Guide to chtMultiRegionSimpleFoam

David Tranter

University of Exeter

February 24, 2014

Overview

chtMultiRegionSimpleFoam is a multi region solver that couples conjugate heat transfer between solid and fluid volumes.

To run chtMultiRegionSimpleFoam you will need to provide

- A multi-region mesh
- Boundary conditions for each region
- Material properties for each region
- Solver properties for each region

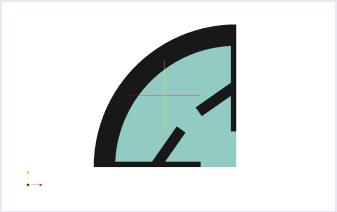


Figure: Solid region is black and the fluid region blue.

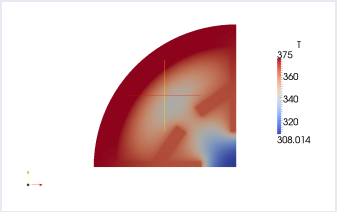


Figure: Temperature distribution across the slice.

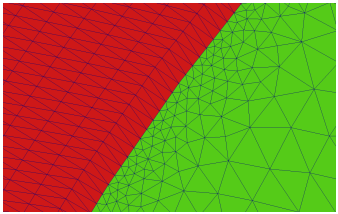
Sliced plane of a multi-region single tube snowflake model.

Mesh

chtMultiRegionSimpleFoam requires a multi region mesh. This means providing the following files in addition to the standard files
-cellZones
-faceZones
-sets/<fluid-region>-cells
-sets/<solid-region>-cells

In Pointwise v17.1R4 this can easily be generated by selecting different blocks and selecting the volume to cell selection.

Currently chtMultiRegionSimpleFoam requires the mesh to be conformed at the boundary between volumes. That is the boundary that seperates the regions must be edges that define one edge in a cell in the solid region and one edge in a cell in the fluid region.



Example of two regions seperated by a boundary made up of edges that are members of cells in both regions.

Boundary conditions

The standard boundary conditions available in OpenFOAM can also be used to define the behaviour of the velocity/temperature/pressure values at the boundary. However they must be provided in 0/<fluid-cells>/ and 0/<solid-cells>/ directories.

In addition template files must be provided that declare all the boundaries in the 0/ directory but these are not used in determining the boundary conditions. They are used to set up the boundary conditions after splitMeshRegions has been executed. After this has been run the 0/<fluid-cells>/ and 0/<solid-cells>/ will be replaced with default ones as defined in 0/. Therefore it might be easier to create a backup folder (0.bu for example) with the desired boundary conditions. Then once splitMeshRegions has been executed cp -r 0.bu/ 0/ will update the default conditions with the ones desired. changeDictionary command will also update the boundary conditions based on entries declared in /system/<region>/changeDictionaryDict

Boundary conditions

After splitMeshRegions has been executed OpenFOAM will create new boundaries that define the interface between fluid region and solid regions.

The naming convention for these new boundaries are <fluidregion> _to_<solid-region> and <solidregion>_to_<fluid-region>. To define how to solve for temperature at the solid-fluid boundary a new boundary condition has to be used called turbulentTemperatureCoupledBaffleMixed.

This enforces that the temperature values/gradient agree either side of the boundary that seperates the solid/regions. The uniform value is simply a placeholder for the boundary condition to let it know it is a scalar value.

```
fluid_cells_to_solid_cells
{
    type            compressible::turbulentTemperatureCoupledBaffleMixed;
    value           uniform 293;
    neighbourFieldName T;
    K               basicThermo;
    KName           none;
}
```

Region properties

After splitMeshRegions has been executed new directories will be created in the constant folder for each region that include a polyMesh subdirectory with files that contain the information on cells that define that region. In the constant/<fluid-region> folder you will need to include the following files

g
radiationProperties
RASProperties
thermophysicalProperties
turbulenceProperties

In the constant/<solid-region> folder you will need to include the following files

solidThermophysicalProperties

It is the solidThermophysicalProperties and the thermophysicalProperties that define the relevant constants that describe heat transfer within the region.

Example of fluid thermodynamic constants

```
thermoType
hRhoThermo<pureMixture<sutherlandTransport<specieThermo<hConstThermo<perfectGas>>>>>>;

mixture
{
    specie
    {
        nMoles          1;
        molWeight       28.96;
    }
    thermodynamics
    {
        Cp              1006.43;
        Hf              0;
    }
    transport
    {
        As              1.458e-6;
        Ts              110.4;
        Pr              0.71;
    }

}
```

In this declaration we define the chemical properties (for use in the perfect gas law), the specific heat constant (Cp) and the constants for the sutherland law for viscocity which is a temperature dependent function. The thermal conductivity isn't explicitly given but is given implicitly via the Prandtl number. Hf is the heat of fusion constant and is only relevant in multi phase flows. The density is solved using the PV = NRT perfect gas law.

Region properties

Example of fluid thermodynamic constants

```
thermoType constSolidThermo;

constSolidThermoCoeffs
{
    //thermal properties
    rho rho [1 -3  0  0 0 0 0] 8193.25;
    Cp  Cp  [0  2 -2 -1 0 0 0] 435;
    K   K   [1  1 -3 -1 0 0 0] 14.19;

}
```

In this declaration we define the density (rho), specific heat constant (Cp) and the thermal conductivity (K) all given in SI units. In version 2.2.x version of OpenFOAM solid/fluid regions can have polynomial functions of temperature for these constants.

Finite volume settings

In the system directory each region will need its own subdirectory with a fvSchemes, fvSolution and decomposeDict files. The syntax for these files are as normal. In the fvSolutions file the GAMG solver can be the optimal choice for solving for the pressure field. The syntax is given below. The only setting to change between models is the nCellsInCoarsestLevel setting which should be the square root of the number of cells in the region.

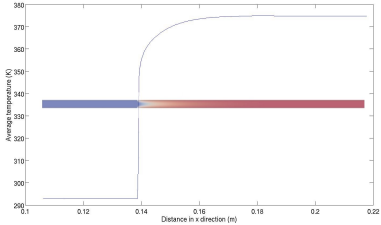
```
p_rgh
{
    solver  GAMG;
    tolerance 1e-6;
    relTol 0.0001;
    smoother GaussSeidel;
    nPreSweeps 1;
    nPostSweeps 2;
    cacheAgglomeration on;
    agglomerator faceAreaPair;
    nCellsInCoarsestLevel  10000;
    mergeLevels 1;
}
```

Finite volume settings

Focusing on generating a high quality mesh and using higher order finite difference schemes can be a good idea especially if the mesh is to be used in subsequent simulations. Below is an example of higher order bounded schemes that can be used in the fvSchemes file.

```
ddtSchemes
{
        default         steadyState;
}

gradSchemes
{
        default         Gauss linear;
}

divSchemes
{
        default         Gauss linear;
}

laplacianSchemes
{
        default         Gauss linear limited 1.0;
}

interpolationSchemes
{
        default         linear;
}

snGradSchemes
{
        default         limited 1.0;
}

fluxRequired
{
        default         no;
        p_rgh;
}
```

To judge convergence of the simulation it is more appropriate to monitor the values of temperature over a line that runs throught the fluid domain. The simulation can be said to have converged when the values of temperature over the line do not change over subsequent iterations.



After convergence the command paraFoam -touchAll will create files that allow for postprocessing of the individual regions. Post processing in paraview can then be done in the normal way.

"*Time step continuity errors increasing which leads to divergence*"

Time step continuity errors are related to the pressure solver tolerance. If these are not tight enough this can cause the time step continuity error to grow. The solution is to tighten up the pressure solver tolerance/relative tolerance in the /system/<fluid-region>/fvSolution file.

"*The temperature residual is very small yet clearly the temperature has not converged to a physical solution*"

To judge convergence of temperature it is more appropriate to monitor the values of temperature over a line that runs throught the fluid domain rather than the residual. Unfortunately this takes a lot longer to converge than the velocity and pressure fields. Try increasing the relaxation factor to increase convergence.

" *The velocity/pressure residuals are oscillating*"

Check the boundary conditions/relaxation factors/solver tolerances and mesh quality. If that all seems fine it is possible, especially with the wavy models, that the actual solution is transient in nature. If choosing a laminar flow it is possible that the actual solution is turbulent.

" *The temperature field contains temperatures lower than physically possible*"

This can happen at areas of high temperature gradients. I.e the entrance of the heated section. In OF 2.2.x you can bound the temperature fields by upper/lower limits but this can also be easily implemented in other versions.