SOMA Documentation

Javier Villarreal

09/27/2021

1 Introduction

The purpose of this document is to keep track of the project to port the C++ SOMA code to Fortran.

2 C++ Code Architecture

The C++ code is broadly arranged in the following way:

1. Reads in simulation parameter text files and defines some constants.
2. constructs the objects that hold the variables: `Domain`, `SimFluid`, and `Approximator`
3. Reads in geometry-based data files to appropriate object variables.
4. Enforces boundary conditions and calculates derivatives on initial data.
5. Defines parameters for the genetic algorithm optimization code.
6. (optional) Reads initial values from text files and re-calculates BC's and derivatives.
7. Prints initial data to output text files.
8. Loops over time steps. (SOMA proper)
9. Prints latest values to output text files.

In the SOMA step, the code splits into one of two modes, explicit (using Runge-Kutta) or implicit (using RBF addition). Different mechanisms within the code based on convergence criteria or iteration counts switch the code between one mode or the other. Each one will be explained in its own section.

3 Pre-Processing

Geometry data used to run the simulation is currently created on a case by case basis using Matlab codes to generate .txt files that are then read in by SOMA. SimulationValues.txt defines the configuration of the simulation, and Sizes.txt contains metadata used to properly allocate arrays.

- SimulationValues.txt (Mach, AOA, Re)
- Sizes.txt (# of domain, body, farfield, cloud, ghost, extrapolation, total nodes)

The geometry data files themselves are:

- x,y.txt (node coordinates)
- DX,DY.txt (DQ coefficients)
- EC.txt (extrapolation coefficients)
- Jd,Jb,Jf.txt (domain, body, farfield node indices)
- nxb,nyb.txt (body node unit normal vectors)
- nxf,nyf.txt (farfield node unit normal vectors)
- s11,s12,s21,s22.txt (Flow tangency matrices)

Most of the data files are read by the code into variables holding "carbon copies" of the data, with a few exceptions:

- The data in SimulationValues and Sizes are stored in individual variables for each number, rather than vectors.
- The normal vectors, which are created in separate files by components (e.g. nxb and nyb are the x- and y- components of the normal vectors), are read by the code into arrays containing the entire vectors, `b_normal` and `f_normal`, where each row corresponds to a node and the columns correspond to [x y].
- The matrices used to enforce flow tangency boundary conditions are stored in the array `slip`, where each row corresponds to a node, and the columns correspond to [s11 s12 s21 s22]

The structure of the normals and matrices were defined to reflect the text files to maintain readability, but it means the code will loop over those matrices as in

```
do node=1,n
    use array(node,:)
end do
```

which is slower than looping over the right-most index due to how Fortran stores data in memory. Eventually, the text files and arrays should be restructured for optimization.