore using the coarse CFD grid.**



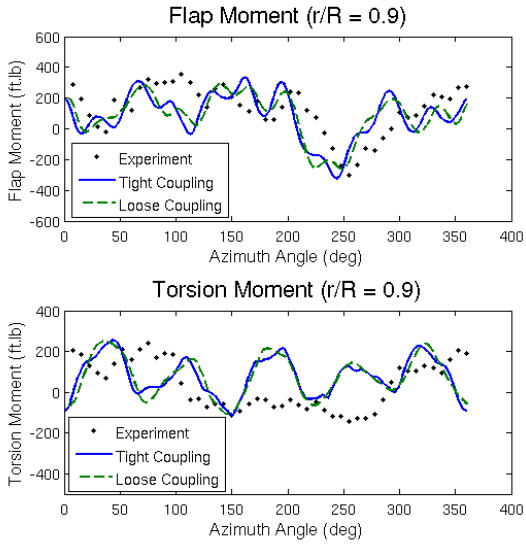
a) r/R = 0.2 **b) r/R = 0.3**

c) r/R = 0.4 **d) r/R = 0.5**

Figure 24: Inboard UH60-A structural moments computed with OVERFLOW-Dymore using the coarse CFD grid.



a) r/R = 0.6 **b) r/R = 0.7**



c) r/R = 0.9

Figure 25: Outboard UH60-A structural moments computed with OVERFLOW-Dymore using the coarse CFD grid.

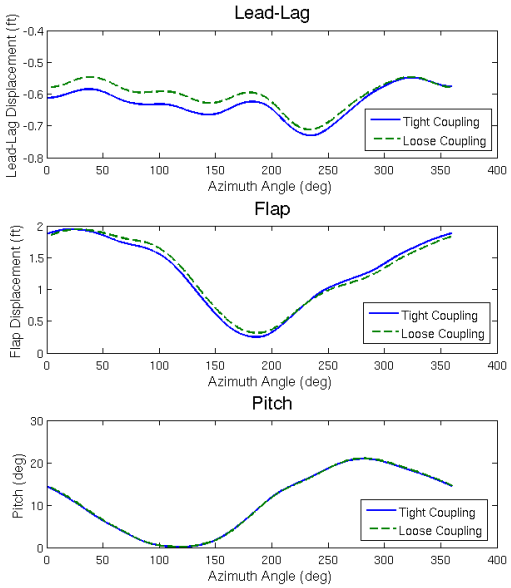


Figure 26: UH60-A tip deflections computed with OVERFLOW-Dymore using the coarse CFD grid.

The influence of the grid was next evaluated for both the prediction capabilities and its influence on the tight coupling convergence. As observed in Figure 27, the airloads for this case change only minimally for the refined grid. These changes, which consist primarily of a phase shift and mild amplitude adjustments, are comparable again to those observed by Potsdam et al.[23] using the same grids, but CAMRAD-II to resolve the structural model.

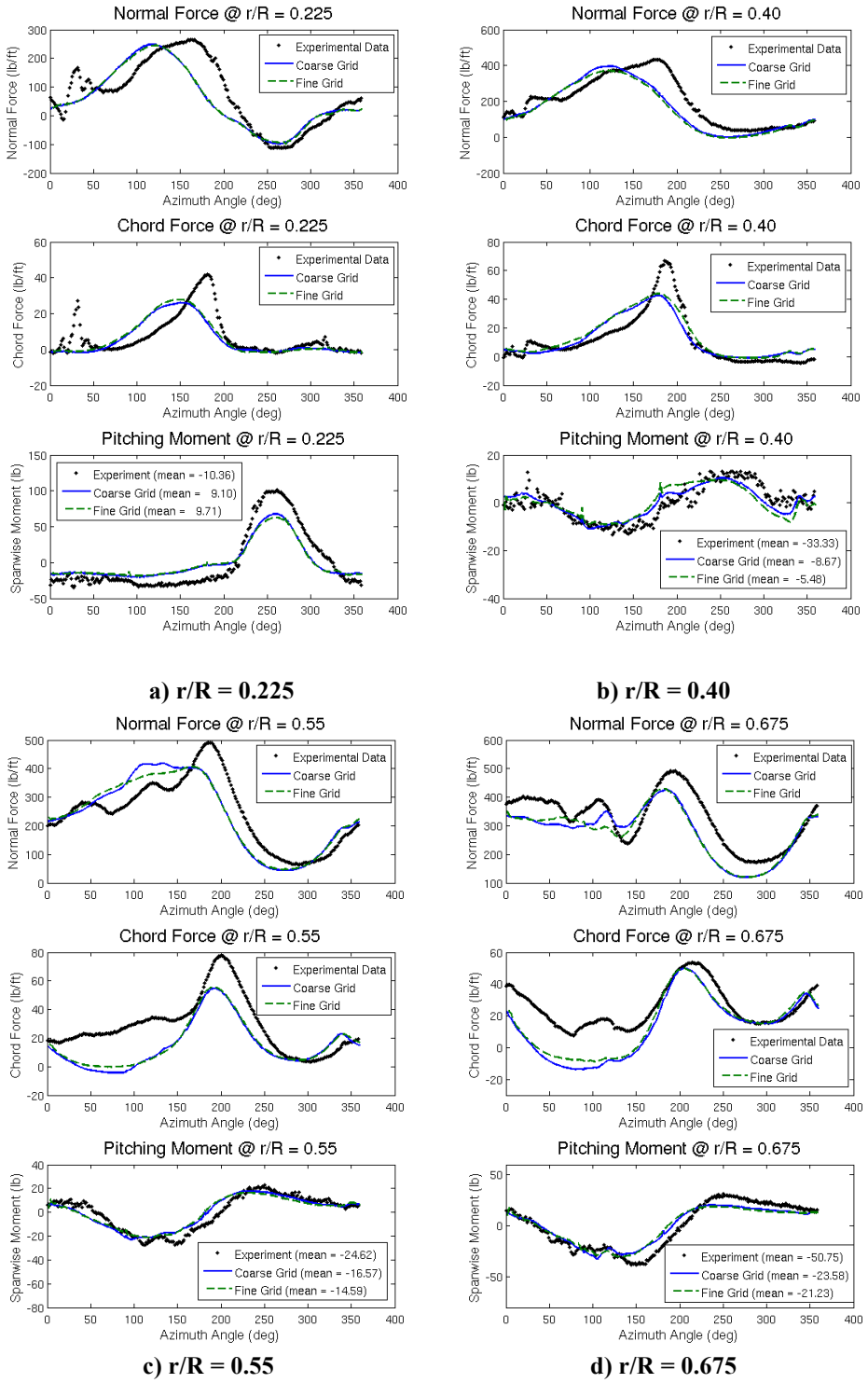
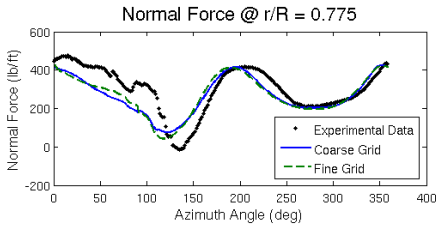
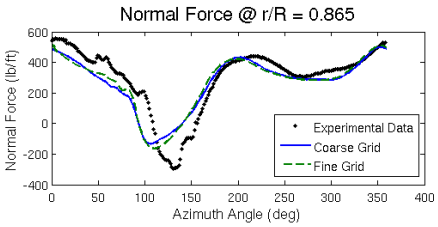
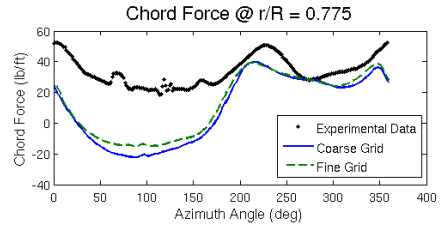
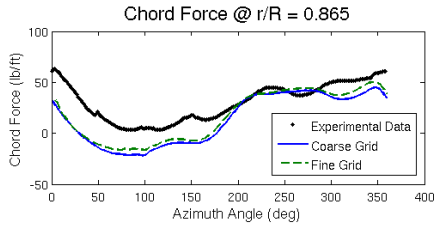


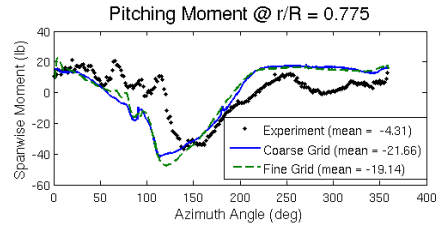
Figure 27: UH60-A airloads computed with OVERFLOW-Dymore using refined CFD grids.

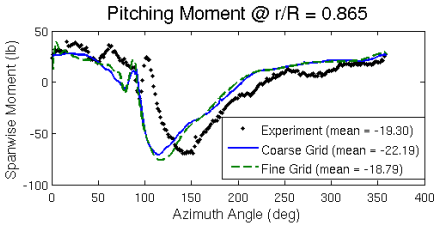




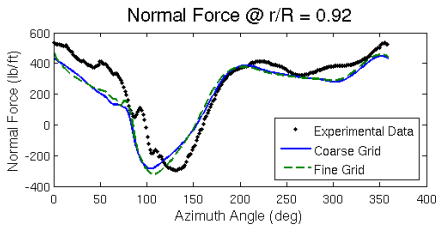


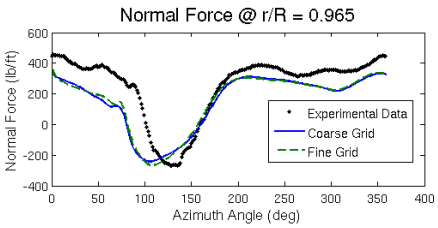


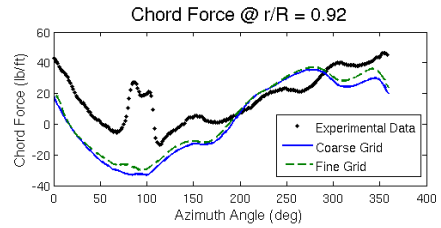


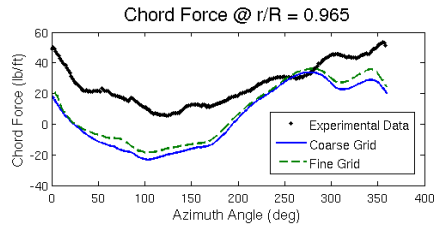


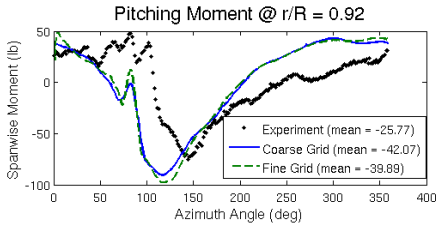
g) r/R = 0.775 **h) r/R = 0.865**

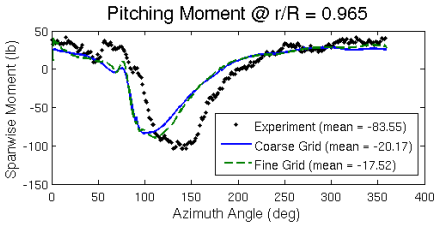






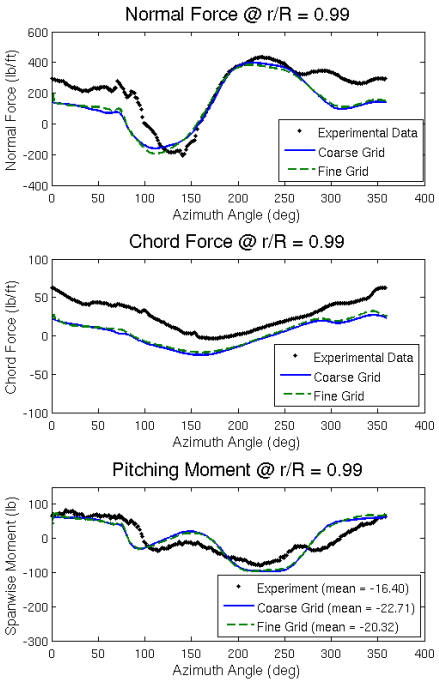






i) r/R = 0.92 **j) r/R = 0.965**

Figure 27 (cont.): UH60-A airloads computed with OVERFLOW-Dymore using refined CFD grids.



k) r/R = 0.99

Figure 27 (cont.): UH60-A airloads computed with OVERFLOW-Dymore using refined CFD grids.

UH60-A C9017

Case 9017 poses a more interesting case for loose and tight coupling, as dynamic stall occurs on the retreating side of the blade. Since dynamic stall is very sensitive to simulation parameters, the impact of tight coupling on the predictions has been a topic of speculation at the UH60-A airloads workshop. For this case, the coarse grid was again run to minimize computational time, but the hybrid RANS-LES (HRLES) turbulence model was selected to help quantify differences in the coupling approaches. Loose coupling was started after one revolution and updated every ¼ revolution until convergence. Tight coupling was started after ¼ revolution and controls were updated every degree (20 timesteps) during the revolution, with the Jacobian updated every 36 degrees. Once again convergence (Figure 28) occurred within approximately one revolution of each approach, depending on the point at which the solution isdeemed to be converged. Similar to the 8534 case, the loose and tight coupling airloads for the 9017 case for the most inboard radial sections show comparable predictions (Figure 29a) and b)). As one moves outboard, however, significant differences in the predictions appear, in particular for the retreating side of the rotor. Phase and amplitude differences result, with loose coupling results providing more accurate correlation to experiment in some instances, with tight coupling results more accurate in others. It is difficult to determine the cause of these differences in the prediction due to the plethora of numerical options that may affect this simulation (grid fidelity, turbulence model, time step/subiteration, etc.). These are further explored and discussed in the 2011 European Rotorcraft Forum article that is under preparation.

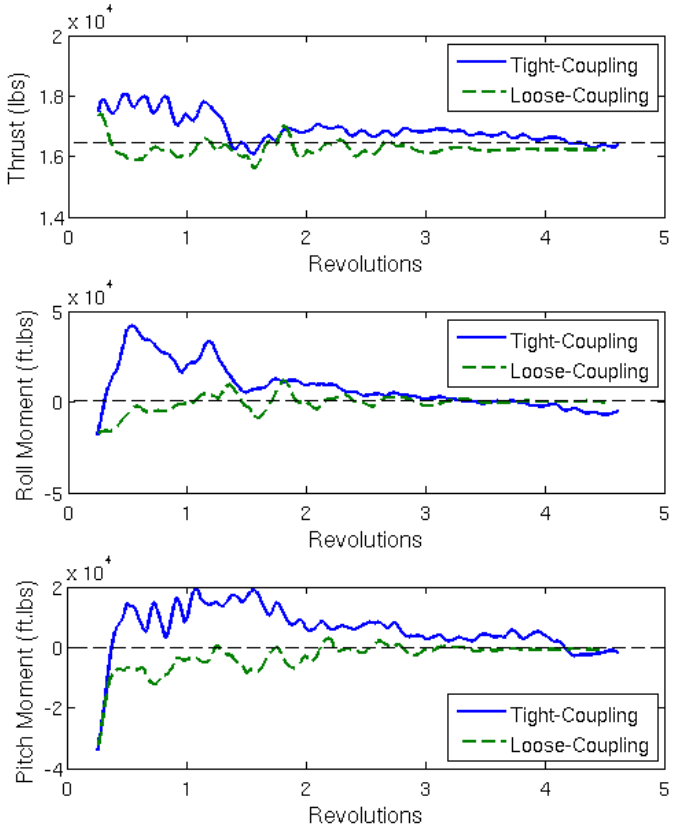
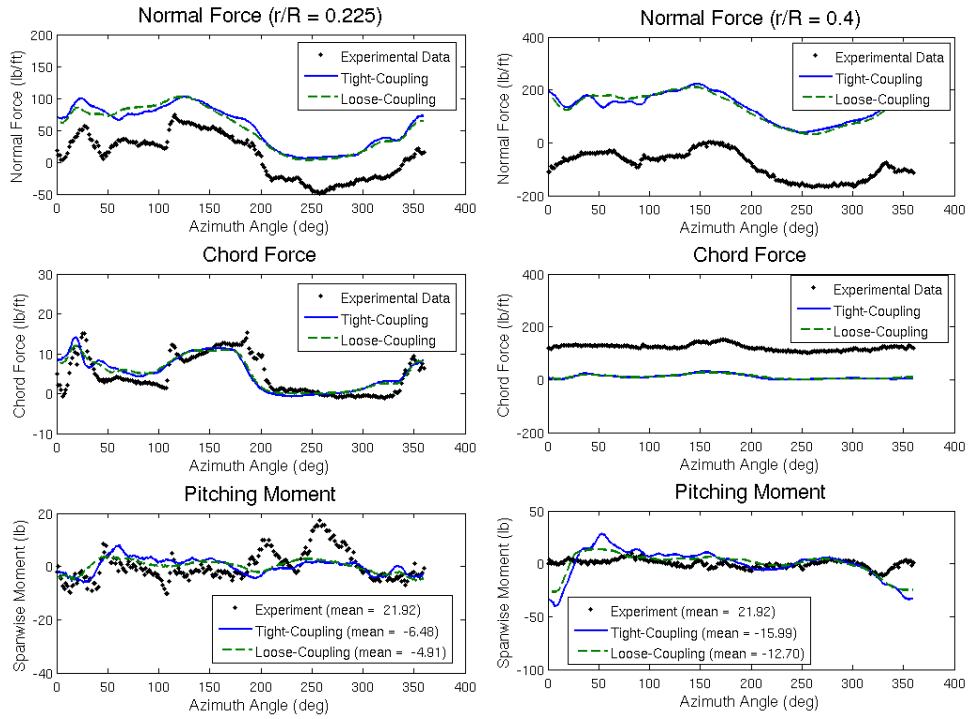
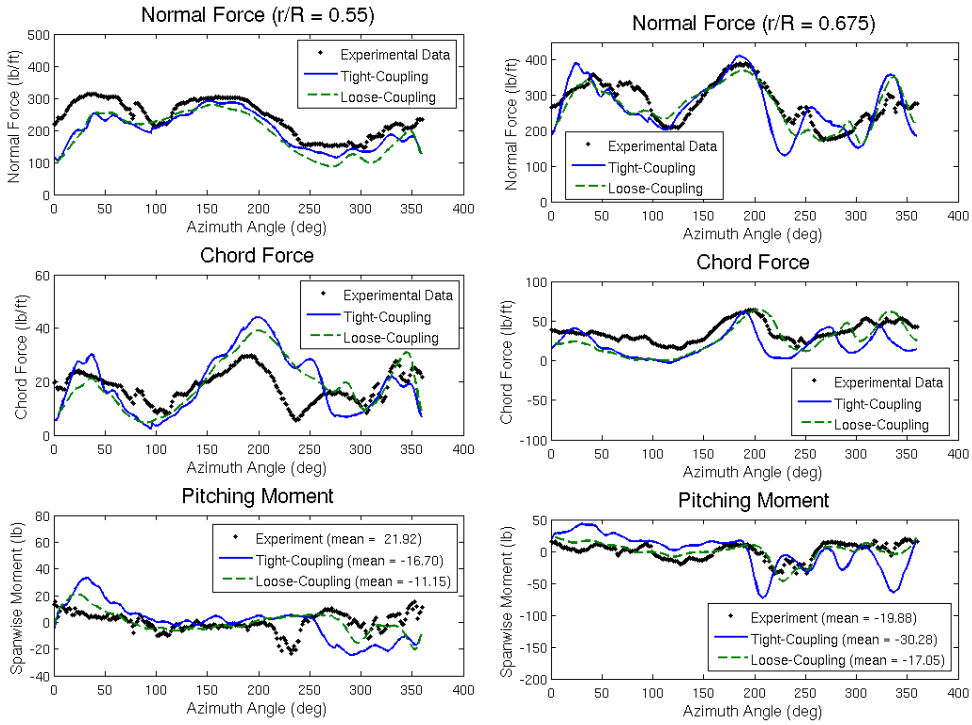


Figure 28: C9017 convergence with loose and tight coupling using OVERFLOW-Dymore on the coarse grid.

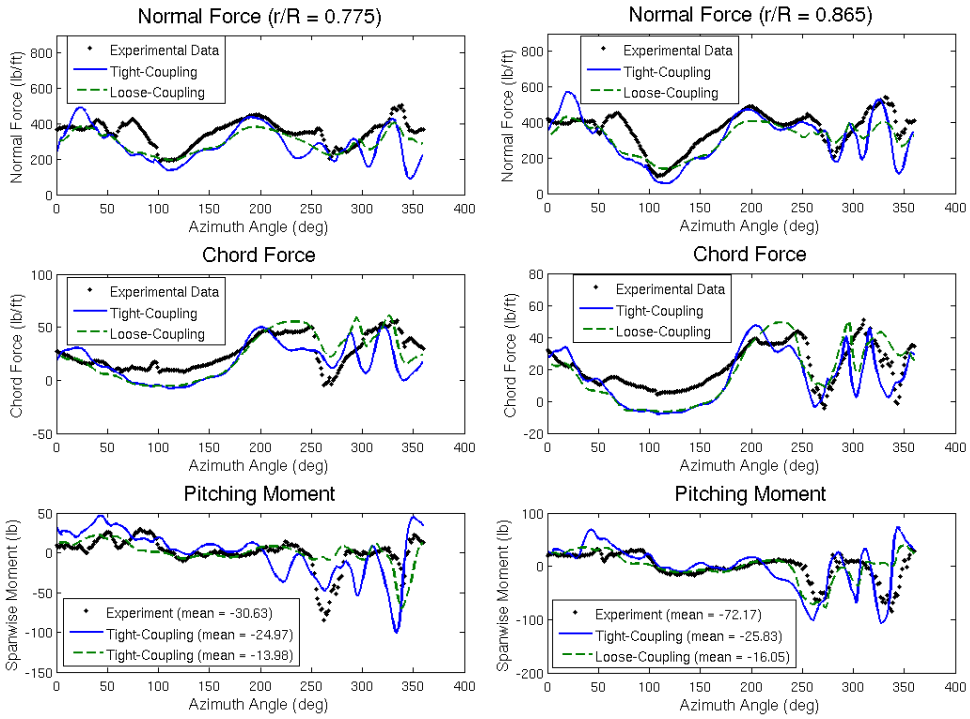


a) r/R = 0.225 **b) r/R = 0.40**

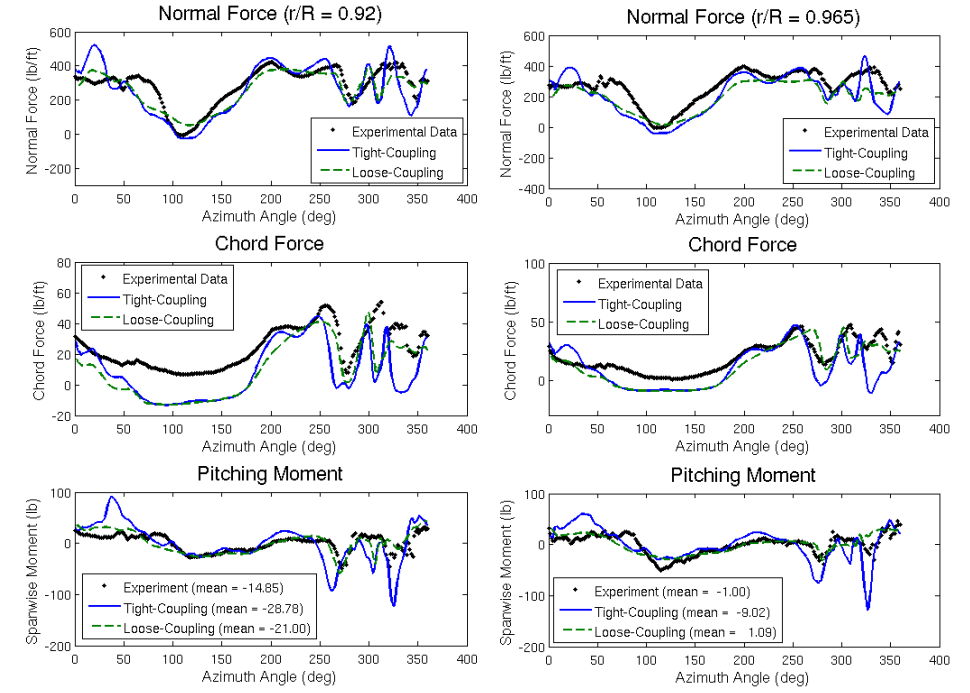


c) r/R = 0.55 **d) r/R = 0.675**

Figure 29: C9017 airloads predictions for loose and tight coupling using OVERFLOW-Dymore on the coarse grid with the HRLES turbulence model.

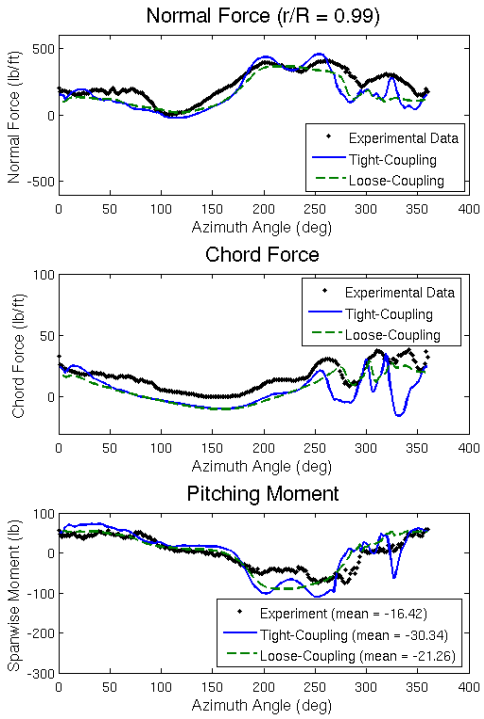


e) r/R = 0.775 **f) r/R = 0.865**



g) r/R = 0.92 **h) r/R = 0.965**

Figure 29(cont.): C9017 airloads predictions for loose and tight coupling using OVERFLOW-Dymore on the coarse grid with the HRLES turbulence model.



j) r/R = 0.99

Figure 29(cont.): C9017 airloads predictions for loose and tight coupling using OVERFLOW-Dymore on the coarse grid with the HRLES turbulence model.

HART-II Baseline

Another test case chosen to validate the coupling methodology is the HART-II rotor test[64], which was conducted in 2002 to study the effect of higher harmonic pitch control on rotor noise and vibration. The rotor was situated on a model fuselage, which was at the end of a long sting. Various geometric parameters for the rotor are given in Table 8. To emulate descending flight, the shaft was tilted aft by 5.4 degrees by tilting the entire fuselage/rotor assembly. This has been calculated to correspond to an effective shaft tilt of 4.5 degrees aft when wind tunnel walls are not present. As discussed previously, a number of researchers have demonstrated loose CFD/CSD coupling with the HART-II rotor[65,66,67].

Table 8: Geometric parameters for the HART-II rotor.

Rotor radius, R	2 m
Rotor chord, c	0.121 m
Solidity, σ	0.077
Pre-cone angle	2.5^{o}
Actual shaft tilt, α_{shaft}	5.4^{o} aft
Effective shaft tilt, α_{eff}	4.5^{o} aft
Rotor speed, Ω	108.9 rad/s

The HART-II rotor lacks flap and lead-lag hinges between the hub and the blades. The blades attach directly to the hub, and the inboard portion of the blade, which has an elliptical cross-section and is much stiffer than the outboard portions, is allowed to bend elastically to

absorb some of the bending moment that would otherwise be transferred to the hub. The Dymore model used in these simulations, shown in Figure 30, has a number of simplifications compared to the actual rotor. First, most of the hub hardware is omitted. The blades simply attach to a revolute joint at the hub axis. While the HART blade is physically a single piece, here each blade is modeled as two beams: a relatively stiff inboard flex beam and a more flexible outboard main blade beam. The flex beam is constructed from a single third-order finite element. The main blade consists of eight third-order elements. Figure 31 shows the relationship between the number of main blade elements and the first torsion frequency. With eight finite elements, the first torsion frequency can be computed to within 0.5% of the measured frequency. It is important to minimize the number of elements used in the model since Dymore's performance scales with $\mathcal{O}(N_e\, m^2)$, where N_e is the number of elements, and m is the bandwidth of the stiffness matrix. Since adding more one-dimensional beam elements does not change m, performance scales linearly with N_e.

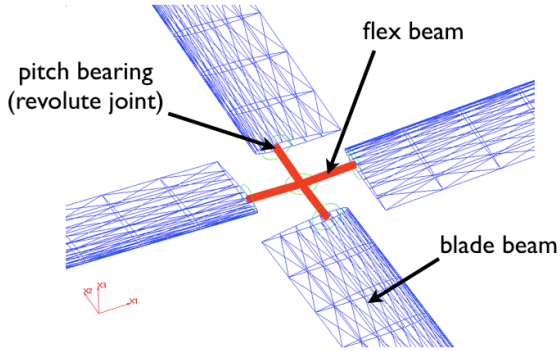


Figure 30: Hub of the HART Dymore model.

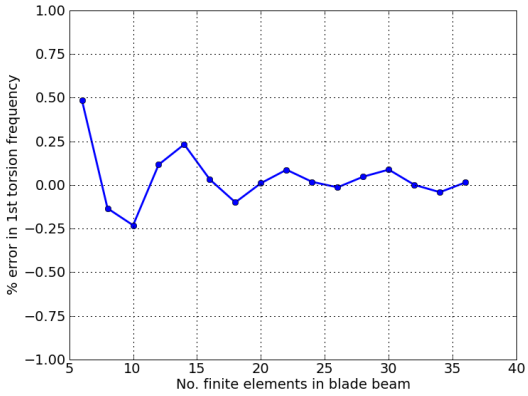


Figure 31: Convergence of the 1st torsion frequency with the number of finite elements in the blade beam

The pitch link is also omitted since there is a great deal of uncertainty as to its structural properties. To approximate the effect of the control system stiffness on the blade response, a torsional spring connects the flex beam and the main blade beam. In order for the torsional spring to provide the best possible approximation of the control system stiffness, the spring constant must be adjusted until the natural frequency associated with the first torsional mode matches measured values. This *tuning* process resulted in a stiffness of 1632.4 N-m/rad, yielding the target natural frequency of 419.57 rad/s. Figure 32 and Figure 33 show a fan plot and the first six mode shapes at the nominal frequency, respectively. The computational model follows

the trend of the measured frequencies, and the mode shapes are nearly identical to the measured modes.

As stated in Table 8, the HART blades had a small pre-cone angle of 2.5°, which was included in the Dymore model in the reference configuration. This means that the deflections output by Dymore are relative to the pre-coned state. The aft shaft tilt was applied to the model via the far field velocity used by the aerodynamic interface, which mirrors the manner in which shaft tilt was applied to the CFD model by setting the angle of attack to 4.5°.

The CFD portion of the coupled simulation followed as closely as possible the work of Biedron and Lee-Rausch[67], in order to provide a correlation of the loosely-coupled analysis prior to tight coupling. A time step corresponding to one degree of azimuthal rotation was used, and the number of Newton subiterations was dynamically determined by a temporal error controller[68], insuring that at each time step, residuals were always reduced to less than 5% of an estimate of the temporal error. On average, using the error controller resulted in about 30 Newton subiterations per step. To correlate with Reference 67, the Spalart-Allmaras turbulence model was selected.

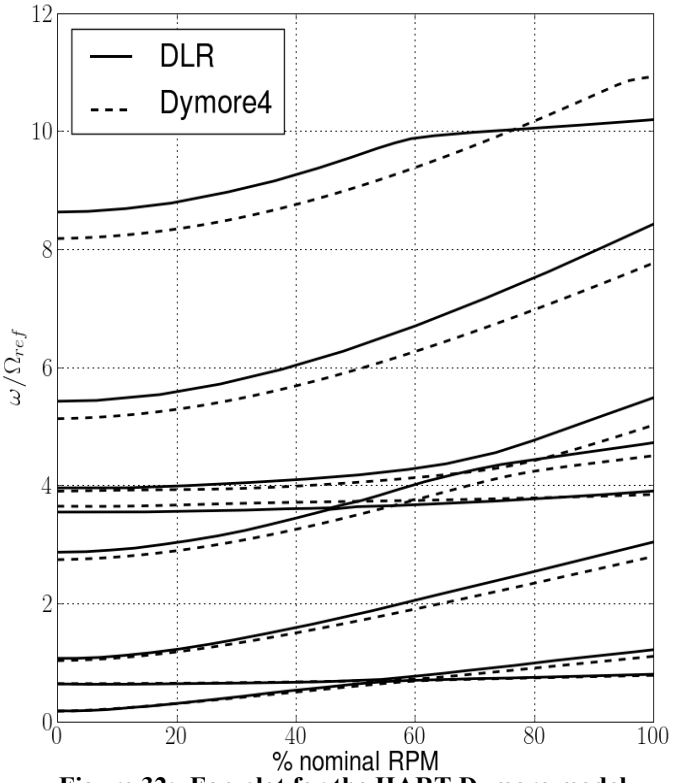


Figure 32: Fan plot for the HART Dymore model

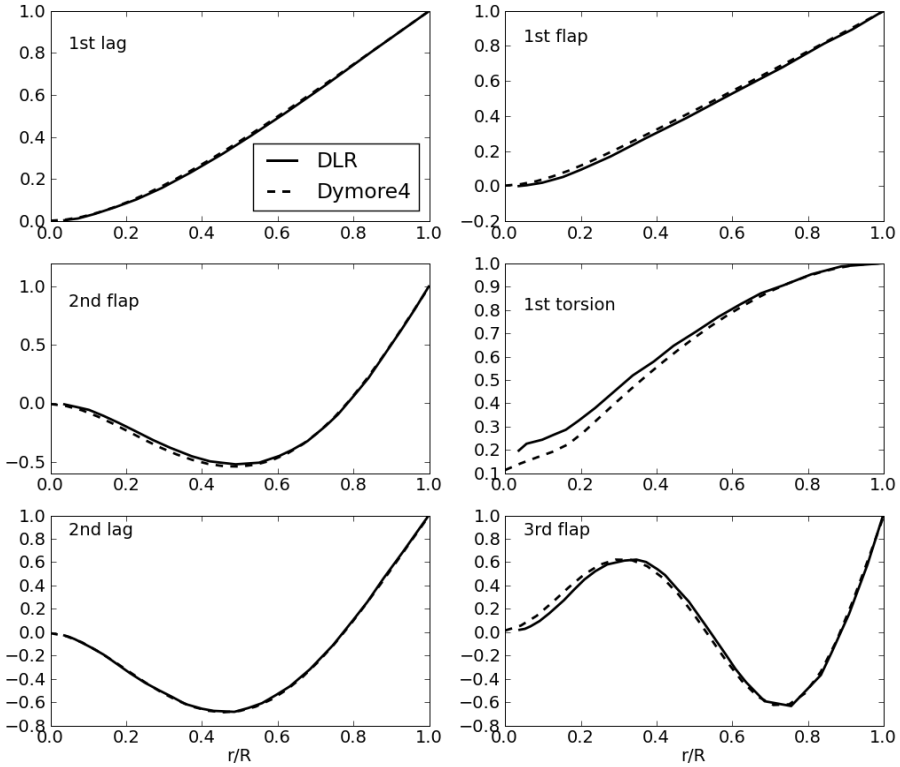


Figure 33: First six mode shapes for the HART Dymore model

The loose coupling phase was allowed to converge via five coupling iterations. The zeroth iteration was run for two revolutions, and subsequent iterations each ran for half a revolution. It should be noted that for a rotor with n_b blades a coupling iteration needs only to include $1/n_b$ of a revolution. At that point, the airloads from all the blades can be combined to create a map of airloads over an entire revolution. However, as the blade controls undergo a step change at the beginning of each coupling iteration, leading to transients in airloads, running $2/n_b$ revolutions allows the airloads to become quasi-periodic state before applying them to the Dymore model.

Figure 34 shows the convergence of blade collective and cyclic pitch, θ_o, θ_{1c}, and θ_{1s}, during the initial loosely-coupled trim phase. The controls change very little after the second iteration, though as will be shown later, the airloads continue to change. Only the Dymore portion of the sixth iteration was simulated. The controls from the sixth iteration were then fixed for the later tight coupling phase. The final controls are given in Table 9, along with controls computed via FUN3D/CAMRAD-II coupling from Reference 67.

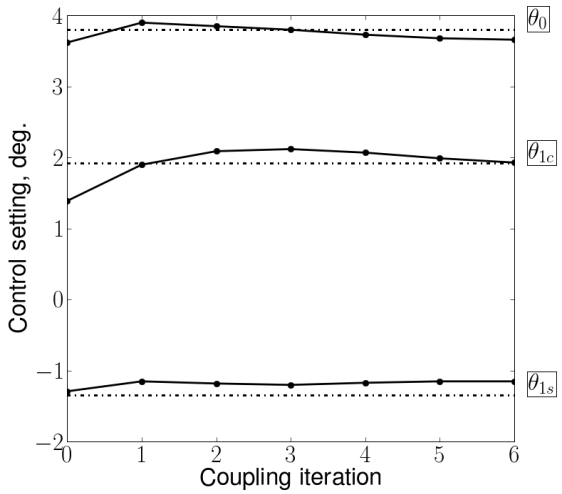


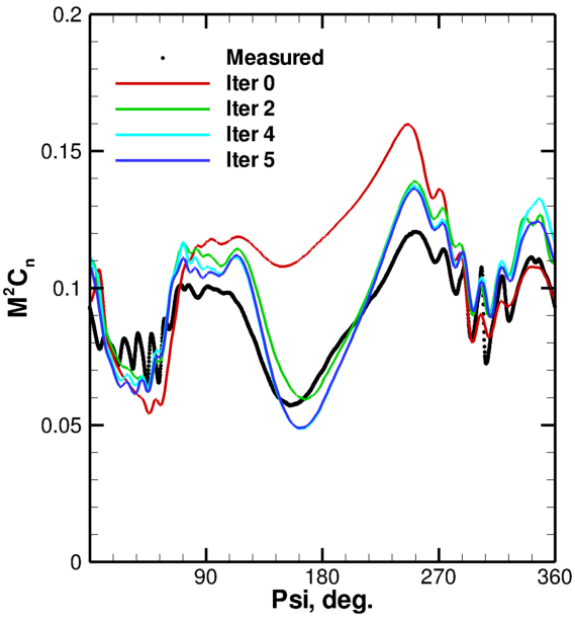
Figure 34: Convergence of blade pitch controls with loose coupling iterations

Table 9: Converged blade pitch controls, in degrees.

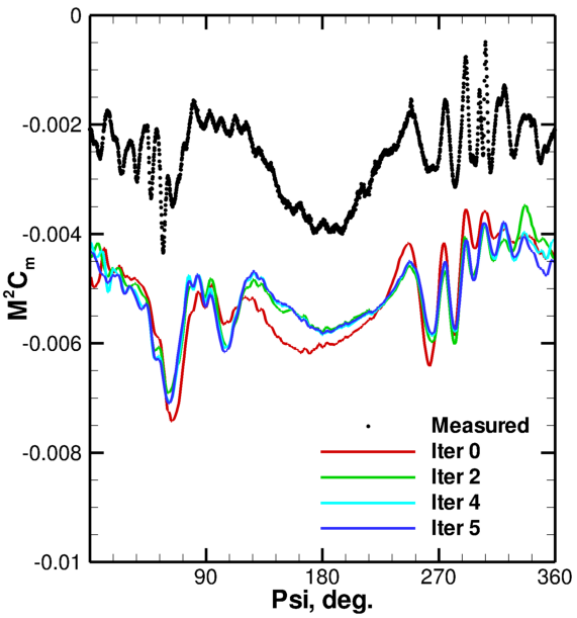
	θ_0	θ_{1c}	θ_{1s}
Measured	3.80	1.92	-1.34
Computed	3.66	1.93	-1.15
Computed (Ref. 67)	3.47	1.77	-0.97

The convergence of airloads at 87% span is shown in Figure 35. Prior work with the HART rotor shows that many predictions, including airloads and structural loads, have finite offsets from the measured values. Since the fluctuating component is of greater engineering interest than the mean value, Figure 35(c) shows pitching moment with the mean component removed to show the fluctuating component only.

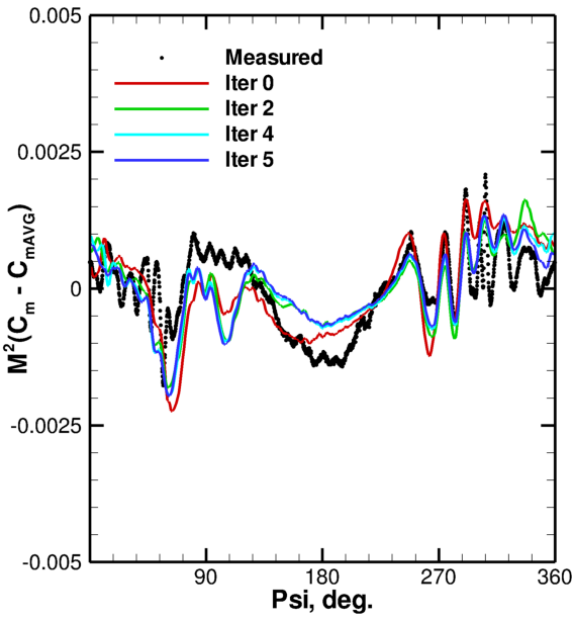
It is clear that by the fifth loose coupling iteration, the airloads have nearly converged on their final values. The primary differences between the ultimate and penultimate iterations are observed for the normal force predictions in the first and fourth quadrants, which feature prominent blade-vortex interactions. Since the controls remain relatively constant after the second iteration, it can be concluded that most of the adjustments made by Dymore after that point are elastic deformations in an attempt to resolve the BVI phenomena. In an effort to further establish the validity of the structural model, structural moments from the present simulation were compared against measured moments and those computed by Biedron and Lee-Rausch. The present loosely-coupled simulations (Figure 36) capture the character and magnitude of the flap moment better than FUN3D/CAMRAD-II. The torsion and lag moments in are nearly identical, and Dymore has the same difficulty as FUN3D/CAMRAD-II in capturing the lag moment, which is very lightly damped. For reasons discussed earlier, the second column of Figure 36 shows the structural loads with their means removed. Those mean values are listed in Table 10.



a) Normal force, C_nM^2

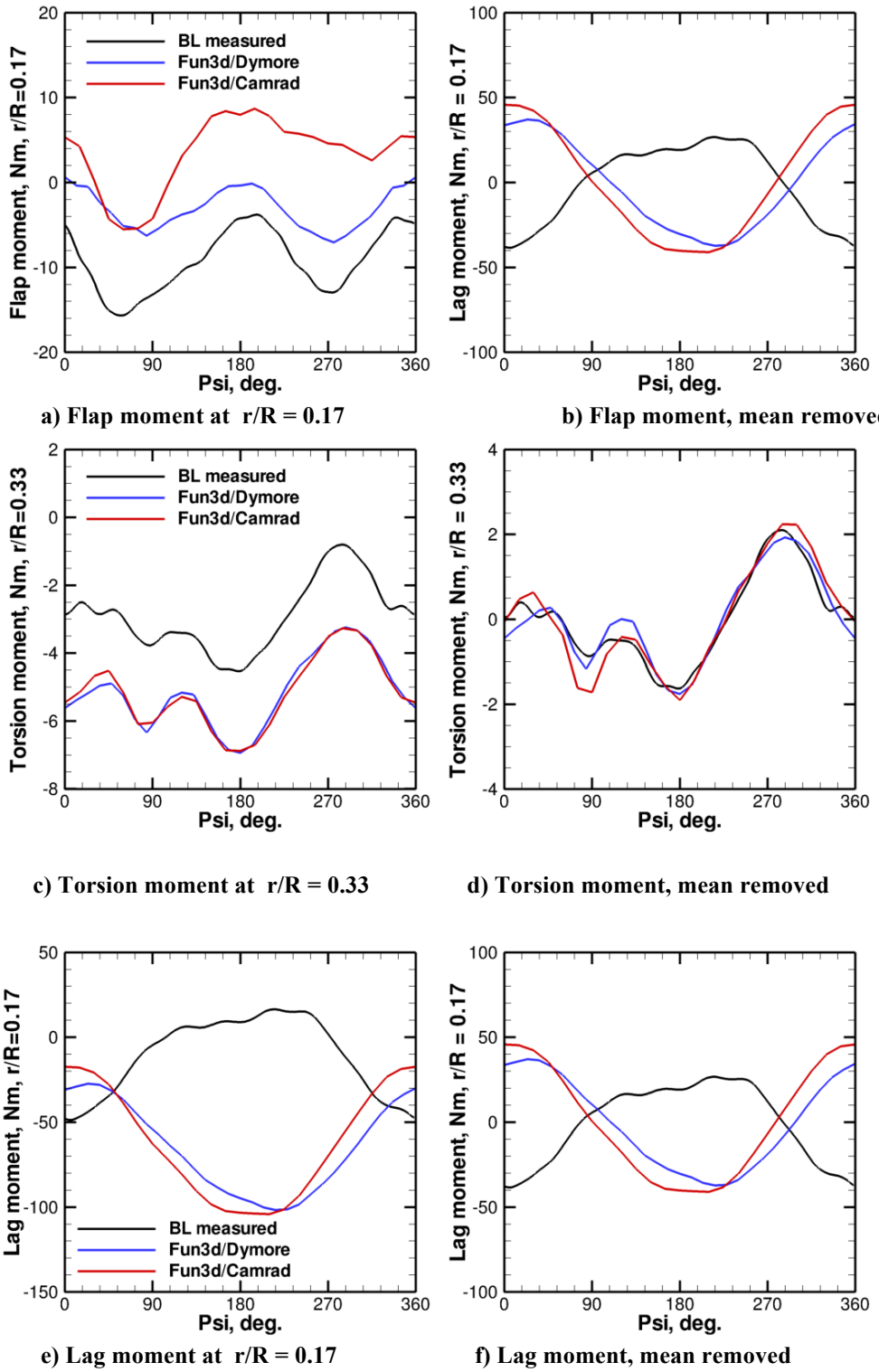


b) Pitching moment, C_mM^2



c) Pitching moment with mean removed, C_mM^2

Figure 35: Convergence of airloads at 87% span for the loosely-coupled trim phase. For measured pitching moment, the mean value is -0.00258; for iteration 5, the mean pitching moment is -0.00513.



a) Flap moment at r/R = 0.17

b) Flap moment, mean removed

c) Torsion moment at r/R = 0.33

d) Torsion moment, mean removed

e) Lag moment at r/R = 0.17

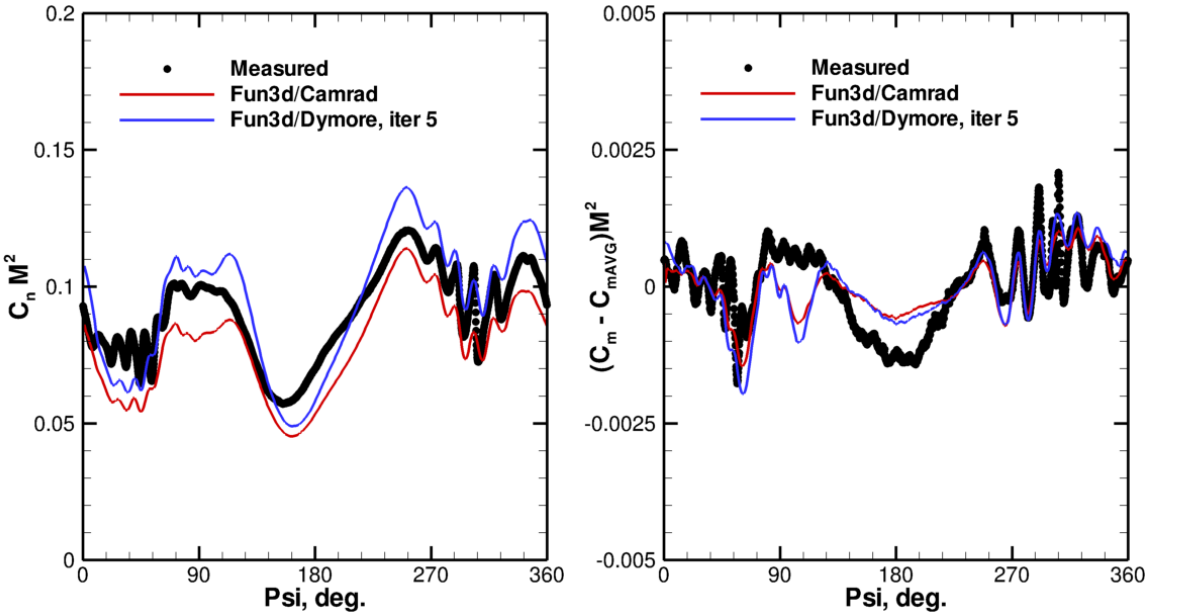
f) Lag moment, mean removed

Figure 36: Comparison of structural moments computed using FUN3D/Dymore and FUN3D/CAMRAD-II in Reference 67. Mean values removed in second column are listed in Table 10. Moment is negative for FUN3D results.

Table 10: Mean values subtracted from structural moments in second column of Figure 36.

	Flap, Nm r/R=0.17	Torsion, Nm r/R=0.33	Lag, Nm r/R=0.17
Measured	-9.27	-2.90	-10.26
FUN3D/Dymore	-3.05	-5.17	-64.49
FUN3D/CAMRAD-II	-3.36	-5.18	-63.18

A comparison of FUN3D/Dymore airloads from the final loose coupling iteration with those predicted by FUN3D/CAMRAD-II is provided in Figure 37. The two coupling methodologies provide similar results at all azimuths, with FUN3D/Dymore capturing the BVI regions more accurately, particularly the one in the fourth quadrant.

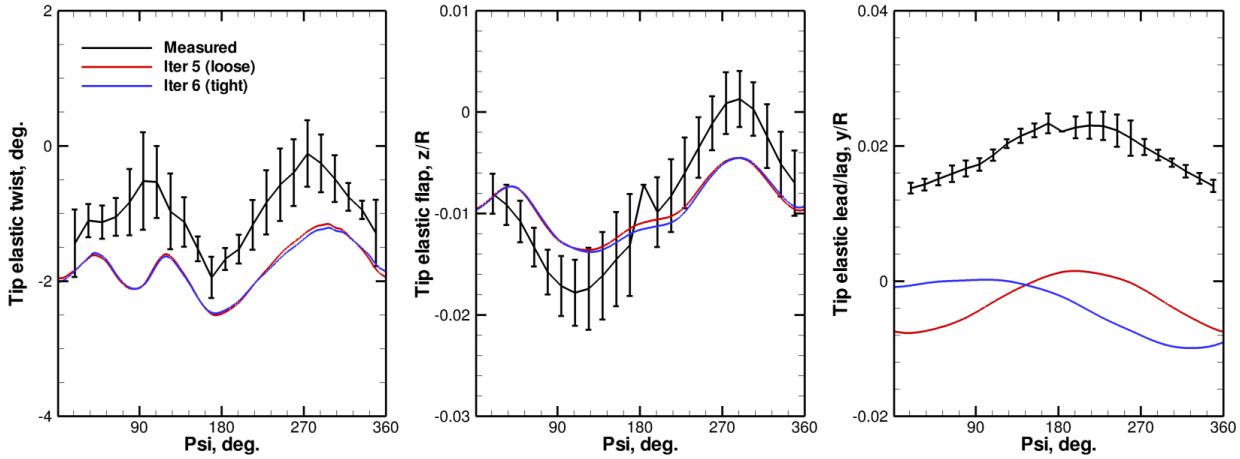


a) Normal force, C_nM^2 b) Pitching moment with mean removed, C_mM^2

Figure 37: Comparison of airloads at 87% span computed using FUN3D/Dymore and FUN3D/CAMRAD-II in Reference 67. Mean pitching moments are -0.00258, -0.00424, and -0.00513 for measured, FUN3D/CAMRAD-II, and FUN3D/Dymore, respectively.

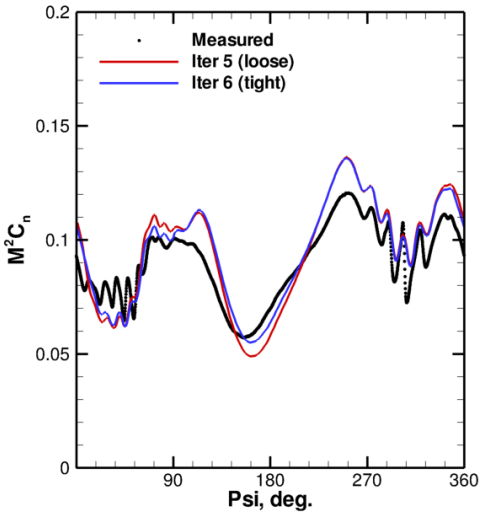
When a tightly-coupled solution is initialized from a previously converged loosely-coupled solution, the computed elastic deformations and airloads should not change.

Tip torsion, flap, and lag predicted after switching to tight coupling are shown in Figure 38, along with the motions from the final iteration of loose coupling. There is very little change in torsion and flap, although there is a small decrease in flap near 180° azimuth. Airloads at 87% span, shown in Figure 39, also show minimal change, particularly in pitching moment. Due to the small change in flapping motion at 180°, there is a small change in normal force. There are other small variations in normal force in the first quadrant BVI events. Since the tightly-coupled solution moves closer to the measured data, it can be surmised that the initial loosely-coupled solution was not completely converged. Otherwise, the solution is quite stable.

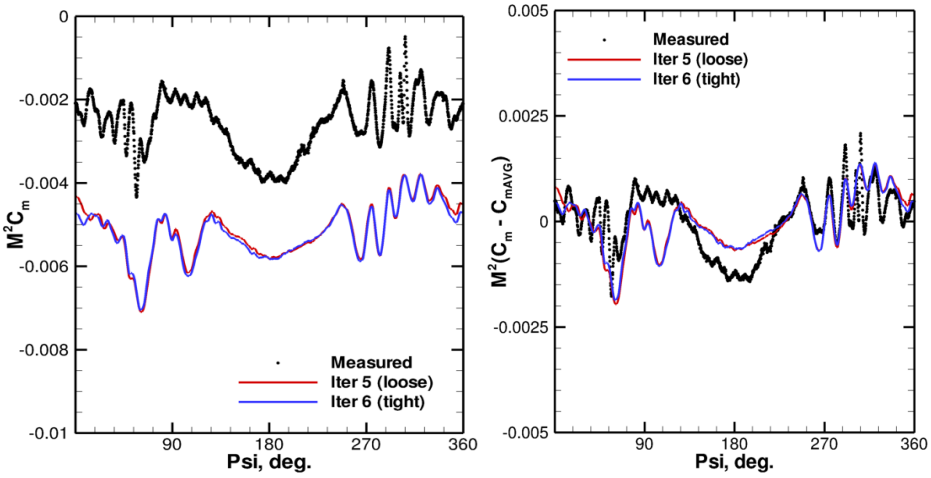


a) Elastic torsion b) Flap c) Lead-lag

Figure 38: Tip deflections predicted in tight coupling and the final iteration of loose coupling. Bars on the measured data indicate blade-to-blade variation.



a) Normal force, C_nM^2



b) Pitching moment, C_mM^2 c) Pitching moment with mean removed, C_mM^2

Figure 39: Sectional normal force and pitching moment 87% span, comparing loose and tight coupling.

The mean measured thrust is T=3300 N. The mean measured rolling and pitching moments are M_x=20 N-m (positive for rolling to the left) and M_y=-20 N-m (positive for nose-up pitching). At the end of the fifth loose coupling iteration, the computed mean hub loads were T=3326 N, M_x =11.9 N-m, and M_y =-29.2 N-m. After a revolution of tight coupling, the hub loads are nearly the same, with T=3328 N, M_x =10.2 N-m, and M_y =-10.7 N-m. Though the change in pitching moment as a fraction of the target value is large, instantaneous pitching moment varies from -200 N-m to 160 N-m, so the change in the mean is only 5.1% of the total variation.

Iso-surfaces of the second invariant of the velocity gradient tensor, also known as the "Q-criterion", are shown in Figure 40. Q-criterion has the property that it is only positive near vortices, and as such, it is a useful tool for distinguishing between sheets of high vorticity and actual vortices. The roll-up of the individual tip vortices into a single horseshoe vortex is clearly visible in these visualization, though the horseshoe dissipates just after of the rotor disc after exiting the refined region of the background grid. Just before 90^o and just after 270^o are regions where the blade tip can be observed to pass through several tip vortices in rapid succession, leading to the blade-vortex interactions discussed previously. In Figure 40(a), it is also clear that the system of tip vortices resides primarily within the plane of the rotor, as one would expect since the rotor is tilted aft to simulate a descending flight case.

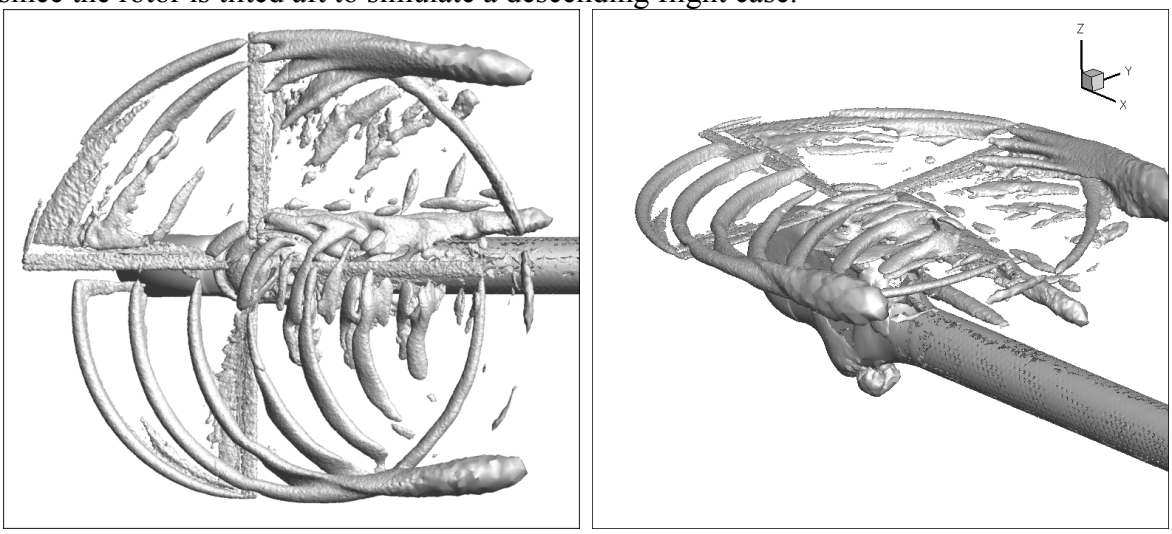


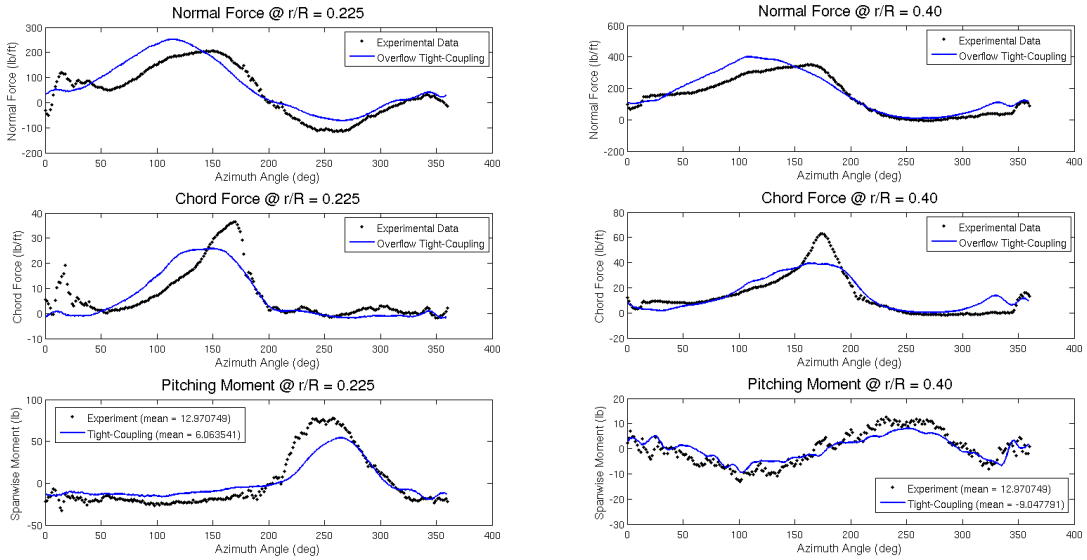
Figure 40: Q-Criterion iso-surfaces. Q=0.0075 and is non-dimensionalized by speed of

The flow solution was run on a Cray XT5 supercomputer. Each cluster node had two 2.3 GHz quad-core CPUs, for a total of eight cores per node. Four hundred cores were used for the flow solution. A single core was used for hole-cutting with the other seven cores on that node sitting idle to increase the memory available to the hole-cutting process. Each revolution required 7.3 hours of wall time. A total of 5.5 revolutions were simulated, including the final revolution of tight coupling, resulting in a total wall time of about 84 hours or 34,272 CPU hours. During tight coupling, Dymore had a negligible impact on CPU time and memory usage (less than 1% each). Dymore required about 45 minutes to trim each loose coupling iteration on a single processor. The Suggar hole-cutting and overset assembly process, which runs as a single process on the first MPI rank, takes 25% of the total wall clock time in both loose and tight coupling. A single overset assembly process is currently used because the mesh partitioning

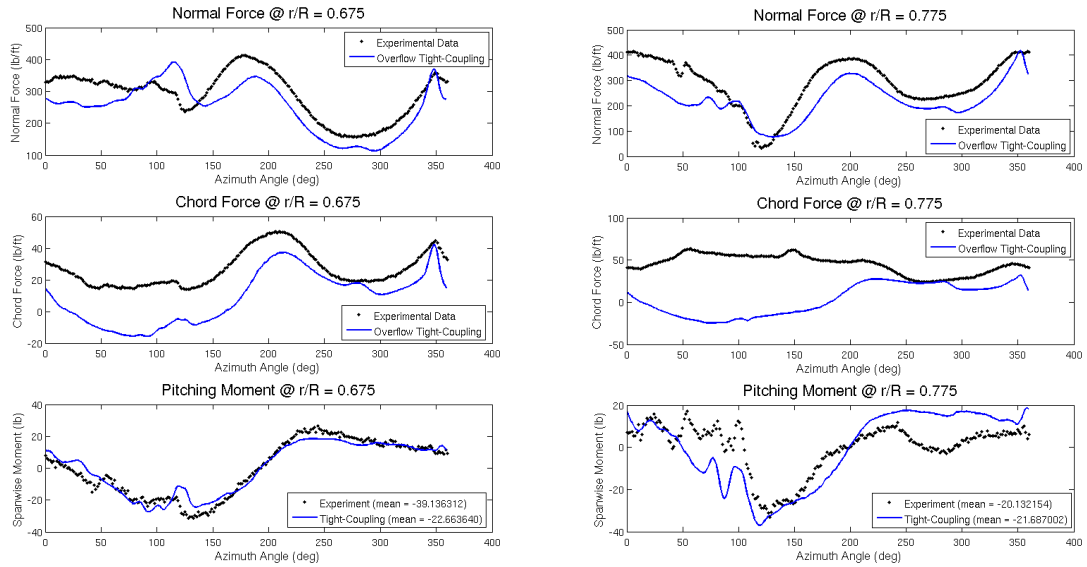
scheme in FUN3D, which yields compact partitions that minimize communication (and therefore maximize scalability in the flow solver), limits the parallel scalability of Suggar.

7.3. Maneuvering Flight Demonstration

The UH60-A 11029 test case was examined to demonstrate maneuvering flight. Only tight coupling was run for this case, and again the coarse mesh with the Spalart-Allmaras turbulence model was employed to demonstrate the capabilities of the methodology. Similar tightly-coupled behavior to the prior cases examined are observed with the 11029 test case, as shown in Figure 41 and Figure 42. Again the simulation was first initialized with a partial "0^{th}" CFD simulation similar to the start of a loosely-coupled analysis, and the tight coupling started after the transients due to the CFD initialization have past. Kriging is turned on after the transients due to the loose-to-tight coupling switch has been past, and the simulation is "flown" using inputs to the controller to simulate the maneuver.

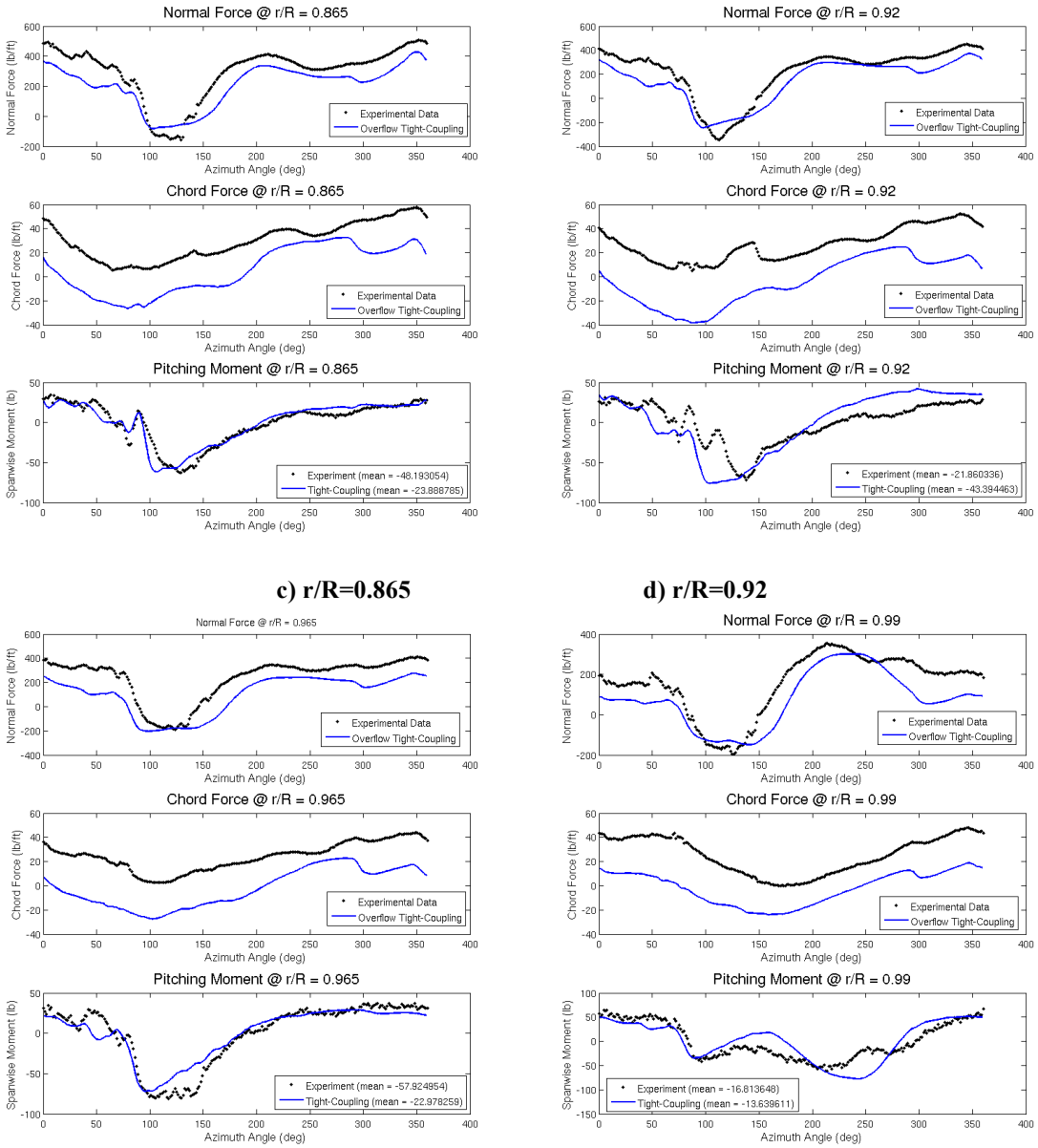


a) r/R=0.225 **b) r/R=0.40**



c) r/R=0.675 **d) r/R=0.775**

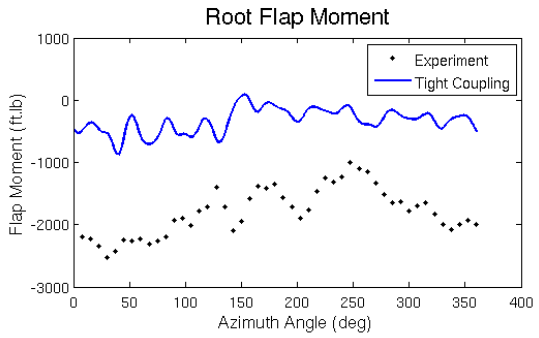
Figure 41: UH60-A C11029 tightly-coupled airloads using OVERFLOW-Dymore on the coarse grid.

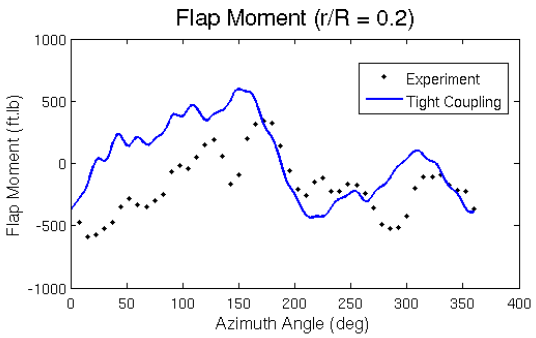


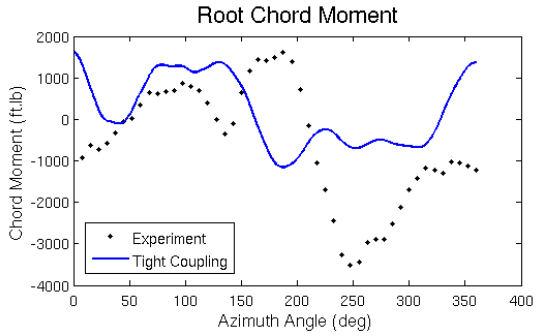
c) r/R=0.865 **d) r/R=0.92**

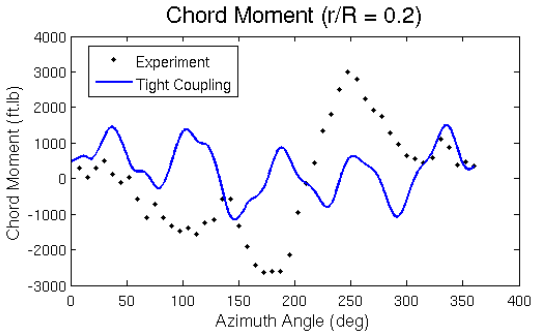
e) r/R=0.965 **f) r/R=0.99**

Figure 41 (cont.): UH60-A C11029 tightly-coupled airloads using OVERFLOW-Dymore on the coarse grid.

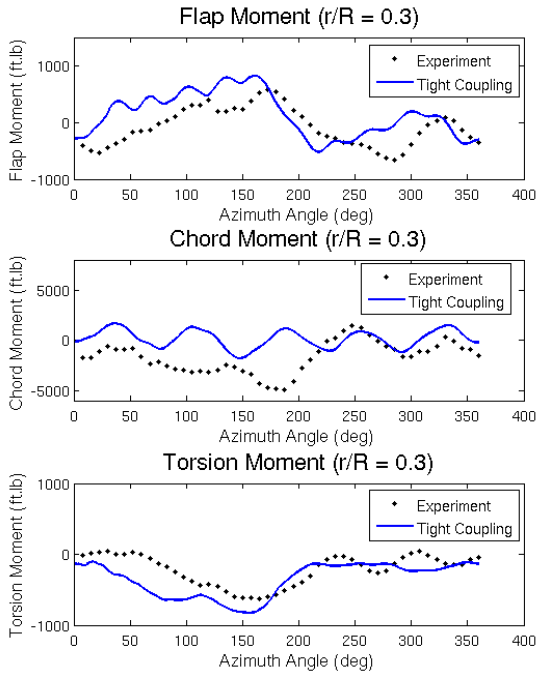


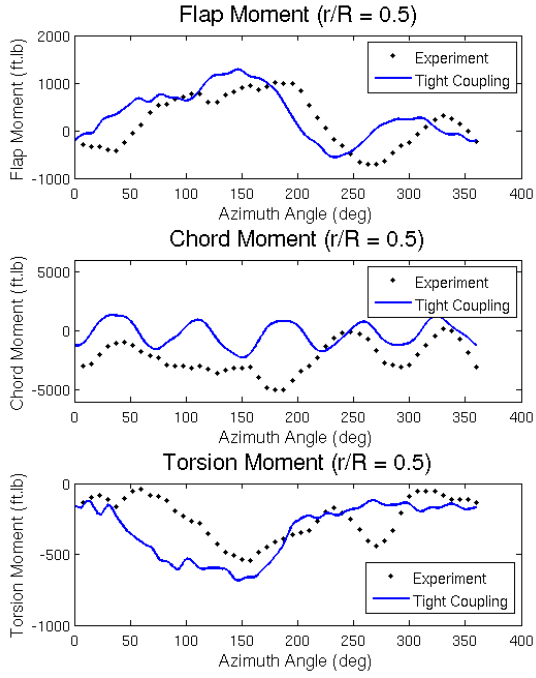






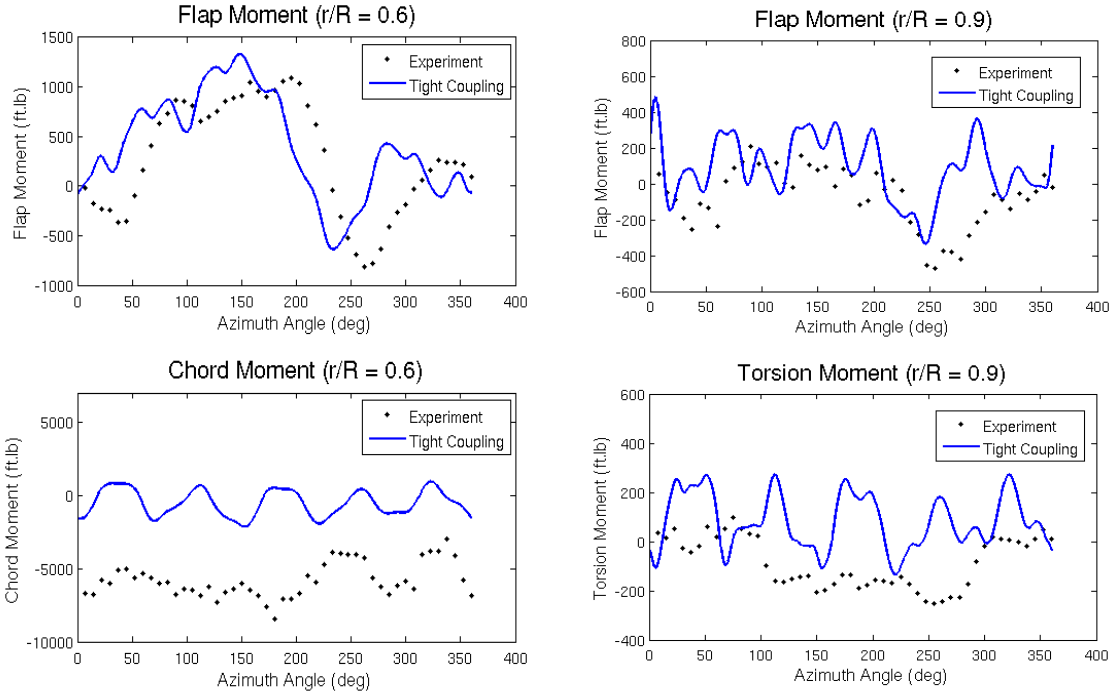
a) r/R=0.0 **b) r/R=0.20**





c) r/R=0.30 **d) r/R=0.50**

Figure 42: UH60-A C11029 tightly-coupled structural moments using OVERFLOW-Dymore on the coarse grid.



e) r/R=0.30 **f) r/R=0.50**

Figure 42 (cont.): UH60-A C11029 tightly-coupled structural moments using OVERFLOW-Dymore on the coarse grid.

7.4. Tight Coupled Application: Stability Analysis

Rotorcraft systems operate under flight conditions that contain nonlinearities such as transonic shocks, turbulent flow and interactions with their own wakes. The rotors themselves are nonlinear multi-body elastic systems. An aeroelastic analysis of these systems must include an assessment of potential instabilities and the determination of damping ratios for all modes. The tightly-coupled methodology developed in this effort was applied to this problem to determine its ability to be used for additional rotor aeroelastic problems of interest.

The stability assessment is undertaken using an algorithm based on a partial Floquet approach that has been successfully applied with CSD tools on rotors by Bauchau and Wang[69] and wind turbines by Bauchau and Skjolden[70]. The stability analysis approach is computationally inexpensive and consists of post processing aeroelastic data, which can be used with any aeroelastic rotorcraft code or with experimental data. Unlike classical stability analysis methodologies, it does not require the linearization of the equations of motion of the system, which makes it ideally suited for use in conjunction with the tightly-coupled CFD/CSD simulations.

If the governing equations of motion for a system can be formulated as linear, ordinary differential equations with constant coefficients, classical stability evaluation methodologies based on the characteristic exponents of the system can rapidly and accurately provide the system's stability characteristics. For systems described by linear, ordinary differential equations with periodic coefficients, Floquet's theory[71,72] is the preferred approach.

While these methods provide excellent results for simplified linear models with a moderate number of degrees of freedom, they become quickly unwieldy as the number of degrees of freedom increase[73]. Therefore, to accurately analyze rotorcraft aeroelastic periodic

systems, a formal linearization is virtually impossible. Instead, a fully nonlinear, coupled simulation tool is used to determine the response of the system to perturbations about an equilibrium configuration and determine the presence of instabilities and damping ratios.

Stability analysis of rotors and other complex multi-body systems

A multi-body system, such as a helicopter, can be modeled by a set of nonlinear equations of the form $\dot{\vec{y}} = \vec{g}(t,\vec{y})$, where $\vec{y}$ is a vector containing N state variables, t denotes time, and superscript dot represents a derivative with respect to time. The rotor in level forward flight can be assumed to initially operate at a constant mean rotor speed and a periodic steady state, $\vec{y}_{SS}$. Assuming small perturbations, $\vec{x} = \vec{y} - \vec{y}_{SS}$, about the steady state allows the nonlinear system to be approximated by a linear system with periodic coefficients:

$$\dot{\vec{x}} = \overline{\overline{A}}(t)\vec{x} + \vec{f}(t)$$

where $\overline{\overline{A}}(t) = \overline{\overline{A}}(t+T)$ is the periodic system matrix of period $T = 2\pi/\Omega$. This system and small perturbations about it can then be analyzed through several different computational methods, as next described.

The theoretical basis for the analysis of periodic systems, such as a rotor was developed by Floquet[74] who demonstrated that the response of a linear system with periodic coefficients can be modeled as the sum of modal contributions multiplied by an time-dependent exponential term, $\rho_k = exp(\lambda_k t)$. Floquet analysis determines the change in the states of the system at times t and $t+nT$, where nT is the time required for n integer-based periods of the system. λ_k are known as the characteristic exponents and is defined as a complex variable of the form $\lambda_k = \sigma_k \pm i\sigma_k$. The damping coefficient, σ_k, and principal frequency, $\omega_{p,k}$, of mode k are determined from the eigenvalues, ρ_k, as

$$\sigma_k = \frac{1}{T} ln(|\rho_k|)$$

$$\omega_{p,k} = \frac{1}{T} arctan(\rho_k), \quad \omega_{p,k} \in [-\Omega/2, \Omega/2]$$

To obtain more physically meaningful frequencies for engineering analysis, integer multiples (j_k) of the rotor speed can be added to the principal frequencies as $\omega_k = \omega_{p,k} + j_k\Omega$. The damping ratio can be computed from $\varsigma_k = -\sigma_k / \left(\omega_k \sqrt{1 + (\sigma_k/\omega_k)^2} \right)$. The system is stable if, for all values of k, $\zeta_k < 0$.

For large multi-body models, a formal linearization of the governing equations is difficult and costly to obtain both for time invariant systems and periodic systems. To overcome these difficulties, Bauchau and Wang[69,75,76] developed several approaches to stability analysis and demonstrated their applicability to large scale multi-body systems, verifying that these approaches provide identical results with a number of other, more complex methods, such as Prony's method or Poincare mapping.

A distinctive feature of these methods is that they can analyze one or multiple discrete time signals characterizing the dynamic response of the system for either time invariant or periodic systems, and unlike classical stability analysis methodologies, linearization of the system's equations of motion is not required. Consequently, these approaches are computationally inexpensive, consisting of post processing that can be used with any multi-

physics computational tool or with experimental data. Singular value decomposition is used systematically as a means of dealing with the noisy, highly redundant data sets obtained from nonlinear systems, and this approach is referred to as the partial Floquet analysis (PFA), and is described in Reference 77.

When performing any aeroelastic analysis, the accuracy of the prediction is driven by the weakest link of the analysis, which must include consideration of the structures, dynamics, and aerodynamics modeling. While the linear aerodynamics used routinely in aerodynamic analyses provides efficient turn-around, the aerodynamics of a rotor are highly nonlinear. The use of CFD-CSD coupling can provide an estimate of the nonlinearities of the aerodynamics, but the cost is much greater. Thus, to perform a stability analysis, it is critical to determine the least amount of data necessary to provide accurate results without incurring overly costly computations

Once the periodic trimmed solution has been obtained, a tightly-coupled simulation can be applied to model the perturbation and subsequent rotor response. The initial condition applied in the tight coupling is the solution obtained at the last iteration of a loosely-coupled simulation, which provides an equilibrium starting point to the tightly-coupled stability analysis.

For this evaluation, tight coupling was achieved by compiling the Dymore code into a static library with a CFD interface that could then be linked into the OVERFLOW CFD code. This way, the CFD code controls the coupling process from inside itself. It is also possible to achieve tight coupling by linking the codes through Python or another scripting language. This tightly-coupled time stepping continues until the simulation is complete, at which point the CFD and CSD enter into their respective post-processing routines. Since the stability analysis was performed using the PFA method described earlier, it is only necessary to save the temporal data during the simulation at locations using sensors defined within the CSD code. No interactive stability analysis is required.

The demonstration computations were obtained on the UH60-A coarse grid described previously, using the compressible 4th order time-accurate URANS equations with a 0.05° azimuthal time integration. The Spalart-Allmaras turbulence model is utilized to resolve the boundary layer as it has been shown (using OVERFLOW) to be comparable to more advanced turbulence models for the C8534 flight condition[78].

The stability analysis results were obtained using the following procedure. First, the loose-coupling strategy with the delta-airloads approach was used to obtain a converged, periodic solution of the UH60-A at the C8534 flight condition. Next, using this solution as an initial condition, the tight-coupling procedure was applied to predict the aeroelastic response of the system. A perturbation was applied to the system, which consisted of three time-dependent concentrated loads applied at the blade's three-quarter span (75%R) location: a load applied upward in the direction perpendicular to the rotor plane, a load acting in the plane of the rotor directed towards the trailing edge, and a torque. These three loads were applied as over time, linearly increasing to the maximum and then decreasing to zero. This loading creates triangular pulses with maximum amplitudes of 500 lbs, 100 lbs, and 2,200 ft-lbs, respectively in the normal, in-plane and torsional directions. For each case, the triangular pulse lasted for 0.1 revolution.

The applied perturbation changed the blade aeroelastic response, resulting also in modified airloads. After the end of the perturbation, the released rotor continued to respond to the perturbation and subsequent aeroelastic effects. Four signals were identified for the stability analysis procedure: the flap and lag displacements and lag rotation angle at the blade three-

quarter span location, and the root flap angle. All four signals were sampled at 180 or 290 points per revolution.

Finally, the proposed stability analysis method was applied to these four signals. Both partial Floquet and Prony methods were applied to the resulting CFD-CSD response, and very similar results are obtained from both approaches.

Figure 43 shows the singular values obtained from the SVD of the Hankel matrix H_0. The largest singular value is normalized to unity, and the subsequent singular values are arranged in descending order. Note the logarithmic scale on the ordinate. The magnitude of the 16th singular value is less than one percent of the first singular value. Since the singular values are an indicator of the energy associated with each mode, Figure 43 implies that over 99% of the total energy in the signal is concentrated in those first 16 modes. Each pair of singular values allows the determination of the frequency and damping of one mode of the system.

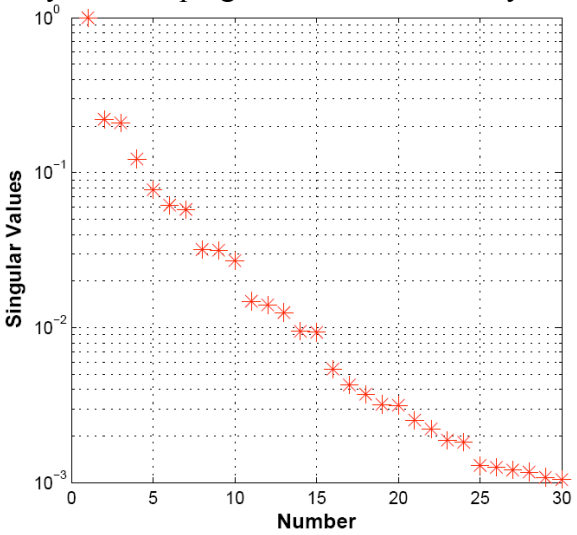


Figure 43: Singular values arranged in decreasing order for the 3.5 revolution analysis.

At first, a window starting 0.01 seconds after the end of the perturbation and lasting for 3.5 revolutions was investigated. Five different approaches (Table 11) were applied to evaluate the system modal parameters. Table 12 lists the damping ratios for the 1st lead-lag, 1st flap, 2nd flap, and 1st torsion modes. The first two approaches use the partial Floquet algorithm with rank numbers of 32 and 16, respectively, and sampling rates of 32 and 16 points per revolution, respectively. The last three approaches use Prony's method with rank numbers of 32, 16, and 18, respectively and sampling rates of 180, 180 and 290 points per revolution, respectively.

Table 11: Characteristics of the stability approaches.

Notation	Algorithm	Rank Number	Sampling Rate [/rev]
PF	Floquet	32	180
PF6	Floquet	16	290
PR	Prony	32	180
PR2	Prony	16	180
PR5	Prony	18	290

Table 12: Damping ratios for the four rotor modes.

Case	Lead-Lag	1st Flap	2nd Flap	Torsion

PF	-31.85	-21.24	0.13	-0.66
PF6	-47.11	-30.84	-0.87	-0.25
PR	-31.85	-21.24	0.13	-0.66
PR2	-48.37	-30.90	-0.86	-0.25
PR5	-32.44	-17.49	-0.23	-0.39

To ensure the validity of the prediction, the various signals were reconstructed based on the identified modal parameters. Figure 44 illustrates the original and reconstructed signals for the four signals selected for the analysis. Excellent correlations are observed between the original and reconstructed signals, indicating the robustness of the proposed approach.

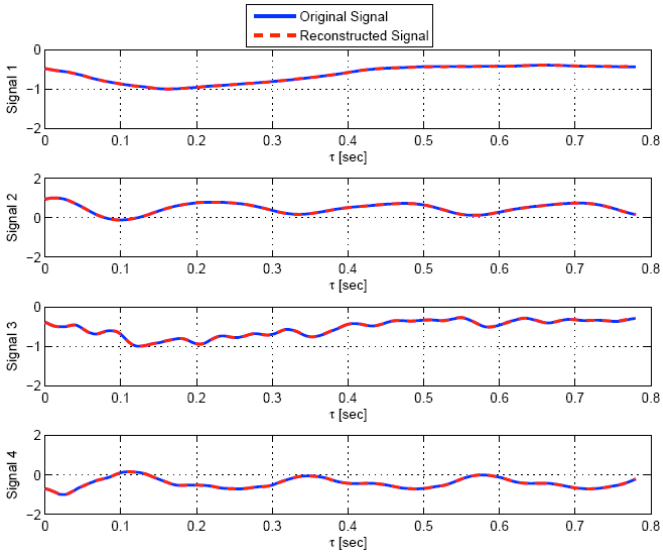


Figure 44. Comparison of the original and reconstructed signals for the four signal locations.

The results presented in this section focus on four rotor modes: the first lead-lag, first flap, second flap and first torsion modes. The predicted frequencies are shown in Figure 45 and are in good agreement with the frequencies predicted with lower-order aerodynamic models, as expected.

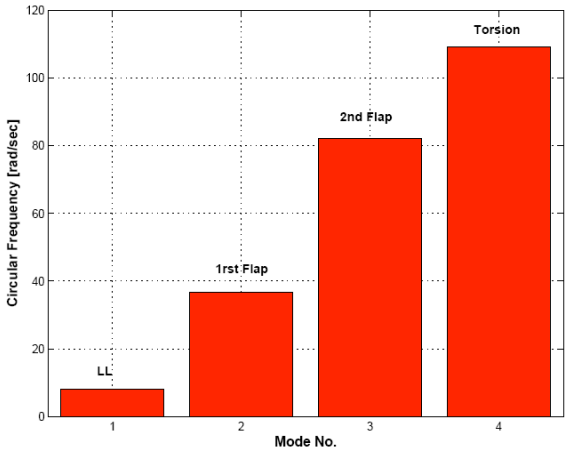


Figure 45: Predicted frequencies for the four rotor modes.

Figure 46 shows the corresponding damping ratio. Table 12 lists the damping ratios for the five approaches investigated here. Predictions are consistent for the four rotor modes. As expected, the first flap mode is well damped, in contrast with the second flap and torsion modes that are very lightly damped. Note that the 3.5 revolution data used here is not long enough to obtain reliable predictions of these lightly damped modes.

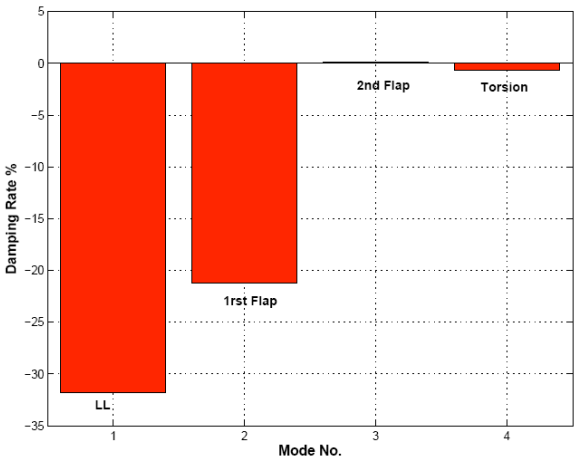


Figure 46: Predicted damping ratios for the four rotor modes.

The results presented in Figure 46 show that the identified damping of the first lag mode is very high, about 35%. This result is unexpected, since the first lag mode of the UH60-A rotorcraft is known to be very lightly damped. Upon investigation, it was first observed that CFD analysis predicted a doubling of the overall lift over the perturbation time period, compared to a 40% increase for lower-fidelity aerodynamics. The elastic deflection from that loading indicated a significant, but physically feasible response in the blade attitude, as shown in Figure 47. The UH60-A case selected for demonstration, C8534, is a high-speed flight case, and the tip may enter the transonic regime. Increases in twist angle will cause the blade outer span to enter further into the transonic regime, giving rise to shocks, which can dramatically influence the lead-lag damping. The Mach contours on the upper surface of the rotor at the 90° azimuth for the advancing side of the rotor in Figure 48 clearly illustrate this aerodynamic phenomenon. The perturbation immerses the rotor outer span in a higher transonic flow compared to the pre-perturbed state, and this state remains for a significant portion of the window used for the stability analysis. The result of the transonic flow comes via the presence of stronger shocks, which dramatically increase the drag. Thus, the damping in the lead-lag mode is disproportionately high compared to the damping encountered during most of the level flight regime.

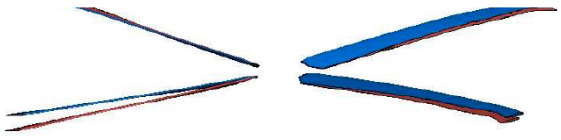
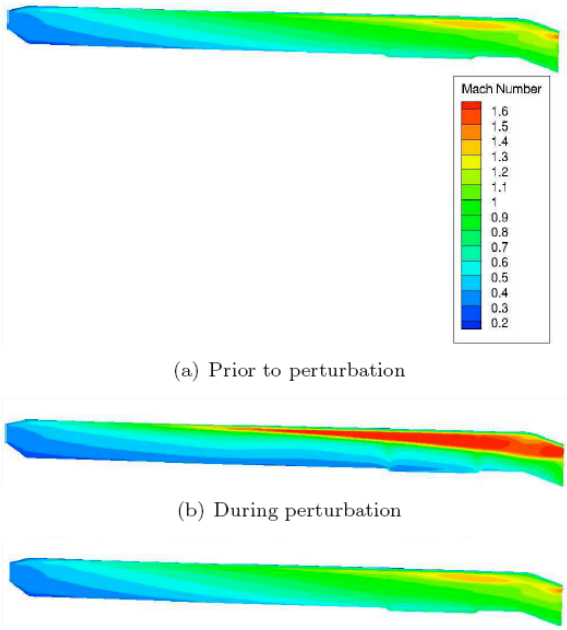


Figure 47: Blade response at maximum perturbation location.



(a) Prior to perturbation

(b) During perturbation

(c) Three revolutions after perturbation

Figure 48: UH60-A CFD Mach number variation during blade perturbation

The tightly-coupled computational fluid dynamics and computational structural dynamics (CFD-CSD) technique, employing an algorithm based on a partial Floquet approach provides the tools to accomplish successful stability analysis of a rotor system. The UH60-A rotor in high-speed level forward flight has been used to demonstrate the validity of this method, as well as its ability to predict modal characteristics within a moderate number of rotor revolutions. A number of observations can be made:

- The predicted frequencies of the rotor modes correlate very well with the known rotor frequencies of the UH60-A.
- Signal reconstruction indicates that the partial Floquet analysis reliably extracts the modal parameters from the computed signals.
- As few as 3.5 revolutions after the system is perturbed yields good prediction of the rotor frequencies, but it is not long enough to reliably predict the damping characteristics of the lightly damped modes.
- The number of CFD-CSD revolutions to determine the stability characteristics after the perturbation is released appears to be dependent on the magnitude of the perturbation.
- A large perturbation in high-speed forward flight may generate large damping ratios in lead-lag due to the presence of strong shocks.
- The proposed approach may be able to qualitatively assess the dependency of damping on amplitude. Further studies are proposed beyond this effort to demonstrate this important feature of the method.

8. Parallelization and Portability of Codes

Concurrent to the efforts described in the preceding technical results and development sections, whenever code was developed, it was parallelized (as required) and tested for compatibility on various compilers and computers. Scripts, including makefiles and known conflicts/resolutions that allow portability with minimal user input have been provided to NASA and other code users during the course of the project.

The OVERFLOW and Dymore tightly-coupled code have been successfully ported to the following machines and operating systems:

- Georgia Tech local Beowulf PC cluster; Fedora Linux (snowshoe)
- Georgia Tech local PC; ubuntu
- NAS SGI ICE; SUSE linux (Pleiades)
- NAS SGI Altex ; SUSE Linux (columbia)
- DoD HPC Cray XE6; CLE 3.1 UP02 (raptor)
- DoD HPC Cray XT5; CLE 2.2.UP02 (einstein)

The FUN3D and Dymore tightly-coupled framework has been similarly ported to these machines and operating systems:

- DoD HPC Cray XE6; CLE 3.1 UP02 (raptor)
- DoD HPC Cray XT5; CLE 2.2.UP02 (einstein)
- DoD HPC, Cray XE6; CLE (garnet)
- NAS SGI ICE; SUSE linux (Pleiades)

The tight coupling protocol includes two methods of coupling wherein one method has both codes compiled and the CSD method is called as a library, wherein the second method using Python to accomplish the coupling. Both methods rely on arrays and shared memory to exchange information, rather than file transfers.

The Python versions of the coupled codes do have limitations based on the parallel system type. Specifically, in creating shared libraries, this particular problem is encountered on Cray XT and SGI Altex machines when compiling in MPI:

```
cc -fPIC -c hello.c
/opt/cray/xt-asyncpe/2.0/bin/cc: INFO: linux target is being used
cc -fPIC -c goodbye.c
/opt/cray/xt-asyncpe/2.0/bin/cc: INFO: linux target is being used
cc -shared -o libsay.so hello.o goodbye.o
/opt/cray/xt-asyncpe/2.0/bin/cc: INFO: linux target is being used
/usr/bin/ld: /usr/lib64/libpthread.a(pt-system.o): relocation
R_X86_64_32 against `a local symbol' can not be used when making a
shared object; recompile with -fPIC
/usr/lib64/libpthread.a: could not read symbols: Bad value
```

This situation exists because on these computers, any binary created by ftn or cc will be statically linked, and it is not possible to have a dynamically linked library (dll). This limitation was discussed with Army Helios developers, and DoD User Productivity Enhancement and Technology Transfer (PET) help desk during 2009-2010. As a result of Georgia Tech's early detection of this problem, an independent effort to provide a work-around for the static-dynamic

library issue was funded by the Army[79], as this situation impacts many Python-based CFD-CSD coupling efforts, including for example Helios. The current Python coupling methods have been successfully tested on the local Georgia Tech personal computers and on the NAS SGI ICE running SUSE linux (Pleiades).

9. Documentation and Technology Transfer

The following papers have been presented/published or abstracts accepted

- Riviello, L., Bottasso, C., and Bauchau, O. A. "Effect of Modeling Approximations on the Stability of Autopilot Controllers," Proceedings of the 64th Annual American Helicopter Society Forum, Montreal, Canada, April 29 - May 1, 2008.

- **Zaki, A., Reveles, N., Bauchau, O., and Smith, M., "Using Tightly-Coupled CFD/CSD Simulation for Rotorcraft Stability Analysis," Proceedings of the 66th Annual American Helicopter Society Forum, Phoenix, AZ, May 2010.

- Reveles, N., Zaki, A., Smith, M., and Bauchau, O., "An Evaluation of Loose and Tight CFD-CSD Coupling for UH60-A Airloads," UH60-A Airloads Workshop, Ames Research Center, CA, Feb 2011.

- **Reveles, N., Zaki, A., Smith, M., and Bauchau, O., "A Neural-Network-Based Trim Algorithm for Computational Rotor Aeroelasticity," to be presented at the European Rotorcraft Forum, Milan, Italy, September 2011.

- Reveles, N., and Smith, M. J., "Analysis of a Rotorcraft in Descent," (working title), under preparation for the *AIAA Journal of Aircraft*, expected in September, 2011.

** Indicates that this paper is also being/has been submitted to an archival journal.

Contributions to PhD theses at Georgia Tech

- Lynch, C. E., *Advanced CFD Methods for Wind Turbine Analysis*, PhD dissertation, Georgia Institute of Technology, January 2011. (HART-II tight coupling)

- Zaki, A., *Using Tightly-Coupled CFD/CSD Simulation for Rotorcraft Stability Analysis*, PhD dissertation, Georgia Institute of Technology, planned defense Summer 2011. (Kriging implementation and application to stability)

- Reveles, N., *Tightly-Coupled Rotor CFD-CSD for Steady and Transient Aeroelastic Simulations* (working title), PhD dissertation, Georgia Institute of Technology, planned defense in 2012. (Tight coupling, including Python-based framework, UH60-A simulations)

Technology Transfer

- The modified codes have been provided to NASA development teams for both FUN3D and OVERFLOW.

- Engineers at both Bell Textron Helicopter and Sikorsky Helicopter have modified versions of OVERFLOW with Dymore tight coupling and are applying them to rotor configurations.

Patents

- No patents have been filed as a result of this effort.

10. Conclusions

A tightly-coupled computational fluid dynamics and computational structural dynamics (CFD-CSD) technique, employing an algorithm based on a partial Floquet approach provides the tools to accomplish aeroelastic analysis of a time-accurate rotor system. The approach has been demonstrated for both unstructured and structured CFD methods, using both compilation- and Python-based frameworks to replace file data transfers between the CFD and CSD methods. The UH60-A and HART-II rotor configurations have been used to demonstrate the validity of this method. Eight metrics of success were defined at the beginning of the research effort, and all have been met or exceeded.

A number of technical observations can be concluded with respect to this research:

- Both neural networks, as well as an alternative, more efficient method, kriging, were examined. Based on the results, it was determined that kriging provided a viable alternative to neural networks in the development of the computational trimmer.
- Moderate numbers of CFD/CSD data have been used to train the metamodel of the new trim algorithm. CSD data can also be used to train the metamodel of the kriging algorithm for some run configurations.
- Initialization of the tight coupling with loose coupling results improves the convergence of the tight coupling algorithm for level flight cases. Optimization indicates that ¼ - ½ revolution of the initial (zeroth) loose coupling iteration is sufficient to remove transients that affect the tightly-coupled solution stability.
- Updating the controls via the kriging algorithm immediately upon commencing the tight coupling also creates instabilities, so a period of fixed controls is necessary. However, the result of this fixed control period, if sufficiently periodic, can be added to the training database for the current and future kriging estimates.
- Tight coupling yields comparable results to loose coupling when dynamic stall in conjunction with advanced turbulence models are not present. The differences that are yielded when these are used may provide clues to the physics of the rotor behavior.
- Tight coupling, when optimized and used with the kriging controller, appears, within the context of the test cases examined in this effort, to converge in approximately the same number of total revolutions as loose coupling.
- The tight coupling has been demonstrated to determine the stability characteristics of rotors.
- Signal reconstruction indicates that the partial Floquet analysis reliably extracts the modal parameters from the computed signals with as few as 3.5 revolutions after the system is perturbed yields good prediction of the rotor frequencies, but may not be sufficient to reliably predict the damping characteristics of the lightly damped modes.

11. References

1. Strawn, R. C., Caradonna, F. X. and Duque, E. P. N., "30 Years of Rotorcraft Computational Fluid Dynamics Research and Development", *Journal of the American Helicopter Society*, Vol. 51, No. 1, 2006, pp. 5–21.
2. Duque, E. P. N., Sankar, L. N., Menon, S., Ruffin, S., Smith, M., Ahuja, A., Brentner, K. S., Long, L. N., Morris, P. J., and Gandhi, F., "Revolutionary Physics-Based Design Tools for Quiet Helicopters," AIAA-2006-1068, Aerospace Sciences Exhibit and Conference, Reno, NV, January 2006.
3. Brown, R.E., and Line, A.J., "Efficient High-Resolution Wake Modeling Using the Vorticity Transport Equations," *AIAA Journal*, Vol. 43, No. 7, April 2005, pp. 1434-1443.
4. Inthra, P. A., "Development of Rotorcraft Wake Capturing Methodology Using Fully Coupled CFD and Particle Vortex Transport Methods," Proceedings of the 62nd AHS Annual Forum, Phoenix, AZ, May 2006.
5. Chan, W., Meakin, R., and Potsdam, M., "CHSSI Software for Geometrically Complex Unsteady Aerodynamic Applications," AIAA Paper 2001-0593, AIAA 39th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 2001.
6. Kunz, D. L., "Comprehensive Rotorcraft Analysis: Past, Present and Future," 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, Austin, TX, April, 2005.
7. Saberi, H., Khoshlahjeh, M., Ormiston, R. A., and Rutkowski, M. J., "Overview of RCAS and Application to Advanced Rotorcraft Problems,"American Helicopter Society 4th Decennial Specialists' Conference on Aeromechanics, San Francisco, CA, January 2004.
8. Bir, G. and Chopra, I., "Development of UMARC (University of Maryland Advanced Rotorcraft Code," Proceedings of the 46th American Helicopter Society Annual Forum, Washington D.C., May 1990.
9. Johnson, W., "Rotorcraft Dynamics Models for a Comprehensive Analysis", Proceedings of the 54th American Helicopter Society Annual Forum, Washington D.C., May 1998.
10. Sopher, R. and Hallock, D., "Time-History Analysis for Rotorcraft Dynamics Based on a Component Approach," 2nd Decennial Specialists' Meeting on Rotorcraft Dynamics, Moffett Field, CA, November 1984.
11. Benoit, B. , Dequin, A. M., Kampa, K., Grunhagen, W., Basset, P.-M., and Gimonet, B., "HOST: A General Helicopter Simulation Tool For Germany and France", Proceedings of the 56th American Helicopter Society Annual Forum,Virginia Beach, VA, May 2000.
12. Bauchau, O. A., Bottasso, C. L. and Nikishkov, Y. G., "Modeling Rotorcraft Dynamics with Finite Element Multi-body Procedures," *Mathematical and Computer Modeling*, Vol. 33, pp. 1113-1137, 2001.
13. Berdichevsky, V. L., "On the Energy of an Elastic Rod," *Journal of Applied Mathematics and Mechanics*, Vol. 45, pg. 518-529, 1982.
14. Cesnik, C. E. S. and Hodges, D. H., "VABS: A New Concept for Composite Rotor Blade Cross-Sectional Modeling," Journal of the American Helicopter Society, Vol. 42, No. 1, pp. 27-38, 1997.

15. Bauchau, O. A., Choi, J. Y., and Bottasso, C. L., “On the Modeling of Shells in Multi-body Dynamics,” *Multi-body System Dynamics*, Vol. 8, No. 4, pp. 459-489, 2002.
16. Nikravesh, P. E.: *Computer-Aided Analysis of Mechanical Systems*, Prentice-Hall, Englewood Cliffs, New Jersey, 1988
17. Bauchau, O. A., and Bottasso, C.L. and Nikishkov, Y.G.: “Modeling Rotorcraft Dynamics with Finite Element Multi-body Procedures.” *Mathematical and Computer Modeling*, Vol. 33, No. 10-11, pp. 1113-1137, 2001
18. Bauchau, O. A., and Hodges, D.H.: “Analysis of Nonlinear Multi-Body Systems with Elastic Couplings.” *Multi-body System Dynamics*, Vol. 3, pp. 168-188, 1999
19. Bridgeman, J. O., Strawn, R. C., Caradonna, F. X., and Chen, C. S., “Advanced Rotor Computations with a Corrected Potential Method,” Proceedings of the 45th American Helicopter Society Annual Forum, Boston, MA, May 1989.
20. Smith, M. J., *A Fourth-Order Euler/Navier-Stokes Prediction Method for the Aerodynamics and Aerolasticity of Hovering Rotor Blades*, PhD Dissertation, Georgia Institute of Technology, 1994.
21. Bauchau, O. A., and Ahmad, J. U., "Advanced CFD and CSD Methods for Multidisciplinary Applications in Rotorcraft Problems," AIAA-1996-4151, 6th NASA and ISSMO Symposium on Multidisciplinary Analysis and Optimization, Bellevue, WA, September 4-6, 1996.
22. Datta, A., Nixon, M., and Chopra, I., "Review of Rotor Loads Prediction with the Emergence of Rotorcraft CFD." Presented at the 31st European Rotorcraft Forum, Florence, Italy, September 13-15, 2005.
23. Potsdam, M., Yeo, H., and Johnson, W., "Rotor Airloads Prediction Using Loose Aerodynamic Structural Coupling." Proceedings of the American Helicopter Society 60th Annual Forum, Baltimore, Maryland, June 7-10, 2004.
24. Makinen, S. et al., "OVERFLOW-RCAS CFD-CSD Coupling." Proceedings of the American Helicopter Society 62th Annual Forum, Phoenix, Arizona, May 9-11, 2006.
25. GIT/PSU Team, “Revolutionary Physics-Based Design Tools for Quiet Helicopters,” Final Report, DARPA Helicopter Quieting Program, December 2006.
26. O'Brien, D. M., and Smith, M. J., "Understanding The Physical Implications Of Approximate Rotor Methods Using An Unstructured CFD Method," Proceedings of the 31st European Rotorcraft Forum, Florence, Italy, September 2005.
27. O'Brien, D. M., *Analysis of Computational Modeling Techniques for Complete Rotorcraft Configurations*, Georgia Institute of Technology, Advisor: Prof. M. J. Smith, May 2006.
28. Rutkowski et al., presentations from the UH60-A Airloads Workshop, February 2007, unpublished. See, e.g., Potsdam’s presentation for an overview of convergency problems with UH60-A case 9017.
29. Nygaard, T. A., Saberi, H., Ormiston, R. A., Strawn, R. C., and Potsdam, M., “CFD and CSD Coupling Algorithms and Fluid Structure Interface for Rotorcraft Aeromechanics in Steady and Transient Flight Conditions,” Proceedings of the 62nd American Helicopter Society Annual Forum, Phoenix, AZ, May 2006.
30. Rajmohan, N., Phanse, S., and Sankar, L. N., “UH60-AA Airloads Prediction using First-Principles Based Aerodynamic Modeling,” presented at the UH60-A Airloads Meeting, August, 2006, Ames Research Center, unpublished.

31. Peters, D. A., and Barwey, D., "A General Theory of Rotorcraft Trim." Proceedings of the 36th Structures, Structural Dynamics, and Materials Conference, New Orleans, LA, April 1995.
32. Bottasso, C. L., and Riviello, L., "Rotor Trim by a Neural Model-Predictive Auto-Pilot," Proceedings of the American Helicopter Society 62nd Annual Forum, Phoenix, AZ, May 2006.
33. Bottasso, C. L., Chang, C.-S., Croce, A., Leonello, D., and Riviello, L., "Adaptive Planning and Tracking of Trajectories for the Simulation of Maneuvers with Multi-body Models," *Computer Methods in Applied Mechanics and Engineering*, Vol. 195, No. 50-51, pp. 7052-7072, October 2006.
34. Bottasso, C.L., Croce, A., Leonello, D., and Riviello, L., "Procedures for Enabling the Simulation of Maneuvers with Comprehensive Codes," Proceedings of the 31st European Rotorcraft Forum, Florence, Italy, September 2005.
35. Kim, C. H., and Arora, J.S., "Nonlinear Dynamic System Identification for Automotive Crash using Optimization: A Review," *Structural and Multidisciplinary Optimization*, Vol. 25, No. 1, pp. 2-18, 2003.
36. Viberg, M., "Subspace-based methods for the Identification of Linear Time-Invariant Systems," *Automatica*, Vol. 31, No. 12, pp. 1835-1851, 1995.
37. Ho, B., and Kalman, R., "Efficient Construction of Linear State Variable Models from Input/output Functions." *Regelungstechnik*, Vol. 14, pp. 545-548, 1966.
38. Moore, B. C., "Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction." *IEEE Transaction on Automatic Control*, Vol. AC-26, No. 1, pp. 17-32, 1981.
39. Juang, J. N., and Pappa R. S., "An Eigensystem Realization Algorithm for Modal Parameter Identification and Model Reduction." *Journal of Guidance, Control, and Dynamics*, Vol. 8, No. 5, pp. 620-627, 1985.
40. Lall, S., Marsden, J. E., and Glavaški, S., "A Subspace Approach to Balanced Truncation for Model Reduction of Nonlinear Control Systems." *International Journal of Robust and Nonlinear Control*, Vol. 12, No. 5, pp. 519-535, 2002.
41. Peeters, B., and de Roeck, G., "Stochastic System Identification for Operational Modal Analysis: A Review." *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 123, pp. 659-667, 2001.
42. Overschee, P., and de Moor, B., *Subspace Identification for Linear Systems: Theory-Implementation-Applications*, Springer, New York, New York, 1996.
43. G. Gubenko "Approximations by Syperpositions of Sigmoidal Functions" *Mathematics of Control, Signals, and Systems*, Vol. 2, pp. 303-314, 1989.
44. Calise, A. J., Hovakimyan, N., and Idan, M., "Adaptive Output Feedback Control of Nonlinear Systems using Neural Networks," *Automatica*, Vol. 37, No. 8, pp. 1201-1211, 2001.
45. Kutay, A. T., Calise, A. J., Idan, M., and Hovakimyan, N., "Experimental results on adaptive output feedback control using a laboratory model helicopter**,"** *Control Systems Technology, IEEE Transactions*, Vol. 13, No. 2, pp. 196-202, 2005.

46. Meijer, P.B.L., "Neural Networks for Device and Circuit Modeling," *Scientific Computing in Electrical Engineering*, Proc. SCEE-2000, August 20-23, 2000, Warnemünde, Germany, U. van Rienen, M. Günther and D. Hecht, Eds., Springer-Verlag, 2001, pp. 251 – 258.
47. S. M. Amin, Gerhart, V., and Rodin, E. Y., "System Identification via Artificial Neural Networks: Application to On-line Aircraft Parameter Estimation," AIAA 97-5612.
48. Van Rossum, G., *Python/C API Reference Manual, Release 2.5, F. L. Drake, Jr., ed., Python Software Foundation, September, 2006. (See also: http://www.python.org)*
49. Peters, D.A., Chouchane, M., and Fulton, M. "Helicopter trim with flap-lag-torsion and stall by an optimized controller." *Journal of Guidance, Control and Dynamics*, Vol. 13, pp. 824-834, 1990.
50. Peters, D.A., and Barwey, D. "A General Theory of Rotorcraft Trim." *Proceedings of the 36th Structures, Structural Dynamics, and Materials Conference*, New Orleans, LA, April, 1995.
51. Bauchau, O.A., and Wang, J.L., "Stability Evaluation and System Identification of Flexible Multi-body Systems," *Multi-body System Dynamics*, Vol. 18, No. 1, pp. 95-106, October 2007.
52. Riviello, L., Bottasso, C., and Bauchau, O. A. "Effect of Modeling Approximations on the Stability of Autopilot Controllers," Proceedings of the 64th Annual American Helicopter Society Forum, Montreal, Canada, April 29 - May 1, 2008.
53. Matheron, G., "Principles of Geostatistics", *Economic Geology*, Vol. 58, pp. 1246–1266, 1990.
54. Cressie, N., "The Origins of Kriging", *Mathematical Geology*, Vol. 22, No. 3, pp. 239–252, 1990.
55. Zaki, A., Using Tightly-Coupled CFD/CSD Simulation for Rotorcraft Stability Analysis, Ph.D. Dissertation, Georgia Institute of Technology, expected August 2011.
56. Martin, J. D., and Simpson, T. M., "A Study on the Use of Kriging Models to Approximate Deterministic Computer Models," Proceedings of DETC'03 ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference, September 2003.
57. Warnes, J. J., and Ripley, B. D., "Problems with Likelihood Estimation of Covariance Functions of Spatial Gaussian Processes," *Biometrika*, Vol. 74, No. 3, pp. 640-642, September 1987.
58. Mardia, K. V., and Watkins, A. J., "On Multimodality of the Likelihood in the Spatial Linear Model," *Biometrika*, Vol. 76, No. 2, pp. 289-295, June 1989.
59. Runze, L., and Sudjianto, A., "Analysis of Computer Experiments using Penalized Likelihood in Gaussian Kriging Models," *Technometrics*, Vol. 47, No. 2, pp. 111-120, May 2005.
60. Thepvongs, S., Cook, J., Cesnik, C., and Smith, M.J., "Computational Aeroelasticity of Rotating Wings with Deformable Airfoils," Proceedings of the American Helicopter Society 65th Annual Forum, Grapevine, Texas, May 2009.
61. Abras, J., *Enhancement Of Aeroelastic Rotor Airload Prediction Methods,* Ph.D. Dissertation, Georgia Institute of Technology, May 2009.

62. Simo, J.C., "A Finite Strain Beam Formulation. The Three-Dimensional Dynamic Problem. Part I," *Computer Methods in Applied Mechanics and Engineering*, Vol. 49, pp. 55–70, 1985.
63. Potsdam, M., Yeo, H., and Johnson, W., "Rotor Airloads Prediction Using Loose Aerodynamic/Structural Coupling," American Helicopter Society 60th Annual Forum, Baltimore, MD, June 7-10, 2004.
64. Yu, Y. H., Tung, C., van der Wall, B., Pausder, H., Burley, C., Brooks, T., Beaumier, P., Delrieux, Y., Mercker, E., and Pengel, K., "The HART-II Test: Rotor Wakes and Aeroacoustics with Higher-Harmonic Pitch Control (HHC) Inputs - The Joint German/French/Dutch/US Project," American Helicopter Society 58th Annual Forum Proceedings, Montreal, Canada, June 11-13, 2002.
65. Lim, J. W., Nygaard, T. A., Strawn, R., and Potsdam, M., "BVI Airloads Prediction Using CFD/CSD Loose Coupling," AHS San Francisco Bay Area Chapter Specialists' Conference on Vertical Lift Aircraft Design, San Francisco, CA, January 18-20, 2006.
66. Makinen, S., Hill, M., Gandhi, F., Long, L. N., Vasilescu, R., and Sankar, L., "A Study of the HART-I Rotor with Loose Computational Fluid/Structural Dynamic Coupling," American Helicopter Society 62nd Annual Forum Proceedings, Phoenix, AZ, May 9-11, 2006.
67. Biedron, R. T. and Lee-Rausch, E. M., "Rotor Airloads Prediction Using Un- structured Meshes and Loose CFD/CSD Coupling," 26th AIAA Applied Aerodynamics Conference, AIAA-2008-7341, Honolulu, HI, Aug. 18-21, 2008.
68. Vatsa, V. N. and Carpenter, M. H., "Higher Order Temporal Schemes with Error Controllers for Unsteady Navier-Stokes Equations," 17th AIAA Computational Fluid Dynamics Conference, AIAA-2005-5245, Toronto, Ontario, Canada, June 6-9, 2005.
69. Bauchau, O. A., and Wang, J. L., "Efficient and Robust Approaches to the Stability Analysis of Large Multi-body Systems," *Journal of Computational and Nonlinear Dynamics*, Vol. 3, No. 1, pp. 1-12, 2008.
70. Skjoldan, P.F., and Bauchau, O. A., "Determination of Modal Parameters in Complex Nonlinear Systems," *Journal of Computational and Nonlinear Dynamics*, Vol. 6, No. 3, pp. 031017 1-19, 2010.
71. Friedmann, P.P., "Numerical Methods for Determining the Stability and Response of Periodic Systems with Applications to Helicopter Rotor Dynamics and Aeroelasticity," *Computers and Mathematics with Applications*, Vol. 12A, No. 1, 1986, pp. 131–148.
72. Gaonkar, G.H., and Peters, D.A., "Review of Floquet Theory in Stability and Response Analysis of Dynamic Systems with Periodic Coefficients," *R.L. Bisplinghoff Memorial Symposium Volume on Recent Trends in Aeroelasticity, Structures and Structural Dynamics*, University Press of Florida, Gainesville, pp. 101-119, 1986.
73. Subramanian, S., Gaonkar, G.H., Nagabhushanam, J., and Nakadi, R.N., "Parallel Computing Concepts and Methods for Floquet Analysis of Helicopter Trim and Stability," *Journal of the American Helicopter Society*, Vol. 41, No. 4, pp. 370–382, 1996.
74. Floquet, G., "Sur les equations Differentielles Lineaires a Coefficients Periodiques,"*Annales Scientifiques de l'Ecole Normale Superieure*, Vol. 12, pp. 47–88, 1983.
75. Bauchau, O.A., and Wang, J.L., "Stability Analysis of Complex Multi-body Systems," *Journal of Computational and Nonlinear Dynamics*, Vol. 1, No. 1, pp. 71–80, 2006.

76. Bauchau, O. A., and Wang, J., "Efficient and Robust Approaches for Rotorcraft Stability Analysis," *Journal of the American Helicopter Society*, Vol. 55, No. 3, pp. 032006 1-9, 2010.
77. Zaki, A., Reveles, N., Bauchau, O., and Smith, M.J., "Using Tightly-Coupled CFD/CSD Simulation for Rotorcraft Stability Analysis," Proceedings of the American Helicopter Society 66th Annual Forum, Phoenix, Arizona, May 11-13, 2010.
78. Shelton, A. B., Braman, K., Smith, M. J., and Menon, S., "Improved Turbulence Modeling for Rotorcraft," Proceedings of the American Helicopter Society 62nd Annual Forum, Phoenix, AZ, May 9-11, 2006.
79. Sankaran, Venkateswaran, private communication, Ames Research Center, Feb. 2011.