Project ANTASME

WP 3
Report on Innovative Finite Element Methods For Aeroelastic Analysis

Compiled by Roberto Flores

CIMNE: International Center for Numerical Methods in Engineering

North East South West INTERREG IIIC MATEO PROJECT PART-FINANCED BY THE EUROPEAN UNION

This report describes the activities carried out at CIMNE to improve the analysis tools for coupled fluid-structure interaction analysis

In order to develop the capabilities of CIMNE's software in the Fluid-Structure Interaction (FSI) domain, research and development activities have been focused in three different fronts:

- Advances in fluid solver technology.
- Development of a unified software platform (KRATOS) to allow integration of single-field codes into a coupled multiphysics solver.
- Development of new coupling algorithms.

Each of these aspects is summarized in the reports included in this document.

- Improvements In CFD Solver Technology, Some Theoretical Aspects (contributed by Enrique Ortega)
- A Short Description Of A 3D Navier-Stokes Compressible Flow Solver (contributed by Roberto Flores)
- Basic Structure of KRATOS, An Object-Oriented Environment For The Development Of Multi-Physics Analysis Software (contributed by Pooyan Dadvand)
- Analysis Of Some Fluid-Structure Interaction Algorithms (contributed by Riccardo Rossi)

1. IMPROVEMENTS IN CFD SOLVER TECHNOLOGY, SOME THEORETICAL ASPECS

1.0. OVERVIEW

CIMNE's traditional focus on the field has been on low-speed applications. FE software for near-incompressible CFD has been actively developed in the past. A notorious example is the TDYN software package for ship hydrodynamics, which after being initially developed at CIMNE is now marketed by "COMPASS Ingeniería y Sistemas" (A CIMNE spin-off company). However, high-speed solver capabilities were lagging behind so it was decided to develop a new solver for highly compressible flows. In order to harmonize the new development with existing in-house software, a decision was made to use a finite element based code instead of the more traditional finite volume approach. To achieve optimal performance, an edge based data structure has been chosen. While the actual code is a 3D Navier-Stokes solver, in the following pages the general algorithm is described using the bidimensional Euler equations for the sake of simplicity.

1.1. INTRODUCTION

When element-based data structures are used in finite elements calculations, certain redundancies of information occurs. For linear solvers based on triangular and tetrahedral elements, an alternatively and more efficient data structure using only edge information can be considered. Edge-based data structures for finite elements calculations have been introduced by Morgan *et al.* drawing from earlier finite volume schemes [2]. This mesh representation allows both to take advantage of unstructured meshes and reduce CPU time and memory required by the calculations [1,3,4] which is of vital importance when considering three-dimensional flow calculations. Additionally, when compressible or incompressible flow solvers are dealt with, edge-based representations make a straightforward implementation of upwind schemes in the finite element method context possible. Several comparisons between element and edge-based data structures performance can be found in the literature, see for instance [1]. An excellent review of the most common upwind schemes and its application on unstructured grids is set forth in [4]. .

1.2. EULER EQUATIONS

The Euler equations model a non-viscous, compressible and non-conductive fluid flow. This set of conservation laws is represented by a coupled non-linear system of first order partial differential equations, which can be written in different ways. The conservative form of the equations must be used when it is necessary to take into account discontinuities in the fluid

field, such as contact discontinuities or shocks. The conservative form of the Euler equations can be expressed in a compact manner (omitting source terms) as follows

$$\frac{\partial U}{\partial t} + \frac{\partial F^{\kappa}}{\partial x_{\kappa}} = \tilde{0} \tag{1}$$

where U is the vector of the conservative variables and F^{κ} is the inviscid flux vector in the direction x_{κ}. For two-dimensional flow these vectors are defined by

$$U = \begin{bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho e_t \end{bmatrix} \quad ; \quad F^{\kappa} = \begin{bmatrix} \rho u_{\kappa} \\ \rho u_1 u_{\kappa} + \delta_{1\kappa} p \\ \rho u_2 u_{\kappa} + \delta_{2\kappa} p \\ (\rho e_t + p) u_{\kappa} \end{bmatrix} \tag{2}$$

where ρ is the fluid density, u_{κ} is the component of the velocity vector in the direction x_{κ}, p is the pressure and δ_{ij} denotes the Kronecker delta. In addition, the total specific energy e_t is composed of the internal and kinetic energy of the fluid particle. Finally, the equation system (1) is closed by the following state relation

$$p = \rho(\gamma - 1)\left[e_t - \tfrac{1}{2} u_{\kappa} u_{\kappa} \right] \tag{3}$$

in which $\gamma = C_p/C_v$ is the constant pressure and volume specific heat ratio, adopted as $\gamma = 1.4$ in the present work.

The solution of the system of equations (1) over a closed domain Ω with boundary $\Gamma = \Gamma_0 \cup \Gamma_n$ for a given time t requires additionally initial and boundary conditions given by

$$\begin{aligned} U(x,0) &= U_0(x) \qquad t = 0 \,,\, x \in \Omega \\ U(x,t) &= U_n(x) \qquad t \geq 0 \,,\, x \in \Gamma_0 \\ F^n &= F \cdot \hat{n} = \bar{F}^n \qquad t \geq 0 \,,\, x \in \Gamma_n \end{aligned} \tag{4}$$

where $\hat{n}$ is the unit outward normal vector to boundary Γ_n.

1.3. FINITE ELEMENT DISCRETIZATION

Consider a discretization of the whole domain Ω into subdomains or elements Ω^e such as $\Omega = \sum_{nelem} \Omega^e$. Then the weak formulation for the system of equations (1) can be expressed as

$$\int_{\Omega} W \frac{\partial U}{\partial t} d\Omega - \sum_k \int_{\Omega} \frac{\partial W}{\partial x_{\kappa}} F^{\kappa}(U)\, d\Omega + \int_{\Gamma} W\, \bar{F}^n\, d\Gamma = \tilde{0} \tag{5}$$

being W a set of arbitrary weighting functions. Note that the advective fluxes terms are integrated by parts in equation (5). To continue, an approximation to the conservative variables vector U is defined in each subdomain Ω^e in the following manner

$$\widehat{U} = \sum_{j=1}^{nnode} U_j N_j \tag{6}$$

where *nnode* is the total number of nodes in the element, N_j is the standard finite elements shape function associated with node j and U_j is the value of $\widehat{U}$ at the same node. Due to the fact that the flux vectors F^κ are generally a non-linear function of U, the formers are also approximated by the element shape functions as follows (group representation)

$$F^\kappa(\widehat{U}) = \sum_{j=1}^{nnode} F_j^\kappa N_j = \sum_{j=1}^{nnode} F^\kappa(\widehat{U}_j) N_j \tag{7}$$

Assuming the Galerkin finite element formulation, $W_j = N_j$ and the approximate weak formulation can be written for a generic node i as

$$\sum_{e\in i}\int_{\Omega^e} N_i \frac{\partial \widehat{U}}{\partial t} d\Omega = \sum_{e\in i}\sum_{\kappa}\int_{\Omega^e} \frac{\partial N_i}{\partial x_\kappa} F^\kappa(\widehat{U})\, d\Omega - \sum_{b\in i}\int_{\Gamma} N_i \bar{F}^n\, d\Gamma \tag{8}$$

where the summation extends over all the elements e and boundaries b which contains the generic node i.
Taking into consideration triangular elements with standard C^0 shape functions and the approximate forms for the conservative variables vector and the flux vectors given in eqs. (6) and (7) respectively, the integral terms in eq. (8) can be evaluated as

$$\sum_{e\in i}\int_{\Omega^e} N_i \frac{\partial \widehat{U}}{\partial t} d\Omega = \sum_{e\in i}\left[\int_{\Omega^e} N_i N_j\, d\Omega\right]\frac{dU_j}{dt} = \left[M\frac{dU}{dt}\right]_i \tag{9}$$

$$\sum_{e\in i}\sum_{\kappa}\int_{\Omega^e} \frac{\partial N_i}{\partial x_\kappa} F^\kappa(\widehat{U})\, d\Omega \;=\; \sum_{e\in i}\sum_{\kappa}\left[\frac{\Omega^e}{3}\frac{\partial N_i}{\partial x_\kappa}\right]\left(F_i^\kappa + F_j^\kappa + F_k^\kappa\right) \tag{10}$$

$$\sum_{b\in i}\int_{\Gamma} N_i \bar{F}^n\, d\Gamma = \sum_{b\in i}\left[\frac{\Gamma_b}{6}\left(2\bar{F}_i^n + \bar{F}_j^n\right)\right] \tag{11}$$

where M is the consistent mass matrix and Ω^e is the area of element e with nodes i, j and k. In the last expression Γ_b is the length of the boundary edge b defined by nodes i and j.

1.3.1 Edge-Based formulation

Using and edge-based data structure, nodal values of the diverse quantities are obtained adding edge contributions. The typical edge-data consists of the nodal coordinates and a list of all the edges in the mesh and its connectivities, i.e. the nodes to define each edge. A list of boundary edges where physical boundary conditions are imposed is also necessary. Several routines to obtain an edge-data structure for finite elements calculations can be found in [1].
Once the edge-data is obtained for a particular problem, the right hand side of eq. (8) can be evaluated for triangular linear elements in the following manner [4]

$$\left[M \frac{dU}{dt} \right]_i = \sum_{ed=1}^{ne_i} \sum_{\kappa} C_{ij}^{\kappa} \left(F_i^{\kappa} + F_j^{\kappa} \right) + \left[\sum_{f=1}^{2} D_f \left(4\bar{F}_i^n + 2\bar{F}_{j_f}^n + F_i^n - F_{j_f}^n \right) \right]_i \tag{12}$$

where C_{ij}^{κ} is the weight that must be applied to the sum of the fluxes in the x_κ direction on the edge *ed*, with nodes *i* and *j*, to obtain the contribution made for the edge to node *i*. The first summation term extends over all the edges in the mesh that contain node *i*, i.e. ne_i, and for each spatial direction x_κ. The term between brackets is only non-zero when the node *i* lies on the boundary and *f* denotes each edge that contains nodes i-j_1 and i-j_2, i.e. the boundary edges associated with node *i*. The bar over normal fluxes F_i^n and $F_{j_f}^n$ denotes prescript fluxes that become equal to the calculated normal fluxes in the bracket term if no boundary conditions are imposed on the edge. The weight coefficients are calculated as follows and the complete process to obtain these weights is presented in [4]

$$C_{ij}^{\kappa} = \sum_{e \in ij} \frac{\Omega^e}{3} \left[\frac{\partial N_i}{\partial x_\kappa} \right]_e - \left[\frac{\Gamma_f}{12} n_{ij}^{\kappa} \right]_{ij} \tag{13}$$

$$D_f = -\frac{\Gamma_f}{12} \tag{14}$$

Again, the second term in the right hand side of eq. (13) is only non-zero when the node *i* lies on a boundary. To obtain the C_{ij}^{κ} weights, the summation extends over all the elements that contains the edge *ij* , Γ_f is the length of the boundary edge *ij* and n_{ij}^{κ} is the vector normal κ component at edge *ij*.

Similarly, the weights C_{ji}^{κ} to be applied to the sum of the fluxes in the x_κ direction on the edge *ed*, with nodes *i* and *j*, to obtain the contribution made for the edge to node *j* can be calculated as

$$C_{ji}^{\kappa} = -C_{ij}^{\kappa} \tag{15}$$

and can be demonstrated by geometrical arguments [4].

As can be noticed, the weight coefficients (13) and (14) must be determined in a post-process stage. Then eq. (12) is evaluated through a loop over each edge in the mesh and adding the edge contributions to the appropriate nodes. A second loop over the boundary edges is also necessary and the boundary contributions are added to the appropriate boundary nodes, too.
Finally, in order to write eq. (12) in a more compact format, fluxes in the weight coefficients direction are defined by

$$\begin{aligned} f_i &= S_{ij}^{\kappa} F_i^{\kappa} \\ f_j &= S_{ij}^{\kappa} F_j^{\kappa} \end{aligned} \tag{16}$$

where

$$S_{ij}^{\kappa} = \frac{C_{ij}^{\kappa}}{\left|C_{ij}\right|} \quad \text{and} \quad \left|C_{ij}\right| = \sqrt{C_{ij}^{\kappa} C_{ij}^{\kappa}} \tag{17}$$

and taken into account the above expressions, eq. (12) becomes

$$\left[M \frac{dU}{dt} \right]_i = \sum_{ed=1}^{ne_i} \left|C_{ij}\right| \underbrace{\left(f_i + f_j \right)}_{\mathcal{F}_{ij}} + \left[\sum_{f=1}^{2} D_f \left(4\bar{F}_i^n + 2\bar{F}_{j_f}^n + F_i^n - F_{j_f}^n \right) \right]_i \tag{18}$$

Due to certain properties of the edge weights it is observed that the discretization scheme is conservative in the sense that the sum of the contributions made for any interior edge is zero. It is demonstrable too that the discretization scheme is a central difference type discretization for the spatial derivatives and some dissipation terms must be introduced in order to provide the necessary stabilization for the scheme. This fact brings about the replacement of the flux function $\mathcal{F}_{ij}$ defined in (18) by new consistent numerical fluxes. Adopting different forms for the latter, it is possible to obtain a wide variety of algorithms in which the numerical dissipation is introduced in a explicit manner or the flux function is modified according to the physics of the problem. An excellent recompilation and a comparative study of the most common algorithms is presented in [4]. Here, following Löhner's work [1], a first order numerical flux according to the Roe's approximate Riemman solver is employed in conjunction with a limiting stage with the aim of reducing the amount of dissipation and increasing the order of the scheme in regions where the flow is smooth.

1.4. ROE'S APPROXIMATE RIEMMAN SOLVER

The Roe's solver for the Euler equations based on flux difference splitting is one of the most popular and less dissipative approximate Riemman solvers and was developed by Roe in 1981. The idea behind this method is to solve the Riemman problem (Godunov) at the interface of two piecewise constant states U_L and U_R in a approximate manner reducing the computational cost and obtaining equally good results. The first order flux for this solver is defined for each edge by

$$\mathcal{F}_{ij} = f_i + f_j - \left| A(U_i, U_j) \right| \left(U_j - U_i \right) \tag{19}$$

where U_i and U_j is the vector of conservative variables at the edge nodes i and j and $|A(U_i,U_j)|$ is the absolute value of the Roe matrix calculated in the direction of the edge e_{ij}. The last is obtained projecting the conservative Jacobian matrices A^{κ} in the direction of the edge e_{ij} and replacing the variables in the resulting Jacobian matrix by the density-average Roe variables. Then the absolute value of the Roe matrix is achieved decomposing the last matrix by means of its eigenvalues and eigenvectors matrices. General expressions for the Jacobian, eigenvalues and eigenvectors matrices can be found in [5].
The density-average Roe variables are obtained by

$$\begin{aligned}\tilde{\rho}_{ij} &= \sqrt{\rho_i \rho_j} \\ \tilde{u}^{\kappa}_{ij} &= \frac{\sqrt{\rho_i}\, u^{\kappa}_i + \sqrt{\rho_j}\, u^{\kappa}_j}{\sqrt{\rho_i} + \sqrt{\rho_j}} \\ \tilde{H}_{ij} &= \frac{\sqrt{\rho_i}\, H_i + \sqrt{\rho_j}\, H_j}{\sqrt{\rho_i} + \sqrt{\rho_j}}\end{aligned} \tag{20}$$

where u^{κ} is the velocity component in direction κ and H is the total enthalpy $H=e_t+p/\rho$. The density-average values can be obtained with a higher computational efficiency through the definition of the following auxiliary parameter [5]

$$R = \sqrt{\rho_j / \rho_i} \tag{21}$$

which allows us to express

$$\begin{aligned}\tilde{\rho}_{ij} &= R\,\rho_i \\ \tilde{u}^{\kappa}_{ij} &= \frac{R\,u^{\kappa}_j + u^{\kappa}_i}{R+1} \\ \tilde{H}_{ij} &= \frac{R\,H_j + H_i}{R+1}\end{aligned} \tag{22}$$

Using these interface or intermediate variables, the average speed of the sound is obtained by

$$\tilde{c}_{ij} = \sqrt{(\gamma-1)\left(\tilde{H}_{ij} - \tfrac{1}{2}\left(\sqrt{\tilde{u}^{\kappa}_{ij}\tilde{u}^{\kappa}_{ij}}\right)^2\right)} \tag{23}$$

In order to avoid the eigenvalue and eigenvector decomposition of the Roe matrix $A(U_i,U_j)$ for the calculation of its absolute value in eq. (19), a calculation that is more computationally efficient is presented in the next section.

1.4.1 A practical calculation of the dissipative term

Following the pioneer ideas of Turkel [6] and a simplification of his model proposed in [7], the dissipative term, $|A(U_i,U_j)|\cdot(U_j - U_i)$, in eq. (19) can be evaluated in such a manner that matrix-vector and vector-vector multiplication are avoided increasing the computational efficiency. Then, the general three-dimensional expressions for the dissipation term of eq. (19) are presented and the complete derivation of these terms can be found in [7].

Due to the hyperbolic nature of the Euler equations and its derived properties, any linear combination of the Jacobian matrices, in our case the Roe matrix evaluated in the direction of the edge e_{ij}, is diagonalizable with real eigenvalues. Thus, there exists a diagonal matrix Λ_{ij} containing the eigenvalues of $A(U_i,U_j)$ and an associated matrix of eigenvectors R_{ij} such that the Roe matrix for the edge e_{ij} admits the following factorization

$$A\left(U_i,U_j\right)=R_{ij}^{-1}\Lambda_{ij}R_{ij} \tag{24}$$

and

$$\left|A\left(U_i,U_j\right)\right|=R_{ij}^{-1}\left|\Lambda_{ij}\right|R_{ij} \tag{25}$$

where $\left|\Lambda_{ij}\right|=diag\left\{\left|\lambda_1\right|,\left|\lambda_1\right|,\left|\lambda_1\right|,\left|\lambda_2\right|,\left|\lambda_3\right|\right\}$ and

$$\begin{aligned}\lambda_1&=\hat{u}_{ij}\\ \lambda_2&=\hat{u}_{ij}+\tilde{c}_{ij}\\ \lambda_3&=\hat{u}_{ij}-\tilde{c}_{ij}\end{aligned} \tag{26}$$

It must be remembered here that all the variables correspond with the average-density Roe variables calculated for the edge e_{ij}. In eqs. (26) $\hat{u}_{ij}$ is the interface velocity projected on the edge, i.e. $\tilde{u}_{ij}^{\kappa}\cdot n^k$, and n=($n^{(1)}$,$n^{(2)}$,$n^{(3)}$) is a unitary vector in the direction of the edge e_{ij}.

Introducing the factorization (25), the dissipative term of the Roe first order numerical flux can be written as

$$D=\left|A(U_i,U_j)\right|\left(U_j-U_i\right)=\left(R_{ij}^{-1}\left|\Lambda_{ij}\right|R_{ij}\right)\Delta U_{ji} \tag{27}$$

where ΔU_{ji}=(U_j-U_i) is a five-component vector which entries are the differences between the conservative variables vector components at edge nodes j and i. After some algebraic manipulations [7] the dissipative term in the first order Roe flux can be evaluated as follows

$$D=\left|\lambda_1\right|\Delta U_{ji}+\frac{\left|\lambda_2\right|-\left|\lambda_1\right|}{2}\Psi_1\begin{bmatrix}1\\ \tilde{u}_{ij}^{(1)}/\tilde{c}_{ij}+n^{(1)}\\ \tilde{u}_{ij}^{(2)}/\tilde{c}_{ij}+n^{(2)}\\ \tilde{u}_{ij}^{(3)}/\tilde{c}_{ij}+n^{(3)}\\ \tilde{H}_{ij}/\tilde{c}_{ij}+\hat{u}_{ij}\end{bmatrix}+\frac{\left|\lambda_3\right|-\left|\lambda_1\right|}{2}\Psi_2\begin{bmatrix}1\\ \tilde{u}_{ij}^{(1)}/\tilde{c}_{ij}-n^{(1)}\\ \tilde{u}_{ij}^{(2)}/\tilde{c}_{ij}-n^{(2)}\\ \tilde{u}_{ij}^{(3)}/\tilde{c}_{ij}-n^{(3)}\\ \tilde{H}_{ij}/\tilde{c}_{ij}-\hat{u}_{ij}\end{bmatrix} \tag{28}$$

where

$$\begin{aligned}\Psi_1=&\left(\frac{\gamma-1}{\tilde{c}_{ij}}q-\hat{u}_{ij}\right)\Delta U_{ji}^{(1)}+\left(-\frac{\gamma-1}{\tilde{c}_{ij}}\tilde{u}_{ij}^{(1)}+n^{(1)}\right)\Delta U_{ji}^{(2)}+\left(-\frac{\gamma-1}{\tilde{c}_{ij}}\tilde{u}_{ij}^{(2)}+n^{(2)}\right)\Delta U_{ji}^{(3)}\\ &+\left(-\frac{\gamma-1}{\tilde{c}_{ij}}\tilde{u}_{ij}^{(3)}+n^{(3)}\right)\Delta U_{ji}^{(4)}+\frac{\gamma-1}{\tilde{c}_{ij}}\Delta U_{ji}^{(5)}\end{aligned} \tag{29}$$

$$\Psi_2 = \left(\frac{\gamma - 1}{\tilde{c}_{ij}} q + \hat{u}_{ij} \right) \Delta U_{ji}^{(1)} + \left(-\frac{\gamma - 1}{\tilde{c}_{ij}} \tilde{u}_{ij}^{(1)} - n^{(1)} \right) \Delta U_{ji}^{(2)} + \left(-\frac{\gamma - 1}{\tilde{c}_{ij}} \tilde{u}_{ij}^{(2)} - n^{(2)} \right) \Delta U_{ji}^{(3)} + \left(-\frac{\gamma - 1}{\tilde{c}_{ij}} \tilde{u}_{ij}^{(3)} - n^{(3)} \right) \Delta U_{ji}^{(4)} + \frac{\gamma - 1}{\tilde{c}_{ij}} \Delta U_{ji}^{(5)} \tag{30}$$

with $q = \frac{1}{2}\left(\tilde{u}_{ij}^{\kappa} \cdot \tilde{u}_{ij}^{\kappa} \right)$. The two-dimensional form of eq. (28) can be obtained by setting $\tilde{u}_{ij}^{(3)} = n^{(3)} = 0$ and removing the fourth component in both, the differences vector ΔU_{ji} and the column vectors introduced in (28).

As can be noted in the definition of the eigenvalues of the Roe matrix (26), λ_1 approaches zero near stagnation points while λ_2 and λ_3 approaches to zero near sonic lines. In view of the structure of the dissipation term, vanishing eigenvalues leads to the dissipation term at these points approaches to zero with the consequence of numerical instabilities in general calculations. A possibility to avoid this behaviour is to limit the eigenvalues defined in (26) in the following manner

$$\begin{aligned} |\lambda_1| &= \max\left(|\lambda_1|, \alpha_1 \rho(A) \right) \\ |\lambda_2| &= \max\left(|\lambda_2|, \alpha_2 \rho(A) \right) \\ |\lambda_3| &= \max\left(|\lambda_3|, \alpha_2 \rho(A) \right) \end{aligned} \tag{31}$$

where α_1 and α_2 are parameters that limit the eigenvalues associated with the linear and non-linear characteristic fields to a value that is a fraction of the spectral radius of the Roe matrix $\rho(A)$. These parameters must be numerically determined taking into consideration both sharpness and oscillation-free capture of discontinuities as well as convergence rate of a given numerical calculations. A value of $\alpha_1 = \alpha_2 = 0.2$ is recommended in [6,8]. It should be noticed that when these parameters are set to zero no limiting is applied to the eigenvalues and when they are set to the unity the dissipation term mimics an scalar dissipation model. Vanishing eigenvalues can also be related to entropy violation problems and several ways to treat it can be found in the literature, see for instance [5].

1.4.2 High order scheme

Common procedures to obtain oscillation-free high-order schemes are based on slope-limited or geometric methods which were originated in Van Leer's observations (1979). These methods are based on the modification of the piecewise constant states used in the Godunov projection stage. In order to achieve high-order spatial approximations, a piecewise linear reconstruction of the interface variables is adopted using neighbouring values. Then, a slope limiting stage must be applied to maintain the monotonicity of the numerical solution.

Adopting the MUSCL formulation for the reconstruction stage, the interface values in the middle of the edge can be obtained with up to third-order accuracy in space in the following manner

$$U_i^+ = U_i + \frac{1}{4}\left[(1-k)\Delta_i^- + (1+k)(U_j - U_i)\right]$$
$$U_j^- = U_j - \frac{1}{4}\left[(1-k)\Delta_j^+ + (1+k)(U_j - U_i)\right] \tag{32}$$

where Δ_i^-, Δ_j^+ are difference operators given by

$$\Delta_i^- = U_i - U_{i-1} = 2l_{ji}\cdot\nabla U_i - (U_j - U_i)$$
$$\Delta_j^+ = U_{j+1} - U_j = 2l_{ji}\cdot\nabla U_j - (U_j - U_i) \tag{33}$$

and l_{ji} is a vector from edge node i to node j, k is a parameter that allows to obtain different spatial order approximations and the gradients of the variables at edge nodes, ∇U_i and ∇U_j, are obtained via a recovery of the first derivatives at nodes procedure. It is possible to note that the approximations of the difference operators in (33) belong to an approximation to the gradient of the variables at nodes i and j using a central difference formulae. As was mentioned earlier, the parameter k in eqs. (32) allows to obtain different order or approximation, for example

- k = -1 → second-order fully upwind scheme
- k = 0 → From's scheme (see [9])
- k = 1/3 → third-order upwind scheme
- k = 1 → three-point central difference scheme

A graphical representation of the high-order approximation is presented in Figure 1.

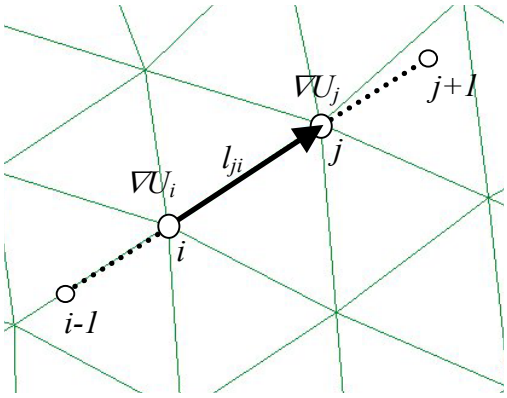


Figure 1: High-order approximation

The proposed extrapolation for the interface values (32) does not guarantee monotonicity properties, then in order to obtain an oscillation-free solution, non-linear limiters are introduced in the extrapolation stage through

$$U_i^+ = U_i + \frac{s_i}{4}\left[\left(1-ks_i\right)\Delta_i^- + \left(1+ks_i\right)\left(U_j - U_i\right)\right]$$
$$U_j^- = U_j - \frac{s_j}{4}\left[\left(1-ks_j\right)\Delta_j^+ + \left(1+ks_j\right)\left(U_j - U_i\right)\right] \tag{34}$$

where s_i and s_j are the flux limiters. Two-parameter non-linear limiters are used in this work based on the Van Albada model [1]. These can be defined by

$$s_i = \max\left\{0\,,\ \frac{2\Delta_i^-\left(U_j - U_i\right)+\varepsilon}{(\Delta_i^-)^2 + \left(U_j - U_i\right)^2 + \varepsilon}\right\}$$
$$s_j = \max\left\{0\,,\ \frac{2\Delta_j^+\left(U_j - U_i\right)+\varepsilon}{(\Delta_j^+)^2 + \left(U_j - U_i\right)^2 + \varepsilon}\right\} \tag{35}$$

where ε is a small constant included to avoid possible divisions by zero. It is possible to note that when the limiters are equal to the unity a high-order approximation is done while when the limiters are equal to zero the first-order approximation is recovered. At this limiting stage, other models for the limiters calculation can be employed. In order to calculate the limiters, different sets of variables can be used, i.e. primitive, conservative or characteristic variables, but conservative variables are used in the present work.
Once the high-order approximations for interface values (34) are calculated, these values are replaced in the first-order numerical flux defined in (19) leading to

$$\mathcal{F}_{ij} = f\left(U_i^+\right) + f\left(U_j^-\right) - A(U_i^+, U_j^-)\left(U_j^- - U_i^+\right) \tag{36}$$

and all the expressions presented in paragraph 4 and 4.1 remain valid replacing U_i for U_i^+ and U_j for U_j^-. These changes in the fluxes along the weight coefficients direction f_i and f_j have only a minor effect in the solution and could be omitted [1].

1.5. TIME DISCRETIZATION

The temporal discretization of eq. (18) is done in a fully explicit manner by means of a multi-stage method that is a subset of the Runge-Kutta family of schemes. These methods allow to increase the stability range and obtain high-accuracy solutions for transient problems. Assuming the vector of conservative variables U is known at each node at time $t=t^n$, the right hand side of eq. (18), called the residual of the equation, at each node RHS_i is evaluated and it is possible to advance the solution in time from t^n to $t^{n+1}=t^n+\Delta t$ by means of an *s-stage* scheme given by

$$
\begin{aligned}
&U_i^{(0)} = U_i^n \\
&\vdots \\
&U_i^{(s)} = U_i^n + \alpha_s \Delta t \left[M_L^{-1} \right]_i RHS_i^{s-1} \\
&\vdots \\
&U_i^{n+1} = U_i^{(s)}
\end{aligned}
\tag{37}
$$

where $\left[M_L^{-1} \right]_i$ is the lumped mass matrix at node $\underline{i}$ and α_s are coefficients that depend on the number of stages *s* employed. For three and four stages these parameters are set according to

3 stages $\rightarrow \alpha_1 = 3/5$, $\alpha_2 = 3/5$ and $\alpha_3 = 1.0$
4 stages $\rightarrow \alpha_1 = 1/4$, $\alpha_2 = 1/3$, $\alpha_3 = 1/2$ and $\alpha_4 = 1.0$

With the aim of reducing the computational cost in the residual evaluation, the dissipative term (28) is calculated at time $t=t^n$ and remains frozen for the next *s* stages of the scheme. Other possibilities proposed by Jameson, Schmidt and Turkel can be found in the literature.

1.5.1 Time step calculation

In the last paragraph the value of the time increment Δt must be bounded by stability criteria. In the present work, the time increment is determined at each node *i* as follows

$$
\Delta t_i = \min \left\{ C \, \Delta t_{ij} \right\} \tag{38}
$$

where *C* is the Courant number and Δt_{ij} is calculated, for each edge in the mesh that contains node *i*, according to

$$
\Delta t_{ij} = \frac{l_{ji}}{\left| V_i^l \right| + c_i} \tag{39}
$$

and

$$
V_i^l = V_i \cdot l_{ji} \tag{40}
$$

In the equations above, l_{ji} is a vector from node *i* to node *j* and V_i and c_i are the velocity vector and the speed of sound at node *i* respectively. If a global time step is chosen to advance in time, Δt in eqs. (37) corresponds to the minimum Δt_i calculated over the mesh.

REFERENCES

[1] Löhner R., '*Applied CFD Techniques*', John Wiley & Sons Ltd., 2001

[2] Morgan K., Peraire J., Peiró J., *'Unstructured Grid Methods for Compressible Flows'*, In Report 787 – Special Course on Unstructured Grid Methods for Advection Dominated Flows. AGARD, 1992.

[3] Morgan K., Peraire J., '*Unstructured Grid Finite-Element Methods for Fluid Mechanics*', Rep. Prog. Phys. 61: 569-638 (1998)

[4] Lyra P. R. M., Morgan K., '*A Review and Comparative Study of Upwind Biased Schemes for Compressible Flow Computation. Part III: Multidimensional Extension on Unstructured Grids*', Arch. Comput. Meth. Engrg Vol 9: 207:256 (2002)

[5] Hirsch C., '*Numerical Computation of Internal and External Flows*', Volume 2, John Wiley & Sons. (1990)

[6] Turkel E., '*Improving the Accuracy of Central Difference Schemes*', ICASE Report 88-53, September 1988

[7] Hu G., '*The Development and Applications of a Numerical Method for Compressible Vorticity Confinement in Vortex Dominant Flows'*, PhD Thesis, Virginia Polytechnic, 2001

[8] Swanson R.C., Turkel E., '*Multistage Schemes with Multigrid for Euler and Navier-Stokes Equations. Components and Analysis'*, NASA Technical Paper 3631, August 1997

[9] Hirsch C., '*Numerical Computation of Internal and External Flows*', Volume 1, John Wiley & Sons. (1990)

2. A SHORT DESCRIPTION OF A 3D NAVIER-STOKES COMPRESSIBLE FLOW SOLVER

2.0. OVERVIEW

To solve real problems of interest for the aerospace industry, there is a need to deal with extremely large models (having tens of millions of elements). Therefore computational efficiency is of prime importance. To achieve this goal the solver uses the edge-based data structure described in chapter 1. It has been optimized for use in shared memory parallel architectures using OPEN-MP directives and features extensive optimizations to minimize memory access overheads (fully vectorized code with access pattern adapted for minimum cache misses) . In this chapter the algorithm used for the general 3D case is described.

2.1 NAVIER-STOKES EQUATIONS

The 3D Navier-Stokes equation set in conservative form can be written as:

$$\frac{\partial \mathbf{\Phi}}{\partial t}+\frac{\partial \mathbf{F}_k}{\partial x_k}+\frac{\partial \mathbf{G}_k}{\partial x_k}=0 \quad for \quad k=1...3 \qquad \textbf{(1)}$$

with the vector of conservative variables, convective and diffusive fluxes being:

$$\mathbf{\Phi}=\begin{bmatrix}\rho\\U_1\\U_2\\U_3\\e\end{bmatrix}\quad \mathbf{F}_i=\begin{bmatrix}U_i\\u_iU_1+p\delta_{i1}\\u_iU_2+p\delta_{i2}\\u_iU_3+p\delta_{i3}\\u_ih\end{bmatrix}\quad \mathbf{G}_i=\begin{bmatrix}0\\-\tau_{1i}\\-\tau_{2i}\\-\tau_{2i}\\q_i-\tau_{ik}u_k\end{bmatrix} \qquad \textbf{(2)}$$

where the different terms can be expressed (in the case an ideal gas with Newtonian behaviour) as:

$$U_i=\rho u_i, e=\rho\left(c_vT+\frac{u^2}{2}\right), h=e+p, q_i=-k\frac{\partial T}{\partial x_i}, \tau_{ij}=2\mu\dot{e}_{ij}+\mu_v\dot{\varepsilon}_{kk}\delta_{ij} \qquad \textbf{(3)}$$

In the case of high-Reynolds number flows the effect of turbulence is accounted by solving the Reynolds average of the equations. The Reynolds stresses thus appearing are modelled using a variety of existing turbulence models. These can be also expressed in a manner similar to **(1)** (usually with the addition of a source term to the left-hand side) therefore the basic scheme remains the same.

2.2 FEM DISCRETIZATION

The weak form of the equation set **(1)** is:

$$\int_\Omega W(\vec{x})\left(\frac{\partial \mathbf{\Phi}}{\partial t}+\frac{\partial \mathbf{F}_k}{\partial x_k}+\frac{\partial \mathbf{G}_k}{\partial x_k}\right)d\Omega=0 \ \ \forall \ \ W \qquad \textbf{(4)}$$

W being an arbitrary test function.

Using the standard FE shape interpolation functions and the Galerkin method

$$\begin{aligned}\tilde{\mathbf{\Phi}}(\vec{x})&=N_j(\vec{x})\tilde{\mathbf{\Phi}}(\vec{x}^j)=N_j\tilde{\mathbf{\Phi}}^j\\ W(\vec{x})&=N_i(\vec{x})\end{aligned} \qquad \textbf{(5)}$$

we find the following semi-discrete form:

$$\int_\Omega N_i \left(N_j \dot{\tilde{\mathbf{\Phi}}}^j + \frac{\partial \tilde{\mathbf{F}}_k}{\partial x_k} + \frac{\partial \tilde{\mathbf{G}}_k}{\partial x_k} \right) d\Omega = 0 \;\; for \;\; i = 1...n_{node} \quad \textbf{(6)}$$

At this point, we will assume that the flux terms can be interpolated in the same way as the conservative variables (this is equivalent to using Lobato quadrature for the flux terms and does not affect the end result in any significant way)

$$\tilde{\mathbf{F}}_k \cong N_j \tilde{\mathbf{F}}_k(\vec{x}^j) = N_j \tilde{\mathbf{F}}_k^j \quad \textbf{(7)}$$

$$\int_\Omega N_i \left(N_j \dot{\tilde{\mathbf{\Phi}}}^j + \frac{\partial N_j}{\partial x_k} \left(\tilde{\mathbf{F}}_k^j + \tilde{\mathbf{G}}_k^j \right) \right) d\Omega = 0 \;\; for \;\; i = 1...n_{node} \quad \textbf{(8)}$$

The system can be written in matrix notation as:

$$\begin{aligned} \dot{\tilde{\mathbf{\Phi}}}^j &= \mathbf{M}^{-1}\,\mathbf{r} \\ \mathbf{M} &= \int_\Omega N_i\, N_j\, d\Omega \\ \mathbf{r} &= -\int_\Omega N_i \frac{\partial N_j}{\partial x_k} d\Omega \left(\tilde{\mathbf{F}}_k^j + \tilde{\mathbf{G}}_k^j \right) \end{aligned} \quad \textbf{(9)}$$

The expressions can be arranged in a way similar to what was described in the previous chapter. However, remark that in this case no assumption is made with respect to the shape of the elements. Therefore, the discussion is quite general (valid for 2D and 3D as well as for elements with different number of nodes and faces)

$$\mathbf{r^i} = -\sum_{j \neq i} \int_\Omega N_i\, N_{j,k}\, d\Omega\, \tilde{\mathbf{F}}_k^j - \int_\Omega N_i\, N_{i,k}\, d\Omega\, \tilde{\mathbf{F}}_k^i \quad where \quad N_{j,k} = \frac{\partial N_j}{\partial x_k} \quad \textbf{(10)}$$

To keep the notation compact, from now on sum will be assumed for the index ***j***, with the sum extended to all values of ***j*** except ***i***; there is no sum on index ***i***. To the same effect, in **(10)** only the convective terms have been included, the treatment of the diffusive fluxes is identical.

Integration by parts of **(10)** yields:

$$\begin{aligned} \mathbf{r^i} &= \int_\Omega N_{i,k}\, N_j\, d\Omega\, \tilde{\mathbf{F}}_k^{ij} - \int_\Omega N_{i,k}\, N_j\, d\Omega\, \tilde{\mathbf{F}}_k^i - \int_\Gamma N_i\, N_j\, n_k d\Gamma\, \tilde{\mathbf{F}}_k^j - \\ &- \frac{1}{2} \int_\Gamma N_i\, N_i\, n_k d\Gamma\, \tilde{\mathbf{F}}_k^i \quad where \quad \tilde{\mathbf{F}}_k^{ij} = \tilde{\mathbf{F}}_k^i + \tilde{\mathbf{F}}_k^j \end{aligned} \quad \textbf{(11)}$$

The expression must now be symmetrised to achieve the benefits of the edge storage

$$\begin{aligned} \mathbf{r^i} &= \frac{1}{2} \int_\Omega \left(N_{i,k}\, N_j - N_i\, N_{j,k} \right) d\Omega\, \tilde{\mathbf{F}}_k^{ij} + \int_\Gamma N_i\, N_j\, n_k d\Gamma\, \tilde{\mathbf{F}}_k^{ij} - \\ &- \int_\Omega N_{i,k}\, N_j\, d\Omega\, \tilde{\mathbf{F}}_k^i - \int_\Gamma N_i\, N_j\, n_k d\Gamma\, \tilde{\mathbf{F}}_k^j - \frac{1}{2} \int_\Gamma N_i\, N_i\, n_k d\Gamma\, \tilde{\mathbf{F}}_k^i \end{aligned} \quad \textbf{(12)}$$

Using the shape function property $N_i = 1 - \sum_{j \neq i} N_j$

$$\mathbf{r}^{\mathbf{i}} = d_k^{ij}\tilde{\mathbf{F}}_k^{ij} + b_k^{ij}\tilde{\mathbf{F}}_k^{ij} + c_k^{i}\tilde{\mathbf{F}}_k^{i}$$

$$d_k^{ij} = \frac{1}{2}\int_{\Omega}\left(N_{i,k}\,N_j - N_i\,N_{j,k}\right)d\Omega$$

$$b_k^{ij} = -\frac{1}{2}\int_{\Gamma} N_i\,N_j\,n_k\,d\Gamma \qquad \textbf{(13)}$$

$$c_k^{i} = -\int_{\Gamma} N_i\,N_i\,n_k\,d\Gamma$$

The ***d*** coefficients are antisymmetric, meaning that only half of them need be stored. Moreover, the ***b*** and ***c*** terms are zero for any interior edge so the storage requirements are greatly reduced.

The scheme is conservative, as for any couple of internal nodes the total contribution to the residual is zero.

$$\mathbf{r}_{\mathbf{e}}^{\mathbf{i}} + \mathbf{r}_{\mathbf{e}}^{\mathbf{j}} = d_k^{ij}\tilde{\mathbf{F}}_k^{ij} + d_k^{ji}\tilde{\mathbf{F}}_k^{ji} = d_k^{ij}\tilde{\mathbf{F}}_k^{ij} - d_k^{ij}\tilde{\mathbf{F}}_k^{ij} = \mathbf{0} \qquad \textbf{(14)}$$

When solving the complete set **(1)**, diffusive fluxes must be added to eq. **(13)** yielding:

$$\mathbf{r}^{\mathbf{i}} = d_k^{ij}(\tilde{\mathbf{F}}_k^{ij} + \tilde{\mathbf{G}}_k^{ij}) + b_k^{ij}(\tilde{\mathbf{F}}_k^{ij} + \tilde{\mathbf{G}}_k^{ij}) + c_k^{i}(\tilde{\mathbf{F}}_k^{i} + \tilde{\mathbf{G}}_k^{i}) \qquad \textbf{(15)}$$

To evaluate **(15)** the nodal values of the solution gradient are required in order to calculate the **G** terms. These are obtained through a smoothing procedure. Assuming the derivatives can be interpolated using the shape functions and using a weighted average we have:

$$\int_{\Omega} N_i N_j\,d\Omega\,\nabla_k\mathbf{\Phi}^j = \int_{\Omega} N_i N_{j,k}\,d\Omega\,\mathbf{\Phi}^j$$

$$\nabla_k\mathbf{\Phi}^j = \mathbf{M}^{-1}\int_{\Omega} N_i N_{j,k}\,d\Omega\,\mathbf{\Phi}^j \qquad \textbf{(16)}$$

Solution of the system of equations **(9)** and **(16)** can be achieved in an efficient way by means of an iterative procedure involving the lumped mass matrix

$$\mathbf{M}^{\mathbf{d}} = \delta_{ij}\sum_j\int_{\Omega} N_i\,N_j\,d\Omega \qquad \textbf{(17)}$$

$$\mathbf{M}^{\mathbf{d}}\,\tilde{\tilde{\mathbf{\Phi}}}_0 = \mathbf{r}$$

$$\mathbf{M}^{\mathbf{d}}\left(\tilde{\tilde{\mathbf{\Phi}}}_m - \tilde{\tilde{\mathbf{\Phi}}}_{m-1}\right) = \mathbf{r} - \mathbf{M}\tilde{\tilde{\mathbf{\Phi}}}_{m-1} \qquad \textbf{(18)}$$

2.3 CONVECTIVE STABILIZATION

As the basic Galerkin discretization is inherently unstable in presence of large convective fluxes, the scheme thus devised is prone to spurious oscillations which render it completely unusable. To overcome this limitation the interface fluxes are modified according to Roe's upwind scheme:

$$\tilde{\mathbf{F}}_k^{ij} \rightarrow \mathbf{F}_k^{ij} = \tilde{\mathbf{F}}_k^{i} + \tilde{\mathbf{F}}_k^{j} - \frac{1}{2}\left|\mathbf{A}_{\vec{\mathbf{u}}^{ij}}\right|\Delta^{ij}\,\vec{\mathbf{u}}_k^{ij}$$

$$\vec{\mathbf{u}}^{ij} = \frac{\vec{\mathbf{l}}^{ij}}{\left\|\vec{\mathbf{l}}^{ij}\right\|} \qquad \vec{\mathbf{l}}^{ij} = \vec{\mathbf{x}}^{j} - \vec{\mathbf{x}}^{i} \qquad \Delta^{ij} = \tilde{\mathbf{\Phi}}^{j} - \tilde{\mathbf{\Phi}}^{i} \qquad \textbf{(19)}$$

$\left|\mathbf{A}_{\vec{\mathbf{u}}^{ij}}\right|$ being the positive flux Jacobian evaluated along the direction of the edge. As explained in the previous chapter, the scheme **(19)** is only first order accurate. To recover a higher order of accuracy the nodal states are replaced with extrapolated values at the interface as explained in part 1.4.2

$$\mathbf{F}_k^{ij} = \tilde{\mathbf{F}}_k^i + \tilde{\mathbf{F}}_k^j - \frac{1}{2}\left|\mathbf{A}_{\vec{\mathbf{u}}^{ij}}\right|\left(\tilde{\mathbf{\Phi}}^{j-} - \tilde{\mathbf{\Phi}}^{i+}\right)\vec{\mathbf{u}}_k^{ij} \qquad \textbf{(20)}$$

These extrapolated values are calculated using a limited MUSCL scheme in the same way described in 1.4.2 using the recovered gradients **(16)** to carry out the extrapolation.

2.4 TIME INTEGRATION

A multi-stage Runge-Kutta scheme is used as described in 1.5. The local time step is calculated accounting for convective and diffusive transport, thus:

$$\Delta t_i = \min\left\{CFL\frac{h_i}{\|\vec{u}\| + a}, \frac{h_i^{\,2}}{\upsilon}\right\} \qquad a^2 = \gamma\frac{p}{\rho} \qquad \textbf{(21)}$$

In order to accelerate convergence an implicit residual smoothing step is carried out before solving

$$\bar{\mathbf{r}}^i = \mathbf{r}^i + \varepsilon\sum_j\left(\bar{\mathbf{r}}^j - \bar{\mathbf{r}}^i\right) \quad \text{for all } j \text{ connected to } i \qquad \textbf{(22)}$$

Eq. (22) is solved using a Jacobi iterative process

$$\bar{\mathbf{r}}_n^{\,i} = \frac{\mathbf{r}^i + \varepsilon\sum_j \bar{\mathbf{r}}_{n-1}^{\,j}}{1 + \varepsilon\sum_j 1} \qquad \textbf{(23)}$$

2.5 A METHOD FOR COUPLED EULER-BOUNDARY LAYER SOLUTION

In many cases of interest the viscous effects are confined into a small volume next to the wall (the so called boundary layer). A common approach in these cases is to solve the thin layer equations to determine the effect of diffusion while the Euler equations account for the outer inviscid flow. The coupling between the two fields is achieved by means of the displacement thickness. This algorithm is less costly than the solution of the complete RANS equations over the whole domain while providing results of acceptable quality as long as the boundary layer remains attached.

The traditional way of solving the boundary layer equations is to cut 2D slices of the solid body and solve the 2D version of the boundary layer equations over each slice using integral formulations. This approach, while quite simple, cannot account for 3D effects inside of the boundary layer. To overcome this limitation a 3D variant has been devised which combines low computational cost with increased accuracy with respect to the schemes in use.

The 3D boundary layer equations (the energy term has been dropped to make the exposition more compact) can be written in a curvilinear reference system as

$$\frac{\partial\mathbf{\Phi}}{\partial t} + \frac{\partial\mathbf{F}_\xi}{\partial\xi} + \frac{\partial\mathbf{F}_\eta}{\partial\eta} + \frac{\partial\mathbf{F}_n}{\partial n} = 0 \qquad \textbf{(24)}$$

where ***n*** represents the direction perpendicular to the wall. The flux vectors (in this case **F** includes both the convective and diffusive parts) are simplified due to the fact that normal gradients are larger than tangential ones

$$\mathbf{\Phi} = \begin{bmatrix} \rho \\ U_\xi \\ U_\eta \end{bmatrix} \mathbf{F}_\xi = \begin{bmatrix} U_\xi \\ u_\xi U_\xi + p \\ u_\xi U_\eta \end{bmatrix} \mathbf{F}_\xi = \begin{bmatrix} U_\eta \\ u_\eta U_\xi \\ u_\eta U_\eta + p \end{bmatrix} \mathbf{F}_n = \begin{bmatrix} U_n \\ u_n U_\xi - \tau_{\xi n} \\ u_n U_\eta - \tau_{\eta n} \end{bmatrix} \qquad \textbf{(25)}$$

Solving these equations requires the definition of a local reference frame at each point of the solid surface. Whereas this may be achieved without much trouble when using structured a structured mesh and dealing with simple geometries, in the case of an unstructured grid over an arbitrary shape the problem becomes quite difficult.
To overcome this problem the equations are written instead in the global reference frame, much in the same way as **(1)**. The equations are solved using a cell centered finite volume discretazation on a hybrid (structured in normal direction). To achieve this goal a "virtual" hybrid mesh of the boundary layer is created using the surface discretization of the solid.

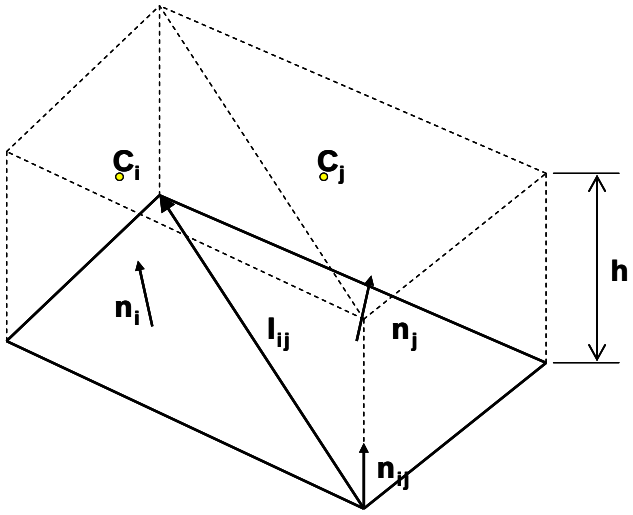


Fig. 1 - Virtual Cells Over Solid surface

As these cells only exist internally to the solver, no real mesh has to be created for the boundary layer; the boundary mesh from the fluid volume is all that is needed.
The average normal between two surface facets is defined as

$$\mathbf{n}_{ij} = \frac{\mathbf{n}_i + \mathbf{n}_j}{\left\| \mathbf{n}_i + \mathbf{n}_j \right\|} \qquad \textbf{(26)}$$

then, a transfer coefficient from cell ***i*** to cell ***j*** can be defined as

$$\mathbf{C}_{ij} = h\, \mathbf{n}_{ij} \otimes \mathbf{l}_{ij} \qquad \textbf{(27)}$$

The flow of a conservative variable from cell ***i*** to cell ***j*** can be written as

$$\int_{\Gamma_{ij}} \mathbf{F} \cdot \mathbf{n}\, d\Gamma = \tilde{\mathbf{F}}_{ij} \cdot \mathbf{C}_{ij} \qquad \textbf{(28)}$$

where the interface flow being calculated trough interpolation between states ***i*** and ***j***. During this stage the thin layer assumption is enforced by retaining only the diffusive fluxes in normal direction (which, due to the hybrid nature of the grid can be easily calculated using standard finite differences).
The 3D system of momentum equations is solved to calculate a trial momentum value at the end of the step:

$$V_i \frac{d\mathbf{U}_i}{dt} = \sum \tilde{\mathbf{F}}_{\mathbf{ij}} \cdot \mathbf{C}_{\mathbf{ij}} \Rightarrow \mathbf{U}_i^*(t+\Delta t) \quad \textbf{(29)}$$

Then, the normal component is subtracted from **(29)**

$$\mathbf{U}_i^{**} = \mathbf{U}_i^* - \mathbf{n}_i\left(\mathbf{U}_i^* \cdot \mathbf{n}_i\right) \quad \textbf{(30)}$$

The true momentum is obtained by adding the correct value of the normal component:

$$\mathbf{U}_i = \mathbf{U}_i^{**} + \mathbf{n}_i U_i^u \quad \textbf{(31)}$$

The correct value of the momentum is determined by forcing the conservation of mass on each cell which yields the equation for U_i^n

$$\int_{\Gamma_{cell}} \mathbf{U} \cdot \mathbf{n}\, d\Gamma = 0 \quad \textbf{(32)}$$

Once the displacement thickness has been determined from the solution of the boundary layer, coupling with the outer inviscid flow is achieved by forcing a transpiration velocity on the wall surface

$$v_n = v_{Euler} \frac{\partial \delta^*}{\partial s} \quad \textbf{(33)}$$

s being the streamline direction and $\mathbf{v}_{\text{Euler}}$ representing the slip velocity from the inviscid solution.

2.6 AN EXAMPLE APPLICATION

The code thus developed has been successfully applied to industrial scale aerodynamic problems. One such case is the analysis of the flow field over a transonic jet test model in order to determine the interference and aeroelastic effects due to the support mechanism. While the problems is highly complex due to its size, the analysis was carried out efficiently on desktop computers due to the highly optimized nature of the code.

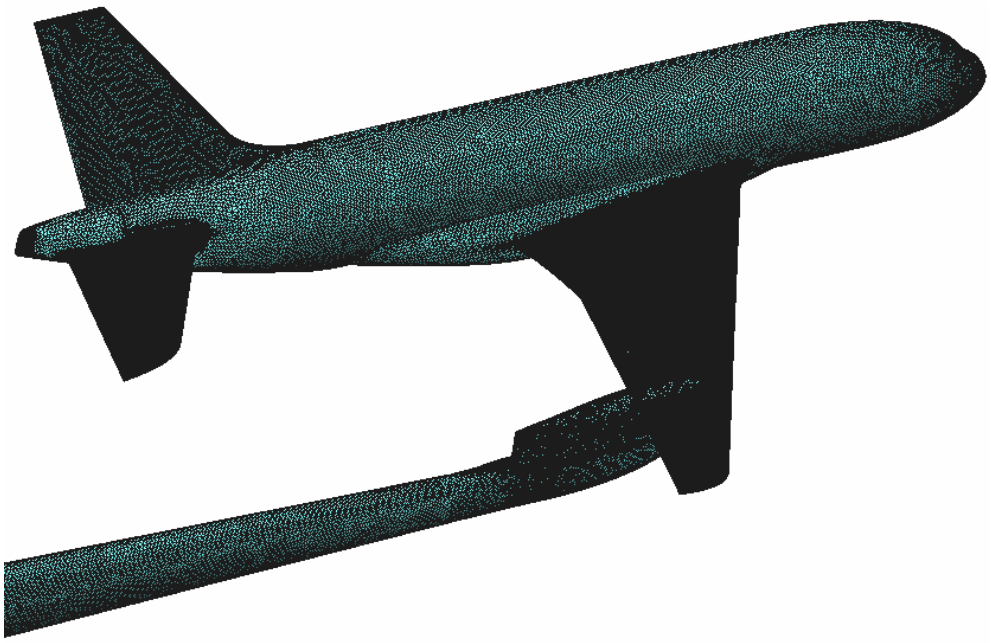


Fig. 2 - Surface Mesh of Test Model (3D mesh includes 11M cells)

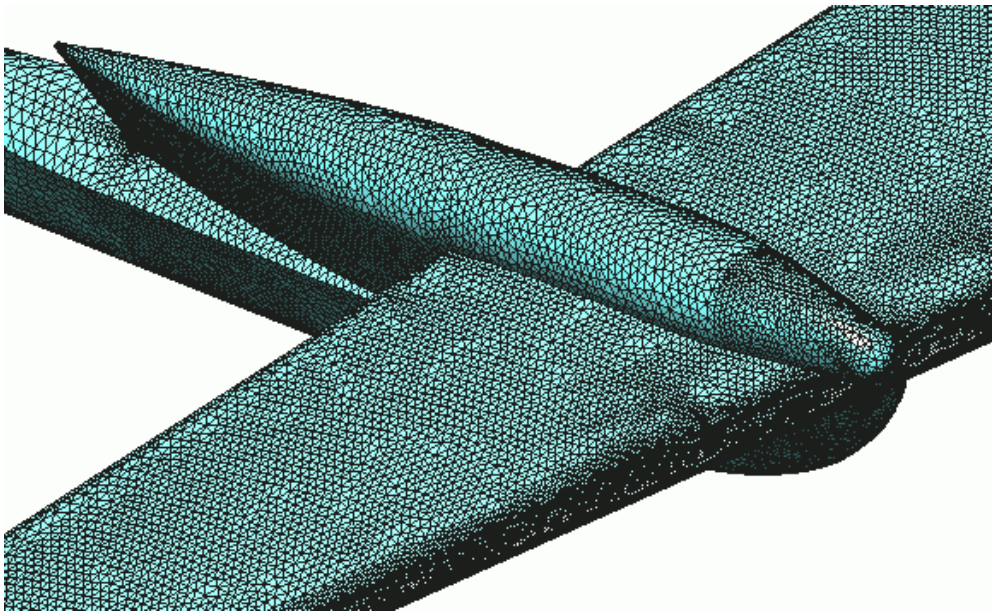


Fig. 3 - Detail of Mesh Around Support Attachment

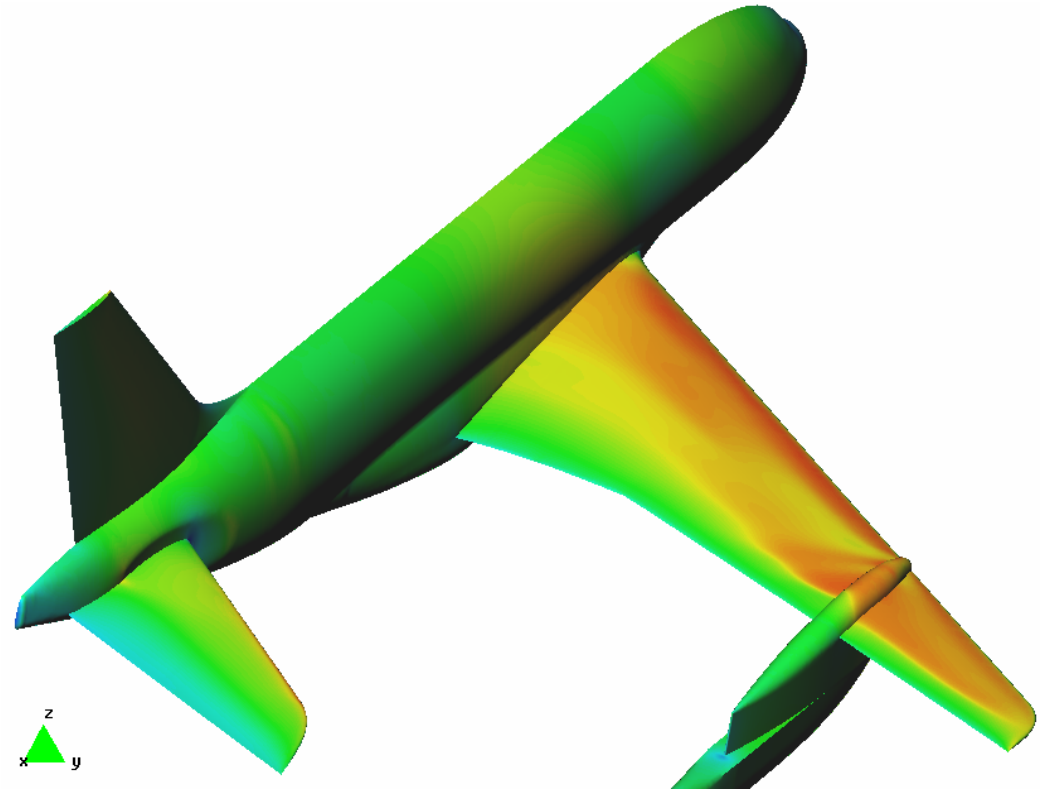


Fig. 4 - Pressure Contours In Transonic Flight

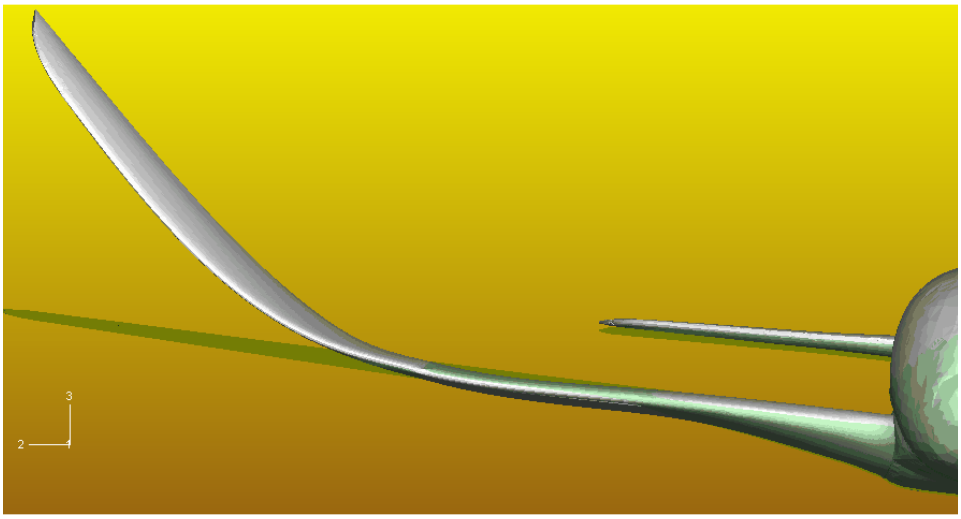


Fig. 5 - Wing Deformation Due To Aerodynamic Loads

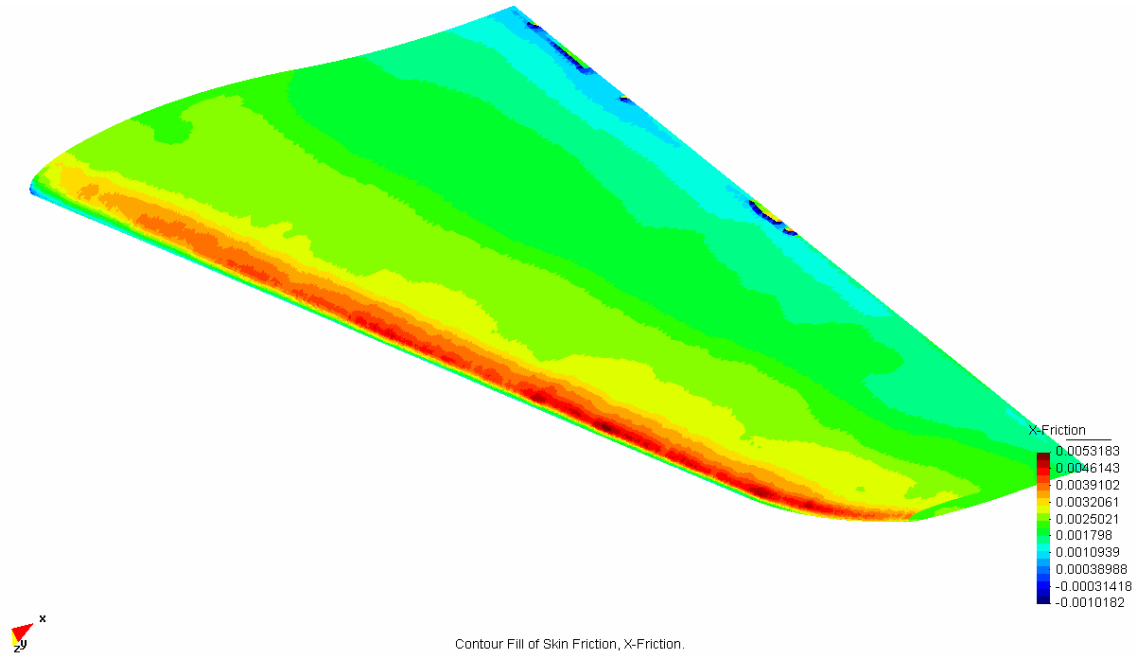


Fig. 6 - Skin Friction Contours Calculated With The 3D Boundary Layer Code

BASIC STRUCTURE OF KRATOS
An Object-Oriented Environment for the Development of Multi-Physics Analysis Software

Overview

Complexity of multi-physics problems from one side, and the wide variety of problem-types from the other, lead the finite element developers to design their program structures as flexible and extendable as possible in order to avoid an unnecessary growing of the implementation effort. On the other hand, for development of interesting applications in this field a group of professionals with a variety of knowledge is needed. Providing a framework and a proper structure where this group of developers can work with each other are relevant aspects to design a new multi-physics code.

Kratos has been designed to be a common environment for several kinds of activities in the field of multi-physics analysis area. Developers of finite element formulations can embed their problem type in Kratos while interacting with others in order to solve a multi-physics problem. Application developers who try to develop a new application in specific or general multi-physics area also can enjoy from this common environment. Even more, within this framework several contributions can interact together without conflict. Finally, end users can take advantages of multi-physics applications generated with Kratos as well as of Kratos itself.

In this report, we will discuss the object-oriented structure of Kratos. The methodology to achieve the structure, key aspects and advantages of each part are also briefly presented. We also comment the multi layer nature of Kratos and its benefits to achieve the goals described above.

1 Introduction

One of the relevant topics in numerical methods nowadays is the combination of different analysis (thermal, fluid dynamic, structural) with optimization methods, adaptive meshing and different formulations in one global software package with just one user interface and the possibility to extend the implemented solution to new types of problems, as an approach to a multi-physics simulation environment.
The automotive, aerospace or building sectors have traditionally used simulation programs to improve their products or services, focusing the computation in the major physical phenomena: fluid dynamics, structural, metal forming, etc. Nevertheless, the new needs for safer, cheaper and more efficient goods together with the impressive growth of the computing facilities, demand an equivalent solution for multi-physics problems. Moreover, new applications in the food, chemical and electromagnetic industry, among others, are not useful at all without a multi-physics approach.
Some illustrative cases can be found in the design of sails and sailboats where structural and fluid dynamic computations have to be taken in account, sterilization processes (thermal and fluid dynamic analysis), strong magnet design (structural, thermal and magnetic analysis) or photovoltaic cells (thermal and electrical computations), among many others.

1.1 Demands

The world of computing simulation has experienced great progresses in recent years, and requires more exigent multidisciplinary challenges to satisfy the new upcoming demands.
Due to the industrial maturity of one-purpose codes for the different fields, the production sector has increased its expectations, because realizes the need for solving in a more realistic way the problems they deal with, in order to stay competitive. Strong simplifications are the reason for which difficult problems could be solved in the past. These simplifications lead to a model as different from the real one, as the severity of the assumptions made. If we add the every required accuracy in a concurrent world, then we need to relax the assumptions and become more general in the way of solving multidisciplinary problems.
The situation is not really new. What is novel is the need for solving multidisciplinary problems combining most of the information coming from different fields, obtaining a more precise information and therefore better optimization methods. This means to solve more complex problems. There are many strategies to approach the solution of this kind of problems. A simple approach assumes that the work done is worth enough and that people should use already existing codes to try to solve Multi-Physics problems. Unfortunately in most of the cases this strategy will simply not work when there is a strong coupling between the different domains that each code has to solve.
One of the crucial problems in the numerical simulations corresponds to the mesh generation, that means the necessary expertise to get good solutions of the problem, by avoiding numerical errors due to the use of bad elements (incorrect size or aspect ratio). This ability can be different for each physical field, because a suitable mesh is directly related with the obtained error that depends on the simulation. As consequence, a multi-physics environment should contribute to improve this situation. In this sense, knowing the physics of the problem helps to define a good mesh that will represent better the results.
It has been also planned to integrate another interesting topic in Multi-Physics such as Optimization.
Transversal demanded features for the Kratos framework are extendibility, efficiency and continuity of work. The extendibility of the code guarantees the state of the art in the framework. It makes easy to add new capabilities to the programs. The efficiency makes the program attractive to either researchers and end-users: as above said, concurrence is a constant value nowadays.

1.2 Background

Other existing initiatives can be found to provide a general framework to implement numerical methods without paying attention on the programming. Two good examples are POOMA (Parallel Object-Oriented Methods and Applications, a framework for applications in computational science requiring high-performance parallel computers) [2] and Deal.II (a C++ program library targeted at adaptive finite elements and error estimation) [3]. A previous project in CIMNE called FemLab created a C++ library to use for any finite element formulation [4]. All of these initiatives are an excellent starting point which has been taken in account to build Kratos.

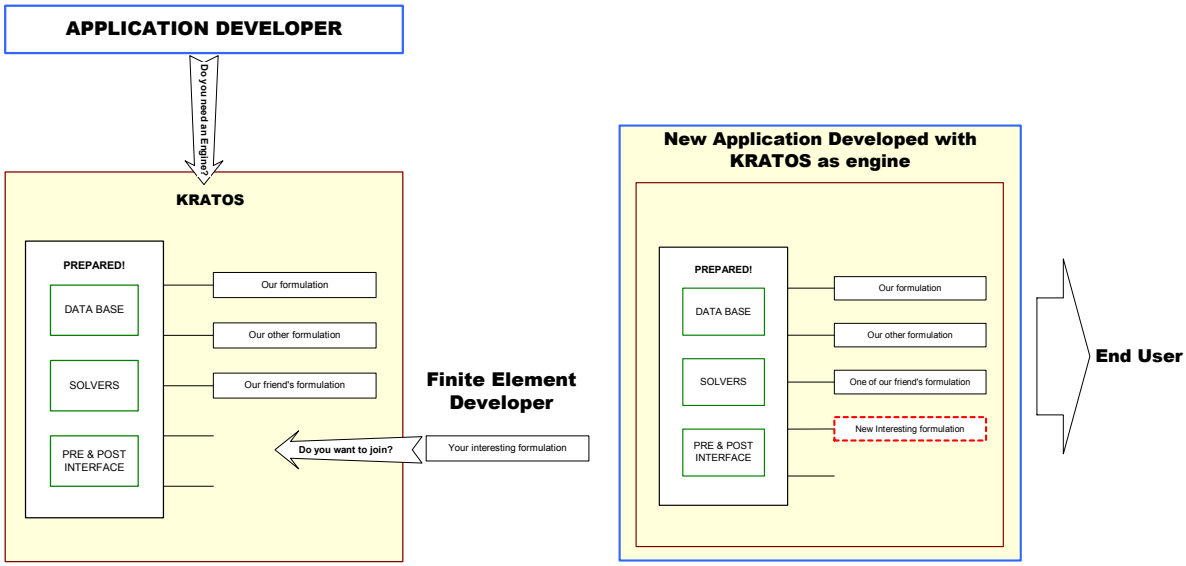


Figure 1: Kratos framework and extensions

2 Kratos Approach

We have identified that in a general enough framework, it is possible to fit any FEM code with very little effort. It is important to provide this facility in order to extend the capabilities of each program. Thus, we aim to provide a very general framework in which a code can be written by combining basic modules putting the effort just in the differences respect already existing modules. This information is typically the formulation of the element. However, many solving procedures differ from the standard FEM strategy, and the framework has to be able to find room for them. Moreover, it is essential to provide easy tools to customize the environment to the needs of any algorithm.

Kratos has as main objective to establish a framework, a methodology and a computing structure to allow the building of multi-physics finite element programs at different levels of implementation. Depending on the interest of the user, the package can be used as a high level library to explore new finite element formulations or to check features of new implementations for industrial applications.

Kratos consists in a set of classes and methods for programmers, providing the ability to handle multi-physics, adaptive meshing and optimization problems. It has been designed as a user-developer approach, considering different levels of contributions to the Kratos system, as well as plug-in extensions (Figure 1).

Kratos will help to build a numerical application in C++ from the simplest formulation (heat conduction) to the most complex one (optimization techniques).

Kratos includes a revision of the state of the art on C++ programming for numerical applications: existing codes and external free sources.
The aim of the first development phase is to provide simulation programs' developers with the ability to implement a new application in a few days, using a few test programs and the Kratos library.
A set of initial user-developers has been defined, in view of the simulations to be performed:

- basic heat conduction problem;
- transient thermal or low frequency electromagnetic problems;
- thermo-structural-fluid dynamic coupled problems;
- adaptive meshing optimization problems;

Last but not least, reusability and program maintenance is one of the main priorities within the Kratos philosophy. For this purpose, special attention is put in the modular aspects of the Object Oriented Programming and in the creation of an intranet platform to share all the Kratos documentation including methodology, structure, project progress and source code.

3 Kratos' structure

3.1 Object-oriented structure

History of object-oriented design for finite element programs turns back to early 90's and even more [7, 9, 10, 11, 13]. Before that, many large finite element programs were developed in modular ways. Industry demands for solving more complex problems from one side, and the problem of maintaining and extending the previous programs from the other side, has lead developers to address their design strategy towards an object-oriented one [2, 3, 4, 5, 6].
The main attempt of an object-oriented structure is to split the whole problem into several objects and to define their interfaces. There are many possible ways to do this for each kind of problem we want to program and the functionality of the resultant structure depends largely on it. In the case of finite element problems there are also many approaches such as constructing objects based on partial differential equations solving methods [2] or in the finite element method itself [9].
In Kratos we have chosen the second approach and have constructed our objects based on a finite element general methodology. This approach was selected because our goal was to create a finite element environment for multidisciplinary problems. Also our users were, in general, more familiar with this methodology than with physical properties. In addition, this approach has given us the necessary generality mentioned above in the objectives of Kratos. Within this scope we take our main objects from various parts of the finite element method structure. Then, we define some abstract objects for implementation purposes. Finally we define their relation and try to balance their responsibility.
The main design effort is focused on the "element" object. An element has all the data necessary for its self calculation, and this makes it independent. In fact, element and its accessories together make the Finite Element Developer Layer. This will help the developers of new formulations to write their own elements and add them to the structure without any change in other parts of the program.
All the necessary data during the process is encapsulated and structured in one single object. We hide the memory management and the data management through the implementation of model, also using a façade pattern [8] for the model, preparing a simpler interface for the whole database. This should help other developers to write their problems without being concerned on this side of the implementation. From the point of view of multi-physics problems, this database provides an unified method through which any problem-type can have access to the data belonging to other problem-types. This feature is the most important part of a multi-physics environment.

3.2 Multi-Layer Structure

One of the strategies of Kratos is its multi-layer approach. Why did we try to layer the Kratos' structure and which is the benefit of it? On one hand, dividing the structure into layers allows us to concentrate and specialize our design on objects and their interfaces, taking advantage of the knowledge of the developers that will program in that layer. On the other hand, each layer working group must use a restricted interface to the other layers and this will reduce the dependency inside the program. Of course by decreasing the dependency, understanding each part will be easier and, meanwhile, maintaining the code for a long time would be facilitated.

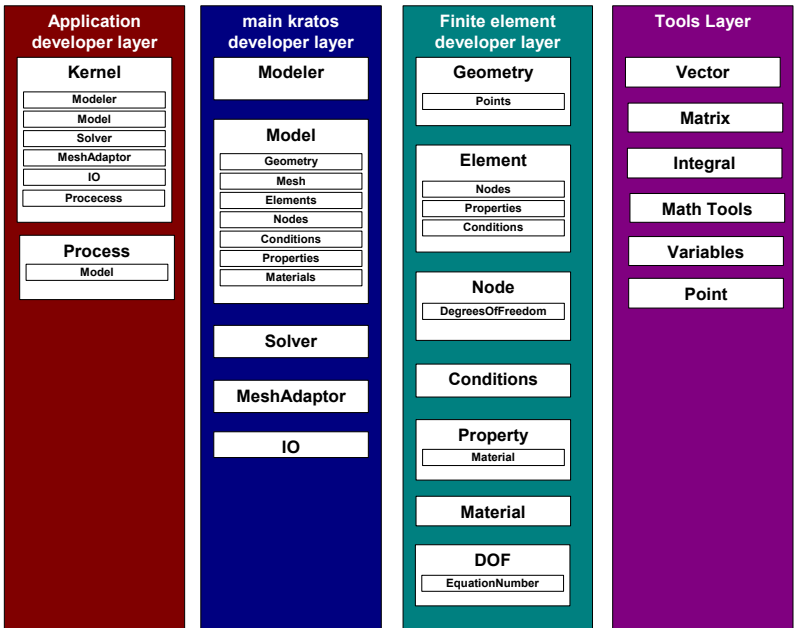


Figure 2: Layers of Kratos

In designing the layers of the structure we assume three types of developers:

- main developers of the program, who are involved in the most technical programming part, and try to expand the basic abilities of the program (also called bricks developers);
- element developers, who are mainly interested on finite element programming aspects, from the physical and mathematical point of view, without particular experience in C++ programming;
- application developers who will use the entire library as a customizable finite element program where they can just to activate or deactivate certain functionalities or program a new global algorithms (i.e. interface programmers in finite element application projects and/or application managers).

Each users group is respectively assigned one out of the three structure layers as shown in Figure 2 plus one layer for the programming basic tools. The first and top layer is the Application Developer Layer and is provided by an object that hides other parts of the implementation from the application point of view. This layer has an external interface for other applications and provides the necessary tools and methods. The Main Developer Layer gathers most general parts from libraries. This part is defined to hide C++ and algorithm implementation technical concepts from the two other layers. It has an interface with the Application Developer Layer and another one with the Finite Element Developer Layer in order to provide a standard method for implementation. The Finite Element Developer Layer provides a common group of objects, that are important from the point of view of a new formulation. This layer provides an interface to the previous layers and also to the Tools Layer. Therefore, these

interfaces are used as common standards for connecting to other parts of the program. The lowest layer is the Tools Layer. It provides the basic components for the rest of the program. Using these standard tools through the project ensures the portability and maintenance of the code.

3.3 Performance and Efficiency

For the design of many parts of the program we focus on the performance (understood as computation speed) rather than on the efficiency (understood as memory management). This strategy comes from the fact that for usual problems there is usually enough computer memory, and for large scale ones it is always cheaper to find a machine with enough memory than finding a faster machine. Meanwhile, we try to store every data just once with several references to it. Thus, in this way we preserve memory while the references increase the performance as we save the searching time for several entities.

4 Kratos and Multi-physics

4.1 Element sets

One of the most important aspects of a multi-physics environment is the ability to handle several domains and process them in a different manner. The idea of "Element sets" comes from this aspect. Each element set in Kratos is a union of elements to be processed in the same way. Solver, result-tables, and other processes that work on elements can be different for each set of elements. There is no restriction for an element to be just in one set. This strategy allows us to group some sets of elements in one set. This is specially helpful when the whole set of equations needs to be solved simultaneously as the value of certain physical unknowns is needed to solve other physical domains.

4.2 Sharing data

As mentioned before, we have one single database for all the different problem-types. To achieve full multi-physics capability we need also to establish a uniform database interface for all domains. Just by having this uniform database interface we can guarantee the accessibility of all the objects to their necessary data. A set of functions provides this interface in Kratos. Polynomial functions, interpolators, and table search functions are examples of these interface functions. By this function the interrelation between the parameters and unknowns can be established in a very generic way, for example, allowing conductivity depend on temperature values. Properties, sources and other objects in the Finite Element Developer Layer use these functions to extract data from the model. Kratos provides also a uniform interface for all the properties and sources. This is the final key to have a general interface between the element and the model as a database.

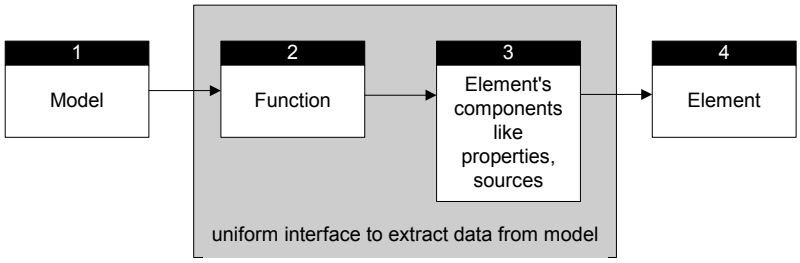


Figure 3: Interface between the elements and the model

4.3 Degrees of freedom

One of the representative features of multi-physics analysis is the variety of degrees of freedom in each node. Each degree of freedom in Kratos is encapsulated in a Dof object. Each Dof provides the information of what variable stores (temperature, x displacement, etc.), its location in the global system of equations, its condition and also its value. This Dof encapsulation provides an easy and flexible way of adding new unknowns to the system.

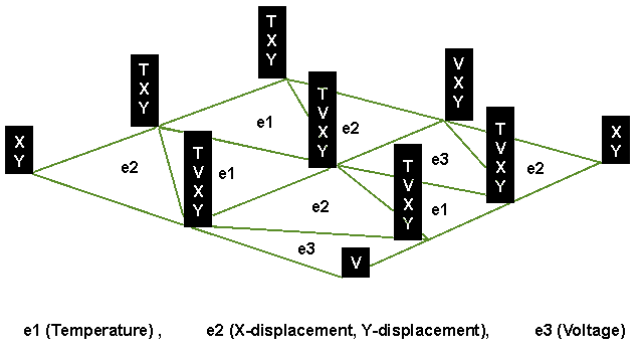


Figure 4: Dynamic arrays of the degrees of freedom

Each node has a dynamic list of its degrees of freedom. Using this dynamic list of degrees of freedom gives us the desired flexibility while avoiding unnecessary use of the space. These lists are filled during run time by the elements. The idea of filling these lists by the elements comes from the fact that only each element knows what kind and how many unknowns it has. This idea provides a simple but effective mechanism to work with several elements in different problem-types. At the initializing time, each element asks to its nodes to add its Dofs to their lists. Then, each node adds these Dofs if they do not already exist in the list.

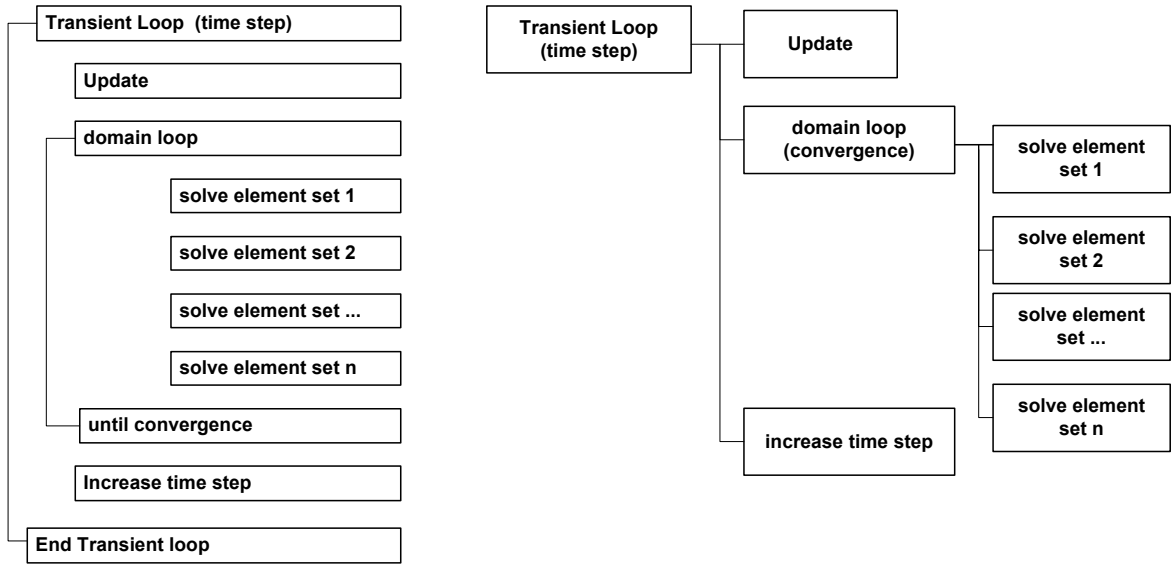
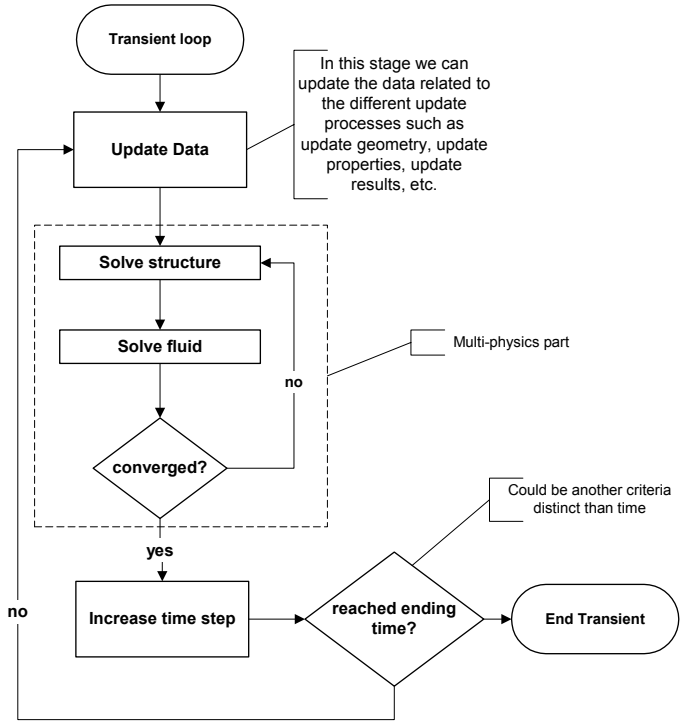


Figure 5: Scheme of a Multi-Physics transient problem and the corresponding process tree

4.4 Program flow

In general, each group of problem-types has its own solving algorithm [1]. Transient and steady-state, one domain or multi-domain, two-three-four field problems, etc. are examples of the variety of processes in different problems. A possible approach is to provide some high level classes to manage these algorithms in a global form [12].
In Kratos, the high level classes mentioned above are called "processes", and they are used to achieve a flexible and customizable flow of the program. You can consider each process as a construction unit for building the global algorithm. Each single step of the finite element methodology is encapsulated in one process. Each process can have several features too, like loops, conditions and normal statements. A compound pattern [8] is used to design the process. Each process can have several sub-processes and will execute them as well as its own statements. Finally, by having a good interface we can customize Kratos' flow very easily from outside as well as from inside (application layer).




```
1  Begin Transient Loop

2  Update Data

3  Solve element set
   3.1 Repeat 3 for each element set

4  Check convergence
   4.1 If not converged:
            go to 3
   4.2 Else: continue

5  Increase time step

6  Check ending time
   6.1 If not reached ending time:
            go to 2
   6.2 Else: continue

7  End Transient Loop
```

Figure 6: Flow-chart and pseudo-code of the transient loop for a fluid-structure interaction code

4.5 Equation solver

A very simple example of how a wide scope programming can be effective and integrate a wide range of different cases, is the implementation of equation solver. This class is responsible for solving any system of equations generated by Kratos, either linear or non-linear. It works using a Newton-Raphson algorithm [14], where the Jacobian matrix and the residual vector are provided and assembled by the element class. However, as the Jacobian matrix is computed outside of the algorithm, the element can provide to the equation solver any kind of matrix. If the problem is too complex to compute the derivatives, the Element Developer can choose to provide a secant matrix instead of a tangent one, and then the system is equally solved assuming that the approach is good enough. For either linear or non-linear systems of equations, the element must give to the equation solver the residual vector instead of the classical forces vector. In the case of a linear problem there is the possibility to fix the number of

non-linear iterations to 1. Finally, the Jacobian matrix in linear systems of equations is equal to the classical stiffness matrix, so it is straight forward for the Element Developer to integrate his element inside Kratos.
The way of fixing the degrees of freedom in the System Matrix is by zeroing the rows related to the Dof's fixed, except for the diagonal value that is set to 1. The values in the System Vector related to the fixed Dof's are set to zero as it is assumed that their residuals are zero. This algorithm destroys the possible symmetry of the matrix. There is no need to modify the System Vector in any other way, because it is calculated as a residual and the values of the fixed Dof's are already taken into account inside the elements.

5 Further Work

One of the main objectives of Kratos is to contribute to facilitate the easy and fast implementation of finite element programs. Extensive documentation and web-based presentations will be prepared to disseminate Kratos progresses, as well as to invite researchers to develop their own applications and element formulations within Kratos.
Apart from working on the quality of current parts, the work will be focused on a robust and flexible script language interpreter to complete the environment, as well as on tools for converting partial differential equations into Kratos' elements.

For more information, please visit http://www.cimne.com/kratos

References

[1] F.J. Royo. *Something About Multi-physics Problems and Programs.* Internal Report, CIMNE, (2001)

[2] Pooma: http://www.acl.lanl.gov/Pooma

[3] Deal.II: http://gaia.iwr.uni-heidelberg.de/~deal

[4] M. Galindo. *FemLab v. 1.0.* Technical Report n. IT-114. Ed. CIMNE. Barcelona (1994).

[5] FemLab MATLAB http://www.femlab.com/femlab/

[6] OOFELI, An object finite element libray: http://wwwlma.univ-bpclermont.fr/~touzani/ofeli.html

[7] J. Lu, D. White and W.F. Chen. *Applying Object-Oriented Design to Finite Element Programming*. Purdue University. ACM (1993).

[8] E. Gamma, R. Helm, R. Johnson and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley Longman, Inc. (1998).

[9] T. Zimmermann, Y. Dubois-Pèlerin and P. Bomme. *Object-oriented finite element programming: I Governing principles*. Computer Methods in Applied Mechanics and Engineering 98 (1992) 291-303.

[10] Y. Dubois-Pélerin, T. Zimmermann and P. Bomme. *Object-oriented finite element programming: II. A prototype program in Smalltalk*. Computer Methods in Applied Mechanics and Engineering 98 (1992) 361-397.

[11] Y. Dubois-Pélerin and T. Zimmermann. *Object-oriented finite element programming: III. An efficient implementation in C++*. Computer Methods in Applied Mechanics and Engineering 108 (1993) 165-183.

[12] Y. Dubois-Pélerin and P. Pegon. *Improving modularity in object-oriented finite element programming*. Communications in Numerical Methods in Engineering. Vol 13 (1997) 193-198.

[13] B.W.R. Forde, R.O. Foschi and S.F. Stiemer. *Object–Oriented Finite Element Analysis*. Computers and Structures, 34 (1990) 355-374.

[14] O.C. Zienkiewicz and R.L. Taylor. *The Finite Element Method*. Volume 2. Fifth Edition. Butterworth-Heinemann 2000.

Some example applications of KRATOS to FSI analysis

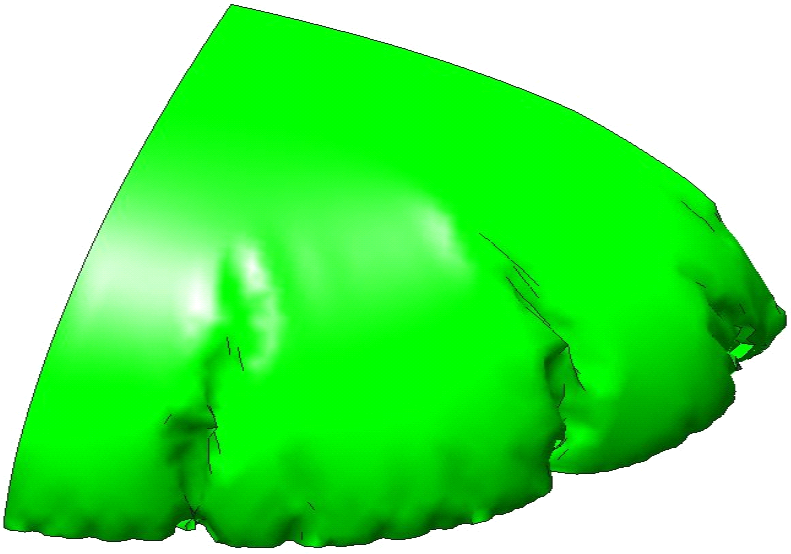


Fig. 1 - Airbag Deployment

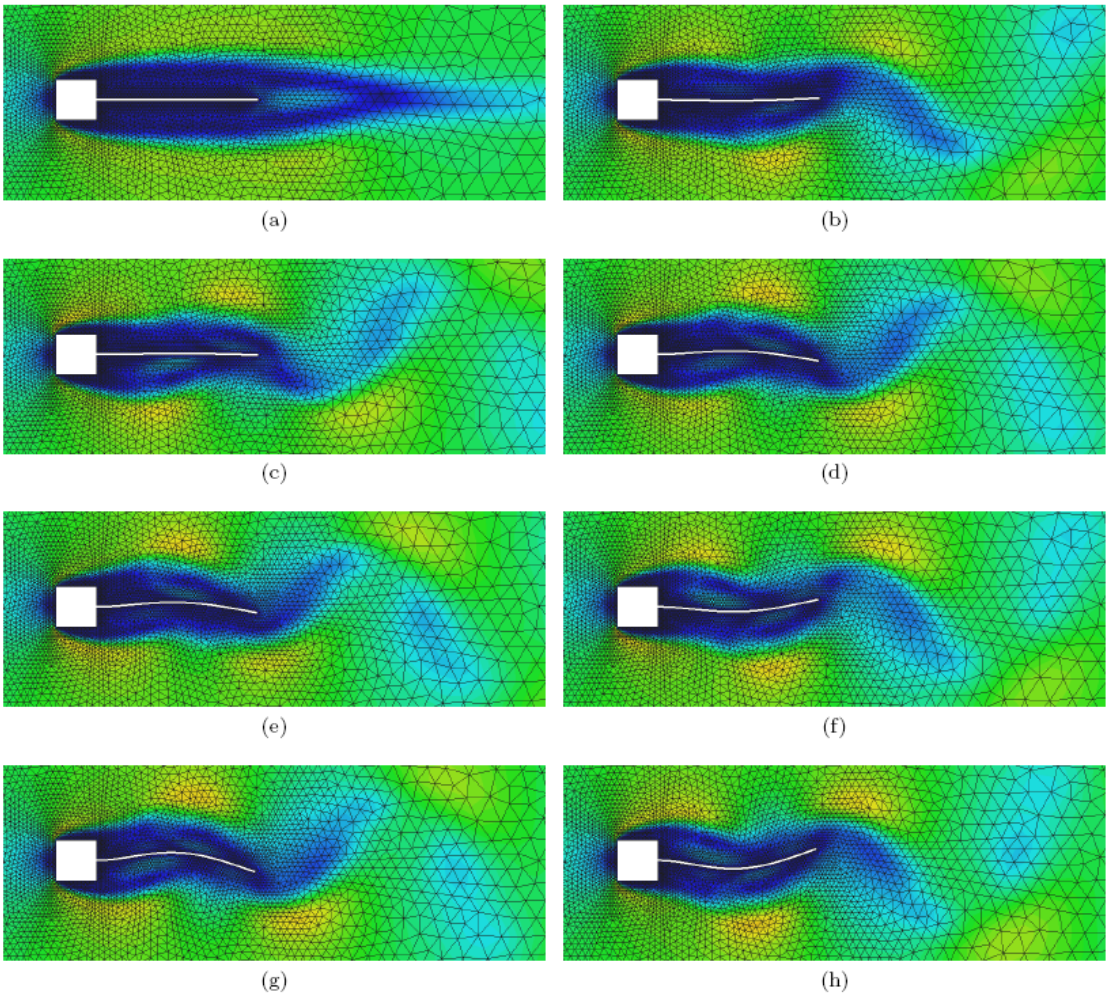


Fig. 2 - Flag Flutter

Analysis of some fluid-structure interaction algorithms

1 Overview

Over the last 40 years the FEM has experienced an exponential growth leading to the definition of a set of reliable computational techniques for many different problems. This maturity, together with the increasing availability of powerful computational resources gave rise in relatively recent times to an increasing interest in coupled problems. The field of Fluid-Structure Interaction (FSI) has important applications ranging from the bio-medical to the aeronautical fields. Different approaches were developed over the years for solving a FSI problem which remains however far from being closed. This document addresses the theoretical analysis of a partitioned coupling procedure for FSI. This is achieved by choosing a simple but representative model problem, which allows the development of analytical results. A loose coupling procedure is presented in the first part of the paper. An iterative version of the same algorithm is presented next together with an analysis of its convergence properties and a brief discussion on a possible acceleration strategy. Finally a new iterative strategy, based on a modified equations approach is described. Some examples of application of the proposed algorithm are then presented in application of problems of bridge engineering.

2 Introduction

The coupling between different physical phenomena constitute a classical challenge in many applications of practical interest. Empirical or analytical techniques, which classically provide a satisfactory description of coupled phenomena at the asymptotic limits, do not guarantee the necessary guidance when the coupling becomes strong. The field of FSI (fluid-structure interaction) represents an important and active area of research due to its great economical importance and its relevance in many practical problems in science and engineering.

Until recent times the numerical simulation of strongly coupled FSI problems was considered to be unviable. Experimental wind-tunnel testing constituted for long the only real approach to the design of complex aeroelastic interaction-sensitive structures such as bridge decks or slender towers. Although well established, wind tunnel testing is often expensive and time-consuming. In addition some concerns on the quality of its predictions may also arise when severe scale reduction is needed to fit the model in the test facility.

The increasing maturity of numerical Finite Element (FE) techniques in the fields of Computational Fluid Dynamics (CFD) and structural dynamics, together with the increasing computational power start nowadays to make competitive the purely numerical approach for solving FSI problems.

From a mathematical point a wide class of coupled problems can be repre-

sented symbolically in a form of the type

$$f_S(x,y) = 0 \tag{1}$$
$$f_F(x,y) = 0 \tag{2}$$

where $f_S(x,y)$ and $f_F(x,y)$ are functions that represent the behaviour of the two coupled systems. which can be linearized following the Newton-Raphson approach to give a system in the form

$$\begin{pmatrix} \frac{\partial f_S}{\partial x} & \frac{\partial f_S}{\partial y} \\ \frac{\partial f_F}{\partial y} & \frac{\partial f_F}{\partial y} \end{pmatrix} \begin{pmatrix} dx \\ dy \end{pmatrix} = \begin{pmatrix} -f_S(x^k, y^k) \\ -f_F(x^k, y^k) \end{pmatrix} \tag{3}$$

where the off-diagonal terms of the tangent (or Jacobian) stiffness matrix express the dependency of one field on the variations of the second. This form guarantees theoretically optimal quadratic convergence properties. In practice unfortunately the two symbolic operators $f_S(x,y)$,$f_F(x,y)$ are not explicitly known and the computation of the derivatives may be very difficult or indesiderable for other reasons.

Even if theoretically possible, the linearization of the fluid field is usually not performed, as it is often preferred to rely on fixed point type iterations to solve the non linearities. This implies that the calculation of the jacobian in Eqn(3) would involve a redefinition of the fluid strategy which is not desiderable from the point of view of software modularity. This makes generally unattractive the use of full Newton schemes for the solution of FSI problems. Further the resulting "monolithic" matrix, would inherit from the corresponding fluid counterpart the same conditioning problems, making its inversion difficult using any iterative technique.

Two main approaches exist for solving the impasse. The "purely algebraic" one focuses on the techniques to solve iteratively Eqn(1) approaching as much as possible the optimal behavior of the Newton system in Eqn(3). The second focuses on "explicit" coupling techniques where convergence to the coupled solution is guaranteed by a careful adjustment of the predicted variable and of the data transferred between the domains.

The first approach has its easiest expression in the so called "Block-Jacobi" or "Block-Seidel" iterations which consist in an iterative solution involving the following steps

- $f_S(x^{k+1}, y^k) = 0 \rightarrow y^{k+1}$
- $f_F(x_\omega^{k+1}, y^{k+1}) = 0 \rightarrow x^{k+1}$

The above scheme can be eventually modified to include a relaxation step such as $x_w^{k+1} = \omega x^{k+1} + (1-\omega)x^k$. This technique is very effective for some problems, nevertheless in some conditions it may not be computationally competitive with the "explicit" technique. The choice of the "optimal" relaxation factor is still matter of research. Particularly noteworthy is the work in [7],[19] which advocates the use of the Aitken accelerator to improve the convergence of the

scheme. Alternatively a number of quasi-Newton techniques were presented over the years in an effort of achieving the quadratic convergence behavior of Eqn(3) without requiring the explicit calculation of the Jacobian matrix. An interesting description can be found in [13],[12] where the use of iterative solvers (for which exclusively the matrix-vector multiplication is needed) is exploited for the solution of the coupled problem without building explicitely the Jacobian matrix.

This matrix-free approach reveals to be optimal from the point of view of software modularity, and naturally tends to the solution of the monolythic formulation of which it inherits the stability properties. The typical disadvantage is the high computational cost due to the need of solving the implicit system to a very high degree of accuracy.

The "explicit" approach on the other hand relies on a careful design of the prediction and correction phases in order to minimize the spurious energy introduced in the system. Pioneering work on the subject was done by Farhat and Piperno who developed explicit solution strategies for aeronautical applications [23] [24]. Their work focused mainly on large scale computations with emphasis on parallel solution issues.

Given the absence of a common mathematical formulation for the two coupled fields the investigation of the stability properties of the different methods is in this case extremely challenging. A 1D case is investigated in [23] (for a compressible flow case) while in [24] a simple test is advocated to assess the tendency of a given method to instability. Even if the proposed method has no general mathematical validity it was shown to discriminate successfully stable methods from unstable techniques. The techniques proposed were tested on large aeronautical examples (see [5]) and provided satisfactory results in describing experimental flutter envelopes on real aircrafts. They are believed to be extremely effective in dealing with linear and non-linear problems as long as the displacements involved are small. Some concern rises however regarding their robustness in dealing with the interaction problem between highly flexible structures and truly incompressible fluids (recents results seem however to prove their applicability to general cases when using compressible flow solvers).

The application of space-time FE based approaches can be found in [10] and [4] with reference to both civil and aeronautical applications. Finally some examples of applications to large scale cases of flexible civil engineering structures can be found in [1],[9].

The objective of this work is to discuss a coupling procedure by considering both an "explicit" approach to the couping and a possible improvement by iteration. The effort is to provide a framework to study the properties of the coupling procedure proposed by evaluating the impact of different choices in the structural time integration scheme. The layout of the paper is the following: first a simple but representative 1D model problem is chosen. A coupling strategy is identified and applied "symbolically" to the model problem, obtaining the definition of an "error matrix" which depends on the choice of the time integration scheme. The possibility of iterative improvement of the method is then discussed and an estimate of the evolution of error with the iterations is

given. Finally the explicit technique proposed is applied to the solution of a real problem in bridge aerodynamics, proving its efficiency and its ability to reproduce satisfactorely wind tunnel results.

3 The Model Problem

Staggered or "explicit" coupling techniques are very appealing for the simulation of coupled FSI problems as they allow the use of state-of-the-art single-field solvers greatly easing the development and maintenance of coupling codes. As such they are widely used in the literature, and represent an important tool in many fields of application. The disadvantage is often connected to a lack of robustness in dealing with some cathegory of problems, typically involving truly incompressible fluids or requiring long-term analyses. Strong coupling techniques on the other hand "guarantee" a convergence to the real, monolithic solution by providing a completely "implicit" coupling between the different fields. In the practice however, they tend to be costly and may experience convergence difficulties in dealing with complex problems.

The aim of the paper is to investigate the reasons behind this difficulties. The same effort was attempted in different works, see for example [24] for a discussion of explicit methods or [18] for an analysis of a semi-implicit technique. Both works present interesting results and allow the respective authors to define satisfactory algorithms for application to the fields of interest.

The present article follows a slightly different path by considering a further requirement: an ideal "coupling framework" should be robust upon changes in the single-field solvers, in other words its properties should be independent from the particular solver chosen for each of the fields involved. As an immediate consequence, the particular features of the solvers used should not be exploited in discussing a coupling algorithm, unless the interest focuses exclusively on a very particular choice (as is the case for the articles mentioned). In order to highlight our approach, a simple linearized model problem is chosen and the coupling algorithm is applied to its solution.

The choice for such a simple model problem is unfortunately non univocal in FSI, and needs to be justified. To do so we considered (see for example [22]) that in the "common" engineering practice the fluid solution is often avoided and the interaction between a structure and the surrounding fluid is often described by the exchange of a pressure force between the fluid and the structure. Such pressure force is often modeled by expressing the fluid pressure "p" as

$$p = A(\omega)\dot{x} + B(\omega)x \tag{4}$$

where the coefficients "A" and "B" generally depend on the frequency ω of the motion. It is well known that a careful choice of such aeroelastic constants (generally a good fit to wind tunnel data) can lead to an excellent description of the most relevant features of the fluid action on the structure. This, in other words, implies that this simple model reproduces correctly the coupling between the structure and the fluid.

A load with the form of Eqn(4), which is naturally suited for the frequency domain, needs to be transformed for use in the time domain. Such transformation is not trivial (see for example [17]) and will not be discussed in this article. The interesting result is that Eqn(4) can be substituted in the time domain by a relation of the type

$$p = K_{fsi}x + C_{fsi}\dot{x} + M_{fsi}\ddot{x} \tag{5}$$

eventually generalized including history dependent terms. This implies that the instantaneous force exerted by the fluid on the structure due to the interaction, depends directly on the acceleration, velocity and displacement of the structure itself. The dependence on the acceleration in particular is crucial and the corresponding "mass-like" term is known in the literature as "added mass".

Interestingly enough a similar dependence of the interaction forces on the structural motion cand be obtained (at least concerning the dependence on the structural acceleration) from the analysis of the continuum problem [18] or from a simple algebraic manipulation of the discrete Navier Stokes problem. The important point is in any case that a simple model as the one described in Eqn(5) captures satisfactorily the important features of the coupling, and is thus appropriate to define a "test" problem.

The idea we will exploit in the following is now conceptually simple: let us consider a 1D problem representing the interaction between a given structure and the surrounding fluid. Without loss of generality we can assume that the abstract problem defined by Eqn(1) can be particularized to:

$$f_S(x,p) \rightarrow M\ddot{x} + C\dot{x} + Kx = p \tag{6}$$

$$f_F(x,p) \rightarrow p = K_{fsi}x + C_{fsi}\dot{x} + M_{fsi}\ddot{x} \tag{7}$$

Where "p" is a force that originates from the coupling (and that plays the role of the "coupling variable" in the study of the interaction). We can now consider that most explicit or iterative "coupling procedures" represent techniques for solving the set of equations 1 (defined in implicit form) without taking advantage on the knowledge of the mathematical structure of the different equations involved. The performance of a given coupling algorithm can be therefore assessed by applying it directly to the problem in Eqn(6) of which we know everything.

It is now intuitive that procedures that "behave well" on a test system in the form of Eqn(6) for any physically relevant choice of the parameters K_{fsi} , C_{fsi} and M_{fsi} are idoneous to describe correctly the effect of the fluid on the structure. In other words approximate procedures that solve accurately the test system can be expected to be well suited for the simulation of the FSI interaction for all cases where a model in the form $p = A(\omega)\dot{x} + B(\omega)x$ is acceptable for design purposes.

Before proceeding to the actual analysis it is convenient to highlight the final goal. The crucial point is that the time discretization of Eqn(6) implicitly contains an error connected to the particular choice for the time integration algorithm. This error has no interest for us as it does not depend on the way we enforce the coupling. Our interest is rather connected to the error between the

approximate and exact discrete solution of the coupling problem. In other words we assume that no error cancellation manifests due to the interaction of the different solving technologies. It should be noted that this is a strong assumption. Different approaches, in particular the works of Farhat and Piperno [24] , exploit exactly the error cancellation to tune the features of their coupled solvers. Not considering the error cancellation is however mandatory if the requirement of "independence from the single-field solvers" needs to be enforced.

As a final comment we point out that necessary features for a "good" coupled time integrators are (see Felippa [11])

1. preserve the stability of stable mathematical models
2. manifest the instability of unstable mathematical models.

Some effort will be therefore devoted to describing the damping properties of the different coupling algorithms analysed.

3.1 A simple form for the analysis of the discrete coupling problem

The system described by Eqn(6) can be rewritten as a modified "monolithic" problem by simply substituting the second equation into the first. To simplify the subsequent developments we will express the aeroelastic mass and stiffness (K_{fsi} and M_{fsi}) as a ratio to the correspondent structural quantities. This gives

$$K = \omega^2 \ ; \ K_{fsi} = \beta K; \tag{8}$$

$$C = 2\xi\omega \ ; \ C_{fsi} = 2\xi_{fsi}\omega \tag{9}$$

$$M = 1 \ ; \ M_{fsi} = \alpha M \tag{10}$$

where the mass is taken to be the unit without loss of generality.

Substituting the value of p in Eqn(7) in Eqn(6) and using Eqns(8) yelds after collecting the similar terms the "monolithic" system

$$(1-\alpha)\,\ddot{x} + 2\omega\,(\xi - \xi_{fsi})\,\dot{x} + (1-\beta)\,\omega^2 x = f \tag{11}$$

which is equivalent to Eqn(6) and Eqn(7) and expresses synthetically the dynamic behaviour of the coupled problem. We are mostly interested in the case where the presence of the fluid modifies sensibly the time behaviour of the structure. We shall however assume that even in presence of a very important coupling the mathematical behaviour of the coupled system is preserved, or, in other words that the overall mass and stiffness are both non negative. Note however that a negative aeroelastic damping C_{fsi} (leading to a divergent structural behaviour) can be considered and indeed it constitutes a very important case. These considerations allows us to assume that

$$\alpha < 1 \tag{12}$$

$$\beta < 1 \tag{13}$$

Note that in general we can expect the coupled system to be heavier than the single system which suggests that the stricter condition $\alpha < 0$ will hold.

Up to this point no approximation was introduced and Eqn(11) is still completely equivalent to the system described in Eqn(6) and Eqn(7). To proceed in our analysis it is now necessary to discretize in time the equation of interest. Without loss of generality we shall consider that a very general class of time integrators can be expressed symbolically as

$$\mathbf{y}_{n+1} = \mathbf{A}\mathbf{y}_n + \mathbf{L}_n p_n + \mathbf{L}_{n+1} p_{n+1} \;\; ; \;\; \mathbf{y}_n = \begin{pmatrix} x_n \\ \dot{x}_n \end{pmatrix} \tag{14}$$

where the choice of the linear operators $\mathbf{A}$ and $\mathbf{L}_n$, $\mathbf{L}_{n+1}$ defines the time integrator used and the effect of external forces (note that the "exact" time integrator can be expressed this way). Such form allows us to predict the evolution of a system between the time stations "n" and "n+1" when subjected to a force varying linearly between the values p_n and p_{n+1}. This defines a general framework which simplifies studying the behavior of different time integration schemes.

By defining the auxiliary vectors

$$\mathbf{G} = \begin{pmatrix} K & C \end{pmatrix} \qquad \mathbf{G}_{fsi} = \begin{pmatrix} K_{fsi} & C_{fsi} \end{pmatrix} \tag{15}$$

it is possible to express the model problem of Eqn(11) in the simple matrix form

$$\ddot{x} + \mathbf{G}\mathbf{y} = \mathbf{G}_{fsi}\mathbf{y} + M_{fsi}\ddot{x} \tag{16}$$

and the "pressure force" (the term exchanged between the first and second equation in Eqns(6) and (7)) at the initial and final time stations as

$$p_{n+1} = M_{fsi}\ddot{x}_{n+1} + \mathbf{G}_{fsi}\mathbf{y}_{n+1} \qquad p_n = M_{fsi}\ddot{x}_n + \mathbf{G}_{fsi}\mathbf{y}_n \tag{17}$$

This will suffice to express in matrix form both the exact solution and the approximate one.

3.2 Discrete solution of the exact coupling problem

The evaluation of the exact coupled solution can be performed preserving the matrix notation introduced in the previous section. To do so it is necessary to express the link between the pressure and the structural motion. Dynamic equilibrium (Eqn(6)) gives

$$\ddot{x}_{n+1} = p_{n+1} - \mathbf{G}\mathbf{y}_{n+1} \tag{18}$$

substituting back into Eqn(17) and using the definitions in Eqn(8), we recover the relation

$$(1-\alpha)\, p_{n+1} = (\mathbf{G}_{fsi} - \alpha\mathbf{G})\, \mathbf{y}_{n+1} \tag{19}$$

On the other hand the structure is advanced in time following the time integration rule in Eqn(14). Substituting Eqn(19) into Eqn(14) and collecting terms we obtain

$$\begin{pmatrix} \mathbf{I} & -\mathbf{L_{n+1}} \\ \alpha\mathbf{G} - \mathbf{G}_{fsi} & 1-\alpha \end{pmatrix} \begin{pmatrix} \mathbf{y}_{n+1} \\ p_{n+1} \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{L}_n \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{y}_n \\ p_n \end{pmatrix} \tag{20}$$

which allows to write synthetically

$$\mathbf{z}_{n+1} = \mathbf{A}_{ex}\mathbf{z}_n \tag{21}$$

where

$$\mathbf{A}_{ex} := \begin{pmatrix} \mathbf{I} & -\mathbf{L}_{n+1} \\ \alpha\mathbf{G} - \mathbf{G}_{fsi} & 1-\alpha \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{A} & \mathbf{L}_n \\ 0 & 0 \end{pmatrix} \tag{22}$$

and

$$\mathbf{z}_{n+1} := \begin{pmatrix} \mathbf{y}_{n+1} \\ p_{n+1} \end{pmatrix} \tag{23}$$

Eqn(21) yelds the amplification form of the "exact solution" for the coupled problem.

3.3 Approximate solution of the coupled problem

In the previous section we expressed the exact solution in an amplification form, relating pressures, velocities and displacements at two consecutive time steps.

Unfortunately in real life not all the information needed is available at the same time (otherwise indeed the solution of the problem would be known). Therefore an approximate coupling strategy needs to be devised. In this paragraph we propose and analyse a "fractional step-like" approach to the solution in the form

- predict the fluid forces acting on the structure
- advance in time the structure subjected to the predicted forces
- solve for the fluid domain according to the structural prediction
- correct the structural position according to the newly calculated fluid forces

This simple strategy can be of course improved by successive corrections leading to the definition of an iterative strategy which minimizes the coupling error inside each time step. For such a strategy our interest focuses on the convergence properties of the iteration sequence rather than on the stability of the strategy which will be guaranteed if a a converged solution is achieved inside each time step.

To ease the development it is useful to distinguish between a first prediction phase and successive corrections.

Prediction phase In the prediction step the pressure is calculated as

$$p^I_{n+1} = \eta p_n + \phi p_{n-1} \tag{24}$$

The structural prediction is performed by advancing in time the solution of the structure under the predicted pressure. Taking into account the form assumed for the structural integrator we obtain

$$\mathbf{y}^I_{n+1} = \mathbf{A}\mathbf{y}_n + \mathbf{L}_n p_n + \mathbf{L}_{n+1} p^I_{n+1} \tag{25}$$

grouping the two equations we get

$$\begin{pmatrix} \mathbf{I} & -\mathbf{L_{n+1}} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{y}^I_{n+1} \\ p^I_{n+1} \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{L}_n \\ 0 & \eta \end{pmatrix} \begin{pmatrix} \mathbf{y}_n \\ p_n \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & \phi \end{pmatrix} \begin{pmatrix} \mathbf{y}_{n-1} \\ p_{n-1} \end{pmatrix} \tag{26}$$

By introducing the auxiliary matrices

$$\mathbf{A}^I_1 := \mathbf{C} \begin{pmatrix} \mathbf{A} & \mathbf{L}_n \\ 0 & \eta \end{pmatrix} \; \mathbf{A}^I_2 := \mathbf{C} \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ 0 & \phi \end{pmatrix} \tag{27}$$

with

$$\mathbf{C} := \begin{pmatrix} \mathbf{I} & -\mathbf{L_{n+1}} \\ 0 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{I} & \mathbf{L}_{n+1} \\ 0 & 1 \end{pmatrix} \tag{28}$$

we can express the prediction as

$$\mathbf{z}^I_{n+1} = \mathbf{A}^I_1 \mathbf{z}_n + \mathbf{A}^I_2 \mathbf{z}_{n-1} \tag{29}$$

Correction phase The first step of the correction phase is the computation of the corrected pressure field. To include this in our simplified model we need to calculate the predicted accelerations. Dynamic equilibrium (see Eqn(6)) gives

$$\ddot{\mathbf{x}}^I_{n+1} = p^I_{n+1} - \mathbf{G}\mathbf{y}^I_{n+1} \tag{30}$$

Using the definitions in Eqn(8) the corrected value for the pressure can be expressed as

$$\mathbf{p}^{II}_{n+1} = \mathbf{M}_{fsi}\ddot{x}^I_{n+1} + \mathbf{G}_{fsi}\mathbf{y}^I_{n+1} = \alpha p^I_{n+1} + (\mathbf{G}_{fsi} - \alpha\mathbf{G})\,\mathbf{y}^I_{n+1} \tag{31}$$

while the successive correction step for the structure takes the usual form

$$\mathbf{y}^{II}_{n+1} = \mathbf{A}\mathbf{y}_n + \mathbf{L}_n p_n + \mathbf{L}_{n+1} p^{II}_{n+1} \tag{32}$$

This can be rewritten in an amplification form as

$$\mathbf{z}^{II}_{n+1} = \mathbf{B}_0 \mathbf{z}^I_{n+1} + \mathbf{B}_1 \mathbf{z}_n + \mathbf{B}_2 \mathbf{z}_{n-1} \tag{33}$$

where

$$\mathbf{B}_0 := \mathbf{C} \begin{pmatrix} 0 & 0 \\ \mathbf{G}_{fsi} - \alpha\mathbf{G} & \alpha \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{L}_{n+1} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ \mathbf{G}_{fsi} - \alpha\mathbf{G} & \alpha \end{pmatrix} \tag{34}$$

$$\mathbf{B}_1 := \mathbf{C}\begin{pmatrix}\mathbf{A} & \mathbf{L}_n\\ 0 & 0\end{pmatrix} = \begin{pmatrix}\mathbf{I} & \mathbf{L}_{n+1}\\ 0 & 1\end{pmatrix}\begin{pmatrix}\mathbf{A} & \mathbf{L}_n\\ 0 & 0\end{pmatrix} \tag{35}$$

$$\mathbf{B}_2 := \mathbf{C}\begin{pmatrix}\mathbf{0} & \mathbf{0}\\ 0 & 0\end{pmatrix} = \begin{pmatrix}\mathbf{0} & \mathbf{0}\\ 0 & 0\end{pmatrix} \tag{36}$$

The matrix relative to the second backward step is empty and was included only for generalization purposes. Substituting Eqn(29) into Eqn(33) we obtain

$$\mathbf{z}_{n+1}^{II} = \mathbf{B}_0\left(\mathbf{A}_1^I\mathbf{z}_n + \mathbf{A}_2^I\mathbf{z}_{n-1}\right) + \mathbf{B}_1\mathbf{z}_n + \mathbf{B}_2\mathbf{z}_{n-1} \tag{37}$$

which allows us to obtain the final amplification form for the fractional step procedure as

$$\mathbf{z}_{n+1}^{II} = \left(\mathbf{B}_0\mathbf{A}_1^I + \mathbf{B}_1\right)\mathbf{z}_n + \left(\mathbf{B}_0\mathbf{A}_2^I + \mathbf{B}_2\right)\mathbf{z}_{n-1} \tag{38}$$

This form will be used in the following to address the stability and accuracy of the proposed formulation

3.4 Analysis of the truncation error

Assuming that the exact solution is known in the steps 1 to n, the truncation error can be obtained by difference between the approximate and exact solutions.

The amplification form obtained in the previous section (Eqn(38)) allows us the evaluation of this error. To perform this operation we observe that $\mathbf{y}_n$ and $\mathbf{p}_n$ can not be prescribed independently as the exact solution is used as a starting point. Typically only the displacements and velocities are prescribed and the pressure is calculated subsequently. A dynamic equilibrium relation necessarily holds relating pressures displacements and velocities in the initial steps. The definition of $\mathbf{z}_n$ implies

$$\mathbf{z}_n = \mathbf{D}\mathbf{y}_n \tag{39}$$

with

$$\mathbf{D} := \begin{pmatrix}\mathbf{I}\\ \frac{1}{1-\alpha}\left(\mathbf{G}_{fsi} - \alpha\mathbf{G}\right)\end{pmatrix}$$

The definition of the exact amplification matrix on the other hand allows us to write

$$\mathbf{z}_n^{ex} = \mathbf{A}_{ex}\mathbf{z}_{n-1}^{ex} \tag{40}$$

This together with Eqn(39) allows us to express the exact solution at step "n+1" in the form

$$\mathbf{z}_{n+1}^{ex} = \mathbf{A}_{ex}\mathbf{A}_{ex}\mathbf{D}\mathbf{y}_{n-1} \tag{41}$$

Following the same rationale, the approximate solution at the given step becomes

$$\mathbf{z}_{n+1}^{app} = \left(\left(\mathbf{B}_0\mathbf{A}_1^I + \mathbf{B}_1\right)\mathbf{A}_{ex} + \left(\mathbf{B}_0\mathbf{A}_2^I + \mathbf{B}_2\right)\right)\mathbf{D}\mathbf{y}_{n-1} \tag{42}$$

By subtracting Eqn(41) and Eqn(42) it is possible to define a "reduced" rectangular error matrix relating pressure and structural variables at the step n+1 with the structural variables at the former step. In symbols

$$\mathbf{z}_{n+1}^{ex} - \mathbf{z}_{n+1}^{app} = \mathbf{E}^{red}\mathbf{y}_{n-1} \tag{43}$$

with

$$\mathbf{E}^{red} := \left(\mathbf{A}_{ex}\mathbf{A}_{ex} - \left(\mathbf{B}_0\mathbf{A}_1^I + \mathbf{B}_1\right)\mathbf{A}_{ex} + \left(\mathbf{B}_0\mathbf{A}_2^I + \mathbf{B}_2\right)\right)\mathbf{D}\mathbf{y}_{n-1} \tag{44}$$

This last equation contains all the information needed concerning the order of accuracy. The fact that $\mathbf{E}^{red}$ is not square indicates that for the exact solution only the structural variables can be prescribed, as the pressure can be calculated as a dependent variable.

3.5 Newmark and exact time integration

The results up to this point are still general and hold for any time integration scheme with the form assumed (Eqn(14)). This allowed us to describe "symbolically" an approximate fractional-step coupling procedure and to evaluate an expression for its error matrix. To proceed further in our analysis it is necessary to choose a time integration scheme and to evaluate how the symbolic relations identified will behave depending on the choice. In the following we will consider the "exact" time integrator and the Newmark time integrator and evaluate for the two cases the behaviour of the error.

We highlight once again that the "exact" integrator should guarantee exact results once provided the initial conditions and the values of the force at the beginning and end of the step. Applying the *exact* time integrator inside an *approximate* procedure for computing the solution of the system in Eqn(6) does NOT guarantee an exact result. The interesting result is that the coupling error obtained with both Newmark and the Exact integrator is very similar and can be therefore considered to depend on the coupling procedure rather than on the time integration method.

The amplification matrix and the force terms for the Newmark time integration scheme are widely known and can be found for example in [14]. In the most general case they take the form

$$\mathbf{A} := \mathbf{A}_1^{-1}\mathbf{A}_2 \tag{45}$$

with

$$\mathbf{A}_1 := \begin{pmatrix} 1 + h^2\beta_n K & h^2\beta_n C \\ h\gamma_n K & 1 + h\gamma_n C \end{pmatrix}$$

$$\mathbf{A}_2 := \begin{pmatrix} 1 - \frac{h^2}{2}(1-2\beta_n)K & h(1 - h(1-2\beta_n)\frac{C}{2}) \\ -h(1-\gamma_n)K & 1 - h(1-\gamma_n)C \end{pmatrix}$$

and

$$\mathbf{L}_n := \mathbf{A}_1^{-1}\begin{pmatrix} \frac{h^2}{2}(1-2\beta_n) \\ h(1-\gamma_n) \end{pmatrix}$$

$$\mathbf{L}_{n+1} := \mathbf{A}_1^{-1} \begin{pmatrix} \frac{h^2}{2}(2\beta_n) \\ h\gamma_n \end{pmatrix}$$

where h represents the time step size. We will consider in the following exclusively the second order accurate, non dissipative case.

By assuming that the forces vary linearly within the time step, it is possible to work out a similar result corresponding to the exact time integrator (which can be written in the form of Eqn(14)). In this case the amplification matrix becomes.

$$\mathbf{A} := \begin{pmatrix} cos(\omega h) & \frac{sin(\omega h)}{\omega} \\ -sin(\omega h)\omega & cos(\omega h) \end{pmatrix}$$

and the force terms have the form

$$\mathbf{L}_{n+1} := \begin{pmatrix} \frac{\omega h - sin(\omega h)}{\omega^3 h} \\ -\frac{-1+cos(\omega h)}{\omega^2 h} \end{pmatrix}$$

$$\mathbf{L}_{n} := \begin{pmatrix} \frac{-(-sin(\omega h)+cos(\omega h)\omega h)}{\omega^3 h} \\ \frac{(sin(\omega h)\omega h-1+cos(\omega h))}{(\omega^2 h)} \end{pmatrix}$$

Substituting into Eqn(44) is straight-forward but implies some heavy algebraic calculations. Performing this operation in Maple or a similar Computer Algebra System (CAS) program presents no particular difficulty even for the general case. It is however interesting to focus on the undamped case (no structural damping) as this leads to simpler results. It can be verified that the estimates obtained hold as well for the general damped case

An important (although not unexpected) result is that consistency requires the condition $1 = \eta + \phi$ for the "free" parameters in the prediction step (Eqn(24)). The two choices $\eta = 1, \phi = 0$ and $\eta = 2, \phi = -1$ are appealing and correspond to first or second accuracy for the predicted force. The order of this initial prediction determines the order of the overall approximate time integration scheme for the coupled problem.

Table 1 contains the first non-zero terms for the Taylor expansion of the error matrix. The analysis of these results shows how the same error estimates hold for the Newmark and exact time integration schemes. This feature is desiderable as it suggests that different choices for the time integration scheme can be taken without affecting the coupling error.

The properties of the coupled fractional step depend however on the values assumed by the aeroelastic mass and the damping term. For values of $\alpha > 1$ or $\beta > 1$ the mathematical structure of the problem changes as the overall coupled problem assumes a negative mass matrix. By assuming $\alpha < 1$ or $\beta < 1$ (see remark 12 and 13) we obtain the inequalities

$$\frac{\alpha^2}{\alpha - 1} < 0$$

$$\frac{\beta - 1}{\alpha - 1} > 0$$

Table 1: Error matrix for the fractional step coupling procedure as a function of the time integrator, the time step size "h" and the prediction order

FIRST ORDER PRESSURE PREDICTION $\eta = 1\ \phi = 0$
Exact time integration

$$E_1^{red} := \frac{\alpha^2}{\alpha-1}\frac{(\alpha-\beta)}{\alpha}\begin{pmatrix} \frac{1}{4}\frac{\beta-1}{\alpha-1}\omega^4h^4 & -\frac{1}{6}\omega^2h^3 \\ \frac{3}{4}\frac{\beta-1}{\alpha-1}\omega^4h^3 & -\frac{1}{2}\omega^2h^2 \\ \frac{3}{2}\frac{\beta-1}{\alpha-1}\omega^4h^2 & -\omega^2h \end{pmatrix}$$

Newmark time integration

$$E_1^{red} := \frac{\alpha^2}{\alpha-1}\frac{(\alpha-\beta)}{\alpha}\begin{pmatrix} \frac{3}{8}\frac{\beta-1}{\alpha-1}\omega^4h^4 & -\frac{1}{4}\omega^2h^3 \\ \frac{3}{4}\frac{\beta-1}{\alpha-1}\omega^4h^3 & -\frac{1}{2}\omega^2h^2 \\ \frac{3}{2}\frac{\beta-1}{\alpha-1}\omega^4h^2 & -\omega^2h \end{pmatrix}$$

SECOND ORDER PRESSURE PREDICTION $\eta = 2\ \phi = -1$
Exact time integration

$$E_{ord=2,it=0}^{redExact} := \frac{\alpha^2}{\alpha-1}\frac{(\alpha-\beta)}{\alpha}\frac{\beta-1}{\alpha-1}\begin{pmatrix} \frac{1}{6}\omega^4h^4 & -\frac{1}{6}\omega^4h^5 \\ \frac{1}{2}\omega^4h^3 & -\frac{1}{2}\omega^4h^4 \\ \omega^4h^2 & -\omega^4h^3 \end{pmatrix}$$

Newmark time integration

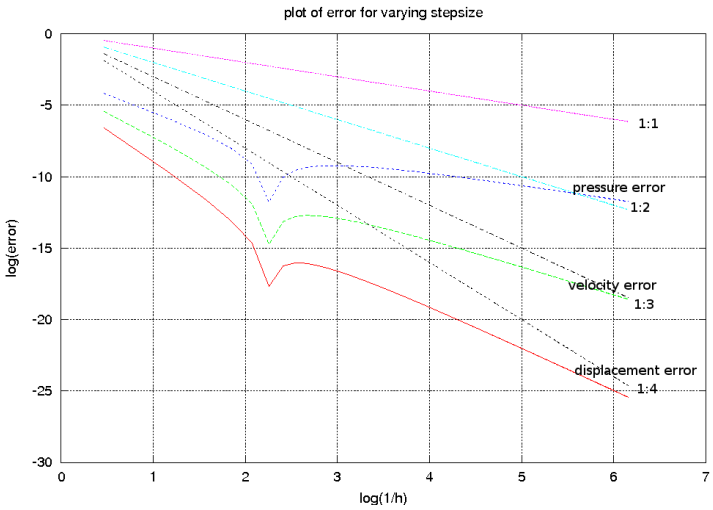
$$E_{ord=2,it=0}^{redExact} := \frac{\alpha^2}{\alpha-1}\frac{(\alpha-\beta)}{\alpha}\frac{\beta-1}{\alpha-1}\begin{pmatrix} \frac{1}{6}\omega^4h^4 & -\frac{1}{6}\omega^4h^5 \\ \frac{1}{2}\omega^4h^3 & -\frac{1}{2}\omega^4h^4 \\ \omega^4h^2 & -\omega^4h^3 \end{pmatrix}$$

which give some insight on the damping properties of the proposed fractional step procedure. The exactness of these results was tested using the Newmark scheme to verify that the theoretical predictions match the numerical values. The numerical approach allows as well the direct assessment of the accuracy order. The Figures (1) and (2) are obtained for the general case (including non zero structural damping) and contain a log-log plot of the error for diminishing time step size. It can be seen how the error on the pressure dominates the error, which is not surprising as the pressure term depends on the acceleration.

It is of interest that these estimates hold in the general case. When the aeroelastic mass is negligible the performance of the fractional step procedure is greatly enhanced and the overall order of accuracy is 3. Using the exact time integrator we obtain the error matrix

$$E := \begin{pmatrix} -\frac{1}{6}\xi\omega^5(-1+\beta)(\beta+4\xi^2)h^5 & -\frac{1}{3}\xi^2\omega^4(2\beta+4\xi^2-1)h^5 \\ -\frac{1}{2}\xi\omega^5(-1+\beta)(\beta+4\xi^2)h^4 & -\xi^2\omega^4(2\beta+4\xi^2-1)h^4 \\ -\xi\omega^5(-1+\beta)(\beta+4\xi^2)h^3 & -2\xi^2\omega^4(2\beta+4\xi^2-1)h^3 \end{pmatrix} \tag{46}$$

Figure 1: First order fractional step. log-log error plot.



3.6 Stability issues

To complete the discussion we make some considerations on the stability of the scheme proposed. Let us consider for simplicity the one-step (first order) fractional step. We already know that the exact solution of the coupled problem (including both the pressure and the structural variables) can be written as

$$\mathbf{z}_{n+1}^{ex} = \mathbf{A}_{ex}\mathbf{z}_n^{ex} \tag{47}$$

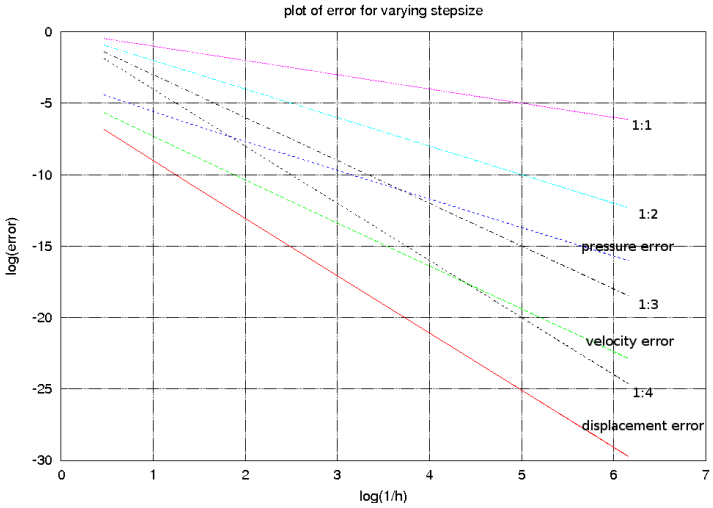
while the approximate solution, obtained via some coupling procedure one can be expressed as

$$\mathbf{z}_{n+1} = \mathbf{A}_{app}\mathbf{z}_n + \mathbf{E}\mathbf{z}_n \;\; ; \;\; \mathbf{E} := \mathbf{A}_{app} - \mathbf{A}_{ex} \tag{48}$$

where we have introduced a new "complete" error matrix $\mathbf{E}$ which relates both the structural variables and pressure. Some confusion might arise concerning the amplification matrices used. $\mathbf{A}_{ex}$, $\mathbf{A}_{app}$ are here order three square matrices and should not be confused with the "reduced" forms (rectangular) used for assessing the accuracy.

A classical discussion of the stability is not trivial due to the complexity of the approximate amplification matrix. We know however that stability is governed by the spectral radius of the amplification matrix. For the one-step fractional step procedure, the eigenvalues can be computed analytically. For the general case the results are unfortunately very lenghty and difficult to handle. Some useful information can however be obtained by expressing the eigenvalues as functions of the auxiliary variable $\Omega := \omega h$ which expresses a more adimensional measure of the number of steps used for the integration of one structural

Figure 2: Second order fractional step. log-log error plot.



period. By expanding in Taylor series the expressions obtained for the eigenvalues , the leading terms of these series (which represents the results for the time step size h tending to zero) give:

$$e_I = \frac{\alpha - 1 + i\Omega\sqrt{1-\alpha}\sqrt{\alpha - 2\beta + 1}}{\alpha - 1} \tag{49}$$

$$e_I = abs\,(\alpha) \tag{50}$$

$$e_{III} = \frac{\alpha - 1 - i\Omega\sqrt{1-\alpha}\sqrt{\alpha - 2\beta + 1}}{\alpha - 1} \tag{51}$$

where the index represents the eigenvalue number. A necessary requirement for stability is that at the limit for $\Omega = \frac{\omega}{h}$ tending to zero, the spectral radius ρ is lower or equal to one. By taking the limit for the simplified expressions we obtain

$$\rho_{max} = max\,(\|\alpha\|, 1) \tag{52}$$

which suggests that the stability is problem dependent and the proposed procedure is **unconditionally unstable** for $\alpha < -1$.

We should remark that this definition of stability is not completely appropriate for the coupling problem of interest. For the FSI coupling both divergent and convergent solutions are possible. This implies that the exact amplification matrix may be characterized by eigenvalues of value greater than one. When, as in the case we are treating here, no damping is considered, the target value for the spectral radius is exactly one. A value greater than 1 (but tending to 1 for diminishing step size) may be accepted, even if it corresponds to a numerically divergent solution. When however the system is characterized by $\alpha < -1$ even for little time steps the solution is divergent and the situation can not be improved by reducing the time step. This leads to explosive instabilities.

3.7 Case of negligible aeroelastic mass

As a final consideration it is interesting to consider the behaviour of the algorithm for α tending to zero. When the aeroelastic mass disappears the equations simplify sensibly and one order of accuracy can be gained for the second order prediction case.

When the aeroelastic damping is zero the equations simplify sensibly and the stability conditions can be investigated analytically (for the first order prediction case). Under these conditions the procedure is unconditionally stable if $\beta > -1$ and conditionally stable otherwise. For $\beta < -1$ it can be verified that the stability condition is $h \leq \frac{2}{\omega\sqrt{-1-\beta}}$.

The general damped case, on the other hand, leads to very long expressions which are not well suited for analytical treatment. The spectral radius can be expressed in this case as $\rho(n_{div}, \xi_{fsi}, \beta)$ where $n_{div} = \frac{2\pi}{\Omega}$ represents the number of time steps used for discretizing the natural period of the structure alone. Useful informations concerning the behaviour of the procedure of interest can be obtained by plotting the spectral radius versus ξ_{fsi}, β for an increasing number of divisions of the structural period.

The analysis of Figure (3.7) shows clearly how the spectral radius is always less than one when the aeroelastic damping constant is negative (which implies a positive damping for the system). It can be also seen how the stability region extends to the positive damping range for very low values of the parameters n_{div}. This suggests that the algorithmic damping is strong when the number of divisions is very low. For "high" values of the number of divisions the isolines of the ρ plot become straight and parallel which testifies how the additional algorithmic damping vanishes.

The plots in Figure 3.7 show a measure of the algorithmic damping versus the numerical one. The scale of the graphs changes as the error vanishes very quickly. However the plots indicate the areas of the ξ_{fsi}, β plane where the algorithmic damping is greater or lesser than that for the newmark scheme applied to the exact case.

4 Iterative Enhancements

The "fractional step" procedue described, represents still an example of "explicit" coupling strategy. An appealing possibility is to attempt improving the properties of the method by iterating through the different domains. Interestingly the analysis framework introduced can be applied directly to such an iterative "improvement" and allows to describe analytically the behaviour of error during the iterations.

This is done in symbols by replacing the predicted value in Eqn(33) with the last known approximation of the system variables, which gives

$$\mathbf{z}_{n+1}^{i+1} = \mathbf{B}_0 \mathbf{z}_{n+1}^{i} + \mathbf{B}_1 \mathbf{z}_n + \mathbf{B}_2 \mathbf{z}_{n-1} \tag{53}$$

where the upper index "i" identifies the iteration count. Note that simply

Table 2: Maple procedure for evaluating the "local" and "total" work difference

```
the first operation needed is to set the ratio between the natural
frequency of the structure and the frequency of the load. This is
needed only because the case of beta = 1(exact resonance) should
be treated as special
> Omega:=beta*omega; beta:=1; phi:=phi;
> pr := (t) -> p * sin(beta*omega*t + phi);
here we compute the "exact" structural solution by solving the
(undamped) single-DOF system described below, starting from a
static configuration
> eq1 := D(D( x ))(t)+omega^2*x(t) = pr(t)/m;
> init1 := x(0) = 0, D(x)(0) = 0;
> solution1:=dsolve({eq1,init1},x(t)):
> expr1 :=   subs(solution1,x(t)):
> x := a->subs(t=a,expr1);  v := a->subs(t=a, diff(expr1,t));
The "guess" solution is calculated by starting from the "true"
solution of the previews time step and advancing in time with a
modified (delayed) load. This is achieved by calculating the
theoretical solution starting from a parametric value for the
initial displacement and velocity
> eq2 := D(D( y ))(t)+omega^2*y(t) = pguess/m;
> init2 := y(0) = xin, D(y)(0) = vin;
> solution2:=dsolve({eq2,init2},y(t)):
> expr2 :=   subs(solution2,y(t)):
> xaux := a -> subs([t=a],expr2);
> vaux := a -> subs([t=a], diff(expr2,t) );
>x1:=(t,h)->subs([xin=x(t-h),vin=v(t-h),pguess=pr(t-h)],xaux(h));
>x2:=(t,h)->subs([xin=x(t),vin=v(t),pguess=pr(t)],xaux(h));
Here we calculate the different "works", on the structural side
(true displacement and pressure) and on the fluid side (true
pressure but "guess" for the velocities). It is important to
calculate consistently the different works
> Wstruct := (t,h)->((x(t+h)-x(t))/h) * int ( pr(t),x=t..t + h);
> Wfluid  := (t,h)->((x2(t,h)-x1(t,h))/h)*int pr(t),x = t..t+h);
> Wdiff   := (t,h)->(Wstruct(t,h) - Wfluid(t,h));
> prova:=simplify(Wdiff(t,h));localdiff:=simplify(subs(t=k*h,prova));
here we evaluate the order of the LOCAL error
> simplify(taylor(localdiff,h,5));
> localexpansion:=factor(convert(%,polynom));
now we evaluate the sum of the errors after n steps, and finally
the order of approximation OF THE TOTAL ERROR
> TOT:=simplify(sum(localdiff,k=0..n));
> TOTexpansion:=simplify(taylor(TOT,h,5));
> TOTapprox := convert(TOTexpansion,polynom):
```

Table 3: Maple procedure for studying the accuracy and stability of the model problem

```
> restart; DATA FOR ANALYSIS
> Mae := alpha;
> Kae := beta*omega^2;
> Cae := 0; #2*omega*xi_ae;
> K := omega^2;  M := 1; C := 0;
DEFINITION OF INTEGRATOR and of auxiliary matrices
> Gae := Matrix([Kae,Cae]); G := Matrix([K,C]);
> > #NEWMARK
>#Ln1 := Matrix([[h^2/(4+2*h*C+h^2*K)], [2*h/(4+2*h*C+h^2*K)]]);
>#Ln :=Ln1;
>#A:=Matrix([[-(-4+h^2*K-2*h*C)/(4+2*h*C+h^2*K),4*h/(4+2*h*C+h^2*K)],
 [-4*h*K/(4+2*h*C+h^2*K),-(h^2*K-4+2*h*C)/(4+2*h*C+h^2*K)]]);
> #EXACT (under the hypothesis of linearly varying pressures)
> A := Matrix([[cos(omega*h), sin(omega*h)/omega],
                          [-sin(omega*h)*omega, cos(omega*h)]]);
> Ln1 := Matrix([[(omega*h-sin(omega*h))/(omega^3*h)],

[-(-1+cos(omega*h))/(omega^2*h)]]);
>Ln:=Matrix([[-(-sin(omega*h)+cos(omega*h)*omega*h)/(omega^3*h)],
>        [(sin(omega*h)*omega*h-1+cos(omega*h))/(omega^2*h)]]);
EXACT SOLUTION
> Aex_rhs := Matrix([[A[1,1],A[1,2],Ln[1,1]],
                             [A[2,1],A[2,2],Ln[2,1]],[0,0,0]]);
> Aex_lhs := Matrix([[1,0,-Ln1[1,1]],[0,1,-Ln1[2,1]],
[Mae*G[1,1]-Gae[1,1],Mae*G[1,2]-Gae[1,2],1-Mae]]);
> Aexact := Aex_lhs^(-1).Aex_rhs:
> Aexact := map(simplify,Aexact);
APPROX SOLUTION
>approx_lhs:=Matrix([[1,0,-Ln1[1,1]],[0,1,-Ln1[2,1]],[0,0,1]]);
first prediction step - different choices are left
> Aapprox_rhs1 := Matrix([[A[1,1],A[1,2],Ln[1,1]],
[A[2,1],A[2,2],Ln[2,1]],[0,0,eta]]);
> Aapprox_rhs2 := Matrix([[0,0,0],[0,0,0],[0,0,phi]]);
> Aapprox_1 :=map(simplify, approx_lhs^(-1).Aapprox_rhs1 );
> Aapprox_2 :=map(simplify, approx_lhs^(-1).Aapprox_rhs2 );
correction step
> temp1 := (Gae-Mae*G)*1;
> temp2 := (Gae-Mae*G)*0;
```

```
Bapprox_rhs0:=Matrix([[0,0,0],[0,0,0],[temp1[1,1],temp1[1,2],alpha]])
> Bapprox_rhs1 := Matrix([[A[1,1],A[1,2],Ln[1,1]],
[A[2,1],A[2,2],Ln[2,1]],[temp2[1,1],temp2[1,2],0]]);
> Bapprox_rhs2 := Matrix([[0,0,0],[0,0,0],[0,0,0]]);
> Bapprox_0 :=map(simplify, approx_lhs^(-1).Bapprox_rhs0);
> Bapprox_1 :=map(simplify, approx_lhs^(-1).Bapprox_rhs1);
> Bapprox_2 :=map(simplify, approx_lhs^(-1).Bapprox_rhs2);
composition of the first steps
>A1expanded:=map(simplify,Bapprox_0.Aapprox_1+Bapprox_1):
>A2expanded:=map(simplify,Bapprox_0.Aapprox_2+Bapprox_2):
other correction step (including fluid solution);
>#A1expanded:=map(simplify,Bapprox_0.A1expanded+Bapprox_1):
>#A2expanded:=map(simplify,Bapprox_0.A2expanded+Bapprox_2):
third correction step
>#A1expanded:=map(simplify,Bapprox_0.A1expanded+Bapprox_1):
>#A2expanded:=map(simplify,Bapprox_0.A2expanded+Bapprox_2):
ACCURACY STUDY matrix linking the exact pressure to the
displacements and velocities
> ttt := (Gae-alpha*G)/(1-alpha);
> Ap := Matrix([[1,0],[0,1],[ttt[1,1],ttt[1,2]]]);
matrix to extract displacements and velocities from the order-3
vector contag the pressures
> Aytransform := Matrix([[1,0,0],[0,1,0]]);
reduced matrices definition (to study accuracy)
> Aex := Aexact.Ap; A1 := A1expanded.Ap: A2 := A2expanded.Ap:
definition of multistep equivalent matrices to study the accuracy
> A1.Aytransform.Aex:  Aapp_eq := A1.Aytransform.Aex + A2:
> Aex_eq := Aex.Aytransform.Aex:
definition and assessment of the "reduced" error matrix
> diff1 := map(simplify,Aapp_eq-Aex_eq):
> eta := 1; #2; phi := 0; #-1;
> map(simplify,diff1):
> differenza := map(simplify,map(series,diff1,h,6));
STUDY OF STABILITY: (only for first  - 1step approximation)
> assume(alpha<1); assume(beta<1); omega:=Omega/h;
> eiga:=LinearAlgebra[Eigenvalues](A1expanded):
> e1 := simplify(convert(series(eiga[1],Omega,5),polynom));
> e2 := simplify(convert(taylor(eiga[2],Omega,5),polynom));
> e3 := simplify(convert(taylor(eiga[3],Omega,5),polynom));
> ro1 := factor(sqrt(e1*conjugate(e1)));
> ro2 := factor(sqrt(e2*conjugate(e2)));
> ro3 := factor(sqrt(e3*conjugate(e3)));
> simplify(eval(ro1,Omega=0));
> simplify(eval(ro2,Omega=0));
> simplify(eval(ro3,Omega=0));
```

using the last known values corresponds to a Jacobi type iteration and may not converge in some situations.

Basing on the theoretical framework proposed above the error matrix corresponding to the (unrelaxed) iterative application of the correction step can be directly assessed. The result is simple and interesting. By identifying as $\mathbf{E}^{red}$ the "reduced" error matrix containing the leading terms in the series expansion for the error of the "basic" fractional step, the following relations hold

$$\mathbf{E}_{i=0}^{red} = \frac{1}{\alpha}\mathbf{E}^{red} \quad \text{(explicit prediction)}$$

$$\mathbf{E}_{i=1}^{red} = \mathbf{E}^{red} \quad \text{(end of basic fractional step procedure)}$$

$$\mathbf{E}_{i=2}^{red} = \alpha\mathbf{E}^{red}$$

$$\mathbf{E}_{i=3}^{red} = \alpha^2\mathbf{E}^{red}$$

This result was verified using a CAS (Computer Algebra System) for the first 4 iterations, by induction we can assume

$$\mathbf{E}^{i} = \alpha^{i-1}\mathbf{E} \tag{54}$$

This result is important as it states that the simple Jacobi iteration will NOT converge when $\alpha < -1$ which corresponds to a case in which the fluid is much heavier than the structure (or more correctly a case for which the added mass effect is crucial). Under these conditions an accelerator will be needed to ensure convergence. This behavior is well known in many practical FSI applications.

It is also important to observe that for values of $\alpha \approx -1$, even on the "safe side", the simple iterative strategy will converge very slowly. Fast convergence will on the other hand be achieved for cases in which the added mass effect is not too important. The conclusion is thus that the basic iterative strategy works well only when "explicit" schemes are effective, which indeed reduces its practical importance.

In practice a relaxation factor ω is generally introduced to guarantee convergence. The determination of the optimal value for this relaxation parameter can be done by trial and error or eventually using some acceleration technique for alternating vector series. Mok and Wall [7],[19] for example advocate the use of the Aitken accelerator for FSI problems (the performance appears to be comparable to the best gradient accelerated techniques). No theoretical justification is however given for this behavior.
Interestingly, the optimal value of the relaxation factor appears to depend on the aeroelastic mass. To show this let us consider first the definition of error

$$\mathbf{y}_{n+1}^{i} - \mathbf{y}_{n+1}^{ex} \approx \alpha^{i-1}\mathbf{E}\mathbf{y}_n \tag{55}$$

by making the difference between successive iterations and pre-multiplying by a given vector $\mathbf{v}^T$ we obtain

$$\mathbf{v}^T\left(\mathbf{y}_{n+1}^{i} - \mathbf{y}_{n+1}^{i-1}\right) \approx \alpha^{i-2}\left(\alpha - 1\right)\mathbf{v}^T\mathbf{E}\mathbf{y}_n \tag{56}$$

and

$$\mathbf{v}^T \left(\mathbf{y}_{n+1}^{i-1} - \mathbf{y}_{n+1}^{i-2} \right) \approx \alpha^{i-3} \left(\alpha - 1 \right) \mathbf{v}^T \mathbf{E} \mathbf{y}_n \tag{57}$$

The ratio between the two above equations gives finally an estimate for the aeroelastic mass coefficient as

$$\alpha = \frac{\mathbf{v}^T \left(\mathbf{y}_{n+1}^{i} - \mathbf{y}_{n+1}^{i-1} \right)}{\mathbf{v}^T \left(\mathbf{y}_{n+1}^{i-1} - \mathbf{y}_{n+1}^{i-2} \right)} \tag{58}$$

This estimate is useful for computing the "optimal" relaxation parameter. By definition the relaxation takes the form

$$\mathbf{y}^* = \omega \mathbf{y}^i + \left(1 - \omega \right) \mathbf{y}^{i-1} \tag{59}$$

Using Eqn(55) and simplifying we obtain

$$\mathbf{y}^* \approx \mathbf{y}^{ex} + \left(\omega + \left(\omega - 1 \right) \alpha \right) \mathbf{E} \mathbf{y}_n \tag{60}$$

which allows us to obtain the optimal relaxation factor as

$$\omega = \frac{\alpha}{1+\alpha} = \frac{\frac{\mathbf{v}^T \left(\mathbf{y}_{n+1}^{i} - \mathbf{y}_{n+1}^{i-1} \right)}{\mathbf{v}^T \left(\mathbf{y}_{n+1}^{i-1} - \mathbf{y}_{n+1}^{i-2} \right)}}{1 + \frac{\mathbf{v}^T \left(\mathbf{y}_{n+1}^{i} - \mathbf{y}_{n+1}^{i-1} \right)}{\mathbf{v}^T \left(\mathbf{y}_{n+1}^{i-1} - \mathbf{y}_{n+1}^{i-2} \right)}} = \frac{\mathbf{v}^T \left(\mathbf{y}_{n+1}^{i} - \mathbf{y}_{n+1}^{i-1} \right)}{\mathbf{v}^T \left(\mathbf{y}_{n+1}^{i-1} - \mathbf{y}_{n+1}^{i-2} \right) + \mathbf{v}^T \left(\mathbf{y}_{n+1}^{i} - \mathbf{y}_{n+1}^{i-1} \right)} \tag{61}$$

or simplifying.

$$\omega = \frac{\mathbf{v}^T \left(\mathbf{y}_{n+1}^{i} - \mathbf{y}_{n+1}^{i-1} \right)}{\mathbf{v}^T \left(\mathbf{y}_{n+1}^{i} - \mathbf{y}_{n+1}^{i-2} \right)} \tag{62}$$

This expression allows us to reduce to zero the leading term of the error (on the SDOF test system) which leads to a higher order of accuracy in time.

It is interesting to highlight how this estimation procedure delivers results which are very similar to the so called Aitken acceleration (see for example Eqn(5)) which justifies the success in its application to FSI.

4.1 On the use of dissipative algorithms for the structural integration

All the results shown up to this moment refer to an exact or non-dissipative (Newmark) time integration scheme. In many problems of practical interest it becomes important the use of a dissipative solver to diminish the noise connected to the high frequencies or simply to guarantee the stability of the time integrator for non linear problems. Unfortunately the assumption in Eqn(14) is not valid for various dissipative schemes which does not allow using this framework (which could be however generalized for the purpose).

We would like however to remark that "surprises" may arise due to a perverse combination of errors between the coupling iterations and the structural solution. This result can be reproduced easily on the test model problem by

choosing a rhs in the form $RHS = \alpha\ddot{x}$ and using a Bossak time integration scheme. Even if this aspect is not studied in detail in this work we believe this is a interesting problem to be addressed in the future.

The results in Fig. 6 shows results of the iterative coupled scheme vs the ones obtained using the same integrator and a "monolithic" approach. It is clear how the iterative solution exibits some spurios damping which is not present when a non-dissipative algorithm is used. This spurious damping appears to be proportional to the aeroelastic mass and disappears as the time steps tends to zero. As a consequence, the importance of this effect is reduced as long as the mass of the structure is sensibly greater than that of the surrounding fluid; however the phenomena may become of major concern in dealing with heavy fluids.

As a final comment we would like to point out that a very simple modification allows to use the fractional step procedure in combination with a dissipative algorithm. The accuracy can be recovered by using a undamped solver for the structural prediction while keeping the numerically damped solver for the solution of the correction step.

5 An improved Iterative Coupling Procedure

As discussed above the simple iterative strategy may encounter convergence difficulties or even diverge depending on the features of the flow. Even relaxed iterative strategies may be challenged for the solution of complex interaction problems when the aeroelastic mass term becomes very important.

From an heuristic point of view this can be justified by considering that most relaxation strategies perform a relaxation on the displacements term which in turn reflects on some form of control over the acceleration dependent contributions. Given the high dependency of the acceleration on small variations of the displacements. This turns out to be ineffective to control the (aeroelastic) mass-induced oscillations.

A natural solution for this problem is to control the variation of the acceleration in order to improve the convergence of the terms that are most problematic in the interaction. The idea is better expressed in mathematical terms. The differential equations governing the dynamic equilibrium, follow a "basic" iterative scheme of the form

$$M\ddot{x}_{n+1}^{i+1} + D\dot{x}_{n+1}^{i+1} + Kx_{n+1}^{i+1} = p_{n+1}^{i} \tag{63}$$

This equation can be modified symbolically to give

$$M\ddot{x}_{n+1}^{i+1} - \alpha M\ddot{x}_{n+1}^{i+1} + D\dot{x}_{n+1}^{i+1} + Kx_{n+1}^{i+1} = p_{n+1}^{i} - \alpha M\ddot{x}_{n+1}^{i} \tag{64}$$

where α^i is a scalar term known from the i-th iteration. Clearly as $\left\|\ddot{x}^{i+1} - \ddot{x}^{i}\right\| \rightarrow 0$ the solution of this modified equation model tends to the original solution. Nevertheless the parameter α has an important impact on how the modified problem converges to the coupled solution.

Once again our simple model problem is of great help in investigating the convergence properties for this modified equation. We already know that the case $p = M_{fsi}\ddot{x}$ is the main responsable of the convergence difficulties for the iterative strategy. We will therefore develop our analysis on the simpler case

$$M\ddot{x}_{n+1}^{i+1} + Kx_{n+1}^{i+1} = \alpha M\ddot{x}_{n+1}^{i} \tag{65}$$

which has the same convergence properties as the more general case. The modified problem becomes

$$(M - \alpha M)\,\ddot{x}_{n+1}^{i+1} + Kx_{n+1}^{i+1} = (\alpha^{ex} - \alpha)\,M\ddot{x}_{n+1}^{i} \tag{66}$$

by dividing by $1 - \alpha^i$ this can be recasted as

$$M\ddot{x}_{n+1}^{i+1} + Kx_{n+1}^{i+1} = \frac{\alpha^{ex} - \alpha}{1 - \alpha} M\ddot{x}_{n+1}^{i} \tag{67}$$

were we modified accordingly the stiffness term (which does not play a role in the stability). The system assumes now the form of Eqn(65), of which it inherits the same properties. The condition for convergence becomes

$$\left| \frac{\alpha^{ex} - \alpha}{1 - \alpha} \right| < 1 \tag{68}$$

As observed before (see Eqn(12)) $\alpha_{ex} < 1$. The parameter α is an approximation of α_{ex} we can therefore take $\alpha < 1$. Under this assumption the unequality can be solved to give

$$\alpha < \frac{1}{2}\alpha^{ex} + \frac{1}{2} \tag{69}$$

Basing on the results shown before we can conclude that for a system in the form of Eqn(67) the error varies with iterations as $\mathbf{E}^i = \left(\frac{\alpha^{ex} - \alpha}{1 - \alpha}\right)^{i-1} \mathbf{E}^0$. It follows that the optimal choice for the modified equation method is given by $\alpha = \alpha^{ex}$, for which the spurious oscillations are immediately damped out.

The presence of the modification factor α affects sensibly the condition for stability, hereby greatly increasing the "area" of convergence. Any choice of the type $\alpha < \alpha^{ex}$ guarantees stability and monothonic convergence ($0 < \frac{\alpha^{ex} - \alpha}{1 - \alpha} < 1$) while the choice $\alpha^{ex} < \alpha < \frac{1}{2}\alpha^{ex} + \frac{1}{2}$ guarantees stability but oscillatory convergence. Unfortunately the exact value for the term "alpha" is not known. However Eqn(69) ensures that the convergence is robust and that a rough estimation of the optimal parameter is sufficient to ensure convergence.

Before proceeding any further let's make some heuristic considerations on the properties of the iterative scheme proposed. The crucial point is that at every iteration we solve a problem that is different from the original one and which solution becomes increasingly similar to that of the original problem as the iterative process converges. Even "at convergence", an error exists which implies that the solution found corresponds to a slightly different problem. The choice of an error tolerance therefore imply how closely the modified equations will "mimic" the real problem. The important point is however to guarantee "a priori" the stability of the coupled procedure while improving the accuracy as needed. Note that this is not the case for a general relaxation scheme.

5.1 Estimating alpha

The real interest of any acceleration scheme is of course in application to real models. For such problems we can observe that the interface force should be oriented as the normal to the interface and proportional to the surface area, while an heuristic consideration tells us that only the acceleration normal to the interface will play a role in the coupling. This suggests that $\mathbf{M}_{fsi}$ should take, on each interface node, a form of the type $\mathbf{M}_{fsi} = \alpha \mathbf{A}_n \mathbf{A}_n^T$ where $\mathbf{A}_n$ represents the normal at one interface node multiplied by the corresponding influence area.

Several possibilities exist for the practical evaluation of the parameter alpha. A good estimation could be derived by physical reasoning or by considering the actual choice of the flow solver.

An alternative procedure, which makes use exclusively of the data at different iterations is derived next for the non-scalar case. Assuming α^i to be constant, the application of Eqn(67) to two successive iterations gives

$$\mathbf{M}\ddot{\mathbf{x}}_{n+1}^{i+1} = \frac{\alpha^{ex} - \alpha}{1 - \alpha^i} \mathbf{M}\ddot{\mathbf{x}}_{n+1}^{i} - \frac{1}{1 - \alpha^i} \mathbf{K}\mathbf{x}_{n+1}^{i+1} \tag{70}$$

$$\mathbf{M}\ddot{\mathbf{x}}_{n+1}^{i} = \frac{\alpha^{ex} - \alpha}{1 - \alpha^i} \mathbf{M}\ddot{\mathbf{x}}_{n+1}^{i-1} - \frac{1}{1 - \alpha^i} \mathbf{K}\mathbf{x}_{n+1}^{i} \tag{71}$$

Subtracting the above equations and pre-multiplying by $\mathbf{M}^{-1}$ gives

$$\ddot{\mathbf{x}}_{n+1}^{i+1} - \ddot{\mathbf{x}}_{n+1}^{i} = \frac{\alpha^{ex} - \alpha}{1 - \alpha^i} \left(\ddot{\mathbf{x}}_{n+1}^{i} - \ddot{\mathbf{x}}_{n+1}^{i} \right) - \frac{1}{1 - \alpha^i} \mathbf{M}^{-1} \mathbf{K} \left(\mathbf{x}_{n+1}^{i+1} - \mathbf{x}_{n+1}^{i} \right) \tag{72}$$

For small time steps the accelerations governs the phenomena (a small variation in displacements in a small time induces a high acceleration to rise). This allows to neglect the term $\mathbf{M}^{-1} K \left(\mathbf{x}_{n+1}^{i+1} - \mathbf{x}_{n+1}^{i} \right)$. Under this assumption, pre-multiplying both sides by $\ddot{\mathbf{x}}_{n+1}^{i} - \ddot{\mathbf{x}}_{n+1}^{i}$ we obtain the scalar relation

$$c = \frac{\alpha^{ex} - \alpha}{1 - \alpha^i} \tag{73}$$

where

$$c := \frac{\left(\ddot{\mathbf{x}}_{n+1}^{i+1} - \ddot{\mathbf{x}}_{n+1}^{i} \right)^T \left(\ddot{\mathbf{x}}_{n+1}^{i+1} - \ddot{\mathbf{x}}_{n+1}^{i} \right)}{\left(\ddot{\mathbf{x}}_{n+1}^{i+1} - \ddot{\mathbf{x}}_{n+1}^{i} \right)^T \left(\ddot{\mathbf{x}}_{n+1}^{i} - \ddot{\mathbf{x}}_{n+1}^{i-1} \right)} \tag{74}$$

Eqn(73) can be solved for α^{ex} and used for for calculating $\alpha^{i+1} := \alpha^{ex}$

We remark that this procedure represents only one of the many possibilities to obtain a value for alpha. In many cases it may be attractive to use aconstant obtained by other sort or reasoning. In any case, once α has been estimated (by any means) the relaxation method proposed can be easily implemented with a minor modification to the structural solution code.

5.2 A simple example to verify the implementation

A very simple example can be used to verify the correctness of the implementation. This is obtained by taking a 1m cube of elastic material subjected to a variable pressure on one side.

The external pressure is made dependent on the acceleration following a law of the type $p = M_{fsi}\ddot{x}_{norm}$. For this numerical setting, the proposed method is expected to recover exactly the aeroelastic matrix and, apart for an initial transient phase, to reproduce the analytical result in just one iteration (two iterations are actually needed for convergence to be detected).

Taking $\rho = 1000\frac{Kg}{m^3}$, $E = 5e9\frac{N}{m^2}$, $\nu = 0$, $t = 1m$ and using a lumped mass matrix the actual system to be solved is equivalent to the scalar problem

$$\frac{1}{2}M\ddot{x} + \frac{EA}{L}x = M_{fsi}\ddot{x} \tag{75}$$

Assuming $M_{fsi} = -1000$, for which the standard staggered approach would not converge, the system takes the following form

$$500\ddot{x} + 5000000x = 1000\ddot{x} \tag{76}$$

Fig. 7 shows a comparison between the exact and analytical results. Convergence is obtained at the second iterations except during the first three steps. A plot of the error norm $\left\|\ddot{x}^{i+1} - \ddot{x}^{i}\right\|$ is given in Fig. 8

6 Restrictor Flap

This test was proposed by Wall and Mok [19] and [7], as a challenging test for the stability of coupled procedures. The geometry is described in Fig. 9. The data used for the analysis are $\rho_F = 956Kg/m^3$, $\mu = 0.145Kg/ms$, $\rho_S = 1500Kg/m^3$, $E = 2.3MPa$, $\nu = 2.3MPa$. The resulting pressure histories are shown in Fig. 10 and directly compared with the results of the literature. The frames shown in Fig. 11 show the evolution of the coupled fluid-structure deformation and compare well with the results of the same example as reported in Mok [19]. A second order fractional step solver is used in solving the problem. The resulting pressure histories are shown in Fig. 10 and directly compared with the results of the literature

7 Flag flutter

An interesting validation example is that of a square bluff body followed by a flexible cantilever plate as shown in Fig. 6. The prescribed distribution of velocities is shown in the same figure. The numerical setup is reproduced following [4]. The fluid characteristics are $\rho_f = 1.18\frac{Kg}{m^3}, \mu = 1.8e-5\frac{Kg}{ms}$ the structure has a elastic modulus of $200000\frac{N}{m^2}$ and a density $\rho_s = 2000\frac{Kg}{m^3}$ with the first three natural frequencies at 0.61,3.8 and 10.6 Hz.

By applying a constant velocity at the inflow $U_{inflow} = 0.315\frac{m}{s}$, the bluff body sheds Von Karman vortices with a frequency of 3.7 Hz. (on the rigid system) close to the second natural frequency of the structure. In the rigid case the pressure oscillations are relatively small, consequently only small displacement motion could be expected due to the vortex shedding.

Given the proximity between the shedding frequency and the structural frequency on the other hand lock-in can manifest leading to large amplitude "resonant" motion. After a first phase of growth, the amplitude of vibration is however expected to reach a "stable" solution with a peak amplitude of around 0.08m.

Loose coupling procedures based on high order prediction steps (see [24]) are known to be unstable for this example. On the other hand, procedures based on a first order prediction step enforce poorly the energy conservation at the interface, leading to an important numerical positive damping (unless the time step is very small). This example was in fact originally proposed as an argument for the necessity of implicit procedures for non-linear aeroelasticity. The results shown were however obtained using the (loosely coupled) fractional step procedure described in this work. The Fourier transform of the displacement history is given in Fig. 6. The leading peak is found at a frequency of 3.04Hz in good agreement with the value of 3.1Hz published in [4]. The displacement history of the tip is reported in Fig. 6 on top of the history reported in the reference article. The excellent agreement found with the literature proves the effectivenes of our scheme even for challenging simulations.

8 A Practical example

To conclude this work we propose an example of application for the fractional step algorithm proposed in the first part of the paper. The goal of this example is twofold: from one side our interest is in showing that the algorithm actually "works", also we want to prove that the coupling error is dominated by the fluid solution phase rather than by the coupling procedure.

To do so we will focus our attention on the calculation of the flutter derivatives for a long span bridge deck (The Great Belt bridge section) subjected to a constant wind speed. First we present briefly the engineering theory, then we review two methods used for the experimental determination of the coefficients needed, and finally we will show that the two techniques lead to similar results, proving that the coupled procedure delivers satisfactory results.

Even if the final design still relies on experimental testing, a well established theory of bridge aerodynamics [20] was developed in the early 70s and is still used in engineering applications. In a modern revision [16] the theory is based on the assumption of a mathematical model in the form

$$L(t) = \rho\, U^2 B \left(\frac{K\, H_1\, \dot{y}}{U} + \frac{K\, H_2\, B\dot{\alpha}}{U} + K^2 H_3\, \alpha + \frac{K^2 H_4\, y}{B} \right) \qquad (77)$$

$$M(t) = \rho U^2 B^2 \left(\frac{K A_1 \dot{y}}{U} + \frac{K A_2 B\dot{\alpha}}{U} + K^2 A_3 \alpha + \frac{K^2 A_4 y}{B} \right) \qquad (78)$$

where $K = \frac{\omega B}{U}$. Eqns(77) and (78) express the forces exerted by the fluid on the structure due to the motion of the latter. The coefficients in these equations are known as "aeroelastic derivatives" and are obtained experimentally. Interestingly enough this model can be shown to be equivalent to the model $A(\omega)\mathbf{v} + B(\omega)\mathbf{x}$ which we assumed for the analysis of our coupling algorithms.

All the coefficients in Eqns(77) and 78 need to be identified experimentally through the analysis of sectional models.

The solution of the bridge aerodynamic problem has been addressed in many CFD works over the past years, nevertheless the most frequent approach in numerical analysis is focused on the direct determination of the flutter limit [21, 25, 8, 15]. This is generally measured through the direct assesment of the system's energy, or possibly by the analysis of the displacement history. Unfortunately many factors come into play in this process making difficult the identification of the error sources. Even if in general these errors are on the numerical side it is interesting that even the experimental results may be affected by factors such as the correspondence of the numerical Reynolds to the real one, or the blockage ratio for the model tested.

In the practice, at least two techniques exist for the experimental determination of the flutter derivatives. One based on the prescription of the section motion and on the analysis of the history of the forces applied from the fluid to the structure. The second one based on the analysis of the displacement history of the section model mounted on springs. Details on the procedure used for the identification of the aeroelastic derivatives in the two cases are prodvided in the Appendix.

Both of those procedures can be reproduced numerically. The forced vibration approach does not involve any coupling error and can be considered as a reference solution for the evaluation of the equivalent coupled approach. Its performance is governed exclusively by the properties of the fluid solver and by the computational mesh used.

In the hypothesis that the same initial solution, mesh and boundary conditions are used, the aeroelastic derivatives computed using a "free vibration" (coupled FSI) approach can be directly compared to the ones obtained from the forced analysis. This comparison allows to isolate the coupling error from the other error sources thus reaching the desired objective.

The advantages of this approach are self evident. A similar result could be achieved by direct comparison with the experimental results once the ability of the fluid solver to describe correctly the phenomena is guaranteed. This would require fine meshes greatly increasing the computational cost, thus making the testing procedure unsuitable for the study of different coupling strategies. On the other hand by comparing two numerical results obtained under the same condition we can directly asses the impact of the coupling on the final results, which is the objective of current work.

Table 4: Description of the validation procedure

- Construct the model and compute a starting solution by keeping the section fixed
- Prescribe the motion of the section and identify the aeroelastic derivatives for different frequences
- Suspend the section on springs calculating the spring so to be in the range of reduced velocities of interest.
- Calculate the aeroelastic derivatives in free motion
- Evaluate the coupling properties by comparing the two curves

8.1 Numerical Setup

The identification techniques described above were used for the identification of the aeroelastic derivatives of a bridge cross-section. Due to the availability of validation data the cross-section of the great belt suspension span in Denmark was chosen for the benchmark exercise. This section, was studied numerically and experimentally [15, 3, 2]. The experimental plots used for the comparison were taken from [6].

The domain used in the simulation, as represented in Fig.(8.1), was chosen so to keep around 6% the ratio between the section chord and the vertical dimension of the domain in order to minimize the blockage effect. An unstructured mesh composed of 9197 nodes and 17862 triangular elements with a minimum element size (around the bridge deck) of 0.15m was used.

Only pressure force are transferred to the structure, negleting the contribution of viscous forces (which would be poorly approximated by a mesh as the one described). This approximation is commonly accepted for bridges which behave as bluff bodies.

The actual dimensions of the cross-section were used with a Reynolds number that was consequently far greater than the one reproduced in wind tunnel testing. Even if this could theoretically lead to important discrepancies with the experimental data, it is currently believed that for the case of interest the derivatives are fairly independent of the Reynolds number of the flow. The inflow velocity was therefore taken as 10m/s corresponding to a $Re = \frac{v_\infty B}{\nu} \approx 2 * 10^7$.

No turbulence model was included in the simulation however a subgrid scale stabilization was used.

The structural displacements were kept small with a maximum pitching angle of 3deg and a maximum motion in heave of 1.57m in accordance to the values reported in [20, 2]. The reduced frequency $\frac{U_{inflow}}{nB}$, was varied by changing the value of the frequency n of the prescribed sectional motion. This allowed to start all the simulations exactly from the same initial conditions. An initial velocity was prescribed to the structure in order to study the decay or growth of the

initial solution. This velocity was chosen so to provide a maximum displacement "in vacuo" corresponding to the one for the prescribed motion setting. This was enforced by considering that in absence of fluid the system is conservative, from which follows

$$\frac{1}{2}Mv_{in}^2 = \frac{1}{2}Kx_{max}^2 \rightarrow v_{in} = \sqrt{\frac{K}{M}}x_{max} \tag{79}$$

This setup was used for the assessment of the explicit coupling procedures implemented in Kratos, a Multi-Physics code developed at CIMNE (www.cimne.com) . Both first and second order solvers were used, the latter being vastly superior in reproducing the experimental curves. For the first order case, showed in Fig. 15 the mesh and step size are not sufficiently fine to reproduce the experimental curves. The results of interest are however clearly met as the identified derivatives follow a similar behavior using both identification techniques, proving that the error is governed by the fluid solution and not by the coupling error. This allows in the practice to use a much coarser mesh than that normally be needed to get rid of the errors in the fluid solution, which in turn provides an important advantage for the quick assesment of the performance of the coupling procedure.

We remark that the use of a second order solver for the fluid solution (on the same computational setup) reproduces accurately the experimental curves using the free vibration procedure (see Fig. 16). This confirms the good performance of the coupling procedure used.

9 Conclusions

A simple model problem is presented and used to predict the behaviour of some coupling algorithm for applications of FSI. In particular a "Fractional Step" type algorithm and a coupling strategy based on the modification of the dynamic equilibrium equations have been analysed. Although simple, the model allows to justify the stability or instability of some coupled strategies and to justify the success of some accelerators proposed in the literature. The effectivness of the algorithms analysed is proved first by comparing their performance with other results from the available literature and later by showing their effectiveness for the solution of the problem of bridge aeroelasticity. This latter case is treated using free and forced vibration approaches so to be able to assess the error induced by the FSI coupling. The results confirm that the coupling error is negligible when compared to the errors in the fluid solution.

10 Acknowledgements

Present work was partially supported under the auspices of the Beatriu de Pinos Program of the Generalitat de Catalunya and by the SEDUREC project of the Consolider Programme of the Ministerio of Educacion y Ciencia of Spain. The authors would like to thank Dr. Lazzari, Prof. Saetta and all the group of Prof. Vitaliani in Padova for the many helpful discussions on the topics analysed.

References

[1] M. Gluck A. Halfmann, E. Rank et al. A partitioned solution approach for the fluid-structure interaction of wind and thin walled structures. Technical report, TU Munchen, Universitat Erlangen-Nurnberg, 2000.

[2] J.H.Walther A.Larsen. Aeroelastic analysis of bridge girder sections based on discrete vortex simulations. *Journal of Wind Engineering and Industrial Aerodynamics*, 67:253–265, 1997.

[3] J.H.Walther A.Larsen. Discrete vortex simulation of flow around five generic bridge deck sections. *Journal of Wind Engineering and Industrial Aerodynamics*, 77:591–602, 1998.

[4] D. Dinkler B. Hubner, E. Walhorn. A monolithic approach to fluid-structure interaction using space-time finite elements. *CMAME*, 2003.

[5] G. Brown C. Farhat, P. Geuzaine. Application of a three-field non-linear fluid-structure formulation to the prediction of the aeroelastic parameters of an f-16 fighter. *Computers and Fluids*, 32:3–29, 2003.

[6] D. Cobo del Arco and A.C. Aparicio Bengoechea. An analysis of wind stability.improvements to the response of suspension bridges. Technical report, CIMNE, 1999.

[7] W.A. Wall D.P. Mok. Partitioned analysis schemes for the transient interaction of incompressible flows and nonlinear flexible structures. In *trends in computational structural mechanics*, Barcelona, 2001.

[8] S.Piperno E. Briand. Validacion du code nsi3fs sur des ecoulements turbulents autour d'un tablier de pont elementaire. Technical report, INRIA, 2002.

[9] A.Halfmann E. Rank, D. Scholz et al. Fluid-structure interaction in civil engineering. In *Second MIT conference on Computational Fluid And Solid Mechanics*, 2003.

[10] B. Hubner E. Walhorn, A. Kolke and D. Dinkler. Fluid-structure coupling within a monolithic model involving free surface flows. *computers and structures*, submitted:2100–2111, 2005.

[11] C. Felippa. Fsi course: chapters 1 - 11. http://www.colorado.edu/engineering/CAS/courses.d/FSI.d/Home.html, 2004.

[12] J. Steindorf H.G. Matthies. Partitioned strong coupling algorithms for fluid-structure interaction. Technical report, Technical University Braunschweig, Brunswick, 2002.

[13] J. Steindorf H.G. Matthies. Strong coupling methods. Technical report, Technical University Braunschweig, Brunswick, 2002.

[14] T.J.R. Hughes. *The Finite Element Method*. Dover, 2000.

[15] J.B.Frandsen. Simultaneous pressures and accelerations measured full-scale on the great belt east suspension bridge. *Journal of Wind Engineering and Industrial Aerodynamics*, 89:95–129, 2001.

[16] A. Larsen. Advances in aeroelastic analyses of suspension and cable stayed bridges. *Journal of Wind Engineering and Industrial Aerodynamics*, 74:73–90, 1998.

[17] M. Lazzari. Time domain modelling of aeroelastic bridge decks: a comparative study and an application. *International Journal For Numerical Methods in Engineering*, 62, 2005.

[18] Celine Grandmont Miguel Angel Fernandez, Jean-Frederic Gerbeau. A projection semi-implicit scheme for the coupling of an elastic structure with an incompressible fluid. Technical report, INRIA, 2005.

[19] D.P. Mok. *Partitionierte Lungsansze in der Strukturdynamik und der Fluid-Struktur-Interaktion*. PhD thesis, Institut fr Baustatik, Fakult Bauingenieur- und Umweltingenieurwissenschaften, 2001.

[20] J.J. Tomko R. Scanlan. Airfoil and bridge deck flutter derivatives. *J. Mechanical Division*, EM6, 1971.

[21] S.Govindaswamy R.P.Selvam. Aeroelastic analysis of bridge girder section using computer modeling. Technical report, University Of Arkansas, 2001.

[22] R.Rossi. *Ligh-weight structures - Numerical Analysis and Coupling Issues*. PhD thesis, University of Padova, Italy, 2005.

[23] B. Larroutorou S.Piperno, C.Farhat. Partitioned procedures for the transient solution of coupled aeroelastic problems - part1 - model problem, theory and two dimensional applications. *Computer Methods in Applied Mechanics and Engineering*, 124:79–112, 1995.

[24] C.Farhat S.Piperno. Partitioned procedures for the transient solution of coupled aeroelastic problems - part2 - energy transfer analysis and three dimensional applications. *Computer Methods in Applied Mechanics and Engineering*, 190:3147–3170, 1995.

[25] E.Bournet. S.Piperno. Numerical simulation of wind effects on flexible civil engineering structures. Technical report, INRIA, 1999.

11 Appendix

11.1 Forced motion procedure

One first approach is based on the application of a prescribed sinusoidal motion to the bridge section which is then subjected to the fluid flow. A description

of the testing procedure can be found in [3] or, with a similar approach in [21]. The "experimental" setup can be expressed as follows:

- fix the horizontal and vertical motion of the cross section
- prescribe a sinusoidal pitching motion (a prescribed rotation) with a known angular frequency
- measure the forces and moments on the section
- perform a (linear) least-square fitting of the measured forces with an equation in the form $x(t) = Asin(\omega t + \phi)$ both for the moment history and lift history.
- use the fitted values for A and ϕ to identify the coefficients as described below

The identification is performed through a (linear) least square fit of the recorded moment and lift histories using an expression of the form $Asin(\omega t+\phi)$. The simple trigonometric identity

$$Asin(\omega t + \phi) = Bsin(\omega t) + Ccos(\omega t)) \ ; \ A = \sqrt{B^2 + C^2}\,, \ \phi = tan^{-1}\left(\frac{C}{B}\right) \tag{80}$$

together with the auxiliary vectors (N is the number of measured time steps, δt the stepsize)

$$\mathbf{v}_s = \begin{pmatrix} sin(\omega\delta t) \\ ... \\ sin(\omega N\delta t) \end{pmatrix} \quad \mathbf{v}_c = \begin{pmatrix} cos(\omega\delta t) \\ ... \\ cos(\omega N\delta t) \end{pmatrix} \tag{81}$$

allows to rewrite the fitting problem as

$$\mathbf{M} = B_m\mathbf{v}_s + C_m\mathbf{v}_c \ ; \tag{82}$$

$$\mathbf{L} = B_l\mathbf{v}_s + C_l\mathbf{v}_c \ ; \tag{83}$$

where the moment and lift histories are written in vector form as

$$\mathbf{M} = \begin{pmatrix} M(\delta t) \\ ... \\ M(N\delta t) \end{pmatrix} \quad \mathbf{L} = \begin{pmatrix} L(\delta t) \\ ... \\ L(N\delta t) \end{pmatrix} \tag{84}$$

by multiplying each of them once by $\mathbf{v}_s^T$ and then by $\mathbf{v}_c^T$ it is possible to obtain the systems

$$\begin{pmatrix} \mathbf{v_s} \bullet \mathbf{v_s} & \mathbf{v_s} \bullet \mathbf{v_c} \\ \mathbf{v_s} \bullet \mathbf{v_c} & \mathbf{v_c} \bullet \mathbf{v_c} \end{pmatrix} \begin{pmatrix} B_m \\ C_m \end{pmatrix} = \begin{pmatrix} \mathbf{v_s} \bullet \mathbf{M} \\ \mathbf{v_c} \bullet \mathbf{M} \end{pmatrix} \tag{85}$$

and

$$\begin{pmatrix} \mathbf{v_s} \bullet \mathbf{v_s} & \mathbf{v_s} \bullet \mathbf{v_c} \\ \mathbf{v_s} \bullet \mathbf{v_c} & \mathbf{v_c} \bullet \mathbf{v_c} \end{pmatrix} \begin{pmatrix} B_l \\ C_l \end{pmatrix} = \begin{pmatrix} \mathbf{v_s} \bullet \mathbf{L} \\ \mathbf{v_c} \bullet \mathbf{L} \end{pmatrix} \tag{86}$$

Table 5: Relations for computing the aeroelastic derivatives: the upper index indicates the prescribed sectional motion, the lower index the equation used in fitting the load history

- Prescribed heave:

$$H_4 = \frac{B_{lift}^{heave}}{\rho\, B^2 w^2 y_0} \;\; ; \;\; H_1 = \frac{C_{lift}^{heave}}{\rho\, B^2 w^2 y_0}$$

$$A_4 = \frac{B_{mom}^{heave}}{\rho\, B^3 w^2 y_0} \;\; ; \;\; A_1 = \frac{C_{mom}^{heave}}{\rho\, B^3 w^2 y_0}$$

- Prescribed rotation:

$$H_3 = \frac{B_{lift}^{pitch}}{\rho\, B^3 w^2 \alpha_0} \;\; ; \;\; H_2 = \frac{C_{lift}^{pitch}}{\rho\, B^3 w^2 \alpha_0}$$

$$A_3 = \frac{B_{mom}^{pitch}}{\rho\, B^4 w^2 \alpha_0} \;\; ; \;\; A_2 = \frac{C_{mom}^{pitch}}{\rho\, B^4 w^2 \alpha_0}$$

which can be solved for the desired fitting coefficients. By considering further that the measured load histories were obtained by prescribing a sinusoidal motion (pitching) of the type

$$\alpha = \alpha_0 sin(\omega t) \tag{87}$$

$$\dot{\alpha} = \alpha_0 \omega sin(\omega t) \tag{88}$$

$$\ddot{\alpha} = -\omega^2 \alpha \tag{89}$$

and substituting this in Eqn(78) we obtain

$$\mathbf{L} = B_l \mathbf{v}_s + C_l \mathbf{v}_c = (\rho B^3 \omega^2 H_3 \alpha_0)\mathbf{v}_s + (\rho B^3 \omega^2 H_2 \alpha_0)\mathbf{v}_c \tag{90}$$

$$\mathbf{M} = B_m \mathbf{v}_s + C_m \mathbf{v}_c = (\rho B^4 \omega^2 A_3 \alpha_0)\mathbf{v}_s + (\rho B^4 \omega^2 A_2 \alpha_0)\mathbf{v}_c \tag{91}$$

which links the fitting coefficients to the aeroelastic derivatives. By proceeding in a totally analogous way prescribing a displacement history in the form

$$y = y_0 sin(\omega t) \tag{92}$$

$$\dot{y} = y_0 \omega sin(\omega t) \tag{93}$$

$$\ddot{y} = -\omega^2 y \tag{94}$$

which represents a heaving motion, it is possible to identify the remaining four derivatives. Table (5) summarizes the relevant formulae for the calculation of the aeroelastic derivatives.

11.2 Free vibration method

The second procedure for the determination of the aeroelastic coefficients is based on the analysis of the displacement history of the sectional model mounted on a set of springs of known stiffness. This testing procedure is more challenging for the interaction as it involves a completely coupled analysis, which introduces a further error source. An excellent example of application to bridges can be found in [25, 8]. In these works however the free vibration procedure is used to assess directly the flutter speed by evaluating the energy variation.

The original identification procedure as proposed by Scanlan [20], is based on a three step analysis in which the cross section mounted on springs is first restrained in heave, then in rotation and finally left completely free. This was needed as the available setup allowed to record exclusively the displacement history but not the restrain forces. The approach described in the following is slightly different and it involves only two sets of simulations by making use of both displacement histories and forces records.

The dynamical equation of motion in rotation for a bridge section subjected to small displacements takes the form

$$I\ddot{\alpha} + C_\alpha\dot{\alpha} + K_\alpha\alpha = M(t) \tag{95}$$

In order to identify the first set of aeroelastic derivatives it is convenient to restrain the motion in heave while leaving "free" the rotation. Under this assumption, substituting Eqn(78) into Eqn(95) gives

$$I\ddot{\alpha} + C_\alpha\dot{\alpha} + K_\alpha\alpha = \rho B^3 w\,(BA_2\dot{\alpha} + wBA_3\alpha) \tag{96}$$

which provides a description of the motion of the section in its interaction with the fluid. Eqn(96)can be formally rewritten as

$$I\ddot{\alpha} + 2I\omega\xi\dot{\alpha} + I\omega^2\alpha = 0 \tag{97}$$

with

$$2I\omega\xi = C_\alpha - \rho\omega B^4 A_2 \tag{98}$$

and

$$I\omega^2 = K_\alpha - \rho\omega^2 B^4 A_3 \tag{99}$$

The differential equations in the form described in Eqn(97) have an analytical solution of the type.

$$\alpha_{teo}(t) = Ae^{-\xi\omega t}sin(\omega t + \phi) \tag{100}$$

Assuming that the mathematical model proposed by Scanlan describes correctly the real structural behaviour, it is possible to fit the "experimental" solution with a mathematical expression given by Eqn(100). A non-linear fitting of the motion history allows therefore computing the coefficients $A, \xi, \omega, \phi.$ which can be used for the determination of A_2, A_3 as

$$A_2 = -\frac{-C_\alpha + 2I\xi\omega}{\rho B^4\omega} \;\;;\;\; A_3 = -\frac{-K_\alpha + I\omega^2}{\rho B^4\omega^2} \tag{101}$$

The derivatives H_2 and H_3 can be identified making use of the recorded lift forces using a technique that is very similar to the one performed in the previous section. The lift history can be seen as generated from an imposed displacement in pitch described by an equation such as Eqn(97). The first step is therefore to fit the lift history with the following function

$$L_{fitted} = B_{lift}^{pitch} e^{-\xi\omega t} sin(\omega t + \phi) + C_{lift}^{pitch} e^{-\xi\omega t} cos(\omega t + \phi) \tag{102}$$

The second step is to substitute Eqn(100) into the moments equation to obtain a theoretical expression in the form

$$L_{teo} = -\rho B^3 \omega^2 A e^{-\xi\omega t} \left(H_2 \xi \sin(\omega t + \phi) - H_2 \cos(\omega t + \phi) - H3 \sin(\omega t + \phi) \right) \tag{103}$$

by equating the last equations, collecting the "cos" and "sin" terms and equating them to zero, we obtain

$$H_2 = \frac{C_{lift}^{pitch}}{\rho B^3 \omega^2} \;\; ; \;\; H_3 = \frac{B_{lift}^{pitch} + C_{lift}^{pitch} \xi}{\rho B^3 \omega^2} \tag{104}$$

which is the desired result.

11.3 Free heave

The identification of the remaining four derivatives follows a similar path. Once the non linear fitting of the lift equation is performed, two of the aeroelastic parameters can be identified as

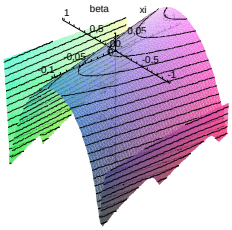
$$H_1 = -\frac{-C_y + 2m\xi\omega}{\rho B^2 \omega} \;\; ; \;\; H_4 = -\frac{-K_y + m\omega^2}{\rho B^2 \omega^2} \tag{105}$$

The remaining two parameters, after fitting the moment equation as described above, take the form

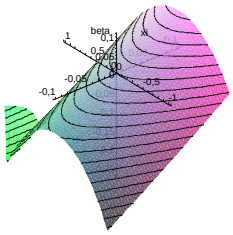
$$A_1 = \frac{B_{mom}^{heave} + C_{mom}^{heave} \xi}{\rho B^3 \omega^2} \;\; ; \;\; A_4 = \frac{C_{lift}^{pitch}}{\rho B^3 \omega^2} \tag{106}$$

The identification of the parameters can be performed by non linear fitting techniques. The nonlinear fitting routines provided in "scipy" a scientific library included in "python" (see www.python.org) were used in the computations. The convergence of the fitting process was ensured by taking as initial value for the fitting process an estimate for all the relevant parameters.

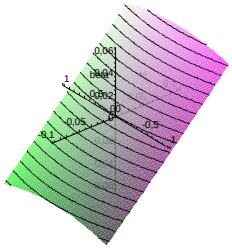
Figure 3: Plot of $\rho_{approx} - 1$



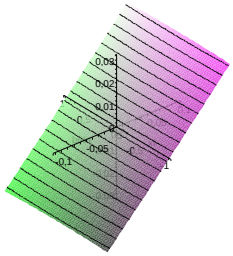
(a) $n_{div} = 2$



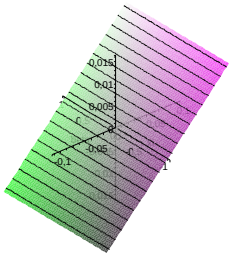
(b) $n_{div} = 4$



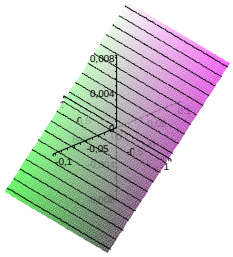
(c) $n_{div} = 10$



(d) $n_{div} = 20$

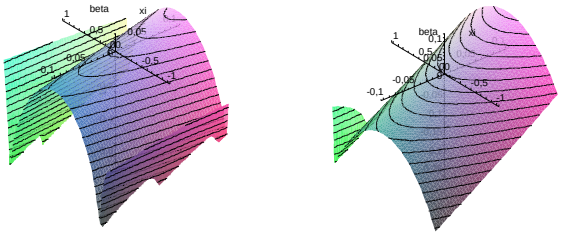


(e) $n_{div} = 40$



(f) $n_{div} = 80$

Figure 4: Plot of $\frac{\rho_{approx} - \rho_{exact}}{\rho_{exact}}$



(a) $n_{div} = 2$ (b) $n_{div} = 4$

Figure 5: Error plot: Aitken Acceleration - red line, "Consistent Acceleration" - green line. $\alpha = 0.5$, $\beta = 0.0$

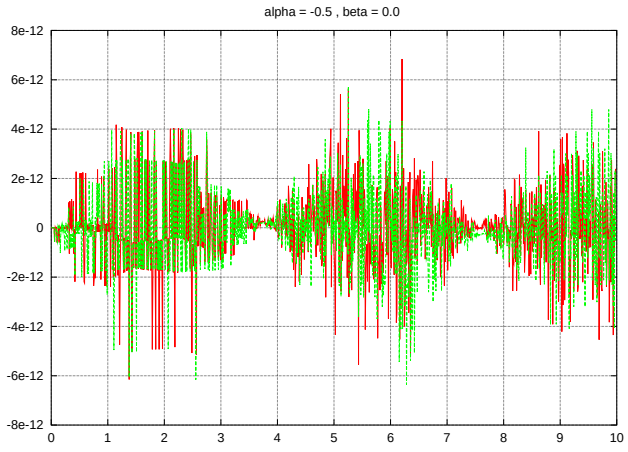


Figure 6: Comparison between exact and iterative coupled solutions using the Bossak Algorithm. Bossak parameter = -0.1, period divided in 50 steps - Monolithic solution is dashed while the solution of iterative coupling is represented with a continuum line

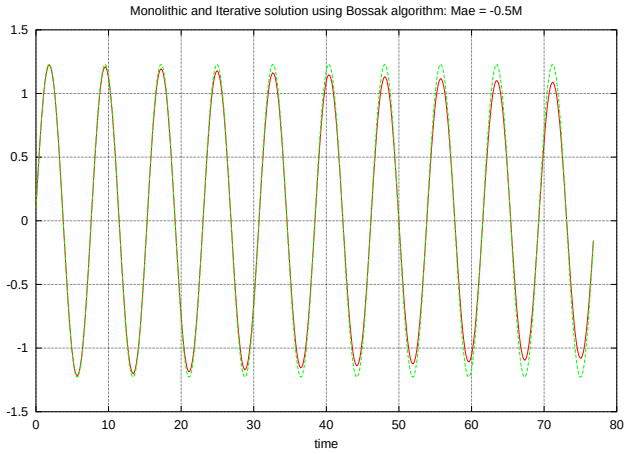


Figure 7: Comparison between analytical and numerical solution for the cube model problem

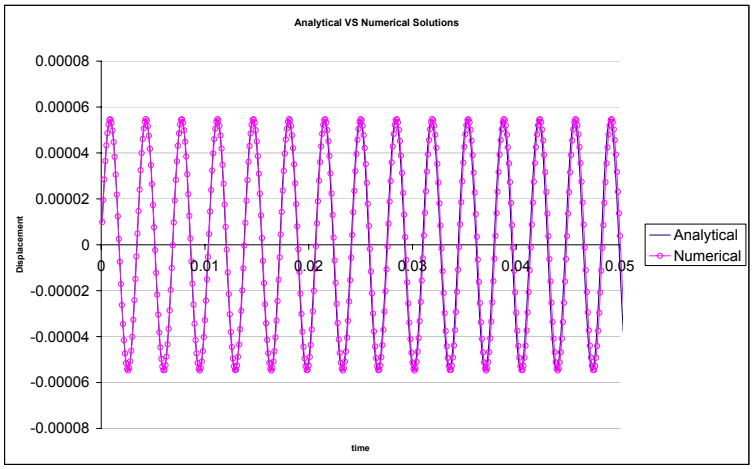


Figure 8: Acceleration error for the simple cube benchmark scheme

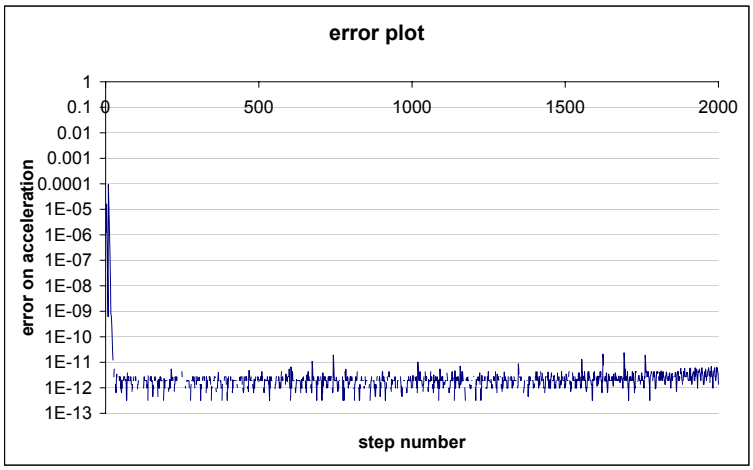


Figure 9: Restrictor flap - mesh of 7928 elements

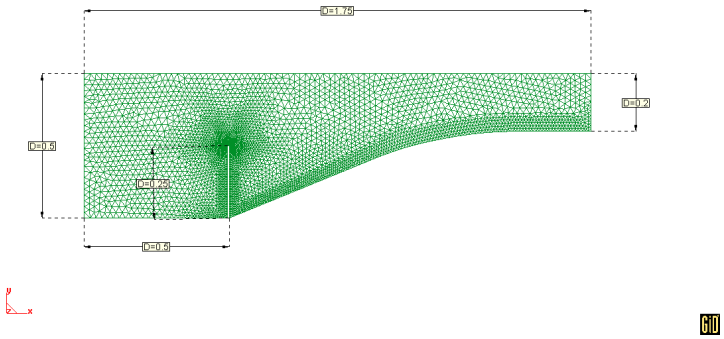
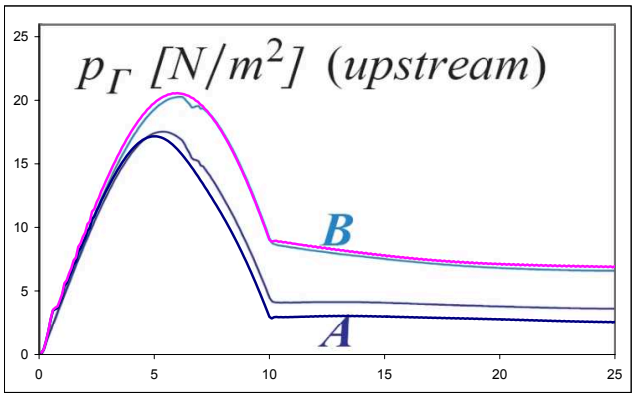


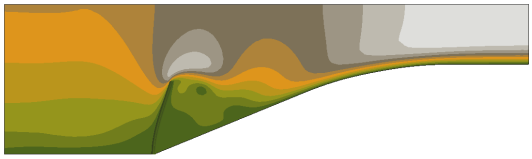
Figure 10: Pressure vs time at mid-height (A) and at the tip (B)
, Results of pressure VS time are shown on the top of the curves found in the



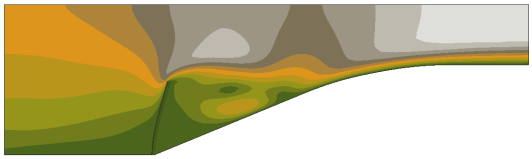
literature

Figure 11: Analysis of a restrictor flap. Contours of velocity. Symmetry boundary condition is used on the top boundary

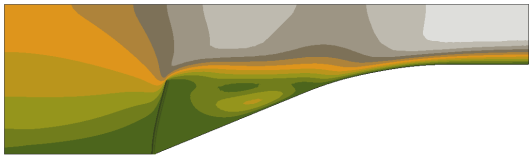
(a) t = 5s



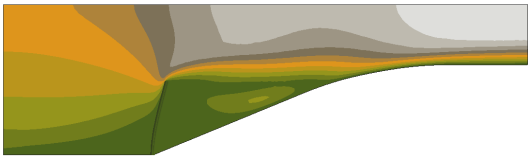
(b) t = 10s



(c) t = 15s



(d) t = 20s



(e) t = 25s

Figure 12: Flexible plate behind a square bluff body - view of the domain

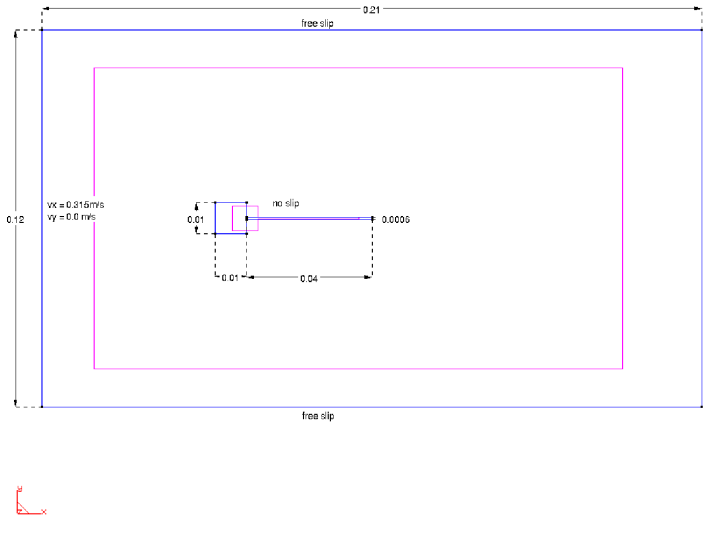


Figure 13: Flexible plate, tip displacement history - Numerical results

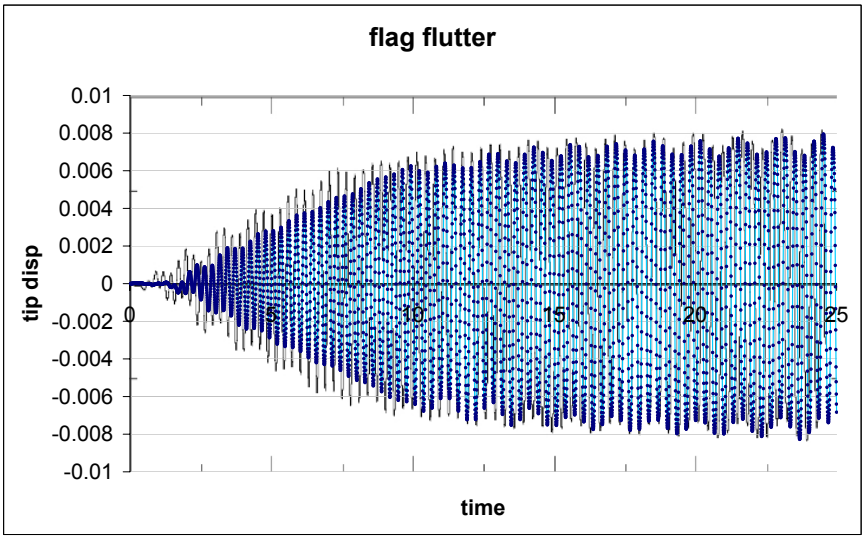


Figure 14: Flexible plate, Fourier analysis of tip displacement

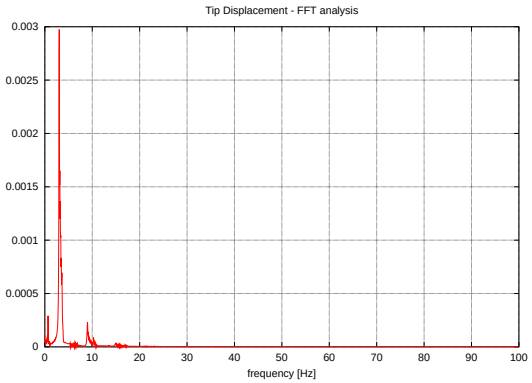


Figure 15: Fitting of experimental aeroelastic derivatives, using a first order accurate solver for the fluid domain

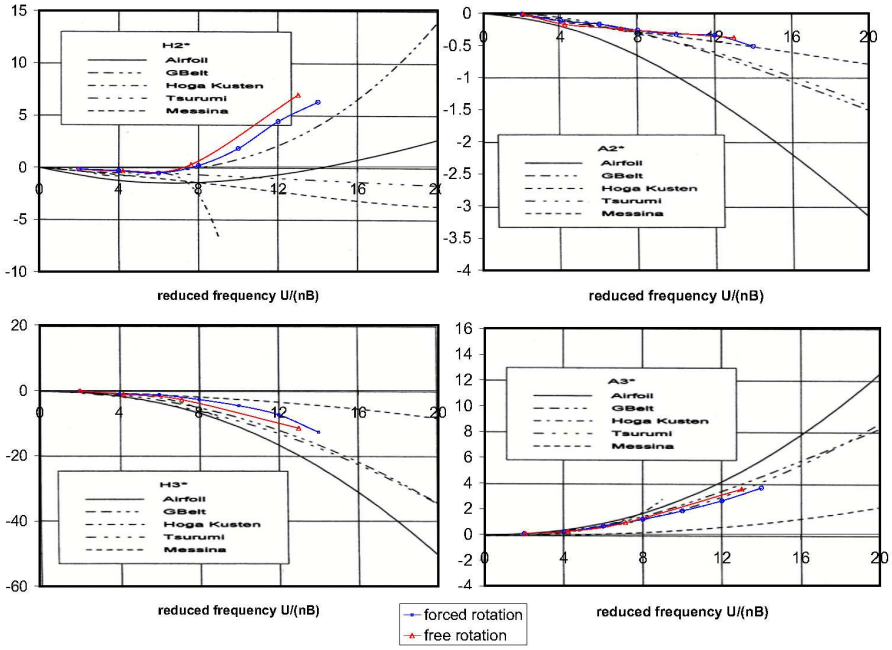
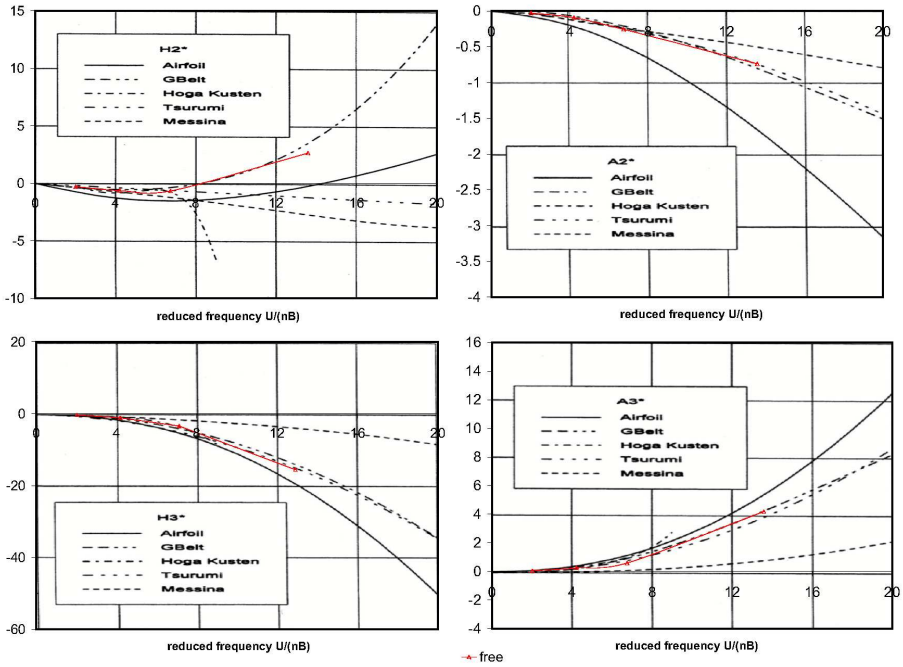


Figure 16: Free rotation fitting of experimental aeroelastic derivatives using a second order accurate solver for the fluid domain



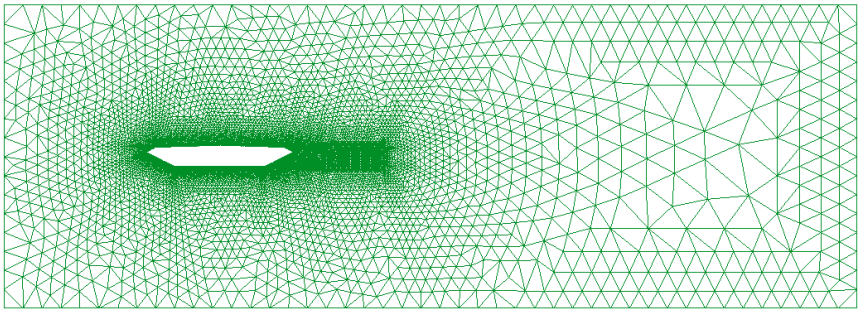


Figure 17: Cross section of Great Belt suspension bridge. View of the mesh used in the computations.