051176 - Computational Techniques for Thermochemical Propulsion

Master of Science in Aeronautical Engineering

The Equation of Energy

Prof. **Federico Piscaglia**

Dept. of Aerospace Science and Technology (DAER)

POLITECNICO DI MILANO, Italy

federico.piscaglia@polimi.it

Bibliography

https://cfd.direct/openfoam/energy-equation

Introduction

The roles of the various equations in compressible flows are quite different from the ones they play in incompressible flows. Also note that the nature of the pressure is completely different:

- for incompressible flows there is only the dynamic pressure whose absolute value is of no consequence;
- for compressible flows, it is the thermodynamic pressure whose absolute value is of critical importance.

Additional differences are related to the significance of the conservation of energy.

Conservation of Energy

The rate of change of total energy for a particle of material must equal the input of mechanical and thermodynamic power from fluxes and sources acting on the particle.

$$\boxed{\text{Total Energy}} = \boxed{\text{Mechanical Power}} + \boxed{\text{Thermodynamic Power}}$$

In the limit where particle size is infinitesimally small, it states:

$$\underbrace{\rho \frac{De}{Dt}}_{\text{thermo}} + \underbrace{\rho \frac{DK}{Dt}}_{\text{mech}} = \underbrace{-\nabla \cdot \boldsymbol{q} + \rho r}_{\text{thermo}} + \underbrace{\nabla \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{U}) + \rho \boldsymbol{g} \cdot \boldsymbol{U}}_{\text{mech}}$$

Conservation of Energy

The law of conservation of energy states that the total energy of an isolated system remains constant, i.e. it is conserved over time and energy is not created or destroyed but is transformed from one form to another. In integral form the energy equation is:

$$\frac{\partial}{\partial t}\int_V \rho h \, dV + \int_S \rho h \boldsymbol{v} \cdot \boldsymbol{n} dS - \frac{\partial}{\partial t}\int_V p \, dV = -\int_V \nabla \cdot q \, dV + \int_S [\boldsymbol{v}\nabla p + \boldsymbol{\tau} : \nabla \boldsymbol{v}] \cdot \boldsymbol{n} dS + \dot{S}$$

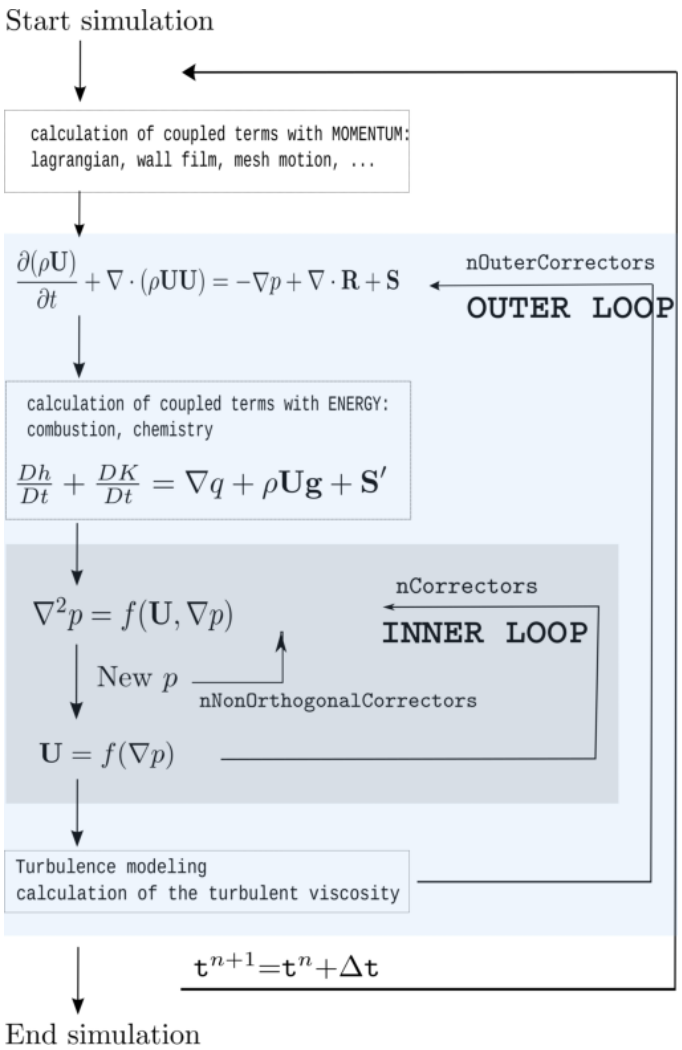
being:

- h is the enthalpy per unit mass;
- q is the energy flux, including heat diffusion (Fourier) and specie diffusion:

$$q_i = -\lambda \frac{\partial T}{\partial x} + \rho \sum_{k=1}^{N} h_k Y_k V_{k,i}$$

  where λ is the thermal conductivity and T the absolute temperature (K);
- $\boldsymbol{\tau}$ is the viscous part of the stress tensor ($\boldsymbol{\sigma} = \boldsymbol{\tau} - p\boldsymbol{I}$).

Compressible Solvers in OpenFOAM



- SIMPLE (steady) :
  - `nOuterCorrectors`>1
  - `nCorrectors`=1
  - `nNonOrthogonalCorrectors` $\in$ [0;∞)
- PISO (unsteady):
  - `nOuterCorrectors`=1
  - `nCorrectors`=2
  - `nNonOrthogonalCorrectors` $\in$ [0;∞)
- PIMPLE (unsteady, transient SIMPLE):
  - `nOuterCorrectors`>1
  - `nCorrectors`>1
  - `nNonOrthogonalCorrectors` $\in$ [0;∞)

An Important Note

The energy equation should be written as:

$$\frac{D(\rho e)}{Dt} + \frac{D(\rho K)}{Dt} = -\nabla \cdot \boldsymbol{q} + \rho r + \nabla \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{U}) + \rho \boldsymbol{g} \cdot \boldsymbol{U}$$

while density often appears out of the total derivatives.

$$\rho \frac{De}{Dt} + \rho \frac{DK}{Dt} = -\nabla \cdot \boldsymbol{q} + \rho r + \nabla \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{U}) + \rho \boldsymbol{g} \cdot \boldsymbol{U}$$

The two equations above are equivalent, despite the first formulation is denoted in CFD as **conservative form**.

NOTE: from mass conservation

$$\frac{D\rho}{Dt} = 0$$

As a consequence, **total derivatives in the energy equation** can be written as follows:

$$\frac{D(\rho e)}{Dt} \equiv \rho \frac{De}{Dt} + e \frac{D\rho}{Dt} = \rho \frac{De}{Dt}$$

and, similarly:

$$\frac{D(\rho K)}{Dt} \equiv \rho \frac{DK}{Dt} + K \frac{D\rho}{Dt} = \rho \frac{DK}{Dt}$$

Mechanical Power (1/3)

The **rate of change of mechanical, or kinetic, energy** is:

$$\rho \frac{DK}{Dt} \equiv \rho \frac{D(|\boldsymbol{U}|^2/2)}{Dt} \equiv \rho \frac{D\boldsymbol{U}}{Dt} \cdot \boldsymbol{U}$$

where $\boldsymbol{U}$ is the flow velocity, $K \equiv \boldsymbol{U}^2/2$ is the specific kinetic energy (kinetic energy per unit mass) and ρ is the mass density.

- The power flux, or rate of change of strain energy, is

$$\nabla \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{U}) \equiv (\nabla \cdot \boldsymbol{\sigma}) \cdot \boldsymbol{U} + \boldsymbol{\sigma} : \nabla \boldsymbol{U}$$

  where $\boldsymbol{\sigma}$ is the mechanical stress tensor. The stress tensor $\boldsymbol{\sigma}$ may be decomposed into a scalar thermodynamic pressure p and viscous stress tensor τ by $\boldsymbol{\sigma} = \boldsymbol{\tau} - p\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity tensor.

- The power source, or rate of change of potential energy, is

$$\rho \boldsymbol{g} \cdot \boldsymbol{U}$$

  where $\boldsymbol{g}$ is a body acceleration, e.g. gravity.

Mechanical Power (2/3)

Mechanical contributions to the equation of energy can be grouped and may be written as:

$$\dot{\boldsymbol{W}} \equiv \rho\frac{DK}{Dt} - \nabla\cdot(\boldsymbol{\sigma}\cdot\boldsymbol{U}) - \rho\boldsymbol{g}\cdot\boldsymbol{U} = \rho\boldsymbol{U}\frac{D\boldsymbol{U}}{Dt} - \underbrace{[(\nabla\cdot\boldsymbol{\sigma})\cdot\boldsymbol{U} + \boldsymbol{\sigma}:\nabla\boldsymbol{U}]}_{\nabla\cdot(\boldsymbol{\sigma}\cdot\boldsymbol{U})} - \rho\boldsymbol{g}\cdot\boldsymbol{U}$$

The $\boldsymbol{\sigma}:\nabla\boldsymbol{U}$ term represents the contribution of mechanical power to internal energy, and thus, random motion of particles. The expression $(\nabla\cdot\boldsymbol{\sigma})\cdot\boldsymbol{U}$ must then represent a power due to the bulk motion of particles. If we combine it with the momentum equation:

$$\rho\frac{D\boldsymbol{U}}{Dt} = \nabla\cdot\boldsymbol{\sigma} + \rho\boldsymbol{g}$$

It follows:

$$\begin{aligned}\dot{\boldsymbol{W}} &= \rho\frac{D\boldsymbol{U}}{Dt}\cdot\boldsymbol{U} - [(\nabla\cdot\boldsymbol{\sigma})\cdot\boldsymbol{U} + \boldsymbol{\sigma}:\nabla\boldsymbol{U}] - \rho\boldsymbol{g}\cdot\boldsymbol{U} = \\ &= (\nabla\cdot\boldsymbol{\sigma} + \rho\boldsymbol{g})\cdot\boldsymbol{U} - [(\nabla\cdot\boldsymbol{\sigma})\cdot\boldsymbol{U} + \boldsymbol{\sigma}:\nabla\boldsymbol{U}] - \rho\boldsymbol{g}\cdot\boldsymbol{U} = \\ &= -\boldsymbol{\sigma}:\nabla\boldsymbol{U}\end{aligned}$$

The mechanical contributions reduce to:

$$\dot{\boldsymbol{W}} = \rho\frac{D\boldsymbol{U}}{Dt}\cdot\boldsymbol{U} - \nabla\cdot(\boldsymbol{\sigma}\cdot\boldsymbol{U}) - \rho\boldsymbol{g}\cdot\boldsymbol{U} = -\boldsymbol{\sigma}:\nabla\boldsymbol{U}$$

Mechanical Power (3/3)

An equation for internal energy is produced by simplifying the mechanical contributions from the total energy:

$$\rho\frac{De}{Dt} + \rho\frac{D\boldsymbol{U}}{Dt}\cdot\boldsymbol{U} = -\nabla\cdot\boldsymbol{q} + \rho r + \nabla\cdot(\boldsymbol{\sigma}\cdot\boldsymbol{U}) + \rho\boldsymbol{g}\cdot\boldsymbol{U}$$

$$\rho\frac{De}{Dt} + \underbrace{\rho\frac{D\boldsymbol{U}}{Dt}\cdot\boldsymbol{U} - \nabla\cdot(\boldsymbol{\sigma}\cdot\boldsymbol{U}) - \rho\boldsymbol{g}\cdot\boldsymbol{U}}_{\dot{\boldsymbol{W}}=-\boldsymbol{\sigma}:\nabla\boldsymbol{U}} = -\nabla\cdot\boldsymbol{q} + \rho r$$

Hence, the **conservation of Internal Energy** takes the form:

$$\rho\frac{De}{Dt} = -\nabla\cdot\boldsymbol{q} + \rho r + \boldsymbol{\sigma}:\nabla\boldsymbol{U}$$

An important note - 2

We can express our equations in terms of the local (or partial) derivative $\frac{\partial}{\partial t}$, where for the tensor ϕ:

$$\frac{D\phi}{Dt} \equiv \frac{\partial \phi}{\partial t} + \boldsymbol{U} \cdot \nabla \phi$$

so:

$$\rho \frac{D\phi}{Dt} = \frac{D(\rho\phi)}{Dt} \equiv \frac{\partial(\rho\phi)}{\partial t} + \nabla \cdot (\rho \boldsymbol{U} \phi)$$

Conservation of Total Energy

$$\frac{D(\rho e)}{Dt} + \frac{D(\rho K)}{Dt} = -\nabla \cdot \boldsymbol{q} + \rho r + \nabla \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{U}) + \rho \boldsymbol{g} \cdot \boldsymbol{U}$$

Decomposing the stress tensor in its principal and deviatoric part:

$$\boldsymbol{\sigma} = \boldsymbol{\tau} - p\boldsymbol{I}$$

it follows:

$$\frac{\partial(\rho e)}{\partial t} + \nabla \cdot (\rho \boldsymbol{U} e) + \frac{\partial(\rho K)}{\partial t} + \nabla \cdot (\rho \boldsymbol{U} K) = -\nabla \cdot \boldsymbol{q} + \rho r + \underbrace{\nabla \cdot (\boldsymbol{\tau} \cdot \boldsymbol{U}) - \nabla \cdot (\boldsymbol{U}\, p)}_{\nabla \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{U})} + \rho \boldsymbol{g} \cdot \boldsymbol{U}$$

so:

The **conservation of Total Energy** takes the form:

$$\frac{\partial(\rho e)}{\partial t} + \nabla \cdot (\rho \boldsymbol{U} e) + \frac{\partial(\rho K)}{\partial t} + \nabla \cdot (\rho \boldsymbol{U} K) + \nabla \cdot (\boldsymbol{U}\, p) = -\nabla \cdot \boldsymbol{q} + \rho r + \nabla \cdot (\boldsymbol{\tau} \cdot \boldsymbol{U}) + \rho \boldsymbol{g} \cdot \boldsymbol{U}$$

Conservation of Total Energy

The **conservation of Total Energy** takes the form:

$$\frac{\partial(\rho e)}{\partial t}+\nabla\cdot(\rho \boldsymbol{U} e)+\frac{\partial(\rho K)}{\partial t}+\nabla\cdot(\rho \boldsymbol{U} K)+\nabla\cdot(\boldsymbol{U}\, p)= -\nabla\cdot\boldsymbol{q}+\rho r+\nabla\cdot(\boldsymbol{\tau}\cdot\boldsymbol{U})+\rho \boldsymbol{g}\cdot\boldsymbol{U}$$

NEXT STEP: being

$$h \equiv e+\frac{p}{\rho} \rightarrow \rho\, e=\rho\, h-p$$

we want to write the conservation of total energy as conservation of total enthalpy. In particular:

$$\frac{\partial(\rho e)}{\partial t}+\nabla\cdot(\rho \boldsymbol{U} e)+\nabla\cdot(\boldsymbol{U}\, p)=\frac{\partial(\rho h-p)}{\partial t}+\nabla\cdot[\rho \boldsymbol{U}(e+p)]=\frac{\partial(\rho h)}{\partial t}-\frac{\partial p}{\partial t}+\nabla\cdot(\rho \boldsymbol{U} h)$$

Conservation of Total Enthalpy

Being $h \equiv e + \frac{p}{\rho}$, it follows:

$$\underbrace{\frac{\partial(\rho h)}{\partial t} - \frac{\partial p}{\partial t} + \nabla\cdot(\rho \boldsymbol{U} h)}_{\frac{\partial(\rho e)}{\partial t}+\nabla\cdot(\rho \boldsymbol{U} e)+\nabla\cdot(\boldsymbol{U}\, p)} + \frac{\partial(\rho K)}{\partial t} + \nabla\cdot(\rho \boldsymbol{U} K) = -\nabla\cdot\boldsymbol{q} + \rho r + \nabla\cdot(\boldsymbol{\tau}\cdot\boldsymbol{U}) + \rho \boldsymbol{g}\cdot\boldsymbol{U}$$

and:

$$\underbrace{\frac{\partial(\rho h)}{\partial t} + \nabla\cdot(\rho \boldsymbol{U} h) + \frac{\partial(\rho K)}{\partial t} + \nabla\cdot(\rho \boldsymbol{U} K)}_{\frac{D(\rho h)}{Dt} + \frac{D(\rho K)}{Dt}} - \frac{\partial p}{\partial t} = -\nabla\cdot\boldsymbol{q} + \rho r + \nabla\cdot(\boldsymbol{\tau}\cdot\boldsymbol{U}) + \rho \boldsymbol{g}\cdot\boldsymbol{U}$$

NOTE: the diagonal part of the stress tensor $p\boldsymbol{I}$ is now embedded into the advective term!

The **conservation of Total Enthalpy** takes the form:

$$\frac{\partial(\rho h)}{\partial t} + \nabla\cdot(\rho \boldsymbol{U} h) + \frac{\partial(\rho K)}{\partial t} + \nabla\cdot(\rho \boldsymbol{U} K) - \frac{\partial p}{\partial t} = -\nabla\cdot\boldsymbol{q} + \rho r + \nabla\cdot(\boldsymbol{\tau}\cdot\boldsymbol{U}) + \rho \boldsymbol{g}\cdot\boldsymbol{U}$$

Conservation of Total Enthalpy

The **conservation of Total Enthalpy** takes the form:

$$\frac{\partial(\rho h)}{\partial t} + \nabla\cdot(\rho \boldsymbol{U} h) + \frac{\partial(\rho K)}{\partial t} + \nabla\cdot(\rho \boldsymbol{U} K) - \frac{\partial p}{\partial t} = -\nabla\cdot\boldsymbol{q} + \rho r + \nabla\cdot(\boldsymbol{\tau}\cdot\boldsymbol{U}) + \rho \boldsymbol{g}\cdot\boldsymbol{U}$$

Conservation of Total Energy

We wrote the **conservation of total energy** as:

$$\frac{\partial(\rho e)}{\partial t}+\nabla\cdot(\rho \boldsymbol{U} e)+\frac{\partial(\rho K)}{\partial t}+\nabla\cdot(\rho \boldsymbol{U} K)+\nabla\cdot(\boldsymbol{U}\, p)=-\nabla\cdot\boldsymbol{q}+\rho r+\nabla\cdot(\boldsymbol{\tau}\cdot\boldsymbol{U})+\rho\boldsymbol{g}\cdot\boldsymbol{U}$$

An alternative form of the conservation of total energy is:

$$\frac{\partial(\rho E)}{\partial t}+\nabla\cdot(\rho \boldsymbol{U} E)+\nabla\cdot(\boldsymbol{U} p)=-\nabla\cdot\boldsymbol{q}+\rho\, r+\nabla\cdot(\boldsymbol{\tau}\cdot\boldsymbol{U})+\rho\boldsymbol{g}\cdot\boldsymbol{U}$$

where Total Energy E is defined as

$$E \equiv e + K$$

Energy Equation in OpenFOAM solvers

In OpenFOAM, the energy equation is generally implemented in the form of

- Total Energy (see slide 13)

$$\frac{\partial(\rho e)}{\partial t} + \nabla \cdot (\rho \boldsymbol{U} e) + \frac{\partial(\rho K)}{\partial t} + \nabla \cdot (\rho \boldsymbol{U} K) + \nabla \cdot (\boldsymbol{U}\, p) = \alpha_{\text{eff}} \nabla e + \rho\, r$$

- Total Enthalpy (see slide 15)

$$\frac{\partial(\rho h)}{\partial t} + \nabla \cdot (\rho \boldsymbol{U} h) + \frac{\partial(\rho K)}{\partial t} + \nabla \cdot (\rho \boldsymbol{U} K) - \frac{\partial p}{\partial t} = \alpha_{\text{eff}} \nabla e + \rho\, r$$

With respect to its complete form (slides 13 and 15), in the implementation of the energy equation in OpenFOAM:

- **mechanical sources** $\nabla \cdot (\tau \cdot \boldsymbol{U})$ and $\rho \boldsymbol{g} \cdot \boldsymbol{U}$ are neglected;
- a **heat flux** $q = -\alpha_{\text{eff}} \nabla e$ is assumed, where the effective thermal diffusivity α_{eff} is the sum of laminar and turbulent thermal diffusivities;
- **thermal source terms** $\rho\, r$ are specific to the particular solver.

Note: the `rhoCentralFoam` solver includes an implementation of an energy equation best represented by slide 13 that includes the mechanical source $\nabla \cdot (\tau \cdot \boldsymbol{U})$. But `rhoCentralFoam` is an explicit solver and it is not the most suitable for the kind of problems discussed in this class.

Energy Equation in OpenFOAM Solvers

The solution of the energy equation is included in several solvers in OpenFOAM for compressible flow, combustion, heat transfer, multiphase flow and particle tracking. The source code can be found for these solvers within files in sub-directories of the `$FOAM_SOLVERS` directory of OpenFOAM (including the `compressible`, `combustion`, `heatTransfer`, `multiphase` and `lagrangian` sub-directories).

Energy Equation in OpenFOAM Solvers

More commonly, in the the `rhoPimple`-based solvers, the energy equation is implemented in terms of both internal energy e and enthalpy h, allowing the user to choose the solution variable, e or h, at run time. The general implementation of the equation is the following:

```
volScalarField& he = thermo.he();

fvScalarMatrix EEqn
(
    fvm::ddt(rho, he) + fvm::div(phi, he)
  + fvc::ddt(rho, K) + fvc::div(phi, K)
  + (
        he.name() == ''e''
      ? fvc::div
        (
            fvc::absolute(phi/fvc::interpolate(rho), U),
            p,
            ''div(phiv,p)''
        )
      : -dpdt
    )
  - fvm::laplacian(turbulence->alphaEff(), he)
 ==
    fvOptions(rho, he)
);
```

Here, `he` represents either *h* or *e*. The fifth term switches between $\nabla \cdot (\boldsymbol{U}\, p)$ and $-\frac{\partial p}{\partial t}$, depending on the solution variable chosen by the user.

Energy Equation - Source Terms

The equation of energy in OpenFOAM is written for open and closed systems. Please note that for any specific problem, formulations of the energy equations differ in the sink term ρr, which in OpenFOAM is is represented by the term:

```
fvOptions(rho, he)
```

As we will see, `fvOptions` is the C++ class to write/insert source terms to ANY governing equation (mass, energy, momentum). Most of the times, what you need is in the class `fvOptions`!

Thermophysical Properties

Thermophysical Properties

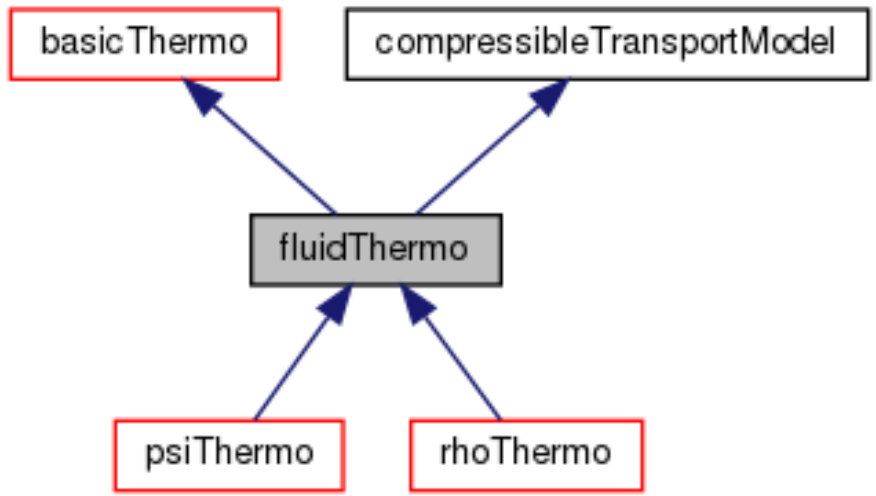
Thermophysical models are concerned with energy, heat and physical properties and they are needed for the closure of the system of governing equations.

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

A thermophysical model is constructed in OpenFOAM as a pressure-temperature p-T system from which other properties are computed. There is one compulsory dictionary entry in the file `constant/thermophysicalProperties` called `thermoType` which specifies the package of thermophysical modelling that is used in the simulation.

Thermophysical Properties

Thermophysical models are grouped into two families:



1) **thermophysical models based on compressibility** $\psi = \frac{1}{RT}$.

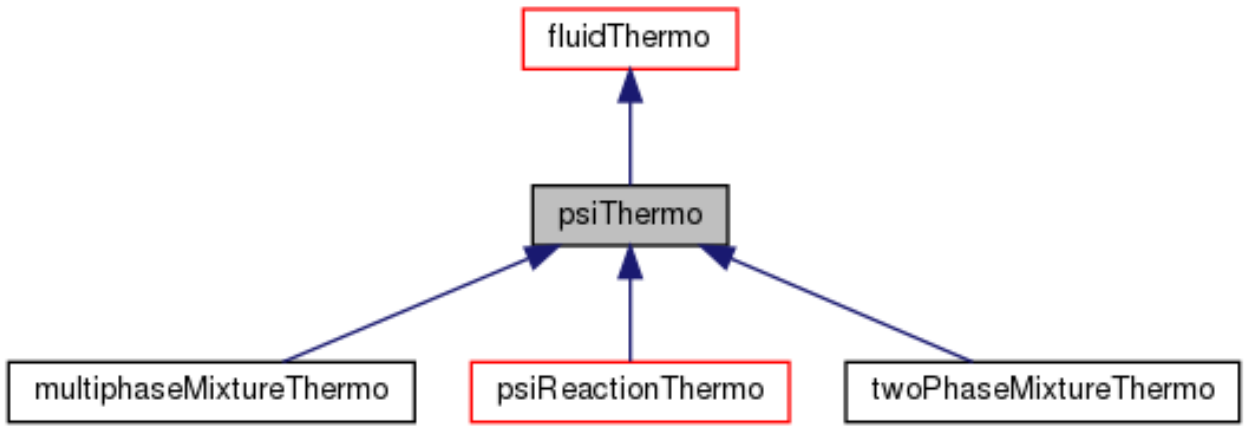
2) **thermophysical models based on density** ρ: In this case density is NOT calculated by an equation of state, but is calculated somewhere else (in the transport equation `rhoEqn.H`, for instance).

`psiThermo` template class

`psiThermo`: thermophysical models based on compressibility $\psi = \frac{1}{RT}$, where density is calculated as:

$$\rho = \psi \cdot p$$

and the compressibility ψ is provided by the equation of state.

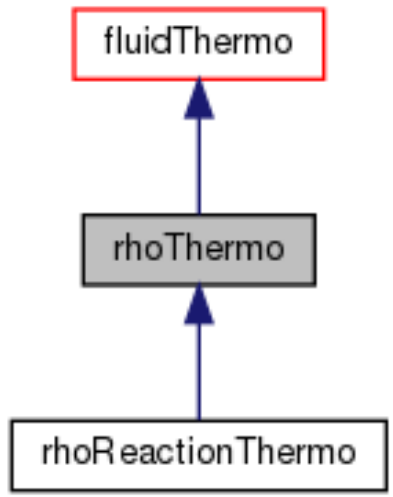


In OpenFOAM, ρ, ψ and μ are stored as `volScalarField`(s) by the `psiThermo` abstract class.

NOTE: ν=μ/ρ is defined in `fluidThermo`.

`rhoThermo` template class

`rhoThermo`: thermophysical models based on density ρ. In this case density is NOT calculated by an equation of state, but is calculated somewhere else (in the transport equation `rhoEqn.H`)



IMPORTANT NOTES:

- the solvers that construct `rhoThermo` include the heatTransfer family of solvers (`buoyantSimpl` CHT solvers, etc. , excluding Boussinesq solvers) and `rhoPorousSimpleFoam`, `twoPhaseEule` and `thermoFoam`.
- The solvers that construct `rhoReactionThermo` include some combustion solvers, e.g. rhoReactingFoam, rhoReactingBuoyantFoam, and some lagrangian solvers, e.g. reactingParcelFoam and simpleReactingParcelFoam.

Density vs compressibility based `thermo`

To SUMMARIZE, thermodynamics in OpenFOAM is handled by:

`rhoThermo`	`basic thermodynamic properties based on density`
`psiThermo`	`basic thermodynamic properties based on compressibility`

Both `rhoThermo` and `psiThermo` calculate the properties ρ, ψ and μ, BUT:

- `rhoThermo` directly defines density ρ.
- `psiThermo` calculates ρ from pressure and compressiblity ($\rho = p \cdot \psi$),

IMPORTANT NOTE: the way the pressure equation is written depends on the approach used to model thermodynamics in OpenFOAM.

The choice of the thermophysical model depends on the application, as it will be outlined in the next slides.

Density-based thermodynamics

`rhoThermo`

Buoyant solvers, manly changing their density due to temperature changes, use the `rhoThermo` model. They make use of a form of the pressure equation where ψ is included in explicit fashion.

Large pressure jumps are not represented correctly by density-based thermodynamics; this approach is valid for liquid of for gas with low compressibility, working under low pressure differences.

Density-based thermodynamics (`rhoThermo`) is less and less used in OpenFOAM solvers.

$\rightarrow$ **NOTE**: `rhoThermo` (density-based thermodynamics) $\neq$ `heRhoThermo`, that is the class for handling energy for a mixture based on density.

Compressibility-based thermodynamics

In contrast, most of the solvers (including transsonic solvers) in OpenFOAM use the `psiThermo` model. They have another pressure equation where the change in time of pressure is treated implicitly:

```
fvScalarMatrix p_rghDDtEqn
(
    fvc::ddt(rho) + psi*correction(fvm::ddt(p_rgh))
  + fvc::div(phiHbyA)
  ==
    fvOptions(psi, p_rgh, rho.name())
);
```

that reads:

$$\nabla^2 \, p^{n+1} = \frac{1}{\Delta t} \left[\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho^* \mathbf{u}^*) \right]$$

Larger pressure jumps are now possible. Such solvers are used when the flow is mainly driven by pressure changes and temperature change is only an effect of large compression and expansion. To conclude: use `rhoThermo` for fluids with low compressibility like water where the density change is mostly due to temperature variation. The `psiThermo` model should be used for gas.

New templating: `fluidThermo`

New templating: `fluidThermo`

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

The `type` keyword specifies the underlying thermophysical model; options are:

1) `hePsiThermo`: for solvers that construct `psiThermo` and `psiReactionThermo`;
2) `heRhoThermo`: for solvers that construct
   - `rhoThermo` → heat transfer
   - `rhoReactionThermo` → combustion/lagrangian solvers (non-premixed combustion)
   - `multiphaseMixtureThermo` → compressible multiphase interface-capturing solvers
3) `heheuPsiThermo`: → solvers that construct `psiuReactionThermo` (e.g. premixed combustion)

New templating: `fluidThermo`

```
thermoType
{
    type            <ThermoType>;
    mixture         <MixtureType>;
    ....
}
```

New class `hePsiThermo` includes:

`psiThermo`

Thermophysical model for fixed composition, based on compressibility ψ=$(RT)^1$, where R is Gas Constant and T is temperature. The solvers that construct `psiThermo` include the compressible family of solvers (`sonicFoam`, `simpleFoam`, etc. , excluding `rhoPorousSimpleFoam`) and `uncoupledKinematicParcelFoam` and `coldEngineFoam`

`psiReactionThermo`

Thermophysical model for reacting mixture, based on ψ. The solvers that construct `psiReactionThermo` include many of the combustion solvers, e.g. `sprayFoam`, `chemFoam`, `fireFoam` and `reactingFoam`, and some lagrangian solvers, e.g. `coalChemistryFoam` and `reactingParcelFilmFoam`.

New templating: `fluidThermo`

```
thermoType
{
    type            <ThermoType>;
    mixture         <MixtureType>;
    ....
}
```

New class `heRhoThermo` includes:

`rhoThermo`

Thermophysical model for fixed composition, based on density ρ. The solvers that construct `rhoThermo` include the heatTransfer family of solvers (`buoyantSimpleFoam`, CHT solvers, excluding Boussinesq solvers) and `rhoPorousSimpleFoam`, `twoPhaseEuler` and `thermoFoam`.

`rhoReactionThermo`

Thermophysical model for reacting mixture, based on ρ. The solvers that construct `rhoReactionThermo` include some solvers for non-premixed combustion.

`multiphaseMixtureThermo`

Thermophysical models for multiple phases. The solvers that construct `multiphaseMixture` include compressible multiphase interface-capturing solvers.

New templating: `fluidThermo`

```
thermoType
{
    type            <ThermoType>;
    mixture         <MixtureType>;
    ....
}
```

New class `psiuReactionThermo` includes:

`psiuReactionThermo`

Thermophysical model for combustion, based on compressibility of unburnt gas ψ_u. The solvers that construct `psiuReactionThermo` include combustion solvers that model combustion based on laminar flame speed and regress variable, e.g. `XiFoam`, `PDRFoam` and `engineFoam`.

Transport model

```
thermoType
{
    ...
    transport       <transportType>;
    thermo          <thermoType>;
    ...
}
```

The transport modelling concerns evaluating dynamic viscosity μ, thermal conductivity k and thermal diffusivity α (for internal energy and enthalpy equations). The current transport models are as follows:

1) **const**

   assumes a constant μ and Prandtl number $Pr = c_p\mu/k$ which is simply specified by a two keywords, μ and Pr, respectively.

2) **sutherland**

   calculates μ as a function of temperature T from a Sutherland coefficient A_s and Sutherland temperature T_s, specified by keywords A_s and T_s; μ is calculated according to

$$\mu = \frac{A_s\sqrt{T}}{1 + T_s/T}$$

Transport model

```
thermoType
{
    ...
    transport       <transportType>;
    thermo          <thermoType>;
    ...
}
```

3) **polynomial**

calculates μ and k as a function of temperature T from a polynomial of any order N:

$$\mu = \sum_{i=0}^{N-1} a_i T^i$$

4) **logPolynomial**

calculates $ln(\mu)$ and $ln(k)$ as a function of ln(T) from a polynomial of any order N; from which ν, k are calculated by taking the exponential:

$$ln(\mu) = \sum_{i=0}^{N-1} a_i [ln(T)]^i$$

energy variable

```
thermoType
{
    ...
    specie          specie;
    energy          <energyVariable>
}
```

- The user must specify the form of energy to be used in the solution, either internal energy `e` and enthalpy `h`, and in forms that include the heat of formation Δh_f or not. This choice is specified through the energy keyword.

- We refer to absolute energy where heat of formation is included, and sensible energy where it is not. For example absolute enthalpy `h` is related to sensible enthalpy h_s by

$$h = h_s + \sum_i c_i \Delta h_f^i$$

where c_i and h_i^f are the molar fraction and heat of formation, respectively, of specie i. In most cases, we use the sensible form of energy, for which it is easier to account for energy change due to reactions. Keyword entries for energy therefore include e.g. `sensibleEnthalpy`, `sensibleInternalEnergy` and `absoluteEnthalpy`.

Composition of each constituent

```
thermoType
{
    ...
    specie          specie;
    ...
}
```

There is currently only one option for the specie model which specifies the composition of each constituent. That model is itself named specie, which is specified by the following entries.

- `nMoles`: number of moles of component. This entry is only used for combustion modelling based on regress variable with a homogeneous mixture of reactants; otherwise it is set to 1.
- molWeight in grams per mole of specie.

Thank you for your attention!

contact: federico.piscaglia@polimi.it