A Brief tutorial for XiFoam in OpenFOAM 1.7.x

- A brief introduction to premixed turbulent combustion.
- XiFoam solver:
  - ⋆ Capability of XiFoam:
  - ⋆ Create your own XiFoam solver
  - ⋆ Look inside bEqn.H
  - ⋆ Add subroutine to calculate flame radius(optional)
- Setup a case and preprocess:
  - ⋆ geometry and mesh generation
  - ⋆ combustion and thermophysical properties
  - ⋆ boundary and initial condition
- Implement a new algebraic combustion model

Brief introduction to premixed turbulent combustion

- Fuel and oxidizer are mixed at the molecular level prior to ignition.
- Combustion occurs as a flame front propagating into the unburnt reactants.
- The basic parameter is known to be the progress variable c. $c = \frac{T - T_f}{T_b - T_f}$
- In burnt gas b=0 and in fresh gas b=1.
- The flame front propagation is modelled by solving a transport equation.
- $$\frac{\partial}{\partial t}(\rho b) + \nabla . (\rho \vec{u} b) - \nabla . (\rho D \nabla b) = -\rho_u S_u \Xi |\nabla b|$$

XiFoam solver

XiFoam solver used for:

- compressible
- premixed
- partially-premixed
- combustion
- with turbulence modelling.

create new XiFoam solver

Copy the original solver and rename it to myXiFoam:

```
cd $WM_PROJECT_USER_DIR
cp -r $FOAM_APP/solvers/combustion/XiFoam myXiFoam
cd myXiFoam
```

rename both XiFoam.C and bEqn.H

```
mv XiFoam.C myXiFoam.C
mv bEqn.H myBEqn.H
```

Now, we also have to modify the `files` in `Make` directory,

```
sed -i s/"XiFoam"/"myXiFoam"/g  Make/files
sed -i s/"FOAM_APPBIN"/"FOAM_USER_APPBIN"/g  Make/files
```

so we would have

```
myXiFoam.C
EXE = $(FOAM_USER_APPBIN)/myXiFoam $
```

Continue: create new XiFoam solver

Also we should replace bEqn.H with myBEqn.H in all files:

```
sed -i s/"bEqn.H"/"myBEqn.H"/g *.*
```

and then run the `wmake` command:

```
wmake
```

Now we have myXiFoam solver, and we can modify it.

myBEqn.H

Following items are in `myBEqn.H` file

- Transport equation for regress variable b
- Laminar flame speed based on different models
- Weller combustion modell for calculation Xi=St/Su

Continue: myBEqn.H: Transport equation for regress variable b

Here is the implemenation for transport equation

- $\frac{\partial}{\partial t}(\rho b)+\nabla.(\rho\vec{u}b)-\nabla.(\rho D\nabla b)=-\rho_u S_u \Xi|\nabla b|$

```
fvScalarMatrix bEqn
   (
       fvm::ddt(rho, b)
     + mvConvection->fvmDiv(phi, b)
     + fvm::div(phiSt, b, "div(phiSt,b)")
     - fvm::Sp(fvc::div(phiSt), b)
     - fvm::laplacian(turbulence->alphaEff(), b)
   );
```

Continue: myBEqn.H: Laminar flame speed

Three different model used to calculate laminar flame speed:

- `unstrained`
- `equilibrium`
- `transport`

For implemenation of the these models you can refer to line 111-161 `myBEqn.H`

Continue: myBEqn.H: Weller combustion model

Three methods implemented to calculate the Xi parameters:

- 1- fixed : Do nothing, Xi is fixed!
- 2- algebraic $\Xi^*_{eq} = 1 + 0.62\sqrt{\frac{u'}{S_u}}R_\eta$ $\Xi_{eq} = 1 + 2(1-b)(\Xi^*_{eq} - 1)$

```
Xi == scalar(1) +
    (scalar(1) + (2*XiShapeCoef)*(scalar(0.5) - b))
   *XiCoef*sqrt(up/(Su + SuMin))*Reta;
```

- 3- transport: solve a transport equation for Xi

For implementation you can check line 179 `myBEqn.H`

Subroutine to calculate the flame propagating radius

Flame propagation radius is one of the important parameters which must be measured during the simulation.

$$R=[(\frac{3}{4\pi\rho_b})\iiint \rho(1-b)\,dxdydz]^{1/3}$$

```
touch  radiusFlame.H
gedit  radiusFlame.H
```

and then write:

```
Info<< "Reading radiusFlame.H file "<<endl;
#include "mathematicalConstants.H"
volVectorField centres = mesh.C();
scalar SummationRho=0.0;
scalar RadiusMinRho=0.0;
const scalar coeff=3./(4.*mathematicalConstant::pi);
forAll(centres,k) { SummationRho=SummationRho+(mesh.V()[k]*rho[k]
                   *(scalar(1.)-b[k]))/(min(rho).value());}
RadiusMinRho=Foam::pow(coeff*SummationRho,(1./3.));
Info<< "RadiusMinRho = "<< RadiusMinRho  <<endl;
```

Continue:Subroutine to calculate the flame propagating radius

Add `radiusFlame.H` to `myXiFoam.C` after the `runTime.write();`

```
#include "radiusFlame.H"
```

Write and save radius during simulation:

```
touch createXiFoamOutput.H
```

Add the following lines:

```
OFstream RadiusFlame("XiFoamOutput.txt");
```

Also it is necessary to add these header files in myXiFoam.C after `#include "Switch.H"`

```
#include "IFstream.H"
#include "OFstream.H"
```

Then:

```
touch writeXiFoamOutput.H
```

and write :

```
RadiusFlame << "Time= "<<runTime.timeName() <<"\tRadiusMinRho= "<<
RadiusMinRho<<"\tMin(rho)= "<< min(rho).value()<<endl;
```

Continue:Subroutine to calculate the flame propagating radius

Then, we have to add :

```
#include "createXiFoamOutput.H"
```

In myXiFoam.C before the :

```
Info<< "\nStarting time loop\n" << endl;
```

And also add

```
#include "writeXiFoamOutput.H"
```

After the line

```
#include "radiusFlame.H"
```

Finally run the `wmake` command:

```
wmake
```

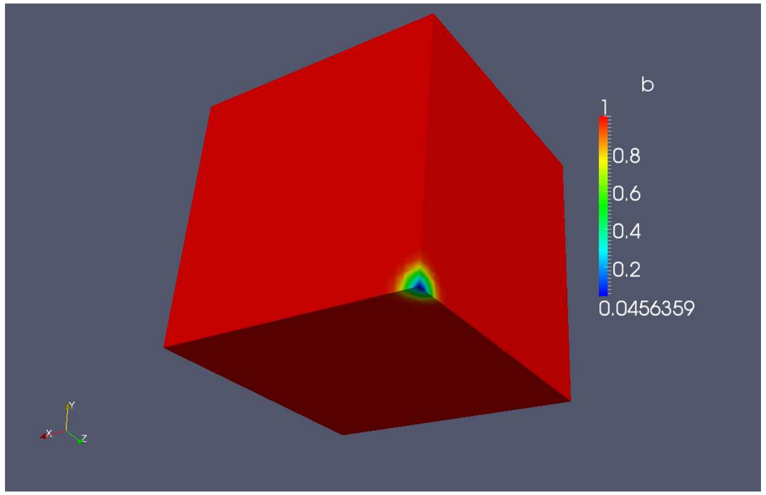
- Now we have a solver the same as XiFoam which calculate the the flame propagating radius

Setting up the case:preprocess and run

- Case description
- Constant/polyMesh Folder
- Constant Folder
- Initial and boundary condition
- System Folder

Setting up the case:description

- The case study is a cubic combustion chamber, which ignition occurred at its centre in 0.001 ms.
- The fuel is propane which premixed with air.
- The domain consists of a square with length= 35 mm.
- A uniform mesh of 35*35*35 is used to have cell size= 1mm.



Setting up the case: constant/polyMesh

Copy the default case of XiFoam

```
    run
    cp  -r $FOAM_TUTORIALS/combustion/XiFoam/ras/moriyoshiHomogeneous chamber
    cd chamber
    tree L 2
    gedit constant/polyMesh/blockMeshDict
```

Modify the blockMesh to have:

```
convertToMeters 0.001;
vertices
(   (0 0 0) //vertex No.1
    (0 35 0) //vertex No.2
    (35 0 0) //vertex No.3
    (35 35 0) //vertex No.4
    (0 0 35) //vertex No.5
    (0 35 35) //vertex No.6
    (35 0 35) //vertex No.7
    (35 35 35) //vertex No.8
);
```

Continue: Setting up the case: constant/polyMesh

```
blocks
(    hex (0 2 3 1 4 6 7 5) (35 35 35) simpleGrading (1 1 1)  //Block No.1);
edges                (); 
patches    (    symmetryPlane left     ( (0 4 5 1) )
                symmetryPlane right    ( (2 3 7 6) )
                symmetryPlane top      ( (1 5 7 3) )
                symmetryPlane bottom   ( (0 2 6 4) )
                symmetryPlane front    ( (4 5 7 6) )
                symmetryPlane back     ( (0 1 3 2) )
           );
mergePatchPairs ();
```

Mesh the geometry using `blockMesh`

```
blockMesh
```

View the geometry in paraview

```
paraFoam
```

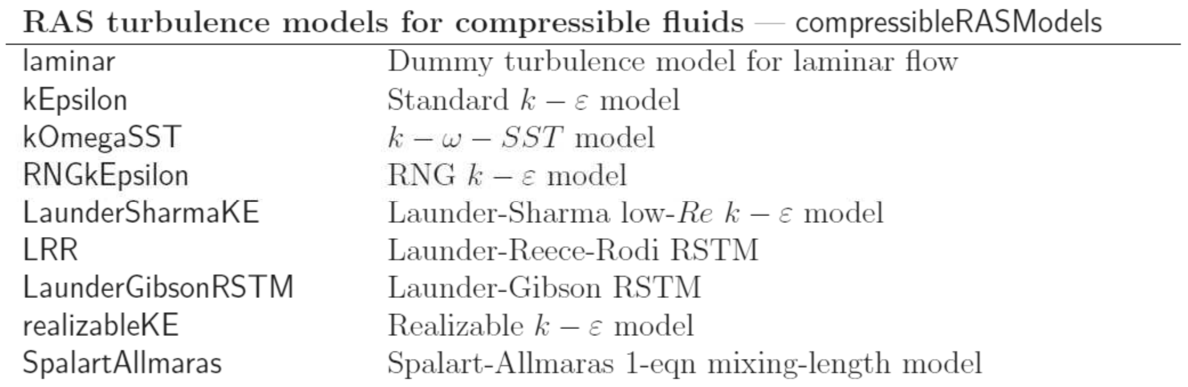
Setting up the case:constant folder

- turbulenceProperties `1- RASModel  2- LESModel`

```
simulationType  RASModel;
```

- RASProperties

```
RASModel        LaunderSharmaKE;
turbulence      on;
printCoeffs     on;
```



RAS turbulence models for compressible fluids — compressibleRASModels	
laminar	Dummy turbulence model for laminar flow
kEpsilon	Standard $k-\varepsilon$ model
kOmegaSST	$k-\omega-SST$ model
RNGkEpsilon	RNG $k-\varepsilon$ model
LaunderSharmaKE	Launder-Sharma low-Re $k-\varepsilon$ model
LRR	Launder-Reece-Rodi RSTM
LaunderGibsonRSTM	Launder-Gibson RSTM
realizableKE	Realizable $k-\varepsilon$ model
SpalartAllmaras	Spalart-Allmaras 1-eqn mixing-length model

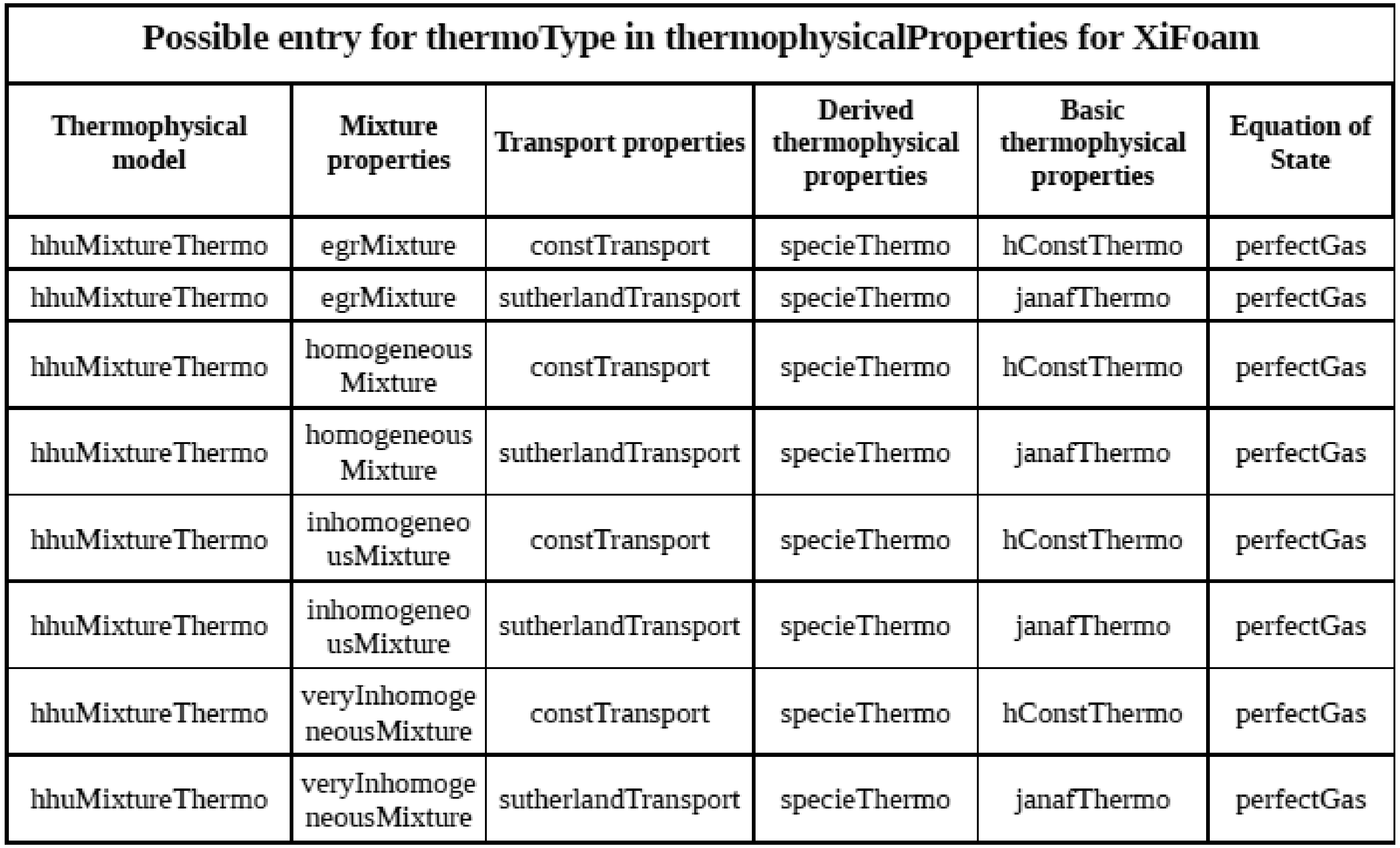
- g

```
dimensions      [0 1 -2 0 0 0 0];
value               ( 0 0 0 );
```

Continue: Setting up the case:thermophysicalProperties

- keyword: `thermoType`

Possible entry for `thermoType` in thermophysicalProperties for XiFoam

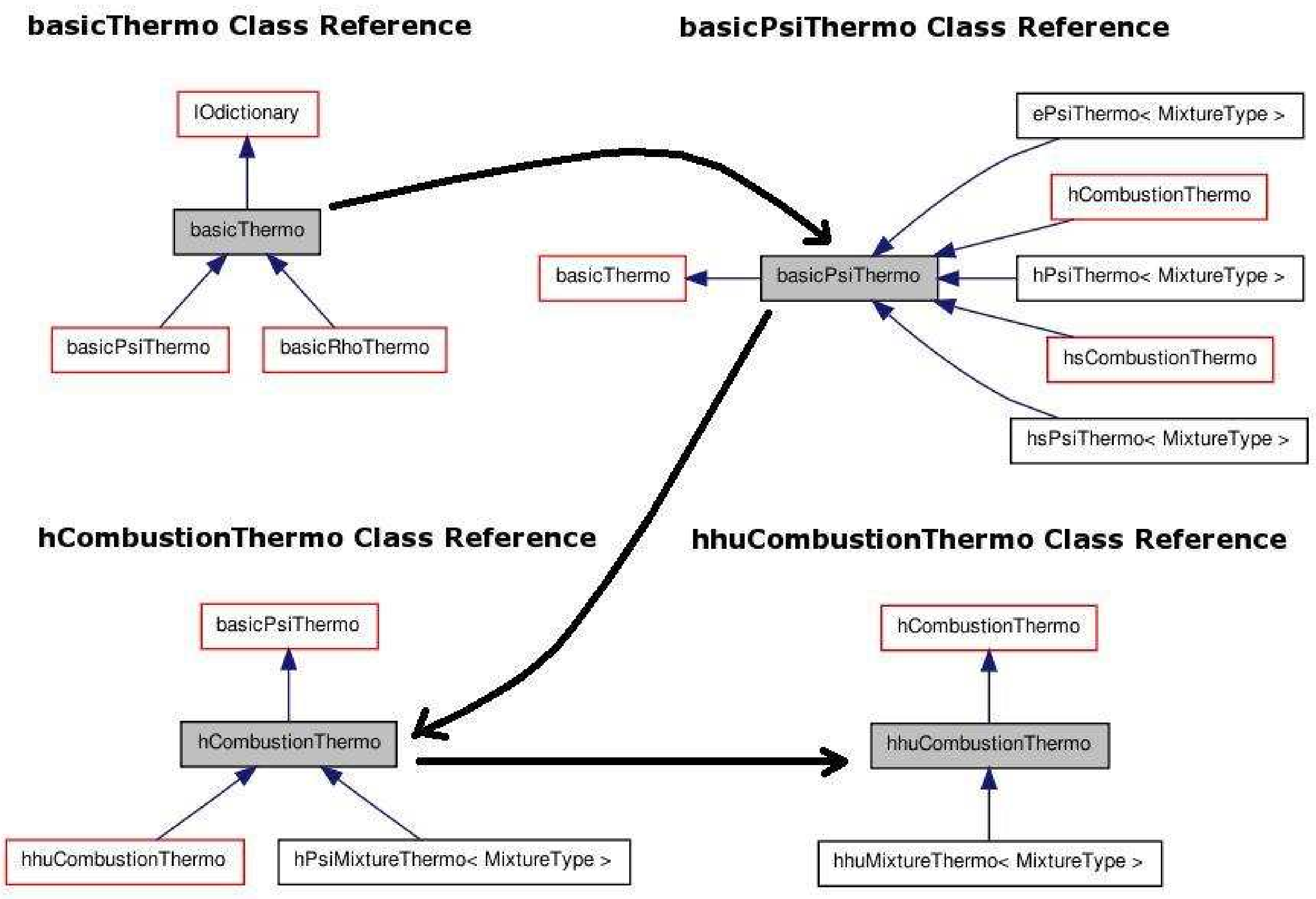


Possible entry for thermoType in thermophysicalProperties for XiFoam

Thermophysical model	Mixture properties	Transport properties	Derived thermophysical properties	Basic thermophysical properties	Equation of State
hhuMixtureThermo	egrMixture	constTransport	specieThermo	hConstThermo	perfectGas
hhuMixtureThermo	egrMixture	sutherlandTransport	specieThermo	janafThermo	perfectGas
hhuMixtureThermo	homogeneous Mixture	constTransport	specieThermo	hConstThermo	perfectGas
hhuMixtureThermo	homogeneous Mixture	sutherlandTransport	specieThermo	janafThermo	perfectGas
hhuMixtureThermo	inhomogeneo usMixture	constTransport	specieThermo	hConstThermo	perfectGas
hhuMixtureThermo	inhomogeneo usMixture	sutherlandTransport	specieThermo	janafThermo	perfectGas
hhuMixtureThermo	veryInhomoge neousMixture	constTransport	specieThermo	hConstThermo	perfectGas
hhuMixtureThermo	veryInhomoge neousMixture	sutherlandTransport	specieThermo	janafThermo	perfectGas

Continue: Setting up the case:thermophysicalProperties

Here is the Inheritance diagram for `hhuMixtureThermo`:


basicThermo Class Reference
IOdictionary
basicThermo
basicPsiThermo
basicRhoThermo
basicPsiThermo Class Reference
ePsiThermo< MixtureType >
hCombustionThermo
basicThermo
basicPsiThermo
hPsiThermo< MixtureType >
hsCombustionThermo
hsPsiThermo< MixtureType >
hCombustionThermo Class Reference
basicPsiThermo
hCombustionThermo
hhuCombustionThermo
hPsiMixtureThermo< MixtureType >
hhuCombustionThermo Class Reference
hCombustionThermo
hhuCombustionThermo
hhuMixtureThermo< MixtureType >

Continue: Setting up the case:thermophysicalProperties

- keyword: `stoichiometricAirFuelMassRatio`

stoichiometric ratio of Air-Fuel, and is read on line 52 of:

```
src/thermophysicalModels/reactionThermo/mixtures/inhomogeneousMixture.C
```

- keyword: `fuel, oxidant, burntProducts` read by:

```
src/thermophysicalModels/reactionThermo/mixtures/inhomogeneousMixture.C
```

- keyword: `reactants, products` read by:

```
src/thermophysicalModels/reactionThermo/mixtures/homogeneousMixture.C
```

Explanation of the coefficients:

```
Line 1: fuel
Line 2: fuel 1 44.0962
Line 3: 200 5000 1000
Line 4: 7.534 0.01887 -6.271e-06 9.147e-10 -4.783e-14 -16467.5 -17.892
Line 5: 0.9335 0.02642 6.105e-06 -2.197e-08 9.514e-12 -13958.5 19.201
Line 6: 1.67212e-06 170.672;
```

Continue: Setting up the case:thermophysicalProperties

```
Line 1: keyword
Line 2: <specieCoeffs>: n_moles       Molecular weight(W(kg/kmol))
Line 3: Lower, Upper and Common temperature Respectively
Line 4: High temperature coeff: a1-a7(a6:enthalpy offset,a7:entropy offset)
Line 5: Low  temperature coeff: a1-a7(a6:enthalpy offset,a7:entropy offset)
Line 6: Sutherland coefficient
```

$$\frac{C^o_{pk}}{R}=a_{1k}+a_{2k}T_k+a_{3k}T_k^2+a_{4k}T_k^3+a_{5k}T_k^4$$

$$\frac{H^o_k}{RT_k}=a_{1k}+\frac{a_{2k}}{2}T_k+\frac{a_{3k}}{3}T_k^2+\frac{a_{4k}}{4}T_k^3+\frac{a_{5k}}{5}T_k^4+\frac{a_{6k}}{T_k}$$

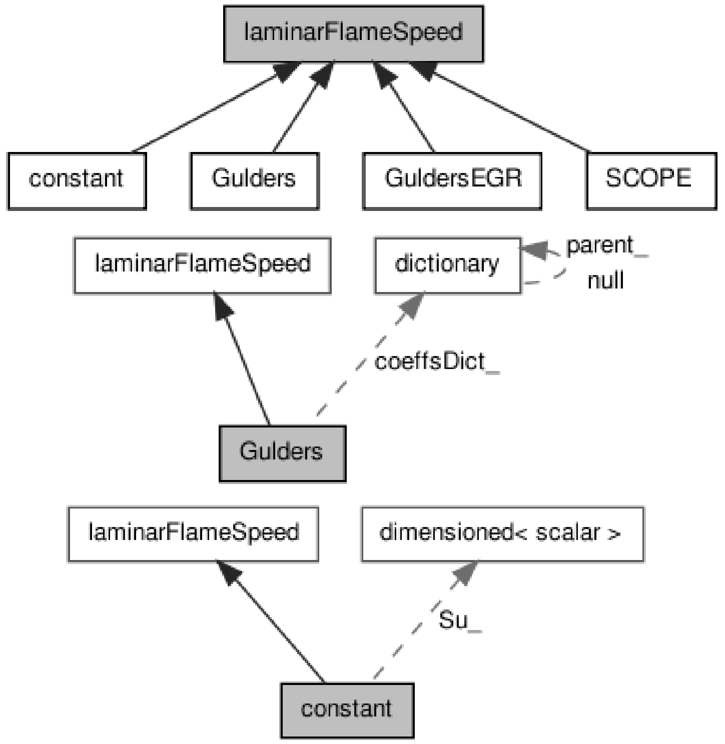
$$\frac{S^o_k}{R}=a_{1k}\ln T_k+a_{2k}T_k+\frac{a_{3k}}{2}T_k^2+\frac{a_{4k}}{3}T_k^3+\frac{a_{5k}}{4}T_k^4+a_{7k}$$

$$\mu=A_s\frac{T^{1/2}}{(1+T_s/T)}$$

Continue: Setting up the case:combustionProperties

- keyword: `laminarFlameSpeedCorrelation`

Three options for this entry:1-`Gulders` 2-`GuldersEGR` 3-`constant`



Laminar flame speed based on Gulders formulation:

```
src/thermophysicalModels/laminarFlameSpeed/Gulders/Gulders.C
src/thermophysicalModels/laminarFlameSpeed/GuldersEGR/GuldersEGR.C
```

Continue: Setting up the case:combustionProperties

- keyword: `fuel`

`fuel` must be specified,if `Gulders/GuldersEGR` is selected in `laminarFlameSpeedCorrelation`
`fuel` keyword is read on line 47 of the following file:
`src/thermophysicalModels/laminarFlameSpeed/laminarFlameSpeed/laminarFlameSpee`

And then it is used on line 57, 56 of the following files, respectively:

```
src/thermophysicalModels/laminarFlameSpeed/Gulders/Gulders.C
src/thermophysicalModels/laminarFlameSpeed/GuldersEGR/GuldersEGR.C
```

- keyword: `Su`

If we choose `constant` laminar flame speed(Su) in `laminarFlameSpeedCorrelation`

In line 57 of the following file, constant laminar flame speed(Su) is read.
`src/thermophysicalModels/laminarFlameSpeed/constant/constant.C`

Continue: Setting up the case:combustionProperties

- keyword: `equivalenceRatio`

Defined as ratio of the fuel-to-oxidizer ratio to the stoichiometric fuel-to-oxidizer ratio.

$$\Phi = \frac{\frac{m_{fuel}}{m_{oxidizer}}}{(\frac{m_{fuel}}{m_{oxidizer}})_{st}}$$

This keyword is read by:

```
/src/thermophysicalModels/laminarFlameSpeed/laminarFlameSpeed/laminarFlameSpee
```

- keyword: `SuModel`

There are three options for this entry: `1-unstrained` `2-equilibrium` `3-transport`
These options read by :

```
    applications/solvers/combustion/XiFoam/readCombustionProperties.H
```

And the implementation of these model are in line 120:

```
    /applications/solvers/combustion/XiFoam/bEqn.H
```

Continue: Setting up the case:combustionProperties

- keyword: `sigmaExt`

The strain rate at extinction which obtained from the Markstein length by extrapolating to `Su--> 0`
This keyword is read by `readCombustionProperties.H` and used in `bEqn.H`

- keyword: `XiModel`

Three different models for flame wrinkling Xi:1- `fixed` 2- `algebraic` 3- `transport`
This keyword is read by `readCombustionProperties.H` and used in `bEqn.H`

- keyword: `XiCoef and XiShapeCoef`

These coefficients used in algebraic model for Xi in line 175 of `bEqn.H`
And read by:`readCombustionProperties.H`

- keyword: `uPrimeCoef`

`uPrimeCoef` is used in calculation the velocity fluctuation on line 74 of the `bEqn.H`

Continue: Setting up the case:combustionProperties

- keyword: `GuldersCoeffs GuldersEGRCoeffs`

These coefficients used to calculate laminar flame speed according to the Gulders formulation for specific `fuel`.
These coefficients are read by the following codes depend on the selected model for `laminarFlameSpeedCorrelation`.

```
src/thermophysicalModels/laminarFlameSpeed/Gulders/Gulders.C
src/thermophysicalModels/laminarFlameSpeed/GuldersEGR/GuldersEGR.C
```

$$S_u = W\,\Phi^{\eta}\exp[-\xi(\Phi-1.075)^2](\frac{T}{T_0})^{\alpha}(\frac{P}{P_0})^{\beta}$$

- keyword: `ignite`

If we have ignition we must specify here: 1- `yes` 2- `no`
This entry read by the `readCombustionProperties.H` file on line 45

Continue: Setting up the case:combustionProperties

- keyword: `ignitionSites`

The `location, diameter, duration` and `strength` of ignition are specified here.
These data is read by the following code:

```
src/engine/ignition/ignitionSiteIO.C
```

- keyword: `ignitionSphereFraction, ignitionThickness, ignitionCircleFraction ignitionKernelArea`

These are some correction factor based on the ignition shape,and the geometry using `mesh.nGeometricD()`.
These coefficients are read by the following files:

```
src/engine/include/stCorr.H
```

And return the `StCorr` which is used in calculation the turbulent flame speed flux in `bEqn.H` in line 37
`StCorr` varies between 1-10 during the simulation, and must be reduced during the simulation.

Setting up the case:Initial and boundary condition

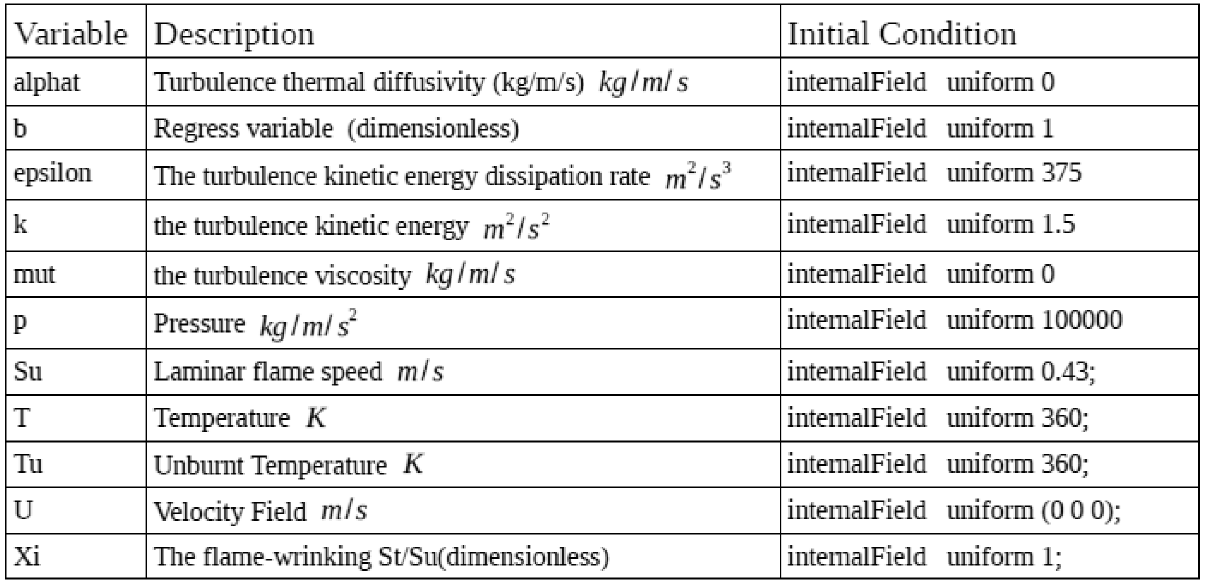
- keyword: `boundary condition:`

We use a `symmetryPlane` boundary condition for the case:

- keyword: `initial condition:`

We have the following files in 0 direcrtory:

```
alphat,  b,  epsilon,  k,  mut,  p,  Su,  T,  Tu,  U,  Xi
```



Variable	Description	Initial Condition
alphat	Turbulence thermal diffusivity (kg/m/s) $kg/m/s$	internalField uniform 0
b	Regress variable (dimensionless)	internalField uniform 1
epsilon	The turbulence kinetic energy dissipation rate m^2/s^3	internalField uniform 375
k	the turbulence kinetic energy m^2/s^2	internalField uniform 1.5
mut	the turbulence viscosity $kg/m/s$	internalField uniform 0
p	Pressure $kg/m/s^2$	internalField uniform 100000
Su	Laminar flame speed m/s	internalField uniform 0.43;
T	Temperature K	internalField uniform 360;
Tu	Unburnt Temperature K	internalField uniform 360;
U	Velocity Field m/s	internalField uniform (0 0 0);
Xi	The flame-wrinking St/Su(dimensionless)	internalField uniform 1;

Setting up the case: system folder and run

There is no change required here, so run the case:

```
    myXiFoam >log &
    paraFoam
```

Implement a new combustion model:

Please refer to report for implementation.