FOEuler User Guide

Mianzhi Wang

December 23, 2019

Contents

1 Installing FOEuler

1.1 System Requirement

FOEuler is a component of the project FOSolverS, which is build upon other open-source packages. The building FOSolverS is orchestrated by CMake, which you may want to install:

```
# dnf install cmake
```

CMake will guide you through the installation of missing dependencies. Please see Sec. 1.3 for details. You know the drill.

Besides the packages necessary for building, FOEuler reads unstructured grid generated by GMSH and writes post-processing data files in the VTK(ParaView) format. The following command installs these optional external tools:

```
# dnf install gmsh paraview
```

1.2 Getting FOEuler

The source code of the project FOSolverS is hosted on GitHub. The home page of the project FOSolverS is `https://github.com/mianzhi/fosolvers`. You may want to download the zip file which is a snapshot of the source code, but it is recommended to clone the full git repository by the following command:

```
$ git clone https://github.com/mianzhi/fosolvers.git
```

1.3 Building and Installing FOEuler

The project FOSolverS uses modern build system CMake. The building and installation of FOEuler is straightforward:

```
$ cd fosolvers # the top-level CMakeList.txt should be here
$ mkdir build
$ cd build
$ cmake .. # install the missing dependencies. You know the drill.
$ make
# make install
```

2 Using FOEuler

2.1 Overview of the Work Flow

Figure 1 shows the work flow of a simulation with FOEuler. FOEuler reads grid file "grid.msh" in GMSH format. Boundary conditions, initial conditions, simulation control, fluid property, and user-defined function (optional) file are all text files. Their file name must exactly be "bc", "ic", "sim", "fl", and "udf" respectively. As the computation runs, post-processing files "rst_n.vtk" ("n" is an integer starting from 0) are generated. These post-processing files can be read by visualization software ParaView.

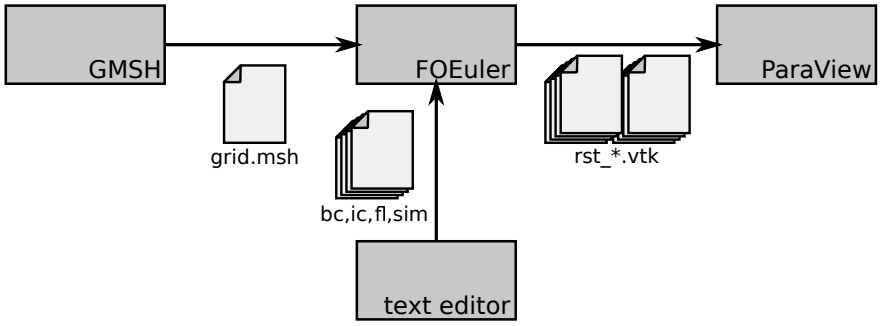

Figure 1: Work flow of FOEuler simulation.

2.2 Preparing Input Files

2.2.1 Grid File

Preparation of the grid file "grid.msh" follows general procedure of grid generation with GMSH. Note that the GMSH "geometric entity" of all the surfaces on which boundary condition are specified should be recorded. The value of "geometric entity" will be used as "geometric group identifier" to associate boundary condition to a specific surface.

2.2.2 Simulation Control File

Figure 2 shows an example of simulation control file. A "sim" file consists of two lines, the first line is the total simulation time, the second line is the time interval between post-processing outputs.

```
2e-2          # final time
5e-4          # output interval
```

Figure 2: An example of "sim" file.

Table 1: All supported boundary conditions.

Boundary Type	Code	# R. Par.	Real Parameters
Wall	0	0	N/A
In-flow (static)	10	5	static pressure, static temperature, velocity/direction (3 components)
(UDF)	110	5	UDFs of the above variables (integers in float form)
In-flow (total)	11	5	total pressure, total temperature, velocity/direction (3 components)
(UDF)	111	5	UDFs of the above variables (integers in float form)
Out-flow	20	1	static pressure
(UDF)	120	1	UDF of the above variable (integers in float form)
Far-field	30	5	static pressure, static temperature, velocity (3 components)
(UDF)	130	5	UDFs of the above variables (integers in float form)

2.2.3 Initial Condition File

Figure 3 shows an example of initial condition file. The "ic" file has six lines. The first line is an integer switch of user-defined function (UDF). If the value is "0", UDF is not used; If the value is "1", UDFs are used for all the five state variables of initial condition. The state variables are static pressure, static temperature, and the three components of velocity. The initial state is set uniformly through the computation domain according to the given state variables.

```
0             # do not use UDF
0.4e5         # initial pressure
200.          # initial temperature
350.          # initial velocity, x component
0.            # initial velocity, y component
0.            # initial velocity, z component
```

Figure 3: An example of "ic" file.

2.2.4 Boundary Condition File

Figure 4 shows an example of boundary condition file. The "bc" file contains several blocks. The blocks are separated by exactly one empty line. The first block has two lines. The first line specifies the number of boundary conditions, which should equal to the number of the following blocks. The second line specifies the maximum number of real parameters for one boundary condition. The value should be set such that the boundary condition with the largest number of real parameters does not have more real parameters than this value. Each following block associates boundary condition for one geometric group, a.k.a. all surfaces with the same geometric group identifier that are recorded as in Sec. 2.2.1. The geometric group identifier is specified in the first line of each block, followed by another two integer lines, type of the boundary condition, and the number of real parameters. The number of real parameters determines the number of real parameter lines of the boundary condition. The supported types of boundary condition and required real parameters are shown in Tab. 1.

If any of the surfaces is not explicitly associated with a boundary condition, the wall boundary condition is used for this surface.

```
2           # number of conditions
5           # maximum number of parameters per condition

12          # geometric group identifier
11          # condition type: inflow with total properties
5           # number of real parameters
1.01325e5   # real parameter: total pressure
300.        # real parameter: total temperature
382.22      # real parameter: velocity/direction of the flow, x component
0.          # real parameter: velocity/direction of the flow, y component
0.          # real parameter: velocity/direction of the flow, z component

1           # geometric group identifier
20          # condition type: outflow
1           # number of real parameters
0.73146e5   # real parameter: static pressure
```

Figure 4: An example of "bc" file.

For the two in-flow boundary conditions (static and total), FOEuler computes in-flow Mach number M based on the given in-flow velocity and the thermal state. In case that $M < 1$, the given velocity will be normalized to set the direction of flow, and the magnitude of the velocity is computed from inside of the domain.

2.2.5 Fluid Property File

Figure 5 shows an example of fluid property file. The "fl" file has two lines, specifying the mass-specific specific gas constant and the ratio of heat capacities.

```
287.058     # specific gas constant
1.4         # gamma
```

Figure 5: An example of "fl" file.

2.2.6 UDF File (Optional)

Figure 6 shows an example of UDF file. The first line of the "udf" file is the total number of UDFs. Each of the UDFs takes a single line. Each of the UDF should be a scalar function of $\{x, y, z, t\}$. Details about the supported operators and rules of ODF follows the documentation of open-source project libmatheval.

In case that boundary conditions and(/or) initial conditions defined by UDFs are used, the "udf" must be provided. The real parameters of boundary condition in "bc" file and(/or) the real state variables in "ic" file should be replaced by the integer index of the associated UDF in float form, i.e. "3." can be used to associate with the third UDF.

2.3 Run FOEuler

Once all the input files are prepared, the FOEuler solver can be started by:

```
$ cd my_case # this folder contains all input files
$ foeuler
```

```
5
90000.-x*5000
300.
118.+x*35.
x*4.
0.
```

Figure 6: An example of “udf” file.

The execution of the program may take a long time depending on the size and nature of the problem. Post-processing files are generated in the same folder as the solver is running.