interPhaseChangeFoam

This assignment is solved in OpenFOAM 2.0.x.

How to apply a heat source to interPhaseChangeFoam to evaporate water.
In this presentation will go through the following steps:

- Short description of interPhaseChangeFoam.
- Add temperature dependence - The idea.
  - August-Roche-Magnus formula.
- Copy the solver.
- Add temperature dependence - The implementation.
  - Add temperature field to interPhaseChangeFoam.
  - Include temperature dependent phase change in interPhaseChangeFoam.
    - Create an extra volume scalar field for speed optimization.
- Simple test case.

interPhaseChangeFoam

In *interPhaseChangeFoam.C* the solver is described as follows:

> *Solver for 2 incompressible, isothermal immiscible fluids with phase-change (e.g. cavitation). Uses a VOF (volume of fluid) phase-fraction based interface capturing approach.*
>
> *The momentum and other fluid properties are of the "mixture" and a single momentum equation is solved.*
>
> *The set of phase-change models provided are designed to simulate cavitation but other mechanisms of phase-change are supported within this solver framework.*
>
> *Turbulence modelling is generic, i.e. laminar, RAS or LES may be selected.*

Merkle Mass Transfer Model

Merkle is the simplest of the cavitation algorithms implemented.

Liquid $\rightarrow$ vapor:

$$\dot{m}^- = \frac{C_\mathrm{v}\rho_\mathrm{v}}{\frac{1}{2}\rho_\mathrm{l}U_\infty^2 t_\infty}\alpha\min(0, p - p_\mathrm{Sat}) \tag{1}$$

Vapor $\rightarrow$ liquid:

$$\dot{m}^+ = \frac{C_\mathrm{c}}{\frac{1}{2}U_\infty^2 t_\infty}(1-\alpha)\max(0, p - p_\mathrm{Sat}) \tag{2}$$

where:
C_c, C_v, U_∞ and t_∞ are empirical constants based on the mean flow,
ρ_l and ρ_v are the density of the liquid and vapor,
p is the pressure,
p_{Sat} is the vaporisation pressure.

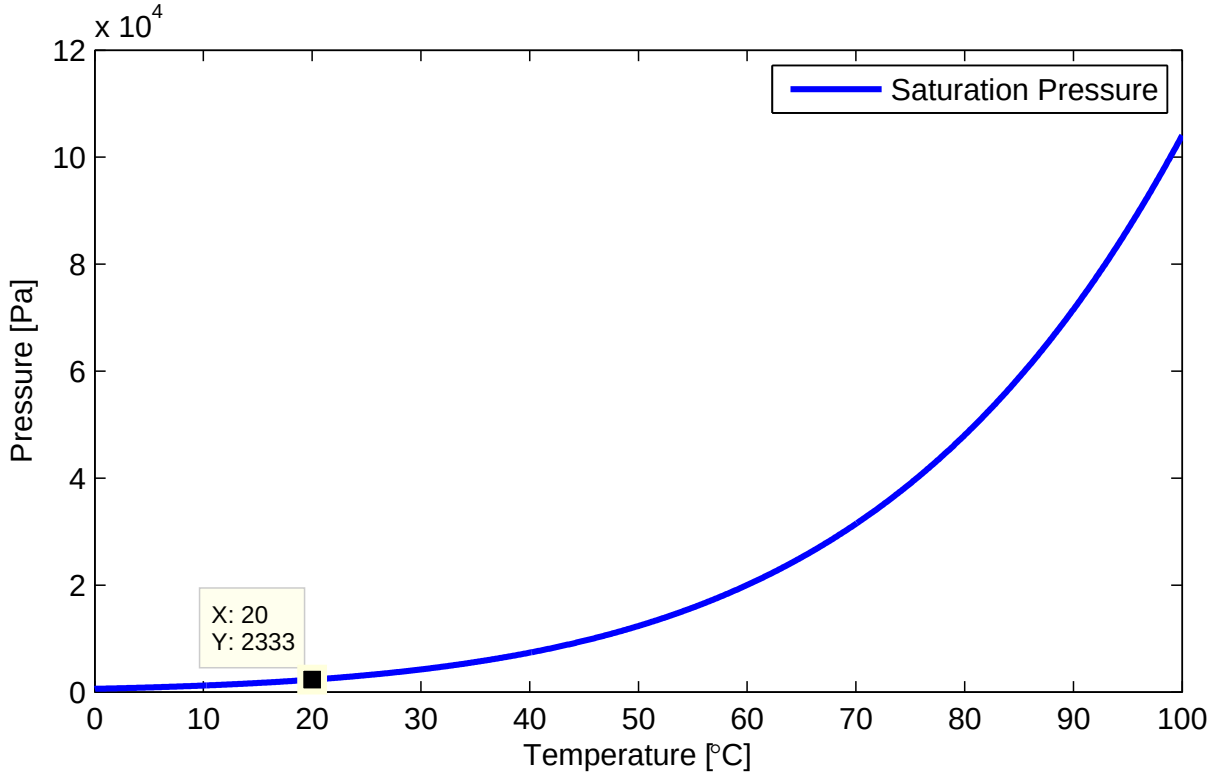
Total mass transfer rate (positive for vapor $\rightarrow$ liquid):

$$\dot{m} = \dot{m}^+ + \dot{m}^- \tag{3}$$

Add Temperature Dependence - The Idea

Total mass transfer rate (positive for vapor $\rightarrow$ liquid):

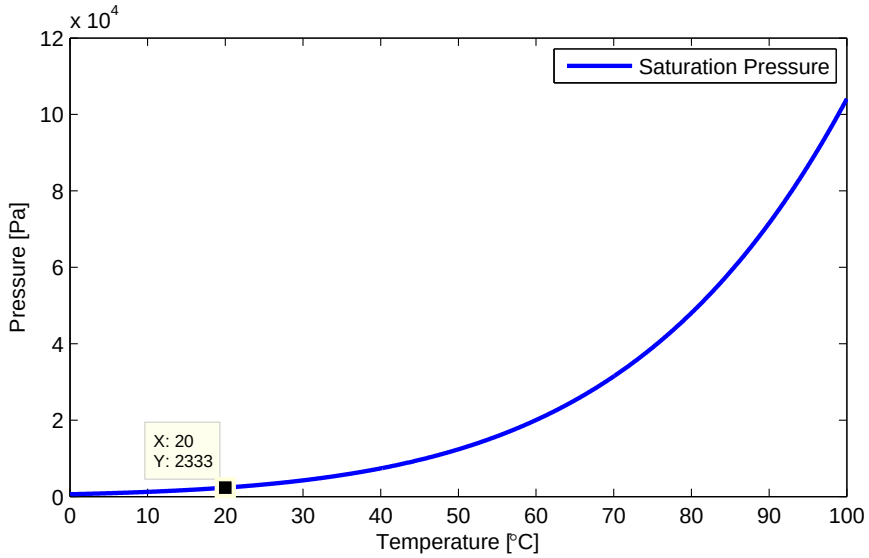
$$\dot{m} = \begin{cases} K_1 \alpha (p - p_{\text{Sat}}), & (p - p_{\text{Sat}}) < 0 \\ K_2 (1 - \alpha)(p - p_{\text{Sat}}), & (p - p_{\text{Sat}}) \geq 0 \end{cases}$$



August-Roche-Magnus formula

Implement temperature dependent saturation pressure using August-Roche-Magnus formula.

$$p_{\text{Sat}} \approx 610.94 \cdot \exp\left(\frac{17.625 \cdot T}{T + 243.04}\right), (T \text{ in } ^\circ C) \tag{4}$$



Need a temperature field.

Copy the solver 1/2

Before we start modifying the *interPhaseChangeFoam* we copy it to a new location and rename it to *myInterPhaseChangeFoam*.

```
mkdir -p $WM_PROJECT_USER_DIR/applications/solvers/multiphase
cd $WM_PROJECT_USER_DIR/applications/solvers/multiphase
cp -r $FOAM_SOLVERS/multiphase/interPhaseChangeFoam .
mv interPhaseChangeFoam myInterPhaseChangeFoam
cd myInterPhaseChangeFoam
mv interPhaseChangeFoam.C myInterPhaseChangeFoam.C
cp $FOAM_SOLVERS/multiphase/interFoam/correctPhi.H .
wclean
```

Find the line `#include "../interFoam/correctPhi.H"` in *myInterPhaseChangeFoam.C* and change it to `#include "correctPhi.H"`

Copy the solver 2/2

Edit the build files to fit the *myInterPhaseChangeFoam* solver. The file *Make/files* shall look like this:

myInterPhaseChangeFoam.C
phaseChangeTwoPhaseMixtures/phaseChangeTwoPhaseMixture/phaseChangeTwoPhaseMixture.C
phaseChangeTwoPhaseMixtures/phaseChangeTwoPhaseMixture/newPhaseChangeTwoPhaseMixture.C
phaseChangeTwoPhaseMixtures/Kunz/Kunz.C
phaseChangeTwoPhaseMixtures/Merkle/Merkle.C
phaseChangeTwoPhaseMixtures/SchnerrSauer/SchnerrSauer.C

EXE = $(FOAM_USER_APPBIN)/myInterPhaseChangeFoam

Compile the solver:

```
wmake
```

If everything worked correctly, the new solver binary should appear here:

```
ls $FOAM_USER_APPBIN
```

Add Temperature Field 1/2

Edit *createFields.H* by adding the following to the top of the file

```
Info<< "Reading transportProperties\n" << endl;
IOdictionary transportPropertiesDict
(
    IOobject
    (
        "transportProperties",
        runTime.constant(),
        mesh,
        IOobject::MUST_READ,
        IOobject::NO_WRITE
    )
);
dimensionedScalar DT
(
    transportPropertiesDict.lookup("DT")
);

Info<< "Reading field T\n" << endl;
volScalarField T
(
    IOobject
    (
        "T",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);
```

Add Temperature Field 2/2

Add temperature transport equation. Create a file named *TEqn.H* with the following lines:

```
{
    fvScalarMatrix TEqn
    (
        fvm::ddt(T)
        + fvm::div(phi, T)
        - fvm::laplacian(DT, T)
    );
    TEqn.solve();
}
```

And inset the line

```
#include "TEqn.H"
```

after the Pressure-velocity PIMPLE corrector loop, but before *runTime.write();* in *myInterPhaseChangeFoam.C*. Compile

```
wmake
```

Temperature Dependent Phase Change

Merkle Mass Transfer:

$$\dot{m} = \begin{cases} K_1\alpha(p - p_{\text{Sat}}), & (p - p_{\text{Sat}}) < 0 \\ K_2(1-\alpha)(p - p_{\text{Sat}}), & (p - p_{\text{Sat}}) \geq 0 \end{cases}$$

August-Roche-Magnus formula:

$$p_{\text{Sat}} \approx 610.94 \cdot \exp\left(\frac{17.625 \cdot (T - 273.15)}{T - 30.11}\right), (T \text{ in Kelvin})$$

Merkle Mass Transfer is used in a pressure loop and an α-loop in the main function. The temperature is adjusted outside these loops.
p_{Sat} only depends on T. To optimize the computation a p_{Sat}-Field will be created.

Create p_{Sat} Field

Open the file *createFields.H* and remove the line

```
const dimensionedScalar& pSat = twoPhaseProperties->pSat();
```

And include this the following lines in file after *volScalarField T* and *volScalarField p_rgh* but before *Creating phaseChangeTwoPhaseMixture*

```
volScalarField pSat
(
    IOobject
    (
        "pSat",
        runTime.timeName(),
        mesh,
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    p_rgh  // initial value will be overwritten by calcPSatField.H
);
#include "calcPSatField.H"
```

Update the p_{Sat} Field

Implement August-Roche-Magnus formula:

$$p_{\text{Sat}} \approx 610.94 \cdot \exp\left(17.625 \cdot \frac{T - 273.15}{T - 30.11}\right), (T \text{ in Kelvin})$$

Create the file *calcPSatField.H* with the following content:

```
{
    const dimensionedScalar t30_11("30.11", dimensionSet(0,0,0,1,0,0,0), 30.11);
    const dimensionedScalar t273_15("273.15", dimensionSet(0,0,0,1,0,0,0), 273.15);
    const dimensionedScalar t1("1", dimensionSet(0,0,0,1,0,0,0), 1);
    const dimensionedScalar p610_94("610.94", dimensionSet(1,-1,-2,0,0,0,0), 610.94);
        // dimensionSet( [kg], [m], [s], [K], [kg*mol], [A], [cd]), [kg/(m*S^2)]=[Pa]


    // August-Roche-Magnus formula
    pSat = p610_94 * exp(  17.625*(T-t273_15) / max(t1, T-t30_11) );
        //max(1,...) is included to avoid problems with devision by 0
}
```

Also update the new field each time the temperature field is re-calculated. Include the line

`#include "calcPSatField.H"`

In *myInterPhaseChangeFoam.C* after `#include "TEqn.H"`.

Remove stationary p_{Sat}

Remove the stationary p_{Sat}. Search for `psat` in the files
phaseChangeTwoPhaseMixtures/phaseChangeTwoPhaseMixture/phaseChangeTwoPhaseMixture.C
and
phaseChangeTwoPhaseMixtures/phaseChangeTwoPhaseMixture/phaseChangeTwoPhaseMixture.H

In the later file, replace

```
const dimensionedScalar& pSat() const
{
    return pSat_;
```

with

```
const volScalarField& pSat() const
{
const volScalarField& pSat=alpha1_.db().lookupObject<volScalarField>("pSat");
return pSat;
```

Compile

```
wclean
wmake
```

Test Case

Simple closed box with 90°C water. Heated on the right with 110°C hot wall.

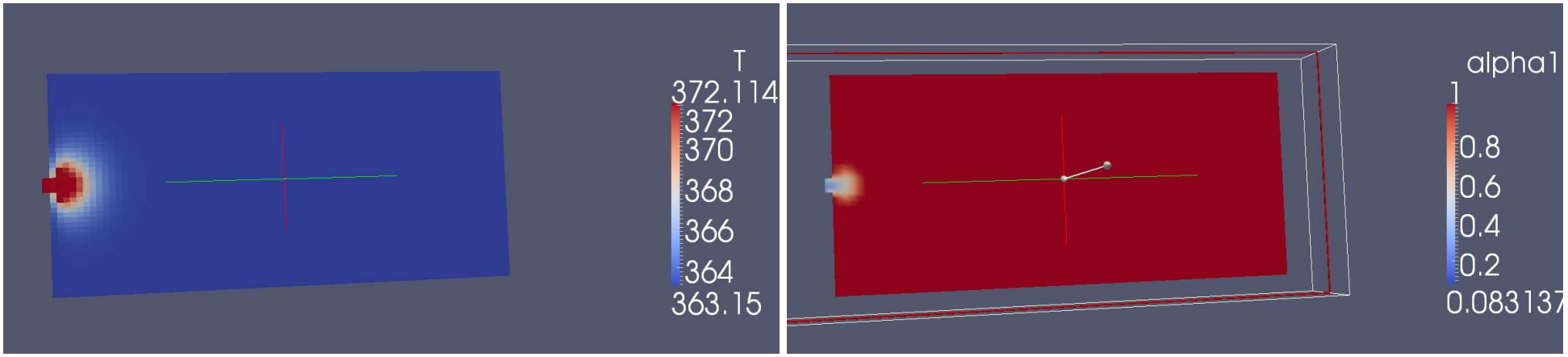


Figure 1: Left: Temperature. Right: Alpha