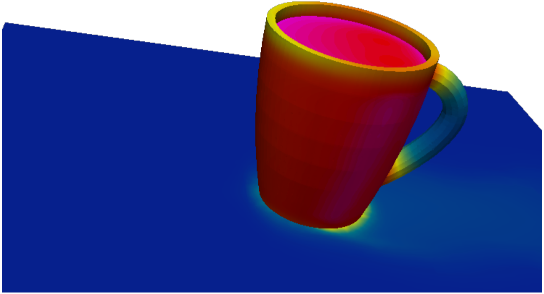
Heat transfer in OpenFOAM

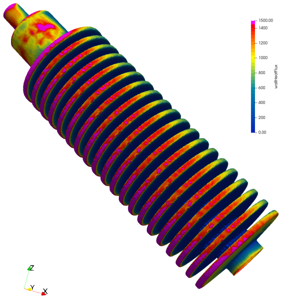
Dr. Johann Turnow,
silentdynamics GmbH

2018-02-22



1. Heat conduction
2. Heat convection
3. Heat radiation

- steady, transient
- compressible, incompressible
- ...



Overview of OpenFOAM solvers for heat transfer analysis

- **laplacianFoam:**
  Transient, incompressible, thermal diffusion according to Fourier's law
- **scalarTransportFoam:**
  Steady-state, incompressible, laminar, passive scalar e.g. temperature for a given velocity field
- **buoyantBoussinesqSimpleFoam:**
  Steady-state, thermal, natural convection, incompressible, Boussinesq's approximation
- **buoyantBoussinesqPimpleFoam:**
  Transient, thermal, natural convection, incompressible, Boussinesq's approximation

Overview of OpenFOAM solvers for heat transfer analysis

- **buoyantSimpleFoam:**
  Steady-state, natural convection, compressible (sub-sonic), including radiation
- **buoyantPimpleFoam:**
  transient, natural convection, compressible(sub-sonic), including radiation
- **rhoSimpleFoam:**
  Steady-state, thermal, compressible(sub-sonic)
- **rhoSimplecFoam:**
  Steady-state, thermal, compressible(sub-sonic) -Pressure under relaxation =1
- **rhoPimpleFoam:**
  Transient, thermal, compressible(sub-sonic)

Overview of OpenFOAM solvers for heat transfer analysis

- **chtMultiRegionFoam:**
  Transient, compressible, conjugate heat transfer between solid and fluid
- **chtMultiRegionSimpleFoam:**
  Steady-state, compressible, conjugate heat transfer between solid and fluid
- **thermoFoam:**
  Transient, evolves the thermophysical properties for a frozen velocity field

Basic solver: laplacianFoam

- Simple heat conduction equation according to Fourier's law

$$\frac{\partial T}{\partial t} = \frac{\lambda}{\rho c_p} \frac{\partial^2 T}{\partial x^2} \tag{1}$$

- Take a look at the solver
  - `cd $FOAM_SOLVERS` or `sol`
  - `cd basic/laplacian`
  - `gedit laplacianFoam.C`

```
solve
(
fvm::ddt(T) - fvm::laplacian(DT, T)
);
```

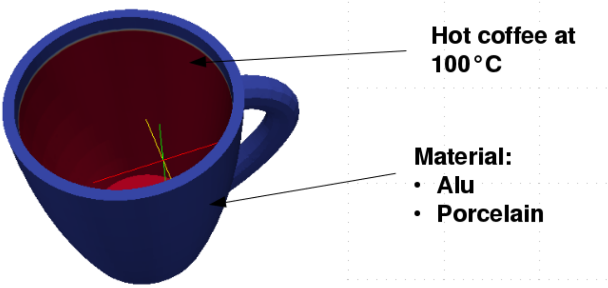
Basic solver: laplacianFoam

- Define the heat diffusivity DT:
  - `gedit constant/transportProperties`

```
//DT = heat diffusivity
DT DT [ 0 2 -1 0 0 0 0 ] 1.6667e-05; //air
//DT DT [ 0 2 -1 0 0 0 0 ] 0.144e-06; //water
//DT DT [ 0 2 -1 0 0 0 0 ] 9.3e-05; //alu
```

Example coffee cup

- Using `laplacianFoam` to simulation usual problems
- Let's try to analyze the temperature distribution in our coffee cup
- Question: Can you touch the cup without any pain?



Example coffee cup

- Setting the boundary conditions
- `gedit 0/T`

```
internalField uniform 273;
boundaryField
{
   sideWalls
  {
   type zeroGradient; //adiabatic
  }
   coffee
   {
   type fixedValue; // fixed Temperature b.c.
   value uniform 373;
   }
}
```

Example coffee cup

- Setting the boundary conditions
- `gedit 0/T`

```
internalField uniform 273;
boundaryField
{
   sideWalls
  {
   type zeroGradient; //adiabatic
  }
   coffee
   {
   type fixedGradient; //fixed heat flux b.c.
   gradient 10000;
   value uniform 373;
   }
}
```

Example coffee cup

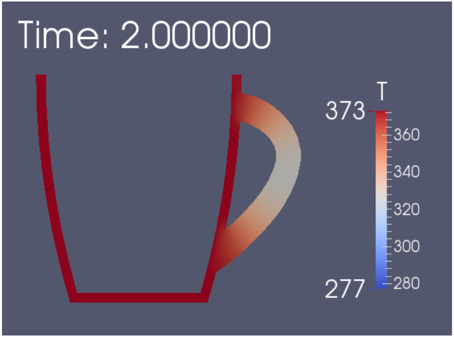
- Define the heat diffusivity DT for **alu**:
  - `gedit constant/transportProperties`

```
//DT = heat diffusivity
//DT DT [ 0 2 -1 0 0 0 0 ] 1.6667e-05; //air
//DT DT [ 0 2 -1 0 0 0 0 ] 0.144e-06; //water
DT DT [ 0 2 -1 0 0 0 0 ] 9.3e-05; //alu
```

- `decomposePar`
- `foamJob -parallel laplacianFoam`
- `tail -f log`

Example coffee cup

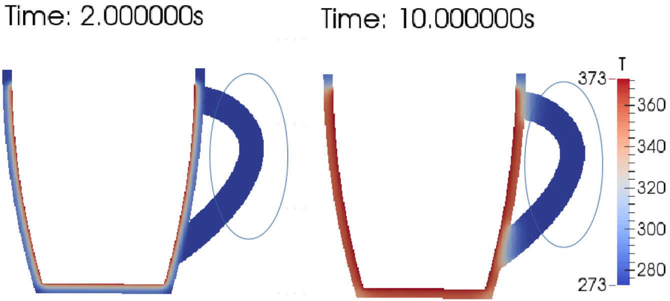
- Take a look at the temperature after 2.0sec for our **alu** cup



- The **alu** gives pretty hot fingers after 2.0sec ☺

Example coffee cup

- Comparison to a usual porcelain cup



- The **porcelain** cup gives us cool fingers fingers after 2.0sec and 10.0sec (-:

Outcome

- Laplacian solver gives a fairly good overview for simple heat conduction problems
- Always the first choice for simple heat conduction solutions
- First step: Think about which results you expect
- Important to avoid nonphysical solutions ... :-)
- Always take a look at the residuals
- Always remember that the mesh resolution influences the results in case of heat transfer dramatically!
- A Priori: Which boundary conditions should be applied?
- Be careful with the constant heat flux boundary condition

Which solvers can we use?

- **scalarTransportFoam** for laminar, unsteady/steady flows
- **buoyantBoussinesqSimpleFoam:**
  Steady-state, thermal, natural convection, incompressible, Boussinesq's approximation
- **buoyantBoussinesqPimpleFoam:**
  Transient, thermal, natural convection, incompressible, Boussinesq's approximation

→ Set the gravitation to Zero for simple passive scalar flows

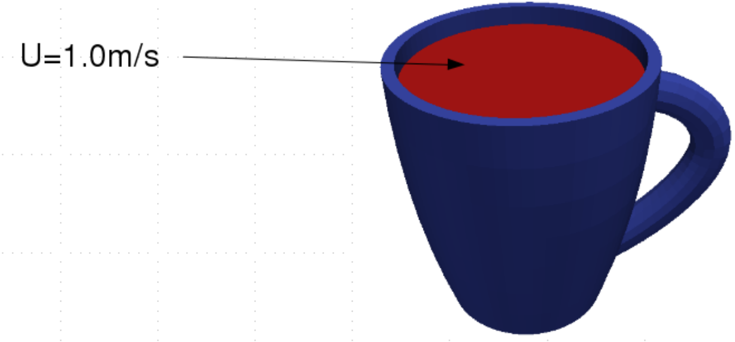
Wich equation is solved?

```
volScalarField alphaEff("alphaEff", turbulence->nu()/Pr
+ alphat);
```

```
fvScalarMatrix TEqn
(
    fvm::div(phi, T)
  - fvm::laplacian(alphaEff, T)
  ==
    radiation->ST(rhoCpRef, T)
    + fvOptions(T)
);
```

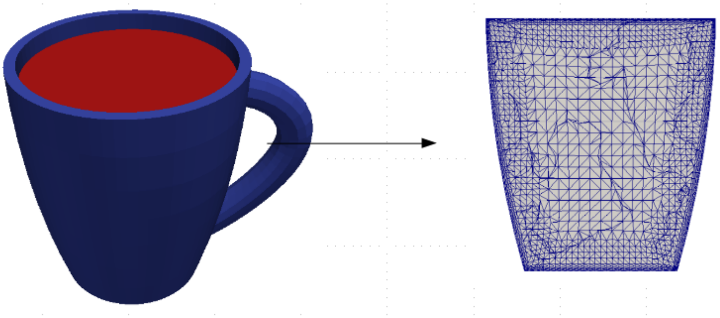
Let's take a look at our cup!

- **Question**: How much is the coffee cooled down when you hold the cup in the cold wind of 0°C and a wind speed of 1.0m/s



Let's take a look at our cup!

- Only the fluid is treated first



Let's take a look at our cup!

- Do we need turbulence?

$$\mathrm{Re} = \frac{U \cdot L}{\nu} = \frac{1 m/s \cdot 0.05 m}{0.3 \cdot 10^{-06} m/s^2} = 16666 \tag{2}$$

- Yes weed need turbulence.
- Turbulence model $\rightarrow$ kOmegaSST (wallbounded)
- `gedit constant/RASProperties`

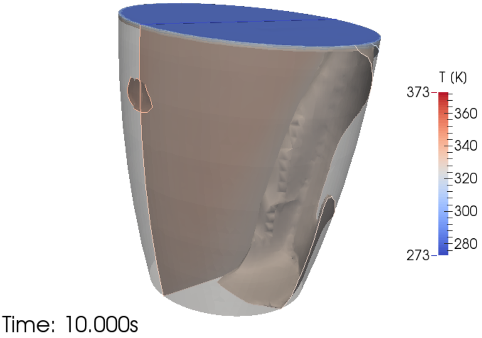
```
 simulationType RAS;
RAS
{
 RASModel kOmegaSST;
 turbulence on;
 printCoeffs on;
}
```

Let's take a look at our cup!

- We need Prandtl numbers for coffee
- Assuming hot water at 373K
  - $\mathrm{Pr} = 1.75$
  - Turbulent Prandtl number Prt ?
  - Normally a dynamic calculation!
  - Here: fixed at $\mathrm{Pr}_t = 0.9$
- Please remember: turbulent Prandtl number is not a constant
- Varies through the boundary layer!
- Set the value in `constant/transportProperties`

Get the simulation started!

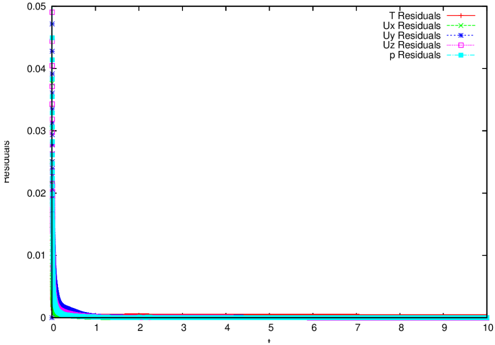
- `foamJob -parallel buoyantBoussinesqPimpleFoam`
- Result after 10sec



- Mean temperature using paraview volume integration 62.4°C

Analyze your results

- Mistakes may occur. Any ideas?
- Look at the residuals



- Ok, high residuals within the first time step! Smaller timesteps at the beginning of the simulation!

Analyze your results

- Mistakes may occur. Any ideas?
- Look at the mesh resolution for heat transfer analysis
- Remember the theory of a flate plate

$$\mathrm{Re}_l = \frac{U \cdot L}{\nu} = \frac{1 m/s \cdot 0.025 m}{0.3 \cdot 10^{-06} m/s^2} = 16666 \tag{3}$$

$$\frac{\delta_h}{L} = 5.0 \mathrm{Re}_l = 0.0173 m \tag{4}$$

$$\delta_h = 0.4 \cdot 10^{-03} m \tag{5}$$

- Let's check our mesh!

Analyze your results

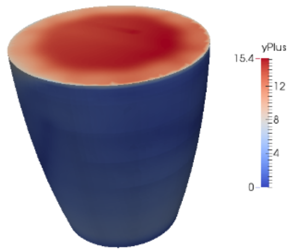
- `buoyantBoussinesqPimpleFoam -postProcess -func yPlus -latestTime`
  ```
  Patch 0 named cup_fluid_surface, wall-function
  nutLowReWallFunction, y+ :  min:  8.21955 max:
  15.8498 average:  13.0949
  Patch 1 named cup_fluid_wall, wall-function
  nutLowReWallFunction, y+ :  min:  0.287126 max:
  7.86749 average:  3.2015
  ```
- Not good, we need to generate a finer mesh!
- Also remember the correlation of thermal and hydraulic boundary layer

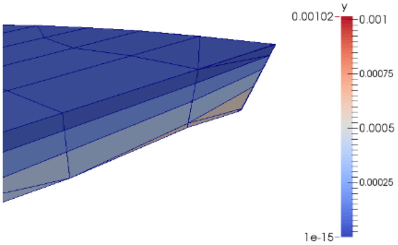
$$\frac{\delta_h}{\delta_t} = \mathrm{Pr}^{1/3} \tag{6}$$

- We need to be finer at the coffee surface!
- Fields of y and yPlus are written to the time folder

Analyze your results

- Mesh resolution





- Mesh is too coarse near the wall!

Analyze your results

- Now you have the choice:
  1. Generate a finer mesh.
  2. Application of wall functions.
- OpenFOAM gives us a wallfunction called `alphatJayatillekeWallFunction`
- Application of the wallfunction to obtain the turbulent thermal conductivity at the wall to ensure realistic heat flux

$$\mathrm{alpha}_t = \frac{\nu}{\mathrm{Pr}} + \frac{\nu_t}{\mathrm{Pr}_t} \tag{7}$$

Analyze your results

- OpenFOAM gives us a wallfunction called `alphatJayatillekeWallFunction`
-
```
 cup_fluid_surface
  {
     type alphatJayatillekeWallFunction;
     Prt 0.9;
     value uniform 0;
  }
  cup_fluid_wall
  {
     type alphatJayatillekeWallFunction;
     Prt 0.9;
     value uniform 0;
  }
```

Get back starting the simulation

- `foamJob -parallel buoyantBoussinesqPimpleFoam`
- Result after 10sec
- Mean temperature using paraview volume integration is now 58.4°C compared to previous 62.4°C
- Higher temperature gradients need to be captured using a finer mesh or by application of wallfunctions.

Remember

- a) Residuals
- b) Mesh resolution
- c) turbulent boundary conditions
- d) upwind schemes for velocity and temperature are too diffusiv! (see `system/fvSchemes`)
- application of finer and high quality meshes allow us to use second order schemes like `Gauss linear` or `linearUpwind` or blended schemes like `Gauss linearLimited`

Including buoyant forces

- Calculate temperature profiles in case of natural convection problems using Boussinesq approximation for density changing in stratified flows

$$\rho_{eff} = 1 - \beta(T - T_{ref}) \tag{8}$$

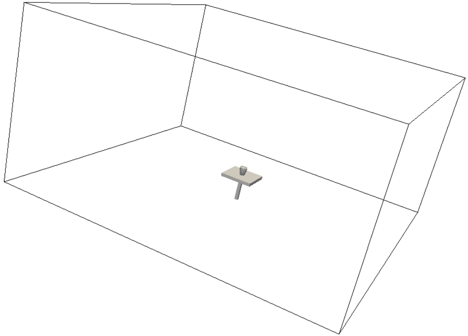
- where

ρ_{eff}	effective driving density
β	thermal expanison coefficient
T	temperature
T_{ref}	reference temperature

- Note:
  - Boussinesq approximation is only valid for $\beta(T - T_{ref} \ll 1.0)$
  - According to *Peric* the failure is below 1% for temperature differences of max. 2K for water and 15K for air

Including buoyant forces

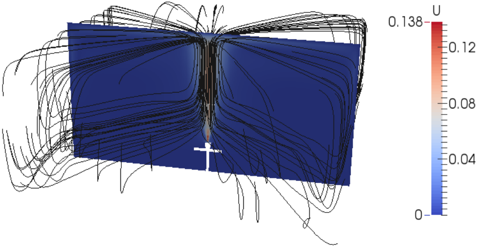
- Let's place our cup in a room on a small table



- `foamJob -parallel buoyantBoussinesqSimpleFoam`

Including buoyant forces

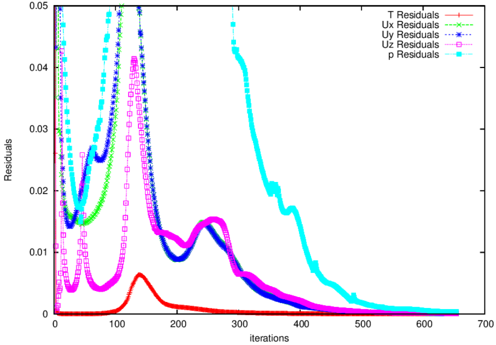
- Look at the results:



- Streamlines seems to be physically reasonable

Including buoyant forces

- But, take a look at the residuals!



- Seems to be ok, but remember that the convergence of steady simulations using Boussinesq approximation is hard to get.

Including buoyant forces

- Remember, that we have a temperature difference about 100K, Boussinesq approximation is not guilty!
  max 15K for air
- I have used upwind to get convergence.
  The applied interpolation schemes are to diffusive –> temperature disappears in the solution after a short range better use `bounded Gauss linearUpwind grad(U)`
- Better divergence schemes shows no convergence for this case :-)
- Use `buoyantBoussinesqPimpleFoam` if possible!

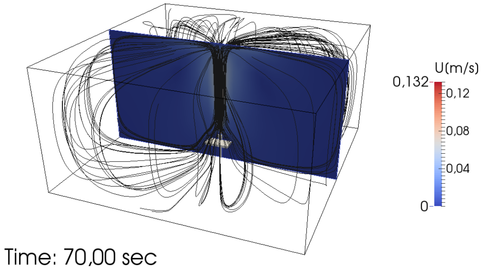
Including buoyant forces

- Run `foamJob buoyantBoussinesqPimpleFoam`
- Trying to get convergence for each timestep → good for initial heat transfer calculations
- `gedit log`

```
DILUPBiCG: Solving for T, Initial residual =
2.04079e-06, Final residual = 2.53797e-08, No
Iterations 1
DICPCG: Solving for p_rgh, Initial residual =
0.0287143, Final residual = 0.000274784, No Iterations
33
DICPCG: Solving for p_rgh, Initial residual =
0.00027785, Final residual = 2.6717e-06, No Iterations
53
```

Including buoyant forces

- Here is the result after 70sec of realtime



Compressible buoyant forces

- Since our coffee is too hot for the Boussinesq approximation we have to include the variation of material properties through pressure and temperature Relevant solvers are
- `buoyantSimpleFoam:`
  Steady-state, natural convection, compressible (sub-sonic), including radiation
- `buoyantPimpleFoam:`
  transient, natural convection, compressible(sub-sonic), including radiation

Compressible buoyant forces

- Changing of material properties requires underlaying thermophysics of the fluids
- Generally the thermophysics within OpenFOAM are a little bit of a mysterium since it is not well documented
- Let's bring light into the darkness
- Thermophysical properties for each case are defined in `constant/thermophysicalProperties`
- All models are located under `$FOAM_SRC/thermophysicalModels`
  - Fluid and solid properties (water, air)
  - Mixture and pre-definitions for combustion (really complicated ....)

Thermophysical models

- Thermomodels are the basis for determination of all material quantities
- Most of the models are implemented for combustion simulations since the temperature and pressure variations are enormously
- Models needed for heavy reactions are based on compressibility
- For heat transfer analysis **density** based models are preferable
- Otherwise phase changing is present which requires VOF methods including a fast interface capturing (see Level Set methods, big pain for unstructured meshes ...)

Thermophysical models

- `gedit constant/thermophysicalProperties`

```
thermoType
{
    type heRhoThermo;
    mixture pureMixture;
    transport const;
    thermo hConst;
    equationOfState perfectGas;
    specie specie;
    energy sensibleEnthalpy;
}
```

Thermophysical models

- Types of thermo class

`hePsiThermo`	General thermophysical model calculation based on compressibility $\psi = 1/(RT)$ Only gas
`hRhoThermo`	General thermophysical model calculation based on density ρ Gas, liquid, solids
`hSolidThermo`	Only solids

Thermophysical models

- Let's look for the air
- `gedit constant/thermophysicalProperties`

```
thermoType
{
    type heRhoThermo;
    mixture pureMixture;
    transport polynomial;
    thermo hPolynomial;
    equationOfState icoPolynomial;
    specie specie;
    energy sensibleEnthalpy;
}
```

Thermophysical models

- Let's look for the air
- `gedit constant/thermophysicalProperties`

```
mixture
{
// coefficients for air
    specie
  {
    nMoles 1;
    molWeight 28.85;
    }
    equationOfState
    {
    rhoCoeffs<8> ( 4.0097 -0.016954 3.3057e-05
-3.0042e-08 1.0286e-11 0 0 0 );
    }
```

Thermophysical models

- Let's look for the air
- `gedit constant/thermophysicalProperties`

```
thermodynamics
   {
   Hf 0;
   Sf 0;
   CpCoeffs<8> ( 948.76 0.39171 -0.00095999 1.393e-06
-6.2029e-10 0 0 0 );
   }
  transport
   {
  muCoeffs<8> ( 1.5061e-06 6.16e-08 -1.819e-11 0 0 0 0
0 );
   kappaCoeffs<8> ( 0.0025219 8.506e-05 -1.312e-08 0 0
0 0 0 );
   }
```

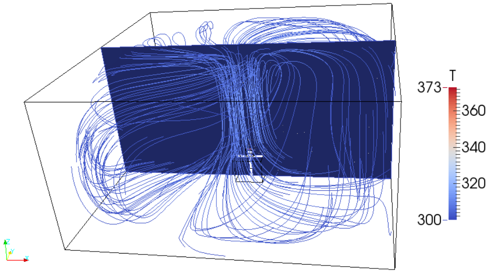
Thermophysical models

- Just make a small mistake to see which combination is possbile!

```
thermoType
{
    type heRhoThermo;
    mixture pureMixture;
    transport polynomial;
    thermo hPolynomial;
    equationOfState icoPolynomial;
    specie bananas;
    energy sensibleEnthalpy;
}
```

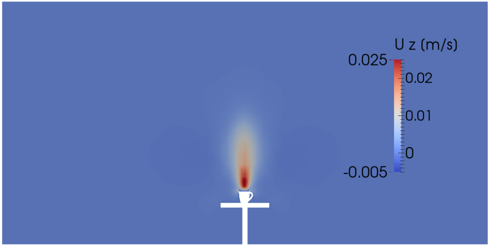
Run the compressible case

- Now we are able to run the simulation with changing material parameters
- `foamJob -parallel buoyantSimpleFoam`



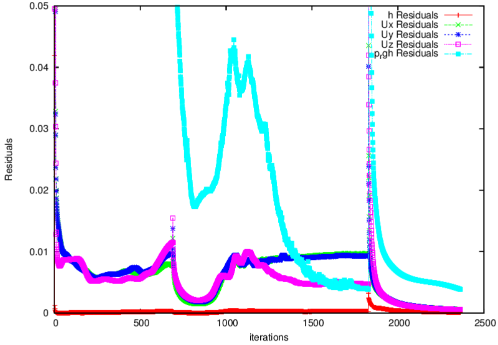
Run the compressible case

- Now we are able to run the simulation with changing material parameters
- `foamJob -parallel buoyantSimpleFoam`



Run the compressible case

- Keep care of the residuals



- Large residuals $\rightarrow$ hard to get convergence for steady simulations.
- Better use unsteady solver `buoyantPimpleFoam`

Case Setup

- Let's get to interesting stuff
- Including solids and more fluids in the analysis
- Names of the regions are defined in the file `constant/regionProperties`
- For our case:

```
regions
  (
   fluid (air coffee)
   solid (cup)
  );
```

Case Setup

- Eeach region properties are defined separately in the folders `0,constant,system`
- All other parameters for each region are defined in the region folders (e.g. `ls system/air`)
- A useful tool to setup the simulations: `changeDictionaryDict`
- Initialize the start fields for e.g. the region air `changeDictionary -region air`
- However be careful, empty fields are required

Case setup

- gedit 0/air/T

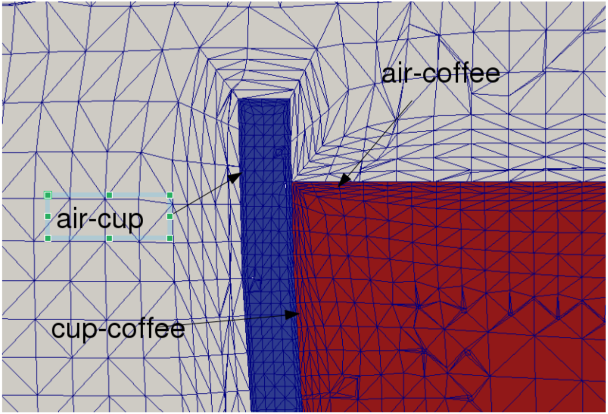
```
air_cup
    {
    type
compressible::turbulentTemperatureCoupledBaffleMixed;
    Tnbr T;
    kappa fluidThermo;
    kappaName none;
    value uniform 300;
    }
```

- Additional multiple layers with different thermal resistances can be specified at the interface:

```
thicknessLayers (1e-3);
kappaLayers (5e-4);
```

Case setup

- Lets's look at our interfaces:



Edit mesh setup

- Entry in the `snappyHexMeshDict`
- Example: `refinementSurfaces`

```
zone1
{
 level (0 0);
 faceZone faceZone1;
 cellZone cellZone1;
 cellZoneInside inside;
 boundary internal;
}
```

Case setup

- Coupling is based on nearest neighbor search!
- So please be careful to couple meshes with totally different mesh resolutions at the wall
- Otherwise the interpolation will give bad results
- Also remember, that the heat fluxes are not strictly conservative
- Too strong differences in the mesh resolution will induce heat sinks or heat source at the coupled patches

Case setup

- `gedit constant/air/polyMesh/boundary`

```
air_cup
  {
   type mappedWall;
   sampleMode nearestPatchFace;
   sampleRegion cup;
   samplePatch cup_air;
   nFaces 3307;
   startFace 616900;
  }
```

Run the CHT Case

- After the long road of setting up the case
- `decomposePar -allRegions`
  `foamJob -parallel chtMultiRegionFoam`
- After finish the simulation
- `paraFoam -touchAll`
- `paraview`

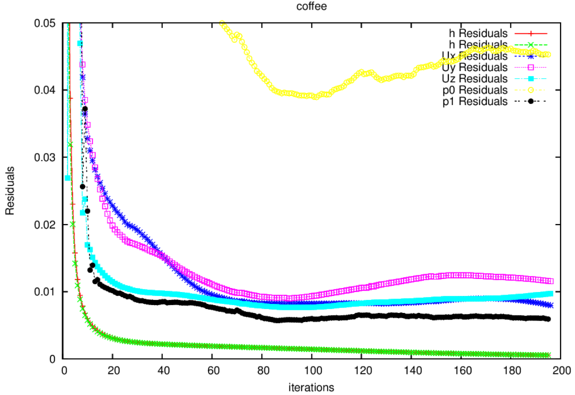
Analyze the results

- Let's have look what our alu cup says

- Your hand will be quite hot after 1 sec :-)

Analyze the results

- Check the residuals!



- Not good for the coffee fluid.

Analyze the results

- Use `potentialFoam` to get initial flow fields
- Use strong under relaxation for *p_rgh* and *h*
- Especially for heat transfer the temperature range is enlarged for in areas of bad cells or high velocity gradients
- Easy way to limit the temperature range is to use the very comfortable `fvOptions` method
- `fvOptions` can be added individually to the solver (e.g. porosity, ..)
- No need to recompile and adopt solver properties
- Located `$FOAM_SRC/fvOptions`

Stabilize the results

- gedit system/air/fvOptions

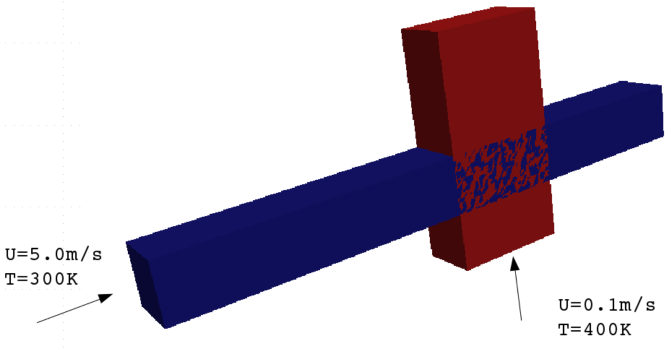
```
temperature_corrections
  {
   type limitTemperature;
   active yes;
   selectionMode all;
      Tmin 300;
      Tmax 373;
  }
```

Using `fvOptions`

- OpenFOAM gives us the following possibilities
  - `constantHeatTransfer`
    Constant heat transfer coefficient, need Area to Volume ratio (AoV)
  - `variableHeatTransfer`
    Calculates heat transfer coefficient using Nusselt number correlation
    $Nu = a * pow(Re, b) * pow(Pr, c)$
  - `tabulatedHeatTransfer`
    Calculates heat transfer coefficient using a predefined 2D table for heat transfer coefficient and velocity
- Interpolation of enthalpy h between each fluid region

Using `fvOptions`

- Let's solve the heat exchange between to cross streams of water and air



Using fvOptions

- The coupling is defined in system/air/fvOptions
- gedit system/air/fvOptions

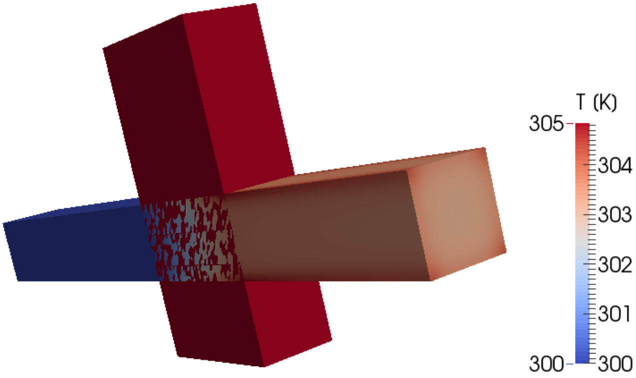
```
air_water
   {
  type constantHeatTransfer;
  active on;
  selectionMode mapRegion;
  interpolationMethod cellVolumeWeight;
  nbrRegionName water;
  master true
...
```

Using `fvOptions`

- We have to provide the Area of Volume ratio (AoV)
- `gedit 0/air/AoV`
- And the constant heat transfer coefficient
- `gedit 0/air/htcConst`
- `foamJob chtMultiRegionSimpleFoam`

Using `fvOptions`

- Look at the results



Using `fvOptions`

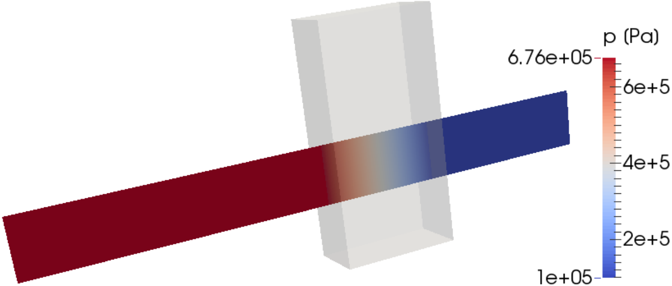
- However, the regions do not only interact through heat transfer
- Flow resistance due to e.g. heat exchanger pipes is present inducing a pressure drop
- Without modeling each pipe the flow resistance is included using porosity models
- OpenFOAM uses Darcy-Forchheimer law to calculate pressure drop

$$S_i = -[\mu d_i + 0.5\rho |u_i| f_i] u_i \tag{9}$$

- Please note, that the porosity can be defined for a cellZone (`explicitPorositySource`) or a region (`interRegionExplicitPorositySource`)

Using `fvOptions`

- If we add the porosity we get pretty physical results inside complex heat exchangers



- There are more fvOptions available
- see `$FOAM_SRC/fvOptions`

Basic background

- Radiation is very important and is often not considered
- Interaction of different devices in respect of thermal radiation is basis of thermal problems
- Throw radiation heat transfer beside will often lead to wrong physical results a
- Radiation heat transfer takes place in form of electromagnetic waves
- Wave length for heat transfer: $0.8 - 400 \mu m$ (ultrared)
- At higher temperatures, the amount of visible radiation is larger and can be seen e.g. lightning bulb

Basic background

- With increasing temperatures the intensity of heat radiation increases e.g. the human body radiates continuously about 1000W in a vacuum
- (note: no media is required for thermal radiation)
- From surrounding walls the human adsorbs thermal energy of about 900W
- So the typical loss of a non-working human is about 100W
- Electromagnetic waves can be adsorbed, reflected or transmitted according to the surface properties

$$\epsilon + \tau + \rho = 1 \tag{10}$$

- Coefficients depend also on wave length

Basic background

- For simplification a black body is introduced
  - All waves are adsorbed
  - Waves are emitted with maximum of intensity
- Emission coefficient for a black body is $\epsilon = 1$
- Law of Kirchhoff $\epsilon = \alpha$

Basic background

- The emission for a black body is independent of the wave length and solid angle
- Stephan-Boltzmann-law for hemispheric thermal radiation

$$Q/A = \epsilon\sigma T^4 \qquad \sigma = 5.6696 \cdot 10^{-8} W/m^2 K^4 \tag{11}$$

- Remember: include radiative heat transfer when the radiant heat flux, is large compared to the heat transfer rate due to convection or conduction

$$q_{rad} = \sigma(T^4_{max} - T^4_{min}) \tag{12}$$

Basic background

- OpenFOAM gives us three radiation models to calculate the heat fluxes
  - P1 model
  - fvDOM (finite volume discrete ordinates model)
  - viewFactor model
- We don't have time to review the models!
- But let us take a closer look

Decision of radiation model

- Indicator is the optical length $a * L$ where L is typical length scale and a absorption coefficient
- If $a * L >> 1$ then use P1 model
- Otherwise if $a * L < 1$ use fvDOM
- Since fvDOM also captures the large optical length scales it is the most accurate model
- P1 model tends to overpredict the heat flux
- fvDOM consumes a lot of CPU power since it solves the transport equation for each direction
- fvDOM can handle non gray surfaces (dependence of the solid angle is included)
- viewFactor is used if non participating mediums are present (space craft, solar radiation)

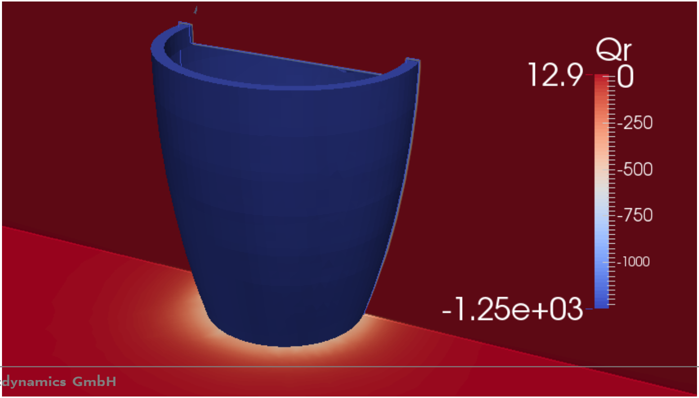
Get the case started

- gedit constant/radiationProperties

```
radiation on;
radiationModel P1;
// Number of flow iterations per radiation iteration
solverFreq 1;
absorptionEmissionModel constantAbsorptionEmission;
constantAbsorptionEmissionCoeffs
{
absorptivity absorptivity [ m^-1 ] 0.5;
emissivity emissivity [ m^-1 ] 0.5;
E E [ kgm^-1s^-3 ] 0;
}
scatterModel none;
sootModel none;
```

Get the case started

- We have to define the incident radiation field G for the P1 model
- And the field for radiation intensity I in case of the fvDOM model
- Let's look at the radiative heat flux Qr for the P1 model



Get the case started

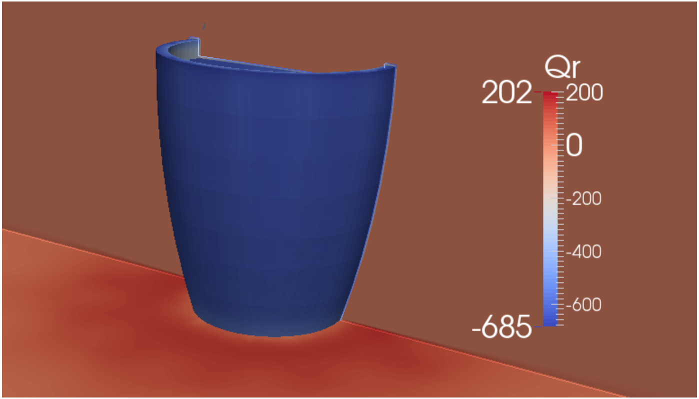
- Properties for the fvDOM

```
nPhi 3; // azimuthal angles in PI/2 on X-Y.(from Y to X)
nTheta 4; // polar angles in PI (from Z to X-Y plane)
convergence 1e-3; // convergence criteria for radiation
iteration
maxIter 10; // maximum number of iterations
cacheDiv false; //only for upwind schemes
```

- Hence for 4 Octants this gives us 48 equations for the intensity
- To get a numerical stable solution, a maximum iteration of 10 is defined
- Very time consuming: 480 Iterations per timeStep
- Thus only every 10 iterations the number of equations are solved (`solverFreq 10`)

Get the case started

- Radiative heat flux for the fvDOM



Outcome

- FvDOM model much more physical
- P1 model overpredict heat flux at cup and table surface
- Remember the optical length a*L!
- Radiative heat transfer from the hot cup to cold table has a fairly small
- length scale –> small optical length –> fvDOM
- FvDOM requires large CPU resources

Thank you very much!

Dr.-Ing. Johann Turnow
Email: johann.turnow@silentdynamics.de
Tel.: +49 381 36 76 84 11

silentdynamics GmbH
http://www.silentdynamics.de