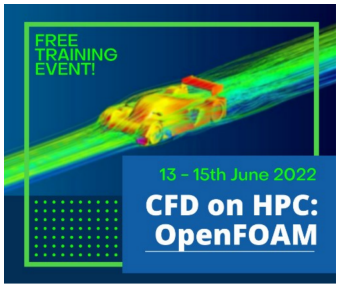
Programming in OpenFOAM



EuroCC workshop

Aleksander GRM

June 15, 2022

Vsebina

Programming Introduction

Simple Thermal case
- Solver
- Case

Programming Introduction

To understand OpenFAOM programming one must understand basics of `c++` language.

All sources are stored in

- solvers: `$WM_PROJECT_DIR/applications`
- API: `$WM_PROJECT_DIR/src`

New explorers should search over this two folders for a solution!

OpenFOAM uses special environment for compiling: `WMake`

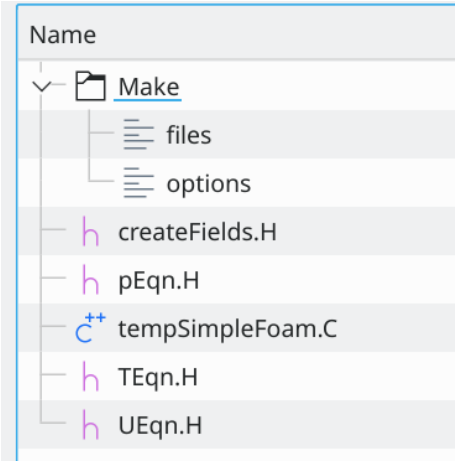
Each new application, boundary condition, ..., is first designed and later compiled.

```
First one must clean old stuff
    $> wclean

Latter comile is executed with
    $> wmake
```

In each code folder there is a file structure

```
Root folder:
    - *.H  this are header files
    - *.C  this are source files

Make folder:
    - files : containing directives for wmake of executable
    - options: add include folders and lib folders
```

Compiling OF code structure – example



Simple Thermal case

For the purpose of the thermal process simulation, we develop new solver.

The idea is to merge `simpleFoam` and `scalarTransportFoam` solvers. This is not correct modelling of thermal process, but it shows the complete process how to upgrade `simpleFoam` with thermal conservation law.

We add additional equation for scalar transport

$$\frac{\partial T}{\partial t} + (\mathbf{v} \cdot \nabla) T = \beta \Delta T,$$

where is β thermal diffusivity of specific media (air, water).

File `TEqn.H` code snippet

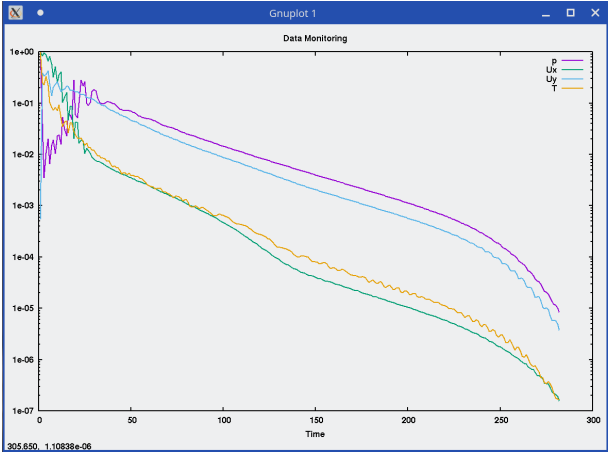
```
1   fvScalarMatrix TEqn
2   (
3       fvm::ddt(T)
4       + fvm::div(phi, T)
5       - fvm::laplacian(DT, T)
6       ==
7       fvModels.source(T)
8   );
9
10  TEqn.relax();
11  fvConstraints.constrain(TEqn);
12  TEqn.solve();
13  fvConstraints.constrain(T);
```

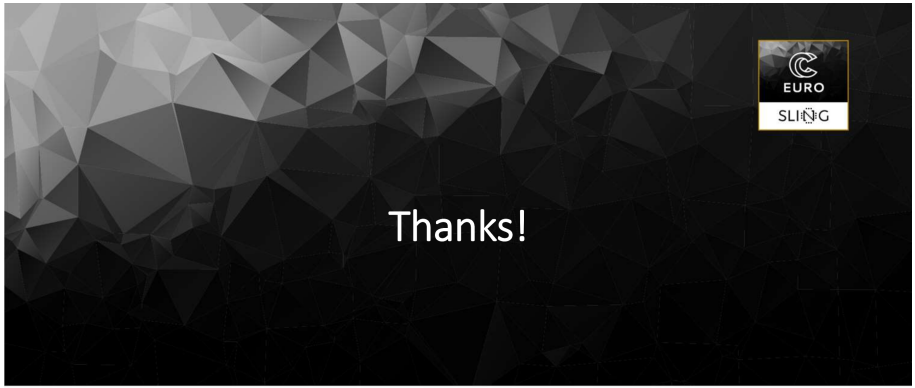
Code of inclusion of T in `simpleFoam`


```
    while (simple.loop(runTime))
    {
        fvModels.correct();

        // --- Pressure-velocity SIMPLE corrector
        {
            #include "UEqn.H"
            #include "pEqn.H"
        }

        #include "TEqn.H"   (!additional line!)

        laminarTransport.correct();
        turbulence->correct();

        runTime.write();
    }
```

Test case is simple 2D geometry with a channel flow (inlet+outlet) with specific fixed values of T on a boundary.

Results residuals.





This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 951732. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Bulgaria, Austria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, Greece, Hungary, Ireland, Italy, Lithuania, Latvia, Poland, Portugal, Romania, Slovenia, Spain, Sweden, United Kingdom, France, Netherlands, Belgium, Luxembourg, Slovakia, Norway, Switzerland, Turkey, Republic of North Macedonia, Iceland, Montenegro

EuroHPC Joint Undertaking