A NUMERICAL METHOD FOR FULLY NONLINEAR AEROELASTIC ANALYSIS

A Dissertation

by

JOAQUIN IVAN GARGOLOFF

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2007

Major Subject: Aerospace Engineering

A NUMERICAL METHOD FOR FULLY NONLINEAR AEROELASTIC ANALYSIS

A Dissertation

by

JOAQUIN IVAN GARGOLOFF

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,	Paul Cizmas
Committee Members,	Thomas Strganac
	Othon Rediniotis
	Theofanis Strouboulis
	Ergun Akleman
Head of Department,	Helen Reed

May 2007

Major Subject: Aerospace Engineering

ABSTRACT

A Numerical Method for Fully Nonlinear Aeroelastic Analysis. (May 2007)
Joaquin Ivan Gargoloff, B.S., La Plata National University, Argentina
Chair of Advisory Committee: Dr. Paul Cizmas

This work presents a numerical method for the analysis of fully nonlinear aeroelastic problems. The aeroelastic model consisted of a Navier-Stokes flow solver, a nonlinear structural model, and a solution methodology that assured synchronous interaction between the nonlinear structure and the fluid flow.

The flow around the deforming wing was modeled as unsteady, compressible and viscous using the Reynolds-averaged Navier-Stokes (RANS) equations. To reduce the computational time, a three-level multigrid algorithm was implemented and the flow solver was parallelized. The message-passing interface (MPI) standard libraries were used for the parallel interprocessor communication.

The computational domain was divided into topologically identical layers that spanned from the root to past the tip of the wing. A novel mesh deformation algorithm was developed to deform the mesh as the structure of the wing was being displaced. The mesh deformation algorithm was able to handle wing tip deformations of up to 60 % of the wing semi-span. Besides being robust, the mesh deforming algorithm was computationally more efficient than regriding, since deforming an existing mesh was computationally less expensive than generating a new mesh for each wing position.

Results are presented for the validation and verification of both the flow solver and the aeroelastic solver. The flow solver was validated using: (1) the flow over a flat plate, to validate the turbulent model implementation, and (2) the flow over the NACA 0012 airfoil and over the F-5 wing, to validate the implementation of the convective and viscous fluxes, the time integration algorithm, and the boundary conditions. The aeroelastic solver was validated using: (1) the unsteady F-5 wing undergoing forced pitch motion, and (2) the Nonlinear Aeroelastic Test Apparatus (NATA) wing. In addition, aeroelastic results were generated for the Goland wing.

The aeroelastic solver developed herein allows the analysis of aeroelastic phenomena using a fully nonlinear approach. Limit cycle oscillations, which are highly nonlinear phenomena, were captured by the nonlinearities of the flow solver and the structural solver. The impact of the nonlinearities was assessed for the Goland wing, where nonlinear terms changed dramatically the aeroelastic behavior of the wing.

ACKNOWLEDGMENTS

I would like to thank my advisor and committee chair, Dr. Paul Cizmas, and my committee members, Dr. Thomas Strganac, Dr. Othon Rediniotis, Dr. Theofanis Strouboulis, Dr. Ergun Akleman, and also Dr. Philip Beran from AFRL/VASD for their guidance, valuable insights, and continuous support and encouragement throught the development of this research.

I would also like to thank my fellow graduate students Brian Richardson, Tom Brenner, and those who already graduated, Chetan Nichkawde, Steven Chambers, Kyusup Kim, Celerino Resendiz, and Tao Yuan. Special thanks to my classmates and friends at the Aerospace Department and to the fine department faculty and staff, who assisted me during my studies at Texas A&M University.

Finally, I would like to thank my parents, Angel and Marta, my brother Matias, and my beloved girlfriend Lisa, for their love, support, and encouragement. Special thanks to my friends and family in Argentina who despite the distance kept a close contact with me and shared the exciting experience of living in College Station and Texas.

TABLE OF CONTENTS

CHAPTER Page

I INTRODUCTION 1

- A. Statement of work 1
- B. Aeroelastic problem 2
- C. Background 6
- D. Flow solver 8
- E. Grid generation and grid deformation algorithms 8
- F. Parallel algorithm 9
- G. Multigrid implementation 9
- H. Original contributions of the present work 9
- I. Outline of dissertation 10

II PHYSICAL MODEL 11

- A. Aerodynamic model 11
  - 1. Reynolds-averaged Navier-Stokes equations 11
  - 2. Turbulence model 14
- B. Structural model 17
- C. Coupling of the aerodynamic and structural models 22

III NUMERICAL METHOD 23

- A. Mesh generation algorithm 23
- B. Flow solver 25
  - 1. Navier-Stokes equations 25
  - 2. Turbulence model 28
  - 3. Spatial discretization 29
  - 4. Vector fluxes implementation 32
  - 5. Second-order upwind scheme 38
  - 6. Time integration 39
  - 7. Boundary conditions 42
- C. Mesh deformation algorithm 45
  - 1. Translational deformations 45
  - 2. Rotational deformations 46
    - a. Rotation about the x-axis 46
    - b. Rotation about the y-axis 48

CHAPTER Page

            c. Rotation about the z-axis 51
        3. Computation of the rotation angles 52
        4. Cubic mapping function 54
        5. Parametric study of mesh deformation algorithm . . . 58
            a. Goland wing 58
            b. F-5 wing 59
    D. Parallel algorithm 60
    E. Multigrid 64
        1. Introduction 65
        2. Multigrid mesh subdivision 67
        3. Interpolation transfer operators 70
            a. Interpolation schemes for internal quadrilateral cells 71
            b. Interpolation schemes for boundary quadrilateral cells 73
            c. Interpolation schemes for triangular cells 74
            d. Interpolation schemes for mixed triangular and quadrilateral cells 75
        4. Implementation of multigrid on the flow solver 76
        5. Multigrid versus one-level grid solvers 79
            a. Inviscid flow results 80
            b. Turbulent flow results 83

IV RESULTS 90

    A. Validation and verification of the flow solver 91
        1. Steady flow over a flat plate 91
        2. Steady flow over the NACA 0012 airfoil 94
        3. Steady flow over the F-5 wing 97
    B. Aeroelastic results 100
        1. F-5 wing undergoing forced pitching motion 100
        2. Aeroelastic results for the NATA wing 102
        3. Aeroelastic results for the Goland wing 108
            a. Goland wing at low Mach number 108
            b. Impact of nonlinear structural model 110
            c. Stability boundary of Goland wing 113

V CONCLUSIONS AND FUTURE WORK 115

    A. Conclusions 115

CHAPTER Page

B. Future work 117

REFERENCES 118

VITA 125

LIST OF TABLES

TABLE Page

I Efficiency of parallel computation. 64

II Average and maximum difference between the one-level grid solution and the multigrid solution. 87

III NACA 0012 grid parameters. 96

IV Pressure and suction side peak comparative. 98

V F-5 wing grid parameters. 101

LIST OF FIGURES

FIGURE Page

1 Aeroelasticity as an interdisciplinary interaction. 3

2 Linear and nonlinear aeroelastic response 5

3 Structural model of the wing. 17

4 Modal shape functions for bending and torsion. 21

5 O-grid generated around the wing. 24

6 O-grid and unstructured mesh, close up and whole domain views. . . 25

7 Mesh with topologically identical layers. 26

8 Median and centroid dual meshes. 30

9 Median dual mesh for unstructured mixed grids. 31

10 Dual mesh face associated with the edge (i, j). 32

11 Linear reconstruction of the solution. 39

12 Quadrature points to calculate boundary fluxes. 44

13 Mesh before and after the translational deformations. 46

14 Rotations about the x-axis. 47

15 Mesh before and after the x-axis rotational deformation. 49

16 Rotations about the y-axis. 50

17 Rotations about the z-axis. 51

18 Cubic mapping function and front view of rotational deformations. . 55

19 3D view of rotational deformations. 57

FIGURE Page

20 Goland wing, quality measure as the wing deforms. 60

21 Front and 3D view of Goland wing with tip deformation equal to 60% of wing semi-span. 61

22 F-5 wing: Deformed and undeformed meshes. 62

23 Schematic view of the parallel implementation. Subindex indicate the sub-grid number. Active nodes are marked as black circles. Ghost nodes are marked as white circles. Arrows indicate the communication path. 63

24 Arbitrary sinusoidal initial solutions. 67

25 Error history for sinusoidal initial solutions with different k frequencies 68

26 Error history for initial solution composed of sinusoidal functions with different frequencies 68

27 Triangular and quadrilateral element subdivision. 69

28 True shape of boundary is recovered as the grid is refined. 70

29 Boundary node relocation. Black circles represent coarse grid nodes, white squares represent fine grid nodes. 71

30 Coarse and fine meshes for internal quadrilateral cells. 72

31 Coarse and fine meshes for boundary quadrilateral cells. 73

32 Coarse and fine meshes for triangular cells. 74

33 Coarse and fine meshes for mixed triangular and quadrilateral cells. . 75

34 Coarse mesh for the NACA 0012 airfoil. 80

35 Three level multigrid meshes. The coarse mesh is plotted in red, the medium mesh in blue, and the fine mesh in green. 81

36 Maximum residual for one-level grid and multigrid flow solvers. . . . 82

37 Maximum Mach number for one-level grid and multigrid flow solvers. 83

FIGURE Page

38 Lift on NACA 0012 airfoil for one-level grid and multigrid flow solvers. 84

39 Maximum residual for one-level grid and multigrid flow solvers. . . . 85

40 Maximum Mach number for one-level grid and multigrid flow solvers. 88

41 Lift on NACA 0012 airfoil for one-level grid and multigrid flow solvers. 89

42 Mesh used for the turbulent flat plate simulation. 92

43 Turbulent boundary layer for flat plate. 95

44 NACA 0012 pressure coefficients. 97

45 NACA 65-A-004.8 airfoil and O-grid for the F-5 wing. 99

46 Three-dimensional view of the surface of the F-5 wing. 100

47 Steady F-5 wing, station 2. 101

48 Steady F-5 wing, station 5. 101

49 Pitching angle as a function of time. 103

50 Unsteady F-5 at Mach = 0.6 and frequency = 20 Hz, real and imaginary components of pressure coefficients, station 2. 104

51 Unsteady F-5 at Mach = 0.6 and frequency = 20 Hz, real and imaginary components of pressure coefficients, station 5. 105

52 Two-dimensional aeroelastic wing model with two degrees of freedom. 106

53 Schematic view of the NATA wing setup. 106

54 Isometric view of the NATA wing setup. 107

55 Pitch amplitudes for NATA wing with Coulomb damping. 109

56 First generalized modal deformations for Heavy Goland Wing at Mach = 0.09. 110

57 Linear versus non-linear structural model: Out-of-plane and torsional modal amplitudes. 112

FIGURE Page

58 Linear versus non-linear structural model: In-plane modal amplitudes. 112

59 Modal amplitudes of out-of-plane bending for Original Goland Wing at velocities ranging from 300 ft/sec to 350 ft/sec. 114

60 Torsional modal amplitudes for Original Goland Wing at velocities ranging from 300 ft/sec to 350 ft/sec. 114

CHAPTER I

INTRODUCTION

A. Statement of work

The objective of this research was to develop a numerical method for a fully nonlinear aeroelastic analysis. The aeroelastic model consisted of: (1) an unsteady, viscous aerodynamic model that captures compressible flow effects for transonic flows with shock/boundary layer interaction, (2) a nonlinear structural model that captures in-plane, out-of-plane, and torsional couplings, and (3) a solution methodology that assures synchronous interaction between the nonlinear structure and fluid flow, including a consistent geometric interface between the highly-deforming structure and flow field.

High fidelity aerodynamic models like the Euler or Navier-Stokes equations have a high computational cost in the order of weeks or months. The computational cost is affected by the large time scales involved in the aeroelastic simulations, where in some cases tens of seconds of real time have to be simulated to capture the physics of the problem. Different approaches are available to reduce the computational cost of the aeroelastic simulations. The most simple aerodynamic model is the quasi-steady approach, in which the effective angle of attack depends upon the rates of displacement and rotations of the airfoil. The transonic small disturbance [1] model solves the transonic potential flow equations and it has a relatively low computational cost. Proper Orthogonal Decomposition (POD) can be applied to the Euler or Navier-Stokes equations and is used to convert the partial differential equations (PDEs) into a set of ordinary differential equations (ODEs) which are considerably cheaper to

---

The journal model is *AIAA Journal.*

solve.

Parallelization techniques allow a direct reduction of the computational time by distributing the computational effort among several computers. To achieve substantial computational time reductions and high parallel efficiencies the loads on the processors must be balanced. To obtain balanced processor loads the mesh used in this work was constructed with layers that were topologically identical, thus each processor was equally loaded and processor communication was kept to a minimum.

As the wing deformed, the computational domain followed the new shape of the wing. A new mesh could be generated for each wing configuration, but this approach would be computationally expensive and penalize the turnaround time for the aeroelastic solver. Instead of generating a new mesh for each wing configuration, the same original mesh was deformed without changing the mesh connectivity. A moving mesh algorithm was developed that allowed a robust and efficient strategy to deform the mesh without penalizing the quality of the mesh.

B. Aeroelastic problem

Aeroelasticity is the discipline concerned with the physical interaction among inertial, elastic and aerodynamic forces [2]. The aeroelastic phenomena can be visualized as the result of the interaction between these three forces, as shown in Fig. 1.

The intersection between aerodynamic and elastic forces is the static aeroelasticity discipline. Common static aeroelastic phenomena are divergence, reversal control, and lift effectiveness. Divergence is a static aeroelastic instability in which the deformation-dependent aerodynamic forces or loads exceed the elastic restoring capabilities of the structure. Reversal control is another static aeroelastic instability in which maneuver loads (roll, pitch, yaw) due to a control input (aileron, elevator,

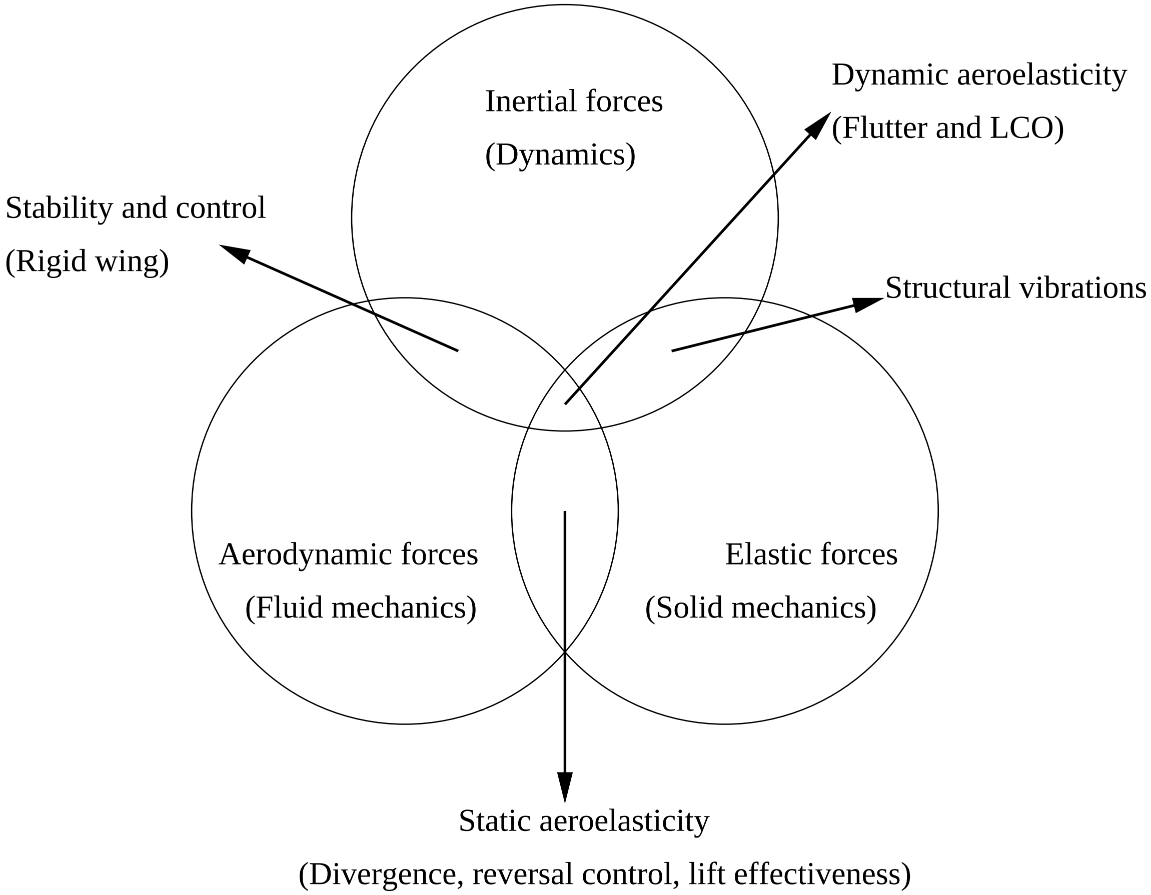


Fig. 1. Aeroelasticity as an interdisciplinary interaction.

rudder) are lost due to the flexibility of the primary structure. As the control surface deflects, the flexible primary structure deflects in the opposite direction in such a way that the total resulting control forces become zero. Lift effectiveness is the change of lift characteristics due to the flexibility of the structure.

The intersection between the aerodynamic and inertial forces is the stability and control discipline, or flight mechanics, where the structure is assumed to be infinitely rigid. The intersection between inertial and structural forces is the structural dynamics discipline, which deals with structural vibrations.

The intersection between the aerodynamic, structural and inertial forces is the dynamic aeroelasticity discipline. In this case the interactions between the flow and the structure are taken into account, while being influenced by the dynamic characteristics of the wing. Flutter and Limit Cycle Oscillation (LCO) are phenomena that occur when the aerodynamic loads excite one or more structural modes. The flutter instability is characterized by a sudden growth in the amplitude of the deformation, which growths unbounded until ultimately the wing breaks down. The LCO instability is characterized by a growing amplitude of the deformations, until a certain limit is reached, the LCO amplitude. This LCO amplitude is dependent on the structural parameters and the flow conditions.

While linear aeroelastic models are capable of predicting divergence, control surface reversal, and flutter, they are not capable of predicting the onset of LCO. Figure 2 presents a typical amplitude versus velocity plot. If the velocity of the airplane is smaller than the flutter velocity $V_{Flutter}$, the linear analysis predicts a zero deformation, or given any initial deformation, the deformation will be reduced in time. If the velocity is equal to or higher than the $V_{Flutter}$, the linear theory predicts a continuous growth in the amplitude of the deformations, until ultimately the wing breaks down.

Nonlinear aeroelastic analysis is capable of providing much richer insights into the aeroelastic phenomena. From Fig. 2 it can be seen that non-zero amplitudes of deformations can be present even at speeds below the flutter velocity $V_{Flutter}$. The plot also shows that for a velocity below $V_{Flutter}$, if the initial deformations are below a certain threshold, the deformations will be reduced in time. However, once the initial deformations grow beyond this threshold, a limit cycle oscillation phenomenon occurs, in which the wing will continuously oscillate at a certain amplitude and frequency. The deformations are then finite and confined, but in order to eliminate this LCO motion the velocity needs to drop considerably below the $V_{Flutter}$ velocity.

The fundamental issues associated with managing nonlinear aeroelastic effects

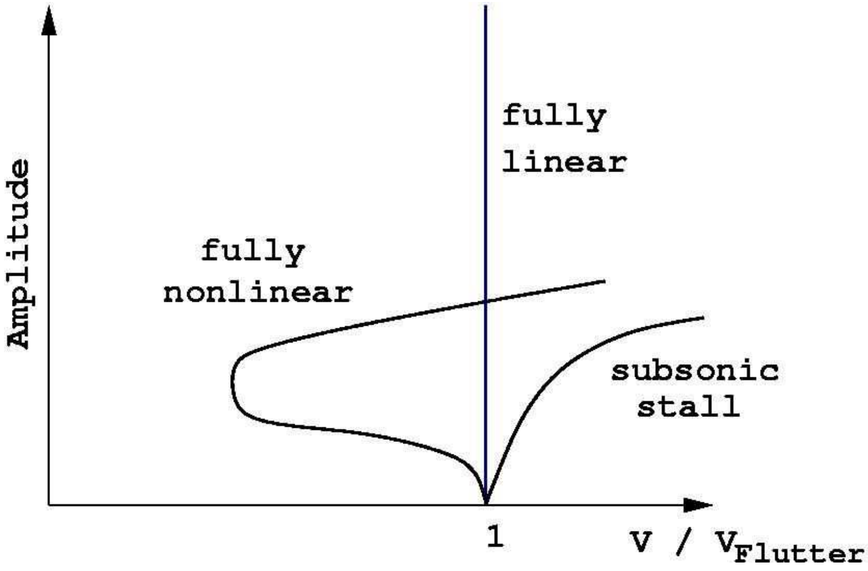


Fig. 2. Linear and nonlinear aeroelastic response [3].

are present in configurations of very flexible wings. Active wing technologies, such as those explored in the Active Aeroelastic Wing (AAW) program, utilize pronounced twisting and bending to achieve desired aerodynamic loads. Structural nonlinearities that are inherent in the flexible, high-aspect-ratio wing are exacerbated in envisioned joined-wing configurations. Nonlinear responses driven by these phenomena are anticipated to be of very large amplitudes.

In recent years, studies of nonlinear fluid-structure interactions have been motivated by evidence that there are adverse aeroelastic responses attributed to system nonlinearities. For example, limit-cycle oscillations (LCOs) occur in nonlinear aeroelastic systems and remain a persistent problem on fighter aircraft with store configurations. Nonlinear phenomena such as LCOs have been observed on the F16 aircraft [4, 5]. Such LCOs are unacceptable since aircraft performance, aircraft certification, mission capability, and human factor issues such as pilot fatigue are adversely affected. The existence of residual pitch oscillations (RPOs) has also been found during flight tests of the B-2 [6]. The RPO phenomenon is a persistent small amplitude oscillation, and is linked to the aeroelastic nonlinearities associated with the B-2 airplane.

C. Background

This section presents background information on aerodynamic models typically used in aeroelastic analysis. This section targets at the aerodynamic models since the focus of this dissertation is in the aerodynamic modeling of the aeroelastic problem.

Different approaches have been employed to model the aerodynamics of the aeroelastic problem. The unsteady vortex lattice model was used to model the aerodynamic loads on the wing [7]. Point vortices were placed on the wing and in the wake. The strength of the discrete vortices was calculated by specifying that the velocity induced by the vortices had to be equal to the downwash arising from the unsteady motion of the wing.

Unsteady transonic small disturbance theories were introduced in the 1980s and solved the three-dimensional potential-flow equations [8, 9]. A more rigorous development (CAPTSDv) coupled the inviscid transonic small disturbance theories with an integral boundary layer model [10]. The outer inviscid flow solution provided the surface pressure distribution needed to solve the boundary layer equations. The boundary layer thickness distribution calculated was used to modify the airfoil surface tangency boundary condition for the inviscid flow solver.

More advanced flow models involve solving the Euler or Navier-Stokes equations that govern the fluid motion. Among the Euler flow solvers developed there is the Computational-Structural-Mechanics (CSM) code developed at the University of Colorado [11]. The flow was modeled using the Arbitrary Lagrangian-Eulerian (ALE) conservative formulation of the Euler equations.

The Euler or Navier-Stokes flow solvers can be divided into structured or unstructured, depending on the approach taken to discretize the computational domain. Structured solvers are normally less complex and more efficient than unstructured solvers. Unstructured solvers have the advantage of being much more flexible for discretizing the domain, since the unstructured cells have more freedom to adjust to arbitrary boundaries. Structured Navier-Stokes flow solvers include the Euler/Navier-Stokes 3D Aero-Elasticity (ENS3DAE) code [12] and the Computational Fluids Laboratory 3-Dimensional (CFL3D) flow solver [13]. Unstructured Navier-Stokes include the Air Force Air Vehicles Unstructured Solver (AVUS) [14].

The ENS3DAE flow solver was developed at Lockheed Martin [15, 16]. ENS3DAE solved the three-dimensional compressible flow modeled by the Euler or Reynolds-averaged Navier-Stokes equations. Central finite differences were used for spatial discretization with blended second and forth-order dissipation terms. Turbulence effects were modeled using the Baldwin-Lomax algebraic turbulence model or the Johnson-King model. The flow solver was parallelized to reduce the computational time. The solver accepted either single or multiple-block curvilinear structured grids and required a one-to-one match of grid points at block interfaces.

The CFL3D solver [13] is a Reynolds-averaged thin-layer Navier-Stokes flow solver for structured grids. The spatial discretization involved a semi-discrete finite-volume approach. Upwinding-biasing was used for the convective and pressure terms, while central differencing was used for the shear stress and heat transfer terms. Numerous turbulence models were provided, including zero, one, and two-equation models. Multiple-block topologies were possible with the use of one-to-one blocking (that is, nodes from different blocks shared a face having a one-to-one correspondence), patching, overlapping, and embedding.

The Air Vehicles Unstructured Solver (AVUS) [14], formerly known as Cobalt$_{60}$, is a finite-volume, cell-centered, second-order accurate Euler/Navier-Stokes flow solver. Second-order accuracy was achieved using linear variation of the solution within each cell and least squares with QR decomposition to compute the solution gradients. The turbulence model used was Menter's two equation Shear-Stress Transport (SST) model [17]. The grid can be composed of cells of arbitrary type.

D. Flow solver

The flow solver used in this work was developed by Cizmas and Han [18] and is an unstructured mixed-type grid with a finite-volume discretization. The code is capable of solving the Euler and Navier-Stokes equations using a two-equation $k-\omega$ SST turbulence model [17]. The mixed-type grid allows an O-grid close to the surface of the bodies to have better control of the discretization in the boundary layer region. An unstructured mesh was generated outside of the structured mesh to efficiently fill the computational domain.

E. Grid generation and grid deformation algorithms

High-fidelity flow solvers for aeroelastic applications require the use of computational meshes that deform as the structure is being displaced. High-aspect-ratio wings increase the demands on the robustness of the deforming mesh algorithm because these wings are extremely flexible and attain deformations that are a significant fraction of the span of the wing. Besides being robust, the mesh deforming algorithm must be computationally inexpensive to avoid penalizing the turnaround time of the aeroelastic computations. A methodology was developed herein to generate a robust and efficient grid deformation algorithm for wings with large deformations.

F. Parallel algorithm

The coupled flow-structure solver was parallelized to reduce the computational time. The computational domain was divided into topologically identical layers that spanned from the root to past the tip of the wing. The fact that the layers were topologically identical simplified the parallel algorithm and increased the parallelization efficiency. The message-passing interface (MPI) standard libraries were used for the interprocessor communication.

G. Multigrid implementation

To reduce the computational time of the flow solver, a three-level multigrid technique was implemented in the flow solver code. This technique is based on the solution of the governing equations on a series of successively coarser grids which reduce the high-frequency components of the solution error and accelerate the convergence of the solution on the finer grid. The evolution on the coarser grids is driven by the residuals on the finer grid, therefore maintaining the overall accuracy of the finer grid. The correction obtained for the coarse grid is then interpolated to the finer grid using an interpolation scheme.

H. Original contributions of the present work

- Development of a robust and efficient grid deformation algorithm.
- Update and validation of an unstructured finite-volume flow solver.
- Integration of the flow solver with the structural solver, load calculation, and grid deformation algorithms.
- Implementation of a parallelization strategy for the flow solver.

- Implementation of multigrid techniques.

I. Outline of dissertation

Chapter II presents the aerodynamic and structural models of the aeroelastic phenomena. Chapter III describes the numerical methods used to solve the problem. This chapter includes the mesh generation and deformation algorithms, the implementation of the flow solver, the parallelization strategy, and the multigrid technique. Chapter IV presents the results for the validation and verification of the flow solver and aeroelastic results for the F-5 wing, the Nonlinear Aeroelastic Test Apparatus (NATA), and the Goland wing. Chapter V presents the conclusions and the suggested future work.

CHAPTER II

PHYSICAL MODEL

The first part of this chapter presents the aerodynamic model, which consisted of the Reynolds-averaged Navier-Stokes equations and a two-equation eddy-viscosity turbulence model. The second part describes the equations of motion for the wing modeled as a nonlinear beam. The third part describes the coupling of the aerodynamic and structural models.

A. Aerodynamic model

1. Reynolds-averaged Navier-Stokes equations

The flow around the deforming wing was modeled as unsteady, compressible and viscous using the mass, momentum and energy conservation equations. These equations are collectively known as the Navier-Stokes equations.

The mass conservation equation states that mass cannot be created nor it can disappear, and is expressed as [19]

$$\frac{\partial}{\partial t}\int\limits_{V} \rho \cdot dV + \oint\limits_{S} \rho \cdot (\vec{v} \cdot \vec{n}) \cdot dS = 0, \tag{2.1}$$

where V is the cell control volume, ρ is the flow density, $\vec{v}$ is the velocity of the flow, $\vec{n}$ is the unit normal vector to the cell surface, and S is the area of the face.

The first term of Eq. (2.1) represents the time rate of change of the total mass inside the cell. The second term represents the mass flow of the fluid through the surface of the cell.

The linear momentum conservation equation is expressed as [19]

$$\frac{\partial}{\partial t}\int\limits_V \rho \cdot \vec{v} \cdot dV + \oint\limits_S \rho \cdot \vec{v} \cdot (\vec{v} \cdot \vec{n}) \cdot dS = \int\limits_V \rho \cdot \vec{g} \cdot dV + \oint\limits_S (-p \cdot \vec{n} + \tau \cdot \vec{n}) \cdot dS, \tag{2.2}$$

where $\vec{g}$ is the body force per unit mass, p is pressure imposed by the fluid on the boundary, and τ are the shear and normal stresses, resulting from the friction between the fluid and the boundary surface.

The first term in Eq. (2.2) represent the time variation of linear momentum. The second term represents the convective momentum flux or the transfer of momentum across the boundary of the control volume. The third term represents the volume (or body) forces. The last term represents the surface forces.

The energy conservation equation is expressed as [19]

$$\begin{aligned}\frac{\partial}{\partial t}\int\limits_V \rho \cdot E \cdot dV + \oint\limits_S \rho \cdot E \cdot (\vec{v} \cdot \vec{n}) \cdot dS = \int\limits_V (\rho \cdot \vec{g} \cdot \vec{v} + q_h) \cdot dV + \\ + \oint\limits_S k \cdot (\nabla T \cdot \vec{n}) \cdot dS - \oint\limits_S p \cdot (\vec{v} \cdot \vec{n}) \cdot dS + \oint\limits_S (\tau \cdot \vec{n}) \cdot \vec{n} \cdot dS,\end{aligned} \tag{2.3}$$

where E is the total energy per unit of mass, q_h is the heat source, and $\vec{g}$ is the body force per unit mass.

The first term in Eq. (2.3) represent the time variation of the total energy of the fluid in the control volume. The second term represents the convective energy flux or the transfer of energy across the boundary of the control volume. The third term represents the heat sources and the work done by the body forces. The fourth term is called the diffusive or dissipative flux, and it represents the diffusion of heat due to molecular thermal conduction. The last two terms represent the rate of work done by the pressure as well as the shear and normal stresses on the fluid element.

The total energy E per unit mass is expressed as

$$E = e + \frac{|\vec{v}|^2}{2} = e + \frac{u^2 + v^2 + w^2}{2}, \tag{2.4}$$

where e is the internal energy, and u, v, and w are the x-, y-, and z-components of the velocity vector $\vec{v}$.

The internal energy e for a calorically perfect gas is expressed as

$$e = c_v \cdot T, \tag{2.5}$$

where c_v is the specific heat at constant volume and T is the temperature of the fluid.

The total enthalpy H per unit mass is expressed as

$$H = h + \frac{|\vec{v}|^2}{2} = E + \frac{p}{\rho}, \tag{2.6}$$

where h is the internal enthalpy, p is the pressure, and ρ is the density of the fluid.

The internal enthalpy h for a calorically perfect gas is expressed as

$$h = c_p \cdot T, \tag{2.7}$$

where c_p is the specific heat at constant pressure and T is the temperature of the fluid.

Using the total enthalpy Eq. (2.6), the energy conservation equation Eq. (2.3) is expressed as

$$\begin{gathered}\frac{\partial}{\partial t} \int\limits_V \rho \cdot E \cdot dV + \oint\limits_S \rho \cdot H \cdot (\vec{v} \cdot \vec{n}) \cdot dS = \int\limits_V (\rho \cdot \vec{g} \cdot \vec{v} + q_h) \cdot dV + \\ + \oint\limits_S k \cdot (\nabla T \cdot \vec{n}) \cdot dS + \oint\limits_S (\tau \cdot \vec{n}) \cdot \vec{n} \cdot dS,\end{gathered} \tag{2.8}$$

where the convective $(\rho E \vec{v})$ and pressure $(p\vec{v})$ terms have been gathered into the (ρH)

term.

The Navier-Stokes equations are complemented with the equation of state for a perfect gas

$$P = \rho \cdot R \cdot T, \tag{2.9}$$

where p is the pressure, ρ is the density, R is the gas constant, and T is the temperature.

Other useful relationships are $R = c_p - c_v$ relating the gas constant, R, to the specific heat coefficients at constant pressure and volume. The ratio of specific heat capacities is $\gamma = c_p/c_v$. With these relationships, the pressure p is expressed in terms of the total energy E as

$$p = (\gamma - 1)\rho \left[E - \frac{u^2 + v^2 + w^2}{2} \right] \tag{2.10}$$

2. Turbulence model

The turbulence effects were modeled by using the two-equation eddy-viscosity Shear Stress Transport (SST) model of Menter [17]. The turbulent eddy viscosity μ_T was calculated as the ratio between the turbulence kinetic energy k and the specific dissipation of turbulence ω

$$\mu_T = \rho \cdot \frac{k}{\omega} \tag{2.11}$$

Two additional equations have to be solved to calculate the values of k and ω. These equations are the transport equations for the turbulent kinetic energy and the specific dissipation of turbulence.

The transport equation for the turbulent kinetic energy is expressed in differential form as [17]

$$\frac{\partial \rho k}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho v_j k\right) = \frac{\partial}{\partial x_j}\left[\left(\mu_L + \sigma_k \mu_T\right) \cdot \frac{\partial k}{\partial x_j}\right] + \tau_{ij} \cdot S_{ij} - \beta^* \rho \omega k, \tag{2.12}$$

where μ_L is the molecular viscosity, μ_T is the eddy viscosity, τ_{ij} are the turbulent stresses, S_{ij} is the strain-rate tensor, and $\beta^* = 0.09$ is a turbulent model parameter.

The first term in Eq. (2.12) represents the time variation of the turbulent kinetic energy of the fluid in the control volume. The second term represents the convective turbulent kinetic energy flux or the transfer of turbulent kinetic energy across the boundary of the control volume. The third term represents the conservative turbulent kinetic energy diffusion. The fourth term is the eddy-viscosity production of turbulent kinetic energy. The last term in this equation is the turbulent kinetic energy dissipation.

The transport equation for the specific dissipation of turbulence is expressed in differential form as

$$\begin{gathered}\frac{\partial \rho \omega}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho v_j \omega\right) = \frac{\partial}{\partial x_j}\left[\left(\mu_L + \sigma_w \mu_T\right) \cdot \frac{\partial \omega}{\partial x_j}\right] + \frac{C_w \rho}{\mu_T} \cdot \tau_{ij} \cdot S_{ij} - \beta \rho \omega^2 + \\ +2\left(1 - f_1\right) \cdot \frac{\rho \sigma_{w2}}{w} \cdot \frac{\partial k}{\partial x_j} \cdot \frac{\partial \omega}{\partial x_j},\end{gathered} \tag{2.13}$$

where C_w and β are turbulent model parameters. The function f_1 is called the blending function because it blends the model coefficients of the $k - \omega$ model in boundary layers with the model coefficients of the transformed $k - \epsilon$ model in free-shear layers and freestream zones.

The first term in Eq. (2.13) represents the time variation of the specific dissipation of turbulence of the fluid in the control volume. The second term represents the convective specific dissipation of turbulence flux or the transfer of specific dissipation of turbulence across the boundary of the control volume. The third term represents the conservative specific dissipation of turbulence diffusion. The fourth term is the eddy-viscosity production of specific dissipation of turbulence. The fifth term represents the specific dissipation of turbulence dissipation. The last term in this equation represents the turbulent cross diffusion.

The blending function f_1 is expressed as [17]

$$f_1 = tanh\left(arg_1^4\right) \tag{2.14}$$

The value of arg_1 is obtained from the following equation [17]

$$arg_1 = min\left[max\left(\frac{\sqrt{k}}{0.09\omega d}, \frac{500\mu_L}{\rho\omega d^2}\right), \frac{4\rho\sigma k}{CD_{kw}d^2}\right], \tag{2.15}$$

where d is the distance from the cell to the nearest wall. CD_{kw} is the positive part of the cross diffusion term in Eq. (2.13) and is expressed as [17]

$$CD_{kw} = max\left(2 \cdot \frac{\rho\sigma_{w2}}{w} \cdot \frac{\partial k}{\partial x_j} \cdot \frac{\partial \omega}{\partial x_j}, 10^{-20}\right) \tag{2.16}$$

The constants for the turbulent model are calculated using the blending function f_1 from Eq. (2.14) and the following blending equation [17]

$$\phi = f_1 \cdot \phi_1 + (1 - f_1) \cdot \phi_2, \tag{2.17}$$

where ϕ can be any parameter that needs to be blended, like σ_k, σ_ω, β, and C_w. The coefficients for the inner model ($k-\omega$) are given by $\sigma_{k_1} = 0.85$, $\sigma_{\omega_1} = 0.5$, $\beta_1 = 0.075$, and $C_{w_1} = 0.533$. The coefficients for the outer model ($k - \epsilon$) are given by $\sigma_{k_2} = 1.0$, $\sigma_{\omega_2} = 0.856$, $\beta_2 = 0.0828$, and $C_{w_2} = 0.44$.

B. Structural model

The nonlinear behavior of the cantilever wing was modeled using a nonlinear beam theory. This theory accounted for bending about multiple axes, as well as torsional coupling. The nonlinear equations of motion for a cantilever wing were derived from the equations of motion for flexural-flexural-torsional vibrations. The formulation followed an approach developed by Crespo da Silva [20]. This approach contained structural coupling terms and included both quadratic and cubic nonlinearities due to curvature and inertia.

Figure 3 shows a beam segment of length s. In this figure X-Y-Z are the undeformed or reference axes, and ξ-η-ζ are the deformed axes. Figure 3 also shows the in-plane bending v and the out-of-plane bending w. The torsion ϕ was taken about the deformed elastic axis ξ.

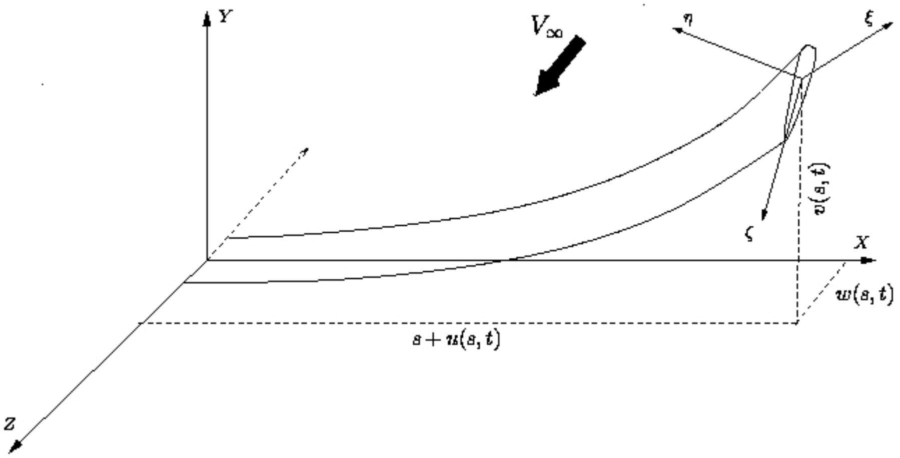


Fig. 3. Structural model of the wing [3].

The beam was assumed to be inextensional lengthwise. This constraint was expressed mathematically by the following relation

$$(1+u')^2+v'^2+w'^2=1 \tag{2.18}$$

This constraint, although artificial, is a valid approximation since the extensional stiffness of the wing is considerably larger than the in-plane and out-of-plane stiffnesses.

The equations of motion for in-plane bending w, out-of-plane bending v, and torsion ϕ response were expressed as [3]

$$\begin{gathered} m\ddot{w}-I_\eta\ddot{w}''+D_\eta w^{IV}=G'_w+F_{A_w} \\ m\ddot{v}-me\ddot{\phi}-I_\zeta\ddot{v}''+D_\zeta v^{IV}=G'_v+F_{A_v} \\ I_\xi\ddot{\phi}-me\ddot{v}-D_\xi\phi''=G_\phi+M \end{gathered} \tag{2.19}$$

where m is the mass of the wing per unit length, I is the moment of inertia, D is the stiffness coefficient, e is the center of gravity offset from the elastic axis. G'_w, G'_v and G_ϕ are the structural nonlinear components [3]. F_{A_w} and F_{A_v} are the aerodynamic forces and M is the aerodynamic moment.

In Eqs. (2.19), the linear terms were written on the left-hand side and the corresponding nonlinear terms were expressed as G'_w, G'_v, and G_ϕ. The nonlinear structural terms were expanded using a Taylor series about the static undeformed wing configuration. Terms up to third order were retained in the final expression. Higher order nonlinearities were neglected assuming they would have negligible effect on the system dynamics. The formulation for the nonlinear components are presented in the following expressions

$$\begin{aligned} G'_w &= \{D_\xi \left(\phi' + v''w'\right) v'' - \left[\left(D_\eta - D_\zeta\right)\left(\phi v'' + \phi^2 w''\right)\right]' \\ &\quad -w'\left(D_\zeta v''^2 + D_\eta w''^2\right) - I_\xi \left(\dot{\phi} + \dot{v}'w'\right)\dot{v}' - \left[\left(I_\eta - I_\zeta\right)\left(\phi\dot{v}' + \phi^2\dot{w}\right)\right]^{\bullet} \\ &\quad +w'\left(I_\zeta \dot{v}'^2 + I_\eta \dot{w}'^2\right) + \lambda w'\}' \end{aligned} \tag{2.20}$$

$$\begin{aligned} G'_v &= \{-D_\xi \left(\phi' + v''w'\right) w'' - \left[\left(D_\eta - D_\zeta\right)\left(\phi^2 v'' - \phi w'' - v'w'w''\right)\right]' \\ &\quad -v'\left(D_\zeta v''^2 + D_\eta w''^2\right) + I_\xi \left(\dot{\phi} + \dot{v}'w'\right)\dot{w}' + \left[\left(I_\eta - I_\zeta\right)\left(\phi^2\dot{v}' - \phi\dot{w}' - v'w'\dot{w}'\right)\right]^{\bullet} \\ &\quad +v'\left(I_\zeta \dot{v}'^2 + I_\eta \dot{w}'^2\right) + \lambda v' + w'qb^2 C_M\}' \end{aligned} \tag{2.21}$$

$$\begin{aligned} G_\phi &= D_\xi (w'v'')' - \left(D_\eta - D_\zeta\right)\left[(v''^2 - w''^2)\phi - v''w''\right] \\ &\quad -I_\xi (w'\dot{v}')^{\bullet} + \left(I_\eta - I_\zeta\right)\left[(\dot{v}'^2 - \dot{w}'^2)\phi - \dot{v}'\dot{w}'\right] \end{aligned} \tag{2.22}$$

where $\dot{}$ and $\bullet$ denote time derivatives, and $'$ denotes spatial derivatives.

The solution was assumed to be separable in space and time. The variables were expressed in series form as follows

$$w(x,t) = \sum_{i=1}^{\infty} W_i(x) w_i(t), \tag{2.23}$$

$$v(x,t) = \sum_{j=1}^{\infty} V_j(x) v_j(t), \tag{2.24}$$

$$\phi(x,t) = \sum_{k=1}^{\infty} A_k(x) \phi_k(t) \tag{2.25}$$

In these expressions, the capitalized terms, U_i, W_j and A_k represent the shape functions derived from a vibrating, nonrotating uniform cantilever beam, and they were defined as

$$\begin{aligned} W_i(x) = F_i(x) &= \cosh(\frac{\beta_i x}{L}) - \cos(\frac{\beta_i x}{L}) - \sigma_i \left[\sinh(\frac{\beta_i x}{L}) - \sin(\frac{\beta_i x}{L})\right] \\ V_j(x) = F_j(x) &= \cosh(\frac{\beta_j x}{L}) - \cos(\frac{\beta_j x}{L}) - \sigma_j \left[\sinh(\frac{\beta_j x}{L}) - \sin(\frac{\beta_j x}{L})\right] \\ A_k(x) &= \sqrt{2}\sin(\frac{\gamma_k x}{L}) \end{aligned} \tag{2.26}$$

where β and γ are the roots of the characteristics equations for pure bending and torsion respectively, and σ is a function of β.

The $F_i(s)$, $F_j(s)$ and $A_k(s)$ in Eq. (2.26) are the shape functions or mode shapes for in-plane bending, out-of-plane bending and torsion motion respectively.

The root of the characteristic equation for pure bending was defined as

$$1 + \cos(\beta_i)\cosh(\beta_i) = 0 \tag{2.27}$$

The root of the characteristic equation for pure torsion was defined as

$$\sin(\gamma_k) = 0 \tag{2.28}$$

The parameter σ_i was defined as

$$\sigma_i = \frac{\cosh(\beta_i) + \cos(\beta_i)}{\sinh(\beta_i) + \sin(\beta_i)} \tag{2.29}$$

Figure 4 shows the first and second bending and torsion mode shapes.

By using the Galerkin method, the corresponding ordinary differential equations (ODEs) were obtained from the partial differential equations (2.19). The procedure was applied to obtain linear mass, damping and stiffness matrices. These matrices were time invariant and they were computed at the beginning of the algorithm only once.

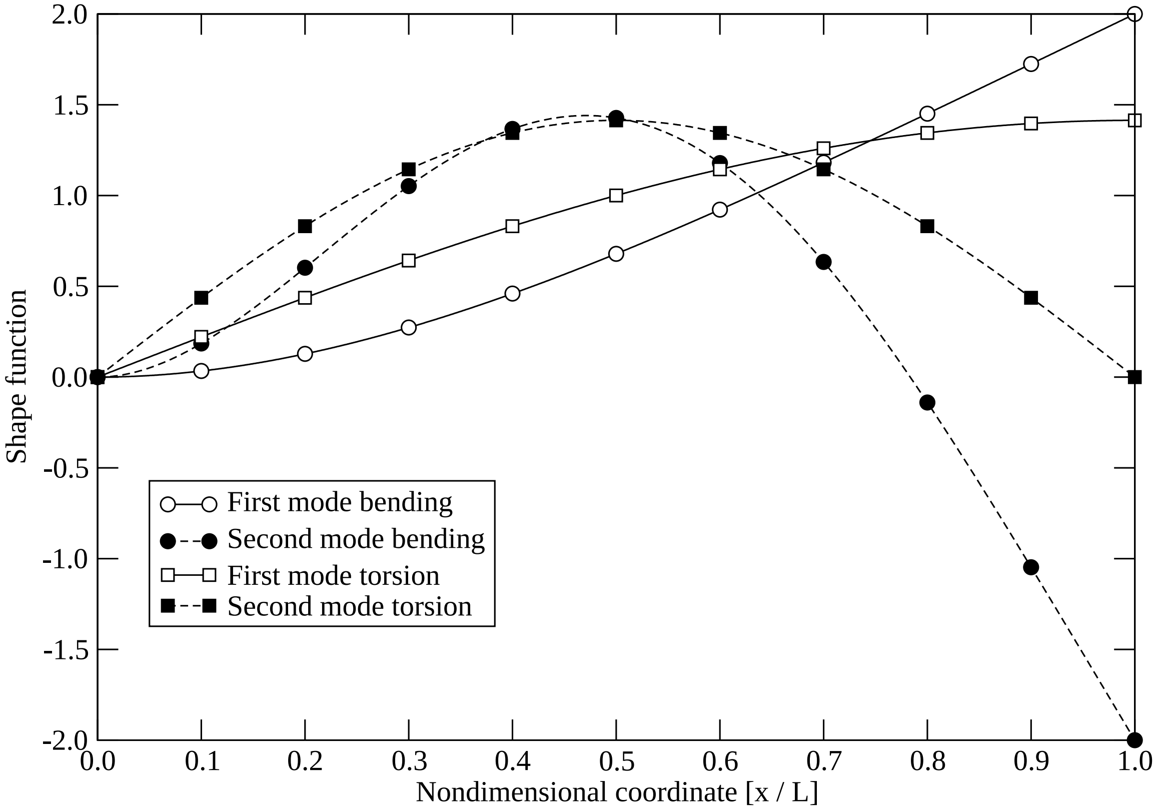


Fig. 4. Modal shape functions for bending and torsion.

The ODEs obtained were expressed in matrix form as follows [3]

$$[M_L]\left\{\begin{array}{c}\ddot{w}\\ \ddot{v}\\ \ddot{\phi}\end{array}\right\}+[C_L]\left\{\begin{array}{c}\dot{w}\\ \dot{v}\\ \dot{\phi}\end{array}\right\}+[K_L]\left\{\begin{array}{c}w\\ v\\ \phi\end{array}\right\}=[M_{NL}]\left\{\begin{array}{c}\ddot{w}\\ \ddot{v}\\ \ddot{\phi}\end{array}\right\}+F_{NL}+F_A \quad (2.30)$$

where F_{NL} is the vector of the structural nonlinear terms excluding the inertia terms. F_A is the vector of aerodynamic forces and moments obtained after the application of Galerkin procedure to the actual physical loads. This procedure interpolates the aerodynamic loads between the modal degrees of freedom. Equation (2.30) was integrated in time using a Runge-Kutta algorithm.

C. Coupling of the aerodynamic and structural models

The coupling of aerodynamic and structural models was done using a tightly-coupled solution [21] which allowed the two models to communicate during every time step. After the structural equations of motion were solved for the incremental deflections, the wing location and the flow solver grid were updated and a new aerodynamic solution was obtained at the next time step using these deflections. The structural model loads lagged the aerodynamic forces by one time step. This time step was considered to be small enough such that no significant phase lags were introduced.

The aerodynamic loads were generated by integrating the pressure over the wing surface. The pressure values were obtained from the unsteady flow solver. The aerodynamic loads were then passed to the structural solver and were used to calculate the wing deformation. The coordinates of the new position of the wing were passed to the grid deformation algorithm which updated the grid of the flow solver. The flow solution was then calculated and the aerodynamic loads generated for a new wing position.

CHAPTER III

NUMERICAL METHOD

The first part of this chapter presents the details of the mesh generation algorithm. The chapter continues with the details of the Reynolds-averaged Navier-Stokes flow solver implementation, the mesh deformation algorithms, the parallel algorithm implementation, and finalizes with the multigrid technique implementation on the flow solver.

A. Mesh generation algorithm

This section presents the details of the mesh generation algorithm. The mesh generation algorithm presented herein was developed to satisfy the following requirements: (1) allow a good control of the grid size in the boundary layer, (2) allow large wing deformations without the need to remesh the domain, (3) reduce the computational time for grid deformation as the wing deforms, and (4) facilitate parallel computation. As a result of these requirements, the grid generation and deformation algorithm used: (a) layers of topologically identical elements in the spanwise direction, (b) a structured O-grid around the wing surface, and (c) an unstructured grid outside of O-grid mesh that deformed using the spring analogy method.

The topologically identical layers spanned from the root of the wing past the tip of the wing to the far-field boundary. The topologically identical layers simplified the mesh deformation algorithm and made the parallelization of the flow solver associated with this mesh more efficient. Each layer of the mesh had a hybrid configuration. A structured O-grid was constructed around the wing, to have better control of the mesh in the viscous region. An unstructured grid was constructed outside the O-grid to fill the rest of the layer [22]. Figure 5 shows the structured O-grid generated around the wing. Note how the size of the elements grow from smaller close to the surface of the wing to larger at the O-grid boundary. Figure 6 shows the O-grid and unstructured mesh, with a close up and whole domain views. The figure on the left shows the smooth transition of the element size in the unstructured mesh. The triangles are of comparable size to the outer layer of quadrilaterals at the O-grid boundary, and grow larger away from the wing. Figure 6 on the right shows the whole domain, where the O-grid is barely visible since it is much smaller than the domain size. Note how the unstructured mesh covers the domain with relatively few elements that grow in size as the distance to the wing increases. Figure 7 shows the original undeformed shape of the mesh. Note the identical topology of the layers of the mesh. This mesh has 16 layers that span from the root of the wing past the tip of the wing. For clarity only four layers are shown, which are the layers corresponding to the root, the mid-span, the wing tip, and the far-field boundary sections.

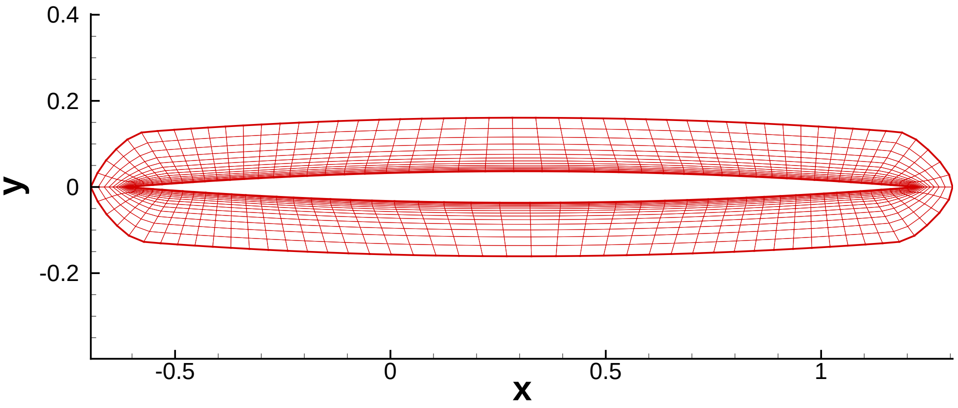


Fig. 5. O-grid generated around the wing.

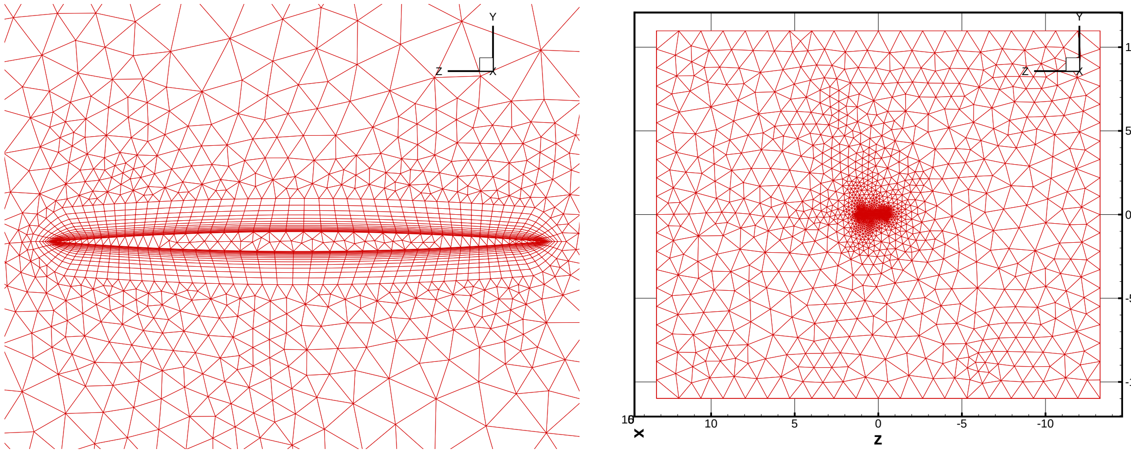


Fig. 6. O-grid and unstructured mesh, close up and whole domain views.

B. Flow solver

The governing equations (2.1, 2.2, 2.8, 2.12 and 2.13) were solved using a finite volume method. The computational domain was discretized using an unstructured mixed mesh. The mesh was deformed according to the wing displacement. Time-marching was used to calculate the unsteady solution. This section presents the discretization of the computational domain, the spatial discretization of the governing equations, including the numerical implementation of the vector fluxes and the second-order upwind scheme, the time integration, and the implementation of the boundary conditions.

1. Navier-Stokes equations

The governing equations of the flow, the Navier-Stokes equations (2.1, 2.2, 2.8), were expressed in vectorial form as [18, 23]

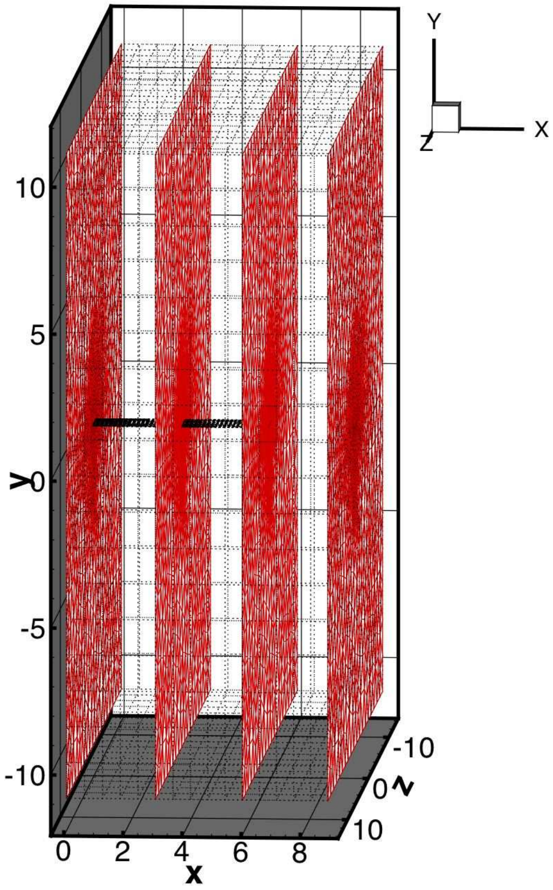


Fig. 7. Mesh with topologically identical layers.

$$\frac{\partial}{\partial t}\int\limits_V \vec{Q}\cdot dV + \oint\limits_S \left(\vec{F}_{conv} - \vec{F}_{vis}\right)\cdot\vec{n}\cdot dS = \int\limits_V \vec{G}\cdot dV, \tag{3.1}$$

where $\vec{Q}$ is the vector of conservative variables

$$\vec{Q} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix} \tag{3.2}$$

$\vec{F}_{conv}$ is the vector of the convective or inviscid fluxes

$$\vec{F}_{conv} = \begin{pmatrix} \rho V \\ \rho u V + n_x p \\ \rho v V + n_y p \\ \rho w V + n_z p \\ \rho H V \end{pmatrix} \tag{3.3}$$

where V is the contravariant velocity or velocity normal to the surface element $V = n_x u + n_y v + n_z w$.

$\vec{F}_{vis}$ is the vector of the viscous fluxes

$$\vec{F}_{vis} = \begin{pmatrix} 0 \\ n_x \tau_{xx} + n_y \tau_{xy} + n_z \tau_{xz} \\ n_x \tau_{yx} + n_y \tau_{yy} + n_z \tau_{yz} \\ n_x \tau_{zx} + n_y \tau_{zy} + n_z \tau_{zz} \\ n_x \Theta_x + n_y \Theta_y + n_z \Theta_z \end{pmatrix} \tag{3.4}$$

where the terms describing the work of viscous stresses and the heat conduction in the fluid are expressed as

$$\begin{aligned} \Theta_x &= u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + \lambda \frac{\partial T}{\partial x} \\ \Theta_y &= u\tau_{yx} + v\tau_{yy} + w\tau_{yz} + \lambda \frac{\partial T}{\partial y} \\ \Theta_z &= u\tau_{zx} + v\tau_{zy} + w\tau_{zz} + \lambda \frac{\partial T}{\partial z} \end{aligned} \tag{3.5}$$

The thermal conductivity coefficient λ is calculated with

$$\lambda = c_p \cdot \frac{\mu}{Pr} \tag{3.6}$$

where Pr is the Prandtl number.

The components of the viscous stresses are

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

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

$$\tau_{zz} = \frac{2}{3}\mu\left(2\frac{\partial w}{\partial z} - \frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}\right) \tag{3.7}$$

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

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

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

2. Turbulence model

The eddy viscosity was modeled by using the two-equation $k-\omega$ Shear Stress Transport (SST) turbulence model of Menter [17]. The time dependent integral form of these equations was expressed in a vectorial form similar to the Navier-Stokes equations

$$\frac{\partial}{\partial t}\int\limits_V \vec{Q}_T \cdot dV + \oint\limits_S \left(\vec{F}_{conv,T} - \vec{F}_{vis,T}\right) \cdot dS = \int\limits_V \vec{G}_T \cdot dV, \tag{3.8}$$

where $\vec{Q}_T$ is the state vector of turbulent conservative variables, $\vec{F}_{conv,T}$ is the vector of turbulent convective fluxes, $\vec{F}_{vis,T}$ is the vector of turbulent viscous fluxes, and $\vec{G}_T$

is the vector of turbulent source terms. The state vector of turbulent conservative variables is

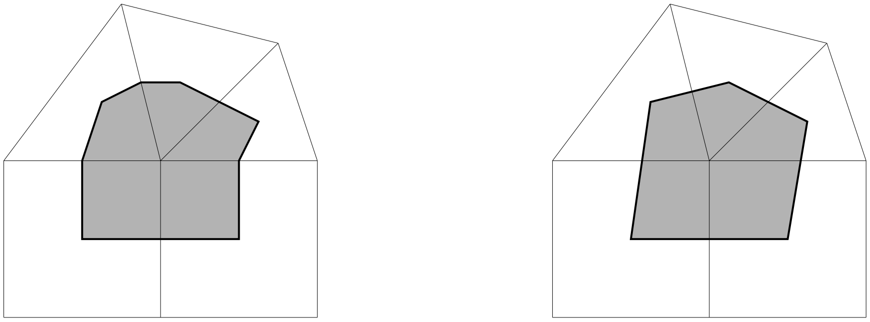
$$\vec{Q}_T = \begin{pmatrix} \rho k \\ \rho \omega \end{pmatrix} \tag{3.9}$$

where k is the turbulence kinetic energy and ω is the specific dissipation rate.

3. Spatial discretization

The computational domain was discretized using hexahedra and triangular prisms cells. The cell-averaged variables were stored at the nodes of the grid, which were the vertices of the cells. The governing equations were discretized using mesh duals as control volumes. Two different mesh duals can be used, the median dual mesh and the centroid dual mesh. The median dual mesh is defined by the surfaces crossing the cell centroid, the face centroids, and the mid-edge points. The centroid dual mesh skips the mid-edge points, and is defined by the surfaces crossing the cell centroid and the face centroids. Figure 8 shows an example of a median dual mesh and a centroid dual mesh. The median dual mesh was adopted because of its flexibility to handle unstructured mixed meshes [24]. Figure 9 shows an example of a dual mesh unstructured mixed grid, where an hexahedral and a prism cells are neighbors and the median dual mesh is constructed in the corner of these two cells.

Fluxes can be calculated using either an edge-based or a cell-based data structure. An edge-based data structure takes advantage of the fact that each edge in the mesh has only one associated face. Consequently, the fluxes through the faces of the mesh are calculated by looping over the edges. In a cell-based data structure, the fluxes are calculated by looping over the cells, then looping over each face of the cells. Thus, in a cell-based data structure, each face is visited twice, one time for each of the



Median dual mesh

Centroid dual mesh

Fig. 8. Median and centroid dual meshes.

two cells that share that face. As a consequence, the number of operations for the cell-based data structure is about twice than the number of operations for the edge-based data structure. In this work the edge-based data structure was utilized for the discretization of the governing equations Eq. (3.1) and Eq. (3.8).

The numerical scheme was node-based because the solution was obtained at the nodes of the mesh. The control volume averaged flow variable $\vec{Q}$ over the volume V_i was calculated as

$$\vec{Q}_i = \frac{\int\limits_{V_i} \vec{Q} \cdot dV_i}{V_i} \tag{3.10}$$

The surface integral of the convective and viscous fluxes was approximated by a sum over the faces of each control volume,

$$\oint\limits_{S} \left(\vec{F}_{conv} - \vec{F}_{vis} \right) \cdot \vec{n} \cdot dS = \sum_{j=k(i)} \left(\vec{F}_{conv} - \vec{F}_{vis} \right) \cdot S_{ij} \tag{3.11}$$

where $k(i)$ is the set of vertices adjacent to node i, $(\vec{F}_{conv} - \vec{F}_{vis})$ is the flux normal to the dual-mesh cell face, and S_{ij} is the cell face surface area. Both the surface area and the flux are associated with the dual mesh face that corresponds to the edge (i, j).

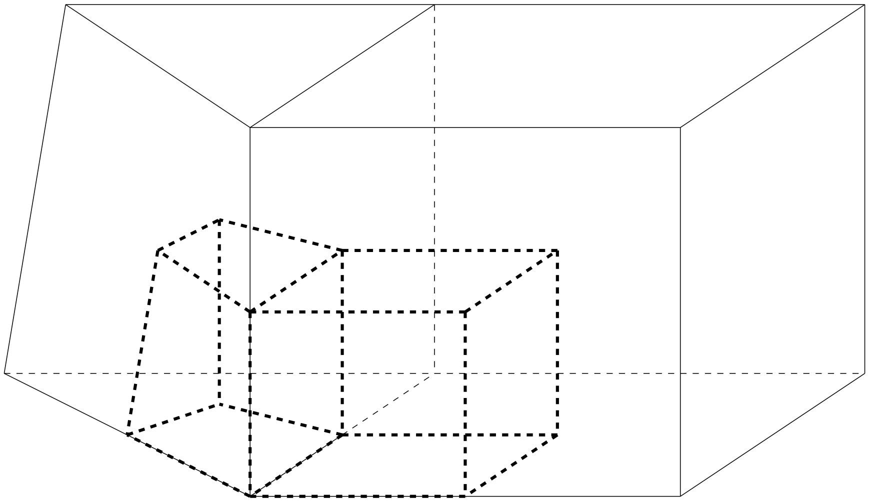


Fig. 9. Median dual mesh for unstructured mixed grids.

Figure 10 shows the dual mesh face associated with the edge (i, j).

The source terms were calculated using the control volume averaged solution and the derivatives of flow variables at the cell centroid,

$$\vec{G}_i = \frac{\int\limits_{V_i} \vec{G} \cdot dV_i}{V_i} \tag{3.12}$$

The term semi-discrete indicates that the conservative variables were represented as an average value and the spatial operators, such as the integrations in space, were approximated by the sum of the contributions over each face. Equation (3.1) may be rewritten in a semi-discrete form as

$$\frac{\partial\left(\vec{Q}_i \cdot V_i\right)}{\partial t} = \frac{\partial \vec{Q}_i}{\partial t} V_i + \frac{\partial V_i}{\partial t} \vec{Q}_i = -\sum_{j=k(i)} \left(\vec{F}_{conv} - \vec{F}_{vis}\right) \cdot S_{ij} + \vec{G}_i \cdot V_i \tag{3.13}$$

Equations (3.1) and (3.8) have the same form. Consequently, the spatial discretization of the turbulence model equation is similar to the discretization of the

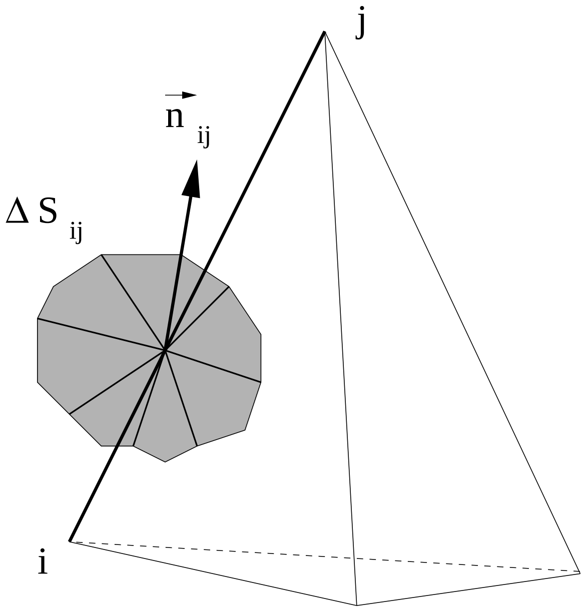


Fig. 10. Dual mesh face associated with the edge (i, j).

Navier-Stokes equations, shown in Eq. (3.13). The Navier-Stokes and the turbulence model equations were not solved simultaneously, but were staggered, with the turbulence model lagging one time step behind the Navier-Stokes equations. Therefore, the equations (3.1) and (3.8) were solved alternatively, with each equation using the variables from the other equation corresponding to the previous time step.

4. Vector fluxes implementation

The convective flux in Eq. (3.13) was calculated using Roe's flux-difference splitting scheme [25]. Roe's approximate Riemann solver is based on the decomposition of the flux difference over a face of the control volume into a sum of wave contributions.

The convective fluxes were evaluated at the faces of a control volume using the following expression [19]

$$\vec{F}_c = \frac{1}{2}\left[\vec{F}_c\left(\vec{Q}_R\right) + \vec{F}_c\left(\vec{Q}_L\right) - \left|A_{Roe}\right| \cdot \left(\vec{Q}_R - \vec{Q}_L\right)\right] \tag{3.14}$$

where $|A_{Roe}|$ is the Roe matrix. This matrix is identical to the flux Jacobian with respect to the conservative variables. The flow variables were replaced by the Roe-averaged variables, which are expressed as [19]

$$
\begin{aligned}
\tilde{\rho} &= \sqrt{\rho_L \cdot \rho_R} \\
\tilde{u} &= \frac{u_L\sqrt{\rho_L} + u_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\
\tilde{v} &= \frac{v_L\sqrt{\rho_L} + v_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\
\tilde{w} &= \frac{w_L\sqrt{\rho_L} + w_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\
\tilde{H} &= \frac{H_L\sqrt{\rho_L} + H_R\sqrt{\rho_R}}{\sqrt{\rho_L} + \sqrt{\rho_R}} \\
\tilde{c} &= \sqrt{(\gamma - 1) \cdot \left(\tilde{H} - \frac{\tilde{q}^2}{2}\right)} \\
\tilde{V} &= \tilde{u} \cdot n_x + \tilde{v} \cdot n_y + \tilde{w} \cdot n_z \\
\tilde{q}^2 &= \tilde{u}^2 + \tilde{v}^2 + \tilde{w}^2
\end{aligned}
\tag{3.15}
$$

The product of $|A_{Roe}|$ and the difference of the left and right states was evaluated as follows [19]

$$
|A_{Roe}| \cdot \left(\vec{Q_R} - \vec{Q_L}\right) = |\Delta\vec{F_1}| + |\Delta\vec{F}_{2,3,4}| + |\Delta\vec{F_5}| \tag{3.16}
$$

where

$$|\Delta\vec{F}_1| = |\vec{V} - \vec{c}| \cdot \left(\frac{\Delta p - \tilde{\rho}\cdot\tilde{c}\cdot\Delta V}{2\cdot\tilde{c}^2}\right)\cdot\begin{bmatrix} 1 \\ \tilde{u} - \tilde{c}\cdot n_x \\ \tilde{v} - \tilde{c}\cdot n_y \\ \tilde{w} - \tilde{c}\cdot n_z \\ \tilde{H} - \tilde{c}\cdot\tilde{V} \end{bmatrix} \tag{3.17}$$

$$|\Delta\vec{F}_{2,3,4}| = |\vec{V}| \cdot \left(\Delta\rho - \frac{\Delta p}{\tilde{c}^2}\right)\cdot\begin{bmatrix} 1 \\ \tilde{u} \\ \tilde{v} \\ \tilde{w} \\ \tilde{q}^2/2 \end{bmatrix} + \tag{3.18}$$

$$+|\vec{V}|\cdot\tilde{\rho}\cdot\begin{bmatrix} 0 \\ \Delta u - \Delta V\cdot n_x \\ \Delta v - \Delta V\cdot n_y \\ \Delta w - \Delta V\cdot n_z \\ \tilde{u}\cdot\Delta u + \tilde{v}\cdot\Delta v + \tilde{w}\cdot\Delta w - \tilde{V}\cdot\Delta V \end{bmatrix}$$

$$|\Delta\vec{F}_5| = |\vec{V} + \vec{c}| \cdot \left(\frac{\Delta p + \tilde{\rho}\cdot\tilde{c}\cdot\Delta V}{2\cdot\tilde{c}^2}\right)\cdot\begin{bmatrix} 1 \\ \tilde{u} + \tilde{c}\cdot n_x \\ \tilde{v} + \tilde{c}\cdot n_y \\ \tilde{w} + \tilde{c}\cdot n_z \\ \tilde{H} + \tilde{c}\cdot\tilde{V} \end{bmatrix} \tag{3.19}$$

To evaluate the viscous flux in Eq. (3.4), the velocity and temperature derivatives were required at edge midpoints. For this edge-based data structure, the gradients of a generic variable U at edge midpoints can be computed by averaging the nodal values

$$\overline{(\nabla U)}_{ij} = \frac{1}{2}\left[(\nabla U)_i + (\nabla U)_j\right] \tag{3.20}$$

The disadvantage of this approach is that decoupling occurs, particularly for hexahedra and prismatic cells. To prevent decoupling, the directional derivatives along an edge were taken into account [26]

$$(\nabla U)_{ij} = \overline{(\nabla U)}_{ij} - \left[\overline{(\nabla U)}_{ij} \cdot \frac{\Delta\vec{r}}{|\Delta\vec{r}|} - \frac{U_j - U_i}{|\Delta\vec{r}|}\right] \cdot \frac{\Delta\vec{r}}{|\Delta\vec{r}|} \tag{3.21}$$

where $\Delta\vec{r}$ is the vector that goes from node i to node j.

The same approach as outlined above was also employed for the discretization of the viscous term of Eq. (3.8).

The least-squares method was used to calculate solution gradients at vertices of the mesh [24]. The least-squares approach is based upon the use of a first-order Taylor series approximation for each edge which is incident to the central node i. The change of the solution of a generic variable U along an edge ij was computed from

$$(\nabla U_i) \cdot \vec{r}_{ij} = U_j - U_i \tag{3.22}$$

where $\vec{r}_{ij}$ is the vector that goes from node i to node j.

When Eq. (3.22) is applied to the N number of edges incident to node i, the following over-constrained system of linear equations is obtained

$$\begin{bmatrix} \Delta x_{i1} & \Delta y_{i1} & \Delta z_{i1} \\ \Delta x_{i2} & \Delta y_{i2} & \Delta z_{i2} \\ \vdots & \vdots & \vdots \\ \Delta x_{ij} & \Delta y_{ij} & \Delta z_{ij} \\ \vdots & \vdots & \vdots \\ \Delta x_{iN} & \Delta y_{iN} & \Delta z_{iN} \end{bmatrix} \begin{bmatrix} \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \\ \frac{\partial U}{\partial z} \end{bmatrix} = \begin{bmatrix} U_1 - U_i \\ U_2 - U_i \\ \vdots \\ U_j - U_i \\ \vdots \\ U_N - U_i \end{bmatrix} \tag{3.23}$$

Equation (3.23) can be expressed as

$$\bar{A} \cdot \vec{x} = \vec{b} \tag{3.24}$$

Solving for the gradient vector $\vec{x}$ requires the inversion of the matrix $\bar{A}$. To prevent problems with ill-conditioning, particularly on stretched grids, the matrix $\bar{A}$ can be decomposed into the product of an orthogonal matrix $\bar{Q}$ and an upper triangular matrix $\bar{R}$ [27].

The solution of the gradients was obtained using the following expression

$$\vec{x} = \bar{R}^{-1} \cdot \bar{Q}^T \cdot \vec{b} \tag{3.25}$$

The entries in the upper triangular matrix $\bar{R}$ were obtained from [27]

$$r_{11} = \sqrt{\sum_{j=1}^{N} \left(\Delta x_{ij}\right)^2}$$

$$r_{12} = \frac{1}{r_{11}} \cdot \sum_{j=1}^{N} \Delta x_{ij} \cdot \Delta y_{ij}$$

$$r_{22} = \sqrt{\sum_{j=1}^{N} \left(\Delta y_{ij}\right)^2 - r_{12}^2} \tag{3.26}$$

$$r_{13} = \frac{1}{r_{11}} \cdot \sum_{j=1}^{N} \Delta x_{ij} \cdot \Delta z_{ij}$$

$$r_{23} = \frac{1}{r_{22}} \cdot \left(\sum_{j=1}^{N} \Delta y_{ij} \cdot \Delta z_{ij} - \frac{r_{12}}{r_{11}} \cdot \sum_{j=1}^{N} \Delta x_{ij} \cdot \Delta z_{ij} \right)$$

$$r_{33} = \sqrt{\sum_{j=1}^{N} (\Delta z_{ij})^2 - (r_{12}^2 + r_{23}^2)}$$

The gradient at node i was calculated with the weighted sum of the edge differences [27]

$$\nabla U_i = \sum_{j=1}^{N} \vec{w}_{ij} \cdot (U_j - U_i) \tag{3.27}$$

with the vector of weights $\vec{w}_{ij}$ defined as

$$\vec{w}_{ij} = \begin{bmatrix} \alpha_{ij,1} - \frac{r_{12}}{r_{11}} \cdot \alpha_{ij,2} + \beta \cdot \alpha_{ij,3} \\ \alpha_{ij,2} - \frac{r_{23}}{r_{22}} \cdot \alpha_{ij,3} \\ \alpha_{ij,3} \end{bmatrix} \tag{3.28}$$

The terms in Eq. (3.28) were expressed as [27]

$$\alpha_{ij,1} = \frac{\Delta x_{ij}}{r_{11}^2}$$

$$\alpha_{ij,2} = \frac{1}{r_{22}^2} \cdot \left(\Delta y_{ij} - \frac{r_{12}}{r_{11}} \cdot \Delta x_{ij} \right) \tag{3.29}$$

$$\alpha_{ij,3} = \frac{1}{r_{33}^2} \cdot \left(\Delta z_{ij} - \frac{r_{23}}{r_{22}} \cdot \Delta y_{ij} + \beta \cdot \Delta x_{ij} \right)$$

$$\beta = \frac{r_{12} \cdot r_{23} - r_{13} \cdot r_{22}}{r_{11} \cdot r_{22}}$$

5. Second-order upwind scheme

The flow variables had a piecewise linear distribution over all cells. Therefore, the numerical method was second-order accurate in space [28]. The left and right fluid states were determined using linear reconstruction. To produce a monotone solution, a limiter was applied to the linearly reconstructed fluid states.

A piecewise linear redistribution of the cell-averaged flow variables is represented by [29]

$$Q\left(x, y, z\right) = Q\left(x_0, y_0, z_0\right) + \nabla Q_0 \cdot \vec{r}, \tag{3.30}$$

where $\vec{r}$ is the vector from point (x_0, y_0, z_0) to any point (x, y, z) in the cell, and ∇Q_0 represents the solution gradient in the cell. Using this piecewise linear redistribution in Eq. (3.30), the left and right fluid states Q_L and Q_R are

$$\begin{aligned} Q_L &= Q_i + \tfrac{1}{2}\nabla Q_i \cdot \Delta\vec{r} \\ Q_R &= Q_j + \tfrac{1}{2}\nabla Q_j \cdot \Delta\vec{r}, \end{aligned} \tag{3.31}$$

where $\Delta\vec{r} = \vec{r}_j - \vec{r}_i$, Q_i and Q_j are cell-averaged fluid states associated with cells V_i and V_j, respectively, and ∇Q_i and ∇Q_j are gradients of Q at the end nodes i and j of an edge (i, j), respectively. Figure 11 shows an example of reconstructed solution, where the left and right fluid states Q_L and Q_R were obtained from the constant cell values Q_i and Q_j augmentated by the gradient contributions ∇Q_i and ∇Q_j.

To ensure that the linearly reconstructed fluid states produced a monotone solution, a limiter function was introduced into Eq. (3.31) [30]

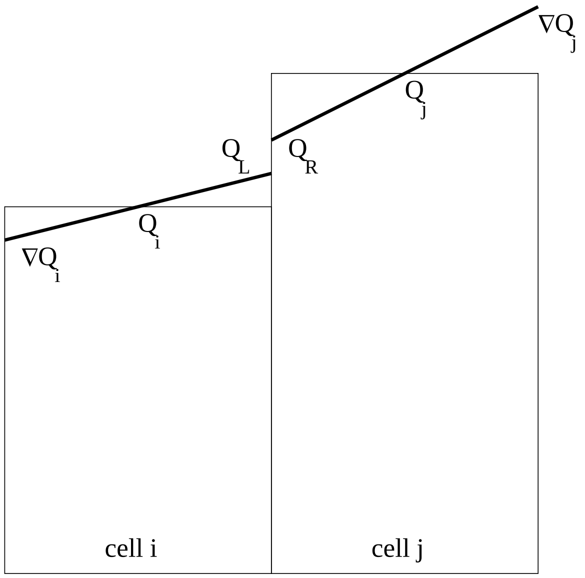


Fig. 11. Linear reconstruction of the solution.

$$\begin{aligned} Q_L &= Q_i + \tfrac{1}{2}\alpha\left[(1-\eta)\,\nabla Q_i \cdot \Delta\vec{r}, \eta\,(Q_j - Q_i)\right] \\ Q_R &= Q_j + \tfrac{1}{2}\alpha\left[(1-\eta)\,\nabla Q_j \cdot \Delta\vec{r}, \eta\,(Q_j - Q_i)\right], \end{aligned} \tag{3.32}$$

where $\eta = \frac{1}{3}$ and the limiter function was

$$\alpha\,[a,b] = \frac{\left[1 + sign\,(1,ab)\right] \cdot \left[\left(a^2+\epsilon\right)b + \left(b^2+\epsilon\right)a\right]}{2 \cdot \left(a^2+b^2+2\cdot\epsilon\right)} \tag{3.33}$$

ϵ is a very small number (in the order of machine zero) that prevents division by zero in smooth regions of the flow. The limiter reduces the solution accuracy in regions of large gradients, in order to avoid the generation of new extrema.

6. Time integration

The semi-discrete form Eq. (3.13) of the Navier-Stokes equations Eq. (3.1) and of the turbulence model equations Eq. (3.8) were expressed for a grid node i as

$$\frac{\partial Q_i}{\partial t} V_i = R_i \tag{3.34}$$

with

$$R_i = -\oint\limits_{S_i} \vec{F} \cdot \vec{n} \cdot dS + E_i \cdot V_i - \frac{\partial V_i}{\partial t} Q_i \tag{3.35}$$

where R_i is the residual.

The term $\frac{\partial V_i}{\partial t}$ is the time evolution of the cell volume and relates to the deformation of the grid. This term was calculated using the Geometric Conservation Law [31] which relates the time derivative of the volume to the motion of the faces of the cell

$$\frac{\partial V_i}{\partial t} = \oint\limits_{S_i} \vec{V}_S \cdot \vec{n} \cdot dS = \sum_{j=k(i)} \vec{V}_S \cdot \vec{n}_{ij} \cdot S_{ij}, \tag{3.36}$$

where $\vec{V}_S$ denotes the velocity vector corresponding to the center of the face S_{ij} whose unitary normal vector is $\vec{n}_{ij}$. The integral was solved using the same edge-based numerical scheme used to solve the inviscid fluxes.

To obtain the time evolution of the solution, Eq. (3.34) has to be integrated in time. The turbulence model equations were uncoupled from the Navier-Stokes equations. The Navier-Stokes equations were solved first, assuming the eddy viscosity μ_T equal to the value from the previous time step. Subsequently, the turbulence model equations were solved, k and ω were advanced in time, and μ_T was updated. Both sets of equations were integrated in time using the same explicit, four-stage Runge-Kutta scheme

$$\begin{aligned}
Q_i^{(0)} &= Q_i^n \\
Q_i^{(1)} &= Q_i^{(0)} + \alpha_1 \cdot \tfrac{\Delta t_i}{V_i} \cdot R\left(Q_i^{(0)}\right) \\
Q_i^{(2)} &= Q_i^{(0)} + \alpha_2 \cdot \tfrac{\Delta t_i}{V_i} \cdot R\left(Q_i^{(1)}\right) \\
Q_i^{(3)} &= Q_i^{(0)} + \alpha_3 \cdot \tfrac{\Delta t_i}{V_i} \cdot R\left(Q_i^{(2)}\right) \\
Q_i^{(4)} &= Q_i^{(0)} + \alpha_4 \cdot \tfrac{\Delta t_i}{V_i} \cdot R\left(Q_i^{(3)}\right) \\
Q_i^{n+1} &= Q_i^{(4)}
\end{aligned} \tag{3.37}$$

where Δt_i is the time step at node i, the superscript is the time-stepping level, and α_1, α_2, α_3 and α_4 are the stage coefficients. The values for these stage coefficients were $\alpha_1 = 0.1084$, $\alpha_2 = 0.2602$, $\alpha_3 = 0.5052$, and $\alpha_4 = 1.0$.

A global time step was used by all the cells to ensure the time accuracy of the solution. This global time step was the minimum time step size of all the cells. The time-step calculation was based strictly on stability considerations for each node [32], and has the following expression

$$\Delta t_i = CFL \cdot \frac{V_i}{\left(\Lambda_c^x + \Lambda_c^y + \Lambda_c^z\right)_i + 4 \cdot \left(\Lambda_v^x + \Lambda_v^y + \Lambda_v^z\right)_i} \tag{3.38}$$

where the CFL number is the Courant-Friedrichs-Levy stability condition for explicit time integration methods.

The convective spectral radii are expressed as

$$\Lambda_c^x = (|u| + c) \cdot \Delta S_x$$

$$\Lambda_c^y = (|v| + c) \cdot \Delta S_y \tag{3.39}$$

$$\Lambda_c^z = (|w| + c) \cdot \Delta S_z$$

The viscous spectral radii are expressed as

$$\Lambda_v^x = max\left(\frac{4}{3\cdot\rho}, \frac{\gamma}{\rho}\right)\cdot\left(\frac{\mu_L}{Pr_L}+\frac{\mu_T}{Pr_T}\right)\cdot\frac{(\Delta S^x)^2}{V_i}$$

$$\Lambda_v^y = max\left(\frac{4}{3\cdot\rho}, \frac{\gamma}{\rho}\right)\cdot\left(\frac{\mu_L}{Pr_L}+\frac{\mu_T}{Pr_T}\right)\cdot\frac{(\Delta S^y)^2}{V_i} \tag{3.40}$$

$$\Lambda_v^z = max\left(\frac{4}{3\cdot\rho}, \frac{\gamma}{\rho}\right)\cdot\left(\frac{\mu_L}{Pr_L}+\frac{\mu_T}{Pr_T}\right)\cdot\frac{(\Delta S^z)^2}{V_i}$$

The variables ΔA^x, ΔA^y and ΔA^z represent the projections of the volume V_i on the y-z-, x-z, and x-y planes. They are defined as

$$\Delta A^x = \frac{1}{2}\cdot\sum_{j=1}^{N}|n_x\cdot\Delta A_j|$$

$$\Delta A^y = \frac{1}{2}\cdot\sum_{j=1}^{N}|n_y\cdot\Delta A_j| \tag{3.41}$$

$$\Delta A^z = \frac{1}{2}\cdot\sum_{j=1}^{N}|n_z\cdot\Delta A_j|$$

7. Boundary conditions

Boundary conditions were applied as conditions on the flux at boundary surfaces as opposed to being applied directly to state variables. This approach, called “weak condition” [33], avoids singularities at mesh points located at corners. To close the dual-meshes at boundary points a face-based, as opposed to an edge-based, data structure is utilized. The boundary face-based data structure and the adjacent edge-based data structure were used to evaluate the fluxes and gradients of the flow field. The boundary surface elements were discretized in the same manner as the elements of interior grid. The vector normal to the boundary face was found in the same manner as the normal to a dual mesh face in the interior of the grid.

The tangency flow condition was imposed by requiring that the flux through the wall be zero, so that $\vec{u} \cdot \vec{n} = 0$, where $\vec{n}$ is the normal to the boundary dual mesh. The flux normal to the boundary face is [33]

$$F_{B_i} = \begin{pmatrix} 0 \\ p_i \cdot \hat{n}_{B_i} \cdot \hat{i} \\ p_i \cdot \hat{n}_{B_i} \cdot \hat{j} \\ p_i \cdot \hat{n}_{B_i} \cdot \hat{k} \\ 0 \end{pmatrix}. \qquad (3.42)$$

The value of the pressure in Eq. (3.42) was obtained at the quadrature points. Figure 12 shows a boundary cell with the interior and boundary faces. The quadrature points appear marked with an X in the plot.

The expression of the normal flux is valid for both viscous and inviscid flow. In the viscous flow case, the no-slip boundary condition was applied after each time step, such that $\vec{u} = \vec{V}_{wall}$ for moving walls and $\vec{u} = 0$ for stationary walls.

The inflow/outflow boundary flux was defined in terms of the fluid state at the boundary node i and was specified as [33]

$$F_{B_i} = \begin{cases} -\hat{F}(Q_{-\infty}) \cdot \hat{n}_{B_i} & \text{if} \quad \hat{U}_i \cdot \hat{n}_{B_i} < -c_i \\ \hat{F}(Q_i, Q_{-\infty}) \cdot \hat{n}_{B_i} & \text{if} \quad \hat{U}_i \cdot \hat{n}_{B_i} \in [-c_i, c_i] \\ \hat{F}(Q_i) \cdot \hat{n}_{B_i} & \text{if} \quad \hat{U}_i \cdot \hat{n}_{B_i} > c_i. \end{cases} \qquad (3.43)$$

where $Q_{-\infty}$ is the freestream value of the state vector, Q_i is the boundary cell state vector, and c_i is the local speed of sound.

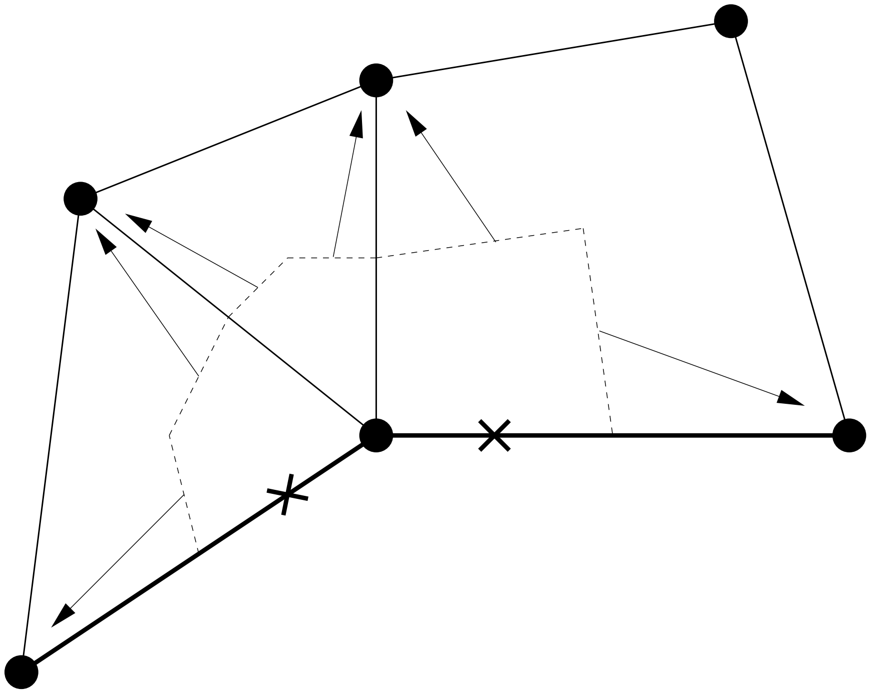


Fig. 12. Quadrature points to calculate boundary fluxes.

The flux normal to the boundary face imposed for supersonic inflow (*i.e.*, $\hat{U}_i \cdot \hat{n}_{B_i} < -c_i$) was computed using the state vector at upstream infinity. The flux normal to the boundary face imposed for subsonic inflow and outflow (*i.e.*, $\hat{U}_i \cdot \hat{n}_{B_i} \in [-c_i, c_i]$) was computed using an intermediate state calculated from the conditions at cell i and upstream infinity. The intermediate state was calculated using Riemann invariants. The turbulent flow variables, k and ω, were specified at the wall and inlet boundaries, and extrapolated at the outlet boundary. The value of k was set to zero at walls, since this region of the flow corresponds to the laminar sublayer which is characterized for having laminar flow behavior.

C. Mesh deformation algorithm

This section presents the details of the mesh deformation algorithm that was developed to deform the mesh as the wing evolved in time. The section presents the translational and rotational deformations, the details of the computation of the rotation angles, the cubic mapping function used to deform the mesh, and a parametric study of mesh deformation algorithm to determine the evolution of the mesh quality as a function of the mesh deformation.

1. Translational deformations

The computational mesh was deformed to allow for the wing displacement. The mesh connectivity was not modified in this deformation process. The mesh deformation algorithm was applied in two steps: first a translation, and then a rotation about the three axes. The O-grid layers were translated, without being deformed, since the currently used structural model does not account for cross-sectional deformation. The unstructured grid was deformed in the y- and z-directions using a spring analogy technique [34]. The nodes of each layer were also translated in the x-direction according to the spanwise wing deformation. The length of the wing was assumed constant, using the constraint presented in Eq. (2.18). The magnitude of the translational deformations were set by the new coordinates of the deformed elastic axis $(X_{e.a.}, Y_{e.a.}, Z_{e.a.})$, which were calculated by the structural solver.

Figure 13 shows the mesh after the translational deformation. The original undeformed mesh is shown in grey color, and the deformed mesh is shown in red. In this case the tip deformation along the y-axis is 20% of the wing semi-span.

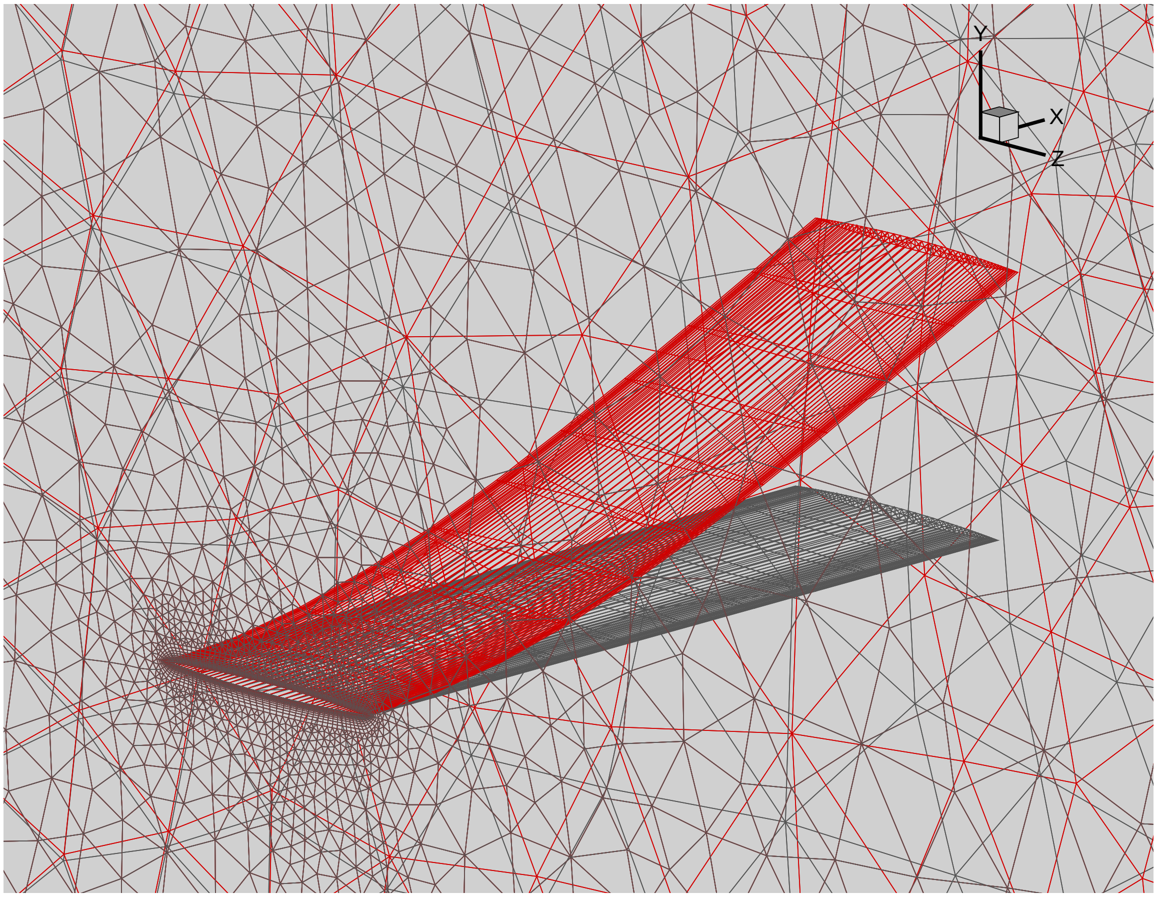


Fig. 13. Mesh before and after the translational deformations.

2. Rotational deformations

a. Rotation about the x-axis

The first rotation applied to the mesh slice was a rotation about the x-axis, which is analogous to a torsional deformation. This rotation about the x-axis took place about the elastic axis. Figure 14 shows a sketch and a picture of the mesh after rotations about the x-axis. The sketch on the left shows the initial and final positions of the leading and trailing edges. The picture on the right shows the layer corresponding to the tip of the wing. Note that even after large angular deformations no negative volumes are created and the triangles still have a high quality associated.

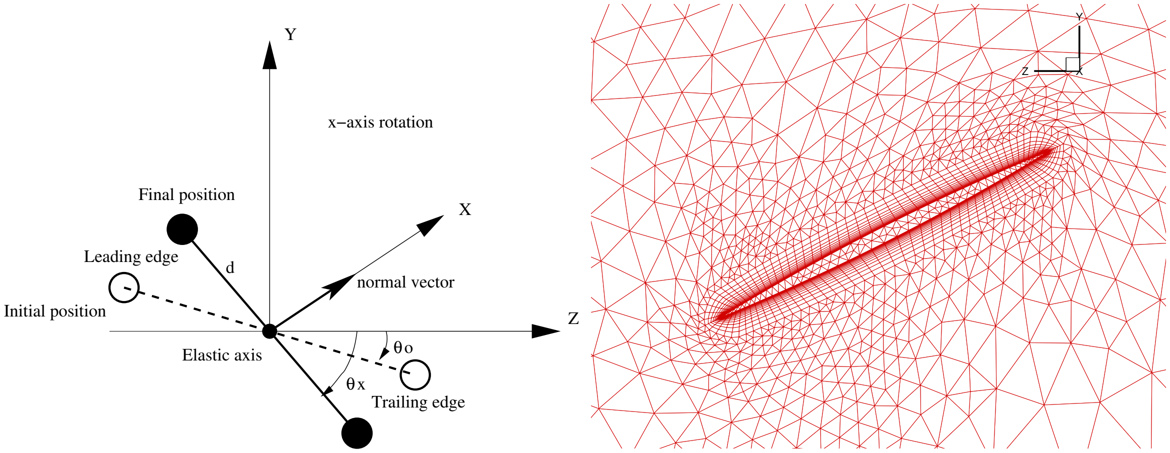


Fig. 14. Rotations about the x-axis.

The mesh slice was contained in the y-z plane. The mesh slice was assumed to already have an initial rotation about the x-axis. Since the mesh slice is contained initially on the y-z plane, the vector normal to the mesh slice is oriented along the x-axis. The parameter d is the distance from the leading edge to the elastic axis.

The initial leading edge vector and normal vector have the following components:

$$\left\{ \begin{matrix} X \\ Y \\ Z \end{matrix} \right\}_{L.E.}^{(0)} = \left\{ \begin{matrix} 0 \\ d \cdot sin\left(\theta_0\right) \\ -d \cdot cos\left(\theta_0\right) \end{matrix} \right\} \tag{3.44}$$

$$\left\{ \begin{matrix} n_x \\ n_y \\ n_z \end{matrix} \right\}^{(0)} = \left\{ \begin{matrix} 1 \\ 0 \\ 0 \end{matrix} \right\} \tag{3.45}$$

The final position of the leading edge after a rotation along the x-axis was obtained by rotating the mesh slice the amount $(\theta_x - \theta_0)$ about the x-axis. The coordinates of the leading edge were:

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}^{(1)}_{L.E.} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\left(\theta_x - \theta_0\right) & -sin\left(\theta_x - \theta_0\right) \\ 0 & sin\left(\theta_x - \theta_0\right) & cos\left(\theta_x - \theta_0\right) \end{bmatrix} \cdot \begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}^{(0)}_{L.E.}$$

which yields

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}^{(1)}_{L.E.} = \begin{Bmatrix} 0 \\ d \cdot sin\left(\theta_x\right) \\ -d \cdot cos\left(\theta_x\right) \end{Bmatrix} \tag{3.46}$$

The rotation with respect to the x-axis does not change the normal vector:

$$\begin{Bmatrix} n_x \\ n_y \\ n_z \end{Bmatrix}^{(1)} = \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix} \tag{3.47}$$

Thus the normal vector still points in the x-axis and the wing slice is still normal to the x-y plane. Figure 15 shows the mesh before and after the x-axis rotational deformation. The figure on the left shows a 3D view of the mesh with only translational deformation. The figure on the right shows the mesh after the x-axis rotational deformation.

b. Rotation about the y-axis

The second rotation applied to the mesh slice was a rotation about the y-axis, which is analogous to an in plane bending deformation. This rotation about the y-axis took place about the elastic axis. Figure 16 shows a sketch and a picture of the mesh after rotations about the y-axis. The sketch on the left shows the initial and final

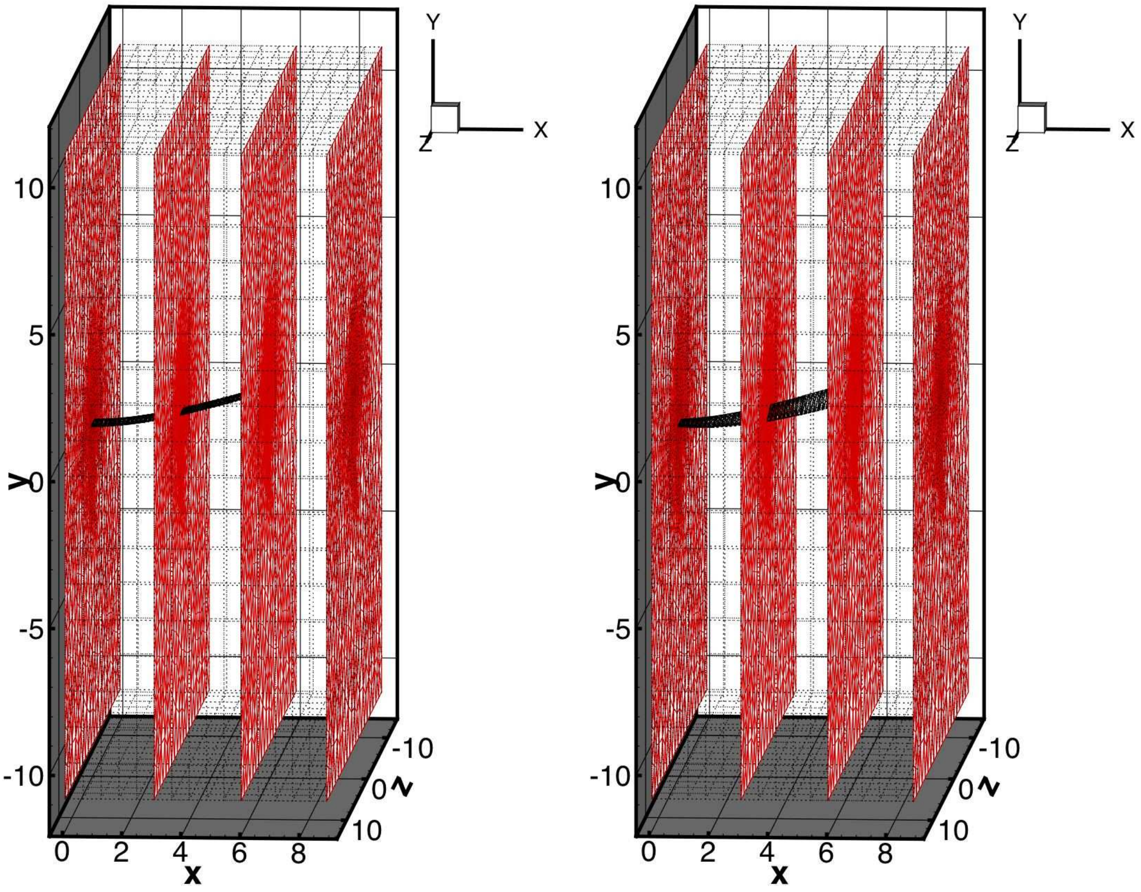


Fig. 15. Mesh before and after the x-axis rotational deformation.

positions of the leading and trailing edges. The picture on the right shows a top view of the mesh after the rotations about the y-axis. The amplitudes of these rotations are small because the wing has a high stiffness associated with the in-plane motion. Nevertheless, it is possible to see the wing in-plane deformation as well as the rotation of all the layers about the y-axis.

The final position of the leading edge after a rotation along the y-axis was obtained by rotating the mesh slice the amount θ_y about the y-axis. The coordinates of the leading edge were:

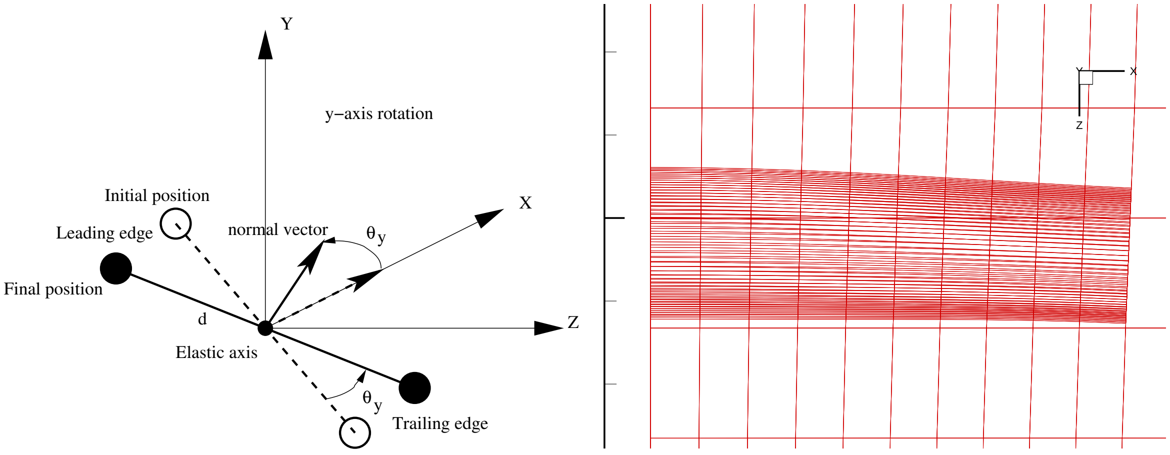


Fig. 16. Rotations about the y-axis.

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}^{(2)}_{L.E.} = \begin{bmatrix} cos\left(\theta_y\right) & 0 & sin\left(\theta_y\right) \\ 0 & 1 & 0 \\ -sin\left(\theta_y\right) & 0 & cos\left(\theta_y\right) \end{bmatrix} \cdot \begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}^{(1)}_{L.E.}$$

which yields

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}^{(2)}_{L.E.} = \begin{Bmatrix} -d \cdot cos\left(\theta_x\right) \cdot sin\left(\theta_y\right) \\ d \cdot sin\left(\theta_x\right) \\ -d \cdot cos\left(\theta_x\right) \cdot cos\left(\theta_y\right) \end{Bmatrix} \tag{3.48}$$

The expression for the normal vector after a rotation along the y-axis was

$$\begin{Bmatrix} n_x \\ n_y \\ n_z \end{Bmatrix}^{(2)} = \begin{Bmatrix} cos\left(\theta_y\right) \\ 0 \\ -sin\left(\theta_y\right) \end{Bmatrix} \tag{3.49}$$

Thus the normal vector is now contained on the x-z plane.

c. Rotation about the z-axis

The third rotation applied to the mesh slice was a rotation about the z-axis, which is analogous to an out-of-plane bending deformation. This rotation about the z-axis took place about the elastic axis. Figure 17 shows a sketch and a picture of the mesh after rotations about the z-axis. The sketch on the left shows the initial and final positions of the leading and trailing edges. The picture on the right shows a front view of the mesh after the rotations about the z-axis. It is clear from this figure how the layers rotate following the out-of-plane deformations of the wing.

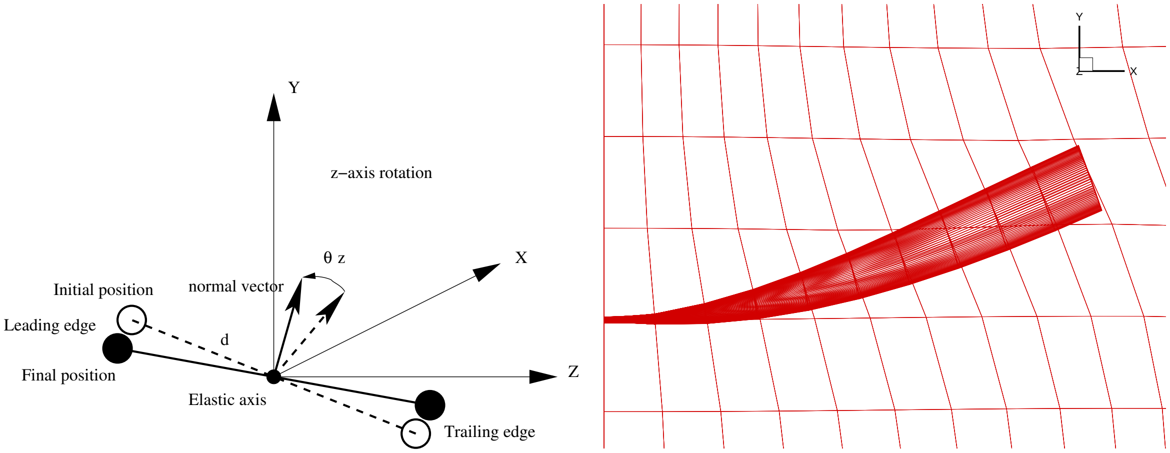


Fig. 17. Rotations about the z-axis.

The final position of the leading edge after a rotation along the z-axis was obtained by rotating the mesh slice the amount θ_z about the z-axis. The coordinates of the leading edge were:

$$\left\{\begin{array}{c} X \\ Y \\ Z \end{array}\right\}_{L.E.}^{(3)} = \left[\begin{array}{ccc} cos\left(\theta_z\right) & -sin\left(\theta_z\right) & 0 \\ sin\left(\theta_z\right) & cos\left(\theta_z\right) & 0 \\ 0 & 0 & 1 \end{array}\right] \cdot \left\{\begin{array}{c} X \\ Y \\ Z \end{array}\right\}_{L.E.}^{(2)}$$

which yields

$$\left\{\begin{array}{c} X \\ Y \\ Z \end{array}\right\}_{L.E.}^{(3)} = \left\{\begin{array}{c} -d \cdot cos\left(\theta_x\right) \cdot sin\left(\theta_y\right) \cdot cos\left(\theta_z\right) - d \cdot sin\left(\theta_x\right) \cdot sin\left(\theta_z\right) \\ -d \cdot cos\left(\theta_x\right) \cdot sin\left(\theta_y\right) \cdot sin\left(\theta_z\right) + d \cdot sin\left(\theta_x\right) \cdot cos\left(\theta_z\right) \\ -d \cdot cos\left(\theta_x\right) \cdot cos\left(\theta_y\right) \end{array}\right\} \tag{3.50}$$

The expression for the normal vector after a rotation along the z-axis was

$$\left\{\begin{array}{c} n_x \\ n_y \\ n_z \end{array}\right\}^{(3)} = \left\{\begin{array}{c} cos\left(\theta_y\right) \cdot cos\left(\theta_z\right) \\ cos\left(\theta_y\right) \cdot sin\left(\theta_z\right) \\ -sin\left(\theta_y\right) \end{array}\right\} \tag{3.51}$$

3. Computation of the rotation angles

This section describes how to calculate the rotation angles θ_x, θ_y and θ_z using the equations (3.46 - 3.51). The data provided by the structural solver comprised the coordinates of the elastic axis $(X_{E.A.}, Y_{E.A.}, Z_{E.A.})$, the coordinates of the leading edge $(X_{L.E.}, Y_{L.E.}, Z_{L.E.})$, and the components of the vector tangent to the elastic axis and normal to the wing slice (n_x, n_y, n_z).

From the last row of Eq. (3.51):

$$n_z = -sin\left(\theta_y\right)$$

therefore the θ_y rotation angle was calculated as

$$\theta_y = asin\left(-n_z\right) \tag{3.52}$$

Next from the second row of Eq. (3.51):

$$n_y = cos\left(\theta_y\right) \cdot sin\left(\theta_z\right)$$

therefore the θ_z rotation angle was calculated as

$$\theta_z = asin\left[\frac{n_y}{cos\left(\theta_y\right)}\right] \tag{3.53}$$

Finally from the last row of Eq. (3.50):

$$Z_{L.E.} - Z_{E.A.} = -d \cdot cos\left(\theta_x\right) \cdot cos\left(\theta_y\right) \tag{3.54}$$

Taking the second row of Eq. (3.50):

$$Y_{L.E.} - Y_{E.A.} = -d \cdot cos\left(\theta_x\right) \cdot sin\left(\theta_y\right) \cdot sin\left(\theta_z\right) + d \cdot sin\left(\theta_x\right) \cdot cos\left(\theta_z\right) \tag{3.55}$$

This last equation is nonlinear in θ_x and could be solved using the Newton's method for nonlinear equations. However, rearranging Eq. (3.54) yields:

$$-d \cdot cos\left(\theta_x\right) = \frac{Z_{L.E.} - Z_{E.A.}}{cos\left(\theta_y\right)} \tag{3.56}$$

This equation cannot be used to solve for θ_x since the $cos()$ function cannot distinguish an angle from its opposite ($cos(x) = cos(-x)$). However, Eq. (3.56) can be replaced in Eq. (3.55) to obtain a linear equation in terms of θ_x:

$$Y_{L.E.} - Y_{E.A.} = \frac{Z_{L.E.} - Z_{E.A.}}{cos\left(\theta_y\right)} \cdot sin\left(\theta_y\right) \cdot sin\left(\theta_z\right) + d \cdot sin\left(\theta_x\right) \cdot cos\left(\theta_z\right) \tag{3.57}$$

therefore the θ_x rotation angle was calculated as

$$\theta_x = asin\left[\frac{Y_{L.E.} - Y_{E.A.}}{d \cdot cos\left(\theta_z\right)} - \frac{Z_{L.E.} - Z_{E.A.}}{d} \cdot tan\left(\theta_y\right) \cdot tan\left(\theta_z\right)\right] \tag{3.58}$$

Using Eq. (3.58), (3.52), and (3.53) the rotation angles θ_x, θ_y, and θ_z, were determined from the parameters provided by the structural solver.

4. Cubic mapping function

This section presents the details of the cubic mapping function that was used to deform the layers of the mesh.

When the mesh was rotated about the y- and z-axes, the nodes between the leading edge and trailing edge were rotated as a solid line. The nodes between the leading edge (or trailing edge) and the outer computational domain were deformed in the x-axis direction such that the layer was as close to being tangential as possible both at the boundary and at the wing. This was achieved using a cubic mapping function.

Figure 18 shows a sketch of the cubic mapping function and a picture with the front view of the mesh after the rotational deformations. The sketch for the mapping function shows how the nodes between the wing and the outer computational domain were deformed in the x-axis direction.

As shown in Fig. 18, the parameters needed by the cubic mapping function are the wing coordinates x_w, y_w, the boundary coordinates x_b, y_b, and the angles for the wing α_w and for the boundary α_b.

The cubic mapping function was expressed as

$$X(y) = A + B \cdot y + C \cdot y^2 + D \cdot y^3 \tag{3.59}$$

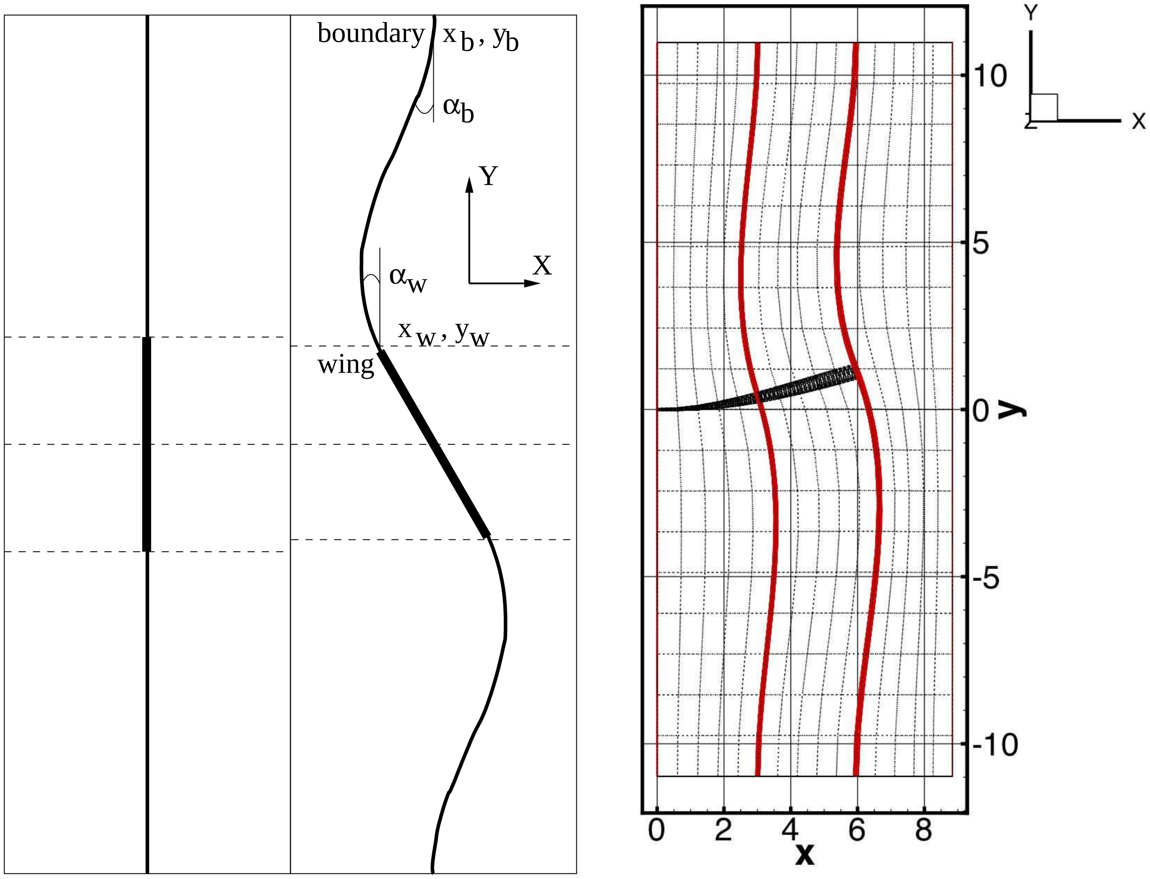


Fig. 18. Cubic mapping function and front view of rotational deformations.

Equation (3.59) has four unknown coefficients, which were obtained imposing four boundary conditions. These boundary conditions were:

1. The value of the cubic mapping function at the wing must be equal to the wing coordinate x_w

$$X(y_w) = A + B \cdot y_w + C \cdot y_w^2 + D \cdot y_w^3 = x_w$$

2. The value of the cubic mapping function at the boundary must be equal to the

boundary coordinate x_b

$$X(y_b) = A + B \cdot y_b + C \cdot y_b^2 + D \cdot y_b^3 = x_b$$

3. The value of the slope of the cubic mapping function at the wing must be equal to the wing slope α_w

$$\frac{\partial X}{\partial y}(y_w) = B + 2 \cdot C \cdot y_w + 3 \cdot D \cdot y_w^2 = \alpha_w$$

4. The value of the slope of the cubic mapping function at the boundary must be equal to the boundary slope α_b

$$\frac{\partial X}{\partial y}(y_b) = B + 2 \cdot C \cdot y_b + 3 \cdot D \cdot y_b^2 = \alpha_b$$

Solving this system of equations yields the values of the cubic mapping coefficients A, B, C, and D. The final expression for the cubic mapping function was:

$$X(y) = A + B \cdot y + C \cdot y^2 + D \cdot y^3$$

$$R_{\alpha y} = \frac{tan\left(\alpha_b\right) - tan\left(\alpha_w\right)}{y_b - y_w}$$

$$D = \frac{2}{\left(y_b - y_w\right)^2} \cdot \left[\frac{tan\left(\alpha_b\right) + tan\left(\alpha_w\right)}{2} - \frac{x_b - x_w}{y_b - y_w}\right]$$

$$C = \frac{R_{\alpha y}}{2} - \frac{3}{2} \cdot D \cdot \left(y_b + y_w\right)$$

$$B = tan\left(\alpha_b\right) - R_{\alpha y} \cdot y_b + 3 \cdot D \cdot y_w \cdot y_b$$

$$A = x_b - B \cdot y_b - C \cdot y_b^2 - D \cdot y_b^3$$

Figure 19 shows the 3D view of the mesh after the rotational deformations. Only four layers are shown, those corresponding to the wing root, the wing midspan, the wing tip, and the lateral boundary of the domain. Note how the layers deform tridimensionally to be as close to perpendicular as possible to both the deforming surface of the wing and the boundaries of the domain.

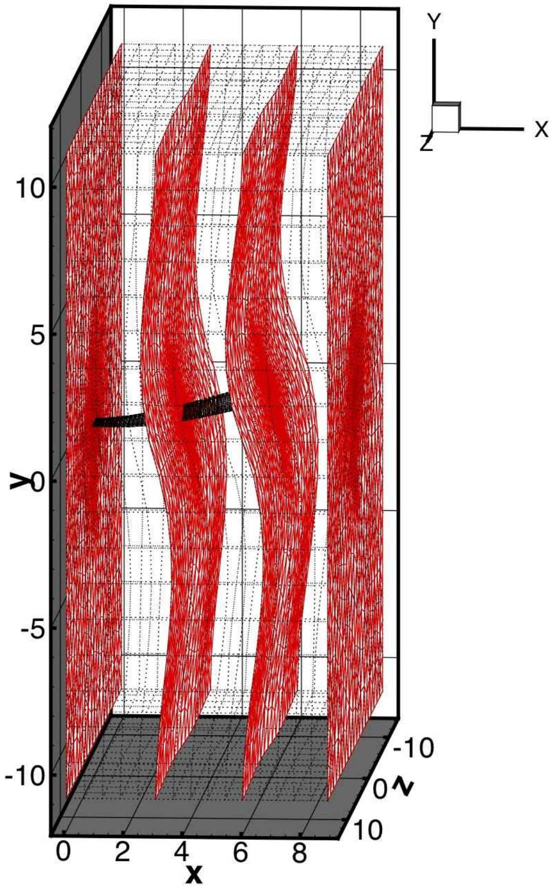


Fig. 19. 3D view of rotational deformations.

5. Parametric study of mesh deformation algorithm

This section presents the results for the parametric study of the mesh deformation algorithm. Two studies were performed, the first was related to the evolution of the mesh quality as a function of the mesh deformation. The second study was to test the limits of the mesh deformation algorithm.

a. Goland wing

The first study performed was to check the quality of the mesh as the wing was deformed. Cells were stretched or compressed as the mesh followed the deformations of the wing. Two measures were selected to evaluate the quality of the mesh, based on the areas and on the angles. The second study performed was to test the limits of the mesh deforming algorithm. Very large deformations were tested, with a wing tip displacement of up to 60% of the wing semi-span.

The area quality measure was defined as the ratio of the area of the triangular face to the sum of the squares of the length of each edge [22]:

$$Q_{area} = C \cdot \frac{A}{L_{12}^2 + L_{23}^2 + L_{31}^2} \tag{3.60}$$

where A is the area of the triangular face and L_{12}, L_{23}, and L_{31}, are the lengths of the three edges that form the triangle. C is a constant that is set equal to $2 \cdot \sqrt{3}$ so the formula returns a value of unity for equilateral triangles.

The angle quality measure was defined as the absolute value of the difference between each internal angle and 60°. This measure was normalized by an angle of 120°:

$$Q_{angle} = 1 - \frac{abs\left(\alpha - 60^\circ\right)}{120^\circ} \tag{3.61}$$

where α is each of the three internal angles of the triangular face. This formula has a value of unity for equilateral triangles, which have internal angles equal to 60°.

The area and angle quality measures were monitored as the wing deformed. Average and lowest values of the quality measures were calculated. The lowest value was obtained by comparing the quality of each cell and sorting the lowest value. The average value was obtained by adding the quality measure of all the cells and dividing by the number of cells.

Figure 20 shows the variation of the quality measures as the wing deformed. Four quality measures are plotted, average and lowest, for the area and angle parameters. The quality measures were calculated for wing tip deformations of up to 60% of the wing semi-span. The range for the average quality measure is between 92% and 96%. The range for the lowest quality measure is between 58% and 73%. Note in Fig. 20 that the average quality varies less than 4% even for very large wing tip deformations of 60% of the wing semi-span. The lowest area quality is more affected than the angle quality as the mesh deforms.

Figure 21 shows the front and 3D views of the deformed mesh, for a tip deformation equal to 60% of the wing semi-span, illustrating that the mesh deformation algorithm is robust and can be applied to very large deformations. The red mesh is the deformed mesh, the gray mesh is the undeformed mesh and is shown for comparison purposes.

b. F-5 wing

The grid generation and deformation algorithm was also used to model the F-5 wing [35]. This wing has a span of 0.6476 m, a root chord of 0.6396 m, and a tip chord of 0.18 m. The cross-section of the wing has a modified NACA 65-A-004.8 airfoil. Figure 22 shows the undeformed and deformed wing configurations. The deformed

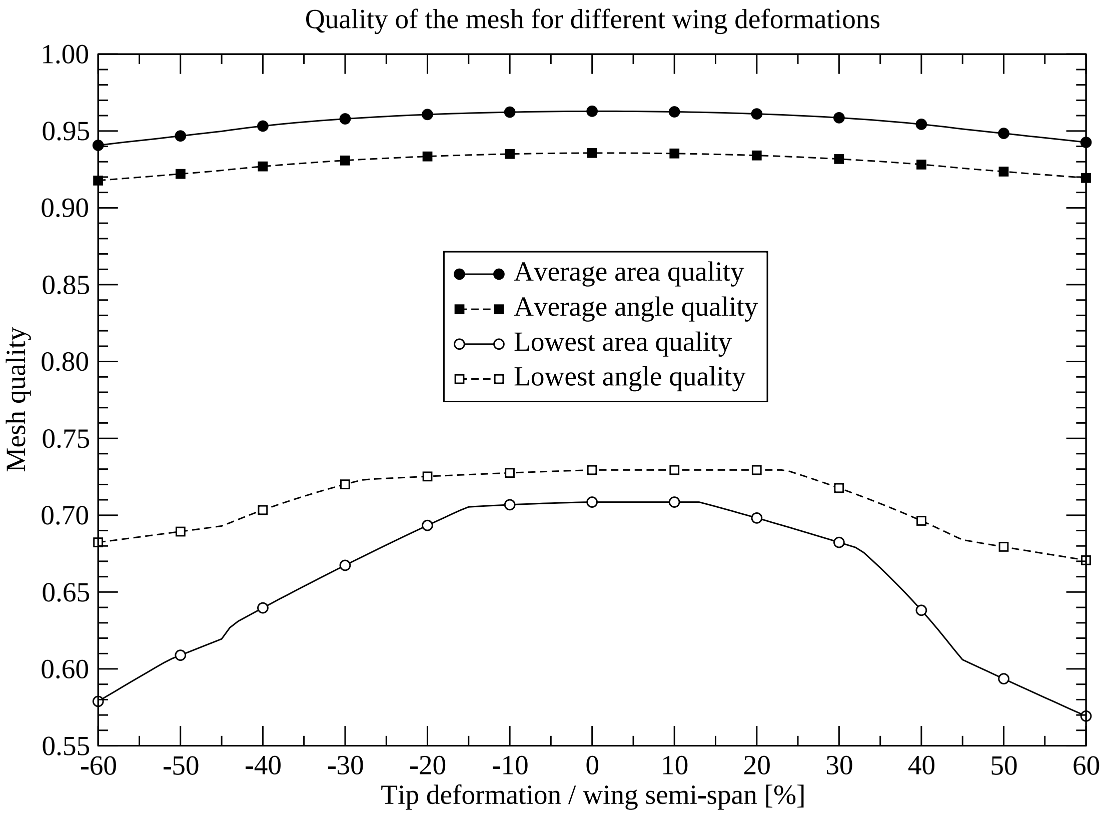


Fig. 20. Goland wing, quality measure as the wing deforms.

configuration has a wing tip displacement of 20% of the wing semi-span.

D. Parallel algorithm

The flow solver was parallelized to reduce the computational time. The message-passing interface (MPI) standard libraries were used for the interprocessor communication. As described in the grid generation section, the computational domain was divided into topologically identical layers that spanned from the root to past the tip of the wing. Having topologically identical layers reduced the communication effort

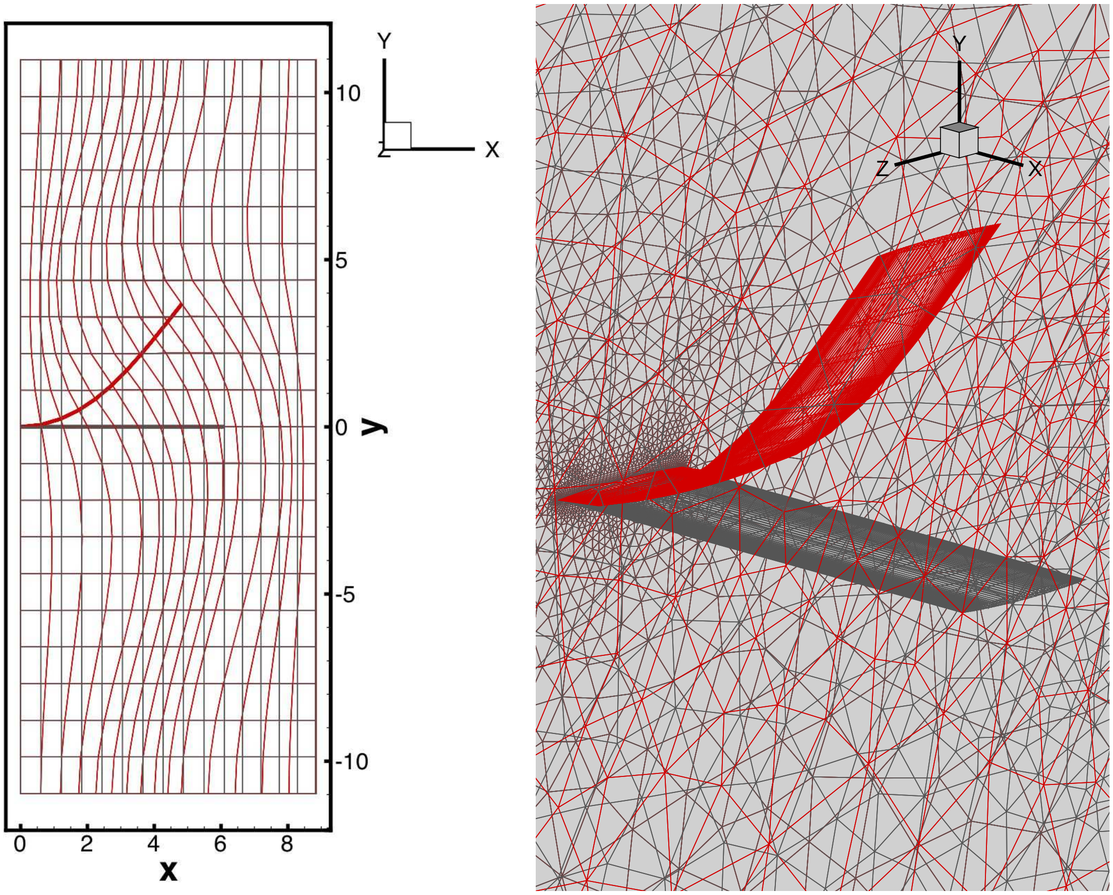


Fig. 21. Front and 3D view of Goland wing with tip deformation equal to 60% of wing semi-span.

and increased the parallelization efficiency. The grid deformation algorithm was designed in such a way that the connectivity of the grid did not change during wing deformation.

The grid was divided along the wing span into several sub-grids, and one processor was allocated for each sub-grid. Each sub-grid could include one or more layers, depending on the number of processors available. Figure 23 shows the sub-grids and the communication process. The nodes in the sub-grids were divided in two cate-

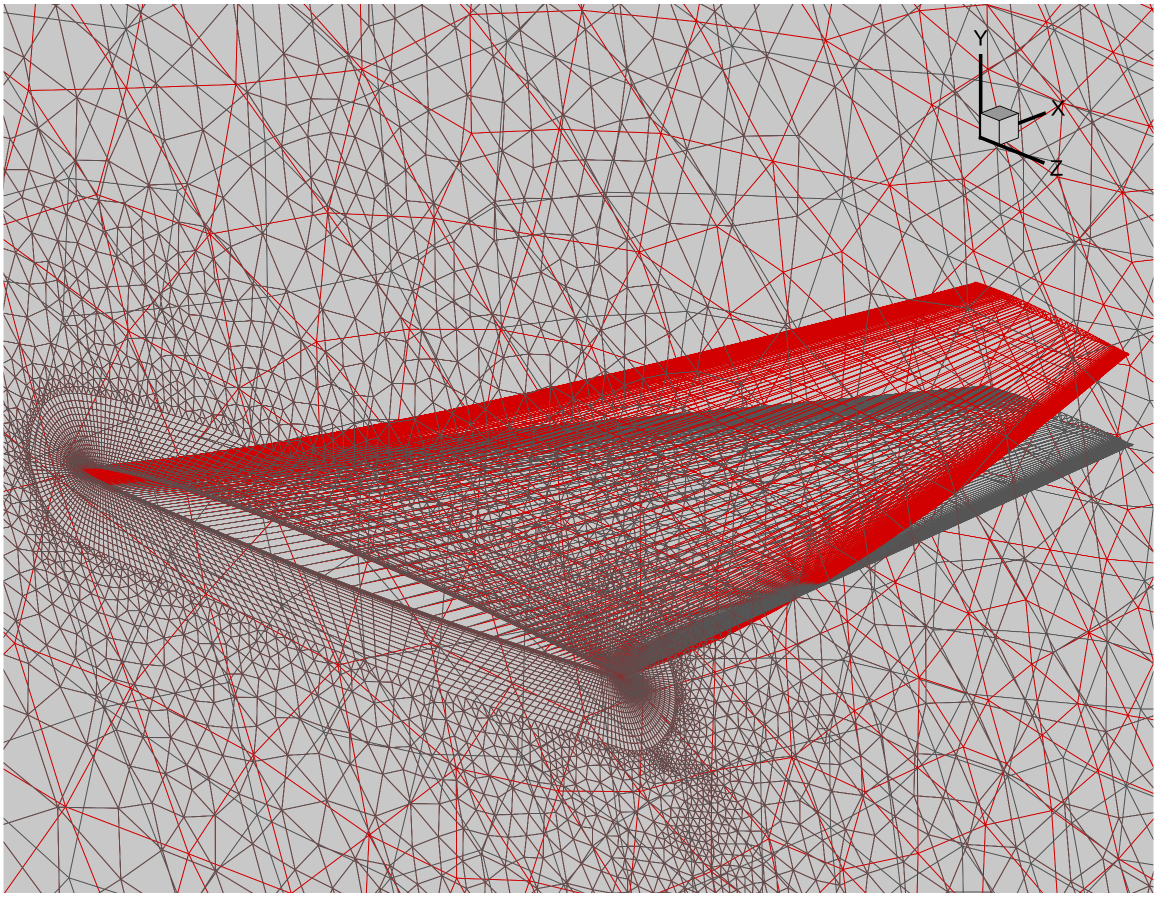


Fig. 22. F-5 wing: Deformed and undeformed meshes.

gories: active and ghost nodes. The active nodes, marked as black circles in Fig. 23, were the nodes where the numerical computation took place. The ghost nodes, marked as white circles in Fig. 23, were only used to compute the edge-based fluxes and the solution gradients. Each sub-grid consisted of n layers of active nodes (layers 1 to n) and two layers of ghost nodes, layers 0 and $n+1$. As shown in Fig. 23, the first two layers (0 and 1) of a sub-grid (i) coincided with the last two layers (n and $n+1$) of the previous sub-grid ($i-1$).

The state vector at the ghost nodes was updated at each time step by the neighbor processor. The state vector information traveled from active nodes to ghost

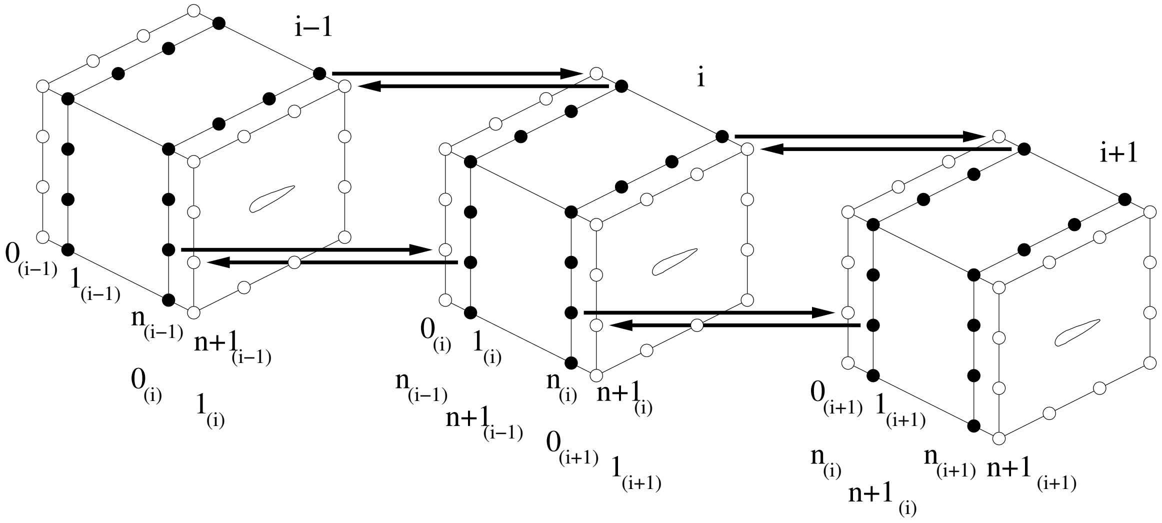


Fig. 23. Schematic view of the parallel implementation. Subindex indicate the sub-grid number. Active nodes are marked as black circles. Ghost nodes are marked as white circles. Arrows indicate the communication path.

nodes, as indicated by the arrows in Fig. 23. Only the active nodes of layers 1 and n were used in the communication process. Each active node from the layers 1 and n communicated with only one ghost node at each time step, as opposed to communicating with multiple ghost nodes. This was possible because the grid was designed to be topologically identical, and each node from a layer was connected to only one node from a neighbor layer. The fact that each active node from the layers 1 and n communicated with only one ghost node increased the parallelization efficiency, since it reduced the communication effort. The processor load was balanced to achieve the maximum possible parallel speedup. The load balance was achieved by evenly splitting the total number of layers of the whole grid among the available number of processors.

Table I summarizes the efficiency of the parallel computation. These results were obtained for the Heavy Goland Wing at Mach=0.09. The computational grid had 64 slices and each slice had 2316 nodes. The runs up to 16 processors were done on a

Table I. Efficiency of parallel computation.

Number of processors	Wall time [sec]	Efficiency [%]	Computer
1	18495	100.00	SGI Altix 3700
2	8963	103.17	SGI Altix 3700
4	4585	100.85	SGI Altix 3700
8	2356	98.13	SGI Altix 3700
16	1244	92.92	SGI Altix 3700
1	9026	100.00	Cray XT3
16	623	90.55	Cray XT3
32	380	74.23	Cray XT3

128-processor (1.3 GHz Intel Itanium) SGI Altix 3700 computer at the Texas A&M Supercomputing Center. Table I shows a small super-linear speed-up for 2 and 4 processors. The efficiency reduced once the number of processors increased to 8 and higher. In addition, 16 and 32 processor runs were done on a 2068-processor (2.4-GHz AMD Opteron) Cray XT3 at the Pittsburgh Supercomputing Center.

E. Multigrid

Parallel algorithms reduce the computational time by dividing the computing effort into several processors. The number of operations slightly increases due to the processor communication, but the overall effect of parallel computation is a reduction in the computational time. To further reduce the computational time, a multigrid technique was applied to the flow solver algorithm. The multigrid technique solves the governing equations on a series of successively coarser grids, accelerating the convergence of the solution on the finer grid while maintaining the overall accuracy of the flow solver. Therefore, multigrid methods achieve a reduction in the computational time by means of reducing the number of iterations necessary to obtain a converged solution.

This section presents the details of the multigrid implementation in the flow solver. The section starts with a general introduction to the multigrid methodology. The section continues with the mesh subdivision algorithm used to generate the different meshes. Then, the interpolation transfer operators are presented, which carry the information between the different meshes. Following this, the step by step implementation of the multigrid on the flow solver is described. Finally, a comparison of the performance of the multigrid solver versus the one-level grid solver is presented.

1. Introduction

The multigrid technique is based on the solution of the governing equations on a series of successively coarser grids that reduce the high-frequency components of the solution error and accelerate the convergence of the solution on the finer grid. The solution on the coarser grid is driven by the residuals of the finer grid, therefore maintaining the overall accuracy of the finer grid. The correction obtained for the coarse grid is then interpolated to the finer grid using an interpolation scheme [36].

A simple example was included herein to show the basic idea behind the multigrid technique [37]. The example consisted of solving iteratively a homogeneous, non-singular linear system $A \cdot u = 0$ with the use of different arbitrary initial solutions. $u = 0$ is the exact solution which is known, v is the numerical solution, and the error ϵ in the approximation v is

$$\epsilon = u - v = 0 - v = -v \tag{3.62}$$

The numerical approach used to solve this problem was the Jacobi method, with a weighting factor $w = 2\ /\ 3$. This relaxation scheme is initiated with an arbitrary initial solution and it relaxes the numerical solution towards the exact solution. Figure 24 shows the initial solutions which consisted of sinusoidal functions with different frequencies. The initial solutions were sinusoidal functions since any type of solution can be decomposed into a sum of harmonic functions with different frequencies and amplitudes. The expression for each initial solution was:

$$v_k = sin\,(k\pi x) \tag{3.63}$$

Figure 25 shows that the error of the numerical solution v decreases with each relaxation iteration. The interesting fact is that this rate of decrease is larger for the higher frequency initial solutions ($k = 6$) than for the lower frequency initial solutions ($k = 1$).

If the initial solution is composed of a combination of sinusoidal functions with different frequencies, the error decreases rapidly in the first few iterations, after which it decreases much more slowly. This initial decrease corresponds to the quick elimination of the high-frequency components of the error of the solution. Figure 26 shows the error history for an initial solution composed of different frequencies. The initial solution was equal to $v = 1/3 \cdot [sin(k\pi x) + sin(6k\pi x) + sin(32k\pi x)]$.

Relaxation schemes are not effective at reducing the error of the numerical solution if this error has predominantly smooth components. Assume that a particular relaxation scheme has been applied until only smooth error components remain. On a coarse grid, this smooth error behave like an oscillatory error. This suggests that when relaxation begins to stall, signaling the predominance of smooth error modes, it is advisable to move to a coarser grid, on which those smooth error modes appear

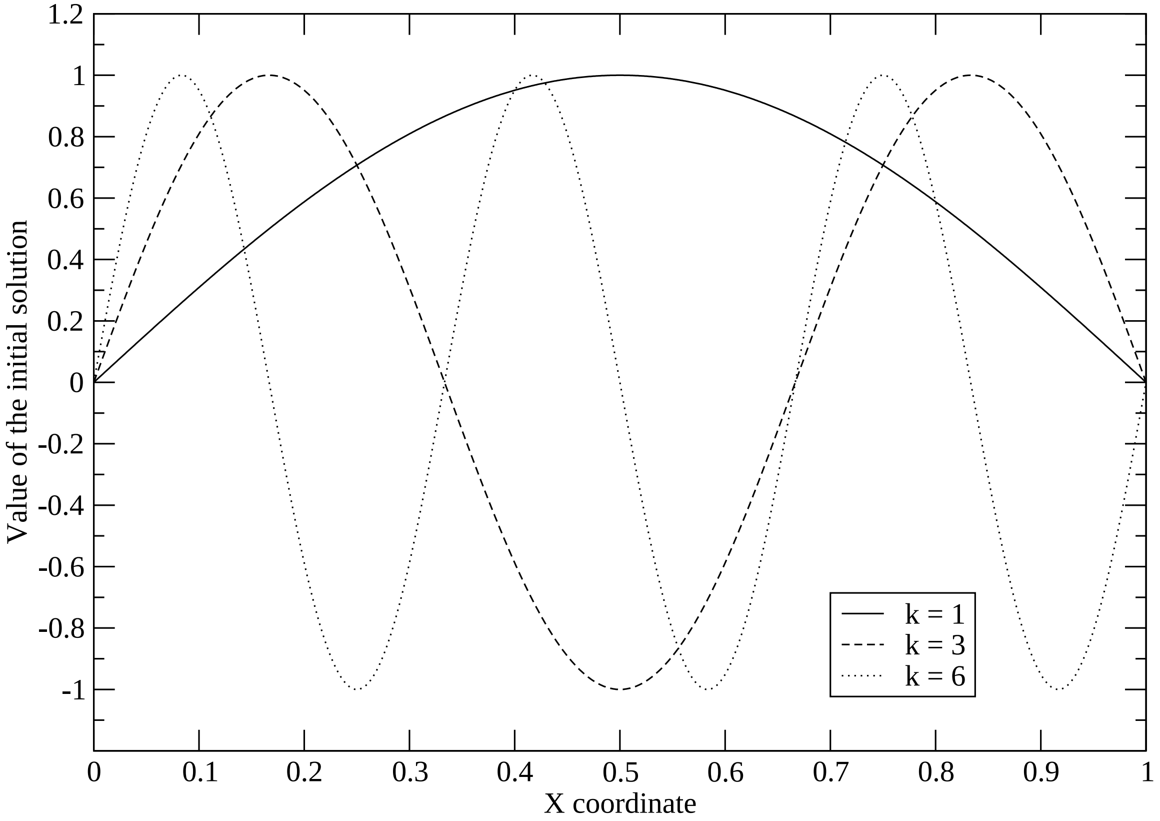


Fig. 24. Arbitrary sinusoidal initial solutions.

more oscillatory and the relaxation scheme will be more effective.

2. Multigrid mesh subdivision

The three grids for the multigrid solver were generated using a topological method called grid refinement. This technique starts from a coarse grid and generates finer grids by element subdivision. New nodes are inserted at the midpoint of the existing edges. Every triangular and quadrilateral elements are subdivided into four elements for the following generation [38]. Figure 27 shows two levels of the multigrid refine-

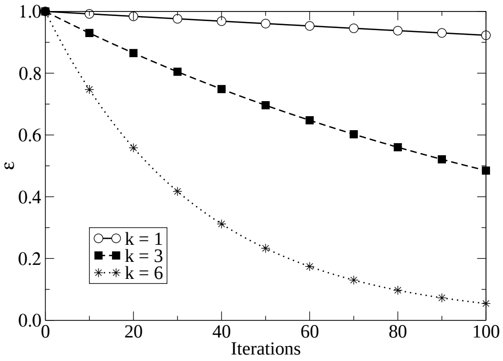


Fig. 25. Error history for sinusoidal initial solutions with different *k* frequencies [37].

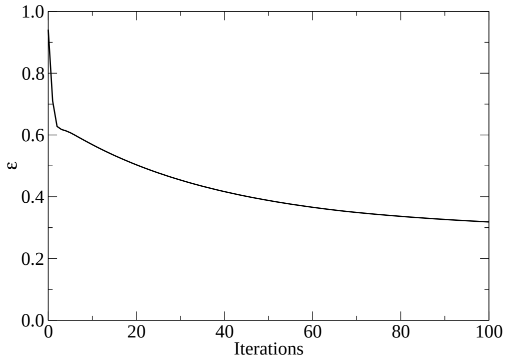


Fig. 26. Error history for initial solution composed of sinusoidal functions with different frequencies [37].

ment technique. The coarse grid is shown in red, and the finer grid is shown in green. Note that each triangular and quadrilateral elements of the grid is subdivided into four new elements.

If the boundaries of the computational domain are non planar it is not sufficient to place the new nodes at the center of the parent edge as is done with the interior edges. If the new nodes are placed at the center of the parent edge, a faceted representation of the original geometry will result. To avoid the faceted representation, the new node is moved from the center of the parent edge to the true shape of the boundary of the domain [39]. Figure 28 shows the boundary elements. Note that the new node is not located at the midpoint of the parent edge, but instead lays on the true shape of the boundary. Consequently, a better representation of the body is obtained as the grid is refined.

The location of the new nodes away from the boundary surface was done by averaging the angles of the edges of the master element, and advancing by a magnitude equal to the average of the length of the edges of the master element. Figure 29

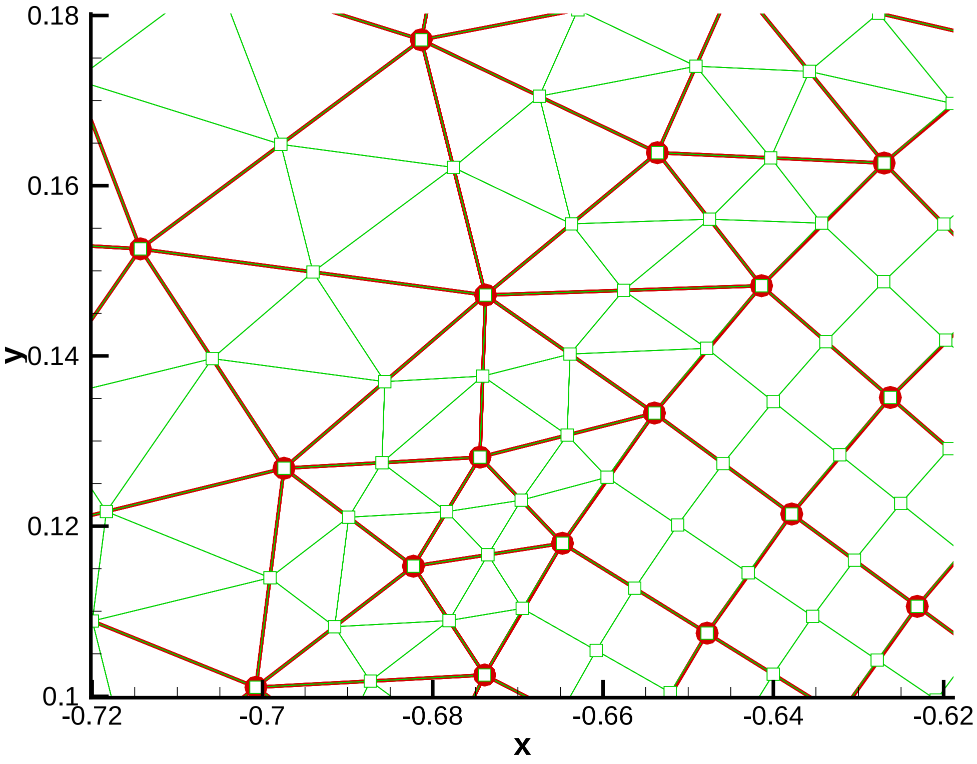


Fig. 27. Triangular and quadrilateral element subdivision.

shows how the new nodes were located, once the boundary nodes were set. Two parameters, α_n and L_n, control the location of these nodes away from the boundary. The parameter α_n is the average angle of inclination of the edge, and L_n is the average distance that the new nodes are to be located. The parameters α_n and L_n were calculated as

$$\alpha_n = \frac{\alpha_1 + \alpha_2}{2} \tag{3.64}$$

$$L_n = \frac{L_1 + L_2}{2} \tag{3.65}$$

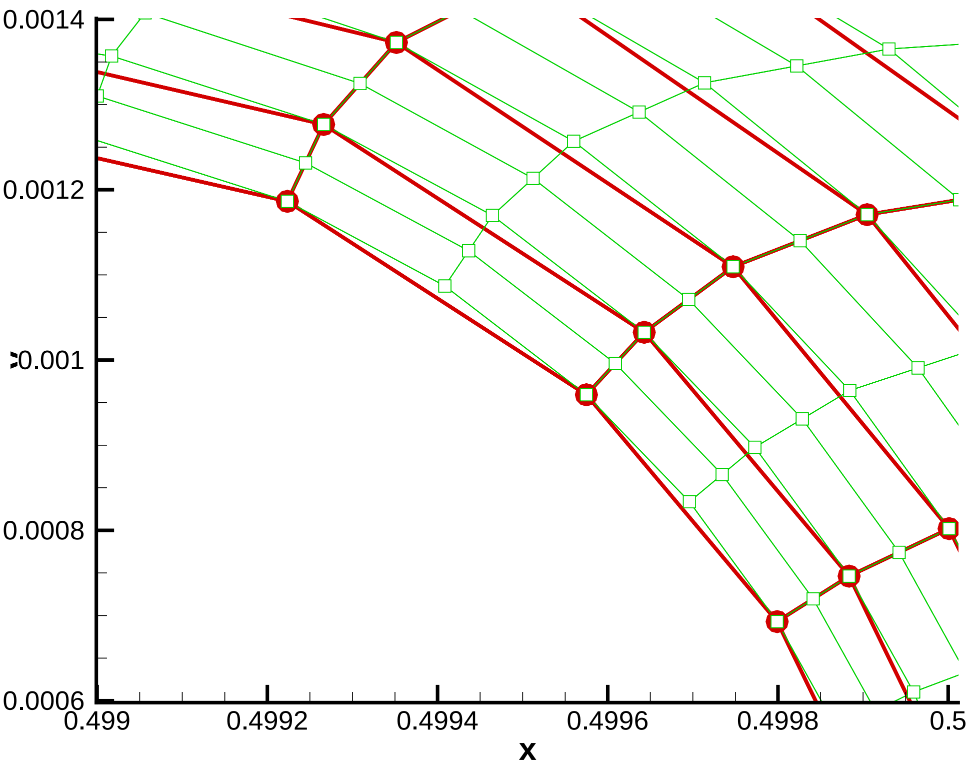


Fig. 28. True shape of boundary is recovered as the grid is refined.

3. Interpolation transfer operators

In the multigrid algorithm, the flow solution obtained on each mesh has to be passed to the other levels of the mesh. This is accomplished by means of interpolation transfer operators, which are linear interpolation schemes used to interpolate the solution from coarser to finer meshes and from finer to coarser meshes. The nodes of the coarse meshes are also nodes of the finer meshes. The solution at these common nodes is transferred from one level to the other by injection, and the value remains the same in both meshes [19]. Four different interpolation schemes were implemented herein for the mesh. One scheme was applied to the internal quadrilateral cells, another to

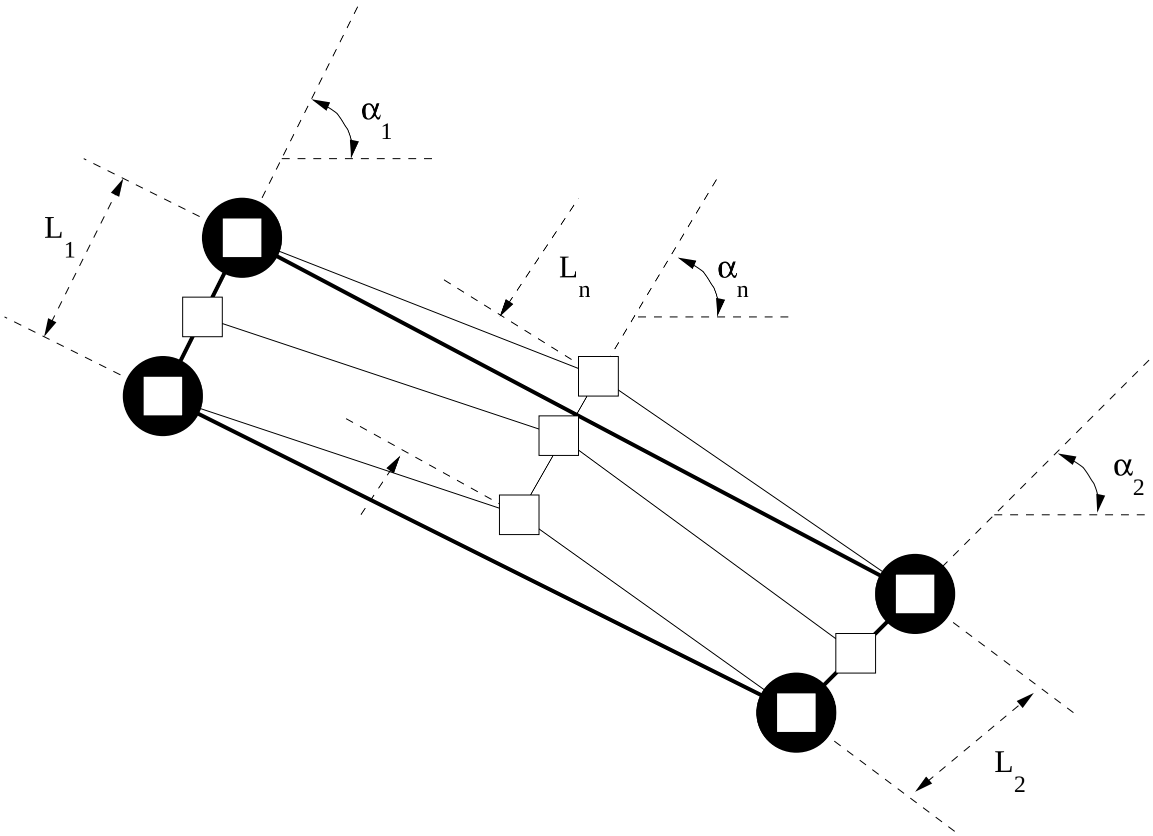


Fig. 29. Boundary node relocation. Black circles represent coarse grid nodes, white squares represent fine grid nodes.

the boundary quadrilateral cells, another one to the triangular cells, and the last one to the cells between the outer boundary of the O-grid and the first layer of triangles. These interpolation schemes are presented in the next four subsections.

a. Interpolation schemes for internal quadrilateral cells

The internal quadrilateral cells are the cells that comprise the O-grid located around the body, excluding the cells located just at the boundary of the O-grid. These internal quadrilateral cells are surrounded by other quadrilateral cells in all directions. Figure 30 shows one of the internal quadrilateral cells. The white nodes are common to the coarse and fine meshes. The black nodes belong only to the fine mesh.

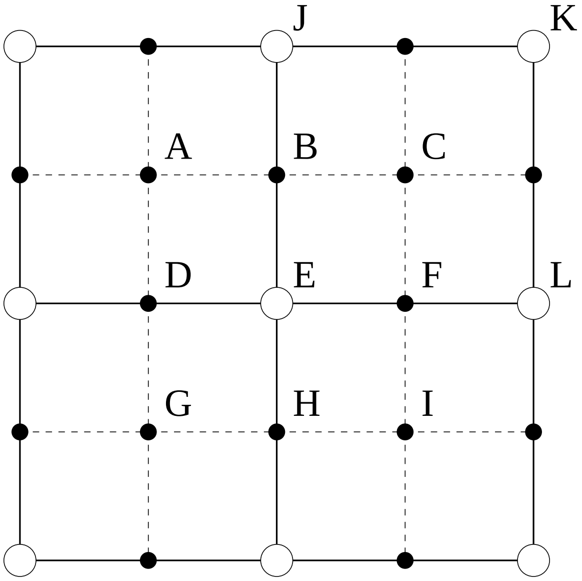


Fig. 30. Coarse and fine meshes for internal quadrilateral cells.

The value of a parameter at node E of the coarse mesh is obtained from the fine mesh using the interpolation formula [19]:

$$Q_E^{coarse} = \frac{1}{4}\left[Q_E + \frac{1}{2}\left(Q_B + Q_D + Q_F + Q_H\right) + \frac{1}{4}\left(Q_A + Q_C + Q_G + Q_I\right)\right]^{fine} \tag{3.66}$$

The value of the parameters at nodes E, B and C of the fine mesh are obtained from the coarse mesh using the interpolation formulas [19]:

$$Q_E^{fine} = Q_E^{coarse}$$

$$Q_B^{fine} = \frac{1}{2} \cdot \left(Q_E + Q_J\right)^{coarse} \tag{3.67}$$

$$Q_C^{fine} = \frac{1}{4} \cdot \left(Q_E + Q_J + Q_K + Q_L\right)^{coarse}$$

b. Interpolation schemes for boundary quadrilateral cells

The boundary quadrilateral cells are the O-grid cells located next to the wall boundary. Figure 31 shows the boundary quadrilateral cells. The white nodes are common to the coarse and fine meshes. The black nodes belong only to the fine mesh.

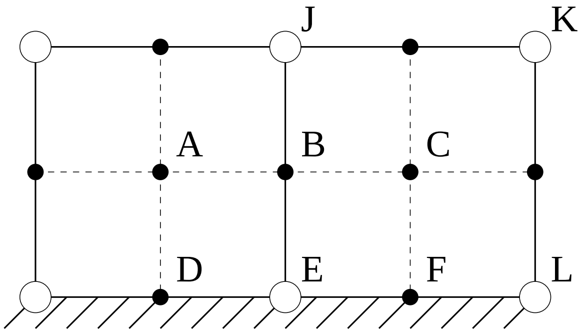


Fig. 31. Coarse and fine meshes for boundary quadrilateral cells.

The value of a parameter at node E of the coarse mesh is obtained from the fine mesh using the interpolation formula:

$$Q_E^{coarse} = \frac{1}{3} \cdot \left[Q_E + \frac{1}{2} \cdot (Q_B + Q_D + Q_F) + \frac{1}{4} \cdot (Q_A + Q_C) \right]^{fine} \tag{3.68}$$

The value of the parameters at nodes E, B and C of the fine mesh are obtained from the coarse mesh using the interpolation formulas:

$$Q_E^{fine} = Q_E^{coarse}$$

$$Q_B^{fine} = \frac{1}{2} \cdot (Q_E + Q_J)^{coarse} \tag{3.69}$$

$$Q_C^{fine} = \frac{1}{4} \cdot (Q_E + Q_J + Q_K + Q_L)^{coarse}$$

c. Interpolation schemes for triangular cells

The triangular cells are the unstructured cells that comprise the region outside of the O-grid. Figure 32 shows a group of triangular cells. The white nodes are common to the coarse and fine meshes. The black nodes belong only to the fine mesh.

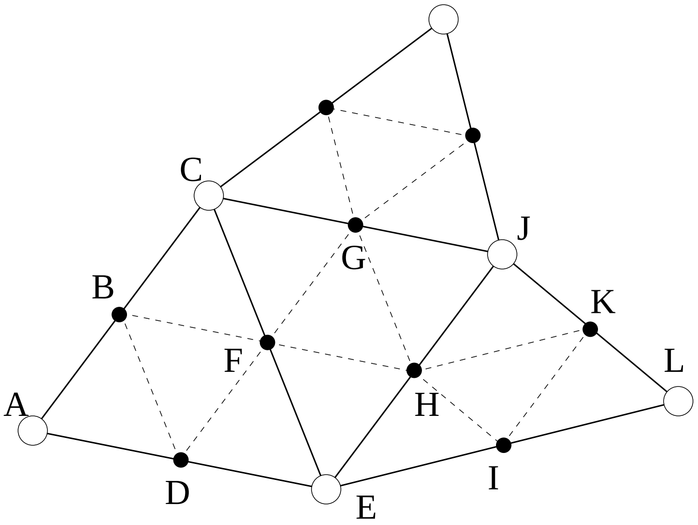


Fig. 32. Coarse and fine meshes for triangular cells.

The value of a parameter at node E of the coarse mesh is obtained from the fine mesh using the interpolation formula:

$$Q_E^{coarse} = \frac{1}{3} \cdot \left[Q_E + \frac{2}{N_T} \sum_i Q_i \right]^{fine} \tag{3.70}$$

where N_T is the number of nodes that are connected to the node being interpolated (node E in this case) and the sum index i extends to all the nodes connected to node E.

The value of the parameters at nodes E, D and F of the fine mesh are obtained from the coarse mesh using the interpolation formulas:

$$Q_E^{fine} = Q_E^{coarse}$$

$$Q_D^{fine} = \frac{1}{2} \cdot (Q_A + Q_E)^{coarse} \tag{3.71}$$

$$Q_F^{fine} = \frac{1}{2} \cdot (Q_C + Q_E)^{coarse}$$

d. Interpolation schemes for mixed triangular and quadrilateral cells

The mixed triangular and quadrilateral cells are the cells located at the outer boundary of the O-grid. These cells have contributions from the boundary quadrilateral cells of the O-grid, and contributions from the first layer of triangular cells. Figure 33 shows a group of mixed triangular and quadrilateral cells. The white nodes are common to the coarse and fine meshes. The black nodes belong only to the fine mesh.

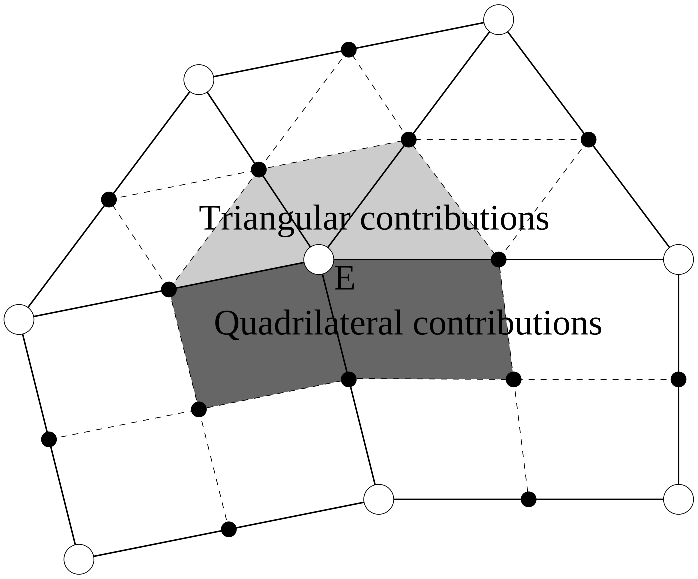


Fig. 33. Coarse and fine meshes for mixed triangular and quadrilateral cells.

The value of a parameter at node E of the coarse mesh is obtained from the contributions of the quadrilateral and triangular cells of the mesh. The triangular contribution is calculated with Eq. (3.70). The boundary quadrilateral contribution is calculated with Eq. (3.68).

4. Implementation of multigrid on the flow solver

This section presents a summary of the multigrid implementation in the flow solver. The flow solver utilizes an explicit time integration algorithm with a four step Runge-Kutta scheme.

1. Calculate the flow solution for the fine mesh.

   $$\frac{d}{dt}\vec{Q}_{fine} = -\frac{1}{\Omega_{fine}}\vec{R}_{fine}$$

   where $\vec{Q}$ is the flow state vector, Ω is the cell volume, and $\vec{R}$ is the residual of the Navier-Stokes equations.

2. Obtain the flow solution for the medium mesh by interpolating the flow solution from the fine mesh.

   $$\vec{Q}_{medium} = I_{fine}^{medium} \cdot \vec{Q}_{fine}$$

   where I_{fine}^{medium} is the interpolation transfer operator from the fine mesh to the medium mesh.

3. Obtain the medium mesh residual by interpolating the residual from the fine mesh.

   $$\vec{R}_{medium} = I_{fine}^{medium} \cdot \vec{R}_{fine}$$

4. Calculate the initial residual $\vec{R}^{(0)}_{medium}$ on the medium mesh using the medium mesh state vector. Only the first stage of the Runge-Kutta scheme is used.

5. Calculate the forcing function for the medium mesh, as the difference between

the interpolated residual and the calculated residual.

$$\vec{F}_{medium} = \vec{R}_{medium} - \vec{R}^{(0)}_{medium}$$

6. Calculate the solution update for the medium mesh, using the four stage Runge-Kutta scheme. The forcing function $\vec{F}$ is added to every stage.

   do k = 1, 4 ! number of Runge-Kutta stages

   $$\vec{Q}^{(k)}_{medium} = \vec{Q}^{(0)}_{medium} + \alpha_k \cdot \frac{\Delta t_{medium}}{\Omega_{medium}} \cdot \left[\vec{R}^{(k-1)}_{medium} + \vec{F}_{medium}\right]$$

   end do

7. Obtain the flow solution for the coarse mesh by interpolating the flow solution from the medium mesh.

   $$\vec{Q}_{coarse} = I^{coarse}_{medium} \cdot \vec{Q}_{medium}$$

   where I^{coarse}_{medium} is the interpolation transfer operator from the medium mesh to the coarse mesh.

8. Obtain the coarse mesh residual by interpolating the residual from the medium mesh.

   $$\vec{R}_{coarse} = I^{coarse}_{medium} \cdot \vec{R}_{medium}$$

9. Calculate the initial residual $\vec{R}^{(0)}_{coarse}$ on the coarse mesh using the coarse mesh state vector. Only the first stage of the Runge-Kutta scheme is used.

10. Calculate the forcing function for the coarse mesh, as the difference between the interpolated residual and the calculated residual.

    $$\vec{F}_{coarse} = \vec{R}_{coarse} - \vec{R}^{(0)}_{coarse}$$

11. Calculate the solution update for the coarse mesh, using the four stage Runge-Kutta scheme. The forcing function $\vec{F}$ is added to every stage.

    do k = 1, 4 ! number of Runge-Kutta stages

    $$\vec{Q}^{(k)}_{coarse} = \vec{Q}^{(0)}_{coarse} + \alpha_k \cdot \frac{\Delta t_{coarse}}{\Omega_{coarse}} \cdot \left[\vec{R}^{(k-1)}_{coarse} + \vec{F}_{coarse}\right]$$

    end do

12. Calculate the solution correction on the coarse mesh.

    $$\delta\vec{Q}_{coarse} = \vec{Q}^{(k)}_{coarse} - \vec{Q}^{(0)}_{coarse}$$

13. Interpolate the solution correction to the medium mesh.

    $$\delta\vec{Q}^{(0)}_{medium} = I^{medium}_{coarse} \cdot \delta\vec{Q}_{coarse}$$

    where I^{medium}_{coarse} is the interpolation transfer operator from the coarse mesh to the medium mesh.

14. Add the coarse mesh interpolated correction to the medium mesh solution.

    $$\vec{Q}^{(k)}_{medium} = \vec{Q}^{(k)}_{medium} + \delta\vec{Q}^{(0)}_{medium}$$

15. Calculate the solution correction on the medium mesh.

    $$\delta\vec{Q}_{medium} = \vec{Q}^{(k)}_{medium} - \vec{Q}^{(0)}_{medium}$$

16. Interpolate the solution correction to the fine mesh.

    $$\delta\vec{Q}^{(0)}_{fine} = I^{fine}_{medium} \cdot \delta\vec{Q}_{medium}$$

    where I^{fine}_{medium} is the interpolation transfer operator from the medium mesh to the fine mesh.

17. Add the medium mesh interpolated correction to the fine mesh solution.

$$\vec{Q}^{(k)}_{fine} = \vec{Q}^{(k)}_{fine} + \delta\vec{Q}^{(0)}_{fine}$$

5. Multigrid versus one-level grid solvers

This section presents the results generated to verify the multigrid flow solver implementation.

The test case chosen is the flow over a NACA 0012 airfoil at an angle of attack of 6.776 degrees. The flow is assumed to be two-dimensional, therefore the mesh has only two layers of nodes in the transversal dimension. The lateral boundary condition used is the no-penetration condition. The inlet and outlet boundaries are located at 12 chords away from the airfoil, and the upper and lower boundaries are located at 9 chords away from the airfoil.

The coarse mesh, shown in Fig. 34, has 64 nodes around the airfoil, 16 layers of nodes for the O-grid around the airfoil, 16 nodes at the inlet and outlet boundaries, and 20 nodes at the upper and lower boundaries. The medium and fine meshes are obtained by mesh subdivision. Each quadrilateral and triangular cell is subdivided in four cells from the coarse to the medium mesh, and again subdivided in four from the medium to the fine mesh. Therefore, the fine mesh has 256 nodes around the airfoil, 64 layers of nodes for the O-grid around the airfoil, 64 nodes at the inlet and outlet boundaries, and 80 nodes at the upper and lower boundaries.

Figure 35 shows the leading edge section of the airfoil on the lower right corner, the O-grid that surrounds the airfoil surface, and the unstructured mesh that covers the rest of the domain. In this figure the three levels of multigrid meshes are present. The coarse mesh is plotted in red, the medium mesh in blue, and the fine mesh in

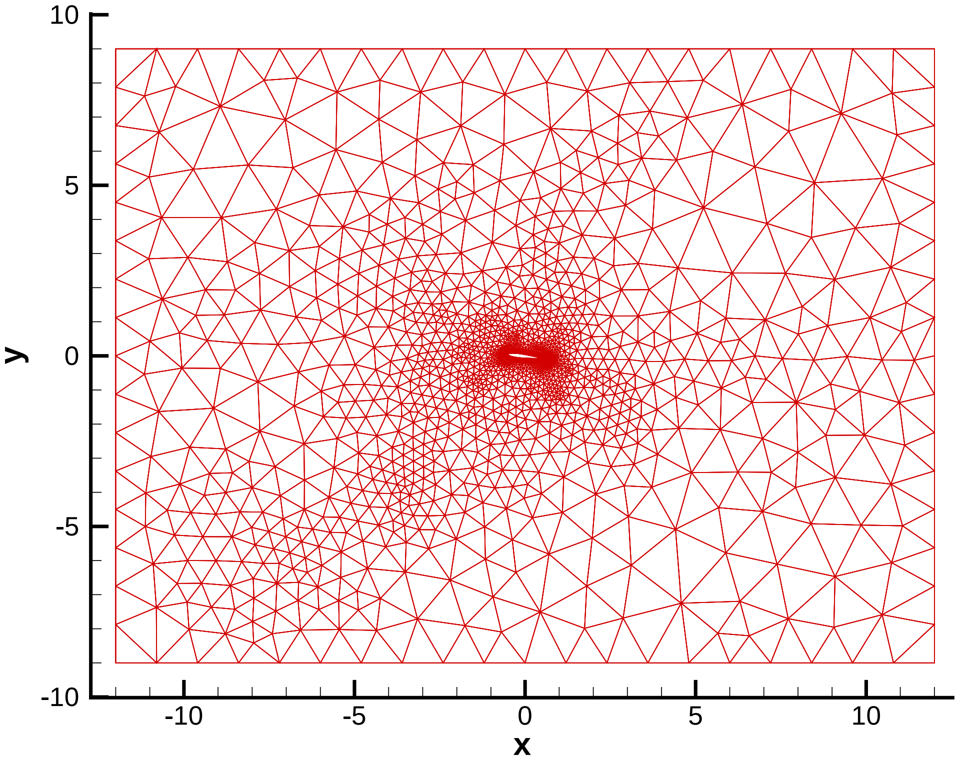


Fig. 34. Coarse mesh for the NACA 0012 airfoil.

green.

a. Inviscid flow results

In order to verify the multigrid inviscid flow solver, two cases were run and compared. One case was the original one-level grid flow solver using only the fine mesh, and the other case was the multigrid flow solver using the three levels of mesh, coarse, medium and fine. The residuals of the Navier-Stokes equations (3.35), the maximum Mach number and the integrated lift over the body were compared for the two cases.

Figure 36 shows the maximum value of the residuals, for both the one-level grid

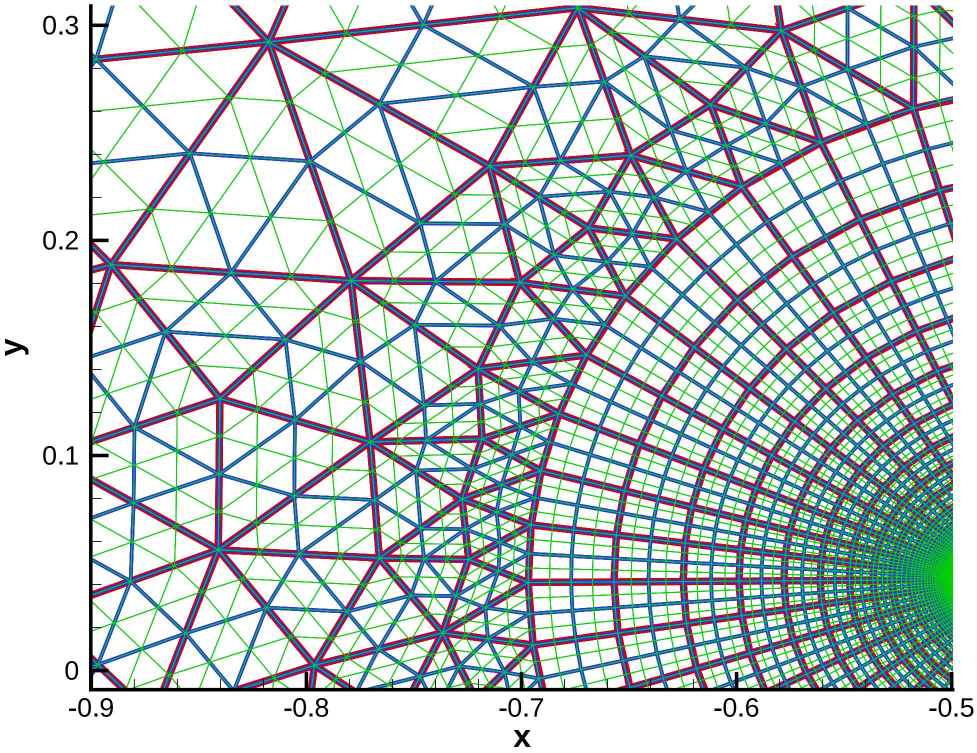


Fig. 35. Three level multigrid meshes. The coarse mesh is plotted in red, the medium mesh in blue, and the fine mesh in green.

and multigrid solvers. Four components of the residual were monitored, the density ρ, the density times the x-axis velocity ρU, the density times the y-axis velocity ρV, and the density times the total energy ρE. The solvers run for 20,000 iterations using first order spatial accuracy, and then 40,000 iterations using second order spatial accuracy. This is the reason for the jump observed in the residuals at 20,000 iterations.

Figure 36 shows that the maximum value of the residuals for the multigrid solver were reduced much faster than the maximum value of the residuals for the one-level grid solver. For the one-level grid solver, the density residual slope for the last 10,000 iterations was -1.20901 / 10,000 iterations. For the multigrid solver, the density residual slope for the last 10,000 iterations was -2.0059 / 10,000 iterations. This means that the multigrid solver reduced the residuals 65.9% faster than the one-level multigrid solver. In addition, it took 33,353 iterations for the one-level grid density residual to reach a value of 10^{-9}. It took 20,054 iterations for the multigrid density residual to reach a value of 10^{-9}. The reduction in the number of iterations corresponded to a factor of 1.663.

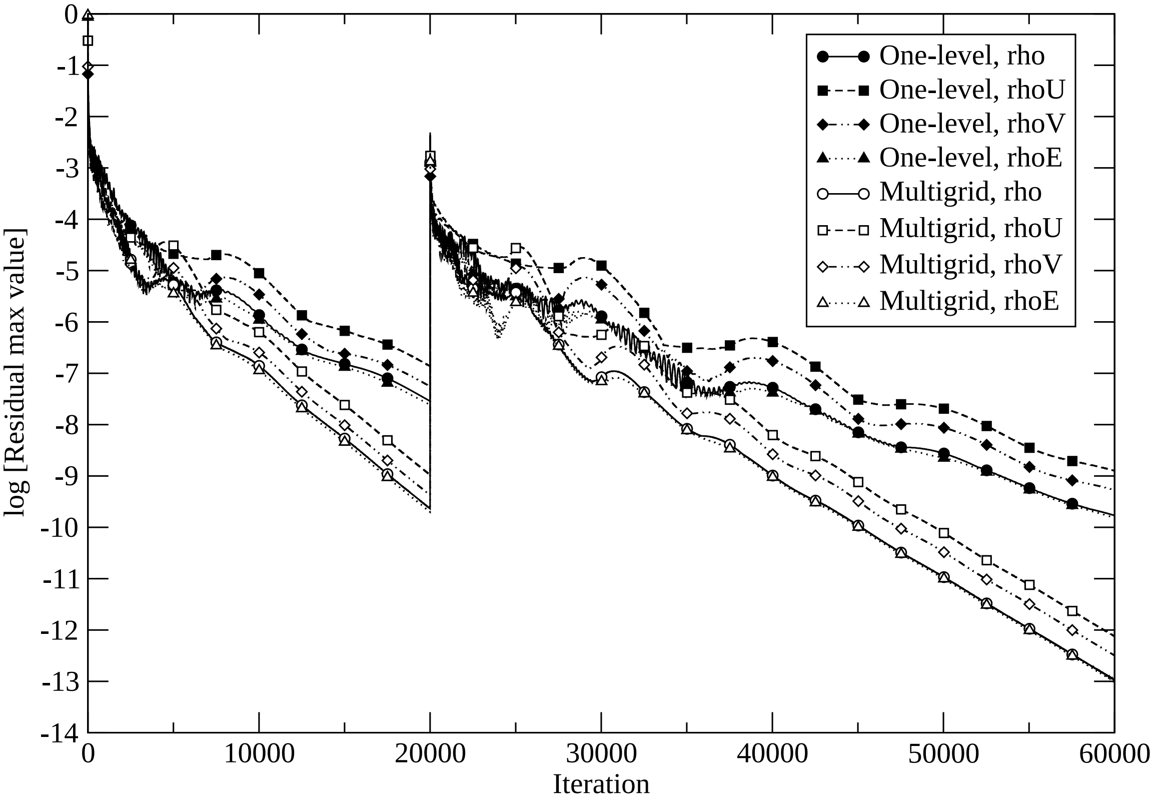


Fig. 36. Maximum residual for one-level grid and multigrid flow solvers.

Figure 37 shows the time history of the maximum value of the Mach number over all the cells of the domain. The final value of the maximum Mach number was 0.63267272 for the one-level grid solver, and 0.63267379 for the multigrid solver. The difference for the maximum value of the Mach number was $1.69 \cdot 10^{-8}\%$. Figure 38 shows the time history of the integrated lift on the airfoil. The final value of the integrated lift was 26544.602 for the one-level grid solver, and 26544.708 for the multigrid solver. The difference for the integrated lift was $3.99 \cdot 10^{-8}\%$.

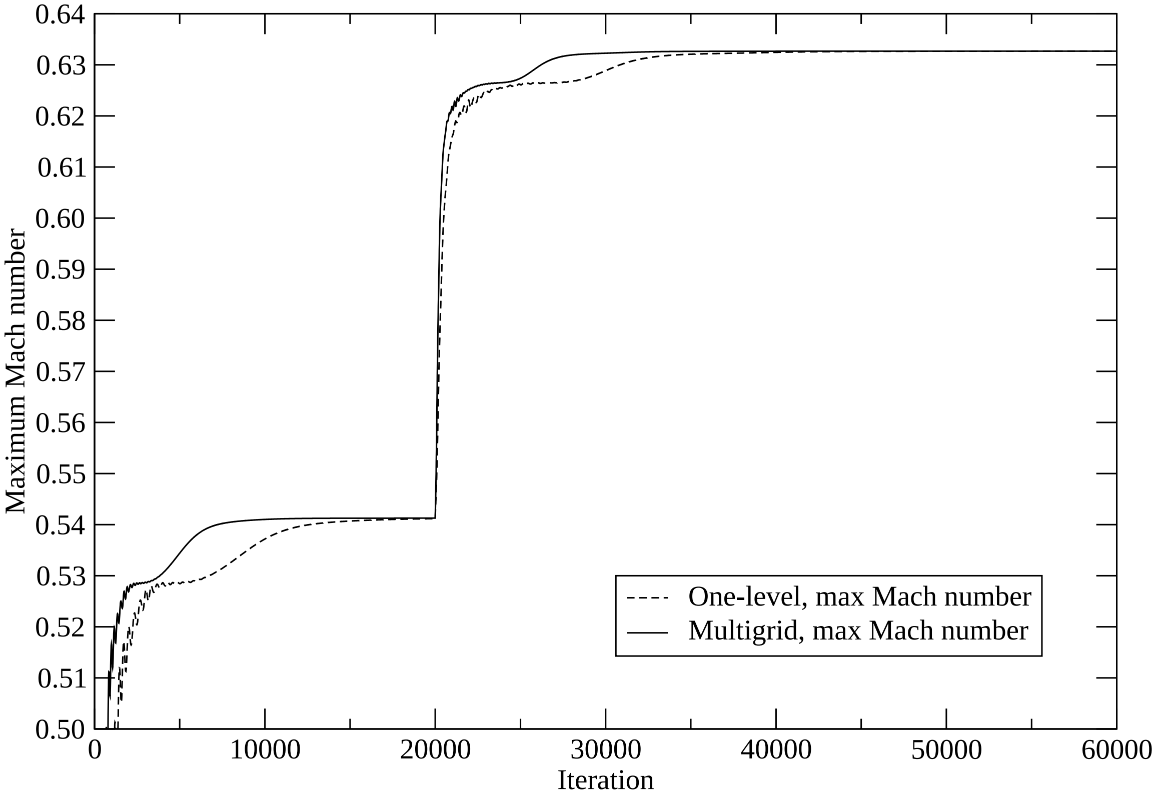


Fig. 37. Maximum Mach number for one-level grid and multigrid flow solvers.

b. Turbulent flow results

In order to verify the multigrid turbulent flow solver, two cases were run and compared in a similar way as with the inviscid flow solver. One case was the original one-level grid flow solver using only the fine mesh, and the other case was the multigrid flow

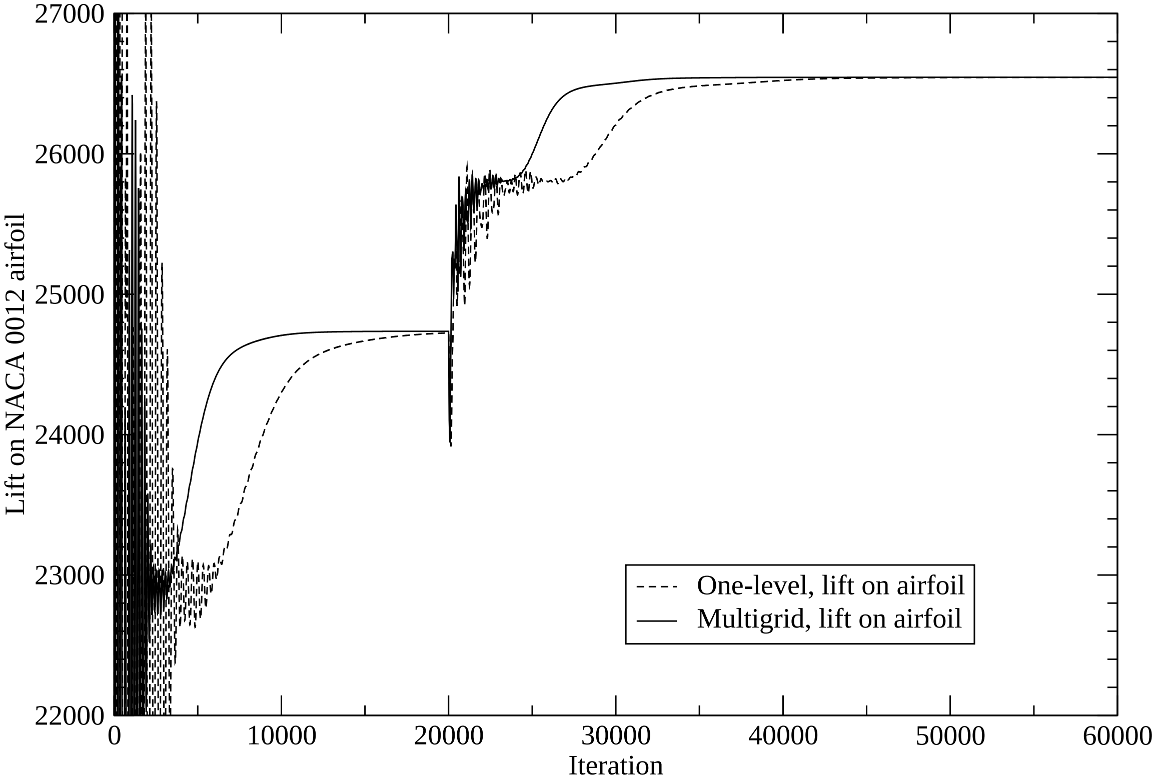


Fig. 38. Lift on NACA 0012 airfoil for one-level grid and multigrid flow solvers.

solver using the three levels of mesh, coarse, medium and fine. The residuals of the Navier-Stokes equations (3.35), the maximum Mach number and the integrated lift over the body were compared for the two cases. The test case chosen was the flow over a NACA 0012 airfoil at an angle of attack of 6.776 degrees.

Figure 39 shows the maximum value of the residuals, for both the one-level grid and multigrid solvers. Four components of the residual were monitored, the density ρ, the density times the x-axis velocity ρU, the density times the y-axis velocity ρV, and the density times the total energy ρE. The solvers run for 20,000 iterations using first order spatial accuracy, and then 40,000 iterations using second order accuracy.

This is the reason for the jump observed in the residuals at 20,000 iterations.

Figure 39 shows that the maximum value of the residuals for the multigrid solver were reduced much faster than the maximum value of the residuals for the one-level grid solver. For the one-level grid solver, the ρU residual slope for the 30,000 - 40,000 iteration segment was -0.84083 / 10,000 iterations. For the multigrid solver, the ρU residual slope for the 30,000 - 40,000 iteration segment was -1.56493 / 10,000 iterations. This means that the multigrid solver reduced the residuals 86.1% faster than the one-level multigrid solver. In addition, it took 36,080 iterations for the one-level grid ρU residual to reach a value of 10^{-7}. It took 14,850 iterations for

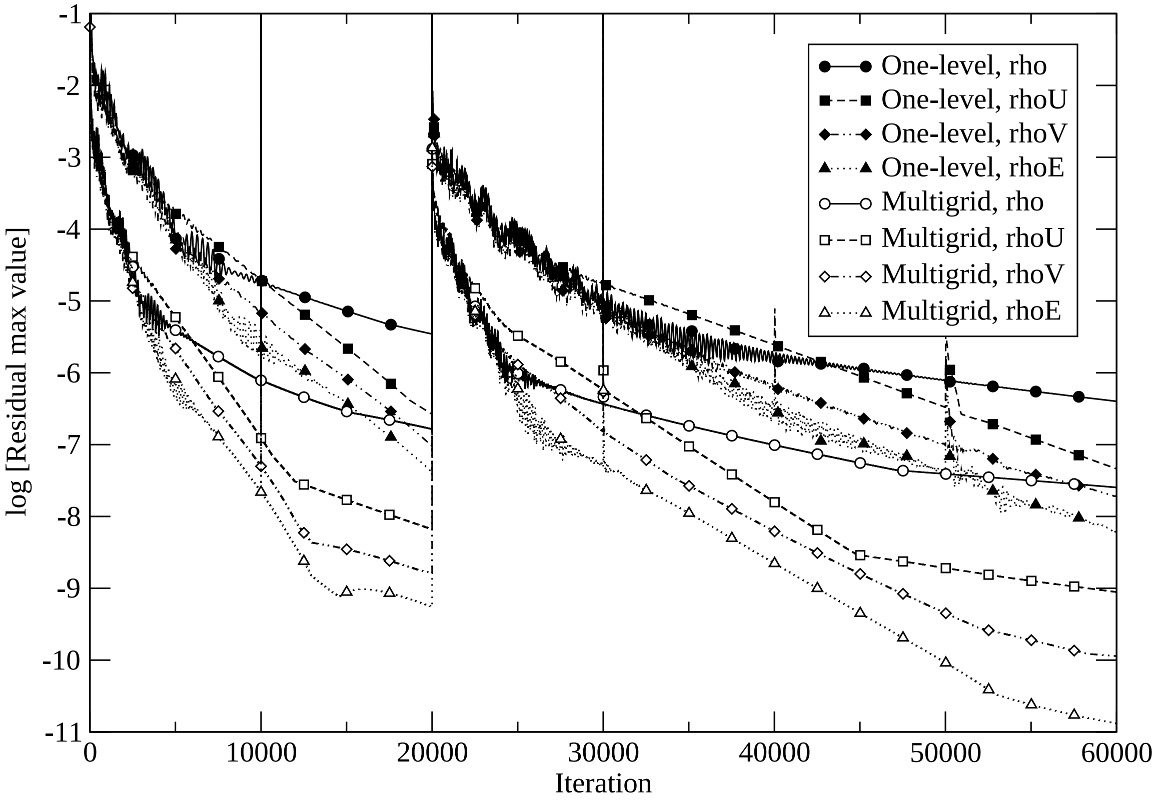


Fig. 39. Maximum residual for one-level grid and multigrid flow solvers.

To verify the multigrid solver implementation, the converged solution obtained with both the one-level grid solver and the multigrid solver has to be almost identical. To calculate the difference between the multigrid and the one-level grid solutions, two parameters were selected, the pressure coefficient C_p and the wall shear stress τ_w. The pressure coefficient is defined as

$$C_p = \frac{P_{local} - P_\infty}{\frac{1}{2}\rho_\infty V_\infty^2} \tag{3.72}$$

where P_{local} is the local pressure on the airfoil, and P_∞, ρ_∞, and V_∞ are the freestream pressure, density and velocity respectively.

The wall shear stress τ_w is defined as

$$\tau_w = \mu \frac{\partial V}{\partial y} \tag{3.73}$$

where μ is the local kinematic viscosity and $\frac{\partial V}{\partial y}$ is the velocity gradient at the wall.

The parameters C_p and τ_w were calculated for each node on the surface of the airfoil. The values of these parameters were compared between the one-level grid solver and the multigrid solver.

The average and maximum difference values were computed using the following formulas [40]

$$\phi_{max} = max_{i\epsilon[1,N]} \frac{|\phi_i^{one-levelgrid} - \phi_i^{multigrid}|}{\phi_{max} - \phi_{min}} \tag{3.74}$$

$$\phi_{avg} = \frac{1}{N} \sum_{i=1}^{N} \frac{|\phi_i^{one-levelgrid} - \phi_i^{multigrid}|}{\phi_{max} - \phi_{min}} \tag{3.75}$$

where ϕ is either the C_p or τ_w variables, and N is the number of nodes around the airfoil. ϕ_{max} and ϕ_{min} were taken as the maximum and minimum values of ϕ among all the nodes around the airfoil and between the one-level grid solution and

Table II. Average and maximum difference between the one-level grid solution and the multigrid solution.

Variable	average difference	maximum difference
C_p	7.137 E-6	4.339 E-5
τ_w	1.106 E-4	9.928 E-4

the multigrid solution.

The values of the maximum and average difference for the C_p or τ_w variables between the one-level grid solution and the multigrid solution are presented in Table II.

Since the multigrid residuals are reduced much faster than the one-level grid residuals, the converged multigrid steady state solution is achieved in fewer iterations. This can be shown plotting a certain parameter as a function of the number of iterations. Figure 40 shows the time history of the maximum value of the Mach number over all the cells of the domain. The final value of the maximum Mach number was 0.5967848 for the one-level grid solver, and 0.5967651 for the multigrid solver. The difference for the maximum value of the Mach number was $3.3 \cdot 10^{-3}\%$. Figure 41 shows the time history of the integrated lift on the airfoil. The final value of the integrated lift was 23218.5 for the one-level grid solver, and 23217.9 for the multigrid solver. The difference for the integrated lift was $2.6 \cdot 10^{-3}\%$.

The fact that the one-level grid solution is almost identical to the multigrid solution indicates that the accuracy of the multigrid flow solver is the same as the accuracy for the one-level grid flow solver, proving that no accuracy loss is associated with the multigrid technique. This is possible in the multigrid solver because the

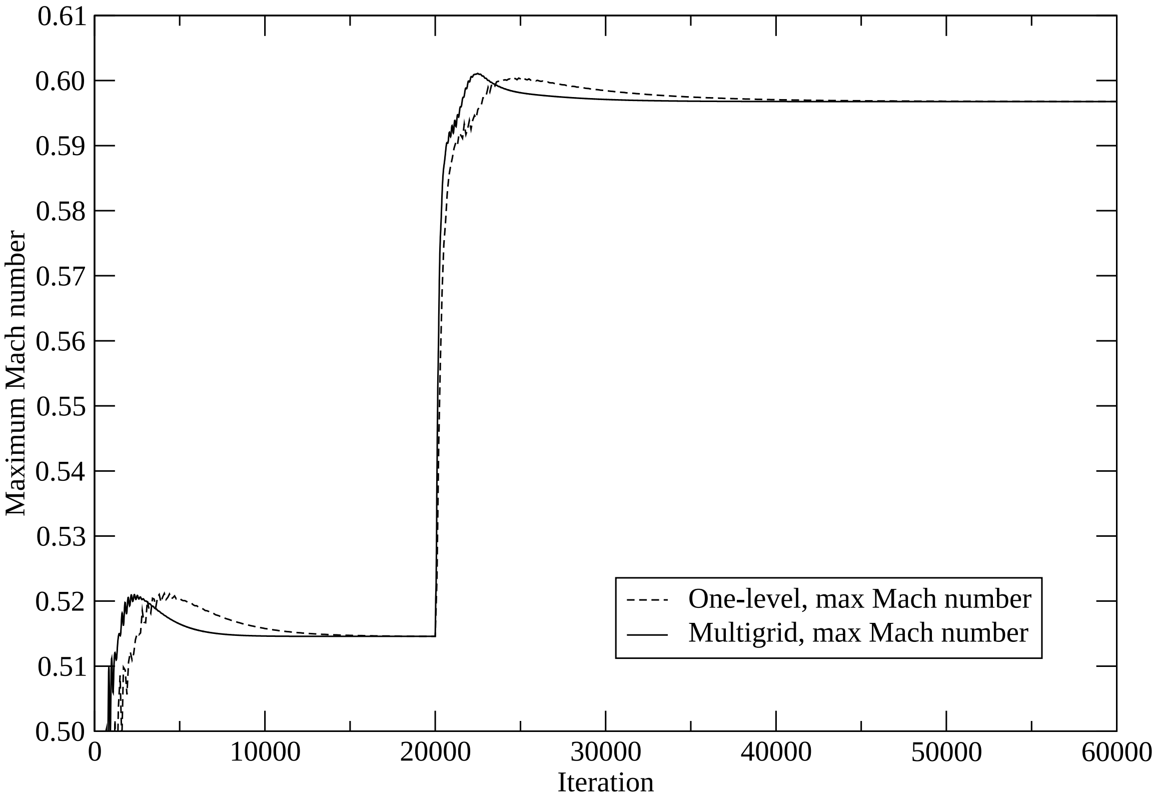


Fig. 40. Maximum Mach number for one-level grid and multigrid flow solvers.

residual of the fine mesh governs the residuals of the medium and coarse meshes, and therefore the accuracy for the system of meshes is maintained.

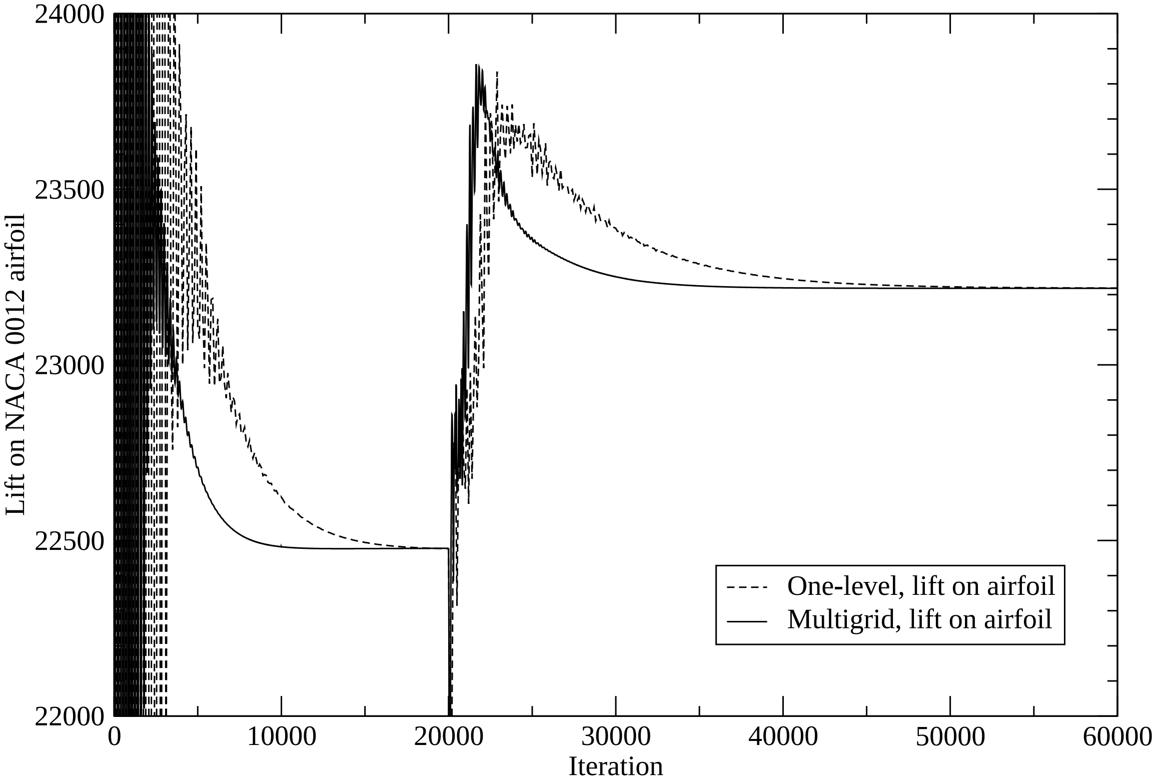


Fig. 41. Lift on NACA 0012 airfoil for one-level grid and multigrid flow solvers.

CHAPTER IV

RESULTS

This chapter presents the results generated for the validation and verification of the flow solver, and aeroelastic results for the F-5 wing, the Nonlinear Aeroelastic Test Apparatus (NATA) wing, and the Goland wing.

The validation results include the turbulent flow over a flat plate. This case was used to validate the implementation of the turbulence model. A second case used for the validation of the flow solver was the NACA 0012 airfoil at an angle of attack of 6.776 degrees. This case was used to validate the implementation of the convective and viscous fluxes, the time integration algorithm, and the boundary conditions. The numerical results for the NACA airfoil were compared against experimental results. Four different grids were used in this simulation to verify that the solution was grid independent. The last validation simulation was for the steady flow over the F-5 wing. The numerical results for this wing were compared against experimental results and other numerical results.

The aeroelastic results include the numerical simulation for the F-5 wing undergoing forced pitch motion. These results were compared against other experimental and numerical results. Aeroelastic results are also presented for the Nonlinear Aeroelastic Test Apparatus (NATA) wing, including a comparison with experimental results. The last set of aeroelastic results correspond to the Goland wing aeroelastic cases, which include a comparison against the quasi-steady aerodynamic model for low Mach numbers, a study of the influence of the nonlinear structural terms in the LCO characteristics, and the calculation of the stability boundary for the Goland wing.

A. Validation and verification of the flow solver

1. Steady flow over a flat plate

This section presents the numerical results for the simulation of the turbulent steady flow over a flat plate. This simulation was used to validate the implementation of the turbulence model in the flow solver. The numerical results were compared against the analytical turbulent boundary layer solutions for the viscous sublayer and logarithm law. Two different grids were used to conduct a grid refinement test. The turbulent effects were modeled using the Shear Stress Transport (SST) model of Menter [17].

The computational domain included the plate region and the freestream region upstream of the flat plate. Figure 42 shows the fine mesh used for the turbulent flat plate simulation. The flat plate region extended from x = 0.0 to x = 0.2. The freestream region extended from x = -0.05 to x = 0.0. The domain extended to y = 0.02 upwards from the plate. Two different grids were used for grid refinement analysis. The coarse grid had 41 nodes in the flow direction and 41 nodes in the transversal direction, and the fine grid had 81 nodes in the flow direction and 81 nodes in the transversal direction.

The flow conditions were Mach number = 0.3, static pressure P_0 = 101,325 Pa, and static temperature T_0 = 288.15° K. The boundary conditions were set as follows. Subsonic inlet was set for the inlet boundary, subsonic outlet was set for the outlet boundary, wall boundary condition for the plate wall, and no penetration condition at the upper boundary and lower boundary upstream from the plate leading edge. The initial condition was constant freestream applied to all the cells in the domain.

To properly capture the turbulent effects close to the wall, the grid has to be refined in the direction transversal to the wall in such a way that the y^+ number is equal to or less than 1. The y^+ number is defined as

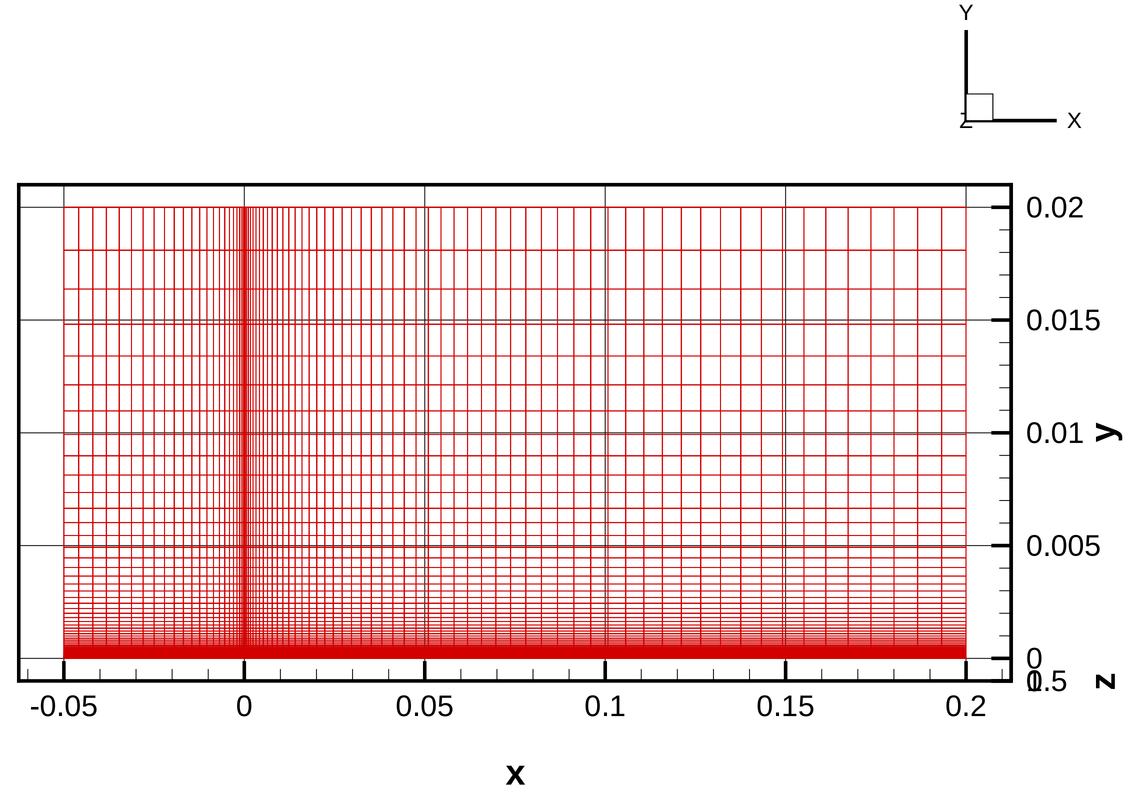


Fig. 42. Mesh used for the turbulent flat plate simulation.

$$y^+ = \frac{u^* \cdot y_1}{\nu} \tag{4.1}$$

where y_1 is the distance from the wall to the first node in the direction transversal to the flow, and $\nu = \mu/\rho$ is the dynamic viscosity of the fluid. The friction velocity u^* is defined as

$$u^* = \sqrt{\frac{\tau_w}{\rho}} \tag{4.2}$$

where τ_w is the shear stress at the wall and ρ is the density of the fluid.

To validate the implementation of the turbulent model, the velocity profile transversal to the flat plate was extracted from the flow field, and the numerical solution was compared against the analytical turbulent boundary layer solutions for the viscous sublayer and logarithm law. The velocity profile was extracted at the middle of the flat plate, for the location $x = 0.1$.

The viscous sublayer, also known as the laminar sublayer, is the region of the flow between the wall and the location where $y^+ = 5.0$. In the viscous sublayer, the flow is not steady, but the velocity fluctuations do not contribute to the total stress because of the overwhelming effects of the viscosity. This means that turbulence cannot sustain itself and the flow behaves as if it was laminar. In the viscous sublayer, the velocity profile is linear [41]:

$$u^+ = y^+ \tag{4.3}$$

where the nondimensional velocity u^+ is defined as $u^+ = u/U$, with U being the freestream velocity.

The inertial sublayer is the region that spans from $y^+ = 30$ to the limit of the turbulent boundary layer. This is a region of approximately constant turbulent stresses. The viscous stresses are much smaller than the turbulent stresses as the distance to the wall increases. This region is called inertial sublayer because of the absence of local viscous effects. The inertial sublayer velocity profile is logarithmic, thus this region is also known as the logarithmic law region. The velocity profile is [41]

$$u^+ = \frac{1}{\kappa} \cdot ln\left(y^+\right) + a \tag{4.4}$$

where $\kappa = 0.40$ is the von Karman constant, and a is a constant equal to 5.0.

There is a third turbulent sublayer, called the buffer layer, which is located between the viscous sublayer and the inertial sublayer, from $y^+ = 5$ to $y^+ = 30$. In this region the viscous and the turbulent effects are equally important, and a transition takes place from the laminar dominated flow of the viscous sublayer to the turbulent dominated flow of the inertial sublayer.

Figure 43 shows the numerical results for the turbulent flat plate using a coarse and a fine grid, and the analytical results for the viscous and inertial sublayers. This figure shows that the solution is grid converged, since both the coarse and the fine grid solutions are almost identical. The figure shows an excellent agreement between the numerical solution and the viscous sublayer solution up to $y^+ = 5.0$, where the viscous sublayer approximation is no longer valid. The inertial sublayer is relatively well captured, with the slopes of the numerical and the analytical solution being similar. The numerical velocity profile reaches a maximum value of $u^+ = 23.5$, which corresponds to the limit of the turbulent boundary layer.

2. Steady flow over the NACA 0012 airfoil

This section presents the numerical results for the simulation of the turbulent steady flow over the NACA 0012 airfoil. This simulation was used to verify and validate the flow solver. The numerical results were compared against experimental results. Four different grids were used to conduct a grid refinement test.

The airfoil chosen for this numerical simulation was the NACA 0012 airfoil, which is defined by the following analytical expression:

$$y(x) = 0.6 \cdot \left(0.2969 \cdot \sqrt{x} - 0.126 \cdot x - 0.3516 \cdot x^2 + 0.2843 \cdot x^3 - 0.1015 \cdot x^4\right) \quad (4.5)$$

The NACA 0012 airfoil was rotated to an angle of attack of 6.776 degrees. The inlet and outlet boundaries were located at 12 chords away from the center of the

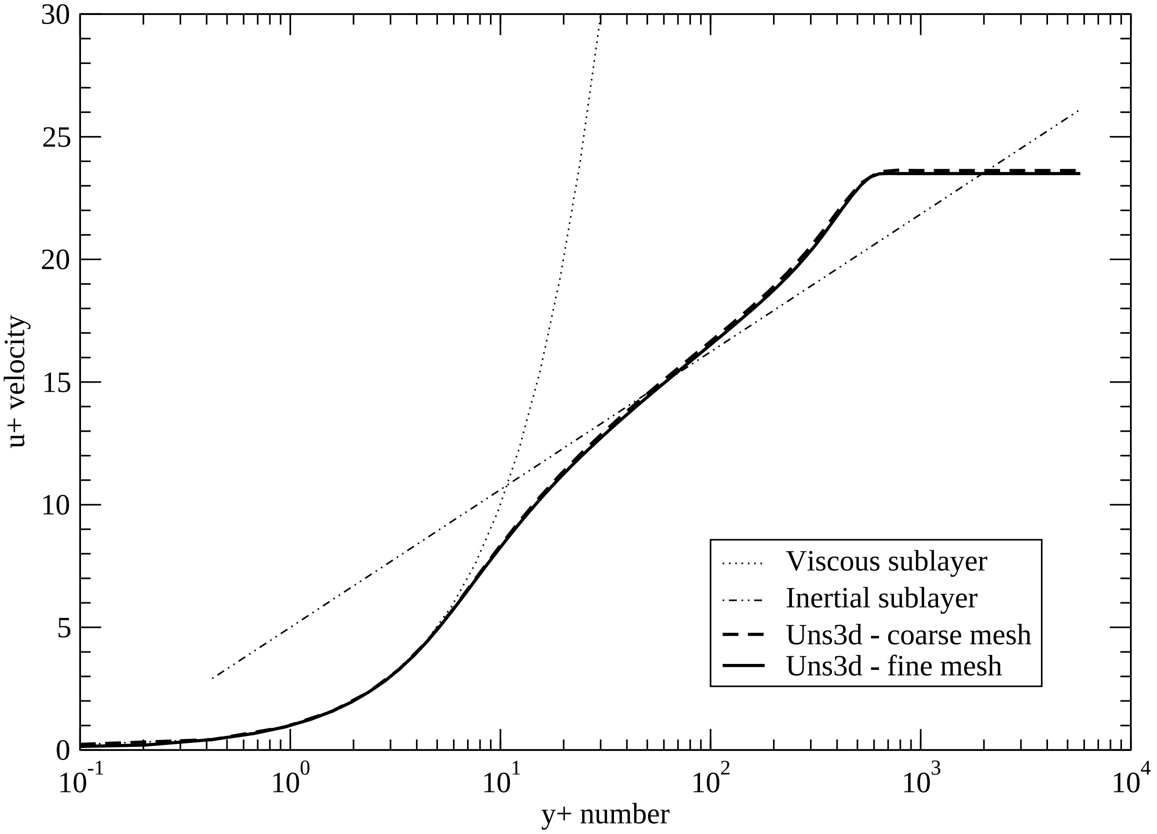


Fig. 43. Turbulent boundary layer for flat plate.

airfoil, and the upper and lower boundaries were located at 9 chords away from the airfoil. Four different grids were used for this numerical simulation. These grids had 64, 128, 256, and 512 nodes around the airfoil. Other parameters of these grids are found in Table III.

The flow conditions were Mach number = 0.3, static pressure P_0 = 101,325 Pa, and static temperature T_0 = 288.15° K. The boundary conditions were set as follows. The flow at the inlet was assumed subsonic, and the values of the stagnation pressure, stagnation temperature, and flow direction were specified. The flow at the

Table III. NACA 0012 grid parameters.

Grid number	Nodes around the airfoil	Nodes across the O-grid	Maximum value of y^+ in the airfoil	Total number of nodes
1	64	16	0.736	5,562
2	128	32	0.614	15,530
3	256	64	0.567	44,658
4	512	128	0.555	150,222

outlet was assumed subsonic, and the value of the static pressure was specified. Wall boundary conditions were specified for the airfoil surface. No penetration condition were imposed at the upper boundary, lower boundary, and the lateral boundaries of the domain. The initial condition was constant freestream applied to all the cells in the domain.

Figure 44 shows the pressure coefficient c_p results for the four different meshes, compared against the experimental results. In this figure, the solution with 64 nodes around the airfoil shows the accuracy limitations of a very coarse grid. The solution compares relatively well with the experimental results, although the accuracy decays close to the peaks in the solution. Figure 44 also shows that the solution with 128, 256, and 512 nodes around the airfoil is very accurate. The solution compares very well with the experimental results, with the accuracy close to the peaks in the solution depending on the number of nodes around the airfoil.

Table IV compares the numerical and experimental c_p values for the peaks at both the pressure and suction sides.

Figure 44 also shows the grid convergence test performed on the NACA 0012 airfoil simulation. The concept of the grid convergence test is to verify that as the

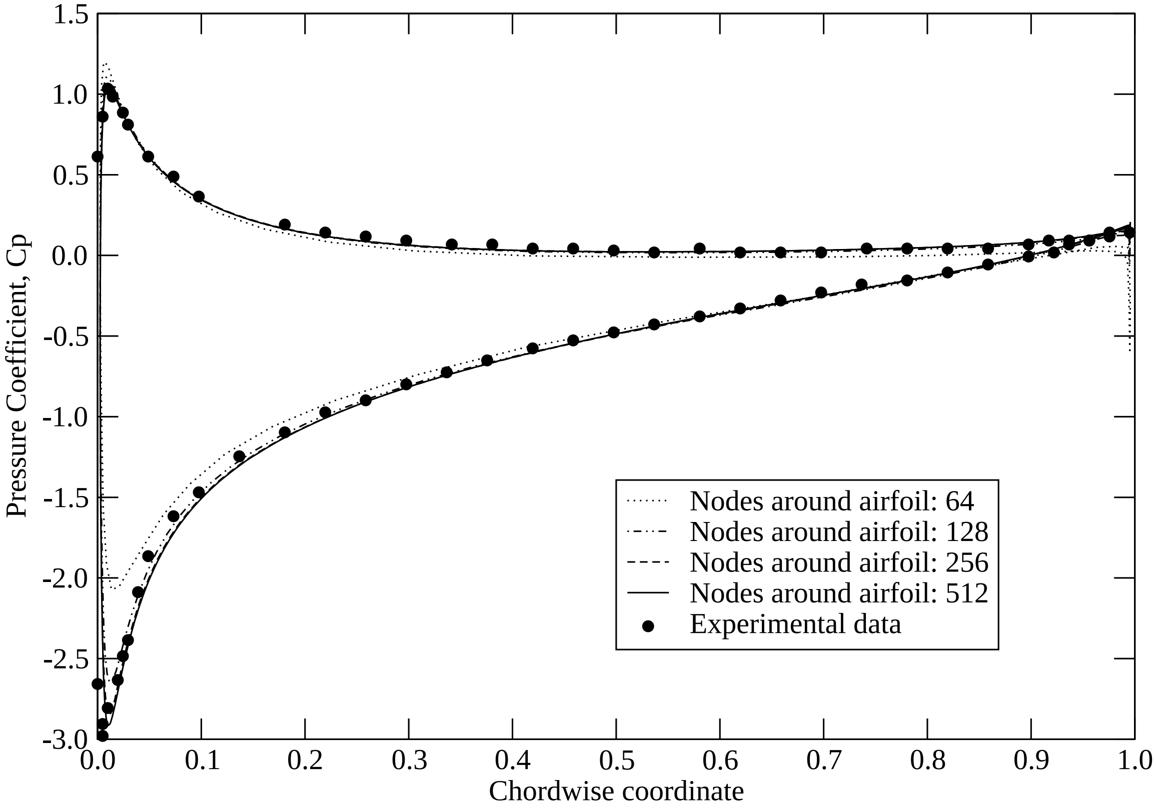


Fig. 44. NACA 0012 pressure coefficients.

mesh is refined, the solution becomes independent of the mesh density. While the solutions obtained with 64, 128 and 256 nodes around the airfoil are clearly different on Figure 44, the solutions obtained 256 and 512 nodes around the airfoil are almost indistinguishable, only at the suction peak can these two solutions be told apart.

3. Steady flow over the F-5 wing

This section presents the numerical results for the simulation of the turbulent steady flow over the F-5 wing. The numerical results were compared against numerical

Table IV. Pressure and suction side peak comparative.

Grid number	Pressure side c_p peak	Error in c_p vs. experimental	Suction side c_p peak	Error in c_p vs. experimental
experimental	1.034	-	2.979	-
1	1.198	15.9 %	2.072	30.4 %
2	1.104	6.8 %	2.648	11.1 %
3	1.065	3.0 %	2.847	4.4 %
4	1.042	0.8 %	2.913	2.2 %

results from Mello and Sankar [42] and experimental data from the National Aerospace Laboratory at the Netherlands [43]. Three different grids were used to conduct a grid refinement test.

The F-5 wing has a span of 0.6476 m, a root chord of 0.6396 m, and a tip chord of 0.18 m. The leading edge angle is 31.9° and the trailing edge angle is 5°. The cross-section of the F-5 wing has a modified NACA 65-A-004.8 airfoil. The coordinates of the airfoil are available at [35]. Figure 45 shows the airfoil section for the root of the wing, as well as the O-grid constructed around the airfoil. Figure 46 shows a three-dimensional view of the surface of the F-5 wing.

The inlet and outlet boundaries were located at 12 chords away from the center of the airfoil, and the upper and lower boundaries were located at 9 chords away from the airfoil. Three different grids were used for this numerical simulation. These grids had 64, 128, and 256, nodes around the airfoil. Other parameters of these grids are found in Table V.

The flow conditions were Mach number = 0.6, static pressure P_0 = 100,000 Pa, and static temperature T_0 = 288.15° K. The boundary conditions were set similarly

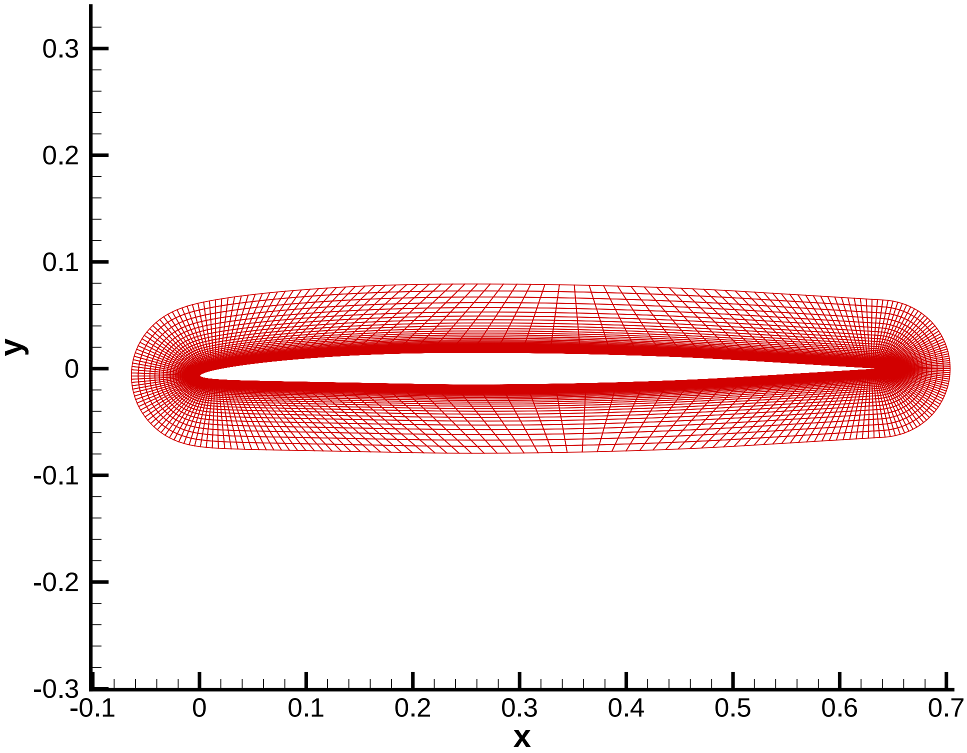


Fig. 45. NACA 65-A-004.8 airfoil and O-grid for the F-5 wing.

to the boundary conditions specified in section 4.1.2.

Figure 47 and 48 show the pressure coefficient c_p results for the three different meshes, compared against numerical results from Mello and Sankar [42] and experimental data from the National Aerospace Laboratory at the Netherlands [43], taken at sections 2 and 5 of the wing. In these figures, the solution with 64 nodes around the airfoil shows the accuracy limitations of a coarse grid. The solution compares relatively well with the experimental results, although the accuracy decays towards the trailing edge, where the predicted c_p is much lower than the experimental results. Figure 47 and 48 also show that the solution with 128 and 256 nodes around the

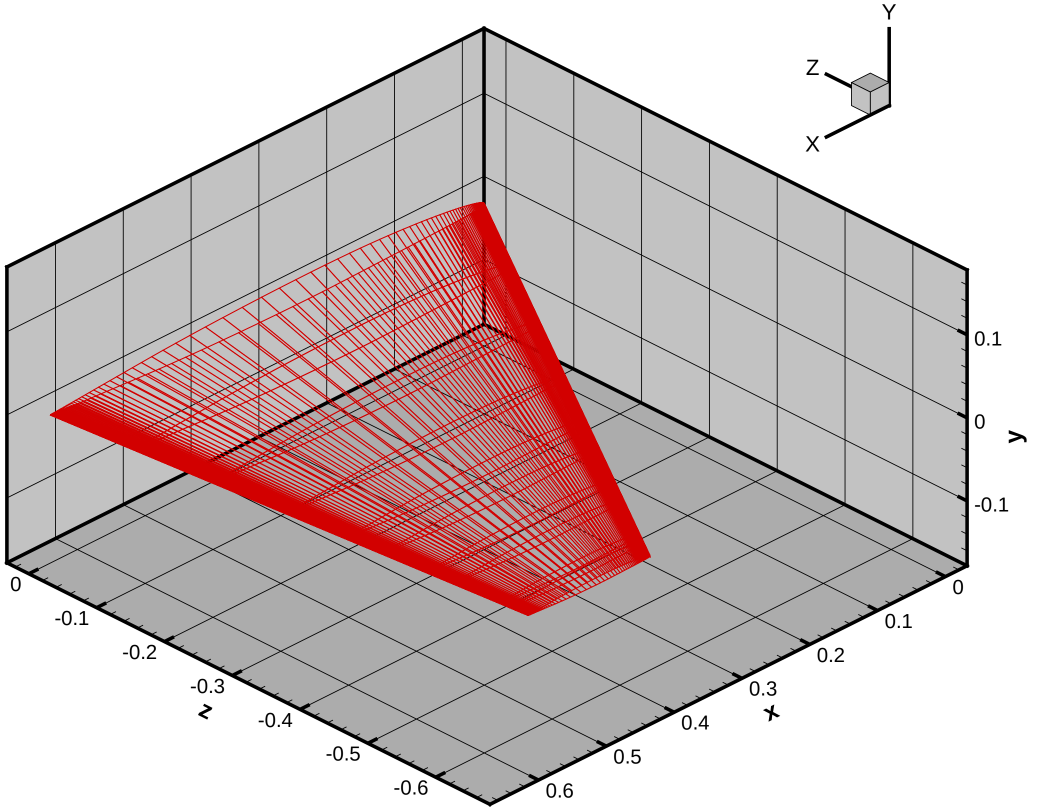


Fig. 46. Three-dimensional view of the surface of the F-5 wing.

airfoil is very accurate for the station 2 results, but is less accurate for the station 5 results.

B. Aeroelastic results

1. F-5 wing undergoing forced pitching motion

This section presents the numerical results for the simulation of the aeroelastic forced pitching motion of the F-5 wing. The numerical results were compared against numerical results from Mello and Sankar [42] and experimental data from the National Aerospace Laboratory at the Netherlands [43]. Two different grids were used to

Table V. F-5 wing grid parameters.

Grid number	Nodes around the airfoil	Nodes across the O-grid	Total number of nodes
1	64	16	45,160
2	128	32	129,072
3	256	64	322,638

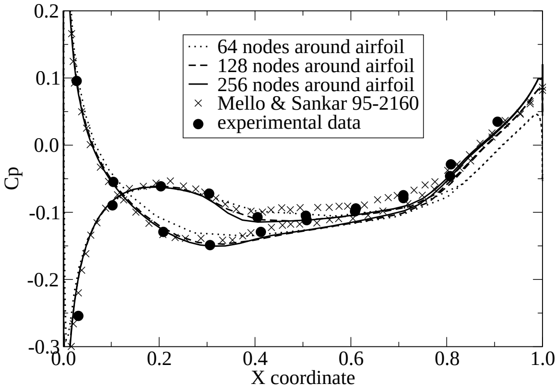


Fig. 47. Steady F-5 wing, station 2.

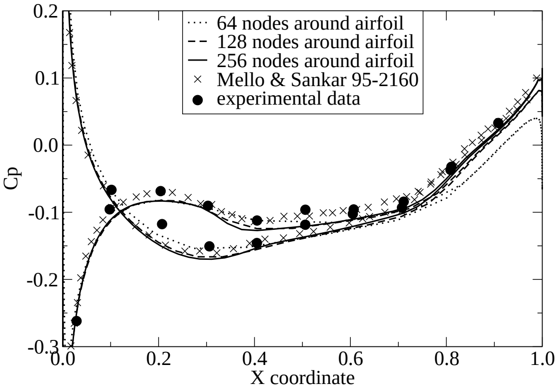


Fig. 48. Steady F-5 wing, station 5.

conduct a grid refinement test.

The F-5 wing geometry was identical to the F-5 wing used for the steady flow computations. The wing was forced to undergo a pitching motion about the elastic axis of the wing, which was located at 50 % of the wing root chord. The pitching motion had a frequency of 20 Hz and an amplitude of $0.11^\circ = 0.00192$ rad. The inlet and outlet boundaries were located at 12 chords away from the center of the airfoil, and the upper and lower boundaries were located at 9 chords away from the airfoil. Two different grids were used for this numerical simulation, with 64 and 128 nodes around the airfoil.

The flow conditions were Mach number = 0.6, static pressure $P_0 = 100{,}000$ Pa, and static temperature $T_0 = 288.15°$ K. The boundary conditions were set similarly to the boundary conditions specified in section 4.1.2.

The unsteady aeroelastic motion was periodic, therefore the solution was expected to be periodic too. The first cycles of the solution contained transient components, so the solution differed from cycle to cycle. After the transient stage was passed, the solution became periodic, with the solution of one cycle being almost identical to the solution of the following cycle. It took the solver 8 cycles to achieve a periodic solution without transient components. Solving for 8 cycles of motion meant solving for 0.4 seconds of real time, at a forcing frequency of 20 Hz. Figure 49 shows the pitching angle as a function of time.

Figures 50 and 51 show the pressure coefficient c_p results for the two different meshes, compared against numerical results from Mello and Sankar [42] and experimental data from the National Aerospace Laboratory at the Netherlands [43]. These results were taken at section 2 and section 5 of the wing. In these figures, both the 64 and 128 node solutions achieve very good results for the real components of the c_p, and the solution is shown to be grid independent, since both the 64 and 128 node solutions almost overlap each other. The accuracy drops for the imaginary components of the c_p, where the 128 node solution is relatively more accurate than the 64 node solution.

2. Aeroelastic results for the NATA wing

The Nonlinear Aeroelastic Test Apparatus (NATA) was developed at Texas A&M University to experimentally test linear and nonlinear aeroelastic phenomena [44]. The NATA setup uses a NACA 0015 wing section with a wing mount that allows for pitching and plunging. Figure 52 shows a schematic view of a two-dimensional

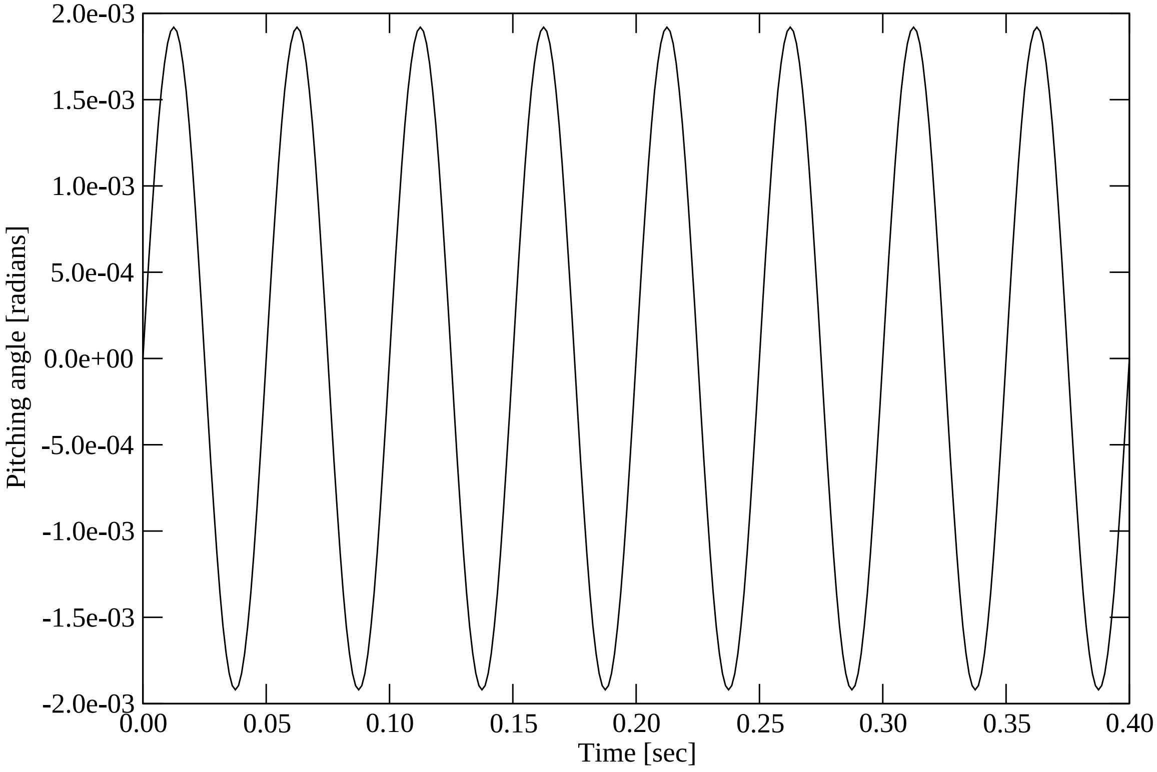


Fig. 49. Pitching angle as a function of time.

aeroelastic wing model with two degrees of freedom.

Each degree of freedom of the NATA wing was supported by its own set of springs. Plunge motion, which mimics the out-of-plane bending motion of the wing, was provided by mounting the wing on a sliding carriage. The wing was attached to a shaft; the shaft was mounted to the plunge carriage via rotational bearings. These bearings allowed the wing to pitch (rotate), simulating the torsional response of the wing. The type of response, linear or nonlinear, depended on the shape of the cam. For the results shown here, the plunge spring was linear and the pitch spring was nonlinear. Figure 53 shows a schematic view of the NATA wing setup.

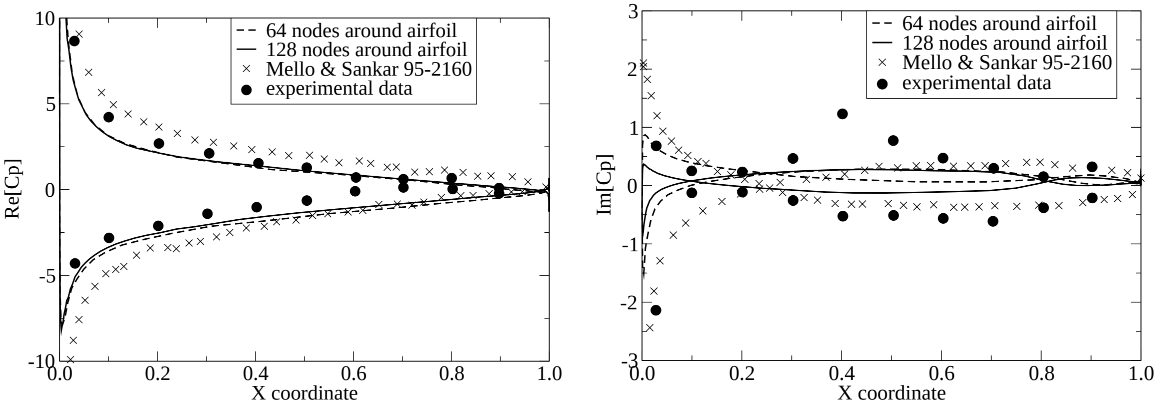


Fig. 50. Unsteady F-5 at Mach = 0.6 and frequency = 20 Hz, real and imaginary components of pressure coefficients, station 2.

The wing section stands vertically in the wing tunnel, spanning the entire tunnel from top to bottom. Figure 54 shows an isometric view of the NATA wing setup.

The equations of motion for the NATA wing are [44]

$$[M]\left\{\begin{array}{c}\ddot{h}\\ \ddot{\alpha}\end{array}\right\}+[C]\left\{\begin{array}{c}\dot{h}\\ \dot{\alpha}\end{array}\right\}+[K]\left\{\begin{array}{c}h\\ \alpha\end{array}\right\}=\left\{\begin{array}{c}-L-F_c\\ M_a-M_c\end{array}\right\} \tag{4.6}$$

where h is the plunge degree of freedom, α is the pitch degree of freedom, $[M]$ is the mass matrix, $[C]$ is the damping matrix, $[K]$ is the stiffness matrix, L is the lift on the airfoil, M_a is the aerodynamic moment, F_c is the Coulomb force, and M_c is the Coulomb moment. The matrices $[M]$, $[C]$ and $[K]$ are expressed as [44]

$$M=\left[\begin{array}{cc} m_T & m_w r_{cg} cos\alpha - m_c r_c sin(\alpha)\\ m_w r_{cg} cos\alpha - m_c r_c sin(\alpha) & I_\alpha \end{array}\right] \tag{4.7}$$

$$C=\left[\begin{array}{cc} c_h & -\left(m_w r_{cg} sin\alpha + m_c r_c cos(\alpha)\right)\dot{\alpha}\\ 0 & c_\alpha \end{array}\right] \tag{4.8}$$

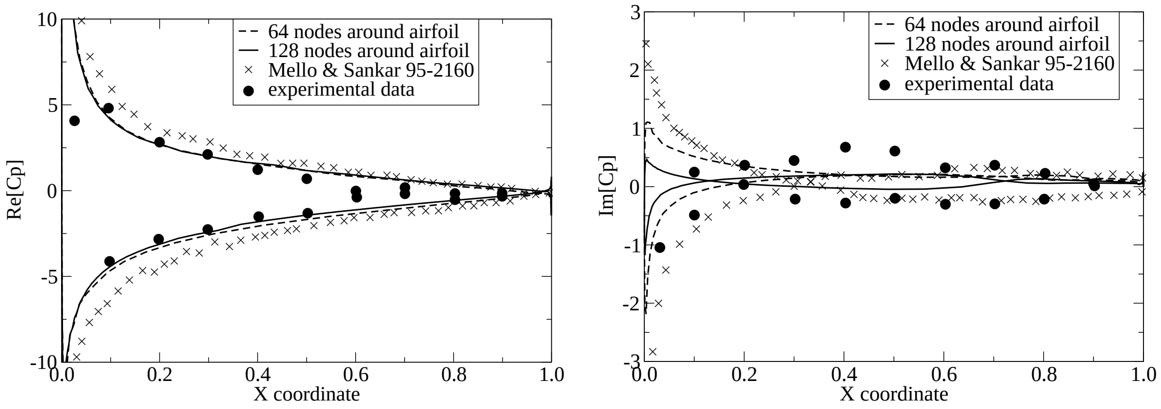


Fig. 51. Unsteady F-5 at Mach = 0.6 and frequency = 20 Hz, real and imaginary components of pressure coefficients, station 5.

$$K = \begin{bmatrix} k_h & 0 \\ 0 & k_\alpha(\alpha) \end{bmatrix} \tag{4.9}$$

The nonlinear pitch stiffness $k_\alpha(\alpha)$ is a function of the pitch angle α. The expression for $k_\alpha(\alpha)$ is [44]

$$k_\alpha(\alpha) = 8.6031 - 27.67\alpha + 867.15\alpha^2 + 376.64\alpha^3 - 7294.6\alpha^4 \tag{4.10}$$

The Coulomb plunging force F_c is expressed as:

$$F_c = -g m_t \mu_h \frac{|\dot{h}|}{\dot{h}} \tag{4.11}$$

The Coulomb pitching moment M_c is expressed as:

$$M_c = -g\left(m_w r_{cg}^2 + m_c r_c^2\right)\mu_\alpha \frac{|\dot{\alpha}|}{\dot{\alpha}} \tag{4.12}$$

The structural parameters corresponding to the NATA wing were presented in Ref. [44].

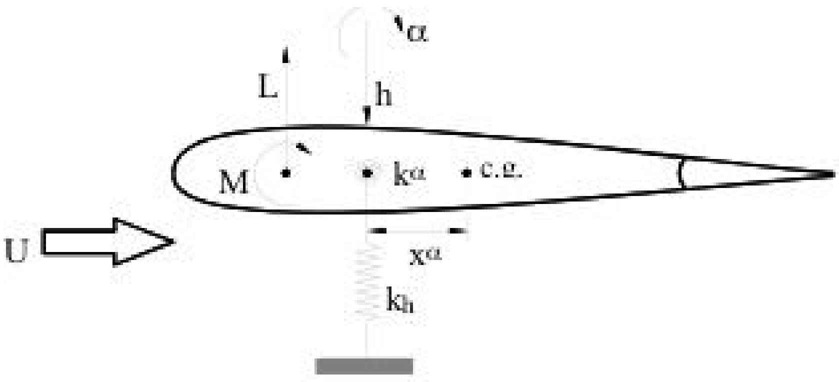


Fig. 52. Two-dimensional aeroelastic wing model with two degrees of freedom [44].

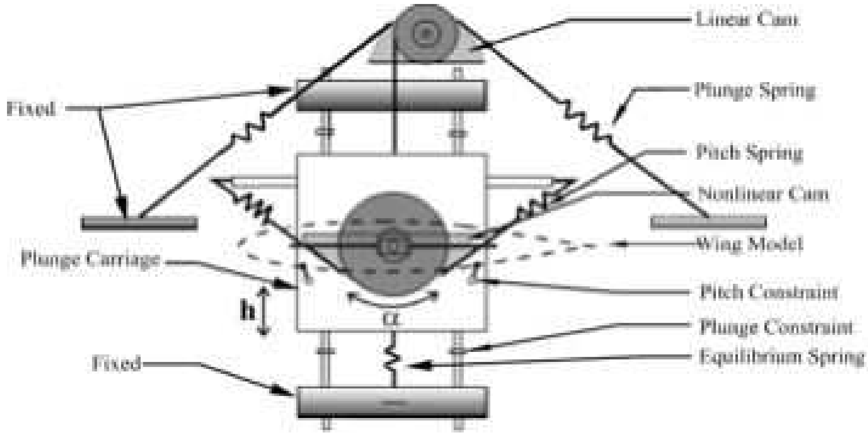


Fig. 53. Schematic view of the NATA wing setup [44].

The equations of motion for the NATA wing, Eq. (4.6) were solved in state space using the state variable X [45]:

$$X = \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} h \\ \alpha \\ \dot{h} \\ \dot{\alpha} \end{Bmatrix} \tag{4.13}$$

The transformed equations of motion become

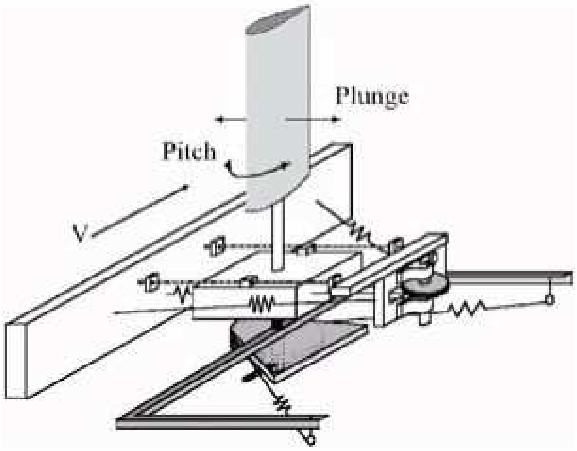


Fig. 54. Isometric view of the NATA wing setup [44].

$$\dot{X} = f(x) \tag{4.14}$$

where

$$f(x) = \left\{ \begin{array}{c} \dot{h} \\ \dot{\alpha} \\ [M]^{-1} \cdot \left[-[C] \left\{ \begin{array}{c} \dot{h} \\ \dot{\alpha} \end{array} \right\} - [K] \left\{ \begin{array}{c} h \\ \alpha \end{array} \right\} + \left\{ \begin{array}{c} -L - F_c \\ M_a - M_c \end{array} \right\} \right] \end{array} \right\} \tag{4.15}$$

Equation (4.14) was solved by using a four-stage Runge-Kutta time integration algorithm.

The analytical estimation of the system's linear flutter velocity was $V_F = 9.5$ m/sec [44]. Eight different cases were tested, with freestream velocities ranging from 4.75 m/sec (0.5 V_F) to 14.25 m/sec (1.5 V_F), corresponding to Mach numbers ranging from 0.01396 to 0.04187 at atmospheric conditions (static pressure $P_0 = 101{,}325$ Pa, static temperature $T_0 = 288.15$ K). The boundary conditions were set similarly to the boundary conditions specified in section 4.1.2.

The flow solver and the structural solver were loosely coupled. The flow solution was computed for a given wing position. The aerodynamic loads (lift and moment) were subsequently used by the structural solver to update the position of the wing, using Eq. (4.14). After the new position of the wing was calculated, the grid deformation algorithm deformed the grid, and the flow solver calculated the new aerodynamic loads.

Figure 55 shows the pitch oscillation amplitude for different velocities. The cases with $V = 0.5V_F$ and $V = 0.9V_F$ were stable, and the amplitudes reduced to zero. The cases with $V \geq V_F$ had a limit cycle oscillation behavior, and the amplitudes of the cycles increased with the velocity. The results were compared with experimental data available in Ref. [44]. The numerical results show a good match of the flutter velocity, with $V = 0.9V_F$ being a stable case and $V = V_F$ being an unstable case. The limit cycle oscillation amplitudes compare relatively well with the experimental data, specially in the $1.1V_F$ to $1.4V_F$ range.

3. Aeroelastic results for the Goland wing

a. Goland wing at low Mach number

The modal amplitudes of the Heavy Goland Wing [1] were compared against results obtained using a quasi-steady aerodynamic model [46]. This quasi-steady aerodynamic model had already been integrated with the structural solver and results were presented in [1].

A freestream velocity of 100 ft/sec was used, in order to satisfy the assumptions of the quasi-steady model. The corresponding Mach number was 0.09 at atmospheric conditions, (static pressure $P_0 = 101{,}325$ Pa, static temperature $T_0 = 288.15$ K). The

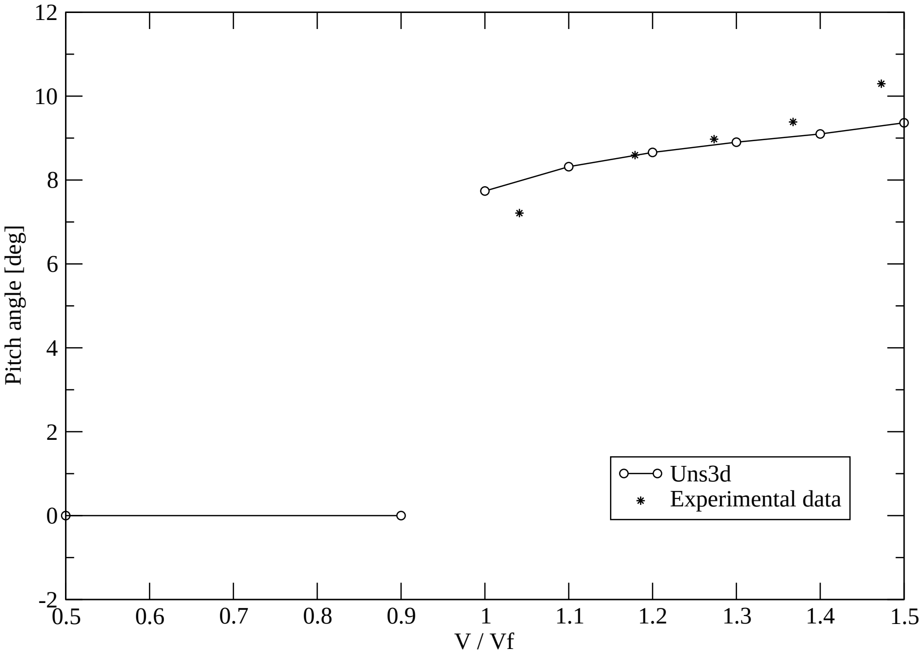


Fig. 55. Pitch amplitudes for NATA wing with Coulomb damping.

boundary conditions were set similarly to the boundary conditions specified in section 4.1.2.

A converged solution was obtained first for the undeformed wing configuration at the flight conditions described above. Then the wing was perturbed by giving initial nonzero values to the modal coordinates w, v and ϕ. w is the in-plane bending, v is the out-of-plane bending, and ϕ is the torsion modal coordinate. The initial perturbation was

$$\begin{Bmatrix} w \\ v \\ \phi \end{Bmatrix}_{initial} = \begin{Bmatrix} 0.0 \\ 0.05 \\ 0.03 \end{Bmatrix} \tag{4.16}$$

The flow solver and the structural solver were loosely coupled. After the flow solution was computed for a given wing position, the aerodynamic loads were used by the structural solver to update the position of the wing. For this new position of the wing, the grid deformation algorithm deformed the grid, and the flow solver calculated the new aerodynamic loads.

Figure 56 shows the time history of the first generalized modal deformations. The results show a good correlation between the Uns3d and quasi-steady flow solvers, for both the out-of-plane and torsional deformations.

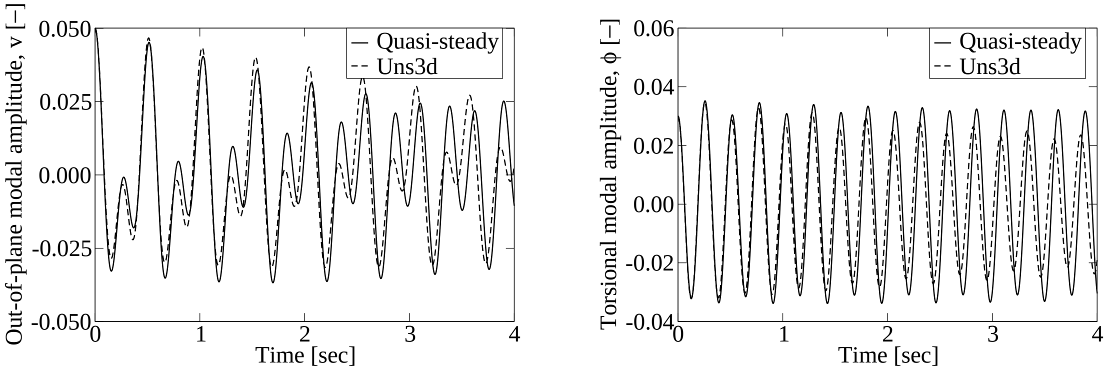


Fig. 56. First generalized modal deformations for Heavy Goland Wing at Mach = 0.09.

b. Impact of nonlinear structural model

The importance of properly capturing the nonlinearities associated with the structural model was assessed in this section by comparing the results obtained using a linear and a nonlinear structural model.

The linear structural solver was a subset of the nonlinear structural solver. The equations of motion for the linear structural solver were obtained from the equations of motion for the nonlinear structural solver, by eliminating the contributions from the nonlinear terms G'_w, G'_v and G_ϕ in Eq. (2.19).

The wing used for this simulation was the Heavy Goland Wing. The static pressure was $P_0 = 101{,}325$ Pa, the static temperature was $T_0 = 288.15$ K, and the Mach number was 0.7. The boundary conditions were set similarly to the boundary conditions specified in section 4.1.2.

A converged solution was first obtained for the undeformed wing configuration at the flight conditions described above. Then the wing was perturbed by giving initial nonzero values to the modal coordinates. The initial perturbation was identical to that set in Eq. (4.16).

At this particular flight condition, the wing exhibited Limit Cycle Oscillation (LCO) behavior. After the transient stage, the wing settled into a periodic oscillatory motion, with a definite amplitude and frequency of motion for each of the three degrees of freedom of the wing.

Figure 57 shows the out-of-plane bending and torsional modal amplitudes for the Heavy Goland Wing at Mach = 0.7. Modal amplitudes were smaller when the nonlinear terms were taken into account. This could indicate that using a linear structural model is a conservative approach, that is, the true (nonlinear) deformation is smaller than the deformation predicted by the linear model. This conclusion was invalidated, however, by results that modeled in-plane deformation, shown in Fig. 58. This was due to the fact that while using a linear structural model, there was no coupling between the modes. The in-plane deformation was positive and oscillated about a non-zero equilibrium. When non-linear terms were taken into account, the deformation was approximately one order of magnitude higher than the deformation of the linear structural model. In addition, the in-plane deformation changed direction.

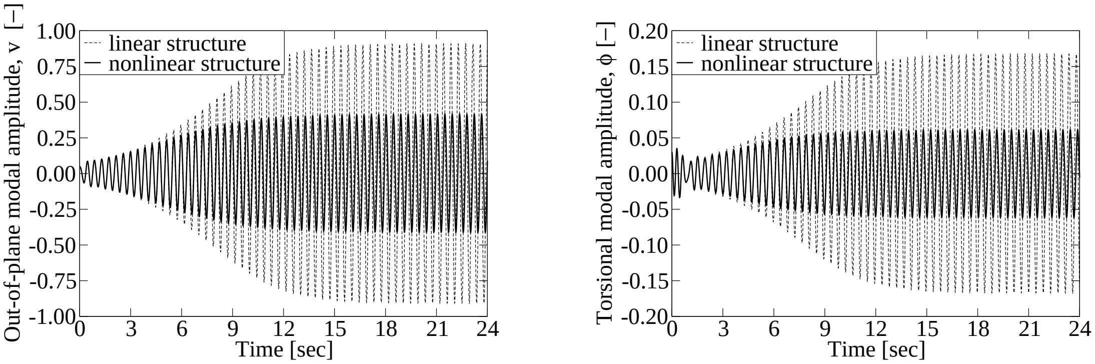


Fig. 57. Linear versus non-linear structural model: Out-of-plane and torsional modal amplitudes.

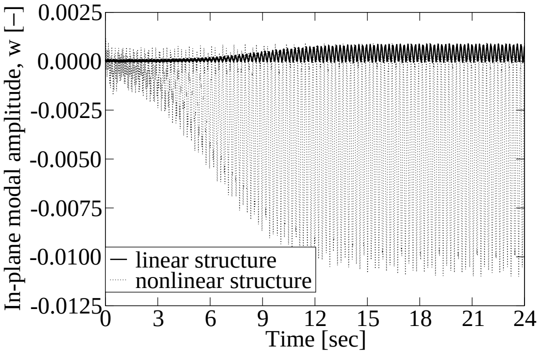


Fig. 58. Linear versus non-linear structural model: In-plane modal amplitudes.

This linear versus nonlinear structural model analysis stresses the importance of properly accounting for the nonlinearities of the system. As shown in Figs. 57 and 58, the response of the numerical simulation can change significantly if the nonlinear terms are not properly accounted for.

c. Stability boundary of Goland wing

This section presents the results obtained for the calculation of the stability boundary of the Goland wing. The stability boundary is the velocity at which the system goes from being stable to being unstable. Any disturbance at a velocity smaller than the stability boundary will be damped due to the internal damping of the structure. On the other hand, disturbances at a velocity larger than the stability boundary will increase the aerodynamic loads on the wing and cause a periodical, cyclic motion. Once the transient stage was passed, the periodical motion settled into an LCO with a definite amplitude and frequency for each modal coordinate. The frequencies were approximately independent of the velocity, but the amplitudes of the LCO were highly correlated with the velocity. As the velocity increased, the transient stage was shorter, and the LCO motion was established in a shorter time.

The stability boundary of limit-cycle oscillation for the Original Goland Wing was calculated by computing the time response of the system for different velocities and standard atmospheric conditions. As the velocity increased, the system became unstable and the disturbances were amplified. It was found that at a velocity of 332 ft/sec the system was stable, whereas at a velocity of 338 ft/sec the system became unstable.

The stable cases, with velocities of 300, 325 and 332 ft/sec, showed a damping of the deformations. This damping decreased as the velocity increased. On the other hand, velocities of 338 and 350 ft/sec showed an LCO type of motion, where the amplitudes grew in time up to the LCO amplitude. This LCO amplitude was larger as the velocity increased.

Figures 59 and 60 show the modal amplitudes for out-of-plane and torsional deformations at velocities ranging from 300 to 350 ft/sec. The plots on the left correspond to the stable cases with velocities less than 332 ft/sec. The unstable case corresponding to a velocity of 338 ft/sec was added to this plot to show the difference between the stable and unstable cases. The plots on the right of Figs. 59 and 60 show the two unstable cases with velocities of 338 ft/sec and 350 ft/sec. They illustrate how much larger the amplitudes are for the larger velocity case, as well as how much sooner the transient stage ends as the velocity increases.

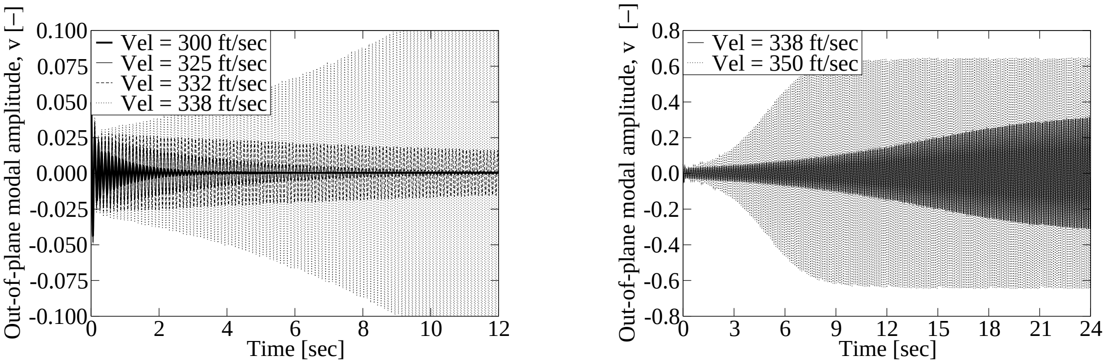


Fig. 59. Modal amplitudes of out-of-plane bending for Original Goland Wing at velocities ranging from 300 ft/sec to 350 ft/sec.

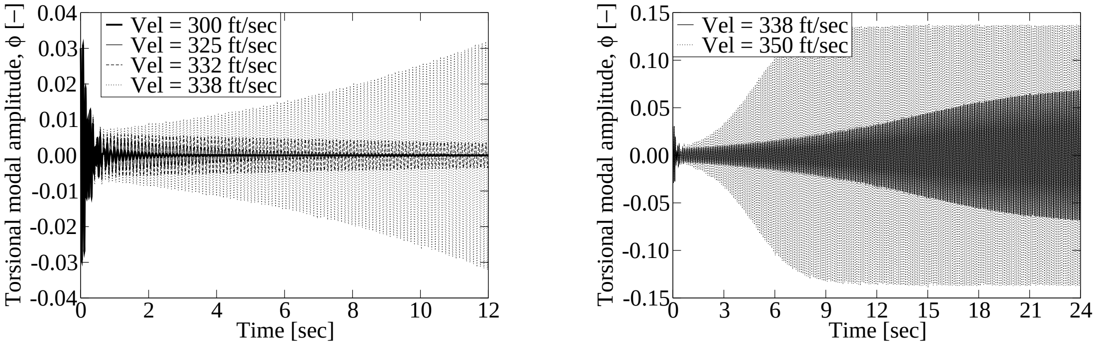


Fig. 60. Torsional modal amplitudes for Original Goland Wing at velocities ranging from 300 ft/sec to 350 ft/sec.

CHAPTER V

CONCLUSIONS AND FUTURE WORK

A. Conclusions

Fully nonlinear aeroelastic analysis presents two major challenges: the high computational cost due to the flow solvers, and the presence of highly deforming structures that challenge the mesh deformation algorithms.

This dissertation presented a mesh deformation algorithm that was both robust and computationally efficient. The mesh deformation algorithm was used for aeroelastic applications with highly deforming high-aspect-ratio wings. The results showed the robustness of this technique for highly flexible wings with deformations of up to 60 % of the wing semi-span.

The coupled flow-structure solver was parallelized to reduce the computational time. The message-passing interface (MPI) standard libraries were used for the interprocessor communication. The computational domain was divided into topologically identical layers that spanned from the root to past the tip of the wing. Dividing the computational domain into topologically identical layers increased the parallelization efficiency since it reduced the communication effort.

To further reduce the computational time, a multigrid algorithm was implemented in the flow solver. The multigrid method consisted of solving the governing equations on a series of coarser meshes, in order to obtain a converged solution in a reduced number of iterations. The multigrid algorithm was used to compute the flow about a NACA 0012 airfoil with a three-level multigrid mesh. The flow solution was computed with both the one-level grid solver and the multigrid solver. The multigrid solver required fewer iterations than the one-level grid solver to reduce the residuals to a certain value. For inviscid flows, the reduction in the number of iterations to achieve a residual value of 10^{-9} corresponded to a factor of 1.663. For turbulent flows, the reduction in the number of iterations to achieve a residual value of 10^{-7} corresponded to a factor of 2.429. The accuracy of the flow solver was not affected by the multigrid scheme, since the residuals of the fine mesh determined the overall residual level of the multigrid scheme. The multigrid method proved to be an efficient tool to accelerate the convergence of a flow solver without compromising the accuracy of the solution.

Results were presented for the validation and verification of both the flow solver and the aeroelastic solver. The flow solver was validated using: (1) the flow over a flat plate, to validate the implementation of the turbulence model, and (2) the flow over the NACA 0012 airfoil and over the F-5 wing, to validate the implementation of the convective and viscous fluxes, the time integration algorithm, and the boundary conditions. The numerical results for the NACA airfoil were compared against experimental results. Four different grids were used in this simulation to verify that the solution was grid independent. The numerical results for the F-5 wing were compared against experimental results and other numerical results.

The aeroelastic solver was validated using: (1) the unsteady F-5 wing undergoing forced pitch motion, and (2) the Nonlinear Aeroelastic Test Apparatus (NATA) wing. Both F-5 wing and NATA wing results were compared against experimental results. In addition, aeroelastic results were generated for the Goland wing, including a comparison against the quasi-steady aerodynamic model for low Mach numbers, a study of the influence of the nonlinear structural terms in the LCO characteristics, and the calculation of the stability boundary for the Goland wing.

The aeroelastic solver developed herein allowed the analysis of aeroelastic phenomena using a fully nonlinear approach. Limit cycle oscillations, which are highly nonlinear phenomena, were captured by the nonlinearities of the flow solver and the structural solver. The impact of the nonlinearities was assessed for the Goland wing, where nonlinear terms changed dramatically the aeroelastic behavior of the wing.

B. Future work

As a continuation of this effort, future work should be directed towards further reducing the computational cost associated with the flow solver. The most effective path to take would be to implement an implicit time integration scheme. These implicit schemes are more difficult to implement than the explicit schemes, and are computationally more costly per iteration, but they have the advantage of being unconditionally stable. While the explicit schemes have a limit on the advancing time step due to stability issues, the implicit schemes lack this limitation, thus they can advance at much larger time steps, resulting in great overall savings in computational time.

Additional structural models should be implemented in this aeroelastic application in order to better represent the wing structure. More accurate representations of the wing could be obtained by using plate and shell structural models. These models could be solved with the finite element method using plate and shell elements.

REFERENCES

[1] Beran, P. S., Strganac, T. W., Kim, K., and Nichkawde, C., “Store Induced Limit Cycle Oscillations Using a Model with Full System Nonlinearities,” *Nonlinear Dynamics*, Vol. 37, No. 4, September 2004, pp. 323–339.

[2] Dowell, E. H., Crawley, E. F., Curtiss, H. C., Peters, D. A., Scanlan, R. H., and Sisto, F., *A Modern Course in Aeroelasticity*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.

[3] Strganac, T. W., Cizmas, P. G., Nichkawde, C., Gargoloff, J., and Beran, P. S., “Aeroelastic Analysis for Future Air Vehicle Concepts Using a Fully Nonlinear Methodology,” AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, AIAA Paper 2005-2171, Austin, Texas, April 2005.

[4] Denegri, Jr., C. M., “Limit Cycle Oscillation Flight Test Results of a Fighter with External Stores,” *Journal of Aircraft*, Vol. 37, No. 5, September-October 2000, pp. 761–769.

[5] Bunton, R. W. and Denegri, Jr., C. M., “Limit Cycle Oscillation Characteristics of Fighter Aircraft,” *Journal of Aircraft*, Vol. 37, No. 5, 2000, pp. 916–918.

[6] Jacobson, S. B., Britt, R. T., Dreim, D. R., and Kelly, P. D., “Residual Pitch Oscillation (RPO) Flight Test and Analysis on the B-2 Bomber,” 39th AIAA Structures, Structural Dynamics, and Materials Conference, AIAA Paper 98-1805, Long Beach, California, April 1998.

[7] Tang, D., Dowell, E. H., and Hall, K. C., “Limit Cycle Oscillations of a Cantilevered Wing in Low Subsonic Flow,” *AIAA Journal*, Vol. 37, No. 3, March 1999, pp. 364–371.

[8] Eastep, F. E. and Olsen, J. J., "Transonic Flutter Analysis of a Rectangular Wing with Conventional Airfoil Sections," *AIAA Journal*, Vol. 18, No. 10, October 1980, pp. 1159–1164.

[9] Batina, J. T., "Efficient Algorithm for Solution of the Unsteady Transonic Small-Disturbance Equation," *Journal of Aircraft*, Vol. 25, No. 7, July 1988, pp. 598–605.

[10] Howlett, J. T., "Efficient Self-Consistent Viscous-Inviscid Solution for Unsteady Transonic Flow," *Journal of Aircraft*, Vol. 24, No. 11, November 1987, pp. 737–744.

[11] Geuzaine, P., Brown, G., Harris, C., and Farhat, C., "Aeroelastic Dynamic Analysis of a Full F-16 Configuration for Various Flight Conditions," *AIAA Journal*, Vol. 41, No. 3, March 2003, pp. 363–371.

[12] Schuster, D. M., Vadyak, J., and Atta, E. H., "Flight Loads Prediction Methods for Aircraft Euler/Navier-Stokes Aeroelastic Method (ENS3DAE), Version 1," Tech. Rep. WRDC-TR-89-3104, Wright Research & Development Center, Wright-Patterson Air Force Base, Ohio, 1989.

[13] Krist, S. L., Biedron, R. T., and Rumsey, C. L., "CFL3D User's Manual (Version 5.0)," Tech. Rep. NASA-TM-1998-208444, NASA, Hampton, Virginia, June 1998.

[14] Hoke, C. and Schwabacher, G., "Aerodynamic Analysis of Complex Missile Configurations Using AVUS (Air Vehicles Unstructured Solver)," Applied Aerodynamics Conference, AIAA Paper 2004-5452, Providence, Rhode Island, August 2004.

[15] Vadyak, J., Smith, M. J., Schuster, D. M., and Weed, R., "Simulation of External Flowfields Using a 3-D Euler/Navier-Stokes Algorithm," AIAA 25th Aerospace Sciences Meeting, AIAA Paper 1987-0484, Reno, Nevada, January 1987.

[16] Schuster, D. M., Vadyak, J., and Atta, E., "Static Aeroelastic Analysis of Fighter Aircraft Using a Three-Dimensional Navier-Stokes Algorithm," *Journal of Aircraft*, Vol. 27, No. 9, September 1990, pp. 820–825.

[17] Menter, F. R., "Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications," *AIAA Journal*, Vol. 32, No. 8, August 1994, pp. 1598–1605.

[18] Han, Z. and Cizmas, P. G. A., "Prediction of Axial Thrust Load in Centrifugal Compressors," *International Journal of Turbo & Jet-Engines*, Vol. 20, No. 1, January 2003, pp. 1–16.

[19] Blazek, J., *Computational Fluid Dynamics: Principles and Applications*, Elsevier Science Ltd., Kidlington, Oxford, United Kingdom, 2001.

[20] da Silva, M. R. M. C. and Glynn, C. C., "Nonlinear Flexural-flexural-torsional Dynamics of Inextensional Beams - Equations of Motion," *Journal of Structural Mechanics*, Vol. 6, No. 9, September 1978, pp. 437–448.

[21] Smith, M. J., Schuster, D. M., Huttsell, L., and Buxton, B., "Development of an Euler/Navier-Stokes Aeroelastic Method for Three-dimensional Vehicles with Multiple Flexible Surfaces," AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, AIAA Paper 1996-1513, Salt Lake City, Utah, April 1996.

[22] Kim, K. and Cizmas, P. G. A., "Three-Dimensional Hybrid Mesh Generation for Turbomachinery Airfoils," *AIAA Journal of Propulsion and Power*, Vol. 18,

No. 3, May-June 2002, pp. 536–543.

[23] Cizmas, P. G. A., "Axial Thrust Load Prediction," Tech. Rep. TRC-PERF-3-00, Turbomachinery Laboratory at Texas A&M University, College Station, Texas, 2000.

[24] Barth, T. J., "A 3-D Upwind Euler Solver for Unstructured Meshes," AIAA Paper 91-1548, 1991.

[25] Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, No. 2, 1981, pp. 357–372.

[26] Blazek, J., Irmisch, S., and Haselbacher, A., "Unstructured Mixed-grid Navier-Stokes Solver for Turbomachinery Applications," 37th Aerospace Sciences Conference, AIAA Paper 99-0664, 1999.

[27] Haselbacher, A. and Blazek, J., "Accurate and Efficient Discretization of Navier-Stokes Equations on Mixed Grids," *AIAA Journal*, Vol. 38, No. 11, November 2000, pp. 2094–2102.

[28] Aftosmis, M., Gaitonde, D., and Tavares, T. S., "Behavior of Linear Reconstruction Techniques on Unstructured Meshes," *AIAA Journal*, Vol. 33, No. 11, November 1995, pp. 2038–2049.

[29] Barth, T. J. and Jesperson, D. C., "The Design and Application of Upwind Schemes on Unstructured Meshes," 27th Aerospace Sciences Conference, AIAA Paper 89-0366, 1989.

[30] Dervieux, A., "Steady Euler Simulations Using Unstructured Meshes," *Computational Fluid Dynamics, VKI Lecture Series 1985-04*, Vol. 1, March 1985.

[31] Thomas, P. D. and Lombard, C. K., “Geometric Conservation Law and Its Application to Flow Computations on Moving Grids,” *AIAA Journal*, Vol. 17, No. 10, January 1979, pp. 1030–1037.

[32] Frink, N. T., “Upwind Scheme for Solving the Euler Equations on Unstructured Tetrahedral Meshes,” *AIAA Journal*, Vol. 30, No. 1, January 1992, pp. 70–77.

[33] Whitaker, D. L., “Three-Dimensional Unstructured Grid Euler Computations Using a Fully-Implicit, Upwind Method,” AIAA Paper 93-3337, 1993.

[34] Batina, J. T., “Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes,” *AIAA Journal*, Vol. 28, No. 8, August 1990, pp. 1381–1388.

[35] Persoon, A. J., Roos, R., and Schippers, P., “Transonic and Low Supersonic Wind-tunnel Tests on a Wing with Inboard Control Surface,” Tech. Rep. AFWAL-TR-80-3146, National Aerospace Laboratory, Amsterdam, The Netherlands, December 1980.

[36] Jameson, A., “Solution of the Euler Equations for Two Dimensional Transonic Flow by a Multigrid Method,” *Applied Mathematics and Computation*, Vol. 13, 1983, pp. 327–355.

[37] Briggs, W. L., *A Multigrid Tutorial*, Society for Industrial and Applied Mathematics, Lancaster, Pennsylvania, 1987.

[38] Holmes, D. G. and Connell, S. D., “Solution of the 2D Navier-Stokes Equations on Unstructured Adaptive Grids,” AIAA Paper 89-1932, 1989.

[39] Connell, S. D. and Holmes, D. G., “A 3D Unstructured Adaptive Multigrid Scheme for the Euler Equations,” AIAA Paper 93-3339, 1993.

[40] Cizmas, P. G. A. and Palacios, A., "Proper Orthogonal Decomposition of Turbine Rotor-Stator Interaction," *Journal of Propulsion and Power*, Vol. 19, No. 2, March-April 2003, pp. 268–281.

[41] Tennekes, H. and Lumley, J. L., *A First Course in Turbulence*, The Massachusetts Institute of Technology, MIT Press, Cambridge, Massachusetts, 1972.

[42] Mello, O. A. F. and Sankar, L. N., "An Improved Hybrid Navier-Stokes/Potential Method for Computation of Unsteady Compressible Flows," 26th AIAA Fluid Dynamics Conference, AIAA Paper 95-2160, San Diego, California, June 1995.

[43] Tijdeman, M., van Nunen, J. W. G., andA. J. Persoon, A. N. K., Poestkoke, R., Roos, R., Schippers, P., and Siebert, C. M., "Transonic Wind Tunnel Tests on an Oscillating Wing with External Stores. Part II. The Clean Wing," Tech. Rep. AFFDL-TR-78-194, National Aerospace Laboratory, Amsterdam, The Netherlands, 1978.

[44] Hill, W. J., Strganac, T. W., Nichkawde, C., McFarland, D. M., Kerschen, G., Lee, Y. S., Vakakis, A. F., and Bergman, L. A., "Suppression of Aeroelastic Instability with a Nonlinear Energy Sink: Experimental Results," 47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, AIAA Paper 2006-1848, Newport, Rhode Island, May 2006.

[45] Ko, J., Kurdila, A. J., and Strganac, T. W., "Nonlinear Control of a Prototypical Wing Section with Torsional Nonlinearity," *Journal of Guidance, Control, and Dynamics*, Vol. 20, No. 6, November-December 1997, pp. 1181–1189.

[46] Kim, K. and Strganac, T. W., "Aeroelastic Studies of a Cantilever Wing with Structural and Aerodynamic Nonlinearities," AIAA/ASME/ASCE/AHS/ASC

Structures, Structural Dynamics and Materials Conference, AIAA Paper 2002-1412, Denver, Colorado, April 2002.

VITA

Joaquin Ivan Gargoloff was born in Dolores, province of Buenos Aires, Argentina on the 28th of January of 1978. He graduated from Colegio Nacional High School in 1990. He then moved to La Plata, where he graduated from La Plata National University in 1996 with a Bachelor of Science degree in Aerospace Engineering. In 2002 he moved to College Station, Texas, to attend Texas A&M University, receiving a Ph.D. degree in Aerospace Engineering in May 2007.

His permanent address is Texas A&M University, Department of Aerospace Engineering, H.R. Bright Building, Ross Street, TAMU 3141, College Station, Texas, TX 77843-3141. His email address is gargoloff@tamu.edu.