A SCALABLE, PARALLEL APPROACH FOR MULTI-POINT, HIGH-FIDELITY AEROSTRUCTURAL OPTIMIZATION OF AIRCRAFT CONFIGURATIONS

by

Gaetan K. W. Kenway

A thesis submitted in conformity with the requirements for the degree of Doctor of Philosophy
Graduate Department of Aerospace Engineering
University of Toronto

Copyright © 2013 by Gaetan K. W. Kenway
Abstract

A Scalable, Parallel Approach for Multi-Point, High-Fidelity Aerostructural Optimization of Aircraft Configurations

Gaetan K. W. Kenway
Doctor of Philosophy
Graduate Department of Aerospace Engineering
University of Toronto
2013

This thesis presents new tools and techniques developed to address the challenging problem of high-fidelity aerostructural optimization with respect to large numbers of design variables. A new mesh-movement scheme is developed that is both computationally efficient and sufficiently robust to accommodate large geometric design changes and aerostructural deformations. A fully coupled Newton–Krylov method is presented that accelerates the convergence of aerostructural systems and provides a 20% performance improvement over the traditional nonlinear block Gauss–Seidel approach and can handle more flexible structures. A coupled adjoint method is used that efficiently computes derivatives for a gradient-based optimization algorithm. The implementation uses only machine accurate derivative techniques and is verified to yield fully consistent derivatives by comparing against the complex step method. The fully-coupled large-scale coupled adjoint solution method is shown to have 30% better performance than the segregated approach. The parallel scalability of the coupled adjoint technique is demonstrated on an Euler Computational Fluid Dynamics (CFD) model with more than 80 million state variables coupled to a detailed structural finite-element model of the wing with more than 1 million degrees of freedom.

Multi-point high-fidelity aerostructural optimizations of a long-range wide-body, transonic transport aircraft configuration are performed using the developed techniques. The aerostructural analysis employs Euler CFD with a 2 million cell mesh and a structural finite element model with 300,000 DOF. Two design optimization problems are solved: one where takeoff gross weight is minimized, and another where fuel burn is minimized. Each
optimization uses a multi-point formulation with 5 cruise conditions and 2 maneuver conditions. The optimization problems have 476 design variables and optimal results are obtained within 36 hours of wall time using 435 processors. The TOGW minimization results in a 4.2% reduction in TOGW with a 6.6% fuel burn reduction, while the fuel burn optimization resulted in a 11.2% fuel burn reduction with no change to the takeoff gross weight.
Acknowledgements

First and foremost, I would like to offer my sincerest thanks to my thesis supervisor, Dr. Joaquim Martins. His knowledge and enthusiasm for all things aircraft design and MDO continues to inspire me. His gentle encouragement of “no more coding!” four months ago finally prompted me to start writing this thesis. Working with Dr. Martins has been very enjoyable and I look forward to continued collaboration at the University of Michigan.

I would also like to thank the other members of my DEC committee, Dr. Grant and Dr. Zingg, for their helpful comments and suggestions over the past five years. There is no doubt their feedback improved the overall quality of my thesis.

My colleagues over the years at the MDO lab have made my time at UTIAS thoroughly enjoyable. Special thanks goes out to Sandy Mader who laid the foundations from which I started my research. He also helped me tremendously during the first years of my Ph.D and our discussions continue to be fruitful. I would also like to thank Graeme Kennedy for the excellent structural solver TACS. Without his code and expertise, the results in this thesis would simply not have been possible.

Finally, I would not be where I am today if it was not for the continuous encouragement from my parents who instilled in me the love of learning.
# Contents

List of Tables vi

List of Figures vii

List of Symbols and Abbreviations x

1 Introduction 1

1.1 Evolution of Jet Transport Configurations .............................. 1
1.2 Multidisciplinary Design Optimization of Aircraft Configurations ...... 3
1.3 Aerostructural Analysis and Optimization ............................... 4
1.4 Thesis Outline and Contributions ......................................... 7

2 Aerostructural Solution Methodology 10

2.1 Geometric Representation and Manipulation ............................... 10
2.2 Mesh Deformation ............................................................ 17
2.3 Aerodynamic Analysis ......................................................... 29
2.4 Skin Friction Estimate ........................................................ 34
2.5 Structural Analysis ............................................................ 35
2.6 Load and Displacement Transfer .............................................. 36
2.7 Aerostructural Analysis ......................................................... 38
2.8 Conclusions ........................................................................... 44

3 Gradient Computation Methodology 45

3.1 Coupled Aerostructural Adjoint Overview ................................. 47
3.2 Aerodynamic Adjoint ............................................................ 48
3.3 Structural Adjoint .................................................................. 53
3.4 Geometry Derivatives ............................................................. 53
3.5 Coupled Adjoint Implementation .............................................. 54
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Nonlinear solution method comparison</td>
<td>43</td>
</tr>
<tr>
<td>3.1</td>
<td>Aerodynamic residual matrix timings</td>
<td>53</td>
</tr>
<tr>
<td>3.2</td>
<td>Coupled adjoint sensitivity verification</td>
<td>60</td>
</tr>
<tr>
<td>3.3</td>
<td>Adjoint solution method comparison</td>
<td>66</td>
</tr>
<tr>
<td>3.4</td>
<td>Mesh sizes</td>
<td>68</td>
</tr>
<tr>
<td>3.5</td>
<td>Aerostructural scaling results</td>
<td>68</td>
</tr>
<tr>
<td>3.6</td>
<td>Coupled adjoint timings</td>
<td>70</td>
</tr>
<tr>
<td>3.7</td>
<td>Design variables</td>
<td>72</td>
</tr>
<tr>
<td>4.1</td>
<td>Objective sensitivities</td>
<td>78</td>
</tr>
<tr>
<td>5.1</td>
<td>CRM specifications</td>
<td>83</td>
</tr>
<tr>
<td>5.2</td>
<td>Operating conditions</td>
<td>84</td>
</tr>
<tr>
<td>5.3</td>
<td>Design variables</td>
<td>85</td>
</tr>
<tr>
<td>5.4</td>
<td>Optimization constraints</td>
<td>87</td>
</tr>
<tr>
<td>5.5</td>
<td>Optimization results summary</td>
<td>101</td>
</tr>
<tr>
<td>6.1</td>
<td>Comparison of file-based and memory-based coupling schemes</td>
<td>107</td>
</tr>
</tbody>
</table>
List of Figures

1.1 Topics addressed in this thesis ................................................. 8
2.1 Common Research Model B-spline surfaces .......................... 13
2.2 Local FFD shape control .................................................. 14
2.3 Multi-block FFD approach for complex configurations .......... 15
2.4 Continuous aileron deformation using a nested FFD approach .... 16
2.5 Complete structural mesh automatically generated with pyLayout . 16
2.6 Exploded view of pyLayoutGeo generated wingbox ................. 17
2.7 Algebraic warping scheme ................................................ 20
2.8 Hybrid mesh perturbation scheme ..................................... 22
2.9 Generic volume approximated with tri-cubic Bézier volume. .... 24
2.10 Solid mesh warping procedure ........................................ 25
2.11 Winglet design variable modification and aerostructural solution . 26
2.12 Quality comparison between original and warped meshes ........ 26
2.13 Aerostructural solution on mesh deformed using nonlinear warping technique . 27
2.14 Performance comparison between linear and nonlinear warping strategies ... 28
2.15 Comparison of cubic backtracking line-search and non-monotone line-search . 33
2.16 Convergence history for Runge–Kutta solver and Newton–Krylov solver ... 34
2.17 Meshes used for skin friction drag computation. .................. 35
2.18 Schematic of load-displacement transfer operation ................ 37
3.1 Scaling behaviour for gradient-based and non gradient-based optimizers ... 46
3.2 $\partial A/\partial w$ stencil, coloring and three dimensional packing ........ 51
3.3 $\partial A/\partial X_v$ stencil, coloring and three dimensional packing .......... 52
3.4 Adjoint procedure for solid mesh warping .......................... 55
3.5 Verification of smoothness of coupled adjoint derivatives .......... 61
3.6 CFD and CSM solution comparison for each mesh level ............. 69
3.7 NLBGS convergence for the three mesh levels ........................ 71
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.8</td>
<td>Convergence history for lift adjoint using the coupled Krylov adjoint solver</td>
<td>71</td>
</tr>
<tr>
<td>3.9</td>
<td>Gradient evaluation cost for finite differencing and the coupled adjoint method</td>
<td>73</td>
</tr>
<tr>
<td>4.1</td>
<td>TOGW and fuel burn objective sensitivities</td>
<td>79</td>
</tr>
<tr>
<td>4.2</td>
<td>Histogram of 101,059 flights for the 777-200ER aircraft during 2011</td>
<td>80</td>
</tr>
<tr>
<td>5.1</td>
<td>Common research model aerostructural solution</td>
<td>82</td>
</tr>
<tr>
<td>5.2</td>
<td>Geometric design variables and FFD volumes for CRM optimization problem</td>
<td>86</td>
</tr>
<tr>
<td>5.3</td>
<td>Structural design variable groups</td>
<td>87</td>
</tr>
<tr>
<td>5.4</td>
<td>Visualization of geometric constraints for optimization</td>
<td>89</td>
</tr>
<tr>
<td>5.5</td>
<td>Processor groups for the multi-point aerostructural optimization</td>
<td>90</td>
</tr>
<tr>
<td>5.6</td>
<td>Optimization convergence histories for each optimization</td>
<td>91</td>
</tr>
<tr>
<td>5.7</td>
<td>Planform view with $C_p$ contours for optimum designs</td>
<td>93</td>
</tr>
<tr>
<td>5.8</td>
<td>Weight and drag evolution during optimization</td>
<td>94</td>
</tr>
<tr>
<td>5.9</td>
<td>Twist and thickness to chord variation for the initial and optimized designs</td>
<td>95</td>
</tr>
<tr>
<td>5.10</td>
<td>Aerostructural deflections for 2.5 g maneuver and 1 g cruise condition</td>
<td>95</td>
</tr>
<tr>
<td>5.11</td>
<td>Spanwise lift distributions for the initial and optimized designs</td>
<td>96</td>
</tr>
<tr>
<td>5.12</td>
<td>Cross section and $C_p$ contours for each operating condition at 66% semi-span</td>
<td>99</td>
</tr>
<tr>
<td>5.13</td>
<td>Thickness and failure parameter distributions for initial and optimized designs</td>
<td>100</td>
</tr>
<tr>
<td>6.1</td>
<td>Process and data flow for a script-driven MDA using file input/output</td>
<td>104</td>
</tr>
<tr>
<td>6.2</td>
<td>UML class diagram overview for multi-point aerostructural optimization</td>
<td>108</td>
</tr>
<tr>
<td>6.3</td>
<td>DVGeometry and DVConstraints UML class diagram</td>
<td>109</td>
</tr>
<tr>
<td>6.4</td>
<td>Aerostructural solution and coupled adjoint UML class diagram</td>
<td>110</td>
</tr>
<tr>
<td>A.1</td>
<td>Simple transonic wing specifications</td>
<td>128</td>
</tr>
<tr>
<td>A.2</td>
<td>Simple transonic wing grids</td>
<td>129</td>
</tr>
</tbody>
</table>
## List of Symbols and Abbreviations

### Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AD</td>
<td>Automatic Differentiation</td>
</tr>
<tr>
<td>ASM</td>
<td>Additive–Schwartz Method</td>
</tr>
<tr>
<td>CAD</td>
<td>Computer Aided Design</td>
</tr>
<tr>
<td>CFD</td>
<td>Computational Fluid Dynamics</td>
</tr>
<tr>
<td>CSM</td>
<td>Computational Structural Mechanics</td>
</tr>
<tr>
<td>CK</td>
<td>Coupled Krylov</td>
</tr>
<tr>
<td>CNK</td>
<td>Coupled Newton–Krylov</td>
</tr>
<tr>
<td>CRM</td>
<td>Common Research Model</td>
</tr>
<tr>
<td>DOF</td>
<td>Degree of Freedom</td>
</tr>
<tr>
<td>FFD</td>
<td>Free Form Deformation</td>
</tr>
<tr>
<td>FGMRES</td>
<td>Flexible-GMRES</td>
</tr>
<tr>
<td>GMRES</td>
<td>Generalized Minimum RESidual</td>
</tr>
<tr>
<td>ILU</td>
<td>Incomplete LU-factorization</td>
</tr>
<tr>
<td>KS</td>
<td>Kreisselmeier–Steinhauser (constraint aggregation technique)</td>
</tr>
<tr>
<td>KKT</td>
<td>Karcsh–Kuhn–Tucker</td>
</tr>
<tr>
<td>JST</td>
<td>Jameson–Schmidt–Turkel artificial dissipation scheme</td>
</tr>
<tr>
<td>MDO</td>
<td>Multidisciplinary Design Optimization</td>
</tr>
<tr>
<td>LBGS</td>
<td>Linear Block Gauss–Seidel</td>
</tr>
<tr>
<td>MAC</td>
<td>Mean Aerodynamic Chord</td>
</tr>
<tr>
<td>MTOW</td>
<td>Maximum TakeOff Weight</td>
</tr>
<tr>
<td>NLGBS</td>
<td>Nonlinear Block Gauss–Seidel</td>
</tr>
<tr>
<td>OEW</td>
<td>Operational Empty Weight</td>
</tr>
<tr>
<td>OML</td>
<td>Outer Mold Line</td>
</tr>
<tr>
<td>PETSc</td>
<td>Portable Extensible Toolkit for Scientific Computing (software)</td>
</tr>
</tbody>
</table>
RANS  Reynolds-Averaged Navier Stokes
RCM   Reverse Cuthill–McKee
SNOPT Sparse Nonlinear OPTimizer (software)
SUmb  Stanford University multi-block CFD solver
SuperLU_DIST SuperLU Distributed memory version (software)
TACS  Toolkit for the Analysis of Composite Structures
TFI   Transfinite Interpolation
TOGW  Takeoff Gross Weight

Symbols

\( A \)  Aerodynamic residuals
\( b \)  Span
\( c \)  Speed of sound
\( C_D \) Coefficient of drag
\( C_F \) Skin friction coefficient
\( C_p \) Coefficient of pressure
\( C_L \) Coefficient of lift
\( c_T \) Thrust-specific fuel consumption
\( \mathcal{D} \) Artificial dissipation
\( F \)  Flux vector
\( F \)  Structural force vector
\( F_A \) Aerodynamic force vector
\( FF \) Friction calculation form factor
\( I \)  Function of interest
\( K \)  Structural stiffness matrix
\( K_M \) Mesh stiffness matrix
\( \mathcal{M} \) Mesh movement operator
\( N_P \) Number of processors
\( p \)  Pressure
\( r \)  Vector between coordinate on \( X_S \) and structural mesh
\( R \)  Aircraft range
\( \mathcal{R} \) All residuals, \( \mathcal{R}^T = [A^T \ S^T] \)
\( R_M \) Mesh restriction operator
\( S \) Structural residuals
\( T \) Displacement transfer matrix
\( T^T \) LOAD Transfer matrix
\( u \) Structural states
\( u_A \) Structural displacements on aerodynamic surface
\( u_t \) Translational components of displacement
\( u_r \) Rotational components of displacement
\( w \) Aerodynamic states
\( W_1 \) Weight at start of cruise
\( W_2 \) Weight at end of cruise
\( x \) Design variables
\( X_J \) Jig coordinates (undisplaced)
\( X_S \) Surface coordinates (displaced)
\( X_{SW} \) Intermediate mesh of Bézier cubic spline interpolation
\( X_V \) Volume mesh coordinates
\( Y_S \) Subset of \( Y_V \) with specified displacements
\( Y_V \) Coarse version of volume mesh, \( X_V \)
\( \beta \) Objective sensitivity with respect to \( W_2 \)
\( \gamma \) Objective sensitivity with respect to \( C_D \)
\( \epsilon_A \) Convergence tolerance for aerodynamic solution
\( \epsilon_{AS} \) Convergence tolerance for aerostructural solution
\( \epsilon_S \) Convergence tolerance for structural solution
\( \epsilon_{SA} \) Convergence tolerance for aerostructural adjoint solution
\( \chi \) Colour index for matrix colouring
\( \theta \) Under-relaxation parameter
\( \psi \) Aerodynamic adjoint vector
\( \phi \) Structural adjoint vector
\( \Psi \) Coupled adjoint vector, \( \Psi^T = [\psi^T \phi^T] \)
Chapter 1

Introduction

1.1 Evolution of Jet Transport Configurations

In the almost 110 years since the first heavier-than-air flight by the Wright brothers, there has been remarkable technological progress. The first Wright Flyer flew at just 20mph and at an altitude barely above the treetops. Less than 50 years later, the first jet airliner to enter commercial service, the De Havilland Comet, flew at over 500mph at altitudes in the lower stratosphere.

The driving philosophy during this early phase of powered flight is best described as “higher and faster” — new technological advances such as the monoplane design, aluminum alloys, NACA cowlings, superchargers, and finally the jet and rocket engine helped push new aircraft to ever faster speeds at ever higher altitudes.

Over the last five decades, the “higher and faster” mantra has been largely replaced by one of “safer, cheaper, more reliable, and more environmentally friendly”. Consider, for example, Boeing’s first commercially successful airliner, the Boeing 707, and Boeing’s most technologically advanced design, the Boeing 787. Both aircraft fly at almost the same cruise speed, at the same altitudes, and both configurations are similar: each aircraft has a cylindrical fuselage, a swept wing, a conventional empennage and engines mounted in pods under the wing. The similarities end there, however. The newer aircraft is built primarily of carbon fibre composites, uses much more efficient high bypass ratio turbofan engines, and has much improved aerodynamic performance. These factors combine to give the Boeing 787 a fuel burn per passenger mile less than half of that of the fifty year old 707 [1].

The drive to reduce fuel burn, through engine improvements, drag reduction, or weight reduction has been a key design driver for commercial jet aircraft since the very beginning. The main driving factors behind the continual quest for lower fuel burn have changed over
the years, however. The first generation of jet airliners such as the De Havilland Comet, the Boeing 707 and the Douglas DC-8 were highly inefficient, due primarily to the use of turbojet engines. During this period, the main reason for reducing fuel consumption was to increase aircraft payload and range capability. The cost of fuel was of little concern.

The oil crises in the 1970’s added an additional motivation to reduce fuel burn: the operational cost. The soaring cost of jet fuel during this period substantially reoriented design priorities. For example, the original target cruising Mach number for the Boeing 747, designed before the oil crisis, was to be Mach 0.90 [2]. In contrast, the Boeing 757 and 767 aircraft designed in the late 1970’s prioritized operating costs from the outset and this resulted in a significant reduction in cruise Mach number to just 0.80 [3, 4].

Over the last twenty years, concerns regarding the environmental impact of aviation have become increasingly prominent. It is now generally accepted that carbon dioxide and other emissions produced since the Industrial Revolution have increased the CO₂ concentration in the atmosphere and that this has contributed to the overall warming of the planet [5]. While aviation’s CO₂ emissions are currently 2% of all human sources [6], the anticipated global growth of air travel [7, 8] in the coming decades coupled with the unknown additional effects of emissions at high altitudes places further pressure for more efficient aircraft designs.

These three key concerns continue to drive improvements in engine and airframe technologies, resulting in each successive generation of aircraft improving the fuel burn of the previous generation. Lee [9] estimates that the fuel burn per passenger mile is up to 70% less than the first generation of jet airliners. While these are certainly impressive gains, new improvements are becoming more difficult to obtain from highly-refined conventional aircraft designs.

In recent years, there have been several proposals for unconventional aircraft designs that will result in significant improvements to aircraft efficiency. These unconventional configurations include the Blended Wing Body (BWB) concept [10], the joined wing concept [11, 12] and the strut-braced wing concept [13]. Liebeck et al. [10] for example, predicts a 30% reduction in fuel burn for an Airbus A380-sized BWB aircraft relative to a conventional design with similar technology.

Novel aircraft configurations certainly have the potential to be more efficient, but what remains to be seen is whether or not the “first generation” of such new configurations will be competitive with existing configurations that have been refined over many generations. Given the immense cost and economic risk associated with unconventional configurations, it is critical the we increase our confidence that an unconventional configuration is capable of
delivering the promised benefits. In essence, for an unconventional aircraft configuration, we need to guarantee that the first design is the best design. However, unlike conventional aircraft designs, it is not possible to draw upon many years of empirical knowledge. We need to find a way to ensure that the lack of empirical knowledge does not result in a sub-optimal design.

### 1.2 Multidisciplinary Design Optimization of Aircraft Configurations

Fortunately, we now have at our disposal, more than at any point in history, the knowledge, expertise, and computing capacity to tackle this problem head on. Computational analysis of aerospace systems has evolved tremendously over the past decades. The critical importance of accurate predictions of aerodynamic drag, weight and structure failure, has positioned the aerospace industry at the forefront of the development and usage of Computational Fluid Dynamics (CFD) and Computational Structural Mechanics (CSM). These tools allow engineers to virtually validate many aspects of the design without physical testing, significantly reducing validation costs. A question remains, however: how can we systematically find promising unconventional designs with limited engineering intuition?

One potential solution is numerical optimization — the systematic selection of the best solution among many alternatives. Optimization of various aspects of aircraft systems is becoming increasingly common \[14, 15, 16, 17\] due in large part, to the exponential growth of computing capacity at ever reducing costs. Optimization’s importance cannot be overstated, as engineering intuition can become unreliable when dealing with complex interacting systems. This is especially true when engineers have no practical experience with the systems under study. The application of optimization algorithms to complex engineering designs represents a significant step forward for the role of computing in engineering design.

The application of numerical optimization to a real-world aircraft design optimization problem is extremely challenging. The design must balance conflicting performance requirements and regulatory constraints covering all aspects of the aircraft. Due to the coupled nature of aircraft design, modifications made to improve the performance in one discipline may degrade performance or violate a constraint in a different discipline.

To ensure candidate designs obtained using numerical optimization are feasible, it is critical to directly model the interaction of coupled disciplines. For a transport aircraft design problem, these disciplines include high-speed aerodynamics, low-speed aerodynamics, structures, weights and balance, stability and control, propulsion, noise, manufacturing, and
Two of the most strongly interacting aircraft disciplines are aerodynamics and structures. Coupling aerodynamic and structural numerical models to compute the static aeroelastic shape of lifting surfaces is essential when designing lifting surfaces that are flexible. Even small changes in shape can have a large effect on the aerodynamic performance, and multiple flow conditions result in multiple shapes. This is particularly important for swept wings, where bending-twist coupling can result in changes in the twist distribution, which have a large effect on the spanwise lift distribution, with strong repercussions in lift induced drag and structural sizing [18]. The analysis of static aeroelastic shapes was introduced as soon as the first simple numerical models for aerodynamics and structures matured. Brown [19], for example, coupled a lifting line model to a beam theory model to obtain the flying shape of a swept wing, and earlier work exists that considered even simpler models [20]. In recent years, this coupling has become even more important, because the trend has been to increase the aspect ratio of aircraft wings, making them more flexible. Wing flexibility impacts not only the static flying shape of the wing, but also its dynamics, resulting in undesirable aeroelastic phenomena such as flutter and aileron reversal. In the present work, however, we restrict ourselves to the analysis of the static aeroelastic shape, which we refer to as *aerostructural* analysis.

The need to include the relevant disciplines in such design problems led to the field of multidisciplinary design optimization (MDO) [21]. MDO provides a means to account for and take advantage of the interactions between disciplines. The efficient optimization of aerostructural systems using high-fidelity analysis methods is the primary topic of this thesis.

### 1.3 Aerostructural Analysis and Optimization

One of the earliest examples of *aerostructural* optimization was by Haftka [22], who combined a panel method with a finite element analysis to iteratively obtain the deformed shape and minimize the weight subject to drag and stress constraints. This enabled the comparison of trade-offs between structural weight and induced drag for aluminium and composite wings.

Grossman *et al.* [23] investigated the aerostructural analysis and optimization of a sailplane and a subsonic forward-swept transport wing. They showed that a sequential optimization approach—where aerodynamic optimization and structural sizing are performed iteratively in sequence—produced a suboptimal result relative to a fully integrated aerostructural op-
timization. This failure of sequential optimization to produce the optimal result is further explained by Chittick and Martins [24].

With the advent of higher fidelity modelling in both structures and aerodynamics, numerical optimization has been extensively applied to each discipline separately. On the structures side, increasingly detailed finite-element models have been used in wing structural sizing optimization [25]. The increase in fidelity of the structural model is required for a more refined sizing of the structure while considering complex structural failure constraints, leading to a better estimate of the optimized structural weight. The fidelity of models used for aerodynamic shape optimization has also been increasing, and it is now possible to use CFD to optimize a design with respect to hundreds of design variables [26, 27, 28, 29]. In the design of transonic wings it is particularly important to use high-fidelity models in order to correctly account for the wave drag. In addition, to take advantage of these models, large numbers of airfoil shape variables are required to effectively reduce the wave drag of a wing. To handle the large numbers of design variables, the efforts above employed gradient-based optimization algorithms together with adjoint methods to compute the required gradients efficiently. Unfortunately, aerodynamic shape optimization alone is insufficient for wing design, since it is impossible to perform the trade-offs for wing thickness, span, and sweep, which require a model of how the wing weight and flexibility varies with respect to these parameters.

Given the importance of coupling the aerodynamics and the structures in wing design, coupling high-fidelity versions of these models for analysis and design optimization is a natural extension of the work done to date. Various techniques have been proposed over the years for coupling CFD to CSM solvers, including different load and displacement transfer schemes [30, 31, 32], and solution techniques for solving the coupled system of equations [33, 34].

On the design optimization side, Maute et al. [35] pioneered high-fidelity aerostructural optimization by coupling an Euler flow solver to a linear finite-element model of the structure. They developed a nonlinear block Gauss–Seidel method with relaxation, for the solution of the aerostructural system, and they deform the CFD mesh by a spring analogy method. They demonstrated the method by performing an optimization with respect to five design variables using gradients computed with the direct method. The number of design variables was severely limited because the cost of computing the gradient with the direct method is proportional to the number of design variables.

Martins et al. [36] proposed the use of a coupled adjoint method for aerostructural design
optimization using Euler CFD and linear finite-element analysis in a two-field formulation (CFD and CSM state variables). They showed that the cost of computing the gradient using this method could be made nearly independent of the number of design variables, and thus were able to compute gradients with respect to thousands of variables [37]. Then, they applied this method to the aerostructural design of a supersonic business jet with respect to 97 shape and sizing variables [38]. One of the main advances in this work was the elimination of the minimum airfoil thickness constraints that must usually be enforced in aerodynamic shape optimization. By including the structure and simultaneously optimizing with respect to the aerodynamic shape and structural sizing, they can use the trade-off between drag and structural weight to naturally determine the optimal thickness-to-chord ratio. The consideration of these aerostructural trade-offs also enables the optimization of wing planform variables, such as span and sweep, but the work did not include these variables because of limitations in the CFD mesh movement scheme.

Maute et al. [39] presented another coupled adjoint formulation using discrete-analytical sensitivities. They compared it to the direct method and showed that the accuracy of the two methods is identical. To improve the robustness and efficiency of aerostructural solution methods, Barcelos et al. [40] developed a class of Newton–Krylov–Schur methods for solving the coupled nonlinear fluid-structure-mesh deformation problem. Their approach used an approximate Newton method for the solution of the nonlinear coupled equations. At each iteration, a Schur-complement approach is used to solve the coupled linear system that results from a linearization of the residual. They found that their technique is more robust and efficient than the original Gauss–Seidel method by Maute et al. [35]. More recently, Barcelos et al. [41] presented an aerostructural solution technique that couples the Reynolds averaged Navier–Stokes (RANS) equations to a linear structural model and a mesh-deformation strategy. They used the direct method for computing the coupled derivatives and they applied it to solve an optimization problem with five design variables.

Other examples of aerostructural optimization include the preliminary design optimization of a business jet by Piperni et al. [42]. A transonic small-disturbance code was used to compute the aerodynamics loads and performance, and a finite element model was used to compute the displacements, mass and stresses. Another example is the work by DeBlois et al. [43], which used a similar framework to investigate the effects of metallic and composite materials on aerostructural optimization. Neither of these two efforts used adjoint methods for computing the gradients and are therefore not scalable. In addition, some of the optimization procedure was sequential, which most likely resulted in suboptimal results. However,
it must be emphasized that this work was focused on developing a framework that can be implemented in an industrial setting and on results that can be usable in practice, so these constitute impressive efforts.

In spite of all these developments, we are still missing a high-fidelity fully integrated aerostructural-design optimization approach that is scalable with respect to the number of design variables and to the number of degrees of freedom in the aerodynamic and structural models. The coupled aerostructural analysis must also be fast enough so that multiple flight conditions and all relevant load cases are considered. This is needed to realize the full potential of this approach to produce wings with optimal static aeroelastic tailoring that are practical. Thus, the objective of this thesis is to develop an approach that enables the simultaneous design optimization of the aerodynamic shape and structural sizing with respect to hundreds (or even thousands) of variables with many practical considerations. The effective use of the resulting aerostructural optimization method is then demonstrated through two sample optimization problems for a conventional configuration.

I have selected a conventional configuration to analyze in an attempt to understand the multidisciplinary trade-offs obtained during the optimization process. As noted previously, the true power of high-fidelity multidisciplinary optimization however, comes from the optimization of unconventional configurations. With many decades of experience with swept-wing configurations, modern transport aircraft are refined designs that incorporate the correct multidisciplinary trade-offs. I would therefore expect the design obtained from the multidisciplinary optimization of transonic transport configuration to not diverge significantly from actual transport aircraft. Close agreement between the optimized configuration and similar actual aircraft would suggest the high-fidelity analysis is accurately capturing the multidisciplinary trade-offs. The type of optimizations presented in this thesis are a critical first step before we can have confidence in the design of unconventional configurations.

1.4 Thesis Outline and Contributions

The central goal of this thesis is to advance the state-of-the art of aircraft design by using high-fidelity aerostructural optimization. The relationships between the topics addressed are shown in Figure 1.1.

To meet the goal of multi-point high-fidelity aerostructural optimization, I made several contributions to the analysis and gradient evaluation of high-fidelity aerostructural systems. I present a multi-level, multiblock FFD approach for geometric parametrization that provides consistent parametrization across disciplines. Since aerostructural analysis is computation-
Figure 1.1: Topics addressed in this thesis. Shaded topics indicated areas with contributions. 

ally demanding, it is critical that each discipline level analysis is as fast as possible. I implemented a Newton–Krylov solution method that uses an explicit Runge–Kutta start-up strategy that is both fast and robust. Significant additional demands are placed on the mesh movement scheme that are not seen with typical aerodynamic optimization. To meet the challenge, I develop a novel hybrid algebraic linear-elasticity mesh movement scheme that is able to satisfy the challenging requirements of robustness and computational efficiency. On the gradient evaluation aspect, I developed a new forward-mode automatic differentiation method for evaluating the residual matrices for CFD that is both computationally efficient and offers increased code maintainability. The first high-fidelity coupled adjoint implementation is then described. This method is unique as it uses entirely machine-accurate sensitivity methods — the accuracy of which has been favorably compared to the complex step method. Finally, the coupled Krylov solution technique for solving the coupled adjoint equations is shown to be more efficient than traditional segregated approaches.

The components comprising the aerostructural analysis and the two aerostructural solution methods are discussed in Chapter 2. The derivative computations for each of these components and the coupled adjoint implementation are described in Chapter 3. The accuracy of the coupled adjoint is compared with the complex-step method, and several studies demonstrate the superior performance of the methods over previous implementations. The coupled adjoint implementation is described in the AIAA journal paper “A scalable parallel approach for high-fidelity steady-state aeroelastic analysis and derivative computations” [44]. Chapter 4 presents a preliminary investigation of the importance of aerostructural objective functions and highlights the additional difficulties of selecting the “right” objective in a multidisciplinary setting. Finally, the results from two optimizations are presented in Chapter 5 using the techniques presented in the preceding chapters. The design problem formulation and optimization results are presented in the Journal of Aircraft paper “Multi-Point High-
Fidelity Aerostructural Optimization of a Transport Aircraft Configuration” [45].
Chapter 2

Aerostructural Solution Methodology

This chapter describes the underlying computational methods and tools that form the high-fidelity aerostructural solver. Each component is discussed individually, including the geometry generation and manipulation, aerodynamic analysis, structural analysis, mesh-deformation algorithm, and load and displacement transfer. Finally, two solution strategies for the coupled aerostructural system are presented and compared. To ensure the overall aerostructural solution method is efficient and scalable, I made contributions to several of the components and these improvements are highlighted in their respective sections.

2.1 Geometric Representation and Manipulation

The representation and manipulation of complex geometries is a central concern for high-fidelity multidisciplinary optimization. The parametrization of the complete multidisciplinary system must be considered. A technique that is suitable for one discipline may not be for the remaining disciplines. The parametrization of the aerodynamic discipline typically requires an accurate, water-tight, representation of the wetted surface, or Outer Mold Line (OML) but not require a representation of the internal structure. The structural discipline, however, requires specification of the internal components, which may not be precisely co-incident with the OML, and how they change with the design parameters. The ideal geometric parametrization technique for high-fidelity aerostructural optimization should fulfill the following requirements:

**Exact geometry:** Arbitrary initial geometries can be represented exactly.

**Global control:** Simple design variables can represent large changes to the geometry. Conceptual aircraft design variables such as span, taper ratio, and twist should be possible.
Local shape control: Detailed shape changes can be applied to an isolated, small portion of the geometry with no effect on the remainder of the geometry. This is known as a compact basis.

Smoothness: Geometric modifications are at least $C^1$ continuous. The introduction of slope discontinuities resulting from a $C^0$ modification can cause convergence problems for the aerodynamics and stress concentrations for the structures.

Constant topology: Design variable changes do not modify the geometry’s topology.

Sensitivities: The derivative of geometry with respect to design parameters is computationally efficient and numerically accurate. Analytic or complex-step derivatives are preferred.

Geometric fidelity independent parametrization: Simple, global, design variables can be used to manipulate a complex geometry.

Consistent parametrization: All disciplines use the same geometric parametrization.

A number of potential techniques were considered including a customized aircraft geometry tool, Aerosurf, B-spline surface patches, a CAD based approach and a Free-Form Deformation (FFD) volume approach. A brief description of each approach and the potential shortcomings for each are described.

Aerosurf: This is a custom geometry generator written specifically for conventional aircraft configurations [46, 47, 48]. An aircraft is defined by a series of wing, fuselage and nacelle type bodies. Each section is specified by a fixed number of cross sections and design variables are used to modify the un-intersected cross sections. Aerosurf then intersects of all bodies automatically to form a water-tight surface. This tool has been successfully applied to aerodynamic and simple aerostructural optimizations in the past [46, 15], but several issues prevented its use for this thesis. The first is common to all surface-based methods. When only the OML is specified by the geometry engine, the corresponding changes to the internal structure are not defined. A “mesh warping” approach for this problem was investigated, but proved cumbersome, not general, and overly complex. Secondly, Aerosurf does not provide any sensitivity information. This was partly remedied by using a complex-variable version and the complex step approach [49], but issues remained due to the non-smooth nature of derivatives resulting
from intersecting bodies. Thirdly, the class of geometries representable by Aerosurf were somewhat limited and did not permit the use of an externally defined geometry without reverse engineering.

**CAD-based:** A CAD based approach has several desirable characteristics, the most important of which is the potential for direct integration into the work flow in a typical industrial engineering setting. CAD approaches are typically very robust, especially when large geometric changes are involved. Alonso *et al.* [50] investigated this approach using the CAPRI API, a vendor neutral approach for accessing information directly from the CAD application [51]. However, there are many drawbacks, including the lack of sensitivity information, very slow rebuild times, and discontinuous derivatives due to topological changes. For aerostructural applications, the complete structural model must be modeled directly. Since the structural model will contain a large number of components (ribs, spars, stringers, etc.) and potentially hundreds of intersections, rebuild times for a complete aerodynamic and structural model are expected to be significantly higher than for an aerodynamic only model. Since finite differencing is typically the only method available to obtain sensitivity information, the prospect of a large number of costly rebuilds was sufficient to consider alternative approaches.

**B-spline based:** This approach has been used successfully for a variety of aerodynamic shape optimization problems [16, 52, 53]. Exact analytic surface derivatives are readily available with this approach and both large global and local design modifications can be incorporated in a continuous, differentiable fashion. However, several issues limited the method’s appeal in a multidisciplinary environment. As with Aerosurf, modification of the internal structure remained an open issue. A subtle issue was identified when the method is applied to complex configurations. Figure 2.1 shows the Common Research Model (CRM) configuration and an expanded view of the wing-fuselage junction. Due to the density of control points required to accurately represent the underlying geometry, control points on the interface have very little freedom to move before overlap occurs, resulting in a folded surface. This effect renders the application of global variables such as chord or twist at the wing root very difficult with this approach. Finally, if B-spline control points are used as design variables providing local geometric control, the number of design parameters is directly tied to the number of control points required to accurately represent the surface. This limits the ability of an engineer to specify a simple optimization problem with a complex geometry with a
few design parameters.

Figure 2.1: Common Research Model surface B-splines. Expanded view at wing-body junction shows the closely spaced control points and possible overlap.

Free-Form Deformation (FFD) Approach: The final and ultimately selected method was the Free-Form Deformation Volume (FFD) approach. The benefits of this approach are described in the next section.

The first three techniques can be classified as constructive techniques. These methods build and maintain the exact representation of the geometry of interest. For constructive techniques, design variables parametrize the geometry directly. In contrast, the free-form deformation approach is a modification technique that parametrizes only the change in the geometry, not the actual geometry itself. Many of the benefits of the Free-Form deformation approach stem from this property. Finally, a survey paper by Samareh [54] outlines several additional shape parametrization techniques including basis-vector, domain element, discrete and analytical approaches and their applicability for MDO. However, none of these methods simultaneously satisfy the outlined requirements for high-fidelity aerostructural optimization.

2.1.1 Free-Form Deformation Volume Approach

The free-form deformation (FFD) approach is borrowed from soft object animation in the computer graphics field [55]. The method can be most easily visualized as embedding an object in a clear, flexible, rubber-like material. This rubber is usually a relatively simple geometry and any \( \mathbb{R}^3 \to \mathbb{R}^3 \) mapping may be used. Typically, tri-variate Bézier, B-spline or
NURBS volumes are utilized with cubic polynomial shape functions. Any discrete geometry may be embedded inside the volume by performing a Newton search to determine the \( u, v, w \) values mapping each point in parameter space to physical space. Provided this search is sufficiently converged, any initial geometry can be represented with numerical precision. Once embedded, high level modifications made to the FFD lattice can be used to indirectly modify the embedded objects.

Moving sets of control points together produces rigid-body motion and moving lines of points can produce geometric variables such as span, sweep and taper. A spatial curve rigidly attached to the control points can facilitate these bulk motions. Moving FFD control points individually produces smooth, high-order shape perturbations. Figure 2.2 shows a NACA 0012 airfoil embedded inside an FFD surface (a 2D analog of a 3D FFD volume). A sequence of control point movements performs a smooth transformation into an ONERA M6 airfoil.

![NACA 0012 Airfoil](a) NACA 0012 Airfoil

![ONERA M6 Airfoil](b) ONERA M6 Airfoil

Figure 2.2: Local shape control modification required to transform NACA 0012 airfoil into an ONERA M6 airfoil

Even for complex geometries, it is always possible to create a single FFD block that fully encloses the geometry. However, a multi-block approach is typically more practical for complete aircraft configurations. An example of a 26 block FFD for the CRM configuration is shown in Figure 2.3. This particular multiblock FFD configuration for the CRM partially solves the intersection issue described in the previous section. Large, linear-basis-function FFDs are used to surround the fuselage near the wing. As the wing-body interface changes, these large blocks can attenuate the perturbation smoothly over a large distance. This type
of modification does not re-intersect the wing and fuselage, as it would with the CAD-based approach, but has proven sufficient for the optimizations performed in Chapter 5.

![Figure 2.3: Multi-block FFD approach for complex configurations](image)

The FFD approach provides a consistent and transparent parametrization across disciplines. Specifically, the entire structural mesh can be embedded in the FFD volumes and modified in the same manner as the aerodynamic OML. Further benefits include a consistent way of manipulating geometric constraints, such as thickness constraints or volume constraints, examples of which are shown in Figure 5.4.

The FFD parametrization approach can also be extended to use nested volumes. Smaller, embedded volumes can be used to provide additional design freedom in certain critical areas of the geometry, such as around a wing-fuselage junction. Control surface deflections can also be approximated using nested FFDs. Figure 2.4 shows a $-7.5^\circ$ aileron deflection superimposed on the simple transonic wing RANS mesh (see Appendix A for details). This deflected geometry only approximates the discontinuous nature of a typical control surface, but since the topology remains unchanged, the mesh-deformation scheme described later in this chapter can automatically produce a valid CFD mesh for this geometry.

### 2.1.2 Structure Geometry Generation

The internal layout of most modern transport aircraft wingboxes consists of stiffened wing skins, ribs and spars. Manually generating this internal geometry for a given wing OML is time consuming and tedious. For conceptual and preliminary design purposes, a method to automatically generate wingbox structures is highly desirable.

**pyLayout** is a module used to generate simple structural meshes for wingbox structures. In the input for **pyLayout** is a surface description of the OML and a topological description of the structural layout. The structural description includes the number and position of
Chapter 2. Aerostructural Solution Methodology

Figure 2.4: Continuous aileron deformation using a nested FFD approach

the ribs, spars, and stringers, as well as parameters controlling the mesh spacing. With this information, pyLayout can quickly generates a complete structural mesh in a fully automatic fashion. An example of a structural mesh generated with pyLayout for the simple transonic wing (see Appendix A for specifications) is shown in Figure 2.5.

Figure 2.5: Complete structural mesh automatically generated with pyLayout

The primary advantage of pyLayout is the speed at which meshes can be generated: it typically requires less than one minute to generate a new mesh. Parametric changes in the structural layout such as changes in the rib and spar counts or changes in the mesh spacing can be quickly accommodated. pyLayout assumes a predetermined structured topological
domain that allows the element connectivities to be determined without resorting to more
general surface meshing techniques. Structural layouts containing more complex components
such as leading edge ribs or auxiliary spars cannot be generated with pyLayout.

For more complex structures, I wrote a separate, but related tool, pyLayoutGeo. Unlike
pyLayout, ribs, spars and stringers can be specified completely independently and inter-
sections are allowed between any two components. In particular, it is possible to generate
structural layouts for tapered wings with a constant stringer pitch, as is typically seen in most
contemporary wing boxes. The geometry for a structural model of the Common Research
Model including constant-pitch stiffeners is shown in Figure 2.6.

![Figure 2.6: Exploded view of pyLayoutGeo generated wingbox geometry for the Common Research Model](image)

2.2 Mesh Deformation

2.2.1 Review of Mesh-Deformation Schemes

Aerostructural analysis places significant demands on the mesh movement algorithm for
two main reasons. First, the algorithm must be sufficiently robust to accommodate large
design variable changes and structural deformations. Second, the algorithm must be suffi-
ciently fast to perform $O(10-100)$ movements during a single aerostructural analysis without
incurring excessive computational cost. Unfortunately, these goals are often mutually exclu-
sive: simple, fast algorithms typically lack the robustness, while robust algorithms incur a
significant computational cost penalty.

The goal of any mesh-deformation scheme is to automatically generate a new mesh suit-
able for desired computational simulation. For the three-dimensional external-flow aerody-
namics considered in this thesis, the mesh-deformation scheme must produce a new volume
mesh, \( X_V \), after a change to the surface mesh, \( X_S \). In general, any mesh movement procedure can be expressed as an operation or sequence of operations, transforming the new desired surface definition into the volume mesh. I denote this general mesh operator as \( \mathcal{M} \).

\[
X_V = \mathcal{M}(X_S)
\] (2.1)

There have been a wide variety of different approaches to the mesh movement problem in the literature. Mesh regeneration is simplest solution to mesh deformation. A new volume mesh is generated in the same manner as the mesh surrounding the original, un-deformed geometry. This technique can be applied to all mesh generation types, but is most applicable to fully automatic generation techniques. Optimization with unstructured mesh \cite{56} and Cartesian meshes \cite{57} have used this approach.

Mesh regeneration has several disadvantages. The method used to produce the initial mesh may be computationally expensive and require human intervention to produce an acceptable mesh. For unstructured meshes, a complete mesh regeneration does not guarantee that new mesh with the same number of cells or the same connectivity, and the design space may be discontinuous as a result. For multiblock structured meshes, obtaining the same number of cells is straightforward, but the procedure may still require human intervention or be computationally expensive. Regeneration approaches rarely, if ever, provide sensitivity information and this information must be computed with finite differencing, a very costly procedure if a large number of design parameters are used.

A second class of mesh-deformation algorithms are algebraic methods. Any method without implicit dependencies in the mesh-deformation operator \( \mathcal{M} \) may be considered algebraic. Algebraic methods are most often used with structured meshes and are typically very fast, but lack the robustness of more complex methods.

For structured meshes, Transfinite Interpolation (TFI) \cite{58} is often used for both multiblock generation and mesh deformation. In three dimensions, transfinite interpolation interpolates the interior of a generic rectangular volume given the definition of the six bounding faces. This approach handles the case where multiple surfaces of the same block are perturbed simultaneously.

A general TFI formulation can use any order of interpolation and blending functions. However, linear blending functions and linear interpolation are typically used. The expressions to compute the coordinates, \( X_V \), in the interior of a rectangular block of size \((i_{\text{max}}, j_{\text{max}}, k_{\text{max}})\) given the coordinates on each of the six boundary faces is given in Equation (2.2)
where \( \xi_i, \eta_j, \zeta_k \) are the arclength-normalized distances along each coordinate line. One advantage of TFI is that once all the corner, edge and face perturbations are known, each block regeneration is independent and thus regeneration is embarrassingly parallel. For mesh movement, an alternative form of Equation (2.2) is typically employed. The change in the coordinates, \( \Delta X \), is replaced by a linear spring analogy first described by Batina [59]. In this approach, each edge of the mesh is replaced by a linear spring with a stiffness inversely proportional to the length, possibly raised to some power. For small displacements on near isotropic meshes, this approach works well. The computational cost is fairly low and can be effectively solved with simple Jacobi iterations. This method guarantees two nodes on an edge will not cross, but does not ensure a node does not cross an edge resulting in a tangled mesh. Several authors have proposed
modifications to the original method that involve the use of torsional springs. The torsional spring method, originally proposed for triangular meshes in two dimensions, penalizes the angular change at nodes by considering the deformation of the surrounding elements. This method has been extended to three dimensions for tetrahedral meshes [60]. Robustness is greatly improved with this method, but at a much higher computational cost. Along similar lines, Truong et al. [61], used a quadrilateral finite element formulation using the cells on a 2D mesh. This approach proved to be particularly robust, but computationally expensive. A similar linear elasticity approach was described by Chen and Jadic [62] and employed a two dimensional surface boundary element formulation for three dimensional deformations.

2.2.2 A Parallel Hybrid Mesh-Deformation Scheme

Overview

The goal of the hybrid mesh-deformation scheme is to combine the speed of an algebraic approach with the mesh quality and robustness of an implicit method. An illustration of one of the shortcomings of an algebraic TFI scheme is shown in Figure 2.7.

![Figure 2.7: Algebraic warping scheme. The block boundaries are highlighted in bold.](image)

The fundamental difficulty is that algebraic TFI is a local method. Information from surface perturbations can only be propagated up to the size of the first block near the surface. As illustrated, if the desired motions is outside the bounds of the first block near the surface, a folded mesh results. Small blocks near the surface are quite common, especially in RANS meshes, and it is important that the hybrid scheme be able to handle this case.

The primary goal of combining the algebraic approach with the an implicit method is to ensure surface information is propagated to the entire volume mesh independent of the blocking topology. To accomplish this goal, an implicit mesh-deformation scheme is applied to a coarse approximation of the volume mesh. This coarse approximation accounts for large low-frequency perturbations and the algebraic approach attenuates small high-frequency
perturbations. The basic outline of the algorithm is as follows:

1. Select a subset of \( m \) nodes from each edge of \( N \) nodes. These nodes will form a coarse super mesh. See Figure 2.8b.

2. Apply a robust, implicit method to the super mesh. See Figure 2.8c.

3. Algebraically regenerate each block using either linear (Figure 2.8d) or cubic Bézier interpolation [63]. (Figure 2.8e)

4. Compute the difference between \( X_{SW} \) and \( X_S \) on the surface and attenuate this perturbation using TFI. See Figure 2.8f.

Hybrid Scheme Implementation Details

In principle, any number of methods can be used to deform the coarse mesh approximation. Since this mesh is relatively small, robust methods that are typically expensive for the entire mesh become attractive. I decided to use a linear elasticity-based method [61, 16] applied to the super nodes. The idea is to treat the mesh deformation as a finite element problem using three dimensional solid elements formed by the super nodes, \( Y_V \). Prescribed displacement conditions are used at all boundary nodes, \( Y_S \), and are determined as follows: nodal displacements on the aerodynamic surfaces are determined from the design variables and structural state vector. Nodes on the far-field boundaries remain fixed, while the nodes on symmetry conditions are constrained to remain in the plane of symmetry.

This hybrid scheme is also similar to the one described Martins et al. [64]. In fact, by choosing only the corners of the blocks \( (m = 2) \) to form the super mesh, the method described by Martins et al. is precisely recovered with the hybrid scheme described in this section. However, if we allow for more super nodes, the larger finite element problem can more easily capture large surface modifications due to either design variables or structural displacements, especially when large rotations are involved. Hicken and Zingg [16] describe a similar approach, using the control points of volumetric B-spline meshes to form the coarse approximation.

I investigated three different methods for solving the mesh finite-element problem: a linear approach, a stepped linear approach, and a nonlinear approach. For the linear approach, three dimensional tri-linear solid elements are used and the stiffness of each element, \( E \), is the inverse of the volume, i.e.

\[
E = \frac{1}{V} \quad (2.3)
\]
Figure 2.8: Hybrid mesh perturbation scheme

This anisotropic scaling causes smaller cells, typically located near the aircraft surface, to be stiffer than larger cells located near in the far-field regions. The stiffer cells near the body deform less helping to preserve the mesh quality [61, 16].

After discretization using the finite element method, a symmetric linear system of equations describes the displacement of the interior volume super mesh nodes:

\[ K_{uu} \Delta Y_V - K_{us} \Delta Y_S = 0 \]  

(2.4)

where the matrices \( K_{uu} \) and \( K_{us} \) are components of the mesh finite element stiffness matrix, \( K_M \). The unknown degrees of freedom are placed first, followed by the specified degrees of
freedom according to:

\[ K_M = \begin{bmatrix} K_{uu} & K_{us} \\ K_{su} & K_{ss} \end{bmatrix}. \] (2.5)

I use PETSc [65, 66, 67] to store the stiffness matrices in a row-distributed parallel format. To solve Equation (2.4), I use a sparse, parallel-direct method, SuperLU_DIST [68], which is accessed directly through the PETSc interface. The advantage of the linear approach and this solution technique is that the stiffness matrix is only factorized once at the beginning of the optimization. All subsequent solutions of Equation (2.4) are efficient back substitution operations.

For many applications the linear mesh-deformation method is sufficiently robust. However, the method can be made more robust by using a stepped approach. Instead of applying the full desired displacement in a single step (\(\Delta Y_S\)), the process is broken into a finite number of smaller steps [16]. After each step the (linear) stiffness matrix is formed using the current mesh and next displacement is applied. Complex movements are possible with this approach given a sufficient number of load increments. However, for reasons discussed in Chapter 3, this approach is not conducive to aerostructural sensitivity analysis.

The last method for determining \(Y_V\) is a nonlinear scheme. As in the stepped linear approach, large mesh deformations are possible. The non-linearity in the solution is introduced in two forms: Firstly, the element constituitive properties, namely the modulus of elasticity is now a function of the mesh solution since the volumes of the elements change. Secondly, a nonlinear strain-displacement relationship is used.

The solution of the nonlinear mesh-deformation equations is somewhat challenging due to Equation (2.3). It is strictly necessary that the volume of all elements is positive at every step towards the nonlinear solution. A negative volume would imply a negative stiffness resulting in an ill-formed finite element problem. I use a Newton-Raphson method for solving the nonlinear system. To determine a suitable starting point, without any inverted cells, I use the stepped linear approach described above. This mesh is typically very close to the nonlinear solution and provides an excellent starting point for the Newton iteration. To increase the computational performance, the nonlinear Jacobian and its SuperLU_DIST computed factorization is lagged and only recomputed when necessary. The cubic backtracking line-search described in the next section is used to increase the robustness of the nonlinear solution.

After the locations of the volume super nodes, \(Y_V\), have been determined, the remainder of the mesh-deformation algorithm is the same for each of the solution techniques. An
approximation to the full volume mesh is generated directly from the super nodes. I have used two different techniques to “fill in” the full volume mesh: a linear technique and a cubic Bézier interpolation technique. For both cases, parametric coordinates resulting from the parametrization of the initial mesh are used in the interpolation technique.

The first approach uses linear shape functions to approximate the interior nodes of each super-mesh element. While this technique is very fast, slope discontinuities can appear at the super nodes, which is not desirable.

The second approach, cubic Bézier interpolation overcomes the discontinuity issue, but at a slightly higher computational cost. The approach I have taken is to determine a tri-variate Bézier volume for each of the super mesh elements and this higher-order interpolation is used to determine the intermediate volume mesh $X_{SW}$. To uniquely determine the values of the 64 ($4 \times 4 \times 4$) Bézier volume coefficients 64 unique pieces of information are required. It is sufficient to use the values at the corners, $Y_V$, and the following 7 derivatives: $\frac{\partial Y_V}{\partial r}$, $\frac{\partial Y_V}{\partial s}$, $\frac{\partial Y_V}{\partial t}$, $\frac{\partial^2 Y_V}{\partial r \partial s}$, $\frac{\partial^2 Y_V}{\partial r \partial t}$, $\frac{\partial^2 Y_V}{\partial s \partial t}$ and $\frac{\partial^3 Y_V}{\partial r \partial s \partial t}$. Here, $r$, $s$, and $t$ are the parametric coordinates within the volume. The derivatives are computed using centered finite differencing with one sided derivative at the block boundaries. The first derivatives determine the control points along each of the edges, the second order mixed partial terms determine the four interior coefficients on each of the six faces and the third order mixed partial term determines the 8 interior control points. A schematic of a generic cubic volume is given in Figure 2.9.

![Figure 2.9: Generic volume approximated with tri-cubic Bézier volume.](image)

After the coefficients of the Bézier volumes are determined, recursive application of De Casteljau’s algorithm [63] is used to determine the intermediate mesh $X_{SW}$.

At the end of both filling techniques, $X_{SW}$ is a complete approximation to the desired volume mesh $X_V$. However, the surface components of $X_{SW}$ and new desired surface $X_S$ will not in general match exactly. To remedy this discrepancy, TFI (Equation (2.2)) is applied to
Figure 2.10: Solid mesh warping procedure

only the difference between $X_S$ and the component of $X_{SW}$ on the aircraft surface. By construction, $X_{SW}$ matches the desired surface $X_S$ precisely at the surface super nodes. When a sufficient number of super nodes are used to construct the coarse mesh approximation, the difference between $X_{SW}$ and $X_S$ is very small and is easily attenuated within a single block of the surface using TFI. The entire mesh-deformation procedure is represented in Figure 2.10.

**Mesh Deformation Example**

To demonstrate the robustness of the hybrid mesh-deformation scheme, I performed a large design variable modification to a simple transonic wing RANS mesh (see Appendix A for details), and then performed an aerostructural analysis. The mesh-deformation scheme described above can be used equally well for Euler meshes. However, I present this example using a RANS mesh as these are typically far more challenging for mesh-deformation schemes. RANS meshes have very thin cells in the boundary layer, which are typically $O\left(10^{-6}\right)$ of the chord length. It is critical the quality of these cells not degrade as a result of the mesh motion. Additionally, the resulting finite element mesh problem is much more poorly conditioned due to the very thin finite elements in the boundary layer region. The stiffness of these elements is many orders of magnitude larger than in the remainder of the mesh. Poorly conditioned problems of this kind favor the direct solution factorization method previously described.

The mesh in this example typical O-mesh surrounding the wing and renders the algebraic mesh-deformation scheme ineffective since even small perturbations exceed the full block dimension of the first off-wall blocks. Figure 2.11 shows the original FFD and the modification performed to produce a near vertical winglet. The surface solution is shown after performing an aerostructural analysis.

This type of modification is particularly challenging for mesh movement schemes, as there is a large rotation component to the displaced surface. Most linear schemes, including the one I have implemented, cannot tolerate large changes in orientation of the displaced surface. The nonlinear scheme, however, easily handles the deformation.

Next I compare the distribution of mesh quality of the original and warped meshes using the both linear and nonlinear solution techniques in Figure 2.12. The quality metric used
Chapter 2. Aerostructural Solution Methodology

Figure 2.11: Winglet design variable modification and aerostructural solution for mesh comparison purposes is the relative determinant:

\[ Q = \frac{\min(\det J_i)}{\max(\det J_i)}, \]

(2.6)

where the \( i^{th} \) metric Jacobian, \( J_i \), is evaluated at the corners of each element in a \( 2 \times 2 \times 2 \) stencil.

Figure 2.12: Quality comparison between original and warped meshes with both linear and nonlinear strategies. The number of elements are computed in quality bins of size 0.02.

The linear solution strategy fails to produce a valid mesh as it contains several hundred cells with negative determinants. The nonlinear solution has not only produced a valid mesh, but has nearly the same quality distribution as the original mesh. The minimum mesh quality of the deformed mesh is 0.34, which is only slightly less than the minimum
quality of the original mesh of 0.36.

Figure 2.13 shows the mesh obtained from the nonlinear solution strategy. Despite the very large design variable modification, the perturbed mesh has retained the desirable characteristics of the original mesh. Most importantly, the very thin cells in the boundary layer region have displaced rigidly with the geometry and have maintained their integrity.

![Figure 2.13: Aerostructural solution on mesh deformed using nonlinear warping technique](image)

**Mesh Deformation Performance**

As previously mentioned, the performance of the mesh-deformation scheme is extremely important for aerostructural analysis. Most analysis will require at least 10 mesh perturbations and potentially many more, especially for the Newton-Krylov solution approach. To ensure overall scalability of the aerostructural analysis, the mesh-deformation scheme must be parallel. Figure 2.14 shows the comparison in the time required to perform a single mesh perturbation for the simple transonic RANS mesh used in the previous section. The mesh finite element problem contains 14,231 unknown degrees of freedom. The number of processors is varied from 1 to 64.

Initially, the parallel performance of both the linear and nonlinear schemes are quite good, but degrades as 64 processors are approached. This performance is reflective of the scaling of SuperLU_DIST with a relatively small linear system. With 64 processors, each processor has on average only 222 unknowns. The communication costs out weight the computational costs in this case. The secondary warping procedures — the block filling and transfinite interpolations — are embarrassingly parallel and scale ideally, provided the distribution of blocks is uniform.
There are several factors that prevent ideal strong parallel scaling. First, the absolute
time required for a mesh perturbation is quite small; with 64 processors the nonlinear scheme
requires less than 5 seconds and the linear scheme just over 0.1 seconds. Secondly, if a larger
finite element mesh is required, a larger number of processors will be beneficial. Finally, the
most important aspect of the hybrid scheme is that the size of mesh finite-element problem is
not dependent on the number of cells in the CFD mesh. The number of super nodes required
to accurately approximate the desired design variable and displacement modifications is
strictly a function of the complexity of the mesh and its topology. Therefore, when larger
mesh sizes are used, the time required for the implicit part of the mesh-deformation scheme
is constant.

Furthermore, the cost of the nonlinear analysis is a function of the magnitude of the
surface displacement from the previous solution. In the case of aerostructural analysis and
design optimization, the magnitude of surface displacements between successive iterations
may be fairly small. In this case, the nonlinear solution converges in far less time. In the
case of the aerostructural analysis performed in Figure 2.13 using 64 processors, 11 mesh
movements were completed in a total time of 17.1 seconds, or an average of 1.55 seconds per
perturbation. This average is only one third of the time shown in Figure 2.14.
2.3 Aerodynamic Analysis

The aerodynamic analysis used in this thesis is SUmb, which solves the Euler, laminar Navier–Stokes, or Reynolds averaged Navier–Stokes (RANS) equations in either steady, unsteady or time-spectral temporal modes [69]. SUmb was originally written for the analysis of large-scale turbo-machinery flows [69] but has also successfully been employed for external flow applications [29].

SUmb uses structured, body-fitted, multi-block meshes resulting in the fluid domain being discretized into many hexahedral control volumes. The finite-volume method is employed with the average of each of the unknown values stored at the center of each cell.

The majority of this thesis will be concerned with Euler equations, including the optimizations in Chapter 5. However, aerostructural solutions can easily be obtained with the RANS equations; see Figure 2.13 for example.

At each cell, there are five aerodynamic unknowns, \( w \), given as:

\[
w = [\rho \; \rho u_1 \; \rho u_2 \; \rho u_3 \; \rho E]^T
\]  
(2.7)

where \( \rho \) is the density, \((u_1, u_2, u_3) = u \) are the three Cartesian components of velocity and \( E \) is the total energy. The Euler flux vector in each of the three spatial direction, \( l = \{1, 2, 3\} \) is defined as

\[
F_l = \begin{bmatrix}
\rho u_l \\
\rho u_1 u_l + p \delta_{1l} \\
\rho u_2 u_l + p \delta_{2l} \\
\rho u_3 u_l + p \delta_{3l} \\
\rho H u_l,
\end{bmatrix}
\]  
(2.8)

where \( \delta_{ij} \) is the Kronecker delta function. The pressure, \( p \), is related to the total energy by the ideal gas law equation of state,

\[
p = (\gamma - 1) \rho \left( E - \frac{u^2}{2} \right)
\]  
(2.9)

where \( \gamma \) is the (constant) ratio of specific heats. The total enthalpy is related to the total energy according to:

\[
H = E + \frac{p}{\rho} = \frac{c^2}{\gamma - 1} + \frac{u^2}{2}
\]  
(2.10)

where \( c = \sqrt{\gamma P/\rho} \) is the speed of sound.
For each hexahedral cell the inviscid flux component of the aerodynamic residual, $A_{F_{i,j,k}}$, is computed according to:

$$A_{F_{i,j,k}} = F_{i+1/2,j,k}S_{i+1/2,j,k} - F_{i-1/2,j,k}S_{i-1/2,j,k}$$

$$+ F_{i,j+1/2,k}S_{i,j+1/2,k} - F_{i,j-1/2,k}S_{i,j-1/2,k}$$

$$+ F_{i,j,k+1/2}S_{i,j,k+1/2} - F_{i,j,k-1/2}S_{i,j,k-1/2}$$

where $F_{i+1/2,j,k} = \frac{1}{2}(F_{i+1,j,k} + F_{i,j,k})$ is the average of the flux values on each side of the face $i + 1/2$ and $S_{i+1/2,j,k}$ is the normal of the face scaled by the area of the face. This central difference approximation is not numerically stable and in practice a small amount of dissipation must be added to prevent oscillations in the solution. I use the Jameson–Schmidt–Turkel (JST) scalar dissipation scheme as it is computationally inexpensive and relatively robust, making it particularly suitable for optimization. The governing equations, including the additional added dissipation $D_{i,j,k}$, are written in semi-discrete form as:

$$\frac{d}{dt}(V_{i,j,k}w_{i,j,k}) + A_{F_{i,j,k}} - D_{i,j,k} = 0$$

where $V_{i,j,k}$ is the volume of given cell. For steady-state analysis, the rate of change of the conserved quantities is zero and the resulting residual equation is:

$$A_{F_{i,j,k}} - D_{i,j,k} = A(w) = 0$$

where $A$ are the aerodynamic residuals, including the fluxes and dissipation terms.

The default method in SUmb for solving Equation (2.15) is an explicit 5-stage Runge–Kutta time integrator. Multi–grid, implicit residual smoothing, and local time stepping are all employed to accelerate steady-state convergence. I have found, however, that the convergence rate typically slows as the flow approaches the steady state solution. This characteristic behaviour tends to be worse when solving a flow–field in the transonic regime. Encouraged by the performance of the Newton–Krylov algorithms developed by Hicken and Zingg [16, 70], I decided to add this capability to SUmb.

Newton’s method seeks an update, $\Delta w$ to the state variables, $w$, by solving the linear system:

$$\left(\frac{\partial A}{\partial w}\right)^{(n)} \Delta w^{(n)} = -A\left(w^{(n)}\right).$$

(2.16)
Updates to $w$ continue until the $\ell^2$ norm of the residual has dropped below some specified tolerance. The solution of the linear system in Equation (2.16) is obtained using PETSc [67, 65] and thus there are variety of options available. Without performing a rigorous survey of available methods, I choose to use GMRES [71] as it performs well for a wide variety of non-symmetric problems. The biggest advantage of iterative Krylov methods for the solution of Equation (2.16) is that they require only matrix–vector products and the explicit formation of the Jacobian matrix is not necessary. For this reason, it is possible to implement a matrix–vector product as a matrix-free operator from the finite difference:

$$\frac{\partial A}{\partial w} v \approx \frac{A(w + h v) - A(w)}{\epsilon}.$$  \hfill (2.17)

where $h$ is a small perturbation computed according to [72]

$$h = \begin{cases} 
  e_{rel} \cdot w^T v / \|v\|_2^2 & \text{if } |w^T v| > u_{min} \|v\|_1 \\
  e_{rel} \cdot u_{min} \cdot \text{sign}(w^T v) \cdot \|v\|_1 / \|v\|_2^2 & \text{otherwise}
\end{cases}.$$  \hfill (2.18)

Assuming the evaluation of $A$ is accurate to machine precision $\epsilon_{\text{Mach}}$, the relative error is set to $e_{rel} = \sqrt{\epsilon_{\text{Mach}}}$ and $u_{min}$ is set to $1 \times 10^{-6}$.

It is well known that the convergence of Krylov methods can be greatly accelerated with preconditioning. To form the preconditioning matrix, I use a first-order approximation to the Jacobian. Only 2nd order dissipation is used and the fourth and second order dissipation constants, $\kappa_4$ and $\kappa_2$ respectively, are combined with a lumping parameter, $\sigma$ [73], according to

$$\hat{\kappa}_2 = \kappa_2 + \sigma \kappa_4.$$  \hfill (2.19)

For three-dimensional Euler flows, typical values of $\sigma$ are $4 \leq \sigma \leq 6$. [70]. I use the additive–Schwartz Method (ASM) preconditioner [74], with a small level of overlap (1–2) and an Incomplete LU Factorization method (ILU) with one level of fill, and Reverse Cuthill–McKee (RCM) [75] reordering on each of the sub–domains. Unlike Gropp’s numerical experiments [76], I have found that a small overlap between domains for the additive–Schwartz preconditioner does decrease the overall solution time. Changes in interconnect speeds and processor architectures over the last decade have made the additional communication and computational cost per iteration competitive with reducing the overall number of sub–iterations.

Additionally, since successive linear solutions of Equation (2.16) are closely related, the
factored form of the preconditioner can be reused between Newton iterations. In my experience, for most relatively well-behaved problems, including transonic flow problems, a single assembly and factorization of the preconditioner is usually sufficient. In practice, I lag the preconditioner by 15 to 20 iterations, which allows for a timely preconditioner update if the Newton algorithm stalls.

Newton algorithms are not globally convergent and it is thus necessary to ensure the starting iterate $w_0$ is sufficiently close to the solution to allow the Newton method to converge. There are a number of different techniques that can be used to obtain the initial iterate. The most common approach is implicit pseudo-time stepping [76, 16]. In this approach, a local time stepping parameter is added to limit the Newton update during initial iterations. As the solution converges, the time step is allowed to approach infinity and full Newton steps are recovered. Alternatively, parameter continuation approaches can be used, including boundary condition continuation and dissipation based continuation [77].

In the Newton–Krylov version of S Umb, which we will call S Umb–NK, an alternative approach is used. The existing explicit multi-grid Runge–Kutta algorithm, with a full multi-grid start-up algorithm, is a very effective approach for producing the initial iterate. The Newton–Krylov algorithm is started after the $\ell^2$ norm of the residual has dropped below a specified tolerance, usually between 1 and 2 orders of magnitude. Often, the iterate obtained at the end of the full-multi-grid start-up procedure is sufficiently converged that the Newton approach can be started immediately without performing multi-grid iterations on the fine mesh.

To further improve the robustness of the Newton algorithm, I have experimented with two different line-search strategies: a cubic backtracking [78] approach and a non-monotone [79] approach. For the cubic backtracking line search, if the function value at the newly computed Newton step, $f(w_{i-1} + \Delta w)$ (where $f = 1/2 \|A\|_2^2$) does not result in a sufficient decrease in the function, the line-search procedure determines a value $\alpha \in (0, 1]$ such that $f(w_{i-1} + \alpha \Delta w)$ satisfies the sufficient decrease condition [80]. The line-search is accelerated by forming a polynomial approximation (up to a cubic degree) to the function along the search direction. The minimum of this polynomial approximation is used to determine the next point to evaluate if the sufficient decrease condition is not met. This approach guarantees that the function will never increase during the solution procedure. This is typically a desirable property for many nonlinear systems. However, for our problems, it is often advantageous to allow the residual to increase for a step or two before requiring the residual to continue to decrease. As the name suggests, the non-monotone approach allows the residual to increase
temporarily within user specified limits. For this technique, the key change is that the
solution is only required to satisfy sufficient decrease with respect to the last $M$ function
evaluations. Typically I use $M = 10$ which has proved satisfactory.

A simple example demonstrates the advantage of the non-monotone approach. Figure 2.15 shows the solution process for a coarse version of the Euler simple transonic wing (see Appendix A). Allowing the residual to increase for a single iteration, allows the remainder of the solution to proceed faster than the cubic strategy. In this particular case, the non-monotone search never limits the Newton step and the same sequence of steps would be recovered if no line-search strategy was used.

![Figure 2.15: Comparison of cubic backtracking line-search and non-monotone line-search](image)

I now present an example comparing the performance of the SUmb Newton–Krylov method with the original Runge–Kutta method. The convergence of both methods for the simple transonic wing Euler mesh is given in Figure 2.16. See Appendix A for further details. The simulation is run with 16 processors on Scinet [81]. The Newton–Krylov algorithm converges the $\ell^2$ norm of the residual 8 orders of magnitude below its free stream value in 211 seconds. The Runge–Kutta method requires 624 seconds to reach the same convergence level, approximately three times longer.
2.4 Skin Friction Estimate

Since the Euler equations are used to model flow over the aircraft, the skin friction drag contribution cannot be modelled. However, for the aerostructural optimizations presented in Chapter 5, an estimate of the total aircraft drag is required. If the skin friction drag is not included, the drag penalty associated with increasing wing area and decreasing wing chords (resulting from lower Reynolds numbers) would not be captured, resulting in unrealistic planforms. A approximation of the skin friction drag can be estimated using a flat plate turbulent skin friction estimate with a form factor correction. The correction accounts for differences in the development of the boundary layer, the higher velocities outside the boundary layer and the pressure drag resulting from a non-zero thickness [82]. I use the van Driest II Method [83] to estimate the turbulent skin friction coefficient, $C_F$ (see Appendix B for more details). For wing-like bodies, the following form-factor correction [84],

$$FF = 1.0 + 2.7 \left( \frac{t}{c} \right) + 100 \left( \frac{t}{c} \right)^4$$

(2.20)
is used, where $t/c$ is the thickness to chord ratio of the wing section being considered. For fuselage like bodies, the form-factor correction is

$$ FF = 1.0 + 1.5 \left( \frac{d}{l} \right)^{1.5} + 50 \left( \frac{d}{l} \right)^{3}, $$

(2.21)

where $d/l$ is the ratio of diameter to length. The contribution of a given component to the drag coefficient is then

$$ C_D = C_F \frac{A_{\text{wet}}}{A_{\text{ref}}} FF $$

(2.22)

where $A_{\text{wet}}$ is the wetted surface area of the component and $A_{\text{ref}}$ is the reference wing area.

In the implementation I developed, 3D surface geometries shown in Figure 2.17 are used. This geometry representation is compatible with the FFD volume technique described earlier. The skin friction drag for the wing and tail are computed in a strip-wise fashion that accounts for local changes in the Reynolds number due to chord modifications, and thickness-to-chord changes due to shape changes. Smooth Kreisselmeier–Steinhauser (KS) functions [85] are used to estimate the $t/c$ ratio from the discrete surface data to ensure smooth, continuous derivatives.

Figure 2.17: The three separate meshes used for the skin friction drag computation. Representative strips are shown for the wing and tail grids.

2.5 Structural Analysis

The structural solver used in this thesis is the Toolkit for the Analysis of Composite Structures (TACS) [86]. This is a parallel finite-element solver that includes both static linear and geometrically nonlinear analysis capabilities. However, only linear analysis is con-
Chapter 2. Aerostructural Solution Methodology

TACS includes and adjoint solver that is able to handle the structural design variables, which in our case are the thicknesses of the structural members. Parallelism is achieved within TACS by using an element-based partitioning of the finite-element mesh. This partitioning is used to parallelize the factorization of the stiffness matrix, the computation and assembly of the stiffness matrix and structural residuals, and the computation of the functions of interest and their derivatives. Typically, the factorization of the stiffness matrix is the most costly operation. To parallelize the matrix factorization and back solutions, TACS uses a Schur-complement-based parallel direct solver. In this technique, each processor independently computes the local contribution to a reduced linear system that is formed from the all unknowns on the domain interface. This reduced problem is the global Schur complement. In TACS, the local contributions to the global Schur complement are computed using a block-based parallel factorization. After the global Schur complement is computed, it is factored in parallel using a sparse block-cyclic algorithm that achieves excellent parallel performance. This direct method enables us to solve poorly conditioned thin-shell structural problems, with condition numbers $O(10^9)$ in an efficient manner. For general nonlinear analysis the structural discipline residuals are

$$S(u) = 0 \quad (2.23)$$

where $u$ is the vector of structural displacements. For linear analysis this equation can be written as $S(u) = Ku - F$, where $K$ is the linear stiffness matrix and $F$ is the load vector.

2.6 Load and Displacement Transfer

To perform aerostructural analysis, we must couple the two disciplines to form a single analysis capable of determining the shape and aerodynamic characteristics of a lifting surface for a given flight condition. Specifically, we must determine how the loads computed by the aerodynamic analysis are transferred to the structural analysis (load transfer) and how the displacements computed by the structural analysis deform the wetted aerodynamic surface (displacement transfer).

The load and displacement transfer scheme follows the work of Brown [30], later employed by Martins et al. [87] and Kennedy [86]. In this approach, rigid links are used to extrapolate the displacements from the structural surface to the CFD surface, as shown in Figure 2.18. These rigid links are constructed between the aerodynamic surface mesh points and the points on the structural model lying closest to this set of points. The consistent force
vector is determined by employing the method of virtual work, ensuring that the force transfer is conservative. The integration of the forces occurs on the aerodynamic mesh and is transmitted back through the rigid links to the structure. The two primary advantages of this scheme are that it is consistent and conservative by construction, and that it may be used to transfer loads and displacements between aerodynamic and structural meshes that are not coincident.

\[ X_S = X_J + u_A \]
\[ = X_J + u_t + u_r \times r. \]

where \( r \) is the vector that connects a point on the aerodynamic surface to the closest location on the structural mesh, \( u_t \) is the translational component of the displacement, and \( u_r \) is the small-angle approximation of the rotations in the global reference frame, as shown in Figure 2.18. We can compactly represent the displacement transfer from \( u \) to \( u_A \) as

\[ u_A = Tu = \left( \frac{\partial u_A}{\partial u} \right) u \]

where \( T \) is the generalized transfer matrix. To transfer loads from the aerodynamic discipline
to the structural discipline, we employ the transpose operation,

\[ F = T^T F_A = \left( \frac{\partial F}{\partial F_A} \right) F_A \]  

(2.26)

where \( F_A \) are the forces on the CFD nodes. More details on these transfers can be found in Brown [30] and Martins et al. [87].

### 2.7 Aerostructural Analysis

I formulate the aerostructural analysis problem using a two-field formulation. The aerodynamic analysis is the first field and the structural analysis is the second. The governing equations of both disciplines are functions of the fluid states, \( w \), structural states, \( u \), and design parameters \( x \). The latter is a vector consisting of global variables and local variables. Global variables affect the two disciplines directly, while local variables affect only a single discipline in a direct manner. Geometric variables that change the aircraft wetted surface or Outer Mold Liner (OML) are global variables. For example, an airfoil thickness variable will not only change the aerodynamic shape but also the height of the ribs and spars in the internal structure. Local variables include the angle of attack, which directly affects only the aerodynamics, and the thicknesses of the structural members, which affects only the structures. Combining the residual equations from the aerodynamic and structural disciplines, previously introduced in Equations (2.15) and (2.23), the combined residual of the multidisciplinary system is

\[ \mathcal{R} = \begin{bmatrix} \mathcal{A}(w, u; x) \\ \mathcal{S}(u, w; x) \end{bmatrix} = 0. \]  

(2.27)

The aerostructural analysis consists of finding a solution, \((w, u)\), that satisfies the coupled residual equations.

#### 2.7.1 Nonlinear Block Gauss–Seidel Method (NLBGS)

The traditional process for solving the coupled aerostructural equations (2.27) is the nonlinear block Gauss–Seidel (NLBGS) method [87, 35]. In this approach, the aerodynamic analysis is first partially converged and the aerodynamic forces are evaluated. The forces are then transferred to the structural analysis and the corresponding displacements are computed. Finally, the displacements are transferred back to the aerodynamic analysis, the mesh is deformed, a new CFD solution is found, and this iterative loop continues until the coupled convergence criterion is met. The NLBGS procedure is listed in Algorithm 1 shown
Three tolerances are defined for the coupled analysis. The relative tolerances, $\epsilon_A$, $\epsilon_S$, and $\epsilon_{AS}$, are the tolerances required by the aerodynamic and structural disciplines, respectively, for each NLBGS iteration. The aerodynamic solver tolerance is typically $O\left(10^{-1}\right)$, while $\epsilon_S$ is typically $O\left(10^{-3}\right)$ or smaller. The third tolerance, $\epsilon_{AS}$, is the aerostructural
solution tolerance and represents the feasibility of the interdisciplinary coupling. Typical values for $\epsilon_{AS}$ range from $1 \times 10^{-3}$ for an engineering solution accurate to three decimal places to $1 \times 10^{-6}$, which is the typical value used when performing design optimization. The CFD convergence is evaluated at the beginning of the $k^{th}$ iteration using the structural displacements and aerodynamic states from the previous iteration. The structural convergence is evaluated at the beginning of the structural solution using the previous structural states with the current force vector. Aitken acceleration [88] (Line 22 of Algorithm 1) is employed to dynamically choose the under-relaxation factor to accelerate convergence. One advantage of this method is that each disciplinary solver can be used without modification. For tightly coupled aerostructural problems with large displacements, however, this method may converge slowly or not at all, as observed by Barcelos et al. [40].

### 2.7.2 Coupled Newton–Krylov Method (CNK)

The second approach I use for the aerostructural solution is a fully coupled Newton–Krylov (CNK) method. This implementation closely follows previous work on low fidelity multidisciplinary systems [86]. Algorithm 2 lists the pseudocode for evaluating the coupled nonlinear residual, $R$.

#### Algorithm 2 Coupled nonlinear residual computation

```plaintext
1: function $R(w,u)$
2: $X_S \leftarrow Tu + X_J$  \hspace{1cm} \triangleright Transfer displacements
3: $X_V \leftarrow M(X_S)$  \hspace{1cm} \triangleright Deform volume mesh to match surface
4: $A \leftarrow A(w,X_V)$  \hspace{1cm} \triangleright Evaluate CFD residuals
5: $F_A \leftarrow F_A(w,X_S)$  \hspace{1cm} \triangleright Evaluate aerodynamics forces
6: $F \leftarrow TT F_A$  \hspace{1cm} \triangleright Transfer forces
7: $S \leftarrow S(u,F)$  \hspace{1cm} \triangleright Evaluate CSM residuals
8: $R \leftarrow (A,S)$  \hspace{1cm} \triangleright Combine residuals
9: return $R$
10: end function
```

The procedure is similar to one iteration of the NLBGS method, except that instead of computing an approximate solution update to the state variables, only the residuals are evaluated. The communication and mesh-deformation costs of this coupled nonlinear residual evaluation are comparable to the cost of a single NLBGS iteration. Provided the communication costs are low and the mesh-deformation algorithm is efficient, the coupled residual evaluation is inexpensive. I use an inexact Newton–Krylov approach to solve the coupled
Chapter 2. Aerostructural Solution Methodology

equation 2.27 by computing the approximate Newton update,

\[
\begin{bmatrix}
\frac{\partial A}{\partial w} & \frac{\partial A}{\partial u} \\
\frac{\partial S}{\partial w} & \frac{\partial S}{\partial u}
\end{bmatrix}
\begin{bmatrix}
\Delta w \\
\Delta u
\end{bmatrix}
= -
\begin{bmatrix}
A(w) \\
S(u)
\end{bmatrix}.
\] (2.28)

After this update is computed, the state variables \(w\) and \(u\) are updated with \(\Delta w\) and \(\Delta u\), respectively.

To reduce the memory requirement, I use a matrix-free method, similar to that used for the aerodynamic solver. The Jacobian-vector products required by the Krylov method are approximated with finite-differences. In theory, with this approach, only residual evaluations are required to solve the coupled problem. However, Krylov methods require effective preconditioning for acceptable performance, especially for very large systems with millions of degrees of freedom. I have implemented a block-Jacobi preconditioner that reuses the preconditioners of the original discipline solvers. This has two advantages: 1) the application of the preconditioners from each discipline can be applied in parallel because this approach eliminates contributions from the off-diagonal terms, which are never explicitly stored, and 2) since both of our discipline codes use implicit methods for their own solutions, this preconditioner simply reuses the same linear solution method from the original solvers. Since these preconditioners vary from iteration to iteration a flexible variant of a Krylov method is required to solve the coupled Newton update (2.28). I choose to use FGMRES [71], which has been shown to work well for a wide range of large unsymmetric systems. The Eisenstat–Walker method is used to adaptively select the forcing tolerance of the linear system solution to prevent over solving during the initial iterations [89].

As with most Newton-type methods, this approach may not converge for certain initial conditions. I have not yet investigated the start up behavior for the coupled Newton–Krylov method in detail. My approach is to run a few relaxed NLBGS iterations (Algorithm 1) before switching to the Newton–Krylov algorithm. This method has worked well for the results in proceeding sections, but it is possible that other methods may be required for other cases, especially for even more structurally flexible designs. The main advantage of the Newton–Krylov method is the elimination of the under-relaxation parameter, which leads to faster convergence, especially for problems with flexible structures.

2.7.3 Solution Methods Comparison

In this section, the two solution strategies for the aerostructural system are compared. The comparison is carried out using a 2 110 096 cell CFD mesh with 64 processors and a
324,180 DOF structural mesh with 8 processors. This is the same as the level-2 discretization listed in Table 3.4. I consider two load cases: a 1 g cruise condition with only moderate elastic deformation and a 2.5 g symmetric maneuver condition with significantly more deflection. I consider several variations of the NLGBS and CNK methods. First, four fixed under-relaxation parameters are considered: \( \theta = 0.25, 0.5, 0.75, \) and 1.0 (the latter corresponding to no under-relaxation). For these analyses, \( \theta^{(k)} = \theta^{(0)} \) for all iterations. An additional analysis with Aitken acceleration (\( \theta^{(0)} = 0.5 \)) is also performed. For all the NLBGS analyses, \( \epsilon_A = 10^{-1} \) and \( \epsilon_S = 1 \times 10^{-3} \). However, since the structural analysis uses a direct factorization method, a lower error than \( \epsilon_S \) is usually achieved.

For the NLBGS analysis, the aerodynamic Newton–Krylov solver uses GMRES(75). This system is preconditioned with Additive Schwartz [74] (overlap 1) and ILU(2) on each of the subdomains. For the Newton–Krylov algorithm, three non-Aitken accelerated relaxed Gauss–Seidel start-up iterations are performed for the 1 g case, and five start-up iterations are performed for the 2.5 g case. Three variants of the algorithm are evaluated, each with a different subspace size for the aerodynamic part of the coupled block-Jacobi preconditioner. To ensure a fair comparison using constant memory, the size of the FGMRES outer subspace is taken as \( m = \lfloor (75 - s)/2 \rfloor \), (where \( s \) is the number of GMRES subspace vectors for the aerodynamic preconditioner) which yields similar memory usage for the CFD processors (See Chapter 3, Section 3.7.3 for more details on memory usage). While the structural processors have additional memory requirements, these requirements are small compared to the memory required for the stiffness matrix and associated factorization, so they are acceptable.

Table 2.1 lists the computational times for each of the test combinations. The \( N_{iter} \) column is the number of iterations required for convergence, which has a slightly different meaning depending on the solution method. For the NLBGS methods, this is the number of iterations required for convergence. For the CNK methods, this is the total number of nonlinear residual evaluations. Since the cost of the preconditioned nonlinear residual evaluations varies with the strength of the preconditioner, i.e., the value of \( s \), the number of iterations is not directly comparable between the two classes of methods. The “—” symbol signifies that a given parameter is not relevant for the particular case.

A number of trends can be observed in these results. For both load cases, fixed under-relaxation parameters with values of 0.25 or 1.0 consistently performed poorly, while values of 0.5 and 0.75 performed reasonably well. The lowest under-relaxation of 0.25 is very robust since there are no over-shoots or under-shoots, but it takes a large number of iterations to converge. Conversely, the solution without under-relaxation (\( \theta = 1.0 \)) case consistently over-
Table 2.1: Aerostructural solution comparison for nonlinear block Gauss–Seidel (NLBGS) with various under-relaxation strategies and coupled Newton–Krylov (CNK) Solver

<table>
<thead>
<tr>
<th>Solver</th>
<th>$\theta^{(0)}$</th>
<th>Aitken accel.</th>
<th>Inner size (s)</th>
<th>Outer size (m)</th>
<th>$N_{iter}$</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>CFD Mesh CSM Total</td>
</tr>
<tr>
<td>1.0 g Load factor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NLBGS</td>
<td>0.25</td>
<td>N</td>
<td>—</td>
<td>—</td>
<td>49</td>
<td>384.4     6.2 23.0</td>
</tr>
<tr>
<td>NLBGS</td>
<td>0.50</td>
<td>N</td>
<td>—</td>
<td>—</td>
<td>22</td>
<td>218.7     2.8 20.3</td>
</tr>
<tr>
<td>NLBGS</td>
<td>0.75</td>
<td>N</td>
<td>—</td>
<td>—</td>
<td>16</td>
<td>190.3     2.0 19.7</td>
</tr>
<tr>
<td>NLBGS</td>
<td>1.00</td>
<td>N</td>
<td>—</td>
<td>—</td>
<td>22</td>
<td>216.8     2.8 21.8</td>
</tr>
<tr>
<td>NLBGS</td>
<td>0.50</td>
<td>Y</td>
<td>—</td>
<td>—</td>
<td>18</td>
<td>200.3     2.3 19.9</td>
</tr>
<tr>
<td>CNK</td>
<td>0.50</td>
<td>N</td>
<td>5</td>
<td>35</td>
<td>202</td>
<td>207.2     25.3 32.5</td>
</tr>
<tr>
<td>CNK</td>
<td>0.50</td>
<td>N</td>
<td>10</td>
<td>32</td>
<td>79</td>
<td>162.4     9.8 24.1</td>
</tr>
<tr>
<td>CNK</td>
<td>0.50</td>
<td>N</td>
<td>20</td>
<td>27</td>
<td>56</td>
<td>183.6     3.9 22.2</td>
</tr>
<tr>
<td>2.5 g Load factor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NLBGS</td>
<td>0.25</td>
<td>N</td>
<td>—</td>
<td>—</td>
<td>52</td>
<td>349.9     6.6 23.2</td>
</tr>
<tr>
<td>NLBGS</td>
<td>0.50</td>
<td>N</td>
<td>—</td>
<td>—</td>
<td>28</td>
<td>254.9     3.5 20.9</td>
</tr>
<tr>
<td>NLBGS</td>
<td>0.75</td>
<td>N</td>
<td>—</td>
<td>—</td>
<td>25</td>
<td>272.2     3.1 20.6</td>
</tr>
<tr>
<td>NLBGS</td>
<td>1.00</td>
<td>N</td>
<td>—</td>
<td>—</td>
<td>52</td>
<td>363.0     6.6 23.2</td>
</tr>
<tr>
<td>NLBGS</td>
<td>0.50</td>
<td>Y</td>
<td>—</td>
<td>—</td>
<td>25</td>
<td>263.1     3.1 20.6</td>
</tr>
<tr>
<td>CNK</td>
<td>0.50</td>
<td>N</td>
<td>5</td>
<td>35</td>
<td>225</td>
<td>234.8     28.2 34.8</td>
</tr>
<tr>
<td>CNK</td>
<td>0.50</td>
<td>N</td>
<td>10</td>
<td>32</td>
<td>118</td>
<td>214.6     14.7 26.8</td>
</tr>
<tr>
<td>CNK</td>
<td>0.50</td>
<td>N</td>
<td>20</td>
<td>27</td>
<td>55</td>
<td>193.9     6.8 22.3</td>
</tr>
</tbody>
</table>

shoots and under-shoots the actual structural displacement, especially for the 2.5 g condition, leading to a large number of iterations. The Aitken acceleration reduces the required number of Gauss–Seidel iterations considerably, resulting in the second-best Gauss–Seidel solution time for each of the two load cases. The advantage of the Aitken acceleration is that the optimal value of the $\theta$ parameter, which is problem dependent, does not need to be known beforehand. For example, the optimal $\theta$ for the 1 g condition is 0.75, while for the 2.5 g load case it is 0.5. Given the ease of implementation and increase in performance, and assuming that we do not know the optimal under-relaxation parameter in advance, the Aitken-accelerated form of the Gauss–Seidel method is the recommended one among the segregated analysis methods.

The CNK method generally performs better than the NLBGS methods, and the lowest solution times for each of the two flight conditions are achieved by this method. The effect of the strength of the inner preconditioner on the number of inner iterations is evident...
in Table 2.1. As expected, $s = 5$ requires the most outer iterations, with each iteration having the lowest computational cost. For the largest number of outer iterations ($m - 35$), the number of iterations increases considerably, exceeding 200. Since our scheme is very efficient, a small inner subspace is still viable. However, better solution times are obtained using a slighter stronger preconditioner. The solution times with $s = 10$ and $s = 20$ are similar and suggest that for computational problems with this discretization level, 10–20 is a reasonable range for this parameter. For the 1 g load case, the best CNK solution time is 9% faster than the best NLBGS time, and it is 22% faster for the 2.5 g load case. Not surprisingly, the advantage of a monolithic strategy is more pronounced in the more strongly coupled 2.5 g load condition.

2.8 Conclusions

In this chapter I have presented the methods I use for aerostructural analysis. I have shown the multiblock FFD approach is well suited for aerostructural optimization. Parametrizing the changes to the geometry rather than the geometry itself results in a consistent parametrization across disciplines. A description the CFD solver, SUm, was given as well as the improvements I contributed including the Newton–Krylov solution technique and the Runge–Kutta startup procedure. The mesh-deformation procedure, uses a hybrid algebraic linear-elasticity approach using either a linear or nonlinear approach. I have shown both methods are computationally efficient and can produce high quality meshes even after undergoing extreme design variable or displacement modifications. The structural solver, TACS, and the coupled procedure were both described. Finally, I presented two methods for solving coupled high-fidelity aerostructural systems: the nonlinear block Gauss–Seidel method (NLBGS) and the coupled Newton–Krylov method (CNK). For the typical size of problems I am interested in for optimization, I showed there is a 10%–20% reduction in wall time for the CNK method and an increase in robustness for highly flexible structures.
Chapter 3

Gradient Computation Methodology

This chapter will describe the sensitivity analysis for each of the components of the aerostructural optimization framework described in the previous chapter.

The main goal of this thesis is to advance the state-of-the-art of high-fidelity aerostructural optimization. The choice of optimization algorithm plays a significant role in determining the overall effectiveness of the design optimization. Generally, optimization algorithms are classified into two categories: gradient based and non-gradient based. Non-gradient based algorithms are often inspired from biological interactions such as genetic algorithms (GA) [90], particle swarm optimization (PSO) [91, 92, 93], ant colony optimization [94], and invasive weed optimization (IWO) [95]. Binitha and Sathya classify approximately 15 other bio-inspired optimization algorithms [96]. Other non-gradient based methods include Nelder–Mead’s algorithm [97], simulated annealing [98] and pattern searches [99].

With no required gradient information, it may appear that non-gradient based methods may be suitable for high-fidelity optimizations given the difficulty and cost of computing gradients. However, a serious drawback with non-gradient based methods is that the computational cost scales exponentially the number of design variables, making high-fidelity optimization extremely expensive.

A simple, scalable analytic function effectively demonstrates this behaviour. Consider the unconstrained minimization of the multidimensional Rosenbrock function,

\[ I(x) = \sum_{i=1}^{N-1} [(1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2], \quad (3.1) \]

where \( N \) is the number of design variables. Four different optimizers are considered: SNOPT [100], a gradient based optimization method, is run with analytic derivatives (SNOPT) and with
finite difference derivatives (SNOPT-FD), NSGA2 is a genetic algorithm [90] and ALPSO is a particle swarm algorithm [91]. The results for design variables counts ranging from 2 to 512 are shown in Figure 3.1 [101].

Figure 3.1: Scaling behaviour for gradient-based and non gradient-based optimizers. All optimizations are converged to a relative tolerance of $1 \times 10^{-2}$.

The non-gradient based algorithms (ALPSO and NSGA2) exhibit very poor scaling behaviour, requiring millions of function evaluations, even for design problems with fewer than 100 design variables. The gradient based algorithms show far better scaling. The SNOPT algorithm with analytic gradients shows the best scaling behaviour; the number of function evaluations is only weakly dependent on the number of design variables. However, the cost of using the same optimization algorithm with finite differencing very quickly becomes too costly when function evaluations are expensive. The non-gradient based approaches do have the advantage of being global and their use increases the probability of getting closer to the global optimum.

If the local nature of gradient based methods is acceptable and the output functions are sufficiently smooth, gradient based methods provide a much more computationally tractable way of dealing with large numbers of design variables and expensive function evaluations. However, as we have seen, the desirable scaling properties of gradient based optimization is only possible when the computational cost of a gradient analysis is approximately equal to the cost of a function evaluation and is independent of the number of variables. Traditional numerical methods for evaluating gradients of functions such as finite differencing and the
complex-step method [49] do not satisfy these conditions. For both of these methods, the
cost of computing the complete gradient is a linear function of the number of design variables
(see Section 3.8.2 for example).

Fortunately, all hope is not lost. Provided the desired optimization problem can be
written with significantly fewer functions than design variables, it is possible to use an
adjoint method. The development, verification and benchmarking of the coupled adjoint
method applied to high fidelity aerostructural systems are discussed subsequently.

3.1 Coupled Aerostructural Adjoint Overview

The goal of the coupled adjoint method is to compute the derivatives of a function of
interest $I$ with respect to many design variables $x$. The function of interest may be the
optimization objective function such as the drag coefficient, or it may be a constraint such
as the lift coefficient. The development begins by assuming that the function of interest is
both a direct function of the design variables $x$ and a function of the state variables, $w$ and $u$,
determined from the solution of the coupled aerostructural analysis problem, Equation (2.27):

\[
I = f(x, (w, u)).
\]  

(3.2)

Applying the chain rule to Equation (3.2), the total sensitivity of the function of interest
can be expanded as:

\[
\frac{dI}{dx} = \frac{\partial I}{\partial x} + \left[ \frac{\partial I}{\partial w} \frac{\partial w}{dx} \right] \left[ \frac{\partial I}{\partial u} \frac{\partial u}{dx} \right].
\]  

(3.3)

In Equation (3.3), a distinction is made between the partial-derivatives denoted “$\partial$” and the
total-derivatives, denoted “$d$”. The total-derivatives indicate that the derivative requires
the solution of the nonlinear system of equations. Once the solution of the aerostructural
analysis is converged, the coupled residual is approximately zero, and in an analogous fashion
to Equation (3.3) the total derivative of the residual equation is

\[
\begin{bmatrix}
\frac{dA}{dx} \\
\frac{dS}{dx}
\end{bmatrix} = \begin{bmatrix}
\frac{\partial A}{\partial x} \\
\frac{\partial S}{\partial x}
\end{bmatrix} + \begin{bmatrix}
\frac{\partial A}{\partial w} & \frac{\partial A}{\partial u}
\end{bmatrix} \begin{bmatrix}
\frac{\partial w}{dx} \\
\frac{\partial u}{dx}
\end{bmatrix} = 0.
\]

(3.4)
Substituting the solution of Equation (3.4) into Equation (3.3) the remaining total-derivative term, \( \left[ \frac{dw}{dx} \quad \frac{du}{dx} \right]^T \), can be eliminated:

\[
\frac{dI}{dx} = \frac{\partial I}{\partial x} - \left[ \frac{\partial I}{\partial w} \frac{\partial I}{\partial u} \right] \Psi^T \left[ \frac{\partial A}{\partial w} \frac{\partial A}{\partial u} \frac{\partial S}{\partial w} \frac{\partial S}{\partial u} \right]^{-1} \left[ \frac{\partial A}{\partial x} \frac{\partial S}{\partial x} \right].
\] 

(3.5)

There are two techniques for solving this equation. The first is to solve Equation (3.4) for \( \left[ \frac{dw}{dx} \quad \frac{du}{dx} \right]^T \) for a given design variable and to use Equation (3.3) to compute the total-derivative. This is known as the direct method. If, however, there are more design variables than functions of interest — which is usually the case for high-fidelity aerostructural optimization — it is computationally more efficient to use the adjoint method and to solve for the coupled adjoint vector, \( \Psi = [\psi^T \phi^T]^T \), where \( \psi \) and \( \phi \) are the aerodynamic and structural adjoint vectors, respectively. Rearranging Equation (3.5) gives

\[
\left[ \frac{\partial A}{\partial w} \frac{\partial A}{\partial u} \frac{\partial S}{\partial w} \frac{\partial S}{\partial u} \right]^T \left[ \begin{array}{c} \psi \\ \phi \end{array} \right] = \left[ \frac{\partial I}{\partial w} \frac{\partial I}{\partial u} \right]^T.
\] 

(3.6)

These are the coupled adjoint equations, which are solved once for each function of interest. After the solution for the coupled adjoint equations (3.6) is obtained, the following equation can be used to compute the total-derivative:

\[
\frac{dI}{dx} = \frac{\partial I}{\partial x} - \psi^T \left( \frac{\partial A}{\partial x} \right) - \phi^T \left( \frac{\partial S}{\partial x} \right).
\] 

(3.7)

The implementation details for the partial-derivative terms in Equations (3.6) and (3.7) are explained in the following sections.

### 3.2 Aerodynamic Adjoint

This section will describe the methods I used to form and solve the aerodynamic adjoint. Excluding the off-diagonal term, the first blocked row of Equation (3.6), \( \frac{\partial A}{\partial w} \psi = \frac{\partial I}{\partial w} \), is the aerodynamic adjoint equation. There are generally two different approaches for the derivation and discretization of the aerodynamic adjoint: the continuous approach [14] and
Chapter 3. Gradient Computation Methodology

The discrete approach [102, 103].

The continuous approach first applies the transpose operation to the governing equations and boundary conditions and then the transformed equations are discretized and solved. Continuous adjoint methods typically use the same discretization and numerical solution approach as for the flow equations. The discrete approach, however, applies the transpose operation directly to the discretized flow equations and then the solution of the resulting linear system is obtained. One of the advantages of the discrete approach is that the computed sensitivity is consistent with the discretized governing equations. Since the solution and gradients are consistent, a numerically exact differentiation method, such as complex-step or forward-mode automatic differentiation, can be used to verify the implementation. As the grid spacing is reduced, both methods produce the same gradients [104]. I have chosen to use the discrete approach primarily for its consistency property; the ability to verify the complete aerostructural adjoint method using the complex-step method is extremely valuable.

The aerodynamic adjoint equation is solved using GMRES, in a similar manner to the solution for the Newton update in Equation (2.16). However, the aerodynamic residual matrix is now transposed and the finite difference matrix-free approach used for the solution cannot be employed. The approach I have used is to compute and store the transpose of the aerodynamic Jacobian. The required matrix-vector products for GMRES are then readily obtained.

The evaluation of the complete aerodynamic Jacobian, $\partial A/\partial w$, can be obtained in a number of different ways. A common approach is to differentiate and assemble the residual contributions by hand [16]. This approach is potentially the fastest computationally, but it comes at the expense of significant coding effort and intricate knowledge of the code structure. A second class of methods I will call forward methods include finite-differencing, the complex-step method and forward-mode AD. With each of these methods the residual routine is evaluated repeatedly with appropriate perturbations on $w$ to simultaneously evaluate one or more columns of the Jacobian. Finally, reverse methods use reverse-mode AD to obtain one or more rows of the Jacobian simultaneously.

A reverse-mode method for evaluating the aerodynamic Jacobian was implemented in SUmb by Mader et al. [103]. This method uses modified residual routines that compute the residual in a single cell only. This is referred to as a single cell routine. To evaluate a given row in the aerodynamic Jacobian, a perturbation is placed on the corresponding $i^{th}$ component of the cell residual, the single cell routine is executed with reverse-mode AD and the sensitivity of the inputs determine the entire row of the Jacobian. As expected and noted
by Mader et al., the reverse-mode cost is approximate 4.5 times the cost of the single cell residual routine. However, the implementation of the single cell routine is relatively naïve, as it performs far more computational work than is strictly necessary. This results in the single cell routine being approximately 10 times slower than the original routine. Presented with the option of improving the reverse-mode method or pursuing a different approach, I choose to implement a forward-mode approach using AD. There are two main advantages of this method: the original routines can be differentiated directly, eliminating the computational penalty of the single cell routine. Secondly, the original routines do not need to be rewritten, thus saving a considerable amount of coding effort.

In contrast to the reverse approach, which evaluates a single row of the Jacobian at a time, forward approaches evaluate (multiple) columns of the Jacobian simultaneously. A naïve approach would require evaluating each column of the Jacobian in sequence, resulting in $O(10^7)$ evaluations for the size of aerodynamic problems I am considering! However, since the aerodynamic Jacobian is sparse, a far more tractable approach can be used.

The general idea is to determine groups of independent columns of the Jacobian. A group of columns is considered independent if no row contains more than one nonzero entry. This allows a group of independent columns to be evaluated simultaneously. The process of determining which columns are independent is known as graph coloring. The determination of an optimal (smallest) set of colors for a general graph is quite challenging. For unstructured grids, a greedy coloring scheme can be used resulting is a satisfactory number of colors [105].

For structured grids with regular repeating stencils, the graph coloring problem is substantially simpler [106]. Consider the 13-cell stencil for the Euler residual evaluation shown in Figure 3.2a. It is clear at least 13 colors will be required. Determining the optimum graph coloring for this case is is equivalent to finding a three dimensional packing sequence that minimizes the unused space between stencils. Fortunately, for this stencil, a perfect packing sequence is possible and precisely 13 colors are required. A three dimensional view of the stencil packing is shown in Figure 3.2b.

A slice in the $i - j$ plane is shown in Figure 3.2c. The center cell on the left side of the figure is arbitrarily assigned color, $\chi$ of 1, and the colors are incremented by one from left to right. It is easily verified the pattern repeats after 13 colors and every cell in the stencil has a been assigned a unique color. This diagram can then be used to determine a formula to identify the color associated with cell at index $i, j, k$:

$$\chi(i, j, k) = \text{mod}(i + 3j + 4k, 13).$$

(3.8)
A similar coloring scheme can be used to compute the derivative of the aerodynamic residuals with respect to the volume grid coordinates, $X_V$. In this case, the coloring is used to determine which nodes can be perturbed simultaneously. The stencil of affected cells surrounding the central node is given in Figure 3.3a and a possible three-dimensional packing is given in Figure 3.3b. However, unlike the $\partial A/\partial w$ stencil, the $\partial A/\partial X_V$ stencil cannot pack perfectly into three-space without holes. These holes are identified by black squares in Figure 3.3c. For this coloring, 38 colors are required to evaluate the 32 values in the stencil. The formula that determines the color of each node at index $i,j,k$ is:

$$\chi(i,j,k) = \text{mod}(i + 7j + 27k, 38).$$

(3.9)

As previously mentioned, all forward methods are conceptually similar and it is quite straightforward to use finite-differences instead of forward-mode AD. The aerodynamic adjoint and the Newton–Krylov solution method in Sumb both require the assembly of a
first-order approximation of the Jacobian matrix, $\partial A_1/\partial w$, which is used to form the preconditioner. Due to the lumped first-order dissipation, the stencil contains only 7 cells. Like the second-order $\partial A/\partial w$ stencil, this stencil packs perfectly and requires precisely 7 colors. High numerical precision is not required in the preconditioner and thus I use forward differencing to further reduce the computational cost of this matrix assembly.

I now present a comparison of the performance of forward AD and reverse AD methods. The test uses the simple transonic wing Euler grid described in Appendix A and is executed on 16 processors. The reverse-mode method can compute $\partial A/\partial w$ and $\partial A/\partial X_V$ simultaneously, but requires a separate evaluation for the first-order preconditioner matrix. The forward-mode AD method requires separate evaluations for each matrix. The timing results are shown in Table 3.1.

The forward method offers significant computational savings, primarily due to the much faster assembly of the first-order Jacobian. The preconditioner assembly is especially important for the Newton–Krylov method, where many assemblies may be required during the
Table 3.1: Aerodynamic residual matrix timings

<table>
<thead>
<tr>
<th>Task</th>
<th>reverse-mode (s)</th>
<th>forward-mode (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\partial A_i/\partial w$</td>
<td>97.8</td>
<td>3.6</td>
</tr>
<tr>
<td>$\partial A/\partial w$</td>
<td>97.8</td>
<td>21.4</td>
</tr>
<tr>
<td>$\partial A/\partial X_v$</td>
<td>—</td>
<td>59.5</td>
</tr>
<tr>
<td>Total</td>
<td>198.2</td>
<td>84.5</td>
</tr>
</tbody>
</table>

course of a nonlinear analysis.

3.3 Structural Adjoint

Excluding the off-diagonal term, the second block row of Equation (3.6), $\frac{\partial S^T}{\partial u} \phi = \frac{\partial I^T}{\partial u}$, is the structural adjoint equation. The structural solver, TACS, includes a structural adjoint implementation and has been used without modification for this thesis [107]. For structural optimization, the residual matrix is just the tangent stiffness matrix. For linear analysis, the Jacobian is symmetric and the same factorization for the solution can be reused for the adjoint. The other partial-derivative terms required for the structural adjoint, $\partial I/\partial u$, $\partial I/\partial x$ and $\partial S/\partial x$ are evaluated analytically. The design variable partial-derivatives include both structural thickness variables and geometric design variables.

3.4 Geometry Derivatives

Analytic sensitivities are easily obtained for the FFD approach. Since the FFD volumes I use are tri-variate B-spline volumes, the sensitivity of any point inside the volume is

$$\frac{\partial X_J}{\partial x} = \frac{\partial X_J}{\partial X_{\text{coef}}} \frac{\partial X_{\text{coef}}}{\partial x}$$  \hspace{1cm} (3.10)$$

where $X_J$ are the coordinates of the jig shape for each discipline, $x$ are the optimization design variables, and $X_{\text{coef}}$ are the spatial coordinates of the FFD lattice control points. The $\partial X_J/\partial X_{\text{coef}}$ term consists of the shape functions at the $u, v, w$ parametric coordinates, and the $\partial X_{\text{coef}}/\partial x$ term relates how the control points move with respect to the optimization design variables. Local shape design variables move individual control points along a given coordinate direction. For global shape control, many control points are grouped and moved simultaneously. User-specified Python functions are used to describe the desired design variable movement of control point groups. The derivatives of the local design variables
are evaluated analytically and the derivatives of the user supplied global design variable functions are evaluated automatically using the complex-step method [49]. These two sets of derivatives from the second term, \( \frac{X_{\text{coef}}}{\partial x} \).

### 3.5 Coupled Adjoint Implementation

Implementing the coupled adjoint, including all the required coupled partial-derivative terms, is a challenging endeavour. Furthermore, ensuring that the partial-derivative computations and solution methods exhibit good parallel scalability is even more difficult. In this section, I present some important aspects involved in the solution of Equation (3.6), as well as in the computation of the required partial-derivative terms. The choice between the various approaches is justified in terms of trade-offs between cost of implementation, computational effort, and memory requirements.

**Aerodynamic Residual Partial-Derivatives**

The diagonal term, \( \partial A / \partial w \) is identical to the aerodynamic adjoint described previously and reused without modification.

The first off-diagonal block, \( \partial A / \partial u \), contains the derivatives of the aerodynamic residual with respect to the structural displacements. This is a challenging matrix to compute in a two-state aerostructural formulation. What makes this term particularly challenging is that a single structural degree of freedom (DOF) can affect all aerodynamic cells. Using the hybrid mesh-deformation scheme described in Chapter 2, any structural DOF that is linked to a mesh super node, perturbs the entire CFD mesh and results in a dense column due to the solution of the linear system of equations. With a large number of structural surface DOFs and a large number of super nodes, storing this matrix would require an excessive amount of memory. Instead, I compute this term in a partially matrix-free fashion using the chain rule:

\[
(\partial A / \partial u)^T \psi = (\partial X_S / \partial u)^T (\partial X_V / \partial X_S)^T (\partial A / \partial X_V)^T \psi
\]

(3.11)

where the first term from the right, \( \partial A / \partial X_V \), contains the sensitivities of the aerodynamic residuals with respect to variations of all the volume mesh coordinates. This term is computed using the forward-mode AD approach outlined in the previous section. This matrix is computed and stored, which allows us to compute the transpose matrix-vector products in a relatively inexpensive manner.

The matrix of derivatives of the mesh deformation, \( \partial X_V / \partial X_S \), requires a careful implementation to be computationally efficient. For the transpose matrix-vector products required
for the adjoint computation, the normal deformation procedure — previously shown in Figure 2.10 — proceeds in reverse. Reverse-mode AD and a mesh adjoint equation are used to compute the variation of all the surface nodes ($X_S$) and the variation of the volume super nodes ($X_V$). The variation of the volume super nodes is related to the surface super nodes through the solution of the mesh adjoint equation given by

$$K_{uu}^T \psi_M = -Y_V,$$  \hspace{1cm} (3.12)

where $\psi_M$ is the mesh adjoint. Finally, the variation $Y_S$ is prolonged to the full mesh surface using the transpose of the restriction operator. A depiction of the procedure is given in Figure 3.4.

This computation is performed just once for each transpose matrix-vector product. To ensure overall scalability of the coupled adjoint solution, all of the computations are executed in parallel with acceptable efficiency. The same parallel direct method used for the forward warping procedure (SuperLU_DIST) is reused for the mesh adjoint system, and the factorization only needs to be performed once. Subsequent mesh perturbations and adjoint solutions can be computed quickly using back substitution operations. Finally, since $X_S$ is given by $X_S = X_J + Tu$, the partial term $(\partial X_S/\partial u)^T$ is simply $T^T$, the transpose of the displacement transfer matrix.

The advantage of the nonlinear mesh-deformation algorithm over the stepped linear approach is now clear. Both the linear and nonlinear mesh-deformation schemes require the solution of a single mesh adjoint using Equation (3.12). In the nonlinear case, $K_{uu}$ is non-symmetric and a single direct factorization of $K_{uu}^T$ is performed at the beginning of the gradient analysis. This factorization is the only addition cost of the nonlinear scheme over the linear scheme for sensitivity analysis. To compute the warping derivative for the stepped linear approach, $N$ mesh adjoint equations must be solved where $N$ is the number of mesh increments. This represents significant additional computational cost and may potentially
suffer from reduced accuracy if the sequential mesh adjoint equations are not tightly converged.

This way of computing $\partial A/\partial u$ contrasts with the finite differences used by Martins et al. [15], where each structural surface DOF was perturbed in turn, followed by a mesh deformation to re-evaluate the aerodynamic residuals. The computational cost of doing these finite differences scaled directly with the number of surface structural surface DOFs and it became clear that this approach would not be suitable for the $O(10^4)$ surface DOFs used in this thesis. As we will see in Section 3.8.1, this method allows us to quickly compute the aerostructural gradient with structural meshes with over 1 million DOFs and problems with thousands of design variables.

**Structural Residual Partial-Derivatives**

The other off-diagonal block in the coupled adjoint equations (3.6), $\partial S/\partial w$, represents the derivatives of the structural residuals with respect to the aerodynamic states. The only contribution to this matrix is due to the aerodynamic forces, $F_A$, and is given by,

$$
\left( \frac{\partial S}{\partial w} \right)^T \phi = - \left( \frac{\partial F}{\partial w} \right)^T \phi
$$

$$
= - \left( \frac{\partial F_A}{\partial w} \right)^T \left( \frac{\partial F}{\partial F_A} \right)^T \phi
$$

$$
= - \left( \frac{\partial F_A}{\partial w} \right)^T \phi.
$$

I compute and store $\partial F_A/\partial w$, and $\partial F_A/\partial X_S$, which I refer to as the coupling matrices. These terms are computed using reverse-mode AD applied to the CFD force evaluation routine. Unlike $\partial A/\partial w$ or $\partial A/\partial X_v$, only cells on the wetted surface have nonzero components. Thus, the cost of storing these terms is very small.

The remaining diagonal block, $\partial S/\partial u$, represents the derivative of the structural residuals with respect to the structural states. For linear aerostructural analysis, this term is not simply the linear stiffness matrix, $K$. Given the structural residuals (2.23), we find that the applied forces, $F$, are actually explicitly functions of $u$. As the flexible structure deforms, the surface normals on the CFD mesh change orientation, giving rise to a nonlinear following force. To produce accurate derivatives, this effect must be included. To obtain the matrix-vector products required by the Krylov-type solution strategy, I compute this term in the
following manner:

\[
\left( \frac{\partial S}{\partial u} \right)^T \phi = \left( K - \left( \frac{\partial F}{\partial u} \right)^T \right) \phi = \left( K - \left( \frac{\partial F_A}{\partial F_A} \right) (\frac{\partial F_A}{\partial X_S})^T \left( \frac{\partial X_S}{\partial u} \right) \right) \phi = \left( K - T^T \left( \frac{\partial F_A}{\partial X_S} \right)^T T \right) \phi.
\]

(3.14)

This product is formed by transferring the \( \phi \) vector to the aerodynamics using a displacement transfer, finding the \( \partial F_A/\partial X_S \) transpose matrix-vector product, and then transferring the result back to the structural solver using a load transfer. Since this operation requires two synchronous data transfers, these products are evaluated only as required and the exact \( \partial S/\partial u \) is never formed.

**Partial-Derivatives of the Function of Interest**

The right-hand side of the coupled adjoint system (3.6) is the derivative of the function of interest with respect to the system states. The derivatives of typical aerodynamic functions such as \( C_L, C_D, \) and \( C_M \) are computed with reverse-mode AD, in a similar fashion to the coupling matrices. The derivatives of typical structural functions, such as individual element stresses, \( \sigma_i \), or the Kreisselmeier–Steinhauser (KS) [85, 108] aggregation functions are evaluated analytically. For the structural functions, \( \partial I/\partial w \) is zero. However, for aerodynamic functions involving surface pressure or traction integration, \( \partial I/\partial u \) is nonzero and is given by

\[
\frac{\partial I}{\partial u} = \left( \frac{\partial I}{\partial X_S} \right) \left( \frac{\partial X_S}{\partial u} \right) = \left( \frac{\partial I}{\partial X_S} \right) T.
\]

(3.15)

The resulting procedure is similar to that used in the computation of \( \partial S/\partial u \), expressed in Equation (3.14).

**Total Sensitivity Partial-Derivatives**

The evaluation of the partial-derivative terms in the total-derivative equation (3.7) is challenging because the load and displacement transfer operations have a dependency on the geometric design variables. Figure 2.18 shows how the displaced surface coordinates, \( X_S \), are extrapolated from the structural domain. The complication arises from the fact that the
length of the extrapolation vector, $r$, changes with the design variables. Accounting for this change leads to a transfer scheme that remains accurate for large geometry changes, but it complicates the computation of the derivatives.

For aerodynamic functions, I compute $\partial I/\partial X_S$ using reverse-mode AD, similar to what was done for the coupling matrices in Equation (3.13). Then, for each design variable, the derivative of the perturbed shape $X_S$ is evaluated with respect to the design variables, $x$, by computing

$$\frac{\partial I}{\partial x} = \frac{\partial I}{\partial X_S} \left( \frac{\partial X_S}{\partial x} \right) \quad (3.16)$$

$$= \frac{\partial I}{\partial X_S} \left( \frac{\partial X_J}{\partial x} + \frac{\partial (u_r \times r)}{\partial x} \right).$$

Since the transfer-operation matrix $T$ is never explicitly formed, I evaluate a single entry of the resulting vector sequentially in a matrix-free fashion. A similar correction must be made for the derivative of the structural states with respect to the geometric design variables, since the load transfer depends on the design variables.

Finally, the evaluation of the $\partial X_S/\partial x$ term is used to complete the computation of $\partial A/\partial x$. These operations are relatively costly, since in our implementation they require a synchronous transfer of data from both disciplines for each design variable. As we will see in Section 3.8.1, the cost of this computation scales very weakly with the number of design variables, and thus makes a small contribution to the overall computational cost of computing the gradient.

### 3.6 Gradient Verification

Accurate derivatives are critical for the performance of any gradient-based optimization. As described in previous sections, all partial-derivative terms for the coupled adjoint equation (3.6) and total sensitivity equation (3.7) are computed either with AD (using Tape-nade [109]) or analytically. These terms can be evaluated with near machine-precision, allowing for extremely accurate derivatives. To ensure an entirely consistent formulation and implementation, we verify the coupled adjoint using the complex-step method [110, 49] as a benchmark applied to the entire coupled system. In my experience, if only a limited number of digits in the derivatives is verified, certain bugs in the computations that may go unnoticed in one case, may cause problems in other cases. Therefore, I do not use finite-difference formulae to verify the results herein.

Using “complexified” versions of all the analysis modules, a complex perturbation can be
placed on any design variable. When an aerostructural solution is computed with a complex perturbation on a design variable, the resulting complex parts in the functions of interest correspond to the derivatives of those functions. Unlike finite differences, the complex-step method does not suffer from subtractive-cancellation. With the elimination of the subtractive cancellation error we can choose an arbitrarily small step size and reduce the truncation error to machine precision. For iterative methods, the complex perturbations on the function of interest converge as the nonlinear system converges [49].

The derivative verification is carried out with the level-1 discretization given in Table 3.4. The aerostructural analysis and adjoint systems are converged to a relative tolerance of $\epsilon_{AS} = \epsilon_{SA} = 10^{-8}$. It is generally not possible to obtain any further convergence for the aerostructural system, since the condition number of the structural Jacobian is typically $O(10^9)$ for the shell structures used in this thesis. The resulting derivatives agree to $O(10^{-5})$. Obtaining better agreement becomes difficult because small discrepancies between the real and complex codes often appear, probably due to differences in compiler implementations and optimizations of certain built-in functions that do not always produce bit-compatible results. However, the level of accuracy achieved with the current coupled adjoint implementation is more than sufficient for gradient-based optimization.

For the verification, I computed the derivative of two aerodynamic functions, $C_L$ and $C_D$, and a structural function, the KS stress aggregation function. The KS function aggregates the von Mises stress on the lower skin and lower stringers. The sensitivities of these three functions are computed with respect to two global variables, span and sweep, a local aerodynamic variable, the angle of attack, $\alpha$, and a local structural variable, $t_{\text{skin}}$, the structural skin thickness of a panel near the root. Table 3.2 shows the derivatives computed using the coupled adjoint method, the complex-step method, and the forward finite-difference formula. A complex-step size of $1 \times 10^{-40}j$ is used for the complex-step method. For the finite-difference derivatives, I show the result for the optimal step size for each variable, $h_{\text{opt}}$. Seven step sizes ranging from $1 \times 10^{-1}$ to $1 \times 10^{-7}$ were considered. I then use the derivative computed with the optimum step, i.e., the step that minimizes the sum of the truncation error and the subtractive cancellation error, as compared with the results given by the complex-step method.

The sensitivities computed with the coupled adjoint method differ from the complex-step method by relative errors of $10^{-3}$ to $10^{-6}$. The sensitivities with respect to the local aerodynamic variable, $\alpha$, are typically more accurate due primarily to a more straightforward total-derivative computation. The finite-difference sensitivities are much less reliable and
contribute to the observed noise. Figure 3.5b also shows vectors representing the derivative with respect to sweep vary smoothly at this scale. Figure 3.5b shows a detailed view where the variation in sweep is six orders of magnitude smaller than the one shown in Figure 3.5a. Here, the noise inherent in the aerostructural solution is visible and is on the order of $O(10^{-7})C_D$. The primary reason for solution noise stems from the from the numerical stiffness of the structural Jacobian and the use of floating point arithmetic. For the structural system, the high numerical stiffness results in variations of the Least Significant Bit (LSB) to be amplified many orders of magnitude and prevents obtaining solutions tighter than $O(10^{-8})$. The observed noise level is consistent with this achievable solution tolerance. The non-differentiable pressure switch in the scalar JST artificial dissipation scheme may also contribute to the observed noise. Figure 3.5b also shows vectors representing the derivative

### Table 3.2: Coupled adjoint sensitivity verification

<table>
<thead>
<tr>
<th>Function Variable</th>
<th>Complex-step Value</th>
<th>Coupled adjoint Value</th>
<th>Rel. Error</th>
<th>Finite-difference Value</th>
<th>Rel. Error</th>
<th>$h_{\text{opt}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_L$ Span</td>
<td>0.0272653249</td>
<td>0.027266123</td>
<td>$-3.21 \times 10^{-5}$</td>
<td>0.027858610</td>
<td>$-2.18 \times 10^{-2}$</td>
<td>$1 \times 10^{-6}$</td>
</tr>
<tr>
<td>Sweep</td>
<td>-0.023601035</td>
<td>-0.023595480</td>
<td>$2.35 \times 10^{-4}$</td>
<td>-0.023634861</td>
<td>$-1.43 \times 10^{-3}$</td>
<td>$1 \times 10^{-4}$</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.1439534101</td>
<td>0.143953092</td>
<td>$5.96 \times 10^{-8}$</td>
<td>0.143632330</td>
<td>$2.23 \times 10^{-3}$</td>
<td>$1 \times 10^{-6}$</td>
</tr>
<tr>
<td>$t_{\text{skin}}$</td>
<td>0.145264733</td>
<td>0.145266846</td>
<td>$-1.45 \times 10^{-5}$</td>
<td>0.144828960</td>
<td>$3.00 \times 10^{-3}$</td>
<td>$1 \times 10^{-5}$</td>
</tr>
<tr>
<td>$C_D$ Span</td>
<td>0.000954129</td>
<td>0.000953880</td>
<td>$2.61 \times 10^{-4}$</td>
<td>0.000911573</td>
<td>$4.46 \times 10^{-2}$</td>
<td>$1 \times 10^{-5}$</td>
</tr>
<tr>
<td>Sweep</td>
<td>-0.001386782</td>
<td>-0.001388121</td>
<td>$9.65 \times 10^{-4}$</td>
<td>-0.001292728</td>
<td>$6.78 \times 10^{-2}$</td>
<td>$1 \times 10^{-3}$</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.0063369099</td>
<td>0.006336908</td>
<td>$1.55 \times 10^{-7}$</td>
<td>0.006234932</td>
<td>$1.61 \times 10^{-2}$</td>
<td>$1 \times 10^{-1}$</td>
</tr>
<tr>
<td>$t_{\text{skin}}$</td>
<td>0.007273797</td>
<td>0.007273570</td>
<td>$3.12 \times 10^{-5}$</td>
<td>0.007141012</td>
<td>$1.83 \times 10^{-2}$</td>
<td>$1 \times 10^{-6}$</td>
</tr>
<tr>
<td>$KS$ Span</td>
<td>0.020412974</td>
<td>0.020414310</td>
<td>$-6.55 \times 10^{-5}$</td>
<td>0.020939720</td>
<td>$-2.58 \times 10^{-2}$</td>
<td>$1 \times 10^{-3}$</td>
</tr>
<tr>
<td>Sweep</td>
<td>-0.006395502</td>
<td>-0.006391973</td>
<td>$1.18 \times 10^{-3}$</td>
<td>-0.005821129</td>
<td>$9.04 \times 10^{-2}$</td>
<td>$1 \times 10^{-3}$</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.086843094</td>
<td>0.086843079</td>
<td>$1.70 \times 10^{-7}$</td>
<td>0.092562735</td>
<td>$-6.59 \times 10^{-2}$</td>
<td>$1 \times 10^{-3}$</td>
</tr>
<tr>
<td>$t_{\text{skin}}$</td>
<td>-0.660819304</td>
<td>-0.660817730</td>
<td>$2.38 \times 10^{-6}$</td>
<td>-0.291215140</td>
<td>$5.59 \times 10^{-1}$</td>
<td>$1 \times 10^{-5}$</td>
</tr>
</tbody>
</table>

I also performed a study to demonstrate that the design space resulting from the aerostructural analysis technique is relatively smooth and continuous with respect to the geometric variables. Geometric variables directly affect both disciplines through mesh changes, and therefore the derivatives with respect to these variables are more challenging to compute. A shearing sweep design variable that displaces all coordinates in the x-direction along the span of the wing is used in this study. Figure 3.5c shows the extent of the sweep change about the reference value, from $\Delta x = -2$ to $\Delta x = 2$. All solutions and gradients are converged to a relative tolerance of $\epsilon_{AS} = \epsilon_{SA} = 10^{-8}$. Figure 3.5a plots the derivative of the drag coefficient over this sweep range. We can see that both the drag coefficient and its derivative with respect to sweep vary smoothly at this scale. Figure 3.5b shows a detailed view where the variation in sweep is six orders of magnitude smaller than the one shown in Figure 3.5a. Here, the noise inherent in the aerostructural solution is visible and is on the order of $O(10^{-7})C_D$. The primary reason for solution noise stems from the from the numerical stiffness of the structural Jacobian and the use of floating point arithmetic. For the structural system, the high numerical stiffness results in variations of the Least Significant Bit (LSB) to be amplified many orders of magnitude and prevents obtaining solutions tighter than $O(10^{-8})$. The observed noise level is consistent with this achievable solution tolerance.
computed with the coupled adjoint method. In spite of the noise in the drag coefficient, the derivative still indicate the correct trend of the noisy function.

(a) Gradient of $C_D$ with respect to shearing sweep for a range of sweep values

(b) Noise level in $C_D$ solution. Vectors denote coupled adjoint computed derivative.

(c) Visualization of the change in sweep

Figure 3.5: Verification of smoothness of coupled adjoint derivatives

To employ gradient-based optimization effectively, smooth functions and continuous first derivatives are required. I have taken great care to ensure that all computational components of the aerostructural analysis, including geometry manipulation, mesh-deformation,
CFD analysis, and CSM analysis have sufficiently smooth responses. It is clear from Figure 3.5a that the derivatives when viewed in an appropriate engineering scale, are not only continuous, but also smooth. The smooth design space and the accurate coupled adjoint derivatives demonstrate that the current framework is well suited for large-scale gradient-based optimization.

### 3.7 Coupled Adjoint Solution Strategy

As with the nonlinear analysis, I consider two techniques to solve the coupled system (3.6): a segregated approach and a monolithic approach. The segregated approach is the lagged coupled adjoint approach [15], which is equivalent to a (linear) block Gauss-Seidel approach. The monolithic approach uses a Krylov subspace method applied directly to the coupled linear system.

#### 3.7.1 Linear Block Gauss–Seidel Method (LBGS)

The analog of the NLBGS method for coupled adjoint equations is the linear block Gauss–Seidel Method (LGBS). This method expresses the interdisciplinary coupling as additional forcing terms to the right-hand side of each respective adjoint solver. The separation of the single blocked-matrix equation yields:

\[
\left( \frac{\partial A}{\partial w} \right)^T \psi^{(k)} = \left( \frac{\partial I}{\partial w} \right)^T - \left( \frac{\partial S}{\partial w} \right)^T \phi^{(k-1)}
\]

\[
\left( \frac{\partial S}{\partial u} \right)_K \phi^{(k)} = \left( \frac{\partial I}{\partial u} \right)^T - \left( \frac{\partial A}{\partial u} \right)^T \psi^{(k)} - \left( \frac{\partial S}{\partial u} \right)_F \phi^{(k-1)}.
\]

The subscripts \(K\) and \(F\) on the partial term, \(\partial S/\partial u\), represent the contributions from the stiffness matrix and external forces, respectively. Since the external-force component is costly to compute and involves a synchronous communication between disciplines, it is lagged, as is the contribution from the off-diagonal term. These equations are iterated in a sequential fashion until the desired convergence level is reached. The main advantage of this approach is that the discipline adjoint solvers can be reused by simply adding the appropriate right-hand side forcing terms. In practice, each system is only partially converged before a data exchange is made. Additionally, depending on the problem, it may be necessary to apply an under-relaxation factor to the structural update for stability reasons. The pseudocode for this approach is listed in Algorithm 3.
Algorithm 3 Block Gauss–Seidel coupled adjoint solution method

1: Given: \( \psi^{(0)}, \phi^{(0)}, k_{\text{max}}, \theta \in (0, 1] \) \( \triangleright \) \( \psi^{(0)}, \phi^{(0)} \) are possibly initialized from previous solution

2: if Aerodynamic Function then

3: \( \frac{\partial I}{\partial u} \leftarrow T^T \left( \frac{\partial I}{\partial X_S} \right) \) \( \triangleright \) Cross-discipline function partial

4: end if

5: for \( k \leftarrow 1, k_{\text{max}} \) do

6: \( \phi_A \leftarrow T \phi^{(k-1)} \) \( \triangleright \) Transfer structural adjoint

7: \( R_A^{(k)} \leftarrow \left( \frac{\partial I}{\partial w} \right)^T \left( \frac{\partial A}{\partial w} \right)^T \psi^{(k-1)} - \left( \frac{\partial F_A}{\partial w} \right)^T \phi_A \) \( \triangleright \) Aerodynamic adjoint RHS

8: \( \left( \frac{\partial A}{\partial w} \right)^T \Delta \psi = R_A^{(k)} \) \( \triangleright \) Partially solve CFD adjoint update

9: \( \psi^{(k)} \leftarrow \psi^{(k-1)} + \Delta \psi \) \( \triangleright \) Update aerodynamic adjoint

10: \( P \leftarrow T^T \left( \frac{\partial X_V}{\partial X_S} \right)^T \left( \frac{\partial A}{\partial X_V} \right)^T \psi^{(k)} \) \( \triangleright \) Transfer aerodynamic contribution to structural RHS

11: \( Q \leftarrow T^T \left( \frac{\partial F_A}{\partial X_S} \right)^T T \phi^{(k-1)} \) \( \triangleright \) Force \( \frac{\partial S}{\partial u} \) contribution

12: \( R_S^{(k)} \leftarrow \left( \frac{\partial I}{\partial u} \right)^T - K \phi^{(k-1)} - P - Q \) \( \triangleright \) Compute Structural RHS

13: \( K \Delta \phi = R_S^{(k)} \) \( \triangleright \) Partially solve CSM adjoint update

14: \( \phi^{(k)} \leftarrow \phi^{(k-1)} + \theta \Delta \phi \) \( \triangleright \) Under-relaxed structural adjoint update

15: if \( \| R_A^{(k)} \| < \epsilon_{SA} \| R_A^{(1)} \| \) and \( \| R_S^{(k)} \| < \epsilon_{SA} \| R_S^{(1)} \| \) then \( \triangleright \) Coupled adjoint convergence check

16: Break

17: end if

18: end for
3.7.2 Coupled Krylov Method (CK)

The second solution approach for the coupled adjoint equations (3.6) is a fully coupled, monolithic method. As with the Newton–Krylov solution method, this method follows previous work with low fidelity systems [107]. This is the first use of a monolithic solution method for the coupled adjoint equations on large-scale, aerostructural problems.

Krylov subspace methods, are particularly attractive for this type of problem, since they require only matrix-vector products, allowing the use of matrix-free methods. In our case, the diagonal blocks of the Jacobian in Equation (3.6) are stored, but the off-diagonal terms are not explicitly stored, resulting in a partially matrix-free implementation. Effective preconditioning is critical to the performance of Krylov methods, especially on large systems of equations such as the ones I am considering.

I choose to use a block-Jacobi preconditioner for the coupled system, since this approach ignores the off-diagonal terms and allows the aerodynamic and structural preconditioning to be carried out in parallel. For the aerodynamic block preconditioner, I reuse the preconditioned Krylov subspace method used for the aerodynamic adjoint. In this case however, I use a fixed number of GMRES iterations, typically between 10 and 20. For the structural block of the preconditioner, I use the matrix factorization of $K$. The pseudocode for the linear adjoint operator for the coupled system is listed in Algorithm 4.

Algorithm 4 Coupled Krylov method linear operator

1: function \text{MULT}(X) \quad \text{// Compute Jacobian-vector product with } X \\
2: \quad (X_A, X_S) \leftarrow X \quad \text{// Extract aerodynamic and structural components} \\
3: \quad Y_A \leftarrow \left(\frac{\partial A}{\partial w}\right)^T X_A \quad \text{// Evaluate diagonal contributions in parallel} \\
4: \quad Y_A \leftarrow Y_A + \left(\frac{\partial F_A}{\partial w}\right)^T TX_S \quad \text{// Add aerodynamic off-diagonal term} \\
5: \quad Y_S \leftarrow Y_S + \left(T^T \left(\frac{\partial X_V}{\partial X_S}\right)^T \left(\frac{\partial A}{\partial X_V}\right)^T X_A\right) \quad \text{// Add structural off-diagonal term} \\
6: \quad Y_S \leftarrow Y_S + \left(T^T \left(\frac{\partial F_A}{\partial X_S}\right)^T TX_S\right) \quad \text{// Add force } \frac{\partial S}{\partial u} \text{ contribution} \\
7: \quad Y \leftarrow (Y_A, Y_S) \quad \text{// Combine aerodynamic and structural components} \\
8: \quad \text{return } Y \\
9: \quad \text{end function}

3.7.3 Memory Requirements

I now consider the additional memory requirements for the fully coupled nonlinear and adjoint solution methods. One advantage of the Gauss–Seidel strategy is that the coupled
solution does not require any significant additional memory. This allows the aerostructural analysis and optimization to inherit the same memory footprint as the stand-alone discipline solvers.

If a Krylov method with a long recurrence relationship is used to solve Equation (2.28) or Equation (3.6), significant additional memory is required. In the current framework, I use restarted FGMRES\( (m) \) (where \( m \) is the size of the Krylov subspace) to solve Equations (2.28) and (3.6). This requires \( 2m \) vectors of size \( \mathcal{R} \) to be stored in memory. For matrix-free disciplinary methods, these vectors require additional memory.

If Newton–Krylov methods are used to solve the disciplinary systems, however, it may be possible to reduce the memory requirement of the individual solvers when a coupled method is used. When our CFD solver is used stand-alone, a preconditioned GMRES\( (s) \) algorithm is used to approximately solve the Newton update equation and tightly solve the adjoint equation. The system is preconditioned using restrictive additive Schwartz and ILU\( (p) \) on each of the subdomains, and it requires \( s \) subspace vectors. For the coupled solution, it is possible to “exchange” some of the Krylov vectors used for discipline preconditioning to the outer Newton–Krylov or Krylov solver.

A fixed number of subspace vectors, \( n \), can be used for either the outer FGMRES solver or the discipline preconditioner. If \( m \) is chosen such that \( m = \lfloor (n - s)/2 \rfloor \), the coupled methods and the Gauss–Seidel methods will use the same memory. In effect, we are transferring computational work (and memory) from the discipline solvers to the coupled solver. This results in more outer iterations, but less expensive inner iterations. The increase in the number of outer iterations also increases the computational cost of evaluating the off-diagonal terms. Eventually, this increasing cost leads to diminishing returns and there is a range of \( s \) values that leads to the best performance. This trade-off is investigated in the following section.

### 3.7.4 Coupled Adjoint Solution Method Comparison

Similar to the comparison of the two nonlinear solution methods, I have performed a performance comparison between the loosely and tightly coupled adjoint solution strategies. For the LBGS solver, I consider two fixed under-relaxation factors, 0.5 and 1.0, as well as three relative tolerances for the aerodynamic adjoint solution, \( \epsilon_{A} = 0.1, 0.25, \text{and} 0.5 \). The same three variants of the CNK solver are considered with inner aerodynamic subspace sizes of 5, 10, and 20. As with the nonlinear solution comparison, the memory requirements are held fixed. The LBGS solver uses GMRES\( (75) \), while the coupled solver uses GMRES\( (s) \) for the aerodynamic preconditioner and FGMRES\((\lfloor (75 - s)/2 \rfloor)\) for the outer subspace. The
results are given in Table 3.3.

Table 3.3: Adjoint solution method comparison

<table>
<thead>
<tr>
<th>Solver</th>
<th>( \theta^{(0)} )</th>
<th>Aitken accel.</th>
<th>Inner size (s)</th>
<th>Outer size (m)</th>
<th>( N_{iter} )</th>
<th>Time (s)</th>
<th>CFD</th>
<th>Mesh</th>
<th>CSM</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>LBGS</td>
<td>0.5</td>
<td>0.10</td>
<td>—</td>
<td>—</td>
<td>21</td>
<td>542.6</td>
<td>9.1</td>
<td>2.1</td>
<td>554.0</td>
<td></td>
</tr>
<tr>
<td>LBGS</td>
<td>0.5</td>
<td>0.25</td>
<td>—</td>
<td>—</td>
<td>22</td>
<td>304.2</td>
<td>9.5</td>
<td>2.3</td>
<td>316.1</td>
<td></td>
</tr>
<tr>
<td>LBGS</td>
<td>0.5</td>
<td>0.50</td>
<td>—</td>
<td>—</td>
<td>23</td>
<td>141.7</td>
<td>10.0</td>
<td>2.3</td>
<td>154.1</td>
<td></td>
</tr>
<tr>
<td>LBGS</td>
<td>1.0</td>
<td>0.10</td>
<td>—</td>
<td>—</td>
<td>17</td>
<td>375.9</td>
<td>7.4</td>
<td>1.7</td>
<td>385.0</td>
<td></td>
</tr>
<tr>
<td>LBGS</td>
<td>1.0</td>
<td>0.25</td>
<td>—</td>
<td>—</td>
<td>16</td>
<td>158.9</td>
<td>7.0</td>
<td>1.6</td>
<td>167.6</td>
<td></td>
</tr>
<tr>
<td>LBGS</td>
<td>1.0</td>
<td>0.50</td>
<td>—</td>
<td>—</td>
<td>21</td>
<td>111.6</td>
<td>9.1</td>
<td>2.1</td>
<td>123.0</td>
<td></td>
</tr>
<tr>
<td>CK</td>
<td>—</td>
<td>—</td>
<td>5</td>
<td>35</td>
<td>120</td>
<td>78.2</td>
<td>53.4</td>
<td>10.6</td>
<td>136.3</td>
<td></td>
</tr>
<tr>
<td>CK</td>
<td>—</td>
<td>—</td>
<td>10</td>
<td>32</td>
<td>58</td>
<td>71.3</td>
<td>25.1</td>
<td>5.1</td>
<td>99.6</td>
<td></td>
</tr>
<tr>
<td>CK</td>
<td>—</td>
<td>—</td>
<td>20</td>
<td>27</td>
<td>40</td>
<td>99.8</td>
<td>17.3</td>
<td>3.5</td>
<td>118.5</td>
<td></td>
</tr>
</tbody>
</table>

The benefits of the monolithic solution method are even more compelling for the adjoint solution. As is the case for the nonlinear solution, all the Gauss–Seidel solutions are able to converge, albeit with some difficulty. For the 1 g flight condition, the under-relaxation parameter of 1.0 yields faster solution times, while for the 2.5 g case, a value of 0.5 is faster. Both cases converge faster with the relatively weak tolerances of only 0.5. Additional accuracy for each approximate solution is unnecessary and only increases the computational cost. The wide range of solution times for the LBGS method indicates a strong dependence on both the forcing tolerances and the under-relaxation factor. All three variants of the coupled Krylov solver converge without issue and are typically significantly faster than the best LBGS combination. The observation regarding the small inner subspace sizes \((s \approx 5)\) made for the aerostructural solution applies to the adjoint solution to an even greater extent. While the reverse-mode mesh-deformation derivatives are computed efficiently, they are more costly than their forward equivalent.
The cumulative time required for the mesh-deformation derivatives represents a much larger proportion of the total solution time in all the coupled solution results. It represents approximately 40% of the overall time for the $s = 5$ case and approximately 15% for the $s = 20$ case. Values of $s = 10$ yielded the best solution times. However, the optimal values of $s$ are problem dependent and may also depend on other solution parameters, such as the level of fill used for the aerodynamic preconditioner. Generally, $s$ values ranging from 10–20 result in fast, robust convergence, a trend that is similar to the aerostructural solution case.

When the best LBGS solution time is compared with the best CK time, we find a 19% reduction for the 1 g load case and a 29% reduction for the 2.5 g case. I conclude that for the type of static aerostructural deformations we expect to find on a subsonic transport aircraft wing, the fully coupled solution methods (nonlinear and adjoint) can yield computational savings in the order of 10–30% with increased robustness.

3.8 Scalability

3.8.1 Parallel Scaling

This section examines the overall parallel scalability of the coupled solution and adjoint techniques. I consider a sequence of three discretization levels, where each level doubles the spatial resolution of the previous level. For the CFD solver each level increases the number of cells by a factor of eight. Since the structural solver uses shell elements, doubling the spatial resolution results in four times the number of elements and approximately four times the number of structural degrees of freedom. Table 3.4 lists the sizes of each level of discretization and the number of processors used for each level.

This weak scaling study attempts to keep the number of degrees of freedom per processor roughly constant. For the CSM problem, near-perfect load balancing is achieved for each mesh level, but this is not the case for the CFD problem. The load imbalance is computed via $\text{Imbalance} = \max (N_i)/N_{\text{even}} - 1$, where $N_i$ is the number of cells on each processor and $N_{\text{even}} = N_{\text{total}}/N_P$ is the number of cells resulting from a perfect balancing. The load imbalance for mesh level-3 indicates that at least one critical processor is assigned 25% more computational work than is required on average, leading to computation times 25% longer than would result from perfect load balancing.

Solutions are generated for a Mach number of 0.85, a fixed angle of attack of $2^\circ$ and a static pressure of 30 kPa, which is close to the 1 g cruise flight condition. The nonlinear equations are solved using the NLBGS method, while the adjoint equations are solved with the CK method. The adjoint solution also evaluates the gradient of the lift with respect to
Table 3.4: Mesh sizes

<table>
<thead>
<tr>
<th>Level</th>
<th>CFD</th>
<th></th>
<th></th>
<th></th>
<th>CSM</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cells</td>
<td>DOFs</td>
<td>$N_p$</td>
<td>Imbalance</td>
<td>Elements</td>
<td>Nodes</td>
<td>DOFs</td>
<td>$N_p$</td>
</tr>
<tr>
<td>1</td>
<td>263,762</td>
<td>1,318,810</td>
<td>8</td>
<td>1.5%</td>
<td>13,512</td>
<td>13,522</td>
<td>81,132</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2,110,096</td>
<td>10,550,480</td>
<td>64</td>
<td>13.1%</td>
<td>54,048</td>
<td>54,030</td>
<td>324,180</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>16,288,076</td>
<td>84,403,840</td>
<td>512</td>
<td>25.2%</td>
<td>216,192</td>
<td>216,118</td>
<td>1,296,708</td>
<td>32</td>
</tr>
</tbody>
</table>

462 design variables, described in Table 3.7. Both the nonlinear and adjoint solutions are converged to relative tolerance $\epsilon_{AS} = \epsilon_{SA} = 10^{-6}$. Figure 3.6 shows the $C_p$ contours for each converged solution as well as the von Mises stress values as a fraction of yield stress on the upper surface of the wing box. A section of the computational meshes near the wing root for each discipline is also provided for visual reference. Comparisons of key aerodynamic and structural performance results are given in Table 3.5.

Table 3.5: Aerostructural scaling results

<table>
<thead>
<tr>
<th>Quantity</th>
<th>Level 1</th>
<th>Level 2</th>
<th>Level 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_L$</td>
<td>0.4871</td>
<td>0.4960</td>
<td>0.4968</td>
</tr>
<tr>
<td>$C_D$</td>
<td>0.0186</td>
<td>0.0121</td>
<td>0.0109</td>
</tr>
<tr>
<td>$C_M$</td>
<td>-0.0250</td>
<td>-0.0224</td>
<td>-0.0216</td>
</tr>
<tr>
<td>Tip displacement (m)</td>
<td>1.54</td>
<td>1.61</td>
<td>1.62</td>
</tr>
<tr>
<td>Lower skin/stringer KS</td>
<td>0.328</td>
<td>0.365</td>
<td>0.491</td>
</tr>
<tr>
<td>Upper skin/stringer KS</td>
<td>0.459</td>
<td>0.492</td>
<td>0.723</td>
</tr>
<tr>
<td>Rib/spar KS</td>
<td>0.367</td>
<td>0.431</td>
<td>0.512</td>
</tr>
<tr>
<td>Nonlinear Gauss–Seidel iterations</td>
<td>16</td>
<td>19</td>
<td>36</td>
</tr>
<tr>
<td>Coupled Krylov adjoint iterations</td>
<td>24</td>
<td>46</td>
<td>117</td>
</tr>
</tbody>
</table>

The general characteristics of the solutions at each mesh level are similar, but a closer investigation reveals several differences. As the CFD mesh is refined, the normal shock on the upper surface of the wing becomes more pronounced. The lift coefficient is predicted reasonably well at all three mesh levels. The drag coefficient is reduced as the mesh is refined due to the reduction of spurious drag and leads to more accurate predictions of the wave drag. As expected from finite element theory, the higher resolution mesh is less stiff, resulting in larger displacements and higher stress levels.

The scalability of the presented methods is demonstrated by breaking down the time required to compute each solution and adjoint. These computations as well as the others in this thesis were performed using Intel Xeon E5540 processors connected with a 4x-DDR non-blocking InfiniBand fabric interconnect [81]. A detailed computational-time breakdown
Chapter 3. Gradient Computation Methodology

(a) Failure parameter (fraction of stress to yield stress) distribution for the upper surface of the wing box

(b) $C_p$ values on upper surface of wing

Figure 3.6: CFD and CSM solution comparison for each mesh level

of the sensitivity analysis performed at the above converged solution is given in Table 3.6. The fraction column is nondimensionalized relative to the aerostructural analysis time for that mesh level.

The aerostructural solution times for all three mesh levels show that the majority of the solution time is taken by the CFD solution. The mesh-deformation scheme comprises only 2–3% of the total solution time. Similarly, the time required for transferring the loads and displacements is less than 1%. We would expect the total solution time for an NLBGS solution to be the sum of the individual disciplinary solution times. However, since a direct solution method is used for the structural solver, the majority of the solution time is due to
Table 3.6: Coupled adjoint timings

<table>
<thead>
<tr>
<th></th>
<th>Level 1</th>
<th></th>
<th>Level 2</th>
<th></th>
<th>Level 3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time (s)</td>
<td>Fraction</td>
<td>Time (s)</td>
<td>Fraction</td>
<td>Time (s)</td>
<td>Fraction</td>
</tr>
<tr>
<td>Aerostructural solution</td>
<td>117.1</td>
<td>1.000</td>
<td>209.7</td>
<td>1.000</td>
<td>897.2</td>
<td>1.000</td>
</tr>
<tr>
<td>CFD solution</td>
<td>112.6</td>
<td>0.962</td>
<td>205.1</td>
<td>0.978</td>
<td>863.2</td>
<td>0.962</td>
</tr>
<tr>
<td>CSM solution</td>
<td>7.24</td>
<td>0.062</td>
<td>20.0</td>
<td>0.095</td>
<td>51.1</td>
<td>0.057</td>
</tr>
<tr>
<td>Mesh deformation</td>
<td>1.8</td>
<td>0.015</td>
<td>2.4</td>
<td>0.011</td>
<td>24.3</td>
<td>0.027</td>
</tr>
<tr>
<td>Coupled adjoint solution</td>
<td>53.2</td>
<td>0.454</td>
<td>112.7</td>
<td>0.537</td>
<td>579.6</td>
<td>0.646</td>
</tr>
<tr>
<td>Setup</td>
<td>42.9</td>
<td>0.366</td>
<td>71.3</td>
<td>0.340</td>
<td>52.6</td>
<td>0.059</td>
</tr>
<tr>
<td>CFD solution</td>
<td>43.7</td>
<td>0.371</td>
<td>90.1</td>
<td>0.430</td>
<td>241.3</td>
<td>0.269</td>
</tr>
<tr>
<td>CSM solution</td>
<td>0.9</td>
<td>0.008</td>
<td>4.0</td>
<td>0.019</td>
<td>19.3</td>
<td>0.022</td>
</tr>
<tr>
<td>Mesh sensitivities</td>
<td>8.9</td>
<td>0.076</td>
<td>20.7</td>
<td>0.097</td>
<td>328.4</td>
<td>0.366</td>
</tr>
<tr>
<td>Total sensitivity</td>
<td>5.4</td>
<td>0.046</td>
<td>5.7</td>
<td>0.027</td>
<td>7.0</td>
<td>0.012</td>
</tr>
</tbody>
</table>

First gradient 0.823 0.913 0.716
Subsequent gradients 0.457 0.573 0.657

the matrix factorization, which is overlapped with the first CFD Gauss–Seidel iteration.

For the coupled adjoint solution, the one-time assembly costs account for the time required to compute the aerodynamic state residual matrix, $\partial A/\partial w$, the spatial residual matrix, $\partial A/\partial X_V$, and the two coupling matrices $\partial F_A/\partial w$ and $\partial F_A/\partial X_S$. This amounts to approximately 35% of the analysis time for the level-2 mesh, but it is quickly amortized over multiple adjoint solutions. Since the stiffness matrix for the structural discipline is symmetric, the original factorization for the structural solution method is reused for the structural adjoint. The bulk of the coupled adjoint solution time is spent in each of the disciplinary solvers, amounting to approximately 35% of the aerostructural solution time. The computational cost for the off-diagonal terms is dominated by the time required to compute $\partial A/\partial u$, listed in Table 3.6 under “Mesh derivatives”. This amounts to 20–30% of the coupled adjoint solution time for the level-1 and level-2 meshes. The time required for the mesh derivatives on the larger mesh is disproportionately large. The culprit is the load balancing of the mesh-deformation algorithm. The mesh-deformation algorithm currently does not support block splitting, which means that once the number of processors exceeds the number of blocks in the mesh (144 in this case), the remaining processors are idle during mesh operations. If the level-3 mesh contained a correspondingly larger number of blocks, this behaviour would not be observed. The total-derivative time includes the calculation of all partial-derivative terms in the total-derivative equation.

It is instructive to examine how the convergence characteristics of the nonlinear aerostruc-
tural solution and linear adjoint solutions change as the size of the computation increases by nearly two orders of magnitude. Figure 3.7 shows the nonlinear convergence and Figure 3.8 gives the convergence for the adjoint.

![NLBGS convergence for the three mesh levels](image)

Figure 3.7: NLBGS convergence for the three mesh levels

![Convergence history for lift adjoint using the coupled Krylov adjoint solver](image)

Figure 3.8: Convergence history for lift adjoint using the coupled Krylov adjoint solver

For both the level-1 and level-2 meshes, convergence to $1 \times 10^{-6}$ is achieved in approximately the same number of iterations (16 and 19, respectively), while engineering accuracy ($1 \times 10^{-3}$) for the lift-to-drag ratio is achieved in approximately 10 iterations. However, the level-3 solution requires 36 iterations to converged to $1 \times 10^{-6}$, and engineering accuracy is not achieved until iteration 20.

Moving from mesh level 2 to 3, the number of NLBGS iterations doubles, but the wall
time increases by a factor of 4.3. Since a fixed aerodynamic forcing tolerance is used, each iteration is also more costly on the larger mesh. The performance for the coupled adjoint solution is similar. In this case, the cost of each iteration is similar for all three mesh levels, such that the number of iterations required for convergence in Figure 3.8 is representative of the overall solution time. All of the adjoint solutions use the same aerodynamic preconditioning settings—ILU(1) and additive Schwartz(1)—resulting in nearly constant memory usage across the mesh levels. For mesh level 3, faster convergence times can be achieved by using stronger preconditioning, which reduces the condition number of the preconditioned system.

### 3.8.2 Design Variable Scalability

The main advantage of using the coupled adjoint method to compute the gradients of the functions of interest is that the computational cost is theoretically independent of the number of design variables. However, as described in Section 3.1, careful implementation of the partial-derivative terms $\frac{\partial I}{\partial x}, \frac{\partial A}{\partial x}, \frac{\partial S}{\partial x}$ in the total-derivative equation (3.7) is required to ensure that the computational cost is truly independent of the number of design variables.

I now consider the time required to compute the gradient of $C_L$ with respect to thousands of design variables. The design variables are distributed according to Table 3.7 and contain both global geometric variables and local variables.

#### Table 3.7: Design variables

<table>
<thead>
<tr>
<th>Global variables</th>
<th>Aerodynamic variables</th>
<th>Structural variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Qty</td>
<td>Description</td>
</tr>
<tr>
<td>Span</td>
<td>1</td>
<td>Angle of attack</td>
</tr>
<tr>
<td>Sweep</td>
<td>1</td>
<td>Tail rotation</td>
</tr>
<tr>
<td>Chord</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Twist</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Shape</td>
<td>1600</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total 1610</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Total 2</td>
<td></td>
</tr>
</tbody>
</table>

I compare the computational time required to compute the gradient using the coupled adjoint method and using first-order finite differencing. The level-2 discretization is used and the computational time is normalized by the time required for a single aerostructural solution. The results are shown in Figure 3.9.

We expect the cost of finite differencing to be linearly dependent on the number of design variables. However, the slope is not equal to one but is significantly lower, since the solution
for each design-variable perturbation uses the previous solution as a starting point, and it is closer to the converged state than is a uniform-flow field solution. For each additional design variable, finite differencing requires a time equivalent to 23% of an aerostructural solution, resulting in a slope of 0.23.

The coupled adjoint method exhibits an extremely small slope. The main contributor to this slope is the design-variable-dependent load transfer, which requires a synchronous data transfer for each geometric design variable. Nevertheless, each additional design variable requires only 0.005% of the aerostructural solution time.

It is worth comparing the current results with the previous work of Martins et al. [87]. In that work, the coupled adjoint cost was found to scale with the number of design variables according to $3.4 + 0.01 N_x$. The present method lowers the scaled cost of computing an adjoint vector from 2.4 to 0.68 times the cost of an aerostructural solution. This is primarily due to the elimination of the finite differencing that was used to compute the off-diagonal coupled adjoint terms. Additionally, the slope has been reduced by over two orders of magnitude, from $1 \times 10^{-2}$ to $5 \times 10^{-5}$. This is achieved by eliminating the use of finite-difference derivatives in the total-derivative equation (3.7).

The coupled adjoint implementation described in this chapter exhibits extremely good design variable scaling. The coupled computational cost can be considered practically inde-
dependent of the number of design variables, and it is now feasible to compute gradients with respect to thousands of design variables.

3.9 Conclusions

In this chapter I first describe the derivative analysis of each of the main components of the aerostructural analysis framework: geometry modification, aerodynamic solver, mesh manipulation and structural solver. For the aerodynamic sensitivity, I presented a new approach for computing the residual matrices using forward-mode AD which is both efficient from a computational standpoint and maintainable from a software perspective. Mesh sensitivities of the hybrid algebraic linear-elasticity method are evaluated using reverse-mode AD and a mesh adjoint. The efficient computation of mesh sensitivities is the key enabler for the coupled Krylov (CK) method.

I developed the coupled adjoint method for the aerostructural equations that significantly improves the accuracy and efficiency relative to previous work. By eliminating the computational expense of the finite differences previously used for the coupled adjoint off-diagonal terms, I am able to solve much larger coupled adjoint systems in a scalable way. The dependency of the computational cost on the number of design variable was reduced to \( O(10^{-5}) \), which can be considered negligible. The elimination of finite differences dramatically improved the accuracy: when compared against the complex-step derivative approximation, and excellent agreement was demonstrated, with relative differences of \( O(10^{-5}) \).

A traditional linear block Gauss–Seidel method was compared with a new fully coupled Krylov method. I showed that the fully coupled solution method is approximately 30\% faster than the linear block Gauss–Seidel method with similar memory requirements.
Chapter 4

Aircraft Design Optimization

Problem Formulation

4.1 Multidisciplinary Trade-offs

The choice of objective function has a significant impact on the optimized aerostructural design. The objective function must be carefully chosen to accurately reflect the performance requirements for a particular aircraft. In many ways, this is a much more challenging problem for aerostructural analysis than it is for either aerodynamic or structural analysis separately. As an example, consider a common multidisciplinary objective, the Breguet range equation:

\[
R = \frac{V}{c_T} \frac{C_L}{C_D} \ln \left( \frac{W_1}{W_2} \right). \tag{4.1}
\]

Here, \( R \) is the range, \( V \) is the flight speed, \( c_T \) is the thrust-specific fuel consumption, and \( W_1, W_2 \) (Equation 5.2) are the initial and final cruise weights, respectively. For the purposes of the following analysis, \( W_2 \) can be considered the aircraft empty weight, since \( \frac{\partial W_2}{\partial W} = 1 \) (where, \( W \) is the aircraft structural weight). If the range equation is used as an objective for aerodynamic optimization, the ratio of the cruise weights remains fixed, and assuming constant cruise Mach number and thrust specific fuel consumption, the range equation reduces to an \( L/D \) maximization or a lift-constrained drag minimization. With this simplification, the design range of the aircraft is irrelevant. Similarly, if we consider only a structural optimization, the only directly relevant metric is the weight ratio, and the optimization becomes a stress-constrained weight minimization.

In both cases, the single discipline optimization problems lead to simplified formulations and, critically, ignore the magnitude of the terms that are assumed constant. An aerostruc-
tural optimization, however, can manipulate the $L/D$ and $\ln (W_1/W_2)$ terms simultaneously, considering the effect of one on the other, and it is critical that the trade-off between these terms reflect the intent of the designer.

A quantitative analysis can be performed to determine the trade-off between drag and weight reduction to answer the question: how many kilograms of structural mass is equivalent to one drag count, in terms of reducing a given objective function? I define $\gamma$ as the partial derivative of the objective function with respect to the drag coefficient $C_D$, and $\beta$ as the derivative of the objective function with respect to the final cruise weight $W_2$. For the Breguet range equation, the linearized objective and the partial derivatives are:

$$R = \gamma C_D + \beta W_2$$

(4.2)

$$\gamma = \frac{\partial R}{\partial C_D} = -\frac{V}{c_T} \frac{C_L}{C_D} \ln \left(\frac{W_1}{W_2}\right)$$

(4.3)

$$\beta = \frac{\partial R}{\partial W_2} = -\frac{V}{c_T} \frac{C_L}{C_D} \frac{1}{W_2}.$$ 

(4.4)

The ratio of these derivative expressions, $\gamma/\beta$ is

$$\frac{\gamma}{\beta} = \frac{\partial W_2}{\partial C_D} = \frac{W_2}{C_D} \ln \left(\frac{W_1}{W_2}\right).$$

(4.5)

The last expression shows that the weight-drag trade-off scales with the final cruise weight, the natural logarithm of the weight ratio, and is inversely proportional to the cruise drag coefficient. Unlike the single discipline optimization cases where a simple reduction of drag or weight was sufficient, this multidisciplinary objective requires knowledge of the aerodynamic performance ($C_D$) and structural performance ($\ln (W_1/W_2)$) to determine the correct multidisciplinary trade-offs. This places additional burden on the analysis, as all components of the drag and weight must be included to achieve the correct aerostructural trade-offs, even if they are not modelled explicitly.

In the results presented in the following chapter, I consider the minimization of two objectives: takeoff gross weight (TOGW) and fuel burn. The Breguet range equation is used to evaluate both objectives. For a fixed desired range, $R$, the weight at the start of cruise, $W_1$, is determined and fuel burn, $W_f$ is then $W_1 - W_2$. The optimization problems only consider the fuel burn during a single cruise segment and ignore the fuel burn during take-off, climb, and descent. For this reason, we assume that the initial cruise weight is equal to $W_1$. Using a similar linearization analysis to the one we performed for the range objective
and applying it to the TOGW objective we find:

\[ I = W_1 = \text{TOGW} \quad (4.6) \]

\[ \gamma = \frac{\partial I}{\partial C_D} = e^{C_T R_{CD} / V C_L} W_2 \left( \frac{C_T R}{V C_L} \right) = \frac{W_1}{C_D} \ln \left( \frac{W_1}{W_2} \right) \quad (4.7) \]

\[ \beta = \frac{\partial I}{\partial W_2} = e^{C_T R_{CD} / V C_L} = \frac{W_1}{W_2} \quad (4.8) \]

\[ \frac{\gamma}{\beta} = \frac{W_2}{C_D} \ln \left( \frac{W_1}{W_2} \right). \quad (4.9) \]

This result is identical to that of the range objective. Therefore, to a first order approximation, maximizing range for a fixed TOGW is equivalent to minimizing TOGW for a fixed range.

For the fuel burn objective, we obtain:

\[ I = W_1 - W_2 = W_f \quad (4.10) \]

\[ \gamma = \frac{\partial I}{\partial C_D} = e^{C_T R_{CD} / V C_L} W_2 \left( \frac{C_T R}{V C_L} \right) = \frac{W_1}{C_D} \ln \left( \frac{W_1}{W_2} \right) \quad (4.11) \]

\[ \beta = \frac{\partial I}{\partial W_2} = e^{C_T R_{CD} / V C_L} - 1 = \frac{W_1}{W_2} - 1 \quad (4.12) \]

\[ \frac{\gamma}{\beta} = \frac{W_1 W_2}{C_D (W_1 - W_2)} \ln \left( \frac{W_1}{W_2} \right). \quad (4.13) \]

The fuel burn objective sensitivity differs from the TOGW sensitivity by a factor of \( W_1 / (W_1 - W_2) = W_1 / W_f \) which is the reciprocal of the fuel fraction. Since the fuel fraction is always less than one, the \( \gamma / \beta \) ratio will always be higher than the ratio for TOGW. A higher \( \gamma / \beta \) indicates one drag count is equivalent to a larger mass. Thus, a higher \( \gamma / \beta \) value favours designs that are more biased towards aerodynamic performance, where weight is more readily increased to reduce the drag. Lower \( \gamma / \beta \) values, on the other hand, favour structural performance, where structural weight reduction is prioritized over drag. Using the data in Table 5.1, we can produce estimates for the objective sensitivities as well as \( \gamma / \beta \).

The fuel fraction for the CRM aircraft, described in Chapter 5, at design payload and range is approximately 0.37 and thus the ratio of \( \gamma / \beta \) for the two objectives is 1/0.37 = 2.69. With the TOGW objective, 1 drag count is equivalent to approximately 313 kg, but for the fuel burn objective 1 drag count is equivalent to 790 kg. These values, however, are only linearizations and do not take into account that a reduction in \( W_2 \) further decreases TOGW and fuel burn when the weight calculation recomputed. The effect of the multidisciplinary
Chapter 4. Aircraft Design Optimization Problem Formulation

Table 4.1: Objective sensitivities

<table>
<thead>
<tr>
<th></th>
<th>TOGW</th>
<th>Fuel burn</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\gamma$ = $\frac{\partial I}{\partial C_{D_{\text{count}}}}$</td>
<td>497.0 kg</td>
<td>497.0 kg</td>
</tr>
<tr>
<td>$\beta$ = $\frac{\partial I}{\partial W_2}$</td>
<td>1.59</td>
<td>0.59</td>
</tr>
<tr>
<td>$\gamma/\beta$ = $\frac{\partial W_2}{\partial C_{D_{\text{count}}}}$</td>
<td>312.6 kg</td>
<td>790.2 kg</td>
</tr>
</tbody>
</table>

objective on the aerostructural optimization is discussed in Chapter 5.

4.2 Multi-point Objective

High-fidelity aerodynamic shape optimization is capable of producing wing shapes that exhibit near optimal induced drag and almost zero wave drag at a specific operating point. Unfortunately, single point optimized designs exhibit poor performance at other equally important on-design operating conditions. To help alleviate this problem, I use a multipoint approach: the optimization considers several operating conditions simultaneously to ensure an even performance improvement across all operating conditions.

Each of the multipoint analyses must be combined to form a composite objective before optimization can proceed. The simplest and most often used approach [111, 46, 112, 113] is a weighted sum, where the weights, $w_i$, are chosen by the designer:

$$ I = \sum_{i}^{N} w_i I_i $$  \hspace{1cm} (4.14)

For the optimizations presented in the next Chapter, I use $N = 5$ and choose a uniform weights corresponding to $W_i = 1/N$.

4.3 Multi-Mission, Multi-point Objective

The objective linearizations perform in Section 4.1 only considered the design range of the CRM aircraft. Figure 4.1 shows the same calculation over all analysis ranges up to and including the design range.

It is clear that the analysis range used to form the objective has a significant impact on the weight-drag trade and thus on the resulting “optimal” design. To investigate this issue further, it is instructive to know how the aircraft is actually used in practice to better...
understand how the multidisciplinary weight-drag trade should be made.

The Bureau of Transportation Statistics (BTS) [114], contains a large database of commercial aircraft flights broken down by aircraft time. Since the overall dimensions of the CRM is very similar to the Boeing 777-200ER, I used the BTS data to compile a list of domestic flights within the United States, as well as international flights to and from the United States. Using this data, we can plot the 2D histogram of the payload and range values from over 100,000 actual flights from 2011, as shown in Figure 4.2. It is immediately clear the nominal design range of the 777-200ER is not representative of how the aircraft is actually used in practice.

Motivated by this observation, I collaborated with Rhea Liem [116] to determine how this real-world data could be incorporated into a multi-point optimization to better match the multidisciplinary trade-off with the actual operation of the aircraft. The method for incorporating this data into a high-fidelity simulation is as follows:

- A set of \( N \) sample locations are chosen in the cruise operational envelope to construct a Kriging meta-model [117, 118] of the \( C_L \), \( C_D \) and \( C_M \).

- The fuel burn for a large representative sample containing approximately 500 missions are analyzed using a numerical integration technique. Since the metamodel is inexpensive to evaluate, analyzing hundreds of missions is tractable.

- The sensitivity of the chosen objective function (e.g. fuel burn), is computed with respect to the chosen sample points. These sensitivities provide the weighting for the multipoint analysis.

Further information and results from this approach can be found in Liem et al. [116].
Figure 4.2: Histogram of 101,059 flights for Boeing 777-200ER aircraft during 2011 using data from the Bureau of Transportation Statistics [114]. The payload-range envelope of the Boeing 777-200ER is provided for reference [115].

4.4 Conclusions

In this chapter I have presented several objectives for aerostructural optimization. To correctly simulate the multidisciplinary trade-offs, it is critical to consider all components of weight and drag — a condition not required by single discipline objectives. I present a method for determining weight/drag sensitivity by evaluating sensitivities of an objective function function with respect to the final aircraft empty weight, $W_2$ and aerodynamic drag, $C_D$. The weight/drag trade-off for both fuel burn and takeoff gross weight objectives increases with mission range and are related by the inverse of the fuel fraction. The weight/drag trade-off for the fuel burn objective is always higher, resulting in more aerodynamically-biased designs than would be obtained using the TOGW objective. Finally, a multi-point multi-mission formulation incorporating real-world operational data is described.
Chapter 5

Aerostructural Design Optimization

This chapter presents results obtained using the solution methods described in Chapter 2 and the coupled adjoint derivative method described in Chapter 3. I present multi-point aerostructural optimization results using both a fuel burn objective and a takeoff gross weight objective.

5.1 Problem Description

The baseline geometry for the optimizations is the NASA common research model (CRM) wing-body-tail configuration, which contains the design features typical of a transonic, wide-body, long-range aircraft [119]. The configuration has been carefully designed and optimized to exhibit good aerodynamic performance across a range of Mach numbers and lift coefficients [119]. This is an excellent aerodynamic starting point for the multidisciplinary optimization, since little additional improvement can be made from isolated aerodynamic design modifications alone.

Since the CRM geometry only defines the OML shape, I created a structural model using *pyLayout* that conforms to the wing OML and is representative of a modern airliner structural wingbox. I only consider the structural model of the wing as this is where the aerodynamic and structural disciplines are most closely coupled. A view of the CFD surface mesh and the structural model is shown in Figure 5.1.

In addition to the OML and the wing structural model, some supplemental information is required to formulate the aircraft design optimization problem. Specifically, we need to know the design mission and payload, the operational empty weight (OEW) and maximum takeoff weight (MTOW). Since the overall dimensions of the CRM are very similar to the Boeing 777-200ER, I use the publicly available documentation [115] to obtain the additional
5.2 Design and Maneuver Conditions

The operating conditions used for the optimizations are now described. To evaluate the TOGW and fuel burn objectives, the weight at the beginning of cruise is estimated using the Breguet range equation:

\[ R = \frac{V}{c_T C_D} \frac{C_L}{C_D} \ln \left( \frac{W_1}{W_2} \right). \]  

(5.1)

The final cruise weight, \( W_2 \), is computed according to

\[ W_2 = W + \text{Fixed Weight} + \text{Reserve Fuel Weight} + \frac{A}{A_{\text{ref}}} \times \text{Secondary Wing Weight} + \text{Payload}, \]  

(5.2)

where \( W \) is the weight of the primary wing structure given based on the volume of the structural finite element model, \( A \) is the projected wing area and \( A_{\text{ref}} \) is the initial projected...
Table 5.1: CRM specifications

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cruise Mach number</td>
<td>0.85</td>
<td>-</td>
</tr>
<tr>
<td>Cruise lift coefficient</td>
<td>0.5</td>
<td>-</td>
</tr>
<tr>
<td>Initial cruise altitude</td>
<td>35 000</td>
<td>ft</td>
</tr>
<tr>
<td>Span</td>
<td>58.6</td>
<td>m</td>
</tr>
<tr>
<td>Aspect ratio</td>
<td>9.0</td>
<td>-</td>
</tr>
<tr>
<td>Reference wing area</td>
<td>383.7</td>
<td>m²</td>
</tr>
<tr>
<td>Sweep (leading edge)</td>
<td>37.4</td>
<td>°</td>
</tr>
<tr>
<td>Maximum take-off weight (MTOW)</td>
<td>298 000</td>
<td>kg</td>
</tr>
<tr>
<td>Operational empty weight</td>
<td>138 100</td>
<td>kg</td>
</tr>
<tr>
<td>Design range</td>
<td>7 725</td>
<td>nm</td>
</tr>
<tr>
<td>Design payload</td>
<td>34 000</td>
<td>kg</td>
</tr>
<tr>
<td>Reserve fuel</td>
<td>15 000</td>
<td>kg</td>
</tr>
<tr>
<td>Initial wing weight</td>
<td>29 200</td>
<td>kg</td>
</tr>
<tr>
<td>Secondary wing weight</td>
<td>8 000</td>
<td>kg</td>
</tr>
<tr>
<td>Fixed weight</td>
<td>100 900</td>
<td>kg</td>
</tr>
<tr>
<td>Thrust specific fuel consumption (c_T)</td>
<td>0.53</td>
<td>lb/(lbf·h)</td>
</tr>
</tbody>
</table>

wing area. The last term is used to account for the weight of fasteners and other wing parts that scale with the wing area. I assume that the aircraft climbs continuously as fuel is consumed to maintain constant \(L/D\) ratio and Mach number. The only aerodynamic input to Breguet range equation is the overall lift to drag ratio of the aircraft.

As discussed in the previous chapter, I only consider multi-point aerostructural optimizations that consider the average performance over multiple flight conditions to obtain more robust designs. A total of five cruise operating conditions are used, centered about the nominal design operating design condition, which is \(M = 0.85, C_L = 0.5\). The first flight condition, matches the CRM design condition closely at the chosen flight altitude. The second and third design points use the same altitude but slightly lower and higher Mach numbers, respectively. The final two flight conditions use the design Mach number of 0.85, but at \(\pm 1000\) ft of altitude, which has the effect of varying the flight \(C_L\).

Two separate maneuver conditions are considered: a 2.5 g symmetric pull-up maneuver and a 1.3 g acceleration due to a gust. The first one represents 2.5 g limit load for the wing structure. The second maneuver condition is a surrogate for a gust load on the aircraft at a cruise condition [120]. The purpose of this maneuver condition is explained in the discussion of the optimization problem constraints, Section 5.3.1.

A final analysis point is also required for the estimate of the aircraft’s static margin. Using static aeroelastic analysis, an estimate of the aircraft’s static margin in the deformed
configuration can be obtained using the following formula:

\[ K_n = \frac{-C_{M\alpha}}{C_{L\alpha}}. \]  \hspace{1cm} (5.3)

Further details on approximating static and dynamic stability derivatives for high fidelity CFD optimization can be found in Mader et al. [29]. The derivatives \( C_{M\alpha} = \frac{\partial C_M}{\partial \alpha} \) and \( C_{L\alpha} = \frac{\partial C_L}{\partial \alpha} \) are estimated with finite differencing and a step size of \( \Delta \alpha = 0.1^\circ \). Since the change in \( C_L \) and \( C_M \) with respect to the angle of attack is nearly linear in the range under consideration, this relatively large finite difference step can be used without significant truncation error. The analysis for cruise condition 1 is used for the baseline value, and the additional stability point provides the perturbed value to complete the derivative calculations.

<table>
<thead>
<tr>
<th>Group</th>
<th>Identifier</th>
<th>Mach</th>
<th>Altitude, (ft)</th>
<th>Load Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cruise</td>
<td>1 0.85</td>
<td>35000</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 0.84</td>
<td>35000</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 0.86</td>
<td>35000</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 0.85</td>
<td>34000</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 0.85</td>
<td>36000</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>Maneuver</td>
<td>1 0.86</td>
<td>20000</td>
<td>2.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 0.85</td>
<td>32000</td>
<td>1.3</td>
<td></td>
</tr>
<tr>
<td>Stability</td>
<td>1 0.85</td>
<td>35000</td>
<td>1.0</td>
<td></td>
</tr>
</tbody>
</table>

### 5.3 Design Variables

The two optimizations presented use 462 design variables to parametrize the aerodynamic shape and structural sizing. As is typical in a multidisciplinary analysis, the design variables can be divided into global variables, which directly affect more than one discipline, and local variables, which only affect a single discipline. Table 5.3 lists all the optimization variables.

The FFD geometry manipulation method described in Chapter 2 is used to make geometric perturbations to the baseline configuration. The geometric design variables, as well as the CFD mesh discretization used for the optimizations are shown in Figure 5.2. Chords are modified at the root, Yehudi break, near tip and tip sections, and the remaining sections are linearly interpolated. Five twist angles are defined similarly, and interpolated linearly in the spanwise direction. Two sweep variables are specified: the first sweep variable ex-
Table 5.3: Design variables

<table>
<thead>
<tr>
<th>Description</th>
<th>Qty</th>
<th>Description</th>
<th>Qty</th>
<th>Description</th>
<th>Qty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Span</td>
<td>1</td>
<td>Angle of attack</td>
<td>1</td>
<td>Upper skin</td>
<td>54</td>
</tr>
<tr>
<td>Sweep</td>
<td>2</td>
<td>Tail rotation</td>
<td>1</td>
<td>Lower skin</td>
<td>54</td>
</tr>
<tr>
<td>Chord</td>
<td>4</td>
<td>Upper stringers</td>
<td>54</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Twist</td>
<td>5</td>
<td>Lower stringers</td>
<td>54</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shape</td>
<td>160</td>
<td>Ribs</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rib stiffeners</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Spars</td>
<td>36</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>172</td>
<td>Total</td>
<td>2</td>
<td>Total</td>
<td>288</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Grand total</td>
<td>462</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The variables controlling the thickness of the structure’s skin are grouped in a grid of 18 stations in the spanwise direction and 3 stations in the chord-wise directions, resulting in 54 variables for each of the upper and lower skins, respectively. The stringer variables are grouped in the same way. The ribs and rib stiffeners each have 18 variables in the spanwise direction as do each of the leading and trailing edge spars. Each of the five cruise conditions and two maneuver conditions have an independent angle of attack and tail rotation angle to provide the required degrees-of-freedom to meet both the lift and pitching moment constraints. Finally, the mean aerodynamic chord (MAC) and center of gravity location ($X_{CG}$) are target variables, each coupled with a consistency constraint using the individual discipline feasible (IDF) MDO architecture [21] that simplifies the implementation. The initial values of geometric and aerodynamic design variables are chosen to regenerate exactly the original CRM geometry.

Since there was no baseline structural sizing, a reasonably efficient initial structure is required to make meaningful comparisons between the initial and optimized designs. The
design of the initial structure is generated by performing a stress-constrained weight minimization optimization. Fixed loads are derived from an aerostructural analysis and the same six KS stress constraints described in Section 5.3.1 are used to ensure a feasible solution. The resulting optimal structural design, is then used as the initial design for the two aerostructural optimization problems. The result of this structural optimization can be seen in Figures 5.13c and 5.13d. Figure 5.3 shows the internal layout of the structure as well as the grouping of the structural design variables.
5.3.1 Design Constraints

For simplicity, I divide the constraints into three groups: geometric and target constraints, aerodynamic constraints, and structural constraints, as shown in Table 5.4. The first two

Table 5.4: Optimization constraints

<table>
<thead>
<tr>
<th>Geometric/target constraints</th>
<th>Aerodynamic constraints</th>
<th>Structural constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Description</td>
<td>Description</td>
</tr>
<tr>
<td>Qty</td>
<td>Description</td>
<td>Qty</td>
</tr>
<tr>
<td>( t_{LE}/t_{LE_{init}} ) \geq 1.0</td>
<td>Cruise: ( L - W = 0.0 )</td>
<td>5</td>
</tr>
<tr>
<td>( t_{TE}/t_{TE_{init}} ) \geq 1.0</td>
<td>Cruise: ( C_{m_y} = 0.0 )</td>
<td>5</td>
</tr>
<tr>
<td>A/A_{init} \geq 1.0</td>
<td>Maneuver: ( L - W = 0.0 )</td>
<td>2</td>
</tr>
<tr>
<td>V/V_{init} \geq 1.0</td>
<td>Maneuver: ( C_{m_y} = 0.0 )</td>
<td>2</td>
</tr>
<tr>
<td>t_{TE \text{ Spar}} \geq 0.20</td>
<td>Static margin: ( K_n ) \geq 0.15</td>
<td>1</td>
</tr>
<tr>
<td>t_{Tip}/t_{Tip_{init}} \geq 0.5</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>MAC - MAC* = 0.0</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>( X_{CG} - X_{CG}^* ) = 0.0</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>36</td>
</tr>
<tr>
<td>Grand total</td>
<td></td>
<td>57</td>
</tr>
</tbody>
</table>
leading edge. The thickness constraints at 97.5% chord prevent the upper and lower surfaces from crossing over near the sharp trailing edge. The projected wing area is constrained to be no less than the initial area ensuring that the takeoff field length and approach speeds are not changed significantly. Even with a complete wingbox structural model, a minimum fuel volume constraint is required. Only the volume inside the spar-box is computed to ensure that the optimized wing is able to carry at least the same amount of fuel as the initial design.

Several additional thickness constraints are also enforced. Minimum trailing edge spar height constraints, $t_{TE\text{ spar}}$, are used over the outboard section of the wing to ensure adequate vertical space is available to attach the actuation devices for the flaps and ailerons. The tip thickness constraint, $t_{tip}$, is used to ensure the optimization does not produce an unrealistically thin wing tip, that would otherwise be sized by manufacturing constraints.

Each cruise and maneuver condition enforces equality constraints on the lift and pitching moment coefficient, so that all aerostructural solutions are trimmed. The static margin of the first cruise condition is constrained to be greater than 15%. The reference point (assumed center of gravity location) for the moment computation is taken to be at 25% of MAC, which changes with the planform design variables. By including a pitching moment constraint, as well as a constraint on the full configuration static margin, the optimization is allowed to trade drag reduction from aft-loaded supercritical profiles with the induced drag penalty required to trim the configuration.

Lastly, we must constrain the stresses on the structure. The structural model used for the optimizations consists of over 50,000 second-order MITC shell elements [121]. Individually constraining the stress in each of these elements would require the solution of the corresponding number of coupled adjoint vectors, which would incur a prohibitive computational cost. To dramatically reduce the required number of coupled adjoint solutions, I use the Kreisslmeier–Steinhauser (KS) constraint aggregation technique [85, 108]. Each maneuver condition uses three KS functions: the first for the lower wing skin and stringers, the second for the upper wing skin and stringers, and the third for the spars, ribs and rib stiffeners. The compression members in the upper wing skin and stringers are assumed to be manufactured from Aluminum 7050 with a maximum allowable stress of 300 MPa. The remainder of the primary wing structure is assumed to be manufactured from Aluminum 2024 with a maximum allowable stress of 324 MPa.

For the first maneuver condition, the maximum von Mises stress must be below the limiting stress, which requires the three KS functions to be less than 1. The second maneuver constraint is used to emulate a fatigue criteria for the lower wing skin and stringers, which
are under tension during normal loading. The stress on the lower wing skin and stringers must be below 138 MPa for the 1.3 g load condition, corresponding to an upper limit of 0.42 for the KS constraint. The remaining two KS functions for the 1.3 g maneuver condition retain the maximum KS value of 1.

Figure 5.4: Visualization of geometric constraints for optimization

5.4 Optimization Algorithm

The introductory discussion in Chapter 3 described the motivation for the use of gradient-based optimization for problems with large numbers of design variables. The optimizer chosen for the two optimizations considered is SNOPT [100]. This optimizer is based on the Sequential Quadratic Programming (SQP) approach and is well suited to large-scale, constrained nonlinear optimization problems. The coupled adjoint method is critical in enabling effective gradient-based optimization.

5.5 Computational Resources

The two optimizations are performed on a massively parallel supercomputer [81]. Each optimization function evaluation requires the solution of eight aerostructural solutions: five for the cruise conditions, two for the maneuver conditions, and one for the stability point. To reduce the wall time required for the optimizations, each of these parallel analyses are
carried out concurrently. The cruise conditions and stability condition use a 2.1 million cell CFD mesh and 52 aerodynamic processors while the CSM solver uses a 324,180 DOF discretization with 4 processors. The two maneuver conditions use a smaller CFD mesh with 1.2 million cells and the same structural discretization. The maneuver conditions use 45 CFD processors and 4 CSM processors. To ensure good overall computational efficiency for the optimization problem, I try to balance the time required to run each concurrent analysis, which should be approximately equal to avoid idle processes. Load balancing is complicated by fact that the each cruise analysis requires the computation of three adjoint vectors (lift, drag, and moment), the maneuver conditions require five adjoint vectors, (lift, moment, and three KS functions) and the stability condition requires only two (lift and moment). The optimization algorithm itself is serial, and does not contribute significantly to the overall computational cost. The number of processors used for each aerostructural computation is shown in Figure 5.5. The total number of processors used for each optimization is $(49 \times 2) + 56 + 1 = 435$. The single additional processor in the far right of the figure is used for the skin friction computation. The TOGW optimization required 34.5 hours of wall time, while the fuel burn optimization required 36.5 hours.

### 5.6 Results

I now examine the optimization results from the two optimizations. The results of each optimization are presented concurrently to compare the effects of considering different objective functions.

First I examine the optimization convergence history for each problem, which is shown in Figure 5.6. Feasibility is the maximum constraint violation, and is a measure of how closely
the nonlinear constraints are satisfied. Optimality refers to the norm of the augmented Lagrangian function used in SNOPT to measure the convergence of the optimization problem. Additionally, the evolution of each objective is shown. SNOPT performs 150 major optimization iterations, which results in approximately two orders of magnitude reduction in the optimality criterion. A lower optimality indicates that the design point satisfies the Karush–Kuhn–Tucker (KKT) conditions, and results in better local minima. I have found that these complex aerostructural optimization problems converge slowly and only produce weak local minima, that is, the KKT conditions are only approximately satisfied. However, the majority of the objective improvement appears to have been achieved and significant additional computational effort is required for further improvements. The optimizations are sufficiently converged to draw overall conclusions and to examine the design trade-offs in the resulting designs.

![Optimization convergence histories for each optimization](image)

**Figure 5.6:** Optimization convergence histories for each optimization

The overall planform changes in the optimized designs are now compared. The chord, sweep and span variables — 7 in total — control the overall planform, including the wing area. Figure 5.7 compares the new planforms with the original design. Despite the absence of bounds on the span, sweep and area, the optimized designs have reasonable planforms, which indicates that the integration of the structural design and constraints successfully prevented the aerodynamic shape optimization from obtaining unrealistic designs. The TOGW optimization does not change the planform area but increases the span from 58.7 m to 62.3
m, and the aspect ratio from 9.0 to 10.1. The most striking feature, however, is the addition of a raked wingtip. While the planform design variables are chosen to facilitate the exploration of this design feature by the optimization procedure, it only appears with the TOGW objective. In contrast, the fuel burn objective results in a significant increase in the wing span and wing area. For this case, the span has increased from 58.7 m to 72.7 m, the exposed planform area has increased by 16.2% and the new aspect ratio is 11.9. Looking closely at the tip, the leading edge is almost straight right to the tip and there is a slight tapering of the chord near the tip. The leading sweep of the TOGW optimization remained essentially unchanged from the baseline except for the raked tip. The fuel burn minimization results in over 5° lower sweep than the original design. This reduction in sweep is required to enable the large span extension without incurring too large of a weight penalty. The wave drag penalty for lowering the sweep angle is minimized through small shape modifications that can only be achieved with a large number of airfoil shape design variables. Additionally, the minimum fuel burn wing is thinner than either the original or TOGW optimum, which further limits the negative aspects of decreasing the sweep angle. These complex interactions can only be predicted with a high-fidelity aerostructural model.

The evolution of the pressure drag, which consists of induced drag and wave drag, skin friction drag, and the weight of the primary wing structure are now considered. The data is shown in Figure 5.8. Here, the vastly different trade-offs made by each optimization to improve the objective function are clear. The TOGW optimization reduces the primary structural weight by 20% while simultaneously reducing the pressure drag by 20%. The skin friction drag is increased slightly by 2.2%, due primarily to the overall increases in the t/c ratio.

The minimum fuel burn design, on the other hand, requires a significantly heavier structure to support the large span increase. In this case, the increase in the skin friction drag is due primarily to the increase in the wing surface area, as the thickness to chord ratios decreased slightly. Despite the increased primary wing structural weight of 39.4% and increased skin friction drag of 5.7%, the total drag coefficient is lower for the fuel burn objective.
Figure 5.7: Planform view with $C_p$ contours for both optimum designs for cruise condition 1; the initial design, on the left, is included for reference.
An examination of the spanwise thickness-to-chord distribution and twist distributions — shown in Figure 5.9 — sheds additional light on the trade-offs for each optimization. The large difference between the optimized t/c ratios of the two optimized designs is immediately apparent. For the TOGW optimized design, the t/c ratios have increased for the majority of the span, peaking at a maximum of 14.8% near the half span position. For the fuel burn optimization, an overall reduction in thickness to chord ratio is observed with the exception of slight increases near and slightly outboard of the Yehudi break. There are two contributing factors to the increased primary structural weight for the fuel burn optimized design: an increase in wing span resulting in higher bending moments and a decrease in the t/c distribution resulting in a heavier structure. The twist distributions plotted in Figure 5.9 are derived from the deformed flying shape of the wing. The optimized twist distributions generally follow the original distribution with both designs lowering the amount of washout near the tip.

The spanwise lift distributions of the initial and optimized designs are shown in Figure 5.11. The first observation is that for all designs, the maneuver lift distributions are shifted inboard compared to the corresponding cruise lift distribution. This beneficial effect is caused by the aeroelasticly induced washout near the tip, resulting in reduced loading and represents a form of static aeroelastic tailoring. The cruise lift distributions for each optimization show more triangular load distributions compared to the original or the el-
Figure 5.9: Twist and thickness to chord variation for the initial and optimized designs (cruise and 2.5 g maneuver conditions).

Figure 5.10: Front view showing aerostructural deflections for the 2.5 g maneuver condition (left) and cruise design point 1 (right)

liptical reference. While the difference is slight, the fuel burn result has a lift distribution closer to elliptical, resulting in a reduction in induced drag. Comparing the lift distribution between the cruise and maneuver conditions, the difference is more pronounced for the fuel burn result. The TOGW optimum has a stiffer structure, and the raked wingtip, due to the lower lift-curve slope, is able to produce the beneficial inboard load shifting for the maneuver conditions. In the fuel burn case, however, even though the wing is heavier, the increased span leads to larger deflections and a more consistent gradual twisting near the tip. The similar aeroelastic effect of reducing the tip load is achieved naturally without raking the tip. A comparison of the deflections of the initial and optimized design as well as the jig shape is given in Figure 5.10.

An examination of the chord-wise $C_p$ distributions and airfoil shapes can explain the
aerostructural trade-offs the optimizer made between the five cruise design points. Figure 5.12 shows the airfoil shape and $C_p$ distribution for the initial and optimized designs for each operating condition at the 66% semi-span location. Design points 1, 4 and 5 are all at the design cruise Mach number of 0.85 and all show a reduction in the shock strength on the wing upper surface lowering the wave drag. Design point 2, at a lower Mach number of 0.84, shows a distinct, weak double shock structure on the upper wing surface. The higher mach number case of 0.86, (design point 3) shows a slightly reduced shock strength compared to the original. If we examine the airfoil cross section for the TOGW optimization, we observe a significant thickening of the aft portion of the airfoil. This is primarily to allow a thicker, more efficient, wingbox structure, which helps explain the weight reduction achieved for the TOGW optimization, despite an increase in wingspan. The vertical displacement of the optimized design at this spanwise location is lower than the initial design. This aft thickening is also present on the fuel burn optimization, but to a lesser extent. This thickening leads to a much more rapid pressure recovery than the baseline wing, which increases the risk of flow separation. A RANS simulation would be required to capture this phenomena.

Figure 5.13 shows the structural sizing resulting from the two aerostructural optimizations. Recall the initial design was first optimized with fixed aerostructural loads and this weight minimization resulted in a failure distribution that meets the KS failure constraints.
and is similar to Figure 5.13b or Figure 5.13f. However, when this design is used in the multidisciplinary optimization, we observe lower stress values. This highlights the need for MDO: even though we performed a structural optimization, we would have to continue the procedure of generating loads from the previous design and re-optimizing to generate a design that meets the multidisciplinary constraint.

Each optimization resulted in similar failure distributions. The plots in Figure 5.13 show the failure load for the symmetric 2.5 g pull-up maneuver. The upper skin and stringers are critical for this load case, but not the lower surface: the lower skin and stringers are dimensioned by the 1.3 g gust load. Since the lift distribution at 1.3 g is close to the 1 g condition, this constraint prevents more aggressive aeroelastic tailoring. The overall distribution of material for the initial and two optimized designs is similar. The largest skin thickness occurs through the mid section of the spar box where the thickness is greatest. The TOGW optimum shows locally increased physical wing thickness near the Yehudi break even with the large increase in $t/c$ ratios described earlier. To support the additional load due to the span extension, the minimum fuel burn wing has significantly increased thicknesses over the majority of the lower skin and stringers.

Finally, Table 5.5 lists the key results from the optimizations. Drag reduction for all operating conditions for both optimizations is achieved, with the higher Mach number point (2) and the higher lift condition (5) exhibiting the largest improvements.

For the fuel burn optimization, the pressure drag is lowered by 30%. As with the TOGW optimization, there is a reduction in wave drag, but there is a much greater reduction in the induced drag due to the higher aspect ratio. In fact, the TOGW for the fuel burn optimization remained essentially unchanged: the increase in structural weight is exactly offset by the reduced fuel load.

The weight breakdown of the primary structure is given in Table 5.5. For the TOGW optimization, the largest weight reduction is from the top and bottom skins and the lower stringers. From Figure 5.11, even the 1 g cruise conditions have more heavily loaded inboard lift distribution than the initial design. Since the 1.3 g lift distribution is similar to the cruise condition, we can conclude that the induced drag penalty from a more linear lift distribution is offset by the weight reduction in the lower skin and stringers. For the fuel burn optimization, weight increases are observed across most of the components, with the largest increases from the skins and rear spar. Given the reduced $t/c$ ratio and increased span, these increases are expected.
5.7 Conclusions

Using the optimization framework developed in this thesis, aerostructural optimizations were performed on the NASA CRM geometry with a structure created to be representative of a modern airliner wing. Multi-point optimizations with 5 cruise conditions and 2 maneuver conditions were performed with a 2.1 million cell CFD mesh and 300,000 DOF structural mesh. The optimization problems used 462 design variables and 57 constraints and required 36 hours of wall time on 435 processors.

The solution of this high-fidelity, high dimensional design optimization problem was made possible by the efficient solution and gradient techniques developed in this thesis. The two chosen objective functions, TOGW and fuel burn, provided valuable insights into the aerostructural trade-offs involved in transonic wing design. The objective sensitivity analysis from Chapter 4, indicated the fuel burn objective should produce a more “aerodynamic” design with lower drag than then TOGW objective. The optimizations show that this is indeed the case: the fuel burn optimization reduced the cruise segment fuel consumption by 11.2%, while the TOGW optimization reduced the fuel burn by 6.6%. Conversely, the TOGW minimization resulted in 4.2% reduction in TOGW while the TOGW for the fuel burn optimization remained effectively unchanged. The optimization results demonstrate the utility of the framework for aerostructural design optimization of full configurations with hundreds of design variables and real world constraints.

There are however, many important aspects of the aircraft design problem that have been not been considered in these optimizations. The most important is effect of viscosity. While an estimate for the skin friction drag is included, it is not possible to predict shock-induced flow separation. The true drag penalty from the large \( t/c \) increase for the TOGW optimization is probably not quantified correctly and using a RANS CFD simulation would result in a different optimum design. Given the high wing aspect ratio of the fuel-burn optimized design, this design may exhibit undesirable aeroelastic phenomena, such as flutter or aileron reversal effects, that are not considered in the optimization. Finally, for the structural analysis, I did not consider buckling failure modes. Typically, the upper wingbox skin and stringers are not sized by strength criteria, but by local or global buckling phenomena. By including buckling constraints into the optimization problem, I would expect to obtain a better estimate of the structural weight and this would impact the overall aerostructural design.
Figure 5.12: Cross section and $C_p$ contours for each operating condition at 66% semi-span. Initial design: dashed lines; optimized design: solid lines.
Figure 5.13: Initial and optimized thickness (left column) and failure parameter (right column) distributions over the wing box
<table>
<thead>
<tr>
<th></th>
<th>Initial TOGW optimization</th>
<th>Fuel burn optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Value</td>
<td>Value</td>
</tr>
<tr>
<td>Induced + wave drag (ct)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cruise 1</td>
<td>134.0</td>
<td>105.3</td>
</tr>
<tr>
<td>Cruise 2</td>
<td>133.1</td>
<td>112.6</td>
</tr>
<tr>
<td>Cruise 3</td>
<td>143.4</td>
<td>111.5</td>
</tr>
<tr>
<td>Cruise 4</td>
<td>122.7</td>
<td>99.4</td>
</tr>
<tr>
<td>Cruise 5</td>
<td>147.7</td>
<td>114.3</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>136.0</td>
<td>108.6</td>
</tr>
<tr>
<td>Friction drag (ct)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>136.0</td>
<td>139.0</td>
</tr>
<tr>
<td>Weight breakdown (kg)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bottom skin</td>
<td>12653</td>
<td>9192</td>
</tr>
<tr>
<td>Bottom stringers</td>
<td>2055</td>
<td>1687</td>
</tr>
<tr>
<td>Top skin</td>
<td>7515</td>
<td>5424</td>
</tr>
<tr>
<td>Top stringers</td>
<td>557</td>
<td>691</td>
</tr>
<tr>
<td>Rib</td>
<td>581</td>
<td>578</td>
</tr>
<tr>
<td>Rib stiffeners</td>
<td>45</td>
<td>56</td>
</tr>
<tr>
<td>Front spar</td>
<td>1235</td>
<td>1307</td>
</tr>
<tr>
<td>Rear spar</td>
<td>723</td>
<td>1145</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>25 368</td>
<td>20 078</td>
</tr>
<tr>
<td>Final cruise weight (kg)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>187 071</td>
<td>181 779</td>
</tr>
<tr>
<td>Initial cruise weight (kg)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cruise 1</td>
<td>289 203</td>
<td>276 875</td>
</tr>
<tr>
<td>Cruise 2</td>
<td>287 321</td>
<td>278 618</td>
</tr>
<tr>
<td>Cruise 3</td>
<td>295 213</td>
<td>280 218</td>
</tr>
<tr>
<td>Cruise 4</td>
<td>290 405</td>
<td>278 340</td>
</tr>
<tr>
<td>Cruise 5</td>
<td>288 858</td>
<td>276 411</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>290 200</td>
<td>278 098</td>
</tr>
<tr>
<td>Fuel burn (kg)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cruise 1</td>
<td>102 132</td>
<td>95 097</td>
</tr>
<tr>
<td>Cruise 2</td>
<td>100 250</td>
<td>96 839</td>
</tr>
<tr>
<td>Cruise 3</td>
<td>108 141</td>
<td>98 440</td>
</tr>
<tr>
<td>Cruise 4</td>
<td>103 334</td>
<td>96 561</td>
</tr>
<tr>
<td>Cruise 5</td>
<td>101 789</td>
<td>96 320</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>103 129</td>
<td>96 320</td>
</tr>
<tr>
<td>Span (m)</td>
<td>58.6</td>
<td>62.1</td>
</tr>
<tr>
<td>Aspect ratio</td>
<td>9.0</td>
<td>10.1</td>
</tr>
<tr>
<td>Leading edge sweep (°)</td>
<td>37.4</td>
<td>37.6</td>
</tr>
<tr>
<td>Reference wing area (m²)</td>
<td>383.7</td>
<td>383.7</td>
</tr>
<tr>
<td>Exposed wing area (m²)</td>
<td>335.0</td>
<td>335.0</td>
</tr>
<tr>
<td>Wing volume (m³)</td>
<td>46.0</td>
<td>53.0</td>
</tr>
</tbody>
</table>
Chapter 6

Software Engineering Methodology

In 1965, Dr. Gorden Moore, an engineer at Fairchild Semiconductor, noted that the number of transistors on a minimum-cost integrated circuit had doubled every 12 months over the previous five years [122]. In 1975, he revised the doubling time to 24 months. Remarkably, this exponential increases in number of transistors per chip has continued to the present day, a period of over five decades.

The performance increases of computer processors have profoundly changed how computational methods are used in engineering design. Time-consuming, computationally intensive simulations that were once used only for final design verifications are now used on a daily basis at the preliminary design stage.

The increasing availability of computing resources can be exploited a number of ways: a) Simulations can be performed with higher spatial or temporal resolution or with a more complex physical model — for example using a RANS simulation instead of a panel method. b) More varied analysis can be performed using the same resolution and physical model — for example we can consider more load cases for a structural simulation or more operating conditions for a fluid dynamics simulation. c) Multiple designs candidates can be evaluated with the aim of improving a figure of merit — this could involve performing parameter sweeps or using a formal optimization algorithm. d) Lastly, computational methods that are traditionally treated in a segregated fashion could be combined to form a multidisciplinary analysis.

Venkatamaran and Haftka [25] considered the historical effects of increasing computational performance on structural analysis and optimization. They note that computational analysis tends to follow Parkinsons Law [123] that states that the work done expands to fill up all the available time. A related, computerized form of the law [124] states that software applications grow to fill up increased computer memory, processing capabilities and
storage space. Venkatamaran and Haftka point out that anecdotal evidence suggests that
time required for “adequate” structural analysis has remained constant at between 6–8 hours
over the last 30 years. This evidence would indicate that computational improvements have
been used to refine the computational models, which describes scenario (a) above. I believe
the main reason for this is that only the first scenario does not fundamentally change the
complexity of an engineering work flow.

The results presented in this thesis, however, have combined scenarios (b) through (d).
This multi-faceted approach to utilizing computational resources has significant advantages
for aircraft design optimization, as shown in the previous chapter. The drawback is that the
computational framework required for the simulations is now vastly more complex from a
software engineering perspective. The difference in complexity for a user running an interac-
tive analysis for a single disciplinary solution compared to a user performing a batch-mode
multidisciplinary, multi-point optimization, is enormous. For example, text based command
scripts that may be suitable for running a batch-mode disciplinary analysis are inadequate for
the complex simulations considered in this thesis. It is critical that computational methods
facilitate the task of setting-up and performing complex multidisciplinary optimizations.

The remainder of this chapter will detail and justify the methods I used to couple codes
in an efficient and scalable manner. I also show the complete breakdown of the software
modules and interactions used to compose the optimization problems detailed in Chapter 5.

6.1 Coupling Software Codes

The task of coupling software codes together begins with ensuring each code has the
ability to perform batch-mode analysis — the ability to perform a specified sequence of
commands without user input. Batch-mode analysis will typically read an input file, perform
a computation, and write the solution data in a fully automatic fashion. Once discipline
analyses can be executed automatically, it is possible to implement a Gauss–Seidel approach
for solving a multidisciplinary system. In the case of aerostructural analysis, for example,
the aerodynamic analysis writes a set of loads to disk, followed by the automatic creation
of the input file for the structural analysis. The structural analysis is executed and the
displacements are written to a file and these displacements are used to deform or regenerate
the CFD mesh. The cycle continues until a specified level of convergence in achieved. A
diagram of this procedure is given in Figure 6.1.

This is the traditional method of using stand-alone codes to perform an aerostructural
analysis. With a driving script configured to automatically generate input files and auto-
matically parse output files, this technique can almost always be used couple codes together. ModelCenter [125], is an example of commercial software that uses the file-I/O approach to couple codes together and is widely used in the aerospace industry. OpenMDAO[126] is an open source NASA project that aims to accomplish similar goals.

There are however, several serious drawbacks with the file-I/O approach. The first one is the limited availability of disk bandwidth and the associated wall-time required to write, read and parse the associated data. On massively parallel computing machines, file I/O is typically a shared resource and thus the decrease in throughput experienced by a given user can become significant. The data transfers speeds that can be sustained between computational nodes far exceeds the speeds obtained with file I/O. Using parallel solution techniques for each discipline, only further complicates the information transfer.

The second concern is the potential loss of accuracy between iterations in a multidisciplinary analysis. During an MDA, subsequent disciplinary solutions are generally closely related to previous solutions. For iterative methods, it is prudent to reuse this information from one multidisciplinary iteration to the next to reduce computational cost. This operation requires a restart capability, which is generally not an onerous requirement, but it does increase the amount of information to be written to disk.

The third concern is the repeated execution of the same program. Each time a discipline is called, a new process must be started, and “one time” initialization functions are typically performed. These operations may not be as optimized for speed as the remainder of the code. In the case of the structural analysis for example, repetitively calling the structural solver requires the assembly and factorization of the tangent stiffness matrix at each iteration requiring significant computational effort. The final concern is creating methods to parse the output from each discipline in a efficient and robust manner.
Unfortunately, the file-I/O approach is the only option if an Application Program Interface (API) is not supplied with direct access to the required functions, as is often the case with commercial codes. However, if the code has an applicable API or the source code is available, a much better approach can be pursued to combine codes.

To avoid the pitfalls of file I/O, it is critical that all interdisciplinary information transfer occur strictly using a direct memory access technique. In this approach, a process script is required to direct the sequence of operations performed by each discipline. The main difference, however, is that the shared information — the aerodynamic forces and structural displacements — is processed directly. The process script then configures the next disciplinary analysis to run directly rather than through an input file. In essence, the Process Script acts as a “glue” controlling the behavior of the subordinate disciplines.

Alonso et al. [64] considered several candidate programming languages for the “glue” including, C++, Java, Perl, Matlab and Python. After evaluating the advantages and disadvantages of each, Python was determined as the most suitable language for the needs of multidisciplinary optimization. I have used Python extensively to couple codes together in this work, and in my experience it continues to be an excellent choice.

6.2 Python Wrapping

Python [127] has many qualities that make it desirable as a scripting language “glue” for high performance scientific computing. Several of the key advantages are:

- The ease of interacting with existing compiled languages, especially C and Fortran
- Multiple programming models including procedural and objected-orientated
- High-quality, open-source numerical libraries such as NumPy and SciPy [128]
- Good support for MPI on compute clusters [129]
- Fully open-source

In addition, the Python syntax is exceptionally clear and makes it easier for others to read and understand the intent of the programmer. The ability to encapsulate the data and methods of a complex scientific code as a Python objects makes configuring sophisticated multidisciplinary optimization problems significantly more straight forward.

A Python wrapper is a library, rather than an executable, that contains an interface to most or all of the functionality of the original code. In its simplest form, a Python wrapper
may emulate the file-I/O driven procedure described in Figure 6.1. The API writes an input file and calls a high-level \texttt{run()} command that executes the same sequence of functions as if the user had run the program in batch mode. While this approach is a good first step, it does not fully alleviate the file-I/O bottleneck. Depending on the application, it is generally desirable to have slightly finer level of control from the calling Python program.

Python is written in C, and it is possible to write the interfaces to call external C code from Python by hand. For large projects, this is a tedious process and can be automated using tools such as using \texttt{swig} [130] or \texttt{cython} [131]. The structural solver TACS uses \texttt{swig} to construct the Python interface.

It is possible to use the same manual technique to wrap Fortran codes, but the \texttt{f2py} [132] tool can largely automate the process. Typically, only a small subset of the functions in a module need to be exposed at the Python level. With \texttt{f2py} these functions are specified using a semi-automatically generated \textit{signature file}. The signature file specifies input and output arguments as well as their rank and size. Using special \texttt{f2py} directives, the signature file can be generated in a completely automatic fashion.

Using compiled libraries to do the bulk of the floating-point computations for each discipline overcomes the speed disadvantage of Python. As an interpreted scripting language, Python is inherently slower than compiled languages, often by very large margins. Since the compiled library does the bulk of the numerical computations, the speed and computational efficiency of the compiled language is retained, while the ease of use and the flexibility and object oriented approach of Python is maintained.

### 6.3 Coupling Method Comparison

To demonstrate the speed advantage of a memory-based code coupling approach using Python over a file-I/O approach, a series of tests were performed to compare the efficiency of both approaches. Since a file-based multidisciplinary analysis would not be possible without several modifications to the existing code, an alternative approach was used. To simulate the file-based MDA, a separate instance of the particular solver (aerodynamic, structural, or mesh) is instantiated when the solver is to be called, and the required output files are written to disk when the solver is complete. This approach effectively simulates the file-I/O approach with little coding effort.

Two different computational environments are considered: A desktop computer and a high performance computing cluster with shared file I/O. For the desktop computer, an MDA is performed with a 147,456 cell CFD grid and a structural model with 84,884 DOFs.
using 3 aerodynamic processors and 1 structural processor. For the cluster case, the grid spacing is reduced by a factor of two yielding a 1179648 cell CFD mesh and a 179628 DOF structural mesh. This simulation is performed using 24 aerodynamic processors and 4 structural processors. Table 6.1 lists the timing breakdown for each case.

Table 6.1: Comparison of file-based and memory-based coupling schemes for a desktop computer and a high-performance parallel cluster

<table>
<thead>
<tr>
<th></th>
<th>Desktop</th>
<th>Cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time (s)</td>
<td>Time (s)</td>
</tr>
<tr>
<td></td>
<td>Fraction</td>
<td>Fraction</td>
</tr>
<tr>
<td>Memory-based MDA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CFD solution</td>
<td>90.6</td>
<td>221.2</td>
</tr>
<tr>
<td></td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>Warp solution</td>
<td>78.5</td>
<td>209.4</td>
</tr>
<tr>
<td></td>
<td>0.866</td>
<td>0.947</td>
</tr>
<tr>
<td>CSM solution</td>
<td>2.3</td>
<td>3.7</td>
</tr>
<tr>
<td></td>
<td>0.025</td>
<td>0.017</td>
</tr>
<tr>
<td>File-based MDA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CFD solution</td>
<td>418.1</td>
<td>435.2</td>
</tr>
<tr>
<td></td>
<td>4.613</td>
<td>1.967</td>
</tr>
<tr>
<td>CFD Input + Setup</td>
<td>87.9</td>
<td>220.1</td>
</tr>
<tr>
<td></td>
<td>0.970</td>
<td>0.994</td>
</tr>
<tr>
<td>CFD Output</td>
<td>7.2</td>
<td>5.8</td>
</tr>
<tr>
<td></td>
<td>0.080</td>
<td>0.026</td>
</tr>
<tr>
<td>Warp solution</td>
<td>30.4</td>
<td>33.2</td>
</tr>
<tr>
<td></td>
<td>0.336</td>
<td>0.150</td>
</tr>
<tr>
<td>Warp Input + Setup</td>
<td>27.5</td>
<td>27.8</td>
</tr>
<tr>
<td></td>
<td>0.303</td>
<td>0.126</td>
</tr>
<tr>
<td>Warp Output</td>
<td>0.7</td>
<td>1.6</td>
</tr>
<tr>
<td></td>
<td>0.008</td>
<td>0.007</td>
</tr>
<tr>
<td>CSM solution</td>
<td>307.6</td>
<td>189.4</td>
</tr>
<tr>
<td></td>
<td>3.393</td>
<td>0.856</td>
</tr>
<tr>
<td>CSM Input + Setup</td>
<td>282.3</td>
<td>170.4</td>
</tr>
<tr>
<td></td>
<td>3.114</td>
<td>0.770</td>
</tr>
<tr>
<td>CSM Output</td>
<td>6.8</td>
<td>3.1</td>
</tr>
<tr>
<td></td>
<td>0.075</td>
<td>0.014</td>
</tr>
</tbody>
</table>

Since the file-based MDA is only simulated, the timings in Table 6.1 can only be compared in a qualitative fashion. Additionally, the file-I/O simulation does not include the time required to parse discipline output files to create input files for the subsequent discipline. If large text files need to be processed for this operation, significant additional costs would be incurred.

For both the cluster and desktop computing environments, the file-based MDA requires more wall-time than the memory-based MDA. In both cases, the largest contribution to the additional cost is the CSM setup, which includes assembly and factorization of the stiffness matrix. It is possible this factorization could be written to disk for subsequent iterations, saving computation at the expense of significantly more file I/O.

The overall time to complete the file-based MDA is almost double for the cluster case and approximately 4.5 times for the desktop case. It is clear the efficiency of the memory-based coupling approach is significant and the preferred approach for coupling codes in a
multidisciplinary environment.

### 6.4 Software Architecture for Aerostructural Optimization

This section will briefly describe the modules, class structure, and dependencies in the MACH framework. The multi-point optimization results presented in Chapter 5 demonstrates usage of the complete framework.

The overview of the entire framework is depicted in Figure 6.2. A single file, `run_script.py`, defines all relevant information that is specific to a particular problem and all other modules are generic. The `run_script.py` is configured to use an object-oriented optimization framework called `pyOpt` [133]. The `pyOpt` modules contain wrappers to many optimization algorithms allowing the user to experiment with different optimization algorithms without changing the problem formulation. For all optimization results in this thesis, I used `pySNOPT`, a Python wrapped version of SNOPT.

![Figure 6.2: UML class diagram overview for multi-point aerostructural optimization](image)

The `multiPoint` module is used to facilitate the parallel evaluation of multiple operating conditions and to communicate the required information between sets of processors.

The four main modules used directly in the run script are `DVGeometry`, `DVConstraints`, `pyFriction` and `pyAeroStruct`. `DVGeometry` uses the FFD approach described in Chapter 2 to manipulate the geometric representations from `pyAeroStruct`, `pyFriction`, and `DVConstraints`. The `DVConstraints` module implements thickness and volume constraints for wing-like structures as well as constraints on the motion of control points at the leading and trailing edges of FFD volumes surrounding wings. The UML class diagrams for the `DVGeometry` and `DVConstraints` modules are given in Figure 6.3.

These two modules effectively demonstrate how a small amount of compiled code can be
used to substantially accelerate Python code. All of the computationally intensive operations including the spline evaluation routines and point inversion routines are implemented in Fortran and are contained in the `pyspline.so` module.

The class structure of `pyAeroStruct` is shown in Figure 6.4.

The implementation of the coupled Newton–Krylov (CNK) solver described in Chapter 2 effectively demonstrates how Python APIs can be exploited in a multidisciplinary setting. To evaluate the nonlinear aerostructural residual described in Algorithm 2, both the aerodynamic and structural solvers implement `setStates()` and `getResidual()` functions. Along with the aerodynamic routine `getForces()` and the `getDisplacements()` structural routine, all of the required information is readily available to the CNK solver. What makes this approach powerful is neither discipline level solver is fundamentally modified, and can be continued to be used in a standalone fashion — potentially within the same optimization problem. The task of solving the multidisciplinary system has been completely removed from each of the disciplines and neither is “aware” a multidisciplinary system is being solved. This Python API approach allows the complete CNK solver to require fewer than 100 lines
of Python code.

### 6.5 Conclusions

This chapter has outlined the details on the software implementation of the MACH architecture. The two main approaches for coupling codes together, file-I/O and direct-memory techniques were discussed. A comparison of both techniques showed the direct memory technique is approximately two times faster on a high performance computing cluster and approximately four times faster on a desktop computer. Lastly, I described the modules required to obtain the aerostructural optimization results in Chapter 5.
Chapter 7

Conclusions, Contributions, and Recommendations

In this chapter, I summarize the main contributions from my thesis, state the overall conclusions, and discuss future research directions.

7.1 Thesis Contributions

Efficient gradient-based optimization of a realistic large-scale optimization problem is a significant undertaking. The multi-point optimizations presented in Chapter 5 represent the current state-of-the-art in gradient-based aerostructural optimization. To advance this field I made contributions to the solution of aerostructural systems and the sensitivity analysis of these systems. Advancements to aerostructural analysis was made through improvements in aerodynamic analysis as well as mesh-deformation algorithm. Contributions to the sensitivity analysis include a new method for evaluating the aerodynamic Jacobian using forward-mode AD, a completely consistent coupled adjoint formulation and a fully coupled adjoint solution strategy. Without the following contributions, the optimization results would not have been possible in a reasonable time-frame.

Geometry manipulation: I showed that the CAD-free multi-block FFD geometric manipulation approach satisfies all the key requirements for gradient-based aerostructural design optimization. (Section 2.1)

Mesh-Deformation: The new hybrid algebraic, linear elasticity mesh-deformation approach for multi-block grids was critical to the success of the optimizations. The coarse super-mesh approach coupled with a robust, but computationally expensive linear or nonlinear solution algorithm, proved to produce high quality meshes with even
with very large design variable changes. I showed that the method can be applied without modification to both Euler and RANS grids. I showed the mesh sensitivity analysis, obtained through the use of reverse-mode AD and a mesh adjoint is fast and accurate. The mesh sensitivities eliminated a significant bottleneck plaguing previous work. (Section 2.2)

**Forward-mode ADjoint:** A new method for computing the state residual matrix of the Euler equations using automatic differentiation was demonstrated. I showed there are several advantages to this technique including reduced computational cost and significant improvements to code maintainability. Several new block structured colouring schemes for sparse matrix evaluations were presented. (Section 3.2)

**Coupled adjoint:** Using the complex step derivative method, I was able to show that the coupled adjoint method is perfectly consistent with the discretized governing equations. This is possible since the high-fidelity implementation is the first where all partial derivative terms are computed using machine-accurate derivative techniques (Forward-mode AD, reverse-mode AD, Complex Step and analytic). (Section 3.1)

**Coupled Krylov method:** I presented the first application of a coupled solution technique applied directly to a high-fidelity aerostructural adjoint system. I showed that the CK method is faster than an optimally tuned LBGS method and has been applied to aerostructural systems with over 80 million aerodynamic variables and 1 million structural variables. (Section 3.7)

**Software engineering:** The techniques described in Chapter 6 demonstrate how separate analysis codes can be coupled in an efficient manner. I showed that the direct memory access technique is substantially faster than the traditional file I/O approach. Using the direct memory technique, I was able to construct efficient coupled solvers (CNK and CK methods) with entirely separate, independent solvers. (Chapter 6)

### 7.2 Conclusions

Increasing concerns of environment impact and operational cost continue to drive aircraft manufacturers to design more fuel efficient, “greener” aircraft. The impressive trend of reduced fuel consumption with each successive generation of aircraft continues, but extracting global performance improvements from isolated sub-systems without considering the impact on the rest of the overall design is becoming increasing difficult. The techniques discussed in
this thesis advance the state-of-the art of aerostructural optimization for aircraft design. The results show that it is possible to capture complex multidisciplinary trade-offs using high-fidelity methods and perform optimizations with hundreds of variables in a timely fashion. The conventional CRM configuration I optimized shows there is still room for improvement in conventional designs. However, since the high-fidelity computational methods used for the optimization do not depend on past engineering experience, they are very well suited to optimize unconventional configuration in the future.

7.3 Future Work

The optimizations presented in Chapter 5 are the most comprehensive, multi-point, high-fidelity aerostructural optimizations completed to date. However, throughout the course of my research, I have identified several areas of the computational process that warrant further investigation. The following topics are ways of improving the accuracy or efficiency of the aerostructural analysis and coupled adjoint analysis. These improvements will make it possible to optimize novel aircraft configurations in the future with increased confidence in the optimized designs.

**Reynolds Averaged Navier–Stokes (RANS) analysis:** The replacement of the Euler equations with the RANS equations is the most important improvement. For the TOGW optimization, the solution of the Euler equations is not sufficient to predict the true drag penalty from the large \( t/c \) increase near the Yehudi break. Even though the Euler equations can capture increases in shock strength, they cannot predict shock-induced flow separation. A RANS CFD solution of the TOGW optimized design shows shock-induced flow separation. The disadvantage of the RANS analysis is the requirement of larger grids with much smaller off-wall spacing. The large cell aspect ratios, the viscous flux computations, and the turbulence model computations all require increased computational effort, making a RANS simulation 3 to 5 times more costly than a corresponding Euler simulation with similar spatial resolution.

**Adjoint memory use:** The coupled adjoint solution techniques presented in Chapter 3 are computationally efficient, but require a substantial amount of memory and for large aerostructural optimization problems, memory usage becomes a critical concern. The majority of the memory is required for the solution of the aerodynamic adjoint
equation. The largest memory contributors along with the typical percentage of total memory usage are: \( \frac{\partial A}{\partial X} V (30\%) \), \( \frac{\partial A}{\partial w} (15\%) \), \( \frac{\partial A_1}{\partial w} (8\%) \), ASM overlap storage (10\%), ILU factorization (10\%).

The largest savings can be obtained by computing \( \frac{\partial A^T}{\partial X} \psi \) products using reverse-mode AD. While the reverse-mode AD method requires storage for the tape variables [109] initial indications are that this is substantially less demanding than storing the full Jacobian.

It is also technically possible to compute transpose \( \frac{\partial A}{\partial w} \) products in a similar manner using reverse-mode AD. However, given the large number of matrix vector products required for the aerodynamic adjoint solutions, it is most efficient to fully store this term.

The remainder of the terms, \( \frac{\partial A_1}{\partial w} \), ASM overlap storage, and ILU factorization, all relate to the particular type of preconditioning I have chosen. Alternative, less memory intensive, methods for preconditioning should be considered to further reduce memory requirements.

**CNK start-up:** The start-up behaviour of the CNK method has not been investigated in detail. Further research is required to determine if a more suitable technique than relaxed NLGBS iterations is required.

**Mesh-Deformation scheme:** Several issues were identified with the mesh-deformation scheme. Block splitting is necessary to allow strong parallel scaling when more processors are used than number of blocks in the grid. The CK adjoint method on the finest grid in the convergence study in Chapter 3 highlighted the need for all components of the entire sensitivity analysis to be scalable.

**Aeroelastic constraints:** The fuel burn optimization results showed a significantly increased span, higher aspect ratio and lower sweep. All of these effects make the wing more susceptible to flutter. Recent advances in this area has been reported by Chen *et al.* [134, 135] using the complex-step approach. The challenge of computing gradients with respect to functions obtained from unsteady aeroelastic analysis with respect to a large number of design variables, will be challenging, but necessary, for integration into the current framework.

**Buckling constraints:** Large parts of the wingbox upper skin and upper stringers are
typically not sized by strength criteria, but by buckling considerations. The addition of buckling constraints to the existing strength criteria, will result in a more accurate prediction of the required primary structural mass [136].

**Composite materials:** The use of composite materials in commercial airliners continues to increase. Recent applications include the Boeing 787 and the Airbus A350, both of which use composite materials for primary load-bearing structures such as the aircraft’s wingbox and fuselage skin. Aerostructural optimization with composite materials is challenging [136], but necessary, as future aircraft designs will most likely use these materials.

**Geometrically nonlinear structural analysis:** The validity of the linear structural model becomes questionable when large displacements are involved. A geometrically nonlinear structural analysis would better predict both the flying shape of the wing, as well as the stresses in the structure. Nonlinear structural analysis will be especially important for even more flexible structures that are possible with composite materials.

**Low-speed analysis:** The leading edge radius plays an important role in both high-speed cruise flight as well as low-speed $C_{L_{\text{max}}}$ performance. Since the optimizations did not consider the impact on low-speed performance, thickness constraints were used to ensure the radius did not decrease compared to the initial design. A method to predict the clean-wing $C_{L_{\text{max}}}$ performance, such as Valarezo’s method could eliminate these constraints [137, 138]. Furthermore, the requirement of a progressive inboard to outboard stalling pattern may place additional constraints on the optimal chord and jig-twist distributions. Other low-speed manoeuvres such as take-off and landing, as well as handling constraints at these conditions, should be considered.

**Unconventional configurations:** I choose to optimize a conventional configuration in order to understand the multidisciplinary trade-offs obtained during the optimization process. The true power of high-fidelity multidisciplinary optimization however, comes from the optimization of unconventional configurations. With physics-based analysis we can have much greater confidence that new designs will perform as predicted in the real world. With many decades of experience with conventional swept-wing configurations, modern transport aircraft are refined designs that largely incorporate the correct multidisciplinary trade-offs. While there are some important differences, the optimized configurations are largely similar to the baseline configuration, suggesting
that the high-fidelity analysis is capturing the most important multidisciplinary trade-offs. The next step is to optimized a promising unconventional configuration such a Blended Wing Body (BWB) design.
Bibliography


Appendix A

Simple Transonic Wing

The simple transonic wing (STW) is a small test case that has been used several times throughout this thesis. The STW is based on the overall dimensions of a Boeing 717. The planform and specifications are given in Figure A.1.

Figure A.1: Simple transonic wing specifications

![Planform view](image)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cruise Mach number</td>
<td>0.78</td>
<td>-</td>
</tr>
<tr>
<td>Cruise lift coefficient</td>
<td>0.5</td>
<td>-</td>
</tr>
<tr>
<td>Span</td>
<td>14.0</td>
<td>m</td>
</tr>
<tr>
<td>Aspect ratio</td>
<td>8.62</td>
<td>-</td>
</tr>
<tr>
<td>Reference wing area</td>
<td>91.0</td>
<td>m²</td>
</tr>
<tr>
<td>Sweep (leading edge)</td>
<td>28.2</td>
<td>°</td>
</tr>
<tr>
<td>Twist</td>
<td>0</td>
<td>°</td>
</tr>
<tr>
<td>Root Profile</td>
<td>RAE 2822</td>
<td>-</td>
</tr>
<tr>
<td>Tip Profile</td>
<td>RAE 2822</td>
<td>-</td>
</tr>
</tbody>
</table>

Two sets of CFD grids have been generated for the wing, one suitable for Euler analysis and the other suitable for RANS analysis. The Euler grid (Figure A.2a) contains 1 179 000 cells with off-wall spacing of $1 \times 10^{-3}$ chord lengths. It uses an H-H type griding strategy and contains a sharp trailing edge. The RANS grid (Figure A.2b) contains 2 803 712 cells with an off-wall spacing of approximate $1 \times 10^{-6}$ chords. It is suitable for flight-speed Reynolds numbers of approximately 20 million. The grid uses an O-H-O griding strategy. An O-grid contains the boundary layer region, a H-mesh region is used in the near field and finally an O-mesh in the far-field region. This grid contains a realistic constant thickness blunt trailing edge.
(a) Euler grid. Flow solution at $M = 0.82$, $\alpha = 2.25^\circ$

(b) RANS grid. Flow solution at $M = 0.78$, $\alpha = 2.25^\circ$

Figure A.2: Simple transonic wing grids.
Appendix B

van Driest II Method

This method is used to estimate the turbulent skin friction coefficient, $C_F$, over a flat plate. A turbulent factor of $r = 0.88$ and ratio of specific heats, $\gamma = 1.4$ are assumed. For a given Mach number ($M$), temperature at the leading edge of the plate ($T_e$), and ratio of wall temperature ($T_W$), to adiabatic wall temperature ($T_{AW}$), $T_W/T_{AW}$, (taken to be 1.0) the following constants are first computed:

$$m = \frac{\gamma - 1}{2} M^2 \quad (B.1)$$

$$F = \frac{T_W}{T_e} = \frac{T_W}{T_{AW}} \cdot \frac{T_{AW}}{T_e} \quad (B.2)$$

where

$$\frac{T_{AW}}{T_e} = 1 + rm \quad (B.3)$$

$$T_w = F \cdot T_e \quad (B.4)$$

$$A = \left( \frac{rm}{F} \right)^\frac{1}{2} \quad (B.5)$$

$$B = \frac{1 + rm - F}{F} \quad (B.6)$$

$$\alpha = \frac{2A^2 - B}{(4A^2 + B^2)^\frac{1}{2}} \quad (B.7)$$

$$\beta = \frac{B}{(4A^2 + B^2)^\frac{1}{2}} \quad (B.8)$$

$$F_c = \frac{rm}{(\sin^{-1} \alpha + \sin^{-1} \beta)^2} \quad (B.9)$$
and

\[ F_\theta = \frac{\mu_e}{\mu_w} = \sqrt{\frac{1}{F} \left(\frac{1 + \frac{122}{T_w} \times 10^{-5/T_w}}{1 + \frac{122}{T_e} \times 10^{-5/T_e}}\right)} \quad \text{(B.10)} \]

\[ F_x = \frac{F_\theta}{F_c} \quad \text{(B.11)} \]

For a given chord-wise Reynolds number, Re\(_x\), we compute

\[ \bar{\text{Re}}_x = F_x \text{Re}_x \quad \text{(B.12)} \]

Next, the following nonlinear equation is solved for \( \bar{C}_F \):

\[ \frac{.242}{\sqrt{\bar{C}_F}} = \log \left(\text{Re}_x \bar{C}_F\right). \quad \text{(B.13)} \]

A suitable initial guess is:

\[ \bar{C}_F = \frac{0.074}{\text{Re}_x^{0.20}}. \quad \text{(B.14)} \]

Once \( \bar{C}_F \) is found, the skin friction coefficient is then computed from:

\[ C_F = \frac{\bar{C}_F}{F_c}. \quad \text{(B.15)} \]