ILLINOIS ROCSTAR

Physics of Coupling

Masoud Safdari
Jessica E. Kress
Michael J. Anderson
Woohyun Kim
Neil Sarwal

OpenMultiphysics

DUNS: xxxx
Illinois Rocstar LLC
1800 South Oak Street, Suite 108
Champaign, IL 61820

This work was funded in whole or in part by DOE Identification Number DE-SC0009596. The U.S. Government has for itself and others acting on its behalf a paid-up, nonexclusive, irrevocable worldwide license to use, modify, reproduce, release, perform, display, or disclose the work by or on behalf of the U.S. Government.

U.S. Department of Energy
Washington DC

Acknowledgments

Illinois Rocstar LLC would like to thank the Department of Energy and the SBIR Program for supporting and funding this work. We would also like to thank Aerojet-Rocketdyne, AMAX, Boeing, Caterpillar, Kitware, NCSA, Orbital-ATK, Sandia National Laboratory, and the Computational Science and Engineering program at the University of Illinois for their support, participation, and invaluable information.

Contents

1 Physics of Coupling in ElmerFoamFSI

1.1 Fluid-Solid Interaction (FSI)

Fluid-solid interaction (FSI) is a multi-physical phenomenon in mechanics that involves non-linear interactions between a fluid and solid. Many physical problems involve FSI phenomenon, example can be taken from different time/space scales and from a variety of systems for example deformations of the blood vessels during blood circulation and the fluctuation of airplane wings during flight both involve FSI. In every FSI problem, a moving fluid (compressible like air, or incompressible like water) interacts with a deformable solid. These interactions usually involve energy and momentum transfer between the phases.

Physical balance laws can be resorted to explain an FSI problem, including conservation of mass, energy and momentum. In some simple configurations these laws can be used to directly develop an analytical solution for an FSI problem. For more complex problems (most of engineering problems in 2D/3D) an analytical solution, if not impossible, is very hard to obtain. Laboratory experiments provide a limited scope of the problem as well and the understanding of the fundamental physics involved in complex interactions between the phases can only be obtained by numerical simulations.

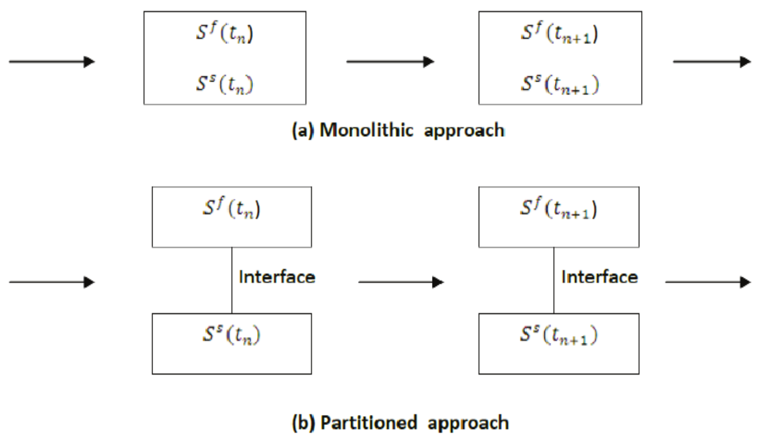
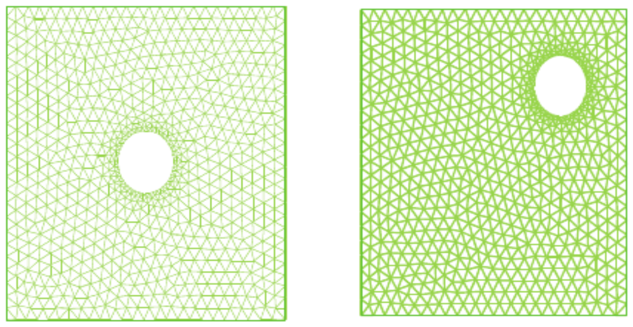


Figure 1: Different types of numerical solution strategies for FSI problems (Figure taken from Hou et al. (2012)).

Based on Hou et al. (2012), numerical treatment of FSI problems can be broadly classified into two major approaches: the *monolithic* approach and *partitioned* approach, shown in Figure 1. In the monolithic approach the interaction between solid and fluid at their interface is treated simultaneously, using a unified mathematical model which leads to a single system of equations following a unified discretization strategy. This approach may result a better accuracy for strong fluid-solid interactions, but it is computationally more expensive. In the partitioned approach, two separate mathematical models are used to describe fluid and solid domains. These two models are used to solve for fluid and solid separately, usually with two different discretization strategies, and they communicate through the common interface. The advantage of this approach is the possibility of using two separate disciplinary solution strategies, possibly two different legacy software to reduce the code development effort substantially. The challenge, however, for this approach is to integrate the two solvers and coordinate them to obtain an accurate and efficient solution with minimal code modification for each side. A better comparison between the monolithic and partitioned approach is provided by Michler et al. (2004).


(a) Conforming mesh. Left: $t = t_1$; Right: $t = t_2$.

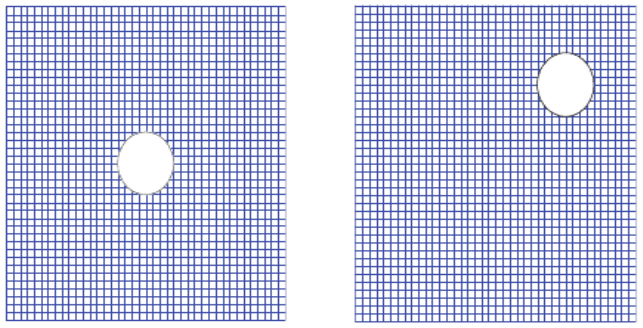

(a) Non-conforming mesh. Left: $t = t_1$; Right: $t = t_2$.

Figure 2: Conformal and non-conformal meshing strategies used in numerical solution of FSI problems (Figure taken from Hou et al. (2012)).

The position of interface between fluid and solid phases is a part of solution in an FSI problem. With regards to moving fluid-solid interface, two different strategies can be used: *conforming* mesh and *non-conforming* mesh, shown in Figure 2. In the more traditional conformal mesh approach, the position and motion of the solid-fluid interface is captured sharply. Because of the movements of common interface, this method requires continuous re-meshing (or mesh updating) during the solution which can be computational expensive and error-prone. In the non-conformal meshes, also known as immersed-methods, the position of the interface is captured by applying some constraints to each solver to avoid the need for a sharp disjoint mesh and the mesh updates. This method is currently constitutes major research efforts in the computational FSI community.

1.2 Partitioned Approach: Strong vs. Weak Coupling

Following partitioned approach strategy, in order to solve for the coupled unknown structural and fluid quantities, two different algorithms can be applied. Figure 3-a illustrates the more traditional weak coupling (also known as one-way coupling) algorithm. In this algorithm for each time step, the solution for the tractions obtained by the fluid solver is passed to the solid solver to find the new deformations. The output of the solid solver then will be used as an input to the fluid solver for the next time step. This approach is more efficient, but its application is limitted to FSI problems with weak fluid-solid interactions. For problems in which strong fluid-solid interactions are present, strong (two-way) coupling algorithm should be used. Figure 3-b illustrates the outline of this algorithm with more details. The major difference for the strong-coupling algorithm is the application of an inner-loop within the external time stepping loop that helps to capture a converged fluid-solid interface solution in each time step. This method is computationally more expensive, but it guarantees a more accurate solution for problems in which strong fluid-solid interactions is expected.

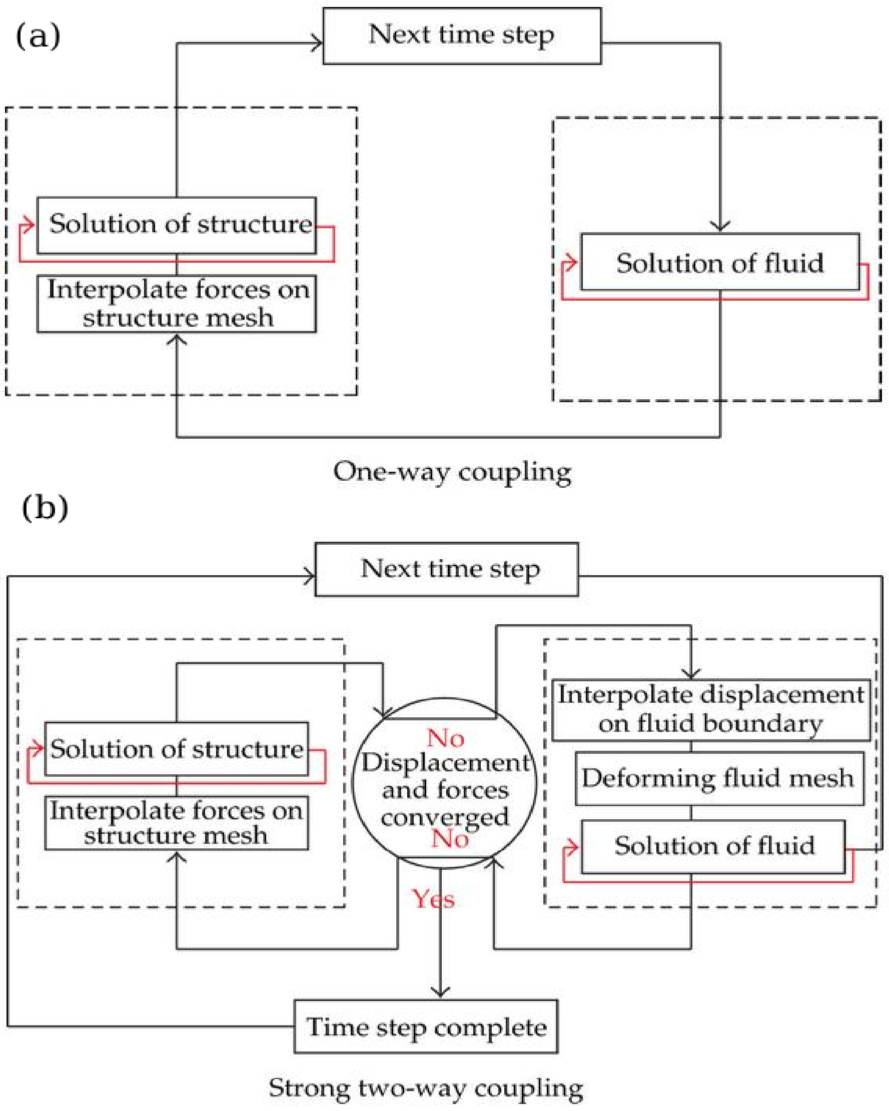


Figure 3: Strong and weak coupling strategies used in numerical solution of FSI problems (Figure taken from Benra et al. (2011)).

2 ElmerFoamFSI: A Partitioned Approach

2.1 Overview

ElmerFoamFSI uses a partitioned approach to solve FSI problems. In *ElmerFoamFSI*, *OpenFOAM* (a finite-volume solver [*]) is used as the fluid solver and *Elmer* (a finite-element solver [†]) is used as the structural solver. In *ElmerFoamFSI* both weak and strong coupling approaches can be used to solve FSI problems. Currently a weak-coupling algorithm is implemented into *ElmerFoamFSI*. In the rest of this document, we briefly explain how to setup and use *ElmerFoamFSI* to solve a FSI problem.

2.2 *Elmer* Input

Elmer is an extensive open-source finite element multiphysics code. This software can be used to find deformation of a solid object (U) subject to different types of loads. *ElmerFoamFSI* uses *Elmer* as its structural solver. At least two different structural solvers (linear and non-linear elasticity) are provided by *Elmer*. To capture large deformations of the solid, *ElmerFoamFSI* uses the non-linear solver. The description of a finite element problem usually starts with input files. The full description of the problem, including: geometry, materials, boundaries conditions, type of equations to solve, etc., is usually provided in the input file(s).

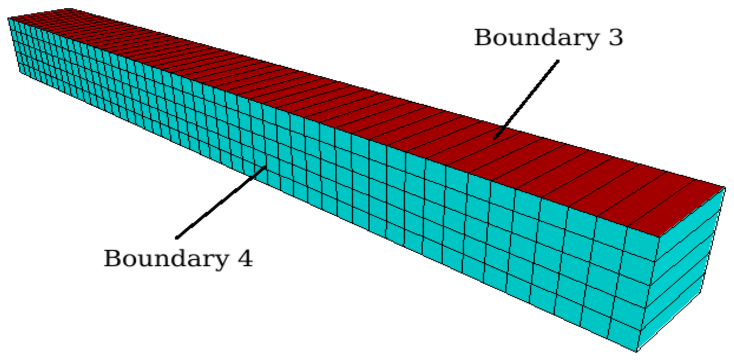


Figure 4: Elmer mesh and boundary 's.

Elmer uses a multiple-part input definition strategy. The description of the geometry of the problem in *Elmer* is provided in a GRD file (*".grd"* file), and the rest of the definitions is described in a SIF file (*".sif"* file). If the *ELMERSOLVER_STARTINFO* is used, upon execution *Elmer* scans this file to figure out which input file should be read (the name of the input file). Elmer uses a separate program (*ElmerGrid*) to read the GRD file and compile geometry description to a format useful for its internal usage. A proper description of problem geometry (also called problem mesh) requires a good level of understanding of finite element method, therefore it is beyond the purpose of the current document [‡]. The SIF file uses a *Keyword*-pair format to describe the rest of the problem. Each keyword can be thought of as a command to the *Elmer* solver which describes certain feature

[*] `http://www.extend-project.de/`

[†] `https://www.csc.fi/web/elmer/`

[‡] `http://www.nic.funet.fi/pub/sci/physics/elmer/doc/ElmerGridManual.pdf`

of the problem. Many of these commands require some parameters to be passed by user. These parameters are defined in front of the command after the "=" symbol. The first section of SIF file defines general settings of the problem as well as the location for geometry file, where to save simulation results, the type of simulation, time step used etc. This definition of this section of the SIF file is very straight forward as shown in the following :

```
Header
  CHECK KEYWORDS Warn
  Mesh DB "." "."
  Include Path ""
  Results Directory ""
End
Simulation
  Max Output Level = 5
  Coordinate System = Cartesian
  Coordinate Mapping(3) = 1 2 3
  Simulation Type = Transient
  Timestep intervals = 100
  Timestep Size = 1e-3
  Output Intervals = 1
  Timestepping Method = BDF
  BDF Order = 2
  Solver Input File = case.sif
  Post File = case.vtu
End
Constants
  Gravity(4) = 0 -1 0 9.82
  Stefan Boltzmann = 5.67e-08
  Permittivity of Vacuum = 8.8542e-12
  Boltzmann Constant = 1.3807e-23
  Unit Charge = 1.602e-19
End
```

In the next section, the types of equations to be solved and material properties for each constituent is described. Depending on what type of solver needed, a series of solver-related parameters should be passed to the *Elmer*. [§]. The following example defines a non-linear elasticity solver for *Elmer* along with the type of equation solver and non-linear solution strategies that should be used for the problem.

```
! Which equation to solve and which materials
Body 1
  Target Bodies(1) = 1
  Name = "Body 1"
  Equation = 1
  Material = 1
End
```

[§]http://www.nic.funet.fi/pub/sci/physics/elmer/doc/ElmerSolverManual.pdf

```
! Solver to use for equation 1
Equation 1
  Name = "Elasticity"
  Calculate Stresses = True
  Active Solvers(1) = 1
End
! Solver 1 definition and the solution strategies
Solver 1
  Exec Solver = Always
  Equation = Nonlinear elasticity
  Variable = Displacement
  Variable Dofs = 3
  Procedure = "ElasticSolve" "ElasticSolver"
  Nonlinear System Convergence Tolerance = 1.0e-4
  Nonlinear System Max Iterations = 1
  Nonlinear System Newton After Iterations = 3
  Nonlinear System Newton After Tolerance = 1.0e-2
  Nonlinear System Relaxation Factor = 1.0
  Linear System Solver = Iterative
  Linear System Iterative Method = BiCGStab
  Linear System Max Iterations = 1000
  Linear System Convergence Tolerance = 1.0e-6
  Linear System Abort Not Converged = True
  Linear System Residual Output = 1
  Steady State Convergence Tolerance = 1.0e-5
  Linear System Preconditioning = Diagonal
  Time Derivative Order = 2
End
! Definition of the material 1
Material 1
  Name = "blahBlah"
  Youngs modulus = 1.4e6
  Density = 10
  Poisson ratio = 0.4
End
```

The rest of SIF file defines the boundary conditions for the problem. For a structural problem different types of boundary conditions can be used, including displacement and force (traction). The boundary condition definition starts with a *Target Boundaries(x) = y* statement that specifies how many boundaries are using this description (x: an integer) and which boundary in the geometry file should be used (y: separated by space if more than one, look at Figure 4). Finally we specify which direction our loads/displacement should be applied (here $1, 2, 3$ correspond to X, Y and Z directions in a Cartesian grid, respectively). Some of the boundary conditions require special treatments that are defined by some extra keywords. For example look at the boundary condition 4 that describes a FSI boundary. In this case, we tell *Elmer* to apply a variable force to the sections of the geometry that reside on this boundary. We also specify: 1) which direction loads should be applied (*direction 2* is *Force 2*), 2) they are variable in time, and 3) *Elmer* should resort to function *LoadYDirection* in *LoadFunctionLibrary* whenever it is needed. Care most be practiced with the proper formating. Further definition of the keywords can be found in the *Elmer* manuals.

```
Boundary Condition 1
  Target Boundaries(1) = 6
  Name = "Wall"
  Displacement 3 = 0
  Displacement 2 = 0
  Displacement 1 = 0
End
Boundary Condition 2
  Target Boundaries(1) = 2
  Name = "stabilizer1"
  Displacement 1 = 0
End
Boundary Condition 3
  Target Boundaries(1) = 4
  Name = "stabilizer2"
  Displacement 1 = 0
End
Boundary Condition 4
  Fsi BC = True
  Target Boundaries(1) = 3
  Name = "FSI"
  Force 2 = Variable Time
    Real Procedure "LoadFunctionLibrary" "LoadYDirection"
End
```

2.3 *OpenFOAM* Input

ElmerFoamFSI uses *OpenFOAM* as its fluid solver module to solve for the fluid pressure (p) and velocity (U). To capture the deformations of the solid/fluid interface, *OpenFOAM* has to also perform mesh update and re-meshing procedures in order to maintain the quality of the computational grid during the simulation (in a fluid problem, gird is equivalent to mesh for solid problems). *OpenFOAM* uses a finite-volume discretization scheme to obtain pressures and velocities and a Laplace solver to perform the re-meshing tasks. User has to specifies proper parameters for these solvers in the input files.

The definition of a problem in *OpenFOAM* is performed based on a multi-part input file strategy. For FSI problems, *OpenFOAM* requires both the definition of the solid and the fluid sections of the grid. In *ElmerFoamFSI*, however, the solid definition for *OpenFOAM* is disregarded and will be replaced with that of *Elmer*. Similarly, all other computations for the solid section will be performed solely by *Elmer*. To setup a fluid problem in *OpenFOAM*, we have to follow a strict folder/file hierarchy as *OpenFOAM* seeks the definition of a problem in multiple files residing in a set of subfolders. The input files in *OpenFOAM* are also called dictionary files. In each dictionary file, a set of keyword-value pairs specify proper settings for some part of the problem and solution procedure. There should be at-least three major sub-folders in the main fluid problem definition folder *./fluid/*. These sub-folders are *1)./fluid/0/, 2)./fluid/constant/ and 3)./fluid/system/.* In the rest we briefly describe each folder and its contents. Readers are strongly encouraged to resort to *OpenFOAM* documentation for more details.¶

In *OpenFOAM*, *./fluid/0/* contains initial and boundary conditions for the problem. In this folder, we specify initial and boundary conditions for the pressures, velocities and grid deformations in three separate files with the proper names. Similarly, *./fluid/system/* folder contains a series of input files to specify which solvers should be used and what solution strategies should be used for them. Finally, *.fluid/constant/* folder contains the description of the grid and material properties. In this folder, a sub-folder exists which contains all information about the problem geometry, specified in *blockMeshDict*. These geometric information will be read by *blockMesh*, an accessory of *OpenFOAM* that generates simple block-shaped grids from the geometric data defined in the dictionary. The overall structure of an example blockMesh file is as following:

¶valuable information can also be found in `http://www.cfd-online.com/`

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | foam-extend: Open Source CFD                    |
|  \\    /   O peration     | Version:     3.0                                |
|   \\  /    A nd           | Web:         http://www.extend-project.de       |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
convertToMeters 1;
vertices
(
    (0.1 0.1 0.0)     // 0
    (0.1 0.1 1.0)     // 1
    (0.1 1.0 1.0)     // 2
    (0.1 1.0 0.0)     // 3

    (0.0 0.1 0.0)     // 4
    (0.0 0.1 1.0)     // 5
    (0.0 1.0 1.0)     // 6
    (0.0 1.0 0.0)     // 7
);
blocks
(
    hex (0 1 2 3 4 5 6 7) (50 50 1) simpleGrading (1 1 1)
);
edges
(
);
patches
(
    wall left
    (
        (0 3 7 4)
    )
    wall right
    (
        (1 5 6 2)
    )
    wall fsiFace
    (
        (0 4 5 1)
    )
    wall top
    (
        (3 2 6 7)
    )
    empty front
    (
        (4 7 6 5)
    )
    empty back
    (
        (0 1 2 3)
    )
);
mergePatchPairs
(
);
// ************************************************************************* //
```

The structure and the commands used in this file is fairly easily to understand.[‖] After the *OpenFOAM* banner, at the top section of the file *blockMesh* accessory to be used and the format of the file is specified. In the *vertices* section, the coordinates for the vertices of the grid are specified (starting from 0, each line specifies a vertex coordinate). In the *block* section, the connectivity for the blocks of the mesh are specified. In the *patches* section, we specify boundaries of the domain and their connectivity with respect to the vertex indices.

2.4 *ElmerFoamFSI* Input

Currently, *ElmerFoamFSI* uses a very basic input file (named *"test.config"*). This file should reside in the *./fluid/* folder where the *ElmerFoamFSI* executable is called, and it should be passed to the executable as a command-line parameter. Following example shows the structure of this keyword-value pair file. In this example the name of fluid and structure solver modules that should be loaded are specified in the first two lines. The third line specifies the solution transfer module (in this case *SurfX* which a part of *IMPACT*). In the rest of the file the total analysis time and the time step are specified.

```
FluidSolver=OpenFoamFSI
SolidSolver=ElmerCSC
TransferService=SurfX
FinalTime=0.05
TimeStep=2.0e-3
```

[‖]for more details resort to `http://cfd.direct/openfoam/user-guide/`

2.5 *ElmerFoamFSI* Call Procedure

To execute *ElmerFoamFSI* simulation, *Elmer* input files and *ElmerFoamFSI* input file should be copied to the *./fluid/* folder of the *OpenFOAM* input hierarchy. Following directions specify steps to take to run a problem:

- Create a folder and copy *Allclean*, *Allrun* (scripts), */fluid/* and */solid/* folders into this folder, make sure to copy all sub-folders as well.
- Copy elmer SIF, GRD and ELMERSOLVER_STARTINFO files to */fluid/*.
- run ./Allclean and ./Allrun to clean-up and prepare input files for the *OpenFOAM*.
- Go to *./fluid/* and type *ElmerGrid 1 2 name* where "name" should be replaced with the name of the GRD file for *Elmer*.
- Run *ElmerFoamFSI* the FSI simulation by "*elmerformfsi name*" where *name* should be replaced with the name of *ElmerFoamFSI* input file.

References

Benra, F.-K., Dohmen, H. J., Pei, J., Schuster, S., and Wan, B. (2011). A comparison of one-way and two-way coupling methods for numerical analysis of fluid-structure interactions. *Journal of Applied Mathematics*, 2011:16.

Hou, G., Wang, J., and Layton, A. (2012). Numerical methods for fluid-structure interaction – a review. *Communications in Computational Physics*, 12:337–377.

Michler, C., Hulshoff, S., van Brummelen, E., and de Borst, R. (2004). A monolithic approach to fluid-structure interaction. *Computers and Fluids*, 33(5-6):839–848. Applied Mathematics for Industrial Flow Problems.