Graz University of Technology

Pharmaceutical Multiphase Reactors
CHE.782

Design of Multiphase Flow Processes
669.266

Hands on the „CPPPO“ library

A – General Information
B – Guide - Verification Cases

Ass.Prof. Dr. Stefan Radl,
MSc. Federico Municchi
Email: radl@tugraz.at
Institute for Process and Particle Engineering
Inffeldgasse 13/III
TU Graz

A part of this teaching material has been prepared for NanoSim (http://sintef.no/NanoSim/

A – Gerneral Information

Graz University of Technology

Why verfication/test cases:

- Ensure right implementaion (verification cases)
- Showcases in order to demonstrate software capabilities (test cases to produce „eye candy“)
- Ensure software backward compability

Where to find:

- Verification cases: *„examples/verification/“*
- Test cases: *„/examples/codeTest“*

When to run:

- After compiling the *CPPPO* core library
- Test cases can be run after compiling the C3PO_CSV stand-alone binary
- Verification cases can be run after compiling the C3PO-OpenFOAM interface and the solvers in *„/applications“*
- **AFTER** reading all ***.md** files in *„doc/“* for general informations regarding structure and input CPPPO

Verification case - potentialStokesFilter

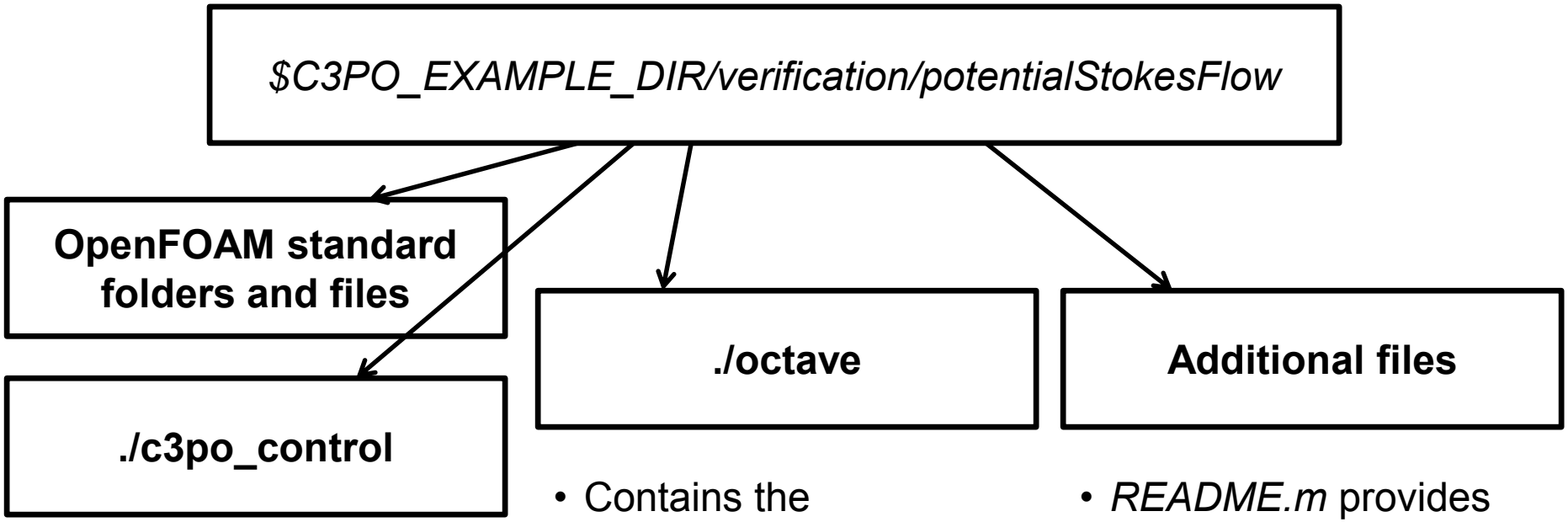
Requirements:

- Be sure that the *CPPPO-OpenFOAM interface* is compiled correctly.
- Ensure that the application in *applications/stokesFilter* is compiled correctly.
- Ensure that the application in *applications/irrotationalFilter* is compiled correctly.

To check if any solver is compiled just type: which «solver-name»
For example: *which stokesFilter*

If the prompt returns the file location, then your solver is compiled

Case structure



- Contains c3po.input and the json files required to run CPPPO (see *./doc*)
- It should **always** be named c3po_control

- Contains the octave/matlab scripts required to collect and plot data from cpppo output

- *README.m* provides a brief description of the case
- *Allclean/Allrun.sh* allow to clean and run the whole case
- *runPotential.sh/runStokes.sh* allow to run just one of the solvers

Walk-through (1)

Have a look at ./c3po_control/c3po.input and ./c3po_control/c3po.json

These files contain all the operation and selectors used for this case as well as the main settings for CPPPO.

You can refer to *doc/02_c3poInput.md* for a detailed description.

The file*./c3po_control/mesh.json* allows to specify the tolarance for filtering operations and mesh check.

Walk-through (1) – c3po.input

Branch: master ▾ C3PO / examples / verification / potentialStokesFilter / c3po_control / **c3po.input**

cgonIva picking up status of commit b08c04b4842ca064004f6c3a0f1d3d9d14bd49ad ...

1 contributor

24 lines (22 sloc) | 1.17 KB | Raw | Bla

Declaration of filtering operations

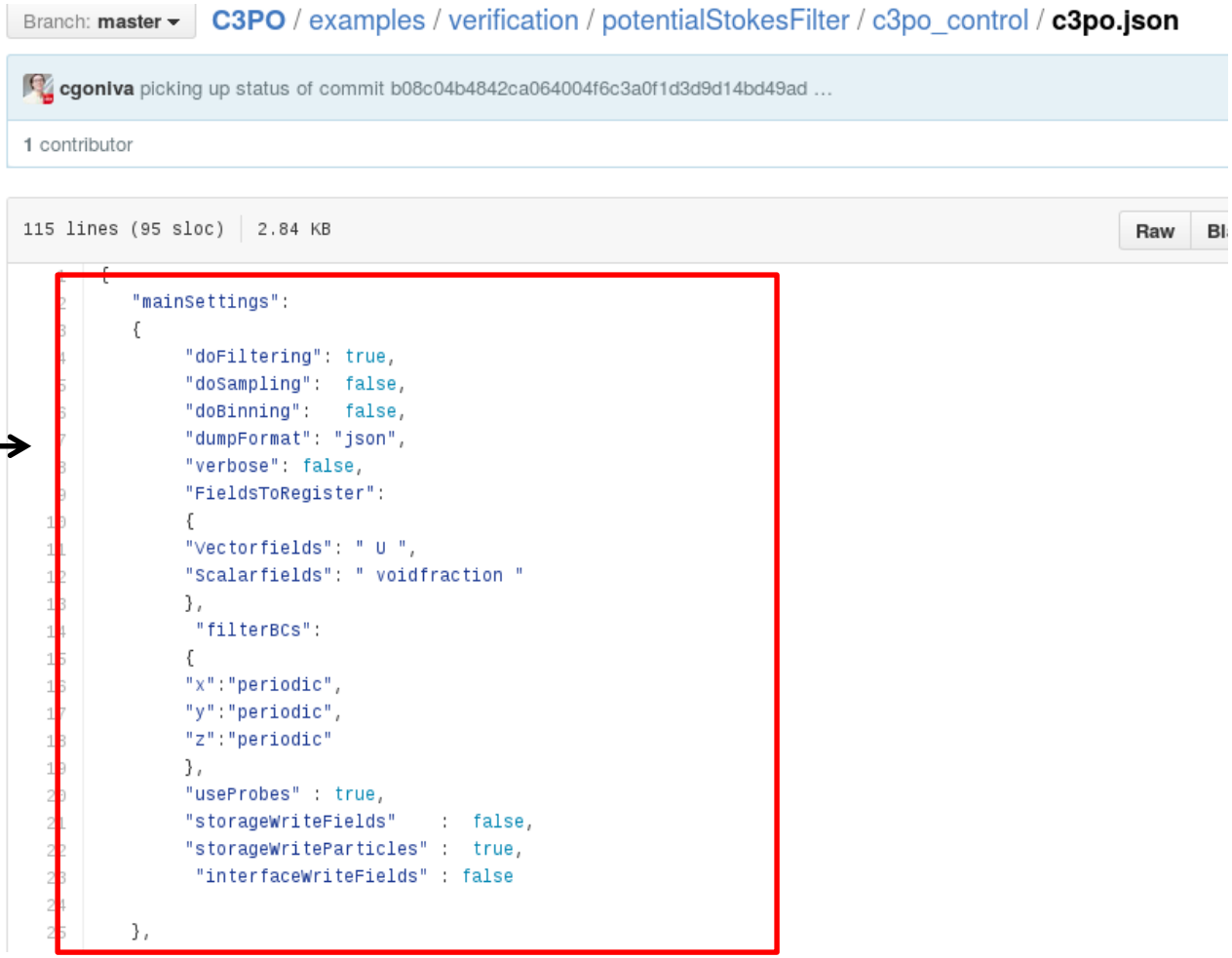
Declaration of selectors

Declaration of filters

```
1  operation   filtering    Favre              #Favre averaging
2  operation   filtering    Favre1             #Favre averaging
3  selector    cellUnstruct                 myCell0        #selectors for cell index calculation#
4  selector    cellUnstruct                   myCell1
5  selector    cellUnstruct                   myCell2
6  selector    cellUnstruct                   myCell3
7  selector    cellUnstruct                   myCell4
8  selector    cellUnstruct                   myCell5
9  selector    cellUnstruct                   myCell6
10 selector    cellUnstruct                   myCell7
11 selector    cellUnstruct                   myCell8
12 selector    cellUnstruct                   myCell9
13
14 selector    filter                   0.75          #filters named as their filter-size
15 selector    filter                   1.00
16 selector    filter                   1.25
17 selector    filter                   1.50
18 selector    filter                   1.75
19 selector    filter                   2.00
20 selector    filter                   2.25
21 selector    filter                   2.50
22 selector    filter                   2.75
23 selector    filter                   3.00
```

Walk-through (1) – c3po.json

Main settings to control I/O and set general run parameters



Branch: master ▾ C3PO / examples / verification / potentialStokesFilter / c3po_control / c3po.json

cgoniva picking up status of commit b08c04b4842ca064004f6c3a0f1d3d9d14bd49ad ...

1 contributor

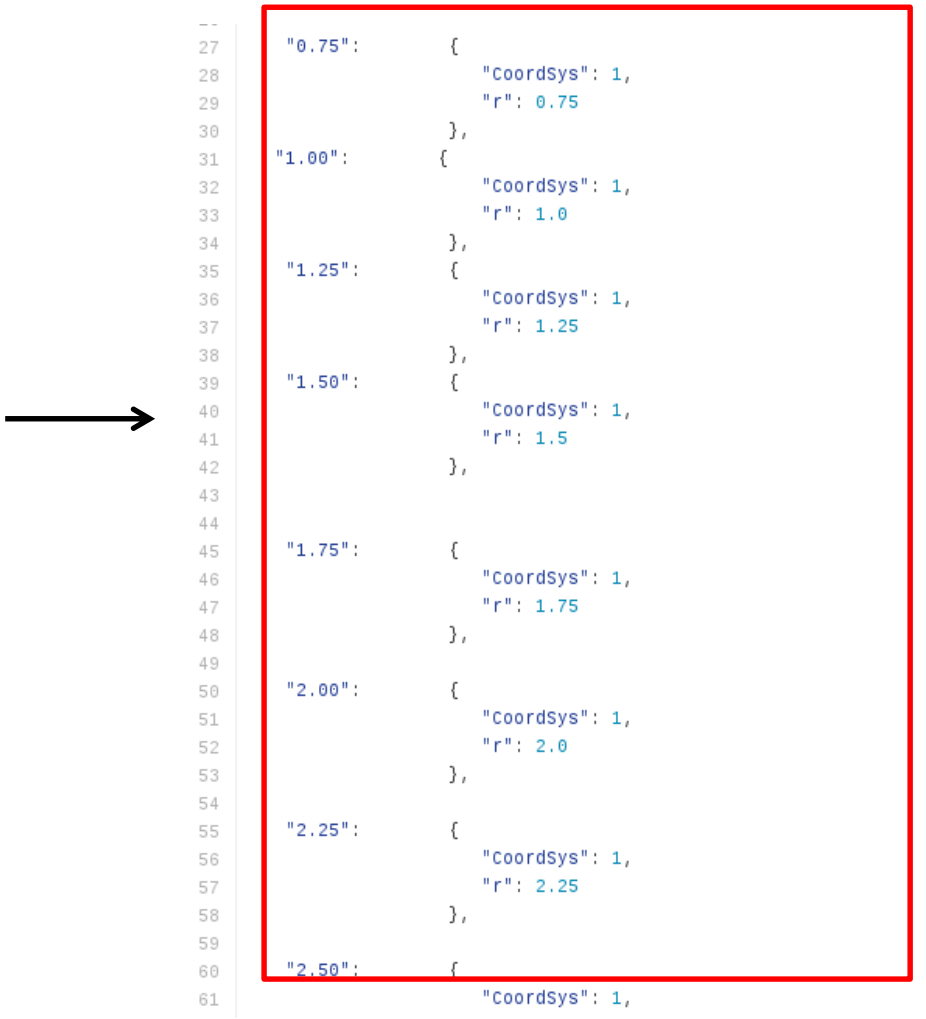
115 lines (95 sloc) | 2.84 KB | Raw | Bl

```
{
    "mainSettings":
    {
         "doFiltering": true,
         "doSampling":  false,
         "doBinning":   false,
         "dumpFormat": "json",
         "verbose": false,
         "FieldsToRegister":
         {
         "vectorfields": " U ",
         "Scalarfields": " voidfraction "
         },
          "filterBCs":
         {
         "x":"periodic",
         "y":"periodic",
         "z":"periodic"
         },
         "useProbes" : true,
         "storageWriteFields"    :  false,
         "storageWriteParticles" :  true,
          "interfaceWriteFields" : false

    },
```

Walk-through (1) – c3po.json

Definition of filters according to their naming in *c3po.input*



```
"0.75":         {
                    "CoordSys": 1,
                    "r": 0.75
                },
"1.00":         {
                    "CoordSys": 1,
                    "r": 1.0
                },
"1.25":         {
                    "CoordSys": 1,
                    "r": 1.25
                },
"1.50":         {
                    "CoordSys": 1,
                    "r": 1.5
                },


"1.75":         {
                    "CoordSys": 1,
                    "r": 1.75
                },

"2.00":         {
                    "CoordSys": 1,
                    "r": 2.0
                },

"2.25":         {
                    "CoordSys": 1,
                    "r": 2.25
                },

"2.50":         {
                    "CoordSys": 1,
```

Walk-through (1) – c3po.json

Definition of the operations to perform according to their naming in *c3po.input*

```
"Favre":
        {
         "type":"Favre",
         "VectorfieldsToFilter": "U",
         "ScalarfieldsToFilter": " ",
         "phaseFractionField": "voidfraction",
         "VectorfieldsForVarianceName1": "U",
         "VectorfieldsForVarianceComputeOffDiagonal": [false],
         "ScalarfieldsToFilter": " ",
         "ScalarfieldsForVectorScalarMixedVariance"  : " off",
         "ScalarfieldsForVarianceName1" : "",
         "ScalarfieldsForVarianceName2" : "",
         "lagrangian": true,
         "probesName":"particleCenter"
        },


"Favre1":
        {
         "type":"Favre",
         "VectorfieldsToFilter": "U",
         "ScalarfieldsToFilter": " ",
         "phaseFractionField": "voidfraction",
         "VectorfieldsForVarianceName1": "U",
         "VectorfieldsForVarianceComputeOffDiagonal": [false],
         "ScalarfieldsToFilter": " ",
         "ScalarfieldsForVectorScalarMixedVariance"  : " off",
         "ScalarfieldsForVarianceName1" : "",
         "ScalarfieldsForVarianceName2" : "",
         "lagrangian": true,
         "probesName":"particleCenter"
        }
```

Walk-through (2)

Check ./c3po_control/probeSettings.json

This file allows to define probes (or particles in CPPPO).
Refer to $C3PO_SRC_DIR/doc/20_probesAndParticles.m

Check ./octave

This folder contains two octave/matlab scripts:

- *postproc.m.potential:* collects and plot the json output from CPPPO against the analytical solution for the potential flow.
- *postproc.m.stokes:* collects and plot the json output from CPPPO against the analytical solution for the Stokes flow.

Walk-through (3)

Run the verification case

- Pre-processing operations:
  - Run *blockMesh*
  - Run *setFields*

- Run *irrotationalFilter* **or** *stokesFilter*
- Post-processing operations:
  - Type «cd octave»
  - Run «octave postproc.m.potential» or «octave postproc.m.stokes»
  - Visualize «verification.eps»

Alternatively, **you can run the whole case using ./Allrun.sh**

Remember to run Allclean before leaving the directory!

Walk-through (4)

Modify the verification case

- ***Add a new filter:***
  - Add a new filter and a new selector in c3po.input, name them as you like.
  - Copy an existing filter in *c3po.json* and name it accordingly to your new filter.
  - Modify the coordinate system to cartesian and set a filter volume of 8 following the documentation in *doc/14_selector_type.md*
  - Modify the *postproc.m* script in */octave* to account for the new filter.
  - Run the modified case.

Impressum & Disclaimer

Graz University of Technology