Description of reactingTwoPhaseEulerFoam solver with a focus on mass transfer modeling terms

Submitted by:
Thummala Phanindra Prasad,
Environmental Engineering Department,
Anadolu University of Technology,
Esksiehir, Turkey

This presentation is submitted as part of project work for the course "CFD with Open Source Software" , taught by Prof.Hakan at Chalmers University of Technology, Sweden

CHALMERS
UNIVERSITY OF TECHNOLOGY

<u>Submitted to:</u>

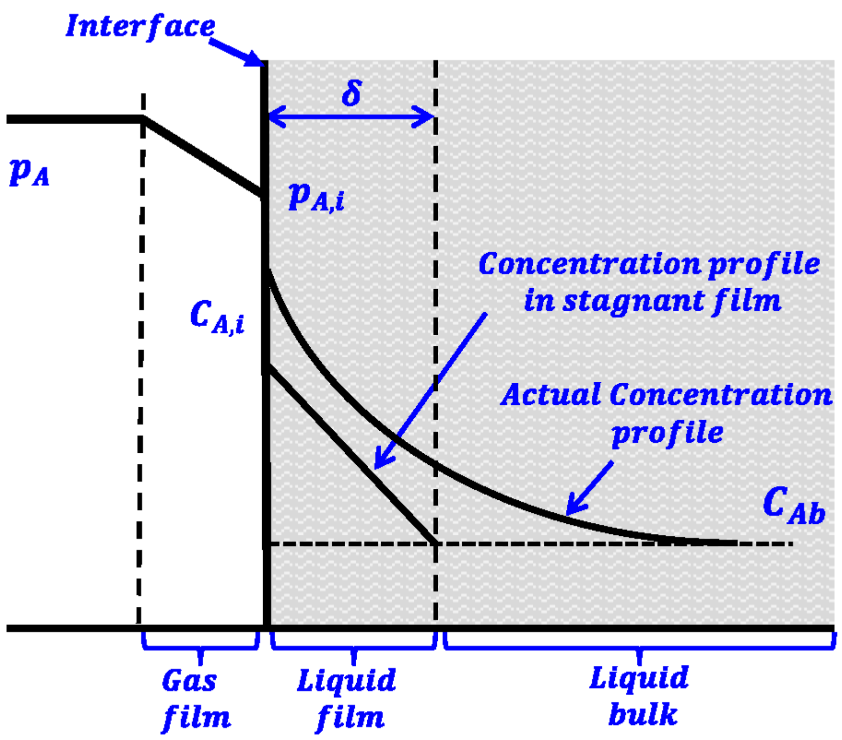
Prof. Hakan Nilsson,

Department of Applied Mechanics, Division of Fluid Dynamics, at Chalmers University of Technology in Sweden.

Overview

- Mass Transfer process
- Overview of the code
- Species transport equation
- Tutorial
- Limitations

Mass transfer process



The main difference between heat and mass transfer processes is that at the interface the temperature will remain same in both the phases but the concentration varies . It can be determined from VLE curves or any other equilibrium relations like Henry's law, Saturation model law etc…

Overview of the Source code

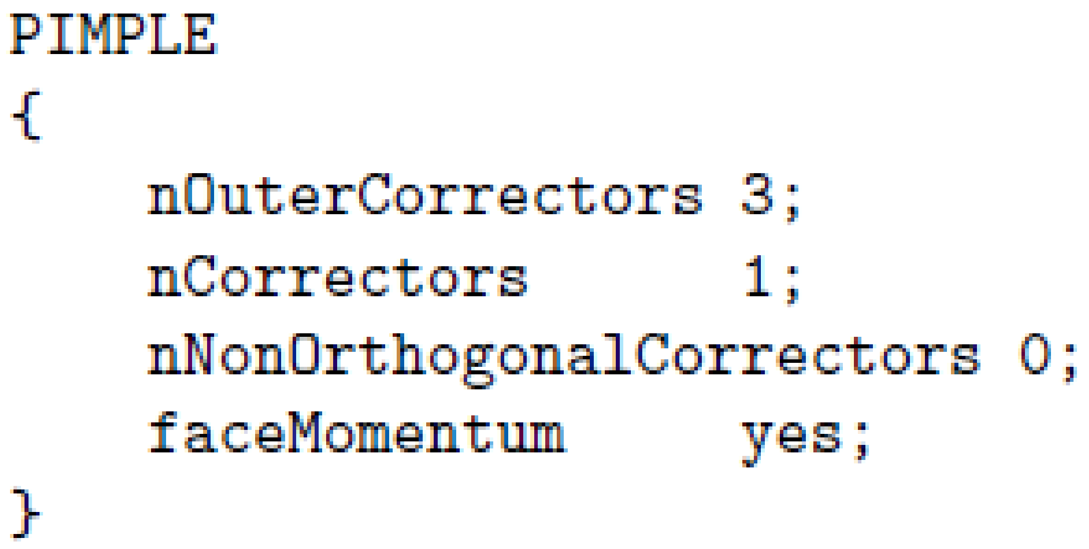
- The source code of the solver is located at:

$FOAM_APP/solvers/multiphase/reactingEulerFoam/reactingTwoPhaseEulerFoam/reactingTwoPhaseEulerFoam.C

- Face momentum formulation (#include pUf/createDDtU.H) is given in additional to ccollocated momentum formulation.
- Currently it is not clear if this face-based momentum equation formulation is preferable for all Euler-Euler simulations (comment from developers) and hence it must be activated explicitly

Overview of the Source code

- The facemomentum formulation can be activated by explicitly specifying in PIMPLE controls in *casedirectory/system/fvsolution* file as follows:



```
PIMPLE
{
    nOuterCorrectors 3;
    nCorrectors      1;
    nNonOrthogonalCorrectors 0;
    faceMomentum     yes;
}
```

Species transport equation

- In the reactingTwoPhaseEulerFoam.C file the species transport equation is given by :

  #include "YEqns.H”

- The location of this file can be found at:

$FOAM_APP/solvers/multiphase/reactingEulerFoam/reactingTwoPhaseEulerFoam/YEqns.H

- The header file connects to an auto pointer where the mass transfer rates table will be defined for the given species:

  autoPtr<phaseSystem::massTransferTable>

Species transport equation

- The definition of the auto pointer to mass transfer table can be found at:

$FOAM_APP/solvers/multiphase/reactingEulerFoam/phaseSystems/PhaseSystems/InterfaceCompositionPhaseChangePhaseSystem/InterfaceCompositionPhaseChangePhaseSystem.C

- Using the code in this file the amount of mass transfer of the species/phase as a whole is calculated.

- The term referring to the mass transfer rate is given by *dmdt*.

- The definition of mass transfer involved in other phase systems like HeatandMassTransferphasesystem and Thermal-PhaseChangephasesystem is one and same like this.

Species transport equation

- The default available mass transfer models are : Frossling and Spherical.
- Their definition can be found in:

$FOAM_APP/solvers/multiphase/reactingEulerFoam/interfacialCompositionModels/massTransferModels

- The Frossling equation is given by:

$$\mathrm{Sh} = 2 + 0.552 * Re^{1/2} * Sc^{1/3}$$

Species transport equation

Even though we explicitly mention Sc number for individual phases in phase properties file in case directory/constant folder, the Sc (Schmidt number) in Frossling mass transfer model implementation in Open Foam is calculated from Prandtl Number(Pr) and Lewis number(Le) as:

$$Sc = Le*Pr$$

Where the Le number is read from phase properties in case directory/constant folder and Pr is calculated from phase pair properties calculated from phasepar.C file located at:
$FOAM_APP/solvers/multiphase/reactingEulerFoam/phaseSystems/phasePair/phasePair/phasePair.C

Species transport equation

The way the mass transfer coefficient is derived from Sherwood number is bit tricky as explained here:

Let us define:

Rate of Mass transfer = RMT

mass transfer coefficient= MTC

interfacial area =ai

concentration gradient = dY

volume fraction of gas = α_g

sauter diameter = Ds

diffusivity = Df

Then, RMT = MTC* ai * dY ------(1)

For any multiphase flow the interfacial area in a cell is generally related to phase fraction as:

ai= 6 * α_g / Ds ---------(2)

Species transport equation

Combining (1) and (2) we get:

RMT = MTC *6* $\boldsymbol{\alpha_g}$***dY / Ds** --------(3)

The MTC is derived from Sh as: **MTC = Sh*Df/Ds** ----(4)

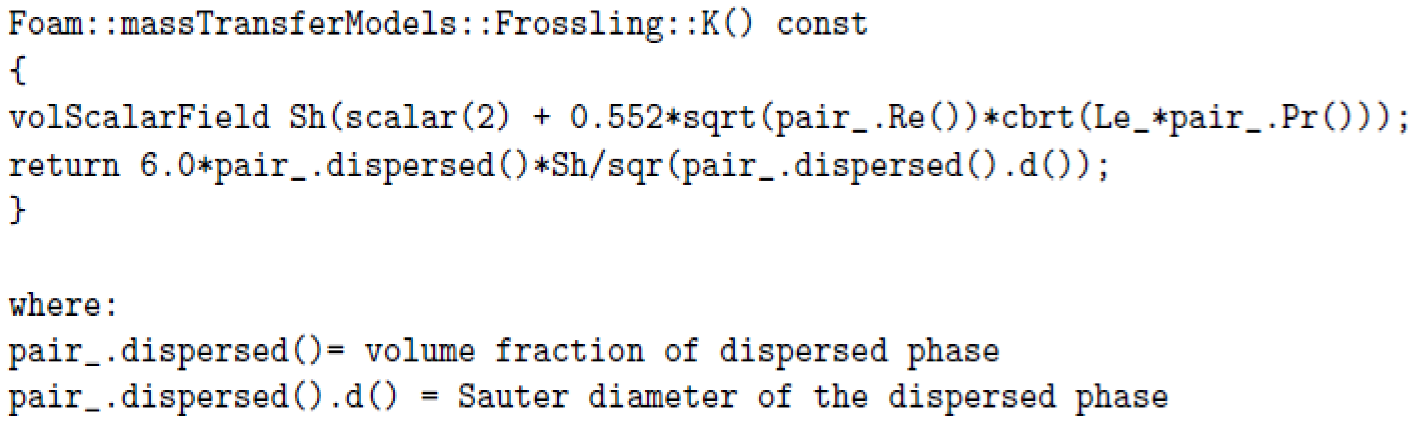
Combining (3) and (4) we get:

RMT = Sh * Df*6* $\boldsymbol{\alpha_g}$***dY /** $\mathbf{Ds^2}$ -------(5)

In the equation (5) , since the Df and dY are calculated as part of solution, all the remaining terms are combined and calculated in the MTC calculation it self and the MTC K() function returns :

$$\mathbf{6*Sh*\alpha_g / Ds^2}$$

Hence, the same can be seen from the Frossling equation code as return argument:

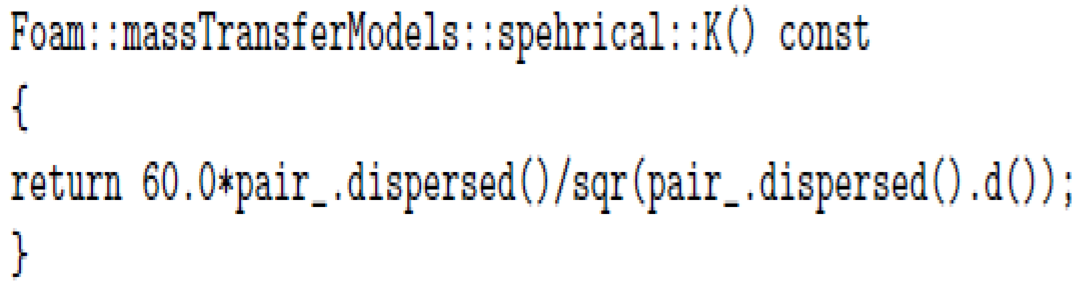


```
Foam::massTransferModels::Frossling::K() const
{
volScalarField Sh(scalar(2) + 0.552*sqrt(pair_.Re())*cbrt(Le_*pair_.Pr()));
return 6.0*pair_.dispersed()*Sh/sqr(pair_.dispersed().d());
}

where:
pair_.dispersed()= volume fraction of dispersed phase
pair_.dispersed().d() = Sauter diameter of the dispersed phase
```

Species transport equation

- Similarly, the Spherical mass transfer coefficient is based upon the assumption of laminar flow where Sh=10 and hence return statement will be, by substituting Sh=10 in (5) :

$$60*\alpha g/ Ds^2$$



```
Foam::massTransferModels::spehrical::K() const
{
return 60.0*pair_.dispersed()/sqr(pair_.dispersed().d());
}
```

Species transport equation

- Once the mass transfer terms are calculated as described above, the solution proceeds to source these *dmdt* term in the place *massTransfer[Y1[i].name()] of the equation and proceed for solving the equation of the code:

```
Y1iEqn == *massTransfer[Y1[i].name()]+ fvOptions(alpha1, rho1, Y1[i])
```

The left hand side of this equation is a species fraction equation which is created depending upon the phase model that has been selected. For example, if the phase model of gas is reactingPhaseModel then on reading the command the OpenFoam creates the equations for all individual species as described in:

$FOAM_APP/solvers/multiphase/reactingEulerFoam/phaseSystems/phaseModel/MultiComponentPhaseModel/MultiComponentPhaseModel.C

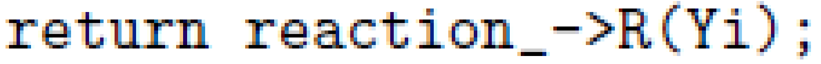
Species transport equation

```
return
(
    fvm::ddt(alpha, rho, Yi)
  + fvm::div(alphaRhoPhi, Yi, "div(" + alphaRhoPhi.name() + ",Yi)")
  - fvm::Sp(this->continuityError(), Yi)

  - fvm::laplacian
    (
        fvc::interpolate(alpha)
       *fvc::interpolate(this->turbulence().nut()*rho/Sc_),
        Yi
    )
 ==
    this->R(Yi)

  + fvc::ddt(residualAlpha_*rho, Yi)
  - fvm::ddt(residualAlpha_*rho, Yi)
);
```

Species transport equation

- The R(Yi) terms are contribution from reactive terms defined for reacting mixture and accessed by reacting phase model at:

$FOAM_APP/solvers/multiphase/reactingEulerFoam/phaseSystems/phaseModel/ReactingPhaseModel/ReactingEulerFoam.C



```
return reaction_->R(Yi);
```

where the term "reaction_" term is a table of species, their reactions and thermo information retrieved using "foamChemistryReader" key word when reacting mixture is used as a phase.
The source term R(Yi) is calculated using combustion models available for "rhoCombustionModel" setting through combustion properties.
They are called
upon when the fluid model (phase model) is "ReactingPhaseModel" as defined in:

Species transport equation

- The source term R(Yi) is calculated using combustion models available for "rhoCombustionModel" setting through combustion properties.
- They R(Yi)are called upon when the fluid model (phase model) is "ReactingPhaseModel" as defined in:

$FOAM_APP/solvers/multiphase/reactingEulerFoam/phaseSystems/phaseModel/phaseModel/phaseModels.C

- The combustion model intern refers to the Chemistry model setting which will be described in a seperate chemistry file corresponding to the phase.
- The chemistry models will generate the source terms R(Yi) and feed back to the species code. The details of the various chemistry models and solvers for chemistry can be found at:

$FOAM_SRC/thermophysicalModels/chemistryModel/chemistryModel/chemistryModel/chemistryModel.H

Tutorial

- For understanding the reactingTwoPhaseEulerFoam code let us use a tutorial named bubbleColumnEvaporatingReacting tutorial located in $FOAM_TUTORIALS folder
- To copy the tutorial type the following command:

```
cp -r
   $FOAM_TUTORIALS/multiphase/reactingTwoPhaseEulerFoam/RAS/b
   ubbleColumnEvaporatingReacting $FOAM_RUN
```

And open the tutorial directory:

```
cd $FOAM_RUN/bubbleColumnEvaporatingReacting
```

- The file structure is similar to any other standard tutorials and hence I’m not introducing it explicitly

Tutorial

- **Problem description**:

This tutorial describes how to pre-process, run and post-process a Water Gas Shift Reaction (WGSR)taking place in a 3D bubble column reactor. The reacting fluids are reacting gas (AIR,CO) and pure liquid water. The products formed were CO_2 and H_2 along with water vapor. The flow of phases were modeled with Eulerian-Eulerian Solver. Also, Separate species mass fraction transport equations were solved for all the individual species involved:CO,H_2O,AIR,CO_2,H_2. The tutorial helps in understanding the modeling of multiphase reactive fuids involving reaction, heat and mass transfer between phases.

Tutorial

- **Problem description**:

The geometry consists of a 3D block with a 0.15x0.1 m2 base and a length of 1m .The reactor is filled with water till height of 0.5m and assumed to have 0.1% dissolved air. Air enters from the bottom inlet at 0.1m/s. The system is initially assumed to be at 400K. The overall reaction involved is:

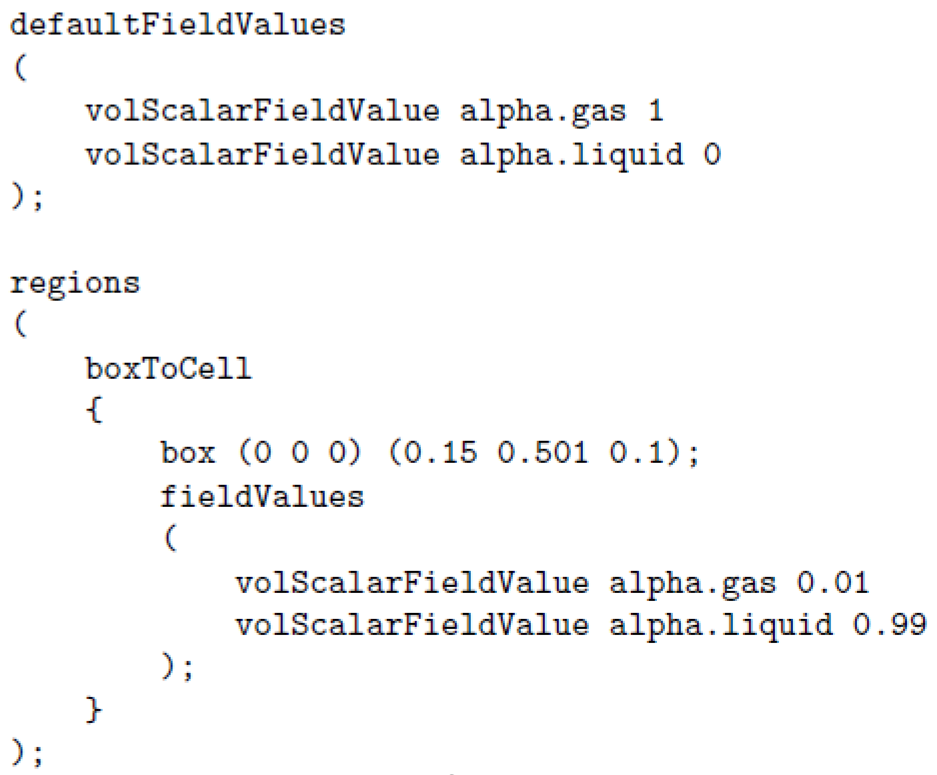
$$0.93CO + 0.24H2O \rightarrow 0.69CO2 + H2$$

Tutorial

- **Boundary and initial conditions:**
- The boundary conditions for the bubbleColumnEvaporatingReacting tutorial are very simple. All walls are modeled as adiabatic. The reacting gas enters from the bottom of the column at a speed of 0.1m/s. The AIR mass fraction in the inlet gas is 0.9 and that of CO is 0.1.

Tutorial

- The presence of water initially up to a height of 0.5m is declared by using setFieldsDict (given in system folder) functionality as given below:



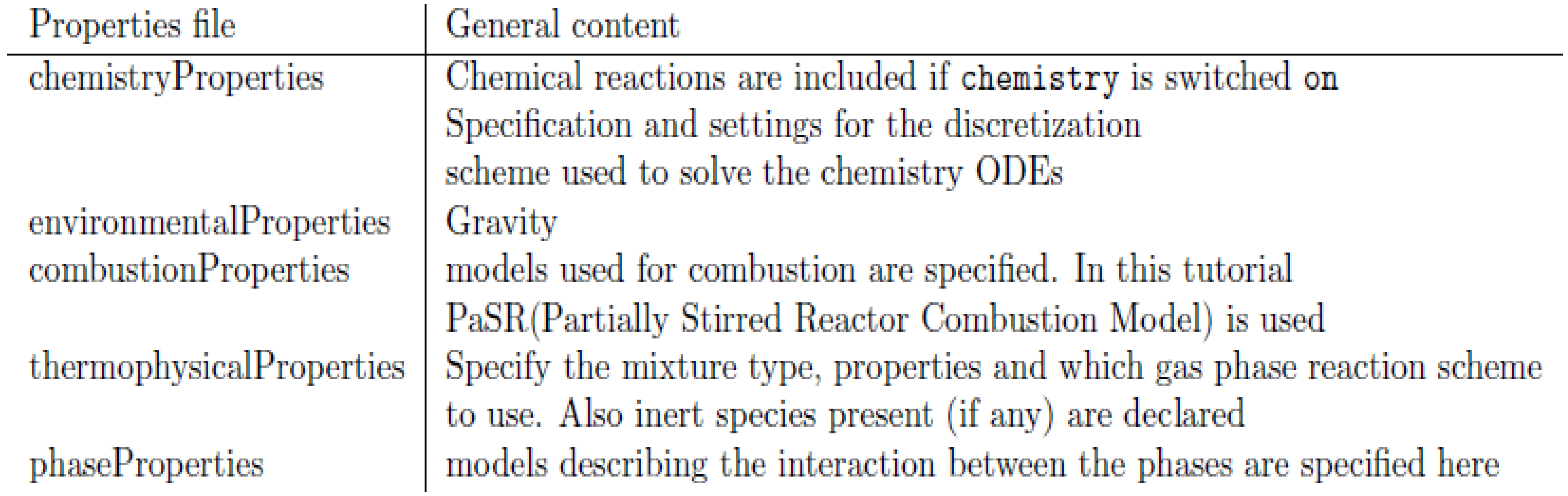
```
defaultFieldValues
(
    volScalarFieldValue alpha.gas 1
    volScalarFieldValue alpha.liquid 0
);

regions
(
    boxToCell
    {
        box (0 0 0) (0.15 0.501 0.1);
        fieldValues
        (
            volScalarFieldValue alpha.gas 0.01
            volScalarFieldValue alpha.liquid 0.99
        );
    }
);
```

Note:-For understanding the usage of setFieldsDict the user is recommended to look into the OpenFoam basic damBreak tutorial

Tutorial

- The initial specie mass fraction for all the species over the entire geometry field distribution can be found in the 0 folder file Ydefault file. It can be observed that the values were set to 0.
- **The Physical Properties:**

In the **/constant** directory the properties files for chemistry,environmental, combustion, reaction, combustion, RAS and thermophysical. All the properties are thoroughly described in the Open-FOAM user guide and are therefore not described here.

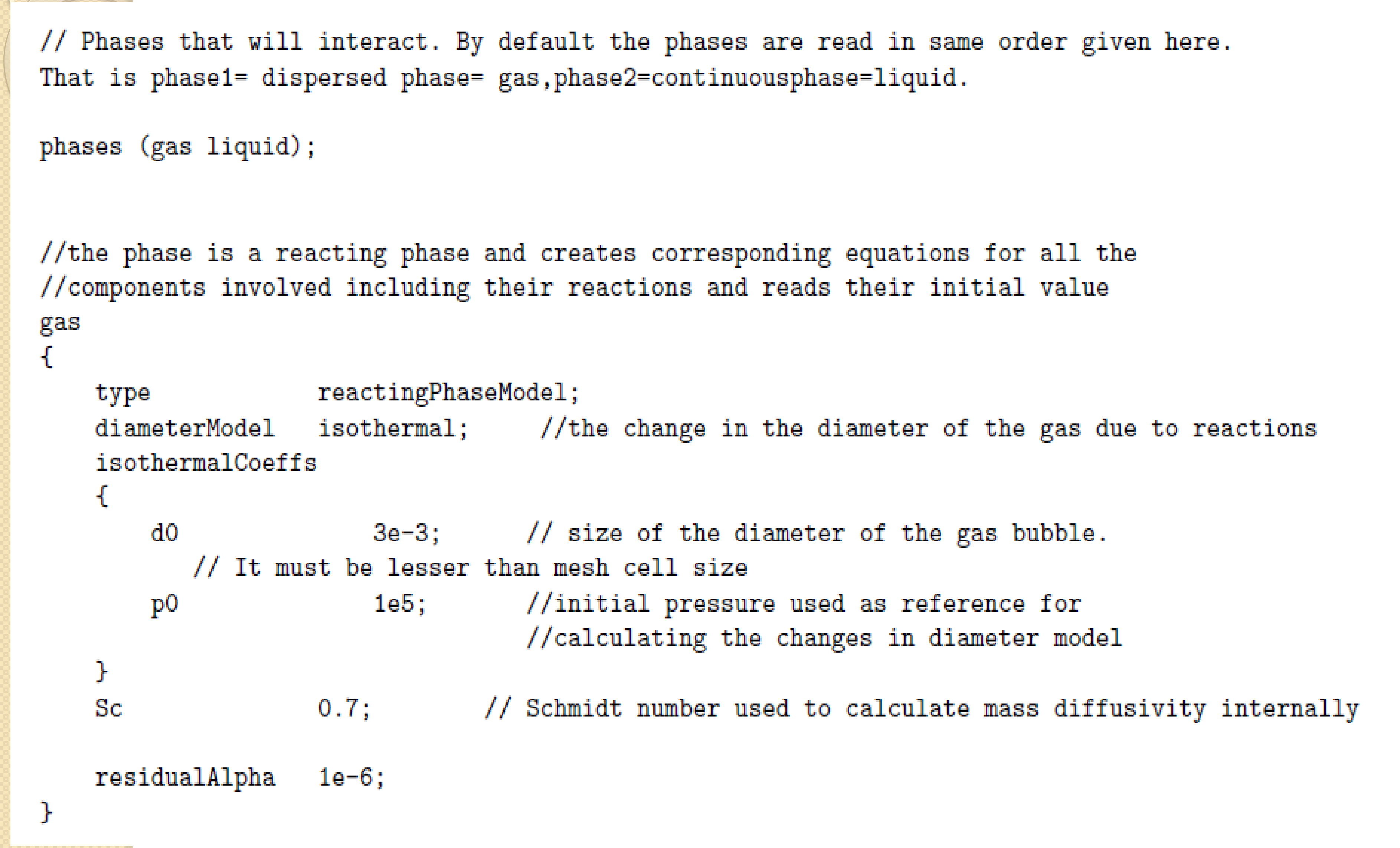


Properties file	General content
chemistryProperties	Chemical reactions are included if `chemistry` is switched `on` Specification and settings for the discretization scheme used to solve the chemistry ODEs
environmentalProperties	Gravity
combustionProperties	models used for combustion are specified. In this tutorial PaSR(Partially Stirred Reactor Combustion Model) is used
thermophysicalProperties	Specify the mixture type, properties and which gas phase reaction scheme to use. Also inert species present (if any) are declared
phaseProperties	models describing the interaction between the phases are specified here

Tutorial

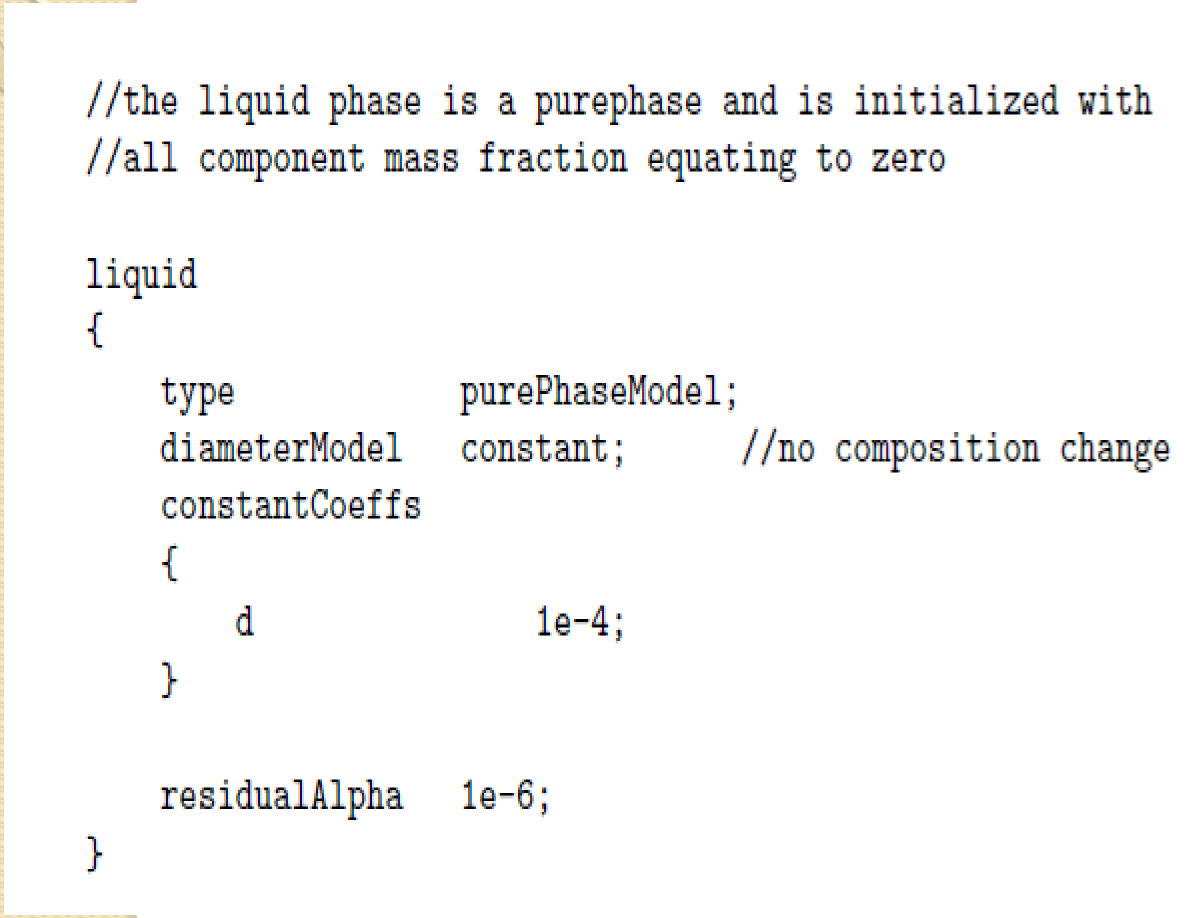
- The models specified in phaseProperties file will dictate the interaction between phases. In the current case, the reactions were confined to gas phase only and the liquid phase is assumed to be a purephase.
- Also, the phases were assumed to be a combination of small droplets (or bubbles) with fixed or variable diameters.
- The details of these diameter models are given in user guide and hence are not discussed in detail here.
- Some important observations regarding phaseProperties file:

Tutorial



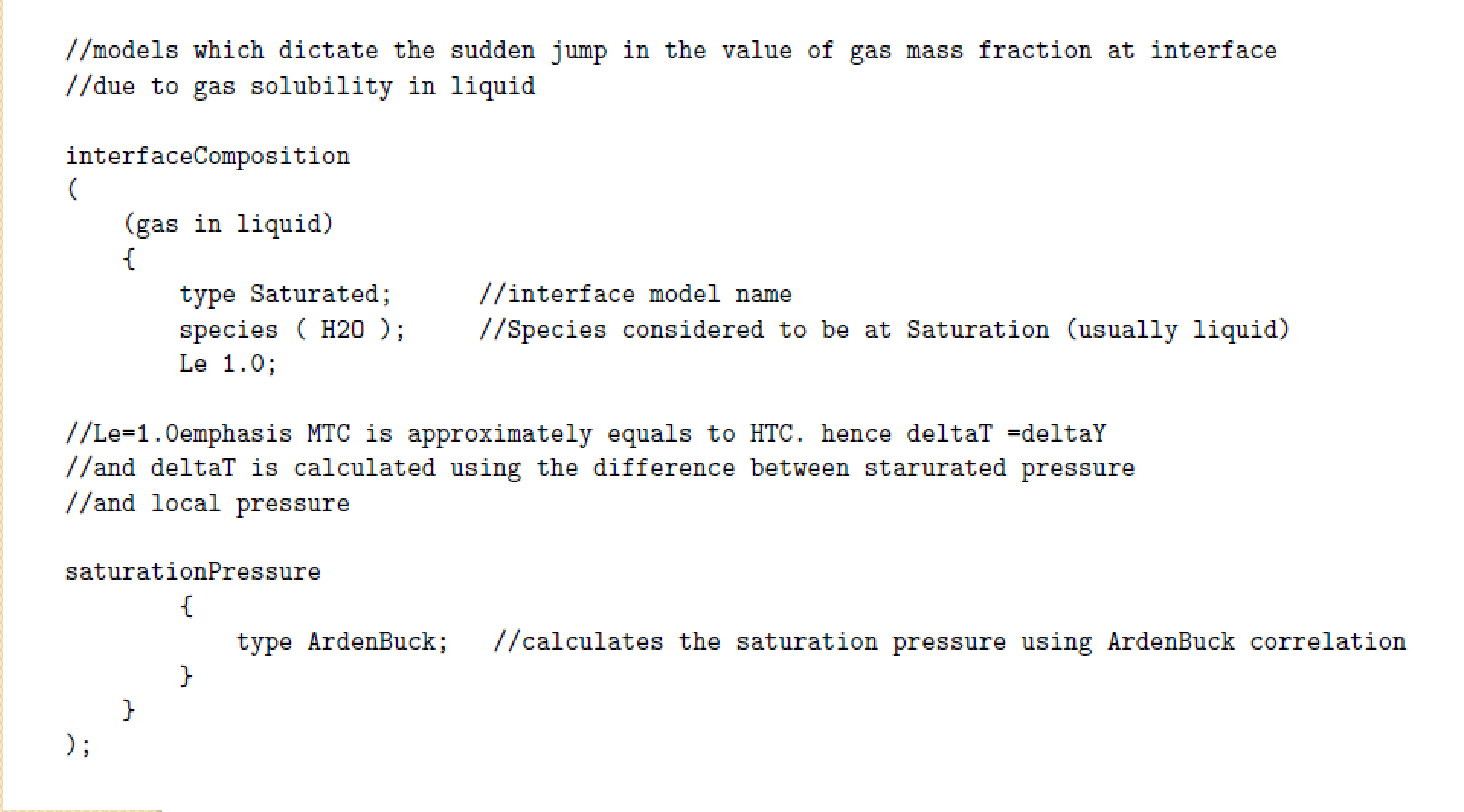
```
// Phases that will interact. By default the phases are read in same order given here.
That is phase1= dispersed phase= gas,phase2=continuousphase=liquid.

phases (gas liquid);



//the phase is a reacting phase and creates corresponding equations for all the
//components involved including their reactions and reads their initial value
gas
{
    type            reactingPhaseModel;
    diameterModel   isothermal;     //the change in the diameter of the gas due to reactions
    isothermalCoeffs
    {
        d0              3e-3;      // size of the diameter of the gas bubble.
           // It must be lesser than mesh cell size
        p0              1e5;       //initial pressure used as reference for
                                   //calculating the changes in diameter model
    }
    Sc              0.7;        // Schmidt number used to calculate mass diffusivity internally

    residualAlpha   1e-6;
}
```

Tutorial



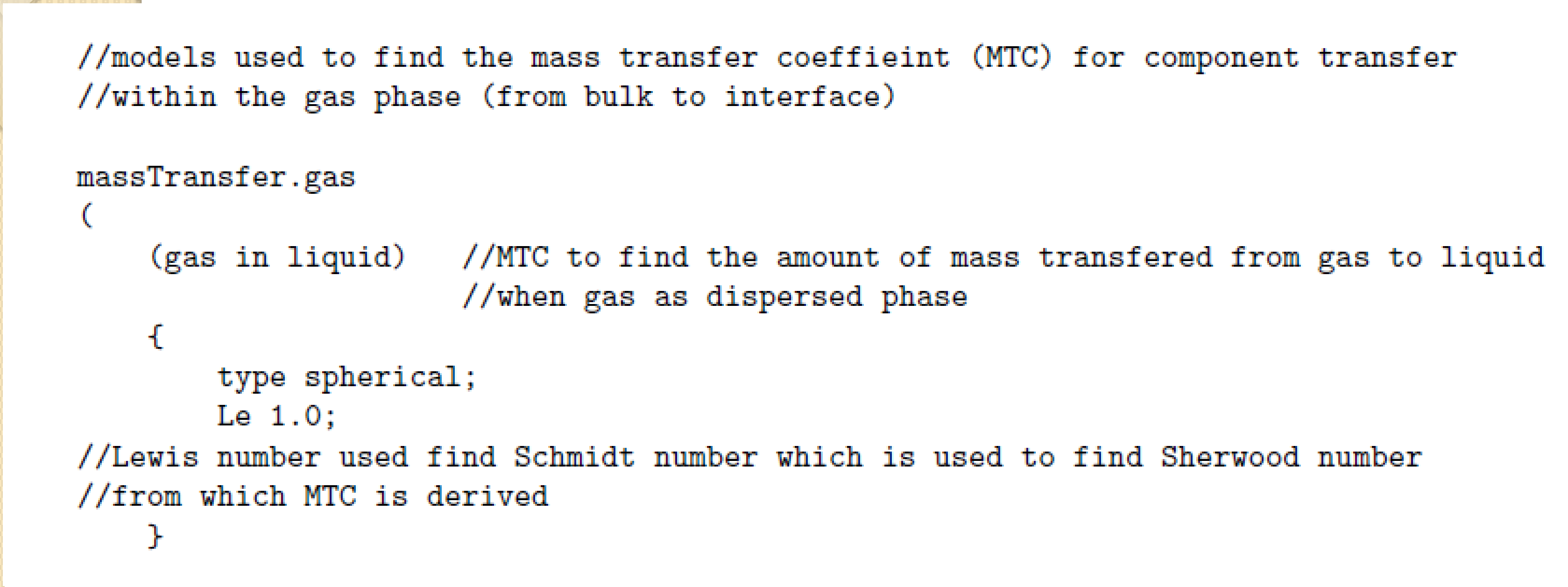
```
//the liquid phase is a purephase and is initialized with
//all component mass fraction equating to zero

liquid
{
    type            purePhaseModel;
    diameterModel   constant;      //no composition change
    constantCoeffs
    {
        d               1e-4;
    }

    residualAlpha   1e-6;
}
```

Tutorial

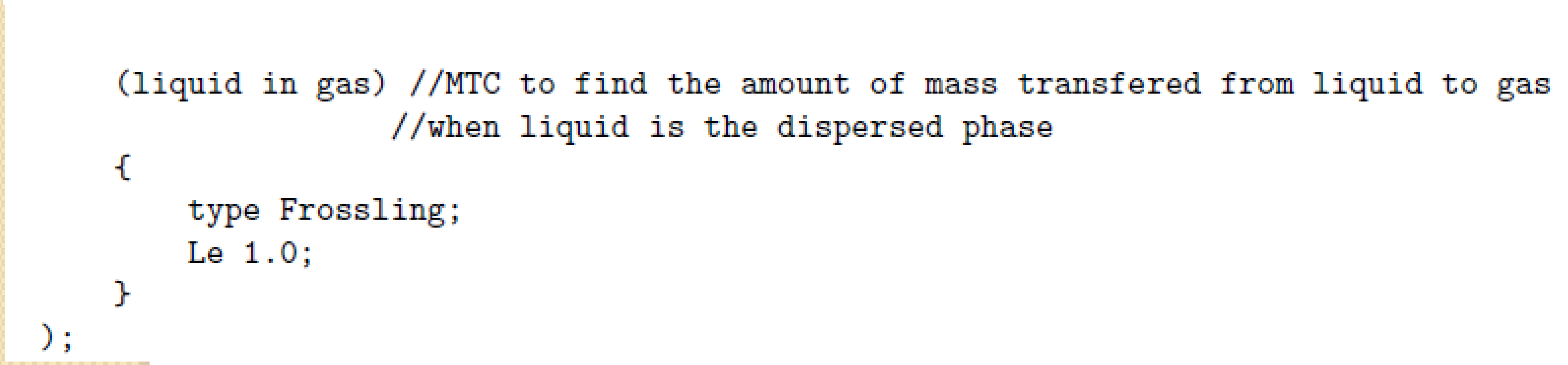


```
//models which dictate the sudden jump in the value of gas mass fraction at interface
//due to gas solubility in liquid

interfaceComposition
(
    (gas in liquid)
    {
        type Saturated;      //interface model name
        species ( H2O );     //Species considered to be at Saturation (usually liquid)
        Le 1.0;

//Le=1.0emphasis MTC is approximately equals to HTC. hence deltaT =deltaY
//and deltaT is calculated using the difference between starurated pressure
//and local pressure

saturationPressure
        {
            type ArdenBuck;   //calculates the saturation pressure using ArdenBuck correlation
        }
    }
);
```

Tutorial

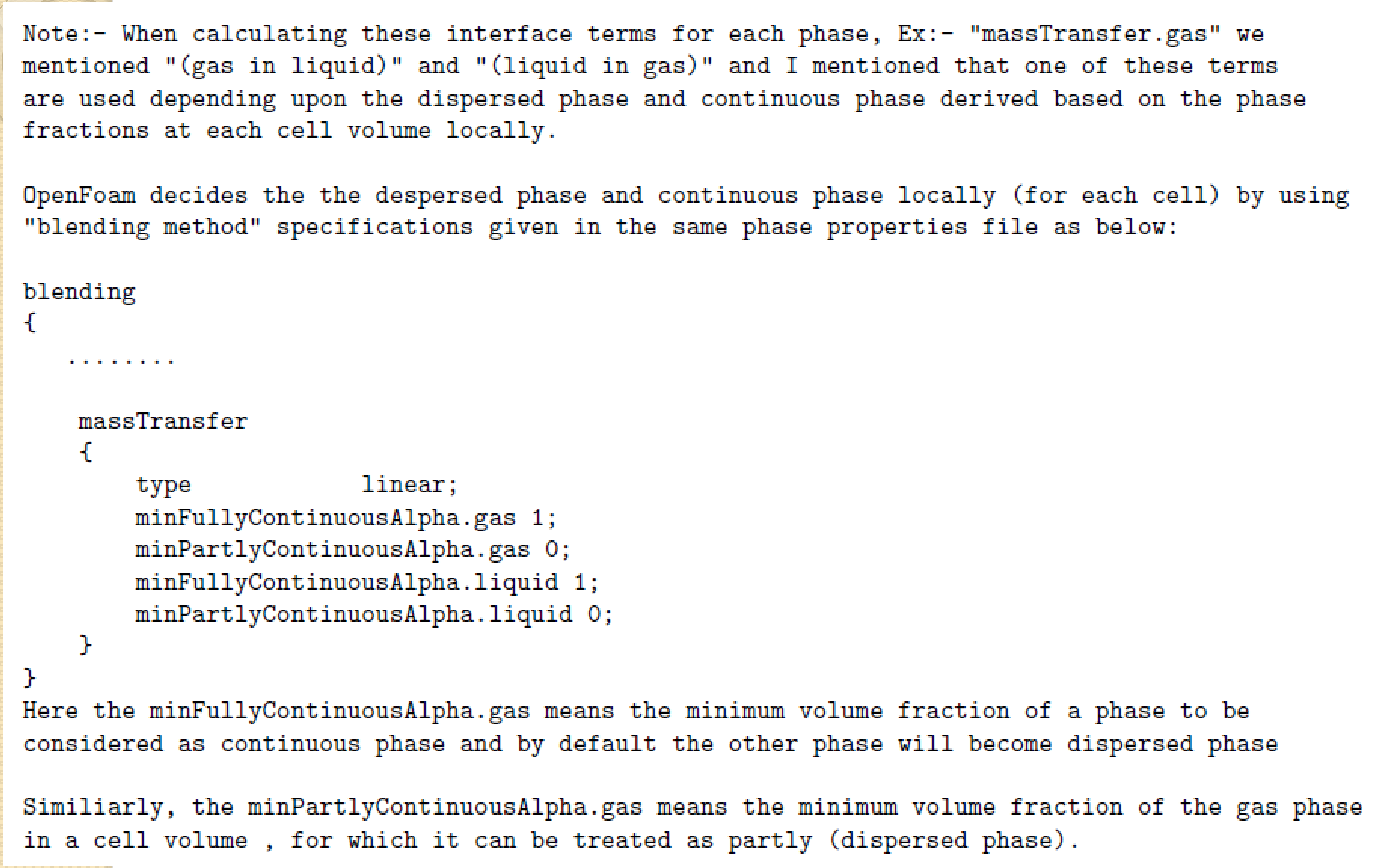


```
//models used to find the mass transfer coeffieint (MTC) for component transfer
//within the gas phase (from bulk to interface)

massTransfer.gas
(
    (gas in liquid)   //MTC to find the amount of mass transfered from gas to liquid
                      //when gas as dispersed phase
    {
        type spherical;
        Le 1.0;
//Lewis number used find Schmidt number which is used to find Sherwood number
//from which MTC is derived
    }
```



```
  (liquid in gas) //MTC to find the amount of mass transfered from liquid to gas
                 //when liquid is the dispersed phase
  {
      type Frossling;
      Le 1.0;
  }
);
```

Tutorial



Note:- When calculating these interface terms for each phase, Ex:- "massTransfer.gas" we mentioned "(gas in liquid)" and "(liquid in gas)" and I mentioned that one of these terms are used depending upon the dispersed phase and continuous phase derived based on the phase fractions at each cell volume locally.

OpenFoam decides the the despersed phase and continuous phase locally (for each cell) by using "blending method" specifications given in the same phase properties file as below:

```
blending
{
   ........

    massTransfer
    {
        type            linear;
        minFullyContinuousAlpha.gas 1;
        minPartlyContinuousAlpha.gas 0;
        minFullyContinuousAlpha.liquid 1;
        minPartlyContinuousAlpha.liquid 0;
    }
}
```

Here the minFullyContinuousAlpha.gas means the minimum volume fraction of a phase to be considered as continuous phase and by default the other phase will become dispersed phase

Similiarly, the minPartlyContinuousAlpha.gas means the minimum volume fraction of the gas phase in a cell volume , for which it can be treated as partly (dispersed phase).

Tutorial

- The details of this method can be found in the file located at:

$FOAM_APP/solvers/multiphase/reactingEulerFoam/phaseSystems/BlendedInterfacialModel/BlendedInterfacialModel.C

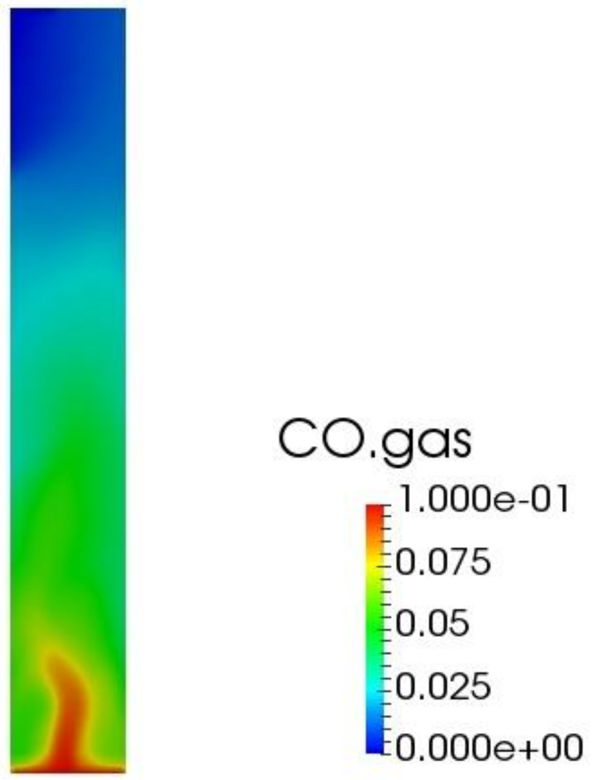
- where in which for this file the fuction weightage values f1 and f2 are determined using blending methods.
- In our current case the method is linear and the files describing this method and calculation of f1 and f2 are described in:

$FOAM_APP/solvers/multiphase/reactingEulerFoam/phaseSystems/BlendedInterfacialModel/blendingMethods/linear/liner.C

- Remaining settings are similar to any other combustion tutorial settings
- Retaining them as it is, we can run the case by initially typing blockmesh command followed by reactingTwoPhaseEulerFoam command

Tutorial

- Post processing is done in paraview opened by command paraFoam.
- On completion of simulation the r CO distribution within the bubble column will be as shown below:



Thank you