uFVM v1.5 - Quick Guide

CFD Group @ AUB

Computational Fluid Dynamics Lab — cfd@aub.edu.lb
Maroun Semaan Faculty of Engineering And Architecture

March 10, 2019

1 Preamble

Credits of developing this code goes to Dr. Marwan Darwish, a full-time instructor at the mechanical engineering department in the American University of Beirut, and Mr. Mhamad Mahdi Alloush a PhD candidate at the same department. We would like also to acknowledge all those who contributed to the code, by either instruction or implementation.

Contents

2 Introduction

2.1 Using The Guide

2.1.1 About The Guide

This quick guide provides an overview about uFVM code. It describes with brief details the structure of the code, the range of applicability and who may use it. The way through which a CFD case is prepared is then described and plenty of tutorials are accordingly provided.

2.1.2 Terms of Use

uFVM is an academic CFD tool made for learning purposes. It provides a package of libraries and algorithms that the user can comfortably follow up. Handling, distributing or modifying is fully permissible; the user has the full permission to add any piece of code or modify an existing one.

3 About the Code Development and Authors

3.1 What's uFVM?

The name of the code presents an abbreviation letters of the finite volume method (FVM) that the code is based on. The u at the beginning of the name points for a fluid flow. The code is developed in Matlab environment because it is assumed that the majority of interested people are familiar with this environment.

3.2 The Place and Time of Development

The code is developed in the computational mechanics lab at the American University of Beirut, Beirut, Lebanon. The development has started in 2003 and was built and updated gradually through years. Lots of versions were made each of them had a different structure but necessarily the same theoretical background.

3.3 Development

The code was thoroughly based on the book by [1] and other published references. It is a revolutionized version of the previous code uFVM v1.0. The code is a direct accomplishment of the CFD group at the American University of Beirut. The CFD group is a team of professors, graduate and undergraduate students. Their main objective is to build computational knowledge and work on plenty of related topics in both tracks, development and application.

The major contributor to the code is Professor Marwan Darwish, a CFD professor at AUB, and Mhamad Mahdi Alloush, a PhD candidate at AUB as well. The other contributors to the code are Master and PhD students who accomplished their theses and dissertations from the computational mechanics lab at the American University of Beirut.

3.4 CFD Group at AUB

The CFD group at AUB is a research group that includes a group of professors, graduate students and undergraduate students who undergo a wide range of studies and simulations related to computational fluid dynamics. Working with both development and applied numerical studies, the group have gained a great expertise and knowledge in the CFD domain, which is currently the widest as well as the most efficient fluid flow testing tool. The groups accomplishments, research topics and published work are posted on their website: `https://www.aub.edu.lb/msfea/research/Pages/cfd.aspx`.

3.5 Range of Applicability

uFVM works for incompressible and compressible single-phase fluid flows with any type of mesh (structured and unstructured). For consistency, the code doesnt include any geometry modeling or meshing capabilities. It accepts mesh files of OpenFOAM format. It is worth mentioning that uFVM is a solver which solves the conservation equations (transport equations) where the user is able to investigate any physical quantity of interest which is transported by means of a physical phenomenon like convection and diffusion. A transient treatment is also included. The domain usually assumes a fluid flow, however, the code may still apply to solid domains if the user seeks certain transport quantities in a solid, obviously, like the temperature distribution in a metal. It is not the purpose of this code to provide a CFD tool for conducting fluid flow simulations for heavy/complex applications. There are two issues to raise here. First, this code is made for those who are mainly learning CFD and/or interested in CFD code development. This code provides a very useful and helpful means for those people. Second, the user should necessarily realize that Matlab is a highly user friendly language; this makes it very convenient for learning issues much more than it is for conducting real life engineering applications. However, this friendly user specialty had an expense at the computational time; Matlab is slower than other lower level languages.

3.6 A Glance into uFVM's Discretization and Solution Methods

Only pressure-based methods are available in uFVM with SIMPLE method as the default scheme (Other algorithms in the SIMPLE-family are not implemented). The default convection scheme is the Upwind scheme, but however, second order upwind (Gauss linear) convection scheme is also there. Gradient computation is based on the cell-based first-order Gauss approximation. The ILU and SOR solvers are available along with an algebraic multi-grid (AMG) solver which utilizes a V-Cycle. Under-relaxation factors for any given transport equation are treated implicitly in the equations. All these solution methods and controls are presented in a more detailed framework in later parts of the manual.

4 Structure of the Code

The uFVM directories are distributed into sources, which are the routines that make up the code, and tutorials that include the test cases. The source code is available in the uFVM/src directory, which includes all of the main sources. uFVM/utilities contains all auxiliary functions and applications. A third directory 'uFVM/tutorials' exists which contain some tutorials to run the code. These tutorials are classified as basic, incompressible, compressible, and heatTransfer. Cases that are to be simulated are to be made in OpenFOAM format. OpenFOAM cases include 3 main directories: 0, system and constant. The 0 directory is where initial and boundary conditions are specified. The system directory includes the solution methods, the finite volume schemes and the time and write controls of the simulation. The constant directory includes the mesh files, the fluid properties (transport and/or thermophysical), gravity properties and turbulence properties. For further information, refer to the tutorials.

5 How to Use uFVM?

5.1 Main Entrance to the Code

The code runs from a main script, usually called cfdRun. The cfdRun file has to be located in an OpenFOAM case as mentioned earlier. It represents the case study or the problem definition. This is the only file of importance for the user. The main file contains a set of functions that build up the model. However, the user has to add the path of uFVM source files, by pressing the 'HOME' tab, then 'Set Path', and after that click 'Add with Subfolders', and choose the uFVM directory. The user may also add the path by typing the following command into the command window:

$$addpath(genpath(< uFVMPath >));$$

5.2 Highlighting the Main File

The incompressible elbow example will be presented here. The user should change the directory to ' /uFVM/tutorials/incompressible/elbow'. The corresponding run script ('cfdRun') is shown here:

```
%---------------------------------------------------------------------------
%
%  written by the CFD Group @ AUB, 2018
%  contact us at: cfd@aub.edu.lb
%===========================================================================
% Case Description:
%     In this test case a water flow in an elbow is
%     simulated
%---------------------------------------------------------------------------

% Initialize
cfdStartSession;

% Read OpenFOAM Files
cfdReadOpenFoamFiles;

% Define equations to be solved
cfdDefineMomentumEquation;
cfdDefineContinuityEquation;

% Run case
cfdRunCase;
```

The code proceeds by starting the session, reading OpenFOAM files, defining the intended transport equations, and running the simulation.

5.2.1 Start Session

The session starts by printing a header which defines the program. A global structure 'Region' is initialised at this point. This global variable is the data base which will contain all the simulation information.

5.2.2 Read OpenFOAM Files

All OpenFOAM files are read within this function, the mesh, the system files, and the initial/boundary conditions.

5.2.3 Define Transport Equations

The equations to be solved are defined here. If a fluid flow problem is intended (this is a typical case), the momentum and continuity equations are indispensable.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \tag{1}$$

$$\frac{\partial \rho \mathbf{v}}{\partial t} + \nabla \cdot (\rho \mathbf{v}\mathbf{v}) = \nabla \cdot \tau - \nabla p + \mathbf{B} \tag{2}$$

Energy equation can also be added if temperature distribution is intended. Other scalar equations can be included; this is to be highlighted later on.

5.2.4 Running the Code

Before running the code, it is a good practice to visualize the mesh first and witness its properties. typing into Matlab's command window the following command will plot the mesh of the correspoding case:

$$cfdPlotMesh;$$

The following mesh is plotted:

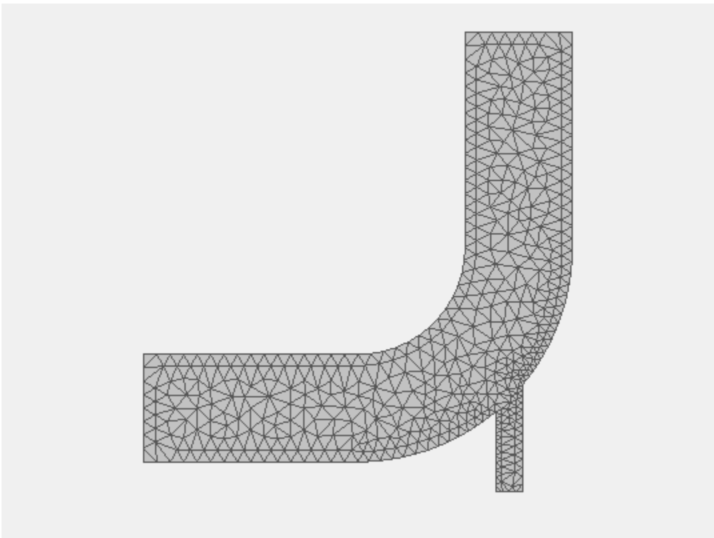

Figure 1: Elbow Mesh

Additionally, the user may want to print some useful mesh info like the number of elements, number of faces, number of boundary patches, etc. Typing the following will do that:

$$cfdCheckMesh;$$

The output in the command window is the following:

```
Mesh stats
    points:           1074
    faces:            3290
    internal faces:   1300
    cells:            918
    faces per cell:   5
    boundary patches: 6

Checking geometry...
    Overall domain cfdBounding box (0.00 -4.54 -0.94) (64.00 64.00 0.94)
    Boundary openness (-0.000000 0.000000 0.000000) OK.
    Max cell openness = 0.000000 OK.

End
```

Figure 2: Mesh Info

Now typing in the command window the command

$$cfdRun;$$

will run the code. The code gets executed here following up the previous steps (Initializing the data base, reading files, etc) and running the simulation by looping over time steps until convergence.

Running of the code will proceed, during which simulation information at each iteration will be printed as shown in the figure below:

```
|     U-y      |  7.544E-04 |  1.402E-02 |    1.896E+01    |   1.847E-06   |
|     U-z      |  2.509E-08 |  2.107E-07 |    1.187E-03    |   6.796E-10   |
|      p       |  4.930E-06 |  1.061E-04 |    1.388E+04    |   8.876E-01   |
|=========================================================================|
Total Elapsed Time (s): 73.546912

|=========================================================================|
                            Global Iter 24
|-------------------------------------------------------------------------|
|-------------------------------------------------------------------------|
|   Equation   |    RMS     |    MAX     | initialResidual | finalResidual |
|-------------------------------------------------------------------------|
|     U-x      |  4.863E-04 |  6.145E-03 |    1.707E+01    |   2.184E-06   |
|     U-y      |  7.067E-04 |  1.327E-02 |    1.756E+01    |   1.766E-06   |
|     U-z      |  1.548E-08 |  1.124E-07 |    7.951E-04    |   8.222E-10   |
|      p       |  4.495E-06 |  9.281E-05 |    1.268E+04    |   9.607E-01   |
|=========================================================================|
Total Elapsed Time (s): 75.702160

|=========================================================================|
                            Global Iter 25
|-------------------------------------------------------------------------|
|-------------------------------------------------------------------------|
|   Equation   |    RMS     |    MAX     | initialResidual | finalResidual |
|-------------------------------------------------------------------------|
|     U-x      |  4.451E-04 |  5.462E-03 |    1.566E+01    |   2.140E-06   |
|     U-y      |  6.621E-04 |  1.249E-02 |    1.619E+01    |   1.657E-06   |
|     U-z      |  9.773E-09 |  5.725E-08 |    5.212E-04    |   9.598E-10   |
|      p       |  4.101E-06 |  8.102E-05 |    1.166E+04    |   1.294E+00   |
|=========================================================================|
Total Elapsed Time (s): 78.208797

|=========================================================================|
                            Global Iter 26
|-------------------------------------------------------------------------|
|-------------------------------------------------------------------------|
|   Equation   |    RMS     |    MAX     | initialResidual | finalResidual |
```

Figure 3: Iterations

Meanwhile, a figure will pop-up and show residuals on-fly as shown in the figure below:

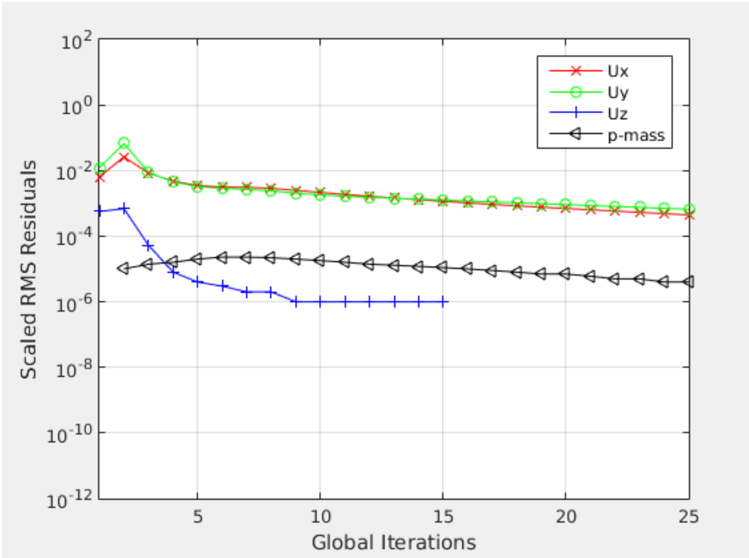

Figure 4: Residuals

5.3 Results

After the simulation finished according to the time controls (in the controlDict file located in the system directory), the user can open the postProcessing GUI to visualize the results by typing

$$postProcessing;$$

The postProcessing GUI looks like:

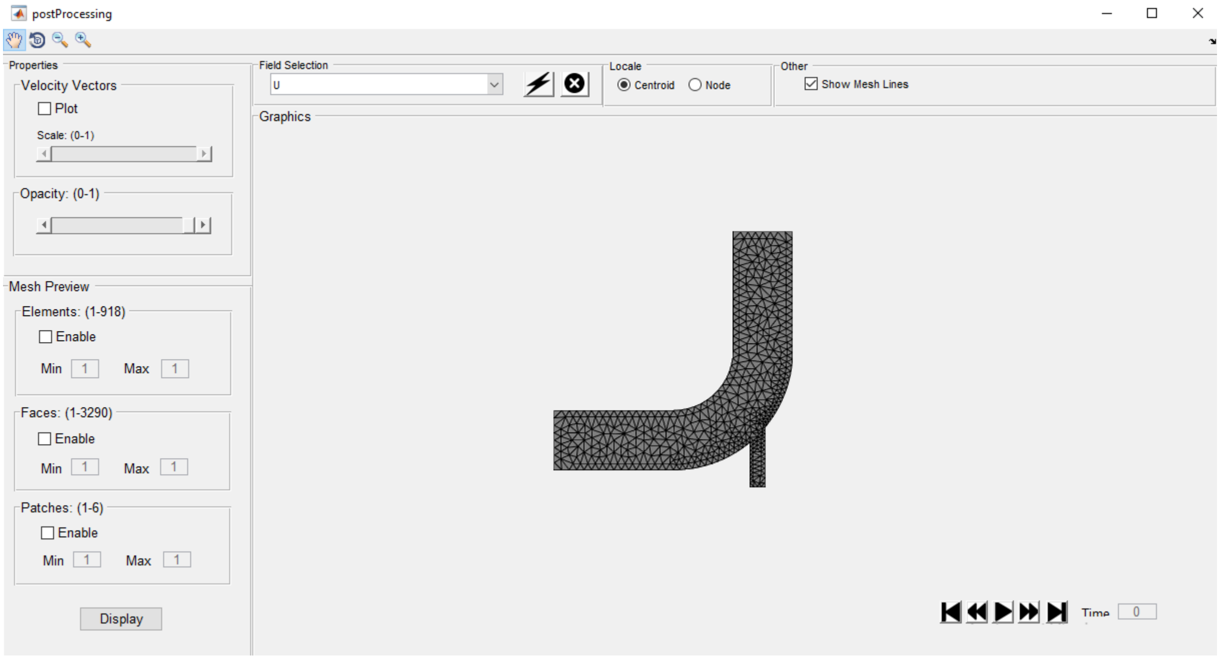

Figure 5: Mesh

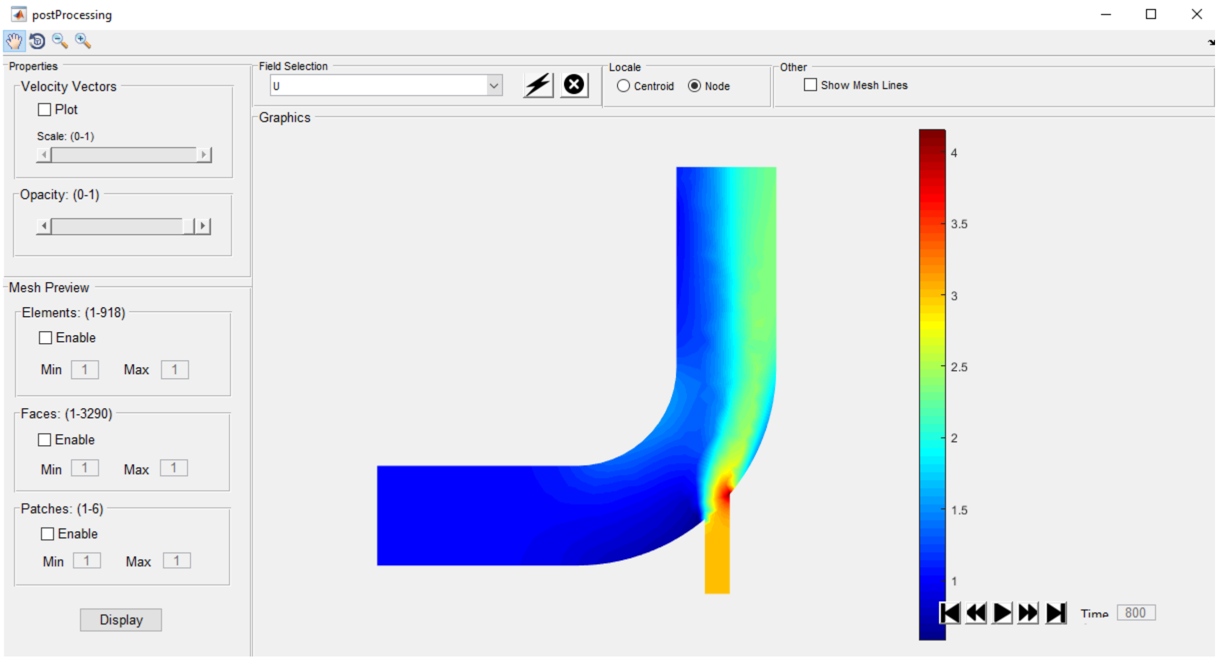

Figure 6: Velocity Contours

On another hand, the user may manually visualize the results by calling the functions

$$cfdPlotField('U');$$

or

$$cfdPlotVelocity;$$

References

[1] F Moukalled, L Mangani, M Darwish, et al. The finite volume method in computational fluid dynamics. *An Advanced Introduction with OpenFOAM and Matlab*, pages 3–8, 2016.