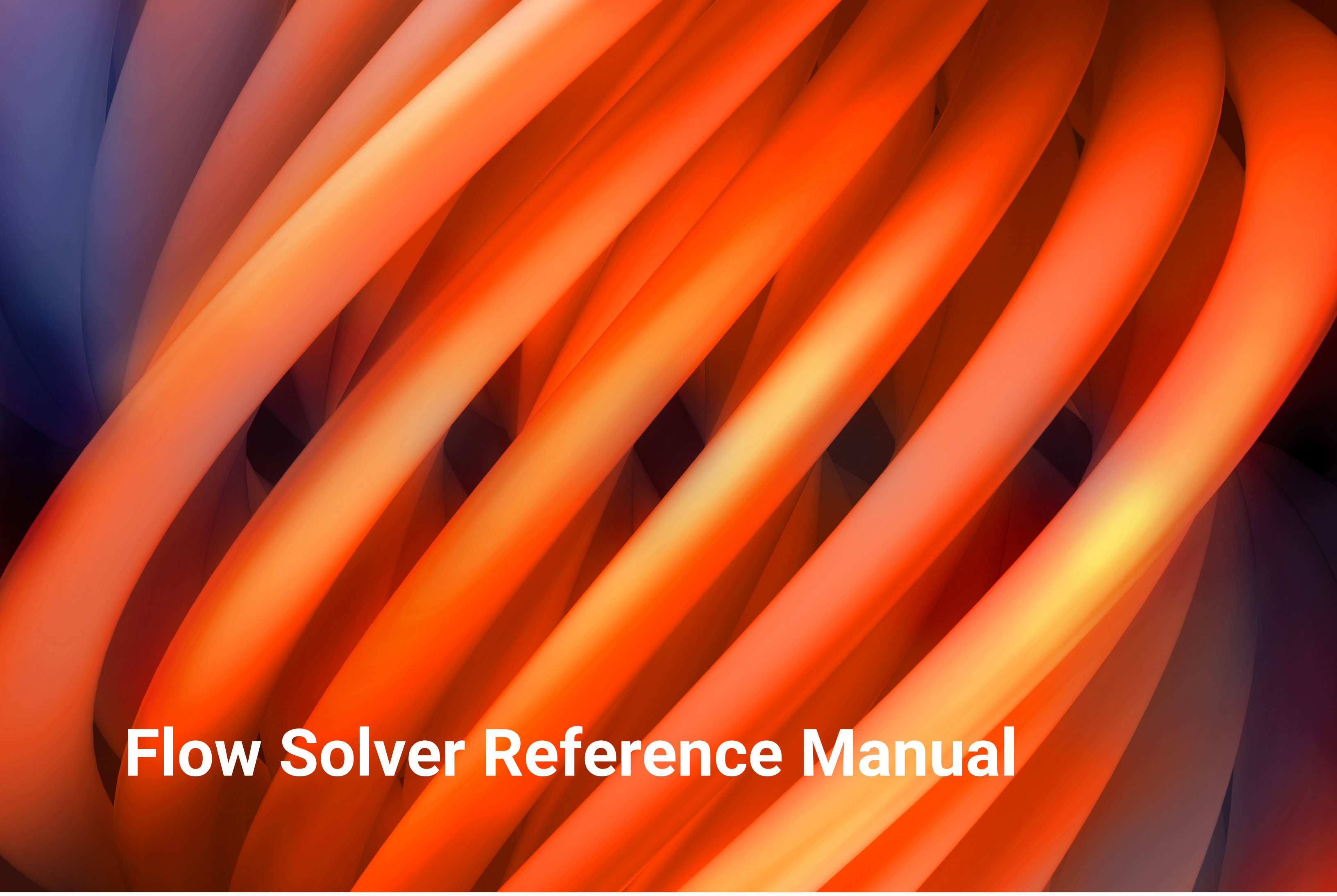


Flow Solver Reference Manual

June 2024

Maya HTT Better thinking Better future.

1 Proprietary & Restricted Rights Notice

© 2023 Maya Heat Transfer Technologies Ltd. All Rights Reserved. This software and related documentation are proprietary to Maya Heat Transfer Technologies LTD.

All trademarks belong to their respective holders.

2 Table of Contents

1 Proprietary & Restricted Rights Notice 2
2 Table of Contents 3
3 Introducing the Flow Solver Reference Manual 6
4 Nomenclature 7
5 Governing equations 14
5.1 Mass and momentum equations 14
5.2 Energy equation 15
5.3 Scalar equation 17
5.4 Equations of state 19
6 Source term 22
6.1 Buoyancy force 22
6.2 Flow resistance through porous materials 24
6.3 Heat sources 26
7 Turbulence models 27
7.1 Laminar flow 27
7.2 Turbulent flow 27
7.3 Assumptions 28
7.4 Mixing length turbulence model 28
7.5 Standard k-epsilon model 30
7.6 RNG k-epsilon model 31
7.7 Realizable k-epsilon model 32
7.8 k-omega model 34
7.9 Shear stress transport model 36
7.10 Spalart-Allmaras model 38
7.11 Initializing turbulence models 39
7.12 Turbulence model tuning coefficients 41

7.13 Turbulent structures 44

8 Boundary conditions 46

8.1 Walls 46

8.2 Fans 75

8.3 Openings 83

8.4 Convective outflow 86

8.5 Bursting membrane and flap 87

8.6 High speed flow boundary condition 90

8.7 Screens 91

8.8 Symmetry boundaries 95

8.9 Periodic boundaries 95

8.10 Mixing plane 97

8.11 Rotating frames of reference 98

8.12 1D network connected to 3D fluid region 100

9 Modeling a gas mixtures 103

9.1 Pressure and density of the gas mixture 103

9.2 Specific heat at constant pressure of the gas mixture 103

9.3 Thermal conductivity and dynamic viscosity of the gas mixture 104

9.4 Species enthalpy diffusion in the homogeneous gas mixture 105

10 Modeling humidity 107

10.1 Conversion from relative humidity to mass ratio 107

10.2 Conversion from specific humidity to mass ratio 108

10.3 Condensation and evaporation 109

10.4 Fog formation 110

10.5 Semi-permeable flow surface 112

11 Modeling non-Newtonian fluid 115

11.1 Non-Newtonian fluids 115

11.2 Power-Law model 115

11.3 Herschel-Bulkley model 115
11.4 Carreau model 116
12 Modeling particle tracking 117
12.1 Modeling of the particle motion 117
12.2 Drag force 118
12.3 Buoyancy 119
12.4 Non-drag forces 119
12.5 Brownian and turbulent diffusion 120
12.6 Particle motion equations 121
12.7 Particle slip correction factor 122
12.8 Initial conditions for injected particles 123
12.9 Boundary treatment 123
12.10 Wall boundaries 124
12.11 Inlet and symmetry boundaries 125
12.12 Periodic boundaries for particle tracking 125
13 Modeling fluid structure interaction 127
13.1 Transient fluid structure interaction 127
13.2 Static fluid structure interaction 128
14 Discretization 129
14.1 Control-volume method 129
14.2 Spatial discretization 130
14.3 Temporal discretization 140
14.4 Other discretization methods 143
15 Solver numerical methods 151
15.1 Linear equation solution 151
15.2 Solver methods 151
16 Flow solver files 154
17 Flow solver references 157

3 Introducing the Flow Solver Reference Manual

The flow solver computes a solution to the non-linear, partial differential equations for the conservation of mass, momentum, energy, and general scalars in general complex 3D geometries. It uses an element-based finite volume method and iterative Krylov methods to discretize and solve the governing equations.

Physical models include laminar or turbulent, incompressible or compressible flow, natural convection, rotating frame of reference, non-newtonian fluids, porous blockages, mixtures, humidity, condensation and evaporation at walls, and general boundary conditions for fluid flow and heat transfer. Details of the mathematical model, the discretization of the equations, and of the solution method used in the flow solver are presented in the following sections of this document.

4 Nomenclature

Symbol	Description	Dimensions
A	Area	m^2
A_ϵ	Damping function constant	3.8
A_μ	Damping function constant	63
C_μ	k-ε turbulence model constant	*0.09*
C	Cunningham correction factor	-
$C_{\varepsilon 1}$	k-ε turbulence model constant	*1.44*
$C_{\varepsilon 2}$	k-ε turbulence model constant	*1.92*
C_D	Drag coefficient	-
C_A	Added mass coefficient	-
c_p	Specific heat at constant pressure	*J/kg K*
D_B	Brownian diffusivity	m^2/s
D_p	Particle diffusivity	m^2/s
D_T	Turbulent diffusivity	m^2/s
D_v	Water vapor diffusivity	m^2/s
D_ϕ	Scalar diffusion coefficient	m^2/s
d	Diameter	*m*

Symbol	Description	Dimensions
E	k-ε turbulence model constant	*8.43*
e	Internal energy	*J*
$\mathbf{F}$	Convective flux term	-
f_ϵ	Damping function	-
f_l	Damping factor	-
f_μ	Van Driest damping factor	-
f_ω	Damping function	-
$\mathbf{G}$	Viscous flux term	-
g	Gravitational acceleration constant	*m^2/s*
h	Enthalpy of the fluid	*J/kg*
h	Heat transfer coefficient	*W/m^2 K*
h	Head loss coefficient	-
h_0	Total enthalpy	*J/kg*
I_x	Turbulence intensity	-
$\boldsymbol{J}$	Diffusive mass flux	*kg/m^2 s*
K	Permeability	*m^2*
K	Consistency index	-

Symbol	Description	Dimensions
Kn	Knudsen number	-
k	Turbulence kinetic energy	*J/kg*
k	Thermal conductivity	*W/m K*
k_s	Specified equivalent sand grain roughness.	*m*
k_B	Boltzmann constant	*1.3806488*10^{-23} J/K*
l	Turbulent eddy length scale	*m*
$\dot{m}$	Mass flow rate	*kg/s*
m''	Mass flux	*kg/m^{2}s*
$\vec{N}$	Unit normal vector	-
Nu	Nusselt number	-
P	Pressure	*Pa*
Pr	Prandtl number	-
q	Heat flux	*W/m^2*
R	Universal gas constant	8.3144 *J/mol K*
R	Location vector	*m*
R	Internal loss coefficient	*m^{-1}*
R	Humidity diffusion resistance	*s/m*

Symbol	Description	Dimensions
$\mathbf{R}$	Discrete convective flux	-
R_g	Specific gas constant	*J/kg* K
Ra	Rayleigh number	-
Re	Reynolds number	-
r	Radius	m
S	Modulus of the mean strain rate	-
S	Sutherland constant	-
S	Closed surface	m^2
S_{ij}	Mean rate-of-stain tensor	s^{-1}
S_h	Energy source term	W/m^3
S_m	Mass source	$kg/m^3\ s$
S_U	Momentum source	N/m^3
Sc	Schmidt number	-
T	Temperature	*K*
$\mathbf{T}$	Discrete viscous flux	-
t	Time	s
U	Velocity	*m/s*

Symbol	Description	Dimensions
$\mathbf{U}$	Conserved quantity	-
U_m	Mean flow velocity	*m/s*
U_r	Relative velocity magnitude from a rotating frame	*m/s*
u	Fluctuating velocity	*m/s*
u^*	Shear velocity	*m/s*
V_d	Volume of the domain	m^3
V_e	Mesh element volume	m^3
V_m	Molar volume of the gas	m^3/mol
V_p	Volume of a particle	m^3
V	Volume of the fluid domain	m^3
$\dot{v}$	Volume flow rate	m^3/s
w	Weighting function	-
$\vec{\mathrm{x}}_p$	Particle centroid trajectory	*m*
$\dot{\mathbf{x}}$	Velocity of the control volume	m/s
X_i	Molar fraction	-
y^+	Non-dimensional wall distance	-
α	Thermal diffusivity	m^2/s

Symbol	Description	Dimensions
α	Coefficient of restitution	-
β	Thermal expansion coefficient	K^{-1}
Γ	Effective diffusion coefficient	-
$\dot{\gamma}$	Shear rate	s^{-1}
δ_{ij}	Kronecker delta function	-
ε	Dissipation rate of the turbulent kinetic energy	m^3/s^3
ε	Smoothing factor	-
η	Avogadro's number	$6.02214*10^{23}$ mol^{-1}
κ	Von Karman constant	0.41
λ	Time constant	s
λ	Mean free path of the particle	m
μ	Dynamic viscosity	Pa s
μ_t	Turbulent viscosity	Pa s
ν	Kinematic viscosity	m^2/s
Ω	Rotational velocity vector of a rotating frame	m/s
ρ	Density	kg/m^3
ρ_v	Water vapor density	kg/m^3

Symbol	Description	Dimensions
σ_k	Turbulence model constant	*1.0*
σ_ε	Turbulence model constant	*1.3*
τ_w	Wall shear stress	*Pa*
τ_0	Yield stress	*Pa*
ϕ	Mass fraction	-
ϕ	Angle	*degrees*
ϕ	Radial basis function	-
φ_r	Relative humidity	-
φ_s	Specific humidity	-
ω	Specific dissipation rate	s^{-1}
$\vec{\omega}$	Rotational velocity vector	*rad/s*

5 Governing equations

A model can be comprised of multiple flow enclosures. Each enclosure is a separate group of three-dimensional elements that form a separate fluid domain from the remaining geometry. The physical scales of each enclosure, such as the length scale, time scale, pressure, and temperature are independent of the other enclosures. The flow solver solves governing equations separately for each flow enclosure.

5.1 Mass and momentum equations

The flow solver solves conservation equations for mass and momentum for general flow, which can be expressed in Cartesian coordinates and using the tensorial notation [1, 2, 3] as follows:

$$\frac{\partial \rho}{\partial t} + \frac{\partial(\rho U_j)}{\partial x_j} = 0$$

Mass conservation equation

$$\frac{\partial(\rho U_j)}{\partial t} + \frac{\partial(\rho U_i U_j)}{\partial x_i} = -\frac{\partial P}{\partial x_j} + \frac{\partial}{\partial x_i}\left(\mu\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) - \overline{\rho u_i u_j}\right) + S_{U_j}$$

Momentum conservation equation

In these equations:

- The Einstein convention is used (i, j, k = 1,2,3).
- U_j and u_j are the components of the mean and the fluctuating velocity in the x_j direction.
- P is the pressure.
- ρ is the density of the fluid.
- μ is the dynamic viscosity of the fluid.
- $-\overline{\rho u_i u_j}$ represents the turbulent (or Reynolds) stresses.
- S_{U_j} is the source terms for momentum equation.

These equations are valid for both, incompressible and compressible flows.

The terms of the momentum conservation equation are:

- The transient term
- The convection term
- The pressure gradient term
- The stress term
- The source term

The source term S_{U_j} can represent body forces or flow resistance forces:

- For natural convection flows, S_{U_j} includes the buoyancy force. See Buoyancy force for more information.

- For flows through porous blockages, S_{U_j} contains additional resistance terms. See Flow resistance through porous materials for more information.

5.2 Energy equation

The instantaneous total energy equation [1] in tensorial notation is:

$$\frac{\partial\left[\rho\left(e+1/2U^2\right)\right]}{\partial t}+\frac{\partial\left[\rho U_i\left(e+1/2U^2\right)\right]}{\partial x_i}=-\frac{\partial q_i}{\partial x_i}-\frac{\partial PU_i}{\partial x_i}-\frac{\partial}{\partial x_j}\left[U_i\mu\left(\frac{\partial U_i}{\partial x_j}+\frac{\partial U_j}{\partial x_i}\right)\right]+q'$$

Total energy conservation equation

where:

- e is the internal energy.
- U is the velocity magnitude.
- q_i is the heat flux in the direction x_i .
- q' is a heat generation or heat sink per unit volume.

In energy conservation equation, the terms represent, in order:

- The rate of energy gain per unit volume
- The rate of energy input per unit volume due to convection
- The rate of energy addition due to conduction
- The rate at which work is done on the fluid by pressure
- The rate of energy addition due to viscous forces (dissipation term)
- The rate of heat generation by internal sources

The second and fourth terms in the above equation can be combined and using the definition of the total enthalpy, h_o as:

$$h_o=h+\frac{1}{2}U^2=e+\frac{P}{\rho}+\frac{1}{2}U^2$$

where h is the static enthalpy of the fluid and combining with total energy conservation equation, gives:

$$\frac{\partial(\rho h_o)}{\partial t}+\frac{\partial(\rho U_j h_o)}{\partial x_j}=\frac{\partial P}{\partial t}+\frac{\partial q_j}{\partial x_j}-\frac{\partial}{\partial x_j}\left[U_i\mu\left(\frac{\partial U_i}{\partial x_j}+\frac{\partial U_j}{\partial x_i}\right)\right]+S_h$$

Energy conservation equation

where S_h is the energy source term.

5.2.1 High speed equation

After modeling the conduction and taking the Reynolds average of energy conservation equation [1], it becomes:

$$\frac{\partial(\rho h_o)}{\partial t} + \frac{\partial(\rho U_j h_o)}{\partial x_j} = \frac{\partial P}{\partial t} + \frac{\partial}{\partial x_j}\left(k\frac{\partial T}{\partial x_j} - \overline{\rho u_j h'}\right) - \frac{\partial}{\partial x_j}\left(U_i\left[\mu\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) - \overline{\rho u_i u_j}\right]\right) + S_h$$

This equation expresses the conservation of the total energy for example, the thermal energy plus the mechanical energy. It is valid for all flow situations, but you should limit it to supersonic flow regimes. Because the flow solver is not density based, hypersonic flows are not supported.

5.2.2 Low speed equation

For low speed flows (Mach < 0.3), the energy equation is simplified for robustness and for round-off purposes. For low speed incompressible and compressible flows, the pressure work and dissipation terms in energy conservation equation are neglected. The simplified form of the energy equation has the mechanical energy subtracted from the total energy, and becomes a thermal energy equation [2]:

$$\frac{\partial(\rho h)}{\partial t} + \frac{\partial(\rho U_j h)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(k\frac{\partial T}{\partial x_j} - \overline{\rho u_j h'}\right) + S_h$$

where:

- k is the thermal conductivity.
- h' is the fluctuating static enthalpy.
- $-\overline{\rho u_j h'}$ is the turbulent or Reynolds flux.

This form of the equation ensures conservation of the thermal energy, and avoids round-off problems. The low speed form of the energy equation is used by default in the flow solver.

The transient term in the low-speed form of the energy equation should represent storage of internal energy, however, by default, it is implemented as a storage of enthalpy. In transient cases where the pressure changes in time, such as tank charging problems with ideal gas fluids, this approximation may lead to unacceptable errors. You can include pressure time derivative, $\frac{\partial P}{\partial t}$, to the right side of the energy equation by using the `INCLUDE DP/DT TERM IN LOW-SPEED ENERGY EQUATION` advance parameter, so that the correct energy storage rate is recovered.

The viscous heating term (dissipation term) can be activated for the low speed energy equation by using the `INCLUDE_VISCOUS_HEATING` advanced parameter. This term should be included to the energy equation for the flows with high velocity gradients and incompressible flows. The term is computed on nodes based on the contributions of all surrounding nodes. In addition, a correction is added to account for the different flux calculation across a wall boundary. In this case this wall flux value is added to the existing value of nodes on wall boundaries.

5.3 Scalar equation

In addition to the mass, momentum, and energy equations, the flow solver solves general scalar equations that have the following form:

$$\frac{\partial(\rho\phi)}{\partial t} + \frac{\partial(\rho U_j \phi)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\rho D_\phi \frac{\partial\phi}{\partial x_j} - \overline{\rho u_j \phi'}\right)$$

where the mass fraction is defined as:

$$\phi = \frac{\rho_{\phi_i}}{\rho} = \frac{\rho_{\phi_i}}{\rho_a + \sum_i \rho_{\phi_i}}$$

- ρ_a , ρ_{ϕ_i} , and ρ represent the density of the main fluid, the density of the scalar for gas i , and the density of the fluid mixture, respectively.
- D_ϕ is the scalar diffusion coefficient.
- $-\overline{\rho u_j \phi'}$ is the turbulent or Reynolds flux where ϕ' is the fluctuating mass fraction of the scalar.

The left hand side terms of the scalar equation are the transient term and the convective term, respectively, and the right hand side term is the diffusion term.

5.3.1 Modeling a passive scalar

The general scalar equation can be used to model a passive scalar. In this case, it is assumed that the second component is present in the main fluid in very small quantities.

- $\phi << 1$
- $\rho \approx \rho_a$

The passive scalar does not have any influence on the flow. The fluid properties are that of the main fluid, and the properties of the second component are not needed.

The general scalar equation is also used for:

- Modeling a gas mixture
- Modeling humidity

5.3.2 Temperature dependent diffusion coefficient

The flow solver supports temperature-dependent diffusion coefficient in active and passive scalar equations.

To use this functionality, do the following before you run your simulation:

Step 1. Add the following line in the *user.prm* file.

For more information on the *user.prm* file, see Flow solver files.

```
VARIABLE_MOLECULAR_DIFFUSION_COEFFICIENT = TRUE
```

Step 2. Specify the table for the molecular diffusion coefficient.

You specify the table in the *flow* `.vdc` file, which you create using a text editor in your run directory. The following table lists the field values that you need to specify in the *flow* `.vdc` file and indicates the values used in the example below:

Field	Value	Description	Value from example
S	int	The number of tables	1
Name_S	char	The name of the affected scalar S	Carbon_Dioxide_Gas
N	int	Length of the table	8
T_0 D_0	double	Temperature, T_0, and molecular diffusion coefficient, D_0	0.00000 14.00000
...	...	...	...
T_N D_N	double	Temperature, T_N, and molecular diffusion coefficient, D_N	50.00000 19.00000

The temperature units must be consistent with the solution units.

Example

```
1
```

```
Carbon_Dioxide_Gas
8
  0.00000    14.00000
  7.14286    14.71429
 14.28571    15.42857
 21.42857    16.14286
 28.57143    16.85714
 35.71429    17.57143
 42.85714    18.28571
 50.00000    19.00000
```

5.4 Equations of state

An equation of state is a constitutive equation which provides a mathematical relationship between thermodynamic state variables for a given material. With the mass, momentum, and energy equations, it completes the mathematical representation of your fluid model.

The following state variables need to be defined:

- Density, ρ
- Dynamic viscosity, μ
- Specific heat a constant pressure, c_p
- Conductivity, k
- Specific enthalpy, h

You define all of these material variables except the specific enthalpy in this software. These material variables can vary with time, temperature, pressure, or they can vary with both temperature and pressure (bivariate properties).

In the case of bivariate properties, and in the absence of the standard state models such as ideal gas, the flow solver reads the user-specified bivariate table, and then performs a bi-linear interpolation with respect to temperature and pressure in order to calculate the value of the state variables.

The flow solver uses the state variables differently depending if your fluid is a gas or a liquid.

Note

The flow solver differentiates between a liquid and a gas as follows:

- If the gas constant, R_g , is defined, the fluid material is a gas.
- If the gas constant, R_g , is not defined, the fluid material is a liquid.

When your fluid is a liquid, the density ρ and the specific heat c_p are assumed constant. The flow solver calculates the specific enthalpy, h, from the energy equation, and uses the following equation to calculate the temperature T:

$$h = c_p T + \frac{P}{\rho}$$

When your fluid is a gas, an equation of state is used to model the relationship between thermodynamic state variables.

The flow solver supports the following models that you specify in the software when you define the fluid material:

- Ideal gas law
- Redlich-Kwong real gas equation of state

The flow solver calculates the specific enthalpy, h, from the energy equation, and uses the following equation to calculate the temperature T:

$$h = c_p T$$

where c_p is the specific heat at constant pressure of the gas.

5.4.1 Ideal gas law

The ideal gas law equation of state used by the flow solver is given by:

$$P = \rho R_g T$$

where:

- P is the pressure of the gas.
- T is the temperature of the gas.
- ρ is the density of the gas.
- R_g is the specific gas constant (J/kg K in SI units).

5.4.2 Redlich-Kwong real gas equation of state

The Redlich-Kwong real gas equation of state is given by:

$$P = \frac{R \cdot T}{V_m - b} - \frac{a}{\sqrt{T} V_m (V_m + b)}$$

where:

- P is the pressure of the gas.
- T is the temperature of the gas.
- ρ is the density of the gas.
- V_m is the molar volume of the gas.
- R is the universal gas constant (8.314472 J/mol K).

The constants a and b are defined as:

$$a = 0.42748 \cdot \left(R^2 \cdot \frac{T_c^{2.5}}{P_c} \right)$$
$$b = 0.08662 \cdot \left(R \cdot \frac{T_c}{P_c} \right)$$

where:

- P_c is the critical pressure of the gas.
- T_c is the critical temperature of the gas.

The Redlich-Kwong equation of state is more realistic than the ideal gas law at high pressure and valid when:

$$\frac{P}{P_c} < \frac{T}{2T_c}$$

6 Source term

6.1 Buoyancy force

For buoyancy-driven flows, the flow solver uses the following models:

- *Full gravity model* for multiphase and multicomponent flow, or for the flows with the density difference due to sources other than temperature variation
- *Boussinesq model* for the flows with density variation due to small temperature changes

6.1.1 Full gravity model

For buoyancy calculations, the gravity force is included in the source term S_{U_j} of the momentum conservation equation, as follows:

$$\frac{\partial(\rho U_j)}{\partial t} + \frac{\partial(\rho U_i U_j)}{\partial x_i} = -\frac{\partial P}{\partial x_j} + \frac{\partial}{\partial x_i}\left(\mu\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) - \overline{\rho u_i u_j}\right) + \rho g_j$$

However, incorporating the gravity force directly into the source term can lead to round-off errors. Therefore, the flow solver computes the gravity force based on the following equation:

$$\rho = \rho_r + (\rho - \rho_r)$$

where ρ_r is a reference density.

The momentum conservation equation in presence of buoyancy is re-written as:

$$\frac{\partial(\rho U_j)}{\partial t} + \frac{\partial(\rho U_j U_i)}{\partial x_i} = -\frac{\partial P^*}{\partial x_j} + \frac{\partial}{\partial x_i}\left(\mu\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) - \overline{\rho u_i u_j}\right) + (\rho - \rho_r) g_j$$

where:

$$\frac{\partial P^*}{\partial x_j} = \frac{\partial P}{\partial x_j} - \rho_r g_j$$

$$P^* = P - (P_0 + \rho_r g_j)$$

In these equations:

- P^* is the pressure field with respect to the hydrostatic variation.
- P_0 is called the offset pressure. It is the pressure when $z = 0$.

The flow solver adds the hydrostatic contribution at the openings where a relative pressure is specified and does not add it at the openings where an absolute pressure is specified. For example, you would not want to add the hydrostatic contribution at the opening at the bottom of a tank draining under gravity, but you would want to add it at the opening at the outlet of a channel through which fluid flows horizontally.

6.1.2 Boussinesq Model

The flow solver uses the Boussinesq model for the flows with the density variation due to small temperature changes. The buoyancy force per unit volume term $(\rho — \rho_r)g_j$ is modeled for incompressible liquid as follows:

$$(\rho - \rho_r)g_j \approx -\rho\beta(T - T_r)g_j$$

$$\beta = -\frac{1}{\rho}\left(\frac{\partial\rho}{\partial T}\right)_P$$

where:

- β is the coefficient of thermal expansion.
- T_r is a reference temperature, at the same condition as ρ_r .

Note

It is assumed that the pressure-density effects are negligible.

Thus, momentum conservation equation in presence of buoyancy is:

$$\frac{\partial(\rho U_j)}{\partial t} + \frac{\partial(\rho U_j U_i)}{\partial x_i} = -\frac{\partial P^*}{\partial x_j} + \frac{\partial}{\partial x_i}\left(\mu\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) - \overline{\rho u_i u_j}\right) - \rho\beta\,(T - T_r)\,g_j$$

The reference temperature, T_r , differs depending on the boundary conditions applied to the fluid domain.

6.1.3 Cavities

Enclosed cavities

In an enclosed cavity, with no communication of pressure level to the outside, T_r is arbitrary: the temperature and velocity fields are independent of reference temperature. The pressure reacts with differing linear variations superimposed on an unchanging field. In effect, different T_r values only create different hydrostatic pressure variations in P^* . T_r is selected such that the total buoyancy force summed over the whole domain is zero.

The buoyancy source term is

$$S_{U_j} = -\rho\beta g_j(T - T_r)$$

This code reference temperature is computed internally to minimize the summed effect of the buoyancy term:

$$T_r = \frac{\sum \rho\beta g T dV}{\sum \rho\beta g dV}$$

Reference temperature equation for enclosed cavities

where V is the volume of fluid, and the sums are over the whole domain. Substituting this expression for T_r into equation for the buoyancy source term and then summing the result over the whole domain gives a zero buoyancy force.

Cavities with openings

Consider a simple cavity, open to a large domain at the top and the bottom, and heated from its internal walls. Physically, natural convection results with fluid entering through the bottom and heated fluid leaving through the top.

If T_r is computed from the reference temperature equation for enclosed cavities in a model with cavities open to external fluid domains then, by design, there will be a zero total buoyancy force and with the zero force from the P^* pressure boundary conditions there will be no net movement of the fluid through the cavity. Because of this potential problem, the reference temperature for open cavities is it taken as the maximum of all the openings temperatures in the enclosure, is given by:

$$T_r = \max(T_{opening})$$

6.2 Flow resistance through porous materials

The pore structure resists the flow of the fluid passing through the porous material.

- An isotropic porous material has the same loss coefficient in all directions.
- An orthotropic porous material has three different loss coefficients that correspond to the three orthogonal principal axes.

The resistance to flow is included in the source term S_{U_j} of the momentum conservation equation as described by the Darcy-Forchheimer law [26].

6.2.1 Isotropic resistance

For an isotropic porous material, the source term accounts for the resistance to flow as follows:

$$S_{U_j} = -\frac{\mu}{K} U_j - \frac{1}{2}\rho R |\bar{U}| U_j$$

where

- K is the permeability of the porous material that you specify.
- R is the inertial loss coefficient of the porous material that you specify, and represents the fraction of the dynamic head lost per unit distance and has a dimension of inverse length.
- $|\bar{U}|$ is the magnitude of the velocity. The velocity is expressed in unit vectors of the global coordinates as $\bar{U} = U_j \bar{e}_j$.

6.2.2 Orthotropic resistance

An orthotropic material has three orthogonal principal axes $X_1,\ X_2,\ X_3$. Each axis has a different loss coefficient.

The components of the resultant resistance force per unit volume along the principal axes $X_1,\ X_2,\ X_3$ are defined as follows:

$$\begin{aligned} S_{U1} &= -\tfrac{\mu}{K_{11}} U_1 - \tfrac{1}{2}\rho R_{11} |\bar{U}| U_1 \\ S_{U2} &= -\tfrac{\mu}{K_{22}} U_2 - \tfrac{1}{2}\rho R_{22} |\bar{U}| U_2 \\ S_{U3} &= -\tfrac{\mu}{K_{33}} U_3 - \tfrac{1}{2}\rho R_{33} |\bar{U}| U_3 \end{aligned}$$

where

- R_{11} , R_{22} and R_{33} are the specified inertial loss coefficients of the porous material in the directions of X_1 , X_2 , and X_3 respectively.
- K_{11} , K_{22} and K_{33} are the specified permeabilities of the porous material in the directions of X_1 , X_2 , and X_3 respectively.
- U_1 , U_2 and U_3 are the velocity components in the directions of X_1 , X_2 , and X_3 respectively.

The equation above is written in matrix form as follows:

$$\begin{pmatrix} S_{U1} \\ S_{U2} \\ S_{U3} \end{pmatrix} = -\begin{pmatrix} \frac{\mu}{K_{11}} + \frac{1}{2}\rho R_{11}|\bar{U}| & 0 & 0 \\ 0 & \frac{\mu}{K_{22}} + \frac{1}{2}\rho R_{22}|\bar{U}| & 0 \\ 0 & 0 & \frac{\mu}{K_{33}} + \frac{1}{2}\rho R_{33}|\bar{U}| \end{pmatrix} \begin{pmatrix} U_1 \\ U_2 \\ U_3 \end{pmatrix}$$

The resistance force in the global Cartesian coordinates is:

$$\begin{pmatrix} S_{U_x} \\ S_{U_y} \\ S_{U_z} \end{pmatrix} = [T] \begin{pmatrix} S_{U1} \\ S_{U2} \\ S_{U3} \end{pmatrix}$$

where $[T]$ is the transformation matrix between the coordinates along the principal axes and the global Cartesian system. The components of the velocity along the principal axes are expressed in the global Cartesian velocity components as follows:

$$\begin{pmatrix} U_1 \\ U_2 \\ U_3 \end{pmatrix} = [T]^{-1} \begin{pmatrix} U_x \\ U_y \\ U_z \end{pmatrix} = [T]^T \begin{pmatrix} U_x \\ U_y \\ U_z \end{pmatrix}$$

where superscript ^{T} represents the transpose of the matrix.

Combining equations, the source term for the momentum conservation equation in the global Cartesian coordinate system is written as:

$$\begin{Bmatrix} S_{U_x} \\ S_{U_y} \\ S_{U_z} \end{Bmatrix} = -[T] \begin{bmatrix} \frac{\mu}{K_{11}} + \frac{1}{2}\rho R_{11}|\bar{U}| & 0 & 0 \\ 0 & \frac{\mu}{K_{22}} + \frac{1}{2}\rho R_{22}|\bar{U}| & 0 \\ 0 & 0 & \frac{\mu}{K_{33}} + \frac{1}{2}\rho R_{33}|\bar{U}| \end{bmatrix} [T]^T \begin{Bmatrix} U_x \\ U_y \\ U_z \end{Bmatrix}$$

The resistance force is nonlinear and is re-computed at each iteration.

6.3 Heat sources

The flow solver supports transient analysis with a constant or time-dependent heat source inside the fluid domain. The source term, S_h in low speed equation and high speed equation, corresponds to the heat generation per unit volume. The heat source is imposed on selected fluid elements.

7 Turbulence models

In the flow solver, the flow field can be solved as laminar or as turbulent using governing equations. The following turbulence models are available to model the Reynolds (or turbulent) stresses and fluxes:

- Mixing length turbulence model
- Standard two-equation
- RNG k-epsilon model
- Realizable k-epsilon model
- k-omega model
- Shear stress transport model (SST)
- Spalart-Allmaras turbulence model

7.1 Laminar flow

For laminar flows, the Reynolds stresses, $-\overline{\rho u_i u_j}$, and the Reynolds fluxes, $\overline{\rho u_j h'}$, $\overline{\rho u_j \phi'}$ and $-\overline{\rho_v u_j}$ are equal to zero.

7.2 Turbulent flow

All turbulent models use Boussinesq eddy viscosity assumption [4] to evaluate the Reynolds stresses and fluxes as follows:

$$-\overline{\rho u_i u_j} = \mu_t \left(\frac{\partial U_i}{\partial x_j}\right)\left(\frac{\partial U_j}{\partial x_i}\right) - \frac{2}{3}\rho k \delta_{ij}$$

$$-\overline{\rho u_j h'} = \frac{\mu_t}{\mathrm{Pr}_t}\left(\frac{\partial T}{\partial x_j}\right)$$

$$-\overline{\rho u_j \phi'} = \frac{\mu_t}{\mathrm{Sc}_t}\left(\frac{\partial \phi}{\partial x_j}\right)$$

$$-\overline{\rho_v u_j} = \frac{\mu_t}{\rho \mathrm{Sc}_t}\left(\frac{\partial \rho_v}{\partial x_j}\right)$$

In these equations:

- μ_t is the turbulent viscosity.
- k is the turbulence kinetic energy.
- δ_{ij} is a Kronecker delta function.
- Pr_t is the turbulent Prandtl number.
- Sc_t is the turbulent Schmidt number.

- $P^* = -\frac{2}{3}\rho k\delta_{ij}$ is the pressure field with respect to the hydrostatic variation. This term is included in the pressure term of the momentum equation and is not calculated explicitly.
- $-\overline{\rho_v u_j}$ is the turbulent or Reynolds flux for cases that involve mixture species transport, where the transported quantities are constituent mass fractions ρ_v/ρ contributing to the mixture density ρ.

7.3 Assumptions

The derivation of the near-wall relations for turbulent flows depends on several assumptions presented in this section. It is assumed that the flow close to the wall is in the x direction.

7.3.1 Fully developed flow

This means that all streamwise derivatives vanish, i.e. $\frac{\partial()}{\partial x} = 0$, and that the flow is parallel to the wall.

7.3.2 Constant shear layer flow

This is equivalent to requiring that there is no pressure gradient or other momentum source term.

$$-\rho\overline{U_i U_j} = \tau_{ij}$$

7.3.3 Boussinesq or eddy viscosity assumption

This states that the Reynolds stress can be expressed as the product of an effective eddy viscosity and the mean flow strain rate. Combining the Boussinesq approximation to equation above gives:

$$\tau_{ij} = \mu_t \left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right)$$

7.4 Mixing length turbulence model

Prandtl mixing length hypothesis states that the eddy viscosity could be expressed as the product of a turbulence length, $l_t = \kappa y$, and a velocity scale, $U_t = l_t \frac{dU}{dy}$, i.e.

$$\mu_t = \rho U_t l_t = \rho \kappa^2 y^2 \frac{dU}{dy}$$

The mixing length turbulence model is a zero-equation model, which uses the following relationship based on Prandtl mixing length hypothesis to calculate the turbulent viscosity:

$$\mu_t = \rho l^2 S$$

- l is the mixing length.
- S is the modulus of the mean strain rate.

The mixing length l and damping factor f_l are defined as:

$$l = \min(f_l \kappa y,\ 0.09 y_{\max})$$

$$f_l = 1 - \exp\frac{-y^+}{26}$$

$$y^+ = \frac{\rho u^* y}{\mu}$$

$$u^* = \sqrt{\frac{\tau_w}{\rho}}$$

In these equations:

- κ is the Von Karman constant ($\kappa = 0.41$).
- y is the normal distance from the node to the wall.
- y^+ is the dimensionless wall distance.
- $y_{\max}$ is a characteristic length scale for the model.
- u^* is the shear velocity.
- τ_w is the wall shear stress.

If you do not specify the length scale, the flow solver uses a default value defined as:

$$y_{\max} = 2\frac{V_d}{A_w}$$

- V_d is the volume of the fluid domain.
- A_w is the wetted area.

The flow solver computes a length scale for each fluid domain in the model.

Note

For a square or cylindrical duct, the above length scale $y_{\max}$ is equivalent to half the hydraulic diameter.

For internal nodes (i.e. nodes which are not touching a wall), the modulus of the mean strain rate is given by:

$$S = \sqrt{2S_{ij}S_{ij}}$$

$$S_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right)$$

7.5 Standard k-epsilon model

The standard k-epsilon model uses the turbulent viscosity evaluated from:

$$\mu_t = \rho C_\mu \frac{k^2}{\varepsilon}$$

- k is the turbulent kinetic energy.
- ε is the dissipation rate of the turbulent kinetic energy.
- C_μ is a k-ε turbulence model constant.
- ρ is the density of the fluid.

The turbulent kinetic energy, k, and the dissipation rate of turbulent kinetic energy, ε, are obtained by solving a conservation equation for each of these two quantities given by:

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial(\rho U_j k)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\Gamma_k \frac{\partial k}{\partial x_j}\right) + P_k + P_b - \rho\varepsilon$$

$$\frac{\partial(\rho\varepsilon)}{\partial t} + \frac{\partial(\rho U_j \varepsilon)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\Gamma_\varepsilon \frac{\partial \varepsilon}{\partial x_j}\right) + C_{\varepsilon 1}\frac{\varepsilon}{k}(P_k + C_{\varepsilon 3} P_b - C_{\varepsilon 2}\rho\varepsilon)$$

In these equations:

- Γ_k is the effective diffusion coefficient of k:

$$\Gamma_k = \mu + \frac{\mu_t}{\sigma_k}$$

- Γ_ε is the effective diffusion coefficient of ε:

$$\Gamma_\varepsilon = \mu + \frac{\mu_t}{\sigma_\varepsilon}$$

- P_k is the production rate of the turbulent kinetic energy defined as:

$$P_k = -\rho \overline{u_i u_j} \frac{\partial U_i}{\partial x_j}$$

- P_b is the production rate of the turbulent kinetic energy due to buoyancy defined as:

$$P_b = \beta g_i \frac{\mu_t}{\sigma_t} \frac{\partial T}{\partial x_i}$$

where

- β ia the coefficient of thermal expansion.
- g_i is a component of the gravity vector g in x_i direction.
- $\sigma_t = 0.87$ is a turbulent Prandtl number. You can modify it using the `TURBULENT PRANDTL NUMBER` advanced parameter.

You can disable the production term, P_b, using the `BUOYANCY TKE PRODUCTION TERM` advanced parameter.

- By default, $C_{\varepsilon 3} = 0$ means there is no turbulence dissipation due to buoyancy. You can specify the value for the $C_{\varepsilon 3}$ coefficient using the `BUOYANCY EPS DISSIPATION TERM OPTIONS` with `BUOYANCY C3EPS COEFFICIENT` advanced parameters `.`
- The constants in these equations are:

$$\begin{aligned} C_\mu &= 0.09 \\ C_{\varepsilon 1} &= 1.44 \\ C_{\varepsilon 2} &= 1.92 \\ \sigma_k &= 1.0 \\ \sigma_\varepsilon &= 1.3 \end{aligned}$$

7.6 RNG k-epsilon model

The RNG k-epsilon model is derived from the application of the Re-Normalization Group (RNG) method to the Navier-Stokes equations, [51]. The RNG k-epsilon model uses the same equation for computing the turbulent viscosity as the standard k-epsilon model. Compared to the standard k-epsilon model, the RNG k-epsilon model has an additional term in the turbulence dissipation rate equation that accounts for the different scales of motion of turbulent flows:

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial(\rho U_j k)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\alpha(\mu + \mu_t)\frac{\partial k}{\partial x_j}\right) + P_k + P_b - \rho\varepsilon$$

$$\frac{\partial(\rho \varepsilon)}{\partial t} + \frac{\partial(\rho U_j \varepsilon)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\alpha(\mu + \mu_t)\frac{\partial \varepsilon}{\partial x_j}\right) + \frac{\varepsilon}{k}(C_{\varepsilon 1} P_k + C_{\varepsilon 1} C_{\varepsilon 3} P_b - C^*_{\varepsilon 2} \rho\varepsilon)$$

where:

- $C^*_{\varepsilon 2}$ is a modified constant defined as follows:
  $$C^*_{\varepsilon 2} = C_{2\varepsilon} + \frac{C_\mu \eta^3 (1 - \eta/\eta_0)}{1 + \beta\eta^3}$$
  where
  - $\eta = Sk/\varepsilon$
  - $S = (2S_{ij}S_{ij})^{1/2}$ is a strain rate magnitude.
- P_b is the production rate of the turbulent kinetic energy due to buoyancy defined as:
  $$P_b = \beta g_i \frac{\mu_t}{\sigma_t} \frac{\partial T}{\partial x_i}$$
  where
  - β is the coefficient of thermal expansion.
  - g_i is the component of the gravity vector g in x_i direction.
  - $\sigma_t = 0.87$ is the turbulent Prandtl number. You can modify it using the `TURBULENT PRANDTL NUMBER` advanced parameter.

You can disable the production term, P_b , using the `BUOYANCY TKE PRODUCTION TERM` advanced parameter.

- By default, $C_{\varepsilon 3} = 0$ means there is no turbulence dissipation due to buoyancy. You can specify the value for the $C_{\varepsilon 3}$ coefficient using the `BUOYANCY EPS DISSIPATION TERM OPTIONS with BUOYANCY C3EPS COEFFICIENT` advanced parameters.
- The constants in these equations are:

$$\begin{aligned} \alpha &= 1.39 \\ C_\mu &= 0.0845 \\ C_{\varepsilon 1} &= 1.42 \\ C_{\varepsilon 2} &= 1.68 \\ \eta_0 &= 4.38 \\ \beta &= 0.012 \end{aligned}$$

7.7 Realizable k-epsilon model

The realizable k-epsilon model uses a new model for the turbulence dissipation rate equation as well as a new realizable eddy viscosity formulation, [52], which oppose to "non-realizable" turbulent flow where the Reynolds stresses can be negative. In this model, the C_μ quantity which was involved in the standard k-epsilon model eddy viscosity formulation is no longer a constant but a variable.

The realizable k-epsilon model uses the following formulation for calculation of the turbulent viscosity:

$$\mu_t = \rho C_\mu \frac{k^2}{\varepsilon}$$

where:

- C_μ is evaluated from the following equation:

$$C_\mu = \frac{1}{4 + A_s \frac{kU^*}{\varepsilon}}$$

$$A_s = \sqrt{6}\cos\left(\frac{1}{3}\cos^{-1}(\sqrt{6}\frac{S_{ij}S_{jk}S_{ki}}{(S_{ij}S_{ij})^{3/2}})\right)$$

- The quantity U^* is defined as follows:

$$U^* = \sqrt{S_{ij}S_{ij} + \Omega_{ij}\Omega_{ij}}$$

where:

- The strain rate S_{ij} and vorticity Ω_{ij} are defined as follows:
  $S_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right)$
  $\Omega_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} - \frac{\partial U_j}{\partial x_i}\right)$

The turbulent kinetic energy, k, and the dissipation rate of turbulent kinetic energy, ε, are computed by solving the following modified conservation equations:

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial(\rho U_j k)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\Gamma_k \frac{\partial k}{\partial x_j}\right) + P_k + P_b - \rho\varepsilon$$

$$\frac{\partial(\rho\varepsilon)}{\partial t} + \frac{\partial(\rho U_j \varepsilon)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\Gamma_\varepsilon \frac{\partial \varepsilon}{\partial x_j}\right) + \rho C_1 S_\varepsilon - \rho C_2 \frac{\varepsilon^2}{k + \sqrt{\frac{\mu}{\rho}\varepsilon}} + C_{\varepsilon 1}\frac{\varepsilon}{k} C_{\varepsilon 3} P_b$$

where:

- C_1 is a modified constant:
  $C_1 = \max\left[0.43, \frac{\eta}{\eta+5}\right]$
  $\eta = Sk/\varepsilon$

where:

- $S = (2S_{ij}S_{ij})^{1/2}$ is a strain rate magnitude.

- P_b is the production rate of the turbulent kinetic energy due to buoyancy defined as:

  $P_b = \beta g_i \frac{\mu_t}{\sigma_t}\frac{\partial T}{\partial x_i}$

  where:

  - β ia the coefficient of thermal expansion.
  - g_i is a component of the gravity vector g in x_i direction.
  - $\sigma_t = 0.87$ is a turbulent Prandtl number. You can modify it using the `TURBULENT PRANDTL NUMBER` advanced parameter.

You can disable the production term, P_b, using the `BUOYANCY TKE PRODUCTION TERM` advanced parameter.

- By default, $C_{\varepsilon 3} = 0$ means there is no turbulence dissipation due to buoyancy. You can specify the value for the $C_{\varepsilon 3}$ coefficient using the `BUOYANCY EPS DISSIPATION TERM OPTIONS with BUOYANCY C3EPS COEFFICIENT` advanced parameters.
- The constants in these equations are:

$$\begin{aligned} C_{\varepsilon 1} &= 1.44 \\ C_2 &= 1.9 \\ \sigma_k &= 1.0 \\ \sigma_\varepsilon &= 1.2 \end{aligned}$$

7.8 k-omega model

With the standard *k*-ω turbulence model, the turbulent viscosity is given by:

$$\mu_t = \rho\frac{k}{\omega}$$

- k is the turbulent kinetic energy.
- ω is the specific dissipation rate of the turbulent kinetic energy.
- ρ is the density of the fluid.

The turbulent kinetic energy, k, and the specific dissipation rate of turbulent kinetic energy, ω, are obtained by solving a conservation equation for each of these two quantities given by:

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial(\rho U_j k)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\Gamma_k \frac{\partial k}{\partial x_j}\right) + P_k - \beta^* \rho k \omega$$

$$\frac{\partial(\rho\omega)}{\partial t} + \frac{\partial(\rho U_j \omega)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\Gamma_\omega \frac{\partial \omega}{\partial x_j}\right) + \alpha \frac{\omega}{k} P_k - \beta \rho \omega^2$$

In these equations:

- Γ_k is the effective diffusion coefficient of k:

$$\Gamma_k = \mu + \mu_t \sigma_k$$

- Γ_ω is the effective diffusion coefficient of ω is:

$$\Gamma_\omega = \mu + \mu_t \sigma_\omega$$

- P_k is the production rate of turbulent kinetic energy:
  $P_k = -\overline{\rho u_i u_j} \frac{\partial U_i}{\partial x_j}$
- The quantity β is defined as $\beta = \beta_0 f_\beta$ where:

$$f_\beta = \frac{1 + 70\chi_\omega}{1 + 80\chi_\omega}, \quad \chi_\omega \equiv \left|\frac{\Omega_{ij}\Omega_{jl}S_{li}}{(\beta_0^* \omega)^3}\right|$$

- The quantity β^* is defined as $\beta^* = \beta_0^* f_{\beta^*}$ where:

$$f_{\beta^*} = \begin{cases} 1, & \chi_k \leq 0 \\ \frac{1+680\chi_k^2}{1+400\chi_k^2}, & \chi_k > 0 \end{cases}, \quad \chi_k \equiv \frac{1}{\omega^3}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}$$

- The strain rate S_{ij} and vorticity Ω_{ij} are defined as follows:

$$S_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right)$$

$$\Omega_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} - \frac{\partial U_j}{\partial x_i}\right)$$

- The constants in these equations are:

$$\begin{aligned} \mathrm{a} &= 0.52 \\ \beta_0 &= 0.072 \\ \beta_0^* &= 0.09 \\ \sigma_k &= 0.5 \\ \sigma_\omega &= 0.5 \end{aligned}$$

7.9 Shear stress transport model

With the *shear stress transport* (SST) turbulence model the turbulent viscosity is:

$$\mu_t = \rho \frac{a_1 k}{\max(a_1 \omega, \Phi F_2)}$$

The flow solver uses the strain rate magnitude $\Phi = S = \sqrt{2 S_{ij} S_{ij}}$

The strain rate S_{ij} and vorticity Ω_{ij} are defined as follows:

$$S_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right)$$
$$\Omega_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} - \frac{\partial U_j}{\partial x_i}\right)$$

The turbulent kinetic energy, k, and the specific dissipation rate of turbulent kinetic energy, ω, are obtained by solving a conservation equation for each of these two quantities given by:

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial(\rho U_j k)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\Gamma_{k3} \frac{\partial k}{\partial x_j}\right) + \tilde{P}_k - \beta^* \rho k \omega$$

$$\frac{\partial(\rho \omega)}{\partial t} + \frac{\partial(\rho U_j \omega)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\Gamma_{\omega 3} \frac{\partial \omega}{\partial x_j}\right) + $$
$$+ (1 - F_1) 2\rho \frac{1}{\sigma_{\omega_2} \omega} \frac{\partial k}{\partial x_i} \frac{\partial \omega}{\partial x_i} + \alpha_3 \frac{\mu_t}{\rho} P_k - \beta_3 \rho \omega^2$$

In these equations:

- F_1 is the blending function, given by:

$$F_1 = \tanh\left(\psi_1^4\right)$$

$$\psi_1 = \min\left(\max\left(\frac{\sqrt{k}}{\beta^* \omega y}, \frac{500\mu}{\rho y^2 \omega}\right), \frac{4\rho k}{CD_{k\omega} \sigma_{\omega_2} y^2}\right)$$

$$CD_{k\omega} = \max\left(2\rho \frac{1}{\sigma_{\omega_2}\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}, 10^{-10}\right)$$

Note

When:

- $F_1 = 0$, the transport equations are equivalent to the *k*-*ε* model.
- $F_1 = 1$, the transport equations are equivalent to the k-ω model.

- F_2 is the second blending function, given by:

$$F_2 = \tanh\left(\psi_2^2\right)$$

with

$$\psi_2 = \max\left(\frac{2\sqrt{k}}{\beta^* \omega y}, \frac{500\mu}{\rho y^2 \omega}\right)$$

A production limiter is used to prevent build-up of turbulence in stagnation regions:

$$\tilde{P}_k = \min\left(P_k,\ 10\rho\beta^* k\omega\right)$$

with

$$P_k = -\rho\overline{u_i u_j}\frac{\partial U_i}{\partial x_j}$$

The following lists the various coefficients that are given as blended constants:

- Γ_{k_3} is the effective diffusion coefficient of k:

$$\Gamma_{k_3} = \mu + \frac{\mu_t}{\sigma_{k_3}}$$

- Γ_{ω_3} is the effective diffusion coefficient of ω:

$$\Gamma_{\omega_3} = \mu + \frac{\mu_t}{\sigma_{k_3}}$$

- The quantity α_3 is defined as:

$$\alpha_3 = \alpha_1 F_1 + \alpha_2 (1 - F_1)$$

- The quantity β_3 is defined as:

$$\beta_3 = \beta_1 F_1 + \beta_2 (1 - F_1)$$

- The quantity σ_{k_3} is defined as:

$$\sigma_{k_3} = \sigma_{k_1} F_1 + \sigma_{k_2} (1 - F_1)$$

- The quantity σ_{ω_3} is defined as:

$$\sigma_{\omega_3} = \sigma_{\omega_1} F_1 + \sigma_{\omega_2} (1 - F_1)$$

- The constants in these equations are:

$$\begin{aligned} a_1 &= 0.31 \\ \beta^* &= 0.09 \\ \alpha_1 &= 5/9 \\ \alpha_2 &= 0.44 \\ \beta_1 &= 0.075 \\ \beta_2 &= 0.0828 \\ \sigma_{k1} &= 1/0.85 \\ \sigma_{k2} &= 1 \\ \sigma_{\omega_1} &= 2 \\ \sigma_{\omega_2} &= 1/0.856 \end{aligned}$$

7.10 Spalart-Allmaras model

The Spalart-Allmaras (SA) model adds one partial differential equation for the modified eddy viscosity, identified by $\tilde{\nu}$. The modified eddy viscosity is related to the turbulent viscosity by an algebraic equation:

$$\mu_t = \rho \tilde{\nu} \frac{\chi^3}{\chi^3 + C_{v1}^3}$$

$$\chi = \frac{\tilde{\nu}}{\nu}$$

The equation above is non-linear. It is solved using Newton`s method.

The modified eddy viscosity is governed by the following transport equation [50]:

$$\frac{\partial \tilde{\nu}}{\partial t} + \frac{\partial(\tilde{\nu} U_j)}{\partial x_j} = \frac{1}{\sigma}\frac{\partial}{\partial x_j}\left[(\nu + \tilde{\nu})\frac{\partial \tilde{\nu}}{\partial x_j}\right] + C_{b1}\,\widehat{S}\,\tilde{\nu} - \left[C_{w1} f_w\right]\left(\frac{\tilde{\nu}}{d}\right)^2 + \frac{C_{b2}}{\sigma}\left(\frac{\partial \tilde{\nu}}{\partial x_j}\right)^2$$

where:

- ν is the molecular kinematic viscosity.
- d is the distance to the nearest solid wall.
- $\widehat{S}$ is the modified vorticity term that maintains log-law behavior to the nearest solid wall defined as follows:

$$\widehat{S} = \Omega + \frac{\tilde{\nu}}{k^2 d^2}\left(1 - \frac{\chi}{1+\chi f_{v1}}\right)$$
$$f_{v1} = \frac{\chi^3}{\chi^3 + C_{v1}^3}$$

- The magnitude of vorticity is computed as follows:

$$\Omega = \sqrt{2\Omega_{ij}\Omega_{ij}}$$
$$\Omega_{ij} = \frac{1}{2}\left(\frac{\partial U_i}{\partial x_j} - \frac{\partial U_j}{\partial x_i}\right)$$

- The non-dimensional function, f_w, which accelerates the decay of the destruction term in the outer region of the boundary layer is given by:

$$f_w = g\left[\frac{1+C_{w3}^6}{g^6+C_{w3}^6}\right]^{\frac{1}{6}}$$
$$g = r + C_{w2}(r^6 - r)$$
$$r = min\left[\frac{\tilde{\nu}}{\widehat{S}\,k^2 d^2}, 10\right]$$

- The constants in these equations are:

$$C_{b1} = 0.1355$$
$$C_{b2} = 0.622$$
$$\sigma = 2/3$$
$$k = 0.41$$
$$C_{w1} = \frac{C_{b1}}{k^2} + \frac{(1 + C_{b2})}{\sigma}$$
$$C_{w2} = 0.3$$
$$C_{w3} = 2.0$$
$$C_{v1} = 7.1$$

7.11 Initializing turbulence models

You can initialize the turbulence model by providing one of the following quantity pairs:

- Turbulent intensity, I_x, and viscosity ratio, μ_t/μ
- Turbulent intensity, I_x, and eddy length, l_t
- Turbulent kinetic energy, k, and dissipation rate, ε
- Turbulent kinetic energy, k, and specific dissipation rate, ω

The flow solver calculates the appropriate turbulence quantities for the selected turbulence model (see table below) and for the provided quantity pair using the following relationships:

$$k = \frac{3}{2}(U_0 I_x)^2$$

$$\varepsilon = \frac{k\sqrt{k}}{l_t} = 0.09\rho\frac{k^2}{\mu}(\frac{\mu_t}{\mu})^{-1} = 0.09\frac{k^2}{\nu}(\frac{\nu_t}{\nu})^{-1}$$

$$\omega = \frac{\sqrt{k}}{0.09 l_t} = \rho\frac{k}{\mu}(\frac{\mu_t}{\mu})^{-1} = \frac{k}{\nu}(\frac{\nu_t}{\nu})^{-1}$$

$$\varepsilon = 0.09k\omega$$

$$\tilde{\nu} = \sqrt{\frac{3}{2}}U_o I_x l_t = \sqrt{k} l_t$$

where:

- U_0 is the mean flow velocity at the boundary.
- μ is the fluid dynamic viscosity in the independent fluid domain.
- μ_t is the turbulent dynamic viscosity.
- ν is the fluid kinematic viscosity in the independent fluid domain.
- ν_t is the turbulent kinematic viscosity.
- $\tilde{\nu}$ is the modified viscosity.

If you initializes the fluid domain variables in such a way that ε or ω equal to zero everywhere, the flow solver uses the mixing length turbulence model to initialize the fluid domain for the first five iterations. This prevents convergence issues with the model. At the sixth iteration, the solver activates the specified turbulence model and uses specified turbulence quantities.

The following table lists the turbulence quantities that each turbulence model solves for using addition equations.

Turbulence model	Turbulence quantities
Standard k-epsilon RNG k-epsilon Realizable k-epsilon	Turbulent kinetic energy, *k* Dissipation rate, *ε*

Turbulence model	Turbulence quantities
k-omega SST (shear stress transport)	Turbulent kinetic energy, *k* Specific dissipation rate, ω
Spalart-Allmaras	Modified viscosity, $\tilde{\nu}$

7.12 Turbulence model tuning coefficients

You can modify the turbulence model constants using the advanced parameters in the *user.prm* file. For example, to set the standard σ_k coefficient to 2.0, add the following line to the *user.prm* file using a simple text editor, which you save in the run directory:

```
SIGMA_K_KEPS_STD = 2.0
```

For more information on the *user.prm* file, see Flow solver files.

The following table lists the advanced parameters you use to modify the coefficients in the Standard k-epsilon model.

Advanced parameter	Default	Description
`SIGMA_K_KEPS_STD`	1.0	Sets the σ_k coefficient.
`SIGMA_EPS_KEPS_STD`	1.3	Sets the σ_ε coefficient.
`C_MU_KEPS_STD`	0.09	Sets the C_μ coefficient.
`C_EPSILON1_KEPS_STD`	1.44	Sets the $C_{\varepsilon 1}$ coefficient.
`C_EPSILON2_KEPS_STD`	1.92	Sets the $C_{\varepsilon 2}$ coefficient.

The following table lists the advanced parameters you use to modify the coefficients in the RNG k-epsilon model.

Advanced parameter	Default	Description
`ALPHA_K_KEPS_RNG`	1.39	Sets the coefficient α for k equation.
`ALPHA_EPS_KEPS_RNG`	1.39	Sets the coefficient α for epsilon equation.
`C_MU_KEPS_RNG`	0.0845	Sets the C_μ coefficient.
`C_EPSILON1_KEPS_RNG`	1.42	Sets the $C_{\varepsilon 1}$ coefficient.
`C_EPSILON2_KEPS_RNG`	1.68	Sets the $C_{\varepsilon 2}$ coefficient.
`ETA0_KEPS_RNG`	4.38	Sets the η_0 coefficient.
`BETA_KEPS_RNG`	0.012	Sets the β coefficient.

The following table lists the advanced parameters you use to modify the coefficients in the Realizable k-epsilon model.

Advanced parameter	Default	Description
`SIGMA_K_KEPS_REAL`	1.0	Sets the σ_k coefficient.
`SIGMA_EPS_KEPS_REAL`	1.2	Sets the σ_ε coefficient.
`C2_KEPS_REAL`	1.9	Sets the C_2 coefficient.
`A0_KEPS_REAL`	4.0	Sets the A_0 coefficient.

The following table lists the advanced parameters you use to modify the coefficients in the k-omega model.

Advanced parameter	Default	Description
`SIGMA_K_KOMG`	0.5	Sets the σ_k coefficient.
`SIGMA_OMG_KOMG`	0.5	Sets the σ_ω coefficient.
`ALPHA_KOMG`	0.52	Sets the α coefficient.
`BETA0_KOMG`	0.072	Sets the β_0 coefficient.
`BETA_STAR0_KOMG`	0.09	Sets the β_0^* coefficient.

The following table lists the advanced parameters you use to modify the coefficients in the Shear stress transport model.

Advanced parameter	Default	Description
`SIGMA_K1_SST`	1.0/0.85	Sets the σ_{k1} coefficient.
`SIGMA_K2_SST`	1.0	Sets the σ_{k2} coefficient.
`SIGMA_OMG1_SST`	2.0	Sets the $\sigma_{\omega 1}$ coefficient.
`SIGMA_OMG2_SST`	1.0/0.856	Sets the $\sigma_{\omega 2}$ coefficient.
`A1_SST`	0.31	Sets the $a1$ coefficient.
`ALPHA1_SST`	5.0/9.0	Sets the $\alpha 1$ coefficient.
`ALPHA2_SST`	0.44	Sets the $\alpha 2$ coefficient.
`BETA1_SST`	0.075	Sets the β_1 coefficient.

Advanced parameter	Default	Description
`BETA2_SST`	0.0828	Sets the β_2 coefficient.
`BETA_STAR_SST`	0.09	Sets the β^* coefficient.

The following table lists the advanced parameters you use to modify the coefficients in the Spalart-Allmaras model.

Advanced parameter	Default	Description
`SIGMA_SA`	2.0/3.0	Sets the σ coefficient.
`CB1_SA`	0.1355	Sets the C_{b1} coefficient.
`CB2_SA`	0.622	Sets the C_{b2} coefficient.
`KAPPA_SA`	0.41	Sets the k coefficient.
`C_OMG2_SA`	0.3	Sets the $C_{\omega 2}$ coefficient.
`C_OMG3_SA`	2.0	Sets the $C_{\omega 3}$ coefficient.
`C_V1_SA`	7.1	Sets the C_{v1} coefficient.

7.13 Turbulent structures

The flow solver identifies turbulent structures using the Q-criteria [39]. This criteria, which is used to detect vortices in the computation domain, is defined with a positive invariant of the velocity gradient tensor $\nabla\vec{U}$:

$$Q \equiv \frac{1}{2}(U_{i,i}^2 - U_{i,j}U_{j,i}) = -\frac{1}{2}U_{i,j}U_{j,i} = \frac{1}{2}(\| \mathbf{\Omega} \|^2 - \| \mathbf{S} \|^2) > \mathbf{0}$$

where:

- $\mathbf{\Omega}$ is a vorticity tensor.

- $\mathbf{S}$ is a strain-rate tensor.

The norm of any tensor is defined as follows:

$$\|\mathbf{G}\| = [\mathrm{tr}(\mathbf{G}\mathbf{G}^{\mathrm{T}})]^{1/2}$$

The vortex exists where the Q-criteria is positive. This is the regions where the vorticity magnitude prevails over the strain-rate magnitude. The pressure in the vortex region is required to be lower than the ambient pressure.

8 Boundary conditions

8.1 Walls

Because the fluid is unable to penetrate a solid wall, the mass flow on the wall is equal to zero.

8.1.1 Wall treatment

The treatment of momentum and energy quantities at the wall depends on the boundary condition type.

Slip wall

A slip wall boundary condition is used to simulate the flow next to a frictionless surface. The wall shear stress and the velocity gradients normal to the wall are zero. The velocity of the fluid relative to the wall is non-zero. For the k-ε turbulence model, the gradients of k and ε normal to the wall and mass flow are set to zero.

Immersed boundary method

When you use the standard k-epsilon turbulence model with immersed boundary meshes, the gradient of the k and ϵ values normal to the slip wall boundary are set to zero. When you use the SST turbulence model with immersed boundary meshes, the gradient of the k and ω values normal to the slip wall boundary are set to zero. In both turbulence models, the specified boundary conditions are applied based on the Neumann boundary condition in the IBM discretization method.

No-slip wall

The velocity of the fluid at a no-slip wall is set equal to the velocity of the wall. For a stationary wall, the fluid velocity at the wall is zero. For translating or rotating walls, it is non-zero. The calculation of the wall shear stress, τ_ω , depends on whether the flow is laminar or turbulent.

For a laminar flow, the viscous stress tensor, τ_{ij} , is given by:

$$\tau_{ij} = \mu\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) - \frac{2}{3}\mu\frac{\partial U_k}{\partial x_k}\delta ij$$

- μ is the dynamic viscosity of the fluid.
- δ_{ij} is the Kronecker delta function defined as:

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

The viscous stress vector applied on the wall boundary is then obtained as:

$$\tau_i = \tau_{ij} n_j$$

- where n_j is the unit vector normal to the wall boundary.

The shear stress vector is calculated by removing the normal component from the viscous stress vector as follow:

$$\tau_w = \tau_i - (\tau_k n_k) n_i$$

Note

The mesh close to the wall must be sufficiently fine to resolve the velocity gradient at the wall.

For turbulent flow, the equations used depend on the turbulent model and whether wall functions are applied or not.

You can impose the wall velocity $U_w = 0$ on the wall boundary nodes using the `WALL MOMENTUM DIRICHLET BC` advanced parameter, which applies the Dirichlet boundary conditions for momentum equations on the walls.

8.1.2 Wall distance calculation methods

Wall distance is used in turbulence models, such as Spalart-Allmaras, K-epsilon, K-omega and SST. The flow solver uses the following methods to compute the wall distance:

- Geometric search method.
- Partial differential equation (PDE) method.

The geometric search method provides more accurate results compared to the PDE method, for fluid nodes that are located far from the wall boundary. However, the geometric method requires more computation time.

Geometric search method

The flow solver uses the geometric search method to compute geometrically the exact distance from each fluid node to their closest no-slip wall boundary node. This method is useful when accurate wall distance is required in the fluid domain. With this method, computation time increases significantly by increasing the number of fluid nodes.

Partial differential equation method

The flow solver computes the wall distance function, ϕ, using the Poisson equation:

$$\nabla^2 \phi = -1$$

The no-slip wall boundary condition for the Poisson equation is:

$$\phi = 0$$

The other types of boundary condition for the Poisson equation are defined as:

$$\frac{\partial \phi}{\partial x_i}.n = 0$$

where:

- n is the normal to the boundary.
- x_i are the Cartesian coordinates.

The wall distance, d, is computed as:

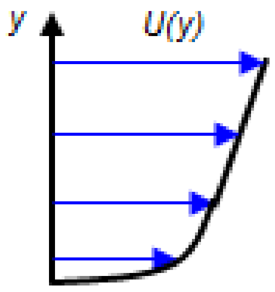
$$d = \sqrt{\sum_{i=1,3}\left(\frac{\partial \phi}{\partial x_i}\right)^2 + 2\phi} - \sqrt{\sum_{i=1,3}\left(\frac{\partial \phi}{\partial x_i}\right)^2}$$

8.1.3 Wall functions

Wall functions describe the flow velocity as a function of distance from the wall within the near-wall region in the turbulent boundary layer. By using wall functions to approximate the mean velocity field in the near-wall region, you avoid mesh requirements needed to resolve the viscous sublayer.

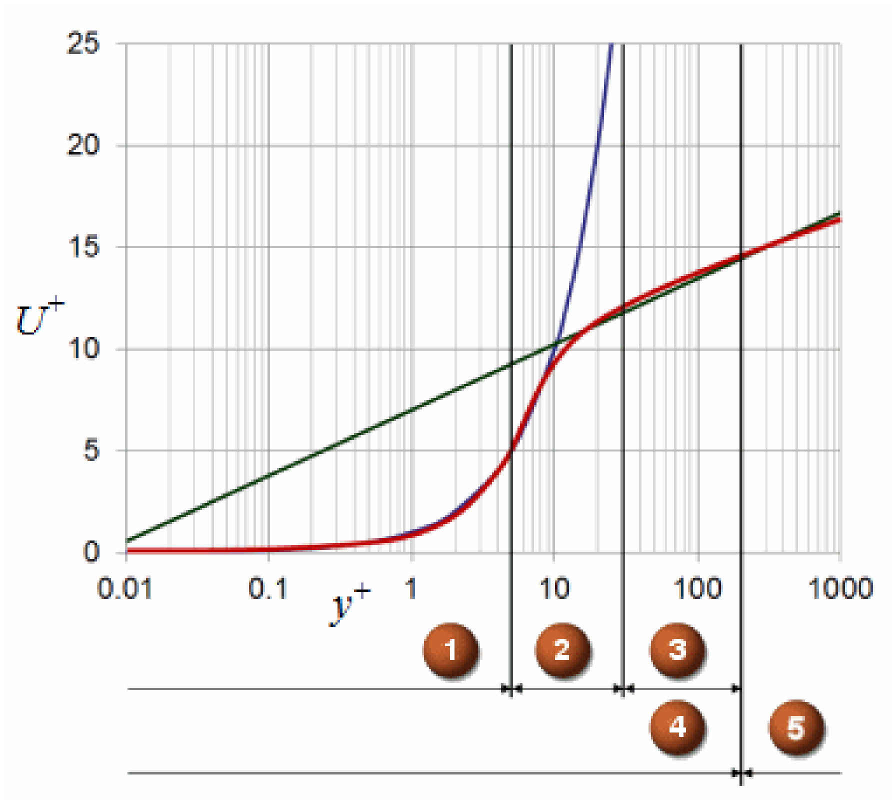
Turbulent boundary layer

The turbulent boundary layer that forms over a flow surface is characterized by the sharp velocity gradient along the direction normal to the wall as shown in the following picture.



Velocity profile near the wall

The turbulent boundary layer is divided into the inner layer (4) and the outer layer (5).



Velocity in the turbulent boundary layer

The inner layer in turn is divided into:

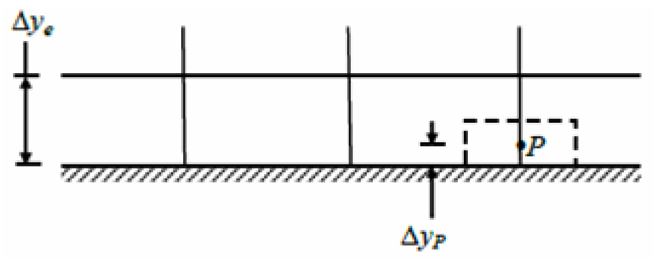
- The viscous sublayer (1): $y^+ < 5$
- The buffer layer (2): $5 < y^+ < 30$
- The log-law region (3): $30 < y^+ < 200$

Boundary layer meshes, which are high aspect ratio grids with very fine spacing along the wall normal direction, are typically used inside boundary layers in order to capture the sharp velocity gradient while maintaining a reasonable overall grid count and computational efficiency. The appropriate mesh size next to walls is determined by the non-dimension distance value y^+ defined as follows:

$$y^+ = \frac{\Delta y \sqrt{\rho \tau_\omega}}{\mu}$$

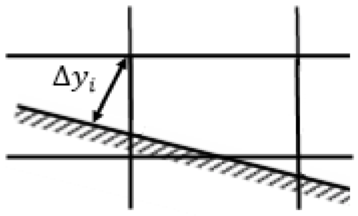
where:

- For body-fitted fluid meshing, $\Delta y = \Delta y_p$ is the distance between the centroid (point P) of the wall-adjacent control volume and the wall. You can approximate Δy_p from the distance of the first node and the wall, Δy_e , as $\Delta y_e/4 \leq \Delta y_p \leq \Delta y_e/3$. In the following graphics, the centroid is represented by the point P.



For immersed boundary meshing, $\Delta y = \Delta y_{e,avg}$ is the average value of the normal wall distances between the boundary element's N nodes and the closest immersed wall boundary.

$$\Delta y_{e,avg} = \frac{\sum_{i=1}^{N} \Delta y_i}{N}$$



- ρ and μ are density and dynamic viscosity of the fluid.
- τ_ω is the wall shear stress.

In the viscous sublayer, the mean velocity represented by the red curve, is given by the following equation represented by the blue curve:

$$U^+ = y^+$$

- where U^+ is the dimensionless velocity computed as:

$$U^+ = U\sqrt{\frac{\rho}{\tau_\omega}}$$

In the log-law region, the mean velocity is given by the log-law equation represented by the green curve:

$$U^+ = \frac{\ln y^+}{\kappa} + C^+$$

- where C^+ is an experimental constant.

Standard wall function

The standard wall function is represented by the red curve in the graphic: Velocity in the turbulent boundary layer.

The flow solver uses general form of exponential blending function between the linear relation in the viscous sublayer and the classical log-law from the log-law region as follows:

$$U^+(y^+) = y^+ e^{-\Gamma} + \left(\tfrac{1}{\kappa}\ln(y^+) + 5.2\right)e^{-\frac{1}{\Gamma}}$$
$$\Gamma = \tfrac{0.01(y^+)^4}{1+5y^+}$$

When you request the standard wall function, you should place the first computational nodes above the wall in the log-law region. If the node is located in the viscous sublayer, the results are inaccurate. You can use wall function only for streamlined geometries without flow separation.

Hybrid wall function for K-Omega and SST models

Standard wall function performance deteriorates when the first layer grid is in the buffer layer or viscous sublayer. Hybrid wall function ensures a locally appropriate near-wall resolution adapted to the flow structures and it is robust even when $y^+ < 30$. Hybrid wall function is also more robust, when the flow approaches the separation point.

The hybrid function blends the Reichardt's law $F_{Rei_{mix}}$ and Spaldings law $F_{Sp,3}$ as follows [36]:

$$U^+ = (1 - \phi_{k\omega})F_{Sp,3} + \phi_{k\omega}F_{Rei_{mix}}$$

- $\phi_{k\omega} = \tanh\left(\left(\frac{y^+}{50}\right)^2\right)$

Reichardt's law of the wall is defined as follows:

$$F_{Rei}(y^+) = \frac{\ln(1 + 0.4y^+)}{\kappa} + 7.8\left(1 - e^{-\frac{y^+}{11}} - \frac{y^+}{11}e^{-\frac{y^+}{3}}\right)$$

The following blending is applied with classical log-law:

$$F_{Rei_{mix}} = (1 - \phi_{b1})F_{Rei} + \phi_{b1}(\ln(y^+)/\kappa + 5.1)$$

- $\phi_{b1} = \tanh\left(\left(\frac{y^+}{27}\right)^4\right)$

Spaldings law is given by the following inverse formula:

$$y^+ = F_{Sp,3}^{-1} = u^+ + e^{-5.2\kappa}\left(e^{\kappa u^+} - \sum_{n=0}^{3}\frac{(\kappa u^+)^n}{n!}\right)$$

Hybrid wall function for Spalart-Allmaras model

The hybrid wall function blends Reichardt's law $F_{Rei_{mix}}$ and Spaldings law $F_{Sp,5}$ as follows [35]:

$$U^+ = (1 - \phi_{SA})F_{Sp,5} + \phi_{SA}F_{Rei_{mix}}$$

where:

- $\phi_{SA} = \tanh\left(\left(\frac{y^+}{24}\right)^3\right)$

Reichardt's law of the wall is defined as follows:

$$F_{Rei}(y^+) = \frac{\ln(1 + 0.4y^+)}{\kappa} + 7.8\left(1 - e^{-\frac{y^+}{11}} - \frac{y^+}{11}e^{-\frac{y^+}{3}}\right)$$

The following blending is applied with classical log-law:

$$F_{Rei_{mix}} = (1 - \phi_{b1})F_{Rei} + \phi_{b1}(\ln(y^+)/\kappa + 5.1)$$

where:

- $\phi_{b1} = \tanh\left(\left(\frac{y^+}{27}\right)^4\right)$

Spaldings law is given by the following inverse formula:

$$y^+ = F_{Sp,5}^{-1} = u^+ + e^{-5.2\kappa}\left(e^{\kappa u^+} - \sum_{n=0}^{5}\frac{(\kappa u^+)^n}{n!}\right)$$

8.1.4 Wall functions with roughness effect

In addition to the linear relation in the viscous sublayer, the wall function can include the roughness effect as follows:

$$U^+ = \frac{1}{\kappa}\ln\left(\frac{y^+}{1+0.3k_s^+}\right) + 5.2$$

- $k_s^+ = k_s\frac{\rho}{\mu}u^*$
- k_s is the specified equivalent sand grain roughness.
- u^* is the shear velocity. For more information, see Mixing length turbulence model.

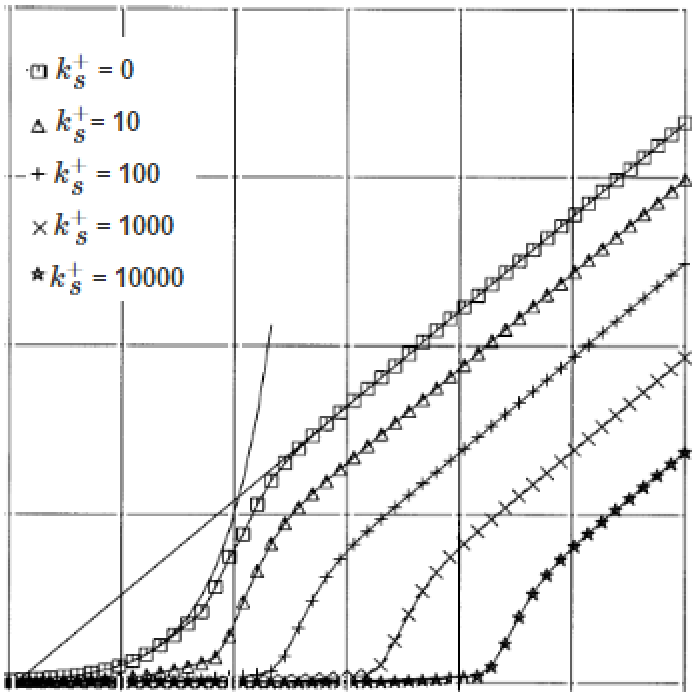
The roughness effect shifts the log-law, but does not affect the slope.

The equation for the standard wall function with roughness effect is as follow:

$$U^+ = \frac{y^+}{1+0.3k_s^+}e^{-\Gamma} + \left(\frac{1}{\kappa}\ln\left(\frac{y^+}{1+0.3k_s^+}\right) + 5.2\right)e^{-\frac{1}{\Gamma}}$$

- $\Gamma = \frac{0.01(z^+)^4}{1+5z^+}$
- $z^+ = \frac{y^+}{1+0.03k_s^+}$

Plots of wall function dimensionless velocity, U^+ for varying dimensionless roughness is shown in the following figure. For reference, the smooth wall log-law and the linear function are also plotted as solid lines.



Wall function for varying roughness heights

The equation for the hybrid wall function with roughness effect is as follow:

$$U^+ = (1 - \phi_{k\omega})F_{Sp,3} + \phi_{k\omega} F_{Rei_{mix}}$$

- $\phi_{k\omega} = \tanh((C_1 y^+)^2)$
- C_1 is an empirical constant that depends on the dimensionless roughness height k_s^+.

Reichardt's law that takes into account the roughness effect:

$$F_{Rei}(y^+) = \frac{\ln(1 + \frac{0.4y^+}{1+0.3k_s^+})}{\kappa} + 7.8\left(1 - e^{-\frac{y^+}{11(1+0.3k_s^+)}} - \frac{y^+}{11(1 + 0.3k_s^+)}e^{-\frac{y^+}{3(1+0.3k_s^+)}}\right)$$

$F_{Rei_{mix}}$ blends Reichardt's law with classical log-law as follows:

$$F_{Rei_{mix}} = (1 - \phi_{b1})F_{Rei} + \phi_{b1}\left(\ln\left(\frac{y^+}{1 + 0.3k_s^+}\right)/\kappa + 5.1\right)$$

- $\phi_{b1} = \tanh((C_2 y^+)^4)$
- C_2 is an empirical constant that depends on the dimensionless roughness height k_s^+.

Spaldings law is given by the following inverse formula:

$$F_{Sp,3}^{-1} = U^{+} + e^{-5.2\kappa}\left(e^{\kappa u^{+}} - \sum_{n=0}^{3}\frac{(\kappa U^{+})^{n}}{n!}\right)$$

8.1.5 Wall shear stress for turbulence models

The calculation of the wall shear stress in turbulent flows depends on the applied wall treatment method. The application of wall functions and their types affects the shear stress calculation.

Shear stress on a resolved wall

The wall viscous stress tensor on a resolved wall in turbulent flows is defined as follows:

$$\tau_{ij} = (\mu + \mu_t)\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) - \frac{2}{3}(\mu + \mu_t)\frac{\partial U_k}{\partial x_k}\delta_{ij} - \frac{2}{3}\rho k \delta_{ij},$$

where:

- μ is the specified dynamic viscosity of the fluid.
- μ_t is the turbulent viscosity calculated from the specified turbulence model.
- k is the turbulent kinetic energy.
- δ_{ij} is the Kronecker delta function defined as:

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

The viscous stress vector applied on the wall boundary is then obtained as:

$$\tau_i = \tau_{ij} n_j$$

where n_j is the unit vector normal to the wall boundary.

The shear stress vector is calculated by removing the normal component from the viscous stress vector as follow:

$$\tau_w = \tau_i - (\tau_k n_k)\, n_i$$

Shear stress with applied wall functions

When wall functions are applied, the flow solver computes the magnitude of the wall shear stress $\tau_w = |\tau_{w,\,i}|$ from the wall function formulation. It is assumed that the wall shear stress vector is aligned with the tangential component of the velocity at the centroid of the wall-adjacent control volume. The tangential component of the velocity is given by:

$$U_{t,\,i} = \bar{U}_i - \left(\bar{U}_k n_k\right) n_i$$

where $\bar{u}_i$ is the vector of the relative velocity at the centroid of the wall-adjacent control volume. The flow solver computes the relative velocity with respect to the wall.

The wall shear stress vector is then calculated as follows:

$$\tau_{w,\,i} = \tau_w \frac{U_{t,\,i}}{|U_{t,\,i}|}.$$

The calculation of τ_w depends on the type of the wall function: standard wall function or hybrid wall function.

Wall functions provide the relation between $U^+ = |U_{t,\,i}|^+$ and y^+ at centroids of wall-adjacent control volumes. Since both U and y at the centroid are available from the solution of the conservation equations and the model geometry, respectively, the only unknown left is the wall shear velocity, U^* . The shear velocity, which is directly related to the magnitude of the wall shear stress vector, τ_w , via $U^* = \sqrt{\tau_w/\rho}$, is used to convert U and y to corresponding dimensionless forms of U^+ and y^+ .

Due to the complexity of wall function relations, an iterative method is required to solve for the wall shear stress.

Standard wall functions

When you use the standard wall function, instead of an iterative method, the flow solver uses the turbulent kinetic energy, k , which is computed from the solution of turbulence conservation equations, to calculate the wall shear velocity as follows:

$$U^* = C_\mu^{\frac{1}{4}} k^{\frac{1}{2}}$$

The shear velocity is used to convert the wall distance, y , to dimensionless form y^+ . U^+ is then computed by plugging y^+ into the wall function relation and converted to U using the shear velocity. The wall shear stress is calculated from the following relation:

$$\tau_w = \mu \frac{U}{y} \frac{y^+}{U^+}$$

The calculation of the shear velocity from the turbulent kinetic energy makes this procedure more sensitive to y^+ values and less accurate when $y^+ \ll 1$ in comparison to the iterative procedure of hybrid wall functions. At separation points, the wall shear stress and y^+ diminish. The shear velocity, according to the definition, should follow the shear stress and vanish. However, the turbulent kinetic energy coming from the solution of turbulence conservation equations maintains non-zero values at separation locations above the wall. Any non-zero turbulent kinetic energy leads to a finite shear velocity when standard wall function are used. The flow solver reverts back to an iterative procedure to calculate the wall shear stress when the employed turbulence model involves no equation for the turbulent kinetic energy. This is the case for the mixing length turbulence model.

Hybrid wall functions

When hybrid wall functions are employed, the iterative method described by Knopp et al.[36] is used in the flow solver to calculate the magnitude of the wall shear velocity and the wall shear stress.

8.1.6 Wall boundary conditions for turbulence models

Mixing length turbulence model

The flow solver calculates the near wall turbulent viscosity using Eddy viscosity equation with the appropriate mixing length l and the strain rate S. This model always uses the standard wall function to approximate the mean velocity field in the near-wall region of the boundary layer.

Standard k-epsilon, RNG k-epsilon, Realizable k-epsilon turbulence models

The flow solver calculates the near wall turbulent viscosity as

$$\mu_t = \rho l_t^2 \frac{C_\mu^{1/4} k^{1/2}}{\kappa y}$$

- $l_t = f_l \kappa y$ is the turbulence length scale.
- f_l is the damping factor. For more information, see Mixing length turbulence model.

In the log-law region, the near-wall relation is given by:

$$U^+ = \frac{U_f}{u^*} = \frac{1}{\kappa}\ln(y^+)$$

- U_f is the fluid velocity at distance y from the wall.
- κ is the Von Karman constant.
- $y^+ = \frac{\rho u^* y}{\mu}$ is a dimensionless wall distance for a wall bounded flow:
- $u^* = \sqrt{\frac{\tau_\omega}{\rho}}$ is the shear velocity.

Using the previous equations, the wall shear stress, τ_ω , is given by:

$$\tau_\omega = \frac{\rho U_f^2}{(\frac{1}{\kappa}\ln(y^+))^2}$$

The near wall transport equation for Standard k-epsilon model is replaced by:

$$\varepsilon = \frac{\rho C_\mu k^2}{\mu} \frac{1}{\kappa y^+ f_\varepsilon}$$

where f_ε is a near wall damping function [17] given by:

$$f_\varepsilon = 1 - \exp\left(- \frac{y^+}{3.8 C_\mu^{1/4}} \right)$$

The near wall production of turbulent kinetic energy is evaluated from:

$$P_k = \frac{\tau_\omega^2}{\mu} \frac{1}{\kappa y^+}$$

The *k*-ε models always uses the standard wall function to approximate the mean velocity field in the near-wall region of the boundary layer.

Immersed boundary method

When you use the standard *k*-ε turbulence model with immersed boundary meshes, the following equations are used based on the Dirichlet boundary condition in the IBM discretization method, to compute the k and ϵ values, respectively:

$$k = \frac{u^{*2}}{\sqrt{C_\mu}}$$

$$\epsilon = \frac{u^{*3}}{f_\epsilon \kappa \overline{y_n}}$$

where:

- $\overline{y_n}$ is the node sector average of the wall distance value, assessed at the halo node, where the flow solver applies the imposed boundary condition.
- f_ϵ is the near wall damping function.

The near wall damping function is:

$$f_\epsilon = max\left(0.005, 1 - exp\left(-\frac{\rho u^* \overline{y_n}}{\mu A_\epsilon C_\mu^{0.25}} \right)\right)$$

where $A_\epsilon = 3.8$ is the damping function constant.

The near wall production of turbulent kinetic energy is computed as:

$$P_{k,wall} = \frac{\rho u^{*3}}{f_\epsilon \kappa \overline{y_n}}$$

The near wall turbulent viscosity is computed as:

$$\mu_{t,wall} = f_\mu \rho \kappa u^* \overline{y_n}$$

where f_μ is the damping function and computed as:

$$f_\mu = max\left(0.005, 1 - exp\left(-\frac{\rho u^* \overline{y_n}}{\mu A_\mu C_\mu^{0.25}}\right)\right)$$

where $A_\mu = 63$ is the damping function constant.

K-omega turbulence model

The flow solver uses the following values for the wall resolved k-ω turbulence model boundary conditions on a smooth wall:

$$k = 0$$
$$\omega = \frac{60\mu}{\rho \beta_1 y^2}$$

The boundary conditions for the wall resolved k-ω turbulence model on a rough wall are given by:

$$k = 0$$
$$\omega = \frac{\rho S_R u^{*2}}{\mu}$$
$$S_R = \begin{cases} \frac{2500}{k_s^{+2}} & \text{if} \quad k_s^+ < 25 \\ \frac{100}{k_s^+} & \text{if} \quad k_s^+ \geq 25 \end{cases},$$

where k_s^+ is the equivalent sand grain roughness height in wall units and is calculated as follow:

$$k_s^+ = \frac{\rho k_s u^*}{\mu}$$

Standard wall function

For k-ω turbulence model that uses the standard wall function, the flow solver calculates the boundary conditions similarly as to the k-ε turbulence model, with exception that instead of ε value, it uses ω value:

$$\omega = \frac{\rho k}{\mu} \frac{1}{\kappa y^+ f_\omega}$$

where f_ω:

$$f_\omega = 1 - \exp\left(- \frac{y^+}{3.8C_\mu^{1/4}} \right)$$

For the k equation, the convective and diffusive fluxes on the wall boundary are set to zero.

Hybrid wall function on smooth walls

The turbulent kinetic energy is zero on the wall.

$$k = 0$$

The near-wall solution for ω is defined as described in [36] and can be written in the wall units as follow:

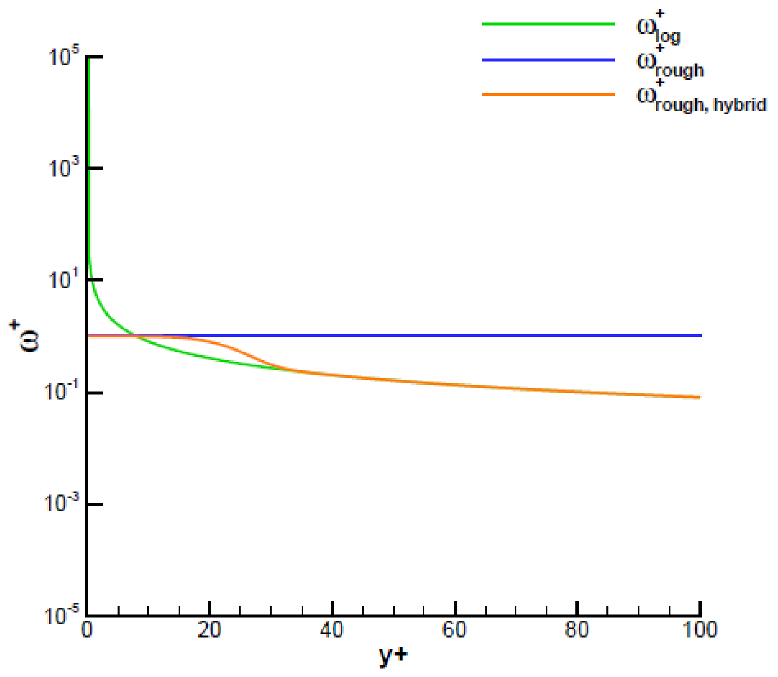
$$\begin{aligned}
\omega^+_{\text{vis}} &= \frac{6\nu}{\beta_\omega y^{+2}} \\
\omega^+_{\text{log}} &= \frac{u^*}{\sqrt{\beta_k}\kappa y^+} \\
\omega^+_{\text{b1}} &= \omega^+_{\text{vis}} + \omega^+_{\text{log}} \\
\omega^+_{\text{b2}} &= \left(\omega^{+1.2}_{\text{vis}} + \omega^{+1.2}_{\text{log}}\right)^{\frac{1}{1.2}} \\
\omega^+_{\text{BC}} &= \phi\omega^+_{b1} + (1-\phi)\omega^+_{b2}
\end{aligned}$$

where

$$\phi = \tanh\left(\frac{y^+}{10}\right)^4$$

Hybrid wall function on rough walls

Turbulent kinetic energy is set to zero on rough walls. The boundary condition for ω is a blend of the resolved rough wall value ω^+_{rough} and the log layer value ω^+_{log} . Unlike the smooth resolved boundary value, ω^+_{vis} , the rough wall boundary value, ω^+_{rough} , is independent of y^+ .



Comparison between different boundary ω^+ values: blended $\omega^+_{rough,hybrid}$, rough wall implementation ω^+_{rough} and the log layer based value ω^+_{log} for $k^+_s = 100$

Shear Stress Transport (SST) model

On resolved smooth walls, the flow solver uses the same boundary condition as those for the *k*-ω turbulence model.

The boundary conditions for the wall resolved SST turbulence model on a rough wall are defined as follows:

$$k^+ = \frac{1}{\sqrt{\beta^*}} \tanh\left[\left(\frac{\ln \frac{k_s^+}{30}}{\ln 10} + 1 - \tanh \frac{k_s^+}{125} \right) \tanh \frac{k_s^+}{125} \right]$$

$$\omega^+ = \frac{\frac{300}{k_s^{+2}}}{\tanh \frac{15}{4k_s^+}} + \frac{191}{k_s^+} \left(1 - \exp(-\frac{k_s^+}{250}) \right)$$

where k^+ and ω^+ are the turbulence kinetic energy and turbulent specific dissipation in wall units [37]:

$$k^+ = \frac{k}{u^{*\,2}}$$
$$\omega^+ = \frac{\mu\omega}{\rho u^{*\,2}}$$

k^+ should satisfy the condition $k^+ = \max(k^+, 0)$.

Standard wall function

The flow solver uses the same boundary conditions for SST model as those used for the k-ω turbulence model in conjunction with the standard wall functions.

Hybrid wall function on smooth walls

Turbulent kinetic energy is set to zero on rough walls. The same blended formula used for the k-ω turbulence model is applied to calculate the boundary value of ω on the wall surface.

Hybrid wall function on rough walls

Turbulent kinetic energy is non-zero and equal to the value used on a resolved rough wall [37]. The value of ω^+ is calculated by blending the resolved rough wall value [37] and ω^+_{log}.

Immersed boundary method

When you use the SST turbulence model with immersed boundary meshes, the following equations are used based on the Dirichlet boundary condition in the IBM discretization method, to compute the k and ω values, respectively:

$$k = \frac{u^{*\,2}}{\sqrt{\beta^*}}$$

$$\omega = \frac{u^*}{f_\omega \sqrt{\beta^*} \kappa \overline{y_n}}$$

where:

- $\beta^* = 0.09$ is the turbulence model constant.
- f_ω is the near wall damping function.

The near wall damping function is:

$$f_\omega = max\left(0.005, 1 - exp\left(-\frac{\rho u^* \overline{y_n}}{\mu A_\epsilon C_\mu^{0.25}}\right)\right)$$

where $A_\epsilon = 3.8$ is the damping function constant.

The near wall production of turbulent kinetic energy is computed as:

$$P_{k,wall} = \frac{\rho u^{*3}}{f_\omega \kappa \overline{y_n}}$$

The near wall turbulent viscosity is computed as:

$$\mu_{t,wall} = f_\mu \rho \kappa u^* \overline{y_n}$$

where f_μ is the damping function and computed as:

$$f_\mu = max\left(0.005, 1 - exp\left(-\frac{\rho u^* \overline{y_n}}{\mu A_\mu C_\mu^{0.25}}\right)\right)$$

where $A_\mu = 63$ is the damping function constant.

Spalart-Allmaras model

For all types of wall treatments, the flow solver uses the following boundary condition for modified viscosity at the wall:

$$\tilde{\nu}_{wall} = 0$$

Smooth walls

The flow solver calculates the near wall vorticity magnitude, Ω, in the same way as for the rest of the fluid domain using the equation that is described in Spalart-Allmaras model.

Standard and hybrid wall functions

The flow solver calculates the near wall vorticity magnitude, Ω, as follows:

$$\Omega_{wall} \approx \frac{\tau_{wall}}{\mu_{eff}}$$

where:

- τ_{wall} is the wall shear stress. For more information, see Turbulent boundary layer.
- μ_{eff} is the effective viscosity, $\mu_{eff} = \mu + \mu_t$, where μ is the dynamic viscosity and μ_t is the turbulent viscosity.

8.1.7 Wall function for the thermal boundary layer

For the energy equation, a solid wall can either be specified as:

- Adiabatic, which means no heat transfer allowed across the wall. In this case, a heat flux of zero is imposed at the wall.
- Convecting, where the wall temperature, the heat flux, or the heat load is specified at the wall.

For laminar flows, the heat flux between the wall and the fluid, q_w, is related to the wall and fluid temperatures through:

$$q_\omega = -k\frac{dT}{dy}\bigg|_\omega \approx -k\frac{(T_\omega - T_f)}{y_f}$$

- k is the thermal conductivity of the fluid.
- y_f is the distance from the wall at which T_f is evaluated.

For turbulent flows, the temperature variation in the thermal boundary layer generally follows the general wall function. For more information, see the Wall functions.

A dimensionless temperature, T^+, is defined in terms of fluid properties, wall heat flux, and the near-wall velocity scale u^*:

$$T^+ = \frac{\rho C_p u^*(T_\omega - T_f)}{q_\omega}$$

The following general equation describes the universal boundary layer profile for temperature:

$$T^{+} = (\mathrm{Pr}\, y^{+})e^{-\Gamma} + \left(\frac{1}{\kappa}\ln(\mathrm{Pr}\, y^{+}) + C\right)e^{-\Gamma}$$

- $\Gamma = \frac{0.01(\mathrm{Pr}\, y_f^{+})^4}{1+5\,\mathrm{Pr}^3\, y^{+}}$
- $C = (3.85 Pr^{1/3} - 1.3)^2$
- Pr is the Prandtl number.

This definition of the thermal wall functions, proposed by B.A. Kader [9], is accurate right to the wall, and applies for a wide range of Prandtl numbers. Notice that there is no explicit reference to roughness and it is assumed that roughness effects are captured implicitly by the wall when y^{+} is defined through u^{*}. Roughness τ_{ω} and near wall k will increase so that u^{*} will be increased suitably.

The equation for q_{ω} is re-arranged as follows:

$$q_{\omega} = \frac{\rho C_p u^{*}(T_{\omega} - T_f)}{T^{+}}$$

Given a near wall fluid temperature, T_f, this equation can be used to compute q_{ω} which is substituted directly into the finite volume energy equation at the wall.

For consistency with the momentum treatment, the equation for q_{ω} is defined as follows:

$$q_{\omega} = \frac{\lambda}{y}(T_{\omega} - T_f)\left(\frac{y^{+}}{T^{+}}\mathrm{Pr}\right) = h(T_{\omega} - T_f)$$

The effect of the wall function is to amplify the laminar (i.e. molecular) thermal diffusion to the wall. The amplification term, $(y^{+}/T^{+})Pr$, tends to 0 as y^{+} tends to 0. Because the dependent variable for the energy equation is not temperature but enthalpy, h. The equation for q_{ω} is modified so that an approximate dependency of q_{ω} on h is retained, while not changing the converged answer for q_{ω}. So, denoting the "new" solution with superscript "n" and the "old" solution by "o", Equation for q_{ω} is written as:

$$q_{\omega}^{n} = -\frac{\mu}{y}(h^{n} - h^{0})\frac{y^{+}}{T^{+}} + \frac{\lambda}{y}\left(T_{\omega} - T_f^{0}\right)\left(\frac{y^{+}}{T^{+}}\mathrm{Pr}\right)$$

On convergence, the wall heat flux does not depend on enthalpy.

8.1.8 Thermal wall function for high speed flows and flows with viscous heating

The flow solver computes the wall heat flux as follows:

$$q_\omega = h(T_\omega - T_f)$$

- T_w is the wall temperature.
- T_f is the fluid temperature at the near wall node.
- h is the heat transfer coefficient.

For flows with significant viscous heating (such as high speed flows), the heat flux is not dependent on the difference between T_w and T_f but rather between T_w and an adiabatic wall temperature. This adiabatic wall temperature depends on the local flow conditions and can be derived from the thermal wall function. Denoting LS as low speed and HS as high speed, the general thermal wall function can be written as:

$$T^+ = T^+_{LS} + T^+_{HS}$$

where

$$T^+_{LS} = (\Pr y^+_f)e^{-\Gamma} + \left(\frac{1}{\kappa}\ln(\Pr y^+_f) + C\right)e^{-\frac{1}{\Gamma}}$$

$$T^+_{HS} = \frac{u^*}{q_\omega}\frac{1}{2}\rho\left[(\Pr U^2_f)e^{-\Gamma} + (\Pr U^2_f + (\Pr - Pr_t)U^2_C)e^{-\frac{1}{\Gamma}}\right]$$

$$T^+ = \frac{\rho C_p u^*}{q_\omega}(T_\omega - T_f)$$

$$y^+_f = \frac{\rho y_f u^*}{\mu}$$

$$\Gamma = \frac{0.01(\Pr y^+_f)^4}{1 + 5\Pr^3 y^+}$$

In the above equations:

- T^+_{LS} is the standard thermal wall function for low speed flows.
- T^+_{HS} is the correction to the standard thermal wall function which accounts for the contribution from the viscous heating on the temperature profile.
- $\Pr$ is the laminar Prandtl number.
- $\Pr_t$ is the turbulent Prandtl number.
- y_f is the normal distance from the wall to the near wall node.
- u^* is the shear velocity.

- q_ω is the wall heat flux.
- ρ is the fluid density.
- μ is the fluid dynamic viscosity.
- U_f is the fluid velocity at the near wall node.
- U_C is the fluid velocity at the intersection of the linear and logarithmic temperature profiles (for low speed flow).

Combining above equations gives:

$$T^+ = \frac{\rho C_p u^*}{q_\omega}(T_\omega - T_f) = T^+_{LS} + \frac{u^*}{q_\omega}T^*_{HS}$$

which allows to express the wall heat flux as:

$$q_\omega = \frac{\rho C_p u^*}{T^+_{LS}}(T_\omega - T_f) - u^* \frac{T^*_{HS}}{T^+_{LS}}$$

The heat flux q_ω is positive when heat flows from the solid. In these equations the terms:

- $\frac{\rho C_p u^*}{T^+_{LS}}(T_\omega - T_f)$ represents the standard low speed flow wall heat flux.
- $\frac{u^* T^*{}_{HS}}{T^+_{HS}}$ is the contribution to the wall heat flux due to viscous heating.

Equation for q_ω can be rewritten as:

$$q_\omega = \frac{\rho C_p u^*}{T^+_{LS}}(T_\omega - T^*_f)$$

where:

$$T^*_f = T_f + \frac{T^*_{HS}}{\rho C_p}$$

In the literature for high speed flows, T^*_f is defined as the recovery temperature or the adiabatic wall temperature and is expressed as:

$$T^*_f = T_f + r\frac{U_f^2}{2C_p}$$

where r is the recovery factor.

If y_f is located in the laminar sublayer (small y_f^+ value), the equation for T_f^* is defined as:

$$T_f^* = T_f + \Pr \frac{U_f^2}{2C_p}$$

which gives the standard recovery factor for the Couette flow equal to $\Pr$.

For the thermal wall function for high speed flow, the flow solver uses the standard formulation for the heat transfer coefficient given by:

$$h = \frac{\rho C_p u^*}{T_{LS}^+}$$

The effects of viscous heating are then accounted for by correcting the fluid temperature so that T_f^* replaces T_f when the heat flux is computed.

8.1.9 Thermal wall function for natural convection

For natural convection in turbulent flows, the flow solver uses a new h correlation derived from a general temperature wall function for natural convection. The flow solver computes the temperature wall function [23], which is valid for turbulent natural convection for any Prandtl number, and for any surface inclination. A complete treatment of turbulent natural convection boundary layers includes the implementation of a natural convection velocity wall function. Implementing such a wall function however, has implications in the treatment of the production and the dissipation of the turbulence kinetic energy in the *k*-ε turbulence model.

The flow solver uses the thermal wall function, which takes into account the effect of Prandtl number as well as the influence of the surface angle with respect the gravity vector, in general form:

$$T^* = \begin{cases} y^*, & \text{for} \quad y^* < 1 \\ a_t + b_t \ln y^* + C_t \ln^2 y^*, & \text{for} \quad 1 < y^* < 100 \\ \left(\Pr/C_t^3\right)^{1/4} y^*, & \text{for} \quad y^* > 100 \end{cases}$$

The coefficients in the thermal wall function are:

$$\alpha_T = 1\,; \quad b_T = 0.4\left(\left(\frac{Pr}{C_t^3}^{1/4}\right) - 1\right); \quad c_T = -0.0397\left(\left(\frac{Pr}{C_t^3}^{1/4}\right) - 1\right)$$

C_t is dependent on the $\Pr$ number as well as on the angle of surface with respect to the gravity vector. Raithby et al [24] proposed the following general formulation:

$$C_t = [C_u(\cos^{1/3}(\phi), 0)_{\max}, \quad C_v \sin^{1/3}(\phi)]_{\max}$$

with the following definitions for the angle ϕ:

For $T_w > T_0$

- $\phi = 0^\circ$ if the surface is horizontal, facing upward
- $\phi = 180^\circ$ if the surface is horizontal, facing downward

for $T_w < T_0$

- $\phi = 0^\circ$ if the surface is horizontal, facing downward
- $\phi = 180^\circ$ if the surface is horizontal, facing upward

The correlations of Raithby [24] for C_u is adopted with a small modification to make it compatible with the value of $C_u = 0.15$ for air. The C_u correlation is:

$$C_u = \frac{0.15(1 + 0.0107\mathrm{Pr})}{(1 + 0.01\mathrm{Pr})}$$

A new correlation for C_v as a function of Pr is derived so that the Nusselt relationship $Nu = C_v Ra^{1/3}$ matches as close as possible:

$$C_v = \frac{0.16}{\left(1 + \left(\frac{0.426}{\mathrm{Pr}}\right)^{9/16}\right)^{16/27}}$$

8.1.10 Local heat transfer coefficient correlation for thermal wall function

The local heat transfer coefficient in the flow solver is defined as:

$$h_l = \frac{q_\omega}{\mathrm{T}_\omega - \mathrm{T}_f}$$

- q_ω is the wall heat flux.
- T_ω is the wall temperature.
- T_f is the local fluid temperature at a normal distance from the wall y_f.

The temperature scale based on wall heat flux is:

$$T_q = \frac{(T_\omega - T_f)}{T_f^*} = \left(\frac{q_\omega^3}{g\beta\alpha(\rho C_p)^3} \right)^{1/4}$$

- α is the thermal diffusivity.
- β is the coefficient of thermal expansion.
- ρ is the fluid density.
- C_p is the specific heat at constant pressure.

Combining equations, the local heat transfer coefficient is expressed as:

$$h_l = pC_p(g\beta\alpha)^{1/3} \frac{T_q^{1/3}}{T_f^*}$$

For laminar flow, $y^* < 1$, the local heat transfer coefficient becomes:

$$h_l = pC_p(g\beta\alpha)^{1/3} \frac{T_q^{1/3}}{y_f^*} = \frac{k}{yf}$$

The local heat transfer coefficient is computed based on the above equation and the natural convection thermal wall function using information on the local wall temperature and heat flux on the surface of a wall element, as well as on the surface orientation angle.

8.1.11 Special considerations when calculating local heat transfer coefficient

There are uncertainties regarding the validity of the correlations found in the literature when the orientation of the solid surface is such that $\phi > 150^\circ$. In addition, as the angle ϕ approaches 180° (heated surface facing downward or cooled surface facing upward), laminar flow prevails even for very high Ra numbers. In the flow solver, the methodology adopted for $150^\circ < \phi < 180^\circ$ is to evaluate laminar and turbulent heat fluxes and select the highest one.

The heat flux for laminar flow can be written as:

$$q_{w,lam} = \frac{k}{y_f(T_w - T_{f,lam})}$$

The near wall fluid temperature for laminar flow $T_{f,lam}$ is expressed in terms of the nodal temperature T_n using:

$$T_{f,lam} = T_w - 2(T_w - T_N)$$

so that:

$$q_{w,lam} = 2\frac{k}{y_f(\mathrm{T}_w - \mathrm{T}_N)}$$

For turbulent flow, the heat flux is:

$$q_{w,turb} = pC_p(g\beta\alpha)^{1/3}\frac{\mathrm{T}_q^{1/3}}{\mathrm{T}_f^*(\mathrm{T}_w - \mathrm{T}_{\mathrm{f,turb}})}$$

The near wall fluid temperature for turbulent flow $\mathrm{T}_{f,turb}$ is expressed in terms of the nodal temperature T_N using:

$$\mathrm{T}_{f,turb} = \mathrm{T}_w - \frac{(\mathrm{T}_w - \mathrm{T}_N)\mathrm{T}_f^*}{\mathrm{T}_N^*}$$

which gives:

$$q_{w,turb} = pC_p(g\beta\alpha)^{1/3}\frac{\mathrm{T}_q^{1/3}}{\mathrm{T}_N^*(\mathrm{T}_w - \mathrm{T}_{f,turb})}$$

The turbulent heat flux corresponds to an amplification of the laminar heat flux, that is:

$$q_{w,turb} = \frac{y_f^*}{(2NT_N^*)q_{w,lam}}$$

The amplification factor $\frac{y_f^*}{(2NT_N^*)q_{w,lam}}$ tends to 1 when $y_f < 1$ (laminar flow). Test done with the flow solver indicate that the turbulent heat flux tends naturally towards the laminar heat flux for cases where $\phi > 150^\circ$ (heated surface facing downward or cooled surface facing upward).

8.1.12 Slip wall treatments for gases

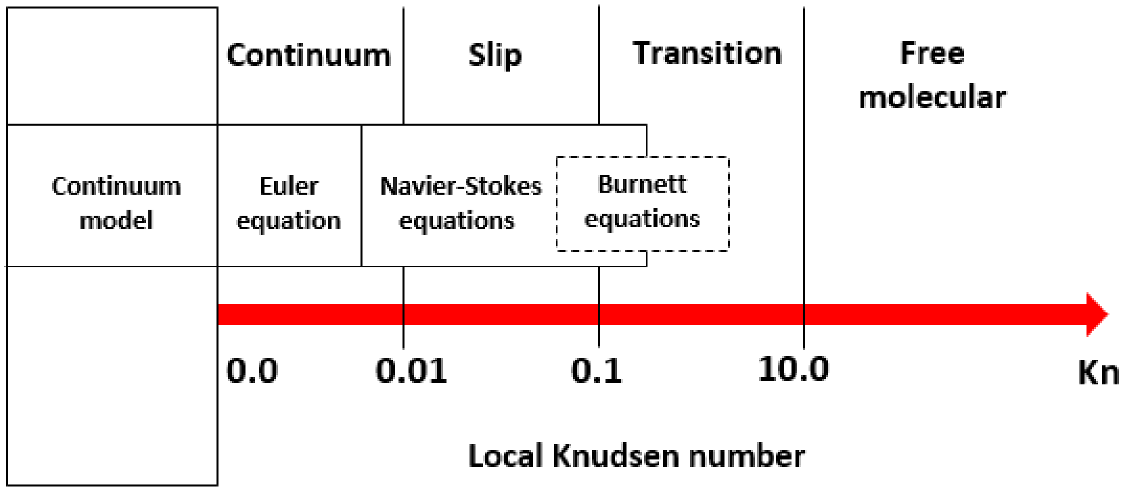
For the flow regimes at low absolute pressure or in gas flows involving micro-channels, where the Knudsen number Kn is small, the no-slip boundary condition is no longer valid. The flow solver uses slip wall treatment for gas flows with Knudsen numbers ranging from 0.01 to 0.1:

$$Kn = \frac{\lambda}{L_0}$$

where

- L_0 is the characteristic of the flow.
- λ is the mean free path of the gas.

This regime is referred to as the *slip regime*, [43].



Within this regime, the flow solver uses the standard momentum and energy equations to model the gas flow with the following modifications to the boundary treatments:

- The velocity has a non-zero finite value, called slip velocity.
- Thermal creep effects can be taken into account.
- The gas temperature at the wall can be different from the wall temperature. There can be a temperature jump at the wall.

Slip velocity

The flow solver uses a second order extension of Maxwell's model [44] to approximate the slip velocity. The traditional no-slip boundary condition is relaxed to allow the rarefied gas to slip at the wall by imposing a tangential component of the slip velocity at the solid boundary. The expression of the slip velocity is defined as:

$$U_s - U_w = C_1 \lambda \left(\frac{\partial U}{\partial n} \right) - C_2 \lambda^2 \left(\frac{\partial^2 U}{\partial n^2} \right)$$

where:

- U_s is the slip velocity.

- U_w is the wall velocity.
- n is the local normal direction of the wall.
- C_1 and C_2 represents respectively the first-order and second-order slip coefficients.

To impose the Maxwell model to all walls that have slip wall specified, use the `USE LOW-PRESSURE MAXWELL SLIP WALL TREATMENT` advanced parameter. This advanced parameter replaces the standard slip wall treatment by the Maxwell model.

By default, $C_1 = 1$ and $C_2 = 0$. You can specify them using the `MAXWELL SLIP WALL C`1 and `MAXWELL SLIP WALL C2` advanced parameters.

You have to enable the **Velocity Adjusted** result set to view the results for the slip velocity. You can view the results in the output `slipValues.dat` file located in the run directory when you specify the `OUTPUT LOW-PRESSURE MAXWELL SLIP WALL SUMMARY` advanced parameter.

Thermal creep effect

The Maxwell's model with included thermal creep effect, induced in rarefied flow with large tangential wall temperature gradient is given by:

$$U_s - U_w = C_1 \lambda \left(\frac{\partial U}{\partial n} \right) - C_2 \lambda^2 \left(\frac{\partial^2 U}{\partial n^2} \right) + \frac{3}{2\pi} \frac{(\gamma - 1)}{\gamma} \left(\frac{\lambda^2 \rho c_p}{\mu} \right) \left(\frac{\partial T}{\partial s} \right)$$

where

- γ is the specific heat ratio defined as $\gamma = c_p / c_v$, where c_p is the specific heat at constant pressure and c_v is the specific heat at constant volume.
- T is the absolute temperature of the gas.
- s is the local tangential direction of the wall.
- μ is the dynamic viscosity of the gas.
- ρ is the density of the gas.

You can include the thermal creep effect only in the case of a non-isothermal fluid using the `INCLUDE THERMAL CREEP TERM IN MAXWELL MODEL` advanced parameter.

Temperature jump

The flow solver uses a temperature jump condition derived by von Smoluchowski [45] to account for the effect of gas temperature. The temperature jump expression is defined as:

$$T_s - T_w = \frac{2-\sigma_T}{\sigma_T}\left(\frac{2\gamma}{\gamma+1}\right)\frac{\lambda}{Pr}\left(\frac{\partial T}{\partial n}\right)$$

where

- T_s is the slip temperature of the gas.
- T_w is the wall temperature.
- σ_T is the thermal accommodation coefficient.
- $Pr = \frac{\mu c_p}{k}$ is the Prandtl number.
- k is the thermal conductivity of the gas.

To impose the Maxwell low-pressure temperature jump model, use the `USE LOW-PRESSURE MAXWELL TEMPERATURE JUMP TREATMENT` advanced parameter. By default, the value for the thermal accommodation coefficient is 1. You can modify it using the `MAXWELL SLIP WALL SIGMA T` advanced parameter. Use the `OUTPUT LOW-PRESSURE MAXWELL SLIP WALL SUMMARY` advanced parameter to view the results in the output *slipValues.dat* file located in the run directory.

Mean free path of the gas

The mean free path of the gas is calculated locally at every wall-adjacent element by using the local gas density, pressure, and temperature.

$$\lambda = \lambda_0\left(\frac{T}{T_0}\right)\left(\frac{P_0}{P}\right)\left(\frac{1+\frac{S}{T_0}}{1+\frac{S}{T}}\right)$$

where

- T_0 is the specified reference temperature of the gas. You specify it using the `REFERENCE TEMPERATURE FOR SLIP CORRECTION` advanced parameter.
- P_0 is the specified reference pressure of the gas. You specify it using the `REFERENCE PRESSURE FOR SLIP CORRECTION` advanced parameter.
- S is the specified Sutherland constant for the viscosity of the gas. You specify it using the `SUTHERLAND CONSTANT FOR SLIP CORRECTION` advanced parameter.
- λ_0 is the reference mean free path of the gas at temperature T_0 and pressure P_0, given by the following equation:

$$\lambda_0 = \frac{\mu_0}{0.491\rho_0}\sqrt{\frac{\pi R}{8k_B T_0 \eta R_g}}$$

- μ_0 is the reference viscosity of the ideal gas. You specify it using the `REFERENCE VISCOSITY FOR SLIP CORRECTION` advanced parameter.
- ρ_0 is the reference density of the ideal gas, which is calculated using ideal gas law.
- k_B is the Boltzmann constant.
- η is the Avogadro's number.
- $R = 8.31434$ (J/mol) is the universal gas constant.
- R_g is the specific gas constant, which is obtained from the gas material properties.

8.2 Fans

In the flow solver, you can use the following fan types to model flow boundary conditions:

- Inlet Flow
- Outlet Flow
- Internal Fan
- Recirculation Loop

8.2.1 Immersed boundary method

When you use the standard k-epsilon or SST turbulence models with immersed boundary meshes, the k, ϵ and ω values that you specify at the Inlet Flow fan type boundary are applied based on the Dirichlet boundary condition in the IBM discretization method.

When you use the standard k-epsilon turbulence model with immersed boundary meshes, the gradient of the k and ϵ values normal to the Outlet Flow fan type boundary are set to zero. When you use the SST turbulence model with immersed boundary meshes, the gradient of the k and ω values normal to the Outlet Flow fan type boundary are set to zero. In both turbulence models, the specified boundary conditions are applied based on the Neumann boundary condition in the IBM discretization method.

8.2.2 Fan boundary condition for energy equation

Specified temperature

The fluid temperature can be specified as a boundary value on inlet flow only. The temperature imposed can either be the ambient temperature or specified temperature.

Temperature change from extract to return

For recirculation loop, you can specify a temperature change from the extract side of the fan to the return side of the fan as:

$$\Delta T = T_{return} - T_{extract}$$

This is equivalent to specifying the value of the return temperature

$$T_{return} = T_{extract} + \Delta T_{spec}$$

Heat generation

The heat generation condition is in fact applied as a temperature specification at the exhaust side or return side of the fan.

Heat generation can be specified on internal fans as well as on recirculation loops. The amount of heat generated by the fan can be expressed as:

$$Q = \dot{m} c_p \Delta T$$

- $\dot{m}$ is the mass flow rate through the fan.
- c_p is the specific heat at constant pressure of the fluid.
- $\Delta T = T_{exhaust} - T_{intake}$ for internal fans.
- $\Delta T = T_{return} - T_{extract}$ for recirculation loops.

8.2.3 Fan boundary condition for scalar equation

General scalar equation

An inlet value of the mass ratio may be specified at inlet flows.

Specified relative humidity or specific humidity

The relative humidity or the specific humidity can be specified at inlet flows even though the transport equation for water vapor in air is in terms of the mass ratio. The conversions from relative humidity to mass ratio and from specific humidity to mass ratio are described in conversion from relative humidity to mass ratio.

8.2.4 Pressure rise

In the context of a fan curve, the term *pressure rise*, often referred to as *delta pressure*, is a fundamental parameter that plays a pivotal role in understanding the performance characteristics of a fan system. The flow solver computes the static pressure rise as the difference between the fan exhaust static pressure, P_E^S , and the fan intake static pressure, P_I^S .

$$\Delta P = P_E^S - P_I^S$$

This section demonstrates the flow solver's process for computing static pressure within the context of a fan with pressure rise, as defined by its fan curve. This page also explains the calculation methods when the fan serves as an inlet, outlet, internal, or recirculation fan.

Inlet fan with pressure rise

When the solver computes the pressure rise for the inlet fan, it assumes that the intake of an inlet fan is linked to the environment with ambient conditions, while the fan's exhaust is connected to the inlet of the computational domain.

Therefore, the pressure imposed at the exhaust of the domain is the static pressure, P_E^S . It is defined as:

$$P_E^S = P_I^T + \Delta P$$

where the total inlet pressure is equal to the ambient pressure since the flow is on the incoming side of the fan, $P_I^T = P_a$.

In post-processing, you visualize the relative pressures, which lacks the static pressure component. Thus, what you observe is not precisely the total pressure P_I^T as defined above, which is absolute. This term is now equal to zero since $P_I^T = P_a$.

$$P_{E,POST}^S = \Delta P$$

Outlet fan with pressure rise

When the solver computes the pressure rise for the outlet fan, it assumes that the intake of an outlet fan is linked to the outlet of the computational domain, while the fan's exhaust is connected to the environment with ambient conditions.

The pressure imposed at the outlet is a static pressure at the fan intake, P_I^S . The ambient pressure is known and is equal to the static pressure since the flow is leaving the fan.

$$P_I^S = P_E^S - \Delta P$$

On the other hand, the static pressure at the fan intake is

$$P_I^S = P_I^T - P_I^{dyn}$$

where P_I^{dyn} is the dynamic pressure at the fan intake.

Using the pressure rise equation, the total pressure at the fan intake is:

$$P_I^T = P_E^S - \Delta P$$

Therefore, the static pressure at the fan intake is:

$$P_I^S = P_E^S - \Delta P - P_I^{dyn}$$

In post-processing, you visualize the relative pressures, which lacks the static pressure component. Thus, what you observe is not precisely the static pressure P_I^S as defined above, which is absolute. This term is now equal to zero since $P_I^T = P_a$.

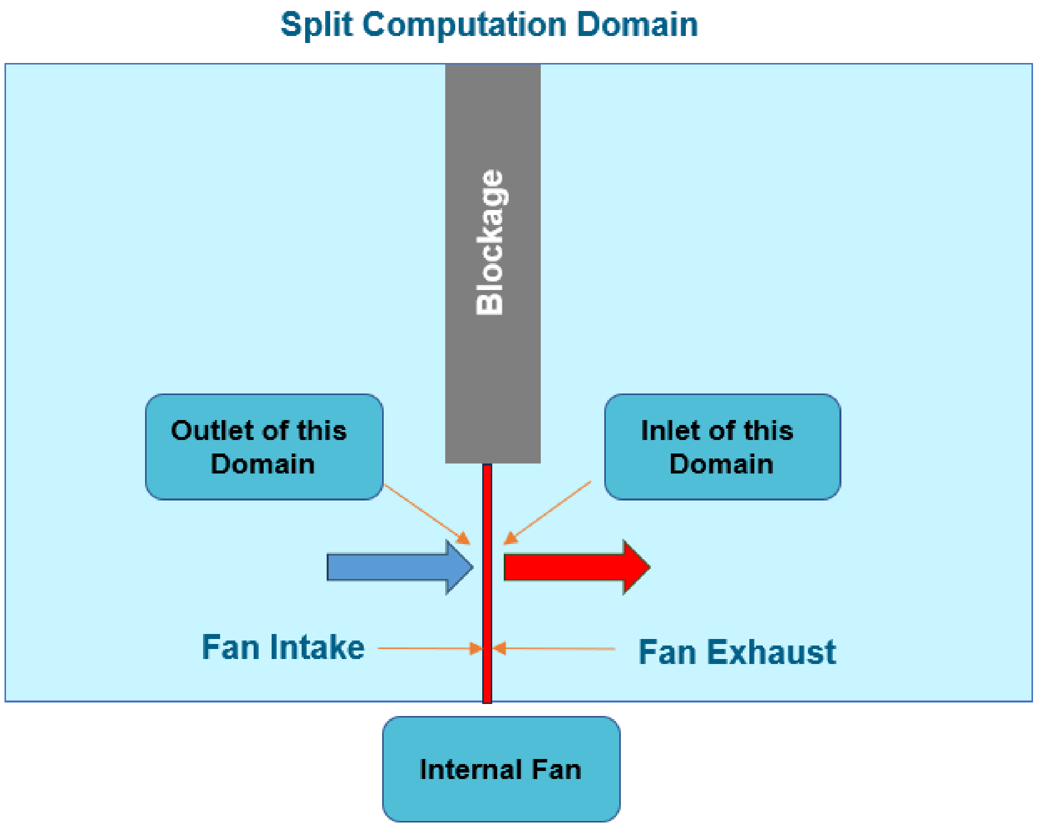
$$P_{I,POST}^S = -\Delta P - P_I^{dyn}$$

Internal fan with pressure rise

For an internal fan, the flow solver duplicates the face, splits the computational domain into an outlet and an inlet at the fan location. The intake of an internal fan is connected to the outlet of the computational domain. The fan's exhaust is connected to the inlet.

The pressure imposed at the inlet of the domain is the static pressure:

$$P_{in}^S = P_E^S = P_I^S + \Delta P = P_{out}^S + \Delta P$$

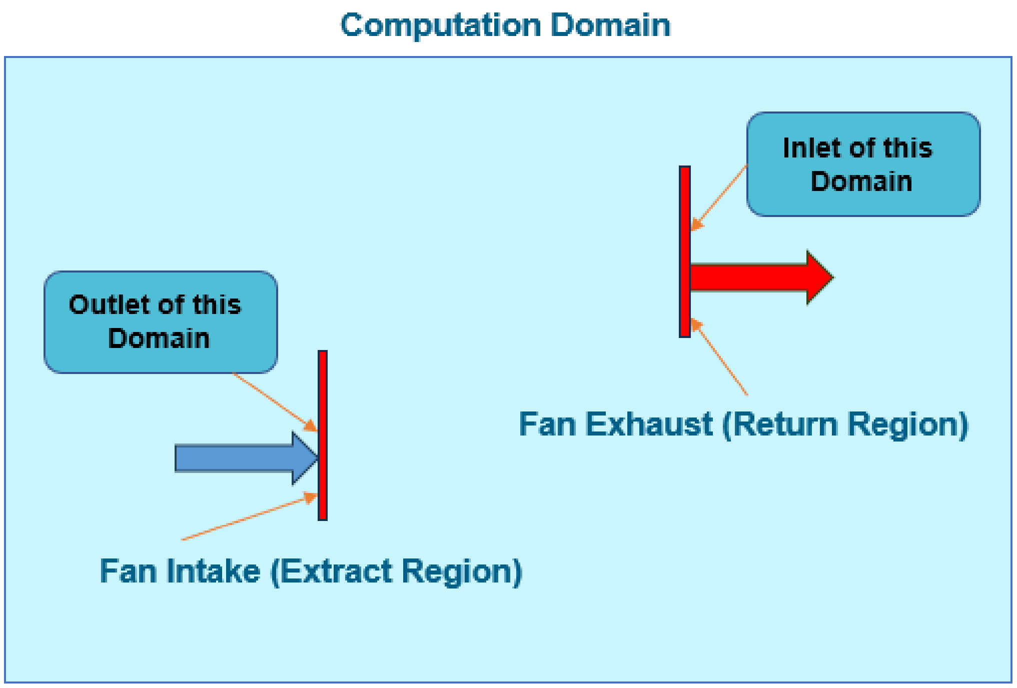


Recirculation fan with pressure rise

A recirculation fan connects an outlet in a computational domain to an inlet of the fan. The intake of a recirculation fan is connected to the outlet. The fan's exhaust is connected to the inlet of the computational domain.

The pressure imposed at the inlet of the domain is the static pressure:

$$P_{in}^{S} = P_{E}^{S} = P_{I}^{S} + \Delta P = P_{out}^{S} + \Delta P$$



8.2.5 Fan Curve

Using the fan curve, you can specify the relationship between the static pressure rise, ΔP, across the outlet, and the volume flow rate $\dot{v}$ circulated by the fan as follows:

$$\dot{v} = f(\Delta P, h)$$

The fan's pressure rise ΔP is computed depending on the fan curve type (inlet fan curve, outlet fan curve, internal fan curve and recirculation fan curve) using the equations in Pressure rise.

Head loss coefficient, h, appears if the fan's pressure rise reduces due to a head loss. For more information, see Head loss on fan.

You can use the implicit or explicit approach to define to define the fan's volume flow rate from the fan curve.

Implicit approach

In the implicit approach, the solver implicitly calculates the volume flow rate at current iteration $(n+1)$ based on the pressure rise at the same iteration $(n+1)$ and the active coefficients of the linear system. The solver obtains the linear system coefficients from the fan curve derivative, based on the fan's pressure rise at previous iteration (n). Therefore, both the fan's pressure rise and fan's volume flow rate are the results of the linear system solution.

Explicit approach

In the explicit approach, the solver computes the volume flow rate at current iteration $(n+1)$ directly from the fan curve itself using the fan's pressure rise at previous iteration (n), and imposes the volume flow rate as an explicit boundary condition to form the linear system. The software then obtains the pressure rise at the current iteration $(n+1)$ by solving the linear system.

In the explicit approach, the pressure rise and volume flow rate must converge through explicit iterations. This could lead to strong fluctuations in the solution convergence based on the characteristics of the fan curve. You have to control these fluctuations using the relaxation techniques. The implicit approach suppresses these fluctuations by making the pressure rise and volume flow rate better converge at each iteration, through implicit coupling in the linear system.

Implicit vs explicit approach

The flow solver supports both implicit and explicit approaches to define the volume flow rate for all fan curve types, except for internal fan that uses always explicit approach.
By default, the flow solver uses the implicit approach. To switch to the explicit approach, you need to set the value of the USE EXPLICIT FAN CURVE advanced parameter to TRUE.

Head loss on fan

In the case of a head loss, the fan curve is modified that for a given volume flow rate, the pressure rise produced by the fan is reduced by the head loss. In the fan curve, the pressure rise is shifted below the nominal fan curve by the amount of ΔP_{loss}.

$$\Delta P_{loss} = \frac{1}{2} U^2 (C_I^s \rho_I + C_E^s \rho_E)$$

where:

- C_I^s and C_E^s are the specified intake and exhaust loss coefficients, respectively.
- ρ_I and ρ_E are the intake and exhaust densities, respectively.
- U is the flow velocity.

Therefore, you can rewrite ΔP_{loss} as follows:

$$\Delta P_{loss} = \frac{1}{2} \dot{v}^2 \frac{C_I^s \rho_I}{A_I^2} + \frac{C_E^s \rho_E}{A_E^2}$$

- A_I and A_E are the intake and exhaust areas.

To obtain the volume flow rate, $\dot{v}$, the solver also needs the value for the head loss coefficient, h, that is defined as:

$$h = \frac{C_I^s \rho_I}{A_I^2} + \frac{C_E^s \rho_E}{A_E^2}$$

Note

If there is no head loss on the fan, h is set to zero.

Volume flow rate calculation with implicit approach

Using the implicit approach, the solver defines the relationship between the fan's pressure rise and volume flow rate using the Taylor series expansion:

$$\dot{v}^{n+1} = \dot{v}|_{\Delta P^n} + \left(\frac{\partial f(\Delta P)}{\partial P}\right)_{\Delta P^n} (\Delta P^{n+1} - \Delta P^n)$$

- $\dot{v}$ is a volume flow rate.
- $f(\Delta P)$ represents the fan curve.
- The superscripts n and $n+1$ refer to the previous and current iterations, respectively.

The pressure rise, ΔP^n, volume flow rate, $\dot{v}|_{\Delta P^n}$, and gradient, $\left(\frac{\partial f \Delta P}{\partial P}\right)_{\Delta P^n}$, are evaluated from the solution at iteration, n. The updated volume flow rate, $\dot{v}^{n+1}$, is evaluated implicitly at the current iteration, $n+1$.
In the implicit approach, the solver takes the fan curve data local to elements over the two faces on opposite sides of the fan and obtains a normal velocity vector for each element of a face to impose the Dirichlet boundary condition. The solver evaluates $\dot{v}|_{\Delta P^n}$ on the fan curve for a particular face element and obtains $\left(\frac{\partial f \Delta P}{\partial P}\right)_{\Delta P^n}$ from the derivative of the fan curve at the specified operating point. In this approach, the solver inserts the local velocity into the solution matrix at a particular point based on the pressure difference across a face element. This gives a relationship between the velocity and pressure.

Volume flow rate calculation with explicit approach

The explicit approach consists of calculating the fan's pressure rise at a given iteration, and then obtaining the corresponding volume flow rate from the fan curve:

$$\dot{v}^{n+1} = f(\Delta P^n, v_2^n)$$

The superscripts n and $n+1$ refer to the previous and current iteration levels, respectively. The pressure rise, ΔP^n and v_2^n are evaluated at iteration, n. The updated volume flow rate, $\dot{v}^{n+1}$, is evaluated at the current iteration, $n+1$. This volume flow rate is then imposed as the boundary condition for the next iteration and a new value of pressure rise is calculated. However, when using the explicit approach, the pressure rise and the volume flow rate at a given iteration level are out of sync until the solution starts to reach some level of convergence, which can lead to some fluctuations in the solution convergence.

The solver then uses the calculated volume flow rate through the fan to obtain the normal velocity vectors on both faces of the fan: s_1 and s_2. The normal velocity vectors, $\bar{U}_n(s_1)$ and $\bar{U}_n(s_2)$, become Dirichlet boundary conditions on the opposing sides.

8.2.6 Flow direction

Flow angle on fan

The flow through an inlet fan or an internal fan can be specified to be at an angle θ from the fan normal. In such cases, the velocity component normal to the fan, U_n , is the velocity obtained from the fan curve, velocity, mass flow, volume flow, or pressure rise specification. The velocity component parallel to the fan plane, U_t , is then calculated so that the total velocity vector is in the specified flow direction:

$$U_t = U_n \tan \theta$$

Swirl on fan

Swirl can also be imposed on inlet and internal fans by specifying an axis of rotation and an angle θ from the fan normal. At each computation point on the fan, the velocity is calculated such that it has a component normal to the fan plane and a component tangential to that plane in the direction of rotation.
The velocity component normal to the fan plane, U_n , is obtained from the mass flow rate, velocity, fan curve or pressure rise specification, as usual, whereas the tangential component, U_t , is calculated from:

$$U_t = U_n \tan \theta$$

8.3 Openings

Openings, also called vents, are pressure and temperature specified boundaries. The boundary condition imposed depends on the direction of the flow: inflow or an outflow. The direction of the flow at a vent can vary during the solve.

The flow solver uses the node-based treatment, which means that the control volume around each node of a vent can have incoming or outgoing flow separately from its neighbors. For separated or reversed flow at an opening, you can activate the faceset-based treatment for openings by using the `USE FACESET-BASED TREATMENT FOR`

`OPENINGS` advanced parameter. The direction of the total mass flow through the opening faceset is used to determine if the flow is coming in or going out of the domain for all elements of the opening.

8.3.1 Inflow vent

Pressure

At inflow vents, you specify the total pressure, $P_{spec} = P_{total}$, but the solver imposes the static pressure:

$$P_{static} = P_{spec} - \frac{1}{2}\rho U_n^2$$

where U_n is the velocity component normal to the vent.

You can also specify a head loss coefficient, h , at a vent to simulate a screen or a filter at that opening. In such cases, the static pressure is computed as follows:

$$P_{static} = P_{spec} - \frac{1}{2}\rho h U_n^2$$

When an inflow vent is attached to a rotating frame of reference, the static pressure is computed as follows:

$$P_{static} = P_{spec} - \frac{1}{2}\rho h U^2$$

where U is the absolute velocity magnitude.

Temperature

The temperature is applied only at inflows. The temperature is either the specified value or ambient temperature.

Turbulence quantities

The solver applies turbulence quantities only at inflow vents. Depending on the turbulence model and the specified turbulence quantities, the following equations are used to compute the turbulence kinetic energy, k , the dissipation rate, ε , or the specific dissipation rate, ω :

$$k = \frac{3}{2}(U_0 I_x)^2$$

$$\epsilon = \frac{k\sqrt{k}}{l_t}$$

$$\omega = \frac{\sqrt{k}}{0.09 l_t}$$

$$\epsilon = 0.09 k\omega$$

When you do not specify any turbulence quantities, the solver computes them using:

- A turbulence intensity, I_x , of 4%.
- An eddy length scale, l_t , computed as follows:

$$l_t = 101,284 \times \frac{\mu}{I_x}\sqrt{\frac{\rho}{2|\Delta P|}}$$
$$0.01\sqrt{A} \le l_t \le 0.25\sqrt{A}$$

- U_0 is the mean flow velocity at the boundary condition.
- μ is the fluid dynamic viscosity.
- ρ is the fluid density.
- ΔP is the maximum pressure difference of the complete fluid domain.
- A is the surface area of the vent.

Immersed boundary method

When you use the standard k-epsilon or SST turbulence models with immersed boundary meshes, the k , ϵ and ω values that you specify at the inflow boundary are applied based on the Dirichlet boundary condition in the IBM discretization method.

Relative humidity or specific humidity

When you model humidity and condensation, you can specify the value of the relative humidity or of the specific humidity at vents. This value is applied only at inflow vents.

Mass ratio for general scalar equation

When you model additional general scalar quantities, you can specify an inlet value of the mass ratio at vents. This value is applied only at inflow vents.

Angle from vent

You can specify the angle, θ, at which the flow enters the fluid domain. It is the angle between the flow direction and the vent normal. The flow solver calculates the velocity component parallel to the vent, U_t, so that the total velocity vector is in the specified flow direction, as follows:

$$U_t = U_n \tan\theta$$

8.3.2 Outflow vent

At outflow vents, you can only specify the pressure. The solver imposes the static pressure that you specify locally or the specified ambient pressure.

$$P_{spec} = P_{static}$$

$$P_{spec} = P_{ambient}$$

Immersed boundary method

When you use the standard k-epsilon turbulence model with immersed boundary meshes, the gradient of the k and ϵ values normal to the outflow boundary are set to zero. When you use the SST turbulence model with immersed boundary meshes, the gradient of the k and ω values normal to the outflow boundary are set to zero. In both turbulence models, the specified boundary conditions are applied based on the Neumann boundary condition in the IBM discretization method.

8.4 Convective outflow

The convective outflow boundary condition models flow exiting the fluid domain without specifying the pressure. It lets the flow field exit and enter the flow domain on each element of the boundary as necessary conserving the mass.

At the centroids of each face element lying on the convective outflow boundary, the dependent variable fields, φ, are computed from a discretized form of the advection equation:

$$\frac{\partial\varphi}{\partial t}|_{bnd} + \vec{U}\cdot\vec{\nabla}\varphi|_{bnd} = 0$$

For the purposes of the above boundary condition equation, the advecting velocity field is taken as uniform, in the direction normal to the boundary. The solver computes the magnitude of the velocity, U, from the average velocity of all other flow boundaries to conserve mass of the flow domain.

$$\vec{U} = U\vec{n}$$

To compute the diffusive and advective fluxes, the solver derives an expression for the values of the dependent variables at the boundary face elements as a function of the values at the nodes of the adjacent solid element, and imposes them implicitly.

For the mass equation on the convective outflow boundary, the flow solver computes the outflow velocity field explicitly based upon the current nodal field. It is then scaled to conserve mass, accounting for density variation with time.

8.4.1 Transient initialization

Because the temporal derivative is often dominant in the boundary value equation, a reasonable initial condition is crucial. Therefore, a small number of initializing steps are performed in which the outflow boundary is treated as an opening, to obtain a reasonable and conservative starting point for the solution of the evolution of the boundary field.

8.4.2 Steady state assumption

For better stability, in steady state runs, the flow solver computes the advected values from a zero-normal-derivative condition, with temporal derivative explicitly neglected. This assumption is true in the converged steady state solution. It removes the need for the initializing iterations and improves convergence and robustness. Because the conservation of mass is explicitly enforced over the entire domain, the pressure field at the outflow boundary can evolve naturally, so that the upstream flow features are not strongly influenced by nearness of the boundary.

8.5 Bursting membrane and flap

The *bursting membran*e boundary condition models a flow surface that bursts when a specified static pressure difference across the membrane or a specified time is reached. After the burst, the fluid flows freely from one side to the other in the fluid domain.

The *flap* boundary condition models the opening and the closing of the flap. A closed flap is represented by a flow surface; while, an open flap is represented by an opening where the fluid flows freely from one side to the other in the fluid domain.

The specified criteria for flap opening include:

- Specified static pressure difference across the flap
- Specified time

The specified criteria for flap closing include:

- Falling below a specified dynamic pressure difference across the flap
- Specified time

The flow solver uses the following assumptions:

- When the membrane bursts, the opening that replaces it remains even if the pressure is decreased to lower values.
- A displacement or motion of the membrane or flap surface is not modeled, and effects of the motion of the membrane or flap on the flow are neglected.
- When the bursting membrane or flap is embedded in the fluid domain, the fluid mesh is continuous across the interface.
- When the bursting membrane or flap is on the fluid domain boundary, it is assumed that the fluid follows the ideal gas law.
- When the bursting membrane or flap is on the fluid domain boundary, only one bursting membrane or one flap can be defined.

8.5.1 Embedded bursting membrane or flap

Switching between closed and open modes leads to a sudden change from an impermeable surface to a fully permeable interface in a flow direction. This cause a numerical pressure wave propagation destabilizing the convergence of the solver.

To alleviate this issue, the flow solver uses a screen relaxation method, where switching between the embedded flow surface and the zero head loss screen is carried out by imposing variable head loss for the screen through many iterations:

- In a steady state model, the relaxation is performed through steady state iterations.
- In a transient model, the relaxation is performed in one time step and through many non-linear iterations.

Once switching the condition is initiated, the flow solver calculates automatically the variable screen head loss coefficient with the dynamic pressure-based screen formulation, according to:

$$h = \frac{\Delta P}{0.5\bar{\rho}\bar{U}^2}$$

where

- h is the head loss coefficient.
- $\bar{\rho}$ is the area averaged density.
- $\bar{U}$ is the average velocity in the upsteam fluid domain

At each iteration, the head loss coefficient decreases towards zero.

For a flap, the switching process from the open state to the closed state starts with increasing the head loss coefficient, h, at each iteration from zero to its most up-to-date value at previous closed state. The number of iterations to complete the switching process depends on the maximum number of iterations in the simulation and the simulation type: transient or steady state.

8.5.2 Boundary bursting membrane or flap

After the membrane bursts or the flap opens, the flow solver defines a closed fluid region to insure that the pressure value in the fluid domain accurately follow the Gay-Lussac's Law of pressure-temperature with constant, [49]. The closed region consists of a group of 3D elements that are enclosed by boundary of the fluid domain and one bursting membrane or flap.

The flow solver must re-level the pressure value in the closed fluid region. When the flow solver detects a closed region, it computes the initial mass m_{ini} using the initial density ρ_{ini} :

$$m_{ini} = \int \rho_{ini} dV$$

The conservation of mass must be globally satisfied in the closed region through the ideal gas law:

$$m_{ini} = \sum_{n \in \Omega} \rho_n \Delta V_n = \sum_{n \in \Omega} \left(\frac{P_n^{abs}}{RT_n^{abs}} \right)_n \Delta V_n$$

where:

- n is the index of the control volume around a mesh node.
- Ω is the set of all control volumes within the closed region,
- ρ_n is the density of the control volume.
- ΔV_n is the volume of the control volume.
- T_n^{abs} is the absolute temperature of the control volume.
- $R = 8.31434$ (J/mol) is the universal gas constant.
- P_n^{abs} is the absolute pressure of the control volume, defined as follows:

$$P_n^{abs} = P_{offset}^{\Omega} + P_n$$

- P_n is the pressure of the control volume after the linear system solve.

The pressure offset within the closed region is defined as follows:

$$P_{offset}^{\Omega} = \frac{(m_{ini} - \alpha)}{\beta}$$

where

$$\alpha = \sum_{n \in \Omega} \left(\frac{P_n}{RT_n^{abs}} \right)_n \Delta V_n$$

and

$$\beta = \sum_{n \in \Omega} \left(\frac{1}{RT_n^{abs}} \right)_n \Delta V_n$$

The pressure re-leveling procedure is complete by the following relation:

$$P_n' = \begin{cases} P_n & : n \notin \Omega \\ P_n + P_{offset}^{\Omega} - P_{offset} & : n \in \Omega \end{cases}$$

The pressure relative to the global offset pressure, P_n' , is used in all calculations. This procedure accounts for an accurate increase in the static pressure difference across the boundary membrane or flap.

8.6 High speed flow boundary condition

You can use a high speed flow boundary condition for inlets or outlets, to specify the Mach number, pressure, and turbulence values at the domain boundary when the flow has velocities above Mach 0.3, which implies that the flow has density changes of more than 5% and is generally treated as compressible.

The *Mach* number is defined as:

$$M = \frac{U}{c}$$

- U is the local speed.
- c is the speed of sound.

The speed of sound c is the speed at which sound propagates through a medium under specific conditions. In ideal gases, c is dependent on the molecular weight, and it is a function of temperature:

$$c = (kRT)^{0.5}$$

- k is the adiabatic exponent.
- R is the gas constant.
- T is the absolute temperature.

In real gases, in which the density and specific heat capacity are functions of temperature and pressure, the specific heat at constant volume, C_v , is defined as:

$$C_v = C_p - \frac{T}{\rho^2} \left(\frac{\partial \rho}{\partial T} \right)_p^2 \left(\frac{\partial \rho}{\partial p} \right)_T^{-1}$$

where:

- C_p is the temperature dependent, or bivariate pressure and temperature dependent specific heat.
- ρ is the pressure dependent, or bivariate pressure and temperature dependent density.
- p is the pressure.

For real gases, the speed of sound is computed by:

$$c = \sqrt{\frac{C_p}{C_v}\left(\frac{\partial \rho}{\partial p}\right)_T^{-1}}$$

The specified *Mach* number is converted in a velocity using the speed of sound in the fluid. This velocity is then applied to the faces you select as a supersonic or subsonic inlet or outlet. The flow solver selectively applies the boundary values you specified depending on the local flow condition.

The flow solver uses the general energy equation for all flows. For more information, see energy equation.

For low speed flows (Mach < 0.3) the energy equation is simplified. The pressure work and dissipation terms are neglected. In this case, the flow solver uses the low speed equation.

8.7 Screens

The flow solver models planar resistances such as screens by calculating the pressure drop, ΔP, across them:

$$\Delta P = \frac{1}{2}\rho h U_n^2$$

- U_n is the velocity component normal to the screen.
- h is the head loss coefficient.

The direction of the velocity components on either side of the screen are identical, unless a flow angle is specified at the screen. This flow angle and the pressure drop are accounted for in the mass and momentum equations.

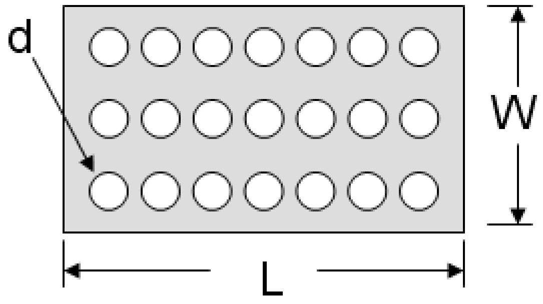
You must specify one of the following:

- The total head loss coefficient h.
- The coefficient b, which is the linear proportionality constant, that relates the pressure drop to the normal velocity component: $\Delta P = bU_n$. In this case, b has units of mass flux.
- A correlation to calculate the head loss coefficient.

When you choose to compute the head loss coefficient, h, the following correlations are available:

- Thin perforated plate screen correlation
- Wire screens correlation
- Silk thread screens correlation

8.7.1 Thin perforated plate screen correlation



For this geometry the following correlation applies:

$$h = (h'(1 - F_{AR})^{0.5} + 1 - F_{AR})^2 / F_{AR}^2$$

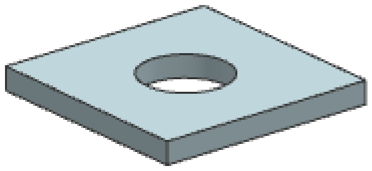
Thin perforated plate screen correlation

The term, h' , depends on the geometry of the perforations from the screen. To obtain, h' , you can select between the following edge geometries: sharp edge, rounded edge, beveled edge.

The following definitions are used to define the thin perforated plate:

- L is the length of the screen.
- W is the width of the screen.
- d is the diameter of a single orifice.
- P_0 is the perimeter of a single orifice.
- A_0 is the area of a single orifice.
- $A_{FS} = LW$ is the area of the free stream.
- $F_{AR} = (\sum A_0)/A_{FS}$ is the free area ratio.
- t is the thickness of the plate.
- $d_h = 4A_0/P_0$ is hydraulic diameter of a single orifice.
- U is the viscosity of the fluid.
- $Re_d = V_n d_h / v$ is the orifice Reynolds number.

Sharp Edge



Sharp edge orifice

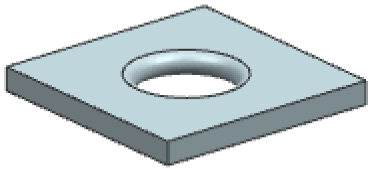
The term, h' , for sharp edge orifice is:

$$h' = 0.707$$

The following assumptions apply:

- $t/d_h < 0.015$
- $Re_d > 10^5$ where Re_d

Rounded Edge



Rounded edge orifice

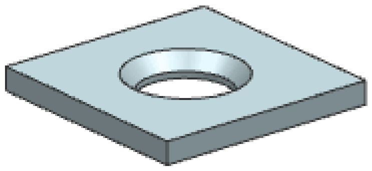
The term, h' , for rounded edge orifice is:

$$h' = 0.03 + 0.47.10^{\frac{-7.7r}{dh}}$$

The following assumptions apply:

- Grid thickness is the same as orifice radius.
- $Re_d > 10^3$.

Beveled Edge



Beveled edge orifice 40 - 60 degrees

The term, h' , for beveled edge orifice is:

$$h' = 0.13 + 0.34.10^{f''}$$

$$h'' = -\left(\frac{3.4t}{d_h} + 88.4\left(\frac{t}{d_h}\right)^{2.3}\right)$$

The following assumptions apply:

- The beveled edge of the orifice is facing the flow.
- The bevel angle is between 40° and 90°.
- $0.01 < t/d_h < 0.16$
- $Re_d > 10^4$

8.7.2 Wire screens correlation

For wire screens the following correlation applies:

$$h = 1.3(1 - F_{AR}) + ((1/F_{AR}) - 1)^2$$

- A_0 is the area of a single orifice.
- A_{FS} is the area of the free stream.
- $F_{AR} = \sum A_0/A_{FS}$ is the free area ratio.

The following assumption applies:

- $Re_d > 10^3$

8.7.3 Silk thread screens correlation

For silk thread screens the following correlation applies:

$$h = 1.62[1.3(1 - F_{AR}) + ((1/F_{AR}) - 1)^2$$

- A_0 is the area of a single orifice.
- A_{FS} is the area of the free stream.
- $F_{AR} = \sum A_0/A_{FS}$ is the free area ratio.

The following assumption applies:

- $Re_d > 500$

8.8 Symmetry boundaries

A model is symmetric about a plane when the flow on one side of the plane is a mirror image of the flow on the opposite side of the plane. In such case, the use of a symmetry plane condition enables efficient use of computer resources by allowing the numerical solution to be obtained on a fraction of the original domain.

The application of a symmetry plane condition to a planar surface of the grid means that the solution on the complete geometry is a reflection of itself about the symmetry plane.

For the mass equation and all scalar equations (energy, water vapor, passive component, k omega, and ε), a zero flow condition is imposed at the symmetry plane.

For the momentum equations, the symmetry condition is specified such that all vector values are parallel to the plane of symmetry.

8.8.1 Immersed boundary method

When you use the standard k-epsilon turbulence model with immersed boundary meshes, the gradient of the k and ε values normal to the symmetry boundary are set to zero. When you use the SST turbulence model with immersed boundary meshes, the gradient of the k and ω values normal to the symmetry boundary are set to zero. In both turbulence models, the specified boundary conditions are applied based on the Neumann boundary condition in the IBM discretization method.

8.9 Periodic boundaries

Periodic boundary conditions are used to simulate a flow leaving through a boundary A and entering through a boundary B under identical conditions (velocity, temperature, scalar values, etc).

The periodic boundary conditions act as if the solution domain is rolled up so that boundaries A and B become adjacent. Any pair of periodic boundaries must have similar shape and size. They can be either parallel to one another, or one boundary is the copy of the other periodic boundary, rotated by an angle ϕ with respect to a specified axis of rotation.

8.9.1 Periodicity with a pressure drop

The flow solver applies the boundary conditions for translational periodicity as follows:

$$\rho_A = \rho_B, \quad U_A = U_B$$

The flow solver computes pressure drop for incompressible or isentropic relations, for compressible cases, using Bernoulli's equation:

$$P_A + \rho_A g h_A + \frac{\rho U_A^2}{2} = P_B + \rho_B g h_B + \frac{\rho U_B^2}{2} + H$$

where:

- h_A and h_B are the heights with respect to the gravity vector.
- H is a head loss on the fan.

From this equation, the pressure drop is:

$$\Delta P = -H = \alpha U^2 \approx \alpha \left(\frac{Q}{\rho A} \right)^2$$

where:

- α is the proportionality coefficient.
- Q is the calculated mass flow rate.
- A is the cross-sectional area of the region where the periodic boundary condition is specified.

Periodicity with a specified mass flow rate

You can also specify a mass flow rate when you set the translational periodic boundary condition, using the advanced parameters.

When you specify the mass flow rate, the flow solver compares the specified mass flow rate Q^* with the calculated mass flow rate Q and adjusts the pressure drop, based on the relation proposed by Bernoulli's equation. The difference between Q^* and Q lies within the specified relative tolerance.

You apply the mass flow rate Q^* at translational periodic boundary conditions with a pressure drop using the *user.prm* file with the following advanced parameters.

Advanced Paramter	Value	Description
`GLOBAL_MASS_FLOW_RATE_AT_PERIODIC_FACES_TARGET`	double value	Sets a mass flow rate applied to all periodic boundaries. The mass flow rate is applied to the primary periodic region. The mass flow rate units must be consistent with the solution units.
`GLOBAL_MASS_FLOW_RATE_AT_PERIODIC_FACES_RELAX`	double value	Sets the value to relax the approximate pressure drop computed from the mass flow rate and helps to convergence. The default value is 0.5 and it should not be reduced unless the solver experience big oscillation in the solution.
`GLOBAL_MASS_FLOW_RATE_AT_PERIODIC_FACES_RELATIVE_TOL`	double value	Sets the relative tolerance that controls how precise the computed mass flow rate is compared to the specified mass flow rate. By default, the relative tolerance is 1e-4, which is equivalent to 0.01%.

For more information on the *user.prm* file, see Flow solver files.

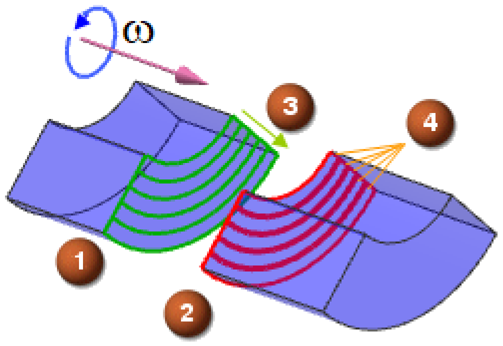
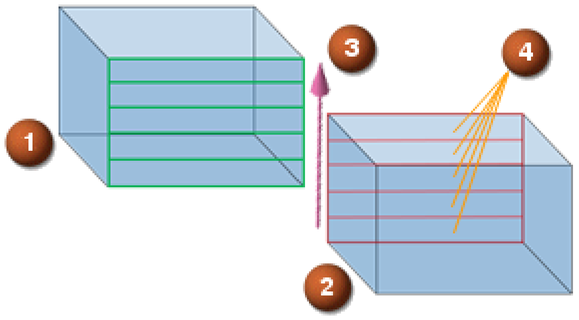
8.10 Mixing plane

A mixing plane boundary condition interfaces two or more fluid volumes with different flow conditions.
The two interfacing faces:

- Can have different geometries.
- Can have a connected interface or be geometrically separated.
- Do not need to be parallel.

The algorithm solves each fluid volume independently. It uses flow variable values from the adjacent fluid volume as boundary conditions.
You can use a mixing plane on two or more interfacing faces with different geometries. The interfacing faces are subdivided in a number of areas you specify. The areas are matched to its closest corresponding area according to their geometry and the averaging method you choose.

Averaging method examples

 **Along radius**	 **Along vector**
1 Upstream 2 Downstream	3 Averaging direction 4 Number of segments (for example: 5)

For each area, the averaged flow variables are transformed to a 1D field using an averaging technique and the information is applied to the other side as a boundary condition.

Depending on the flow direction, for the momentum equation along each area, the downstream static pressure is applied to the upstream side, whereas the total pressure of the upstream side is applied to the downstream side in an explicit order.

The values are mapped between areas to account for different geometry and mesh size at the interfaces.
You define either a radial or axial averaging method.

- When the defined axis is perpendicular to the mixing plane, the flow variable values are averaged along radial arcs at the interfaces.
- When the defined axis is parallel to the mixing plane, the flow variable values are averaged perpendicular to axial segments at the interfaces.

8.11 Rotating frames of reference

The flow solver uses the frozen rotor method for rotating frames of reference (RFR). This method is useful for models where the flow distribution around the interface between the fixed and the rotating frames is non-uniform. The flow solver uses the following Navier-Stokes and Energy equations that use the absolute velocity formulation:

$$\frac{\partial \rho}{\partial t} + \nabla \rho \vec{U_r} = 0$$

Mass equation

$$\frac{\partial \rho \vec{U_a}}{\partial t} + \nabla(\rho \vec{U_r} \vec{U_a}) + \rho(\vec{\omega} \times \vec{U_a}) = -\nabla P + \nabla \bar{\bar{\tau}} + \vec{F}$$

Momentum equations

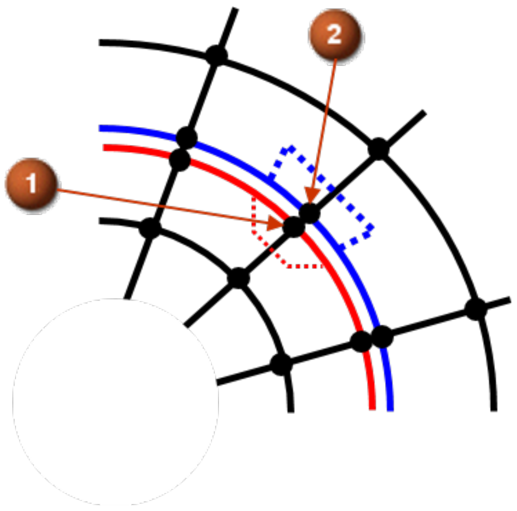
$$\frac{\partial \rho e}{\partial t} + \nabla(\rho \vec{U_r} H) + \nabla(P \vec{U_f}) + = \nabla(k \nabla T + \bar{\bar{\tau}} \vec{U_a}) + S_h$$

Total energy conservation equation

where:
$\vec{U_a}$ is the absolute (inertial) velocity.
$\vec{U_r}$ is the relative velocity in the rotating frame.
$\vec{U_f}$ is the rotating frame velocity relative to the inertial reference frame.
$\vec{\omega}$ is the angular frame velocity.
$\vec{r}$ is the position vector from the origin of the rotating frame.
H is the total enthalpy.

The interface between the frames allows the flow solver to model the frame change with a zero head loss and the flow recirculation across the interface.

The following figure is a schematic representation of the RFR interface with coupled nodes and the control volumes. These nodes are physically at the same location but not in the same frame of reference.



1 The attached node to rotating frame 1 with rotational velocity $\vec{\Omega}_1$.

2 The attached node to rotating frame 2 with rotational velocity $\vec{\Omega}_2$.

At the RFR interface, the flow variables, such as absolute velocities and static pressure, and the total enthalpy are equal between node 1 and node 2:

$$\vec{U}_1 = \vec{U}_2$$

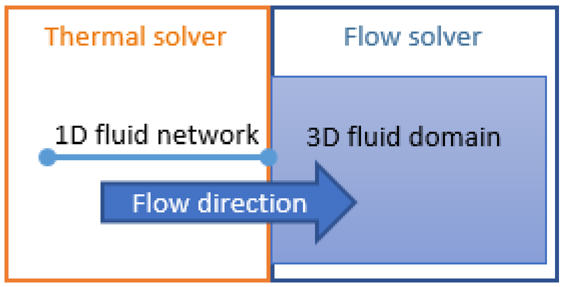
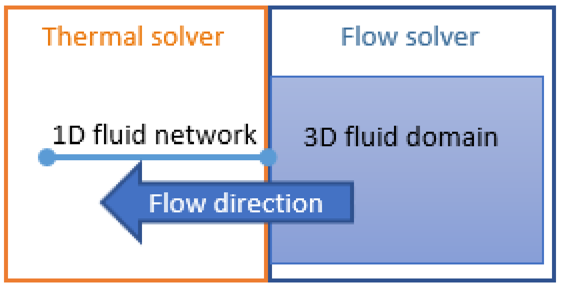
$$P_1 = P_2$$

$$H_1 = H_2$$

8.12 1D network connected to 3D fluid region

In a coupled thermal-flow analysis, you can couple a 3D fluid domain to a set of thermal streams and ducts with mass flow from a 1D fluid network that has the same fluid material. The flow solver solves the equations on the 3D fluid domain using the information from the 1D fluid network solved by the thermal solver.

8.12.1 Flow Directions

Fluid flows from 1D network to 3D domain	Fluid flows from 3D domain to 1D network
 Thermal solver Flow solver 1D fluid network 3D fluid domain Flow direction	 Thermal solver Flow solver 1D fluid network 3D fluid domain Flow direction

When the fluid flows from the 3D domain to the 1D network, you can specify the flow alignment, turbulence characteristics, and define a radially varying swirl in the fluid if your model requires it.

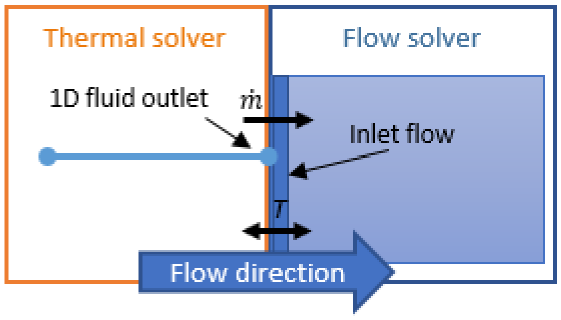
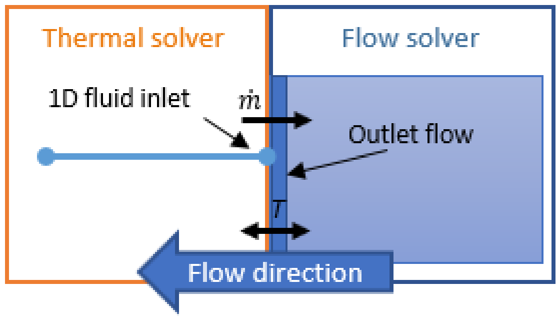
8.12.2 Transfer options between solvers

Irrespective of the flow direction, the thermal solver always transfers the fluid quantities from the 1D fluid network to the 3D fluid domain at the junction. You specify whether the mass flow ($\dot{m}$) or pressure (*P*) is transferred. The flow solver computes the other quantity.

The thermal and the flow solvers exchange the temperature at the junction as the boundary temperature of the other solver.

If the flow direction is from the 3D domain to the 1D network and you request that the thermal solver automatically determines the inlet temperature of the stream, the flow solver transfers the temperature from the 3D flow surface to the inlet of the stream.

8.12.3 Transferring mass flow rate to the 3D fluid domain

Fluid flows from 1D network to 3D domain	Fluid flows from 3D domain to 1D network
 Thermal solver Flow solver 1D fluid outlet $\dot{m}$ Inlet flow T Flow direction	 Thermal solver Flow solver 1D fluid inlet $\dot{m}$ Outlet flow T Flow direction
The flow solver treats the selected 2D surface as an inlet flow with mass flow rate value transferred from the junction by the thermal solver, which is the outlet value of the thermal stream or duct.	The flow solver treats the selected 2D surface as an outlet flow with mass flow rate value transferred from the junction by the thermal solver, which is the inlet value of the thermal stream or duct.

Multiple 1D fluid connections

When transferring mass flow rate, you can connect multiple streams and ducts to the 3D fluid domain at the junction. At each junction, the flow solver controls the mass flow using the conservation of mass equation:

$$\Sigma(\dot{m}_{1D}) = \dot{m}_{3D}$$

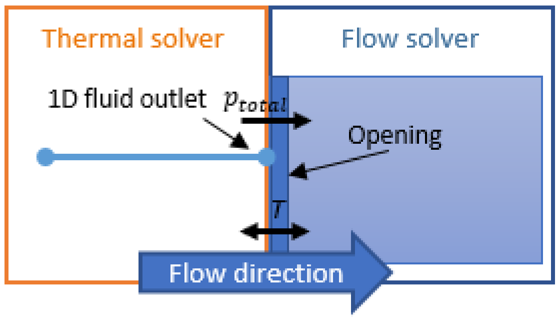
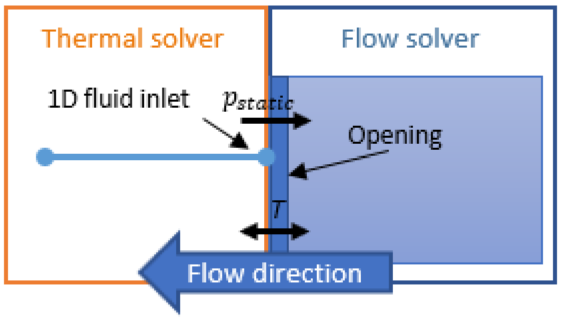
Where:

- $\dot{m}_{1D}$ is the mass flow rate of a 1D duct or stream
- $\dot{m}_{3D}$ is the mass flow rate transferred to the 3D fluid domain.

Because the mass flow value of each stream or duct can be positive or negative, the solvers adapt automatically to the direction of the 1D fluid.

8.12.4 Transferring pressure to the 3D fluid domain

When you specify the pressure transfer, the flow solver always treats the selected 2D surface of the 3D fluid domain as an opening.

Fluid flows from 1D network to 3D domain	**Fluid flows from 3D domain to 1D network**
 Thermal solver Flow solver 1D fluid outlet p_{total} Opening T Flow direction	 Thermal solver Flow solver 1D fluid inlet p_{static} Opening T Flow direction
The 1D fluid pressure from the outlet of the stream or duct is specified as the opening total pressure. For more information, see Inflow vent.	The 1D fluid pressure from the inlet of the stream or duct is specified as the opening static pressure. For more information, see Outflow vent.

The thermal solver allows only one stream or duct to be connected to the 3D fluid domain when a pressure transfer is specified.

You can manually override the pressure value transferred by the thermal solver using an expression, a formula, a table, or a table of fields. For example, this may be useful if you would like to keep a constant pressure value at some locations.

9 Modeling a gas mixtures

The flow solver uses the scalar equation to model a gas mixed in any proportion with the main gas. The software supports up to five gases in the mixture. All gases are assumed to behave as ideal gases using the ideal gas law. For more information, see Ideal gas law.

The flow solver uses the conservation equations of mass, momentum, and energy to compute the gas mixture.

When modeling a gas mixture, the following properties are calculated at every iteration:

- ρ the gas mixture density
- C_p the specific heat at constant pressure of the gas mixture
- k the thermal conductivity of the gas mixture
- μ the dynamic viscosity of the gas mixture

This property update is based on the assumption that the two gases behave as ideal gases. Ideal gas equations of state for two gases, defined as follows:

$$p_1 = \rho_1 R_1 T$$
$$p_2 = \rho_2 R_2 T$$

- p_1 , p_2 represent the partial pressures of gas 1 and gas 2.
- R_1 , R_2 are the gas constants of gas 1 and gas 2.

9.1 Pressure and density of the gas mixture

The pressure of the gas mixture is given by:

$$p = p_1 + p_2$$

The density of the gas mixture is given by:

$$\rho = \rho_1 + \rho_2$$

9.2 Specific heat at constant pressure of the gas mixture

The specific heat of the mixture, c_p , is calculated from the following equations [18]:

$$c_p = \frac{\hat{c}_p}{\hat{M}}$$

- $\hat{c}_p$ is the molar specific heat of the mixture, defined as:

$$\hat{c}_p = \frac{p_1}{p} c_{p1} \hat{M}_1 + \frac{p_2}{p} c_{p2} \hat{M}_2$$

- $\hat{M}$ is the molar mass of the mixture, defined as:

$$\hat{M} = \frac{p_1}{p} \hat{M}_1 + \frac{p_2}{p} \hat{M}_2$$

In these equations, $\hat{M}_1$ and $\hat{M}_2$ are the molar masses of gas 1 and gas 2.

9.3 Thermal conductivity and dynamic viscosity of the gas mixture

The thermal conductivity, k, and the dynamic viscosity, μ, of the mixture are calculated using the method of Wilke [19], which is valid for gases at low pressures.

The property η_m, that represents the the gas mixture thermal conductivity or the gas mixture dynamic viscosity, is given by:

$$\eta_m = \frac{X_1 \eta_1}{X_1 + X_2 \phi_{12}} + \frac{X_2 \eta_2}{X_2 + X_1 \phi_{21}}$$

where ϕ_{12} and ϕ_{21} are given by:

$$\phi_{12} = \frac{[1 + (^{\eta_1}/_{\eta_2})^{1/2} (^{\hat{M}_1}/_{\hat{M}_2})^{1/4}]^2}{\{8[1 + ^{\hat{M}_1}/_{\hat{M}_2}]\}^{1/2}}$$

$$\phi_{21} = \phi_{12} \frac{\eta_2}{\eta_1} \frac{\hat{M}_1}{\hat{M}_2}$$

In these equations:

- η_m, η_1, and η_2 represent the property, thermal conductivity or viscosity, of the gas mixture, gas 1, and gas 2, respectively.
- X_1 and X_2 are the mole fractions of the two gases given by:

$$X_1 = \frac{\hat{M}_2 (1 - \phi)}{\hat{M}_2 (1 - \phi) + \hat{M}_1 \phi}$$

$$X_2 = \frac{\hat{M}_1 \phi}{\hat{M}_2 (1 - \phi) + \hat{M}_1 \phi}$$

9.4 Species enthalpy diffusion in the homogeneous gas mixture

You can include the enthalpy flux to the low-speed energy equation for the homogeneous gas mixture using the `INCLUDE SPECIES ENTHALPY DIFFUSION TERM` advanced parameter. It accounts for the energy changes in the gas mixtures due to species diffusion [38]. You should include the enthalpy diffusion term when species mass fraction gradients are significant as in combustion problems or for the gas mixture with different temperatures at low speed.

The term in the energy equation related to the transport of enthalpy due to the species diffusion is defined as follows:

$$S_h = \nabla \cdot \left[\sum_{i=1}^{n} h_i \vec{J}_i \right]$$

where:

- h_i is the sensible enthalpy for the species i.
- J_i is the diffusive mass flux.

For laminar flow, the value of the diffusive mass flux J_i is defined as follows:

$$\vec{J}_i = -\rho D_i \nabla \phi_i$$

where:

- ρ is the gas mixture density.
- D_i is the molecular diffusion coefficient for species i.
- ϕ_i is the mass fraction of species i in the mixture.

For turbulent flow, the value of the diffusive mass flux J_i is defined as follows:

$$\vec{J}_i = -\left(\rho D_i + \frac{\mu_t}{Sc_t} \right) \nabla \phi_i$$

where:

- μ_t is the turbulent viscosity.
- Sc_t is the turbulent Schmidt number.

The flow solver uses the binary molecular diffusion coefficient D_{i1} as a value for D_i for each solved species i relatively to the primary gas ($i = 1$). For the primary gas, the flow solver uses the following expression:

$$D_1 = \frac{1 - X_1}{\frac{X_2}{D_{21}} + \frac{X_3}{D_{31}} + \ldots + \frac{X_i}{D_{i1}}}$$

where X_i is the molar fraction defined as:

$$X_i = \frac{\phi_i M_{mix}}{M_i}$$

where:

$$M_{mix} = \frac{1}{\sum_{j=1}^{n} \frac{\phi_j}{M_j}}$$

10 Modeling humidity

The flow solver uses the gas mixture scalar equation to model humidity that is defined as a mixture of water vapor in air. For more information, see Modeling a gas mixtures.

Rather than specifying the boundary and initial conditions in terms of mass ratio, in this case, they are specified either in terms of relative humidity or specific humidity. The flow solver supports the specification of relative humidity and specific humidity as function of time.

The preprocessor converts the humidity values to mass ratio values. After the solution of the conservation equations is obtained, the postprocessor coverts the mass ratio values back to relative and specific humidity values.

You do not need to specify the water vapor properties, as they are available in the flow solver.

10.1 Conversion from relative humidity to mass ratio

The flow solver sets to zero the gradient normal to the wall of the mass ratio, when it solves the humidity equation or the other general scalar equation.

By definition, the relative humidity, φ_r , is the ratio of the partial pressure of the water vapor, p_v , to the saturation pressure of the water vapor at a given temperature $p_{v.\ sat}(T)$:

$$\varphi_r = \frac{p_v}{p_{v,sat}(T)} \times 100$$

A value of φ_r equals to 100% indicates the onset of condensation. The air/vapor mixture gets closer to the condensation state if:

- Water vapor is added to the fluid, the partial pressure of water vapor then increases.
- The temperature goes down.

The flow solver uses analytical formulas, proposed in [13], to calculate the water vapor saturation pressure for $-100^\circ C < T < 0^\circ C$ given by:

$$\ln(p_{v.sat}) = C_1/T + C_2 + C_3T + C_4T^2 + C_5T^3 + C_6T^4 + C_7 \ln(T)$$

with the following coefficients:

- $C_1 = 5.6473590 \cdot 10^3$
- $C_2 = 6.3925247$
- $C_3 = 9.6778430 \cdot 10^{-3}$
- $C_4 = 6.2215701 \cdot 10^{-7}$
- $C_5 = 2.0747825 \cdot 10^{-9}$
- $C_6 = -9.4840240 \cdot 10^{-13}$
- $C_7 = 4.1635019$

The water vapor saturation pressure for $0^{\circ}C < T < 200^{\circ}C$ given by:

$$\ln(p_{v.sat}) = C_8/T + C_9 + C_{10}T + C_{11}T^2 + C_{12}T^3 + C_{13}\ln(T)$$

with the following coefficients:

- $C_8 = -5.8002206 \cdot 10^3$
- $C_9 = 1.3914993$
- $C_{10} = -4.8640239 \cdot 10^{-2}$
- $C_{11} = 4.1764768 \cdot 10^{-5}$
- $C_{12} = -1.4452093 \cdot 10^{-8}$
- $C_{13} = 6.5459673$

In these equations, the temperature T is in Kelvin and $p_{v.\ sat}$ is in Pascal.

The mass ratio is then obtained as:

$$\phi = \frac{p_v}{\frac{R_v}{R_a}(p - p_v) + p_v}$$

- p is the pressure of the water vapor/air mixture.
- R_v is the water vapor gas constant.
- R_a is the air's gas constant.

10.2 Conversion from specific humidity to mass ratio

By definition, the specific humidity φ_s is the ratio of the water vapor density to the dry air density, defined as:

$$\varphi_s = \frac{\rho_v}{\rho_a}$$

The conversion from specific humidity to mass ratio is given by:

$$\phi = \frac{\rho_v}{\rho_v + \frac{\rho_v}{\varphi_s}} = \frac{\varphi_s}{1 + \varphi_s}$$

10.3 Condensation and evaporation

In transient analysis, the flow solver computes condensation and evaporation at walls when the ambient fluid is air.

The model assumes film type condensation or evaporation with the following conditions:

- The water film on the surface of the walls is very thin.
- The temperature of the liquid film is assumed to be the same as that of the wall.
- The rate of mass transfer between condensate and the air is small.
- The presence of the condensate does not affect the heat transfer coefficient of the surface.

10.3.1 Mass and scalar conservation equations

The flow solver calculates the flux of water vapor from the film to the air as follows [13]:

$$\dot{m}'' = h_m[\rho_{Bsat}(T_s) - \rho_{Bf}] \quad \text{for} \quad \rho_{Bf} \leq \rho_{Bsat}(T_f)$$

If the water vapor density is greater than the density at saturation (i.e. 100% relative humidity), then the flow solver evaluates the water vapor flux from:

$$\dot{m}'' - \frac{V}{A\Delta t}[\rho_{Bf} - \rho_{B,sat}(T_f)] \text{ for } \rho_{Bf} > \rho_{B,sat}(T_f)$$

The mass transfer coefficient, h_m , is evaluated from the Lewis relation:

$$h_m = \frac{h}{C_{pm}\rho_{Af}}(\frac{\alpha}{D_v})^{2/3}$$

- $\rho_{Bsat}(T_s)$ is the saturation density of water vapor at the wall temperature.
- ρ_{Bf} is the density of the water vapor in the fluid.
- h represents the heat transfer coefficient computed by the thermal solver.
- C_{pm} is the specific heat per unit volume of the air-vapor fluid mixture.
- ρ_{Af} is the density of air in the fluid.
- $\frac{\alpha}{D_v}$ is the ratio of the thermal diffusivity of the fluid mixture to the diffusivity of the water vapor, also called the Lewis number.

Evaporation corresponds to a positive flux of water vapor to the fluid, while condensation corresponds to a negative flux.

The flow solver uses the mass flux as a source term in both the mass and momentum equations and the scalar equation.

10.3.2 Energy conservation equation

The addition or retrieval of water vapor to or from the air as a result of evaporation or condensation also affects the overall enthalpy of the fluid mixture. The energy flux to the fluid mixture resulting from evaporation or condensation is evaluated as:

$$\dot{q}'' = \dot{m}'' C_{p,bf} T_f$$

- $C_{p,bf}$ is the specific heat of the water vapor at the fluid temperature, T_f .

This energy flux is a source term in the energy equation.

10.4 Fog formation

The flow solver computes the fog formation in a gas mixture. The gas mixture constitutes the continuum phase, and the airborne condensate that forms the dispersed phase. The Quadrature Method of Moments (QMOM) scheme, proposed in [53], is used for computing the droplet distribution in the mixture. In addition to the mixture mass, momentum, and energy conservation equations, the flow solver solves the conservation equations for the water vapor and the first 2k moments of the droplet number density function, where k represents the kth moment of the number density function. By default, only the first and second moments are solved.

The conservation equation for the water vapor is as follows:

$$\frac{\partial(\rho\phi_v)}{\partial t} + \frac{\partial(\rho U_j \phi_v)}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\rho D_{\phi_v} \frac{\partial \phi_v}{\partial x_j}\right) - S_{\phi_v}$$

where:

- ρ is the density of the fluid mixture.
- $\phi_v = \frac{\rho_v}{\rho}$ is the vapor mass fraction, where ρ_v is the vapor density.
- D_{ϕ_v} is the effective diffusion coefficient of the vapor into air. It includes both laminar and turbulent diffusions. The turbulent diffusion is computed from the turbulent viscosity and the Schmidt number.
- S_{ϕ_v} is the source term. It includes the water droplet formation and the droplet growth/evaporation effects.

The droplet distribution in the domain is described by a number density function $f(x_i, r, t)$, which is the function of Cartesian coordinates, x_i , the droplet radius, r, and time, t.

The kth moments of the number density function is given by:

$$\frac{\partial(\rho\mu_k)}{\partial t} + \frac{\partial(\rho U_j \mu_k)}{\partial x_j} - k\int r^{k-1} \rho G(r) f dr = \frac{\partial}{\partial x_j}\left(\rho D_f \frac{\partial \mu_k}{\partial x_j}\right) + Jr^{*k}$$

where:

- $\mu_k = \int r^k f dr$
- D_f is the effective diffusion coefficient for the droplet number density function. It includes both laminar and turbulent diffusions. The laminar diffusion coefficient is computed using the mixture dynamic viscosity and the assumption of a Schmidt number of 0.61. The turbulence contribution is obtained using the turbulence viscosity and a turbulence Schmidt number of unity.
- $G(r) = \frac{dr}{dt}$ is the droplet growth rate.
- J is the number of droplets at critical radius, r^* , generated per unit volume of the mixture per unit time, proposed in [54].

The solver uses the nucleation model that corresponds to the homogeneous nucleation where droplets are formed as a result of random collisions of water vapor molecules. The homogeneous nucleation source term is given by [54]:

$$J_{hom} = \frac{q_c}{1+\eta}\left(\frac{2\sigma}{\pi m^3}\right)^{\frac{1}{2}}\frac{\rho_v^2}{\rho_l}exp\left(-\frac{4\pi r^{*2}\sigma}{3k_B T_g}\right)$$

where

- q_c is the condensation coefficient.
- m is the mass of one molecule of water.
- σ is the surface tension.
- k_B is the Boltzmann constant.
- η is the correction factor.
- T_g is the temperature of the saturated vapor.
- ρ_v and ρ_l are the vapor and liquid density, respectivly.

The heterogeneous nucleation where water droplets are formed as a result of the condensation around small suspended aerosol particles is the dominant mode of the nucleation. A model for the heterogeneous nucleation is provided in [55] as follow:

$$J_{het} = 4\pi R^2 n_p J_0 exp\left(-\frac{4\pi r^* \sigma}{3k_B T_g}f(m,z)\right)$$

where:

- R is the average radius of aerosol particles.
- n_p is the number of particles per unit mass of the mixture.
- $J_0 = 10^{25}$ (cm^{-2}s^{-1}) is the nucleation pre-factor.
- f is the factor that depends on the contact angle between the water and the niclei and given by:

$$f(m,z) = \frac{1}{2}\left\{1 + \left(\frac{1-mz}{\kappa}\right)^3 + z^3\left[2 - 3\left(\frac{z-m}{\kappa}\right) + \left(\frac{z-m}{\kappa}\right)^3\right] + \right.$$
$$\left. +3mz^2\left(\frac{z-m}{\kappa} - 1\right)\right\}$$

where m, z and κ are calculated as:

$$m = \cos\theta;$$
$$z = \frac{R}{r^*};$$
$$\kappa = (1 + z^2 - 2mz)^{1/2}$$

Where θ is the contact angle between water and the aerosol particles.

The following table lists the advanced parameters to model fog formation in your simulation.

Advanced parameter	**Default**	**Description**
`SOLVE EXTRA EQUATIONS TO MODEL FOG FORMATION`	False	When you set it to True, it activates the QMOM scheme to model fog formation.
`NUMBER OF MOMENT QUADRATURES TO SOLVE IN THE QMOM SCHEME`	2	Sets the number of moments of the number density function to be solved in the QMOM scheme. This number must be even.
`HETEROGENEOUS NUCLEATION NUCLEI AVARAGE RADIUS`	-1.0	Sets the average radius of aerosol nuclei, r, used for modeling the heterogeneous nucleation.
`HETEROGENEOUS NUCLEATION NUCLEI NUMBER DENSITY`	-1.0	Sets the concentration of aerosol nuclei, n_p, used for modeling the heterogeneous nucleation.
`HETEROGENEOUS NUCLEATION CONSTANT ANGLE`	80.0	Sets the contact angle, θ, between water and the aerosol particles used for modeling the heterogeneous nucleation.

10.5 Semi-permeable flow surface

The flow solver supports the following methods to model semi-permeable flow surfaces for humid air:

- *Humidity based permeability* permits the diffusion of only water vapor.
- *Mixture based permeability* permits the diffusion of humid air mixtures, including air, water vapor, and other gas components.

10.5.1 Humidity based permeability

In this method, the flux of the water vapor, m'', across the boundary is given by the following equation:

$$m'' = \frac{\rho_v^a - \rho_v^e}{R}$$

where

- ρ_v^a is the specified ambient water vapor density. If it is not defined, its value is set to zero.
- ρ_v^e is the water vapor density at the flow surface location in the computational domain.
- R is the specified humidity diffusion resistance. This value can be a constant or time-varying.

The flux crossing the semi-permeable surface depends solely on the difference between the water vapor densities across the interface. An outflow is predicted from a humid enclosure to a dry ambient even when the ambient pressure is much higher than the enclosure pressure. This method does not allow the pressure to be equalized across the boundary in a steady state simulation.

10.5.2 Mixture based permeability

This method is derived from the concept of head loss in the opening boundary condition. The relationship between the specified ambient pressure, P^a, the pressure at the opening, P^e, and the specified head loss, h, is given by the following equations:

- For inflows:
  $$P^a - P^e = \frac{1}{2}\rho(h+1)U^2$$
- For outflows:
  $$P^e - P^a = \frac{1}{2}\rho h U^2$$

Assuming that the velocities, U, in these equations are normal to the surface, the mass flux at the semi-permeable surface is calculated as:

$$m'' \cong \pm\sqrt{\frac{2\rho|P^a - P^e|}{h}}$$

The plus and minus sign corresponds to the sign of the pressure difference, $P^a - P^e$. For an inlet, the following further assumption is made: $h \cong h + 1$. This assumption is valid for very large values of h.

The density ρ is either equal to the mixture density at semi-permeable flow surface or to the ambient density:

$$\rho = \begin{cases} \rho^a & \text{if } P^a - P^e \geq 0 \\ \rho^e & \text{if } P^a - P^e < 0 \end{cases}$$

To suppress the occurrence of alternating inflow and outflow conditions in individual semi-permeable cells at each iteration, the flow solver uses a minimum mass flux criterion. With this criterion, when the mass flux is smaller than the specified minimum mass flux, the mass flux is set to zero. You specify the minimum mass flux using the `MINIMUM SEMI PERMEABLE MEMBRANE FLUX` advanced parameter. By default, the specified minimum mass flux is set to zero.

By default, the mass flux magnitude of each cell is compared to the specified minimum mass flux. If the cell mass flux magnitude is smaller that the specified minimum mass flux, it is set to zero. You can also apply this criterion on the complete semi-permeable flow surface. It this case, the average mass flux magnitude of all flow surface cells is compared to the specified minimum mass flux. If the average value is smaller than the specified minimum mass flux, the mass fluxes of all flow surface cells are set to zero. You specify the global criterion using the `PERFORM MINIMUM FLUX CHECK GLOBALLY` advanced parameter.

The minimum mass flux can be computed from the values specified by the manufacturers of semi-permeable membranes, as follows:

$$m''_{min} = \frac{\rho \dot{v}_{min}}{A_v}$$

where:

- ρ is the density of air that can be approximated to 1.0 kg/m^3.
- $\dot{v}_{min}$ is the minimum volume flow rate given in manufacturer specifications.
- A_v is the vent area that can be computed from the design and dimensions given in manufacturer specifications.

You can specify a constant humidity flux m'' instead of the loss coefficient. To do this add the following line to the *user.prm* file.

```
EXPLICIT_HUMIDITY_FLUX=TRUE
```

The flow solver applies the specified loss coefficient value as the value of the constant humidity flux in SI unit $\left(\frac{kg}{m^2 s}\right)$ crossing the specified wall face.

For more information on *user.prm* file, see Flow solver files.

11 Modeling non-Newtonian fluid

11.1 Non-Newtonian fluids

In non-Newtonian fluids, the shear stress of the fluid is not proportional to the rate of deformation. Therefore, the viscosity is no longer a constant and an additional model is required to model viscosity.

You can use one of the following models in accordance with fluid behavior:

- *Power-Law model* for fluids with dilatant, pseudo-plastic, and similar behaviors.
- *Herschel-Bulkley model* for fluids with Bingham plastic, viscoplastic, and similar behaviors.
- *Carreau model* that is a generalized model that behaves as Newtonian at low shear rate and non-Newtonian for high shear rates.

11.2 Power-Law model

The fluid viscosity, μ, of a Power-Law fluid is defined in the flow solver using the following equation:

$$\mu = K\dot{\gamma}^{n-1} e^{T_0/T}$$
$$\mu_{\mathtt{min}} < \mu < \mu_{\mathtt{max}}$$

where:

- K is the consistency index.
- $\dot{\gamma}$ is the shear rate.
- n is the power law index.
- T_0 is the reference temperature.
- T is the fluid temperature.
- $\mu_{\mathtt{min}}$ is the minimum viscosity limit.
- $\mu_{\mathtt{max}}$ is the maximum viscosity limit.

11.3 Herschel-Bulkley model

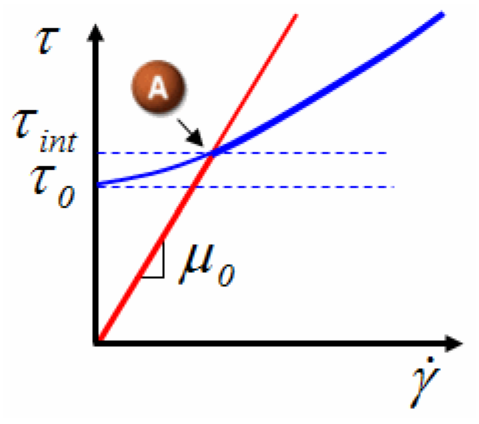
The fluid viscosity, μ, of a Herschel-Bulkley fluid is modelled in the flow solver using the following equation:

$$\mu = K\dot{\gamma}^{n-1} + \frac{\tau_0}{\dot{\gamma}}, \qquad \dot{\gamma} > \frac{\tau_{\mathtt{int}}}{\mu_0}$$
$$\mu = \mu_0, \qquad 0 < \dot{\gamma} < \frac{\tau_{\mathtt{int}}}{\mu_0}$$

where:

- K is the consistency index.
- $\dot{\gamma}$ is the shear rate.
- n is the power law index.

- τ_0 is the yield stress.
- τ_{int} is the stress value at the intersection point A shown in the following figure.
- μ_0 is the yield viscosity or plastic viscosity.



Herschel-Bulkley model for shear rate

To smooth convergence, a fluid defined with Herschel-Bulkley model, behaves like a Newtonian fluid until the local shear stress reaches intersect (A), shown as the red curve. The yield viscosity, μ_0, is a mathematical artefact introduced to improve convergence of the Herschel-Bulkley model when the stress is less than the yield stress. The blue curve shows the non-Newtonian Herschel-Bulkley behavior after intersect (A).

11.4 Carreau model

The fluid viscosity, μ, of a Carreau fluid is defined in the flow solver using the following equation:

$$\mu = \mu_\infty + (\mu_0 - \mu_\infty)\left[1 + \left(\lambda\dot{\gamma}e^{T_0/T}\right)^2\right]^{\left(\frac{n-1}{2}\right)}$$

where:

- λ is the time constant.
- $\dot{\gamma}$ is the shear rate.
- n is the power law index.
- T_0 is the reference temperature.
- T is the fluid temperature.
- μ_∞ is the infinite-shear viscosity.
- μ_0 is the zero-shear viscosity.

12 Modeling particle tracking

12.1 Modeling of the particle motion

The flow solver uses a one-way coupling between the flow field and the injected particles. Thus, the flow field is not affected by the particle transport problem.

The particle transport model, is defined by:

- The equations of motion, arising from the statement of the conservation of particle momentum.
- The initial conditions to be imposed on a newly-injected particle.

The equations of the particle motion are integrated in time to determine the particle centroid trajectory, $\vec{\mathrm{x}}_p(t)$.

The motion of a spherical particle is caused by the force $\vec{F}_{fp}$ acting on a single particle. Thus, the basic form of the system of governing equations for the particle trajectory is a set of ordinary differential equations for position and velocity in time:

$$\begin{cases} \rho_p V_p \dfrac{d\vec{U}_p}{dt} = \vec{F}_{fp} \\ \dfrac{d\vec{\mathrm{x}}_\mathrm{p}}{dt} = \vec{U}_p \end{cases}$$

where:

- ρ_p is the particle density.
- V_p is the volume of a particle.
- $\vec{U}_p$ is the particle velocity vector.
- $\vec{F}_{fp}$ is the fluid/particle interaction force, decomposed into four portions as:

$$\vec{F}_{fp} = \vec{F}_{fp}^{D} + \vec{F}_{fp}^{B} + \Delta\vec{F}_{fp} + \Delta\vec{F}'_{fp}$$

- $\vec{F}_{fp}^{D}$ is the drag force, which is exerted upon a spherical particle traveling at a constant velocity $\vec{U}_p$ in a flow with constant mean freestream velocity.

See Drag force for more information.

- $\vec{F}_{fp}^{B}$ is the buoyancy force due to gravity.
  See Buoyancy for more information.
- $\Delta\vec{F}_{fp}$ is a set of non-drag forces (such as added mass force and pressure gradient), which accounts for the particle perturbations of flow field.
  See Non-drag forces for more information.

- $\Delta \overrightarrow{F'_{fp}}$ is a set of other forces, which models the impact upon the particle transport problem of both the unresolved turbulent fluctuations of the flow field, and the chaotic motions due to sub-continuum scale phenomena.
  See Brownian and turbulent diffusion for more information.

The governing equations are integrated to obtain the particle trajectory over the range of times $t \in [t_0^p, t_f^p]$, where $t_0^p \geq 0$ is the injection time for the particle of interest, and the final particle time t_f^p is the lesser of the time, at which the particle exits the domain and the total simulation time for the particle transport problem, t_{tot} .

Note

The flow solver uses the following assumptions for particle tracking:
- Spherical particles
- No particle-particle iteration
- No flow field alteration due to particle motion

12.2 Drag force

The drag force on the particle is defined follows:

$$\overrightarrow{F_{fp}^D} = \frac{\pi d^2}{8} C_D(Re_l)\rho_f \left[\mid \vec{U}_\infty - \vec{U}_p \mid (\vec{U}_\infty - \vec{U}_p) \right] \Big|_{\mathrm{x}=\mathrm{x}_p(t)}$$

with local Reynolds number:

$$Re_l = \left(\frac{\rho_f \mid \vec{U}_\infty - \vec{U}_p \mid d)}{\mu_f} \right) \Bigg|_{\mathrm{x}=\mathrm{x}_p(t)}$$

where $\vec{U}_\infty$ is the local fluid velocity field in the absence of any perturbation due to the presence or motion of the particle.

The flow solver uses the following drag coefficient correlation, by default:

$$C_D(Re_l) = \frac{24}{Re_l} + \frac{4}{\sqrt{Re_l}} + 0.4$$

You can specify a fixed drag coefficient C_D^0 . In this case, the flow solver uses it as:

$$C_D(Re_l) = C_D^0$$

12.3 Buoyancy

The buoyancy force acting on a spherical particle immersed in a fluid is given by:

$$\overrightarrow{F_{fp}^{B}} = V_p(\rho_p - \rho_f)\vec{g}$$

where:

- $V_p = \pi d^3/6$ is the volume of a particle of diameter, d.
- ρ_p is the particle density.
- ρ_f is the fluid density.

12.4 Non-drag forces

The non-drag force $\Delta\vec{F}_{fp}$ consists of:

$$\Delta\vec{F}_{fp} = \vec{F}_p + \vec{F}_A$$

where:

- $\vec{F}_p$ is the force applied on the particle due to the pressure gradient in the fluid surrounding the particle that is caused by fluid acceleration. It is defined as follow:

$$\vec{F}_p = \frac{3}{2}V_p\rho_f(\nabla P_\infty)\Big|_{\mathrm{x=x_p(t)}}$$

- $\vec{F}_A$ is added mass force to accelerate the virtual mass of the fluid in the volume occupied by the particle. This force arises due to the particle moving in an unsteady manner, which accelerates a certain amount of fluid surrounding it. According to Batchelor (1967) [28], this force depends on the orientation of the particle shape relative to the flow. The added mass coefficient C_A corresponds to a symmetric tensor whose components depend on the geometry of the particle. For a sphere, C_A is equal to 0.5.

$$\vec{F}_A = -\rho_f\Big|_{\mathrm{x=x_p(t)}} C_A V_p \frac{d\vec{U}_p}{dt}$$

Final expression for the non-drag force term becomes:

$$\Delta\vec{F}_{fp} = -\rho_f\Big|_{\mathrm{x=x_p(t)}}\left(\frac{1}{2}V_p\frac{d\vec{U}_p}{dt} + \frac{3}{2}V_p(\nabla P_\infty)\Big|_{\mathrm{x=x_p(t)}}\right)$$

12.5 Brownian and turbulent diffusion

The phenomenon of small scale chaotic motion of particles through a fluid, due to the sub-continuum interactions with individual fluid particles, is known as Brownian motion. In the case of a turbulent flow, chaotic motion of particles is also observed. It is due to the instantaneous forces arising from the turbulent, continuum-scale fluctuations of the velocity and pressure fields. This chaotic motion is estimated within the boundaries of an element during a particle time step.

A particle moves an average distance $|r|$ in Δt according to $< |r|^2 > \approx D_p \Delta t$, where D_p is the particle diffusivity:

$$D_p = D_T + D_B$$

- D_T is the turbulent diffusivity associated with the unresolved turbulent motions.
- D_B is the Brownian diffusivity associated with the sub-continuum effects.

The turbulent diffusivity is computed based on the turbulent eddy diffusivity, as follows:

$$D_T = \mu_t / \rho_f$$

The Brownian diffusivity is given by:

$$D_B = \frac{(k_B T)}{3\pi\mu_f d}$$

In these equations:

- μ_t is the eddy viscosity based on the turbulent model.
- ρ_f is the fluid density.
- μ_f is the fluid viscosity.
- k_B is the Boltzmann constant.
- T is the fluid temperature.
- d is the diameter of the particle.

Therefore the perturbations of velocity field due to Brownian and turbulent forces is estimated as:

$$|\Delta \vec{U}| = \frac{\sqrt{< |r^2| >}}{2\Delta t}$$

The presence of the particle inside the element is estimated as:

$$\Delta t = \frac{|\vec{U}_\infty(\mathrm{x}_p(t), t)|}{V_e^{1/3}}$$

Substituting expressions for Brownian and turbulent diffusivities along with the expression for the perturbation of flow field, into the expression for the drag force, $\Delta\vec{F'_{fp}}$ becomes:

$$\Delta\vec{F'_{fp}} = \frac{\pi d^2}{16} C_D(Re_l)\Big([\rho_f \mid \vec{U}_\infty - \vec{U}_p \mid]\Big|_{x=x_p(t)}\Big)\left(\frac{|\vec{U}_\infty(x_p(t),t)|}{V_e^{1/3}}\left[\frac{k_B T_\infty}{3\pi\mu_f d} + \frac{\mu_t}{\rho_f}\right]\right)^{1/2} \Delta\hat{N}$$

where the unit vector $\Delta\hat{N}$ that represents the direction of the velocity perturbation is generated randomly, and V_e is the element volume of the mesh.

By default, the flow solver computes the Brownian and turbulent diffusivity term $\Delta\vec{F'_{fp}}$. When you deactivated this term, the flow solver uses $\Delta\vec{F'_{fp}} = 0$. Usually, you neglect this term when you apply the particle slip correction. For more information, see Particle slip correction factor.

12.6 Particle motion equations

The previous pages describe in detail the model employed to determine particle trajectories in the flow solver and the final formulation is given by:

$$\begin{cases} \rho_p V_p \dfrac{d\vec{U}_p}{dt} = \vec{F^D_{fp}} + \vec{F^B_{fp}} + \Delta\vec{F}_{fp} + \Delta\vec{F'_{fp}} \\ \dfrac{d\vec{x}_p}{dt} = \vec{U}_p \end{cases}$$

$$\vec{F^D_{fp}} = \frac{\pi d^2}{8} C_D(Re_l)[\rho_f \mid \vec{U}_\infty - \vec{U}_p \mid (\vec{U}_\infty - \vec{U}_p)]\Big|_{x=x_p(t)}$$

$$\vec{F^B_{fp}} = \frac{\pi d^3}{6}(\rho_p - \rho_f)\vec{g}$$

$$\Delta\vec{F}_{fp} = -\rho_f\Big|_{x=x_p(t)}\left(\frac{1}{2}V_p\frac{d\vec{U}_p}{dt} + \frac{3}{2}V_p(\nabla P_\infty)\Big|_{x=x_p(t)}\right)$$

$$\Delta\vec{F'_{fp}} = \frac{\pi d^2}{16} C_D(Re_l)([\rho_f \mid \vec{U}_\infty - \vec{U}_p \mid]\Big|_{x=x_p(t)})\left(\frac{|\vec{U}_\infty(x_p(t),t)|}{V_e^{1/3}}\left[\frac{k_B T_\infty}{3\pi\mu_f d} + \frac{\mu_t}{\rho_f}\right]\right)^{1/2} \Delta\hat{N}$$

$$Re_l = (\rho_f \mid \vec{U}_\infty - \vec{U}_p \mid d/\mu_f)\Big|_{x=x_p(t)}$$

Particle motion equations

- The unit vector $\Delta\hat{N}$ is randomly generated.
- The drag coefficient C_D , see Drag force for details.

- The velocity field $\vec{U}_\infty$ and modified pressure field P_∞ are the fluid velocity and pressure fields found by the flow solver.
- The equations for the initial position $\vec{\mathrm{x}}_\mathrm{p}(t_o^p) = \vec{\mathrm{x}}_\mathrm{p}^0$ and velocity $\vec{U}_p(t_o^p) = \vec{U}_p^0$ of the particle are described in Initial conditions for injected particles.
- For the particle motion assumptions, see Modeling of the particle motion.

12.7 Particle slip correction factor

The flow solver solves the particle motion equations for a rigid sphere by imposing a no-slip assumption at the particle surface. This assumption breaks when the Knudsen number $Kn = \frac{2\lambda}{d}$ is high. This occurs when the particle size, given by its diameter d is the same order as the mean free path λ of the particle. Since the particle diameter d is usually fixed, it is the mean free path λ of the particle that dictates whether this assumption is valid or not.

When the no-slip assumption is no longer valid, the flow solver corrects the particle traction force by dividing it by the Cunningham correction factor C [34]:

$$C = 1 + Kn\left[\alpha + \beta\exp\left(-\frac{\gamma}{Kn}\right)\right]$$

where α, β, and γ are experimentally determined constants that are equal to 1.165, 0.483, and 0.997, respectively for ideal gases. The only unknown in the previous equation is the mean free path of the particle in the Knudsen number. The mean free path of the particle can be determined by evaluating the equation:

$$\lambda = \lambda_0\left(\frac{T}{T_0}\right)\left(\frac{P}{P_0}\right)\left(\frac{1+\frac{S}{T_0}}{1+\frac{S}{T}}\right)$$

where:

- T is the absolute temperature of the ideal gas.
- P is the absolute pressure of the ideal gas.
- S is the specified Sutherland constant for the viscosity of the gas.
- T_0 is the specified reference temperature of the gas.
- P_0 is the specified reference pressure of the gas.
- λ_0 is the reference mean free path of the gas at temperature T_0 and pressure P_0, given by the following equation:

  $$\lambda_0 = \frac{\mu_0}{0.491\rho_0}\sqrt{\frac{\pi R}{8k_B T_0 \eta R_g}}$$

- μ_0 is the reference viscosity of the ideal gas.
- ρ_0 is the reference density of the ideal gas, which is calculated using ideal gas law.
- k_B is the Boltzmann constant.
- η is the Avogadro's number.
- R is the universal gas constant.
- R_g is the specific gas constant, which is obtained from the gas material properties.

The flow solver applies this correction factor to the net traction force acting on the particle surface in the following two ways:

The equations of motion for the particle trajectory is the corrected form given by:

$$\begin{cases} \rho_p V_p \dfrac{d\vec{U}_p}{dt} = \rho_p V_p \vec{g} + \dfrac{\vec{F}_{tr}}{C} \\ \dfrac{d\vec{\mathrm{x}}_{\mathrm{p}}}{dt} = \vec{U}_p \end{cases}$$

Where F_{tr} is the uncorrected traction force $\vec{F}_{tr} = \vec{F}^D_{fp} + \vec{F}^B_{fp} + \Delta\vec{F}_{fp} + \Delta\vec{F'}_{fp} - \rho_p \vec{U}_p \vec{g}$. For detailed equations of these forces, see Particle motion equations.

12.8 Initial conditions for injected particles

In the flow solver, any portion $\delta\Omega_i$ of the piecewise-defined domain boundary may be selected as an injection surface. You must specify the initial conditions $\vec{\mathrm{x}}_p(t^p_0) = \vec{\mathrm{x}}^0_p$ and $\vec{U}_p(t^p_0) = \vec{U}^0_p$ imposed at the injection time t^p_0 for the particle of interest, from which the particle motion equations are integrated. For each particle injected at a given injection surface, a randomly chosen initial location $\vec{\mathrm{x}}^0_p \in \delta\Omega_i$ is imposed. The initial velocity $\vec{U}^0_p$ may either be specified by the user, or be set equal to the local fluid velocity, that is $\vec{U}^0_p = \vec{U}_\infty(\vec{\mathrm{x}}^0_p, t^p_0)$.

In the flow solver, the injection times belong to the range $t^p_0 \in [0, t_{inj}], t_{inj} \le t_{tot}$. In a transient simulation, the total injection time t_{inj} , and the total time t_{tot} , are equal to the total simulation duration for the fluid flow problem. For a steady state simulation, t_{inj} and t_{tot} are specified by the user, and for the purposes of the particle transport problem, the steady state values of the flow field are used at any time t . Denoting by t_i a defined output time for the particle tracking problem, the expected value N^i_{inj} the total number of particles to be injected over the interval $t^p_0 \in [t_i, t_{i+1}]$ is given for a specified injection rate $\dot{N}$ of particles per unit time, as:

$$N^i_{inj} = \dot{N}(t_{i+1} - t_i)$$

Note

Regardless of the method in the flow solver by which the injection rate is specified, it is always transformed by flow solver into a value $\dot{N}$ (particles per unit time). Then, the number of injections over the interval is determined from a rounding of N_{inj} to an integer value. The injection times t^p_0 for each particle to be injected are then randomly distributed on $t_0 \in [t_i, t_{i+1}]$.

12.9 Boundary treatment

Special consideration must be taken when the particle reaches the following boundary types of the flow domain:

- Wall boundaries
- Inlet boundaries

- Symmetry boundaries
- Periodic boundaries

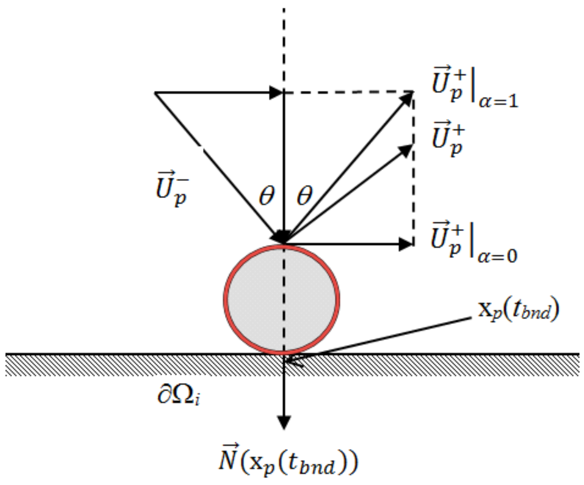
The boundary $\delta\Omega$ of the flow domain is represented by the union of a set of surfaces $\delta\Omega_i$, with a consistent boundary condition applied over each surface $\delta\Omega_i$. The boundary represents a discontinuity in the equations of a particle motion at time t_{bnd} and a position of particle $\overrightarrow{\mathrm{x}_\mathrm{p}}(t_{bnd}) \in \delta\Omega_i$. Therefore special boundary treatment is required. In the flow solver, the forces arising at boundaries are treated as impulses over a vanishingly small time, so that the particle velocities are modified discontinuously.

Note

At boundary surfaces $\delta\Omega_i$ over which particles exit the domain (an outlet or opening boundary), the particles exit smoothly, and no further modeling is required.

12.10 Wall boundaries

The flow solver treats both slip or no-slip wall boundaries identically in the particle transport problem. Figure below depicts the case when the particle, shown in red, reaches a wall boundary $\delta\Omega_i$. In the flow solver, the impact, denoted by $\overrightarrow{\mathrm{x}_\mathrm{p}}(t_{bnd})$, is approximated as an instantaneous event at time t_{bnd} .



Denote the particle approach velocity predicted by the integration of the particle motion equations and the departure velocity after application of the boundary treatment, respectively as:

$$\vec{U}_p^- = \lim_{t \to t_{bnd-}} (\vec{U}_p(t))$$
$$\vec{U}_p^+ = \lim_{t \to t_{bnd+}} (\vec{U}_p(t))$$

The flow solver applies different boundary treatments depending on the particle impact type:

1. At walls with particle impact type set to stick, when the particle reaches the wall boundary, the wall-normal component of particle velocity $\vec{U}_p^+ = 0$ regardless of $\vec{U}_p^-$ and the particle continues to be counted towards the statistics, so that an accumulation of particle density at the wall is captured.
2. At walls with particle impact type set to rebound, when the particle reaches the wall boundary, the wall-normal component of particle velocity is modified as follows, using the collision model with the coefficient of restitution $\alpha \in [0, 1]$ [32]:

$$\vec{U}_p^+ = \vec{U}_p^- + \Delta\vec{U}_p$$
$$\Delta\vec{U}_p = -(1+\alpha)(\vec{U}_p \cdot \vec{N}(\vec{\mathrm{x}}_p(t_{bnd})))\vec{N}(\vec{\mathrm{x}}_p(t_{bnd}))$$

where $\vec{N}(\mathrm{x})$ is the outward unit normal vector to the boundary surface for all $\mathrm{x} \in \delta\Omega_i$.

The coefficient of restitution is a measure of the portion of the energy associated with the wall normal component of the approach velocity, which is retained after the collision. For a purely elastic collision, $\alpha = 1$ and the particle is reflected off of the wall with no loss of energy. When $\alpha = 0$, the particle loses all wall-normal momentum during the collision.

The integration of the particle motion equations is then continued from the predicted location $\vec{\mathrm{x}}_p(t_{bnd})$, with the imposed velocity $\vec{U}_p(t_{bnd}) = \vec{U}_p^+$.

The flow solver continues to track particles that stop at walls and considers these particles in the output statistics.

12.11 Inlet and symmetry boundaries

The flow solver does not allow particles to exit the flow domain across an inlet or symmetry boundary. This condition, similar to the wall boundary treatment with an assumption that the normal particle velocity component is removed. The departure velocity is given by:

$$\Delta\vec{U}_p^+ = \vec{U}_p^- - (\vec{U}_p^- \cdot \vec{N}(\vec{\mathrm{x}}_p(t_{bnd})))\vec{N}(\vec{\mathrm{x}}_p(t_{bnd}))$$

The integration of the particle motion equations is then continued from the modified state defined by the predicted location $\vec{\mathrm{x}}_p(t_{bnd})$ with the imposed velocity $\vec{U}_p(t_{bnd}) = \vec{U}_p^+$.

12.12 Periodic boundaries for particle tracking

In the flow solver, a particle that reaches a periodic boundary is presumed to pass through to the corresponding point on the matching periodic face. If $\vec{\mathrm{x}}_{per}(\vec{\mathrm{x}})$ is the location across the periodic boundary for any $\vec{\mathrm{x}} \in \delta\Omega_i$, then for the approach point obtained from integration of the particle motion equations, denoted by $\vec{\mathrm{x}}_p^-$, the departure point $\vec{\mathrm{x}}_p^+$ is given by:

$$\vec{x}_p^+ = \lim_{t \to t_{bnd^+}} (\vec{x}_p(t)) = \vec{x}_{per}(\vec{x}_p^-)$$
$$\vec{x}_p^- = \lim_{t \to t_{bnd^-}} (\vec{x}_p(t))$$

In addition, at a rotationally periodic boundary defined by a rotation of an angle Θ, about an axis directed along the unit vector ω and with the center of rotation x_o, the departure velocity is obtained by a rotation imposing conservation of the cylindrical components of the velocity vector across the periodic boundary, according to:

$$\vec{U}_p^+ = \mathbf{\Lambda} \cdot \vec{U}_p^-$$
$$\Lambda_{ij} = (1 - \cos(\Theta))\omega_i \omega_j + \cos(\Omega)\delta_{ij} + \sin(\Theta)\varepsilon_{ijk}\omega_k$$

where δ_{ij} is the Kronecker delta tensor and ε_{ijk} is the permutation tensor. In the case of a translational periodicity condition, the following condition is imposed:

$$\vec{U}_p^+ = \vec{U}_p^-$$

The integration of the particle motion equations is then continued from the modified state defined by the imposed location $\vec{x}_p(t_{bnd}) = \vec{x}_p^+$ with the imposed velocity $\vec{U}_p(t_{bnd}) = \vec{U}_p^+$.

13 Modeling fluid structure interaction

Fluid structure interaction (FSI) computations account for motion or deformation of the solid structure boundaries that are interacting with the adjacent fluid flow.

13.1 Transient fluid structure interaction

The flow solver uses the arbitrary-Lagrange-Eulerian (ALE) form to discretize the governing equations for FSI modeling. Integration of the governing Navier-Stokes equations in the ALE form over a moving control volume $V(t)$ [57] is given as:

$$\frac{\partial}{\partial t}\int_{V(t)} \mathbf{U}dV + \int_{\partial V(t)} \left(\mathbf{F}(\mathbf{U}) - \dot{\mathbf{x}}\mathbf{U}\right). \vec{N}dS + \int_{\partial V(t)} \mathbf{G}(\mathbf{U}). \vec{N}dS = 0$$

where:

- $\mathbf{U}$ is a conserved quantity (mass, momentum and energy).
- $\mathbf{F}$ is the convective flux term.
- $\mathbf{G}$ is the viscous flux term.
- $\partial V(t)$ is the control volume boundary.
- S is the control volume boundary surface.
- $\dot{\mathbf{x}}$ is the velocity of the control volume boundary.
- $\vec{N}$ is the normal vector to the control volume boundary.
- t is the time.

The discrete form of the governing equations in the ALE form is given as:

$$\frac{\partial}{\partial t}(V\mathbf{U}) + \mathbf{R}\left(\mathbf{U}, \mathbf{x}(t), \dot{\mathbf{x}}(t), \vec{N}(t)\right) + \mathbf{T}\left(\mathbf{U}, \mathbf{x}(t), \vec{N}(t)\right) = 0$$

where the discrete convective flux in the ALE form is:

$$\mathbf{R}\left(\mathbf{U}, \mathbf{x}(t), \dot{\mathbf{x}}(t), \vec{N}(t)\right) = \int_{\partial V(t)} \left(\mathbf{F}(\mathbf{U}) - \dot{\mathbf{x}}\mathbf{U}\right). \vec{N}dS$$

and the discrete viscous flux is:

$$\mathbf{T}\left(\mathbf{U}, \mathbf{x}(t), \vec{N}(t)\right) = \int_{\partial V(t)} \mathbf{G}(\mathbf{U}). \vec{N}dS$$

and $\mathbf{x}(t)$ are the control volume boundary positions.

The following source term is added to the right hand side of the discrete form of the governing equation, to satisfy the geometric conservation law, by assuming that the state of $\mathbf{U}$ is an exact constant solution for the governing equation.

$$\frac{\partial V}{\partial t} - \mathbf{R}\Big(\mathbf{x}(t), \dot{\mathbf{x}}(t)\Big)$$

13.2 Static fluid structure interaction

The flow solver uses the following discrete form of ordinary differential equation (ODE) to compute FSI modeling with static mesh.

$$V\frac{\partial}{\partial t}(\mathbf{U}) + \mathbf{R}\Big(\mathbf{U}, \mathbf{x}(t), \dot{N}(t)\Big) + \mathbf{T}\Big(\mathbf{U}, \mathbf{x}(t)\Big) = 0$$

where:

- $\mathbf{U}$ is a conserved quantity (mass, momentum and energy).
- $\dot{N}(t)$ is the grid face velocity.
- V is the control volume.

The grid face velocity is defined as:

$$\dot{N}(t) = \dot{\mathbf{x}}(t).\,\vec{N}$$

where $\vec{N}$ is the surface normal.

In static FSI modeling, the velocity of the control volume boundary, $\dot{\mathbf{x}}$, is zero and the control volume depends on the control volume grid coordinates, $\mathbf{x}(t)$ as:

$$V = V\Big(\mathbf{x}(t)\Big)$$

Knowing the grid coordinate positions and velocities as a function of time, for a first order time accurate scheme, the flow solver uses the following discretization of the ODE for the static FSI modeling:

$$V^{n+1}\Big(\mathbf{U}^{n+1} - \mathbf{U}^{n}\Big) = \Delta t\Big(- \mathbf{R}\Big(\mathbf{U}^{n+1}, \mathbf{x}^{n+1}, \vec{N}^{n+1}\Big) - \mathbf{T}\Big(\mathbf{U}^{n+1}, x^{n+1}\Big)\Big)$$

where $\Delta t = t^{n+1} - t^{n}$ is the current time step.

14 Discretization

14.1 Control-volume method

The flow solver uses a finite-element based control-volume method, [5], for discretization of the governing equations. With this method, the governing equations are integrated over a control volume and over a time step. For example, the integration for the conservation equation of the passive component is:

$$\frac{\partial}{\partial t}\int_V \rho\phi dV + \int_V \frac{\partial(\rho U_j\phi)}{\partial x_j}dV = \int_V \frac{\partial}{\partial x_j}\left(\rho D_\phi \frac{\partial\phi}{\partial x_j} - \overline{\rho u_j\phi'}\right)dV + \int_V S_\phi dV$$

- V is the control volume.
- δt is the time step.

Using Gauss's theorem to convert a volume integral to a surface integral, it becomes:

$$\frac{\partial}{\partial t}\int_V \rho\phi dV + \oint_A \rho U_j n_j \phi dA = \int_V \frac{\partial}{\partial x_j}\left(\rho D_\phi \frac{\partial\phi}{\partial x_j} - \overline{\rho u_j\phi'}\right)dV + \int_V S_\phi dV$$

- n_j is the unit outward surface normal of the control volume surface.
- A is the outer surface area of the control volume.

The flow solver approximates numerically the volume and surface integrations over a discrete finite volume defined on a computational grid or mesh. For instance, the advection term is approximated as:

$$\oint_A \rho U_j n_j \phi dA \approx \sum_{ip} \dot{m}_{ip}\phi_{ip}$$

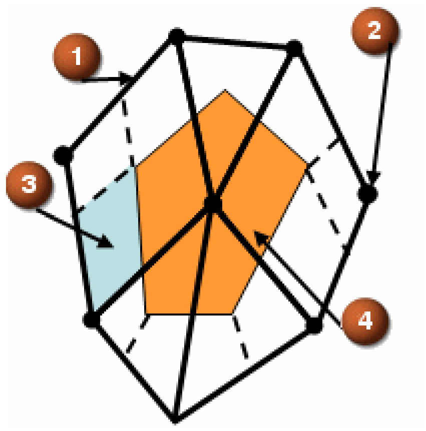
where the discrete mass flow through a finite sub-surface of the finite volume is defined as follows:

$$\dot{m}_{ip} = \rho U_j n_j \Delta A$$

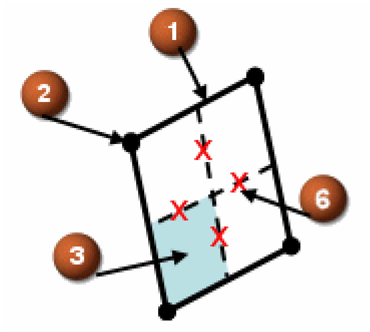
In these equations:

- ip denotes the integration point of the sub-surface.
- ΔA is the sub-surface area.
- ϕ is the transport variable.

The following figure illustrates the finite volume (4) defined from elements (1) in the element-based finite volume method, [5]. All the dependent variables, including the pressure and the velocity components, are stored at the element nodes (2). This is a co-located method. Each finite volume sub-surface is an element bi-sector plane (3). In this example, a complete finite volume results from two surrounding quadrilateral elements and three triangular elements. These sub-surfaces are called integration point surfaces. The software computes the integral at their mid-points (6). This method of finite volume definition extends directly to 3D.



The following figure depicts an isolated element with the integration point surfaces and the finite volume sectors.



The discretization scheme approximates the volume integrals over the element sectors and the surface integrals on the element integration point surfaces. For each element, the flow solver makes the discrete approximations to the terms in the integrals and combines them in the overall equation assembly.

14.2 Spatial discretization

14.2.1 Governing equations discretization

The mass conservation equation gives:

$$V\left(\frac{\rho-\rho^o}{\Delta t}\right)+\sum_{ip}\left(\rho U_j\Delta n_j\right)_{ip}=0$$

where:

- The superscript ^{o} refers to the value at the previous time step.
- $\Delta n_j = n_j \Delta A$.

For gases, the transient term is expressed as:

$$V\left(\frac{\rho-\rho^o}{\Delta t}\right)=\frac{V}{\Delta t}\left(\frac{P}{RT}-\frac{P^o}{RT^o}\right)$$

where R is the ideal gas constant.

The velocities at the integration point, $U_{j,ip}$ are evaluated from momentum equations for the integration point velocities [5]. The resulting form of the mass conservation equation contains a pressure redistribution term, which resolves the pressure-velocity coupling problem that is typical of colocated methods [6].

The discretized form of the momentum conservation equation for velocity U_j, is defined as follows:

$$V\left(\frac{\rho U_j-\rho^o U_j^o}{\Delta t}\right)+\sum_{ip}\dot{m}_{ip}U_{j,ip}=-\sum_{ip}\left(P\Delta n_j\right)_{ip}+\sum_{ip}\left(\mu_{eff}\left(\frac{\partial U_j}{\partial x_i}+\frac{\partial U_i}{\partial x_j}\right)\Delta n_j\right)_{ip}+S_{U_j}V$$

where $\mu_{eff} = \mu + \mu_t$.

Similarly, the discretized conservation equations for energy is given by:

$$V\left(\frac{\rho h-\rho^o h^o}{\Delta t}\right)+\sum_{ip}\dot{m}_{ip}h_{ip}=\sum_{ip}\left(k_{eff}\left(\frac{\partial h}{\partial x_j}\right)\Delta n_j\right)_{ip}+S_h V$$

The equation for the passive component is given by:

$$V\left(\frac{\rho\phi-\rho^o\phi^o}{\Delta t}\right)+\sum_{ip}\dot{m}_{ip}\phi_{ip}=\sum_{ip}\rho D_{\phi,eff}\left(\frac{\partial\phi}{\partial x_j}\Delta n_j\right)_{ip}+S_\phi V$$

where:

- $k_{eff} = k + \mu_t/Pr_t$.
- $\rho D_{\phi,eff} = \rho D_\phi + \mu_t/\rho Sc_t$.

The discretized conservation equations for water vapor and turbulence quantities can be written similarly. The flow solver treats all these equations similarly as they all have similar form.

The terms in these equations are:

- Transient
- Convective
- Effective diffusion
- Source

14.2.2 Diffusion terms

The general form of the diffusion term is:

$$\sum_{ip} \left(\rho D_{eff} \left(\frac{\partial \phi}{\partial x_j} \right) \Delta n_j \right)_{ip}$$

The derivatives at an integration point ip are evaluated using shape functions.

Given the (s , t , u) coordinates of the nodes within an element, the value of the field variable, ϕ , is calculated using the shape functions, as follows:

$$\phi(s,t,u) = \sum_{node=1}^{NNODE} N^{ele}(s,t,u)_{node} \phi_{node}$$

where $NNODE$ is the number of nodes in the element.

The flow solver determines the shape functions for an element, indentifying the (s , t , u) coordinates of the element nodes. An appropriate polynomial in (s , t , u) coordinates and that the shape function corresponding to a particular node is unity at that node and zero at all other element nodes. The derivatives of the field variables are then evaluated using ϕ nodal values and derivatives of the shape functions as follows:

$$\left.\frac{\partial \phi}{\partial x}\right|_{ip} = \sum_{node=1}^{NNODE} \left.\frac{\partial N_{node}}{\partial x}\right|_{ip} \phi_{node}$$

14.2.3 Pressure term

The flow solver calculates the pressure term, $\sum_{ip}(P\Delta n_j)_{ip}$, using shape functions. For more information, see mass conservation equation discretization. The pressure at the integration point ip is defined as:

$$P_{ip} = \sum_{node=1}^{NNODE} N_{ip} P_{node}$$

where P_{node} are the nodal pressure values.

14.2.4 Source terms

The flow solver computes the discretized source terms by multiplying S_{uj} , S_h and S by the volume of the control volume.

14.2.5 Convective terms

In the discretized form of the passive component equation, the term $\sum_{ip} \dot{m}_{ip}\phi_{ip}$ represents the summation of the convective or advective fluxes across the boundaries of a given control volume. This quantity is evaluated using nodal values with first-order or higher-order schemes.

The values ϕ_{ip} or $\phi_{i+1/2}$ from a Taylor's Series are evaluated as follows:

$$\phi_{ip} = \phi_{i+\frac{1}{2}} = \phi_i + \left.\frac{d\phi}{dx}\right|_i \Delta x + \left.\frac{d^2\phi}{dx^2}\right|_i \frac{\Delta x^2}{2!} + H.O.T.$$

where:

- H.O.T. stands for higher order terms.
- When $\phi_{i+1/2}$ is approximated only by the first term of the Taylor series, the truncation error is first order. In this case, the advection scheme is a first order scheme.
- When $\phi_{i+1/2}$ is approximated by the first two terms of the series, the truncation error is second order. The advection scheme is a second order scheme.

The flow solver uses the first order Upwind Differencing Scheme (UDS) by default. The following higher order schemes are also available:

- Quadratic Upwind Interpolation for Convective Kinematics (QUICK) scheme
- Second Order Upwind (SOU) scheme
- Second-Order Central Differencing (CDS) scheme
- Second-Order High Resolution (HI-RES) scheme
- Second-Order Monotonic Upstream-centered Scheme for Conservation Laws (MUSCL) scheme

Higher order schemes help to considerably reduce false diffusion but they still produce a truncation error. Their truncation error is dispersive and can lead to unphysical oscillations and numerical instability. Higher order schemes give more accurate solutions than UDS scheme but they are less stable, which can lead to oscillatory convergence. They generally require higher calculation time.

To eliminate the oscillations that are inherent to most higher order schemes, you can impose a bound on the convected face values. The bounds are imposed through flux limiters. Flux limiters limit the transport variable, ϕ, to ensure it lies between specified values.

First-order UDS scheme

The first-order Upwind Differencing Scheme (UDS) approximates ϕ_{ip} by the value of ϕ at the upstream node:

$$\phi_{ip} = \phi_i$$

With this scheme, the solution is very stable and converges quickly. The UDS generates accurate results only if there are no gradients in the flow fields. Otherwise, there can be false diffusion in the solution field, which is characterized by serious smearing of expected gradients [7]. To reduce the error introduced by false diffusion, it is recommended to use higher order schemes.

Second-order QUICK scheme

The Quadratic Upwind Interpolation for Convective Kinematics (QUICK) scheme approximates ϕ_{ip} with the higher-order values of the transport variable. It uses a two points upstream and one point downstream weighted quadratic interpolation of the transport variable.

$$\phi_{ip} = \frac{6}{8}\phi_{i-1} + \frac{3}{8}\phi_i - \frac{1}{8}\phi_{i-2}$$

This second-order scheme can produce a higher precision solution [41]. However, this scheme has the lowest stability in the regions with strong gradients and is most appropriate for steady flows.

Second-order SOU scheme

The Second-Order Upwind (SOU) scheme approximates ϕ_{ip} by adding the numerical advection correction $\left.\frac{d\phi}{dx}\right|_i \Delta x$ to the Upwind Differencing Scheme (UDS) equation to reduce the diffusive properties of the first-order UDS.

$$\phi_{ip} = \phi_{i+\frac{1}{2}} = \phi_i + \left.\frac{d\phi}{dx}\right|_i \Delta x$$

The numerical advection correction term is an approximation of the nodal gradient.

The SOU advection scheme provides better precision results than the first-order UDS advection scheme. The SOU scheme is less robust than UDS in regions with strong gradients but more stable than the QUICK scheme. When using this scheme, it is necessary to apply flux limiters to the predicted transport variable.

Second-order CDS scheme

The Central Differencing Scheme (CDS) approximates ϕ_{ip} as follows, with reference to the geometry depicted in the figure of the Shape function section:

$$\phi_{ip}^{CDS} = \sum_i \phi_{N_i^e} B_i^e(\mathbf{x}_{ip}) = \phi_{ip} + O(\delta^{N_B})$$

where:

- B_i^e is the shape function associated with the node N_i^e .
- δ is the local characteristic mesh spacing.
- $N_B \geq 2$ is the order of accuracy of the shape function.

When the CDS approximation has the generation of spurious oscillations in the solution field, it is implemented in the flow solver as a correction to the implicit UDS scheme, with the imposed flux limiter $\Psi_{ip} \in [0, 1]$, so that at iteration n of the solution the procedure is defined as:

$$\phi_{ip} \approx \left(\phi_{ip}^{UDS}\right)^n + \Psi_{ip}\left(\phi_{ip}^{CDS} - \phi_{ip}^{UDS}\right)^{n-1}$$

The CDS scheme provides a more accurate representation of the convective fluxes than the UDS scheme and is computationally less time-consuming than the QUICK and SOU second-order schemes.
The CDS advection scheme is appropriate for for low speed flows with small Peclet number.

Second-order HI-RES scheme

The second-order high resolution scheme (HI-RES) approximates ϕ_{ip} by the value of ϕ at the upstream node, plus a higher order correction ϕ' value:

$$\phi_{ip} = \phi_i + \phi_i'$$

Unlike other second-order schemes HI-RES does not require a limiter [42] . HI-RES uses a special numerical technique to calculate the limiter at each node and adapts the discretization to avoid any unwanted unboundedness.

Second-order MUSCL scheme

The second order Monotonic Upstream-centered Scheme for Conservation Laws (MUSCL) approximates the value of ϕ_{ip} with the intercell flux field value $\phi_{i+\frac{1}{2}}$, which is computed by the average of fluxes at the current and next spatial locations:

$$\phi_{ip} = \phi_{i+\frac{1}{2}} = \phi_i + \frac{1}{2}\Psi(r)(\frac{\phi_{i+1} - \phi_i}{\Delta x})$$

where:

- ϕ_i is the field value at the upwind node i relative to the integration point ip.
- ϕ_{i+1} is the field value at the downwind node $i + 1$ relative to the integration point ip.
- $\Psi(r)$ is the specified flux limiter function. For more information, see Flux limiters.
- r is the ratio of the upwind-side gradient to downwind-side gradient. It is defined as:

$$\frac{\phi_i - \phi_{i-1}}{\phi_{i+1} - \phi_i}$$

This advection scheme provides better precision and convergence for simulation with high aspect ratio grids and meshes. The flow solver applies a flux limiter to the predicted transport variable to avoid inaccuracy related to spurious oscillations.

Flux limiters

The concept of flux limiters is similar to Total Variation Diminishing (TVD) schemes that are used in gas dynamics [14]. A limited higher order estimate for the value ϕ_{ip} at an integration point whose upwind and downwind neighbors are nodes C and D, respectively, is expressed according to the following equation:

$$\phi_{ip} = \phi_C + \beta_{C,ip}(\phi_{ip,HO} - \phi_C)$$

where:

- The subscript HO denotes the unlimited higher-order approximation.
- $\beta_{C,ip}$ is the limiter $0 < \beta_{C,ip} < 1$.

The limiter is a blending between the UDS scheme and the higher order scheme. The flow solver selects the limiter as the largest possible value within the solution, which prevents the development of undesirable artifacts, such as unbounded values or spurious oscillations.

SOU, QUICK, and CDS limiters

The limiters used for the SOU, QUICK, and CDS schemes are based on the *convective boundedness criterion* [15]. The integration point value of ϕ is limited as a requirement of the convective boundedness criterion by:

$$\phi_C < \phi_{ip} < \phi_D$$

The limiter is determined using the following equation:

$$\beta_{C,ip} = (\phi_{ip} - \phi_C)/(\phi_{ip,HO} - \phi_C)$$

where ϕ_{ip} is the closest value to $\phi_{ip,HO}$.

When the high-resolution advection scheme is selected, the unlimited second-order numerical advection correction approximation is employed, which is based on the cell-centered gradient, $\nabla\phi_C$, at the upwind node:

$$\phi_{ip,NAC} = \phi_C + (x_{ip} - x_C)\nabla\phi_C$$

In this case, the limiter is determined according to the following equations [27]:

$$\begin{aligned}\beta_{C,ip} &= max_{ip}(\psi_{C,ip})\\ \psi_{C,ip} &= (\Delta_{ext}(\Delta_{ext} + 2\Delta_{ip}) + K\delta^3)/(\Delta_{ext}^2 + 2\Delta_{ip}^2 + \Delta_{ext}\Delta_{ip} \pm K\delta^3)\\ \Delta_{ext} &= \phi_{ext} - \phi_C\\ \Delta_{ip} &= \phi_{ip,NAC} - \phi_C\end{aligned}$$

where:

- δ is the average mesh spacing.
- ϕ_{ext} is an extremal value over the stencil of the node C. Stencil is the geometric arrangement of a nodal group that relate to the node C.
- The factor K is a blending of the limited and unlimited solutions, and determines the magnitude of oscillations in the solution.

The maximization in the first equation is carried out over all integration surfaces bounding the control volume about node C.

MUSCL limiters

The Monotonic Upstream-centered Scheme for Conservation Laws (MUSCL) scheme uses the flux limiter function when approximating the value of ϕ_{ip} .

The flux limiter function, $\Psi(r)$ is based on the ratio, r [63], of the upwind-side gradient to downwind-side gradient, which is defined as:

$$r = \frac{\phi_i - \phi_{i-1}}{\phi_{i+1} - \phi_i}$$

The available limiter functions for the MUSCL scheme are:

- Van Albada [60]:

$$\Psi(r) = \frac{r^2 + r}{r^2 + 1}$$

- Minmod [61]:

$$\Psi(r) = \max\left[0, \min\left(1, r\right)\right]$$

- Superbee [61]:

$$\Psi(r) = \max\left[0, \min\left(2r, 1\right), \min\left(r, 2\right)\right]$$

- Van Leer [62]:

$$\Psi(r) = \frac{r + |r|}{1 + |r|}$$

The flux limiter functions exhibit second-order TVD behavior. They are designed to traverse a specific region of the solution, known as the TVD region, to ensure the stability of the scheme. Second-order TVD limiters adhere to the following criteria:

$$r \leq \Psi(r) \leq 2r, (0 \leq r \leq 1)$$

$$1 \leq \Psi(r) \leq r, (1 \leq r \leq 2)$$

$$1 \leq \Psi(r) \leq 2, (r > 2)$$

$$\Psi(r)(1) = 1$$

14.2.6 Shape function

The flow solver computes solution variables, such as velocity components, pressure, and temperature at the mesh nodes. It then approximates the solution variables at arbitrary points in the computational domain with shape functions. The shape functions interpolate solution variables from mesh nodes to the arbitrary point with given nodal solution variables, and the coordinates of an arbitrary point in the computational domain.

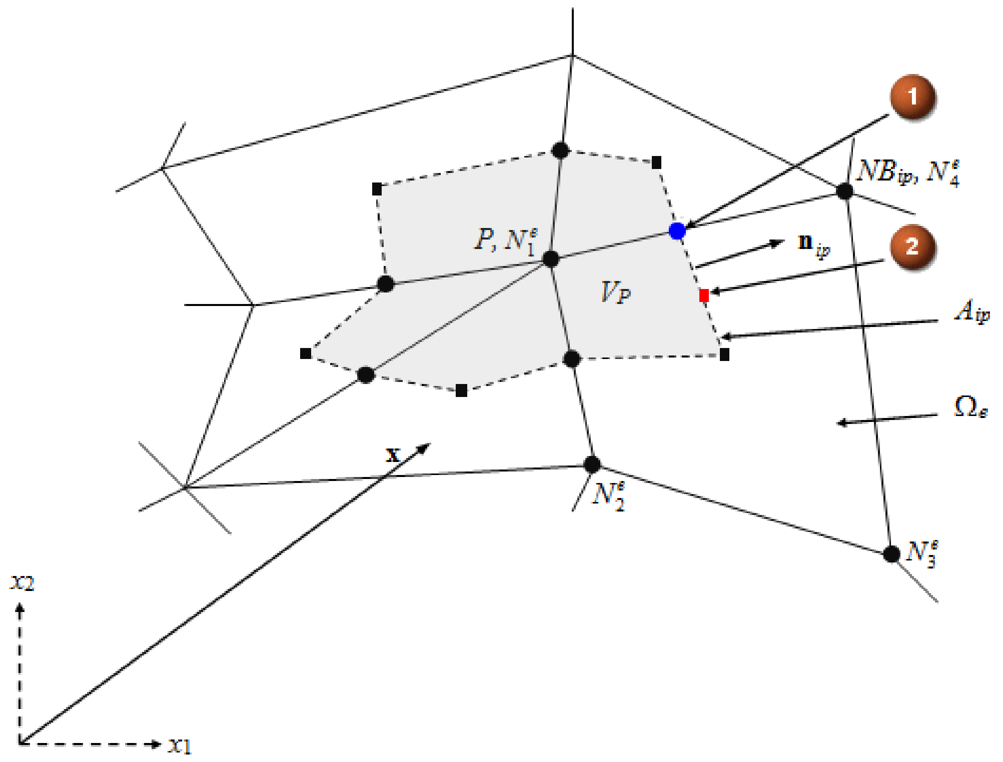
The shape function representation is:

$$\phi(x) = \sum_{i} \phi_i B_i(x), x \in \Omega_e$$

where:

- ϕ_i is the solution value at node N_i^e .
- B_i is the finite element shape function associated with the same node.

The flow solver uses two types linear-linear and tri-linear shape functions for its interpolation approximations. The following figure depicts the locations of the x_{ip} employed in conjunction with the expansion of the shape function for the quadrature of integration surface A_{ip} .



1 Location x_{ip} for linear-linear interpolation 2 Location x_{ip} for tri-linear interpolation

Control volume finite element discretization

The shape functions B_i always represent a tri-linear interpolation over element Ω_e to the arbitrary point x.

The linear-linear shape function imposes a selection for x_{ip} lying at the midpoint of the vector, which joins node P with its neighbor NB_{ip} across A_{ip}. The values of $\phi(x_{ip})$ and the directional derivative $\nabla\phi|_{x_{ip}} . \hat{s}$ along the vector $s = x_{NB_{ip}} - x_p = |s|\hat{s}$, are reduced to the following approximations:

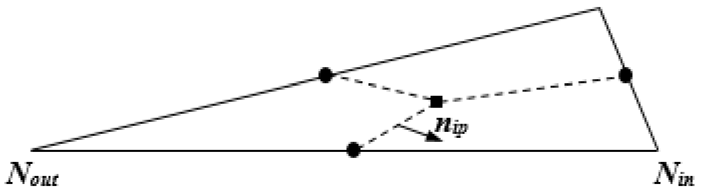
$$\phi(x_{ip}) = \frac{1}{2}\left(\phi_P + \phi_{NB_{ip}}\right)$$

$$\nabla\phi|_{x_{ip}} \cdot \hat{\boldsymbol{s}} = \frac{\phi_{NB_{ip}} - \phi_P}{|\boldsymbol{s}|}$$

By default, the flow solver is set to low resolution mode and uses linear-linear shape functions. When you choose high resolution mode, the flow solver uses tri-linear functions.

Skewed element treatment

The flow solver uses the finite element method to compute the gradient of the solution field $\nabla\phi$ for mesh elements. To improve the solution quality for skewed elements, it uses the finite difference method.



For skewed elements, it is assumed that the tangential component $\nabla\phi_\tau$ is negligible on integration surfaces. The normal component $\nabla\phi_n$ is along the unit normal vector of the integration surface $\hat{\boldsymbol{n}}_{ip}$ and is calculated as follows:

$$\nabla\phi_n = \frac{\phi_{N_{in}} - \phi_{N_{out}}}{(\boldsymbol{s}_{in} - \boldsymbol{s}_{out}) \cdot \hat{\boldsymbol{n}}_{ip}}$$

where:

- $\phi_{N_{in}}$ and $\phi_{N_{out}}$ are the solution values at nodes N_{in} and N_{out} .
- $\boldsymbol{s}_{in} = \{x_{in}, y_{in}, z_{in}\}$, $\boldsymbol{s}_{out} = \{x_{out}, y_{out}, z_{out}\}$ are the coordinates at nodes N_{in} and N_{out} .
- $\hat{\boldsymbol{n}}_{ip}$ is the unit normal vector for each integration surface.

The skewed elements treatment is active by default. To deactivate it, add the following line to the *user.prm* file.

```
SKEWED_ELEMENTS= FALSE
```

For more information on the *user.prm* file, see Flow solver files.

14.3 Temporal discretization

The flow solver uses the following flow time integration methods:

- Backward Euler scheme, which is the fully implicit first orders scheme.
- Crank-Nicolson scheme, which is the semi-implicit second-order scheme.

The semi-discrete form of the integral transport equation for a conserved quantity ϕ, written for the control volume centered at the node P depicted in the figure on the shape function section:

$$V_p \frac{\partial}{\partial t}(\rho_p \phi_p) = V_p S_p^m - \sum_{ip} (\mathbf{F}_{ip} \cdot \mathbf{n}_{ip} A_{ip})$$

where:

- S^m is the local volumetric rate of generation of ϕ.
- $\mathbf{F}$ is the total flux vector of ϕ.
- $\mathbf{n}$ is the outward unit normal vector.

14.3.1 Backward Euler scheme

In the fully implicit first order scheme, the integral transport equation is integrated from the time t_n to the time $t_{n+1} = t_n + \Delta t_n$, yielding:

$$\frac{V_p}{\Delta t_n}\left[(\rho_p \phi_p)\big|_{t_{n+1}} - (\rho_p \phi_p)\big|_{t_n}\right] = V_p S_p^m - \sum_{ip}\left(\mathbf{F}_{ip} \cdot \mathbf{n}_{ip} A_{ip}\right)\Big)\Big|_{t_n} + O\left(\Delta t_n\right)$$

14.3.2 Crank-Nicolson scheme

In the semi-implicit second-order scheme, the integral transport equation is integrated by applying a second-order quadrature rule in the approximation of the time integral of the fluxes and sources, yielding:

$$\frac{V_p}{\Delta t_n}\left[(\rho_p \phi_p)\big|_{t_{n+1}} - (\rho_p \phi_p)\big|_{t_n}\right] = \frac{1}{2}\left(V_p S_p^m - \sum_{ip}\left(\mathbf{F}_{ip} \cdot \mathbf{n}_{ip} A_{ip}\right)\right)\Big|_{t_{n+1}} + \\ + \frac{1}{2}\left(V_p S_p^m - \sum_{ip}\left(\mathbf{F}_{ip} \cdot \mathbf{n}_{ip} A_{ip}\right)\right)\Big|_{t_n} + O\left(\Delta t_n^2\right)$$

In comparison with the fully-implicit time integration option, which results in an $O\left(\Delta t_n^2\right)$ truncation error, the use of the second-order accurate temporal discretization of the equation allows the accurate resolution of transient flow features with a significantly larger timestep.

However, the semi-implicit time integration method is susceptible to spurious, oscillatory temporal variation of the solution field if too large a timestep is employed. This method should not be selected when a timestep is large in comparison with the bulk time scale of the problem.

Notes

When you use the `AUTO TURN-OFF FLUIDS EQUATION SOLVE` advanced parameter in a transient analysis, the flow solver reactivates a frozen solution at the beginning of each time step and iterates at least 3 times, even if the solution converged in prior iterations. If the solution did not converge, the solver continues iterating within the time step until it reaches the convergence.

14.3.3 Steady state - Local time step method

The flow solver computes the local time step method using the quantity, ϕ, between two consecutive iterations:

$$\phi^n = \phi^{n-1} + \Delta\phi^n$$

where:

- n is the number of the iteration.
- $\Delta\phi$ is the change in the quantity ϕ.

Using the control volume centered at the node P, which is depicted in the figure on the shape function section, the discrete form of the transport equation for the conserved quantity is given:

$$A_p\phi_p^n + \sum_{NB} A_{NB}\phi_{NB}^n = b^n$$

where b is the source term.

A combination of the two previous equations gives:

$$A_p\Delta\phi_p^n + \sum_{NB} A_{NB}\Delta\phi_{NB}^n = \tilde{b}^n$$

where $\tilde{b}^n = b^n - \left(A_p\phi_p^{n-1} + \sum_{NB} A_{NB}\phi_{NB}^{n-1}\right)$.

The flow solver computes the local time step at each control volume, using the local velocity and element length scale. It solves the following equation using the local time step method and relaxation factor, α:

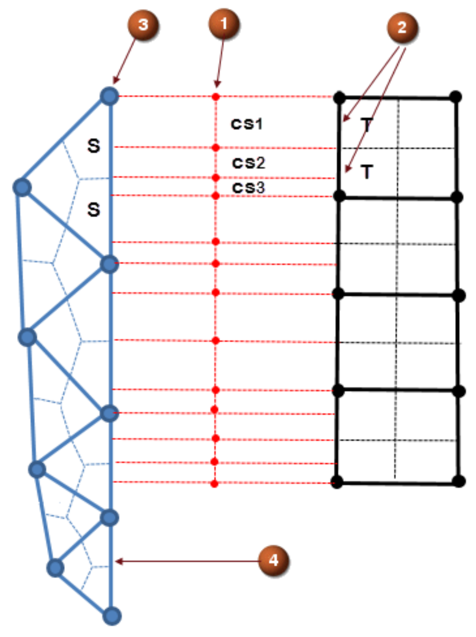
$$\left(1 + \frac{1}{\alpha}\right)A_P\Delta\phi_P^n + \sum_{NB} A_{NB}\Delta\phi_{NB}^n = \tilde{b}^n$$

All the diagonal terms of the coefficient matrix, A_p, are scaled by $\left(1 + \frac{1}{\alpha}\right)$, in which the lower value of the relaxation factor provides more relaxation to the convergence process but usually increases the computational cost.

14.4 Other discretization methods

14.4.1 Disjoint mesh pairing

The flow solver needs to control the fluxes across the interface between two disjoint meshes. The interface is a coplanar face pair where the nodes and elements of the associated fluid meshes are not coincident.

(1) Control surface (2) Sub-faces (3) Source node (4) Unmapped face	

The schematic depicts the interface between the source and target sides with the different element types. Each element face is divided into many sub-faces (2) using integration points: sub-faces S for the source side and sub-faces T for the target side. The sub-faces from the source side and the target side are overlapped creating intersecting control surfaces cs_i (1). The area contribution fraction of each control surface is evaluated based on the intersection of the control surface (1) with the sub-face (2). When there is no intersection between source and target element faces, the unmapped element face is treated as a wall (4).

The area fraction of the sub-face S contributing to the sub-face T is given by:

$$f_{st_i} = \frac{cs_i}{S_S}$$

The area fraction of the sub-face T contributing to the sub-face S is given by:

$$f_{ts_i} = \frac{cs_i}{S_T}$$

So that

$$f_{st_i} S_S = f_{ts_i} S_T = cs_i$$

where:

- cs_i is the i^{th} control surface.
- S_S is the area of the sub-face of the source element face.
- S_T is the area of the sub-face of the target element face.

The solver uses these area fractions for transferring conserved fluxes such as advection, diffusion, mass etc. from one mesh to the other. Conserved fluxes on each control surface are computed as follows:

1. The fluxes of the face elements on each side (F_S , F_T) are first computed independently.
2. The flux of a control surface F_{cs_i} is an area average of the two fluxes:
   $F_{cs_i} = \frac{1}{2}(f_{st_i} F_S + f_{ts_i} F_T),$
   The F_S and F_T are the implicit and conserved fluxes that are normally computed on a face. The previous equation redistributes an average of the computed fluxes on both sides.
3. The redistributed fluxes, F_{cs_i} , are then added to their target nodes on each side. Therefore, the implicit fluxes added to the equations on each side are:
   $F_{Sr} = \sum_{i=1}^{n} F_{cs_i}$
   $F_{Tr} = \sum_{i=1}^{m} F_{cs_i}$

The advection fluxes across the interface between two disjoint meshes are first-order accurate.

With this redistribution procedure, the following conditions are automatically satisfied to define a conservative transfer of fluxes through the interface.

$$\sum_{i=1}^{n} f_{st_i} = 1$$
$$\sum_{i=1}^{m} f_{ts_j} = 1$$

where:

- n is the number of control surfaces to overlap the sub-face S.
- m is the number of control surfaces to overlap the sub-face T.

Pressure drop due to flow through the screen on disjoint meshes

The flow solver computes the pressure drop due flow through the screen on disjoint meshes based on average velocity $U_{average}$ across disjoint fluid mesh pairing:

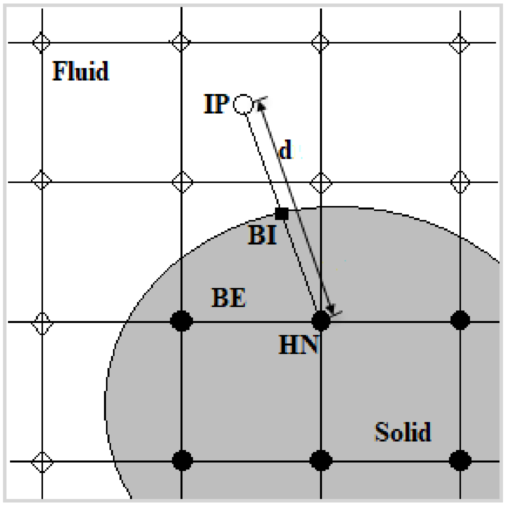
$$\Delta P = \frac{1}{2}\rho h U_{average}^2$$

where h is the total head loss coefficient.

Then, ΔP is added as a surface term (pressure force) to the momentum equations on the disjoint fluid mesh pairing interface.

14.4.2 Immersed boundary method

The flow solver can use the immersed boundary method based on the ghost-cell method to discretize and solve CFD equations. This method allows you to implement boundary conditions for the immersed boundaries without modifying the overall finite-volume algorithms. In this method, the occurence of the immersed boundary is introduced at the halo nodes (HN). Halo nodes belong to immersed boundary elements (BE) and coincide with the solid node. A boundary element in the immersed mesh is a cell that contains part of both the solid body and the fluid domain. The flow variables at the halo nodes are calculated to implicitly satisfy the wall boundary condition using the values at the body intercept points (BI) on the immersed boundary and at the image points (IP). The image point is created by extending a line normal to the immersed boundary, from the halo node to the body intercept point, such that the body intercept point lies at the midpoint of the line connecting the halo node, and the image point. The values near the boundary are calculated using the value at the halo nodes with the appropriate finite difference schemes.



Boundary intercept method

The general variable, ϕ, at an image point, is obtained using trilinear interpolations in an explicit or implicit fashion:

$$\phi_{IP} = \sum_i \alpha_i \phi_i$$

where α is the interpolation coefficient.

The flow solver calculates variables at the halo nodes using linear interpolation between the known values at the image point, and the assigned boundary condition (Dirichlet or Neumann) at the immersed body surface.

The Dirichlet boundary condition assigns the value of the variable at the body intercept, ϕ_{BI} .

A linear interpolation implies that:

$$\phi_{BI} = 0.5(\phi_{IP} + \phi_{HN})$$

Based on the previous equations, the value on the halo node for Dirichlet boundary condition is computed as follows:

$$\phi_{HN} = 2.0 \cdot \phi_{BI} - \sum_i \alpha_i \phi_i$$

The Neumann boundary condition assigns the normal derivative at body intercept point.

$$\frac{\partial \phi_{BI}}{\partial n} = \Gamma_{BI}$$

The central difference method implies that:

$$\frac{\phi_{IP} - \phi_{HN}}{d} = \Gamma_{BI}$$

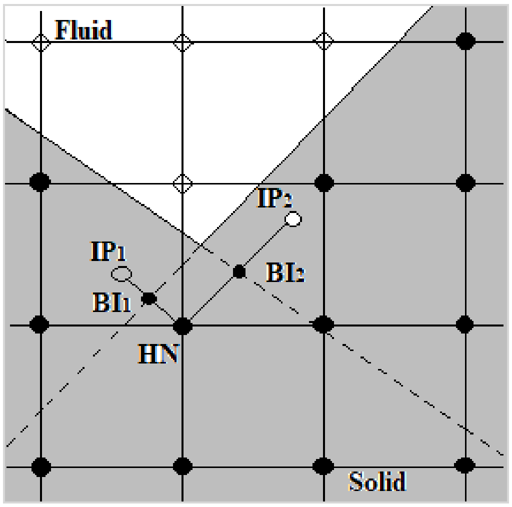
where d is the distance between the halo node and the image point node.

Based on these equations, the value on the halo node for Neumann boundary condition is computed as follows:

$$\phi_{HN} = \sum_i \alpha_i \phi_i - d \cdot \Gamma_{BI}$$

Depending on the type of boundary condition, one of the two equations for ϕ_{HN} replaces the conservation equation for the halo node in the linear system. For more information, see Governing equations.

The trilinear interpolation scheme is used for the image point of the halo node that is located in the fluid region only. If both an image point and a halo node are located in the solid region, for example, a sharp concave corner geometry as shown in the following figure, the versatile sharp interface immersed boundary method is used [56].



14.4.3 Mesh moving methods

The flow solver uses the following methods to compute the mesh movement in fluid structure interaction modeling:

- Radial basis function (RBF) method
- Inverse distance weighting (IDW) method
- Laplace smoothing equation method

The RBF method computes the nodal displacement of the fluid mesh using a radial basis function interpolation. The IDW method computes an interpolated value for the fluid mesh using an inverse distance weighting interpolation technique. The Laplace smoothing equation method computes the fluid nodal displacement by solving the Laplace equations. The IDW method computes translation and rotation separately to preserve a better orthogonality of the bulk fluid mesh region compared to the RBF method. The RBF method requires a linear system computation for each fluid mesh displacement, which makes it less robust than the IDW method. For models with a high number of mesh points, the Laplace smoothing equation performs much faster than both the RBF and IDW. With this method, the flow solver computes the wall distance each time the mesh is moved, to calculate the diffusion coefficient for the next iteration. This is useful in FSI modeling when you use turbulence models that require a wall distance computation.

Radial basis function method

The flow solver applies radial basis function (RBF) interpolation to derive the nodal displacement in the fluid mesh by applying the nodal displacement on the structure boundary for FSI computation [58]. The nodal displacement of the solid structure boundary and fluid internal mesh is computed by:

$$d(\mathbf{x}) = \sum_{j=1}^{n_b} \alpha_j \phi\left(||\mathbf{x} - \mathbf{x}_{b_j}|| \right)$$

where:

- $\mathbf{x}_{b_j}$ is the node position on the boundary and represents the x_{b_j} , y_{b_j} and z_{b_j} coordinate components, respectively.
- n_b is the number of nodes on the boundary.
- $||\mathbf{x}||$ is the Euclidean distance.

The flow solver uses the following RBF:

$$\phi = \sqrt{\frac{1}{1 + \varepsilon r^2}}$$

where:

- ε is the smoothing factor.
- r is the radius.

The α_j coefficients are computed as follows:

$$d_{b_i} = \sum_{j=1}^{n_j} \alpha_j \phi \left(||\mathbf{x}_{b_i} - \mathbf{x}_{b_j}|| \right)$$

where:

- d_b is the displacement of the nodes on the structure boundary.
- $\mathbf{x}_{b_i}$ is the location of the displaced nodes on the solid structure boundary.

The nodal displacement in the fluid mesh, d_I , is computed by:

$$d_{I_i} = \sum_{j=1}^{n_j} \alpha_j \phi \left(||\mathbf{x}_{I_i} - \mathbf{x}_{b_j}|| \right)$$

where $\mathbf{x}_{I_i}$ is the node position in the fluid mesh and represents the x_{I_i} , y_{I_i} and z_{I_i} coordinate components, respectively.

Inverse distance weighting method

The flow solver uses the inverse distance weighting (IDW) method in the FSI modeling to compute the interpolated value, $d(\mathbf{x})$, at a given node, $\mathbf{x}$, in the fluid mesh as a weighted average value of the structure boundary nodes, $d_{b_j}(\mathbf{x}_{b_j})$, [59]. The interpolated value is defined as:

$$d(\mathbf{x}) = \frac{\sum_{b_j=0}^{n} w_{b_j} \left(\mathbf{r}_{b_j} \right) d_{b_j}}{\sum_{b_j=0}^{n} w_{b_j} \left(\mathbf{r}_{b_j} \right)}$$

where:

- $\mathbf{x}_{b_j}$ is the node position on the structure boundary and represents the x_{b_j} , y_{b_j} and z_{b_j} coordinate components, respectively.
- $\mathbf{r}_{b_j}$ is the distance between fluid node and boundary node.
- n is the number of boundary nodes.
- $w_{b_j}\left(\mathbf{r}_{b_j}\right)$ is the weighting function.

The distance between fluid node and boundary node is computed as:

$$\mathbf{r}_{b_j} = \mathbf{x} - \mathbf{x}_{b_j}$$

The flow solver uses the following two-exponent form of the weighting function to compute a rigid deformation near boundaries and a viscous mesh deformation in the fluid mesh.

$$w_{b_j}\left(\mathbf{r}_{b_j}\right) = A_{b_j}\left[\left(\frac{L_{def}}{\mathbf{r}_{b_j}}\right)^p + \left(\frac{\alpha L_{def}}{\mathbf{r}_{b_j}}\right)^q\right]$$

where:

- A_{b_j} is the area weight assigned to the boundary node, b_j .
- L_{def} is the estimated length of the deformation region.
- α is the estimated size of near body influence region.
- $p = 3$.
- $q = 5$.

The estimated length of the deformation region is computed as:

$$L_{def} = \max_{b_j=0}^{n} ||\mathbf{r}_{b_j} - \mathbf{r}_{centroid}||$$

where $\mathbf{r}_{centroid} = \frac{1}{n}\sum_{b_j=0}^{n}\mathbf{r}_{b_j}$.

The estimated size of near body influence region is computed as:

$$\alpha = \frac{\eta}{L_{def}}\max_{b_j=0}^{n} ||d_{b_j} - d_{mean}||$$

where:

- d_{mean} is the average displacement field.
- $\eta = 5$.

The average displacement is computed as:

$$d_{mean} = \frac{\sum_{b_j=0}^{n} A_{b_j} d_{b_j}}{\sum_{b_j=0}^{n} A_{b_j}}$$

Laplace smoothing equation method

The flow solver uses the Laplace smoothing equation method to compute nodal displacement in the fluid mesh [60]. The Laplace operator is:

$$\nabla . \left(\gamma \nabla \mathbf{d} \right) = 0$$

where:

- $\mathbf{d}$ is the nodal displacement.
- γ is the spatially varying diffusion coefficient.

The spatially varying diffusion coefficient is computed as:

$$\gamma = \frac{1}{h^2}$$

where h is the nodal distance from the wall boundary.

The nodal location after displacement, $\mathbf{x}_{new}$, is computed as:

$$\mathbf{x}_{new} = \mathbf{x}_{old} + \mathbf{d}$$

where $\mathbf{x}_{old}$ is the most recent nodal location before displacement.

The flow solver computes the wall distance each time the mesh is moved to calculate the diffusion coefficient for the next iteration. This is useful in FSI modeling when you use turbulence models that require a wall distance computation.

15 Solver numerical methods

15.1 Linear equation solution

The preconditioning method to the basic iterative algorithm is a Incomplete Lower Upper factorization method. As this is not a direct solver, it is used within an iterative refinement loop, in which the exact solution is calculated with iteration.

The system of equations can be written symbolically as:

$$Ax = b$$

- A is the coefficient matrix.
- x *is* the solution vector (e.g. the $U, V, W, P, H, k, \rho_v \Phi$).
- b is the coefficient vector.

Solving this iteratively, one starts with an approximate solution, x^n , that is to be improved by a correction, x' to yield a better solution, x^{n+1} , i.e.,

$$x^{n+1} = x^n + x'$$

where

- x' is a solution of the following equation:

$$Ax' = r^n$$

- r is the the residual given as:

$$r^n = b - Ax^n$$

The approximate iterative solver is used to solve the $Ax' = r^n$.

15.2 Solver methods

The flow solver uses iterative Krylov methods, [46] to solve large set of algebraic equations resulting from the discrete approximation of differential equations. Krylov subspace methods work by forming a basis of the sequence of successive matrix powers A times the initial residual r , the Krylov sequence with iteration number m :

$$K_m(A, r) = span\{r, Ar, A^2r \ldots A^{m-1}r\}$$

The approximations to the solution are then formed by reducing the residual over the subspace formed by the Krylov methods.

The flow solver uses an adaptive multi-methods for efficient solver selection, where the linear solver method is selected dynamically by observing linear system properties as they evolve during the simulation process. Selection of an appropriate solver can lead to benefits such as reduced memory requirement, lower execution time and reliability.

The dynamic linear solver can use four Krylov methods:

- BiCGStab(1)
- BiCGStab (2), where 2 is the second degree polynomials.
- IDR(4), where 4 is the size of the initial orthogonal set of vectors used for the reduction process.
- GMRES (30), where 30 is the size of the Krylov subset.

and two preconditioners:

1. Algebraic multigrid preconditioner for fully coupled flow scheme with the following smoothers:

- Block Gauss-Seidel
- Incomplete LU (ILU) decomposition
- Algebraic Domain Decomposition

The following table shows the 12 combinations of the algebraic multigrid preconditioner:

BiCGStab(1) - Block Gauss-Seidel	BiCGStab(1) - ILU	BiCGStab(1) - Algebraic Domain Decomposition
BiCGStab (2) - Block Gauss-Seidel	BiCGStab (2) - ILU	BiCGStab (2) - Algebraic Domain Decomposition
IDR(4) - Block Gauss-Seidel	IDR(4) - ILU	IDR(4) - Algebraic Domain Decomposition
GMRES (30) - Block Gauss-Seidel	GMRES (30) - ILU	GMRES (30) - Algebraic Domain Decomposition

2. Incomplete LU (ILU) decomposition for single variable transport equation with the following options:

- Fill-in ILU
- Overlap ILU

The following table shows the ILU decomposition combinations:

BiCGStab(1) + Fill-in ILU + Overlap ILU
BiCGStab (2) + Fill-in ILU + Overlap ILU

IDR(4) + Fill-in ILU + Overlap ILU
GMRES (30) +Fill-in ILU + Overlap

16 Flow solver files

The following table describes the flow solver files, which are written in the run directory during the solving process.

File name	Description
`escerr.dat`	Contains the warnings and error messages issued by the flow solver during a solution.
`flowdata.bfd`	Contains the results shared among all the processes of the flow solver (nodal coordinates, element connectivity, list of variables, etc.), which are passed to the results BUN file.
`flowdata_i.bfp`	Contains the results corresponding to each process from the flow solver, which are passed to the results BUN file. Here, *i* indicates the process number associated with the current file.
`flow.bfi`	Contains the boundary condition types and values specified on each selected boundary face.
`flow.cfn`	Relates fluid faces to their correspondent coated shell element surfaces. The flow solver uses fluid faces and the thermal solver uses shell element surfaces in the coupled thermal-flow analysis. This file also contains the method that the flow solver uses to compute the initial heat transfer coefficient values for natural convection problems.
`flow.csc`	Exchanges the control parameters between flow and thermal solvers during the simulations.
`flow.ent`	Contains the number and name of each flow boundary conditions and additional information on flow surfaces.
`flow.fbc`	Transfers boundary conditions defined on the faces from the thermal solver to the flow solver through a converter function.
`flow.gem`	Contains information about geometry: • Element geometry definitions • Nodal coordinates

File name	Description
`flow.ibd`	Contains the boundary condition types and values specified on each selected immersed boundary when you solve an immersed boundary model.
`flow.ibg`	Contains the following information when you solve an immersed boundary model: • Halo nodes • Boundary elements • Immersed triangles
`flow.icp`	Contains the following information when you solve an immersed boundary model in a coupled thermal-flow analysis: • Flow surface types • Convection properties • Flow surfaces information • Thermal elements
`flow.job`	Contains the completion status of the run and summarizes the CFD results, when the run terminates successfully.
`flow.mgd`	Contains global summary information about the flow model, number of nodes, and number of elements.
`flow.opn`	Contains a list of face elements that form openings. These face elements are grouped in sets that geometrically form a contiguous opening.
`flow.prm`	Contains the following solution control parameters, which define the run: • Steady-state relaxation timestep • Iteration limits and convergence criteria • Discretization parameters • Advection scheme • Turbulence model type • Specified advanced parameters
`flow.prp`	Contains the following information: • Material description • Fluid material property definition • Enclosures material information • Variable boundary condition definition • Fan head loss information

File name	Description
`flow.pst`	Contains face-based and nodal-based results for post-processing purposes.
`flow.restart`	Stores a selected flow solver memory image that is used for restarting the flow solver.
`flow.struct`	Contains a memory map of the flow solver in the form that is stores in the `flow.restart` file.
`flow.vsi`	Contains the volumetric boundary condition type and its specified values on the selected polygon body for each volumetric boundary condition in the model.
`groups.unv`	Contains the groups that have an information about warnings and error messages issued by the solver.
`ibm.prm`	Contains advanced parameters used for immersed boundary analysis.
`user.prm`	Contains user-specified optional solver parameters that are not available in the advanced parameters catalog. These parameters, which are provided by the customer support, let you modify the behavior of the flow solver to enhance its capabilities for specific problems. You create the `user.prm` file using a text editor in your run directory.

17 Flow solver references

1. Currie, I.G. Fundamental Mechanics of Fluids, McGraw Hill, 1974.
2. Arpaci, V.S., Larsen, P.S. Convection Heat Transfer, Prentice Hall, 1984.
3. Rosehnow, Hartnett and Cho. Handbook of Heat Transfer, 3rd Edition, McGraw Hill, 1998.
4. Wilcox, DCW Industries. Turbulence Modeling for CFD, Second Edition, D.C, 1998.
5. Schneider, G.E. and Raw, M.J., Control Volume Finite-Element Method for Heat Transfer and Fluid Flow using Colocated Variables- 1. Computational Procedure. Numerical Heat Transfer, Vol.11, pp.363-390, 1987.
6. Rhie, C.M. and Chow, W.L., A Numerical Study of the Turbulent Flow Past an Isolated Airfoil with Trailing Edge Separation, AIAA paper 82-0998, 1982.
7. Patankar, S.V., Numerical Heat Transfer and Fluid Flow, Hemisphere, Washington D.C., 1980.
8. Leonard et al., International Journal for Numerical Methods in Fluids, vol. 20, p. 421, 1995.
9. Kader, B.A.. "Temperature and Concentration Profiles in Fully turbulent Boundary Layers". Int. J. Heat Mass Trans., 21, N0.9, pp 1541-1544, 1981.
10. Raw, M.J., A Coupled Algebraic Multigrid Method for the 3D Navier-Stokes Equations. 10th GAMM Seminar, Kiel 1994.
11. Briggs W.L., A Multigrid Tutorial, SIAM, Philadelphia, 1987.
12. Hutchinson, B.R., Raithby, G.D: A Multigrid Method Based on the Additive Correction Strategy, Numerical Heat Transfer, Vol. 9, pp.511-537, 1986.
13. ASHRAE Fundamentals Handbook. Chapter 6: Psychometrics, 1997.
14. Sweeby, P.K., "High Resolution Schemes Using Flux-Limiters for Hyperbolic Conservation Laws", SIAM J. Numer. Anal., Vol. 21, pp. 995-1011, 1984.
15. Leonard, B.P., Drummond, J.E., "Why you should not use "hybrid", "Power-Law" or related exponential schemes for convective modeling - There are much better alternatives", International Journal for Numerical Methods in Fluids, Vol. 20, pp.421-442, 1995.
16. Gaskell, P.H., Lau, A.K.C., "Curvature-Compensated Convective Transport: SMART, a New Boundedness-Preserving Transport Algorithm", International Journal for Numerical Methods in Fluids, Vol. 8, pp617-641, 1988.
17. Yap, C.R., "Turbulent Heat and Momentum Transfer in Recirculating and Impinging Flows", Ph.D. Thesis, University of Manchester, 1987.
18. Reynolds, W.C., Perkins, H.C.: Engineering Thermodynamics, McGraw Hill, 1977.
19. Wilke, R.C. , "A Viscosity Equation for Gas Mixtures", J.Chem. Phys., vol. 18, p. 517, 1950
20. R. Cheesewright, Turbulent natural convection from a vertical plane surface, J. Heat Transfer 90, 1-8, 1968.
21. T. Tsuji and Y. Nagano, Velocity and temperature measurements in a natural convection boundary layer along a vertical flat plate, Experimental Thermal and Fluid Science 2, 208-215, 1989.
22. X. Yuan, A. Moser, P.Suter, Wall functions for numerical simulation of turbulent natural convection along a vertical plate, Int. J. Heat Mass Transfer 36, 4477-4485, 1993.
23. G.D. Raithby, K.G.T. Hollands, Natural Convection, Handbook of Heat Transfer, Third Edition, W.M. Rohsenow, J.P. Harnet, Y.I. Cho, McGraw Hill, 1998.
24. F.P. Incropera, D.P. DeWitt, Fundamentals of Heat and Mass Transfer, Fourth Edition, John Wiley, Inc., 1996
25. S. Whittaker, Volume Averaging of Transport Equations. In J. Prieur du Plessis (ed.). Fluid Transport in Porous Media. Chap. 1. Computational Mechanics Publications: Southampton, UK, 1997.
26. V. Venkatakrishnan, Convergence to Steady State Solutions of the Euler Equations on Unstructured Grids with Limiters. Journal of Computational Physics 118, 120-130, 1995.
27. G.K. Batchelor, An Introduction to Fluid Dynamics, Cambridge, UK, 2002.
28. H. Schlichting and K. Gersten, Boundary Layer Theory, 8th edition, Springer, Berlin, 2000.
29. A. Einstein, Investigations on the Theory of the Brownian Movement, Dover, NY, 1956.
30. S.B. Pope, Turbulent Flows, Cambridge, UK, 2000.

31. G.R. Fowles and G.L. Cassiday, Analytical Mechanics, 6th edition, Brooks and Cole, Stamford, 1999.
32. J.H. KIm, "Slip Correction Measurements of Certified PSL Nanoparticles Using a Nanometer Differential Mobility Analyzer (Nano-DMA) for Knudsen Number From 0.5 to 83", J.Res.Natl.Inst.Stand.Technol. 110,31-54, 2005.
33. K. Willeke, "Atmospheric Aerosols: Size Distribution Interpretation", J. of the Air Pollution Control Association, 25:5, 529-534, 1975.
34. T. Knopp, T Alrutz, and D. Schwamborn, "A grid and flow adaptive wall-function method for RANS turbulence modeling", J. of Computational Physics, 220, 19-40, 2006.
35. B. Aupoix, "Wall roughness modelling with kw stt model", In 10th International ERCOFTAC Symposium on Engineering Turbulence Modelling and Measurements, 2014.
36. A. Cook, ''Enthalpy Diffusion in Multicomponent Flows", Physics of Fluids, 2008.
37. J. Hunt, "Eddies, stream, and convergence zone sin turbulent flow", Center for Turbulence Research Report CTR-S88, 193-208.
38. P. Galpin, R.B. Broberg and B. Hutchinson, "Three- Dimensional Navier Stokes Predictions of Steady-State Rotor/Stator Interaction with Pitch Change", 3rd Annual Conference of the CFD, Society of Canada, Banff, Alberta, Canada, Advanced Scientific Computing Ltd, June 25-27, 1995.
39. B. Leonard, "A stable and accurate convective modelling procedure based on quadratic upstream interpolation", Computer Methods in Applied Mechanics and Engineering, 19(1): 59-98.
40. T. Barth. D. Jespersen, "The Design and Application of Upwind Schemes on Unstructured Meshes", Technical Report AIAA-89-0366, AIAA 27th Aerospace Sciences Meeting, Reno, 9-12 January 1989.
41. G. Bird, "Molecular Gas Dynamics and the direct Simulation of Gas Flows", Oxford University Press", New York, 1994.
42. J. Maxwell, "On stresses in rarefied gases arising from inequalities of temperature", Philos. Trans. Roy. Soc., Part 1, 170, pp. 231-256, 1879.
43. M. Smoluchowski, "Über wärmeleitung in verdünnten gasen", Ann. Phys. Chem., 64, pp.101-130, 1898.
44. A. Vreman, "An eddy-viscosity subgrid-scale model for turbulent shear flow: Algebraic theory and applications", Physics of Fluids, 2004.
45. H. van der Vorst, "Iterative Krylov methods for large linear system", Cambridge University Press, 2003.
46. P. Sonneveld, van Gijzen, "A family of simple and fast algorithms for solving large nonsymmetric systems of linear equations", SIAM Journal on Scientific Computing 31(2), pp. 1035-1062, 2009.
47. Y. Saad and M Schultz, "A generalized minimal residual algorithm for solving nonsymmetric linear systems", SIAM Journal on Scientific and Statistical Computing, Vol. 7, No.3, pp. 856-869.
48. J. Gay-Lussac, "Mémoire sur la combinaison des substances gazeuses, les unes avec les autres", Mémoires de la Société d'Arcueil 2, 207-234, 1809.
49. T. M. Resource, “The Spalart-Allmaras turbulence model,” 2016. [Online]. Available: turbmodels.larc.nasa.gov/spalart.html.
50. V. Yakhot and S.A. Orszag, "Development of turbulence models for shear flows by a double expansion technique," Physics of Fluids A: Fluid Dynamics 4, 1510, 1992.
51. T.-H. Shih, W.W. Liou, A. Shabbir, Z. Yang, and J. Zhu, "A New - Eddy-Viscosity Model for High Reynolds Number Turbulent Flows - Model Development and Validation," Computers Fluids, 24(3), 227–238. 1995.
52. R. McCraw, "Description of aerosol dynamics by the quadrature method of moments," Aerosol Science and Technology, 27(2): 255-265, 1997.
53. F. Bakhtar, J.B. Young, A.J. White, and D.A. Simpson, "Classical nucleation theory and its application to considering steam flow calculations," Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 219(12): 1315-1333, 2005.
54. A.G. Gerber and A Mousavi, "Representing polydispersed droplet behaviour in nucleating stream flow," Journal of fluids engineering, 129(11): 1404-1414, 2007.
55. Mittal et al, “A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries”, JCP 227, pp 4825-4852, 2008.

56. Mavriplis, D. J. and Yang Z., "Construction of the discrete geometric conservation law for high-order time-accurate simulations on dynamic meshes", Journal of Computational Physics 213, 557–573, 2006.
57. de Boer, A., van der Schoot, M. S., and Bijl, H., "New method for mesh moving based on radial basis function interpolation", European Conference on Computational Fluid Dynamics ECCOMAS CFD 2006.
58. Luke E., Collins E. and Blades E., "A fast mesh deformation method using explicit interpolation", Journal of Computational Physics, 231, 586-601, 2012.
59. Jasak H, and Tukovie Z., Automatic Mesh Motion for the Unstructured Finite Volume Method, Conference Proceedings, 2007.
60. Van Albada, G.D.; Van Leer, B.; Roberts, W.W. (1982), "A comparative study of computational methods in cosmic gas dynamics", *Astronomy and Astrophysics*, **108** (1): 76–84.
61. Roe, P.L. (1986), "Characteristic-based schemes for the Euler equations", *Annu. Rev. Fluid Mech.*, **18**: 337–365.
62. Van Leer, B. (1974), "Towards the ultimate conservative difference scheme II. Monotonicity and conservation combined in a second order scheme", J. Comput. Phys., 14 (4): 361–370.
63. Versteeg, H. K., and W. Malalasekera. "An Introduction to Computational Fluid Dynamics."

Better know-how

Our engineers are skilled in numerical simulation, many with advanced degrees and senior project management experience. Their proficiency in thermal, flow and structural analysis, helps build and analyze thousands of individual components, subassemblies and entire structures around the globe.

Drawing on a portfolio of leading thermal, flow and structural solver technology, we support all stages of the product development cycle. We know that better methodologies leader tobetter design quality, even for the most intricate designs, which means you can trust Maya HTT to bring insight and understanding to the most complex engineering efforts.

Maya HTT Better thinking Better future®

Email
info@mayahtt.com

Web
mayahtt.com

Tel.
+1.514.369.5706

Address
1100 Atwater Avenue, Suite 3000
Westmount, QC H3Z 2Y4 Canada