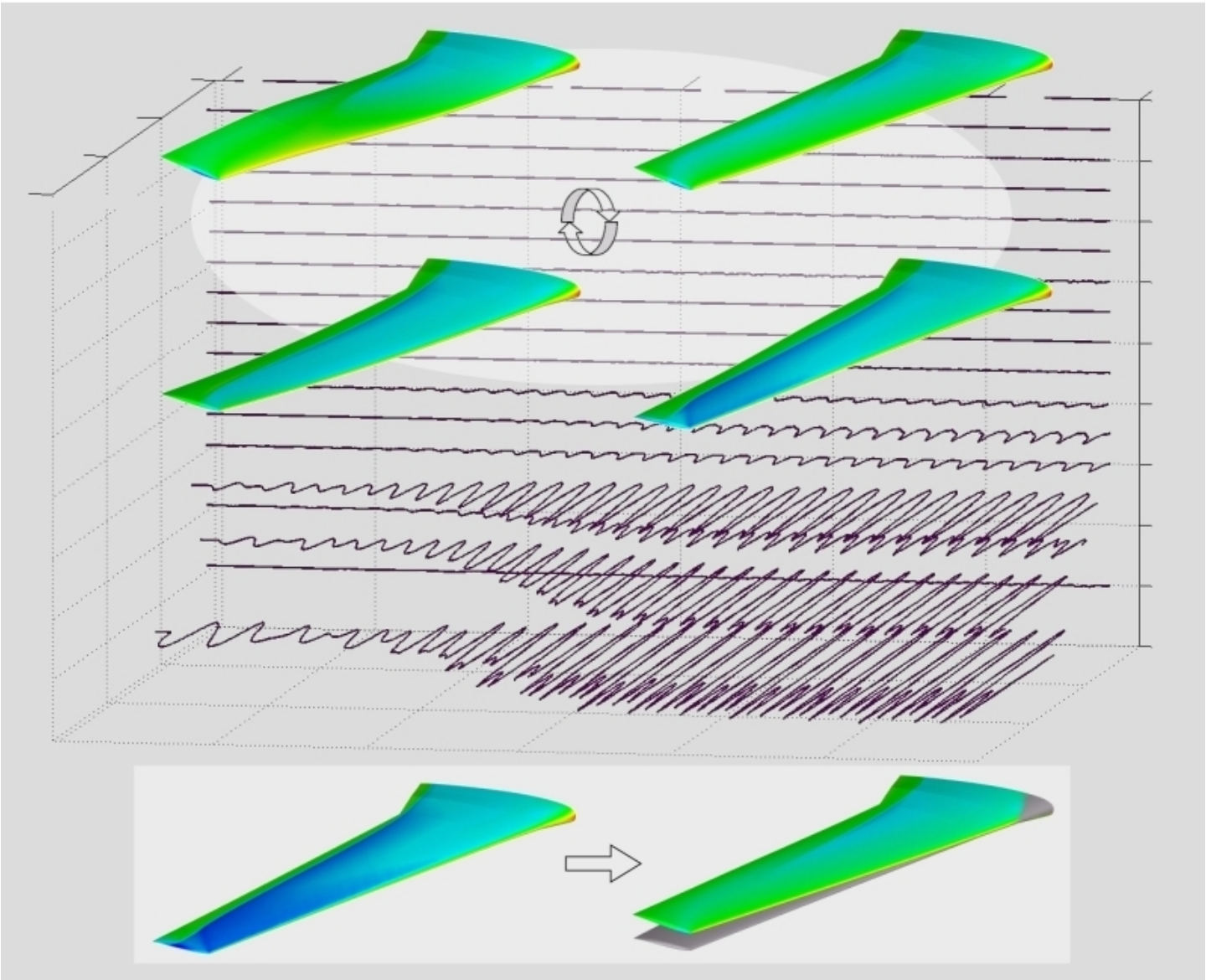
Aeroelastic Functionality in Edge
Initial Implementation and Validation

JONATHAN SMITH



FOI-R--1485--SE Scientific report
ISSN-1650-1942 December 2005

Systems Technology

Jonathan Smith

Aeroelastic Functionality in Edge
Initial Implementation and Validation

FOI-R--1485--SE Scientific report
ISSN-1650-1942 December 2005

Systems Technology

Issuing organisation FOI – Swedish Defence Research Agency Systems Technology SE-164 90 STOCKHOLM	**Report number, ISRN** FOI-R--1485--SE	**Report type** Scientific report
	Research area code Mobility and Space Technology, incl Materials	
	Month year December 2005	**Project no.** E830057
	Sub area code	
	Sub area code 2	
Author/s (editor/s) Jonathan Smith	**Project manager** Peter Eliasson	
	Approved by Monica Dahlén	
	Sponsoring agency Defence Materiel Adminstration	
	Scientifically and technically responsible Bengt Winzell	

Report title
Aeroelastic Functionality in Edge Initial Implementation and Validation

Abstract
The FOI-developed CFD flow-solver Edge has been extended to include functions for Aeroelastic simulation. These new features, which are available in Edge versions 3.3.1 and higher, enable the time-accurate simulation of nonlinear Fluid Structure Interaction phenomena such transonic flutter and limit cycle oscillations (LCO). The initial implementation, which uses a modal formulation, is presented together with two validation studies. The new Edge code is evaluated against experiments with prescribed, rigid-pitch motion (the LANN-wing) and a code-to-code comparison of coupled aeroelastic simulations (the UNSI MDO-wing). It is shown that Edge produces results in good agreement both with experiment and with comparable numerical simulations. Through analysis of the MDO-wing simulations, we assess the accuracy of the time-integration of the coupled equations of motion and the influence of the spatial coupling scheme. The functionality of the new code is evaluated and recomendations are made for its continued development.

Keywords
Aeroelasticity, nonlinear, flutter, Edge, CFD, Matlab, transonic flutter, LCO, FSI, Fluid Structure Interaction, spatial coupling, mesh deformation, multiphysics, unsteady aerodynamics, Stripe, TAURUS, UNSI

Further bibliographic information	**Language** English
ISSN ISSN-1650-1942	**Pages** 66 p.
	Price According to price list

Utgivare FOI – Totalförsvarets forskningsinstitut Systemteknik 164 90 STOCKHOLM	**Rapportnummer, ISRN** FOI-R--1485--SE	**Klassificering** Vetenskaplig rapport
	Forskningsområde Farkost- och rymdteknik, inkl material	
	Månad år December 2005	**Projektnummer** E830057
	Delområde	
	Delområde 2	
Författare/redaktör Jonathan Smith	**Projektledare** Peter Eliasson	
	Godkänd av Monica Dahlén	
	Uppdragsgivare/kundbeteckning FMV	
	Tekniskt och/eller vetenskapligt ansvarig Bengt Winzell	

Rapportens titel
Aeroelastisk funktionaliteten i Edge första implementationen och validering

Sammanfattning
Det FOI-utvecklade CFD programmet Edge har uppgraderats med funktioner för aeroelastisk simulering. Den nya funktionaliteten, som finns i Edge från och med version 3.3.1, möjliggör tidsnogrann simulering av icke-linjära fenomen inom fluidstrukturinteraktion som t ex transoniskt fladder och LCO. Den första implementeringen som är baserad på en modal formulering, samt två valideringsstudier presenteras. Den nya koden utvärderas mot experiment med föreskriven stelkroppsrotation (LANN-vingen) och jämförs med annan CFD-kod för en kopplad, aeroelastisk simulering (MDO-vingen från UNSI-projektet). Resultaten från Edge visar bra överensstämmelse med experiment och jämförbara numeriska simuleringar. Genom analys av simuleringar på MDO-vingen uvärderas även noggrannheten av tidsintegrationen för det kopplade ekvationsystemet samt hur lösningen påverkas av rumskopplingsscheman. Den nya funktionaliten utvärderas och rekommendationer ges för vidareutveckling.

Nyckelord
Aeroelasticitet, icke-linjär, fladder, Edge, CFD, Matlab, transonisk fladder, LCO, FSI, fluidstrukturinteraktion, rumskoppling, nätdeformering, multifysik, instationär aerodynamik, Stripe, TAURUS, UNSI

Övriga bibliografiska uppgifter	**Språk** Engelska
ISSN ISSN-1650-1942	**Antal sidor:** 66 s.
Distribution enligt missiv	**Pris: Enligt prislista**

Contents

Acknowledgments

The implementation of aeroelastic functionality in `Edge`, especially its use of deformable meshes, has required unprecedented changes to the main infrastructure of the code and its preprocessor system. Peter Eliasson, originator and principal developer of `Edge`, has provided essential advice and assistance in this work. Bengt Winzell (FOI, formerly Saab AB) having pioneered the implementation of aeroelastic functionality in earlier CFD code Euranus, has also assisted in both coding and diagnostic work. One key contribution is the interpolation method *polyfit* in the spatial-couDefence Materiel Adminstrationpling program `aedbelast`. Credit is also due for his assistance in the two validation studies.

Computations on the LANN-wing, using both the CFD code TAU and Euranus, were performed by Anders Karlsson (Saab Aerosystems AB). The mesh-deformation tool `aetribend` is a command-line interface for components of TRITET, a mesh-generation system created by Lars Tysell. The mesh for the MDO-wing was constructed by Stephen Conway (Scania AB, formerly FOI).

The bulk of the work presented here was financed by FMV[1] both through direct funding and within the FoT-25 projects *Adaptiva Strukturer* and *Framdrivningsintegration*[2] where the code is being used. The LANN-wing validation study was also backed by EC-funding through the project TAURUS.

[1] FMV: The Swedish Defence Materiel Adminstration `www.fmv.se`

[2] "Adaptive Structures" and "Propulsion Integration"

1 Introduction

Motivation

This report presents the initial implementation and validation of aeroelastic functionality in Edge, an advanced CFD flow-solver for unstructured grids, which is developed and owned by FOI. The work described here extends the capability of Edge beyond that of a conventional flow-solver and enables the simulation of a much wider class of phenomena: Fluid-Structure Interaction. The immediate requirement for this technology is in the Swedish aircraft industry, where tools are needed for the realistic simulation and prediction of aeroelastic effects such as transonic flutter and limit cycle oscillations (LCO). There are, however, many other engineering applications, for example in the analysis of internal flows in propulsion systems. The Edge aeroelastic code is already being used by FOI in the FMV-funded FoT-25 projects *Adaptiva Strukturer* and *Framdrivningsintegration*. Similar computational technology is also being developed elsewhere for biomedical applications, such as the simulation of blood-flow in the human heart. The development of aerolastic functionality in Edge is an attempt to bring such challenging and commercially relevant applications within a nearer grasp.

The purpose of this report is to provide documented quality assurance of the present Edge aeroelastic system and a complete, functional description for the benefit of future users and program developers. The text therefore includes explicit reference to program units, variables and data files. Together with an installed copy of Edge and some example input data, this document can also be used as a practical introduction to CFD-based aeroelastic simulation.

The results presented here and the description of the code apply to Edge version 3.3.1, released in May 2005. The early development of the code is described in [19] and the present version is described in the manual [25]. Current information about Edge an its applications is also published on the FOI website. At the time of writing, a fully-parallel implementation of the aeroelastic code is being tested, however, that work is beyond the scope of this report.

Background

Aeroelasticity is the science of phenomena arising from the interaction between fluid flow and flexible structures. It is a multi-physics discipline and forms part of a wider field known as Fluid Structure Interaction (FSI) [43]. A general introduction to literature on aeroelasticty is given in the review article [6]and more detailed exposition is available in the standard texts [11, 26, 17].

Computational methods for aeroelasticity are of prime importance in the aircraft industry where detailed aeroelastic evaluation is an essential part of the design and certification processes. Other industrial applications of aeroelasticity include the design of turbomachinery, wind turbines and large, flexible land-based structures such as bridges and towers. For low-speed, subsonic flows, or for high-speed, supersonic flows, most aeroelastic problems can be adequately

treated using methods based on linearised aerodynamic theory, such as the doublet lattice model [2]. This is a mature technology and is available in several commercial software packages [44, 72]. However, in the transonic region, Mach 0.8 to 1.2, , which is within the operating range of most modern aircraft, the aerodynamics can not be adequately described by linear theory. Indeed, dynamic aeroelastic instabilities such as transonic limit cycle oscillations (LCO) can not be predicted at all by linear methods. Without adequate predictive methods for such effects, design for transonic flight is compromised by the need for large safety margins. To model and predict transonic aeroelastic phenomena requires aeroelastic methods that exploit modern CFD codes for realistic, nonlinear aerodynamics. This technology is variously known as "structure-coupled CFD" "computational aeroelastic simulation" or simply "Fluid Structure Interaction". The work described here aims to provide self-contained functionality for aeroelastic simulation within the Edge program-system.

Euranus and UNSI

The basic capability for structure-coupled CFD simulation has been available within FOI (and formerly FFA) since as early as 1998. The FFA-developed, structured multiblock flow solver, Euranus, was extended to include functions for elastic mesh deformation and structural coupling. The development work was carried out mainly by the aeroelasticity group at Saab, Linköping, where the aeroelastic Euranus code was used for detailed aeroelastic investigations on the JAS39 aircraft [67, 66, 69]. The aeroelastic functionality of Eurnaus was further developed at FFA and is incorporated into to the final, official FFA release: Euranus version 5.3. Euranus was used by both Saab and FFA in the EU-project UNSI[1] (1998-2000) *Unsteady Viscous Flow in the Context of Fluid-Structure Interaction* [32]. The present development of aeroelastic functionality in Edge closely follows that used for Euranus and the main validation study is based on a coupled simulation test-case from the UNSI-project.

Unstructured Grids and TAURUS

The aeroealstic functionality in Eurnaus is still a valuable asset, however, it is limited by two basic properities of the main CFD code. Firstly, Euranus is code for multiblock, structured grids and secondly it can not be run using large-scale, parallel computer-architectures. For complex geometries, the generation of structured grids is extremely time consuming. For military aircraft, with multiple external stores configurations, even using modern software tools, the cost becomes prohibitive. This was the primary motivation for developing unstructured CFD code Edge. The code's unstructured formulation allows rapid mesh generation for geometries of arbitrary complexity. To achieve the performance required for modern aerodynamic computations, Edge has been built "from the ground up" as a code for scalable, parallel execution.

Whilst unstructured grid technology has many advantages for the end user, it introduces significant additional complexity for aeroelastic applications. This was one motivation for FOI's participation in the EU-project TAURUS [2] (2001-2003) *Technology Development for Aeroelastic Simulations on Unstructured Grids*. The aim of this project was to develop a "common European software" for aeroelastic simulation, a system which could, in principle, be used with any time-accurate, unstructured CFD code. Unfortunately, most of this software was unavailable as source code and direct integration with Edge proved not to be feasible. However, the experience gained from FOI's participation

[1] UNSI: EC Brite/Euram Project BRPR-CT97-0583
[2] TAURUS: EC Contract G4RD-CT-2001-00403

in the TAURUS-project has helped to identify the correct procedures to be implemented in Edge. The TAURUS-project has also made possible a detailed validation of aeroelastic functions in Edge against the DLR-developed, unstructured CFD flow-solver, TAU.

Overview

The immediate aim of the work described here was to provide in Edge functionality matching that available in Euranus (since 1999). However, much of the aeroelastic functionality of Eurnaus relies on a diverse collection of programs developed independently at Saab AB and elsewhere. It was recognised that, for the envisaged applications, a more complete and flexible system was required. The following strategic objectives were identified.

- Aeroelastic functionlaity at least matching Euranus
- Complete system including Aeroelastic pre- and postprocessing tools
- Ease of use and flexibility
- Maintainable code with placeholders for future extension
- Validated for known Aeroelastic test-cases

The Euranus aeroelastic code provides functions for simulations, both prescribed motion using elastic mode shapes and with iterative coupling to an internal, modal, structural model. The movement of the CFD mesh is introduced using a "perturbation field" method[1], in which the quasi-elastic mesh deformation is performed off-line as a separate, precursory operation. As an initial step, the same method has been used for Edge but due to the code's unstructured formulation, this has required new programs for mesh-deformation and subtantial internal modifications to enable the mesh-movement. The code was delivered in two stages. Mesh movement and functions for prescribed motion were delivered in Edge version 3.2 [24] and a validation studies were carried out within the TAURUS-project [51]. Modal coupling was then introduced in Edge version 3.3 [25] and a validation study was been completed using a coupled, aeroelastic simulation test-case from the UNSI-project.

This report describes the first available aeroelastic functionality in Edge. The theoretical formulation and program implementation are presented in detail. This is followed by two validation studies. The first study refers to wind-tunnel experiment with prescribed, rigid-pitch oscillation and the second is a coupled, aeroelastic simulation for which independent numerical results are available. The report concludes with a review of the validation studies, an evaluation of the present code and recommendations for its future development.

[1] also known as "perturbation grid", or in Swedish "*störningsnät*"

2 Theory and Implementation

2.1 Aeroelastic Simulation

The standard computational methods for aeroelasticity are based on linear structural and aerodynamic models with a frequency-domain formulation. The coupled, linear system is analysed to obtain stability boundaries as functions of the aerodynamic conditions (altitude and Mach number). The approach used here is quite different. The coupled structural and aerodynamic models are solved by a "time-marching" procedure to produce an aeroelastic simulation. The result is similar to a single-point flutter flight test, in that it provides the time-domain response for one set of aerodynamic conditions. To locate a stability boundary, or even to estimate a flutter-onset speed, would require the analysis of several such simulations.

Aeroelastic simulation is implemented in Edge using a partitioned[1] method, that is one in which separate fluid (CFD) and structural (CSM) solvers are coupled via an exchange of instantaneous load and displacement data [22]. The flow equations are solved subject to the position and velocity of the moving boundary surface. In turn, the structural equation of motion is solved given the fluid loads on the moving surface. The coupled fluid-structure system is thus advanced through a series of discrete timesteps. At each timestep, the following cycle of operations is repeated.

- **CFD:** Compute the flow field, given the instantaneous surface geometry and its rate of change.
- **Load transfer:** Transform the fluid surface loads to an equivalent load distribution on the structure.
- **CSM:** Compute the structural state, given the current loads.
- **Motion transfer:** Transform the structural motion to the surface geometry.
- **Mesh deformation:** Update the deformable CFD-mesh to comply with the surface geometry.

The CFD and CSM computations are coupled through "load transfer" and "motion transfer" between the fluid and structural domains. The combination of these operations is known as "spatial coupling" since it comprises interpolation between the spatially separated grids used by the fluid and structural solvers. In a further "mesh deformation" computation, the CFD mesh is adjusted to comply with the instantaneous surface geometry. All communication between the CFD and CSM solvers is via the moving surface of contact: a boundary in the CFD-mesh which is here referred to as the "wetted surface".

[1] In contrast, a "monolithic" method, is one in which the fluid and structural domains are represented by a single system of equations. This also known as "strong" coupling.

Within this general framework, a variety of different coupling schemes can be formulated, depending on the scheduling and integration of the operations listed above. The coupling scheme introduced in Edge is an iterative procedure belonging to a set of algorithms known as "Conventional Staggered Schemes" (CSS) [21, 20, 40]. This implementation relies on the use of "dual timestepping" for time-integration of the flow equations [37].

In dual-timestepping, the fluid equation of motion is integrated by regular time-marching, with the instantaneous flow-field at each physical timestep being computed via an inner iteration loop. The inner-loop "pseudo-time" process is similar to a steady-state flow computation, including the use of multigrid for convergence acceleration. In a CSS procedure, the inner loop also provides the coupling point for the structural solver.

The CSS coupling procedure implemented in Edge may be represented as follows. For clarity, the names of subroutines and variables have been changed from those used in the source-code.

```
while in_time_range   ! TIME LOOP

   time = time + delta_t

   while not_converged    ! INNER LOOP

      multigrid_cycles    ! update the flow field

      structural_force    !  "  structural loads

      structural_state    !  "  equation of motion

      mesh_movement       !  "  deformable CFD mesh
```

The structural equation of motion and the full coupling sequence is iterated within the inner loop. This is to ensure that the fluid and structural fields are converged at each real timestep, thus achieving *synchronous* coupling[1] which is an essential prerequisite for time-accuracy. If, instead, the structural state was updated only in the outer loop, then the fluid and structural domains would be separated by a time-lag, `delta_t` and synchronisation could only be acheived in the limiting case. Using the scheme shown above, provided that the fluid and structural domains are converged in the inner loop, the physical timestep need only be made small enough to resolve the structural response. This subiterative, synchronous coupling procedure is therefore both more versatile and more computationally efficient [62, 14, 15].

For each change of the structural state-vector, the deformable CFD-mesh is updated and the structural loads are recomputed for the next iteration. Adjustment of the CFD-mesh is usually a rather slow process, so to speed up the inner loop, the update sequence is usually restricted to a coarser subset of iterations. This is justified since, if the physical timestep is short enough to resolve the structural motion, the variation of the structural state is very slow. Provided the inner loop is well-converged, the coupling remains synchronous.

2.2 Dynamic Computational Mesh

In common with most modern CFD codes, the Edge flow solver uses an unstructured, finite-volume spatial discretisation with a dual-mesh formulation.

[1] The much-used term "strong coupling" is ambiguous and is therefore avoided.

This choice of method introduces a major complication for the implementation of aeroelastic functionality so it described here in some detail.

Static Mesh

The conventional, static computational mesh is constructed in two stages.

1. The primary mesh is generated from the (CAD) surface geometry using a suitable unstructured meshing tool, for example the FOI in-house system TRITET [59, 56, 57, 58]. The mesh comprises sets of elements (for example tetrahedra and triangles) the nodes of which fill the flow domain and cover its boundaries. This dataset is stored in the `Edge.bmsh` file.

2. The dual mesh, used by the Edge flow solver, comprises a set of cells (the control volumes) each of which encloses a node of the primary mesh. Each cell in the dual mesh is constructed from a set of intersecting surfaces, one for each "edge" of the primary grid. The geometric construction used to generate these cells is described in the Edge manual [25] (theory section) and is computed using the Edge preprocessor program `dual`. The dual mesh dataset normally contains node and edge (surface) data only. The element information from the primary grid is not used by the flow solver. This dataset and its modifications are stored in the `Edge.bedg` file.

The basic dual mesh is sufficient to compute a flow solution but in order to use multigrid methods, a further construction is required to define the required series of grid levels. For this, the cells of the dual mesh are selectively merged to form a hierarchy of nested meshes, using an "agglomeration" algorithm [10, 61, 42]. This is carried out using the Edge preprocessor program `agglom`.

The concept of the dual mesh and multigrid agglomeration is illustrated in figure 2.1 for a simple, 2D example. The top image shows the primary mesh (black) and dual mesh (blue) superimposed. The lower three images show the dual mesh again, together with the next two multigrid levels. The primary grid here comprises only simple, triangle and bar, elements but the shapes of the cells in the dual mesh are much more complex. The cells at each multigrid level are constructed from surfaces present at the next finest level.

Dynamic Mesh

In a standard CFD calculation, which uses a rigid mesh, the dual mesh and its agglomeration multigrid structure is computed once only, by the preprocessor programs. However, for an aeroelastic calculation, the surface geometry and the surrounding mesh are both time-dependent. This means that every time the structural state vector changes, the position and velocity of all nodes in the mesh must be updated and the surfaces of the dual mesh must be regenerated, for all multigrid levels.

The movement of the dual mesh is illustrated in figure 2.2. This figure shows the primary mesh (black) and dual mesh (blue) for a simple 2D airfoil under rigid rotation. The primary mesh consists of simple triangular elements, with bar elements forming the boundary surface. Its dual mesh is constructed as described above (Static Mesh 2) and each surface in the dual mesh is constructed from triangular segments (tetrahedra in 3D) though these are not shown here. To clarify the effect of mesh motion, a small "constellation" of primary mesh nodes is highlighted in red. Using these points as a guide, it can be seen that this modest deformation of the primary mesh results in a complex pattern of shape changes in the cells of the dual mesh. Furthermore, these shape change not the same as the continuous, quasi-elastic deformation

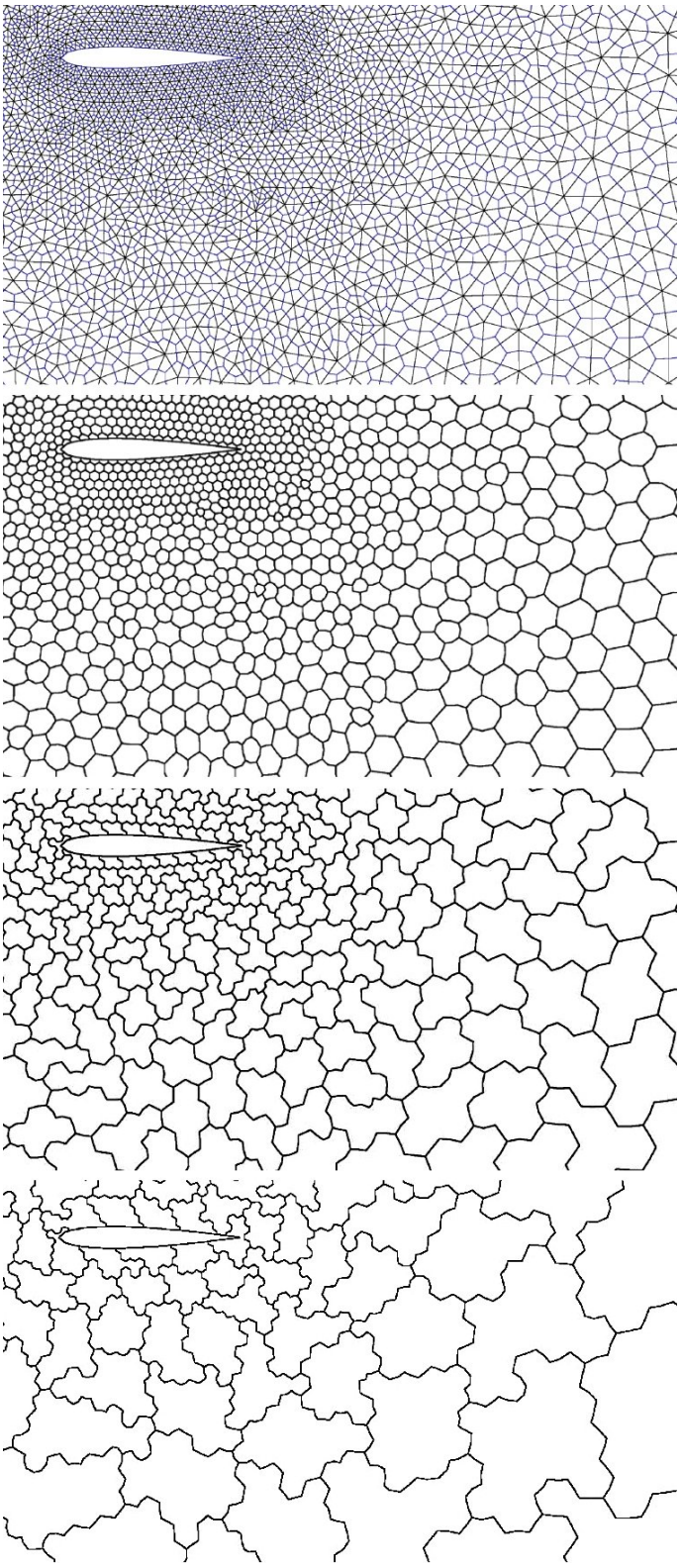


Figure 2.1: Example 2D mesh and its dual with two multigrid levels.

applied to the primary mesh. This is because the surfaces of the dual mesh are separately reconstructed, from the nodes and edges of the deformed primary mesh.

Fortunately, for almost all aeroelastic applications, the structural deformations are small enough to allow the connectivity of the mesh to be kept constant. The same applies to the multigrid agglomeration structure. This allows the vector surfaces and metrics of the dual mesh to be re-computed from the updated coordinates of the primary mesh (control-volume centre points) and then transferred directly to all multigrid levels.

To do this, some of the element information from the primary mesh must be retained after generating the initial dual mesh and functions from the preprocessor program `dual` must be made accessible within the main Edge code. In the present version of Edge, the required element information is stored in the `Edge.bedg` (dual mesh) file, as the datasets `node_edgelist` and `node_edgelist_id` (both 1D, integer fields). The entire wetted-surface and the dual mesh system is updated via the following call sequence.

```
CREMESH  ! update boundary surface and deformable mesh
 \
  DMO2COVE  ! set node coordinates and velocities
  |
  DUALCOVE  ! update the dual mesh
   \
    EDGE_SURFACES     ! compute edge-surface vectors
    |
    VOLUME            !   "   cell volumes
    |
    BOUNDARY_SURFACES !   "   boundary-surface vectors
    |
    VOLCH             ! check control volume sums
    |
    Transfer dual mesh data to all multigrid levels
```

The node coordinates and velocities of the primary mesh are first set via subroutine `DMO2COVE` from the structural state vector. The dual mesh is then recomputed via subroutine `DUALCOVE` using functions from the preprocessor.

Since the subroutine `DUALCOVE` requires only the primary mesh coordinates and velocities, it can be re-used for any computation requiring a deformable mesh, for example a non-modal structural solver. For any such application, the subroutine `DMO2COVE` would be replaced by an appropriate equivalent function.

2.3 Structural Model and Modal Reduction

The present version of Edge, assumes a linear structural model and it is further simplified by using a modal formulation. This is considered appropriate for an initial implementation. The possible extension of Edge to handle more general structural models is described in chapter 5. Here we present the theoretical formulation used for the present, modal, implementation.

For a wide class of mechanical systems, the dynamics can, for small displacements be accurately represented by a system of linear differential equations of the form

$$\mathbf{M}\ddot{x} + \mathbf{C}\dot{x} + \mathbf{K}\dot{x} = f(t) \tag{2.1}$$

where, $x(t)$ is the vector of coordinates for the nodes of the structural model $f(t)$ is the corresponding vector of external forces and the real, symmetric

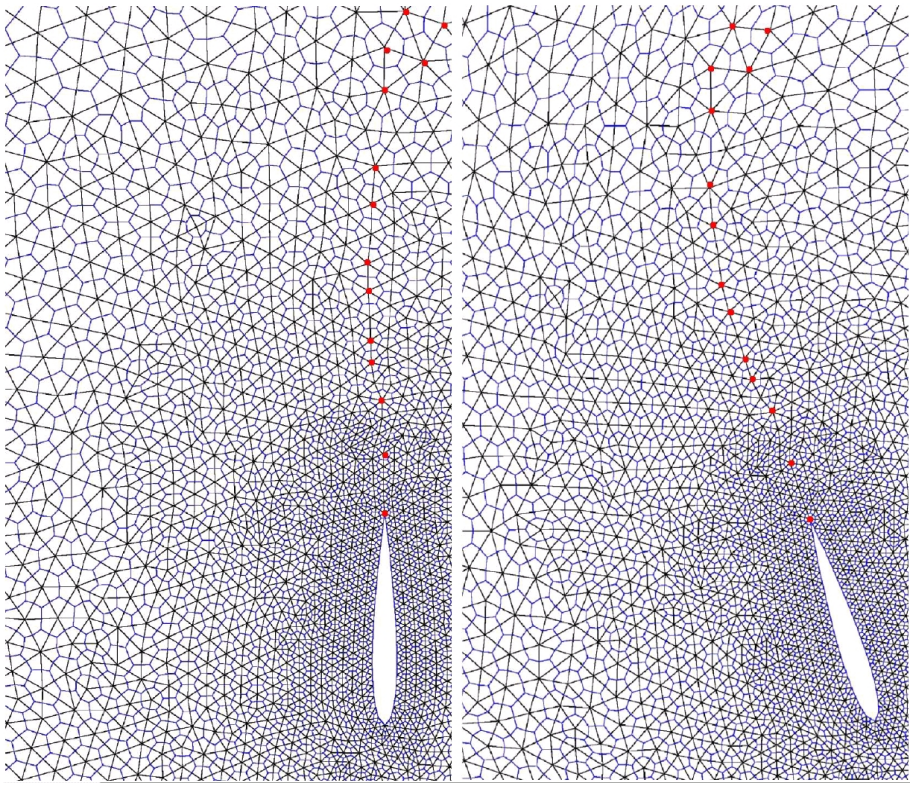


Figure 2.2: Dual mesh update and grid deformation for a 2D surface rotation

matrices M, C and K represent the mass, damping and stiffness terms respectively. This equation of motion is usually the result of spatial discretisation in the form of a Finite Element element model. For a typical FE-model, with several thousand degrees of freedom, these are large, sparse matrices.

Modal Reduction

For structural dynamics applications, the structural model is usually reduced to an approximate representation in terms of it's *normal modes*, the solutions of of equation 2.1 for the free, undamped case. The normal modes are oscillatory solutions of the form $x(t) = \phi e^{j\omega t}$, in which the shape functions, ϕ, are solutions to the generalised eigenvalue problem

$$\mathbf{K}\phi = \omega^2 \mathbf{M}\phi \tag{2.2}$$

Since the matrices $\mathbf{M}$ and $\mathbf{K}$ are real and symmetric, these functions form a complete, orthonormal set, satisfying the conditions

$$\begin{aligned} \phi_j^T \mathbf{M} \phi_k &= \delta_{jk} a_k \\ \phi_j^T \mathbf{K} \phi_k &= \delta_{jk} a_k \omega_k^2 \end{aligned} \tag{2.3}$$

in which ω_k is the "natural frequency" and the normalisation constant a_k is the "generalised mass" for mode k. The eigenvalue problem is solved using an algorithm which extracts the eigenvectors, ϕ_k, in order of increasing frequency, ω_k. This allows modal basis to be truncated, by imposing a cut-off frequency ω_N such that $\omega_k \leq \omega_N$ and $k \leq N$. The structural model in equation 2.1

can thus be reduced to a smaller, modal system which retains only the long-wavelength behaviour of the original FE-model. The modal equation of motion is derived as follows.

Using the normal modes (2.3) as a basis set, the vector of structural coordinates, $x(t)$, may be written as

$$x(t) = \sum_{k=1}^{N} \phi_k q_k(t) \tag{2.4}$$

where $q_k(t)$ is the modal, generalised coordinate for mode k. Substiuting this form in 2.1 and applying the orthogonality conditions 2.3, the equation of motion can be reduced to the form

$$\ddot{q}_k + 2\zeta_k \dot{q}_k + \omega_k^2 q_k = \frac{1}{a_k} Q_k \tag{2.5}$$

where ζ is the *damping ratio* for mode k and the scalar term

$$Q_k(t) = \phi_k^T \mathbf{f}(t) \tag{2.6}$$

is the corresponding modal, generalised force. In this derivation we make the standard assumption that C is a linear combination of M and K, a condition known as "proportional damping" (or Rayleigh damping).

The modal equation of motion, 2.5, is a simple, diagonal system comprising N, 1D harmonic oscillators which are coupled solely by the modal forces, Q_k.

2.4 Spatial Coupling and Modal Forces

The modal force, Q_k, in equation 2.5 is a weighted summation over the forces on the nodes of the structural grid. However, the fluid forces are returned by the flow solver at the nodes of the wetted surface grid. To compute the modal forces, these fluid loads must be transformed onto the structural grid, as noted in section 2.1. This is the "spatial coupling problem", as discussed in section 2.1 and it is fundamental to all partitioned aeroelastic methods. A simplified analysis of this problem is presented below.

Energy Conservation For the present case of a linear structural model, we may assume that there exists a constant, linear transformation matrix, $\mathbf{H}$, which maps displacements from the internal structural grid (S) to the wetted-surface grid (F). The spatial coupling transformation relating forces and displacements on the two grids is then given by

$$\begin{aligned} \delta x_F &= \mathbf{H} \delta x_S \\ \mathbf{H}^T f_F &= f_S \end{aligned} \tag{2.7}$$

where x and f are the vectors of structural coordinates and forces, as defined for equation 2.1.

The force transformation follows immediately from the condition

$$\delta W = (\delta x_S)^T f_S = (\delta x_F)^T f_F \tag{2.8}$$

which states that the virtual work, δW, done by an arbitrary, infinitesimal displacement is invariant under the inter-grid transformation. The transformation (2.7) therefore satisfies both conservation of energy and the preservation of equilibrium. The linear, spatial coupling problem is thus reduced to finding a suitable form for the displacement-transfer matrix, $\mathbf{H}$.

Modal Forces For a modal system, the spatial coupling problem is further simplified because the transformation matrix, $\mathbf{H}$, can be eliminated from the equation of motion. To show this, we first note that the modal basis functions in equation 2.4 are displacements on the structural grid. The transformation in equation 2.7 can be thus used to generate a corresponding set of modeshapes, ϕ_{Fk}, defined on the wetted-surface grid (F).

$$\phi_{Fk} = \mathbf{H}\phi_{Sk} \tag{2.9}$$

Given these modeshapes and the force distribution on the wetted surface, the modal force term can evaluated using equation 2.6.

$$Q_{Fk} = \phi_{Fk}^T \mathbf{f}_F$$

Applying the spatial coupling transformations (2.7) it is easily shown that this modal force, Q_{Fk}, is identical to that evaluated on the structural grid, Q_{Sk}.

$$\begin{aligned} Q_{Fk} &= \phi_{Fk}^T \left[\mathbf{H}^T \mathbf{f}_S\right] \\ &= \left[\mathbf{H}\phi_{Sk}^T\right]^T \mathbf{f}_S \\ &= \phi_{Sk}^T \mathbf{f}_S \\ &= Q_{Sk} \end{aligned} \tag{2.10}$$

The modal force, $Q_k(t)$, being dimensionally equivalent to the virtual work δW in equation 2.8, is thus invariant under the spatial coupling transformation.

It follows that the modal force can therefore be evaluated as a weighted integral of the fluid forces over the wetted-surface.

$$Q_k(t) = \oint_{S(t)} (\delta \mathbf{x}_k) \cdot \left[p(t)\mathrm{I} + \mathbf{T}(t)\right] \mathrm{d}\mathbf{S} \tag{2.11}$$

where $p(t)$ and $\mathbf{T}(t)$ are the local values of the fluid pressure and the viscous stress tensor respectively and $\delta\mathbf{x}_k$ is the constant, local surface displacement for modeshape ϕ_{Fk}^T. Note that apart from the modeshape term, $\delta\mathbf{x}_k$, all variables in this equation are explicity time-dependent, including the surface elements, $\mathrm{d}\mathbf{S}$. In Edge this integral is evaluated, in discrete form, by the subroutine `CMOFORCE`, where the code is similar to that used for the rigid-body forces in subroutines `EBOUIN`, `FCFOAL` and `FVFOAL`.

Surface Normals For applications where, as for many linear flutter codes, the modeshapes are approximated by unidirectional displacements, it is more accurate to compute the modal forces using a constant, undisplaced surface [38]. The selection of constant or variable integration surface is controlled by the parameter `IFIX_B_SURFS` in the main Edge input file. In practice, for most aeroelastic simulations, the surface movement is so small that the solution is insensitive to the choice of this setting.

Geometric Interpolation The above energy conservation analysis alone is not sufficient to define the spatial coupling transformation. Indeed, the conditions 2.7 and 2.8 can be satisfied by any random matrix, $\mathbf{H}$, with consistent size. However, to produce a physically meaningful simulation, it is also necessary that the transformation preserves the shape and scale of the displacement distribution. In particular, for the transformation to be shape-preserving, it must satisfy invariance under rigid-body rotation and translation. It has recently been shown that optimal interpolation schemes satisfying these constraints can be derived using a formulation in terms of Radial Basis Functions [35, 34, 7]. These methods are now widely used for fluid-structure applications. However, as an initial implementation, it was decided to use the same

interpolation method as in Euranus. One reason for this was that much of the original code could be re-used. More importantly, this was also the most effecient way to obtain the matched spatial coupling schemes needed for validation (see chapter 4).

Modeshape Interploation The transformation of modeshapes from the structural grid to the CFD boundary surface is carried out using the Edge program `aedbelast` which is described in the manual [25]. The program is designed to provide several interpolation methods, though currently one is implemented: the *polyfit* procedure developed for use with Euranus [67]. The code is similar to the original Saab-developed program `modcreat` but has been extended to allow general, 3D structural deflections.

In the *polyfit* procedure, the deflection of the boundary surface for each modeshape is computed by a special, least-squares polynomial fit to a set of preselected control points in the structural grid. To allow greater geometric fidelity, the surface is divided up into a set of overlapping "boxes" providing a piecewise smooth fitted surface. The same set of polynomial coefficients are used for all modeshapes and the final transformation is therfore of the form given in equation 2.9, as required. The use of this interpolation procedure is illustrated in section 4.2.

2.5 Time-Integration for Structural Dynamics

The basic structural equation of motion 2.1, is usually integrated using a time-discretisation based on the Newmark method [47, 33]. The implementation used in Edge is based on special case of this method, a centred-difference scheme, in which the continuous-time variables are approximated as follows.

$$\begin{aligned} \widehat{\ddot{x}} &= (\Delta t)^{-2}\,(x_{n+1} - 2x_n + x_{n-1}) \\ \widehat{\dot{x}} &= \frac{1}{2}(\Delta t)^{-1}\,(x_{n+1} - x_{n-1}) \\ \widehat{x} &= \frac{1}{4}\,(x_{n+1} + 2x_n + x_{n-1}) \\ \widehat{f} &= \frac{1}{4}\,(f_{n+1} + 2f_n + f_{n-1}) \end{aligned} \tag{2.12}$$

where Δt is the constant, physical timestep, as discussed on a page 8.

The weights in the expression for $\widehat{x}$ are chosen such that, for the free, undamped case, the total structural energy is conserved exactly [38]. The same weights are used for the force term $\widehat{f}$ but this does not imply energy conservation for the more general case. This energy conservation properties of the scheme can be demonstrated by evaluating the rate of change

$$P_n = \frac{1}{\Delta t}\left[E_{n+\frac{1}{2}} - E_{n+\frac{1}{2}}\right] \tag{2.13}$$

where $E_{n+\frac{1}{2}}$ is the total energy given by

$$\begin{aligned} E_{n+\frac{1}{2}} &= \frac{1}{2}\left(\frac{x_{n+1} - x_{n-1}}{\Delta t}\right)\mathbf{M}\left(\frac{x_{n+1} - x_{n-1}}{\Delta t}\right)^T \\ &+ \frac{1}{2}\left(\frac{x_{n+1} + x_{n-1}}{2}\right)\mathbf{K}\left(\frac{x_{n+1} + x_{n-1}}{2}\right)^T \end{aligned} \tag{2.14}$$

Using the general equation of motion 2.1, it is easily shown that, with $C = 0$ and $f(t) = 0$, the rate P_n is identically zero. It follows that for this case, the

scheme is unconditionally stable [38]. For the more general, forced, damped case, the scheme is conditionally stable [33]. However, in practice, for most systems, stability can usually be acheived simply by making the timestep small enough to resolve the highest structural mode.

From the scheme definition, 2.12, it follows that the update equation for the vector of structural coordinates, x_n, can be written as

$$\mathbf{A}_1 x_{n+1} = \frac{1}{4}\left(f_{n+1} + 2f_n + f_{n-1}\right) - \mathbf{A}_0 x_n - \mathbf{A}_{-1} x_{n-1} \tag{2.15}$$

where the matrices $\mathbf{A}$ are given by

$$\begin{aligned} \mathbf{A}_0 &= -2(\Delta t)^{-2}\mathbf{M} + \frac{1}{2}\mathbf{K} \\ \mathbf{A}_{\pm 1} &= (\Delta t)^{-2}\mathbf{M} \pm \frac{1}{2}(\Delta t)^{-1}\mathbf{C} + \frac{1}{4}\mathbf{K} \end{aligned} \tag{2.16}$$

The update equation is "implicit" in the sense that the coordinates, x_{n+1}, depend on the forces, f_{n+1}, at the same time level, t_{n+1}. The equation must therefore solved iteratively, at at each real timestep, as described on page 2.1. For the general, non-modal case, the update equation (2.15) must be solved as a linear system. However, the matrices $\mathbf{A}$ are constant, symmetric and are usually sparse or small, so for most systems this is not a computationally expensive operation.

For the special case of a diagonal, modal system, the inverse of matrix $\mathbf{A}_1$ is trivial and equation 2.15 can can, for each mode, be written in scalar form as

$$q_{n+1} = \frac{1}{A_1}\left[\frac{1}{4a}\left(Q_{n+1} + 2Q_n + Q_{n-1}\right) - A_0 x_n - A_{-1} x_{n-1}\right]$$

where the coefficients A are given by

$$\begin{aligned} A_0 &= -2(\Delta t)^{-2} + \frac{1}{2}\omega^2 \\ A_{\pm 1} &= (\Delta t)^{-2} \pm (\Delta t)^{-1}\zeta\omega + \frac{1}{4}\omega^2 \end{aligned} \tag{2.17}$$

and modal parameters ω, ζ and a are defined as for the modal equation of motion (2.5) and the modal subscript, k, is omitted for clarity. Note that the generalised mass, a, appears only as a scale factor on the generalised force, Q.

2.6 Mesh Movement

As described in section 2.2, every time the structural state changes, the CFD mesh must be adjusted to comply with the instantaneous surface geometry. In general, a separate, mesh deformation system is required. For this the mesh is usually treated as quasi-elastic medium and the displacement field is computed subject to the state of the wetted-surface boundary. This can be formulated as three-field problem [21] with the mesh deflection as an additional dynamic field. However, with the modal implementation used here, this is not necessary because the mesh deformation is performed off-line.

Perturbation Fields The present implementation of modal aeroelasticity in Edge uses the perturbation field method, as introduced for the earlier Euranus code [68, 67]. The modal shape data, representing the basis functions of equation 2.3, is extended to form a set of displacement fields for the whole CFD mesh. This dataset is stored in the `Edge.bmos` file and is constructed in advance, using the Edge "AE" programs, as described in the Edge manual [25]. The sequence of operations is typically as follows.

1. Program `nastran2edge`: Extract modeshapes, mass and frequency data from NASTRAN (.f06) output files → `Edge.amod`
2. Program `aedbelast`: Transform modeshapes from the structural grid to surface displacements → `Edge_mode.bdis`
3. Program `aetribend`: For each modal surface displacement, compute a deformed mesh → `Edge_mode.bmsh`
4. Program `aeputpert`: For each modal deformed mesh, store the perturbation field → `Edge.bmos`

In this procedure, both the spatial coupling and the mesh-deformation operations are carried out off-line as "aeroelastic preprocessing". The mesh deformation program, `aetribend`, uses components from the mesh-generation system TRITET [58] and the algorithm is based on solution of a modified Laplacian field problem [57]. This operation is computationally expensive but is only required once, for each mode.

Within the Edge solver itself, the mesh coordinates field $X(t)$ is computed directly from the modal coordinates $q_k(t)$ in the form

$$X(t) = X_0 + \sum_{k=1}^{N} (\Delta X)_k \, q_k(t) \tag{2.18}$$

where X_0 is the coordinates field of the base mesh and $(\Delta X)_k$ is the perturbation field for mode k. Given the new mesh coordinates field, the multigrid, dual mesh is then updated as described in section 2.2.

The perturbation field method offers a significant advantage when the available mesh deformation code is slow or, as in the present case, difficult to integrate with the main code. However, these benefits are offset by the method's somewhat heavy use of memory. Specifically, a volume equivalent to that occupied by the entire flow-velocity field, multiplied by the number of modes, is used simply to store the constant, modal perturbations. This "full-field modeshape" data also requires external storage. For simulations with detailed CFD models or with a large number of modes, this overhead[1] becomes unacceptable. Furthermore, the method can not be extended to more general, non-modal, structural models. For these reasons, most contemporary aeroelastic simulation codes are equipped with functions for on-line mesh deformation.

Mesh Velocities and GCL For solutions with moving grids, the present implementation of the *Edge* flow-solver requires both the coordinates and velocities of the primary-mesh nodes. The grid velocities can be computed from the first derivative of the modal coordinates as in equation 2.18, however, the modal structural solver, equation 2.17, does not include this term. The modal velocity vector$\dot{q}(t)$ is therefore computed using the approximation.

$$\dot{q}_{n+1} = \frac{1}{2\Delta t} \left(3q_n - 4q_{n-1} + q_{n-2}\right) \left[1 + O\left((\Delta t)^2\right)\right] \tag{2.19}$$

In future versions of Edge the flow-solver will probably be re-formulated so that the grid velocity field is not required. This is related to the implementation of a time-integration scheme compliant with the Geometric Conservation Law (GCL) [55]. The GCL is, briefly, a mathematical condition which ensures

[1] For example, consider a realistic, full-aircraft simulation with unsteady RANS flow. This could easily require a 10 million cell CFD mesh and 50 modes in the structural model. For this, using 8-byte real data, the perturbation fields would need 12 Gb of memory.

that the free-stream solution is undisturbed by the motion of a deforming volume-mesh. For aeroelastic simulation this is, strictly, a necessary prequisite for both time-accuracy and numerical stability [40, 31]. However, for most aeroelastic systems of practical interest, the grid speeds are sufficiently small for GCL-effects to be negligible. The substantial modifications required for GCL-compliance have therefore been postponed for a future implantation.

2.7 Simulation Options

The aeroelastic simulation options in Edge are selected by the parameter `IAEOPT` in the main input file and are described fully in the program manual [25]. A brief description is included here for completeness. The aeroelastic options currently available are as follows.

`IAEOPT`	0	no aeroelastics (default)
`IAEOPT`	100	modal rigid
`IAEOPT`	101	modal prescribed motion
`IAEOPT`	102	modal coupled: orthogonal modes, central difference

For all three aeroelastic options the required inputs are the modal shape data (perturbation fields) and the modal parameters (files `Edge.bmos` and `Edge.amop` respectively). The response time-history of the modal coordinates forces is recorded in the residuals file, `Edge.bres`.

Option 100: Modal Rigid The usual starting point for any aeroelastic simulation is a well-converged, steady-state flow solution. However, it can be useful to first extend this into a time-accurate solution with the rigid geometry and compute the modal, generalised forces. This provides a check on any intrinsic unsteadiness of the flow solution. If the flow is sufficiently steady, it can be used to start a coupled solution, with the inital modal forces offset to produce a "flight shape" simulation, as explained below (page 19).

Option 101: Modal Prescribed Motion The modal, prescribed motion function enables the simulation of flows subject to an arbitrary motion of the wetted surface. The surface motion is specified as a tabulated timeseries of the modal coordinate vector, $q(t)$ and the response of the flow field is returned both directly and as the vector of surface-integrated, generalised forces, $Q(t)$. The modal timeseries tabulation is set in the `Edge.amot` input file.

The intended application of prescribed motion simulations is the computation of aerodynamic transfer functions, which is standard practice in the Aerospace industry. Typically, the surface motion would be prescribed as single mode input, with a selected pulse profile and generalised forces would be computed for the whole mode set. Using digital signal processing techniques, a linear, frequency domain SIMO (Single Input Multiple Output) analysis would be performed to extract one row of a transfer function matrix, $\mathbf{H}(\omega)$, as defined in the expression

$$\tilde{Q}(\omega) = \mathbf{H}(\omega)\tilde{q}(\omega) \tag{2.20}$$

Linear transfer functions of this form can be used for flutter prediction and dynamic loads calculations. This type of analysis has been performed, using Edge and its Matlab interface, in the FOI-project *Adaptiva Strukturer* [70].

Another application for modal prescribed motion solutions is the simulation of large-displacement motions whilst still using the existing modal, perturbation-field implementation. For example, rigid-body rotations can be approximated by scheduling a sequence of perturbation fields for incremental rotations. In

chapter 3 a simulation with two modes is used to model a rigid pitch oscillation of the LANN-wing wind-tunnel model.

Option 102: Modal Coupled Simulation Dynamic coupled simulations may be of two types, "jig shape" or "flight shape", depending on the assumed structural model. In a "jig shape" simulation it is assumed that the geometry and structural model represents the unloaded shape: the shape of the structure when supported in the "jig" used for it's construction. The first stage in such a simulation is to compute the static equilibrium state at the required flight condition. This then forms the initial state for the dynamic simulation. In a "design shape" simulation it is assumed that the supplied geometry represents the true flight shape. To provide the initial equilibrium state, the modal force is offset by its initial, steady-state value ($Q = Q(t) - Q_{steady}$ in equation 2.5). This option is selected using parameter `IFOFFSET` in the the main Edge input file. The validation study presented in chapter 4 uses a "jig shape" simulation.

In most dynamic simulations, the structure must be excited by some non-aerodynamic energy input. This can be achieved using the initial conditions, for example by fixing an non-zero initial modal velocity (input file `Edge.amop`). Alternatively, a structural force input can be specified as a tabulated, modal time-series using the same system as for Prescribed Motion solutions. This option, activated by parameter `IFORCED`, allows the simulation of processes such as the ejection and release of wing-mounted, external stores.

2.8 Structural Energy Analysis

For any Edge modal simulation, the complete time-history of the modal variables (coordinates, velocities and forces) is recorded in the `Edge.bres` output file. For a coupled simulation, this provides a complete description of the structural state. It also provides a means of checking the time-integration process. Using the orthogonality conditions 2.3 the total structural energy can be written as a modal decomposition

$$E(t) = \sum_{k=1}^{N} E_k(t)$$

where the modal energy, E_k, is given by

$$\begin{aligned} E_k &= E_k^K + E_k^P \\ 2E_k^K &= a_k \dot{q}_k^2 \\ 2E_k^P &= a_k \omega_k^2 q_k^2 \end{aligned} \tag{2.21}$$

The structural energy variables can thus be evaluated directly from the modal time history output and the modal parameters input data. The basic equation of motion 2.1 can be written as the energy conservation condition

$$\dot{E} + \dot{x}^T \mathbf{C} \dot{x} - \dot{x}^T f = 0 \tag{2.22}$$

For the discrete approximation, we may replace the RHS zero in equation 2.22 with an *energy rate error*, $\xi(t)$. This may be written in terms of the modal coordinates as

$$\xi(t) = \dot{E} + 2 \sum_{k=1}^{N} \left[a_k \zeta_k \omega_k \dot{q}_k^2 - \dot{q}_k Q_k \right] \tag{2.23}$$

The energy rate error provides a convenient measure of the accuracy of the discrete time-integration process. It can be easily computed from the modal time-history output using the Edge Matlab interface.

2.9 Edge-Matlab Interface

Aeroelastic simulation requires a more extensive and more flexible pre- and post-processing environment than conventional steady-state CFD. For this reason, an interface has been constructed between *Edge* and the industry-standard computing environment Matlab [41]. This system, the "FFA Matlab Toolbox" is based on a data standard known as "FFA format" (Flexible Format Architecture) which is the basis for all internal and external data structures used by Edge. The same data standard is also used by the CFD code Euranus [23] and the Infra-Red emission analysis code SIGGE [5, 4]. The FFA format is described in the Edge manuals [25, 13] and the original specification [63]. The Edge source package provides support for the FFA format in the programming languages Fortran-90, Java, Matlab and Python (see table 2.1).

The FFA Matlab Toolbox, which was first released in 2002, is documented, with a browser-based manual [52] and Matlab-style, on-line help text. A set of ready-made Matlab functions, including the FFA toolbox, is supplied with Edge. Of these functions, the most important ones for aeroelastic applications are as listed in table 2.2. The functions `pdiff` and `harm_bout` are used for the analysis of periodic, prescribed motion, simulations as explained chapter 3. The functions `mortors` and `mortergs` are used in the MDO-wing validation study (chapter 4) to obtain the structural response in real space and for structural energy analysis (section 2.8). The on-line help texts for these and other Matlab functions are given in appendix A.

language	source
Fortran-90	`Edge/util/*.f90`
Java	`Edge/gui/ffaDS.java`
Matlab	`Edge/matlab/*.m`
Python	`Edge/bin/ffa*.py`

Table 2.1: Location of Edge source code supporting the FFA data format

program	description	input data
`harm_bout`	*Harmonic field analysis* Computes the inter-period change series for a periodic Prescribed Motion solution	solution timeseries files: `Edge_n.bres`
`pdiff`	*Periodic difference* Computes the inter-period change series for a periodic Prescribed Motion solution	modal timeseries file: `Edge.bres`
`mortors`	*Modal response to real-space* Computes the real structural displacement and velocity (equation 2.4)	modal timeseries + mode definition file: `Edge.amod`
`mortergs`	*Modal response to Energy series* Computes the total structural energy, its rate of change and modal decomposition	modal timeseries + modal parameters file: `Edge.amop`

Table 2.2: Some Edge Matlab programs used for Aeroelasitc applications

3 Validation 1: LANN-wing

3.1 Background and Case Definition

The LANN-wing study was completed following the release of Edge version 3.2, with the primary objective being to validate the new, aeroelastic function for modal prescribed motion. The work was carried out within the TAURUS-project, as part of a validation study for the DLR-developed, unstructured CFD code TAU [16]. The results presented here are for an unsteady RANS[1] simulaton for the LANN-wing have previously been issued in an addendum [51] to the final[2] TAURUS-report [18].

The experimental reference is a rigid wing with prescribed sinusoidal pitch oscillation: the LANN wind-tunnel model. For such a system, no structural model is required, so the CFD and mesh movement components of Edge can be tested in isolation. Here we present computational results from Edge and TAU (obtained using the same unstructured mesh) compared with earlier results from the structured solver, Euranus [32] and with the experimental data.

Test Case Description

The LANN wing test case is dataset 9 from the NATO AGARD series of reference results for unsteady aerodynamics [45, 46]. The geometry is an isolated, clean wing with aspect ratio 8 and a leading edge sweep of 28 degrees. The wind-tunnel experiments on the LANN wing were carried out at NLR [48] and comprise surface pressure measurements, in the Mach range 0.6 to 0.95, for various oscillatory rigid pitch conditions. The pitch rotation is about a spanwise axis running through the 62.1% root chord location.

The present study considers only one of the test conditions: the “CT5” test case, the defining parameters of which are given in table 3.1.

Reynolds number	7.31e6	Mach	0.82
Reynolds length	0.3606 m	Incidence	0.6°
Root chord	0.3606 m	Amplitude	0.25°
Rotation axis	(0.000,1,0)	Frequency	24 Hz
Rotation centre	(0.224,0,0)	Reduced Frequency	0.266

Table 3.1: LANN-wing: Defining Parameters for the CT5 test-case

The geometry of the LANN wing, and the axis of pitch rotation is shown in figure 3.1 with the wing surface in the undisplaced position (blue) and at maximum pitch up (red). It can be seen that, the largest surface movement is near the wing tip and that the average displacement amplitude is very small.

[1] RANS: Reynolds Averaged Navier Stokes

[2] The TAURUS report also includes results for inviscid Euler flow.

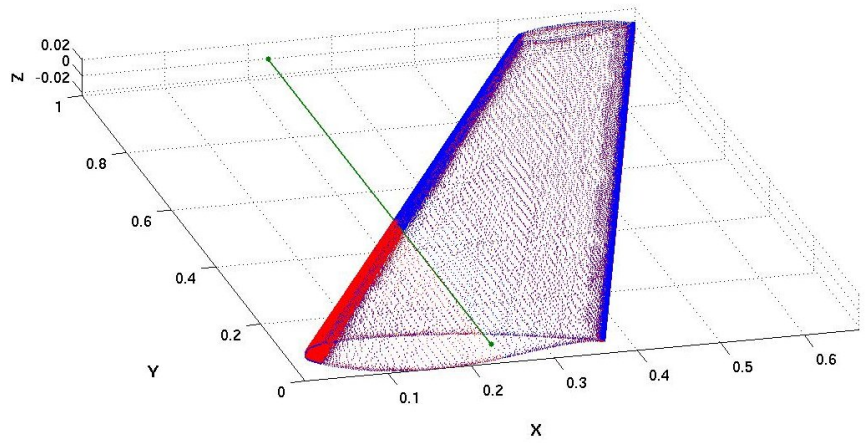


Figure 3.1: LANN-wing: surface points at zero and $\pm 0.25^\circ$ pitch

3.2 Prescribed Motion Simulation

The CFD solutions with Edge and the TAU code were obtained using a 2.5 million cell unstructured mesh supplied from the TAURUS database. Using this mesh, RANS computations were carried out with Edge and TAU for both steady state flow and prescribed-motion simulation. All Edge results were obtained using the default, two-equation, turbulence model: Wallin-Johansson EARSM with standard $k - \omega$ [64]. With the TAU code, due to robustness problems[1] the one-equation, Spalart-Alamaras turbulence model was used.

Time-accurate flow solutions were obtained using the semi-implicit, dual time-stepping method as implemented in both Edge and TAU. The outer, physical, timestep was set to exactly 100 timesteps per cycle and the solution fields were stored at every timestep, equivalent to a sampling rate of 2.4 kHz. The inner "pseudo time" loop was restricted to a fixed number of internal iterations: 10 inner iterations for the TAU computation and 20 for Edge. The use of such a small number of inner iterations was valid only because of the very small amplitude of the surface displacement and the relatively fine time-resolution.

The Edge computation was carried out as a prescribed motion aeroelastic simulation with two normal modes, one each for the "pitch up" and "pitch down" extreme positions. Mesh perturbation fields were created for the "pitch up" and "pitch down" modes, using the mesh deformation program `aetribend`. The mesh movement is synthesised as a linear combination of the two modes with 'half wave' sinusoidal scheduling. This procedure is of course only valid for small angle rotations.

The solution was monitored as the time-history of the modal coordinates and generalised forces. These are respectively the input and response of the two mode system. The two generalised forces are weighted integrals of forces over the moveable surface. Since the input is sinusoidal it is assumed that the response is periodic, with the same fundamental frequency. Convergence of the solution can thus be established using the "Periodic Difference" given by

$$PD(t) = Q(t) - Q(t - T) \tag{3.1}$$

[1] The supplied mesh was found to be of poor quality, having a very sharp transition between the prismatic boundary layer and the tetrahedral volume mesh (shown in the TAURUS report [18]). With TAU, this prevented the use of any two-equation turbulence model.

where T is the period of the input and $Q(t)$ is an output variable, such as one of the modal, generalised force terms.

The modal time history and Periodic Difference for the Edge computation are shown in figure 3.2. It can be seen that the solution is well-converged by the time the fifth pitch cycle starts. However, it should be noted that this assessment is based on the aerodynamic load integrated over the whole moving surface. There may still be non-periodic local fluctuations which are not revealed in this data.

3.2.1 Data Analysis and Representation

The results of rigid body response experiments, like the LANN wing, are usually represented in the terms of a Fourier series in which the fundamental frequency is that of the prescribed surface motion. This retains the style of classical, frequency-domain, aeroelastic methods and affords a convenient means of representation.

For any periodic, scalar variable, $p(t)$, the Fourier series representation may be written as

$$p(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos\left(n\omega t\right) + b_n \sin\left(n\omega t\right) \right)$$

where $\omega = \frac{2\pi}{T}$, and T is the period of the prescribed motion. The Fourier series coefficients are given by

$$\begin{aligned} a_n &= \mathrm{Re}\left[C_n\right] \\ b_n &= \mathrm{Im}\left[C_n\right] \end{aligned}$$

where

$$C_n = \frac{2}{T} \int_{t_1}^{t_1+T} e^{in\omega t} p(t)\, \mathrm{d}t$$

and t_1 is an arbitrary time point. The leading term, $\frac{a_0}{2}$, is the time-averaged value of $p(t)$ and is here denoted "H0". Similar notation is used for the oscillatory terms in the summation: "nth harmonic" or "Hn".

The experimental data on the LANN wing is provided in the form of surface pressure distributions at six streamwise wing cross-sections, comprising the steady state values and the H1 component of the unsteady pressure. The surface motion is defined by the pitch angle, which varies as $\sin \omega t$ (see table 3.1). The coefficient b_n is thus identified as the real (in phase) component and a_n as the imaginary (in quadrature) component. It is important to be aware of this phase convention when comparing calculated and measured data.

The traditional representation in terms of real and imaginary parts is convenient for comparing surface pressure cross sections. However, for visualisation of the unsteady surface pressure distributions and other field variables, it is more compelling to use a phase-magnitude representation. This approach was used by the author for the CFD simulation of the GFSI bump [50]. In the present study, the results are presented as an oscillatory surface pressure field for the H1 component only. For any surface point, the H1 unsteady pressure is of the form

$$p(t) = |p_0| \sin\left(\omega t + \phi\right) \tag{3.2}$$

where $|p_0|$ and ϕ are respectively the magnitude and phase. If ϕ is defined such that $-\pi \leq \phi \leq \pi$, then the phase field will have discontinuities at $\phi = \pm\pi$. However, since ϕ is a periodic variable, we display the phase field using a cyclic colour mapping which renders these discontinuities invisible.

3.2.2 Results and Analysis

Steady State and Mean Position The steady-state flow, measured for zero pitch angle, is not the same as the time averaged unsteady flow obtained with an oscillating wing. The same applies for the CFD solution. Referring to section 3.2.1, the H0 component must not be identified with the steady state flow solution. However, for the present case, the amplitude of the surface motion is so small that the difference between the time-averaged and steady-state solutions is barely visible. The surface pressure distribution and streamlines for the H0 time-averaged solution are shown in figure 3.3.

Figures 3.4 and 3.5 show section C_p distributions for steady state flow computations with Edge, together with the experimental data and results from TAU and Euranus respectively. The results for upper and lower surfaces are shown separately for each section.

There is broadly close agreement between the three sets of CFD results and the measured data. The Edge and TAU results are especially close, showing only slight variation in the shock position at the near mid-span sections. At the spanwise positions $\eta = 0.65$ and $\eta = 0.825$, the Euranus results show slightly closer agreement with experiment whilst Edge shows better agreement at $\eta = 0.325$. With an unstructured CFD code, like TAU or Edge, one would normally use mesh adaption to obtain a more accurate steady state solution. This has not been attempted due to the poor quality of the initial mesh.

Time History and Convergence Figure 3.2 shows the time history for the modal coordinates, velocities and generalised forces together with the periodic difference, as given by equation 3.1, for the "pitch up" modal force. The modal coordinates are prescribed to approximate a sinusoidal pitch rotation. It can be seen that the response in the modal force integral is approximately sinusoidal and that the periodic difference indicates that solution is adequately converged by the end of the fifth period of oscillation. Similar results were obtained using TAU.

First Harmonic Surface Pressures The results described here are for the H1 unsteady surface pressure, as described in section 3.2.1 and for the Edge solutions were computed from the fifth pitch cycle (see figure 3.2).

Fourier analysis of the TAU and Edge solutions was carried out using the FFA Matlab Toolbox [52] as supplied with Edge. The Edge Matlab program `harm_bout` was used to obtain full-field harmonics from the time-sampled flow dumps, in precisely the same format as used for steady solutions. The harmonic field data was then visualised using EnSight [12]. The Euranus results are as formely published on the UNSI-project website.

Figure 3.6 shows the H1 oscillatory surface pressure distribution for the upper and lower wing surfaces, using a phase-magnitude respresentation. Comparing with figure 3.3, it can be seen that, on the upper surface, the peak magnitude of the oscillatory pressure coincides with the maximum gradient of the steady pressure. Similar behaviour would be produced by a linear aerodynamic model, though with a different spatial pressure distribution. The shock structure on both upper and lower surfaces is clearly shown in the oscillatory pressure magnitude distribution.

The phase convention used in figure 3.6 is defined in equation 3.2. The phase field is mapped using a cyclic colour scale, with light green and deep red representing respectively "in phase" and "antiphase" oscillation. The broad features of the phase distribution may be interpreted as follows. As the incidence angle increases, on the upper surface, ahead of the shock, the C_p decreases (antiphase). At the same time, in the equivalent lower surface region,

the C_p decreases (in phase). On both upper and lower surfaces the variation results in lift increasing with incidence. The pressure phase varies continuously over the wing surface but changes sharply through the shock. The phase also shows detailed structure in the unsteady flow near the wing tip downstream of the main shock. These features would not be so clearly visible in a display showing the fields as real and imaginary parts.

Figures 3.7 and 3.8 show the same H1 surface pressure distribution sampled at the six measured spanwise sections, together with the experimental data and results from TAU and Euranus. For the sake of clarity, the results are separated both for the upper and lower wing surface and for the two reference CFD codes. In both figures the Edge results are shown by the solid blue curves.

Referring to the upper surface results in figure 3.7, it can be seen that there is very close agreement between Edge and TAU with the only significant difference being in the imaginary part at the three outer spanwise stations $\eta \geq 0.65$. Both Edge and TAU show good agreement with the experimental data throughout the span. The Euranus results are also in quite good agreement, though less so for the three outer spanwise sections where it predicts a much sharper variation in the unsteady C_p.

Results for the lower surface are shown in figure 3.8, where all three codes are generally in closer agreement with the experimental data. The Edge results show a spike in the C_p at the trailing edge. A similar effect is shown in the steady results, figures 3.4 and 3.5. This is, however, spurious and is known artefact of the post processing. The Euranus results depart from the unstructured codes mostly at the two inner sections, again predicting a sharper variation in the unsteady C_p.

3.3 Summary

For the LANN-wing CT5 test case, Edge performs at least as well as the comparable unstructured code, TAU and shows behaviour consistent with earlier computations using the structured solver, Euranus. We may therefore conclude that the implementation of unsteady RANS flow, with deforming grids, is correct. In Edge, unlike TAU and many other codes, rigid pitch solutions are implemented using exactly the same functions and data paths as used for modal aeroelasticity, including the mesh movement system. The results presented here therefore validate both the unsteady CFD component of Edge and the primary aeroelastic functions.

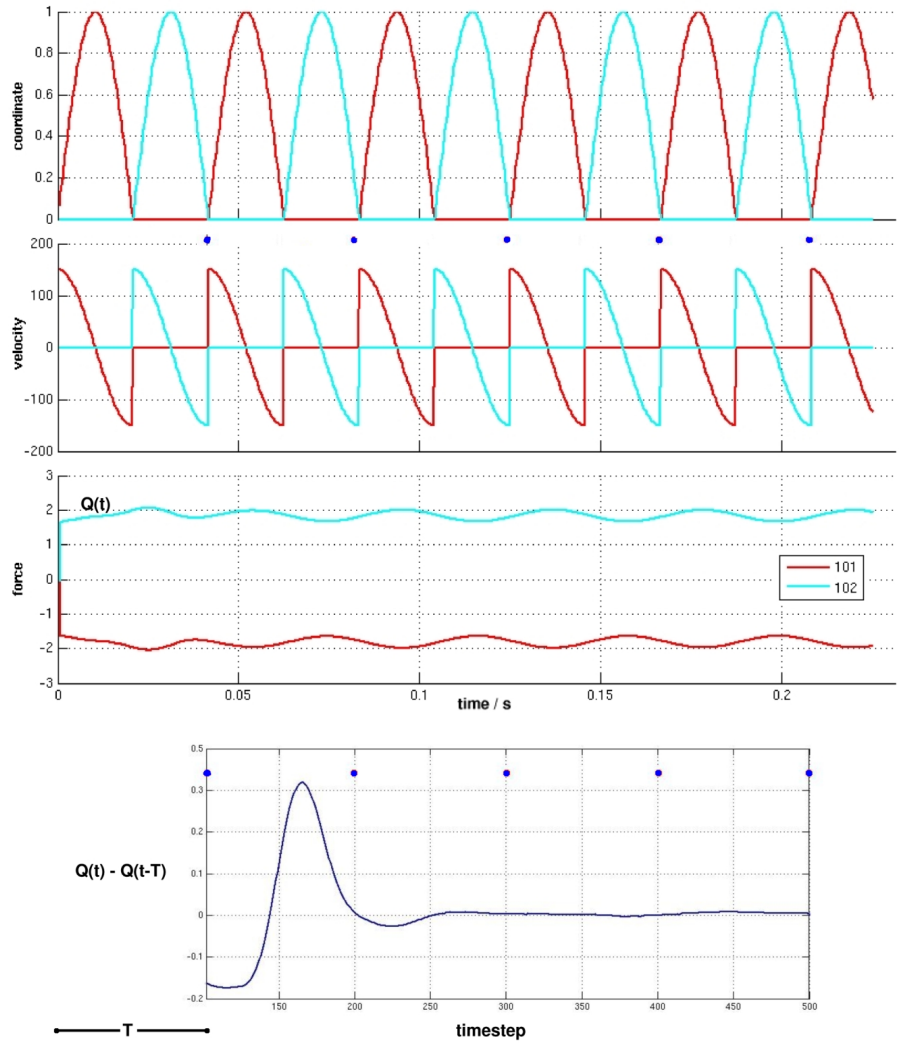


Figure 3.2: LANN-wing: Edge : Time-history of modal variables and Periodic Difference for the, Q_1, modal force

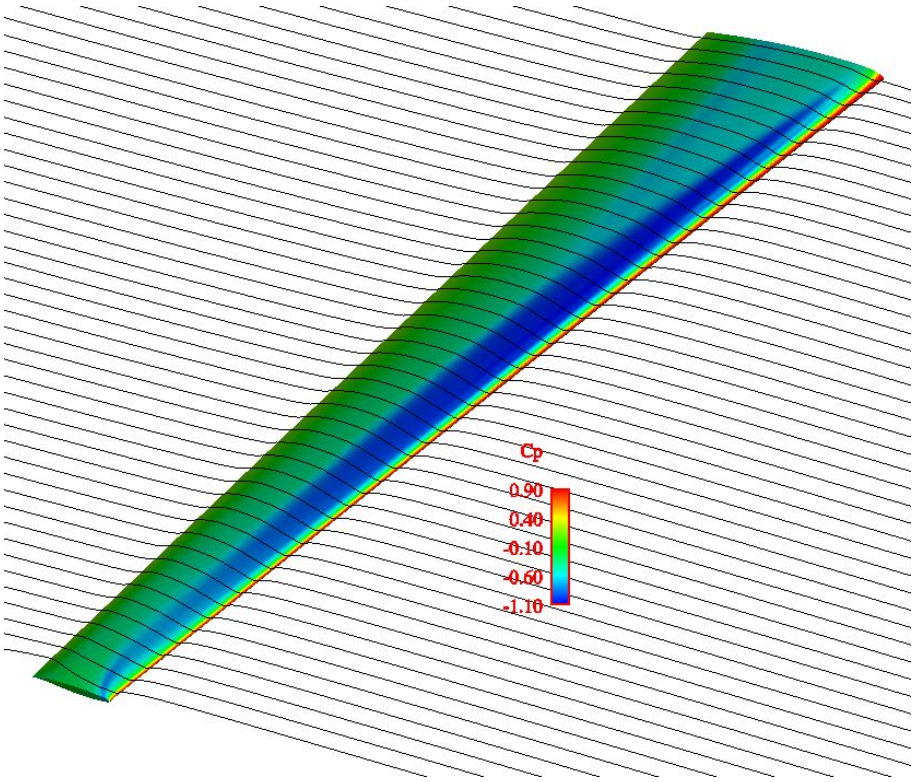


Figure 3.3: LANN-wing: Surface C_p and streamlines for H0 time-averaged solution

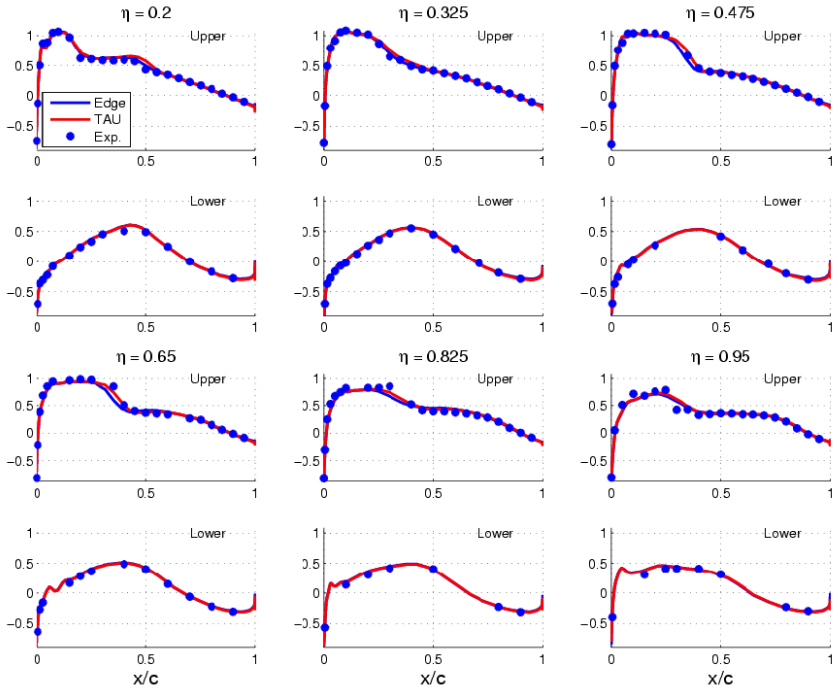


Figure 3.4: LANN-wing: Steady State section-C_p : Edge vs TAU and experiment

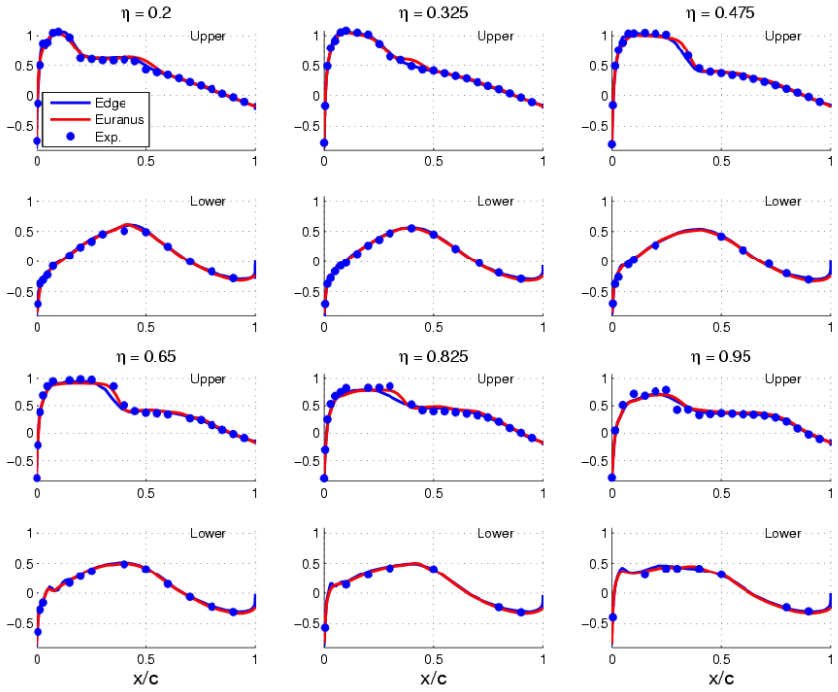


Figure 3.5: LANN-wing: Steady State section-C_p : Edge vs Euranus and experiment

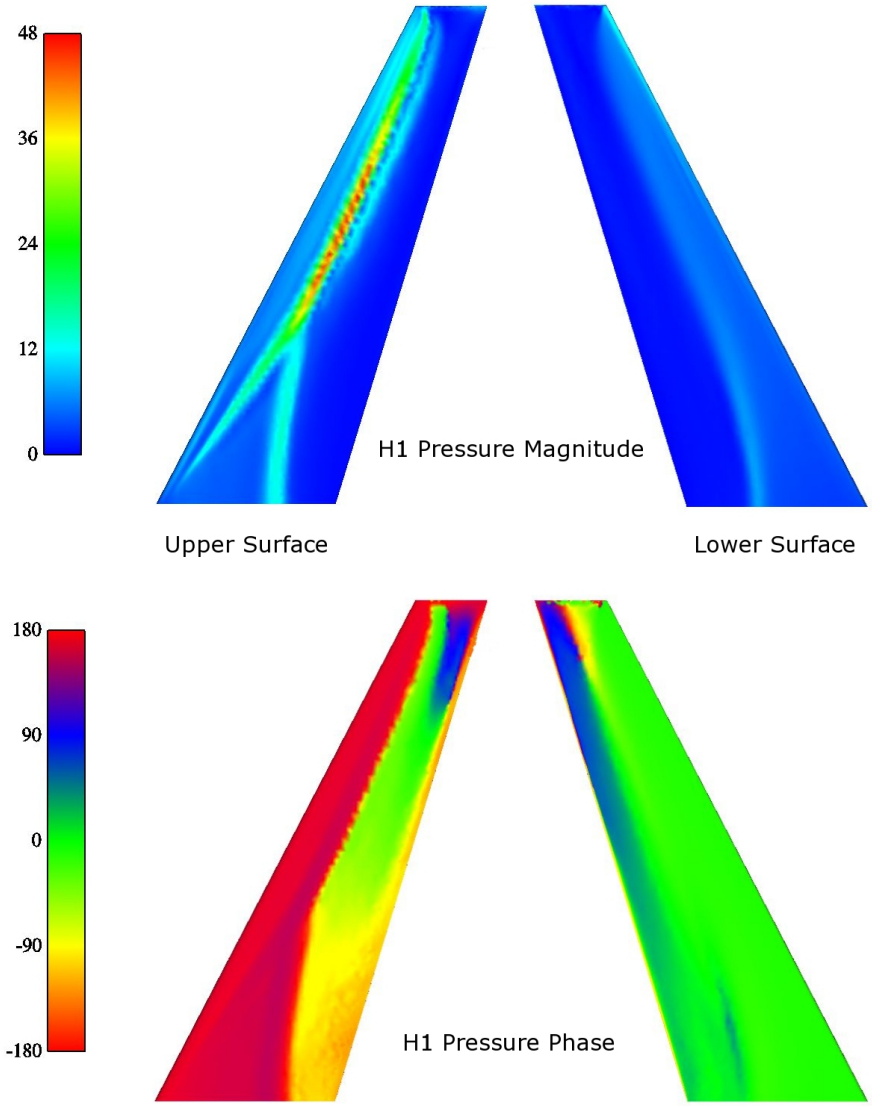


Figure 3.6: LANN-wing: Edge : H1 surface pressure distribution in phase-magnitude form

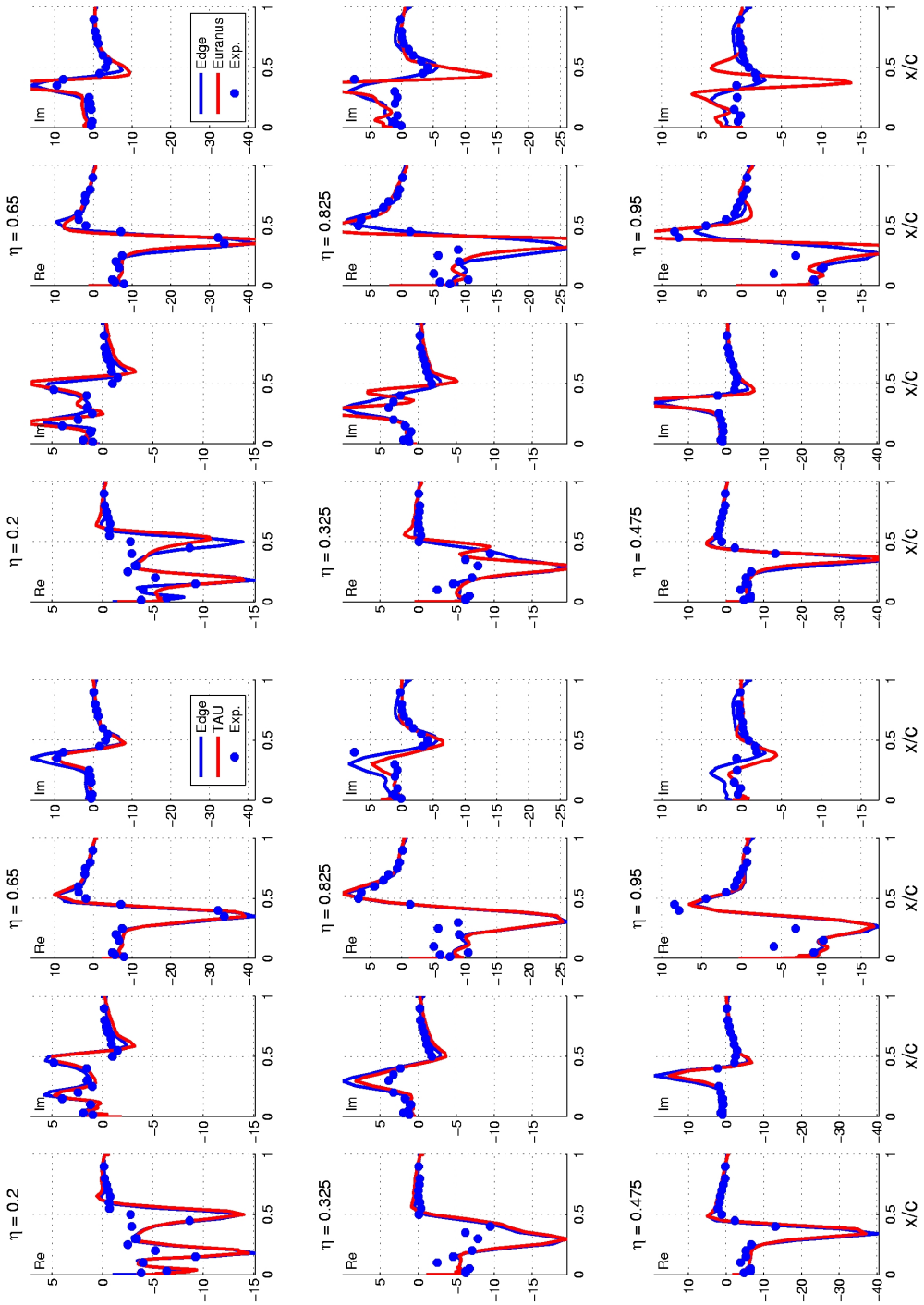


Figure 3.7: LANN-wing: Upper surface sections : H1 unsteady pressure : Edge vs TAU and experiment

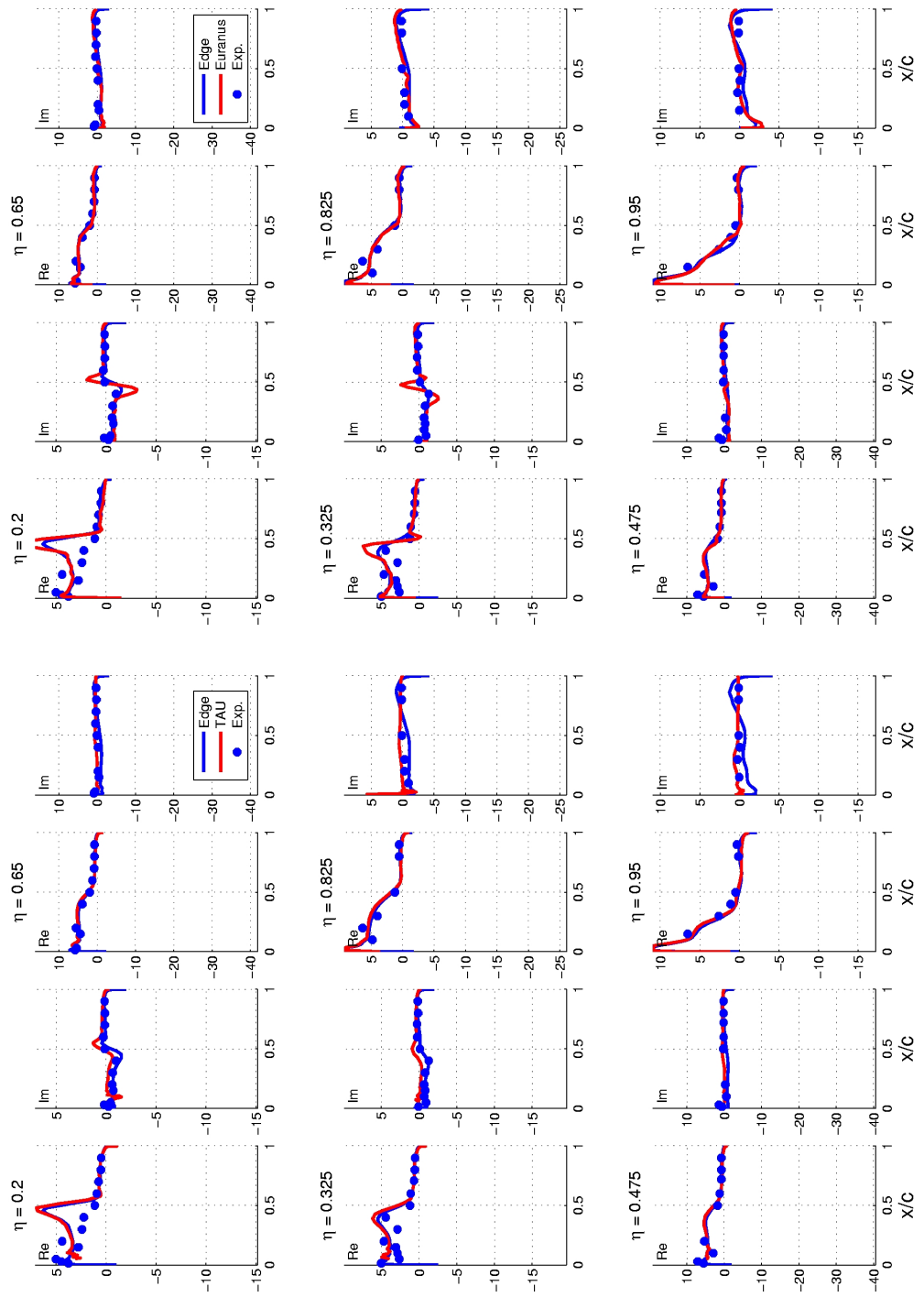


Figure 3.8: LANN-wing: Lower surface sections : H1 unsteady pressure : Edge vs TAU and experiment

4 Validation 2: MDO-wing

4.1 Background and Case Definition

Validation of any coupled aeroelastic simulation is complicated by the presence of three strongly interacting components: the CFD solver, the CSM solver and the coupling system. In a direct comparison with experimental data, it is in general difficult to isolate the effect of any one of these components. Also, due to the high cost and commercial sensitivity of transonic aeroelastic wind tunnel measurements, there is very little of such data in the public domain. It is therefore reasonable to validate Edge against a comparable numerical simulation. The chosen test case is the "MDO wing", a numerical model studied in the EU project UNSI, for which there are comprehensive published results [27, 28, 32].

The MDO-wing model represents a large, flexible aircraft wing, roughly comparable to that of a modern Airbus A380. The geometry is a simplified "exposed wing" with engine nacelles and pylons removed, with length 35.7 m (38.9 m span from fuselage centreline) and root chord 16.7 m. The structural model comprises a detailed wing box, with 3014 nodes and 1135 elements. Fuel and structural masses are included but there are no engine masses and no control surface freedoms. Modal coupled calculations are carried out using the first 18 elastic modes, with the structure rigidly clamped at the wing root. The geometry and structural model represents the unloaded "jig shape" as described in section 2.7.

Two flight conditions are considered: altitudes 7 km and 2 km at Mach 0.88 (ISA standard atmosphere). The assumed aircraft mass is 537 tons and the wing reference area is 725 m^2. To satisfy the conditions for level flight, the angle of incidence is adjusted, to achieve the required lift coefficient (0.3263 at 7 km altitude and 0.1686 at 2 km).

The dynamic simulation is started from a static-equilibrium solution and the structural excitation is introduced by setting a non-zero initial velocity for mode 1. This modal velocity is set to $\dot{q}_1 = 2\pi f_1 q_1$, where q_1 is the value of the modal coordinate equivalent to a vertical displacement of 1 m at the wing tip.

Selected results from the UNSI study are reproduced in figure 4.1, where the graphs show the vertical displacement of a reference point near the wing tip. The four independent simulations are all performed using coupled, inviscid Euler codes and show broadly similar behaviour [28]. At 7 km altitude the wing is aeroelastically stable but at 2 km there is a flutter-like instability. However, there are significant discrepancies between the results, especially for the "flutter" case. It has been noted, [53], that the most probable cause for these discrepancies is the use of widely differing methods for spatial coupling. The bottom pair of graphs in figure 4.1 show results computed at Saab, using the code Euranus 5.3. Having access to the precise details of these calculations it is these results that form our primary validation reference.

The results presented in this chapter include a detailed comparison between coupled simulations performed using Edge and the published Euranus

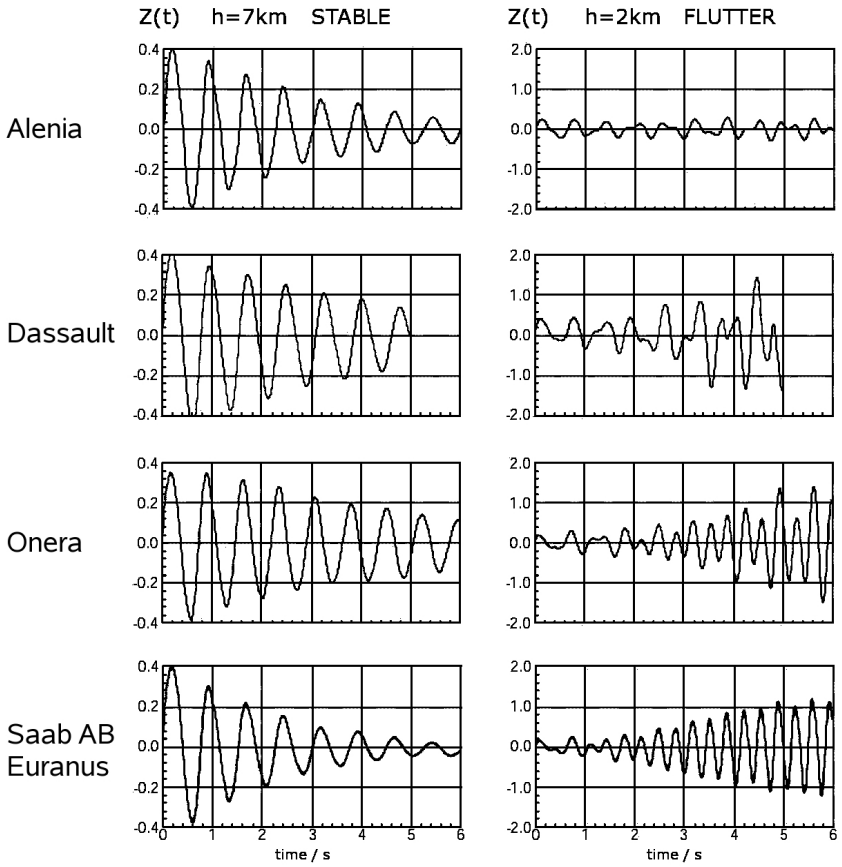


Figure 4.1: MDO-wing: vertical wing-tip response from UNSI simulation data.

computations. To enable a meaningful comparison of the two codes, the Edge computations have been carried out using the same modeshape data and closely matched spatial coupling. The Edge and Eurnaus computations are otherwise completely independent. The unstructured mesh used with Edge was generated from the UNSI geometry data, using ICEM-CFD and has 247,469 nodes. The earlier Euranus computations were performed using a single block, structured mesh with 106,425 nodes.

4.2 Spatial Coupling and Normal Modes

For a modal system, as explained on page 14, the only spatial coupling operation required is the transformation of modeshapes from the structural model to the CFD surface grid. Following the Saab calculation, the simplified, UNSI modeshape data has been used, in which the modes are represented as pure, vertical displacements on a set of "control points" sampled from the structural grid. These modeshapes were transformed onto the surface grid using the Edge program `aedbelast` with the *polyfit* interpolation method. The control points and interpolation set-up and the resulting surface modeshapes are shown in figures 4.2 and figure 4.3 respectively.

Figure 4.2 shows the points of the aerodynamic surface and structural grids together with the internal control points and the regions for the *polyfit* piecewise polynomial interpolation. The polynomial-fit regions form a chain of overlap-

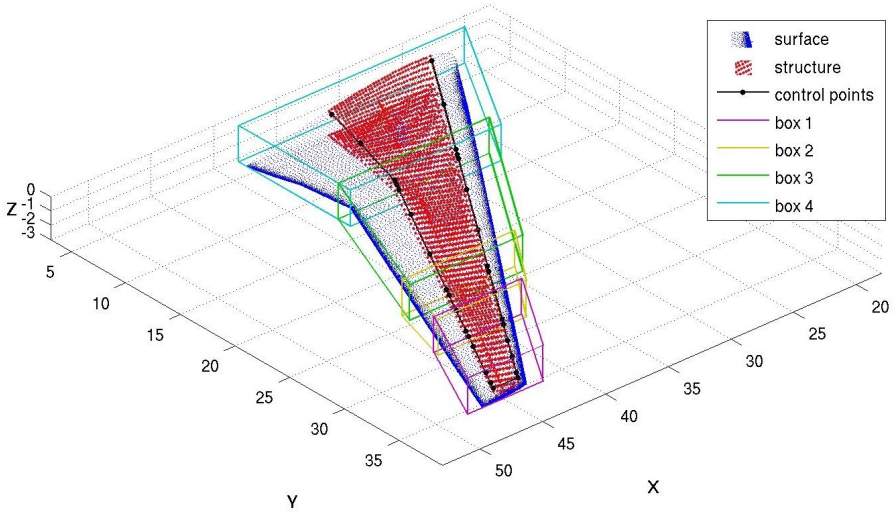


Figure 4.2: MDO-wing: Control points and boxes for `aedbelast` modeshape interpolation

ping boxes enclosing the whole wing. The control points are shown linked to provide a simplified representation of the internal wing-structure.

Figure 4.3 shows the surface modeshapes resulting from the *polyfit* interpolation. The point-cloud of the unstructured surface mesh (blue) is superimposed on the equivalent data for the Euranus structured mesh (red). It can be seen that there is very close agreement between the two sets of modeshapes. This at once validates the interpolation program, `aedbelast` and demonstrates that the spatial coupling used here for Edge is closely matched to that used for the earlier Euranus computation. In both the UNSI Euranus computation and the present Edge simulation, simplified "Z-only" modeshapes are used. For consistency with Euranus the modal forces are therefore computed using the undeflected surface normals (see page 14).

4.3 Coupled Simulation: Static Equilibrium

With Edge, modal, static-coupled simulations are carried out by setting the damping ratio (equation 2.5) of all modes high enough to remove any oscillation (using the parameter `default_damping` in the modal parameters file `Edge.amop`). This stabilises the time-integration scheme (section 2.5) thus allowing the choice of an arbitrarily coarse real timestep. The static equilibrium state is then obtained in a few iterations, at a computational cost comparable to that of a conventional, steady-state solution. Using this procedure, static equilibrium solutions were obtained for a few incidence cases and the target C_L condition was located by a half-interval search. Results for these static-coupled simulations are presented in figures 4.4 and 4.5 and table 4.1.

Figure 4.4 shows, the static displacement of the internal, structural control points, comparing the results of the Edge and Euranus computations. These results were computed from the modal coordinates and the modal definition data using the Edge Matlab program `mortors`. The largest structural deflection occurs for the high altitude case. For both flight conditions, however, the maximum displacement occurs at the control point nearest the wing-tip

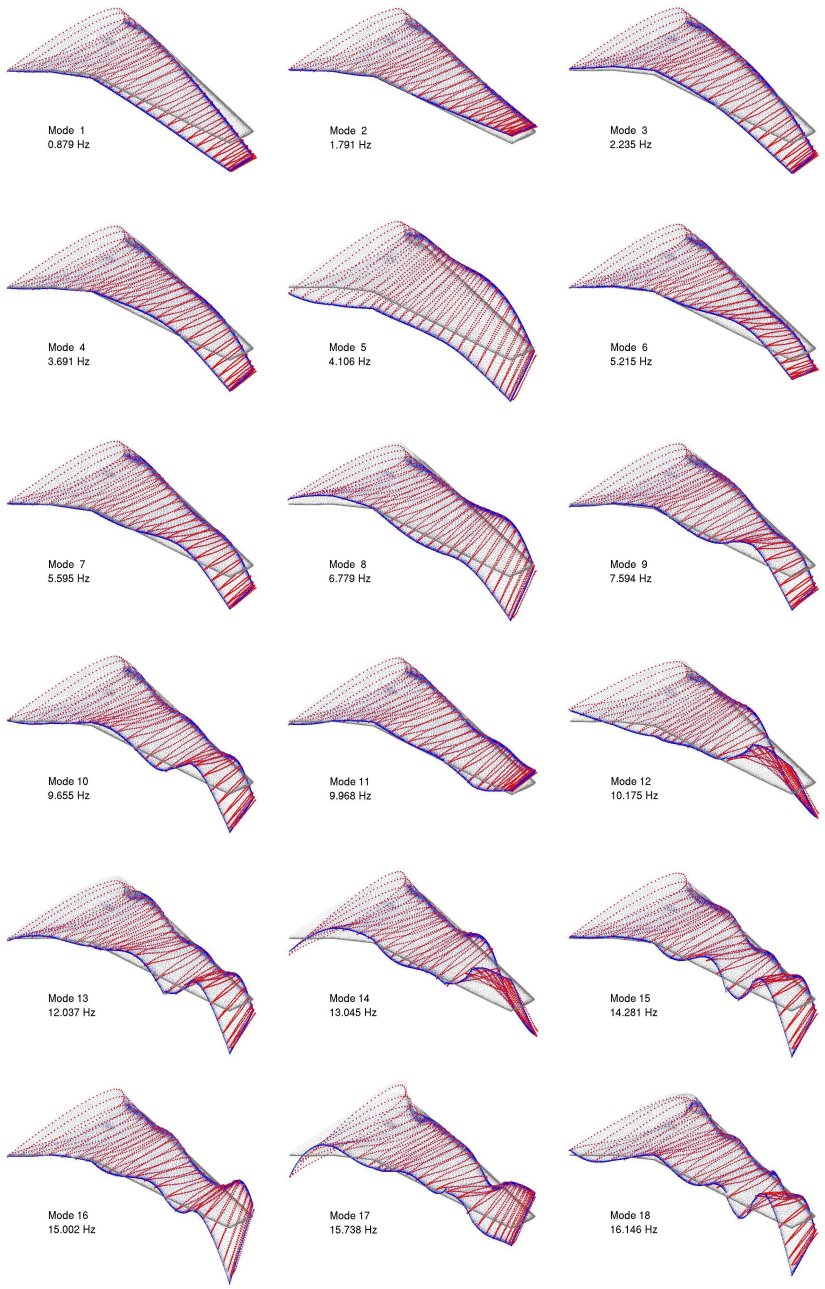


Figure 4.3: MDO-wing: Surface modeshapes for Edge (blue) and Eurnaus (red)

trailing-edge. This node[1] is therefore selected as a common reference point. Most importantly, it can be seen that for both flight conditions, the deflected shapes from the Edge and Euranus simulations are in very close agreement. Taken together with the results of the interpolation comparison, figure 4.3, this implies that there is also close agreement in the static surface shapes. This means that the Edge and Euranus dynamic simulations start from closely matched initial states.

Table 4.1 shows the values of the incidence, α° and lift coefficient, C_L for the two reference flight conditions, together with the, static, vertical displacement of the tip reference point. The results from Edge static-coupled simulations are compared with the results of equivalent computations with the rigid geometry and with the UNSI Euranus data. The Edge static-coupled results show, as expected, a sharp reduction in the lift coefficient compared to the values for the rigid geometry. The load-relief factor is 0.3 at h=2 km and 0.5 at h=7 km. The static-coupled results for both codes are with 0.1% of the target C_L, though there is less than exact agreement between the computed incidence angles.

Figure 4.5 shows surface-pressure and displacement for the two Edge static-coupled solutions at the reference flight conditions, together with the equivalent results for the rigid geometry. For both conditions the wing shows an upward displacement increasing towards the wing tip which results in a sharp reduction of the surface presure loading and a forward shift of the shock position. The wing displacement is largest for the higher altitude (7 km) case, which is consistent with the higher incidence angle required to achieve the required lift.

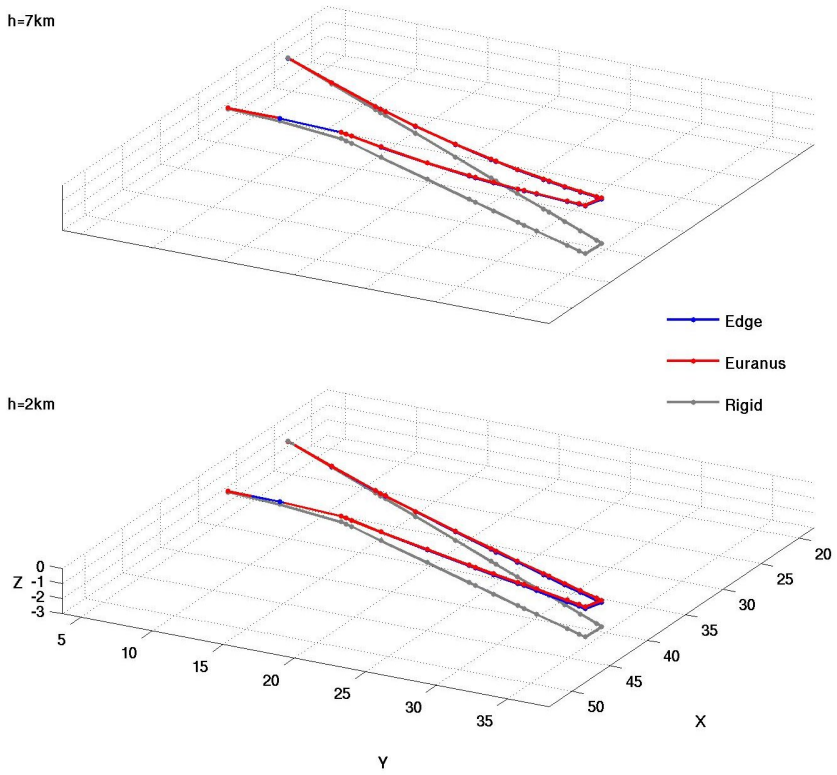


Figure 4.4: MDO-wing: Static-equilibrium position of the structural control points (Edge-Euranus comparison)

[1]Wing-tip reference point: (X,Y,Z) = (47.835,37.695,-0.166)m (structural node)

	h=2 km			h=7 km		
	C_L	α°	ΔZ/m	C_L	α°	ΔZ/m
TARGET	0.1686	-	-	0.3263	-	-
Edge	0.1685	-0.754	1.887	0.3274	+0.074	3.194
Edge rigid	0.5610	"	0.000	0.6486	"	0.000
Euranus	0.1688	-0.649	2.015	0.3266	+0.248	3.235

Table 4.1: MDO-wing: Lift coefficient and incidence angles for static equilibrium

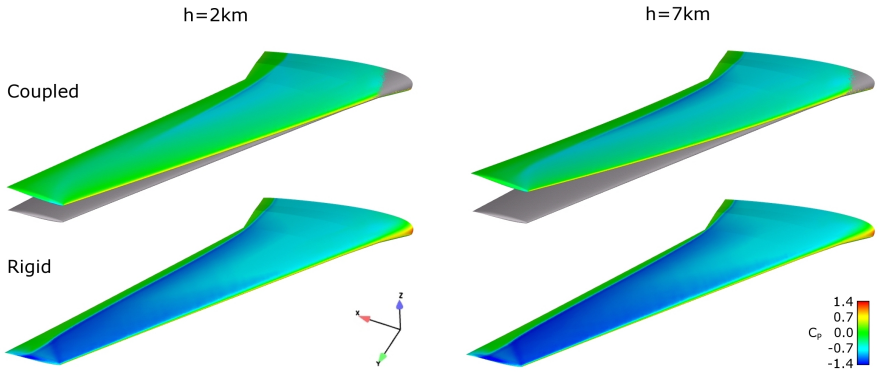


Figure 4.5: MDO-wing: Steady-state surface pressure distributions at target C_L, α

4.4 Coupled Simulation: Impulse Response

4.4.1 Edge-Euranus comparison

Impulse response simulations were carried out by restarting from the previous, static equilibrium solutions and applying the UNSI-specified initial conditions. These time-dependent solutions were obtained using a physical timestep of $\Delta t = 5$ ms, equivalent to 12 points per cycle of the highest structural mode and a constant 20 inner iterations. For the published Euranus computations the equivalent values were $\Delta t = 0.01$ s and 10 iterations. With Edge, adequate convergence of the C_M residual was typically achieved within 15 inner-iterations or less. The results of the Edge and Euranus computations are presented in figures 4.6 to 4.8 and tables 4.2 and 4.3.

Figure 4.6 shows the vertical displacement response at the wing tip structural reference point compared with the equivalent results from the UNSI Euranus computation. For both flight conditions the first 6 s of the time-domain response is shown, together with the modulus of its Fourier transform. The points marked on the time-domain plots indicate the maximum displacement amplitude. From this figure, it is clear that there is very close agreement between the two codes. This is especially true for the stable, 7 km altitude case, where the response is characteristic of a single degree of freedom oscillator with a natural frequency of approximately 1.3 Hz and positive damping ($\zeta = 0.055$ from the UNSI study). For the 2 km case, both codes predict a flutter-like instability with a frequency of approximately 2.8 Hz. The two response curves are almost identical for the first 1.6 seconds and there is close agreement in both the frequency of the emerging oscillation and its rate of growth, though the Euranus code predicts a slightly larger response-amplitude.

Figure 4.7 is similar to figure 4.6 but shows only the Edge results, with two simulations extended to 12 s duration. The behaviour of the stable, 7 km altitude, case is, as expected, a continued, simple exponential decay. In contrast to this, the oscillation in the unstable case, h=2 km, does not show the unlimited, exponential growth predicted a linear flutter model. The oscillation instead stabilises at a constant amplitude and must therefore be classified, not as a "classical flutter" but as a limit cycle oscillation (LCO). Continuation of the Euranus simulation has produces a similar result. In practice, a sustained oscillation of this magnitude, about 45 g at the reference point, would probably cause a catastrophic structural failure within a few cycles.

For the h=2 km case, the amplitude spectrum is dominated by a sharp peak at the fundamental frequency of the LCO, 2.93 Hz, and there are two further peaks at 5.85 and 8.69 Hz which can be identified as the second and third harmonics. This existence of the LCO and it's non-sinusoidal form are characteristic of the action of non-linear aerodynamic forces.

Numerical values from the Edge and Eurnaus simulations are given in tables 4.2 and 4.3. Note that the higher harmonics in the LCO are detected in the first 6 s of the Euranus simulation but not in the equivalent Edge data. This is probably because, due to the lower amplitude of the emergent LCO response from Edge, this part of the spectrum is obscured by the initial impulse response.

The preceding figures and analysis in this section apply only to the motion of the wing-tip structural reference point. The dynamic response is of course distributed over the whole wing. Fortunately, for both flight conditions, the principal response is a clean, low frequency oscillation with well-defined extrema. This allows the response from the two codes to be compared at known, equivalent time-points. Using same procedure as for the static computation (section 4.3) the instantaneous deflected shapes of the internal control points have been computed for the displacement-extrema shown in figure 4.6. The results are shown in figure 4.8. Once again, as in the single pont analysis, there is very close agreement between the two codes, especially for the stable, h=7 km, case. For the LCO case, h=2 km, the Euranus results show, as expected, a slightly larger amplitude. However, more importantly, these results show that, for both flight conditions, the two codes produce the same dynamic response shape.

The single-point analysis presented in figure 4.6 compares the response predicted by Edge Euranus directly in the time-domain and as derived amplitude spectra. The reconstructed, instantaneous structural shapes shown in figure 4.8 compare the response in real space. Both representations show very close agreement between the two codes and from this we may conclude that both simulations quantitatively predict the same aeroelastic phenomena.

4.4.2 Edge Energy Analysis and Surface Pressures

Using the procedure described in section 2.9 structural energy timeseries have been computed from the Edge modal coordinates response data. These results are shown in figures 4.9 to 4.10.

Figure 4.9 shows the total structural energy timeseries for the two flight conditions for the full 12 s of the simulation. For the stable, h=7 km, case the total-energy series closely matches that of the reference point displacement series shown in figures 4.7 and 4.6. After the impulsive excitation at t=0 the total energy oscillates as the wing surface moves alternately with and against the aerodynamic forces and decays back to the initial static equilibrium level. For the LCO case, h=2 km, the initial excitation response is followed by higher frequency oscillation which increases in amplitude together with the total en-

ergy. The system stabilises, at about t=8 s, in an uniform oscillatory cycle with the mean total energy approximately 2.5 times that of the initial, static strain-energy.

Figure 4.10 shows total energy series for the LCO case, h=2 km, together with the energy rate error defined in equation 2.23. Note that the first derivative of the total energy here is evaluated as a centered difference consistent with the internal discretisation scheme. With the exception of the initial excitation point, the error $\xi(t)$ is roughly two orders of magnitude smaller than the corresponding total energy rate $\dot{E}(t)$. From the scheme definition 2.12 it is easily shown that, for the general case, the RMS value of $\xi(t)$ is proportional to the physical timestep, Δt. This consistent with the behaviour shown in here.

In figure 4.11 the structural energy timeseries for the two flight conditions is shown resolved into to modal components, as defined in equations 2.21. For the stable, h=7 km, case the energy response is almost exclusively in the first structural mode. This is consistent which the behaviour observed in the displacement responses which are characteristic of single degree of freedom system. For the LCO case, h=2 km, the response is much more complicated and indicates that true the "flutter" mechanism can only be captured by the inclusion of at least the first eight structural modes. There is very little activity in the higher modes and only very weak response in mode 2. Inspection of the modeshapes from the full structural model shows that this is predominantly an in-plane mode. This is, however, not apparent in the simplified, "Z only" modeshapes used here.

The efficient visualisation of a time-dependent surface pressure field together with a complicated surface motion, without using animation, presents a slight challenge. In contrast to the stable case, the surface motion for the LCO is not a simple, single degree of freedom oscillation and is not sinusoidal. It is therefore not appropriate to select time sample points based on the displacement of an arbitrary structural point. The structural energy, however, represents the state of the entire wing. We therefore sample the solution, over a single cycle of oscillation, at consecutive extrema of the kinetic energy.

In figure 4.12 the time-evolution of the surface-pressure field is shown for the two flight conditions, together with the corresponding four sample points in the energy time-history. For each surface-pressure display, the four images are in clockwise order, with the top left and bottom right frames showing respectively the states with maximum and minimum elastic potential energy. The surface displacements are shown at a scale of 1:1. For the stable, h=7 km, case, the first cycle of the damped oscillation shows a small surface displacement, in the form of the first bending mode, with a modest variation in the surface pressure field. For the h=2 km case, the fuly-developed LCO shows a more complex surface motion with larger displacement amplitude and a much stronger variation in the surface pressure. The largest departure from the equilibrium pressure is seen at the extrema of the elastic energy.

4.5 Summary

The Edge functions for modal-coupled aeroelastic simulation have been demonstrated for the UNSI MDO-wing test-case. Using matched spatial coupling, the simulation results from Edge and Euranus are in very close agreement, contrasting sharply against the large discrepancies observed between the other codes in the UNSI study. Analysis of the modal energy timeseries shows that the coupled time-integration scheme in Edge is, to the expected accuracy, conservative and the modal coupled simulation function produces results which are consistent with those of independent, equivalent computations.

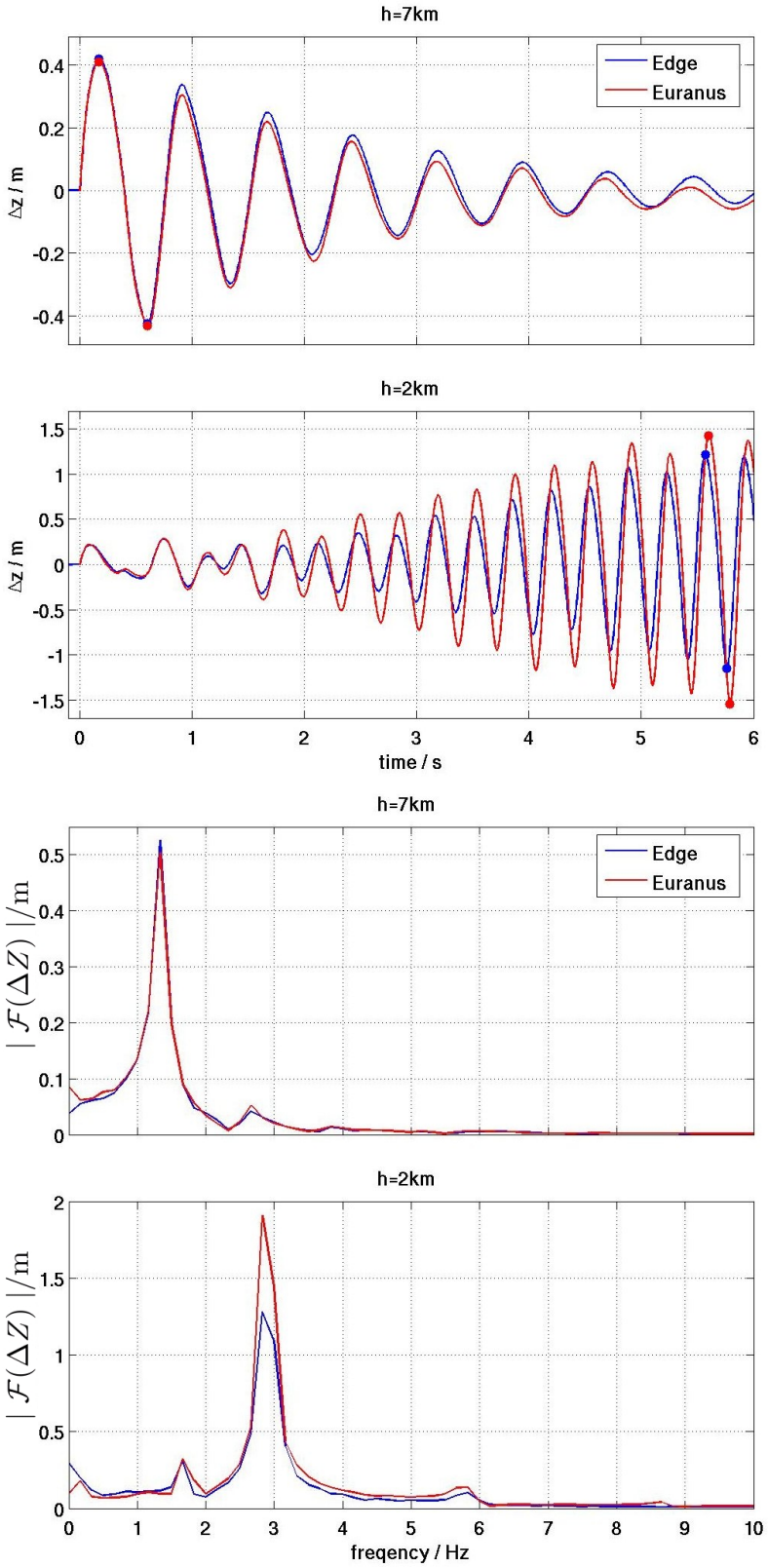


Figure 4.6: MDO-wing: Impulse response Z-displacement at the wing-tip structural reference point (Edge-Euranus comparison)

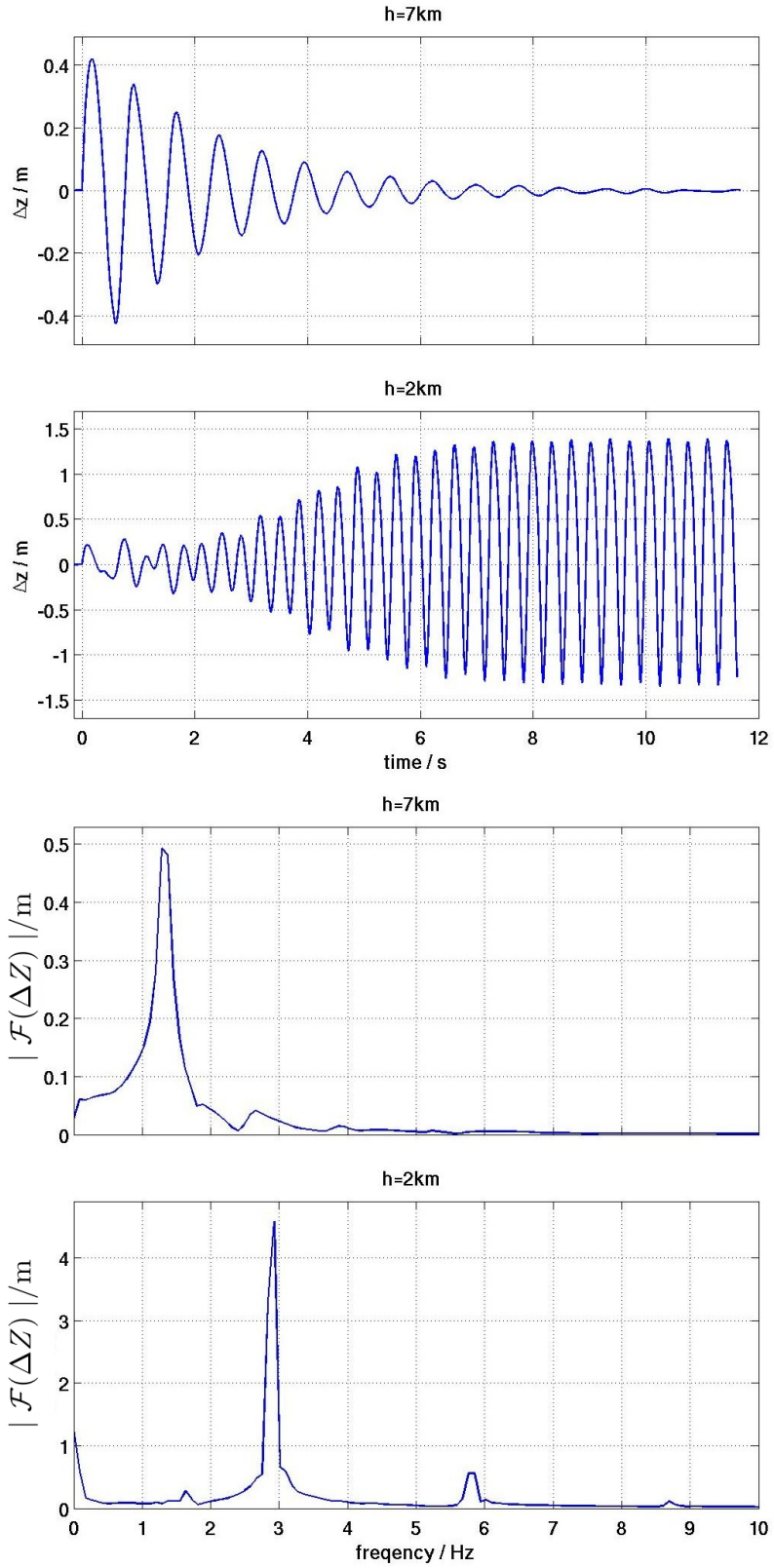


Figure 4.7: MDO-wing: Impulse response Z-displacement at the wing-tip structural reference point (Edge)

condition	data source	ΔZ_{max}/m	t_{max}/s	ΔZ_{min}/m	t_{min}/s
h=7 km	Edge 0-6s	0.419	0.170	-0.424	0.600
	Eurnaus "	0.409	0.170	-0.430	0.600
h=2 km	Edge 0-12s	1.391	10.400	-1.350	10.250
	Edge 0-6s	1.219	5.570	-1.151	5.760
	Eurnaus "	1.431	5.600	-1.544	5.790

Table 4.2: MDO-wing: Impulse-response Z-displacement extrema for the wing-tip structural reference point (see figures 4.6 and 4.7)

	Edge 0-12 s		Edge 0-6 s		Euranus 0-6 s	
h	f/Hz	$\mid \mathcal{F}(\Delta Z) \mid$/m	f/Hz	$\mid \mathcal{F}(\Delta Z) \mid$/m	f/Hz	$\mid \mathcal{F}(\Delta Z) \mid$/m
7 km	1.28	0.4933	1.33	0.5260	1.33	0.5050
	2.74	0.0369	2.66	0.0414	2.66	0.0517
2 km	1.64	0.2745	1.66	0.3064	1.66	0.3234
	2.93	4.5790	2.83	1.2813	2.83	1.9105
	5.85	0.5619	5.82	0.1023	5.82	0.1366
	8.69	0.1125	-	-	8.65	0.0419

Table 4.3: MDO-wing: Maxima of the amplitude spectra for the impulse response at the wing-tip structural reference point (see figures 4.6 and 4.7).

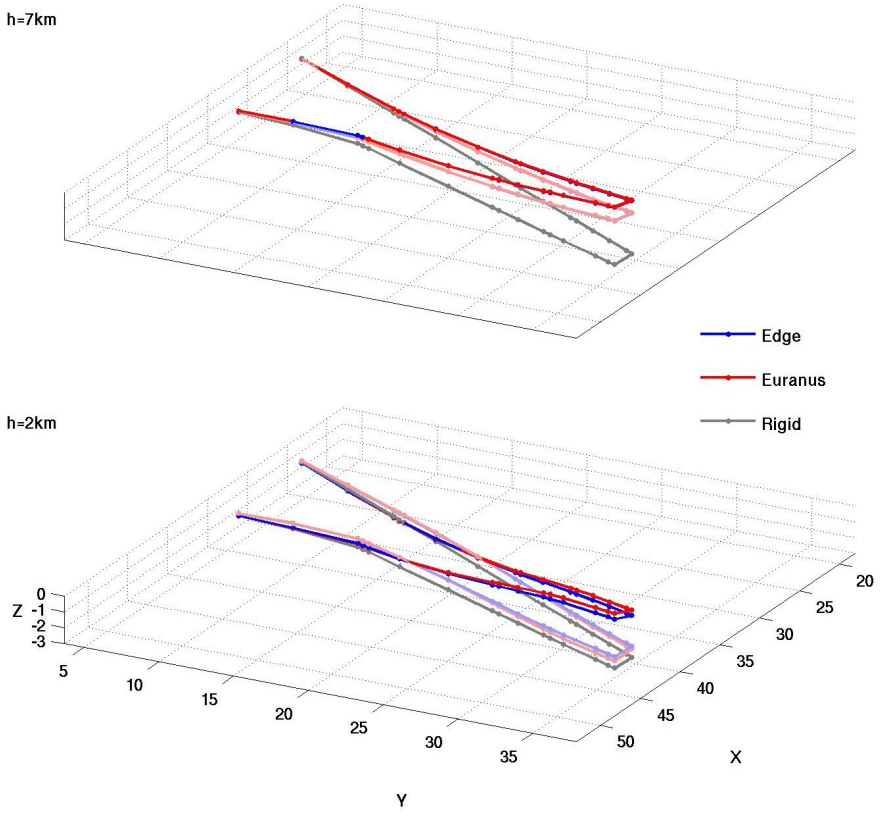


Figure 4.8: MDO-wing: Instantaneous position of structural control-points for displacement-extrema of the wing-tip reference point (Edge-Euranus comparison)

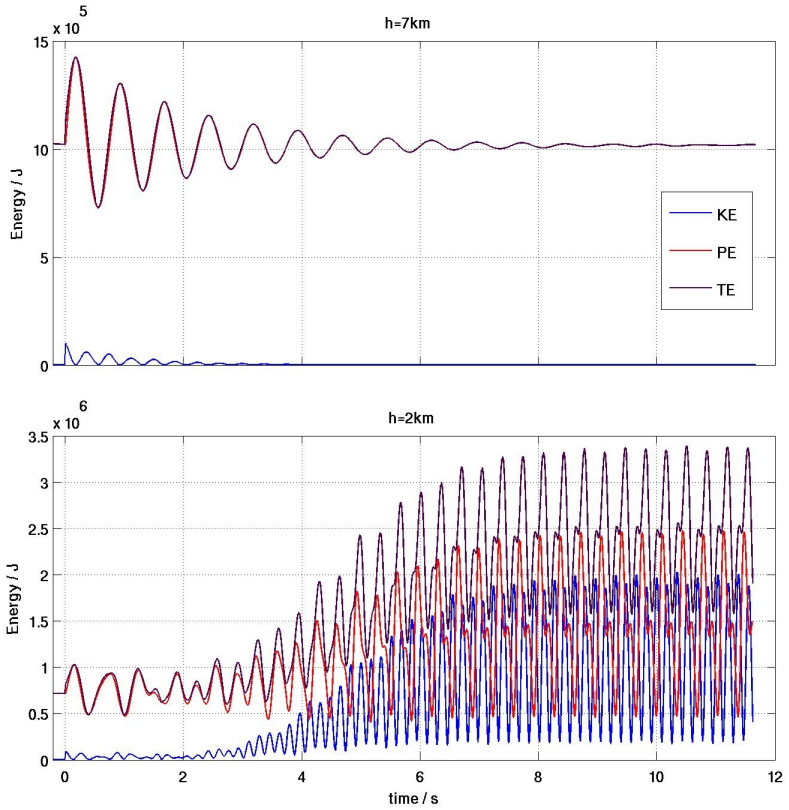


Figure 4.9: MDO-wing: Total Structural Energy timeseries (Edge)

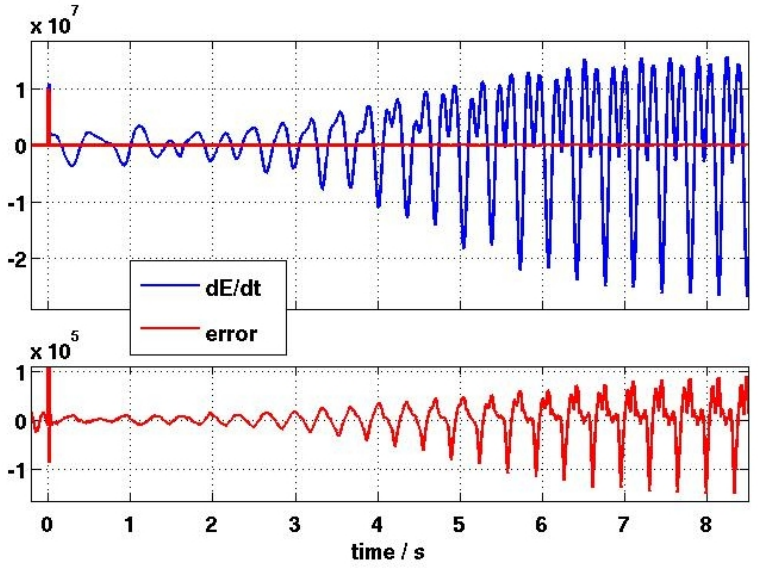


Figure 4.10: MDO-wing: Energy rate error $\xi(t)$ for h=2 km response (Edge)

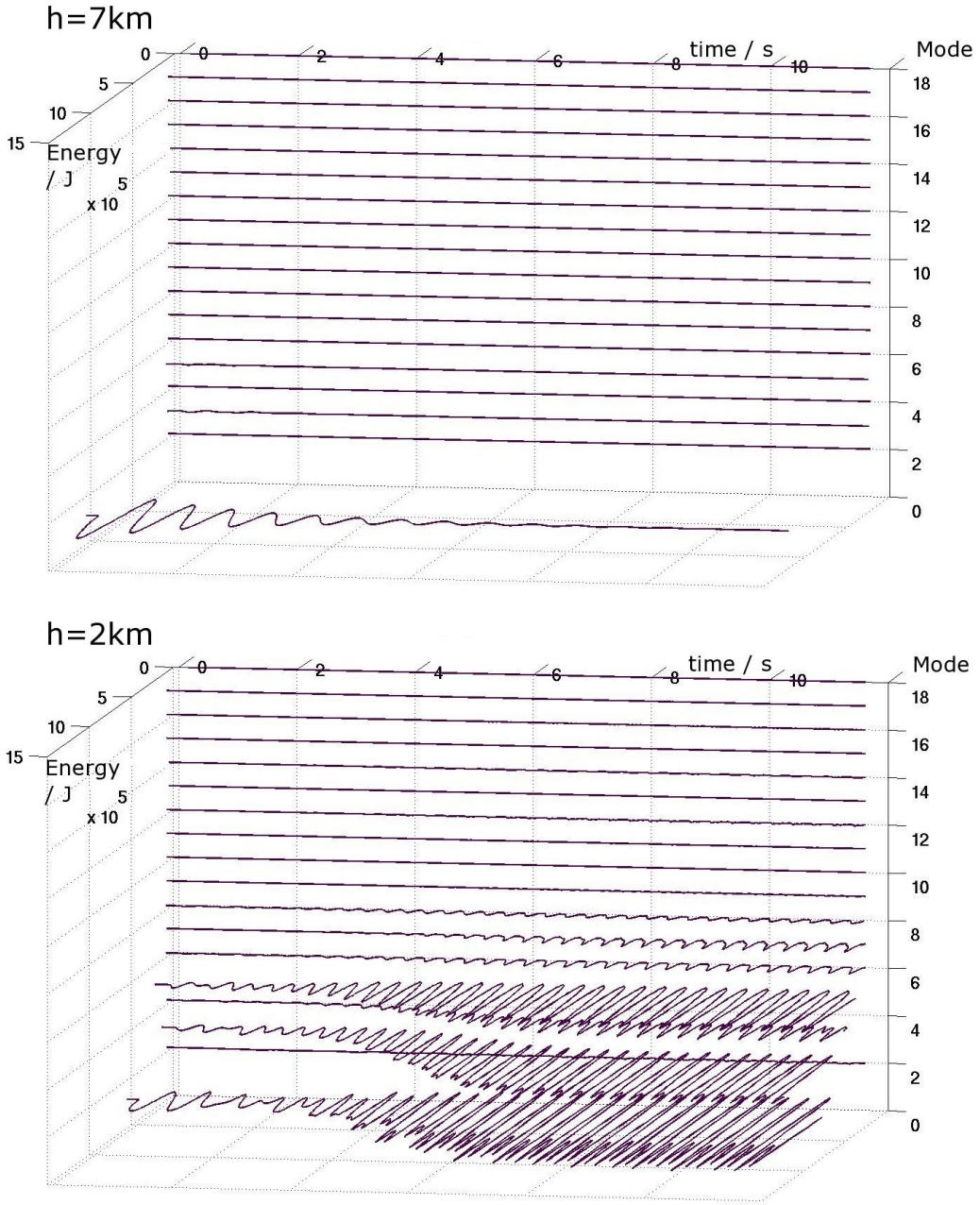


Figure 4.11: MDO-wing: Modal decomposition of the Structural Energy timeseries (Edge)

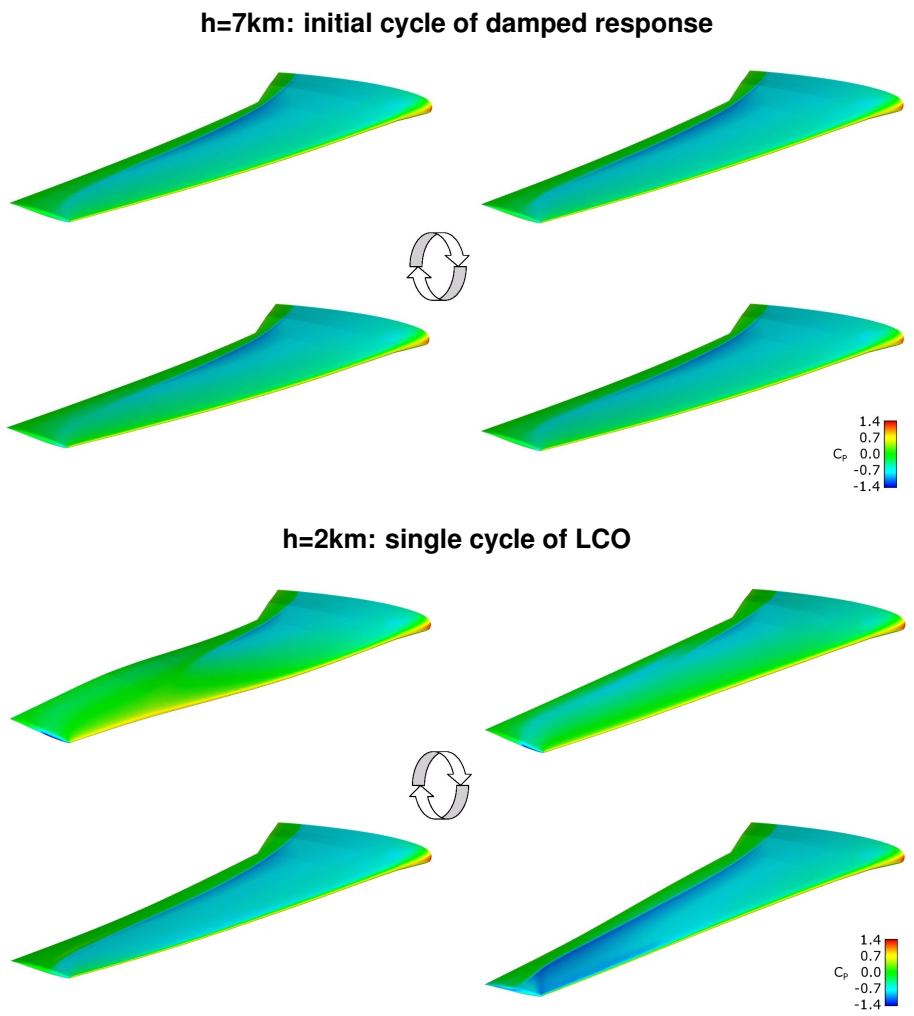


Figure 4.12: MDO-wing: Instantaneous surface shape and pressure-distribution at consecutive extrema of the structural Kinetic Energy for a single cycle of the principal oscillatory response (flight conditions h=2 km and h=7 km)

5 Conclusions and Recommendations

5.1 Conclusions from Validation Studies

The Edge aeroelastic code has been validated for two published test-cases: the LANN-wing rigid-pitch experiment (chapter 3) and coupled simulatons of the MDO-wing from the UNSI project (chapter 4). Based on these two studies, we present the following conclusions.

Validation 1: LANN-wing

In the LANN-wing study, an unsteady RANS simulation with Edge is compared with the experimental data and with results both from the TAU code, using the same, unstructured grid and from earlier work using the structured code, Euranus. The Edge results are in very close agreement with the experimental data, with an accuracy at least as good as the equivalent TAU computations.

The results of the LANN-wing validation study are certainly very positive. However, in one respect, this is a relatively undemanding case. The surface motion is of extremely small amplitude and has a low reduced frequency (table 3.1). Under these conditions, the movement of the mesh and boundary and its perturbation of the flow are both approximately linear effects. For a more thorough validation of the code, further work is required, using a similar test-case with larger amplitude oscillation. In any such an investigation, it would be useful to have pressure data in the time-domain, since this would enable a much more accurate comparison with the simulation results.

Within the limits of the test-case, the LANN-wing results show that, in Edge, the combination of unsteady-RANS and mesh-deformation is correctly implemented. Furthermore, since this rigid-pitch simulation is performed using the Edge modal, prescribed motion function, the results also validate the main data structures used for aeroelastic simulation.

Validation 2: MDO-wing

Using the MDO-wing model, the Edge functions for modal-coupled simulation have been validated, for Euler flow, against a published, comparative study of several CFD codes, including Euranus. The Edge simulations were performed using surface modeshapes closely matched to those of the earlier, Euranus computation, thus providing equivalent spatial coupling.

The simulation results from Edge and Euranus show very close agreement, both in the spatial displacement of the wing and its time-evolution. Furthermore, the Edge and Euranus simulations, which use matched surface modeshapes, show greater consistency than those of the other coupled simulations in the UNSI study. This provides strong evidence in support of the hypothesis [53] that the main cause of the discrepancies between the UNSI simulations is not the variety of Euler flow-solvers but the use of widely different spatial-coupling schemes. The MDO-wing case is particularly sensitive in this respect

due to the large separation between the structural and aerodynamic grids (see figure 4.2). This is, however, a common problem in aerospace applications.

Comparison of Edge with the UNSI study has shown that the implementation of modal coupling is consistent with that used in Euranus. It also highlights the sensitivity of such simulation methods to the choice of spatial coupling. However, since there is no experimental reference for the MDO-wing case, it is not possible to assess the absolute accuracy of the Edge simulations, or indeed any of the other UNSI results. To provide a more thorough validation of Edge for coupled, aeroelastic simulations, it must be evaluated against an experimental test case.

From the results of the structural energy analysis and the close agreement achieved between the Edge and Euranus simulations, we conclude that the modal coupling functions in Edge are correctly implemented and that, at least for Euler flow, the code is reasonably accurate. On the basis of the LANN-wing study, we may extend this conclusion to coupled simulations with RANS flow.

5.2 Code Status and Functionality

Referring to the initial objectives for the Edge aeroelastic code (page 5) and given the results of the two validation studies, we may state that the present implementation has been successful. The aeroelastic functionality in Edge, version 3.3, matches and exceeds that previously available in Euranus 5.3. In contrast to the preceding Euranus aeroelastic code, the Edge system is self-contained, with a set of supporting programs for modal aeroelastic simulations including a powerful extension for the Matlab environment. Two additional, useful features which are new in Edge are its functions for defining arbitrary modal prescribed motion and force inputs and the ability to perform force-offset coupled simulations using an assumed flight-shape. Most importantly, the new code provides aeroelastic functionality with an unstructured CFD flow solver and hence, the capability to model systems with complex geometries.

Whilst the initial Edge aeroelastic code marks a major advance, it retains four features from its predecessor, Euranus, which are somewhat limiting.

1. non-parallel implementation
2. mesh-movement using modal, perturbation fields
3. spatial coupling using the *polyfit* interpolation method
4. restricted to modal structural models

This has been previously discussed in a requirements description for Edge [39] issued by Saab AB in 2004. We now present a brief account of these features in relation to the current status of the code.

Parallelisation The initial Edge aeroelastic code is a serial implementation and it is therefore limited to relatively small-scale simulations, as presented in this report. For example, using a single Intel P4 processor, one of the dynamic, MDO-wing, Euler cases takes about twelve hours per second of simulation time. For the RANS, rigid-pitch simulation of the LANN-wing, the run-time is about six days per period. Clearly, for any realistic applications, a scalable, parallel implementation is essential.

The Edge flow-solver system is purpose-built for parallel execution. However, the extension of this capability to deforming grids is far from trivial. In particular, the re-computation of the dual grid (section 2.2) must preserve

matching cell-surfaces on the partition boundaries and this requires additional element-information from the primary mesh. As noted in the introduction all the necessary modifications have now been completed. For this, the preprocessor program `part` has been completely re-written[1] and substantial changes[2] have been made to the main code. Subject to final testing and validation, the parallel implementation will be available in the next Edge release.

Mesh Movement The present mesh movement system is based on the perturbation field method and an external mesh deformation tool. This system can, in principle, be used for any system with a modal structural model. However, for applications requiring a high-resolution CFD mesh or a large number of modes, the memory requirement becomes a significant limiting factor (see page 17). To remove this restriction and to enable the use of more general structural models, a function for on-line mesh deformation is required. With such a function, Edge would also be easier to use, since there would be no need to generate and manage the, somewhat bulky, perturbation-field datasets.

Spatial Coupling The *polyfit* interpolation method in program `aedbelast`, which has been ported to Edge from Eurnaus provides useful continuity with the earlier code. For example, in the MDO-wing validation study, this has enabled a conclusive "like for like" comparison with Euranus using matched modeshapes. The *polyfit* method in can, in principle, be applied to more complex systems, such as a military aircraft wing with external stores. In practice, the method is limited by the difficulty of manually setting-up the interpolation definition (see section 4.2). This is further complicated by the piecewise-continuous form of the interpolant which, in the "box overlap" regions, can sometimes produce anomalous sharp edges and cusps in the surface shape.

Since it is possible to use any suitable interpolation tools to produce surface modeshapes, spatial coupling in Edge is not restricted to the *polyfit* method in the `aedbelast` program. This means that, for modal systems, alternative interpolation methods can be evaluated without changing the existing code.

Structural Model The present code is restricted to a modal structural model. This is adequate most dynamic simulations. However, for many static aeroelastic problems, or for more advanced, nonlinear structural models, a non-modal approach is required. The implementation of any such model, requires both mesh deformation and spatial coupling functions to be implemented on-line.

Summary of Code Status

The initial Edge aeroelastic code, in version 3.3.1, provides a complete and reasonably well-validated system for modal simulations with unstructured grids. The code has since been extended to a parallel implementation, which is at present under test. With the new, parallel code, it will be feasible to carry out modal-coupled simulations with unsteady RANS flow. This will probably meet the needs of most project applications in the near-term. However, for Edge to be used with more advanced structural models, improvements are needed to the code's functions for mesh-movement and spatial coupling.

[1] Edge Subversion revision 366, 2005-08-24, P. Eliasson
[2] Edge Subversion revision 387, 2005-09-26, P. Eliasson

5.3 Recommendations

The functionality and present status of the Edge aeroelastic code is described in the preceeding section. Based on that analysis, we present the following recommendations.

Parallelisation and Performance The parallel implementation of the present aeroelastic code has been completed, together with some basic functional tests. However, before this code is released, it must be validated to ensure full consistency with the serial code. No experimental reference is needed for this but the chosen test cases must include unsteady-RANS flow with large-amplitude mesh-deformations. These validation cases should be documented and made available for routine testing of the code in subsequent development.

The range of applications for the Edge aeroelastic code depends strongly on the performace of the CFD solver, especially when used with deforming meshes. One way to improve this would be to make solver compliant with the Geometric Conservation Law (see page 17). For Euranus, GCL-compliance was achieved using the Volume Discharge method [9]. Given the construction used for the dual mesh, it should be possible to use a similar method in Edge. However, the formulaton and implementation of this method is a significant challenge. As a first step, tests should be performed to assess the magnitude of GCL-related errors for the present flow-solver and a preliminary study of the formulation problem should be completed.

The performance of the Edge aeroelastic code will also be increased by two forthcoming enhancements the CFD code: the Recursive Projection Method (RPM) [49, 29] and a line-implicit solver [54]. The RPM has been shown to convergence rates in Edge by a factor of two or more [30]. The implementation of a line-implicit solver in Edge is predicted to yield a much larger performance increase, up to two orders of magnitude for some cases. It is planned that both RPM and line-implicit methods will be implemented in Edge during 2006. It is essential that these new features can be used with deforming meshes.

Mesh Movement On-line mesh-deformation in Edge would enable the aeroelastic code to be used with a much greater range of structural models and it could also make the code easier to use (see page 2.6 and the previous section). Similar functionality is also required for applications in aerodynamic shape optimisation, so there is a strong case for development. There are, at present, two feasible options for providing such a function and both of these rely on codes developed, within FOI, for use in aerodynamic shape-optimisation. We note that, whilst the present Edge package contains two programs which can be used for mesh deformation, `aedbelast` and `aetribend`, for practical reasons[2] neither of these codes is suitable for conversion to an on-line function.

Of the two optimisation-related codes, the one that can most directly be implemented in Edge is based on a modified Laplace smoothing method, broadly similar to that used in the program `aetribend`. However, this code uses a more general, edge-based formulation [8] and solves a discrete, Laplace equation with anisotropic diffusivities [3]. The code is quite well-proven and it uses the same, programming language and data structures as Edge. It should therefore be relatively straightforward to implement this function within Edge, though this would still be a sizable programming task.

[2]The program `aetribend` is an interface to components of TRITET, a code which has been developed independently from Edge and has a very different structure. Furthermore, the mesh-deformation program is limited, in 3D, to meshes comprising tetrahedra and triangular prisms. The spatial-coupling program `aedbelast` also has a mesh deformation function but this uses deflections of the structural control points and not the boundary surface.

The second code employs an innovative technique, using Radial Basis Functions, which offers several major benefits for gradient-based aerodynamic shape optimisation [36]. For aeroelastic applications, this method has two several striking advantages. Firstly, for RANS meshes, with a fine, prismatic boundary region, the deformation is much faster than an equivalent Laplace-smoothing operation, without requiring more memory or compromising mesh quality. Secondly, the same, geometric interpolation, code can also be used for spatial coupling interpolation [35]. This technique is quite new and the present code would not be so easy to port into Edge but these issues are strongly countered by the elegance of the method and its dual-purpose functionality.

Spatial Coupling Within FOI, the most immediately accessible and attractive means of improving the available spatial coupling in Edge would be to exploit the methods and code of the RBF-based mesh-deformation program described above. For an initial evaluation, this could be used in its present form, as a stand-alone program, to generate surface modeshapes. With slight modification, the code could also be used to produce an interpolation matrix of the form in equation 2.7, or an equivalent data structure, for on-line spatial coupling. However, to make full use of this method, including its fast mesh-deformation capability, it would need to be built-in to the main Edge code.

Implementation of the in-house methods [36] would provide a substantially improved spatial coupling function in Edge. However, for handling a large structural model, or one with multiple components, such as a high-lift system [60], more advanced methods are required. In particular, the interploation scheme must be both localised and scalable (which in principle means that the interpolation matrix is block-sparse). This can be achieved by using RBFs with compact support, together with a Partition of Unity method [65], an approach which has been successfully demonstrated with the TAU-code [1]. Whilst this method is both powerful and well proven, a usable implementation in Edge would require a substantial investment in code development.

The repeated construction of spatial coupling schemes for a large, detailed, multi-component systems ultimately requires a purpose-built graphical tool. This need was recognised within the TAURUS-project and a protype application, FSCON[1] (fluid-structure connector) was developed by SMR[2] as part of a simulation environment called "FSI". This software, which was used by Saab AB and other TAURUS partners, has since been further developed by SMR and includes advanced, spatial coupling methods similar to those described above. The FSI software, which is supplied as source, could be fully integrated with Edge without requiring any major program development work by FOI.

Structural Model Given the availability of on-line mesh deformation and suitable tools for spatial coupling, Edge can, in principle be coupled to any structural model. Direct coupling to an external CSM code requires, however, a specific interface and for many commercial codes, no such interface is available. A more accessible, intermediate solution is to implement a solver in Edge, based on equation 2.1 and import the (large, sparse) mass, stiffness and damping matrices. This can be done using existing code in the FOI-developed CSM solver Stripe. The required Stripe code would be added Edge as a separate module for optional compilation. With suitable conversion programs, structural matrices could be imported from any CSM code including Stripe and NASTRAN.

To develop a facility for aeroelastic simulation with nonlinear structural models, the most direct route is again to exploit the resources of the existing

[1] FSCON: Specified by J. Smith (FOI) and A. Karlsson (Saab AB)
[2] SMR Engineering & Development `www.smr.ch`

Stripe code. This would require the development of a specific interface for Stripe. For this, however, FOI has the advantage of retaining both the source code for Stripe and its most experienced developers.

Validation One major factor which has limited validation of the present Edge aeroelastic code is the need for very long run-times. Given the availability of a parallel implementation, this problem is greatly reduced. Following completion of consistency checks of the new, parallel code, further validation studies should be completed. This work must at least include dynamic coupled simulations of the AGARD 445.6 elastic wing [71]. However, since this standard case does not include a detailed structural model, additional experimental references are required, in particular to evaluate alterative coupling techniques. Ideally, further validation studies should be carried out as part of collaboration in a dedicated experimental programme.

Summary of Recommendations

The initial Edge aeroelastic code has been throughly tested and validated. As an immediate priority, this validation must be extended to the parallel code.

On-line mesh deformation is a basic requirement for any further development of Edge as an aeroelastic code. At least one of the two identified mesh-deformation codes should be implemented in Edge, within the next year.This must be coordinated with related work for aerodynamic shape optimisation.

To enable the efficient and accurate simulation of systems with widely separated grids, improved methods and tools for spatial coupling should be made available for Edge. As an initial step the new, in-house methods should be evaluated, using existing code. At the same time, a detailed evaluation should be also carried out assessing the SMR software used together with Edge.

Any development of Edge for use with non-modal structural models is dependent on the availablity of on-line mesh deformation. The suggested modifications based on using the, FOI-developed, Stripe code must therefore be postponed until this functionality is available.

The parallel Edge aeroelastic code is, subject verification, capable of realistic, dynamic aeroelastic simulations, with 3D unsteady RANS flow, using unstructured grids. This is a major technological achievement. It is therefore strongly recommended that this new simulation capability is demonstrated in a suitable project application, at the earliest opportunity.

Bibliography

[1] A. Ahrem R., Beckert and H. Wendland, *A new multivariate interpolation method for large-scale spatial-coupling problems in aeroelasticity*, Proc. International Forum for Aeroelasticity and Structural Dynamics, IFASD, June 2005.

[2] E. Albano and W. P. Rodden, *A doublet-lattice method for calculating lift distributions on oscillating surfaces in subsonic flows*, AIAA Journal **7** (1969), 279–285.

[3] O. Amoignon, J. O. Pralits, A. Hanifi, M. Berggren, and D. S. Henningson, *Flap design optimization for take-off performance*, Evolutionary and Deterministic Methods for Design Optimization and Control with applications to Industrial and Societal Problems, EUROGEN 2005 (R. Schilling, W. Haase, H. Periaux, H. Baier, and G. Bugeda, eds.), FLM, Munich, 2005.

[4] M. Andersson, *Derivation of equations used by the computer program SIGGE for calculating IR-intensity*, Tech. report, FOI, August 2002, FOI-R-0553-SE.

[5] M. Andersson and I. Karlsson, *SIGGE v1.0b users guide*, Tech. report, FOI, August 2002, FOI-R-0554-SE.

[6] R.S. Battoo, *An introductory guide to the literature in aeroelasticity*, Aeronautical Journal **103** (1999), 511–518.

[7] A. Beckert and H. Wendland, *Multivariate interpolation for fluid-structure interaction problems using radial basis functions*, Aerosp. Sci. Technol. **1** (2001), no. 11, 1–11.

[8] M. Berggren, *Edge-based mesh movement strategies*, unpublished preprint.

[9] ———, *The volume discharge approach to geometric conservation*, Computational Methods for Fluid-Structure Interaction (T. Kvamsdal *et al.*, ed.), Tapir Publishers, N-7005 Trondheim, Norway, 1999.

[10] T. Berglind, *An Agglomeration Algorithm for Navier-Stokes Grids*, Aerospace Sciences Meeting, no. AIAA Paper 2000-2254, January 2000.

[11] R. L. Bisplinghoff and H. Ashley, *Principles of aeroelasticity*, John Wiley and Sons, Inc., New York-London, 1962. MR MR0151036 (27 #1022)

[12] *Computational Engineering International, CEI*, `www.ceintl.com`.

[13] M. Chevalier, *Xedge v1.2 user manual*, Tech. Report FOI-D-0210-SE, FOI, April 2005.

[14] L. Djayapertapa, *A computational method for coupled aerodynamic-structural calculations in unsteady transonic flow with active control study*, Ph.D. thesis, University of Bristol, 2000.

[15] L. Djayapertapa and C.B. Allen, *Aeroservoelastic simulation by time-marching*, The Aeronautical Journal **105** (2001), no. 1054, 667–678.

[16] *Deutschen zentrum für Luft- und Raumfahrt, DLR*, `www.dlr.de`.

[17] Earl H. Dowell, R. Clark, D. Cox, H. C. Curtiss, Jr., J. W. Edwards, K. C. Hall, D. A. Peters, R. Scanlan, E. Simiu, F. Sisto, and T. W. Strganac, *A modern course in Aeroelasticity*, fourth ed., Solid Mechanics and its Applications, vol. 116, Kluwer Academic Publishers, Dordrecht, 2004, ISBN 1-4020-2039-2. MR MR2107428 (2005h:74020)

[18] Mark Sutcliffe (Editor), *Final report on validaton excersises and comparison to last release of the TAURUS CFD software*, Tech. Report TAURUS Deliverable D6.2-36, Airbus-D, June 2004.

[19] P. Eliasson, *EDGE, a Navier-Stokes solver for unstructured grids*, Proc. of Finite Volumes for Complex Applications III, 2002.

[20] C. Farhat and M. Lesoinne, *Higher-order staggered and subiteration free algorithm for coupled dynamic aeroelasticity problems*, 36th Aerospace Sciences Meeting (Reno), no. AIAA Paper 98-0516, January 1998.

[21] C. Farhat, M. Lesoinne, and N. Maman, *Mixed explicit/implicit time integration of coupled aeroelastic problems: three-field formulation, geometric conservation and distributed solution*, Internat. J. Numer. Methods Fluids **21** (1995), no. 10, 807–835, Finite element methods in large-scale computational fluid dynamics (Tokyo, 1994). MR MR1368695

[22] K. C. Felippa C. A, Park and F. Farhat, *Partitioned analysis of coupled mechanical systems*, Computer methods in applied mechanics and engineering **190** (2001), 3247–3270.

[23] FOI, *Euranus v5.2 users guide*, 1998.

[24] FOI, *Edge 3.2 manual, issue 3.2/1, dnr. 03-2870*, May 2004.

[25] FOI, *Edge 3.3 manual, issue 3.3/1, dnr. 03-2870*, May 2005.

[26] Y. C. Fung, *An introduction to the theory of aeroelasticity*, John Wiley & Sons Inc., New York, 1955. MR MR0071230 (17,101g)

[27] P. Girodroux-Lavigne, J. P. Grisval, S. Guillemot, M. Henshaw, A. Karlsson, V. Selmin, J. Smith, E. Teupootahiti, and B. Winzell, *Comparative study of advanced fluid-structure inteaction methods in the case of a highly flexible wing (results from the UNSI program)*, Proc. International Forum for Aeroelasticity and Structural Dynamics, IFASD, June 2001.

[28] ———, *Comparison of static and fluid-structure interaction solutions in the case of a highly flexible modern transport aircraft*, Aerospace Science and Technology **7** (2003), 121–133.

[29] S. Görtz, *Realistic Simulations of Delta Wing Aerodynamics Using Novel CFD Methods*, Ph.D. thesis, Department of Aeronautical and Vehicle Engineering, Royal Inistitute of Technology KTH, SE-100 44 Stockholm, Sweden, 2005.

[30] S. Görtz and J. Möller, *Evaluations of the Recursive Projection Method for Efficient Turbulent Unsteady CFD Simulations*, Proc. 24th Congress of the International Council of the Aeronautical Sciences (ICAS), Yokohama, Japan, 2004.

[31] H. Guillard and C. Farhat, *On the significance of the Geometric Conservation Law for flow computations on moving meshes*, Comput. Methods Appl. Mech. Engrg. **190** (2000), no. 11-12, 1467–1482. MR MR1807009 (2001j:76081)

[32] W. Haase, V. Selmin, and B. Winzell (eds.), *Progress in computational Flow-Structure Interaction results of the project UNSI, supported by the European Union 1998-2000*, Notes on Numerical Fluid Mechanics and Multidisciplinary Design (NNFM), vol. 81, Springer-Verlag, 2003, ISBN 3-540-43902-1.

[33] J.R. Hughes, *The Finite Element Method*, Prentice-Hall, Inc., 1987, ISBN 0-13-317025-X.

[34] S. Jakobsson, *Frequency optimization, Radial Basis Functions and interpolation*, Tech. Report FOI-R-0984-SE, FOI, February 2004.

[35] ———, *A geometric interpolation algorithm based on Radial Basis Functions*, Tech. Report FOI-MEMO-1142-SE, FOI, December 2004.

[36] S. Jakobsson and O. Amoignon, *Mesh deformation using Radial Basis Functions for gradient-based aerodynamic shape optimization*, Tech. Report FOI-R-1784-SE, FOI, December 2005.

[37] A. Jameson, *Time Dependent Calculations Using Multigrid with Applications to Unsteady Flows Past Airfoils and Wings*, AIAA Paper (1991), no. 91, 1596.

[38] C. Johansson, *Fully Coupled Calculations of Unsteady Fluid-Structure Interaction*, Tech. Report FOI-S-1458-SE, FOI, 1999, KTH (NADA) Master's thesis.

[39] A. Karlsson, *Aeroelastiska applikationer med Edge - önskemål från Saab angående utveckling i närtid*, Tech. Report TDW-2004-0263, Saab Aeroesystems AB, September 2004, (Aerolelastic applicatons with Edge - Saab requirements for near-term development).

[40] B. Koobus and C. Farhat, *Second-order time-accurate and geometrically conservative implicit schemes for flow computations on unstructured dynamic meshes*, Comput. Methods Appl. Mech. Engrg. **170** (1999), no. 1-2, 103–129. MR MR1679256 (99k:76101)

[41] *The Mathworks Worldwide*, `www.mathworks.com`, Matlab.

[42] D. J. Mavriplis, *Directional agglomeration multigrid techniques for high-Reynolds number viscous flows*, AIAA Journal **37** (1999), 1222.

[43] J. P. Morand and R. Ohayon, *Fluid structure interaction*, Wiley, 1995, ISBN 0-471-94459-9.

[44] *MSC NASTRAN*, `www.mscsoftware.com`.

[45] NATO, *Advisory committee compendium of unsteady aerodynamic measurements*, AGARD, no. 702, August 1982.

[46] NATO, *Advisory committee compendium of unsteady aerodynamic measurements*, AGARD, no. 702 Addendum 1, May 1982.

[47] N. M. Newmark, *A method of compuation in structural dynamics*, Journal of the Engineering Mechanics Division ASCE (1959), 67–94.

[48] *National Lucht- en Ruimtevaartlaboratorium, NLR*, `www.nlr.nl`.

[49] M. Schroff, G. and B. Keller, *Stablization of unstable procedures: The recursive projection method*, SIAM J. Numer. Anal. **30** (1993), no. 4, 1099–1120.

[50] J. Smith, *Unsteady RANS calculations for the GFSI bump. Preliminary study using Euranus 5.3*, Tech. report, FOI, February 2003, FOI-R-0814-SE.

[51] ———, *Aeroelastic functionality in Edge 3.2, Validation for the LANN wing CT5 test case*, Tech. report, FOI, December 2004, FOI-MEMO-1168.

[52] ———, *FFA Matlab Toolbox, version 1.5*, FOI, February 2004, HTML manual.

[53] J. Smith and M. J. de C. Henshaw, *Report on Aeroelasticity Methods for Transonic Flow Including Flutter*, Tech. report, BAE Systems, January 2001, UNSI Deliverable D5.4-30.

[54] E. Sterner, *Accuracy and convergence studies of the numerical solution of compressible flow problems*, Ph.D. thesis, Dept. of Scientific Computing, Uppsala University, 1997.

[55] P. D. Thomas and C. K. Lombard, *Geometric Conservation Law and its application to flow computations on moving grids*, AIAA J. **17** (1979), no. 10, 1030–1037. MR MR544850 (80h:65104)

[56] L. Tysell, *Hybrid grid generation for complex 3d geometries*, 7th International Conference on Numerical Grid Generation in Computational Field Simulations, September 2000.

[57] ———, *Grid deformation of 3d hybrid grids*, 8th International Conference on Numerical Grid Generation in Computational Field Simulations, June 2002.

[58] ———, *TRITET on-line manual*, FOI, 2004.

[59] L. Tysell, T. Berglind, and P. Eneroth, *Adaptive grid generation for 3d unstructured grids*, 6th International Conference on Numerical Grid Generation in Computational Field Simulations, July 1998.

[60] J. W. van der Burg, B. Pranata, and B. Soemarwoto, *Aeroelastic CFD studies for geometrically complex aircraft configurations*, Proc. International Forum for Aeroelasticity and Structural Dynamics, IFASD, National Lucht- en Ruimtevaartlaboratorium, NLR, June 2005.

[61] V. Venkatakrishnan and D. J. Mavriplis, *Agglomeration multigrid for the three-dimensional Euler equations*, AIAA Journal **33** (1995), 633.

[62] N. Vlachos, *Aero-structural coupling in transonic flow: Comparison of strong and weak coupling schemes*, Rep. DERA/ASF/3662U - AERO.RK5615, May 1999.

[63] S Wallin, *Stanadardised Data Format*, Tech. Report FFAP-A-950, FFA, 1992.

[64] S. Wallin and A. Johansson, *A complete explicit algebraic reynolds stress model for incompressible and compressible turbulent flows*, J Fluid Mechanics **403** (2000), 89–132.

[65] H. Wendland, *Fast evaluation of radial basis functions: methods based on partition of unity*, Approximation theory, X (St. Louis, MO, 2001), Innov. Appl. Math., Vanderbilt Univ. Press, Nashville, TN, 2002, pp. 473–483. MR MR1924902 (2003f:41033)

[66] B. Winzell, *Aeroelastic coupling with Euranus, a first implementation.*, Tech. Report TUDLF-0-96:79, Issue 2, Saab Military Aircraft, November 1996.

[67] ———, *Generation of grid perturbations for aeroelastic CFD applications with special discussion of the use within Euranus*, Tech. Report TUDLF-0-96:27, Issue 1, Saab Military Aircraft, March 1996.

[68] ———, *Om hur svängningsformerna kan introduceras i instationära CFD beräkningar*, Tech. Report TUDLF-0-96:06, Saab Military Aircraft, January 1996.

[69] ———, *Generation of unsteady airloads for aeroelastic CFD applications with Euranus. manual for pulse excitation of natural modes.*, Tech. Report TUDLF-0-96:40, Issue 3, Saab Military Aircraft, December 2001, First issued April 1996.

[70] B. Winzell and H. Rabia, *Experience with the Aeroelastic Model of the Telescope Wing*, FOI working document, September 2005.

[71] E. C. Yates, *Standard aeroelastic configurations for dynamic response. Candidate configuration I. Wing 445.6.*, Tech. Report 765, AGARD, July 1998.

[72] *ZONA Tecnology Inc*, `www.zonatech.com`.

A Matlab programs used with Edge

A set of reay-made Matlab functions is provided with the Edge source-code package and is stored in the subdirectory `Edge/matlab`. As explained in section 2.8, these programs are based on the FFA Matlab Toolbox (directory `Edge/matlab/ffa_format` which handles the FFA data format. All programs are documented with Matlab-style on-line help text and a listing of all the installed functions can be obtained by entering the commands "help Edge" and "help ffa" at the Matlab prompt.

The main fucntions used in this report are summarised in table 2.2 and the use of these functions is illustrated in the two vaidation studies, sections 3 and 4. We present here a small selection of the on-line help texts that can be accessed at the Matlab prompt. The texts are self-explanatory and therefore presented without further comment.

Matlab help text: Overview of the FFA Matlab Toolbox

```
>> help ffa

   FFA Matlab Toolbox
  ---------------------------------------------------------------
   For help on any function, enter "help" and the function name.
   To find the HTML manual for this toolbox, enter "ffa_helpdesk"
  ---------------------------------------------------------------

   Import/Export
     ffaaload       -   read an ascii FFA file into Matlab
     ffabload       -   read a binary FFA file into Matlab
     ffa_dump       -   write a Matlab FFA dataset out to file

   Inspection
     ffa_list       -   list dataset contents and get pointers
     ffa_diff       -   compare two whole datasets

   Data Access
     ffa_find       -   find sub-datasets by name type or dimesion
     ffa_get        -   extract items from a dataset
     ffa_put        -   modify/place items in a dataset

   Datset Manipulation
     ffa_create     -   create a new dataset with no sub-datasets
     ffa_getsub     -   extract a complete sub-dataset
     ffa_putsub     -   build-in a new sub-dataset
     ffa_delsub     -   delete a sub-datset

  ---------------------------------------------------------------
```

Matlab help text: Overview of Matlab fucntions in Edge 3.3

```
>> help Edge

   Edge Matlab functions
  ------------------------------------------------------------------------
  For help on any function, enter "help" followed by the function name.
  Most of these functions use the FFA Matlab Toolbox (try "help ffa").
  ------------------------------------------------------------------------

   Aeroelastics
      getmores      : extract (.bres) modal time-history
      plotmores     : display (.bres) modal time-history
      plotbdis      : display (.bdis) surface displacement
      plotmot       : display (.amot) modal timeseries tabulation
     mortors      : transform modal time-history to real-space points
      mortergs      : extract energy terms from modal time-history

   Timeseries (periodic)
     plotmorep    : plot periodic differences for modal "res" timeseries
     periodiff    : periodic difference for general timeseries data
     pdiffs       : periodic difference for "res" time history data
      harm_bout     : Fourier decomposition of solution timeseries
   Timeseries (general)
     timeset      : adjust timescale in a pair of 'res'&'out' files

   Mesh operations
      mshplot2d     : display a 2d mesh including voulme
      mshplot3d     : display boundaries for a 3d mesh
      boundcloud    : quick point-cloud display of boundary nodes
      boum_getsrf   : extract a boundary-mesh dataset
     boum_tritet   : write (tri only) boundary mesh in TRITET format
      boum_plot     : plot boundary nodes or elements (slowly!)
      boum_qua2tri  : replace quad elements with tri pairs
      unsymmsh      : mirror a (tri/tet) mesh in a symemtry plane

   Utilities        (used by other Edge Matlab programs)
      claf          : close all figures
      jplot3        : handy plot for 3D point sets
      n_n           : used to print "_" in figure text strings
      nuf           : start new figure
      xyzlabels     : put X,Y,Z labels on a 3D plot
      xyzrot        : arbitrary rotation (as ./ae_util/xyzrot.f)
      xyztriad      : plot a little 3D axes symbol

   Process Coupling (development/debug use only)
      elk.m         : use with Edge/ae_util/elk.f

   ------------------------------------------------------------------------
```

Matlab help text for function MORTORS

```
>> help mortors

   MORTORS : transform MOdal Response TO Real-Space

   usage :  str = mortors(fres,fmod)       (*)
             "    mortors(fres,fmod,nn)

   in:  fres    Edge "res" data (filename or dataset)
        fmod    Edge "mod" data (filename or dataset)
   (*)   nn     node numbers in "mod" grid

   out: str     structure array containing -
        .time : timescale(t)      : NT x 1
        .coor : coordinate(t)     : NT x NS x ND
        .cref : coordinates ref   :      NS x ND
        .velo : velocity(t)       : NT x NS x ND
        .qcrd : modal coordinates : NT x NM
        .qvel : modal velocities  : NT x NM
        .nn   : copy of nn input or default

   (*) If 'nn' is absent then the response is
       is extracted at ALL points in the "mod" grid.

    NT : no. timesteps
    NS : no. structural nodes or length(nn)
    ND : space dimension 2 or 3
    NM : no. modes
```

Matlab help text for function MORTORS

```
>> help mortergs

   MORTERGS - Modal response to Energy series

   Computes variations in the structural energy from
   a set of modal response signals and modal parameters.

  (Data from Edge output Edge.bres a Edge_echo.bmop)

   Usage:    str = mortergs(fres,fmop)

   in:    fres    Edge.bres       filename or dataset
          fmop    Edge_echo.bmod  filename or dataset

   out:   str.    structure array containing
                         energy response timeseries

          .time   time                                NT x 1

          .teto   total energy     E(t)               NT x 1
          .teki         kinetic
          .tepo         potential
          .tpto         power      dE(t)/dt              "
          .tpda         ""         struc damping         "
          .tpfv         ""         fluid F.v             "

          .qeto   modal energy     E(t)               NT x NM
          .qeki         kinetic
          .qepo         potential
          .qpto         power      dE(t)/dt              "
          .qpda         ""         struc damping         "
          .qpfv         ""         fluid F.v             "

   NT = ( length of timeseries )
   NM = ( number of modes      )
```

Matlab help text for function HARM_BOUT

```
>> help harm_bout

  HARM_BOUT  harmonic field variables from a sereies of "bout" files
             - applies to periodic prescribed motion solutions
             - created for Edge 2.3

  Usage:
             harm_bout(prefix, dt, it_start, it_period, n_harm)

  Arguments:

   prefix    [string] prefix for filename e.g. for a series of files like
                 'wobblywing_793.bout' the prefix is 'wobblywing'

    dt         [real] timestep size

    it_start   [integer] initial timestep number

    it_period  [integer] period of oscillation in timesteps

    n_harm     [integer>=1] number of harmonics
                DEFAULT   n_harm=1; zero frequency, f and 2f only

  Returns:  no return variables

  Output:   for each harmonic -

               prefix-Hn-real.bout
               prefix-Hn-imag.bout
               prefix-Hn-magn.bout
               prefix-Hn-phas.bout

                     - where  'Hn' = H0 H1 H2 ... H<n_harm>


  IMPORTANT   The zeroth harmonic is returned as the true "DC" or
              "mean value" component. That is "A0/2" where "An"
              is the nth real coefficient in the Fourier series.
```

FOI is an assignment-based authority under the Ministry of Defence. The core activities are research, method and technology development, as well as studies for the use of defence and security. The organization employs around 1350 people of whom around 950 are researchers. This makes FOI the largest research institute in Sweden. FOI provides its customers with leading expertise in a large number of fields such as security-policy studies and analyses in defence and security, assessment of different types of threats, systems for control and management of crises, protection against and management of hazardous substances, IT-security an the potential of new sensors.

FOI
Swedish Defence Research Agency
SE-164 90 STOCKHOLM

Tel: +46 8 5550 3000
Fax: +46 8 5550 3100

www.foi.se