An accelerated, fully-coupled, parallel 3D hybrid finite-volume fluid–structure interaction scheme

A. G. Malan[1,*], O. F. Oxtoby

Advanced Computational Methods research group, Aeronautic Systems, Council for Scientific and Industrial Research, Pretoria, South Africa

Abstract

In this paper we propose a fast, parallel 3D, fully-coupled partitioned hybrid-unstructured finite volume fluid–structure-interaction (FSI) scheme. Spatial discretisation is effected via a vertex-centered finite volume method, where a hybrid nodal-elemental strain procedure is employed for the solid in the interest of accuracy. For the incompressible fluid, a split-step algorithm is presented which allows the entire fluid-solid system to be solved in a fully-implicit yet matrix-free manner. The algorithm combines a preconditioned GMRES solver for implicit integration of pressures with dual-timestepping on the momentum equations, thereby allowing strong coupling of the system to occur through the inner solver iterations. Further acceleration is provided at little additional cost by applying LU-SGS relaxation to the viscous and advective terms. The solver is parallelised for distributed-memory systems using MPI and its scaling efficiency evaluated. The developed modelling technology is evaluated by application to two 3D FSI problems. The advanced matrix-free solvers achieve reductions in overall CPU time of up to 50 times, while preserving close to linear parallel computing scaling using up to 128 CPUs for the problems considered.

Keywords: 3D fluid-structure-interaction, vertex-centered finite volume, partitioned strongly coupled, implicit, parallel

1. Introduction

Fluid–Structure Interaction (FSI) is a growing field within computational mechanics with important industrial applications, for example aeroelastic flutter response [1, 2, 3], design of valves in air compressors and shock absorbers [4, 5] and analysis of stresses therein, parachute dynamics [6, 7], the effect of

*Corresponding author

Email addresses: Arnaud.Malan@uct.ac.za (A. G. Malan), ooxtoby@csir.co.za (O. F. Oxtoby)

[1]Present address: Department of Mechanical Engineering, University of Cape Town, Private Bag X3, Rondebosch 7701, South Africa

wind on architectural structures [8, 9], and biomechanics [10, 11]. In real three-dimensional systems such as these, geometries are often complex and require the use of hybrid unstructured meshes. In addition, FSI problems are typically transient or oscillatory in nature, requiring the solution of thousands of time-steps. The result is that FSI calculations are burdened by high computational cost, which is at present a major obstacle to becoming a routinely used tool in industry. However, recent progress in FSI modelling technology [4, 12, 13, 14, 15] gives cause for optimism. In this paper we extend the multiphysics code *Elemental* [16, 17, 18, 19, 20, 21] to three-dimensional FSI scenarios, and investigate advanced solver strategies with the aim of achieving significant improvements in parallel computational efficiency.

In order to robustly solve fluid-solid interaction problems which are physically strongly coupled, it is advantageous to fully converge the entire system at each timestep. This is such that both dynamic and kinematic continuity – i.e. continuity of forces and velocities – are satisfied at the fluid/solid interface. So-called monolithic methods ensure this by solving the entire coupled system [22, 23, 24]. Partitioned solvers, on the other hand, are more flexible in allowing independent treatment of the fluid and solid, but require more care to be taken to ensure a stable and efficient coupling procedure. Techniques which have been developed include fixed-point iteration accelerated by Aitken or gradient methods [25, 26, 27, 28], coarse grid predictors [29], approximate [30, 31, 32, 33] and even exact [34, 35] Newton-type methods, and the use of reduced-order models [36]. Here we present a scheme where dual-timestepping [37, 38] is employed to effect the fully-coupled, implicit solution of the FSI system in a partitioned manner. While none of the techniques above have been employed to guarantee coupling stability, the method presented was found to be robust for the test cases presented.

In order to accomodate complex geometries, both fluid and solid governing equations are spatially discretised via a vertex-centered, edge-based finite volume method [18, 39]. Edge-based methods hold the advantages of generic applicability to hybrid-unstructured meshes, ideal parallelisation properties and improved computational efficiency compared to element-based approaches [40, 41]. In the case of the solid domain, we use a hybrid between the traditional node-based finite volume method (which suffers from locking of high aspect-ratio elements) and the element-based strain method [42] (which suffers from odd-even decoupling), to circumvent both of these problems [43].

In this paper fluid incompressibility is dealt with using a new artificial compressibility split-step method. Traditionally, incompressible solvers use pressure projection [44] techniques to derive an equation for pressure from the incompressibility constraint. Alternatively, artificial compressibility [45] may be used to solve for pressure in a matrix-free manner. It was however recently proposed to combine the aforementioned in order to combine the desirable properties of each [46]. Here we propose a simplified strategy where matrix-free solution is maintained through artificial compressibility, but instead of pressure projection, the consistent numerical fluxes of Löhner *et al.* [47, 48] are applied directly to the continuity equation to allow stable solution of pressure and in a manner which

is free from non-physical oscillations. As noted, the use of dual timestepping allows us to achieve a fully-coupled implicit, matrix-free solution.

The use of artificial compressibility for the fluid allows for matrix-free solution and efficient parallelisation. However, since the actual speed of pressure-wave propagation is infinite in incompressible flow, pressure waves have to equalise across the entire fluid domain during every timestep, a process which can markedly slow convergence. Therefore an implicit method of solving the pressure equation, without sacrificing its matrix-free nature, can give significant returns in computation time. When considering the momentum equation, there is evidence that for transient problems, explicit timestepping can be as fast or faster than implicit matrix-free methods [48]. Through the use of dual timestepping, further performance improvement can result [49, 50] due to the increase in the allowable time-step size. In the latter work, however, the pressure equation is still treated in a fully explicit manner. We attempt to combine the most favourable characteristics of the cited schemes in one algorithm in order to furnish an efficient and robust method. As argued, dual timestepping is an attractive option for an FSI solver, and we combine this with an LU-SGS preconditioned GMRES technique, pioneered by Luo *et al.* [51] in the context of compressible flow, applied to the pressure equation. We further show that additional speed-ups may be achieved by applying LU-SGS relaxation to the momentum equations as suggested by Löhner *et al.* [48]. Combined with the concurrent solution of the solid via a Jacobi method, this results in a novel, fast, matrix-free, fully coupled parallel FSI solver which is suited to model complex strongly-coupled 3D problems. The developed technology is evaluated via application to two strongly-coupled three-dimensional problems from literature. Metrics considered include speed-ups achieved by the advanced solver techniques, as well as overall parallel computing efficiency.

The outline of this paper is as follows. In Section 2 we present the govering equations for fluid and solid domains, then describe the spatial discretisation and mesh movement in Section 3. Section 4 details the numerical solvers and coupling algorithm and in Section 5, parallelisation of the code is discussed. We present numerical applications in Section 6 before concluding in Section 7.

2. Governing Equations

The physical FSI domain to be modelled consists of a Newtonian incompressible fluid and homogeneous isotropic elastic solid undergoing geometrically non-linear displacements. The mechanics of each is governed by the appropriate governing equations, which are described next. In this work the fluid mesh is moved in sympathy to the deforming solid and the internal nodes moved using the mesh movement algorithm described in Section 3.3.

2.1. Fluid equations

To account for movement of the mesh, described by velocity field $\mathbf{u}^*$, it is necessary to describe the governing equations in an Arbitrary Lagrangian

Eulerian (ALE) reference frame. To do this we write the governing equations in weak form over an arbitrary and time-dependent volume $\mathcal{V}(t)$ as

$$\frac{\partial}{\partial t}\int_{\mathcal{V}(t)} \mathbf{W} d\mathcal{V} + \int_{\mathcal{S}(t)} \left(\mathbf{F}^j + \mathbf{H}^j - \mathbf{G}^j\right) n_j d\mathcal{S} = \int_{\mathcal{V}(t)} \mathbf{Q} d\mathcal{V}, \tag{1}$$

where

$$\mathbf{W} = \begin{pmatrix} W_0 \\ W_1 \\ W_2 \\ W_3 \end{pmatrix} = \begin{pmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \end{pmatrix}, \quad \mathbf{H}^j = \begin{pmatrix} 0 \\ p\delta_{1j} \\ p\delta_{2j} \\ p\delta_{3j} \end{pmatrix}, \quad \mathbf{G}^j = \begin{pmatrix} 0 \\ \sigma_{1j} \\ \sigma_{2j} \\ \sigma_{3j} \end{pmatrix}, \tag{2}$$

$$\mathbf{F}^j = \mathbf{W}(u_j - u_j^*). \tag{3}$$

Here, $\mathcal{S}(t)$ denotes the surface of the volume $\mathcal{V}(t)$ with $\mathbf{n}$ being a unit vector normal to $\mathcal{S}(t)$; $\mathbf{Q}$ is a vector of source terms (e.g. body forces), $\mathbf{u}$ denotes velocity, p is the pressure, ρ is density, $\boldsymbol{\sigma}$ stress, and δ_{ij} is the Kronecker delta.

The governing equations are closed via the relationship between stress and rate of strain:

$$\sigma_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \tag{4}$$

where μ is the dynamic viscosity and x_i are the fixed (Eulerian) co-ordinates.

2.2. Solid Equations

To account for large deformations without accumulating strain errors due to repeated oscillations, the solid equations are formulated in a total Lagrangian formulation, i.e. in the undeformed reference frame. In this coordinate system the momentum equations are written in weak form as

$$\frac{\partial}{\partial t}\int_{\mathcal{V}_0} \rho_0 v_i d\mathcal{V} = \int_{\mathcal{S}_0} P_{ij} n_j d\mathcal{S} + \int_{\mathcal{V}_0} Q_i d\mathcal{V}, \tag{5}$$

where $\mathbf{v}$ is the solid velocity, $\mathbf{P}$ is the first Piola-Kirchoff stress tensor, $\mathcal{V}_0$ denotes a volume in the material coordinate system and $\mathcal{S}_0$ its surface, with $\mathbf{n}$ being its outward pointing unit normal vector. Further, ρ_0 is the solid density in its undeformed state and $\mathbf{Q}$, again, a vector of source terms.

We use the St. Venant–Kirchoff model to account for finite strains. In this model, the Green-Lagrange strain tensor $\mathbf{E}$ is used, with components

$$E_{ij} = \frac{1}{2}\left(\frac{\partial w_i}{\partial X_j} + \frac{\partial w_j}{\partial X_i} + \frac{\partial w_k}{\partial X_i}\frac{\partial w_k}{\partial X_j}\right) \tag{6}$$

where $\mathbf{w}$ is the total displacement of the solid from equilibrium location $\mathbf{X}$, so that $\mathbf{x} = \mathbf{X} + \mathbf{w}$. The second Piola-Kirchoff stress tensor $\mathbf{S}$ is related to the strain tensor by

$$S_{ij} = 2G(E_{ij} - \tfrac{1}{3}E_{kk}\delta_{ij}) + KE_{kk}\delta_{ij} \tag{7}$$

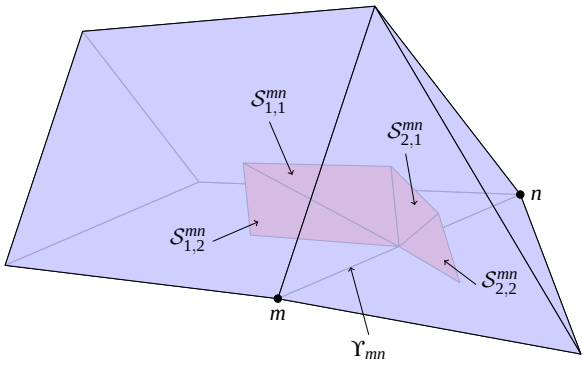

Figure 1: Schematic diagram of the construction of the median dual-mesh on hybrid grids. Here, Υ_{mn} depicts the edge connecting nodes m and n. Shaded is the dual-cell face between nodes m and n. This is composed of triangular surfaces $\mathcal{S}^{mn}_{kl}$ which are constructed by connecting edge centers, face centroids and element centroids as described in the text.

where G is the shear modulus and K the bulk modulus (summation over k implied). Finally, we convert from the second to the first Piola-Kirchoff stress tensor, $\mathbf{P}$, with

$$P_{ij} = F_{ik}S_{kj}, \text{ where } F_{ik} = \delta_{ik} + \frac{\partial w_i}{\partial X_k}. \tag{8}$$

To close the system of equations, the solid velocity $\mathbf{v}$ and displacement $\mathbf{w}$ are related in the obvious way:

$$\frac{Dw_j}{Dt} = v_j. \tag{9}$$

where D/Dt denotes the derivative with respect to a fixed point in the reference (initial) configuration.

3. Spatial Discretisation

3.1. Hybrid-Unstructured Discretisation

In this work, we use a vertex-centred edge-based finite volume algorithm for the purposes of spatial discretisation, where a compact stencil method is employed for all Laplacian terms in the interests of both stability and accuracy [18, 39], for both fluid and solid domains. The method allows natural generic mesh applicability, second-order accuracy without odd-even decoupling [18], and computational efficiency which is factors greater than element based approaches [40]. The edge-based method is also particularly well suited to computation on parallel hardware architectures due to the constant computational cost per edge.

Considering the fluid ALE governing equations (1), all surface integrals are accordingly calculated in an edge-wise manner. For this purpose, bounding surface information is similarly stored per edge and termed *edge-coefficients*.

The latter for a given internal edge Υ_{mn} connecting nodes m and n (see Fig. 1, is defined as a function of time as

$$\mathbf{C}_{mn}(t) = \sum_{k \in \mathcal{E}_{mn}} \left[\mathbf{n}_{k,1}^{mn}(t) \mathcal{S}_{k,1}^{mn}(t) + \mathbf{n}_{k,2}^{mn}(t) \mathcal{S}_{k,2}^{mn}(t) \right] \tag{10}$$

where $\mathcal{E}_{mn}$ is the set of all elements k containing edge Υ_{mn}, $\mathcal{S}_{kl}^{mn}$ is the area of the triangle connecting the centre of edge Υ_{mn} with the centroid of element k and the centroid of one of its two faces which is also incident on Υ_{mn}, $l = 1, 2$. Further, $\mathbf{n}_{kl}^{mn}$ are the associated unit vectors normal to these triangles and oriented from node m to node n. The discrete form of the surface integral in Eq. (1), computed for the volume surrounding the node m, now follows as

$$\int_{\mathcal{S}_m(t)} \left\{ \mathbf{F}^j + \mathbf{H}^j - \mathbf{G}^j \right\} n_j d\mathcal{S} \approx \sum_{\Upsilon_{mn} \cap \mathcal{V}_m(t)} \left\{ \overline{\mathbf{F}^j}_{mn} + \overline{\mathbf{H}^j}_{mn} - \overline{\mathbf{G}^j}_{mn} \right\} C_{mn}^j \tag{11}$$

where all $\overline{\bullet}_{mn}$ quantities denote face values. In the case of the fluid, $\overline{\mathbf{G}^j}_{mn} = \left[\overline{\mathbf{G}^j}_{mn}\big|_{tang} + \overline{\mathbf{G}^j}_{mn}\big|_{norm} \right]$ [18], where $\mathbf{G}^j\big|_{tang}$ is calculated by employing directional derivatives and $\mathbf{G}^j\big|_{norm}$ is approximated by employing the standard finite volume first-order derivative terms.

When considering the solid governing equations (5), the above method of discretising the stress term would enforce continuous gradients at nodes and element boundaries and therefore suffers from stiffness problems on stretched elements [43]. To remedy this we use the hybrid nodal/elemental strain technique [43]. In this method, components of strain E_{ij} with $i = j$ are obtained as per the above. However, the components with $i \neq j$ are evaluated at the element centre and averaged with the neighbouring element's value to obtain the value at the edge centre.

3.2. Geometric Conservation

In order for the fluid solution to be as transparent as possible to the movement of the mesh, an identity known as the Geometric Conservation Law (GCL) should be obeyed [52, 53, 54]. It asserts that the momentum flux into a cell due to the motion of the faces should be consistent with the change in momentum of the cell due to its changing volume. That is, the discretised version of

$$\frac{\partial}{\partial t} \int_{\mathcal{V}(t)} d\mathcal{V} = \int_{\mathcal{S}(t)} u_j^* n_j d\mathcal{S}. \tag{12}$$

should hold exactly, which implies that constant spatial fields will be unaffected by arbitrary mesh deformations. The GCL can therefore be said to impose a specific relationship between mesh deformation and the mesh-velocity field. We could discretise the equation above at node m as

$$\frac{V_m^{t+\Delta t} - V_m^t}{\Delta t} = \sum_{\Upsilon_{mn} \cap \mathcal{V}_m(t)} \frac{\delta V_{mn}^t}{\Delta t}, \tag{13}$$

where V is the volume of dual-cell $\mathcal{V}$, δV_{mn}^t is the volume swept out by the face lying between nodes m and n between time-steps t and $t+\Delta t$. Then, the GCL will hold. Alternatively, to second order accuracy [54]

$$\frac{3V_m^{t+\Delta t} - 4V_m^t + V_m^{t-\Delta t}}{2\Delta t} = \sum_{\Upsilon_{mn} \cap \mathcal{V}_m(t)} \frac{3\delta V_{mn}^t - \delta V_{mn}^{t-\Delta t}}{2\Delta t}. \tag{14}$$

Thus, in order for our discretisation to be consistent with the GCL, the mesh velocity flux $\overline{u_j^*} C_{mn}^j$ in the discretisation of (1) is set equal to $(3\delta V_{mn}^t - \delta V_{mn}^{t-\Delta t})/2\Delta t$.

3.3. Dynamic Mesh Movement

For the purposes of mesh movement, we employ an interpolation procedure [55] which, while offering no guarantees about element quality, has no significant computational cost and is well suited to parallel computing. This approach entails redistributing internal fluid nodes via the following interpolation function:

$$\Delta \mathbf{x} = r\Delta \mathbf{x}_1 + (1-r)\Delta \mathbf{x}_2,$$

where $\Delta \mathbf{x}_1$ and $\Delta \mathbf{x}_2$ respectively denote the displacement of the closest internal and external boundary nodes from their initial locations, and r, which varies between zero and one, is computed as

$$r = \frac{D_2^p}{D_1^p + D_2^p} \quad \text{with } p = 3/2. \tag{15}$$

Here, D_1 and D_2 are the distances to the identified internal and external boundary points in the undeformed configuration. Since the values of r are calculated only once at the beginning of the entire analysis, the parallel computational overhead of the mesh movement function is negligible; furthermore, cumulative deterioration of element quality during repeated oscillations does not occur as the mesh always returns to its initial configuration. While this method has been found to perform well for many problems which include non-linear motion (such as rotations), it is somewhat sensitive to the magnitude of motion as mesh quality is not explicitly enforced.

4. Temporal Discretisation and Solution Procedure

The solution procedure must allow for fast, parallel, fully coupled solution of all descritised equations while allowing independence in terms of discretisation and solution strategy employed for the fluid and solid domains. We therefore advocate a strongly coupled, partitioned, matrix-free iterative solution process where fluid-solid-interface nodes communicate velocities and tractions at each iteration. The resulting proposed solution procedure is detailed below.

4.1. Temporal Discretisation

For the purpose of transient calculations, a dual-time-stepping temporal discretisation [16] is employed such that second-order temporal accuracy is achieved while ensuring that all equations are iteratively solved simultaneously in an *implicit* fashion. This results in a strongly coupled solution. The real-time temporal term is accordingly discretised to second-order and added as a source term to the right-hand-side of the discretised fluid momentum equation as

$$Q_i V\big|^{\tau} = -\frac{3W_i^{\tau+\Delta\tau}V^{\tau} - 4W_i^{t}V^{t} + W_i^{t-\Delta t}V^{t-\Delta t}}{2\Delta t} \quad \text{for } i = 1, 2, 3 \tag{16}$$

where Δt denotes the real-time-step size, the t superscript is the previous (existing) real time-step and τ denotes the latest known solution to the time-step being solved for viz. $t + \Delta t$.

For the solid equation, a similar source term is added to the right-hand side of the discretised version of Eq. (5), namely

$$Q_i^{\tau} V_0 = -\rho_0 \frac{3v_i^{\tau+\Delta\tau} - 4v_i^{t} + v_i^{t-\Delta t}}{2\Delta t} V_0 \tag{17}$$

and also to the discrete form of Eq. (9), to give

$$\frac{Dw_i}{Dt} = v_i - \frac{3w_i^{\tau+\Delta\tau} - 4w_i^{t} + w_i^{t-\Delta t}}{2\Delta t} \tag{18}$$

where the nomenclature is as defined previously.

4.2. Solution Procedure: Fluid

The solution of the incompressible fluid equations (2) presents two numerical difficulties. Firstly, the spatial discretisation of the convective terms via central differencing results in destabilising odd-even decoupling, and secondly, the pressure field must evolve such that the incompressible continuity equation $\nabla \cdot \mathbf{u} = 0$ is satisfied. Since this equation does not involve pressure, solving for it in a matrix-free manner is not straightforward. We use a new artificial compressibility split-step algorithm stabilised with consistent numerical fluxes [55] (CNF-AC method) to overcome this difficulty, as described below.

For incompressible flow it is usually advocated that the pressures are solved implicitly [48] while advective (convective) terms are explicitly integrated [46, 48]. This is as the advective timescales are those of interest, whereas pressure waves propagate instantaneously. Löhner *et al.* [48] have undertaken a thorough investigation into various explicit and implicit integration strategies, with the fastest solution times for transient problems achieved using explicit integration of the advective terms. However, for strongly-coupled FSI systems, the use of an implicit fluid solution methodology allows for strongly coupled concurrent partitioned solution of both fluid and solid domains. We therefore advocate such an approach for this work.

Dual-timesteping has long been used to achieve second-order accurate temporal solutions without introducing the extra computational cost of a more traditional implicit solver [16, 18, 37, 38, 49]. In the context of FSI, it confers the additional advantage of allowing the coupled system to be implicitly solved with the sub-iteration process serving the purpose of effecting numerical strong coupling between fluid and solid. We therefore propose the use of dual-timestepping for the momentum equations combined with implicit integration of pressures.

We now describe the split-step incompressible flow algorithm. The CNF-AC scheme addresses the continuity equation directly by adding artificial compressibility and fourth-order stabilisation terms as follows:

$$\frac{1}{c_\tau^2}\frac{p^{\tau+\Delta\tau_2}-p^\tau}{\Delta\tau_2}V^\tau = -\int_{\mathcal{S}(t)}\left[\rho u_k\Big|^\tau + \frac{\Delta\tau}{\ell}(p_R-p_L)n_k\Big|^{\tau+\alpha\Delta\tau_2}\right]n_k dS \quad (19)$$

where α controls the level of implicitness and $\Delta\tau_2$ is the iterative or pseudo-time-step size (solution involves driving the right-hand-side of the equation to zero). Further, p_R denotes the pressure extrapolated from the node outside the control volume (denoted n below), p_L denotes the pressure extrapolated from the node inside it (denoted m) and ℓ is the length of the edge connecting nodes m and n. In the context of upwinding the aforementioned typically refer to the left and right states respectively. This stabilisation term is essentially the same as the 'consistent numerical flux' of Löhner *et al.* [47, 48]. Using constant interpolation (i.e. the nodal values for p_R and p_L) yields an effective second-order (Laplacian) stabilisation term, whereas an effective fourth-order stabilisation term results from the third-order accurate linear interpolation used in the MUSCL scheme [56]:

$$p_L = p_m + \frac{1}{2}\big[(1-\kappa)\nabla p|_m\cdot\boldsymbol{\ell} + \kappa(p_n-p_m)\big], \quad \kappa = 1/3. \quad (20)$$

Here, $\boldsymbol{\ell}$ is the vector from node m to node n.

In the interest of computational efficiency, we use the implicit formulation ($\alpha = 1$) to approximate the interpolated pressures, and in such a way as to use only nearest-neighbour values of the increment $\Delta p \equiv p^{\tau+\Delta\tau_2} - p^\tau$ being solved for:

$$(p_R-p_L)^{\tau+\Delta\tau_2} \approx (p_R-p_L)^\tau + (\Delta p_n - \Delta p_m). \quad (21)$$

Via numerical experimentation it is found that the computational cost of additional iterations due to the inexact Jacobian is more than offset by the saving due to the greater sparsity of the Jacobian.

Having solved (19) for $p^{\tau+\Delta\tau_2}$, a momentum iteration follows as

$$\frac{W_i^{\tau+\Delta\tau}-W_i^\tau}{\Delta\tau}V^\tau = -\int_{\mathcal{S}(t)}(F_i^j - G_i^j)n_j dS\Big|^{\tau+\beta\Delta\tau} - \int_{\mathcal{S}(t)} H_i^j n_j dS\Big|^{\tau+\Delta\tau_2} + Q_i V|^\tau \equiv R_i(\mathbf{W}), \quad (22)$$

where β is set to 1 for the implicit treatment of velocity terms and zero otherwise.

The rationale for the implicit treatment of velocities is as follows. There are two pseudo-time-step size restrictions on $\Delta\tau$ in Eq. (22): the viscous timestep due to the diffusive terms in $\mathbf{G}$ and the convective timestep due to $\mathbf{F}$. The former is often more restrictive than the latter, scaling with the square of mesh spacing rather than the mesh spacing itself. Moreover, the convective timescales are generally those of interest in transient problems. This means that it is worthwhile to solve the viscous terms implicitly (provided a computationally efficient solver is used) whereas increasing the convective timestep size dramatically would result in a loss of temporal accuracy. The latter would in addition be far more computationally intensive as cross-coupling between all governing equations would have to be accounted for. However, to facilitate modest extensions of the allowable convective pseudo-timestep size, convective velocity components which are not cross coupled are included in the Jacobian. This results in a nett reduction in computational cost (as will be reported on below). Thus, the convective and viscous velocity terms in Eq. (22) are treated as follows:

$$F_i^j|^{\tau+\beta\Delta\tau} \approx W_i|^{\tau+\beta\Delta\tau}(u_j|^{\tau+\delta_{ij}\beta\Delta\tau} - u_j^*|^{\tau}) \tag{23}$$

and

$$G_i^j|^{\tau+\beta\Delta\tau} \approx \mu\left(\frac{\partial u_i}{\partial x_j}\Big|^{\tau+\beta\Delta\tau} + \frac{\partial u_j}{\partial x_i}\Big|^{\tau+\beta\delta_{ij}\Delta\tau}\right) \tag{24}$$

for $i, j = 1, 2, 3$. Note that the second term above does not contribute to the governing equations in the case of constant μ due to the continuity equation and thus the cross-derivative terms may be treated explicitly.

Finally, the fluid mesh velocity $\mathbf{u}^*$ is calculated via second-order backward difference as per Eq. (17).

4.2.1. Preconditioned GMRES Routine

As mentioned, we wish to solve Eqs (19) and (22) implicitly in order to overcome the time-step-size restrictions on $\Delta\tau_2$ and on $\Delta\tau$ to which explicit methods are subject. However, in order to scale efficiently to very large problems, the procedures must be matrix-free. Different approaches are required to efficiently solve the pressure equation (19) and the momentum equations (22), since in the former case the pseudo-time-step size should be increased as much as possible, while in the latter it remains restricted by the convective limit. Thus, it is more efficient to solve the momentum equations with more iterations of a computationally less intensive solver, compared to the pressure equations, which we now consider first.

A popular approximate matrix solver is the Generalised Minimum Residual (GMRES) method of Saad and Schultz [57], which finds an optimum solution within a reduced subspace spanned by the so-called Krylov vectors. The choice of the preconditioner to be applied to the GMRES Krylov-subspace vectors is of utmost importance. It must ensure a good condition number while, importantly, preserving the matrix-free nature of the solution scheme and being computationally efficient. Luo *et al.* [51] were the first to employ LU-SGS [58, 59, 60] for this purpose. As compared to its main competitor, Incomplete

Lower-Upper decomposition (ILU), it does not require the storage of any part of the preconditioning matrix. Furthermore, it has been applied with success to the nonlinear heat conduction equation [20], showing orders of magnitude performance improvement over Jacobi iterations and even over both GMRES and LU-SGS in isolation. As the pressure equations have a diffusive character similar to the heat equation, a similar methodology was adopted for this work.

The discrete form of Eq. (19) can be written in the form

$$\mathbf{A}\Delta\mathbf{p} = \mathbf{Res} \tag{25}$$

where $\Delta\mathbf{p} = \mathbf{p}^{\tau+\Delta\tau_2} - \mathbf{p}^{\tau}$ and $\mathbf{Res}$ is the residual vector. The lengths of the vectors $\Delta\mathbf{p}$ and $\mathbf{Res}$ are equal to the number of nodes N, and $\mathbf{A}$ is a sparse coefficient matrix. An outline of the LU-SGS preconditioned, restarted GMRES procedure follows.

At each iteration the change in the primitive variable vector $\mathbf{p}$ is calculated from $\Delta\mathbf{p} = \mathbf{v}^l a_l, l = 1 \dots L$. Here L denotes the number of preconditioned Krylov-space vectors $\mathbf{v}^l$, and the coefficients a_l are calculated such that they minimise the residual $\mathbf{Res}$ for a given set of Krylov-space vectors (the expression from which a_l is calculated follows below). The latter are calculated via the following GMRES procedure, which is invoked iteratively for a pre-specified number of GMRES iterations.

1. Initialise $\Delta\mathbf{p}_0 = \mathbf{0}$.
2. Starting Krylov-subspace vector:

$$\mathbf{v}^1 = \left[\mathbf{P}^{-1}(\mathbf{Res} - \mathbf{A}\Delta\mathbf{p}_0)\right] \left|\mathbf{P}^{-1}(\mathbf{Res} - \mathbf{A}\Delta\mathbf{p}_0)\right|^{-1} \tag{26}$$

   where $\mathbf{P}^{-1}$ denotes the preconditioning matrix, or $\mathbf{P}^{-1} \approx \mathbf{A}^{-1}$, and $|\cdot|$ denotes the Euclidian norm.

3. For $l = 1, 2, \dots, L-1$, compute Gramm-Schmidt orthogonalisation:

$$\mathbf{w}^{l+1} = \mathbf{P}^{-1}\mathbf{A}\mathbf{v}^l - \sum_{k=1}^{l} h^{kl}\mathbf{v}^k, \quad h^{kl} = \mathbf{v}^k \cdot \mathbf{P}^{-1}\mathbf{A}\mathbf{v}^l \tag{27}$$

$$\mathbf{v}^{l+1} = \frac{\mathbf{w}^{l+1}}{|\mathbf{w}^{l+1}|} \tag{28}$$

4. The change in $\mathbf{p}$ is now calculated from the expression given previously, namely

$$\Delta\mathbf{p} = \Delta\mathbf{p}_0 + \mathbf{v}^l a_l \tag{29}$$

   where a_l are calculated such that the residual is minimised:

$$\left(\mathbf{A}\mathbf{v}^k\right) \cdot \left(\mathbf{A}\mathbf{v}^l\right) a_l = \left(\mathbf{A}\mathbf{v}^k\right) \cdot \left(\mathbf{Res} - \mathbf{A}\Delta\mathbf{p}_0\right) \tag{30}$$

5. Restart from Step 2 using $\Delta\mathbf{p}_0 = \Delta\mathbf{p}$ until a pre-set number of iterations complete.

We now outline the preconditioning procedure. The generic vector ω is preconditioned by the LU-SGS procedure by performing two computational sweeps over the mesh:

- Sweep 1: Calculate ω^* from $(\mathbf{D}+\mathbf{L})\omega^* = \omega$, and
- Sweep 2: Calculate ω^p from $(\mathbf{D}+\mathbf{U})\omega^p = \mathbf{D}\omega^*$,

where $\mathbf{L}$, $\mathbf{D}$ and $\mathbf{U}$ are the strict lower, diagonal and upper parts of $\mathbf{A}$. Further, ω^* and ω^p respectively denote the intermediary and preconditioned versions of ω. Note that this procedure may be implemented in a completely matrix-free form.

As mentioned, a fixed number of GMRES iterations (restarts) are used rather than running the GMRES procedure to convergence. (Recall that the GMRES method runs inside the pseudo-time iteration sequence detailed in Section 4.2, which must itself be iterated in order to reach pseudo-steady state.) For the purposes of this work it was found that efficient convergence of the overall scheme is attained with three GMRES iterations (i.e. two restarts). This is because the residual is rapidly reduced at first, with diminishing returns for further restarts.

For fastest convergence of the pseudo-timestep iterations, $\Delta\tau_2$ should be made as large as possible. On the other hand, if it is made too large the performence of GMRES degrades as the $\mathbf{A}$ matrix loses diagonal dominance. For the purpose of this work, $\Delta\tau_2 = 10^5\Delta\tau$ was employed.

Turning now to the implicit solution of velocity in the momentum equations (22), as mentioned, a computationally cheap solver is necessary in order to effect an overall speedup due to the convective timestep size restriction. For the same reason, however, the Jacobian remains diagonally dominant and so it is found that a single LU-SGS sweep per pseudotime iteration – computationally no more expensive than two Jacobi iterations – is sufficient to remove the viscous timestep restriction altogether [61].

The procedure is as follows. Similar to Eq. (25), the discretised form of Eqs (22) can be written in the form

$$\mathbf{A}_i\Delta\mathbf{u}_i = \mathbf{Res}_i \quad \text{for } i = 1, 2, 3, \tag{31}$$

where $\Delta\mathbf{u}_i = \mathbf{u}_i^{\tau+\Delta\tau} - \mathbf{u}_i^{\tau}$ and $\mathbf{Res}_i$ are the residual vectors. The LU-SGS procedure outlined above is then applied with $\omega = \mathbf{u}_i$, with $\mathbf{u}_i$ then replaced by the calculated ω^p. The speed-up in solution gained via the above will be assessed in Section 6.1.1.

4.3. Solution Method: Solid

For the majority of FSI problems, the solid domain contains far fewer equations to solve as compared to the fluid. As such, a simple Jacobi iterative procedure was selected for this work, which involves the pseudo-temporal discretisation of Eqs (5) and (18). Traction boundary conditions are realised numerically

by excluding external boundaries from the surface integral in (5) and adding in surface integrals of the applied tractions $\boldsymbol{\tau}$. Thus, the equation becomes

$$\frac{\partial}{\partial t}\int_{\mathcal{V}_0}\rho_0 v_i d\mathcal{V} = \int_{\mathcal{S}_{0\text{internal}}} P_{ij}n_j d\mathcal{S} + \int_{\mathcal{S}_{\text{boundary}}}\tau_i d\mathcal{S} + \int_{\mathcal{V}_0} Q_i d\mathcal{V} \equiv R_i(\mathbf{w}), \quad (32)$$

For pseudo-temporal discretisation a single-step procedure proposed in [62] was used:

$$\begin{aligned} w_i^{\tau+\Delta\tau} &= w_i^\tau + \Delta\tau\left[v_i^\tau - \tfrac{3w_i^\tau - 4w_i^t + w_i^{t-\Delta t}}{2\Delta t} + \tfrac{1}{2}\Delta\tau R_i(\overline{\mathbf{w}}^\tau)/(\rho_0^{\tau+\Delta\tau}V_0^{\tau+\Delta\tau})\right] \\ v_i^{\tau+\Delta\tau} &= v_i^\tau + \Delta\tau R_i(\overline{\mathbf{w}}^\tau)/(\rho_0^{\tau+\Delta\tau}V_0^{\tau+\Delta\tau}) \end{aligned} \quad (33)$$

where V_0 is the volume of dual-cell $\mathcal{V}_0$ in the undeformed configuration and $\overline{w}^\tau$ denotes a projected displacement which is calculated as

$$\overline{w_i}^\tau = w_i^\tau + \Delta\tau v_i^\tau \quad (34)$$

where the nomenclature is as previously defined.

4.4. Pseudo-timestep Calculations

The pseudo-timestep local to each computational cell is to be determined in the interest of a stable solution process. For this purpose, the following expression is employed:

$$\Delta\tau = \text{CFL}\left[\frac{|u_i - u_i^*| + c_{\text{unified}}}{\Delta x_i} + \kappa(1-\beta)\frac{2\mu}{\rho\Delta x_i^2}\right]^{-1} \quad (35)$$

where CFL denotes the Courant-Friedrichs-Lewy number, Δx_i is the effective mesh spacing in direction i and κ is equal to 1 in the fluid domain and 0 in the solid domain. Note that if the velocity terms are being treated implicitly ($\beta = 1$), the viscous time-step size restriction falls away. Further,

$$c_{\text{unified}} = \kappa c_\tau + (1-\kappa)\left(\sqrt{K/\rho_0} + \sqrt{\eta/\rho_0}\right). \quad (36)$$

4.5. Solution Procedure

To achieve simultaneous solution of the discretised fluid-solid equations in a manner which effects strong coupling, the following solution sequence is employed in an iterative fashion:

1. The fluid and solid discrete equations are solved concurrently via (19)–(22) and (32). Due to the accelleration of the fluid solver with GMRES, convergence of the fluid domain is typically faster than the solid. Therefore, in order to achieve fast convergence of the coupled system, an adjustable number of iterations of Eq. (32) may be performed for every iteration of the fluid equations.

2. At the interface, the calculated fluid traction is applied to the solid boundary and the solid velocities to the fluid boundary. That is, the following equations for traction and velocity at the boundary are prescribed:

$$\begin{aligned} \tau_j &= p n_j - \sigma_{ij} n_i \\ \mathbf{u} &= \mathbf{v} \end{aligned} \tag{37}$$

   where $\mathbf{n}$ is the normal unit vector pointing outward from the fluid domain.
3. The mesh is only moved if a solid mesh boundary node displacement exceeds 30% of the attached dual-cell size or the residual of the fluid or solid mesh has been reduced by three orders of magnitude (a real-time-step is considered converged when the residual of all fluid and solid equations have dropped by at least 4 orders of magnitude); hence, the mesh is always moved at least once per timestep, and more if the residual is too large or there is significant displacement of the interface nodes during the convergence process. A mesh move involves redistribution of fluid nodes, followed by pre-processing of the mesh. The latter involves computation of edge-coefficients and finite volumes.
4. The residuals are now calculated for all equations, and if larger than the convergence tolerance steps 1–3 are repeated.
5. If the residuals are below the convergence tolerance, the real-timestep is terminated, and the next time-step entered by proceeding to step 1.

As far as fluid–structure interaction is concerned, the method above represents a block-Jacobi method, which suffers from slow convergence compared to block-Gauss-Seidel or block-Newton methods. However, in the current framework the coupling iterations are the same as the dual-timestepping pseudo-timesteps which are numerous (typically on the order of 100 per timestep), and so this does not represent a significant performance penalty. However, in order to ensure robustness, a more sophisticated coupling method (listed in Section 1; see [35] for an overview) should be incorporated. This is because block-Jacobi and block-Gauss–Seidel methods are known to suffer from instabilities due to the 'added mass effect' [63, 64, 65], which are most severe when fluid and structure have comparable densities.

5. Parallelisation

Because of the fully matrix-free nature of the numerical method at solver sub-iteration level, the mesh can be decomposed into separate subdomains for parallel operation. Firstly, in composing the right-hand side, all loops are over edges, with the operation count for each edge being very nearly identical. Secondly, in performing the various sparse-matrix–vector dot products that make up the preconditioned GMRES algorithm, there is one dot product per *node*; however, the number of nonzero entries in the corresponding row of the Jacobian matrix is equal to the number of edges surrounding the node, plus one. Therefore, to balance the operation count for efficient parallelisation of not only

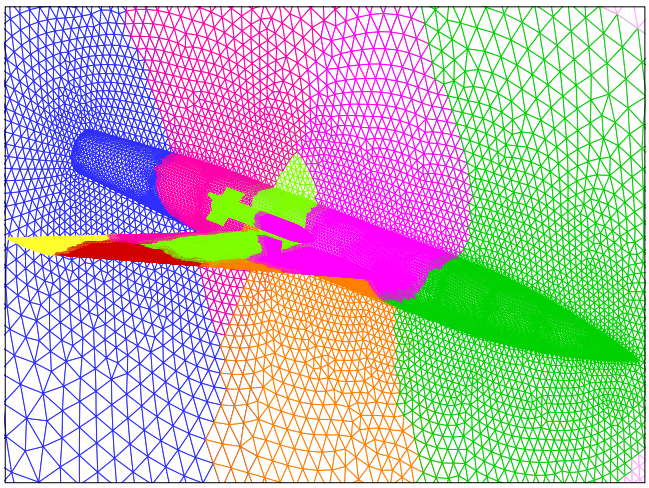

Figure 2: Example of domain decomposition.

the right-hand side calculation but also the LU-SGS/GMRES routine, the number of edges in each domain was balanced. This was done by weighting each node with an integer equal to the number of edges which connect to it, followed by applying the METIS library [66] to its connectivity graph. For interdomain communication, a system of "ghost nodes" is used in this work, with one layer of overlapping nodes at domain boundaries, where 'slave' nodes are updated with the values from corresponding 'master' nodes in the neighbouring domain. For efficiency, data transfer is consolidated into the largest possible packets and communicated using MPI. An example of the decomposition into subdomains is shown in Fig. 2, where individual domains are distinguished by colour.

The GMRES routine consists of sparse-matrix–vector products and dot products which are suited to parallel computing. However, the LU-SGS preconditioning steps are formally serial in nature. These involve sweeps across the mesh, with each subsequent nodal value calculated from the newly updated values of the preceding ones. While this process may be efficiently vectorised on shared-memory architectures [67], on distributed-memory machines excessive communication would be required during each sweep over the mesh, severely impairing performance. In this work we have elected to perform the LU-SGS preconditioning only within each parallel subdomain, thereby eliminating the need for communication altogether. Being merely a preconditioning step, this has no influence on the solution accuracy; neither was it found to impair stability. What does suffer somewhat is the speed of convergence, since information does not propagate across the entire mesh at each iteration to the same extent as the serial implementation. However, as demonstrated below, this effect was found to diminish as the number of parallel domains increases. When using LU-SGS for implicit relaxation of the velocities in the momentum equations,

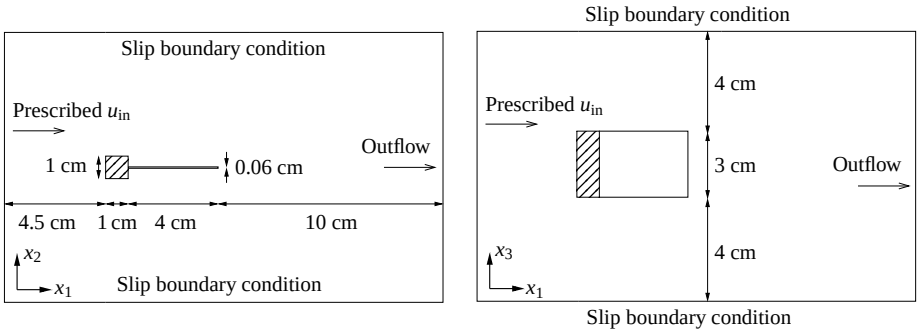

Figure 3: Block with flexible tail: geometry and boundary conditions. Top view (left) and side view (right).

similar arguments apply.

6. Application and Evaluation

6.1. Block with flapping plate

The first FSI problem we consider comprises a block connected to a flexible plate in cross-flow, as proposed by von Scheven and Ramm [29] and also considered by Kassiotis *et al.* [68] As shown in Fig. 3, it is a three-dimensional version of the popular block with flexible tail used to test 2D FSI algorithms [12, 22, 34, 42, 69].

The material properties are as follows: Incompressible fluid density is 1.18×10^{-3} g cm^{-3} and viscosity $\mu = 1.82 \times 10^{-4}$ g cm^{-1} s^{-1}. For the plate, two different material properties have been used. We shall refer to these as the 'stiff' and 'flexible' plate respectively. In the first place, to reproduce the case considered by other researchers [29, 68], we use a density $\rho_s = 2.0$ g cm^{-3}, and a Young's modulus $E = 2.0 \times 10^6$ g cm^{-1} s^{-2}. These are also the same material properties considered in the 2D analyses referenced above. Secondly, for the 'flexible' plate we reduce both the density and elastic modulus by a factor of 20, i.e. $\rho_s = 0.1$ g cm^{-3} and $E = 1 \times 10^5$ g cm^{-1} s^{-2}. This is to decrease the stiffness of the plate without affecting its characteristic frequencies, thereby increasing the three-dimensionality of the response. In both cases we use a Poisson's ratio $\nu = 0.35$.

Two meshes of different densities were employed (Fig. 4) in order to assess the mesh-sensitivity of the solution. The coarse mesh consists of circa 480 000 fluid cells and 8 000 solid cells (with 6 elements through the plate thickness). For the fine mesh the spacing was reduced by a factor of 1.5 in each direction, resulting in 1 580 000 fluid cells and 27 000 solid cells. As pointed out previously, multiple solid sub-iterations are run for every fluid mesh iteration. For the problem considered here, 30 solid sub-iterations for the coarse mesh and 50 sub-iterations for the fine mesh were found to result in similar fluid and solid drop in residual (measured between fluid-solid-boundary communication). The

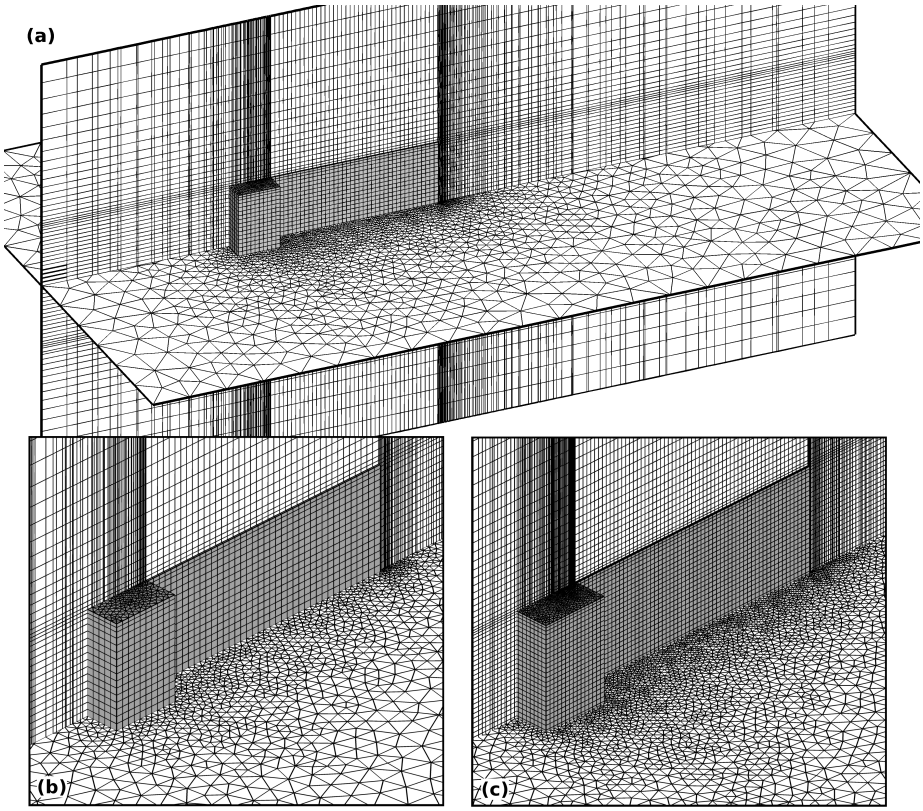

Figure 4: Meshes used for block with flexible plate. Coarse mesh [(a) and (b)] and fine mesh (c).

analysis was run on a Sun Microsystems Constellation cluster with 8-core Intel Nehalem 2.9 GHz processors and Infiniband interconnects at the Centre for High Performance Computing (CHPC), Cape Town. On the course mesh, the 7 500 timesteps completed in 6 days using 23 parallel fluid subdomains and 7 solid subdomains. The fine mesh analysis took 11 days with 64 fluid and 32 solid subdomains.

For the first analysis, with the stiffer plate, the inflow velocity was linearly ramped up to its final value of 100 cm s^{-1} within the first 2 s of the analysis as per [29]. The time-step size was chosen as $\Delta t = 0.004$ s. The calculated time-history of horizontal displacement of the top and bottom corners is depicted in the left panels of Fig. 5. Although there is a significant difference in the rate of growth between the coarse and fine meshes, both settle to similar limit cycles. The initial difference is to be expected since the initial state is an unstable equilibrium, the breaking down of which is initially dominated by discretisation errors. Since the limit-cycle oscillation is modulated by a lower frequency fluctuation, to obtain a quantitative comparison we consider the RMS amplitude

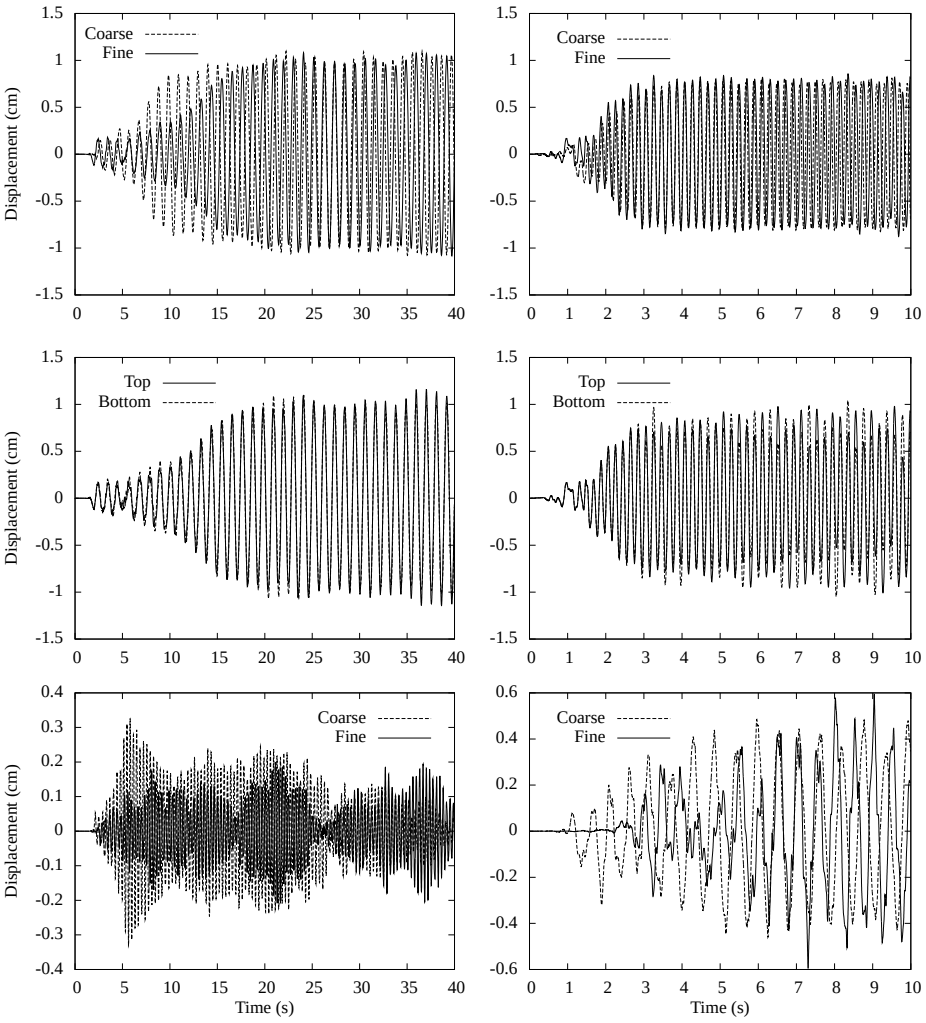

Figure 5: Results for block with flexible plate: x_2-displacement of plate end as a function of time. Analysis of stiff plate (left) and flexible plate (right). Upper panels: Mid point displacement ($x_1 = 4$, $x_2 = x_3 = 0$), comparing fine and coarse meshes. Middle panels: Fine mesh results with top and bottom corner displacement overlaid. Lower panels: Difference between top and bottom corner displacements for fine and coarse meshes, representing the twisting mode.

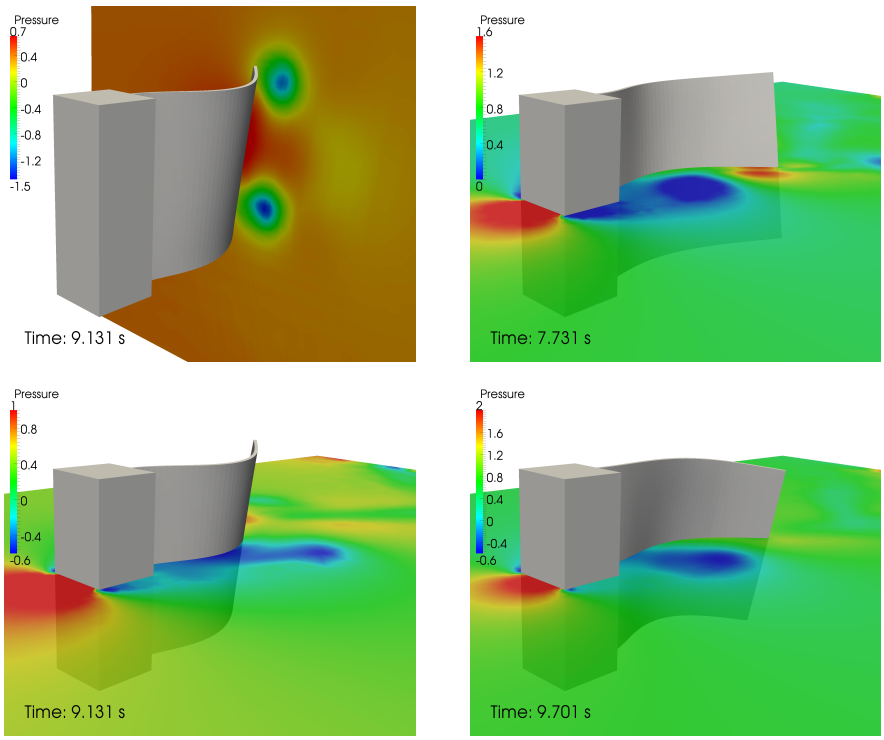

Figure 6: Solution snapshots of flexible plate case. Pressure contours on the plane $x_1 = 4$ (top left) and $x_3 = 0$ (others), given in Pa.

of the vertical-midpoint at the end of the plate ($x_1 = 4$, $x_2 = x_3 = 0$) over the last ten seconds of the simulation, which is 0.734 cm for the coarse mesh and 0.723 cm for the fine mesh; an agreement to within 1.5%. The average frequencies over this same range are 0.970 Hz for the coarse mesh and 0.932 Hz for the fine mesh, a difference of 3.9%. A high degree of mesh independence of the steady-state solution has thus been reached.

It is interesting to compare these results with the equivalent two-dimensional case [22, 55]. In that case, two different regimes of limit-cycle oscillation were observed, the first with an amplitude of approximately 2.0 cm and a frequency of approximately 0.8 Hz, and the second with an amplitude of 0.8 cm and frequency of 2.9 Hz, approximately. These correspond to the first and second mode of vibration of the flexible tail, and depend on initial conditions applied to the flexible tail. In the 3D case analysed here, vibration is in the first mode and the amplitude is almost half of the equivalent 2D case. This is attributable to the fact that pressure can equalise from one side of the plate to the other via flow across its lateral edges. The resulting circulation in the x_2–x_3 plane is clearly visible in the first panel of Fig. 6. The frequency, on the other hand, is of comparable magnitude, with the discrepancy attributable to the nonlinear

Table 1: Speedup in computation time for a representative timestep of the block-plate problem with two different mesh densities. Number of iterations to convergence are also shown. Overall speedup includes time taken on mesh movement and preprocessing, in contrast to solver speedup.

Mesh	Algorithm	Iterations	Overall speedup	Solver speedup
Coarse	Explicit	31016		
	Exp $\mathbf{u}$/Imp p	362	31.7 ×	38.1 ×
	Imp $\mathbf{u}$/Imp p	233	46.0 ×	65.3 ×
Fine	Explicit	49995		
	Exp $\mathbf{u}$/Imp p	1398	17.8 ×	20.0 ×
	Imp $\mathbf{u}$/Imp p	346	51.0 ×	67.6 ×

dependence of frequency on amplitude. In the 3D simulation in [68], on the other hand, the plate begins oscillating in its second mode, with a frequency close to 4 Hz and an amplitude of approximately 0.5 cm – also similar to the equivalent 2D results. The fact that we observe a different mode of vibration to that in [68] may be due to initial conditions. In that paper, the plate is kept rigid for the first two seconds of analysis while the flow velocity is ramped up. This may prevent the plate from receiving the perturbation necessary to excite first-mode vibrations. Also, the form of ramping is different in [68] to that used here. Both results are of much larger amplitude than those reported in [29].

In the second analysis, with the more flexible plate, an inflow velocity of 51.3 cm s^{-1} was used. A smaller time-step size of $\Delta t = 0.002$ s was selected to account for the faster, second-mode oscillations present in this case. Time-histories of corner displacements are shown in the right-hand panels of Fig. 5, and Fig. 6 shows snapshots of the response with pressure contour colouring. As seen, the higher-frequency second-mode of vibration is excited, with additionally a much larger component of the anti-symmetric twisting mode present. The vortices that drive the oscillations are visible in the right-hand panes of Fig. 6, while in the top-left pane, counter-rotating vortices coming off the corners are visible in the vertical plane. The coarse and fine mesh solutions again show good qualitative agreement once oscillations have grown to limit-cycle state. To quantify this we measure the RMS amplitude of the vertical-midpoint oscillation for $6 < t < 10$ s, obtaining a value of 0.579 cm for the coarse mesh and 0.586 cm for the fine mesh, which constitutes agreement to within 1.2%. For the coarse mesh, the average frequency over this range is 4.95 Hz while for the fine mesh it is 4.90 Hz, a difference of 1.0%.

6.1.1. LU-SGS/GMRES performance speedup

In this section, we analyse the performance improvement effected by the advanced solver algorithms applied to the pressure and momentum equations. Quoted values are the improvement in overall performance of the FSI solver,

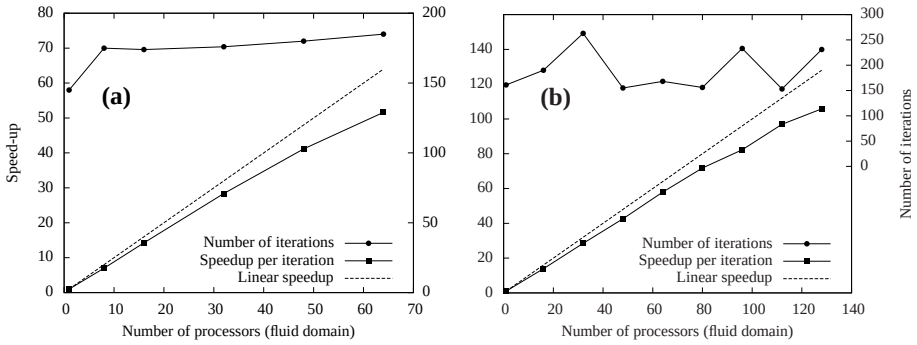

Figure 7: Parallelisation speed-up for the block-tail problem. (a) Coarse mesh (490 000 cells) and (b) fine mesh (1.6M cells). The right axis depicts the number of fluid iterations required per time-step.

even though the advanced solver is only applied to the fluid subdomain. The number of parallel subdomains used was 23 and 7 for the coarse fluid and solid meshes respectively, increased to 42 and 22 in the case of the fine mesh.

For the purpose of solver speed-up assessment, we have first initialised the block with flexible plate problem by running it for one second in order to reach a representative time-step, and then compared the time taken to reduce the residual at the subsequent time-step by five orders of magnitude. Two advanced solver variants were investigated. Firstly, preconditioned GMRES was applied to the pressure equation with Jacobi iterations used to converge the momentum equations (denoted 'Exp $\mathbf{u}$/Imp p' in Table 1). Six Krylov-space vectors were used with three GMRES iterations for each iteration in pseudo-time. The CFL number used was 0.8 for the momentum equations and 1×10^5 for the pressure equation. Secondly, we considered the momentum equations implicitly by setting $\beta = 1$ in Eq. (22) and adding one LU-SGS iteration for each pseudo-timestep ('Imp $\mathbf{u}$/Imp p'). This allowed the CFL number to be increased to 1.4 for the momentum equations. Both of these algorithms are compared to standard Jacobi used for both pressure and momentum equations ('Explicit'), where the CFL number was set to 0.8.

The achieved speed-ups are listed in Table 1. Overall solver speed-ups for the full implicit method were around 65 times. It is interesting to note that the speedup effected by the implicit treatment of the pressure in isolation is less for the fine mesh than for the coarse mesh, whereas with implicit velocity relaxation the speedup is very similar in the two cases. This is most likely due to the restriction in timestep size caused by the viscous terms, which is a quadratic function of mesh spacing. Thus, although the pressure equation is treated implicitly, for the finer mesh the viscous timestep restriction begins to become the bottleneck unless the viscous terms are treated implicitly as well.

Figure 8: Meshes used for pipe problem.

6.1.2. Parallel efficiency

The strong-scaling performance of the entire coupled solver was evaluated by assessing the time taken to complete the first two time-steps of the block with flexible plate problem. The results of the study are depicted in Fig. 7, where the number of iterations achieved per second has been normalised to the value for a single processor. Note that, as a consequence of the preconditioning being done block-for-block on each parallel subdomain, the number of iterations taken to converge the time-step is not constant, and is also plotted in Fig. 7. As seen, this does not show a clear deteriorating trend as one may expect. Also plotted is the speed-up per iteration. After the initial scaling from one to eight cores which is somewhat below linear due to shared-memory bandwidth saturation, the scaling with distributed memory is linear until communication begins to dominate over computation.

6.2. Pressure-pulse in flexible tube

The second test-case comprises a flexible tube 5 cm in length with inner diameter of 1 cm and wall thickness of 0.1 cm. This problem is associated with arterial flow and has been considered by many researchers [27, 70, 71, 72]. It constitutes a worthy test for an incompressible FSI solver as tube flexibility is paramount to the solution. However, a mesh independent solution has never been provided for comparison purposes. Here we aim to establish one, as the problem constitutes a simple and relatively inexpensive test for FSI codes. The walls have density $\rho_s = 1.2$ g cm^{-3} and elastic modulus $E = 3 \times 10^6$ g cm^{-1} s^{-2}. A Poisson's ratio of $\nu = 0.3$ is used. To model a prototypical liquid, the fluid inside has density 1 g cm^{-3} and a viscosity $\mu = 3 \times 10^{-2}$ g cm^{-1} s^{-1}. The tube wall is clamped at both ends, with a pressure boundary condition imposed on the fluid at either side. On the left side the pressure is set to 1.3332×10^4 Pa

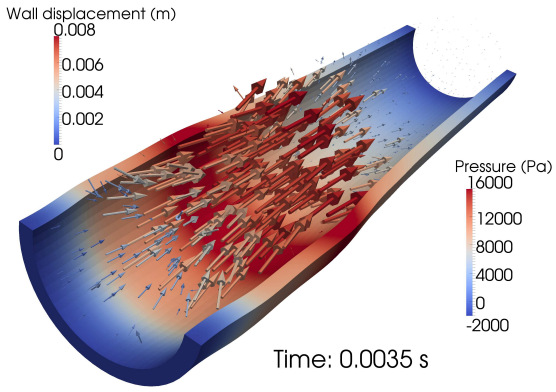

Figure 9: Solution snapshot at $t = 0.0035$ s. Solid deflection is exaggerated by a factor of 10. Velocity vectors are scaled according to velocity magnitude and coloured according to pressure.

for $t < 0.003$ s and zero thereafter, while on the right it is set to zero throughout the analysis. The result is that a pressure wave travels down the tube.

The three computational meshes used are depicted in Fig. 8 and denoted Coarse, Intermediate and Fine. In each case to obtain a finer mesh, the spacing was reduced by a factor of 1.5 in each direction, resulting in a roughly 3.4 times increase in the number of cells. The number of fluid cells is, respectively, 91 000, 301 000 and 1 million for the three meshes, with the solid domain containing 38 000, 130 000 and 454 000 cells respectively. A time-step size of 1×10^{-4} was used as in [27]. For this problem, the calculation of the redistribution of nodes for mesh movement has to be modified, with D_1 and D_2 in Eq. (15) taking on the meaning of shortest distance to the solid boundary and shortest distance to the centreline, $x = y = 0$, respectively.

Figure 9 shows a snapshot of the solution, with wall displacement exaggerated by a factor of 10 for clarity. Figure 10 depicts the radial components of displacement and velocity of the pipe wall at a point on the inner wall halfway along the pipe. It is evident that the fine mesh result is close to converged, with the maximum change in velocity between the coarse and intermediate meshes of 8.2% in peak velocity reducing to 3.4% between the intermediate and fine meshes. The coarse mesh analysis completed in 125 minutes using four Intel Core 2 2.66 GHz computational cores, while the finest mesh took 938 minutes using 31 AMD Opteron 800 MHz cores.

7. Conclusion

A fast parallel strongly-coupled partitioned FSI scheme was developed using a 3D edge-based finite volume methodology. Laminar incompressible flows interacting with structures undergoing large non-linear displacements were considered. The incompressible fluid divergence constraint was dealt with using

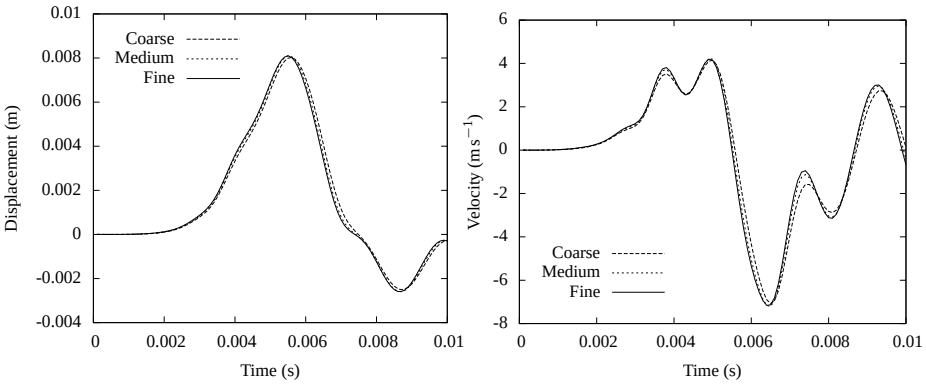

Figure 10: Radial component of dispacement (left) and velocity (right) of the inner tube wall at half the length of the pipe. Solutions are shown for the three mesh densities given in the text.

a new artificial-compressibility algorithm stabilised with Consistent Numerical Fluxes. The fluid solver was accelerated via the implicit treatment of pressures in the continuity equation and implicit treatment of viscous terms in the momentum equations. These were respectively solved via two matrix-free methods viz. LU-SGS preconditioned GMRES and LU-SGS. The solver was parallelised using METIS and MPI for distributed memory architectures. Two strongly-coupled test problems were considered, and solver performance assessed via mesh indepedance studies. A high degree of mesh indepedance was demonstrated throughout. Advanced solver computational performance demonstrated truly fast and efficient parallel solution. Overall speed-ups achieved were circa 50 while ensuring linear distributed parallel-scaling for up to 128 CPUs for the problems considered.

Acknowledgements

The authors wish to thank the Centre for High Performance Computing (CHPC) for access to computing hardware. This work was funded by the Council for Scientific and Industrial Research (CSIR) on Thematic Type A Grant nr. TA-2009-013.

References

[1] R. M. Bennet, J. W. Edwards, An overview of recent developments in computational aeroelasticity, in: Proceedings of the 29th AIAA fluid dynamics conference, Albuquerque, NM, 1998.

[2] C. Farhat, K. G. van der Zee, P. Geuzaine, Provably second-order time-accurate loosely-coupled solution algorithms for transient nonlinear computational aeroelasticity, Computer Methods in Applied Mechanics and Engineering 195 (2006) 1973–2001.

[3] L. Cavagna, G. Quaranta, P. Mantegazza, Application of Navier–Stokes simulations for aeroelastic stability assessment in transonic regime, Computers & Structures 85 (2007) 818–832.

[4] K. J. Bathe, H. Zhang, S. Ji, Finite element analysis of fluid flows fully coupled with structural interactions, Computers & Structures 72 (1-3) (1999) 1–16, doi:10.1016/S0045-7949(99)00042-5.

[5] P. L. Tallec, J. Mouro, Fluid structure interaction with large structural displacements, Computer Methods in Applied Mechanics and Engineering 190 (2001) 3039–3067.

[6] K. Stein, R. Benney, V. Kalro, T. E. Tezduyar, J. Leonard, M. Accorsi, Parachute fluid–structure interactions: 3-D computation, Computer Methods in Applied Mechanics and Engineering 190 (2000) 373–386.

[7] T. E. Tezduyar, Y. Osawa, Fluid–structure interactions of a parachute crossing the far wake of an aircraft, Computer Methods in Applied Mechanics and Engineering 191 (2001) 717–726.

[8] E. Simiu, R. H. Scanlan, Wind effects on structures. Fundamentals and applications to design, John Wiley & Sons, New York, 1996.

[9] R. Wüchner, A. Kupzok, K.-U. Bletzinger, A framework for stabilized partitioned analysis of thin membrane-wind interaction, International Journal for Numerical Methods in Fluids 54 (2007) 6–8.

[10] R. van Loon, F. N. van de Vosse, Special Issue: Fluid–structure interaction in biomedical applications, International Journal for Numerical Methods in Biomedical Engineering 26 (3-4) (2010) 273–275, doi:10.1002/cnm.1371.

[11] C. Taylor, C. Figueroa, Patient-Specific Modeling of Cardiovascular Mechanics, Annual Review of Biomedical Engineering 11 (1) (2009) 109–134, doi:10.1146/annurev.bioeng.10.061807.160521.

[12] W. A. Wall, E. Ramm, Fluid–structure interaction based upon a stabilized (ALE) finite element method, in: S. Idelsohn, E. Oñate, E. Dvorkin (Eds.), Computational mechanics – new trends and applications, Proceedings of WCCM IV, CIMNE, Barcelona, 1998.

[13] E. H. Dowell, K. C. Hall, Modeling of fluid–structure interaction, Annual Review of Fluid Mechanics 33 (1) (2001) 445–490, doi: 10.1146/annurev.fluid.33.1.445.

[14] R. Ohayon, C. E. Felippa, Special Issue: Advances in Computational Methods for Fluid–Structure Interaction and Coupled Problems, Computer Methods in Applied Mechanics and Engineering 190 (2001) 2977–3292, doi: 10.1016/S0045-7825(00)00376-5.

[15] T. Tezduyar, Y. Bazilevs, Special issue on fluid–structure interaction, Computational Mechanics 43 (2008) 1–189, doi:10.1007/s00466-008-0317-8.

[16] A. G. Malan, R. W. Lewis, P. Nithiarasu, An Improved Unsteady, Unstructured, Artificial Compressibility, Finite Volume Scheme for Viscous Incompressible Flows: Part I. Theory and Implementation, International Journal for Numerical Methods in Engineering 54 (5) (2002) 695–714.

[17] A. G. Malan, R. W. Lewis, P. Nithiarasu, An Improved Unsteady, Unstructured, Artificial Compressibility, Finite Volume Scheme for Viscous Incompressible Flows: Part II. Application, International Journal for Numerical Methods in Engineering 54 (5) (2002) 715–729.

[18] A. G. Malan, R. W. Lewis, Modeling Coupled Heat and Mass Transfer in Drying Non-Hygroscopic Capillary Particulate Materials, Communications in Numerical Methods in Engineering 19 (9) (2003) 669–677.

[19] J. Pattinson, A. G. Malan, J. P. Meyer, A Cut-cell non-conforming Cartesian Mesh Method for Compressible and Incompressible Flow, International Journal for Numerical Methods in Engineering 72 (11) (2007) 1332–1354.

[20] A. G. Malan, J. P. Meyer, R. W. Lewis, Modelling Non-Linear Heat Conduction via a Fast Matrix-Free Implicit Unstructured-Hybrid Algorithm, Computer Methods in Applied Mechanics and Engineering 196 (45-48) (2007) 4495–4504.

[21] A. G. Malan, R. W. Lewis, An artificial compressibility CBS method for modelling heat transfer and fluid flow in heterogeneous porous materials, International Journal for Numerical Methods in Engineering 87 (1–5) (2011) 412–423, doi:10.1002/nme.3125.

[22] B. Hübner, E. Walhorn, D. Dinkler, A monolithic approach to fluid–structure interaction using spacetime finite elements, Computer Methods in Applied Mechanics and Engineering 193 (2004) 2087–2104.

[23] C. J. Greenshields, H. G. Weller, A unified formulation for continuum mechanics applied to fluid-structure interaction in flexible tubes, International Journal for Numerical Methods in Engineering 64 (2005) 1575–1593.

[24] M. W. Gee, U. Küttler, W. A. Wall, Truly monolithic algebraic multigrid for fluid–structure interaction, International Journal for Numerical Methods in Engineering 85 (8) (2011) 987–1016, doi:10.1002/nme.3001.

[25] D. P. Mok, W. A. Wall, Partitioned analysis schemes for the transient interaction of incompressible flows and nonlinear flexible structures, in: W. A. Wall, K.-U. Bletzinger (Eds.), Trends in computational structural mechanics, Barcelona: CIMNE, 689–698, 2011.

[26] W. A. Wall, S. Genkinger, E. Ramm, A strong coupling partitioned approach for fluid–structure interaction with free surfaces, Computers & Fluids 36 (2007) 169–183.

[27] U. Küttler, W. A. Wall, Fixed-point fluid–structure interaction solvers with dynamic relaxation, Computational Mechanics 43 (2008) 61–72.

[28] C. Kassiotis, A. Ibrahimbegovic, R. Niekamp, H. G. Matthies, Nonlinear fluid–structure interaction problem. Part I: implicit partitioned algorithm, nonlinear stability proof and validation examples, Computational Mechanics 47 (2011) 305–323.

[29] M. von Scheven, E. Ramm, Strong coupling schemes for interaction of thin-walled structures and incompressible flows, International Journal for Numerical Methods in Engineering 87 (2011) 214–231.

[30] H. G. Matthies, J. Steindorf, Partitioned but strongly coupled iteration schemes for nonlinear fluid–structure interaction, Computers & Structures 80 (2002) 1991–1999.

[31] H. G. Matthies, J. Steindorf, Partitioned strong coupling algorithms for fluid–structure interaction, Computers & Structures 81 (2003) 805–812.

[32] T. E. Tezduyar, S. Sathe, R. Keedy, K. Stein, Spacetime finite element techniques for computation of fluidstructure interactions, International Journal for Numerical Methods in Fluids 195 (2006) 2022–2027.

[33] T. E. Tezduyar, S. Sathe, K. Stein, Solution techniques for the fully discretized equations in computation of fluidstructure interactions with the spacetime formulations, Computer Methods in Applied Mechanics and Engineering 195 (2006) 5743–5753.

[34] W. Dettmer, D. Perić, A computational framework for fluid–structure interaction: Finite element formulation and application, Computer Methods in Applied Mechanics and Engineering 195 (2006) 5754–79.

[35] W. G. Dettmer, D. Perić, A Fully Implicit Computational Strategy for Strongly Coupled Fluid–Solid Interaction, Archives of Computational Methods in Engineering 14 (3) (2007) 205–247.

[36] J. Vierendeels, L. Lanoye, J. Degroote, P. Verdonck, Implicit coupling of partitioned fluid–structure interaction problems with reduced order models, Mathematical Modelling and Numerical Analysis 85 (2003) 970–976.

[37] A. Jameson, Time dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings, AIAA Paper 91-1596.

[38] V. Venkatakrishnan, D. J. Mavriplis, Implicit Method for the Computation of Unsteady Flows on Unstructured Grids, Journal of Computational Physics 127 (1996) 380–397.

[39] P. I. Crumpton, P. Moinier, M. B. Giles, An Unstructured Algorithm for High Reynolds Number Flows on Highly Stretched Meshes, in: C. Taylor, J. T. Cross (Eds.), Numerical Methods in Laminar and Turbulent Flow, Pineridge Press, 561–572, 1997.

[40] Y. Zhao, B. Zhang, A High-Order Characteristics Upwind FV Method for Incompressible Flow and Heat Transfer Simulation on Unstructured Grids, Computer Methods in Applied Mechanics and Engineering 190 (2000) 733–756.

[41] R. Löhner, Applied CFD Techniques, John-Wiley and Sons Ltd., Chichester, 2001.

[42] G. Xia, C.-L. Lin, An unstructured finite volume approach for structural dynamics in response to fluid motions, Computers & Structures 86 (2008) 684–701.

[43] R. Suliman, O. Oxtoby, A. G. Malan, S. Kok, An enhanced matix-free edge-based finite volume approach to model structures, in: 7th South African Conference on Computational and Applied Mechanics (SACAM10), Pretoria, 399–406, 10-13 January 2010.

[44] S. V. Patankar, Numerical Heat Transfer and Fluid Flow, McGraw-Hill, New York, 1980.

[45] A. J. Chorin, A Numerical Method for Solving Incompressible Viscous Flow Problems, Journal of Computational Physics 2 (1967) 12–26.

[46] P. Nithiarasu, An efficient artificial compressibility (AC) scheme based on the characteristic based split (CBS) method for incompressible flow, International Journal for Numerical Methods in Engineering 56 (13) (2003) 1815–1845.

[47] R. Löhner, A fast finite element solver for incompressible flows, AIAA Paper 90-0398 .

[48] R. Löhner, C. Yang, J. Cebral, F. Camelli, O. Soto, J. Waltz, Improving the speed and accuracy of projection-type incompressible flow solvers, Computer Methods in Applied Mechanics and Engineering 195 (2006) 3087–3109.

[49] P. Nithiarasu, C.-B. Liu, An artificial compressibility based characteristic based split (CBS) scheme for steady and unsteady turbulent incompressible flows, Computer Methods in Applied Mechanics and Engineering 195 (2006) 2961–2982.

[50] N. Massarotti, F. Arpino, R. W. Lewis, P. Nithiarasu, Explicit and semi-implicit CBS procedures for incompressible viscous flows, International Journal for Numerical Methods in Engineering 66 (2006) 1618–1640.

[51] H. Luo, J. D. Baum, R. Löhner, A Fast, Matrix-Free Implicit Method for Compressible Flows on Unstructured Grids, Journal of Computational Physics 146 (1998) 664–690.

[52] P. D. Thomas, C. K. Lombard, Geometric conservation law and its application to flow computations on moving grids, AIAA Journal 17 (1979) 1030–1037.

[53] M. Lesoinne, C. Farhat, Geometric conservation laws for flow problems with moving boundaries and deformable meshes, and their impact on aeroelastic computations, Computer Methods in Applied Mechanics and Engineering 134 (1996) 71–90.

[54] K. A. Sørensen, O. Hassan, K. Morgan, N. P. Weatherill, Agglomerated Multigrid on Hybrid Unstructured Meshes for Compressible Flow, International Journal for Numerical Methods in Fluids 40 (3-4) (2002) 593–603.

[55] O. F. Oxtoby, A. G. Malan, A matrix-free, implicit, incompressible fractional-step algorithm for fluid–structure interaction applications, Journal of Computational Physics 231 (2012) 5389–5405.

[56] B. van Leer, Towards the Ultimate Conservative Difference Scheme IV: A New Approach to Numerical Convection, Journal of Computational Physics 23 (1977) 276.

[57] Y. Saad, M. H. Schultz, GMRES: A Generalized Mimimal Residual Algorithm for Solving Nonsymmetric Linear Systems, SIAM Journal on Scientific and Statistical Computing 7 (3) (1986) 856–869.

[58] A. Jameson, S. Yoon, Lower-upper Implicit Schemes with Multiple Grids for the Euler Equations, AIAA Journal 25 (7).

[59] I. Men'shov, Y. Nakamura, Implementation of the LU-SGS Method for an Arbritrary Finite Volume Discretization, in: 9th Japanese Symposium on CFD, Chuo University, Tokyo, Japan, 123–124, 1995.

[60] M. Soetrisno, S. T. Imlay, D. W. Roberts, A Zonal Implicit Procedure for Hybrid Structured-Unstructured Grids, AIAA Paper 94-0645.

[61] R. Löhner, C. Yang, E. Oñate, On The Simulation of Flows With Violent Free Surface Motion, Computer Methods in Applied Mechanics Engineering 195 (2006) 5597–5620.

[62] O. C. Zienkiewicz, R. L. Taylor, The Finite Element Method: Volume 1 – The Basis, Butterworth-Heinemann, Oxford, fifth edn., 2000.

[63] M. M. Joosten, W. G. Dettmer, D. Perić, Analysis of the Block Gauss–Seidel Solution Procedure for a Strongly Coupled Model Problem with Reference to Fluid-Structure Interaction, International Journal for Numerical Methods in Engineering 78 (2009) 757–778.

[64] P. Causin, J. F. Gerbeau, F. Nobile, Added-mass effect in the design of partitioned algorithms for fluid-structure problems, Computer Methods in Applied Mechanics and Engineering 194 (2005) 4506–4527.

[65] C. Förster, W. A. Wall, E. Ramm, Artificial added mass instabilities in sequential staggered coupling of nonlinear structures and incompressible viscous flows, Computer Methods in Applied Mechanics and Engineering 196 (2007) 1278–1293.

[66] G. Karypis, V. Kumar, A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs, SIAM Journal on Scientific Computing 20 (1) (1999) 359–392.

[67] D. Sharov, K. Nakahashi, Reordering of Hybrid Unstructured Grids for Lower–Upper Symmetric Gauss–Seidel Computations, AIAA Journal 36 (3) (1997) 484–486.

[68] C. Kassiotis, A. Ibrahimbegovic, R. Niekamp, H. G. Matthies, Nonlinear fluid–structure interaction problem. Part II: space discretization, implementation aspects, nested parallelization and application examples, Computational Mechanics 47 (2011) 335–357.

[69] P. R. F. Teixeira, A. M. Awruch, Numerical simulation of fluid–structure interaction using finite element method, Applied Mathematical Modelling 34 (2005) 249–273.

[70] L. Formaggia, J. F. Gerbeau, F. Nobile, A. Quarteroni, On the coupling of 3D and 1D Navier–Stokes equations for flow problems in compliant vessels, Comput. Methods Appl. Mech. Engrg. 191 (2001) 561–582.

[71] J.-F. Gerbeau, M. Vidrascu, A quasi-Newton algorithm based on a reduced model for fluid–structure interaction problems in blood flows, Mathematical Modelling and Numerical Analysis 37 (4) (2003) 631–647, doi: 10.1051/m2an:2003049.

[72] M. Á. Fernández, M. Moubachir, A Newton method using exact Jacobians for solving fluid–structure coupling, Computers & Structures 83 (2–3) (2005) 127–142, doi:10.1016/j.compstruc.2004.04.021.