Class EnthalpyFD

# Governing equation

,

in unit . Temperature at liquid-vapor interface is set at the saturation temperature. Based on the color function ,  is given as:

.

# Discretization

## Spatial discretization



The diffusion term is discretized in space with the finite difference method.

The convection term is discretized in space with hybrid of the finite difference method and the finite volume method.



## Discretization in time

Backward Euler method (1st order) is used for the diffusion term, while forward Euler method (1st order) is used for the convection term.

# Structure

enthalpyfd.h

|——enthaplyfd : constructor

|——discretize : discretize

| '——create\_system : create matrix

| |——create\_system\_innertial : 

| |——create\_system\_diffusive : matrix for diffusion

| | '——diff\_matrix : coefficient for matrix

| '——create\_system\_bnd : boundary condition

|

|——new\_time\_step : calculate right hand side

| |——convection : convection term

| | '——setflag : flagging

| '——diffusion\_fd : diffusion term, explicit part

| '——diff\_matrix : coefficient for matrix

|

|——solve : solve linear system

| '——update\_rhs : update right-hand side

|

'——solve\_sor : solve linear system with SOR

(for emergency)

# Prototype etc.

## Create object

In case with solid region:

EnthalpyFD enthFD ( const Scalar & tpr, const Scalar & q, const Scalar & c,

const Vector &uvw, Times & time,

Krylov \* solver, Matter \* mixed ,

const real tsat, Matter \* solid);

In case without solid region:

EnthalpyFD enthFD ( const Scalar & tpr, const Scalar & q, const Scalar & c,

const Vector &uvw, Times & time,

Krylov \* solver, Matter \* mixed ,

const real tsat);

## discretize and new\_time\_step

In case of laminar flow:

enthFD.discretize ();

enthFD.new\_time\_step ();

In case of turbulent flow:

enthFD.discretize ( & mu\_t );

enthFD.new\_time\_step (& mu\_t);

## solve

enthFD.solve(ResRat(1e-16),"enthFD");

If Krylov solver does not converge, try to use SOR.

enthFD.solve\_sor(16, 1.2,"enthFD");

The first argument is number of iteration, the second is omega used in SOR.

# Variables

|  |  |  |  |
| --- | --- | --- | --- |
| Variable | Form | Description | Unit |
| tpr | Scalar | temperature | K or ˚C |
| q | Scalar | source term for enthalpy conservation equation  WARNING: q[W] is converted to [W/m3] in update\_rhs | W |
| c | Scalar | color function |  |
| uvw | Vector | flow velocity | m/s |
| time | Times | object of Times |  |
| tsat | real | saturation temperature | K or ˚C |
| turbP | real | turbulent Prandtl number |  |

# Member functions

## diff\_matrix

Calculate coefficient for diffusion matrix.



### In case: s-f-f (solid-fluid-fluid)



Calculate :















Calculate coefficients:

Finite difference method





Finite volume method

Bottom



### In case: s-s-f (solid-solid-fluid)



Calculate :



Calculate coefficients:

