Altair® AcuSolve® 2025

Altair AcuSolve Training Manual

Updated: 11/18/2024

Contents

AcuSolve Training Manual

1

Introduction of background knowledge regarding flow physics and CFD as well as detailed information about the use of AcuSolve and what specific options do.

Introduction to CFD

2

This chapter covers the following:

Fluid Analysis Overview

This section details the various approaches available for analyzing a general fluid flow problem. It also provides pros and cons for each of these approaches.

Analysis of any physical problem, in general, is governed by a set of basic governing equations. The basic equations governing the motion of fluids are the Navier-Stokes equations, which were developed more than 150 years ago.

Continuity Equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) = 0 \quad (1)$$

Momentum Equation

$$\rho \frac{\partial \vec{u}}{\partial t} + (\rho \vec{u} \cdot \nabla)\vec{u} = -\nabla p + \rho b + \nabla \cdot \tau \quad (2)$$

Energy Equation

$$\rho \frac{\partial h}{\partial t} + (\rho \vec{u} \cdot \nabla)h = \nabla \cdot (k \nabla T) + \nabla \vec{u} : \tau + \frac{Dp}{Dt} + S \quad (3)$$

These equations are a set of coupled partial differential equations which describe the relation among the flow properties like pressure, temperature and velocity. The solution to the above equations is in general complex because of the non-linear terms and coupled nature of the equations. However, in general the following three solution approaches are available:

Analytical Approach

An analytical solution can be obtained by integrating the above boundary value problem, resulting in an algebraic equation for the dependent variables as a function of independent variables. Because of the complexity of the Navier-Stokes equations a general solution for all physical problems is an insurmountable task. However, analytical solution is possible in some simple cases with certain approximations in geometry, dimensionality and compressibility of the flow. A few examples of such flows include Couette flow, Hagen-Poiseuille flow and Parelle flows through straight channels. An obvious limitation of this approach is the practical applications of these flows.

Experimental Approach

This approach uses the physical experiments on the scale down models of the problem at hand and extrapolates the results based on similarity laws. For example, to investigate the flow around an actual aircraft a Reynolds number based miniature model is placed in the wind tunnel and the results are extrapolated to that of the actual aircraft. Although a variety of experimental techniques are available, the choice of the particular type to use depends on capturing the parameter of interest. For example, Particle Image Velocimetry (PIV), Laser Doppler Velocimetry (LDV) and Hot-wire anemometry are used to measure flow velocity. PIV is used to produce 2D or 3D vector fields. Hot-wire anemometry and LDV are used when measuring the rapidly varying velocities at a point with good spatial and time resolution is important. A few

limitations of this approach include the necessity of conducting a large number of experiments for the complete description of flow, cost and the complexity of conducting them.

Numerical Approach (Computational Fluid Dynamics-CFD)

The third approach available for investigating fluid flow is CFD. In this approach, the fluid region of interest is divided into finite regions and the complex governing equations are discretized to obtain a set of algebraic equations that are advanced in time and space using computers. The end result of the CFD analysis is a complete description of the fluid flow within the region of interest.

CFD Brief History

This section details the advancements in CFD research throughout the twentieth century.

Although the study of fluid mechanics is centuries old most of the evolution of CFD took place throughout the twentieth century. By the start of the twentieth century all of the basic governing equations were consolidated. Closed form analytical solution for simple problems were derived putting aside the large set of problems of practical significance that are not suitable for analytical solution. Before the 1950's the main focus was on developing a variety of approximated semi analytical techniques such as perturbation methods for viscous boundary layer calculations, a method of characteristics for compressible flow calculations alongside the numerical solution development. By the mid-century the Finite difference based numerical methods were developed. However, the numerical algorithms until the fifties provided a solution to the single partial differential equations limited by the computation power. With the advent of digital computers the development of numerical simulation was put on an accelerated path. During the fifties and sixties significant developments were made on solution of inviscid compressible flows.

The seventies largely focused on inviscid flows over geometries like airfoils, wings and wing-body combinations. During the eighties intense efforts were made on algorithms for solution of Euler equations which include rotational flows but still inviscid. Research on grid generation was invested in the late eighties which presented a challenge in generating suitable grids for the computational domain of complex geometries. During the late eighties the transonic flow computations with viscous compressible flow over airfoils was made possible. This resulted in designing the super critical airfoils computationally with reduced wave drag. For the first time one could move away from the catalogued airfoil shapes to custom-designed airfoils to suite design end.

In the nineties the power of parallel computing paved the way for turbulence modeling in the flow. Turbulence was incorporated using turbulence modeling techniques such as Reynolds-averaged Navier Stokes (RANS) and Large Eddy Simulation (LES) along with Direct Numerical Simulation (DNS) without any modeling. This period marked the beginning of an era of commercial software capable of impressive visualization of vast numerical data generated by the CFD simulation of flow fields.

CFD Advantages

This section details the diverse advantages of CFD.

Several of the many possible advantages of CFD are listed below:

- Evaluation of performance before modifying or installing a system.

  Without commissioning a real system CFD results can forecast critical regions for improved performance.
- Provides detailed information of the variables on the complete domain.

  CFD simulation can provide all of the information in a single run, unlike the experiments that require multiple runs for evaluating different variables.
- Non-intrusive techniques with powerful visualizing techniques.

  Some of the experiments are intrusive, such as measuring equipment interfering with the flow under consideration, which can be eliminated with CFD.
- The possibility to have a variety of post-processing providing a deeper insight of results.

  CFD can help with evaluating the user-defined variables such as drag and pressure coefficient directly.
- Costs much less than experiments with lesser turnaround times.

  Most of the experimental equipment is costly and requires a large amount of time and expertise in setting up the experiment unlike CFD simulation.

Theoretical Background

3

This chapter covers the following:

Mathematical Background

This section on mathematical background covers the various notations and operators used to formulate and define the equations of fluid flow.

Vector or Dyadic Notation

The conservation laws for a continuum medium involve vector and tensor quantities as well as several operators such as gradient and divergence.

In order to have a comprehensive understanding of these equations it is essential to get a good grasp of the notations and operators used in this manual.

In this notation, the governing equation is independent of the choice of the coordinate system. For the purposes of this manual the scalar quantities would be denoted with italicized letters. For example, pressure field is denoted as p. The vector quantities would be denoted with an arrow above the letters, for example velocity field denoted as $\vec{u}$. Tensor quantities are donated with bold face letters, for example stress tensor represented as $\boldsymbol{\tau}$.

Cartesian Tensor Notation

In this notation, an index subscript is written after the quantity which corresponds to a component of the quantity.

Single subscript is used to denote a component of vector, for example u_i for the vector field $\vec{u}$ and two subscripts are used to denote a component of tensor, for example τ_{ij} for the stress tensor $\boldsymbol{\tau}$.

Kroneker Delta

The Kroneker Delta is a second-order isotropic tensor which is defined as: $\delta_{ij} = 1 \quad \text{for} \quad i = j$, $\delta_{ij} = 0 \quad \text{for} \quad i \neq j$.

Operators

The most frequently used math operator related to flow equations is the ∇ referred to as nabla, grad or del.

The nabla operator operates on the quantity to the right of it and the rules of a derivative of a product still hold. Otherwise the nabla operator behaves like any other vector in an algebraic operation. It is a vector operator and for a Cartesian coordinate system, it is defined as

$$\nabla = \frac{\partial}{\partial x}\hat{i} + \frac{\partial}{\partial y}\hat{j} + \frac{\partial}{\partial z}\hat{k} \tag{4}$$

∇ appears in several different ways when applied to scalar, vector and tensor quantities.

Gradient

When the nabla operator is applied on a scalar quantity φ, it is called Gradient and it gives a vector whose components are the partial derivatives.

$$\nabla\varphi = \frac{\partial\varphi}{\partial x}\hat{i} + \frac{\partial\varphi}{\partial y}\hat{j} + \frac{\partial\varphi}{\partial z}\hat{k} \tag{5}$$

The gradient of φ points to the direction of greatest change of φ and has a magnitude equal to the rate of change of φ with respect to distance in that direction.

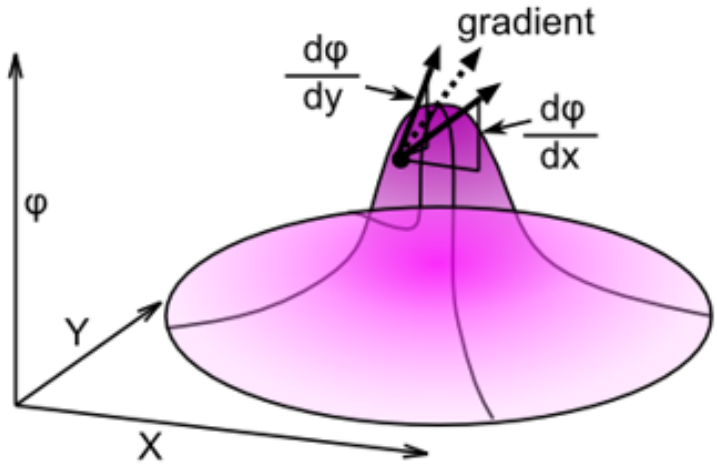


Figure 1: Gradient at a Point

When gradient is applied to a tensor quantity it produces a tensor one rank higher. When applied to a vector quantity $\vec{u}$ it gives a second-order tensor given by

$$\nabla(\vec{u}) = \left(\frac{\partial}{\partial x}\hat{i} + \frac{\partial}{\partial y}\hat{j} + \frac{\partial}{\partial z}\hat{k}\right)\left(u\hat{i} + v\hat{j} + w\hat{k}\right) \equiv \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} & \frac{\partial v}{\partial z} \\ \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} & \frac{\partial w}{\partial z} \end{bmatrix} \tag{6}$$

The gradient of a vector quantity represents the gradient of each component of the vector field individually, each of which is a scalar.

Divergence

The divergence of a vector quantity $\vec{u}$ is defined as a scalar quantity given by

$$\nabla \cdot \vec{u} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} \tag{7}$$

The divergence of a vector field represents the flux generation per unit volume (flux density) at each point on the field.

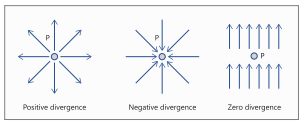


Figure 2: Divergence at a Point

The divergence of a tensor gives a tensor which is one rank lower. When applied to a second-order stress tensor τ it yields a vector field whose ith component is expressed as

$$\left(\nabla \cdot \tau\right)_i = \frac{\partial \tau_{ij}}{\partial x_j} \tag{8}$$

Div

The div operator is defined as $\vec{u} \cdot \nabla$ and is expressed as

$$u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y} + w\frac{\partial}{\partial z} \tag{9}$$

It maps a vector quantity to a scalar which can then be applied to a scalar, vector or higher order tensors.

Laplacian

The Laplacian of a scalar quantity φ is a scalar defined as

$$\nabla^2\varphi = \frac{\partial^2\varphi}{\partial x^2} + \frac{\partial^2\varphi}{\partial y^2} + \frac{\partial^2\varphi}{\partial z^2} \tag{10}$$

Double Dot Product (Scalar Product of Two Tensors)

The double dot product is a doubly contracted product between two tensors. It is expressed as

$$A : B = A_{ij}B_{ji} \tag{11}$$

Total Derivative

The total derivative, also known as material derivative or substantial derivative, describes the rate of change of a physical quantity of a fluid element with time as it moves along a trajectory in a velocity field. The total derivative of a scalar quantity φ is a scalar defined as

$$\frac{D\varphi}{Dt} = \frac{\partial\varphi}{\partial t} + \vec{u} \cdot \nabla\varphi \tag{12}$$

The total derivative of a quantity can be seen as a sum of local time derivative and convective derivative.

Basics of Fluid Mechanics

This section on basics of fluid mechanics covers topics describing the fundamental concepts of fluid mechanics, such as the concept of continuum, the governing equations of a fluid flow, definition of similitude and importance of non-dimensional numbers, different types of flow models and boundary layer theory.

Concept of Continuum

This section introduces the concept of a continuum medium.

Any material, solid, liquid or gas is composed of billions of individual molecules, in a very small region, separated by empty spaces in between them. The continuum assumption considers that matter is continuously distributed and fills the entire region of the space it occupies.

A continuum material can be continually sub divided into infinitesimal elements with properties being those of the bulk material. Properties such as density, pressure, temperature and velocity are taken to be well-defined at infinitely small points.

For a continuum fluid, at each point of the region of the fluid it is possible to construct one volume infinitesimally small enough compared to the region of the fluid and still big enough compared to the molecular mean free path. This condition is mathematically expressed using a non dimensional number known as a Knudsen number.

Describing a fluid flow quantitatively makes it necessary to assume that flow variables, such as pressure and velocity, and fluid properties vary continuously from one point to another. Mathematical description of flow on this basis have proved to be reliable and treatment of fluid medium as a continuum has firmly become established.

Governing Equations

This section describes the definition and formulation of the equations governing the conservations of mass, momentum and energy in a fluid flow and obtaining a closed form solution from these equations.

The equations which govern the motion of a fluid are called the Navier-Stokes equations. These equations refer to the basic governing continuity equations for a compressible, viscous, heat conducting fluid.

They describe the transport of three conserved quantities for a local system: mass, momentum and energy and can be expressed in an integral form when applied to a finite region known as the control volume, or a differential form when applied at an infinitesimally small fluid element.

Continuity Equation

The instantaneous mass conservation equation commonly termed as the continuity equation, is derived by applying mass conservation to a control volume for a general fluid.

Its differential form is written as:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) = 0 \quad (13)$$

where

ρ is the fluid density

t is time

$\vec{u}$ is the flow velocity vector

The first term $\frac{\partial \rho}{\partial t}$ describes the rate of change of density with respect to time and the second term $\nabla \cdot (\rho \vec{u})$ describes the divergence of the vector field $\rho \vec{u}$ at a particular point fixed in space. The continuity equation can also be expressed using the substantial derivative and is written as:

$$\frac{D\rho}{Dt} + \rho \ (\nabla \cdot \vec{u}) = 0 \quad (14)$$

Momentum Equation

The momentum conservation equation is derived by applying Newton's second law of motion to the fluid control volume, which applies as the rate of change of momentum of the fluid particle equals the sum of surface and body forces.

The surface forces comprise of viscous forces (Tensile and shear) and pressure (compressive) and the body (volume) forces can be gravity, centrifugal force, Coriolis force, electromagnetic force, and so on.

The momentum equation is written as:

$$\rho \ \frac{\partial \vec{u}}{\partial t} + (\rho \vec{u} \cdot \nabla)\vec{u} = -\nabla p + \rho b + \nabla \cdot \tau \quad (15)$$

where

p is the pressure

b is the source term (generally gravity)

τ is the viscous stress tensor

If the fluid is Newtonian in nature, the viscous stresses are proportional to the time rate at which strain occurs. Stokes hypothesis for a Newtonian fluid simplifies the stress tensor to:

$$\tau_{ij} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} \mu \delta_{ij} \frac{\partial u_k}{\partial x_k} \quad (16)$$

where μ is the dynamic viscosity of the fluid

The general form of the momentum equation can be seen to be comprised of the following terms:

Time derivative + Convection terms = Forcing (Source) terms + Diffusive terms

The momentum conservation equation for the flow velocity vector $\vec{u}$ can be written as three separate equations in an orthogonal system. These equations describe the conservation of momentum in x, y, z directions with flow velocity components u, v and w.

Energy Equation

The energy conservation equation is derived by applying first law of thermodynamics to fluid control volume, which applies as rate of change of energy of a fluid particle equals the sum of rate of heat addition and rate of work done.

The energy of a fluid particle comprises of internal (thermal) energy and kinetic energy. Heat addition generally takes place in the form of conduction due to a temperature difference through sources such as chemical reactions, potential energy, and so on. Work is done on the fluid volume by the surface and body forces.

The energy equation is written as

$$\frac{\partial(\rho E)}{\partial t}+\nabla\cdot(\rho E\vec{u})= \quad \nabla\cdot(k\nabla T)-\nabla\cdot(p\vec{u})+\vec{u}\cdot(\nabla\cdot\tau)+\nabla\vec{u}:\tau+\rho b\cdot\vec{u}+S \tag{17}$$

where

E is the total energy (internal + kinetic)

$k\nabla T$ is the heat flux given by Fourier's Law

S is the heat source

There are different possible versions of the energy equation. The form written above is known as the total energy form. Another form which describes the convection of internal energy and more commonly used is written as:

$$\frac{\partial(\rho e)}{\partial t}+\nabla\cdot(\rho e\vec{u})= \quad \nabla\cdot(k\nabla T)-p\nabla\cdot\vec{u}+\nabla\vec{u}:\tau+S \tag{18}$$

where

e is the internal energy substituted from the relation $E=e+\frac{1}{2}|\vec{u}|^2$

T is the temperature

k is the thermal conductivity of the fluid

$\nabla\vec{u}:\tau$ is the double dot product defined as $\frac{\partial u_j}{\partial x_i}\tau_{ij}$ describing the irreversible transfer of mechanical energy into heat

Another form known as the enthalpy form is written as:

$$\rho\frac{\partial h}{\partial t}+(\rho\vec{u}\cdot\nabla)h \quad =\nabla\cdot(k\nabla T)+\nabla\vec{u}:\tau+\frac{Dp}{Dt}+S \tag{19}$$

where

h is the enthalpy substituted from the relation $h=e+\frac{p}{\rho}$

$\frac{Dp}{Dt}$ is the energy transfer due to work done by compressive forces, that is, pressure work

The five equation sets described above, one continuity, three momentum and one energy equation contain unknowns: ρ, p, T, e, u, v, w. In order to obtain a closed form solution for this system of differential equations, two more equations are needed. For a compressible flow these equations come in the form of equations of state. They are stated as:

$$p = p(\rho, T) \tag{20}$$

and

$$e = e(\rho, T) \tag{21}$$

For an ideal gas, these equations take the form

$$p = \rho R T \tag{22}$$

and

$$e = C_v T \tag{23}$$

These relations give an additional equation for enthalpy given by $h = C_p T$. When substituted into the energy equation it simplifies to:

$$\rho \quad c_p \frac{DT}{Dt} \quad = \frac{Dp}{Dt} + \nabla \cdot (k \nabla T) + \quad \nabla \vec{u} : \tau + \mathrm{S} \tag{24}$$

Similitude and Non–Dimensional Numbers

This section describes the concepts of similitude and non-dimensional numbers and their importance in fluid mechanics.

Similitude is a concept that relates the behavior of an object in a given flow field with its behavior in a different flow field under different operating conditions. This concept allows you to compare any two scenarios (simulations or experiments) and determine the similarities in the flow fields between them.

This concept becomes useful when there is a prototype of a vehicle, aircraft wing or any such application to be tested under laboratory conditions. It is not always feasible to test the full scale prototype owing to the large size or requirement of large scale flow conditions. Scaled models allow testing of a prototype prior to its production which can greatly increase the efficiency of the development process.

The following similarities need to be met in order to achieve similitude.

Geometric Similarity

Geometric similarity refers to similarity relations between the linear dimensions of the test scale model and the prototype. This means the model has the same shape as the prototype. More precisely, the model can be superimposed on the prototype by geometric operations like scaling, translation, rotation and reflection.

Kinematic Similarity

Kinematic similarity refers to the similarity of motion between the model and prototype in a fluid. This means that the flow streamlines are similar for the model and the prototype. More specifically, the velocities at corresponding points are in the same direction and are related in magnitude by a constant scale factor.

Dynamic Similarity

Dynamic similarity refers to the similarity between the force ratios at corresponding points and boundaries of the model and the prototype. This means that force distributions of the same type on the model and prototype have the same direction and are related in magnitude by a constant scale factor.

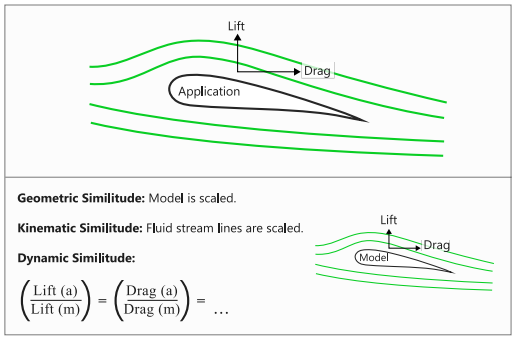


Figure 3: Similitude Between a Model and a Prototype

In order to analyse the above similarities certain non-dimensional numbers are used. The use of these numbers drastically simplifies the task of inferring the experimental data from the model and applying it to prototype design. Another advantage is that these numbers are independent of units of measurement and hence can be interpreted to suit any measurement system.

The most widely used non-dimensional numbers in fluid flow analysis are the following.

Reynolds Number

Reynolds number is the ratio of inertial (momentum) forces to viscous forces. It is an important parameter that can be used to determine the dynamic similarity between two cases of fluid flow or it can be used to characterize the flow regime in a fluid, this is, the laminar or turbulent nature of the flow. It is also used to study the transition between laminar and turbulent flows.

A laminar flow occurs at low Reynolds number when the viscous forces dominate resulting in a smooth velocity and pressure field where as a turbulent flow occurs at high Reynolds number where the inertial forces dominate the flow resulting in instabilities having a large range of time and length scales.

It is defined as

$$\mathrm{Re} = \frac{\rho u L}{\mu} \tag{25}$$

where

- ρ is the density of the fluid
- u is the mean velocity of the flow
- L is the characteristic length
- μ is the dynamic viscosity of the fluid

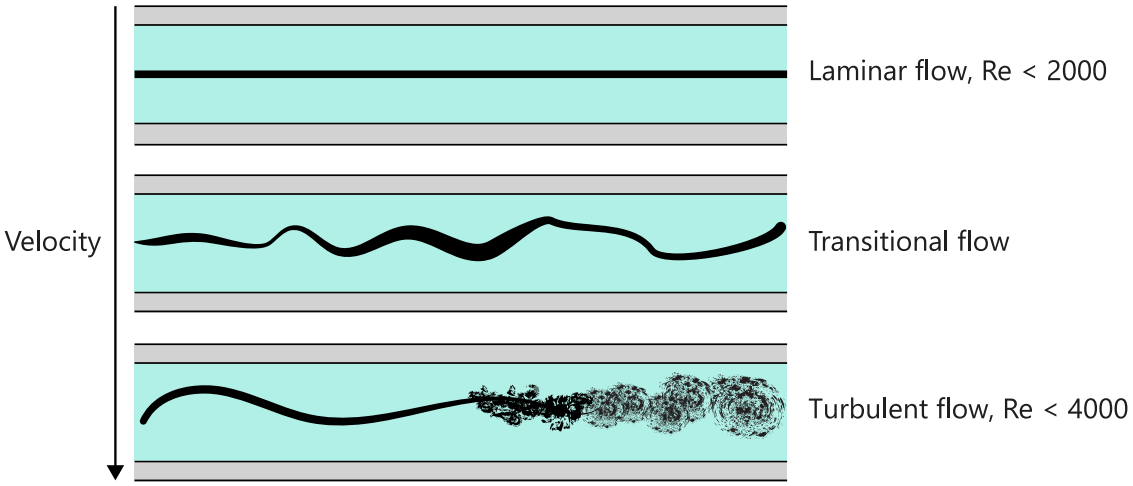


Figure 4: Flow Characterization Based on Reynolds Number

The Reynolds number can also be expressed as the ratio of total momentum transfer to molecular momentum transfer.

The characteristic length is geometry dependent and defines the scale of physical system. For example, the characteristic length for a flow in a pipe with circular, square or annular cross section is taken as the hydraulic diameter of the pipe.

The hydraulic diameter for a pipe is defined as

$$D_H = \frac{4A}{P} \tag{26}$$

where

A is the area of the cross section

P is the wetted perimeter, that is, the perimeter of the pipe in contact with the flow

For a fluid moving between two plane parallel surfaces, where the width is much greater than the space between the plates, the characteristic dimension is twice the distance between the plates.

For an airfoil the characteristic length is considered to be the chord length.

There is no universal definition for a characteristic length. Any geometric length that significantly simplifies the governing equations when converted to their non-dimensional form is considered the characteristic length.

Mach Number

Mach number is the ratio of flow velocity to local speed of sound. It is a key parameter that characterizes the compressibility effects in a fluid flow.

When the Mach number is small (less than 0.3) the inertial forces experienced by the flow can be considered sufficiently small to not cause any changes to the fluid density. Under these conditions the compressibility effects of the fluid can be ignored.

It is written as

$$M = \frac{u}{a} \tag{27}$$

The local speed of sound and hence the Mach number depend on the flow conditions, particularly on temperature and pressure of the fluid.

The Mach number is used to classify the flow regime into subsonic, transonic, supersonic and hypersonic flows, and they are defined as:

Subsonic flow $M < 0.8$

Transonic flow $0.8 < M < 1.0$

Supersonic flow $1.0 < M < 5.0$

Hypersonic flow $M > 5.0$

Knudsen Number

Knudsen number is the ratio of the molecular mean free path length to the representative physical length scale (characteristic length). It is a measure of rarefaction of the flow. Knudsen number determines whether continuum mechanics or statistical mechanics formulation should be used to model the flow.

It is expressed as

$$Kn = \frac{\lambda}{L} \tag{28}$$

where

λ is the molecular mean free path understood as the mean distance travelled by a molecule between two successive collisions with other molecules

L is the characteristic length

Prandtl Number

Prandtl number is the ratio of molecular diffusivity to thermal diffusivity in a fluid. When the Prandtl number is low it means that the heat diffuses quickly compared to momentum and vice versa.

It is written as

$$Pr = \frac{c_p \mu}{k} \tag{29}$$

where

c_p is the molar heat capacity of the fluid at constant pressure

μ is the dynamic viscosity of the fluid

k is the thermal conductivity of the fluid

Nusselt Number

The average Nusselt number is the ratio of convective heat transfer across a chosen boundary or surface to conductive heat transfer within a fluid. The convection heat transfer includes both advection and diffusion heat transfer.

It is defined as

$$Nu = \frac{hL}{k} \tag{30}$$

where

h is the convective heat transfer coefficient

L is the characteristic length

k is the thermal conductivity of the fluid

A Nusselt number close to one is characteristic of a laminar flow whereas a large value corresponds to turbulent flow.

The local Nusslet number at a point at a distance x from the boundary is expressed as

$$Nu_x = \frac{h_x x}{k} \tag{31}$$

Pressure Coefficient

The pressure coefficient is the ratio of pressure difference to the dynamic pressure. It describes the relative pressure across a fluid field. It is expressed as

$$C_p = \frac{p - p_\infty}{\frac{1}{2}\rho_\infty u_\infty^2} \tag{32}$$

where

p is the point at which pressure coefficient is being calculated

p_∞, ρ_∞, u_∞ are the free stream pressure, density and velocity, respectively

Lift and Drag Coefficients

The lift coefficient is the ratio of lift force to the dynamic force experienced by a body in a flow field.

It is expressed as

$$C_L = \frac{L}{qA} = \frac{2L}{\rho_\infty u_\infty^2 A} \tag{33}$$

The drag coefficient is the ratio of drag force to the dynamic force experienced by a body in a flow field. It is expressed as

$$C_D = \frac{D}{qA} = \frac{2D}{\rho_\infty u_\infty^2 A} \tag{34}$$

where

L is the lift force experienced by the body

D is the drag force experienced by the body

q is the dynamic pressure of the fluid defined as $q = \frac{1}{2}\rho_\infty u_\infty^2$

A is the reference area of the body

The reference area depends on the type of coefficient being measured. For automobiles and many bluff bodies the reference area while calculating the drag coefficient is taken as the projected frontal area. For airfoils the reference area is taken as the normal wing area.

The non-dimensional numbers used to determine similitude depend on the type of flow. For example, in a case of incompressible flow (without free surface), the Reynolds number can satisfy the similarity condition. If the flow is compressible, Reynolds number, Mach number and specific heat ratio are required in order to establish similarity.

It is not always possible to achieve absolute similitude between a model and prototype. It becomes even harder as you diverge from the prototype's operating conditions, therefore it becomes important to focus on only the most important parameters.

The following table lists the most frequently encountered non-dimensional number, their description and applications.

Table 1: List of Frequenty Used Non-Dimensional Numbers

Name	Symbol	Numerator	Denominator	Formula	Applications
Reynolds number	Re	Intertial Force	Viscous force	$\frac{\rho u L}{\mu}$	Fluid flow with viscous and inertial forces
Froude number	*Fr*	Intertial Force	Gravitational force	$\frac{u}{\sqrt{(gL)}}$	Fluid flow with free surfaces
Weber number	*We*	Intertial Force	Surface force	$\frac{\rho u^2 L}{\sigma}$	Fluid flow with interfacial forces
Mach number	*M*	Local velocity	Local speed of sound	$\frac{u}{a}$	Gas flow at high velocity

Name	Symbol	Numerator	Denominator	Formula	Applications
Prandtl number	Pr	Viscous diffusion rate	Thermal diffusion rate	$\frac{c_p \mu}{k}$	Fluid flow with heat transfer
Nusselt number	Nu	Convective heat transfer	Conductive heat transfer	$\frac{hL}{k}$	Fluid flow with heat transfer
Grashof number	Gr_L	Buoyancy force	Viscous force	$\frac{g\beta(T_s - T_\infty)D^3}{\mu^2}$	Fluid flow with natural convection
Rayleigh number	Ra_x	(Buoyancy force) * (Viscous diffusion rate)	(Viscous force) * (Thermal diffusion rate)	$\frac{g\beta(T_s - T_\infty)x^3}{\mu\alpha}$	Buoyancy driven flow
Specific Heat ratio	γ	Enthalpy	Internal Energy	$\frac{c_p}{c_v}$	Compressible flow
Pressure Coefficient	C_p	Local pressure	Dynamic pressure	$\frac{p - p_\infty}{\frac{1}{2}\rho_\infty v_\infty^2}$	Pressure drop estimation
Lift Coefficient	C_L	Lift Force	Dynamic Force	$\frac{L}{qA}$	Aerodynamics, Hydrodynamics
Drag Coefficient	C_D	Drag Force	Dynamic Force	$\frac{D}{qA}$	Aerodynamics, Hydrodynamics
Skin friction Coefficient	C_f	Wall Shear Force	Dynamic Force	$\frac{\tau_w}{qA}$	Aerodynamics, Hydrodynamics
Knudsen number	Kn	Molecular mean free path	Characteristic length	$Kn = \frac{\lambda}{L}$	Determination of applicability of continuum mechanics. i.e. suitability of AcuSolve for the application.

Simplification of Governing Equations (Different Types of Flow Models)

This section describes the simplification of the governing equations to various flow models by assumptions on time dependence, density and viscosity of the fluid flow.

Depending on the flow conditions the Navier-Stokes equations, primarily the momentum equations, can be simplified. The simplification of these equations depends on which effects in the flow are significant or insignificant.

The simplified flow models that are most broadly used are the following:

- Steady flow
- Euler flow or Inviscid flow
- Stokes flow
- Incompressible flow

Steady Flow

The time dependence of flow field parameters is an important factor in the analysis of a fluid flow. A majority of flows are not steady but transient in nature. In a steady state flow the flow properties at a point, such as pressure and velocity, do not change with time. Steady state flows are of interest in cases where the flow properties need to be studied after the flow field has stabilized.

This is achieved in simulations by taking a very large time step (Δt) which causes the time derivative of properties in the governing equations to reach to zero.

$$\nabla \cdot (\rho \vec{u}) = 0 \tag{35}$$

$$(\rho \vec{u} \cdot \nabla)\vec{u} = -\nabla p + \rho b + \nabla \cdot \tau \tag{36}$$

$$(\rho \vec{u} \cdot \nabla)h = \nabla \cdot (k \nabla T) + \nabla \vec{u} : \tau + \frac{Dp}{Dt} + S \tag{37}$$

While performing simulations, a steady state flow result gives a preliminary insight into whether the problem is set up correctly or not. If there are a large number of oscillations in the residuals it can be inferred that the flow is transient and not steady. If the residuals show a smooth converge it implies that the flow field becomes stable and steady state is achieved. The results can also be used to fine tune the setup for further simulations or use them as initial conditions for a transient simulation.

Euler Flow or Inviscid Flow

Inviscid flow is a representation of a fluid flow where the dissipative and transport phenomenon of viscosity, mass diffusion and thermal diffusion are neglected. This assumption is valid when the viscous forces are small in comparison to the inertial forces.

Such flow situations can be identified in cases with a high value of Reynolds number, where the viscous effects are concentrated to regions close to solid boundary and can be neglected for the regions far away from the boundary.

The governing equations for such flows are expressed as:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) = 0 \quad (38)$$

$$\rho \frac{\partial \vec{u}}{\partial t} + (\rho \vec{u} \cdot \nabla)\vec{u} = -\nabla p + \rho b \quad (39)$$

$$\rho \frac{\partial h}{\partial t} + (\rho \vec{u} \cdot \nabla)h = 0 \quad (40)$$

These equations can form a closed form solution by assuming the equation of state.

Results obtained from these assumptions in the flow field are widely used in designing flying vehicles, rockets and their engines, turbines and compressors. Studies of inviscid flows are carried out in gas dynamics, acoustics, electro and magneto gas dynamics, the dynamics of rarefied gases, plasma dynamics, and so on.

Stokes Flow

Stokes flow is a representation of fluid flow where the viscosity of the fluid is high. In such flows the viscous effects dominate the advective inertial effects.

The Reynolds number in such flows is low, hence, it is also termed as creeping flow or Low Reynolds number flow.

The governing equations for such flows are expressed as:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) = 0 \quad (41)$$

$$\rho \frac{\partial \vec{u}}{\partial t} = -\nabla p + \rho b + \nabla \cdot \tau \quad (42)$$

$$\rho \frac{\partial h}{\partial t} = \rho s + \nabla \cdot q \quad (43)$$

This results in the linearization of the governing equations and thus they can be solved by a number of linear differential solvers available.

If the governing equations are non-dimensionalized and the Reynolds number is assumed to be very low, the momentum equation reduces to

$$\mu \nabla^2 u = \nabla p - f \quad (44)$$

where

$$f \quad (45)$$

is the external force.

Incompressible Flow

All fluids (gas or liquid) exhibit some change in volume when subjected to compressive stresses. The degree of compressibility for a fluid can be quantified using the bulk modulus of elasticity, E, defined as

$$E = \frac{dp}{\frac{d\rho}{\rho}} \quad or \quad \frac{dp}{-\frac{dV}{V}} \quad (46)$$

where dp is the change in pressure and $d\rho$ or dV is the corresponding change in volume or density.

A flow can be classified as incompressible if the density within fluid particle does not change during its motion. It is also termed as isochoric flow and implies that under certain conditions a compressible fluid can undergoincompressible flow.

$$\frac{D\rho}{Dt} = \frac{\partial \rho}{\partial t} + \vec{u} \cdot \nabla \rho = 0 \tag{47}$$

Incompressibility is a property of flow and not of the fluid itself. Therefore the density field does not need to be uniform for the flow to be incompressible. When the above equation is combined with the continuity equation the following relation is obtained:

$$\nabla \cdot \vec{u} = 0 \tag{48}$$

which states that in an incompressible flow the velocity field is solenoidal (having zero divergence).

When an incompressibility assumption is made it is important to know under what conditions this assumption is valid. For a steady flow the condition is that the flow velocity must be much smaller compared to the local speed of sound, that is, $u \ll a$.

In case of an unsteady flow there is an additional condition which needs to satisfied, stated as $at \ll 1$. Physically this condition states the distance travelled by a sound wave in time t must be much greater than the distance travelled by the fluid particle. This implies that the propagation of pressure signals (sound waves) is instantaneous compared to the interval over which the flow field changes significantly.

When the limit for maximum relative change in density is set to five percent as the criteria for an incompressible flow, the maximum value of Mach number achieved is 0.3. This criteria states that any flow with a Mach number less than 0.3 (without heat source) can be assumed to be incompressible.

One of the implications of assuming the flow to be incompressible is that there is no equation of state as in a case of a compressible flow. In practice this means that the energy equation is decoupled from the continuity and momentum equations, assuming fluid properties are not a function of temperature. If the fluid properties change with temperature the equations again become coupled as in the case of a compressible flow.

The pressure in such a flow is no longer a thermodynamic quantity and cannot be related to temperature or density through an equation of state and must be obtained from the continuity and momentum equation while satisfying zero divergence for the velocity field. In the continuity equation there is no pressure term and in the momentum equation there are only the derivatives of pressure, but not the pressure itself. This means that the actual value of pressure in an incompressible flow solution is not important, only the changes of pressure in space are important.

Basic Boundary Layer Theory

This section covers concepts such as boundary layer type, flow, separation and transition.

Velocity Boundary Layer

Velocity boundary layer or commonly referred to as boundary layer is a thin layer of fluid formed near a boundary surface where the viscous effects of the fluid are significant.

When a fluid moves past an object or an object moves past a fluid, the fluid molecules right next to the boundary surface stick to it. This causes the molecules just above the surface to slow down because of collision with the molecules sticking to the surface. These molecules consequently slow down the flow just above them. As we move farther away from the surface of the object, the collisions affected by the object's surface reduces. Thus a layer of fluid is created within which the velocity of the flow gradually changes from zero at the object's surface to the free stream velocity at a certain distance away from the surface.

This distance at which the flow velocity inside the boundary essentially reaches the free stream velocity is termed as the boundary layer thickness. Mathematically it is defined as the distance at which the flow velocity is 99 percent of the free stream velocity.

$$\text{At } y = \delta, u = 0.99u_\infty$$

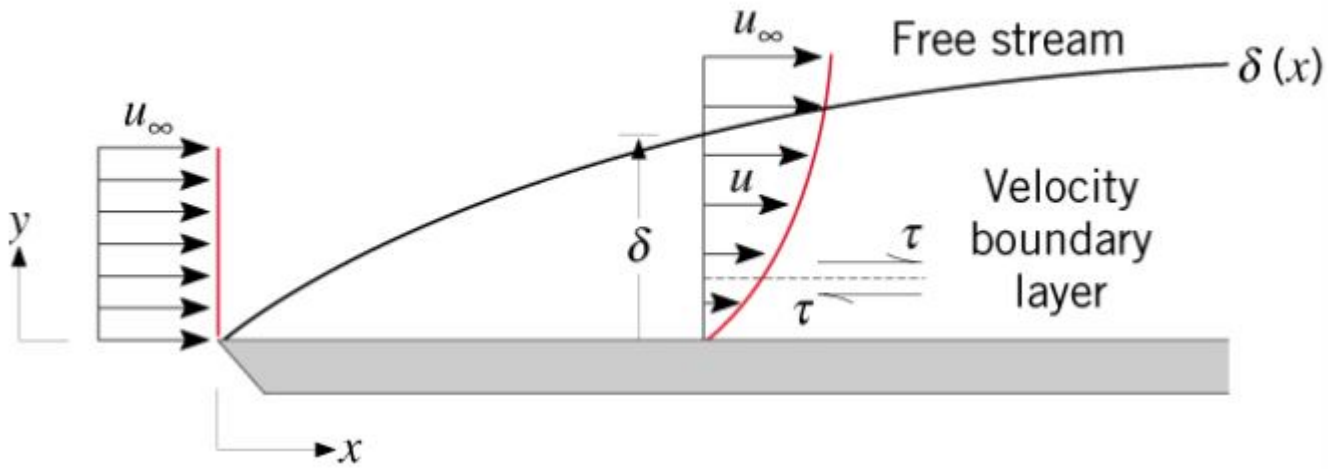


Figure 5: Velocity Boundary Layer

As a result of the loss of velocity, a shear stress is imparted on the object surface defined as

$$\tau_s = \mu \frac{\partial u}{\partial y}\bigg|_{y=0} \qquad (49)$$

Thermal Boundary Layer

Thermal boundary layer is a region of fluid flow near a solid surface characterized by temperature gradients and heat fluxes. It is a consequence of heat transfer between the fluid and the solid surface and the fluid temperatures are directly influenced by heating or cooling from the surface wall.

The distance at which the difference between the flow temperatures inside the boundary layer and the boundary surface essentially reaches the difference between free stream temperature and the boundary surface temperature is termed as the thermal boundary layer thickness. It is mathematically expressed as:

At y $=\delta_t$, $\frac{T_s - T}{T_s - T_\infty}$ = 0.99

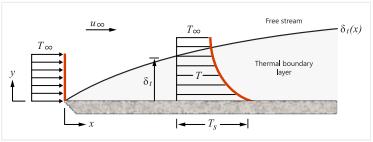


Figure 6: Thermal Boundary Layer

The local heat flux in a thermal boundary layer is analogous to the shear stress in a velocity boundary layer. The heat flux at the surface is proportional to the temperature gradient. It is defined as

$$q''_s = -k_f \left.\frac{\partial T}{\partial y}\right|_{y=0} \tag{50}$$

The flow inside the velocity boundary layer can be classified into two types

- Laminar boundary layer flow
- Turbulent boundary layer flow

Laminar Boundary Layer Flow

The flow inside a laminar boundary layer is smooth and takes place in layers. Each layer slides past the adjacent layers and any exchange of mass or momentum takes place only between these layers on a microscopic scale. Laminar boundary layer is formed when the Reynolds number is low.

The shear stresses associated with this flow can be calculated using the molecular viscosity μ.

Figure 7 depicts the typical velocity profile for a laminar boundary layer.

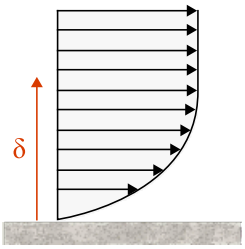


Figure 7: Laminar Boundary Layer

Turbulent Boundary Layer Flow

As the name suggests turbulent boundary layer flow is more chaotic and is characterized by active mixing of the fluid and momentum across several layers. The exchange of mass, momentum and energy takes place on a macroscopic scale with packets of fluid moving across the layers. A turbulent boundary layer is formed only at high Reynolds number.

In this case the scales of mixing cannot be handled by molecular viscosity alone, therefore turbulent or eddy viscosity has to be modeled in order to proceed with the calculations.

Figure 8 depicts the velocity profile for the turbulent boundary layer.

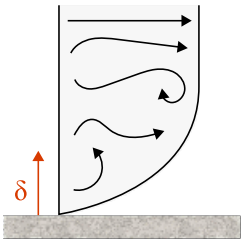


Figure 8: Turbulent Boundary Layer

The velocity profile for a turbulent boundary layer is quite different from a laminar boundary layer. It comprises of three regions or layers:

- Outer layer: This layer is sensitive to the properties of the external flow.
- Inner layer: This layer has turbulent mixing as the dominant physics.
- Laminar or viscous sublayer: This layer is attached to the wall where the no slip condition is applied. The shear stress in this layer is dominated by molecular viscosity.

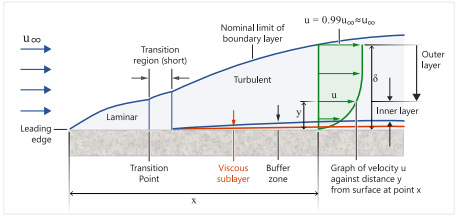


Figure 9: Complete Boundary Layer Profile (Laminar and Turbulent)

These layers blend into each other and the blending region between the inner layer and laminar sublayer is termed as the Buffer Zone.

As a consequence of intense mixing, a turbulent boundary layer has a much steeper gradient of velocity resulting in a larger shear stress and higher skin friction drag at the wall compared to a laminar boundary layer. In addition heat transfer rates are higher for a turbulent boundary layer. The presence of eddies brings fluid particles at different temperatures into close contact with each other and the heat transfer comes more effective.

Another consequence of mixing is that a turbulent boundary layer has higher momentum near the wall, so it can withstand adverse pressure gradient longer and is less easily driven by the changes in the free stream pressure compared to a laminar boundary layer. This property allows a turbulent boundary layer to be attached to the object surface longer.

Boundary Layer Separation

Boundary layer separation is defined as the detachment of the boundary layer from the surface of the object thus leading to formation of eddies and vortices in the wake. The most important factor which leads to boundary layer separation is an adverse pressure gradient.

The pressure gradient is one of the parameters that greatly influences a flow. The shear stress caused by viscosity, near an object surface, has a retarding effect on the flow. This retardation can be overcome by a implementing a negative pressure gradient in the direction of the flow. A negative pressure gradient is termed as a favourable pressure gradient because it enables the flow.

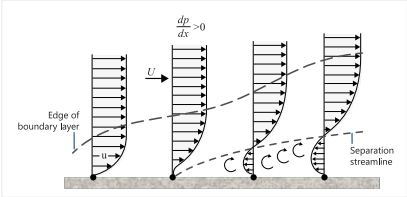


Figure 10: Boundary Layer Separation

A positive pressure gradient, termed as adverse pressure gradient, has the opposite effect on the flow. It further retards the flow leading to a reduction in the velocity near the wall and an increased boundary layer thickness. This continuous retardation causes the shear stress at the wall to drop to zero which then becomes negative causing a reversal in the flow direction near the surface. A region of recirculating flow is developed with formation of large slowly rotating eddies resulting in the flow detaching from the surface of the object. The point at which the velocity gradient and essentially the shear stress reaches zero is termed as the point of separation.

The separation leads to the formation of a large wake of vortices behind the body, where the fluid pressure is much lower compared to the regions where the fluid is attached to the surface since the eddies cannot convert the rotational energy into pressure head. As described in the previous section the flow velocity near the object surface is higher in case of a turbulent boundary layer. This higher fluid momentum means that the fluid separation occurs further downstream than in case of a laminar boundary layer when subject to the same adverse pressure gradient, resulting in a narrower wake and consequently less pressure drag.

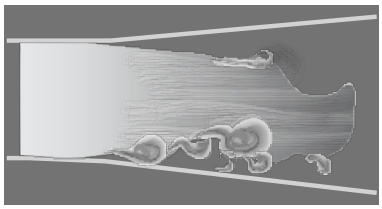


Figure 11: Boundary Layer Separation in a Diverging Channel (source: National committee for Fluid Mechanics Films; http://web.mit.edu/hml/ncfmf.html)

Figure 11 shows the separation of a laminar (top) and turbulent boundary (bottom) layer in a diverging channel.

Depending on the flow conditions, the recirculating flow may reattach to the surface of the body. A number of factors could influence the reattachment. The pressure gradient may turn favourable due to changing flow conditions or the geometry of the body. The other factor is that the boundary layer may transition from laminar to turbulent resulting in a fuller velocity profile that could sustain the adverse pressure gradient and causing the flow to reattach.

Boundary Layer Transition

Boundary layer transition is a very complex process and may be triggered by various types of flow disturbances, which can be due to the following factors:

- Free stream turbulence
- Acoustic Noise (Pressure fluctuations)
- Pressure Gradient
- Surface thermal disturbances
- Surface curvature
- Surface roughness
- Compressibility and high speed flow (transonic, supersonic and hypersonic) transition

These factors along with many other sources of disturbance can cause a smooth laminar flow to break down, enter a transition region and then develop into a turbulent flow. The mechanisms through which a boundary layer transitions can be one of the following:

- Natural transition: This mechanism is prevalent in weak disturbance environments (vibrations, free stream turbulence, adverse pressure gradient, and so on, are absent). The transition occurs gradually and breakdown is reached thorough a linear growth of disturbances. This mechanism is most commonly observed in the flow around a transport aircraft in flight.
- Bypass transition: This mechanism is observed in cases where the initial disturbances in the laminar flow are sufficiently high, caused by factors like surface roughness or high free stream turbulence. The highly non-linear growth of disturbances results in the quick formation of turbulent spots in the flow leading to breakdown, thus bypassing the linear mechanism. This transition

mechanism is observed in flows related to turbomachinery where there is high free stream turbulence present.

- Intermediate mechanisms: Aside from the primary mechanisms of turbulence discussed above, there are additional intermediate mechanisms. They include spanwise modulations in the flow, distortions leading to secondary instabilities and direct bypass.

Importance and Practical Applications of Boundary Layer

The properties associated with boundary layer flow, heat transfer, transition and separation are of great importance. Their application can be found in fields such as aerodynamics, hydrodynamics, meteorology, and so on. The following sub-sections describe the areas of special interest in the various fields which have significant practical applications.

- Flow over an airfoil: In the aerodynamics industry, the boundary layer is particularly important because it is responsible for a considerable amount of drag on the surface. The airfoils are designed to be thin and streamlined in order to keep the boundary layer laminar, thus reducing the skin friction drag. In situations where there are higher chances of flow separation, it is always preferable that the boundary layer separates towards the leading edge of the airfoil and reattaches somewhere downstream. It is dangerous if the separation occurs near the trailing edge in which case the flow would not reattach, resulting in sudden loss of lift and stalling of the aircraft. There are several approaches used to prevent this problem. These approaches include turbulence generation by tripping the boundary layer using vortex generators, boundary layer suction at the leading edge and reenergizing the boundary layer through blowing.
- Heat exchangers: A large number of production facilities in many industries use processes in which heat transfer takes place within the fluid or between different fluids. In such cases it is always preferable to increase the heat transfer and mixing without paying too high a penalty in terms of increased pressure loss. Some of the approaches used for this purpose include use of internal ribs or fins to increase the internal surface area of the domain, deformation in the flow domain to decrease the hydraulic diameter and disruption of the boundary layer to induce turbulence.
- CFD: The approximations obtained from the thin boundary layer theory greatly reduce the complexity of the equations solved. Mathematically the application of boundary layer theory changes the character of the governing Navier-Stokes equation from elliptic to parabolic. This allows marching in the flow direction, as the solution at any location is independent of the conditions farther downstream.

Aside from the topics discussed above, there are numerous cases where the boundary layer and the physics associated with it have significant applications. A few examples are natural boundary layer in the atmosphere, biometrics of marine organisms, automobile aerodynamics and design of sports merchandise such as golf balls.

Turbulence

This section on turbulence covers the topics describing the physics of turbulence and turbulent flow. It also covers the modeling of turbulence with brief descriptions of commonly used turbulence models.

Physics of Turbulent Flows

This section on physics of turbulence introduces a brief history of turbulence and covers the theory behind turbulence generation, turbulence transition and energy cascade in fluid flows.

On a daily basis you encounter turbulent flows in many places including rivers, ocean currents, atmospheric turbulence, bush fires, flows over blunt bodies, for example, cars, aircraft, ships and buildings, cross flows of chimney plumes, internal flows in pipelines, turbines and engines and many other various forms. Despite the prevalence of turbulent flows in your daily life there is no consensus on the definition of turbulence.

The following are several definitions quoted from various sources of literature.

- Bradshaw (1971): “Turbulence is a three dimensional time dependent motion in which vortex stretching causes velocity fluctuations to spread to all wavelengths between a minimum determined by viscous forces and a maximum determined by the boundary conditions. It is the usual state of fluid motion except at low Reynolds numbers.”
- Hinze (1975): “Turbulent flow motion is an irregular condition of flow in which various quantities show a random variation with time and space coordinates, so that statistically distinct average values can be discerned.”
- Launder (1991): “At moderate Reynolds numbers the restraining effects of viscosity are too weak to prevent small, random disturbances in a shear flow from amplifying. The disturbances grow, become non linear and interact with neighboring disturbances. This mutual interaction leads to a tangling of vorticity filaments. Eventually the flow reaches a chaotic, non repeating form describable only in statistical terms. This is the turbulent flow.”
- Pope (2000): "An essential feature of turbulent flows is that the fluid velocity field varies significantly and irregularly in both position and time."

Considering these definitions you can arrive at the following set of characteristic features of turbulent flows:

- Irregularity. Turbulent flow is highly chaotic and irregular in both space and time (Tennekes and Lumley, 1972; Hinze, 1975; Pope, 2000).
- Non-repetition. Turbulent flow is a non-repeating form (Launder, 1991).
- Three dimensional unsteady eddies with different scales. Turbulent flow is always three dimensional and unsteady. Turbulent flow contains eddies with different scales, which extract energy from shear stress of the mean flow and produce fluctuations in the velocity and pressure fields. The eddies also interact with one another and exchange momentum and energy (Bradshaw, 1971; Tennekes and Lumley, 1972).
- Diffusivity. Turbulent flow is diffusive (that is, the spreading of fluctuations) (Tennekes and Lumley, 1972; Pope, 2000). The chaotic eddies (vortices) within the turbulent flow increase the momentum

exchange in boundary layers delaying separation on the surface of curved bodies and increasing the resistance (wall friction) and heat transfer.

- Dissipation. Turbulent flow is dissipative. This suggests that the kinetic energy within small eddies is transformed into thermal energy. The small eddies receive their energy from larger eddies, which receive their energy from even larger eddies, and so on. The largest eddies extract their energy from the mean flow. This process of transferring energy from the larger eddies to the smaller eddies is called the energy cascade process. Turbulence is not self sustainable without a source of energy input.
- Amplification. The flow disturbances amplify when the Reynolds number exceeds critical Reynolds numbers (~2300 for internal flows, ~500000 for external flows).
- Flow property. Turbulent flow is a property of flow, it is not a fluid property.
- Sensitivity. Turbulence is extremely sensitive to flow disturbances (Pope, 2000).

The First Visualization of Turbulent Flow

Leonardo da Vinci was the first one to report the visualization of turbulent flow in his famous sketches from 1510.

He described the existence of whirlpools of water in his notes. He wrote, “Observe the motion of the surface of the water, which resembles that of hair, which has two motions, of which one is caused by the weight of the hair, the other by the direction of the curls; thus the water has eddying motions, one part of which is due to the principal current, the other to random and reverse motion.” This notion is a precursor to the Reynolds flow decomposition of velocity into mean and fluctuating parts, which Osborne Reynolds suggested 400 years later. He also noted, “The small eddies are almost numberless, and large things are rotated only by large eddies and not by small ones and small things are turned by both small eddies and large.” Nowadays, the “small eddies” and the “large things” refer to turbulence and large-scale coherent eddies, respectively. Leonardo da Vinci utilized his sketches as a flow visualization method, providing clear information of turbulence behaviors and their effects.

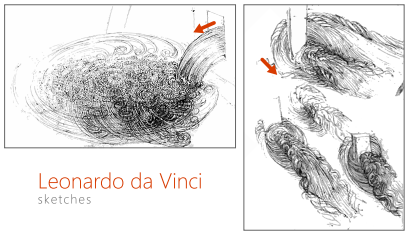


Figure 12: Leonardo da Vinci’s Sketches: Whirlpools of the Water (left), Vortices Behind Obstacles (right): Arrows Indicate Flow Directions

The Reynolds Measurement

In 1883, Osborne Reynolds conducted an experiment where dye was injected into a glass tube with water flowing due to gravity.

He studied the conditions of transition from laminar flow to turbulent flow, indicated by the flow patterns in the tube. He analyzed that the flow patterns differed as the flow velocity was changed. At low speed, the dye showed regular straight lines of a laminar flow pattern. When the flow velocity increased, the dye became random (or chaotic) with eddies filling the entire tube, and mixed with water. This random flow pattern is a turbulent flow. He further noticed that the appearance of turbulent flow highly depended on a dimensionless parameter of velocity and length scale, as well as viscosity. This parameter is now known as the Reynolds number.

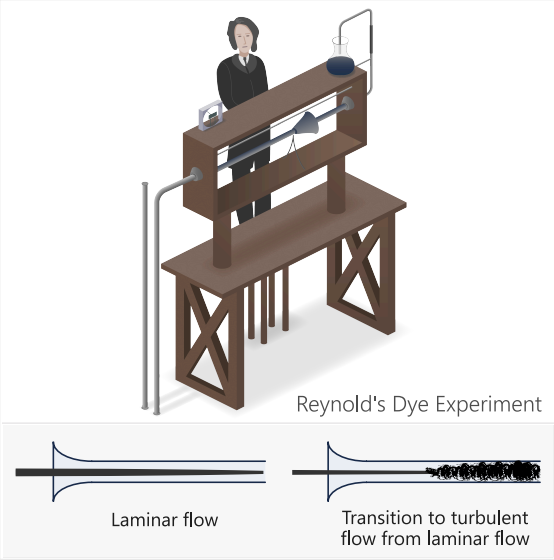


Figure 13: The Dye Experiment Conducted by Osborne Reynolds

The Reynolds Number

The Reynolds number is not only used to characterize the flow patterns, such as laminar or turbulent flow, but also to determine the dynamic similitude between two different flow cases.

The latter is an important concept to ensure valid experimental or computational results of numerous scaled models for industrial applications.

The Reynolds number is the ratio of the inertial force to the viscous force, defined as

$$\mathrm{Re}=\frac{\rho u L}{\mu} \tag{51}$$

where ρ is the fluid density (kg/m3), u the mean flow velocity (m/s), L the characteristic length (m), for example, hydraulic diameter for internal flows, and μ the dynamic viscosity of the fluid (kg/(m-s).

The characteristic length in the Reynolds number can be anything convenient, as long as it is consistent, especially when comparing different geometries. For example, the radius and the diameter are both valid for spheres or circles in flows with the diameter primarily used by convention. When computing the Reynolds number for airfoils and wings, the chord length is often chosen over the span because the

former is a representative function of the lift. For pipes, the diameter is the characteristic length. For rectangular pipes a suitable choice is the hydraulic diameter defined as

$$D_h = \frac{4A}{P} \tag{52}$$

where A is the cross-section area (m2) and P is the wetted perimeter (m). The wetted perimeter is the total perimeter of subject walls in contact with the flow.

Fluid flows are laminar when their Reynolds number is below a certain critical value and they are turbulent when they are larger than this critical value, termed the critical Reynolds number (Recr). This critical Reynolds number is between 2,300 and 4,000 for pipe flows.

The critical Reynolds number is also referred to as the transition Reynolds number. It varies widely depending on the conditions of surface roughness, flow disturbances, flow velocity and geometric considerations. For example, the critical Reynolds number for the boundary layer flow over a flat plate reaches about 500,000 based on the free stream velocity outside of the boundary layer and characteristic length of the distance from the leading edge of the plate.

Turbulent Flow Verses Laminar Flow

At low values of Reynolds number the viscous force is large compared to the inertial force.

In this range, viscous forcing dampens out disturbances in the flow field that are a result of surface roughness or pressure gradients. As the Reynolds number increases, the viscous force becomes relatively smaller and at some point it becomes possible for small perturbation to grow. The flows become unstable and can transition to turbulence where large fluctuations in the velocity field continue to develop.

Figure 14 shows two different flow patterns in pipes for laminar flow and turbulent flow. It is evident in the image that turbulent flow undergoes irregular flow patterns while laminar flow moves in smooth layers while maintaining a constant flow direction. The turbulent flows happen at a high Reynolds number where inertial forces are higher than viscous forces and perturbations can become amplified, whereas the laminar flows occur at a low Reynolds number in which any induced perturbations are damped out due to relatively strong viscous forces.

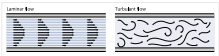


Figure 14: Flow Patterns

Figure 15 shows the time history of local velocity variations for laminar flows and turbulent flows, respectively. These patterns can be obtained from hot wire anemometer measurements or CFD simulations. For the pipe flow cases, laminar flows have nearly constant velocity, while turbulent flows have random or chaotic velocity fluctuation patterns. For a cylinder in cross flow, laminar flows produce a sine wave velocity pattern at a downstream location of the cylinder, while the turbulent flows have similar wave patterns but with embedded fluctuations.

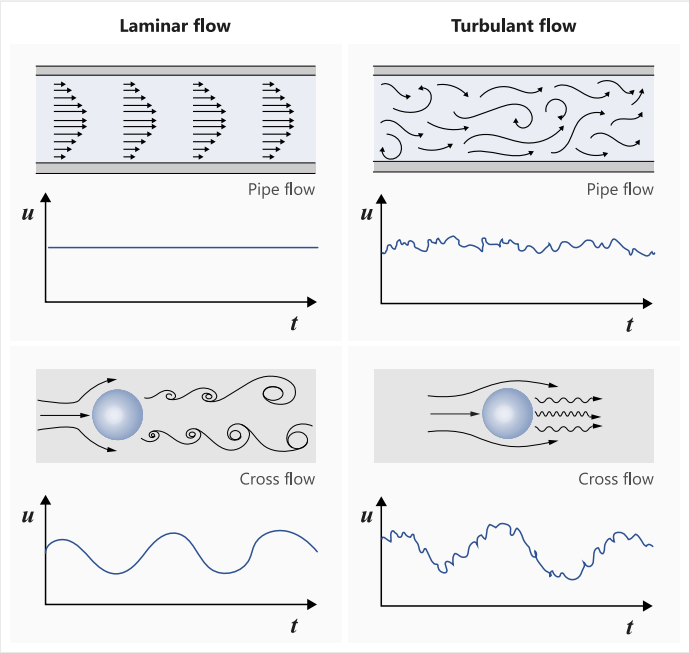


Figure 15: Time History of Instantaneous Velocity at a Local Point

If you consider a flat plate immersed in a flow field with finite viscosity, a thin boundary layer will begin to develop as a function of the distance traveled along the plate (x). The image below shows a comparison of time-averaged velocity profiles for laminar flow and turbulent flow over a flat plate. The time-averaged velocity profile in a turbulent flow appears more uniform than in a laminar flow because the eddy motions in turbulent flow transport momentum more actively from one place to another. This process results in a more uniform profile outside the boundary layer. The velocity gradient near the wall is higher than the one seen in a laminar flow resulting in a larger skin friction coefficient (C_f) than the laminar flow. The skin friction coefficient can be defined as

$$C_f = \frac{\tau_w}{\frac{1}{2}\rho U^2} \tag{53}$$

where $\tau_w = \mu \frac{\partial U}{\partial y}$ is the wall shear stress, μ is the dynamic viscosity, ρ is the density and U is the mean velocity. The local Reynolds number (Re_x) is calculated by using the distance from the leading edge of the flat plate as the length scale. Re_x measures the local ratio of inertial to viscous forcing and provides an indication of the state of the flow regime as it moves from laminar to turbulent.

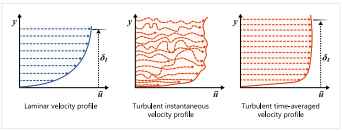


Figure 16: Time Averaged Velocity Profiles for Laminar and Turbulent Flows

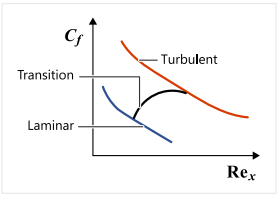


Figure 17: Time Averaged Skin Friction Coefficient for Laminar Flow and Turbulent Flow

The information below includes equations of boundary layer thickness and skin friction coefficient for turbulent flow and laminar flow. These equations where developed through empirical relationships between local Reynolds number and boundary layer characteristics. The information below also provides a summary of general characteristics that are present in turbulent flow and laminar flow.

Table 2: Laminar Flow vs Turbulent Flow: Boundary Layer Thickness and Skin Friction Coefficient

Laminar Flow	**Turbulent Flow**
Boundary layer thickness: $\delta_l = 4.91 \quad x/\sqrt{\mathrm{Re}_x}$	Boundary layer thickness: $\delta_t \cong 0.38 \quad x/(\mathrm{Re}_x)^{1/5}$
Skin friction coefficient: $C_{fl} = \quad 0.664/\sqrt{\mathrm{Re}_x}$	Skin friction coefficient: $C_{ft} \cong \quad 0.059/(\mathrm{Re}_x)^{1/5}$

Table 3: Laminar Flow vs Turbulent Flow

	Laminar Flow	**Turbulent Flow**
Reynolds Number	Low	High
Flow	High degree of orderness	Unsteady three-dimensional flow with random fluctuations

	Laminar Flow	Turbulent Flow
Disturbance	Damped due to high viscous forces	Amplified by inertia forces
Heat transfer	Low heat transfer	High heat transfer because of active mixing
Friction drag (wall shear stress)	Low (low velocity gradient near walls)	High (high velocity gradient near wall)
Separation	Separation can occur at weak adverse pressure gradient	High momentum flow near wall delay flow separation

Transition Flow

In the real world, laminar flow and turbulent flow coexist when obstacles are located inside of a fluid flow.

Industrial examples of the transition from laminar to turbulent flow regimes can be found in many different applications, as shown in Figure 18.

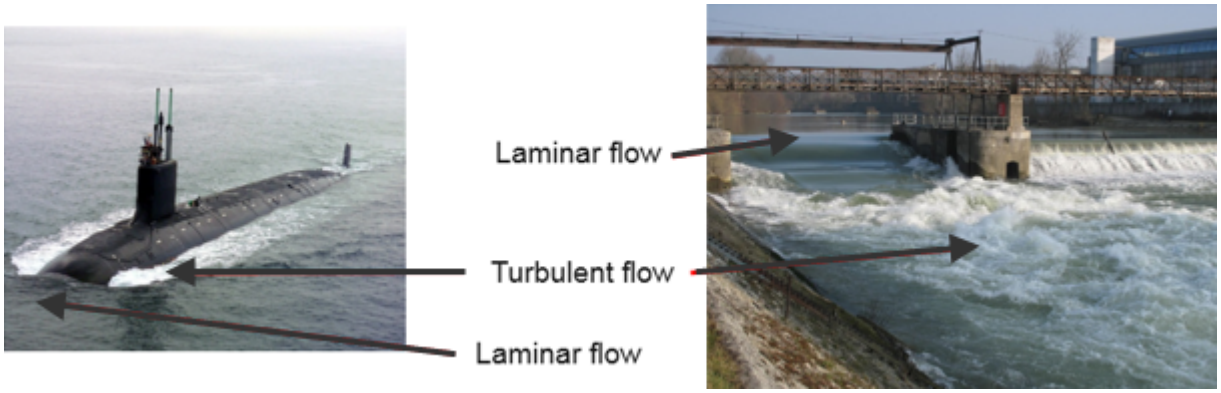


Figure 18: Submarine (left) and Dam (right)

A transition from laminar flow to turbulent flow occurs due to various external factors such as freestream turbulence level (vortical disturbances), sound waves (acoustic disturbances), temperature fluctuations (entropy disturbances), streamwise pressure gradients, surface roughness, surface curvature and vibration. Depending on the external factor or factors, the development of various transition (instability) mechanisms cause the flow to trip from laminar to turbulent. Since transition affects the flow development drag, heat transfer and many other factors it is important to understand the transition phenomena properly. The following section overviews the three main transition processes: natural transition, bypass transition and separation-induced transition.

Natural Transition

In the laminar flow regime viscous forces usually damp out the disturbances.

However, when the free stream turbulence is below one percent and the Reynolds number is higher than the critical Reynolds number, viscous forces destabilize the shear layer such that it becomes unstable two dimensional Tollmien-Schlichting (T-S) waves. These waves develop into three dimensional waves and hairpin vortices, which eventually break down, elongate, and roll up. They then develop into triangular turbulent spots past certain downstream locations. These turbulent spots grow by spreading sideways and burst into a turbulent flow.

Although T-S waves are the main instability mechanisms of the natural transition process for two dimensional boundary layers, such as flows over flat-plates, there are other kinds of instability mechanisms for natural transition. These include Görtler vortices and the crossflow instability.

Görtler vortices generate on any surfaces with concave curvatures from the effects by centrifugal forces. These are counter-rotating pairs of stationary streamwise vortices. Examples of industrial applications with concave surfaces include supercritical airfoils designed for laminar flow control, high speed wind tunnels, as well as forebody compression surfaces ahead of scramjet engine inlets. Details about these vortices are discussed in Saric (1994).

The third instability mechanism is crossflow instability. Crossflow instability develops when the pressure gradient over a swept wing combines with the cross flow from a wing root to a wing tip. It leads to the formation of three dimensional streamwise, co-rotating vortices, inducing transition.

Having an understanding of these instability mechanisms and controlling them with flow controls have received considerable attention over the past decade with the goal of delaying turbulent transition. This is beneficial in reducing the drag of slim bodies, for example wings, by keeping laminar flows as long as possible.

Bypass Transition

Another transition process is bypass transition. When the initial disturbances are high due to surface roughness or freestream turbulence levels higher than one percent, turbulent spots are generated without development of the three initial disturbances, T-S waves, spanwise vorticity and vortex breakdown, observed in the natural transition.

Although the bypass transition phenomena has been investigated in experimental and computational studies it has not been very well understood. A common industrial example of bypass transition is that of turbomachinery. The bypass transition occurs when blades in a gas turbine engine encounter wakes from upstream vanes.

Separation-Induced Transition

The third kind of transition process is separation-induced transition. When a laminar flow experiences adverse pressure gradients, such as airfoil suction surfaces and flow over a sphere, the fluid flow detaches from the wall surface.

If the disturbances are low, separation can cause the generation of structures found in natural transition. Larger disturbances generate Kelvin-Helmholtz instabilities, where vortices roll up before breaking down into turbulence. The process involved in separation-induced transition depends on

the size of the adverse pressure gradient and the presence of additional disturbances, for example obstacles.

Many balls have some type of surface roughness, such as the seams on baseballs and the fuzz on tennis balls. A rough surface can assist in reducing the drag by promoting an early transition.

A number of industrial applications take advantage of this observation for practical design. These include vortex generators on various types of airfoils, such as wind turbine blades, wings and fences on aircraft and automobiles, as well as wings enhanced with surface roughness.

Turbulent Wake

At the low Reynolds number, the downstream flow of the cylinder has a symmetric flow pattern and produces near zero pressure drag.

When flow reaches Re ~ 100,000 vortices begin to shed from each side of the cylinder. This particular shedding pattern is called a Von Kármán vortex street. When the Reynolds number is about 1,000,000 a turbulent wake begins to form behind the cylinder. It is generated by the adverse pressure gradient at the peak position of the cylinder. This forms a large separation region where vortical structures interact and decay downstream. As the Reynolds Number increases the size of the turbulent wake decreases which in turn results in the reduction of the drag. This mainly attributes to the turbulent boundary formation on the cylinder, delaying the flow separation as it has a higher momentum near the cylinder than laminar flow.

As mentioned before, vortex shedding forms behind the cylinder when the Reynolds Number is between 47 and 100,000. This is referred to as a Von Kármán vortex street and is caused by unsteady flow separation around the cylinder (see Figure 19). Since vortices induce low-pressure zones, the cylinder experiences lateral (sideways) forces. When the frequency of lateral forces is close to the natural frequency of the cylinder, unwanted structural vibration will occur. In order to remedy this issue, spirals are sometimes installed on objects subject to wind loading, such as a chimney.

Figure 19: Vortex Shedding Formation Downstream of a Cylinder (left) and Chimney with Spirals (right)

Blunt bodies experience strong pressure drag compared to skin friction. This occurs due to the pressure imbalance between the front cross-section of the blunt bodies and the downstream cross-section. This pressure imbalance is attributed to the formation of downstream turbulent wakes, which reduces the local pressure. The high-pressure drag is a determinant factor of the flying range of a ball. As an example, it is common practice to make dimples on the surface of golf balls so that turbulent boundary

layers form on the surface of a ball. This results in delayed flow separation and consequently reduces the size of the turbulent wake. Since the reduction of turbulent wake induces the smaller pressure drag, the golf ball with dimples should fly for a longer range when compared to the smooth ball.

Compared to the sphere in an airfoil at a low angle-of-attack (AOA) produces a smaller wake size, which results in the reduction of pressure drag. That is why the total drag is smaller for the airfoil than the ball, although the friction (viscous) drag is higher on the airfoil. Consequently, designers focus on the friction drag reduction by employing smooth surfaces. This is contrary to a ball with dimples for long flying range. However, since the turbulent wake size increases for high AOA, designers will shift their focus on reducing wake size by controlling flow on the airfoil suction side where it separates. Several methods are available, including a leading-edge extension, a vortex generator and a flow injection method.

The Navier-Stokes Equations

The physics of turbulent flows have been discussed by presenting experimental observations and comparing it to laminar flows. In this chapter, the focus will shift to the governing equations of these flow fields.

The Navier-Stokes (NS) equations were named after Claude-Louis Navier and George Gabriel Stokes. These equations govern the flow motion of a fluid and usually consist of the mass conservation equation and the momentum conservation equation. Sometimes, the momentum conservation equations alone represent the Navier-Stokes equations. The NS equations govern the behavior of a viscous fluid by balancing the forces with Newton's second law and assuming that the stress in the fluid is the sum of a diffusing viscous term. In the case of incompressible Newtonian fluid the instantaneous continuity equation is the mass conservation equation, which reads in vector notation as:

$$\nabla \cdot \vec{u} = 0 \tag{54}$$

where $\vec{u}$ is the velocity vector.

In tensor notation, the continuity equation is expressed as $\frac{\partial u_i}{\partial x_i} = 0$.

The instantaneous momentum equation using vector notation is written as

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} = \frac{-\nabla p}{\rho} + \frac{\nabla \cdot \tau}{\rho} \tag{55}$$

where ρ is the fluid density, p is the pressure and τ is the viscous stress tensor. In tensor notation, the incompressible momentum equation is given by

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = \frac{-1}{\rho} \frac{\partial p}{\partial x_i} + \frac{1}{\rho} \frac{\partial \tau_{ij}}{\partial x_j} \tag{56}$$

The left-hand side of the equation describes acceleration, which includes the unsteady term and convective term. The right-hand side of the equation represents the summation of pressure and the shear-stress divergence terms. The convective term is non-linear due to an acceleration associated with the change in velocity over position. This term can be disregarded in one-dimensional flow and Stokes flow (or creeping flow). For the Newtonian fluid, the viscous shear stress is assumed to be proportional to the shear strain rate. Thus, viscous stresses can be obtained by

In vector notation, $\tau = \mu\left(\nabla\vec{u} + \left(\nabla\vec{u}\right)^T\right) = \mu\nabla^2\vec{u}$

In tensor notation, $\tau_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)$

Turbulence Scales and Energy Cascade

Turbulence is composed of turbulent eddies of different sizes. At high Reynolds numbers, a scale separation exists between the largest eddies and smallest eddies.

The largest eddies extract kinetic energy from the mean flow as the energy production. The length scale is comparable to the flow dimensions. These processes are highly anisotropic and mostly are not influenced by viscosity. Most transport and mixing happen in this range.

The smallest eddies have universal characters independent of the flow geometry and conditions. Those eddies in this range usually receive energy from the larger eddies and dissipate their energy into heat through the fluid's molecular viscosity. These eddies are isotropic with length scales as described by Kolmogorov scales. It is assumed that the small scale eddies are determined by viscosity and dissipation.

The intermediate eddies are called Taylor length scale eddies with the length scales in the inertial subrange. Compared to the dissipation range, the turbulence in this region is also isotropic but the eddies in this region are independent of both the largest eddies and the smallest eddies in the dissipation range. It is assumed that the eddies in this region can be characterized by the turbulent dissipation ε and the wave number κ (or eddy size l). Through dimensional analysis you obtain

$$E = C_k \kappa^{-5/3} \varepsilon^{2/3} \tag{57}$$

where the Kolmogorov constant C_k = 1.5.

If the flow is fully turbulent flow at a high Reynolds number, the energy spectra should exhibit a -5/3 decay in the inertia region. This is called Kolmogorov spectrum law or the -5/3 law.

Richardson (1922) introduced the energy cascade concept, "Big whorls have little whorls that feed on their velocity; And little whorls have lesser and so on to viscosity."

The largest eddies with integral length scale are unstable and break up, transferring kinetic energy to smaller eddies. A similar break-up process with these smaller eddies transfer their energy to subsequent smaller eddies. These processes continue until the eddy size is reduced to the smallest size and the eddy motion is stable. This process is known as the energy cascade, as shown in Figure 20.

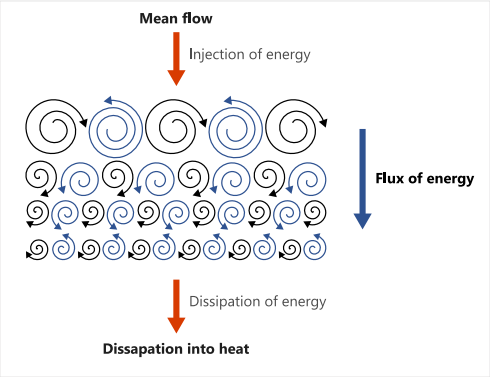


Figure 20: Energy Cascade

The image above illustrates a typical energy spectrum (E) of turbulent eddies in wave number space (κ). Since the turbulent flow has a wide range of eddies with different length scales, it is convenient to utilize the energy spectrum of the velocity field for the classification of eddies into three representative length scales, the integral length scale, the Taylor microscale and the Kolmogorov length scale.

The integral length scale is the largest eddy size of the energy spectrum. These are the most energetic and highly anisotropic eddies that produce energy via the interaction with the mean flow. Eddies in the integral range (A) are very sensitive to flow conditions and their sizes are close to the characteristic length of the flow (for example hydraulic diameter). Its spectrum is defined as $E \propto v^2 l$. The integral length scale is $l = k^{3/2}/\varepsilon$ and corresponding time scales of $t = k/\varepsilon$ and velocity scale $v = \sqrt{k}$.

The Taylor microscale is an intermediate eddy size of the energy spectrum between the largest eddy and the smallest eddy where the flow is inviscid. In this inertial subrange (B), the kinetic energy is transferred from large turbulent eddies to the small eddies in what is defined as the energy cascade. The occurrence of the energy cascade is related to the process of the vortex stretching. The vortex stretching causes the rotational rate of the turbulent eddies to increase and the radius of their cross sections to decrease. The energy spectrum of the inertial subrange is obtained through dimensional analysis and shows that the energy spectrum is proportional to the product of the wave number and the dissipation rate $E \propto \kappa^{-5/3}\varepsilon^{2/3}$.

The Kolmogorov scale is the smallest eddy size and is in the dissipative range (C) of the energy spectrum. Turbulent eddies in the dissipative range are isotropic because the diffusive actions of strong viscosity smear out anisotropic characteristics of the larger eddies. In this region the kinetic energy received from larger eddies is dissipated into heat. The length scale of this region is assumed be a function of the kinematic viscosity (v) and the turbulent dissipation rate per unit mass (ε). Using dimensional analysis you obtain

- The Kolmogorov length scale $\eta = \left(\frac{v^3}{\varepsilon}\right)^{1/4}$
- The Kolmogorov time scale $\tau_\eta = \left(\frac{v}{\varepsilon}\right)^{1/2}$
- The Kolmogorov velocity scale $v_\eta = (v\varepsilon)^{1/4}$

- The energy spectrum $E \propto u^{5/4}\varepsilon^{1/4}$.

The ratio of the integral length scale to the Kolmogorov length scale $l/\eta \sim \left(\frac{\rho v l}{\mu}\right)^{3/4} = \mathrm{Re}^{3/4}$.

Corresponding time scale and velocity scale ratios are, respectively, given as $t/\tau = \sqrt{\mathrm{Re}}$ and $v/v_\eta \sim \mathrm{Re}^{1/4}$.

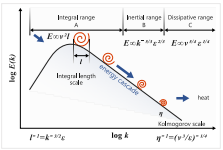


Figure 21: Schematic Representation of Turbulence Energy Spectrum

Considering these scale ratios, you can see larger scale separation between the integral length scale and the Kolmogorov length scale with an increase of the turbulent Reynolds number. This suggests that the Kolmogorov length scale is much smaller than the integral length scale for high turbulent Reynolds number.

References

Bradshaw, P., 1971, "An Introduction to Turbulence and Its Measurement," Pergamon Press, Oxford, England.

Hinze, J. O., 1975, "Turbulence," 2nd Edition, McGraw-Hill, New York, USA.

Launder, B.E., 1991, "An Introduction to the Modeling of Turbulence,' VKI Lecture Series 1991-02.

Pope, S. B., 2000, "Turbulent Flows," University Press, Cambridge, England.

Richardson, L. F., 1922, "Weather Prediction by Numerical Process," University Press, Cambridge, USA.

Saric, W.S., 1994, "Görtler Vortices," Annual Review of Fluid Mechanics, Vol. 26, pp. 379-409.

Tennekes, H. and Lumley, J.L., 1972, "A First Course in Turbulence," The MIT Press, Cambridge, USA.

van Dyke, M., 1982, "An Album of Fluid Motion," Stanford University, California, USA.

White, F.M., 1991, "Viscous Fluid Flow," McGraw-Hill, New York, USA.

Modeling of Turbulence

This section covers the numerical modeling of turbulence by various turbulence models, near wall modeling and inlet turbulence parameters specified for turbulence models.

Although turbulence has been researched for centuries it continues to pose some of the most difficult and fundamental problems in physics because it is a complex, nonlinear phenomenon.

It has the following characteristics:

- Three dimensional and time dependent
- Irregular and chaotic in nature
- Non repetitive
- Contains a vast range of length and time scales
- Has a length scale reduction with increasing Reynolds number
- Is sensitive to boundary and initial conditions

Considering these turbulence characteristics, resolving the turbulent quantities directly by means of Direct Numerical Simulation (DNS) is impractical due to the limitations of currently available computational resources. For this reason, the preferred approach is not to resolve turbulence but to model it with consideration of the following observations:

- Statistical averages of quantities in turbulent flows are reproducible.
- Engineering problems are concerned with the mean flow motion, not the instantaneous motion.

  Given these justifications, turbulence models are developed to account for the effects of eddies on the mean flow field in order to make Computational Fluid Dynamics (CFD) plausible on industrial scale engineering problems.

  In this section representative turbulent flow cases are provided to explain why turbulent flow simulations could be challenging. Resource requirements for DNS are examined in order to justify the need of turbulence modeling. After a brief comparison of resource requirements for various turbulence models the Reynolds-averaged Navier-Stokes equations (RANS) are introduced and turbulence models are presented to close the RANS equations. The focus then shifts to Large Eddy Simulation (LES).

  Overall, this discussion is primarily aimed at motivating the need for turbulence models when investigating high Reynolds number turbulent flow applications and to introduce commonly available turbulence models to new users.

Challenges in Simulating Turbulent Flows

In order to set the context for the modeling of turbulent flow it is essential to understand why turbulent flow simulations are challenging.

First, examine the turbulence characteristic non-repetition. Figure 22 shows two snapshots of smoke plumes from a burning incense stick. Both instantaneous plumes show the chaotic, continuously evolving movement of the smoke. Due to the sensitivity of the smoke plumes to surrounding conditions (temperature, density and cross flow velocity), the pattern of their movement never repeats. Thus, prediction on such instantaneous turbulent smoke evolutions is understood to be quite challenging.

Figure 22: Snapshots of Smoke Plumes

The Navier-Stokes Equations

The Navier-Stokes (NS) equations are the set of equations that govern the motion of a fluid.

These equations consist of the mass conservation equation and the momentum conservation equations for incompressible flows. The NS equations govern the behavior of a viscous fluid by balancing the forces with Newton's second law and assuming that the stress in the fluid is the sum of a diffusing viscous term and a pressure term. In the case of a Newtonian fluid, the instantaneous continuity equation is defined as

$$\frac{\partial \rho}{\partial t} + \frac{\partial \left(\rho u_j\right)}{\partial x_j} = 0 \tag{58}$$

where ρ is the fluid density and u_j is the velocity tensor.

The incompressible momentum equation is given by

$$\frac{\partial \left(\rho u_i\right)}{\partial t} + \frac{\partial \left(\rho u_i u_j\right)}{\partial x_j} = -\frac{\partial \left(p\delta_{ij}\right)}{\partial x_j} + \frac{\partial \tau_{ij}}{\partial x_j} \tag{59}$$

where ρ is the pressure, τ_{ij} is the viscous stress tensor and δ_{ij} is the Kronecker delta.

$$\delta_{ij} = \begin{cases} 0 & if \quad i \neq j, \\ 1 & if \quad i = j. \end{cases} \tag{60}$$

The left-hand side of the equation describes advection, which includes two terms, the acceleration term and convective term. The right hand side of the equation represents the summation of pressure and the shear-stress divergence terms. The convective term is non linear due to an acceleration associated with the change in velocity as a function of position. This term can be disregarded in one-dimensional flow and Stokes flow, or creeping flow.

For Newtonian fluid the viscous shear stress is assumed to be proportional to the shear strain rate, following Stokes Law. Thus, viscous stresses can be given by

$$\tau_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{1}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij} \quad (61)$$

where μ is the molecular (dynamic) viscosity of the fluid.

Now, you have a full set of the Navier-Stokes equations to govern fluid flows. The next step is how to solve these equations. There are two approaches available: analytical approaches and numerical approaches. Since most flows involve convective accelerations, it is difficult to obtain the analytical solutions of the Navier-Stokes equations due to the non-linearity associated with the convective term and coupled nature of the equations. Although analytical solutions do exist for some simple flow cases with certain approximations, they are mostly limited to laminar flows. These are Couette flow and Hangen-Poiseuille flow. The former is a laminar flow between two parallel plates, one of which is moving relative to the others, and the latter is a laminar flow in a long cylindrical pipe of constant cross section. Therefore, a complete description of a turbulent flow can only be obtained by numerically solving the Navier-Stokes equations (Moin and Mahesh, 1998). These are divided into a numerical approach resolving turbulent flows by Direct Numerical Simulation (DNS) and numerical approaches not resolving turbulence but modeling it.

Direct Numerical Simulation

Direct Numerical Simulation (DNS) solves the time dependent Navier-Stokes equations, resolving from the largest length scale of a computational domain size to the smallest length scale of turbulence eddy (Kolmogorov length scale).

Considering a vast range of length scales within the computational domain, you can rightly claim that huge computer resource would be required for DNS. Examining the computer resource requirements for DNS will support this claim.

For a computational square box domain, the number of grid points (n) in one direction depends on the grid spacing ΔL.

$$n = \frac{L}{\Delta L} \quad (62)$$

In order to resolve the Kolmogorov length scale (n), the grid spacing ΔL and the Kolmogorov length scale should have the same order of magnitude.

$$n \sim \frac{L}{\eta} \quad (63)$$

Since the Kolmogorov length scale is a function of the fluid kinematic viscosity (ν) and the turbulent dissipation rate ((ε) $\eta = \left(\frac{\nu^3}{\varepsilon}\right)^{1/4}$), the number of grid points in one direction can be estimated from

$$n \sim L\frac{\varepsilon^{1/4}}{\nu^{3/4}} \quad (64)$$

Utilizing $\varepsilon \sim \frac{U^3}{L}$ the equation for the grid size can be rewritten as

$$n \sim \left(\frac{UL}{\nu}\right)^{3/4} \sim \mathrm{Re}^{3/4} \quad (65)$$

where U is the characteristic velocity.

For three-dimensional flows, the total grid size for DNS can be computed as

$$N = n^3 \sim \mathrm{Re}^{9/4} \tag{66}$$

For a flat plate, turbulent flow occurs when the Re > 500,000 (Schlichting and Gersten, 2000). The above relation shows the estimation of six trillion nodes, which easily exceeds the capacity of even the most advanced high performance computers.

In addition to the demanding grid size requirements of DNS, the computing time step size (Δt) should be small enough to resolve the Kolmogorov time scales. Since the Kolmogorov time scale depends on the Kolmogorov length scale and the characteristic velocity (U), the time step size can be approximated as

$$\Delta t \sim \frac{\eta}{U} \tag{67}$$

Given this, the number of time steps (n_t) can be estimated from the total duration of the simulation (T) and the time step size (Δt)

$$\Delta N_t = \frac{T}{\Delta t} \sim \frac{T}{\eta / U} \tag{68}$$

Utilizing $\eta = \mathrm{Re}^{-3/4} L$ the equation of the number of the time steps can be estimated as

$$N_t \sim \frac{T}{L/U} \mathrm{Re}^{3/4} \propto \mathrm{Re}^{3/4} \tag{69}$$

Finally, the number of floating point operations needed to perform DNS can be computed from the multiplication of total grid size (N) and the number of the time steps

$$N_f = N \; N_t \propto \mathrm{Re}^3 \tag{70}$$

Therefore, the computational cost for DNS is very expensive, confirming that DNS is not feasible for high Reynolds Number turbulent flows.

In addition, high-order (third-order or higher) numerical schemes are commonly used in order to reduce the numerical dissipation and to keep the problem size tractable. These include spectral methods or spectral element methods. Although these methods are very efficient in resolving the small scales of turbulence, they require that the computational domain be relatively simple. The direct result is that these high-order schemes have little flexibility in dealing with complex industrial geometries because of the structured (blocked) mesh approach. Finally, DNS requires special treatments for realistic initial and boundary conditions

Considering these observations it can be concluded that DNS attempting to resolve all turbulent length and time scales is restricted to the low Reynolds number range and is impractical for industrial flows due to huge computing resource requirements. Most DNS applications are served as benchmark databases for tuning turbulence models and have been used for fundamental turbulent flow studies, including homogeneous turbulent flows with mean strain, free shear layers, fully developed channel flows, jets and so on.

Turbulence Modeling

Three-dimensional industrial scale problems are concerned with the time averaged (mean) flow, not the instantaneous motion. The preferred approach is to model turbulence using simplifying approximations, and not resolve it.

Turbulence modeling is a procedure to solve a modified set of the Navier-Stokes equations by means of developing a mathematical model of the turbulent flow that represents the time-averaged characteristics of the flow. Turbulence modeling is used to compute the impact of eddies on the mean flow field. This approach is based on the assumption that the turbulent eddy motion is "universal" and can be related to the large-scale average motion.

Over the course of the past few decades, turbulence models of various complexities have been developed. Depending on the simplifications made to the Navier-Stokes equations, the turbulence models can be classified as shown below.

Large Eddy Simulation (LES)

LES solves the filtered Navier-Stokes equations to resolve eddies down to the inertial range and it uses subgrid models to account for the influence of eddies in the dissipative range. The computing requirement is substantially less than that of DNS but is still not practical for many industrial applications containing wall bounded flows.

Common types include:

- Smagorinsky model
- Germano dynamic model

Hybrid

Hybrid simulations are the bridge between LES and RANS by utilizing RANS for attached boundary layers and LES for separated flow regions. In general, hybrid simulations need spatial filtering processes to determine the local sub grid turbulent viscosity. Compared to LES and DNS, hybrid simulations are much more tractable since the numerical requirement is less severe than the other two approaches.

Common types include:

- Detached Eddy Simulation (DES)
- Delayed DES (DDES)
- Improved DDES (IDDES)
- Scale Adaptive Simulation (SAS)
- Wall Modeled LES (WMES)
- Zonal LES

Reynolds-averaged Navier-Stokes (RANS)

RANS simulations solve directly for the time averaged flow and model the effects of turbulent eddies on the mean flow. This method is the most computationally efficient CFD approach. Since most engineering problems are concerned with the time-averaged properties of the flow, this approach is used most

frequently in the industry. The Reynolds averaging procedure introduces additional unknowns into the Navier Stokes equations, and it is thus necessary to develop additional turbulence model equations to close the set. These additional model equations can be categorized into turbulence models that use the Boussinesq assumption and turbulence models that do not use the Boussinesq assumption. The turbulence models with the Boussinesq assumption have two steps to compute Reynolds stresses for the RANS equations. First, turbulence models are needed for the computations of the eddy viscosity, second, the eddy viscosity is used for the estimation of the Reynolds stress with the Boussinesq assumption. The turbulence models without the Boussinesq assumption, for example, Reynolds stress models or nonlinear eddy viscosity models, determine the Reynolds (turbulent) stresses explicitly by solving an equation for each stress component.

Common types include:

- Seven equation models
- Reynolds stress model (RSM)
- One, two and three equation models + Boussinesq
- v2f, zeta-f
- k-ε, k-ω, SST
- Spalart-Allmaras (SA)

Figure 23 shows turbulence models and their corresponding energy spectrum ranges for modeling. For example, RANS models rely on their transport equations to model the entire wave number range, while LES needs a subgrid model to model behaviors of eddies in the dissipative range, but explicitly resolves the large eddies. The hybrid RANS/LES approach needs a model to cover the inertial range and the dissipative range.

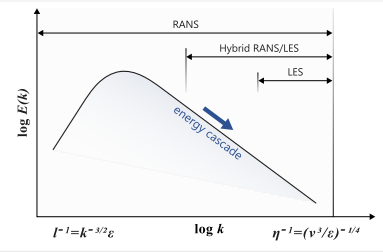


Figure 23: Turbulent Energy Spectrum Modeled by Various Turbulent Flow Simulation Approaches

As expected, computing requirements for these models differ since their models cover a different turbulent spectrum for the range of wave numbers. Spalart (2000) summarized resource requirements

for each model, as shown below. Strictly speaking, DNS is not a turbulence model since it is resolving all scales of motion, however, it is included for comparison purposes. Grid numbers and time steps were estimated for a clean wing (Spalart, 2000). Grid numbers and time steps increase from RANS to DNS.

Table 4: CFD Methods (Spalart, 2000)

CFD methods	Reynolds Number Dependence	Empiricism	Grid	Time steps
DNS	strong	none	10^{16}	$10^{7.7}$
LES	weak	weak	$10^{11.5}$	$10^{6.7}$
DES (hybrid)	weak	strong	10^{8}	10^{4}
RANS	weak	strong	10^{7}	10^{3}

Summary

During the past decades turbulence models of various complexities have been developed. Turbulence models that employ the most assumptions are typically the least demanding from a CPU cost standpoint.

As the number of assumptions associated with the model decreases, the cost and accuracy of the model typically increases. Since no universal turbulence model exists, CFD users need to choose the best turbulence model specific to their application. The most cost effective approach is typically steady Reynolds-averaged Navier-Stokes (RANS). When the flow is naturally unsteady and exhibits oscillations in the quantities of interest, URANS may be employed to improve the accuracy of solutions and gain a better understanding of how much the flow field fluctuates as a function time. For applications when the flow transitions from laminar to turbulent as a function of space, a dedicated transition model is required in order to accurately predict the near wall behavior. When the application demands additional knowledge of the turbulent fluctuations for separated, reattaching or impinging flow fields, hybrid simulations (Detached Eddy Simulation (DES), Delayed Detached Eddy Simulation (DDES)) may be employed. Although the modeling assumption is still made within the boundary layer, the tradeoff between accuracy and computational efficiency is often worthwhile compared to Large Eddy Simulation (LES) or Direct Numerical Simulation (DNS). Finally, when the application requires detailed information about nearfield fluctuations, or for combustion and acoustic applications, LES should be considered.

Based on the information presented it is up to you to understand the complexity of your simulation and make an appropriate decision on which turbulence model to use. Generally, it is recommended to start with a simple turbulence model since a majority of industrial problems can be well defined with a steady state assumption. Even if the application is inherently unsteady, this is still a good place to start as a significant amount of information can be garnered from running RANS. If the initial RANS fails to provide the level of accuracy required for the specific application you will need to consider the use of a hybrid model or LES. Often times the flow regime exhibits sensitivities to the selection of the turbulence model, specifically RANS vs. Hybrid/LES. In any case, the selection of the turbulence model

should be verified against previous simulation or validated with experimental/analytical data for the flow in question.

Numerical Approximation Techniques

This section on numerical approximation techniques covers topics, which describe the numerical modeling of the fluid flow equations on a computational domain, such as spatial discretization using finite difference, finite element and finite volume techniques, temporal discretization and solution methods.

Overview

The governing equations, that is, the Navier – Stokes equations in continuum mechanics are a set of coupled non–linear partial differential equations derived from the conservation laws for mass, momentum and energy.

For a numerical solution of a mathematical continuum model, the focus is to devise efficient, robust, and reliable algorithms for the solution of the partial differential equations. To do this the partial differential equations are converted to a discrete system of algebraic equation using a discretization procedure. In case of a CFD simulation, the discretization of the governing equations, that is, the derivation of equivalent algebraic relations should accurately represent the equations and the physics modeled.

There are a number of discretization techniques that have been developed to solve different kinds of problems in CFD. These methods can be broadly classified into two categories:

- Mesh based methods – These methods require division of the problem domain into a grid or mesh over which the governing equations are discretized. The points are positioned according to a topological connectivity which ensures the compatibility of the numerical technique used. Examples of these methods include Finite Difference (FD), Finite Volume (FV) and Finite Element (FE) methods.
- Mesh free methods – These methods use a collection of nodes in the domain which do not have any apparent connectivity. These methods are particularly important in simulations where the nodes are created or destroyed, simulations where the deformations are so large that the connectivity might introduce distortion and in cases where the domain possesses discontinuities or singularities. A few examples of these methods are: Smoothed Particle Hydrodynamics (SPH), Finite Pointset Method (FPM), Meshless Local Petrov Galerkin (MLPG) and Lattice Boltzman methods.

Of the methods discussed above, mesh based methods are more popular and are among the most widely used in CFD codes. The following sections describe the mesh based discretization methods (FD, FV and FE) and the time discretization techniques used to solve the partial differential equations governing a fluid flow.

Finite Difference Method (FDM)

This section describes the formulation and methodology of finite difference method to solve the governing equations on a computational domain.

The finite difference method is the oldest method for the numerical solution of partial differential equations. It is also the easiest to formulate and program for problems which have a simple geometry.

When calculating derivatives finite difference replaces the infinitesimal limiting process with a finite quantity. The derivative of a function at point expressed as:

$$\frac{\partial f}{\partial x} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x} \tag{71}$$

is replaced by:

$$\frac{\partial f}{\partial x} = \frac{f(x+\Delta x) - f(x)}{\Delta x} + O(\Delta x) \tag{72}$$

The preceding expression is commonly referred to as the forward difference approximation. This derivative can have more refined approximations using a number of approaches such as truncated Taylor series expansions and polynomial fitting.

The term $O(\Delta x)$ gives an indication of the magnitude of the error as a function of the mesh spacing and is therefore termed as the order of magnitude of the finite difference method. In the above formulation, the finite difference approximation employed is first-order accurate. A second-order approximation would have the order of magnitude expressed as $O(\Delta x^2)$. Most of the finite difference methods used in practice are second-order accurate. An example of a second-order method is the central difference approximation of the first derivative. It is expressed as:

$$\frac{\partial f}{\partial x} = \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x} + O(\Delta x^2) \tag{73}$$

The basic methodology in the simulation process using finite difference approach consists of the following steps.

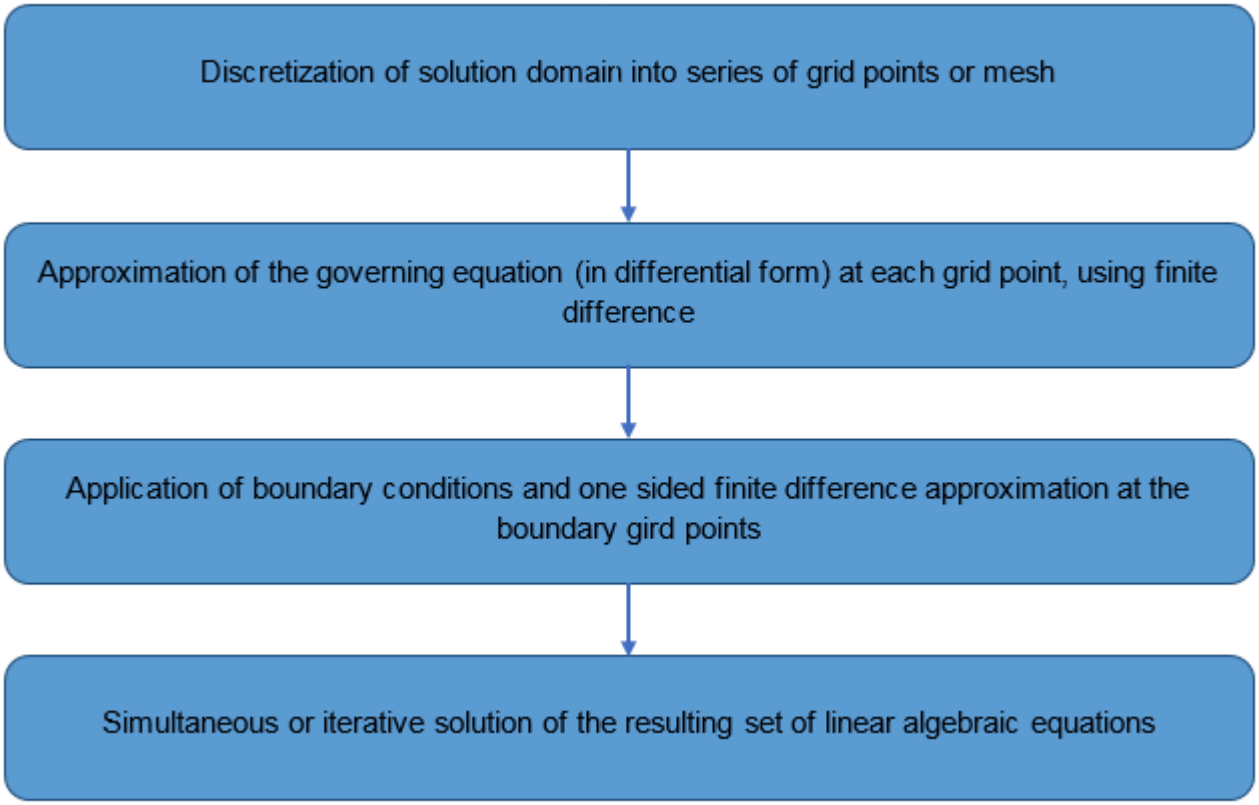


Figure 24: Basic Methodology in the Simulation Process Using Finite Difference Approach

The finite difference formulation generally employs a structured grid. The most commonly used indices are i, j and k for the grid lines at $x = x_i$, $y = y_j$ and $z = z_k$, respectively. The function value at such a grid point is expressed as $f_{i,j,k} \equiv f_{ijk} \equiv f\left(x_i, y_j, \quad z_k\right)$.

Consider the one dimension advection-diffusion equation for a scalar quantity φ governed by

$$\frac{d(\rho u \varphi)}{dx} = \frac{d}{dx}\left(\epsilon \frac{d\varphi}{dx}\right) \tag{74}$$

where u is the specified velocity, ρ is the density of the fluid and ϵ is the diffusivity.

If the domain is defined by the boundary $x = 0$ and $x = L$ and the boundary conditions by $\varphi(0) = \varphi_0$ and $\varphi(L) = \varphi_L$, the domain can be discretised (non-uniformly) by a total of $N+1$ grid points for the finite difference solution of the problem.

The diffusion term can be approximated using the central difference (both the inner and outer derivative) as:

$$\frac{d}{dx}\left(\epsilon \frac{d\varphi}{dx}\right)_i \approx \frac{\left(\epsilon \frac{d\varphi}{dx}\right)_{i+\frac{1}{2}} - \left(\epsilon \frac{d\varphi}{dx}\right)_{i-\frac{1}{2}}}{\frac{1}{2}\left(x_{i+1} - x_{i-1}\right)} \tag{75}$$

$$\frac{d}{dx}\left(\epsilon\frac{d\varphi}{dx}\right)_i \approx \frac{\left(\epsilon_{i+\frac{1}{2}}\frac{\varphi_{i+1}-\varphi_i}{x_{i+1}-x_i}\right)-\left(\epsilon_{i-\frac{1}{2}}\frac{\varphi_i-\varphi_{i-1}}{x_i-x_{i-1}}\right)}{\frac{1}{2}\left(x_{i+1}-x_{i-1}\right)} \tag{76}$$

The convection term can also be approximated using the central difference as:

$$\frac{d}{dx}(\rho u\varphi)_i \approx \frac{(\rho u\varphi)_{i+1}-(\rho u\varphi)_{i-1}}{x_{i+1}-x_{i-1}} \tag{77}$$

If the grid is uniform and the values of density, velocity and coefficient of diffusion are constant throughout the domain the above equations reduce to:

$$\rho u\frac{\varphi_{i+1}-\varphi_{i-1}}{2\Delta x} = \epsilon\frac{\varphi_{i+1}-2\varphi_i+\varphi_{i-1}}{\Delta x^2} \tag{78}$$

The resulting set of linear algebraic equations can be solved to get the values of at the grid points.

Finite difference methods are advantageous for the numerical solution of partial differential equations because of their simplicity, efficiency and low computational cost. Their major drawback is their geometrical inflexibility, such as application on an unstructured grid or moving boundaries. Their formulation increases in complexity as the complexity of the domain increases.

The restrictions resulting from the above mentioned geometrical complexities can be alleviated by the use of methods such as grid transformation and immersed boundary techniques.

Finite Volume Method (FVM)

This section describes the formulation and methodology of finite volume method to solve the governing equations on a computational domain. It also describes the cell centered and face centered approaches used for finite volume formulation.

The finite volume formulation is based on the approximate solution of the integral form of the conservation equations. The problem domain is divided into a set of non overlapping control volumes referred to as finite volumes, where the variable of interest is usually taken at the centroid of the finite volume. The finite volume is also referred to as a cell or element.

The governing equations are integrated over each finite volume and interpolation profiles are assumed in order to describe the variation of the concerned variable of interest between the cell centroids. The resulting discretization equation expresses the conservation principle for the variable inside the finite volume.

The governing differential equations discussed in this manual each have a dependent variable that obeys a generalized conservation principle. If the dependent variable (scalar or vector) is denoted by φ, the generic differential equation is:

$$\underbrace{\frac{\partial(\rho\varphi)}{\partial t}}_{\text{Transient term}} + \underbrace{\nabla\cdot(\rho\vec{u}\varphi)}_{\text{Convection term}} = \underbrace{\nabla\cdot(\epsilon\nabla\varphi)}_{\text{Diffusion term}} + \underbrace{S_\varphi}_{\text{Source term}} \tag{79}$$

where

- ϵ is the diffusion coefficient.

- The transient term $\frac{\partial \rho\varphi}{\partial t}$ accounts for the rate of change of φ inside the control volume.
- The convection term $\nabla \cdot (\rho \vec{u} \varphi)$ accounts for the transport of φ due to the velocity field $\vec{u}$.
- The diffusion term $\nabla \cdot (\epsilon \nabla \varphi)$ accounts for the transport of φ due to its gradients.
- The source term S_φ accounts for any sources or sinks that affect the quantity φ.

When the above equation is integrated over a three dimensional control volume it yields

$$\int_{CV} \frac{\partial(\rho\varphi)}{\partial t} \; dV + \int_{CV} \nabla \cdot (\rho \vec{u} \varphi) \; dV = \int_{CV} \nabla \cdot (\epsilon \nabla \varphi) \; dV + \int_{CV} S_\varphi \; dV \tag{80}$$

The divergence terms (convection and diffusion) can be converted into surface integrals using Gauss divergence theorem and the above equation can be rewritten as

$$\frac{\partial}{\partial t} \int_{CV} \rho\varphi \; dV + \oint_S \hat{n} \cdot (\rho \vec{u} \varphi) \; dS = \oint_S \hat{n} \cdot (\epsilon \nabla \varphi) \; dS + \int_{CV} S_\varphi dV \tag{81}$$

where

- $\hat{n}$ is the unit vector normal to the surface dA
- S is the boundary of the control volume
- $\hat{n} \cdot (\rho \vec{u} \varphi)$ is the convective flux of φ across the boundary dS
- $\hat{n} \cdot (\epsilon \nabla \varphi)$ is the diffusive flux of φ across the boundary dS

The integral conservation in the scalar transport equation applies to each control volume as well as the complete solution domain thus satisfying the global conservation of quantities such as mass, momentum and energy. These quantities can be evaluated as fluxes at the surfaces of each control volume.

In order to obtain an algebraic (discretised) equation for each control volume the surface and volume integrals are approximated using quadrature formulae in terms of function values at the storage location. This may require values of variable at points other than the computational nodes of the control volume. Values at these locations are obtained using interpolation schemes.

The overall finite volume approach involves the following steps.

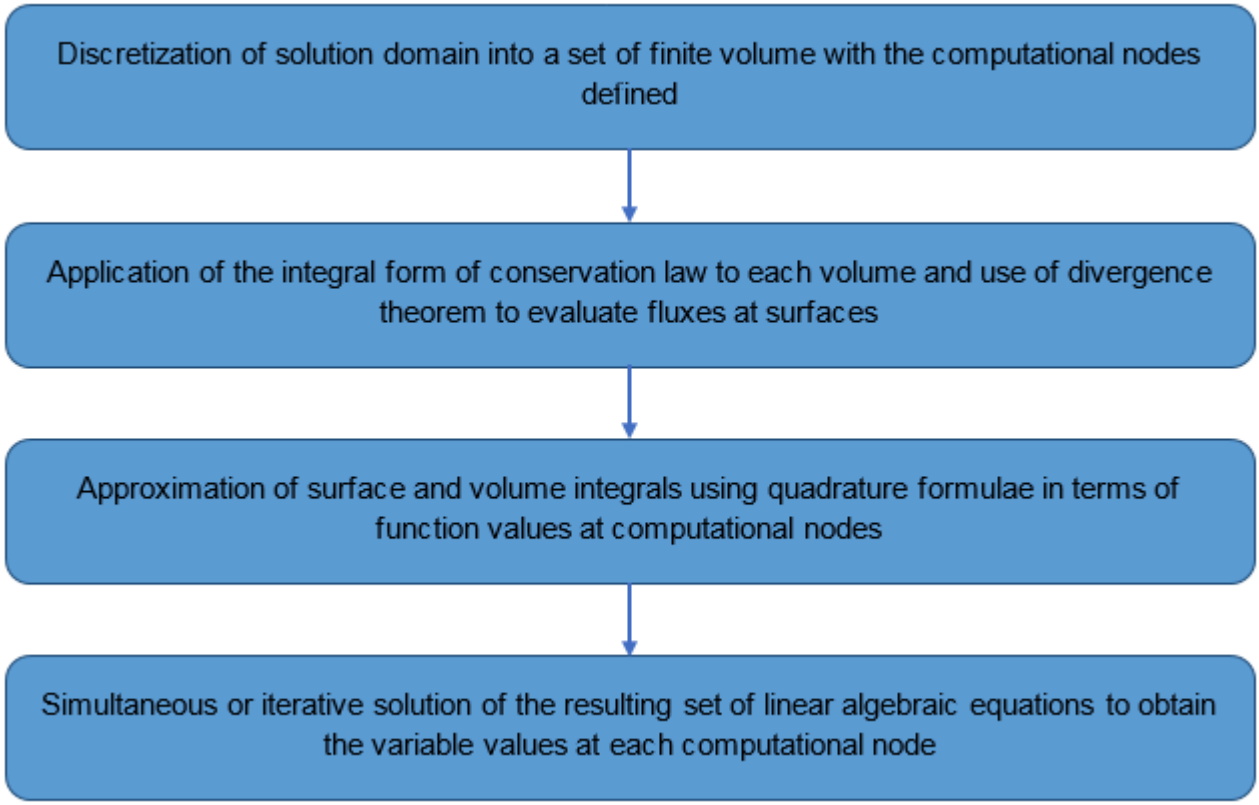


Figure 25: Finite Volume Approach

There are two common approaches to finite volume discretization.

- Cell centered approach: The domain is defined by a suitable grid of finite volumes and the computational nodes are assigned at the centroid of the control volumes.
- Face/Node centered approach: The domain is first defined by a set of nodes and control volumes are constructed around these nodes as cell centers such that the faces of the control volume lie between these nodes.

The cell centered approach and node centered approach have nearly the same accuracy and efficiency for most of the cases which use a structured grid.

To illustrate how the conservation equations in CFD can be discretised using finite volume method an example involving the steady transport of x-momentum in a uniform 2D rectangular grid can be considered:

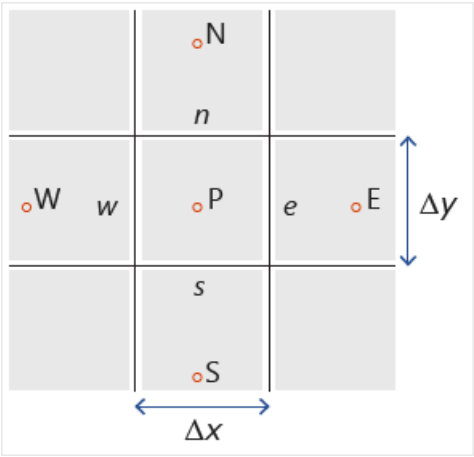


Figure 26: Finite Volume Stencil Around Point P

A cell centered approach is employed with the points P, W, E, N, S representing the cell centers and the notations n, e, s, w representing the faces of cell P. The notation N (n), E(e), S(s), W(w) represent the north, east, south and west directions, respectively.

The velocity u is stored at the nodes N, E, S, W and it is represented as u_N, u_E, u_S, u_W, respectively. The finite volume approximation starts by integrating the x-momentum equation over the control volume P.

$$\iint_V \left[\frac{\partial}{\partial x}(\rho u^2)+\frac{\partial}{\partial y}(\rho uv)\right]dx\ \ dy = -\iint_V \frac{\partial p}{\partial x}dx\ \ dy \tag{82}$$

$$+\iint_V \left[\frac{\partial}{\partial x}\left(\mu\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(\mu\frac{\partial u}{\partial y}\right)\right]dx\ \ dy \tag{83}$$

This leads to:

$$\left[\int \rho u^2 dy\right]_w^e+\left[\int \rho uv dx\right]_s^n=\left[\int \mu\frac{\partial u}{\partial x}dy\right]_w^e+\left[\int \mu\frac{\partial u}{\partial y}dx\right]_s^n-\iint_V \frac{\partial p}{\partial x}dx\ \ dy \tag{84}$$

Evaluation of Source Terms

The source terms can be approximated by evaluating them at cell centers and multiplying them by the volume of the cell.

$$\iint_V \frac{\partial p}{\partial x}dx\ \ dy \approx \left(\frac{\partial p}{\partial x}\right)_P \Delta x\ \ \Delta y$$

In the example considered the pressure gradient at center of cell P can be evaluated by interpolating the pressure values from surrounding nodes. Other source terms can be evaluated similarly, interpolating where necessary to estimate the cell center values.

Evaluation of Diffusive Fluxes

The diffusive fluxes can be approximated as

$$\left[\int \mu\frac{\partial u}{\partial x}dy\right]_w^e+\left[\int \mu\frac{\partial u}{\partial y}dx\right]_s^n \approx \left[\mu\frac{\partial u}{\partial x}\Delta y\right]_w^e+\left[\mu\frac{\partial u}{\partial y}\Delta x\right]_s^n$$

The gradients of velocity at the cell faces (e, w, n, s) can be approximated using central difference comprising of the cell center values. This results in diffusive fluxes being represented as

$$\left[\int \mu \frac{\partial u}{\partial x} dy\right]_w^e + \left[\int \mu \frac{\partial u}{\partial y} dx\right]_s^n \approx (\mu \Delta y)_e \frac{u_E - u_P}{\Delta x} - (\mu \Delta y)_w \frac{u_P - u_W}{\Delta x}$$

$$-(\mu \Delta y)_n \frac{u_N - u_P}{\Delta y} - (\mu \Delta y)_s \frac{u_P - u_S}{\Delta y}$$

The discretised equation which is second-order accurate takes the form

$$a_e u_E + a_w u_W + a_n u_N + a_s u_S - a_p u_P$$

Evaluation of Convective Fluxes

The convective fluxes can be approximated as

$$\left[\int \rho u^2 dy\right]_w^e + \left[\int \rho uv dx\right]_s^n \approx \left[\rho u^2 \Delta y\right]_w^e + \left[\rho uv \Delta x\right]_s^n$$

The discretised equation takes the form

$$c_e u_e - c_w u_w + c_n u_n - c_s u_s$$

where c_e, c_w and so on are mass fluxes through the east, west, north and south faces.

The values of u at cell faces need to be obtained using appropriate interpolation schemes between the cell center values. A few examples of such schemes are linear interpolation (CDS), quadratic upwind interpolation (QUICK), total variation diminishing (TVD).

Finite Element Method (FEM)

The finite element approach is based on the discretization of the domain into a set of finite elements, which are usually triangles or quadrilaterals in a two dimensions and tetrahedral, hexahedra, pyramids or wedges in three dimensions, and using variational principles to solve the problem by minimizing an associated error function or residual.

The unknown variables are approximated over the domain using interpolation procedure in terms of nodal values and set of known functions called shape functions. This approximation is substituted into the governing (conservation) equations in their differential form and the resulting error (residual) is minimized in an average sense using a weighted residual approach.

The variational or weighted residual formulations transforms the governing equations into an integral form called the global weak form. The weak form when applied to each finite element results in a set of discrete equations in terms of the nodal unknown which can then be solved by a number of methods.

In this formulation the governing differential equation for a quantity φ is expressed as:

$$L(\varphi)=\frac{\partial \varphi}{\partial t}+\frac{\partial}{\partial x_j}\left(f_j-f_j^d\right)-s=0 \tag{85}$$

where

- L is a differential operator with its associated initial and boundary conditions.
- $f_j=f\left(u_j, \varphi\right)$ is the convective flux vector.
- $f_j^d=f\left(\epsilon\frac{\partial \varphi}{\partial x_j}\right)$ is the diffusive flux vector.
- s is the contribution due to a source.

An approximate solution $\overline{\varphi}$ of the above equation is assumed, having the form:

$$\varphi(x)\approx\overline{\varphi}(x)=\sum_i N_i\varphi_i(x) \tag{86}$$

where

- N_i is the prescribed shape (interpolation) function.
- φ_i is the unknown value of the variable φ at a discrete spatial point i.

Finite element method requires transformation of the governing differential equation into an integral equation over the domain. This is accomplished through approaches such as weighted residual formulation and least squares formulation.

Overall, the finite element approach contains the following steps.

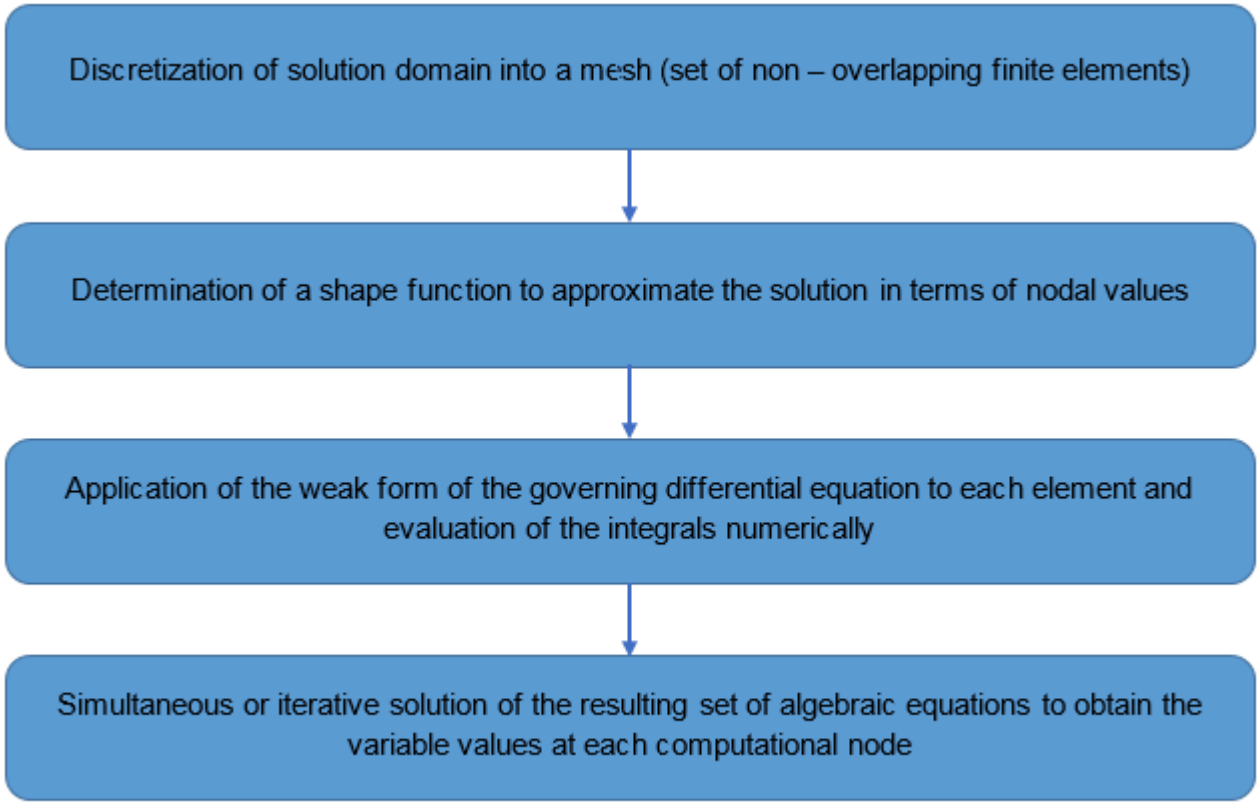


Figure 27: Finite Element Approach

A few of the approaches used to convert the differential equations into their weak integral forms are discussed below.

Weighted Residual Formulation

Substitution of the approximated form into the governing equations results in an error or residual function denoted by R and expressed as:

$$L(\overline{\varphi}) = R \tag{87}$$

In order to determine the nodal values φ_i, the inner product of the residual with a prescribed weighted function w_i is set to zero

$$\int_{\Omega} R w_i \; d\Omega \equiv \int_{\Omega} L(\overline{\varphi}) w_i \; d\Omega = 0 \tag{88}$$

The above equation is called the strong form of the weighted residual method. In many cases it is possible to perform integration by parts and obtain the weak form of the equation which contains lower order derivatives than the ones occurring in L and has reduced continuity requirement for the weighted function. The weak form is expressed as:

$$\int_{\Gamma} A(\overline{\varphi}) w_i \ d\Gamma + \int_{\Omega} B(\overline{\varphi}) C(w_i) \ d\Omega = 0 \tag{89}$$

where A, B and C are the differential operators with lower order derivatives than L and Ω represents the domain and Γ the boundary of the domain.

Different solution methods are obtained based on the choice of the weight function. Some of these approaches are:

- Point Collocation: $w_i(x) = \delta(x, x_i)$ where δ is the Dirac-Delta function. This is analogous to the finite difference approach.
- Subdomain Collocation: $w_i(x) = 1$ for the i^{th} subdomain (element) and zero elsewhere. This leads to a finite volume formulation.
- Galerkin method: $w_i(x) = N_i(x)$ where N is a shape function assumed over an element.
- Petrov Galerkin method: $w_i(x) \neq N_i(x)$. This represents a generalization of all methods except the Galerkin method.

Least Squares Formulation

The least square approach is based on minimizing the residual function in a least square sense. A least square function is constructed by taking the inner product of residual function with itself and defined as:

$$I(\overline{\varphi}) = L(\overline{\varphi})^2 \tag{90}$$

Using variational principles leads to a least square weak form written as:

$$\int_{\Omega} L(w_i) L(\overline{\varphi}) \ d\Omega = 0 \tag{91}$$

Galerkin Least Squares Formulation

Galerkin least square approach is an extension of the Galerkin weighted residual method and the least square approach. It uses stabilizing terms obtained by minimizing the residual of the governing equation. The use of variational principles on the resulting equations results in the following formulation:

$$\langle L(\varphi_i), w_i \rangle + S^{GLS} = \langle L(\varphi_i), w_i \rangle + \sum_k \langle L(\varphi_i), \tau_k L(w_i)_k \rangle = 0 \tag{92}$$

where

- $\langle L(\varphi_i), w_i \rangle$ represents the Galerkin approximation.
- S^{GLS} represents the least squares stabilization.
- $\langle , \rangle$ represents the inner dot product of the comma separated terms.
- τ_k represents the stabilization parameter.

Time Discretization

This section describes the various approaches used to discretize the governing equation in the temporal domain such as two step, multistep and multistage methods.

In order to determine a numerical solution of the governing differential equations the temporal domain must also be discretised apart from the spatial domain. The direction of influence of the time coordinate is only in the future. Therefore, all the solution methods for time dependent problems advance in time from a given initial data.

A vast majority of the methods used for temporal discretization are linear in nature. The time dependent variable is updated using a linear combination of the variable and its time derivatives. Linear approaches can be broadly categorized based on the number of steps, stages and derivatives used in the discretization.

Some of the widely used time discretization approaches are described below.

Generalized Two Step Methods

Two step methods involve function values at two instances in time, generally considering the current time step at which the solution is known and the next time step at which the solution has to be computed.

Consider a first-order ordinary differential equation for a dependent variable φ expressed as:

$$\frac{d\varphi}{dt} = f(t, \varphi) \tag{93}$$

with an initial condition $\varphi(t_0) = \varphi^0$.

A generalized scheme using a weighted average value for the approximation of the variable value at the $n+1^{th}$ time step is expressed as:

$$\varphi^{n+1} = \varphi^n + \Delta t[\theta f^{n+1} + (1-\theta) f^n] \tag{94}$$

where f^n represents the function value at the n^{th} time step and θ represents the weight.

The nature and stability of the temporal discretization scheme depends on the choice of the weight $\theta = 0, \frac{1}{2}$ and 1 represent the forward Euler, Crank-Nicholson and the backward Euler schemes, respectively.

Multistep Methods

Multistep methods involve function values at more than two instances of time. These methods are generally derived by fitting a polynomial to the temporal derivative of the dependent variable, that is, $f(t, \varphi)$.

These methods include the Adams-Bashforth and Adams-Moutlton methods. The order of the method depends on the number of points in time at which the polynomial fitting it used. A third-order accurate Adams-Moulton method is expressed as:

$$\varphi^{n+1} = \varphi^n + \frac{\Delta t}{2}[5f^{n+1} + 8f^n - f^{n-1}] \tag{95}$$

These methods require initial data at many steps, hence they are not self starting.

Multistage Methods

Multistage methods involve computation of the function values multiple times at the same time step. They generally involve predictor and corrector steps to compute the values at the n+1th time step.

Numerical solution schemes are often referred to as being explicit or implicit. When a direct computation of the dependent variables can be made in terms of known quantities the computation is said to be explicit. When the dependent variables are defined by coupled set of equations, and either a matrix or iterative technique is needed to obtain the solution, the numerical method is said to be implicit.

Explicit methods are easy to program but are conditionally stable whereas implicit methods offer better stability but are computationally expensive. Predictor-Corrector methods offer a compromise between these choices. A variety of methods exist based on the choice of base method and the time instants used in the predictor and corrector steps.

The most popular methods in this category are the Runge-Kutta methods. A fourth-order Runge-Kutta method is constructed as follows:

- Explicit Euler Predictor: $\varphi_*^{n+\frac{1}{2}} = \varphi^n + \frac{\Delta t}{2} f^n$
- Implicit Euler Corrector: $\varphi_{**}^{n+\frac{1}{2}} = \varphi^n + \frac{\Delta t}{2} f_*^{n+\frac{1}{2}}$
- Mid-point rule Predictor: $\varphi_*^{n+1} = \varphi^n + \Delta t f_{**}^{n+\frac{1}{2}}$
- Simpsons rule Corrector: $\varphi^{n+1} = \varphi^n + \frac{\Delta t}{6}\left[f^n + 2f_*^{n+\frac{1}{2}} + 2f_{**}^{n+\frac{1}{2}} + f_*^{n+1}\right]$

Generalized-*α* Method

The generalized *α* method is an implicit method of time integration which achieves high frequency numerical dissipation while at the same time minimizing unwanted low frequency dissipation and offers unconditional stability for linear problems. It is a variant of the generalized two step theta scheme discussed above where the first temporal derivatives are evaluated as variables.

For a linear system defined by:

$$\dot{\varphi} = \frac{d\varphi}{dt} = \lambda\varphi \tag{96}$$

The generalized *α* method for integration from time step t_n to t_{n+1} is constructed as follows:

- $\dot{\varphi}_{n+a_m} = \lambda\varphi_{n+a_f}$
- $\varphi_{n+1} = \varphi_n + \Delta t\dot{\varphi}_n + \Delta t\gamma\left(\dot{\varphi}_{n+1} - \dot{\varphi}_n\right)$
- $\dot{\varphi}_{n+a_m} = \dot{\varphi}_n + a_m\left(\dot{\varphi}_{n+1} - \dot{\varphi}_n\right)$
- $\varphi_{n+a_f} = \varphi_n + a_n\left(\varphi_{n+1} - \varphi_n\right)$

where Δt is the time step size $\left(\Delta t = t_{n+1} - t_n\right)$ and a_m, a_f, γ are free parameters.

The above four equations combine to yield the following system:

$$\phi_{n+1} = c\,\phi_n \tag{97}$$

where the solution vector ϕ_n at t_n is defined as $\phi_n = \left\{\varphi_n, \Delta t\dot{\varphi}_n\right\}^T$.

Direct Versus Iterative Solution Methods

This section describes the direct and iterative solution methods used to solve the linear system of equations obtained after spatial and temporal discretization of the governing equations.

Most of the methods of discretization discussed (FDM, FVM and FEM) yield a linear system of equations that need to be solved.

The resultant expression is of the form:

$$Ax = b \tag{98}$$

that is

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_3 \end{bmatrix} \tag{99}$$

where

- A is a matrix of known coefficients.
- b is a vector of given coefficients.
- x is a vector of unknowns such as density, velocity and temperature.

The methods used to solve the linear system of equations can be classified into two categories:

Direct Methods

Direct methods obtain an exact solution of the linear system of equations. These methods generally consist of elementary matrix operations to simplify the equations that will allow an exact computation. Some of these methods are:

- Cramer's Rule: The solution to $Ax = b$ is computed directly by $x = A^{-1}b$ and using Cramer's rule to compute the inverse of the coefficient matrix by calculating the co factor and adjoint of matrix A.
- LU Decomposition: This approach is based on the assumption that matrix A is composed of an upper triangular and a lower triangular matrix, that is $A = LU$. This results in $Ax = (LU)x = L(Ux) = Ly$ after which the systems $Ly = b$ and $Ux = y$ are solved separately.
- Gaussian Elimination: The given linear system of equations is converted to an augmented form given by $\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ \vdots & & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_3 \end{bmatrix}$ which is then converted to an upper triangular form

after elementary operations. The solution to the original system of equations is found by back substitution from the last row of the augmented matrix.

Direct methods work well for simple problems which generate lower order matrices but they are computationally expensive and inefficient for large sparse matrices. Moreover the coefficient matrix A depends on the properties of the fluid, such as viscosity or density which evolves as the solution is refined for the primary variables. Therefore it is not efficient to seek an exact solution to $Ax = b$ for every global iteration because the variables used to build A and b are evolving.

Iterative Methods

Iterative methods obtain an approximate solution of the linear system of equations that is based on iterations. An approximate solution of the primary variables is calculated which is then fed back into the primary equations to refine the variables. For large scale CFD problems it is more efficient to slightly improve over each variable at one turn and then cycle over to all the coupled variables in the subsequent iteration to reduce the error.

Iterative methods generate an approximate solution to the system that tends to converge to an exact solution. After m iterations the approximation is described as:

$$Ax^{(m)} = b - r^{(m)} \tag{100}$$

where $r^{(m)}$ is the residual after m iterations. The error in the approximation is defined as:

$$\epsilon^{(m)} = x - x^{(m)} \tag{101}$$

which results in the expression

$$A\epsilon^{(m)} = r^{(m)} \tag{102}$$

The purpose of the iterative methods is to drive this residual below the convergence criteria set.

The solution of the linear system of equations can be expressed as:

$$x^{(m+1)} = Cx^{(m)} + d \tag{103}$$

Based on the way that the solution at every iteration is updated, iterative methods can be classified into two types:

- Stationary Iterative Methods: The terms C and d are not dependent on the iteration count m. Some examples of these methods include Jacobi method, Gauss-Seidel method, Successive Overrelaxation method (SOR) and Symmetric Successive Overrelaxation method (SSOR).
- Non-Stationary Iterative Methods: The terms C and d are updated at every iteration. Some examples of these methods include Conjugate Gradient method (CG), Generalized Minimal Residual method (GMRES) and Conjugate Gradient Squared Method (CGS).

AcuSolve

4

This chapter covers the following:

Introduction to AcuSolve

This section gives a brief background of Altair HyperWorks- AcuSolve and a brief introduction of AcuSolve. Altair HyperWorks products, which include Radioss, OptiStruct, MotionSolve and HyperXtrude to name a few, is a software giant in the Computer Aided Engineering (CAE) Industry.

Apart from CAE products, Altair operates business in product design consultancy and cloud computing services.

AcuSolve is a leading general purpose CFD solver that is capable of solving the most demanding industrial and scientific applications. Based on the Finite Element method, AcuSolve's robust and scalable solver technology empowers users by providing unparalleled accuracy on fully unstructured meshes. Applications ranging from steady Reynolds-averaged Navier-Stokes (RANS) simulations to complex, transient, multi-physics simulations are handled with ease and accuracy. AcuSolve is used worldwide by scientists and engineers to explore applications in a variety of markets including automotive, aerospace, energy, electronics cooling and chemical processing.

The key benefits of AcuSolve include:

- Robustness: Most problems are solved on the first attempt
- Speed: Fully coupled solver on shared memory and distributed parallel systems
- Accuracy: Highly accurate in space and time while globally and locally conservative

AcuSolve Workflow

AcuSolve Workflow Introduction

This section gives a brief description of a general workflow of a typical CFD process. It describes in short the problem description, pre-processing, solving and post-processing aspects of a CFD workflow.

The entire workflow of a typical CFD analysis can be summarized as follows.

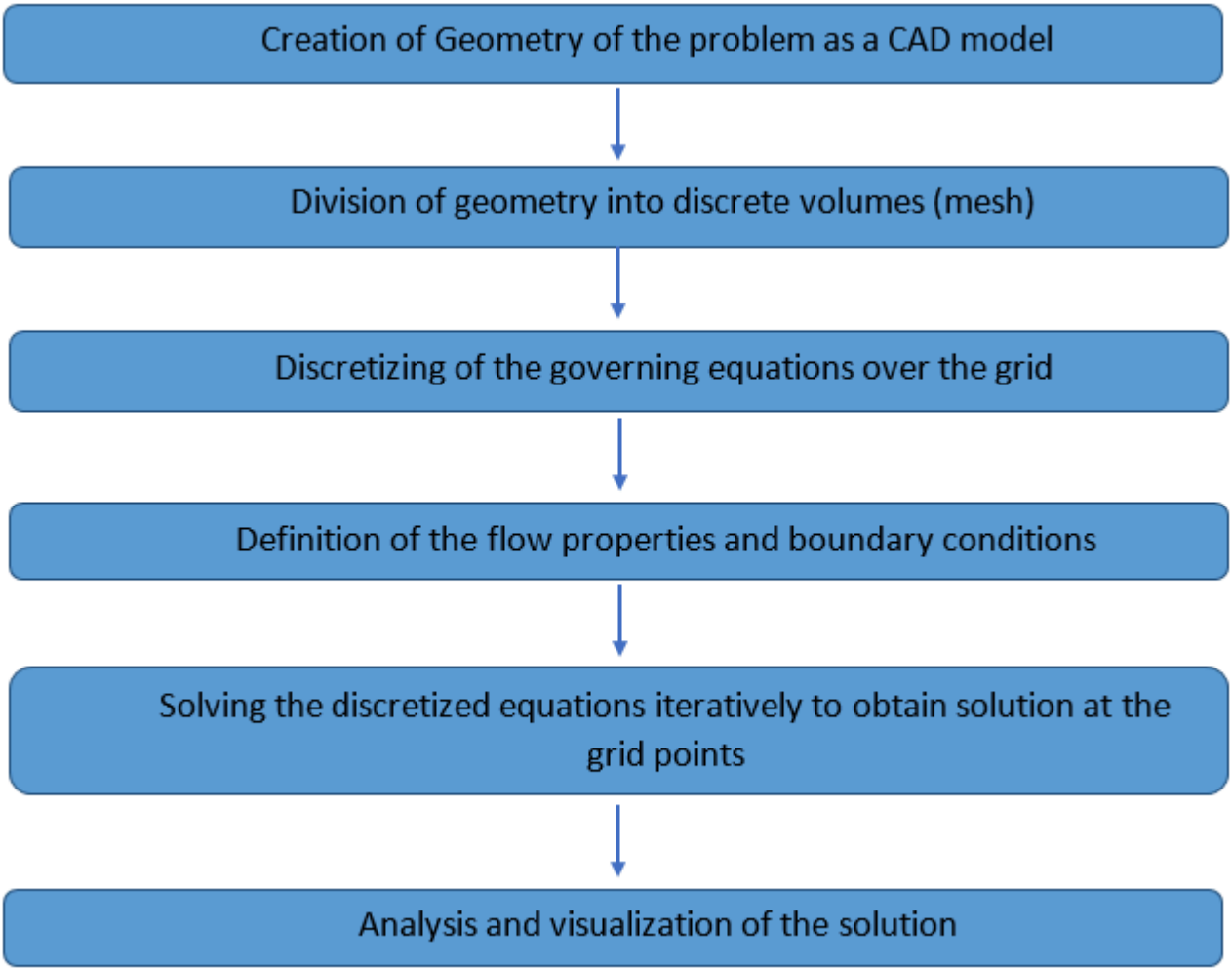


Figure 28: Basic Workflow

AcuSolve is a robust software that does all of the above steps of CFD and provides the accurate solution to the problem.

General Workflow

The accurate simulation of a flow field using CFD requires a number of steps to be taken. In order to set the problem up properly for simulation it is necessary to first understand the physics. Once the physics have been understood a model can be constructed. The following steps describe the set of steps that should be taken when performing a CFD simulation.

Problem Description

Though often neglected, this step is the foremost important aspect of a successful CFD simulation. Before attempting a CFD simulation a clear description of the problem is necessary. This step involves the following tasks.

- Identify the domain of interest: Though in principle domain of any size can be analysed with CFD simulation, it is often wise to choose the critical regions in which flow field is of importance and simulate the flow in that domain. Minimizing the size of the domain keeps the simulation costs low, but comes at the expense of introducing non physical behavior in the simulation by having boundaries too close to regions in the flow where the results are of interest. It is important to select boundaries where the flow character is known so that proper boundary conditions can be applied.
- Identify the nature of the flow: It is always helpful to make an educated guess on the nature of the flow, such as whether it is steady/transient, laminar/turbulent, thermal/non thermal. For example, while analysing a flow in a pipe that has low Reynolds number, say Re ~ 100, it is useful to perform a laminar analysis rather than doing an expensive turbulent flow analysis with a larger mesh count. In case of turbulent flows, it is advisable to choose turbulence model based on the physics present, while keeping in mind the areas where the flow will be critical.

Pre-Processing

This process involves geometry import, mesh generation and specifying the relevant physics equations to be solved. Pre-processing also involves defining the fluid properties, boundary conditions and body force parameters. AcuSolve uses HyperMesh CFD and SimLab for the pre-processing of the geometry and the mesh generation.

Solving

Solving is the critical part of a CFD simulation. The governing flow equations are discretized. All these algebraic equations are then solved for the unknown flow parameters. The final solution is the quantitative representation of these parameters at the mesh generated nodes. As a first step of solving, the AcuPrep module of AcuSolve checks for the compatibility of the information provided in the pre-processing step. The source of errors in AcuPrep could be because of insufficient inputs, syntax errors, and so on. After AcuPrep the actual solving of equations is performed.

Post-Processing

Having solved for the unknowns, the next step is the analysis of this huge data to obtain relevant insight. This data can be analysed either in the form of contour plots or tabular data to identify the critical portions of the flow field.

Pre and Post-Processing with AcuSolve

This section details the integration of AcuSolve with the industrially accepted pre and post-processing tools. AcuSolve is smoothly integrated with pre and post-processing HyperMesh CFD or SimLab.

A user of HyperMesh CFD can prepare the model in HyperMesh CFD and then can either export the input files for AcuSolve or launch the AcuSolve CFD solver directly by clicking the **Run** tool from the Solution ribbon.

After the solution is obtained from AcuSolve, HyperMesh CFD or SimLab can be used for various post-processing applications. For example, AcuSolve `.log` files can be directly read from HyperMesh CFD or SimLab by clicking **File** > **Open** > **Results** from the Post ribbon and then choosing the appropriate AcuSolve `.log` file.

AcuSolve Convergence Criteria

The convergence of the solution provides insight into the accuracy of solving the discretized equation set for the specific application. It reflects the imbalance of terms in the governing equations at each iteration. As a general rule, lower residuals indicate a smaller error resulting from imbalances in the equation terms.

At each time step, AcuSolve determines convergence by comparing the residual ratio and the solution ratio to the convergence tolerance. This convergence check is performed for each equation in the problem, using the residual and solution increment ratios. These ratios are computed individually for solution fields such as pressure, velocity and temperature.

The residual ratio serves as a global measure of how well the solution aligns with the equations being solved. It calculates the relative imbalance between the left and right-hand sides of the equation. On the other hand, the solution ratio provides a global measure of the magnitude of changes in the solution at each node as the solution progresses from one step to the next. Both the residual ratio and solution ratio are normalized using appropriate values. The normalization factor is computed separately for each stagger and is recalculated at each time step.

By default, with a convergence tolerance of 0.001, the residual ratio must be below 0.001 for pressure, velocity, temperature, species, and radiation, and below 0.01 for eddy-viscosity or other turbulent quantities. The solution ratio must be below 0.01 for pressure, velocity, temperature, species, and radiation, and below 0.1 for eddy-viscosity or other turbulent quantities. If these criteria are met during the first pass through the equation set for the time step, the run is considered converged. The convergence tolerance can be adjusted by modifying the `CONVERGENCE_CHECK_PARAMETERS` in the input deck, which determines the relationship between the convergence check and the specified convergence tolerance. The Standard setting means the value must be below the specified convergence tolerance, Looser By 10 allows the value to be up to 10 times larger than the convergence tolerance, and Tighter By 10 requires the value to be less than one-tenth of the convergence tolerance.

While the default convergence criteria are suitable for many applications, it is important to monitor the impact of convergence on the solution for the specific application of interest. For steady-state simulations, this involves monitoring the desired solution quantity as the simulation approaches further convergence. For transient applications, it is necessary to evaluate convergence at each time step to understand its overall impact on the transient behavior.

Computing Time Statistics at the End of the Log File

Following the completion of the last time step, AcuSolve will print information regarding the input/output time, message passing time (for distributed memory parallel simulations), interface search time (if non-conformal interfaces are present), time associated with the solution of each stagger, and total time to complete the simulation. The memory usage of the simulation is also reported in the `.log` file. For distributed memory parallel simulations, the average and maximum CPU Times and Memory usage are printed to indicate if reasonable load balancing was achieved by the domain decomposition.

```
acuSolve: -------------------------------------------------------------
acuSolve: Input       CPU/Elapse time= 1.598300e+01 6.082000e+00 Sec
acuSolve: Output      CPU/Elapse time= 1.144870e+02 2.805000e+01 Sec
acuSolve: Exchange    CPU/Elapse time= 1.559400e+01 1.246370e+02 Sec
acuSolve: Intf.search CPU/Elapse time= 5.824000e+00 3.261900e+01 Sec
acuSolve: Flow stagger "flow":
acuSolve:   No of res/lhs formations =          800          800
acuSolve:   Elem.Form CPU/Elapse time= 1.120542e+03 9.437100e+01 Sec
acuSolve:   Solution  CPU/Elapse time= 4.096300e+01 1.908860e+02 Sec
acuSolve: Turbulence stagger "turbulence":
acuSolve:   No of res/lhs formations =          800          800
acuSolve:   Elem.Form CPU/Elapse time= 2.196290e+02 5.563100e+01 Sec
acuSolve:   Solution  CPU/Elapse time= 5.265000e+00 1.939800e+01 Sec
acuSolve: LnSlv. (host)   Elapse time= 1.891970e+02 Sec
acuSolve: Total CPU/Elapse time      = 1.558624e+03 4.946690e+02 Sec
acuSolve: Total Memory Usage         = 2.500000e+02 Mbytes
acuRun: -*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-
```

Figure 29:

Input and Output times

Report the amount of time that AcuSolve spent reading and writing files to disk. The differences between the input and output CPU/Elapsed times are caused by file system delays. When running AcuSolve in parallel, many processes may be trying to read/write to the same NFS mounted disk. When this happens, delays can occur, forcing AcuSolve to wait for file I/O. This causes the Elapsed time to increase while no CPU time is being used. Differences in CPU/Elapsed time can also be caused by processes competing for run time on an overloaded machine.

Exchange time

Indicates how much time AcuSolve spent passing messages between processes. For most simulations, the value of the Exchange time is 20% or less of the total time required to complete the simulation. When there is a significant difference between the CPU/Elapsed time, the implication is that there is a message passing bottle neck somewhere. This can be caused by a poorly performing network, or a switch that is unable to handle all of the traffic that it is receiving.

The Intf. Search time

Reports the amount of time that AcuSolve spends searching for matching faces for nonconformal interfaces (`INTERFACE_SURFACE` command). When a simulation containing `INTERFACE_SURFACES` is performed in a distributed memory parallel configuration, the search algorithm must communicate with other processes to find matching faces. Because of this, a significant amount of message passing must occur. When significant differences in CPU/Elapsed time are seen for this metric, it is typically due to delays in exchanging messages across the network/switch while searching for interface matches. For each stagger that is solved, AcuSolve reports a total of 3 metrics: No. of res/lhs formations, Elem. Form time, and Solution time.

No. of res/lhs formations

A tally of how many times the residual of the equation was computed, as well as how many times the left-hand-side matrix was formed. By default, these values should match each other, and reflect the total number of times the equation was solved.

> **Note:** The LHS matrix formation frequency can be adjusted, causing these two values to be different. However, this is an advanced option, and is rarely changed.

Elem. Form time

Refers to the amount of time required to form the LHS matrix and loop over all of the elements to compute the residual of the equations. Any significant deviations between the CPU/Elapsed time for this metric are primarily due to delays in exchanging messages across the network/switch or competing processes on a machine that is overloaded.

Solution time

The Solution time that is reported for each stagger refers to the amount of time required by the linear solver routines to converge to the desired tolerance. The total elapsed time of the simulation is typically dominated by the solution time for the various staggers. Any significant deviations between the CPU/Elapsed time for this metric are primarily due to delays in exchanging messages across the network/switch or competing processes on a machine that is overloaded.

Total CPU/Elapsed time

Represents the accounting of all steps that AcuSolve performs in completing the simulation. Any significant deviations between the CPU/Elapsed time for this metric are primarily due to delays in exchanging messages across the network/switch or competing processes on a machine that is overloaded. One other possible cause of large differences is running the machine out of memory, forcing it to write data to swap files.

Note: Simply summing all other timing metrics in the summary of the `.log` file will not yield the same value as the total CPU/Elapsed time. There are additional operations in the solver that are not profiled due to their insignificance.

AcuSolve Files

Files Generated During Solving

When you have a complete problem set up, including problem definitions and a generated mesh, the next step is to run AcuSolve to calculate a solution. When you run AcuSolve from HyperMesh CFD, files and directories are created. These files support two phases of the solution process.

First, HyperMesh CFD exports information about the problem and about the mesh. Next, AcuPrep and AcuSolve read these files and calculate a solution, By default, when you run AcuSolve from HyperMesh CFD, the options for exporting the settings needed for the solver and for launching AcuSolve are both turned on.

After you calculate a solution, you can delete the files used while calculating the solution. Care should be taken to avoid deleting any results files. If you want to calculate a solution for the problem again, you will need to regenerate the input files.

Export Solution Input Files

Table 5:

.inp	
Description	Input file containing all of the commands needed to run AcuPrep
Name	Same as `<problem>` name given by you
File type	ASCII

This file is required in order to calculate a solution. If you have generated an input file but not run AcuSolve, do not delete this file. Once a solution is calculated, this file can be deleted, but will need to be recreated if you want to recreate the solution.

Table 6:

MESH.DIR*.*	
Description	Directory containing all files associated with the mesh
Name	The complete directory should be treated as a single entity

MESH.DIR*.*	
File type	ASCII (default) or binary

The files in `MESH.DIR` contain the coordinate, connectivity, and elemental boundary conditions.

Table 7:

`.cnn`	File containing connectivity of elements within a volume set
`.crd`	File containing coordinates of nodes within the model
`.ebc`	File containing identifiers for element collections associated with each surface set

The files in this directory are required in order to calculate a solution. If you have generated an input file and created this directory, but not run AcuSolve, do not delete this directory. Once a solution is calculated, this directory can be deleted, but will need to be recreated if you want to recreate the solution.

Run AcuSolve

ACUSIM.DIR*.*	
Description	Directory containing files associated with the results from the solution
Name	The complete directory should be treated as a single entity and should not be tampered with
File type	ASCII

These files are required for results visualization. Deletion of any individual file or group of files may render the output database unreadable. If you want to view results after files are deleted, they will need to be recreated by running AcuSolve again.

Acusim.cnf	
Description	AcuSolve configuration file

Acusim.cnf	
Name	`Acusim.cnf`
File type	ASCII

This file is used when AcuSolve is run. This file should not be manually edited or deleted.

.echo	
Description	Complete list of commands provided to AcuSolve
Name	`<problem>.<runID>`, where `<runID>` is the sequential number that indicates the number of runs of AcuSolve for the particular problem
File type	ASCII

This file contains all of the commands needed to run AcuSolve. This file is useful for analyzing differences in settings between runs. This file should not be manually edited or deleted.

.inc	
Description	List of solution-strategy commands provided to AcuSolve
Name	`<problem>.ss.inc`
File type	ASCII

It can be deleted once a solution is reached.

.Log	
Description	Summary of the simulation, containing summary information for each sub-process spawned while calculating a solution. Used as a reference to load results into HyperGraph, the HyperMesh CFD plot utility, and HyperView.
Name	`<problem>.<runID>.Log`, where `<runID>` is the sequential number that indicates the number of runs of AcuSolve for the particular problem

.Log	
File type	ASCII

This file is required for results visualization. Deletion may render the output database unreadable. If this file is deleted, it will need to be recreated by running AcuSolve again.

Once the mesh process is complete, all of the boundary conditions are prepared and the solution strategy has been set, exporting the problem from HyperMesh CFD generates a set of files needed for AcuPrep and AcuSolve. The mesh is exported into a directory with default name `MESH.DIR` and the input file is written to `<problem>.inp`.

The input file is the most critical item associated with an AcuSolve simulation. It provides reference to all settings needed to initialize the solver with a series of command statements. The command statements may reference various other files that are associated with the mesh, initial conditions, boundary conditions or user subroutines.

The `.inp` file should be kept as a reference to the simulation, and along with `MESH.DIR`, these file comprise the simulation input. If either `.inp` or `MESH.DIR` are removed, it will not be possible to run the solver. These will need to be recreated prior to solving the problem.

AcuSolve is typically launched by a wrapper script called AcuRun. AcuRun controls the simulation procedure for AcuSolve. It requires that the input files, as described above, are properly referenced and available for the executable to read. First, AcuRun executes AcuPrep. AcuPrep will read the `.inp` file and generate a series of new files associated with the current run identifier. Since AcuSolve relies on a complete set of commands that are not explicitly defined in the `.inp` file, a new file `<problem>.ss.inc` is created to expand on the `AUTO_SOLUTION_STATEGY` command. The `.inc` file contains several additional commands associated with the solution strategy that may or may not have been defined in the `.inp` file. If not defined in the `.inp` file, commands such as `TIME_SEQUENCE`, `TIME_INCREMENT`, `STAGGER`, and `CONVERGENCE_CHECK_PARAMETERS` are automatically defined in the `.inc` file based on the default settings. The `.inc` file is then referenced by the `INCLUDE` command to fully define the solution strategy.

AcuPrep also generates a file with the extension `.echo`. The `.echo` file gives a complete listing of all commands required to run AcuSolve. Everything included in the `.inp` file and the `.inc` file are written to this file, which serves as a record of the full settings for each run that was made.

Once AcuPrep is complete, AcuRun executes AcuView (if necessary), followed by AcuSolve. AcuSolve generates a series of files that comprise the output from the simulation. The primary file associated with the output that you will interact with is the `.Log` file, by default named `<problem>.<run ID>.Log`. The `.Log` file provides a summary of the simulation progress and includes information from AcuPrep, AcuView, and AcuSolve. The `.Log` file also serves as a reference for loading the simulation results into various post processing applications, including HyperGraph and HyperMesh CFD. Note that these applications only parse the `.Log` file to extract the problem name, run ID, and working directory. The actual data that is loaded into the post-processing tools is read using an API to the AcuSolve solution database, which requires problem name, run ID and working directory as input.

Files in ACUSIM.DIR

`ACUSIM.DIR` is the primary data storage directory for results from a solution. `ACUSIM.DIR` contains various files that completely define the simulation, including results. The directory stores data in a variety of files, each corresponding to a different type of output. These files should not be managed manually.

File Grouping	File Extension	Description
Auxiliary Files	`.slv`	AcuPrep solver input
Basic	`.cli`	Replicated `.echo` file
	`.ddc`	Domain decomposition
	`.oqi`	Mask information
	`.top`	Solutions summary
	`.dat`	Domain connectivity
	`.osn`	Surface composition
Derived Quantity Output	`.odq`	
Nodal Output	`.out`	
Restart	`.rst`	
Time History	`.oth`	

AcuCleanDir

These files are required for results visualization. Deletion of any individual file or group of files may render the output database unreadable. If you want to view results after files are deleted, they will need to be recreated by running AcuSolve again.

A command line tool called AcuCleanDir is also available for managing the field output files in `ACUSIM.DIR`. Using this tool, it is possible to delete selective output types, steps, or an entire run.

> **Note:** This tool does not operate on time series data, only nodal data.

To use AcuCleanDir:

1. Open an AcuSolve Cmd Prompt from the Windows Start menu by clicking **Start** > **Altair <version>** > **AcuSolve Cmd Prompt**.
2. Change to the directory where your problem is saved.
3. Execute the acuCleanDir command with appropriate options.

- To delete run 2 of a problem named "channel":

```
acuCleanDir -pb channel -run 2 -delrun
```

- To delete all but the last time step of each result type:

```
acuCleanDir -pb channel -run 2 -ts abl
```

  where abl is a string that means "all but last".

- To delete nodal output and nodal residual outputs from time steps between the first and last available:

```
acuCleanDir -pb channel -run 2 -ts F:L:2 -types out,onr
```

  where the options for -types are:

out	NODAL_OUTPUT
odq	DERIVED_QUANTITY_OUTPUT
onr	NODAL_RESIDUAL_OUTPUT
ora	RUNNING_AVERAGE_OUTPUT
ota	TIME_AVERAGE_OUTPUT
rst	RESTART_OUTPUT
oee	ERROR_ESTIMATOR_OUTPUT
oae	ERROR_ESTIMATOR_OUTPUT - time averaged

A command line tool called acuCpProbeFiles can be used to summarize the results of a given AcuSolve simulation. Executing the script will copy the pertinent files from `ACUSIM.DIR` and the `.log` file into a directory specified by the command line argument –tdir (default is `PROBE.DIR`). The `PROBE.DIR` directory can be used by itself to interrogate results with AcuProbe.

Note: Nodal output is not stored within `PROBE.DIR`.

```
acuCpProbeFiles -pb channel -run 1 -tdir PROBE.DIR
```

Boundary Conditions

There are two types of boundary conditions to fully specify for a given CFD problem and to define the behavior of the solution at the boundaries of the computational domain: Dirichlet boundary conditions and Neumann boundary conditions. These conditions play a crucial role in determining the accuracy and stability of the numerical solution obtained through simulation.

Dirichlet boundary conditions prescribe the value of the solution at the boundary of the computational domain, often used to set the velocity, temperature, or other flow properties at the boundaries. In AcuSolve, nodal boundary conditions are equivalent to Dirichlet boundary conditions and are applied at three, four, or six nodes on each surface.

On the other hand, Neumann boundary conditions prescribe the normal derivative of the solution at the boundary. They are often used to represent conditions where a certain amount of heat is either entering or leaving the domain through the boundary. Typical examples of Neumann boundary conditions include mass flux (or mass flow rate) and heat flux. In AcuSolve, element boundary conditions are equivalent to Neumann boundary conditions and are applied at the quadrature points (element faces) of the surfaces.

It is possible to over specify a boundary condition by using combinations of element and nodal boundary conditions. In that case, nodal boundary conditions take precedence over element boundary conditions, meaning that the element boundary condition is ignored. If more than one nodal boundary condition is specified for a given variable on a given node, then the boundary condition with the highest precedence takes effect.

In the context of fluid dynamics simulations, the choice between Neumann and Dirichlet boundary conditions depends on the physical problem being modeled and the available information. For example, if the velocity at a boundary is known, a Dirichlet condition on velocity might be applied. If the mass flux at the boundary is known, a Neumann condition might be more appropriate.

All problems in fluid dynamics are governed by the same standard conservation equations. What makes the solution of each problem unique is the set of boundary conditions. Hence, boundary conditions have great impact on the accuracy of the simulation. Therefore, it is ensured that realistic constraints are enforced on the CFD model. This section discusses the variety of boundary conditions used in AcuSolve.

Details of boundary condition guidelines can be found in the Guidelines for Quality CFD Modeling, specifically under the section on Boundary Condition Sensitivity.

Flow Boundary Conditions

- Inflow Boundary Conditions

  The inflow boundary conditions are applied at the boundaries where the flow enters the domain. Various methods exist for specifying inflow conditions.

  - Velocity

    The velocity inflow condition is employed to define the velocity at the inlet. Velocity components can be specified in different coordinate systems (Cartesian, cylindrical system, spherical systems), or the velocity magnitude normal to the boundary surface, with the other two components set to zero. The pressure value at the inlet needs adjustment during calculation to align with the specified velocity provided at the inlet. The velocity value itself remains constant or can be adjusted using a multiplier function. In pipe flows, setting the

inflow with high precedence over the wall is recommended using the precedence parameter to ensure correct mass flow rate.

Additional input parameters are required for this boundary condition type, depending on the activated models. When turbulence is active, turbulence inlet parameters are necessary, including direct, auto, intensity length scale, intensity viscosity ratio and viscosity ratio.

- The direct turbulence input varies depending on the choice of turbulence models.
- Auto turbulence input type automatically assigns a turbulent state based on the selected turbulence model, turbulence intensity type, and turbulence flow type, applicable to all turbulence models.
- The intensity length scale turbulence input type approximates turbulence at the inlet using the turbulence intensity type and length scale. It is valid for all turbulence models.
- The intensity viscosity ratio turbulence input type calculates the turbulence parameters using turbulence intensity type and viscosity ratio. It is valid for two-equation models.
- The viscosity ratio turbulence input type calculates turbulence using the viscosity ratio. It is valid only for the Spalart-Allmaras turbulence model.

When the heat transfer equation is activated, the value temperature type is used to specify the temperature at the inlet. The heat flux temperature type is not used for the inlet boundary condition.

Radiation parameters are needed when using the P1 model or DO model.

> **Note:** The heat transfer (`temperature` = advective diffusive) must be first selected before activating radiation.

◦ Average Velocity

An average velocity is applied at the surface. A boundary layer profiles for velocity and turbulence fields are calculated based on the distance from no-slip walls and the estimated Reynolds number. This boundary condition offers a robust method for automatically specifying physically realistic inlet conditions. It is considerably more realistic than specifying a constant velocity condition for internal flow applications.

◦ Mass flux

A user-prescribed mass flux (equivalent to mass flow rate) is applied to the surface. A boundary layer profiles for velocity and turbulence fields are calculated based on the distance from no-slip walls and estimated Reynolds number. Below is the mass flow rate equation:

$$\dot{m} = \rho V A \qquad (104)$$

where $\dot{m}$: mass flow rate, ρ: density, V: velocity, A: inlet area.

Starting from v2022.3, mass flow inlet conditions are no longer limited to planar surfaces. Cylindrically curved or arbitrarily shaped surfaces can be used to define inlet mass flow conditions. The flow direction is relative to the local surface normal, allowing for inlets on circumferential surfaces.

◦ Flow Rate

A user-prescribed flow rate (equivalent to volumetric flow rate) is imposed on the surface. A boundary layer profiles for velocity and turbulence fields are calculated based on the distance from no-slip walls and estimated Reynolds number.

$$Q = VA \tag{105}$$

Atmospheric Boundary Layer

- User-defined boundary layer profile at the inlet

  As mentioned above, AcuSolve supports built-in boundary condition types (mass flux, flow rate, average velocity) that automatically generate a turbulent or laminar boundary layer profile. However, if these profiles are unsuitable for a particular application, you can generate a user-defined profile using one of the following options:

  - Using the nodal boundary condition, *type* = piecewise_linear or *type* = cubic_spline

    With this approach, AcuSolve interpolates the given velocity profile onto the mesh based on the data provided in a two-column array of dependent and independent variables. This approach is useful for profiles that are a function of a single coordinate direction.
  - Using the nodal boundary condition, *type* = scattered_data

    In this approach, AcuSolve interpolates the given velocity profile onto the mesh based on the data provided in a 4-column array of coordinates and velocity values. AcuSolve triangulates the input data to form a 2D mesh, then projects the coordinates of the inlet plane into this mesh to determine the mapping of the velocity. This approach is used when dealing with a profile that varies in more than one spatial direction.
  - User-defined function for the velocity field

    This is the most general approach, offering flexibility to accomplish a wide range of tasks.
- Pressure

  A pressure value is applied to all the nodes of the surface. This type of condition is useful for simulating scenarios where the pressure at the inlet is known, but the velocity might vary. Turbulence inputs are required when turbulence is activated under equation. Refer to the details of turbulence inlet parameters under the velocity section.
- Stagnation pressure

  A user-specified stagnation pressure (equivalent to total pressure) is explicitly defined at the inlet, while the corresponding velocity and static pressure are calculated as the simulation proceeds. Turbulence inflow parameters need to be specified, as in the velocity inflow type.

  Stagnation pressure (total pressure) is the sum of static pressure and dynamic pressure, reflecting the total energy of the flow. It can be expressed as:

  $$P_0 = P + \frac{1}{2}\rho V^2 \tag{106}$$

  where P: static pressure, ρ: density, V: velocity.

  The stagnation pressure type is weakly imposed boundary condition, making it a suitable choice for simulating flow through components such as fans, where changes in pressure and velocity are significant. It is suitable for both incompressible and compressible flow simulations.
- Velocity_pressure_temperature | mach_pressure_temperature

For a compressible inflow boundary condition, all three variables need to be specified. The simplest choice is primitive variables: velocity, pressure and temperature. Another choice could be Mach number, pressure and temperature. To specify these conditions, two inflow types are introduced: velocity_pressure_temperature or mach_pressure_temperature.

If Mach < 1, velocity is computed based on the given Mach number and temperature, while pressure is ignored. If Mach > 1, velocity is computed based on the given Mach number and temperature, while pressure and temperature are fixed.

- Outflow Boundary Conditions

  This type of boundary condition is applied at the boundaries where the flow exits the domain.

  - Pressure

    The pressure type is typically used for the outflow. AcuSolve offers three pressure types.

    - auto_pressure

      The user-specified pressure is used in subsonic flow, while it is ignored if lower than the computed pressure on the outflow surface.

    - fixed_pressure

      The user-specified pressure is directly applied at the outflow surface.

    - free_pressure

      The user-specified pressure is ignored, and a computed pressure extrapolated from upstream is applied at the outflow surface.

  - pressure_loss_factor

    At the outlet, the pressure loss factor, which can vary up to 1, can be set. Setting the *pressure_loss_factor* = 1 at the outlet may enhance the simulation stability if reverse flow is caused by an excessively short domain.

  - Back flow Condition

    The outlet boundary condition in AcuSolve employs a weakly enforced condition on the pressure field. It attempts to make the integrated pressure over the boundary equal to some value (zero by default). This works well as long as the flow is exiting from the boundary. If, for some reason, the flow reverses and attempts to enter the domain, AcuSolve provides a method for you to specify the temperature of the re-entering flow. This is a function of the back_flow_conditions. When this condition is active, AcuSolve automatically applies nodal boundary conditions to the temperature, turbulence, viscoelastic_stress, and species variables at the nodes of an outlet where the flow has reversed and is entering the domain.

- Wall boundary conditions

  This type of boundary condition is used for the physical walls in the simulation domain, specifying the velocity of the fluid at the wall.

  For instance, for a stationary wall in viscous flow, a wall boundary condition would set all components of the velocity to zero. In a simulation where the wall itself is moving at a certain velocity, a wall boundary condition would impose that velocity on the fluid at the wall. In short, a wall boundary condition can be expressed as $V_{\text{normal}} = 0$ and $V_{\text{tangential}} = V_{\text{wall}}$.

  When heat transfer equation is activated, you must provide the temperature wall boundary condition by selecting either the value temperature type or the flux temperature type. Choosing

the value temperature type requires providing temperature value, while the flux temperature type requires specifying heat flux. For external walls, you have an additional option, convective heat transfer option, which requires specifying convective heat coefficient and convective heat reference temperature.

If the flux specified for an exterior surface is 0, then it behaves as an adiabatic wall, meaning that no heat flux is applied to the external surface.

If the flux specified for an interior surface is 0, it is free to evolve during the simulation. This implies that there is still heat transfer across the surface via conduction, but the contribution of the heat flux from the surface triangles is zero. If you want an internal adiabatic wall, you will need to split the nodes on the surface to prevent heat transfer between the components.

- Slip Boundary Condition

  This type of boundary condition is applied to the surface where there is no normal velocity but has zero shear stress, indicating no change in the tangential velocity. An element boundary condition of type 'free' is imposed on the pressure, which is particularly important for curved surfaces. Additionally, a plane of symmetry can be specified with this type.

- Symmetry Boundary Condition

  For problems that have a plane of symmetry, exploiting the fact that the solution is symmetrical allows modeling only one half of the problem. The boundary type is then defined as symmetry at the symmetry plane. The geometry of the given surface is checked, and appropriate warning or error messages are issued if it is not flat. It is highly recommended to use symmetry instead of slip for all surfaces that are planes of symmetry. Mathematically the symmetry type is equivalent to the slip type, that is, zero flux across the plane and zero shear stress along the plane.

- Far-Field Boundary Condition

  This type of boundary conditions is used to represent conditions far away from the source of disturbance. A far_field boundary condition includes both a velocity-based inflow condition and a pressure-based outflow condition. At each point on the boundary, the angle between the outward normal to the surface and the user-specified velocity is used to determine whether to impose the inflow or outflow conditions. The flow at these boundaries is typically uniform. Far-field is generally used for external flows.

- Free Surface Boundary Condition

  As the name suggests, a free surface is the surface of the fluid not constrained by any physical boundary, such as the top surface of a sloshing box problem. This boundary condition imposes the normal component of mesh velocity to the flow velocity.

  This is used in combination with the ALE mesh motion equation to calculate the shape of the top surface of the fluid, such as wave propagation or generation. However, it does not support breaking waves. Defining the surface tension for the surface is necessary. If the top surface of the fluid is intended to remain flat, this is not needed.

Atmospheric Boundary Layer

The atmospheric boundary layer condition enables the specification of sustainable logarithmic inflow profiles for both wind velocity and turbulence. It is compatible with both one and two-equation turbulence models, offering flexibility in simulations. You can choose from a range of upstream roughness settings, including city, suburb, or forest, to closely reflect real-world conditions.

This boundary condition is indispensable for conducting external flow simulations in architectural engineering and wind turbine placement studies. Figure 1 depicts a schematic of the atmospheric boundary layer, illustrating the wind velocity profile at the inlet plane of the computation domain. This profile serves as a representative model of the roughness characteristics of the upstream terrain not encompassed within the computational domain.

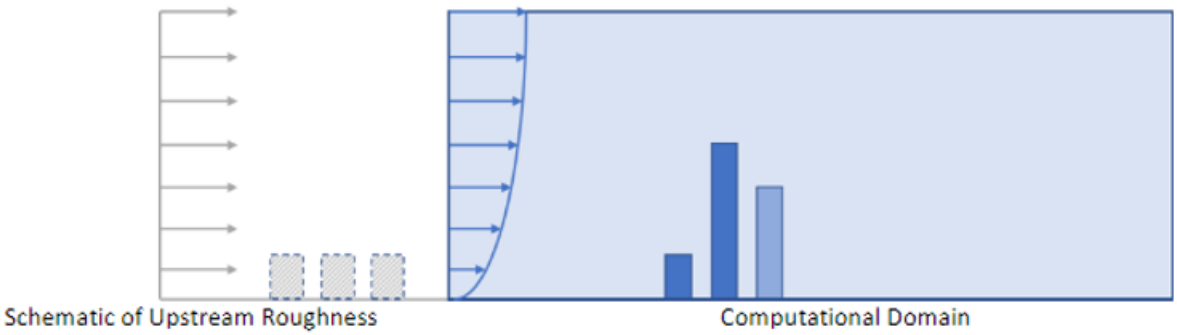


Figure 30: Atmospheric Boundary Layer Schematic

AcuSolve provides two distinct methods for generating inflow wind velocity profiles using the atmospheric boundary layer approach. The first option, friction_velocity, relies on the logarithmic law, while the second sample_height_velocity, utilizes the power law.

In the friction_velocity approach, this method employs the logarithmic law to determine the wind velocity profile in the vertical direction, which is defined as

$$U(y) = \frac{u^*}{\kappa} \ln\left(\frac{y + y_0}{y_0}\right) \tag{107}$$

where y is the distance from the ground, u^* the friction velocity, and κ is the von Karman constant ($0.40 \sim 0.42$). y_0 represents the surface roughness.

You must specify the ground roughness height (y_0) corresponding to the terrain type and the atmospheric friction velocity value (u^*) to generate the atmospheric boundary layer profile at the inlet. The friction velocity, representing the wall shear stress divided by the fluid density, $u^* = \sqrt{\frac{\tau_w}{\rho}}$ poses a challenge for the use of the input value, but a rule of thumb suggests it is roughly one tenth of the wind velocity, ranging from 0.05 m/s in light winds to approximately 1 m/s in strong winds. For atmospheric boundary layer simulations, the fluid density should correspond to air density. Typical surface roughness values for different terrains are available in table 1, or you can specify your own using the atmosphere_ground_roughness command when `atmospheric_roughness_type` is set to user_value.

Table 8: Typical Terrain Types and Surface Roughness

Terrain	**Surface roughness**
ocean	0.001 m
low grass	0.020 m

Terrain	Surface roughness
high grass	0.060 m
forest	1.000 m
suburb	0.300 m
town	1.000 m
city	2.500 m

Inlet turbulence properties are determined based on the chosen turbulence model. For instance, with the Spalart-Allmaras (S-A) model, you can specify the eddy viscosity.

$$v = \kappa\left(y + y_0\right)\left(u^*\right) \tag{108}$$

where v is the eddy viscosity.

For standard, RNG, realizable k-epsilon models, the turbulent kinetic energy (k) and dissipation rate (ε) at the inlet plane are calculated using the following equations:

$$k = \frac{\left(u^*\right)^2}{\sqrt{C_\mu}} \tag{109}$$

$$\varepsilon = \frac{\left(u^*\right)^3}{\kappa\left(y + y_0\right)} \tag{110}$$

Similarly, for k-omega and Shear Stress Transport (SST) models, the turbulent dissipation rate (ω) is determined accordingly. The turbulent kinetic energy is the same as the one used for the k-epsilon turbulence model.

$$k = \frac{\left(u^*\right)^2}{\sqrt{C_\mu}} \tag{111}$$

where C_μ is a model constant of the k-ε model (= 0.09).

$$\omega = \frac{u^*}{\sqrt{C_\mu}\kappa\left(y + y_0\right)} \tag{112}$$

When `atmospheric_reference_velocity_type` is set to sample_height_velocity, the vertical velocity profile at the inlet is determined using the power law,

$$U(y) = U_{\text{ref}}\left(\frac{y}{y_{\text{ref}}}\right)^a \tag{113}$$

where U_{ref} is the reference velocity at the reference height y_{ref}, a is the power law coefficient, which is a function of surface roughness.

An empirical ratio of 30 between inflow roughness and standard wall roughness is typically assumed, aiding in the determination of surface roughness values. For instance, if the atmospheric boundary layer inflow has a roughness of 0.001m, the ground with standard turbulence wall function should have a roughness of 0.03m.

AcuSolve Solver Features

This section on AcuSolve solver features covers the description of various solver features available in AcuSolve such as heat transfer, fluid structure interaction and turbulence modeling. Commercial solvers for fluid flow analysis come in many forms. They use a collection of numerical methods and approximation approaches to solve the governing continuum equations discussed in this manual.

AcuSolve is a general purpose CFD flow solver that is used in a wide variety of applications and industries. The flow solver is architected for parallel execution on shared and distributed memory computer systems and provides fast and efficient transient and steady state solutions for standard unstructured element topologies.

AcuSolve is based on Galerkin Least Squares (GLS) finite element method. The GLS formulation provides second-order accuracy for spatial discretization of all variables and utilizes tightly controlled numerical diffusion operators to obtain stability and maintain accuracy. In addition to satisfying conservation laws globally, the formulation implemented in AcuSolve ensures local conservation for individual elements. Equal order nodal interpolation is used for all working variables, including pressure and turbulence equations. The semi discrete generalized alpha method is used to integrate the equations in time. This approach has been verified as being second-order accurate in time.

AcuSolve uses a time marching procedure to solve both steady state and transient simulations. In the case of steady state simulations, the inertia (mass) terms of the conservation equations are only included in the Galerkin part of the finite element weighted residual formulation. This inclusion adds stability to the nonlinear iterations. Other parts of the finite element formulation (such as the least-squares operator) do not include the inertia terms. This exclusion accelerates non linear convergence to steady state at the expense of time accuracy. The initial time step size is set to a significantly large value (1.0e+10).

For transient simulations the inertia terms are included in all the operators of the finite element formulation in order to preserve time accuracy.

The resultant system of equations is solved as a fully coupled pressure/velocity matrix system using a preconditioned iterative linear solver. The iterative solver yields robustness and rapid convergence on large unstructured meshes even when high aspect ratio and badly distorted elements are present.

AcuSolve consists of multiple features designed to tackle various flow physics such as temperature flow, solar radiation and rigid body dynamics.

Heat Transfer

AcuSolve supports various features for the modeling of the heat transfer phenomenon.

- Convective Heat Transfer: Heat transfer due to the motion of molecules within a fluid. This includes both natural and forced convection.
- Conjugate Heat Transfer: Heat transfer between solids and fluids.
- Radiation Heat Transfer: Heat transfer due to radiation between surfaces having high temperatures.
- Solar Radiation: Heat transfer due to incident solar radiation on a surface.

- Additional features: Thermal shells, viscous heating and compression heating.

AcuSolve Heat Transfer Methodology

AcuSolve uses an advective-diffusive equation governing the transport of enthalpy.

$$\rho\frac{\partial h}{\partial t}+(\rho\vec{u}\cdot\nabla)h \quad =\rho s \quad +\nabla\cdot q \tag{114}$$

This approach permits the conservation of energy with the Galerkin Least Squares formulation.The temperature is derived from the state equation specified for the flow.

There are two common approaches used for the solution of thermal problems in AcuSolve.

- Segregated approach: The enthalpy equation is solved separately from the flow equations. This approach can be further divided into two parts:
- Sequential solving: The enthalpy equation is solved in a sequential manner after solving the coupled flow equations using feed- forward of the velocity field into the enthalpy equation. This approach is appropriate for cases where the temperature field does not affect the flow field significantly. This can offer benefits in speed since the iterations on flow field are not carried along as part of the temperature equation. If the flow field has already converged sufficiently, additional iterations are wasted. This approach also enables “frozen flow” thermal simulations where the flow field is held constant, but different scenarios of the temperature field are investigated.
- Concurrent solving: The enthalpy equation is solved in a sequential manner after the flow equations but there is feedback of the temperature dependent variable from the enthalpy equation back to the flow equations during each iteration. This approach is appropriate for cases involving temperature dependent material properties.
- Coupled approach: The enthalpy equation is coupled into the global system and solved in conjunction with the flow equations. This approach is often times more stable for cases exhibiting a large degree of coupling between the temperature and flow fields, that is, buoyancy driven flows.

AcuSolve Solar Radiation Methodology

AcuSolve uses an ideal grey surface solar radiation model to calculate the solar heat flux. The fluxes are computed using a ray trace algorithm and five optical properties of the surface, specular transmissivity, diffuse transmissivity, specular reflectivity, diffuse reflectivity and absorbtivity. These properties are constrained by the relation:

$$\tau_s(\theta)+\tau_d(\theta)+\rho_s(\theta)+\rho_d(\theta)+\alpha(\theta)=1 \tag{115}$$

AcuSolve Thermal Shell Methodology

AcuSolve permits the definition of thermal shells as material medium to simulate heat transfer in thin solid mediums. This is particularly useful when the thickness of the component makes it inconvenient to use it as a solid medium. The shell medium supports only bricks and wedges.

AcuSolve can model shell layers of varying thickness and material properties. There are two mechanisms by which the heat transfer equation is solved in the thermal shell depending on the number of layers:

- Single Layer: The elements in a single layer shell are considered as full volume elements. The coordinates of the nodes on the opposite faces of the shell are offset by the thickness provided

and the heat transport equations are used to determine the heat transfer within the shell. For this scenario the thermal shell includes all heat transfer effects that are present in a three dimensional volume element.

- Multiple Layer: The nodes on the opposite faces have the same coordinates but have different node numbers in order to support the temperature differences across the shell. A one dimensional heat equation, through the shell thickness at the element nodes, is derived by neglecting thermal inertia and conduction parallel to the surface. This means that the heat flux is the same in all the layers. If each layer material model has a constant conductivity then this simplified heat equation is solved exactly. If any of the conductivities is a function of temperature then a two-pass procedure is used to approximately solve the resulting nonlinear system. Once the temperatures are known at the corners of all the layers in each element they can be interpolated to the quadrature points in the usual manner.

Viscous and Compression Heating

AcuSolve can simulate heat transfer due to viscosity and compressibility of the fluid in case of highly viscous flows and variable density flow models. The terms corresponding to viscous and compression heating are added to the enthalpy transport equation:

$$\rho\frac{\partial h}{\partial t}+(\rho\vec{u}\cdot\nabla)h \quad =\rho s \quad +\nabla\cdot q+\tau:\nabla\vec{u}+\frac{Dp}{Dt} \tag{116}$$

where

- $\tau:\nabla\vec{u}$ is the viscous heating term
- $\frac{Dp}{Dt}$ is the material derivative of pressure which describes the compression heating

AcuSolve Enclosure Radiation

Enclosure Radiation Methodology

AcuSolve simulates the surface to surface radiation using a two step approach:

- View factor computation: In radiative heat transfer view factor is the proportion of radiation incident on one surface due to another surface. The view factors for each facet defining the radiation enclosure are computed using the hemicube algorithm and smoothed using least squares method as a pre-processing step. The view factors are not recomputed during the simulation.
- Heat flux addition: The radiative heat flux, based on view factors, computed using the Stefan-Boltzmann law and the total radiosity, based on Kirchoff's law, is added to the enthalpy transport equation during the solver run.

The enclosure radiation model is supported only on fluid mediums, that is, the fluid side of the fluid/solid surface.

View factor computation is an important point when dealing with moving mesh simulation. Since the view factors are not recomputed during the solver run the boundary elements forming the enclosure should not deform.

The total emissive power of a grey surface is given by the Stefan-Boltzmann law:

$$W_i = \epsilon_i \sigma \left(T_i + T_{off} \right)^4 \tag{117}$$

where ϵ_i is the emissivity of surface i, given by an `EMISSIVITY_MODEL` command; σ is the Stefan-Boltzmann constant, given by *`stefan_boltzmann_constant`*; T_i is the temperature of surface i; and T_{off} is the offset to convert to an absolute temperature, given by the *`absolute_temperature_offset`* parameter of the `EQUATION` command. In addition, each surface receives part of the total radiosity from each of the radiation surfaces:

$$G_i = \sum_j F_{ij} J_j \tag{118}$$

where G_i is the total irradiation of the surface i, F_{ij} is the view factor from the surface i to surface j and J_j is the total radiosity from surface j. The total radiosity of surface i is

$$J_i = W_i + \left(1 - a_i \right) G_i \tag{119}$$

where a_i is the absorptivity and $1 - a_i$ is the reflectivity. From Kirchhoff' law and the grey surface assumption, $a = \epsilon$. The net radiation heat flux is thus $q_i = G_i - J_i = a_i G_i - W_i$ and is added to the temperature equation. The radiation equation solves for all the radiation heat fluxes coupled together with the temperature equation.

View factors F_{ij} are purely geometric entities defined as

$$F_{ij} = \frac{1}{A_i} \int_{A_i} \int_{A_j} \frac{\cos\theta_i \cos\theta_j}{\pi r^2} \delta_{ij} \mathrm{d}A_i \mathrm{d}A_j \tag{120}$$

where A_i and A_j are the areas of surfaces i and j, respectively; θ_i and θ_j are the angles between the line connecting dA_i to dA_j and the normals to surfaces dA_i to dA_j; respectively; r is the distance from dA_i to dA_j; and δ_{ij} is the visibility function, it is equal to one if dA_i to dA_j see each other, otherwise it is zero.

View Factor Nomenclature

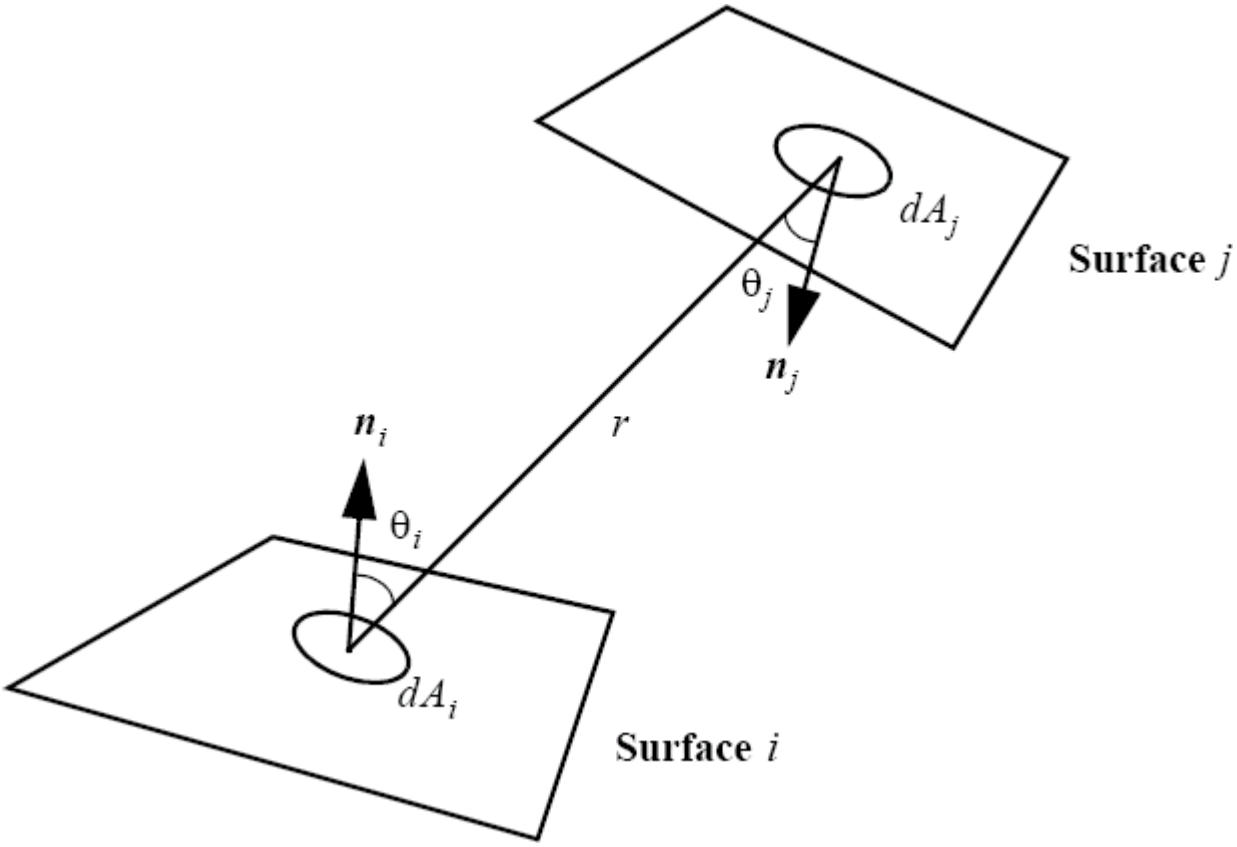


Figure 31:

The view factors are computed using the hemicube algorithm. In a nutshell, in this algorithm a hemicube is placed on the centroid of each radiation facet i. The hemicube is discretized into $3n^2$ pixels; where n is given by `num_hemicube_bins`. All surfaces (facing surface i) are then projected onto this hemicube. The Z-buffering algorithm is used to compute the visibility. Once all surfaces are projected, the pixels weighted by their view factor increments are added up to determine a row of $F_i=\{F_{ij}\}$.

Hemicube Discretization

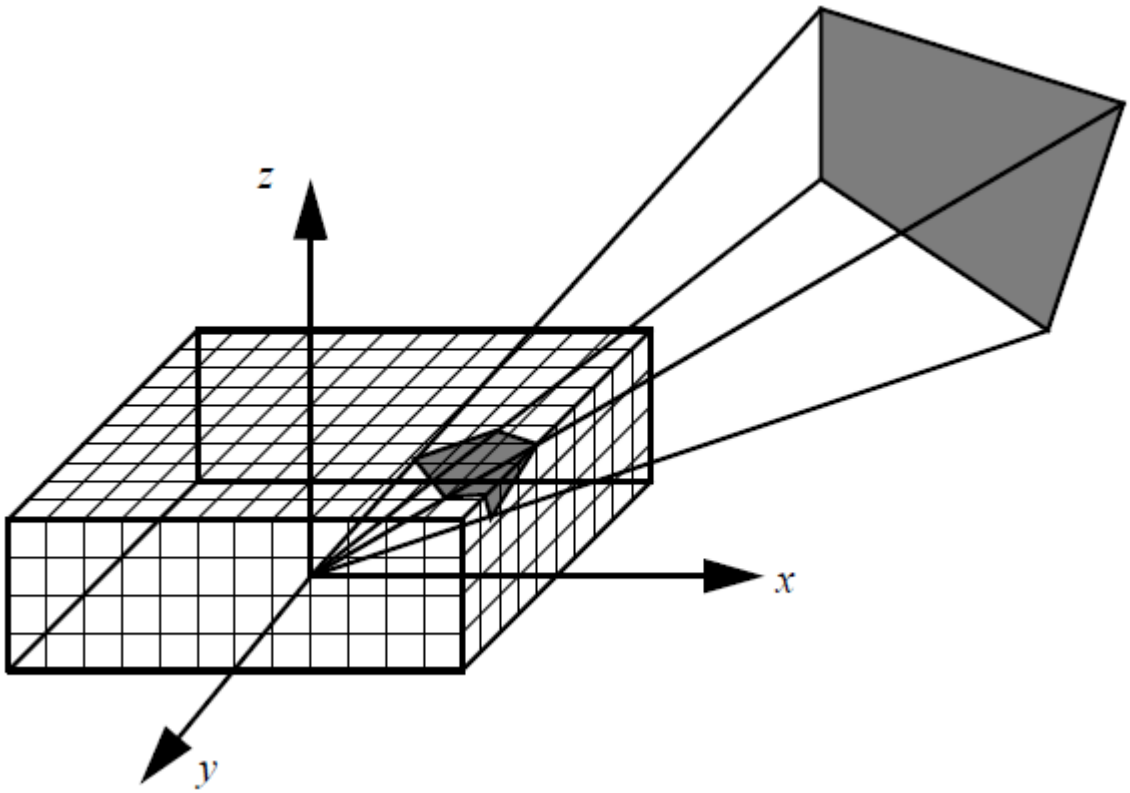


Figure 32:

The hemicube algorithm has three major assumptions, and hence sources of errors: aliasing - the true projection of each visible face onto the hemicube can be accurately accounted for by using a finite resolution hemicube; proximity - the distance between faces is great compared to the effective diameter of the faces; and visibility - the visibility between any two faces does not depend on the position within either face. The aliasing error may be reduced through the use of larger *num_hemicube_bins* values, at the expense of computational cost. The computational cost is proportional to n^2 for sufficiently large values of n. In addition, a random jittering algorithm and a non-uniform hemicube pixelization are employed to minimize the aliasing error. The proximity and visibility errors occur when two facing facets are too close to each other. In this case, the geometry of one or both facets cannot be sufficiently well-represented by their centroids. These errors can be reduced by splitting each poorly-represented facet into smaller segments (sub-surfaces), computing the view factors separately for each sub-surface and then adding them up. The computational cost is proportional to the number of sub-surface divisions. The *max_surface_subdivision* parameter may be used to trade-off accuracy against cost.

A least-squares smoothing is automatically applied to the computed view factor matrix to enforce reciprocity and conservation:

$A_i F_{ij} = A_j F_{ji}$ (reciprocity)

$\sum_j F_{ij} = 1$ (conservation)

> **Note:** Due to the cost of computing the view factor matrix, it is computed in a pre-processing step and stored on disk. This means that the view factors are not updated during a simulation, even when mesh movement is present.

The default for the Stefan-Boltzmann constant is given in MKS units: $\sigma = 5.670 \times 10^{-8}\, W/(m^2 \circ K^4)$.

In British units it is: $\sigma = 4.756 \times 10^{-11}\, Btu/(s\, ft^2 \circ R^4)$.

In general, this constant must be converted to be consistent with the units chosen for use in the problem.

Radiation Modeling with Participating Media

In AcuSolve enclosure or surface to surface radiation models, effect of media between surfaces is ignored. This assumption is acceptable when you are dealing with lower temperature fluids. Nonetheless, while you have semi-transparent media like glass or high temperature gases like in flames, effect of media should be considered in heat transfer analysis. Two models are available in AcuSolve: A simple one equation P_1 model and more detailed but expensive discrete ordinates (DO) model.

Radiative Transfer Equation (RTE)

Radiative energy balance in a participating media is governed by the following integro-differential equation, known as the radiative transfer equation (RTE):

$$\underbrace{\Omega \cdot \nabla I(r,\Omega)}_{\text{rate of change}} + \underbrace{\kappa I(r,\Omega)}_{\text{Absorption}} + \underbrace{\sigma_s I(r,\Omega)}_{\text{Scattering loss}} = \underbrace{f(r)}_{\text{Emission}} + \underbrace{\frac{\sigma_s}{4\pi}\int_{4\pi} I(r,\Omega')\phi(\Omega',\Omega)d\Omega'}_{\text{Scattering addition}} \tag{121}$$

where I is the radiation intensity, r is the spatial vector, Ω is the unit directional vector, Ω' is the scattering directional vector, κ is the absorption coefficient, and σ_s is the scattering coefficient, $f(r)$ is an emission, n the refractive index, and σ is the Stefan-Boltzman constant (5.67×10^{-8} W m^{-2} K^{-4}). T is the temperature (K).

- P1 radiation model
- Discrete Ordinates (DO) model

P1 Radiation Model

The P_1 model is the lowest order P_N (spherical harmonics) type radiation model. The method reduces the five independent variables of the Radiative Transfer Equation (RTE) into a PDE that is relatively simple in comparison. The model is the most computationally efficient of the radiation models in AcuSolve, but it can lose accuracy, under certain conditions, for optically thin media. It performs best in scenarios where the radiative intensity is near isotropic.

Governing Equation

P_1 approximation and assumptions

The P_1 model is derived from the general P_N formulation (a spherical harmonic series expansion of the radiative intensity for the angular variable) by assuming that the series is limited to four terms and integrating over all solid angles. From the first harmonic in the series approximation, the divergence of the radiative flux (q) can be derived by integrating the RTE over all solid angles as

$$\nabla \cdot q = \kappa\left(4n^2\sigma T^4 - G\right) \tag{122}$$

where G is the incident radiation $\frac{\sigma_s}{4\pi}\int_{4\pi} I(r,\Omega')\phi(\Omega',\Omega)d\Omega'$, κ is the absorption coefficient, n the refractive index, and σ is the Stefan-Boltzman constant (5.67×10^{-8} W m^{-2} K^{-4}), q is the radiative flux.

Additionally, a second vector equation can be derived from the other three harmonic terms for the radiative flux

$$q = \Gamma \nabla G \tag{123}$$

where Γ is the diffusion coefficient.

By taking the divergence of (2), and substituting into this the right hand side of equation (1), leads to elimination of the heat flux. The final diffusion reaction equation describing the transport of incident radiation is given by

$$\Gamma \nabla^2 G - \kappa(4n^2\sigma T^4 - G) = 0 \tag{124}$$

where Γ is a diffusion coefficient given by

$$\Gamma = \frac{1}{3(\kappa+\sigma_s) - A_1\sigma_s} \tag{125}$$

where κ is the absorption coefficient, σ_s is the scattering coefficient, and A_1 is the linear-anisotropic phase function.

Anisotropic scattering

The implementation in AcuSolve includes the ability to model linear anisotropic scattering

$$\phi(\Omega' \cdot \Omega) = 1 + A_1\Omega' \cdot \Omega \tag{126}$$

where Ω is the unit directional vector in the scattering direction, Ω' is the unit directional vector in the incident radiation direction. This term is included in the scattering term of the RTE along with the four term spherical harmonic expansion of the radiative intensity field (the first term being isotropic and the other three anisotropic). The final form of the P_1 approximation in equation (3) includes this term. The values of the phase function A_1 have the following meaning:

- A_1 = 1: More radiation is scattered in the forward direction
- A_1= -1: More radiation is scattered in the backward direction
- A_1= 0: Isotropic scattering

Coupling to energy equation

The source term in equation (1) can be substituted into the energy equation as a negative source since a local increase in radiative heat flux is due to a local decrease in thermal energy.

In AcuSolve the default stagger sequence when the P_1 radiative heat transfer solver is enabled is:

1. Solve the energy equation with source term ($\kappa(4n^2\sigma T^4 - G)$), where G is zero for the time step

2. Pass temperature to radiation solver and solve equation (3)
3. Repeat until converged

Boundary Conditions

Marshak's boundary condition

Based on the assumption that the walls are diffused gray surfaces, that is, independent of wavelength, the appropriate wall boundary condition is

$$n \cdot \nabla G = \frac{\varepsilon}{2(2-\varepsilon)} \left(4n^2 \sigma T_w^4 - G_w \right) \tag{127}$$

where ε is the surface emissivity, T_w is the wall temperature, and G_w the wall incident radiation. The boundary radiative heat flux can be calculated from the incident radiation and the temperature at the wall:

$$q_w = \frac{\varepsilon}{2(2-\varepsilon)} \left(G_w - 4n^2 \sigma T_w^4 \right) \tag{128}$$

Discrete Ordinates (DO) Model

Governing equation

The governing equation is the radiative transfer equation limited to a finite number of directions (or ordinates)

$$\Omega \cdot \nabla I(r, \Omega) + \kappa I(r, \Omega) + \sigma_s I(r, \Omega) = f(r) + \frac{\sigma_s}{4\pi} \int_{4\pi} I(r, \Omega') \phi(\Omega', \Omega) d\Omega' \tag{129}$$

$$f(r) = \kappa I_b(r) = \kappa \frac{\sigma T^4}{\pi} \tag{130}$$

Scattering term (source term)

The integral over all the directions in equation (1) is replaced by a numerical quadrature for N different ordinate directions (Ω_i)

$$S = \frac{\sigma_s}{4\pi} \sum_{j=1}^{N} w_j I(r, \Omega_j) \phi(\Omega_j, \Omega_i) \tag{131}$$

The phase function, $\phi(\Omega_j, \Omega_i)$, is given by

$$\phi(\Omega' \cdot \Omega) = 1 + A_1(\Omega' \cdot \Omega) \tag{132}$$

Boundary conditions (RTE)

Diffused surface

If a surface emits and reflects diffusely, the exiting intensity is directionally independent and is given by

$$I(r_w, \Omega_i) = \varepsilon(r_w) I_b(r_w) + \frac{1-\varepsilon(r_w)}{\pi} \sum_{n \cdot \Omega_j < 0} w_j I(r_w, \Omega_j) \left| n \cdot \Omega_j \right|, \quad \ldots\ldots n \cdot \Omega_j > 0 \tag{133}$$

Specular and diffuse surface

The diffused fraction defines the proportion of reflected radiation intensity at a surface which is diffused, that is, the reflection may also have a specular component. If the radiation intensity reflection coefficient at the surface is defined by

$$\rho = \rho^S + \rho^D = 1 - \varepsilon \tag{134}$$

then the diffused reflection coefficient, ρ^D, is defined in terms of the diffused fraction a and the emissivity of the surface by

$$\rho^D = a(1-\varepsilon) \tag{135}$$

and the specular reflection coefficient

$$\rho^S = (1-a)(1-\varepsilon) \tag{136}$$

If a=1, then the reflection at the surface is completely diffused. If a=0 then the reflection is specular. The outgoing intensity, I, at the surface in terms of the above two reflection coefficients is given by

$$I(r_w, \Omega_i) = \varepsilon(r_w) I_b(r_w) + \frac{\rho^D(r_w)}{\pi} \sum_{n \cdot \Omega_j < 0}^{N} w_j I(r_w, \Omega_j) \left| n \cdot \Omega_j \right| + \rho^S(r_w) I(r_w, \Omega_j), \quad n \cdot \Omega_j > 0 \tag{137}$$

where the first terms represent emission from the surface, the second term the diffused component incoming radiation heat flux and the third the specular component. The diffused component represents a sum over all radiation intensities along ordinates that are incident to the surface (that is, a hemisphere of incoming radiation to the surface); n is the normal into the domain and Ω_j the jth ordinate direction. The ordinate direction (Ω), the total number of ordinate directions (N) and the weights (w) are automatically defined by the order of the radiation_quadrature (S2, S4, S6, S8 & S10). The specular ordinate direction (Ω_S) is the direction that the radiation intensity must strike the surface to reflect in a specular fashion along the outgoing ordinate direction, Ω_i, and is given by

$$\Omega_s = \Omega_i - 2(\Omega_i \cdot n) n \tag{138}$$

which means the angle that incident radiation intensity strikes the surface equals the angle of reflection.

Boundary conditions (Energy equation)

Interface and outflow/inflow boundary conditions

At an opaque interface between participating and non-participating media or outflows/inflows a radiative heat flux must be added to the boundaries in the energy equation. This flux is given by

$$Q_{rad} = \varepsilon \left(\sum_{\Omega_j \cdot n > 0} w_j I(r_w) \left| \Omega_j \cdot n \right| - n^2 \sigma T_w^4 \right) \tag{139}$$

For an opening, that is, outflow or inflow, the black body intensity used in the calculation of the outgoing intensity at the surface is based on the opening temperature of the surrounding:

$$I_b = \frac{\kappa n^2 \sigma T_{open}^4}{\pi} \tag{140}$$

while for an interface it is based on the current temperature solution.

Output metrics

Two directionally integrated output metrics can be derived from the radiative intensities: incident radiation and radiative heat flux.

Incident radiation

Incident radiation is the total intensity impinging on a point from all directions and is given by

$$G = \sum_{i=1}^{N} w_i I(r) \tag{141}$$

where I is the intensity in direction i, N the number of ordinates, w the weights.

Interface Between two Semitransparent Media

At the interface between two semitransparent media (referred to as medium 1 and medium 2 below), radiative intensity is both transmitted and reflected. The proportion of transmitted and reflected intensity at the interface depends on: the refractive indices (n_1, n_2) of the two media; the incident angle that radiative intensity strikes the surface; and the diffuse fraction of the surface.

Reflection and transmission for specular interfaces

Reflection at an interface is governed by the angle of incidence of a radiative intensity and the refractive indices of the two media. The image below shows the different refracted and reflected rays between two media.

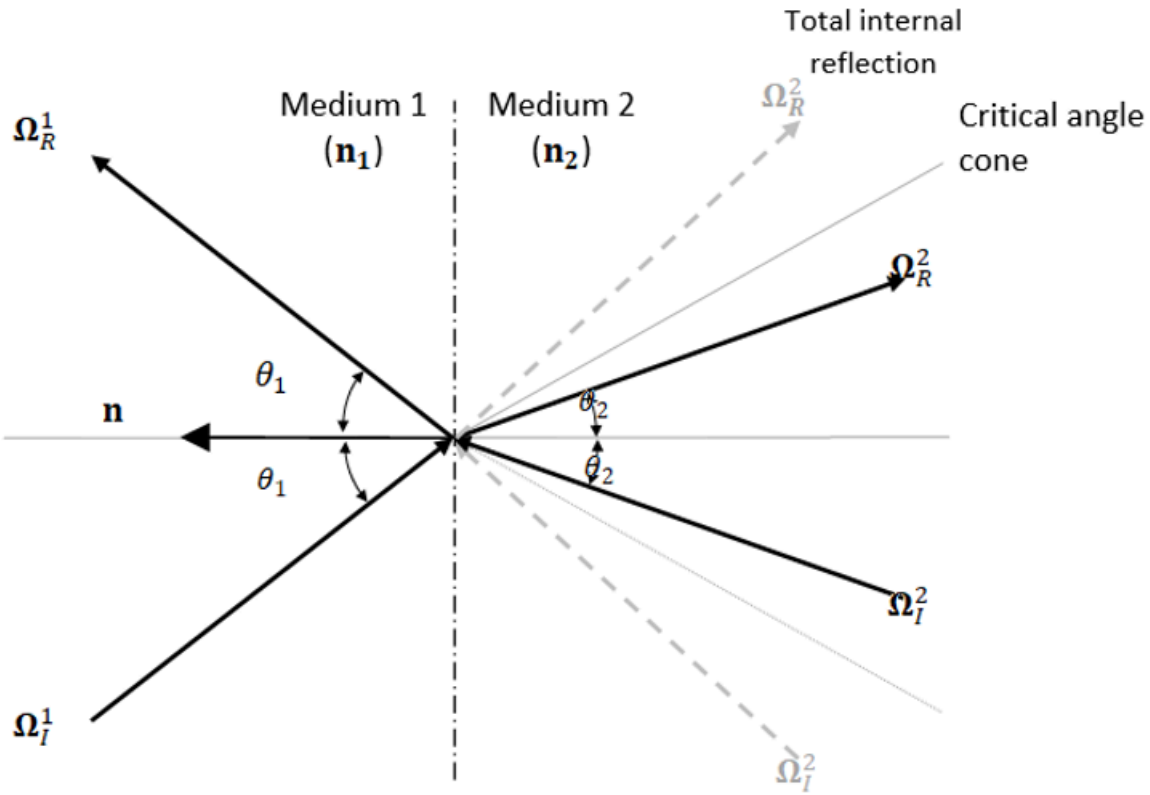


Figure 33: Reflected and refracted directions at the interface between two participating media of different refractive indices ($n_1 < n_2$). For the medium of higher refractive index if the incoming direction is greater than the critical

angle total internal reflection occurs (no transmission occurs across the interface). This is represented by the gray dashed lines in the image.

The cosine of the incident angle for the incoming ordinate is given by

$$\cos\theta_1 = \Omega_I^1 \cdot n \tag{142}$$

where n is the outward facing normal direction at the interface (towards the second medium) and Ω_{I1}^1 is the unit direction of incoming radiative intensity to the surface, given by

$$\Omega_{I1}^1 = \Omega_R^1 - 2(\Omega_R^1 \cdot n)n \tag{143}$$

where Ω_{I1}^1 is the unit reflected ordinate direction vector and also represents the current ordinate direction being solved. The equivalent calculation can also be performed for medium two.

Radiative intensity that is transmitted into a second medium undergoes refraction governed by Snell's law,

$$n_1 \sin\theta_1 = n_2 \sin\theta_2 \tag{144}$$

where n_1 and n_2 are the refractive indices of mediums. θ_1 and θ_2 are the angles of incidence and refraction of radiative intensity relative to the interface normal, respectively. This can also be represented in vector form by

$$n_1\left(n \times \Omega_I^1\right) = n_2\left(n \times \Omega_R^2\right) \tag{145}$$

The incoming direction vector in medium two for a ray refracted from medium two to one is given by

$$\Omega_I^2 = \frac{n_1}{n_2}\Omega_R^1 + \left(\frac{n_1}{n_2}\cos\theta_1 - \sqrt{1 - \left(\frac{n_1}{n_2}\right)^2 (1 - \cos\theta_1)}\right)n \tag{146}$$

The above expression is valid providing the expression under the radicand is greater than zero; otherwise total internal reflection occurs.

The actual reflected and refracted directions differ slightly from the calculated direction since these directions will unlikely coincide with a discrete ordinate direction. Since the number of directions is governed by the order of radiation quadrature, higher quadrature orders are more accurate for interface problems.

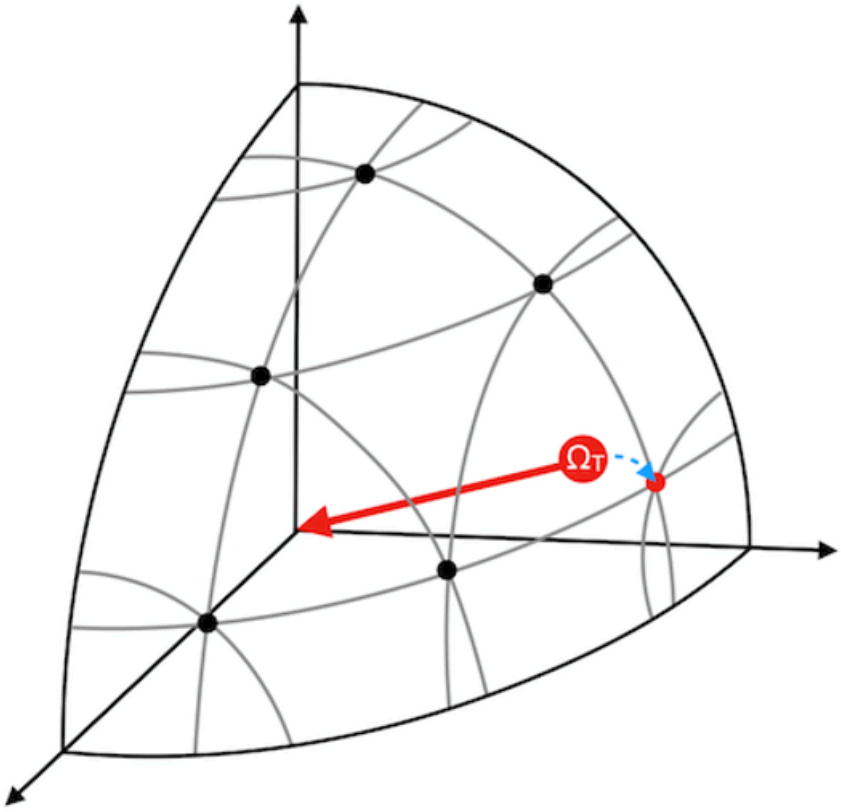


Figure 34: Octant of angular quadrature with transmission direction. The calculated transmission direction (Ω_T) is shifted to match the nearest quadrature point.

Depending on the refractive indices of the two media and the angle of incidence, θ_1, the proportion of radiation intensity that is reflected or transmitted will vary. If $n_1 < n_2$, then the radiative intensity in medium one will be partially reflected and partially transmitted into a cone defined by the critical angle, θ_c, which is given by:

$$\theta_c = \sin^{-1}\frac{n_1}{n_2} \quad \text{when } n_1 < n_2 \tag{147}$$

and defines a cone in 3D (see the images below). The critical angle is defined by a ray that grazes the surface on the side of medium 2 and is transmitted exactly at the critical angle in medium 1. As an example, if $n_1 > n_2$ and $\theta_1 > \theta_c$, the direction of incoming radiation is greater than the critical angle and total internal reflection will occur. This means the incoming ray is reflected at the same angle of incidence and no transmission occurs.

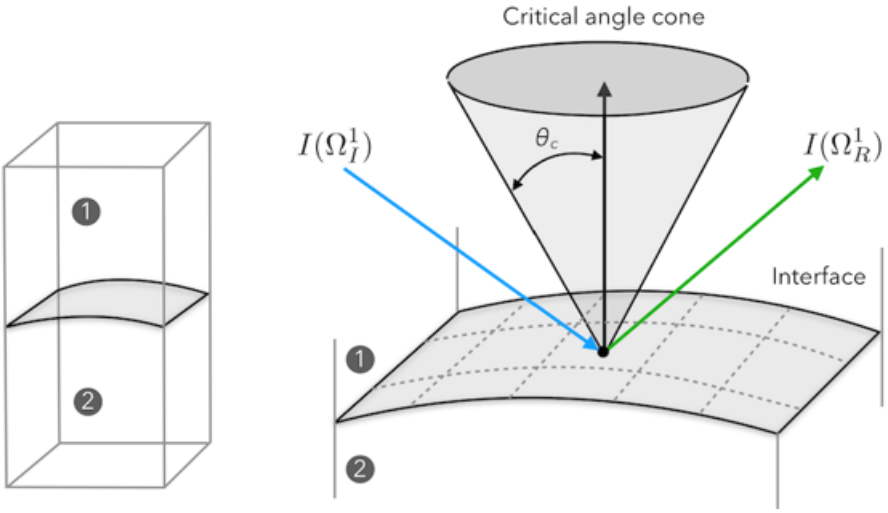


Figure 35: Total internal reflection of intensity rays at an interface ($n_1 > n_2$). The critical angle cone defines the angle outside which total internal reflection occurs, that is, $\theta_1 > \theta_c$.

However, if $\theta_1 < \theta_c$, the outgoing intensity will include transmission from medium 2 to medium 1, as shown in the image below.

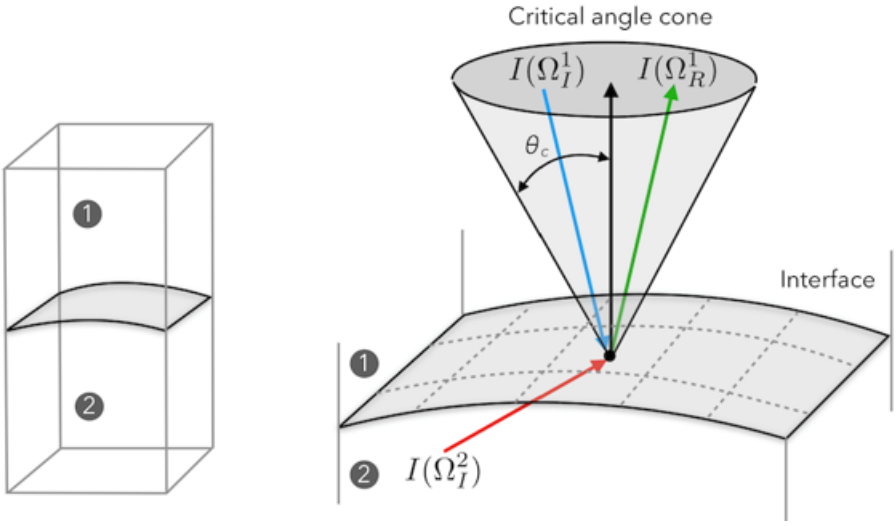


Figure 36:Reflection and transmission of intensity rays at an interface ($n_1 > n_2$) for $\theta_1 < \theta_c$

At the interface, the intensity is partially reflected and transmitted into the other medium if $\theta_1 < \theta_c$ or the outgoing direction is in medium 2. The reflected proportion from $\Omega_I^1 \rightarrow \Omega_R^1$, or reflectance, is given by

$$r_{12} = \frac{1}{2}\left(\frac{n_1 \cos\theta_1 - n_2 \cos\theta_2}{n_1 \cos\theta_1 + n_2 \cos\theta_2}\right)^2 + \frac{1}{2}\left(\frac{n_2 \cos\theta_1 - n_1 \cos\theta_2}{n_2 \cos\theta_1 + n_1 \cos\theta_2}\right)^2 \tag{148}$$

and the transmitted proportion from the $\Omega_I^2 \rightarrow \Omega_R^1$, or transmittance, is given by

$$\tau_{21} = 1 - r_{12} \tag{149}$$

In the second medium, for the current scenario where $n_2 > n_1$, if $\theta_2 < \theta_c$ the radiative intensity is, as for medium one, partially reflected and partially transmitted. The reflection coefficient is as described above since $r_{21} = r_{12}$. If $\theta_2 > \theta_c$, then total internal reflection occurs and $r_{21} = 1.0$ and $\tau_{12} = 0.0$, meaning no transmission of radiative intensity into the second medium or from the first medium. This is shown in the image above with the gray dashed lines.

From the above, the outgoing radiative intensity on the side one of the interface for the current ordinate direction, Ω_R^1 is given by

$$I(\Omega_R^1) = r_{12} I(\Omega_I^1) + \tau_{21} \left(\frac{n_1}{n_2} \right)^2 I(\Omega_I^2) \tag{150}$$

where for medium one, the first term on the right-hand side represents the reflected intensity in medium one and the second term represents the transmitted intensity from medium two to one. For medium two, if the current ordinate direction is Ω_R^2 then the intensity outgoing radiative intensity is given by

$$I(\Omega_R^2) = r_{21} I(\Omega_I^1) + \tau_{12} \left(\frac{n_2}{n_1} \right)^2 I(\Omega_I^1) \tag{151}$$

For $n_2 < n_1$, the subscripts of the above analysis must be exchanged, and total internal reflection will now occur in medium one.

Reflection and transmission for diffuse interfaces

If the interface is diffused, for example, diffused_fraction = 1.0, the reflectivity of the interface is given by the hemispherically averaged reflectance:

$$r_{D,12} = \frac{1}{2} + \frac{(n-1)(3n+1)}{6(n+1)^2} - \frac{2n^3(n^2+2n+1)}{(n^4-1)^2(n^2+1)} + \frac{8n^4(n^4+1)}{(n^4-1)^2(n^2+1)} \ln n + \frac{n^2(n^2-1)^2}{(n^2+1)^3} \ln\left(\frac{n-1}{n+1} \right) \tag{152}$$

where $n = n_1 / n_2$ is the ratio of refractive indices.

> **Note:** n_1 always represents the medium with higher refractive index and n_2 the medium of lower refractive index.

The transmission from medium one to two is given by

$$\tau_{D,12} = 1 - r_{D,12} \tag{153}$$

For the reverse direction the reflectance and transmittance are given by:

$$\tau_{D,21} = 1 - \frac{1}{n^2}(1 - r_{D,12}) \tag{154}$$

and

$$\tau_{D,21} = \frac{1}{n^2} \tau_{D,12} \tag{155}$$

respectively.

The incoming radiative intensity to the interface is given by the hemispherically averaged intensity for medium one and two:

$$Q_1 = \sum_{j=1}^{N} w_j I_j |n \cdot \Omega_j| (n \cdot \Omega_j) \quad \text{when } n \cdot \Omega_j > 0 \tag{156}$$

$$Q_2 = \sum_{j=1}^{N} w_j I_j |n \cdot \Omega_j| (n \cdot \Omega_j) \quad \text{when } n \cdot \Omega_j \leq 0 \tag{157}$$

where n is the outward facing normal. From these fluxes, the outgoing radiative intensity at the wall for the current ordinate direction, Ω, is given by

$$I(\Omega) = r_{D,12}\frac{Q_1}{\pi} + \tau_{D,21}\frac{Q_2}{\pi} \tag{158}$$

Reflection and Transmission for partially specular and partially diffuse interfaces

For partially specular and partially diffuse interfaces 0.0 < diffused fraction < 1.0.

Interfaces between semi-transparent media are typically not 100 percent diffused or specular and the diffuse fraction lies somewhere between zero and one. In this range the outgoing radiative intensity is treated as a linear combination of the specular and diffuse components, for example:

$$I(\Omega) = (1-a) I^S(\Omega) + a I^D(\Omega) \tag{159}$$

where a is the diffuse fraction, $I^S(\Omega)$ is the outgoing specular component of radiative intensity, and $I^D(\Omega)$ is the outgoing diffuse component of radiative intensity. For example, in medium one in the image above the components would be:

$$I^S(\Omega) = I(\Omega_R^1) = r_{12} I(\Omega_I^1) + \tau_{21}\left(\frac{n_1}{n_2}\right)^2 I(\Omega_I^2) \tag{160}$$

and

$$I^D(\Omega) = r_{D,12}\frac{Q_1}{\pi} + \tau_{D,21}\frac{Q_2}{\pi} \tag{161}$$

Specular and diffuse interfaces

For the interface equations to be applied weakly, I_w must be applied in both mediums. If the current ordinate direction, Ω, is outgoing from the interface in medium 1 then I_w is equal to the proportion of radiative intensity reflected and transmitted. From the analysis in the previous section, this would be:

$$I_w = r_{12} I\left(\Omega_I^1\right) + \tau_{21}\left(\frac{n_1}{n_2}\right)^2 I\left(\Omega_I^2\right) \tag{162}$$

In medium 2 since the current ordinate direction, Ω, is incoming to the surface no boundary flux is added to the equation, that is, $\eta(I_w, v)_\Gamma = 0$.

Reflection and Transmission for diffuse interfaces of Type External

Exchange of radiative intensity occurs for external surfaces when the medium inside the computational domain is semitransparent. That is the medium surrounding, which is not modeled using a computational mesh, participates in radiative transfer. For this case a mathematical model of external radiation is used. The model assumes that the surrounding medium has uniform radiative intensity

in all directions, that is, the radiative flux is isotropic. The isotropic radiative intensity is given by the following blackbody source:

$$I_{\text{ext}} = \frac{\varepsilon_{\text{ext}} \sigma T_{\text{ext}}^4}{\pi} \tag{163}$$

where ε_{ext} is the external emissivity and is set to one, σ is the Stefan-Boltzmann constant, T_{ext} is the temperature of the surrounding medium. At the external interface I_{ext} is transferred into the medium. This condition can only be applied to boundaries as the interface is only modeled mathematically.

Variable Property Support

AcuSolve provides an extensive set of features to model the different material properties of the fluid and the surrounding media. The different material properties that can be modeled are discussed in the following sections.

Density

The density models can be specified for all types of media supported by AcuSolve (solid, fluid and shell). They are:

- Constant Density: A constant density for the media. This is generally used for incompressible flow simulations.
- Boussinesq Model: Use of the Boussinesq approximation, where the variation in density is assumed to be a linear function of temperature and the density variation is accounted only in the body force terms.
- Isentropic Model: The constant entropy gas model, where the variation in density is taken into account for all the terms in the momentum equations.
- Ideal Gas Model: The density is based on the ideal gas law. The variation in density is taken into account for all the terms in the momentum equations.
- Customized Models: Custom variable property functions for density can be specified using curve fit options (piecewise linear and cubic spline) as a function of single independent variable. User functions can be used to model more complex density models. In case of customized models the density variation is applicable to all the terms.

Viscosity

The molecular viscosity models used in the flow (momentum) equations, specified for the fluid medium are:

- Constant Viscosity: A constant viscosity for the fluid medium. This is the simplest case of a Newtonian model for calculation of the stress tensor.
- Ramped Viscosity: The viscosity is ramped down 1,000 times from the viscosity value specified at time step one to the viscosity value at time step ten, after which the value from time step ten is used.
- Non-Newtonian Models: Non-Newtonian viscosity models based on Power Law, Bingham model, Carreau-Yasuda model can be specified.

- Customized Models: Custom variable property functions for viscosity can be specified using curve fit options (piecewise linear and cubic spline) as a function of single independent variable. User functions can be used to model more complex viscosity models.

Porosity

AcuSolve uses the Darcy-Forchheimer porosity model for the flow (momentum) equations, specified for the fluid medium. The porosity model modifies the momentum equation as follows:

$$\frac{\rho}{\phi}\left(\frac{\partial \vec{u}}{\partial t}+(\rho\vec{u}\cdot\nabla)\vec{u}+R\vec{f}\right)=-\nabla p+\rho b+\nabla\cdot\tau \tag{164}$$

where

- $\vec{f}$ is the field vector contribution due to porous media, also referred to as porous media forces.
- R is the rotation tensor which rotates $\vec{f}$ to the global coordinate system.

The porous media forces are given by:

$$f_i=\left(\frac{C_{Darcy}\mu}{k_i}+\frac{C_{Forch}\rho}{\sqrt{k_i}}|u|\right)u_i \tag{165}$$

where

- C_{Darcy} and C_{Forch} are the Darcy and Forchheimer coefficients, respectively.
- k_i is the permeability in the principal direction i.

Viscoelasticity

AcuSolve supports simulations involving viscoelastic materials using the Upper Convected Maxwell Model and the Upper Convected Maxwell Log Model. The viscoelastic models used in the flow (momentum) equations, have various sub models to determine the rate of stress build up and rate of stress decay. These include:

- Oldroy-B Model: Stress build up and decay rate obtained by defining the relaxation time and polymer viscosity of the fluid.
- Giesekus Model: Stress build up and decay rate obtained by defining the mobility factor along with the relaxation time and polymer viscosity of the fluid.
- Phan-Thien-Tanner Model (PTT): Stress build up and decay rate obtained by defining the PTT extensibility factor and the diffential ratio along with the relaxation time and polymer viscosity of the fluid.
- Customized Models: Customized models based on a user function.

Additional Properties

AcuSolve also supports the modeling of several additional properties such as Surface Tension, Conductivity, Diffusivity, Contact Angle and Specific Heat of the fluid medium.

All of these material properties except Contact Angle can be specified using customized models based on curve fit options (piecewise linear and cubic spline) as a function of single independent variable or by using user functions.

The Conductivity can be additionally specified to be based on an Isotropic Prandtl number Model, ramped to a value or set as anisotropic using the customized models.

The Diffusivity can be additionally ramped to a value.

Contact Angle for free surfaces with surface tension can be set to a constant value or automatically computed from the boundary conditions.

Eulerian Multiphase Models

Eulerian multiphase models are used to model interpenetrating fields. The concept of volume fraction is used to model different phases. Nonetheless, the velocities of the phases can differ. Generally, in this type of multiphase flows, there are dispersed phases such as bubbles, droplets, or particles within a carrier field. There are three main approaches for modeling:

Euler-Euler	In this approach, separate but coupled momentum equations, along with individual volume fractions for each phase, are solved to compute the velocities and volume fractions of the phases. This is the most accurate but also the most computationally expensive approach.
Algebraic-Euler (Mixture)	In mixture theory, a single mixture momentum equation, together with volume fraction equations for the phase, except one, is solved. The relative velocity of phases is calculated using the Algebraic Slip Model (ASM).
Homogeneous	The homogeneous model is the simplest model and in practice is similar with mixture model, but it assumes that all phases move with the same velocity. Technically, ASM is ignored.

Phasic Equations

As each phase in Eulerian multiphase models moves with different velocities, individual momentum equations govern their motion. The phasic continuity (volume fraction) and momentum equations for each phase k can be written as:

Phasic continuity equation

$$\frac{\partial}{\partial t}\left(a_k \rho_k\right) + \nabla \cdot \left(a_k \rho_k u_k\right) = 0 \tag{166}$$

where a_k is the volume fraction of phase k, ρ_k is the density of phase k, and u_k is the velocity of phase k.

Momentum equation for each phase

$$\frac{\partial}{\partial t}\left(a_k \rho_k u_k\right) + \nabla \cdot \left(a_k \rho_k u_k u_k\right) = -a_k \nabla p + \nabla \cdot a_k(\tau_k) + a_k \rho_k g + M_k \tag{167}$$

where p is the pressure, τ_k is the stress tensor of phase k, g is the gravitational acceleration, and M_k represents the momentum exchange between phases.

The last term in the momentum equation represents the momentum exchange between phases. This momentum exchange occurs due to interfacial forces between phases. These forces include drag force, lift force, turbulent dispersion force, wall lubrication force, virtual mass force, surface tension and solid collision force.

In Eulerian multiphase models, individual continuity and momentum equations are solved for the phasic mass (volume) and velocity, while the pressure field is assumed to be shared among all phases. To calculate the shared pressure field, the mean continuity and momentum equations are derived from the phasic equations and solved for the mean velocity of the phases and the shared pressure. The mean continuity and momentum equations are obtained by summing the individual phasic equations as follows:

$$\frac{\partial}{\partial t}\sum_{k=1}^{n}(a_k\rho_k)+\nabla\cdot\sum_{k=1}^{n}(a_k\rho_k u_k)=0 \tag{168}$$

$$\frac{\partial}{\partial t}\sum_{k=1}^{n}(a_k\rho_k u_k)+\nabla\cdot\sum_{k=1}^{n}(a_k\rho_k u_k u_k)=-\sum_{k=1}^{n}(a_k\nabla p)+\nabla\cdot\sum_{k=1}^{n}(a_k)\tau_k+\sum_{k=1}^{n}(a_k\rho_k g) \tag{169}$$

The mixture density and mixture velocity of the multiphase system are defined as:

$$\rho_m=\sum_{k=1}^{n}a_k\rho_k \tag{170}$$

$$u_m=\frac{1}{\rho_m}\sum_{k=1}^{n}a_k\rho_k u_k \tag{171}$$

Note: The phasic velocity differs from the mixture velocity, and the relative velocity between them is referred to as the drift velocity, which is defined as the velocity difference between the velocity of the disperse phase, for example, bubbles, droplets, or particles, and the velocity of the mixture.

The drift velocity accounts for the relative motion between phases, which arises due to differences in density, buoyancy, or interaction forces such as drag.

$$u_k-u_m=u_k^{drift} \tag{172}$$

Using the mixture density and velocity, the mixture continuity and momentum equations can be derived through straightforward mathematical manipulations as:

$$\frac{\partial\rho_m}{\partial t}+\nabla\cdot(\rho_m u_m)=0 \tag{173}$$

$$\frac{\partial}{\partial t}(\rho_m u_m)+\nabla\cdot(\rho_m u_m u_m)=-\nabla p-\nabla\cdot\sum_{k=1}^{n}(a_k\rho_k u_k^{drift}u_k^{drift})+\nabla\cdot\tau_k+\rho_m g \tag{174}$$

These two equations are solved in AcuSolve using flow stagger to obtain the mixture velocity and pressure fields.

Eulerian-Eulerian

If the preferred multiphase approach is Eulerian-Eulerian, the mixture continuity and momentum equations are solved in the flow stagger for the mean velocity and pressure. For each dispersed field

(k=1,…n-1), individual phasic continuity and momentum equations are solved to obtain the phasic values of volume fraction and velocity.

Algebraic-Eulerian

Similar to Eulerian-Eulerian, the mixture continuity and momentum equations are solved in the flow stagger for the mean velocity and pressure. For each dispersed field (k=1,…n-1), one phasic continuity equation is solved for the phasic values of volume fraction. However, instead of solving for phasic momentum equation, an algebraic equation is solved for the phasic velocity. Algebraic equation is derived following algebra slip model.

Algebraic slip model

The algebraic slip model is used to avoid solving the full phasic momentum equations. To achieve this, the phasic and mean momentum equations are written in non-conservative form (using the mean phasic continuity equations):

$$a_k\rho_k\frac{\partial}{\partial t}(u_k)+a_k\rho_k u_k\nabla(u_k)=-a_k\nabla p+\nabla\cdot a_k(\tau_k)+a_k\rho_k g+M_k \tag{175}$$

$$\frac{\partial}{\partial t}(\rho_m u_m)+\rho_m u_m\nabla(u_m)=-\nabla p+\nabla\cdot\tau_m+\rho_m g \tag{176}$$

These equations can be subtracted to eliminate pressure gradient terms. In addition, the following assumptions are made:

- It is assumed that the disperse phase reaches terminal velocity instantaneously, allowing for the transient terms in the phasic equation to be ignored.
- Viscous and drift stresses are ignored.
- Phasic advection is assumed to be equal to the mean acceleration, as follows:

$$u_k\nabla\cdot(u_k)=u_m\nabla\cdot\quad(u_m) \tag{177}$$

Applying these assumptions to the combined equations results in the following:

$$M_k=a_k(\rho_k-\rho_m)\left(\frac{\partial u_m}{\partial t}+\quad u_m\nabla\cdot\quad(u_m)-g\right) \tag{178}$$

As a final assumption, the momentum exchange between phases is considered to be primarily due to drag force, and optionally lift, as follows:

$$M_k=\frac{1}{2}\rho_c\quad C_D A_k|(u_k\quad-u_c)|(u_k\quad-u_c) \tag{179}$$

where $u_k^{slip}=u_k\quad-u_c$. Therefore, the final form of the algebraic slip model becomes:

$$\left|u_k^{slip}\right|u_k^{slip}=\frac{4}{3}\frac{(\rho_k-\rho_m)}{\rho_k}\quad\left(\frac{\partial u_m}{\partial t}+\quad u_m\nabla\cdot(u_m)-g\right) \tag{180}$$

The slip velocity and mean velocity are used to obtain the phasic velocity as follows:

$$u_k^{drift}=u_k^{slip}-\frac{1}{\rho_m}\sum_{k=1}^{n}a_k\rho_k u_k^{slip} \tag{181}$$

$$u_k=u_m+u_k^{drift} \tag{182}$$

Scalar Transport of Multiple Species

AcuSolve has the capability to track multiple species in a fluid flow by using the scalar transport equations for each of them individually. The species transport equation is similar to the generalized governing equations for the fluid flow and it is expressed as:

$$\rho \frac{\partial \varphi_i}{\partial t} + (\rho \vec{u} \cdot \nabla)\varphi_i = \nabla \cdot \Psi_i + \rho\sigma_i \quad (183)$$

where

- φ_i represents the i^{th} scalar species.
- Ψ_i is the diffusion flux vector for species φ_i.
- σ_i is the source per unit mass for species φ_i.

The diffusivity flux vector is expressed as:

$$\Psi = d\nabla\varphi \quad (184)$$

where d is the diffusivity for species φ. The diffusivity can be modeled as constant, ramped against time step or customized using variable property and user functions.

The material properties can be set to be functions of species concentration. This models a 'miscible' property relative to mixing of multiple fluids.

Turbulence Modeling

AcuSolve supports a variety of turbulence models for fluid flow simulations. For steady state simulations the Reynolds-averaged Navier-Stokes equations are solved to arrive directly at the time averaged flow field. In case of transient simulations the governing equations are integrated in time to yield an accurate description of the flow field. There are also many different turbulence closure methods available for each type of simulation.

The various turbulence models used in AcuSolve are:

One Equation and Two Equation Models	Spalart-Allmaras (SA) turbulence model
	Shear Stress Transport (SST) turbulence model
	$k-\omega$ turbulence model
	$k-\varepsilon$ model
	RNG $k-\varepsilon$ model
	Realizable $k-\varepsilon$ model
Detached Eddy Simulation (DES) Models	SA – DES
	SST – DES
	Dynamic DES (DDES)

Large Eddy Simulation (LES) Models	Classical (Smaroginsky) model
	Dynamic subgrid LES model

Reynolds-Averaged Navier-Stokes (RANS) Simulations

For incompressible turbulent flow the instantaneous velocity field can be decomposed into a time averaged velocity and its corresponding fluctuation.

This is commonly called the Reynolds' decomposition, which is shown in Figure 37.

$$u_i = \overline{u}_i + u_i' \tag{185}$$

where

- u_i: instantaneous velocity,
- $\overline{u}_i = \frac{1}{\Delta t}\int_t^{t+\Delta t} u_i dt$: time averaged velocity over Δt,
- u_i' velocity fluctuation.

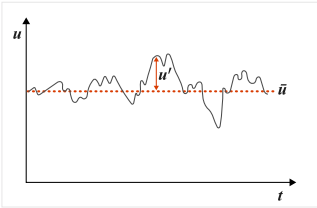


Figure 37: Turbulent Velocity Signals as a Function of Time

Similarly, instantaneous pressure can be decomposed into the time averaged and fluctuation terms.

$$p = \overline{p} + p' \tag{186}$$

where p is the instantaneous pressure.

Once this concept is substituted into the instantaneous Navier-Stokes equations and then time averaging is performed the RANS equations are obtained. The following equations are, respectively, the Reynolds-averaged continuity and Reynolds-averaged momentum equations:

$$\frac{\partial \rho}{\partial t} + \frac{\partial \left(\rho \overline{u}_j\right)}{\partial x_j} = 0 \tag{187}$$

$$\frac{\partial \left(\rho \overline{u}_i\right)}{\partial t} + \frac{\partial \left(\rho \overline{u}_i \overline{u}_j\right)}{\partial x_j} = -\frac{\partial \left(\overline{p}\delta_{ij}\right)}{\partial x_j} + 2\mu \frac{\partial \overline{S_{ij}}}{\partial x_j} + \frac{\partial \tau_{ij}^R}{\partial x_j} \tag{188}$$

where

- $\overline{S_{ij}} = \frac{1}{2}\left(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i}\right)$ is the mean strain rate tensor,
- $\tau_{ij}^R = -\rho\overline{u_i' u_j'}$ is the Reynolds stress tensor.

It is noted that the momentum equation has an unsteady term (the first term in the left hand side of the momentum equation). Wegner et al. (2004) argued that the RANS equations with the time term (unsteady RANS) can be applied to simulate unsteady flows when a time scale (T2) for large scale motions is larger than a time averaging period for RANS (T1) (see Figure 38). It is worth mentioning that RANS time averaging period (T1) should be considerably larger than the largest turbulence (integral) time scale, as URANS can only give a time averaged mean value for the velocity field, not solving turbulence. Thus, a representative URANS solution is demonstrated by the smoothed red line in the image below. Whereas the blue line represents the velocity profile with instantaneous fluctuations.

Steady RANS can be obtained by removing the time term when there is no large scale unsteadiness observed in turbulent flow.

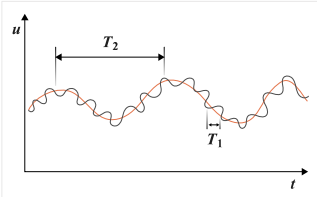


Figure 38: Time Scales for T2 for Large Unsteady Oscillation vs T1 for Turbulent Time Scale

Although Reynolds-averaging eliminates a need to compute the instantaneous flow field, it introduces a new unknown term in comparison to the Navier-Stokes equations. This unknown term is referred to as the Reynolds stress tensor, which represents an added stress due to the turbulent motions. This term arises from the decomposition of the convective term using the Reynolds decomposition. Since your goal is to eliminate any dependence on the instantaneous flow field in these simplified equations, this term should be modeled in terms of the mean flow. This challenge is known as the closure problem of turbulence.

One method that is available for computing the Reynolds Stresses that appear in the RANS equations is to use the Boussinesq approximation. This approach assumes a linear relationship between the turbulent Reynolds stresses and the mean rate of strain tensor:

$$\tau_{ij}^R = -\rho\overline{u_i' u_j'} = \mu_t\left[\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i} - \frac{2}{3}\frac{\partial \overline{u_k}}{\partial x_k}\delta_{ij}\right] - \frac{2}{3}\rho k\delta_{ij} \qquad (189)$$

where

- μ_t: the eddy viscosity,
- $k = \frac{1}{2}\overline{u_i' u_j'}$: the turbulent kinetic energy per unit mass,
- δ_{ij}: the Kronecker delta.

Since $\frac{\partial \overline{u_k}}{\partial x_k} = 0$ for incompressible flow the isotropic eddy viscosity becomes

$$\tau_{ij}^R = \mu_t \left[\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i} \right] - \frac{2}{3} \rho k \delta_{ij} \tag{190}$$

The only unknown in this equation is the eddy viscosity, which must be determined by the turbulence model. There are hundreds of available turbulence models that have been developed to provide this eddy viscosity. Some common Boussinesq models include

- Mixing length model
- Spalart-Allmaras model
- $k-\varepsilon$ model, RNG (Renormalized Group) $k-\varepsilon$model, Realizable $k-\varepsilon$ model
- $k-\omega$ model, SST (Shear Stress Transport) model
- $v2f$ model, zeta-f model

This approach is valid for many turbulent flows, for example, boundary layer, channel flow, round jets, turbulent shear flow and mixing layer, but not for all, for example, impinging flow and swirling flow. Under these circumstances, a model that is capable of predicting anisotropic Reynolds stresses is more appropriate. This can include a full Reynolds stress model (RSM), which determines the turbulent stresses by solving a transport equation for each stress component, or an eddy viscosity model that uses a non linear stress strain relationship to compute each Reynolds stress component separately. Although the improvement in accuracy achieved from this type of modeling is appealing, the introduction of anisotropic stresses is often accompanied by increased compute cost and a decrease in robustness.

The following section provides a review of some commonly used RANS models, ranging from algebraic models (zero equation models) up to the seven equation Reynolds Stress Model.

General Form of Turbulence Models

The general formation of turbulence models can be written as

$$\underbrace{\frac{\partial(\rho\phi)}{\partial t}}_{\text{Unsteady term}} + \underbrace{\frac{\partial\left(\rho \overline{u}_j \phi\right)}{\partial x_j}}_{\text{Convection term}} = \underbrace{\frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma}\right)\frac{\partial \phi}{\partial x_j}\right]}_{\text{Diffusion term}} + \underbrace{P_\phi}_{\text{Production term}} + \underbrace{D_\phi}_{\text{Dissipation term}} \tag{191}$$

where ϕ is a turbulence model variable, μ_t is the eddy viscosity, μ is the material viscosity and σ is a constant.

The left-hand side of the equation describes advection, which includes two terms, the acceleration term and convective term. The right-hand side of the equation represents the summation of the diffusion, production and dissipation terms. The unsteady term represents the time dependence of turbulence model variables, while the convective term accounts for the rate of change of variables due to convection by the mean flow. The diffusion term on the right side of the equation describes the transport of turbulent variables due to the summation of material viscosity and eddy viscosity. The production term indicates the production rate of turbulent variables from the mean flow gradient, while the dissipation term represents the dissipation rate of turbulent variables due to viscous stresses.

One Equation Eddy Viscosity Models

One equation Reynolds-averaged Navier-Stokes (RANS) models solve a single scalar transport equation to compute the eddy viscosity.

Common one equation models include the Spalart-Allmaras (SA) model and the Nut-92 model. The SA model is discussed in this manual due to its application in general purpose CFD codes and its popularity for the simulation of external flows and internal flows. The details of the Nut-92 model can be found in Shur et al. (1995).

Spalart-Allmaras (SA) Model

The SA model uses a transport equation to solve for a modified kinematic eddy viscosity, $\hat{\upsilon}$, as a function of the kinematic eddy viscosity (ν_t) (Spalart and Allmaras, 1992).

In this model, a length scale (d) in a dissipation term of the modified kinematic eddy viscosity transport equation is specified to determine the dissipation rate. This model has an advantage of having economic solutions for attached flows and moderately separated flows, but it is not recommended for massively separated flows, free shear flows and decaying turbulence.

Transport Equations

$$\frac{\partial(\rho\hat{\upsilon})}{\partial t}+\frac{\partial\left(\rho\hat{\upsilon}\overline{u_j}\right)}{\partial x_j} = \frac{1}{\sigma}\left(\frac{\partial}{\partial x_j}\left[(\mu+\rho\hat{\upsilon})\frac{\partial\hat{\upsilon}}{\partial x_j}\right]+\rho C_{b2}\frac{\partial\hat{\upsilon}}{\partial x_j}\frac{\partial\hat{\upsilon}}{\partial x_j}\right)+P+D \tag{192}$$

where σ and C_{b2} are constants and μ is the fluid dynamic viscosity. P and D are the production term and destruction term of the modified turbulent viscosity, respectively.

Production of $\hat{\upsilon}$

$$P=\rho C_{b1}\hat{S}\hat{\upsilon} \tag{193}$$

where

- $\hat{S}=\sqrt{2\Omega_{ij}\Omega_{ij}}+\frac{\hat{\upsilon}}{\kappa^2 d^2}f_{v2}$,
- $\Omega_{ij}=\frac{1}{2}\left(\frac{\partial\overline{u_i}}{\partial x_j}-\frac{\partial\overline{u_j}}{\partial x_i}\right)$ is the rotation tensor,
- $f_{v2}=1-\frac{\chi}{1+\chi f_{v1}}$,
- d is the distance from the nearest wall,
- κ is the Von Kármán constant,
- C_{b1} is a constant.

Destruction of $\hat{\upsilon}$

$$D=-\rho C_{w1}f_w\left(\frac{\hat{\upsilon}}{d}\right)^2 \tag{194}$$

where

- $f_w = g\left(\frac{1+C_{w3}^6}{g^6+C_{w3}^6}\right)^{1/6}$
- $f_{t2} = C_{t3} exp\left(-C_{t4}\chi^2\right)$
- $g = r + C_{w2}\left(r^6 - r\right)$
- $r = \frac{\hat{\upsilon}}{\hat{S}\kappa^2 d^2}$
- C_{w1} is a constant.

Modeling of Turbulent Viscosity μ_t

The kinematic eddy viscosity for the Spalart-Allmaras model is computed using the following relationship

$$\mu_t = \rho\hat{\upsilon} f_{v1} \tag{195}$$

where

- $f_{v1} = \frac{\chi^3}{\chi^3 + C_{v1}^3}$ is the viscous damping function.
- $\chi = \frac{\hat{\upsilon}}{\nu}$.

Model Coefficients

$C_{w1} = \frac{C_{b1}}{\kappa^2} + \frac{1+C_{b1}}{\sigma^2}$, $C_{w2} = 0.3$, $C_{w3} = 2.0$, $C_{b1} = 0.1355$, $C_{b2} = 0.622$, $C_{v1} = 7.1$ $\sigma = \frac{2}{3}$, $\kappa = 0.41$.

Spalart-Allmaras (SA) Model with Rotation/Curvature Correction

The effects of system rotation and streamline curvature are present in turbomachinery components. Some examples include axial turbines, radial turbines, axial fans, compressors and centrifugal impellors.

This is where most linear eddy viscosity models fail. In order to incorporate the rotational and curvature effects for the SA model, Shur et al. (2000) introduced a version with the modified production term of the transport equation by multiplying the rotation function f_{r1}.

Modified Production of $\hat{\upsilon}$

$$P = \rho f_{r1} C_{b1} \hat{S}\hat{\upsilon} \tag{196}$$

where

- $f_{r1} = \left(1+C_{r1}\right)\frac{2r^*}{1+r^*}\left(1 - C_{r3}\ \ tan^{-1}\left[C_{r2}\hat{r}\right]\right) - C_{r1}$
- $r^* = \frac{S}{\hat{\omega}}$
- $S = \sqrt{2S_{ij}S_{ij}}$ is the strain rate magnitude.
- $S_{ij} = \frac{1}{2}\left(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i}\right)$ is the strain rate tensor.

- $\hat{\omega} = \sqrt{2\hat{\omega}_{ij}\hat{\omega}_{ij}}$ is the modified vorticity magnitude.
- $\hat{\omega}_{ij} = \frac{1}{2}\left(\left[\frac{\partial \overline{u_i}}{\partial x_j} - \frac{\partial \overline{u_j}}{\partial x_i}\right] + 2\varepsilon_{mjl}\Omega'_m\right)$ is the rotation tensor.
- $\hat{r} = \frac{2\omega_{ik}S_{jk}}{D^4}\left(\frac{\partial S_{ij}}{\partial t} + \overline{u_{ij}}\frac{\partial S_{ij}}{\partial x_j} + \left[\varepsilon_{imn}S_{jn} + \varepsilon_{jmn}S_{in}\right]\Omega'_m\right)$
- $D = \sqrt{\frac{1}{2}\left(S^2 + \hat{\omega}^2\right)}$
- Ω' is the rotation rate.

Model Coefficients

C_{r1} =1.0, C_{r2} =12.0, C_{r3} =1.0

Two Equation Eddy Viscosity Models

Two equation turbulence models are perhaps the most commonly used in industrial flow applications.

Although there are many two equation models available, some of the more popular ones include:

- $k-\varepsilon$ model, Renormalized Group (RNG) $k-\varepsilon$ model, Realizable $k-\varepsilon$ model,
- $k-\omega$ model, Shear Stress Transport (SST) model.

Standard k-ε Model

Launder and Spalding (1974) proposed the standard k-ε turbulence model utilizing the relationships described below.

The model has been shown to be relatively accurate for high Reynolds number flows in which the turbulence behavior is close to homogeneous, and the turbulence production is nearly balanced by dissipation. Nevertheless, its performance deteriorates when predicting boundary layers under adverse pressure gradients (Bradshaw, 1997). It also has difficulty in predicting the viscous sublayer. To resolve this issue, it is suggested that a correction be made to reproduce the law of the wall for incompressible flat-plate boundary layers (Wilcox, 2000). For low Reynolds number flows, the difference between the turbulent kinetic energy production and the dissipation rates may depart from their equilibrium value of zero, thus, ad-hoc adjustments of empirical parameters are inevitable. Furthermore, the standard k-ε turbulence model does not perform well in shear layers and jets, where the turbulent kinetic energy is not balanced with the dissipation rates (Versteeg and Malalasekera, 2007). Finally, this model is not recommended for high swirling/curvature flows, diverging passage flows as well as flows with a body force under the influence of a rotating reference frame.

Transport Equations

Turbulent Kinetic Energy k

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial\left(\rho \overline{u_j} k\right)}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + P_k + D_k \qquad (197)$$

Turbulent Dissipation Rate ε

$$\frac{\partial(\rho\varepsilon)}{\partial t}+\frac{\partial\left(\rho\overline{u_j}\varepsilon\right)}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial\varepsilon}{\partial x_j}\right]+P_\varepsilon+D_\varepsilon \quad (198)$$

Production Modeling

Turbulent Kinetic Energy k

$$P_k=\mu_t S^2 \quad (199)$$

where

- $S=\sqrt{2S_{ij}S_{ij}}$ is the strain rate magnitude.
- μ_t is dynamic eddy viscosity.

Turbulent Dissipation Rate ε

$$P_\varepsilon=C_{\varepsilon1}\frac{\varepsilon}{k}\mu_t S^2=C_{\varepsilon1}\frac{\varepsilon}{k}P_k \quad (200)$$

Dissipation Modeling

Turbulent Kinetic Energy k

$$D_k=-\rho\varepsilon \quad (201)$$

Turbulent Dissipation Rate ε

$$D_\varepsilon=-C_{\varepsilon2}\rho\frac{\varepsilon^2}{k} \quad (202)$$

Modeling of Turbulent Viscosity μ_t

$$\mu_t=C_\mu\frac{k^2}{\varepsilon} \quad (203)$$

Model Coefficients

$C_{\varepsilon1}$ =1.44, $C_{\varepsilon2}$ =1.92, C_μ =0.09, σ_k =1.0, σ_ε =1.3.

Renormalization Group (RNG) k-ε Model

The RNG k-ε turbulence model (Yakhot and Orszag, 1986) deduces the behavior of large scale eddies from that of the smaller ones by utilizing the scale similarity properties that are inherent in the energy cascade (Bradshaw, 1997).

This model employs a modified coefficient in the dissipation rate equation to account for the interaction between the turbulent dissipation and mean shear. It results in better prediction of flows containing high streamline curvature, flows over a backward facing step (Yakhot et al, 1992), and flows in an expansion duct than the standard k-ε turbulence model. However, its performance worsens when predicting flows in a contraction duct (Hanjalic, 2004).

Transport Equations

Turbulent Kinetic Energy k

$$\frac{\partial(\rho k)}{\partial t}+\frac{\partial\left(\rho\overline{u_j}k\right)}{\partial x_j} =\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]+P_k+D_k \tag{204}$$

Turbulent Dissipation Rate ε

$$\frac{\partial(\rho\varepsilon)}{\partial t}+\frac{\partial\left(\rho\overline{u_j}\varepsilon\right)}{\partial x_j} =\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right]+P_\varepsilon+D_\varepsilon \tag{205}$$

Production Modeling

Turbulent Kinetic Energy k

$$P_k=\mu_t S^2 \tag{206}$$

Turbulent Dissipation Rate ε

$$P_\varepsilon=C_{\varepsilon 1}\frac{\varepsilon}{k}\mu_t S^2=C_{\varepsilon 1}\frac{\varepsilon}{k}P_k \tag{207}$$

Dissipation Modeling

Turbulent Kinetic Energy k

$$D_k=-\rho\varepsilon \tag{208}$$

Turbulent Dissipation Rate ε

$$D_\varepsilon=-C'_{\varepsilon 2}\rho\frac{\varepsilon^2}{k} \tag{209}$$

Modeling of Turbulent Viscosity μ_t

$$\mu_t=C_\mu\frac{k^2}{\varepsilon} \tag{210}$$

Model Coefficients

$C_{\varepsilon 1}$ =1.44, $C_{\varepsilon 2}$ =1.92, C_μ =0.09, σ_k =1.0, σ_ε =1.3, $C'_{\varepsilon 2}=\widetilde{C_{\varepsilon 2}}+\frac{C_\mu\lambda^3\left(1-\lambda/\lambda_0\right)}{1+\beta\lambda^3}$, $\widetilde{C_{\varepsilon 2}}$ = 1.68, C_μ = 0.085, $\lambda = \frac{k}{\varepsilon}S$, λ_0 = 4.38, β = 0.012, σ_k = 0.72, σ_ε = 0.72.

Realizable k-ε Model

The standard k-ε turbulence model and RNG k-ε turbulence model do not satisfy mathematical constraints on the Reynolds stresses for the consistency with physics of turbulence.

These constraints include the positivity of normal stresses ($\overline{u'_\alpha u'_\alpha}\geq 0$) and Schwarz's inequality ($\overline{u'_\alpha u'_\alpha}\quad\overline{u'_\beta u'_\beta}\geq\overline{u'_\alpha u'_\beta}\quad\overline{u'_\alpha u'_\beta}$). The realizable k-ε model proposed by Shih et al. (1995) employs a

formulation of turbulent viscosity with a variation of C_μ to satisfy these constraints. This is why this model is referred to as realizable. The second improvement made with the realizable model is to have a source term in the dissipation rate transport equation, which is derived from the transport equation of the mean square vorticity fluctuation. Compared to the standard k-ε turbulence model, it performs better when simulating planar jet flows, round jet flows and flows under adverse pressure gradients. Although the realizable model has significant performance improvement over the standard model, it has a problem in predicting impinging flow, swirling flow and secondary flows in a square duct since this model is based on the Boussinesq assumption. In addition, this model needs additional functions to simulate the near wall effects.

Transport Equations

Turbulent Kinetic Energy k

$$\frac{\partial(\rho k)}{\partial t}+\frac{\partial\left(\rho\overline{u_j}k\right)}{\partial x_j} =\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right]+P_k+D_k \tag{211}$$

Turbulent Dissipation Rate ε

$$\frac{\partial(\rho\varepsilon)}{\partial t}+\frac{\partial\left(\rho\overline{u_j}\varepsilon\right)}{\partial x_j} =\frac{\partial}{\partial x_j}\left[\left(\mu+\frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right]+P_\varepsilon+D_\varepsilon+\rho C_1 S\varepsilon \tag{212}$$

Production Modeling

Turbulent Kinetic Energy k

$$P_k=\mu_t S^2 \tag{213}$$

Turbulent Dissipation Rate ε

$$P_\varepsilon=C_{\varepsilon 1}\frac{\varepsilon}{k}P_k \tag{214}$$

Dissipation Modeling

Turbulent Kinetic Energy k

$$D_k=-\rho\varepsilon \tag{215}$$

Turbulent Dissipation Rate ε

$$D_\varepsilon=-\rho C_{\varepsilon 2}\frac{\varepsilon^2}{k+\sqrt{\nu\varepsilon}} \tag{216}$$

Modeling of Turbulent Viscosity μ_t

$$\mu_t=C_\mu\frac{k^2}{\varepsilon} \tag{217}$$

Model Coefficients

$C_{\varepsilon 1}$ = 1.44, $C_{\varepsilon 2}$ = 1.9, σ_k = 1.0, σ_ε = 1.22.

where $U^* = \sqrt{S_{ij}S_{ij}+\widetilde{\Omega_{ij}}\widetilde{\Omega_{ij}}}$, $\widetilde{\Omega_{ij}} = \Omega_{ij} - 2\varepsilon_{ijk}\omega_k$, $\Omega_{ij} = \overline{\Omega_{ij}} - 2\varepsilon_{ijk}\omega_k$, $A_0 = 4.04$, $A_s = \sqrt{6}\ \ cos\phi$,

$\phi = \frac{1}{3}cos^{-1}\left(\sqrt{6}\ \ W\right), W = \frac{S_{ij}S_{jk}S_{kj}}{\sqrt{S_{ij}S_{ij}}}$,

$C_1 = max\left[0.43, \frac{\eta}{\eta+5}\right]$

where $\eta = S\frac{k}{\varepsilon}$, $S = \sqrt{2S_{ij}S_{ij}}$ is the strain rate magnitude.

Wilcox k-ω Model

Since all three k-ε turbulence models cannot be integrated all the way to walls, wall damping wall functions must be employed to provide correct near wall behavior. It is also known that the standard k-ε turbulence model fails to predict the flow separation under adverse pressure gradients.

Wilcox proposed a turbulence model similar to the standard k-ε turbulence model but replaced the dissipation rate (ε) equation with the eddy frequency (ω) equation (Wilcox, 2006; Wilcox, 2008). The eddy frequency (ω) is often referred to the specific dissipation rate and is defined as $\omega = \varepsilon / k$. The Wilcox k-ω turbulence model has an advantage over the k-ε turbulence model as the k-ω model does not require any wall functions for the calculation of the velocity distribution near walls. As a result, the k-ω turbulence model has better performance for flows with adverse pressure gradient when compared to the k-ε turbulence models. However, the k-ω model exhibits a strong sensitivity to the freestream boundary condition (Wilcox, 2006) for external flow applications.

Transport Equations

Turbulent Kinetic Energy k

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial\left(\rho\overline{u_j}k\right)}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \sigma_k\mu_t\right)\frac{\partial k}{\partial x_j}\right] + P_k + D_k \qquad (218)$$

Eddy Frequency (Specific Dissipation Rate) ω

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial\left(\rho\overline{u_j}k\right)}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \sigma_k\mu_t\right)\frac{\partial k}{\partial x_j}\right] + P_k + D_k \qquad (219)$$

Production Modeling

Turbulent Kinetic Energy k

$$P_k = \mu_t S^2 \qquad (220)$$

Eddy Frequency ω

$$P_\omega = \frac{\gamma\omega}{k}\mu_t S^2 \qquad (221)$$

where $\gamma = \frac{\beta_0}{\beta^*} - \frac{\sigma_\omega \kappa^2}{\sqrt{\beta^*}}$, $\beta = \beta_0 f_\beta$, $f_\beta = \frac{1+85\chi_\omega}{1+100\chi_\omega}$, $\chi_\omega = \left|\frac{\Omega_{ij}\Omega_{jk}\hat{S}_{ki}}{(\beta^*\omega)^3}\right|$, $\hat{S}_{ki} = S_{ki} - \frac{1}{2}\frac{\partial \overline{u_m}}{\partial x_m}\delta_{ki}$, $S_{ij} = \frac{1}{2}\left(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i}\right)$, $\Omega_{ij} = \frac{1}{2}\left(\frac{\partial \overline{u_i}}{\partial x_j} - \frac{\partial \overline{u_j}}{\partial x_i}\right)$

Dissipation Modeling

Turbulent Kinetic Energy (k)

$$D_k = -\rho\beta^* k\omega \tag{222}$$

Eddy Frequency (ω)

$$D_\omega = -\rho\beta\omega^2 \tag{223}$$

Modeling of Turbulent Viscosity μ_t

$$\mu_t = \frac{k}{\acute{\omega}} \tag{224}$$

where $\acute{\omega} = max\left[\omega, C_{lim}\sqrt{\frac{2\bar{S}_{ij}\bar{S}_{ij}}{\beta^*}}\right]$, $\bar{S}_{ij} = S_{ij} - \frac{1}{3}\frac{\partial \overline{u_k}}{\partial x_k}\delta_{ij}$, $C_{lim} = \frac{7}{8}$,

Model Coefficients

$\sigma_k = 0.6$, $\sigma_\omega = 0.5$, $\beta^* = 0.09$, $\beta_0 = 0.0708$, $\kappa = 0.4$, $\sigma_d = \begin{cases} 0.0 & for \ \frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j} \le 0 \\ \frac{1}{8} & for \ \frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j} > 0 \end{cases}$.

Menter Shear Stress Transport (SST) k-ω Model

Menter (1994) suggested the SST model to overcome the freestream value sensitivity of the standard k-ω turbulence model by transforming the k-ε model into the k-ω model in the near-wall region, and by utilizing the k-ε model in the turbulent region far from the wall.

The SST model employs a modified source term in the eddy frequency equation. Among the different SST versions, the SST 2003 model (Menter, 2003) will be focused on in this section as it has been shown to be more accurate when predicting flows near stagnation zones by introducing a production limiter to constrain the kinetic energy production. This version also employs the strain rate magnitude in the definition of eddy viscosity rather than the vorticity magnitude employed in the standard SST model (1994). When compared to the k-ε model, the SST 2003 model achieves better accuracy for attached boundary layers and flow separation. The SST 2003 model also overcomes the freestream sensitivity of the k-ω turbulence model. However, this model shares a similar range of weakness with the k-ε equation models for the predictions of flows with strong streamline curvature and/or rotation, as well as flows under extra strains and body forces.

Transport Equations

Turbulent Kinetic Energy k

$$\frac{\partial(\rho k)}{\partial t}+\frac{\partial\left(\rho\overline{u_j}k\right)}{\partial x_j} \quad =\frac{\partial}{\partial x_j}\left[\left(\mu+\sigma_k\mu_t\right)\frac{\partial k}{\partial x_j}\right]+\tilde{P}_k+D_k \tag{225}$$

Eddy Frequency (Specific Dissipation Rate) ω

$$\frac{\partial(\rho\omega)}{\partial t}+\frac{\partial\left(\rho\overline{u_j}\omega\right)}{\partial x_j} \quad =\frac{\partial}{\partial x_j}\left[\left(\mu+\sigma_\omega\mu_t\right)\frac{\partial\omega}{\partial x_j}\right]+P_\omega+D_\omega+2\left(1-F_1\right)\frac{\rho\ \ \sigma_{\omega2}}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial\omega}{\partial x_j} \tag{226}$$

where

- The blending function $F_1=tanh\left\{\left(min\left[max\frac{\sqrt{k}}{\beta^*\omega d},\frac{500\upsilon}{d^2\omega},\frac{4\rho\sigma_{\omega2}k}{CD_{k\omega}d^2}\right]\right)^4\right\}$,
- $F_1\begin{cases}0 & :activation\ of\ k-\varepsilon\ model\ for\ turbulent\ core\ flows\\ 1 & :activaition\ of\ k-\omega\ for\ flows\ near\ walls\end{cases}$
- $CD_{k\omega}=max\left(2\rho\sigma_{\omega2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial\omega}{\partial x_i},\quad 10^{-10}\right)$,
- d is the distance to the nearest wall.

Production Modeling

Turbulent Kinetic Energy k

$$\tilde{P}_k=\min\left(P_k,\quad 10\rho\beta^*k\omega\right) \tag{227}$$

where

- Production $P_k=\mu_tS^2$
- Strain rate magnitude $S=\sqrt{2S_{ij}S_{ij}}$
- Strain rate tensor $S_{ij}=\frac{1}{2}\left(\frac{\partial\overline{u_i}}{\partial x_j}+\frac{\partial\overline{u_j}}{\partial x_i}\right)$
- Constant β^*

Eddy Frequency (ω)

$$P_\omega=\alpha\rho S^2 \tag{228}$$

where

- Blending function for specifying constants $\alpha=\alpha_1F_1+\alpha_2\left(1-F_1\right)$
- $\alpha_1=\frac{5}{9}$ for the inner layer
- α_2 = 0.44 for the outer layer.

Dissipation Modeling

Turbulent Kinetic Energy k

$$D_k=-\rho\beta^*k\omega \tag{229}$$

Eddy Frequency ω

$$D_\omega = -\rho\beta\omega^2 \tag{230}$$

Modeling of Turbulent Viscosity μ_t

$$\mu_t = \frac{\rho \ a_1 \ k}{\max\left(a_1\omega, \ S \ F_2\right)} \tag{231}$$

where

- The second blending function $F_2 = \tanh\left(\max\left(2\frac{\sqrt{k}}{\beta^*\omega \ d}, \ \frac{500\nu}{d^2\omega}\right)\right)^2$,
- Fluid kinematic viscosity ν,
- Constant β^*.

Model Coefficients

β^* = 0.09.

Following constants for SST are computed by a blending function $\phi = \phi_1 F_1 + \phi_2\left(1 - F_1\right)$: $\sigma_{\omega 1} = 0.5$, $\sigma_{\omega 2} = 0.856$, $\sigma_{k1} = 0.85$, $\sigma_{k2} = 1.00$, $\beta_1 = \frac{3}{40}$, $\beta_2 = 0.0828$.

Shear Stress Transport (SST) Model with Rotation/Curvature Correction

Since the SST model relies on the Boussineq approximation, it also has poor performance for the prediction of flows with streamline curvature and system rotation because the Reynolds stress tensor is aligned to the mean strain rate tensor.

In order to remedy this problem, the rotation curvature correction for the SST 2003 model modifies the production terms in both the kinetic energy equation and eddy frequency equation (Smirnov and Menter, 2009). The corrected model yields better performance over the SST base model when predicting flows with strong streamline curvature and rotating flows caused by strong swirl or geometric constraints.

Production Modeling

Turbulent Kinetic Energy k

$$\tilde{P}_k = \min\left(f_{r1}P_k, \ 10 f_{r1}\rho\beta^* k\omega\right) \tag{232}$$

where

- $f_{r1} = \max\left(\min\left[f_{rotation}, \ 1.25\right], \ 0.0\right)$
- $f_{rotation} = \left(1 + C_{r1}\right)\frac{2r^*}{1 + r^*}\left(1 - C_{r3} \ \tan^{-1}\left[C_{r2}\hat{r}\right]\right) - C_{r1}$
- $\hat{r} = \frac{2\Omega_{ij}S_{ij}}{\sqrt{\Omega_{ij}\Omega_{ij}}D}\left(\frac{DS_{ij}}{Dt} + \left[\varepsilon_{imn}S_{jn} + \varepsilon_{jmn}S_{in}\right]\Omega_m^{rot}\right)$

- $\Omega_{ij} = \frac{1}{2}\left[\left(\frac{\partial \overline{u_i}}{\partial x_j} - \frac{\partial \overline{u_j}}{\partial x_i}\right) + 2\varepsilon_{mji}\Omega_m^{rot}\right]$
- $S_{ij} = \frac{1}{2}\left(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i}\right)$
- $D = \sqrt{max\left(S^2,\ \ 0.09\omega^2\right)}$
- $r^* = \frac{S}{W}$
- $S = \sqrt{2S_{ij}S_{ij}}$
- $W = \sqrt{2W_{ij}W_{ij}}$

Eddy Frequency ω

$$P_\omega = f_{r1}a\rho S^2 \qquad (233)$$

Model Coefficients

β^* = 0.09, C_{r1} = 1.0, C_{r2} = 2.0, C_{r3} = 1.0.

Turbulent Transition Models

All of the previously described models are incapable of predicting boundary layer transition. To include the effects of transition additional equations are necessary.

Transitional flows can be found in many industrial application cases including gas turbine blades, airplane wings and wind turbines. It is known that conventional turbulence models over predict the wall shear stress for transitional flows. Thus, transition models can be used to improve the accuracy of CFD solutions when flows encounter turbulent transition of the boundary layer.

γ-Reθ Model

Langtry and Menter (2009) proposed one of the most commonly used transition models in industrial CFD applications. The $\gamma - Re_\theta$ model is a correlation based intermittency model that predicts natural, bypass and separation induced transition mechanisms.

The $\gamma - Re_\theta$ model is coupled with the Shear Stress Transport (SST) turbulence model and requires two additional transport equations for the turbulence intermittency (γ) and transition momentum thickness Reynolds number (Re_θ). The turbulence intermittency is a measure of the flow regime and is defined as $\gamma = \frac{t_{\text{turbulent}}}{t_{\text{laminar}} + t_{\text{turbulent}}}$ where $t_{\text{turbulent}}$ represents the time that the flow is turbulent at a given location, while t_{laminar} represents the time that the flow is laminar.

For example, while the intermittency value is zero, the flow is considered to be laminar. A value of one is for fully turbulent flow. The transition momentum thickness Reynolds number is responsible

for capturing the non local effects of turbulence intensity and is defined as $Re_\theta = \frac{\rho \bar{u} \theta}{\mu}$ where the momentum thickness is $\theta = \int_0^\delta \frac{u}{\bar{u}}\left(1 - \frac{u}{\bar{u}}\right) dy$.

To couple with the SST model, Langtry and Menter (2009) modified the production term and dissipation term of the turbulent kinetic energy to account for the changes in the flow intermittency.

Transport Equations

Turbulent Kinetic Energy k

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial\left(\rho \overline{u_j} k\right)}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \sigma_k \mu_t\right)\frac{\partial k}{\partial x_j}\right] + \tilde{P}_k + D_k \tag{234}$$

Eddy Frequency (Specific Dissipation Rate) ω

$$\frac{\partial(\rho \omega)}{\partial t} + \frac{\partial\left(\rho \overline{u_j} \omega\right)}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \sigma_\omega \mu_t\right)\frac{\partial \omega}{\partial x_j}\right] + P_\omega + D_\omega + 2\left(1 - F_1\right)\frac{\rho}{\omega}\sigma_{\omega 2}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j} \tag{235}$$

Turbulent Intermittency γ

$$\frac{\partial(\rho \gamma)}{\partial t} + \frac{\partial\left(\rho \overline{u_j} \gamma\right)}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_f}\right)\frac{\partial \gamma}{\partial x_j}\right] + P_\gamma + D_\gamma \tag{236}$$

Transition Momentum Thickness Reynolds Number Re_θ

$$\frac{\partial\left(\rho\ \mathrm{Re}_\theta\right)}{\partial t} + \frac{\partial\left(\rho \overline{u_j} \mathrm{Re}_\theta\right)}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\sigma_f\left(\mu + \mu_t\right)\frac{\partial \mathrm{Re}_\theta}{\partial x_j}\right] + P_\theta \tag{237}$$

Production Modeling

Turbulent Kinetic Energy k

$$\tilde{P}_k = \gamma_{eff}\ \min\left(P_k,\ \ 10\rho\beta^* k\omega\right)\frac{\partial \overline{u_i}}{\partial x_j} \tag{238}$$

where

- Production $P_k = \mu_t S^2$
- Strain rate magnitude $S = \sqrt{2 S_{ij} S_{ij}}$
- Strain rate tensor $S_{ij} = \frac{1}{2}\left(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i}\right)$
- Constant β^*

Turbulent Dissipation Rate ε

$$P_\varepsilon = a\rho S^2 \tag{239}$$

where

- Blending function for specifying constants $a = a_1 F_1 + a_2(1 - F_1)$
- $a_1 = \frac{5}{9}$ for the inner layer
- a_2 = 0.44 for the outer layer.

Turbulent Intermittency γ

$$P_\gamma = F_{length} C_{a1} \quad S \quad \sqrt{(\gamma F_{onset})}(1 - \gamma C C_{e1}) \tag{240}$$

where $F_{onset} = max\left\{min\left(max\left[\frac{Re_v}{2.193 \quad Re_{\theta 0}}, \left(\frac{Re_v}{2.193 \quad Re_{\theta 0}}\right)^4\right], 2\right) - max\left(1 - \left(\frac{R_T}{2.5}\right)^3, 0\right)\right\}$ $Re_v = \frac{Sd^2}{v}$, $R_T = \frac{kv}{\omega}$

Transition Momentum Thickness Reynolds Number Re_θ

$$P_\theta = C_{\theta t} \frac{1}{t}(\mathrm{Re}_\theta - \overline{\mathrm{Re}_\theta})(1 - F_\theta) \tag{241}$$

where $F_\theta = \min\left(\max\left[F_{wake} e^{-\left(\frac{d}{\delta}\right)^4}, \quad 1 - \left(\frac{\gamma - 1/C_{e2}}{1 - 1/C_{e2}}\right)^2\right], 1\right)$, $F_{wake} = e^{-\left(\frac{Re_\omega}{1E+5}\right)^2}$, $Re_\omega = \frac{\omega d^2}{v}$, $\delta = \frac{50 \Omega d}{\bar{u}} \delta_{BL}$,

$\delta_{BL} = 7.5 \theta_{BL}$, $\theta_{BL} = \frac{\overline{Re_\theta} v}{\bar{u}}$

Dissipation Modeling

Turbulent Kinetic Energy k

$$D_k = -\min\left(\max\left[\gamma_{eff}, \quad 0.1\right], 1.0\right) \rho \beta^* k \omega \tag{242}$$

Turbulent Dissipation Rate ω

$$D_\omega = -\rho \beta \omega^2 \tag{243}$$

Turbulent Intermittency γ

$$D_\gamma = -C_{a2} F_{\text{turbulent}} \Omega \gamma (C_{e2} \gamma - 1) \tag{244}$$

where $F_{\text{turbulent}} = e^{\left(-\frac{R_T}{4}\right)^4}$

Modeling of Turbulent Viscosity μ_t

$$\mu_t = \frac{\rho \quad a_1 \quad k}{\max(a_1 \omega, \quad S \quad F_2)} \tag{245}$$

where

- The second blending function $F_2 = \tanh\left(\max\left(2\frac{\sqrt{k}}{\beta^* \omega \quad d}, \quad \frac{500v}{d^2 \omega}\right)\right)^2$,
- Fluid kinematic viscosity v,

- Constant β^*.

Correlations

$$Re_\theta = \left(1173.51 - 589.428\ \ Tu + \ \frac{0.2196}{Tu^2}\right) F(\lambda_\theta) \tag{246}$$

$$Re_{\theta t} = \begin{cases} \left(1173.51 - 589.428\ \ Tu + \ \frac{0.2196}{Tu^2}\right) F(\lambda_\theta) & for \quad Tu \le 1.3 \\ 331.5(Tu - 0.5658)^{-0.671}\ \ F(\lambda_\theta), & for \quad Tu > 1.3 \end{cases} \tag{247}$$

$$F(\lambda_\theta) \begin{cases} 1 - \left(-12.986\lambda_\theta - 123.66\lambda_\theta^2 - 405.689\lambda_\theta^3\right)\ e^{-\left(\frac{Tu}{1.5}\right)^5} & for \quad \lambda_\theta \le 0 \\ 1 + 0.275\left(1 - e^{-35\lambda_\theta}\right)^{-0.671}\ \ e^{-\left(\frac{Tu}{1.5}\right)}, & for \quad \lambda_\theta > 0 \end{cases} \tag{248}$$

where $\lambda_\theta = \frac{\theta^2}{v}\frac{d\bar{u}}{ds}$ is the pressure gradient.

Model Coefficients

β^* = 0.09.

The following constants for SST are computed by a blending function $\phi = \phi_1 F_1 + \phi_2(1 - F_1)$: $\sigma_{\omega 1}$ = 0.5, $\sigma_{\omega 2}$ = 0.856, σ_{k1} = 0.85, σ_{k2} = 1.00, $\beta_1 = \frac{3}{40}$, β_2 = 0.0828. C_{e1} = 1.0, C_{e2} = 50, C_{a1} = 2.0, C_{a2} = 0.06, $C_{\theta 1}$ = 1.0, $C_{\theta 2}$ = 2.0, σ_f = 1.0.

Large Eddy Simulation (LES)

LES is often regarded as an impractical tool for industrial CFD applications as it requires large computational resources.

LES is associated with the severe requirements of small mesh size and time step size so that the turbulent length scale and time scale in the inertial range is properly resolved. These simulations also have an issue with resolving flows close to walls where energetic vortical structures become very small with the increase of Reynolds number. However, there are cases where LES provides significant benefits, including combustion chemical reaction problems and acoustic problems.

Filtered Navier-Stokes Equations

While Reynolds-averaged Navier-Stokes (RANS) resolves the mean flow and requires a turbulence model to account for the effect of turbulence on the mean flow, Large Eddy Simulation (LES) computes both the mean flow and the large energy containing eddies.

A subgrid model is used to capture the effects of small scale turbulent structures. The spatial filtering process is used to filter out the turbulent structures from the instantaneous flow field which are smaller than a given filter size. This filtering process is based on the decomposition of instantaneous variables (velocity, pressure) into filtered (resolved) and sub filter (unresolved or residual) variables. Here, velocity is used as an example.

$$u_i = \tilde{u}_i + u''_i \tag{249}$$

where

- u_i: instantaneous velocity,
- $\tilde{u}_i$: filtered velocity,
- u''_i: sub filtered (unresolved) velocity.

The filtered velocity field is obtained from a low pass filtering operation due to a weighted filter G, defined as $\tilde{u}_i = \int\int\int G(x, x', \Delta) u_i(x', t) dx'_1 dx'_2 dx'_3$.

The weighted filter includes

- $G(x, x') = \frac{1}{\pi}\left(\frac{\sin\left(\frac{\pi(x-x')}{\Delta}\right)^2}{(x-x')}\right)$: a cut-off filter,
- $G(x, x') = \sqrt{\frac{6}{\pi\Delta^2}}\, exp\left(\frac{-6(x-x')^2}{\Delta^2}\right)$: a Gaussian filter,
- $G(x, x') = \begin{cases} \frac{1}{\Delta} & \forall \quad |x-x'| \le \frac{1}{2}\Delta \\ 0 & \forall \quad |x-x'| > \frac{1}{2}\Delta \end{cases}$: a top-hat filter,

where $\Delta = \sqrt[3]{\Delta x \Delta y \Delta z}$ is a cutoff width, representing a spatial averaging over a grid element. Among those, a top-hat filter (or similar one) is a popular choice for commercial CFD codes where unstructured meshes are usually adopted.

Although the LES decomposition method resembles the Reynolds method, they have an important difference due to the following. $\widetilde{(\tilde{u}_i)} \neq \tilde{u}_i$, $\widetilde{u''_i} \neq 0$

Once this concept is substituted into the instantaneous Navier-Stokes equations, and then the spatial-averaging (or filtering) is made, the filtered Navier-Stokes equations are obtained. The filtered Navier-Stokes equations include the equations for the filtered continuity and filtered momentum equations, which are given below.

$$\frac{\partial(\rho\tilde{u}_i)}{\partial x_i} = 0 \tag{250}$$

$$\frac{\partial(\rho\tilde{u}_i)}{\partial t} + \frac{\partial(\rho\widetilde{u_i u_j})}{\partial x_j} = -\frac{\partial(\tilde{p}\delta_{ij})}{\partial x_j} + 2\mu\frac{\partial\tilde{S}_{ij}}{\partial x_j} \tag{251}$$

where

- $\widetilde{u_i u_j} = \widetilde{(\tilde{u}_i\tilde{u}_j)} + \widetilde{(u''_i\tilde{u}_j)} + \widetilde{(\tilde{u}_i u''_j)} + \widetilde{(u''_i u''_j)}$ is due to the decomposition of the nonlinear convective term in the momentum equation.
- $\tilde{S}_{ij} = \frac{1}{2}\left(\frac{\partial\tilde{u}_i}{\partial x_j} + \frac{\partial\tilde{u}_j}{\partial x_i}\right)$ is the filtered strain rate tensor.

The filtered momentum equation is rearranged as

$$\frac{\partial\left(\rho\widetilde{u}_i\right)}{\partial t}+\frac{\partial\left(\rho\widetilde{u}_i\widetilde{u}_j\right)}{\partial x_j} = -\frac{\partial\widetilde{p}}{\partial x_j}+2\mu\frac{\partial\widetilde{S}_{ij}}{\partial x_j}+\frac{\partial\tau^*_{ij}}{\partial x_j} \quad (252)$$

where

- $\tau^*_{ij}=-\rho\left(C_{ij}+R_{ij}\right)$ is a double decomposition stress tensor.
- $C_{ij}=\widetilde{\left(u"_i\widetilde{u}_j\right)}+\widetilde{\left(\widetilde{u}_iu"_j\right)}$ is the cross stress tensor, representing the interactions between large and small turbulence eddies.
- $R_{ij}=\widetilde{\left(u"_iu"_j\right)}$ is the Reynolds subgrid stress tensor.

Since $\widetilde{\left(\widetilde{u}_i\widetilde{u}_j\right)}$ in the convection term of the filtered momentum equation needs a secondary filtering process, it is rewritten as shown below (Leonard, 1974),

$$\widetilde{\left(\widetilde{u}_i\widetilde{u}_j\right)}=L_{ij}+\widetilde{u}_i\widetilde{u}_j \quad (253)$$

where $L_{ij}=\widetilde{\left(\widetilde{u}_i\widetilde{u}_j\right)}-\widetilde{u}_i\widetilde{u}_j$ is the Leonard stress tensor, representing interactions among large turbulent eddies.

Utilizing the decomposition process shown above, the filtered momentum equations can be rewritten as

$$\frac{\partial\left(\rho\widetilde{u}_i\right)}{\partial t}+\frac{\partial\left(\rho\widetilde{u}_i\widetilde{u}_j\right)}{\partial x_j} = -\frac{\partial\widetilde{p}}{\partial x_j}+2\mu\frac{\partial\widetilde{S}_{ij}}{\partial x_j}+\frac{\partial\tau'_{ij}}{\partial x_j} \quad (254)$$

where

- $\widetilde{S}_{ij}=\frac{1}{2}\left(\frac{\partial\widetilde{u}_i}{\partial x_j}+\frac{\partial\widetilde{u}_j}{\partial x_i}\right)$ is the filtered strain rate tensor,
- $\tau'_{ij}=-\rho\left(C_{ij}+R_{ij}+L_{ij}\right)=\rho\widetilde{u_iu}_j-\rho\widetilde{u}_i\widetilde{u}_j$ is the subgrid stress tensor.

Similar to RANS, the subgrid stress tensor is an unknown and must be computed by a subgrid model. However, the subgrid stress tensor differs from the Reynolds stress tensor. The table below summarizes the differences between time averaging for RANS and spatial averaging for LES. Common subgrid scale (SGS) models available to solve the filtered Navier-Stokes equations include:

- Smagorinsky-Lilly SGS model
- Germano Dynamic Smagorinsky-Lilly model
- Wall-Adapting Local Eddy-Viscosity (WALE)

Table 9: Time-Averaging (RANS) vs. Spatial-Averaging (LES)

	RANS	**LES**
Double averaging	$\overline{\left(\overline{u_i}\right)}=\overline{u_i}$	$\widetilde{\left(\widetilde{u}_i\right)}\neq\widetilde{u}_i$
Turbulence averaging	$\overline{u'_i}=0$	$\widetilde{u"_i}\neq0$

	RANS	**LES**
Averaging (or filtering) of convective term of Navier Stokes	$\overline{u_i u_j} = \left(\overline{\overline{u_i}\,\overline{u_j}}\right) + \left(\overline{u'_i \overline{u_j}}\right) + \left(\overline{\overline{u_i} u'_j}\right) + \overline{u'_i u'_j}$	$\widetilde{u_i u_j} = \left(\widetilde{\widetilde{u_i}\,\widetilde{u_j}}\right) + \left(\widetilde{u''_i \widetilde{u_j}}\right) + \left(\widetilde{\widetilde{u_i} u''_j}\right) + \left(\widetilde{u''_i u''_j}\right)$
Turbulent stress	$\tau^R_{ij} = -\rho\overline{u'_i u'_j}$	$\tau'_{ij} = \rho\widetilde{u_i u_j} - \rho\widetilde{u}_i\widetilde{u}_j$

Smagorinsky-Lilly Subgrid Scale Model

The Smagorinsky-Lilly SGS model is based on the Prandtl's mixing length model and assumes that a kinematic SGS viscosity can be expressed in terms of the length scale and the strain rate magnitude of the resolved flow.

$$\tau'_{ij} = \rho\widetilde{u_i u_j} - \rho\widetilde{u}_i\widetilde{u}_j = -2\mu_s\widetilde{S}_{ij} \qquad (255)$$

where

- $\mu_s = \rho\left(C_S\Delta\right)^2\sqrt{2\widetilde{S}_{ij}\widetilde{S}_{ij}}$ is the subgrid turbulent viscosity,
- Model constant $0.17 < C_s < 0.21$ (Lilly, 1996)
- $\widetilde{S}_{ij} = \frac{1}{2}\left(\frac{\partial\widetilde{u}_i}{\partial x_j} + \frac{\partial\widetilde{u}_j}{\partial x_i}\right)$ is the filtered (resolved) strain rate tensor

A major drawback of this model is that the model constant (C_s) does not vary in space and time. Furthermore, this model has no correct wall behavior and it is too dissipative for laminar turbulent transition cases. These limitations led to the development of a dynamic model for which the model constant is allowed to vary depending on the grid resolution and flow regime.

Dynamic Subgrid Scale Model

Recognizing C_s variations in space and time, Germano et al. (1991) proposed the dynamic model to compute the value of C_s rather than specifying it explicitly.

It is implemented by utilizing two filters: a cutoff filter Δ and a test (coarse) cutoff filter $\tilde{\Delta}$.

The subgrid stress tensor τ'_{ij} with the cutoff filter (Δ) is: $\tau'_{ij} = \rho\widetilde{u_i u_j} - \rho\widetilde{u}_i\widetilde{u}_j = -2\rho\left(C_S\Delta\right)^2|\widetilde{S}|\widetilde{S}_{ij}$. Where $|\widetilde{S}| = \sqrt{2\widetilde{S}_{ij}\widetilde{S}_{ij}}$ is the strain rate magnitude.

Figure 39 shows a resolved turbulence region utilizing Large Eddy Simulation (LES) and a modeled region assuming the subgrid tensor τ'_{ij}.

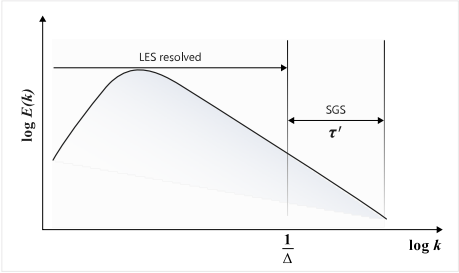


Figure 39: Energy Spectrum for LES Using the Cutoff Filter Width (Δ)

The test subgrid stress tensor T_{ij} with the coarse filter ($\tilde{\Delta}$) can be written as

$$T_{ij} = \rho\widetilde{\widetilde{u}_i\widetilde{u}_j} - \rho\widetilde{\widetilde{u}}_i\widetilde{\widetilde{u}}_j = -2\rho\left(C_S\tilde{\Delta}\right)^2\widetilde{|\tilde{S}|}\widetilde{\tilde{S}}_{ij} \tag{256}$$

where

- $|\tilde{S}| = \sqrt{2\tilde{S}_{ij}\tilde{S}_{ij}}$ is the coarse filtered strain rate magnitude.
- $\hat{S}_{ij} = \frac{1}{2}\left(\frac{\partial\hat{u}_i}{\partial x_j} + \frac{\partial\hat{u}_j}{\partial x_i}\right)$ is the filtered strain rate tensor, using the coarse cutoff filter.

Figure 40 shows a resolved LES region and a corresponding subgrid modelled region (T) when the coarse filter is employed.

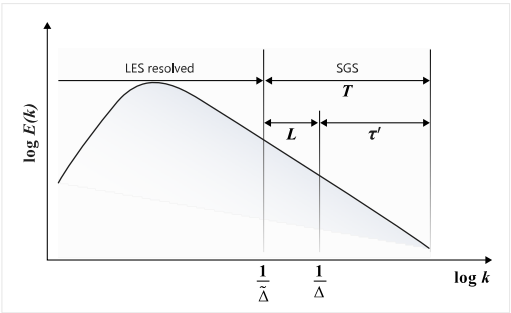


Figure 40: Energy Spectrum for LES Using the Test Cutoff Filter Width ($\tilde{\Delta}$)

Because of the coarse filtering, the test (coarse) subgrid stress tensor T_{ij} should be a summation of the coarse filtered subgrid stress tensor $\widetilde{\tau}_{ij}$ and the Leonard stress tensor L_{ij}.

$$L_{ij} = T_{ij} - \widetilde{\tau}_{ij} = \quad \rho\widetilde{\overline{u}_i}\widetilde{\overline{u}_j} - \rho\widetilde{\overline{u}_i}\widetilde{\overline{u}_j} - \rho\widetilde{\overline{u}_i}\widetilde{\overline{u}_j} + \rho\widetilde{\overline{u}_i}\widetilde{\overline{u}_j} = \rho\widetilde{\overline{u}_i}\widetilde{\overline{u}_j} - \rho\widetilde{\overline{u}_i}\widetilde{\overline{u}_j} \tag{257}$$

where

- $\widetilde{\tau}_{ij}$ is the subgrid tensor for the cutoff filter (or grid filtered), then test filtered.

$$\widetilde{\tau}_{ij} = \rho\widetilde{\overline{u}_i\overline{u}_j} - \rho\widetilde{\overline{u}_i}\widetilde{\overline{u}_j} = -2\rho\left(C_S\Delta\right)^2\widetilde{|\overline{S}|\overline{S}_{ij}} \tag{258}$$

- L_{ij} is the Leonard subgrid stress tensor, representing the contribution to the subgrid stresses by turbulence length scales smaller than the test filter but larger than the cutoff filter.

The Leonard subgrid stress tensor can be arranged as

$$L_{ij} = -2\rho\left(C_S\hat{\Delta}\right)^2\widetilde{|\overline{S}|\overline{S}_{ij}} + 2\rho\left(C_S\Delta\right)^2\widetilde{|\overline{S}|\overline{S}_{ij}} = 2\rho C_S{}^2\Delta^2\left(\widetilde{|\overline{S}|\overline{S}_{ij}} - a^2\widetilde{|\overline{S}|}\widetilde{\overline{S}_{ij}}\right) \tag{259}$$

where $a = \hat{\Delta}/\Delta$.

The Leonard subgrid stress tensor can be rewritten as

$$L_{ij} = 2\rho C_S{}^2\Delta^2\left(\widetilde{|\overline{S}|\overline{S}_{ij}} - a^2\widetilde{|\overline{S}|}\widetilde{\overline{S}_{ij}}\right) = C_S{}^2M_{ij} \tag{260}$$

where $M_{ij} = 2\rho\Delta^2\left(\widetilde{|\overline{S}|\overline{S}_{ij}} - a^2\widetilde{|\overline{S}|}\widetilde{\overline{S}_{ij}}\right)$.

Since the above equation is overdetermined a minimum least square error method is used to determine the coefficient C_S.

$$C_s{}^2 = \frac{L_{ij}M_{ij}}{M_{ij}M_{ij}} \tag{261}$$

In order to avoid numerical instabilities associated with the above equation, as the numerator could become negative, averaging of the error in the minimization is employed.

$$C_s{}^2 = \frac{L_{ij}M_{ij}}{M_{ij}M_{ij}} \tag{262}$$

Hybrid Simulations

In recent years, hybrid methods have increasingly been employed for the simulation of unsteady turbulent flows.

These models form a bridge between Large Eddy Simulation (LES) and Reynolds-averaged Navier-Stokes (RANS) by utilizing RANS for attached boundary layer calculations and LES for the separated regions. In general, hybrid simulations need spatial filtering processes to determine the local sub grid turbulent viscosity. Compared to LES and Direct Numerical Simulation (DNS), hybrid simulations are extremely robust since the numerical requirement is less severe than the two other approaches. Hybrid simulations include Detached Eddy Simulation (DES), Delayed Detached Eddy Simulation (DDES) and Improved Delayed Detached Eddy Simulation (IDDES).

Detached Eddy Simulations

Detached Eddy Simulation (DES) uses one of the one or two equation Reynolds-averaged Navier-Stokes (RANS) turbulence models to define the turbulence length scale or distance from the wall. The distance is then used to determine the region for which the RANS or Large Eddy Simulation (LES) models will be used.

DES using the Spalart-Allmaras (SA) model employs a modified distance function to replace the distance (d) in the SA model's dissipation term (Spalart et al., 1997).

$$\tilde{d} = \min\left(d, \quad C_{DES}\Delta\right) \tag{263}$$

where d is the distance to the closest wall, C_{DES} is a constant and Δ is a metric of the local element size, often chosen as the largest grid spacing in all three directions ($\Delta = \max[\Delta x, \Delta y, \Delta z]$).

The SA model is activated in a boundary layer region where d < Δ, while a Smagorinsky-like LES model is used in regions where Δ < d. Similarly, Strelets (2001) adopted the SST model as the baseline model for DES, which computes a turbulence length scale and compares it with a grid length size for a switch between LES and RANS. However, both approaches need careful determinations of local grid sizes because these grid sizes act as the switch for the activation of LES and RANS.

Delayed Detached Eddy Simulations

Delayed Detached Eddy Simulation is an improved version of DES that avoids the undesired activation of Large Eddy Simulation (LES) in boundary layer regions when the maximum grid size is less than the distance to the closest wall (Δ < d).

In this model (Spalart et al., 2006), the length scale is redefined as:

$$\tilde{d} = d - f_d \max\left(0, d - \quad C_{DES}\Delta\right) \tag{264}$$

where $f_d = 1 - tanh((8h)^3)$, $h = \frac{\tilde{\nu}}{\sqrt{\bar{u}_{ij}\bar{u}_{ij}\kappa^2 d^2}}$, $\kappa = 0.41$.

Improved Delayed Detached Eddy Simulations (DDES)

Shur et al. (2008) proposed an improved DDES to ensure that most of the turbulence is resolved when the model is operating as a wall modeled Large Eddy Simulation (LES).

This is achieved by introducing a blending function of length scales defined as:

$$\tilde{d} = \tilde{f}_d(1 + f_e)d + (1 - \tilde{f}_d)C_{DES}\Delta \tag{265}$$

where

- $\tilde{f}_d = \max([1 - f_d], f_b)$,
- $f_b = \min(2 \quad e^{-9a^2}, \quad 1.0)$,
- $a = 0.25 - \frac{d}{\Delta}$,
- $f_e = \max([f_{e1} - 1], \quad 1.0)f_{e2}$,

- $f_{e1} = \begin{cases} 2\ e^{-11.09a^2\frac{\Delta}{d}}, & if\ \ a \geq 0 \\ 2\ e^{-9a^2\frac{\Delta}{d}}, & if\ \ a < 0 \end{cases}$,
- $f_{e2} = 1.0 - \max\left(\tanh\left((C_t^2 h_t\right)^3\right), \tanh\left((C_l^2 h_l\right)^{10}\right)\right)$,
- $h_t = \frac{v_t}{\sqrt{\bar{u}_{ij}\bar{u}_{ij}}\kappa^2 d^2}$,
- $h_l = \frac{v}{\sqrt{\bar{u}_{ij}\bar{u}_{ij}}\kappa^2 d^2}$.

Near-Wall Modeling

For internal wall bounded flows, proper mesh resolution is required in order to calculate the steep gradients of the velocity components, turbulent kinetic energy, dissipation, as well as the temperature.

Furthermore, the first grid locations from the wall and the stretching ratio between subsequent points are also determinant factors that affect solution accuracy. Since boundary layer thickness is reduced as the Reynolds number increases, the computational cost increases for higher Reynolds number flows due to the dense grid requirements near the walls. Additionally, it is hard to determine the first grid locations for complex three dimensional industrial problems without adequate testing and simulation.

For high Reynolds number flows, it is numerically efficient if flows within boundary layers are modeled rather than resolved down to the wall. This suggests that a coarse mesh is used at the expense of numerical accuracy when compared to fully wall resolved approaches.

Wall Function

Figure 41 shows velocity profiles over a flat plate under the zero pressure gradient.

Velocity profile and wall distances are scaled by the friction velocity ($u_\tau = \sqrt{\frac{\tau_w}{\rho}}$ where τ_w is the wall shear stress and ρ is the fluid density). This is referred to as the log-law velocity profile. As shown in Figure 41, the velocity profiles are divided into three distinguished regions, the viscous sublayer, the buffer layer and the logarithmic layer.

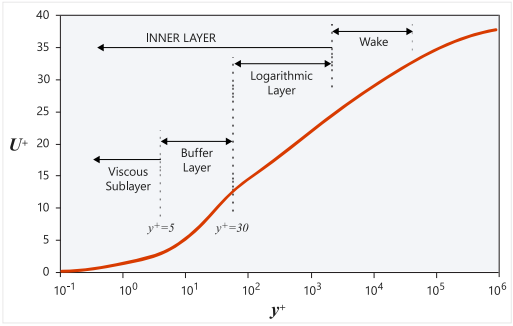


Figure 41: Log-law Velocity Profile

The wall function utilizes the universality of the log-law velocity profile to obtain the wall shear stress. This wall function approach allows the reduction of mesh requirement near the wall as the first mesh location is placed outside the viscous sublayer.

Overall, the log-law based wall model is economical and reasonably accurate in most flow conditions, especially for high Reynolds number flows. However, it tends to display poor performance in situations with low Reynolds number flows, strong body forces (rotational effect, buoyancy effect), and massively separated flows with adverse pressure gradients.

Velocity Profile in the Viscous Sublayer (y+ < 5)

In the viscous sublayer, the normalized velocity profile (U^+) has a linear relationship with the normalized wall distance.

$$U^+ = y^+ \tag{266}$$

where $U^+ = \frac{\overline{u}}{u_\tau}$ is the velocity ($\overline{u}$) parallel to the wall, normalized by the friction velocity. The friction velocity is defined as $u_\tau = \sqrt{\frac{\tau_w}{\rho}}$, τ_w is the wall shear stress and ρ is the fluid density. $y^+ = \frac{\rho u_\tau y}{\mu}$ is the normalized wall distance (or wall unit). The distance from the wall is y and μ is the fluid dynamic viscosity.

Velocity Profile in the Logarithmic Layer (30 < y+ < 500)

In the logarithmic layer the velocity profile can be given by a logarithmic function

$$U^+ = \frac{1}{\kappa}\log(y^+) + B \tag{267}$$

where κ= 0.4 is the Von Kármán constant and B = 5.5 is a constant.

The wall shear stress can be estimated from the above equation via the iterative solution procedure.

Velocity Profile in the Buffer Layer (5 < y+ < 30)

Since two equations mentioned before are not valid in the buffer layer a special function is needed to bridge the viscous sublayer and the logarithmic layer. Details are not covered here.

Kinematic Eddy Viscosity

The kinematic eddy viscosity can be obtained as $v_t = \kappa y u_T$.

Two Layer Wall Model

The two equation turbulence models based on eddy frequency (ω) and the Spalart-Allmaras (SA) model do not require special wall treatments to solve the boundary layer as these models are valid through the viscous sub layer.

However, the two equation turbulence models based on turbulence dissipation rate (ε) need additional functions to simulate the near wall effects. The two layer wall model is one of them. In the two layer model, turbulent kinetic energy is determined from the turbulent kinetic energy transport equation, while dissipation rate is resolved with a one equation turbulence model (Wolfstein, 1969) in the near wall, viscous-affected regions where $Re_y = \frac{\rho y \sqrt{k}}{\mu} < 200$.

$$\varepsilon = \frac{k^{3/2}}{l_\varepsilon} \tag{268}$$

where

- $l_\varepsilon = C_l y \left(1 - \exp\left[-\frac{Re_y}{A_\varepsilon}\right]\right)$,
- $C_l = \kappa C_\mu^{-3/4}$,
- κ is the Von Kármán constant,
- A_ε = 5.08 (Chen and Patel, 1988).

The eddy viscosity can be written as

$$v_t = C_\mu l_\mu \sqrt{k} \tag{269}$$

where

- $l_\mu = C_l y \left(1 - \exp\left[-\frac{Re_y}{A_\mu}\right]\right)$,
- A_μ = 70.

Inlet Turbulence Parameters

CFD simulations require specification of turbulence variables at inlet boundaries.

When measured turbulence data is available you can explicitly specify turbulence variables. For example, eddy viscosity for the Spalart-Allmaras (SA) model, turbulent kinetic energy and eddy frequency/dissipation rate for k-ε and k-ω based models. When measured data is not available, there

are estimations for the turbulence values that are based on the turbulence intensity and turbulence length scale or eddy viscosity ratio.

Turbulence Intensity

The turbulence intensity (I) is defined as the ratio of the root-mean-square of the turbulent velocity fluctuations (u') and the mean velocity ($\overline{u}$):

$$I \equiv \frac{u'}{\overline{u}} \tag{270}$$

where $u' = \sqrt{\frac{\left(\overline{u'_x{}^2} + \overline{u'_y{}^2} + \overline{u'_z{}^2}\right)}{3}} = \sqrt{\frac{2k}{3}}$ and $\overline{u} = \sqrt{\left(\overline{u}_x^2 + \overline{u}_y^2 + \overline{u}_z^2\right)}$.

For fully developed internal flows the turbulence intensity can be estimated as

$$I = 0.16 \quad Re_h^{-1/8} \tag{271}$$

where $Re_h = \frac{\rho \overline{u} D_h}{\mu}$ is the Reynolds number and D_h is a hydraulic diameter.

The following turbulence intensity values can be assumed

- 5 percent < I < 20 percent for rotating machineries, for example, turbines and compressors.
- 1 percent < I < 5 percent for internal flows.
- I~0.05 percent for external flows.

Turbulence Length Scale

The turbulence length scale represents the characteristic size of the turbulent eddies within a flow field. This parameter is often used to characterize the nature of the turbulence and appears in some form in nearly every turbulence model. RANS turbulence models have different definitions of the turbulence length scale based on the turbulence model and the type of application.

Estimated turbulence length scales based on the flow conditions:

- $l = 0.038 D_h$ for fully developed internal flows.
- $l = 0.22\delta$ for developing flows.

where $\delta \approx 0.382 \frac{x}{Re_x^{1/5}}$ is the turbulent boundary layer thickness over a flat plate and $Re_x = \frac{\rho \overline{u} x}{\mu}$ is the Reynolds number and x is the distance from the start of the boundary layer.

Eddy Viscosity Ratio

The eddy viscosity ratio (v_r) is the ratio between the eddy viscosity (v_t) and fluid kinematic viscosity (v). The eddy viscosity ratio is set between 1 and 10 for internal flows, while it should be between 0.2 and 1.3 for external flows.

Inlet Turbulence Specification for the SA Model

The SA model has three options to estimate the turbulence variable at the inlet.

- Option 1: Eddy viscosity v_t

- Option 2: Turbulence intensity I and length scale l. For the eddy viscosity, $\nu_t = \sqrt{\frac{3}{2}}\,\bar{u}\quad I\quad l$.
- Option 3: Viscosity ratio $\quad \nu_r$. For the eddy viscosity, $\nu_t = \nu\quad \nu_r$

Inlet Turbulence Specification for the k-ε Models

- Option 1: Kinetic energy k and dissipation rate ε
- Option 2: Turbulence intensity I and length scale l. For the kinetic energy, $k = \frac{3}{2}(\bar{u}I)^2$. For the dissipation rate, $\varepsilon = C_\mu \frac{k^{3/2}}{l}$, where C_μ = 0.09.
- Option 3: Turbulence intensity I and viscosity ratio $\quad \nu_r$. For the kinetic energy, $k = \frac{3}{2}(\bar{u}I)^2$. For the dissipation rate, $\varepsilon = \frac{C_\mu \quad k^2}{\nu \quad \nu_r}$.

Inlet Turbulence Specification for the k-ω Models

- Option 1: Kinetic energy k and eddy frequency ω
- Option 2: Turbulence intensity I and length scale l. For the kinetic energy, $k = \frac{3}{2}(\bar{u}I)^2$. For the eddy frequency, $\omega = \frac{\sqrt{k}}{l \quad C_{\mu 1/4}}$.
- Option 3: Turbulence intensity I and viscosity ratio $\quad \nu_r$. For the kinetic energy, $k = \frac{3}{2}(\bar{u}I)^2$. For the dissipation rate, $\omega = \frac{k}{\nu \quad \nu_r}$.

Turbulence Wall Function

The first option for computing turbulent boundary layers is to fully resolve them. When computing the near wall gradients explicitly, AcuSolve integrates the governing equations directly to the wall. As a result, this option is more accurate, provided that sufficient mesh density is used. y^+ (defined below) of the first mesh point must be less than 10 (preferably 5); otherwise, gross errors in traction, heat flux, and mass transfer may result. This option is typically used for applications where the near wall flow profile plays an important role in the physics of the simulation, that is, cases having adverse pressure gradients, flow separation, and so on. This option is activated by specifying the `turbulence_wall_type` = low_reynolds_number (or low_re) parameter in the `SIMPLE_BOUNDARY_CONDITION` command.

$$y^+ = \frac{\rho y u^*}{\mu} \tag{272}$$

where μ is the viscosity, $u^* = \sqrt{\frac{\tau_w}{\rho}}$ is the turbulent friction velocity, τ_w is the wall shear stress, ρ is the density.

The second type of wall treatment for turbulent boundary layers allows you to approximate the near wall flow field, without using fine near wall mesh, by employing wall functions. This approach can greatly reduce the size of a mesh by eliminating the need for fine mesh spacing normal to no-slip walls. When this approach is applied, AcuSolve assumes a shape for the near wall flow field. This

assumed profile is based on the "Law of the Wall" for turbulent boundary layers. The "Law of the Wall" is a relation that is based on theoretical and experimental arguments and relates the stream wise velocity profile with the normal distance from the wall. This relation was formulated for fully developed boundary layers with favorable pressure gradients. This option is activated by specifying the *turbulence_wall_type* = wall_function (or func) parameter in the SIMPLE_BOUNDARY_CONDITION command. y^+ of the first mesh point may be as large as 300.

The third type of wall model that is offered by AcuSolve is the running average wall function. When this model is employed, the wall function is evaluated using the running average velocity field and not the instantaneous field. This approach is typically used with LES and DES models, but may also be used with RANS if appropriate.

Note: This requires the Running Average field to be turned on in the simulation. This option is activated by specifying the *turbulence_wall_type* = running_average_wall_function parameter in the SIMPLE_BOUNDARY_CONDITION command.

Thermo-Electric Battery Solution

The design and performance of lithium-ion battery cells, modules and packs is important for a wide variety of battery applications. The thermo-electric battery solution enables detailed analysis of thermal and electric design and performance for these applications under a wide array of scenarios. The analysis is enabled through four core modeling components:

Battery cell core modeling	The cell is modeled as a homogenized region, where the battery voltage response and associated heat generation is determined using the solution of an ODE system. The solution utilizes two approaches: single potential and multi-scale multi-dimensional (MSMD).
Charge conservation	In electrically conducting components, for example, connectors, busbars, and terminals, the charge conservation equation is solved and coupled to the energy equation as a source term.
Transient thermal conduction	Transient thermal conduction in all cell/module/pack components is solved using the energy equation.
Cooling	Optionally, the Navier-Stokes equation can be included to design and enhance cooling systems.

Collectively, the components allow comprehensive analysis and optimization of the thermal and electrical performance of battery systems.

Two overarching methods are employed in the thermo-electric battery solution: single potential and multi-scale multi-dimensional (MSMD). The former solution is applicable for all battery types, for example, cylindrical-cells, prismatic and pouch-type cells, while the latter is only applicable for pouch cells. Each approach solves a variant of charge conservation and the energy equation for the batteries:

Single potential	This approach uses equivalent circuit models (ECM) to characterize the voltage-current response. From the ECM, a source term can be derived and applied to the energy equation and boundary conditions based on current and terminal voltage drop for charge conservation in the connected components of the batteries, for example, terminals/tabs, busbars and any other electrically conducting component.
MSMD	This approach employs a multi-scale method, with a cell scale and a sub-domain scale. On the cell scale, two potential fields represent the potentials of the negative and positive current collectors. The sub-domain model, currently only supporting ECM models, represents the voltage-current response of the battery derived from the solution of an ODE system. Inter-domain coupling between the sub-domain and cell scale is achieved via averaging source terms to eliminate any spatial dependence, in the dual electric potential fields and the energy equation. Cell scale to sub-domain coupling is achieved using spatially resolved variables directly in the sub-domain equations.

The ECM models supported are first, second and third order, where the former has the advantage of simplicity, for example, fewer parameters, and the latter two give a more accurate battery cell voltage response.

A more detailed summary of the models is given in the following sections.

Equivalent Circuit Modeling

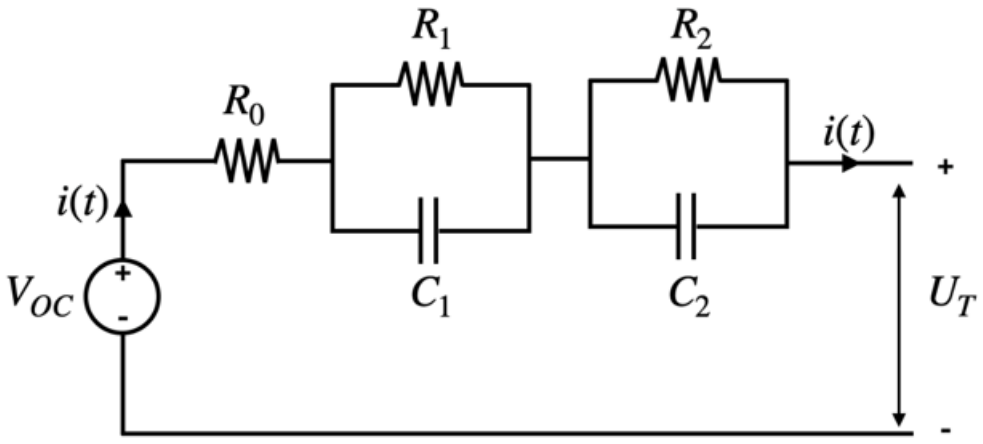


Figure 42: 2nd Order Equivalent Circuit Battery Model

In an equivalent circuit model (ECM), the electric behavior, for example, the voltage-current response, is modeled using a phenomenological electric circuit approach. For example, in a 2nd order ECM (see Figure 42), that is, two RC pairs, the governing set of ODEs are given by

$$U_T = V_{ocv}(soc, T) - V_1 - V_2 - R_0(soc, T)\, i(t) \qquad (273)$$

$$\frac{dV_1}{dt} = -\frac{1}{R_1(soc,T)C_1(soc,T)}V_1 - \frac{1}{C_1(soc,T)}i(t) \quad (274)$$

$$\frac{dV_2}{dt} = -\frac{1}{R_2(soc,T)C_2(soc,T)}V_2 - \frac{1}{C_2(soc,T)}i(t) \quad (275)$$

Where U_T is the terminal voltage, soc is the state of charge, T is the temperature, V_{ocv} is the open circuit voltage, R_0 is the ohmic resistance, R_1 and R_2 are the polarization resistances of the first and second RC pairs respectively, C_1 and C_2 are the polarization capacitance of the first and second RC pairs respectively, $i(t)$ is the input current, and t is the time.

The output terminal voltage is therefore dependent on the input current and the ECM parameters (R_0, R_1, R_2, C_1 and C_2) which can depend on both state of charge and temperature.

In AcuSolve, the sign convention dictates that current is positive during discharge. AcuSolve supports three types of ECM models: 1st, 2nd, and 3rd order. The choice of model order depends on the desired dynamics, such as dynamic current conditions. The 2nd and 3rd order models offer enhanced accuracy for pulse-discharge experiments and dynamic current scenarios. Typically, the 2nd order model is considered the optimal choice, striking a balance between accuracy and ease of parameter determination.

The battery state of charge (soc) is updated using coulomb counting:

$$\frac{dsoc}{dt} = \frac{I}{3600\,Q_{Ah}} \quad (276)$$

where I is the current flux. State of charge (soc) represents the amount of charge in a battery relative to its nominal capacity (Q_{Ah}).

Single Potential Model

In the single potential model, the equivalent circuit model (ECM) computes the voltage drop and current within the homogenized section of the battery cell, automatically applying associated boundary/interface conditions. Briefly, the ECM model is stepped in time and the terminal voltage at time k is updated. For a 1st order ECM model this would be:

$$U_T^k = V_{OCV} - I_\phi^k R_0 - R_1 \exp\left(\frac{-\Delta t}{R_1 C_1}\right) I_{R_1}^{k-1} + R_1 \left(1 - \exp\left(\frac{-\Delta t}{R_1 C_1}\right)\right) I_\phi^{k-1} \quad (277)$$

where I_ϕ is the current flux on the negative terminal of the battery (from the charge conservation PDE solution), V_{ocv} is the open circuit voltage, R_0 is the ohmic resistance, R_1 is the polarization resistance of the 1st RC pair, C_1 is the polarization capacitance of the 1st RC pair, $I_{R_1}^{k-1}$ is diffusion resistor current at time step k-1, and Δt is the time step size.

The terminal voltage is applied to the positive terminal of the battery for the solution of charge conservation in the electrically conducting components of the battery pack. Proper distribution of current is determined from the pack load and how this distributes to individual cells in the modules that make up the pack. This distribution is based on the properties of a cell, for example, SOC level or temperature. The current distribution is applied as a flux condition at the negative terminals of the battery cells.

The governing equation for the thermal (energy equation) and electrical solution (charge conservation) are given by:

$$\rho c_p \frac{dT}{dt} = \nabla \cdot (\kappa \nabla T) + q_B + q_J \quad (278)$$

$$\nabla \cdot (-\sigma \nabla \phi) = 0 \quad (279)$$

where q_B is the heat generation (see later for a description of this heat source) from the battery and q_J is the joule heat generated in electrically conducting components, σ is the conductivity and ϕ is the electric potential.

This source term, derived from Joule's first law, denotes the heat generated per unit volume, calculated as the product of current density j and electric field ($E = -\nabla\phi$):

$$q_J = j \cdot E = \sigma |\nabla \phi|^2 \quad (280)$$

MSMD Model

For pouch batteries the thermal-electric response can be modeled using a MSMD approach which bridges the scale between the cell domain and the electrode scale. In this approach electric potential fields are modeled in a cell composite, or more specifically the current collectors, together with the thermal field. This is then coupled using a multi- scale approach down to the sub-scale, where the electrochemical phenomena, that is, the inner workings of the battery, are modeled.

Cell scale model

At the cell scale, the temperature and a pair of electric potential fields are solved. More specifically, the 3D battery geometry is treated as a homogenized medium with orthotropic electrical and thermal conductivities. Positive and negative electric potential is computed in the battery from charge conservation at the current collectors of the cell composite. For the negative current collector, the governing equation is:

$$\nabla (\sigma_- \nabla \phi_-) - j = 0 \quad (281)$$

Equivalently at the positive current collector the governing equation is:

$$\nabla \left(\sigma_+ \nabla \phi_+\right) + j = 0 \quad (282)$$

where j is the transfer current density and is modeled through an electrochemical sub-model, for example, the ECM model. Additionally, j is subject to the constraint:

$$I_{\text{Cell}} = \int_\Omega j \, d\Omega \quad (283)$$

where I_{Cell} is the current through the cell. The above system of equations comes with associated boundary conditions such as current, for example, constant current discharge, or voltage.

The energy equation in AcuSolve is modified by an additional source term that couples both sources of ohmic (or joule) heating from the current collectors and electrochemical reaction heat. The total heat source term is given by:

$$q = \sigma_- |\nabla \phi_-|^2 + \sigma_+ |\nabla \phi_+|^2 + q_B \quad (284)$$

where the first two terms are ohmic heating and the last term, q_B, is the heat generated by the ECM model.

Note: The ECM model can be closer tied to an electrochemical model by fitting parameters to virtual experimental data provided by a DFN (Doyle Fuller Newman) model.

MSMD multi-cell formulation

In a battery module, batteries are connected in a nSmP pattern, for example, 8s6p, where m batteries are connected in parallel and then n of these parallel connected units are connected in serial to form a pack. In the MSMD method, two potential equations are solved for the ϕ_1 and ϕ_2 electrical field. Depending on battery connectivity, ϕ_1 and ϕ_2 could be the potential from the positive current collector or the negative one. Therefore, in a battery module simulation, the two potential equations are solved with the transfer current density changing sign dependent on the location in the module. The cells are alternating as odd and even stages, see Figure 43. In tabs and busbars, you only solve for one of the fields. In Figure 43, the ϕ_1 and ϕ_2 on the left indicates which field was solved in the tabs/busbars in the line. The governing equations for the cell scale become:

$$\nabla(\sigma\nabla\phi) = j \tag{285}$$

For ϕ_1 in an odd stage:

$$\sigma = \sigma_+ \qquad j = -I/V \tag{286}$$

For ϕ_1 in an even stage:

$$\sigma = \sigma_- \qquad j = I/V \tag{287}$$

For ϕ_2 in an odd stage:

$$\sigma = \sigma_- \qquad j = I/V \tag{288}$$

For ϕ_2 in an even stage:

$$\sigma = \sigma_+ \qquad j = -I/V \tag{289}$$

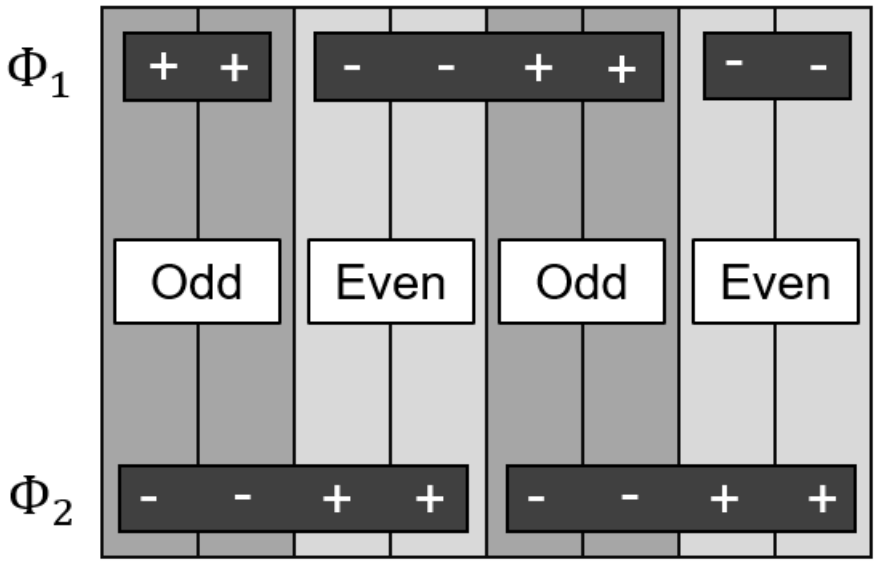


Figure 43: Odd and Even Stages for Connected Cells

Bernardi's Heat Generation Equation

Bernardi's heat generation equation represents the irreversible and reversible heat generation from the battery and is included in the energy equation as a source term derived from the solution of the ECM model, given by

$$q_B = \frac{i(t)}{Volume_{cell}}\left((V_{ocv} - U_T) - T\frac{dV_{oc}}{dT}\right) + \sigma_+\nabla^2\sigma_+ + \sigma_-\nabla^2\sigma_- \quad (290)$$

The first term of the RHS represents the cell over potential and includes irreversibilities, such as ohmic losses and mass transport limitations, while the second term represents entropic heat generation. The last two terms represent ohmic losses and are enabled only for the MSMD approach. Bernardi's heat generation equation is automatically included in the energy equation as a source term for all battery models. The entropic heat can be both temperature and state_of_charge dependent.

Battery Thermal Runaway Model

Thermal runaway in battery packs is a major safety concern with potentially catastrophic outcomes, such as battery pack fires. This phenomenon occurs in batteries due to exothermic degradation reactions when a battery is subjected to abuse conditions, such as physical damage, internal short circuits, overcharging, or overheating. These conditions, along with the associated decomposition of battery components, for example, anode, cathode, and separator, lead to significant heat release and an uncontrollable rise in temperature. When a single cell enters these thermally unstable conditions, the heat release can cause adjacent cells to heat up and enter thermal runaway, eventually propagating through and consuming the entire battery pack. Additionally, the decomposition reactions generate flammable gases that can ignite, causing fires to spread through the battery pack.

SimLab battery solutions for thermal runaway provide a virtual experimental platform to test pack designs for thermal safety, including the assessment of propagation characteristics and heat shield effectiveness. Three models are available to simulate thermal runaway:

- NREL model
- Arc reaction heat model
- Heat rate model

The first two models are chemical kinetic models based on the Arrhenius equation. The NREL model takes a mechanistic approach to model the kinetics and heat release of individual cell components, while the arc reaction heat model adopts a more phenomenological approach, deriving the kinetics directly from ARC data. The heat rate model also uses ARC data but directly reads this data to determine the volumetric heat source.

All of the mathematical models for thermal runaway add an additional heat source (S_{TR}) to the energy equation.

$$\rho c_p \frac{dT}{dt} = \nabla \cdot (\kappa \nabla T) + S_{TR} \tag{291}$$

The form of the heat source (aggregated or direct calculation) depends on the model. A brief description of these models and the heat source term is given in the following sections.

NREL Thermal Runaway Model Formulation

The mathematical model for thermal abuse involves heat generation resulting from series of exothermic decomposition reactions of various battery components. The heat generation from these individual reactions are aggregated and incorporated into the energy equation as a combined source term.

The decomposition reactions within the battery are characterized using Arrhenius equations, accounting for: 1) Solid electrolyte interface (SEI) decomposition at the anode-electrolyte interface ($\sim 80-100\ {}_oC$); 2) Anode decomposition (intercalated lithium reacting with electrolyte, facilitated by SEI decomposition); 3) Cathode conversion (active material decomposition releasing oxygen), which is highly exothermic; and 4) Electrolyte decomposition at very high temperatures. The final source term from the above is written: $S_{TR} = S_{SEI} + S_a + S_c + S_{elec}$, representing the heat release from the exothermic decomposition reactions of the SEI, anode, cathode, and electrolyte, respectively.

The governing equations for these reactions and each exothermic heat source contributing to the energy equation are summarized below.

SEI decomposition reaction rate:

$$\frac{dx_s}{dt} = -x_s \cdot A_s \cdot \exp\left(-\frac{E_s}{k_b T}\right) \tag{292}$$

Where x_s is the fraction of lithium in the anode, A_s the frequency factor for anode decomposition, E_s the activation energy for anode decomposition, T the temperature, and k_b the Boltzmann constant.

SEI decomposition exothermic heat:

$$S_{SEI} = H_{SEI} W_a \frac{dx_s}{dt} \tag{293}$$

Anode decomposition reaction rate:

$$\frac{dx_a}{dt} = -x_a \cdot A_a \cdot exp\left(-\frac{E_a}{k_b T}\right) \cdot exp\left(-\frac{z}{z_0}\right) \quad (294)$$

Where x_a is the fraction of lithium in the anode, A_a the frequency factor for anode decomposition, E_a the activation energy for anode decomposition. As part of anode decomposition, the tunneling effect is considered due to the reduction in thickness of the SEI, where z represents the relative SEI thickness:

$$\frac{dz}{dt} = x_a \cdot A_a \cdot exp\left(-\frac{E_a}{k_b T}\right) \cdot exp\left(-\frac{z}{z_0}\right)$$

Anode decomposition exothermic heat:

$$S_a = H_a W_a \frac{dx_a}{dt} \quad (295)$$

Where H_a is the negative solvent enthalpy, W_a is the specific carbon content in the jelly roll.

Cathode conversion reaction rate:

$$\frac{d\alpha}{dt} = -\alpha(1-\alpha) \cdot A_c \cdot exp\left(-\frac{E_c}{k_b T}\right) \quad (296)$$

Cathode conversion exothermic heat:

$$S_c = H_c W_c \frac{d\alpha}{dt} \quad (297)$$

Where H_c is the positive solvent enthalpy, W_c is the specific cathode active material content in the jelly roll.

Electrolyte decomposition exothermic heat:

$$\frac{dc_{elec}}{dt} = -c_{elec} \cdot A_{elec} \cdot exp\left(-\frac{E_{elec}}{k_b T}\right) \quad (298)$$

Finally, short circuit events can also be included, for example, separator melting leading to direct connection between the anode and cathode. The short circuit leads to charge depletion and is given as a function of state of charge (SOC):

$$\frac{dSoC}{dt} = -T_{isc} \cdot SoC \cdot A_{ec} \cdot exp\left(-\frac{E_{ec}}{k_b T}\right) \quad (299)$$

In which T_{ISC} becomes active when temperature is above a pre-defined internal short circuit temperature.

ARC Reaction Model Formulation

The ARC reaction model is a staged Arrhenius-based kinetic model that can be directly fitted to Accelerating Rate Calorimetry (ARC) data. This enables fitting N number of stages to the data based on the desired level of accuracy. Up to five ODEs (i = 1-5) are employed to represent the data, each with distinct parameters for activation and decay, characterizing a specific stage. The summation of these heat sources represents the ARC heat rate data. The general form of the equation is as follows:

$$\frac{d\alpha_i}{dt} = \alpha_i^n (1-\alpha_i)^m \cdot A_{(a,i)} \cdot exp\left(-E_{(a,i)} / (k_b T)\right) \quad (300)$$

Where $A_{(a,i)}$ is the frequency factor for a specific stage (1/s), $E_{(a,i)}$ is the activation energy (J), k_b is the Boltzmann constant, T the cell temperature (K), n and m are related to the form of the solid reaction model. Depending on the value of m this reaction can be nth order (when m=0) or an auto-catalytic type, that is, the reaction increases as the product is generated (m>0).

The above equation is a more phenomenological modeling approach compared to the NREL model; however, it can still be motivated from a physical perspective providing the stages are divided into bands that represent an approximate physical reaction process occurring, for example, anode decomposition.

The total exothermic heat generation for this model is given by,

$$mc_p\frac{dT}{dt} = \sum h_i \cdot \frac{\mathrm{d}\alpha_i}{\mathrm{d}t} \tag{301}$$

Where h_i is the enthalpy of the component defined by,

$$h_i = m_i c_{p,i} \Delta T \tag{302}$$

The heat contribution for each stage combines to form the heat source, S_{TR}, in the energy equation.

Direct Reading of ARC Data Formulation

A direct reading of thermal runaway ARC data provides a source term directly in the energy equation derived from the heat rate-temperature data. To calculate this heat source using ARC data the following equation is utilized:

$$S_{TR} = \rho_{cell} c_{p,cell} \frac{\mathrm{d}T}{\mathrm{d}t} \tag{306}$$

Where ρ_{cell} is the effective density of the cell, $c_{p,cell}$ is the effective specific heat of the cell and dT/dt is the heat rate data directly read from the ARC test.

Co-Simulations Using AcuSolve

This section details the capabilities of AcuSolve in simulating the multi-physics problems by coupling with structural solvers.

Fluid Structure Interaction (FSI)

One of the rich features of AcuSolve lies in its ability to simulate FSI problems. AcuSolve simulates these problems using two different technologies:

Practical FSI (P-FSI)

In P-FSI simulations, AcuSolve and the structural code are run separately. The Eigen solution obtained from a structural solver is projected onto the CFD mesh and used as boundary conditions to displace the fluid mesh. The mass, stiffness and damping arrays are also transferred from the structural model to flexible body in the CFD model, completing the inputs necessary to compute

	the structural deformation. P-FSI is preferred for simulations that have linear structural response (linear material properties and small mesh deformations).
Direct Coupled FSI (DC-FSI)	For applications involving a nonlinear structural response, AcuSolve is coupled at run time to a structural solver using the DC-FSI approach. The codes are run in tandem and exchange forces and displacements at each time step. This technique captures all nonlinearities in the structural model and can exploit the full capabilities of the structural solver. The DC-FSI technology within AcuSolve performs all projection and interpolation between the structural and CFD meshes without the use of any middleware. The supported structural solvers for AcuSolve include OptiStruct, Abaqus and MD-Nastran.

Coupled Multibody Dynamics (MBD) with AcuSolve

AcuSolve can be directly coupled with MotionSolve, a multibody dynamics solver, to solve multi-physics problems that involve both fluid and rigid body dynamics.

The wetted surfaces are paired with rigid bodies and loads. Displacements are exchanged during the run time using AcuSolve's code coupling interface. The coupling is currently supported for rigid body motion only and not for a flexible body. However, AcuSolve supports both one way and two way coupling with MotionSolve. With one way coupling, the displacements are read from a file generated from MotionSolve and projected onto AcuSolve. With two way coupling, the codes exchange forces and displacement at each time step.

Altair AcuSolve EDEM Coupling

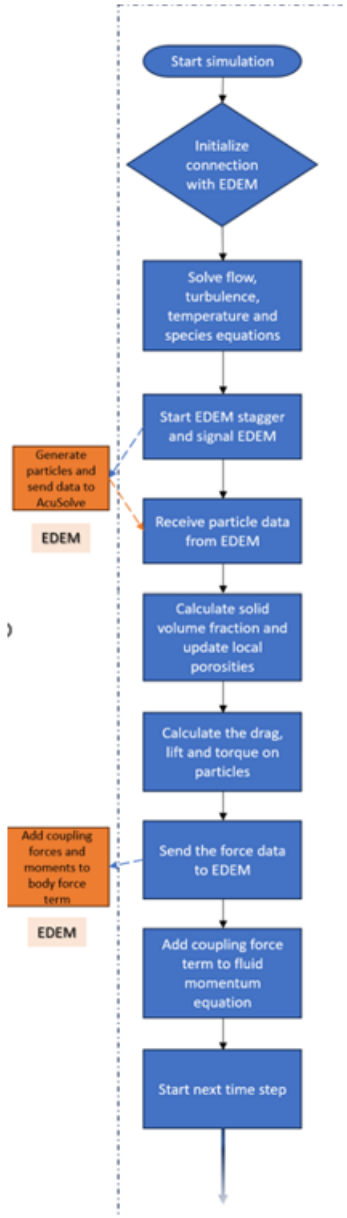


Figure 44:

Note: The detailed DEM simulation sequence is not shown here. For detailed information about the DEM simulation sequence, refer to the EDEM help manual.

Note:

- The temperature and species equations are only solved when the heat transfer and/or mass transfer physics models are active.
- Since the DEM time step is usually multiple orders lower than the CFD time step, the DEM solver loop is repeated multiple times per single CFD time step to ensure that the physical time is synchronized in both the solvers.
- For coupled simulations, the data is always exchanged in SI units.
- For unidirectional coupling, once the coupling forces are calculated and shared with EDEM, the fluid momentum equation is not updated with the coupling force because the effect of particles on fluid is ignored.

AcuSolve-EDEM coupling uses the Eulerian-Lagrangian approach for modeling fluid-particle flows where the fluid transport equations are solved in a Eulerian framework and the dispersed phase is represented as Lagrangian particles. The fluid phase is solved based on the volume-averaged Navier-Stokes equations and the Discrete Element Method (DEM) is used for computing the motion of the solid phase. This coupling strategy allows you to study the momentum and heat transfer at the individual particle scale.

Governing Equations

The volume-averaged Navier-Stokes equations for CFD-DEM momentum coupling are given by:

$$\frac{\partial(\varepsilon_f \rho_f)}{\partial t} + \nabla \cdot (\rho_f \varepsilon_f v_f) = 0 \tag{307}$$

$$\frac{\partial}{\partial t}\left(\varepsilon_f \rho_f v_f\right) + \nabla \cdot \left(\rho_f \varepsilon_f v_f v_f\right) = -\varepsilon_f \nabla p - \varepsilon_f \quad \nabla \cdot \left(\tau_f\right) + \rho_f \varepsilon_f g - F_p \tag{308}$$

where,

- ε_f is the fluid volume fraction or porosity;
- v_f is the fluid velocity;
- ρ_f is the fluid density;
- p is the fluid pressure;
- τ_f is the viscous stress tensor;
- g is the gravity vector;
- $F_p = \frac{1}{V_{cell}} \sum_i \left(f^d + f^l\right)_i$ is the particle-fluid force exchange term;
- V_{cell} is the volume of fluid cell;
- f^d is the fluid drag force;
- f^l is the fluid lift force.

Note: Wherever applicable the following notations are used throughout the document. The subscript *f* denotes a fluid property, *p* denotes a particle property and *i* denotes that the calculation is done for an individual particle.

The drag force on each particle is given by:

$$f_i^d = \beta^D \frac{V_p}{d_p}(\mathbf{v}_{slip}) \tag{309}$$

- $\mathbf{v}_{slip}$ is the slip velocity ($\mathbf{v}_f - \mathbf{v}_p$)
- V_p is the particle volume
- d_p is the particle diameter

Where the formulation of the interphase momentum exchange coefficient (β^D) varies based on the drag model. The drag models available in AcuSolve-EDEM coupling are listed below.

Drag Models

1. Ergun-Wen Yu

The momentum exchange coefficient for the Ergun Wen Yu drag model is given by,

$$\beta^D_{Ergun} = \frac{A(1-\varepsilon_f)\mu_f}{\varepsilon_f \varphi^2 d_p} + \frac{B\rho_f|\mathbf{v}_{slip}|}{\varphi} \qquad \varepsilon_f < 0.8 \tag{310}$$

$$\beta^D_{WenYu} = \frac{3}{4} C_d \rho_f |\mathbf{v}_{slip}| \varepsilon_f^{-1.65} \qquad \varepsilon_f \geq 0.8 \tag{311}$$

Where, the coefficients A and B have a default value of 150 and 1.75 respectively and can be modified by you while specifying the model inputs. v_p is the particle velocity and d_p is the particle's volume equivalent sphere diameter. φ is the sphericity of the particle.

$$C_d = \begin{cases} 0 & \mathrm{Re}_i = 0 \\ \frac{24}{\mathrm{Re}_i}\left(1 + 0.15\mathrm{Re}_i^{0.687}\right) & \mathrm{Re}_i \leq 1000 \\ 0.44 & \mathrm{Re}_i > 1000 \end{cases} \tag{312}$$

Here Re_i is the particle Reynolds number and C_d is the drag coefficient.

$$\mathrm{Re}_i = \frac{\rho_f \varepsilon_f |\mathbf{v}_{slip}| d_p}{\mu_f} \tag{313}$$

The Ergun-Wen-Yu model is one of the most widely used drag models and is recommended for most of the fluid-particle flows since it works well for both dense phase and dilute phase regimes. In this model, the Ergun equation is used for fluid volume fractions less than 0.8 and the Wen-Yu equation for fluid volume fractions greater than 0.8.

2. Di Felice

The momentum exchange coefficient for the DiFelice drag model is given by,

$$\beta^D_{diFelice} = \frac{3}{4} C_d \rho_f |\mathbf{v}_{slip}| \varepsilon_f^{(2-\chi)} \tag{314}$$

$$\chi = \begin{cases} 0 & \mathrm{Re}_i = 0 \\ 3.7 - 0.65e^{(-0.5(1.5-\log_{10}\mathrm{Re}_i)^2)} & \mathrm{Re}_i \neq 0 \end{cases} \tag{315}$$

$$C_d = \begin{cases} 0 & \mathrm{Re}_i = 0 \\ \left(0.63 + 4.8\,\mathrm{Re}_i^{-0.5}\right)^2 & \mathrm{Re}_i \neq 0 \end{cases} \tag{316}$$

Unlike the Ergun-Wen-Yu correlation, the Di Felice correlation is a monotonic function of Reynolds number and porosity and does not have the step change in drag force evaluation.

3. Beetstra

The momentum exchange coefficient for the Beetstra drag model is given by,

$$\beta^D_{Beetstra} = \frac{A\mu_f}{\varepsilon_f d_p} + \frac{B\mu_f \mathrm{Re}_i}{d_p} \tag{317}$$

$$A = 180(1-\varepsilon_f) + 18\varepsilon_f^4\left(1 + 1.5\sqrt{(1-\varepsilon_f)}\right) \tag{318}$$

$$B = \frac{0.31(\varepsilon_f^{-1} + 3(1-\varepsilon_f)\varepsilon_f + 8.4\mathrm{Re}_i^{-0.343})}{1 + 10^{3(1-\varepsilon_f)}\mathrm{Re}_i^{2\varepsilon_f - 2.5}} \tag{319}$$

4. Rong

The momentum exchange coefficient for the Rong drag model is given by,

$$\beta^D_{Rong} = \begin{cases} \frac{150(1-\varepsilon_f)\mu_f}{\varepsilon_f d_p \varphi^2} + \frac{3}{4}\frac{\rho_f |\mathbf{v}_{slip}|}{\varphi} & \varepsilon_f < 0.8 \\ \frac{3}{4} C_d \rho_f |\mathbf{v}_{slip}| \varepsilon_f^{2-\sigma-\lambda} & \varepsilon_f \geq 0.8 \end{cases} \tag{320}$$

where,

- $\sigma = 2.65(\varepsilon_f + 1) - (5.3 - 3.5\varepsilon_f)\ \varepsilon_f^2 e^{\left[-\frac{(1.5 - \log_{10}\mathrm{Re}_i)^2}{2}\right]}$
- $\lambda = (1-\varphi)\left(C - D\, e^{-0.5(3.5 - \log_{10}\mathrm{Re}_i)^2}\right)$
- $C = 39\varphi - 20.6$
- $D = 101.8(\varphi - 0.81)^2 + 2.4$
- φ is the sphericity of the particle

Since the sphericity of the particle is considered while calculating the drag force, this model is strongly recommended for non-spherical particles compared to the other models available in AcuSolve.

5. Syamlal-O'Brien

The momentum exchange coefficient for the Syamlal-O'Brien drag model is given by,

$$\beta^D_{Syamlal} = \frac{3}{4}\frac{C_d \varepsilon_f \rho_f |\mathbf{v}_{slip}|}{v_{r,p}^2} \tag{321}$$

where,

$$C_d = \begin{cases} 0 & \mathrm{Re}_i = 0 \\ \left(0.63 + \frac{4.8}{\sqrt{\mathrm{Re}_i / v_{r,p}}}\right)^2 & \mathrm{Re}_i \neq 0 \end{cases} \tag{322}$$

$$v_{r,p} = 0.5\left(A - 0.06\mathrm{Re}_i + \sqrt{\left(0.0036\mathrm{Re}_i^2\right) + 0.12\mathrm{Re}_i(2B - A) + A^2}\right) \quad (323)$$

$$A = \varepsilon_f^{4.14} \quad (324)$$

$$B = \begin{cases} 0.8\varepsilon_f^{1.28} & \varepsilon_f \le 0.85 \\ \varepsilon_f^{2.65} & \varepsilon_f > 0.85 \end{cases} \quad (325)$$

6. Wen-Yu

The momentum exchange coefficient for the Wen-Yu drag model is given by,

$$\beta^D_{WenYu} = \frac{3}{4} C_d \varepsilon_f^{-1.65} \rho_f |\mathbf{v}_{slip}| \quad (326)$$

$$C_d = \begin{cases} 0 & \mathrm{Re}_i = 0 \\ \dfrac{24}{\mathrm{Re}_i}\left(1 + 0.15\mathrm{Re}_i^{0.687}\right) & \mathrm{Re}_i \le 1000 \\ 0.44 & \mathrm{Re}_i > 1000 \end{cases} \quad (327)$$

7. Schiller Nauman

The momentum exchange coefficient for the Schiller Nauman drag model is given by,

$$\beta^D_{Schiller} = \frac{3}{4} C_d \rho_f |\mathbf{v}_{slip}| \quad (328)$$

$$C_d = \begin{cases} 0 & \mathrm{Re}_i = 0 \\ 24\left(1 + 0.15\mathrm{Re}_i^{0.687}\right) & \mathrm{Re}_i \le 1000 \\ 0.44 & \mathrm{Re}_i > 1000 \end{cases} \quad (329)$$

The drag force calculated does not consider the effect of surrounding particles, that is, volume fraction is not accounted for, and hence this model is strictly valid only for dilute phase flows.

Non-Spherical Drag Coefficient Models

The effect of the particle's shape can be taken into account by using non-spherical drag coefficient models. There are two types of models available in AcuSolve which are listed below. If the non-spherical drag coefficient model is set to none then the particles are assumed to be of spherical shape. But when the drag coefficient model is set to either of the models listed below, the drag coefficient in the drag models will be replaced by the non-spherical drag coefficient.

1. Isometric (Haider Levenspiel)

In this model the drag coefficient is a function of particle Reynolds number and sphericity. The instantaneous orientation of the particle is not considered. This type of model is applicable for particles with shapes closer to a sphere, such as rocks and some grains (beans), and when the orientation of the particles is not critical. The user inputs required for this model are particle's volume and sphericity. The Haider-Levenspiel correlation is given by:

$$C_{dns} = \frac{24}{\mathrm{Re}_c}\left[1 + A_1(\mathrm{Re}_c)^{A_2}\right]\frac{A_3}{\left(1 + A_4/\mathrm{Re}_c\right)} \quad (330)$$

$$Re_c = \min(Re_i, 2.5999e5) \tag{331}$$

Where the constants $A_{1\text{-}4}$ are dependent on the sphericity (φ) of the particle.

Table 10:

$\varphi < 0.67$	$0.67 \le \varphi < 0.99999$	$\varphi \ge 0.99999$
$A_1 = e^{(2.3288 - 6.4581\varphi + 2.4486\varphi^2)}$ $A_2 = 0.0964 + 0.5565\varphi$ $A_3 = e^{(4.905 - 13.8944\varphi + 18.4222\varphi^2 - 10.}$ $A_4 = e^{(1.4681 + 12.2584\varphi - 20.7322\varphi^2 + 15}$	$A_1 = 8.1761e^{(-4.0655\varphi)}$ $A_2 = 0.0964 + 0.5565\varphi$ $A_3 = 73.69e^{(-5.0748\varphi)}$ $A_4 = 5.378e^{(6.2122\varphi)}$	$A_1 = 0.1806$ $A_2 = 0.6459$ $A_3 = 0.4251$ $A_4 = 6880.95$

2. Non-spherical (Ganser and Holzer-Sommerfeld)

The Ganser and Holzer-Sommerfeld models consider both the shape and orientation of the particle. Since the orientation of the particles is also considered, this model is applicable to particle shapes such as disk, ellipsoid and elongated cylinder. The user inputs for these models are volume and aspect ratio of the particles.

The Ganser correlation is given by:

$$C_{d^{ns}} = \frac{24}{k_1 Re_i}\left[1 + 0.1118(k_1 k_2 Re_i)^{0.6567}\right] + \frac{0.4305 k_2}{\left(1 + {}^{3305}\!/_{k_1 k_2 Re_i}\right)} \tag{332}$$

Here k_1 and k_2 are Stokes and Newton shape factors respectively.

$$k_1 = \frac{1}{\left(\frac{\phi_{proj}}{3} + \frac{2}{3\sqrt{\varphi}}\right)} \tag{333}$$

$$k_2 = 10^{1.8148(-\log_{10}(\varphi))^{0.5743}} \tag{334}$$

Where ϕ_{proj} is the particle's projected diameter ratio and is dependent on the aspect ratio (A_R) of the particle and the angle between the principal axis and the fluid velocity vector (a).

$$\phi_{proj} = \begin{cases} \dfrac{\sqrt{\frac{4}{\pi} A_R \sqrt{(\sin^2 a)} + \sqrt{(\cos^2 a)}}}{\left(\frac{3A_R}{2}\right)^{1/3}} & \ldots\ldots A_R = 1 \\ \dfrac{\sqrt{A_R \sqrt{(\sin^2 a)} + \sqrt{(\cos^2 a)}}}{(A_R)^{1/3}} & \ldots\ldots A_R \neq 1 \end{cases} \tag{335}$$

Since this model only takes the aspect ratio as an input, the sphericity (φ) is calculated from the aspect ratio using the following correlation:

$$\varphi = \begin{cases} \frac{(1.5A_R)^{2/3}}{(A_R+0.5)} & A_R = 1 \\ \frac{(A_R)^{2/3}}{\left(\frac{(1+2A_R 1.61)}{3}\right)} & A_R \neq 1 \end{cases} \tag{336}$$

The Holzer-Sommerfeld correlation is given by:

$$C_{dns} = \frac{8}{\mathrm{Re}_i}\frac{1}{\sqrt{\varphi_\perp}} + \frac{16}{\mathrm{Re}_i}\frac{1}{\sqrt{\varphi}} + \frac{3}{\sqrt{\mathrm{Re}_i}}\frac{1}{\varphi^{3/4}} + 0.42^{0.4(-\log_{10}\varphi)^{0.2}}\frac{1}{\varphi_\perp} \tag{337}$$

Where sphericity is calculated from the aspect ratio using the correlation:

$$\varphi = \begin{cases} \frac{(1.5A_R)^{2/3}}{(A_R+0.5)} & A_R = 1 \\ \frac{(A_R)^{2/3}}{\left(\frac{(1+2A_R 1.61)}{3}\right)^{0.62111801242236}} & A_R \neq 1 \end{cases} \tag{338}$$

$\varphi^\perp$ is the crosswise sphericity and is calculated as shown below:

$$\varphi_\perp = \begin{cases} \frac{\pi\left(\frac{3A_R}{2}\right)^{2/3}}{\left(4A_R\sqrt{(\sin^2 a)}+\pi\sqrt{(\cos^2 a)}\right)} & A_R = 1 \\ \frac{A_R{}^{2/3}}{\left(A_R\sqrt{(\sin^2 a)}+\sqrt{(\cos^2 a)}\right)} & A_R \neq 1 \end{cases} \tag{339}$$

Lift Models

Generally, the lift force acts in a direction normal to the relative motion of the fluid and particle. The two components of the lift force considered are Saffman force and Magnus force. The Saffman lift force is due to the pressure gradient on a non-rotating particle in the presence of a non-uniform shear velocity field while the Magnus lift force is due to the particle rotation in a uniform flow. Unlike spherical particles, the behavior of non-spherical particles in turbulent flows is much more complicated and the lift force acting on them can no longer be neglected. As the particle's principal axis becomes inclined with the flow direction, the effect of lift force on the particle motion becomes significant.

The lift force on a particle is given by:

$$f_i^L = \beta_{ls}^L(v_{slip} \times \omega_f) + \beta_{lm}^L(\omega_{slip} \times v_{slip}) + \beta_{\mathrm{ln}}^L(\hat{e}_{Ln}) \tag{340}$$

Where,

- β_{ls}^L is the Saffman lift coefficient.
- β_{lm}^L is the Magnus lift coefficient.
- β_{ln}^L is the non-spherical lift coefficient (used for non-spherical models only).
- ω_f is the curl of local fluid velocity ($\nabla \times \vec{v_f}$).
- ω_{slip} is the curl of slip velocity ($\omega_{slip} = \frac{1}{2}\omega_f - \omega_p$).

- ω_p is the curl of particle velocity ($\nabla \times \overrightarrow{v_p}$).

There are three lift models available in AcuSolve:

1. Saffman-Magnus

   This model is for spherical particles and hence the orientation is neglected whereas the last two models take the particle orientation into account while calculating the lift forces.

   The correlation for the Saffman force is given by:

   $$\beta_{ls}^{L} = SLC \times C_{ls} d_p^2 \sqrt{\mu_f \rho_f}\, |\omega_f|^{-0.5} \tag{341}$$

   SLC is Saffman constant with a default value of 1.615. This value can be modified by you while specifying the model inputs. C_{ls} is given by the expression:

   $$C_{ls} = \begin{cases} e^{-0.1\mathrm{Re}_i} + 0.3314\ \sqrt{\gamma_i}\left(1 - e^{-0.1\mathrm{Re}_i}\right) & \mathrm{Re}_i \le 40 \\ 0.0524\sqrt{\gamma_i \mathrm{Re}_i} & \mathrm{Re}_i > 40 \end{cases} \tag{342}$$

   $$\gamma_i = \frac{|\omega_f| d_p}{2|\mathbf{v}_{slip}|} \tag{343}$$

   The correlation for the Magnus force is given by:

   $$\beta_{lm}^{L} = \frac{MLC \times C_{lm} |\mathbf{v}_{slip}|^2 \pi d_p^2 \rho_f}{|\omega_{slip}||v_{slip}|} \tag{344}$$

   $$\omega_{slip} = \frac{1}{2}\omega_f - \omega_p \tag{345}$$

   MLC is Magnus constant with a default value of 0.125. This value can be modified by you while specifying the model inputs. C_{lm} is given by the expression:

   $$C_{lm} = \begin{cases} d_p \dfrac{|\boldsymbol{\omega}_{slip}|}{|\mathbf{v}_{slip}|} & \mathrm{Re}_i \le 1 \\ d_p \dfrac{|\boldsymbol{\omega}_{slip}|}{|\mathbf{v}_{slip}|}\left(0.178 + 0.822\mathrm{Re}_i^{-0.522}\right) & \mathrm{Re}_i > 1 \end{cases} \tag{346}$$

   For the Saffman Magnus model, β_{ln}^{L} is always equal to zero.

2. Saffman-Magnus non-spherical lift

   The non-spherical version of the Saffman-Magnus lift model is similar to the spherical lift model except the non-spherical lift coefficient (β_{ln}^{L}) is not equal to zero and is calculated as shown in the next section.

3. Non-spherical lift

   When the non-spherical lift model is selected, the Saffman and Magnus lift coefficients are set to 0 and the non-spherical lift coefficient is assumed to be proportional to the drag coefficient and the correlation is given by:

$$\beta_{\text{ln}}^{L} = \frac{0.125 C_D (\sin^2 a \cdot \cos a) \pi d_p^2 |\mathbf{v}_{slip}|^2}{|x' \cdot \mathbf{v}_{slip}| |(x' \times \mathbf{v}_{slip}) \times \mathbf{v}_{slip}|} \tag{347}$$

Here the drag coefficient is obtained from the drag force calculation and a is the angle between the particle's principal axis and the slip velocity vector. The direction of the lift force is given by:

$$\hat{e}_{L_o} = (x'_i \cdot \mathbf{v}_{slip}) ((x' \times \mathbf{v}_{slip}) \times \mathbf{v}_{slip}) \tag{348}$$

where, x'_i is the particle principal axis and $\mathbf{v}_{slip}$ is the slip velocity vector.

Torque Models

By using the torque models, the rotational drag force on the rotating particles due to the inertia of fluid can be considered. When AcuSolve sends the force information to EDEM, this torque is added to the rotational motion equation in EDEM. The torque on a particle is given by,

$$T_i = \beta_{rot}^{T} (0.5\omega_f - \omega_p) + \beta_{pitch}^{T} \left(x'_i \times \left(f_i^d + f_i^l \right) \right) \tag{349}$$

The three types of torque models available in AcuSolve are pitching torque, rotational torque and a combination of both.

1. Pitching torque

   When the center of pressure of the force acting on a non-spherical particle does not coincide with the center of mass, it results in a hydrodynamic pitching torque, also known as offset torque, that acts around the axis perpendicular to the plane of relative fluid velocity and particle orientation vector. The pitching torque can change the angle of incidence of the particle.

   The expression used for calculating the pitching torque coefficient is given by:

   $$\beta_{pitch}^{T} = 0.25 \delta_{AR} \left(1 - (\sin a)^3 \right) \tag{350}$$

   $$\delta_{AR} = \begin{cases} 0.5 d_p (AR)^{2/3} & AR > 1 \\ 0.5 d_p (AR)^{-1/3} & AR \le 1 \end{cases} \tag{351}$$

   Where, AR is the aspect ratio of the particle and a is the angle between the particle's principal axis and the slip velocity vector. When the pitching torque model is selected, the rolling torque coefficient is set to 0 and vice versa.

2. Rotational torque

   A particle experiences rotational torque, also known as rolling friction torque, when there is a difference between the local fluid rotation and the angular velocity of the particle. The rotational torque is applied at the center of mass of the particle and the rotational torque coefficient is given by the expression:

   $$\beta_{rot}^{T} = \frac{\rho_f}{2} \left(\frac{d_p}{2} \right)^5 c_r |0.5\omega_f - \omega_p| \tag{352}$$

$$c_r = \begin{cases} 0 & \mathrm{Re}_r = 0 \\ \frac{64\pi}{\mathrm{Re}_r} & \mathrm{Re}_r \le 32 \\ \frac{12.9}{\mathrm{Re}_r^{0.5}} + \frac{128.4}{\mathrm{Re}_r} & 32 < \mathrm{Re}_r < 1000 \end{cases} \quad (353)$$

Where, Re_r is the Rotational Reynolds number of the particle and is given by,

$$\mathrm{Re}_r = \frac{\varepsilon_f \rho_f d_p^2 |0.5\omega_f - \omega_p|}{\mu_f} \quad (354)$$

3. Pitching rotational torque

   When the pitching rotational torque model is selected, both pitching torque and rotational torque are applied on the particle.

Heat Transfer Governing Equations

When heat transfer is active, in addition to the momentum equation, the energy conservation equations for the fluid and the particle are solved simultaneously to obtain the temperature of each phase. The governing equations for obtaining the temperatures of the particles and fluid are described below:

$$\frac{\partial}{\partial t}\left(\rho_f \varepsilon_f C_{pf} T_f\right) + \nabla.\left(v_f \rho_f \varepsilon_f C_{pf} T_f\right) = \nabla.\left(\varepsilon_f k_f \nabla T_f\right) + Q_p \quad (355)$$

Where Q_p represents the heat source term resulting from the heat transfer from the particle.

$$Q_p = \sum_{i=0}^{N_p} h_{fp} A_p (T_f - T_p) \quad (356)$$

A_p is the surface area of the particle, T_p is the particle temperature and T_f is the fluid temperature. The heat transfer coefficient (h_{fp}) is calculated using the empirical correlation given by $\mathrm{Nu}_p = \frac{h_{fp} d_p}{k_f}$ where the Nusselt number is calculated using the following expression:

$$\mathrm{Nu}_p = \begin{cases} 2 + 0.6\varepsilon_f^{3.5}\mathrm{Re}_i^{0.5}\mathrm{Pr}^{0.333333} & \mathrm{Re}_i < 200 \\ 2 + 0.5\varepsilon_f^{3.5}\mathrm{Re}_i^{0.5}\mathrm{Pr}^{0.333333} + 0.02\varepsilon_f^{3.5}\mathrm{Re}_i^{0.8}\mathrm{Pr}^{0.333333} & 200 \le \mathrm{Re}_i < 1500 \\ 2 + 0.000045\varepsilon_f^{3.5}\mathrm{Re}_i^{1.8} & \mathrm{Re} \ge 1500 \end{cases} \quad (357)$$

Where Pr is the Prandtl number given by $\mathrm{Pr} = \frac{\mu_f C_{pf}}{k_f}$.

The rate of change of the particle's temperature over time is given by,

$$m_p c_p \frac{dT}{dt} = \sum Q_{heat} \quad (358)$$

Where Q_{heat} is the sum of conductive and convective heat fluxes. The convective heat flux is calculated by AcuSolve based on the correlation described above and shared through the coupling interface once per each AcuSolve time step. The conductive heat transfer is due to particle-particle and particle-

geometry contacts and is calculated based on the relative temperatures and the particle overlap. The conductive heat flux between two particles $p1$ and $p2$ is given by,

$$Q_{p1p2} = \frac{4k_{p1}k_{p2}}{k_{p1}+k_{p2}}\left[\frac{3F_N r^*}{4E^*}\right]^{1/3} \tag{359}$$

Where k is the thermal conductivity, F_N is the normal force, r^* is the geometric mean of the particles' radii and E^* is the effective Young's modulus for the two particles.

Mass Transfer Model

The governing equations for the mass transfer in AcuSolve-EDEM coupling are described below:

$$\frac{\partial(\varepsilon_f \rho_f)}{\partial t} + \nabla\cdot(\rho_f \varepsilon_f v_f) = \dot{m} \tag{360}$$

Additionally, a species transport equation is also solved for tracking the transport of the vapor phase in the fluid domain.

$$\frac{\partial}{\partial t}\left(\rho_f \varepsilon_f \psi_v\right) + \nabla\cdot\left(v_f \rho_f \varepsilon_f \psi_v\right) = \nabla\cdot\left(D_v \nabla \psi_v\right) + \dot{m} \tag{361}$$

$$\frac{\partial}{\partial t}\left(\rho_f \varepsilon_f C_{pf} T_f\right) + \nabla\cdot\left(v_f \rho_f \varepsilon_f C_{pf} T_f\right) = \nabla\cdot\left(\varepsilon_f k_f \nabla T_f\right) + Q_p \tag{362}$$

For the mass transfer simulations, the heat source term on the right-hand side consists of both the convective term and a latent heat term due to evaporation of the liquid phase, that is, moisture content of particle. This is calculated as shown below:

$$\frac{dT_p}{dt} = f_2 \frac{Nu}{3\mathrm{Pr}}\left(\frac{\theta_1}{\tau_p}\right)(T_f - T_p) + \left(\frac{L_V}{C_{p,l}}\right)\frac{\dot{m}}{m} \tag{363}$$

Where the first term on the RHS represents the convective term and is scaled by an augmentation factor due to the evaporation of liquid and the second term represents the latent heat effects. θ_1 is the ratio of heat capacity of the carrier gas to that of the liquid phase of the evaporate.

$\dot{m}$ is the rate of evaporation of the liquid content and is given by,

$$\dot{m} = \frac{dm}{dt} = -\frac{Sh}{3Sc_f}\left(\frac{m}{\tau_p}\right)H_M \tag{364}$$

Where,

- Sh is the Sherwood number given by $Sh = 2 + 0.6\varepsilon_f^{3.5}\mathrm{Re}_i^{0.5}Sc_f^{0.333333}$.
- Sc_f is the Schmidt number defined as $Sc_f = \frac{\mu_f}{\rho_f D_f}$, D_f is mass diffusivity of the fluid.
- H_M is the evaporation potential which is analogous to the temperature difference for heat transfer.
- m is the liquid mass on each particle.

Stokes flow time constant of the particle, $\tau_p = \frac{\rho_l d_p^2}{18\mu_f}$

The evaporation potential (H_M), also known as the specific driving potential for mass transfer, is calculated as shown below:

$$H_M = \log(1 + B_M) \tag{365}$$

B_M is the Spalding transfer number for mass given by,

$$B_M = \frac{\psi_{s,eq} - \psi_f}{1 - \psi_{s,eq}} \tag{366}$$

$$\psi_{s,eq} = \frac{X_{s,neq}}{\left(X_{s,neq} + (1 - X_{s,neq})\left(\frac{W_{mol,f}}{W_{mol,v}}\right)\right)} \tag{367}$$

$$X_{s,neq} = X_{s,eq} - \left(\frac{2L_K}{d_p}\right)\beta_{evap} \tag{368}$$

$$\beta_{evap} = -\left(\frac{3\mathrm{Pr}\tau_p}{2}\right)\left(\frac{\dot{m}}{m}\right) \tag{369}$$

$$L_K = \frac{\mu_f\sqrt{2\pi T_p\left(\frac{\bar{R}}{W_{mol,l}}\right)}}{Sc_f p_{f,local}} \tag{370}$$

$$X_{s,eq} = \left(\frac{p_{atm}}{p_{f,local}}\right) e\left(\left(\frac{L_v W_{mol,v}}{\bar{R}}\right)\left(\frac{1}{T_{b,v}} - \frac{1}{T_p}\right)\right) \tag{371}$$

$$f_2 = \frac{\beta_{evap}}{(e^{\beta_{evap}} - 1)} \tag{372}$$

Where,

ψ is the mass fraction of vapor and χ is the mole fraction of vapor. The subscript *s* represents the vicinity of the particle, *f* represents the free stream, *eq* represents equilibrium and *neq* represents the non-equilibrium species fractions. $W_{mol,f}$ is the molecular weight of the carrier gas and $W_{mol,l}$ is the molecular weight of the liquid phase of the evaporate. L_v is the latent heat of vaporization of the liquid, $T_{b,v}$ is the liquid boiling temperature, T_p is the particle temperature. $\bar{R}$ is the universal gas constant, p_{atm} is the standard atmospheric pressure and $p_{f,local}$ is the local fluid pressure.

Topology Optimization

Introduction

Topology optimization for fluid flow in AcuSolve is based on the solution of the flow equations combined with the solution of associated linearized adjoint equations. The objective is to minimize the mechanical losses of the fluid flow system in combination with constraints limiting the used material. To represent the solid geometry a field variable is used to indicate the presence of material which adds a resistance modeled by a porous media force with a positive Darcy coefficient. Refer to Sandboge et al. (2021) for details.

This document shows the equations used to solve the topology optimization problem for flow.

Optimization

Minimize mechanical losses

The considered optimization problems take the form:

Minimize $M(u, \gamma)$ *(373)*

Subject to $D(u, \gamma) \geq 0$ *(374)*

$G_k(u, \gamma) = 0, \quad k = 1, \ldots, N$ *(375)*

$R(u, \gamma) = 0$ *(376)*

where $u = (u_1, u_2, u_3, p, \ldots)$ is the solution vector of flow variables u_1 for velocity vector component; p for pressure, and γ is the design topology field which determines the presence of solid material. The function $M(u, \gamma)$ is the objective, $D(u, \gamma)$ and $G_k(u, \gamma)$ are constraint functions. The flow equations are represented by $R(u, \gamma) = 0$ acting as an additional constraint.

Here, $M(u, \gamma)$ represents the mechanical losses as a single objective, integrating over the inlet and outlet surfaces.

$$M(u, \gamma) = -\int_{S_I} \left(p + \frac{1}{2}\rho u_k^2\right) u_i n_i \, dS - \int_{S_O} \left(p + \frac{1}{2}\rho u_k^2\right) u_i n_i \, dS \quad (377)$$

where ρ is the density, S is the surface.

The function $D(u, \gamma) = D(\gamma)$ is here the function of design fraction occupying the solid volume and is used as an optional constraint to limit the amount of solid material.

$$D(\gamma) = \frac{\int_\Omega \gamma \, d\Omega}{\int_\Omega d\Omega} \quad (378)$$

where Ω is the volume.

As an alternative, the constraint can also be given as an upper bound constraint $D(u, \gamma) \leq 0$.

Further, the functions $G_k(u, \gamma)$ are used to constrain mass flow rates at outlets # for the # outlets,

$$G_k(u, \gamma) = \frac{1}{2}(m_l - \bar{m}_l)^2 = \frac{1}{2}\left(\int_{SO_k} \rho u_j n_j \, dS - \bar{m}_k\right)^2 \quad (379)$$

where m_k is the mass flow rate at exit # and $\bar{m}_k$ is the target mass flow rate at the same exit.

The constraints are naturally optional. It is possible to have an optimization problem without any constraints, except for the flow constraint Equation 376 which is always active.

Minimizing the mechanical losses and fluid volume

If $M(u, \gamma)$ represents the mechanical losses in Equation 373, regions of zero velocity will not change the objective value. Thus, zero velocity "pockets" in the solid region can occur in the final solution. To avoid such unclean solid geometry, it is possible to define $M(u, \gamma)$ as a multi-objective adding the size of the fluid volume in the design domain to the mechanical losses. The strength of minimization of the fluid volume size is defined by a user-defined coefficient c_s multiplied by a normalization coefficient ε_n,

Minimize $$M(u, \gamma) + c_s \varepsilon_n D(\gamma) \quad (380)$$

Subject to $$R(u, \gamma) = 0 \quad (381)$$

Flow Equations

Computational domain

The flow equations are solved in a computational domain which is allowed to vary during the optimization process by modifying the material properties. At the first iteration, the design domain may use material properties representing a fluid, such as air, see Figure 45, and at convergence some parts of the domain have built up regions with material properties representing a solid material, see Figures Figure 46–Figure 47. The computational domain Ω is fixed, but the material properties are functions of a design field # as is explained in the continuous adjoint for topology optimization section.

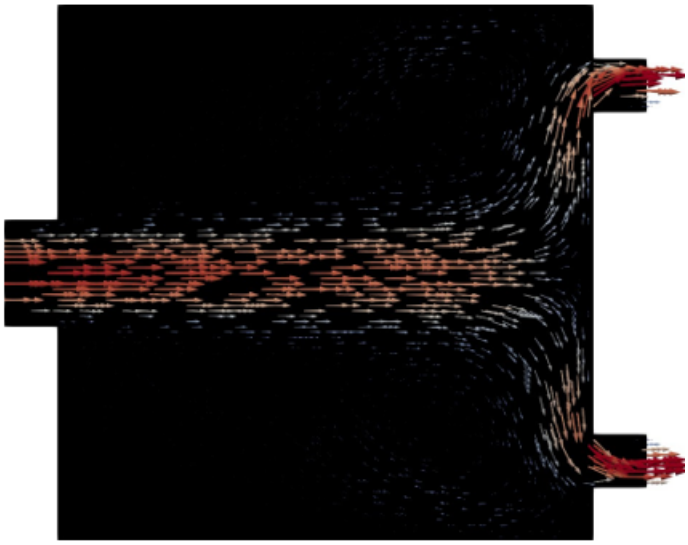


Figure 45: Design domain with all fluid material properties. A starting condition with sub-optimal mechanical losses.

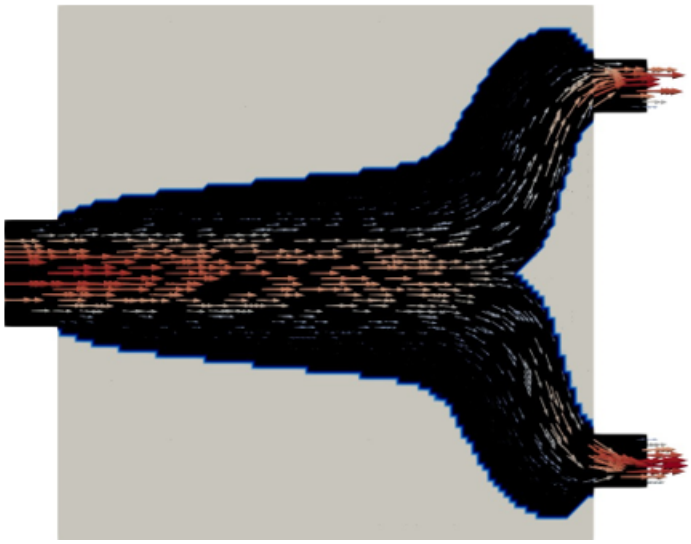


Figure 46: Topology solution where some parts of the design domain have material properties representing a solid material in order to minimize the mechanical losses of the system.

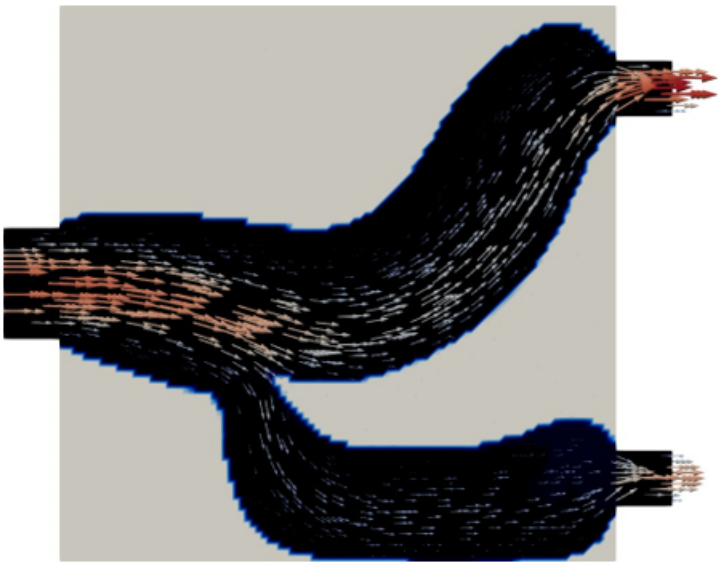


Figure 47: Topology solution minimizing the mechanical losses subject to unequal mass flow constraints at the outlets.

Incompressible Navier-Stokes equations

The state equation residuals of the incompressible equations are defined as, $R_{ui}=0$, $R_p=0$ where

$$R_{ui}=\rho\frac{\partial u_i}{\partial t}+\rho u_j\frac{\partial u_i}{\partial x_j}+\frac{\partial p}{\partial x_j}-\frac{\partial \tau_{ij}}{\partial x_j}-\rho\hat{b}_i \tag{382}$$

$$R_p=\frac{\partial u_i}{\partial x_i} \tag{383}$$

where p, u_i denote the static pressure and velocity components, respectively and # denotes density and $\hat{b}_i$ are the specific body force components. The stresses τ_{ij} are given by

$$\tau_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) \tag{384}$$

where μ is the viscosity.

Continuous Adjoint for Topology Optimization

The optimization problem (Equation 373) is solved using the adjoint methods. The continuous adjoint approach for topology optimization is presented in this section.

Momentum equations for topology optimization

The Darcy porosity model for the momentum flow equations is used; the residual is expressed as

$$R^p_{ui} = \rho\frac{\partial u_i}{\partial t} + \rho u_j\frac{\partial u_i}{\partial x_j} + \frac{\partial p}{\partial x_j} - \frac{\partial \tau_{ij}}{\partial x_j} - \rho\hat{b}_i + f_i = 0 \tag{385}$$

where f_i are the components of the porous media force vector

$$f_i = C_{\text{Darcy}}(\gamma)\,\mu\,u_i \tag{386}$$

and C_{Darcy} is the Darcy coefficient,

$$C_{\text{Darcy}}(\gamma) = C^f_{\text{Darcy}} + \gamma^p\left(C^s_{\text{Darcy}} - C^f_{\text{Darcy}}\right) \tag{387}$$

where the Solid Isotropic Material with Penalization (SIMP) defined by p, a constant whose value is selected $p = 1$ to $p = 3$. Thus, the design field γ determines the level of resistance in the design domain; zero resistance for fluid regions and a high resistance to minimize the flow through the solid regions.

> **Note:** For $\gamma = 0$ you have $C_{\text{Darcy}} = 0$ and Equation 385 becomes identical to Equation 382, and for $\gamma = 1$ you have $C_{\text{Darcy}} = C^s_{\text{Darcy}}$.

The value of C^s_{Darcy} is automatically computed by the optimization algorithm, aiming to minimize the flow through the solid region but at the same time limiting the size of the force f_i to avoid poor convergence.

Adjoint equations

The adjoint equations to the linearized incompressible Navier-Stokes equations are given by,

$$R_{u^a_i} = -\rho u_j\left(\frac{\partial u^a_i}{\partial x_j} + \frac{\partial u^a_j}{\partial x_i}\right) + \frac{\partial p^a}{\partial x_j} - \frac{\partial \tau^a_{ij}}{\partial x_j} + f^a_i + F_\Omega = 0 \tag{388}$$

$$R_{p^a} = \frac{\partial u^a_i}{\partial x_i} = 0 \tag{389}$$

where the adjoint variables u^a and p^a measures the sensitivities of varying the field γ around the fluid solution u and p of the forward problem.

Optimization Algorithm

Derivative of responses

The variation of the mechanical losses objective function (Equation 377) with respect to the design field variables is given by the expression

$$\tilde{F} = \int_{\Omega} \tilde{C}_{\text{Darcy}}(\gamma) \frac{\mu}{k_i} u_i u_i^a \, d\Omega \tag{390}$$

where u_i^a is the adjoint velocity vector. The coefficient $\tilde{C}_{\text{Darcy}}(\gamma)$ is the variation of the darcy factor, which is constant in the design domain for the SIMP-method of order 1. In essence, the derivative of the design material is determined from the scalar product between the velocity and the adjoint velocity vectors.

A volume constraint based on the topology design fraction in Equation 378 gives the variation

$$\tilde{D} = D'(\gamma) \frac{\tilde{\gamma}}{\int_{\Omega} d\Omega} = \frac{\int_{\Omega} \tilde{\gamma} \, d\Omega}{\int_{\Omega} d\Omega} \tag{391}$$

which gives a constant value in the design domain for the same volume variation $\tilde{\gamma}$.

The function $G_k(u, \gamma)$ given by Equation 379, used to account for the constraint on mass flow at outlet k, is specified as an equality constraint and actually treated as a double inequality constraint defined below as

$$\delta_k^{\text{Tol}} \leq m_k - \bar{m}_k \tag{392}$$

or

$$G_k(u, \gamma) \leq \frac{1}{2} \left(\delta_k^{\text{Tol}} \right)^2 \tag{393}$$

where δ_k^{Tol} is a user-defined tolerance value for the exit #. The variation of this function is equal to

$$\tilde{G}_k = (m_k - \bar{m}_k) \int_{S_{Ok}} \rho \tilde{u}_j n_j dS \tag{394}$$

where ### is the surface for outlet #.

Given the values of the objectives and constraints, and the field derivatives with respect to # for the objective, (Equation 390 for mechanical losses), and constraints, (constant for the design fraction constraint), the optimizer can update the design field #. The default optimizer is the methods of moving asymptotes (MMA). Refer to Svanberg (2007) for details of the MMA.

Metrics

The design variable # is continuous, but a well-defined geometry should have the value either # = 0 for a fluid or # = 1 for a solid with small regions of the design domain where 0 < # < 1. To measure the amount a well-defined solution of either fluid or solid, a quality index can be used,

$$I_Q = 1 - 4 \frac{\int_V \gamma (1 - \gamma) \, dV}{V} \tag{395}$$

where 0 ≤ ## ≤ 1, and ## = 1 for a well-defined geometry. Here, # is the volume of the design domain.

Illustration of the optimization algorithm

Equation 14 provides information on how the design material should be altered in the design space at each design iteration. To determine if solid material should be added or removed, it is possible to view the velocity # and adjoint velocity $\#^{\#}$ vectors in conjunction; if the angle is less than 90 degrees, material should be removed and if the angle is greater than 90 degrees material should be added.

Figure 48 shows (i) the initial velocity and adjoint velocity fields where the vectors have an angle greater than 90 degrees in some regions in front and rear of the obstacle; and (ii) the final velocity and adjoint velocity fields where all angles are less than 90 degrees in the entire fluid design domain.

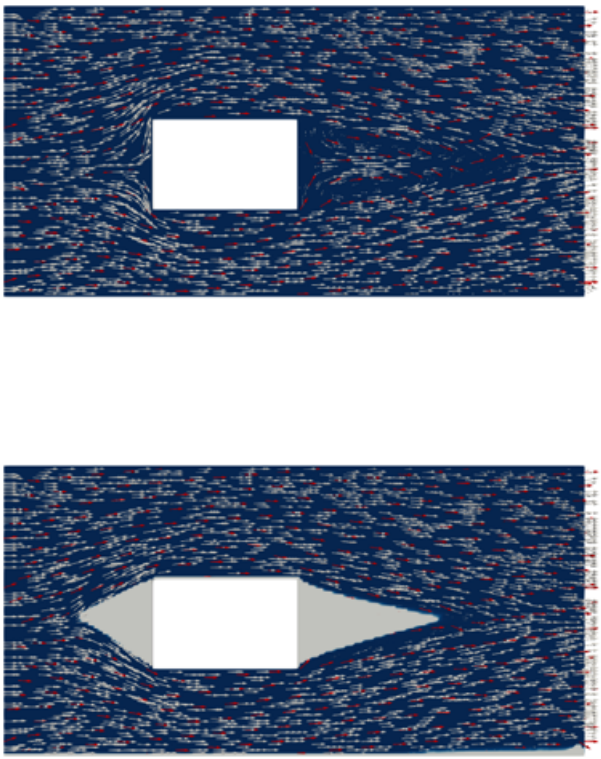


Figure 48: Flow past an obstacle. The velocity fields (white) and adjoint velocity fields (red) at the initial and final iterations: top for the initial field, bottom for the final field.

A close-up of the region behind the obstacle is shown in Figure 49, where the velocity shows some circulation while the adjoint velocity is attached without circulation.

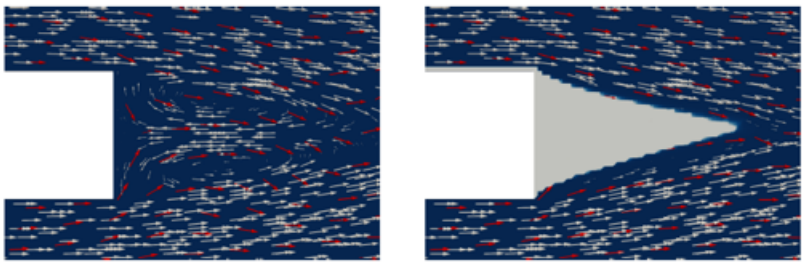


Figure 49: The velocity and adjoint velocity fields behind the obstacle.

Examples

Duct with sharp turn

As a simple example consider the duct domain (Figure 50), where the following optimization problem is solved,

Minimize $M(u,\gamma)$ *(396)*

Subject to $R(u,\gamma)=0$ *(397)*

where #(#,#) is the function of Equation 377, representing the mechanical losses of the fluid flow.

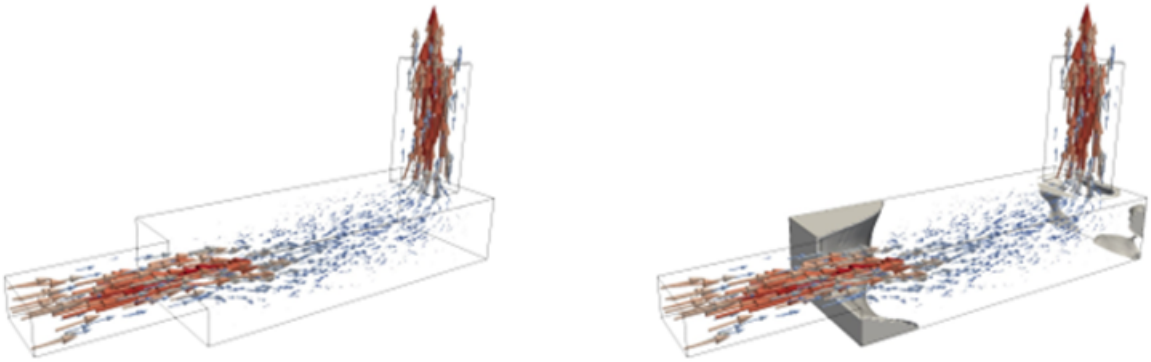


Figure 50: Flow optimization in a duct.
Left: shows the empty design domain; right shows the topology solution of Equation 396 where the solid geometry is shown in grey.

To solve the multi objective problem where you also factor in the size of the occupied fluid volume you have,

Minimize $M(u,\gamma)+c_s\varepsilon_n D(\gamma)$ *(398)*

Subject to $R(u,\gamma)=0$ *(399)*

By using a larger coefficient $\#_{\#}$, the duct cross-section will tend to be smaller due to a higher weight minimizing the occupied fluid volume ().

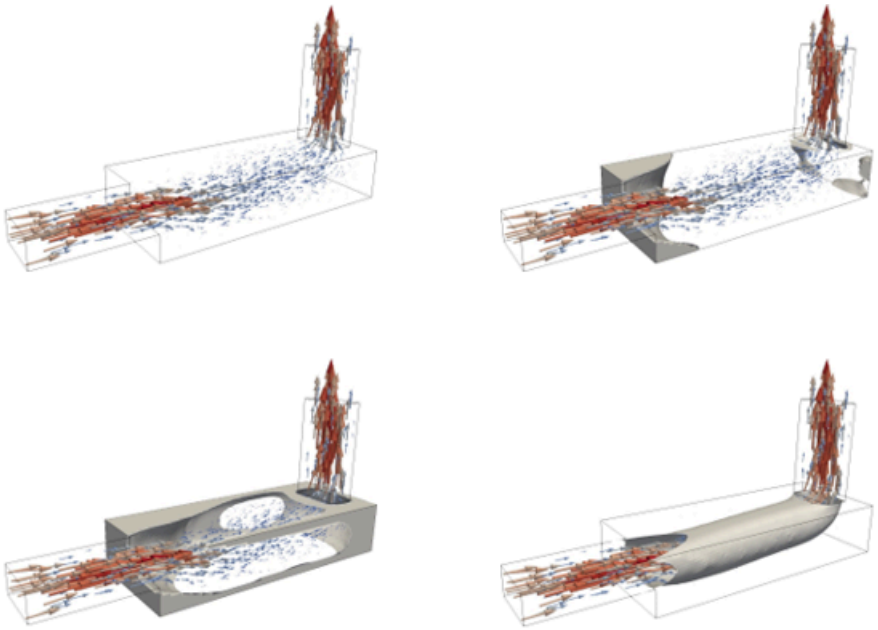


Figure 51: Flow optimization in a duct.
Using Equation 398. Left; shows the topology solution using $\#_{\#}$ =1. Right; shows the topology solution using $\#_{\#}$ =25, here illustrated by the surface between the fluid and solid regions.

Design-optimization of an HVAC Ducting System

The optimization technology is illustrated in this section in the design of the HVAC system of the Altair CX1 car. The cabin geometry of the Altair car is illustrated in Figure 52.

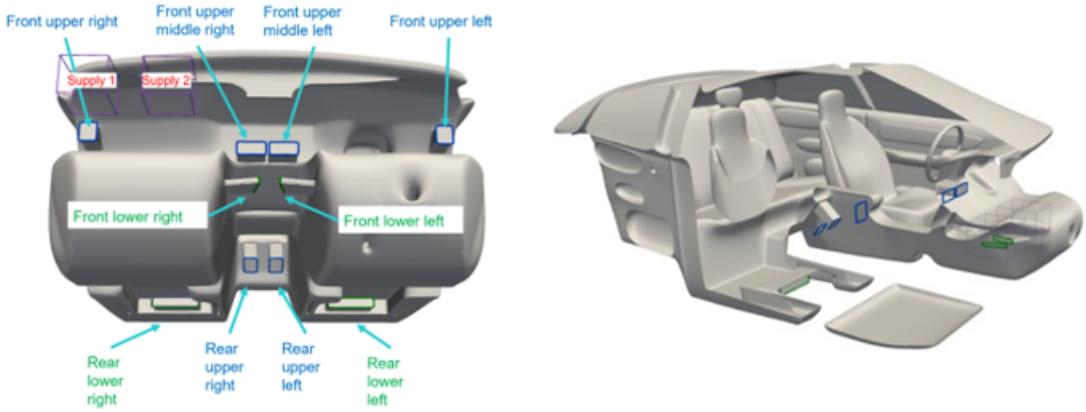


Figure 52: Inlets to the cabin geometry and air supply locations from the blower.

In this figure the ten different inlets to the cabin are shown. The six of them are aiming at the front and rear upper part of the cabin and the remaining four are providing the air to the front and rear lower part of the cabin. The supply of the air from the blower has also been separated into two parts. The first supply has to be connected with the upper inlets and the second supply has to be connected to the lower inlets to the cabin.

The first part of the algorithmic procedure is based on topology optimization for minimum power dissipation or mechanical energy losses to account for minimum fuel consumption and minimum volume to account for minimum manufacturing cost constrained by specific mass flow rates at each exit. In this study all mass flow rates selected as constraints were equal to 0.001kg/s and the tolerance of the constraints were selected equal to 5%.

Given the available space from the cabin and generally the car components, the first step is to define the initial box within which the duct must be confined. Apart from the availability in space the only other restriction to apply for topology optimization is to define the inlet to the box as the air supply location from the blower(s) and the outlet from the box as the inlets to the cabin. The CAD geometry can be in general automatically created and can be extracted from the available space. In this case a set of simple boxes were created to define the initial domain (Figure 53).

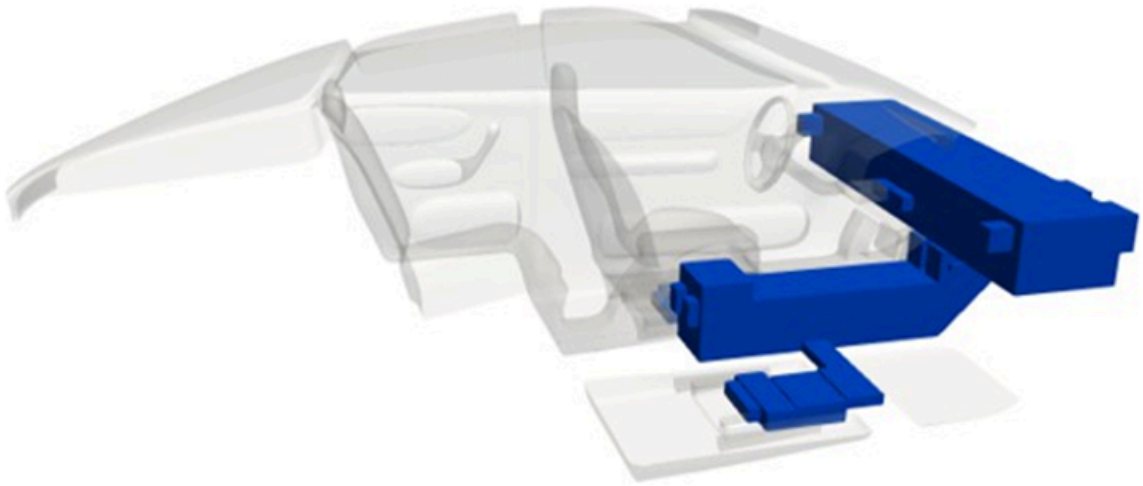


Figure 53: Initial CAD domain based on the available space from the cabin.

The next step is to generate the computational mesh at the initial domain. This mesh can consist of either hexaedral or tetrehedral elements. For this study, a hexahedral mesh with 2.5×10^5 nodes is used, Figure 54, left, which has sufficient resolution for topology optimization.

The topology optimization of the two ducts was performed in a sequential manner. First, the duct connecting the one inlet with the six upper outlets (cabin inlets) was optimized. For that reason, a part of the domain was excluded from the solution at this step to allow the evolution of the design of the first duct in a way that would not block the evolution of the other duct. This exclusion was made possible by applying nodal boundary conditions on the design variable at the excluded part of the domain and not through separate meshing.

After, the first optimal duct geometry is computed, the space it requires is excluded from the initial box, using again nodal boundary conditions on the design variable. To avoid intersections and very close proximity between the two duct geometries a user selected area that surrounds the first duct is also excluded. The second duct is optimized, and the optimal geometry uses the same computational mesh. Thus, a single initial mesh should only be created and no remeshing is required while performing the optimization of the separate ducts. This procedure can be continued to a next round of optimization of the two ducts; the first duct can be optimized again, leaving out not the initial random space but the space covered by the second duct (plus an additional layer), then the second duct can be optimized

again based on the new shape of the first duct and so forth. This approach is expected to result in designs with even better performance but was omitted in this study.

The final design of the two ducts using topology optimization is afterwards extracted and smoothed. The final shape is shown in Figure 54, right.

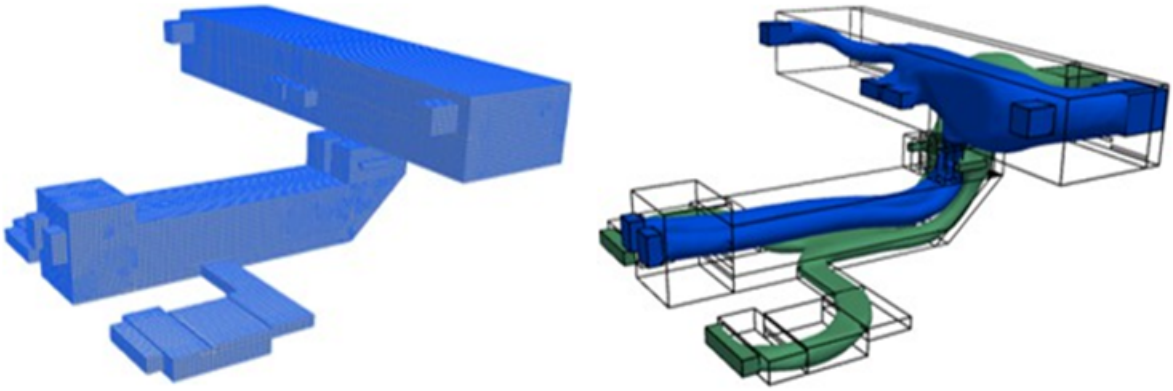


Figure 54: Computational mesh of the initial domain and optimal design of the HVAC ducts based on topology optimization.

A body fitted mesh with 6.5×10^5nodes is generated at this geometry in order to further optimize the shape using the shape optimization methodology with morph shapes. Shape optimization is used to further improve the value of the power dissipation objective function and ensure that the constraints on mass flow rates at the exits of the domain are as accurately as possible satisfied. It has been observed that the values of the mass flow rates at the exits are very sensitive to the geometrical design and mesh and, thus, the smoothing and remeshing of the optimized geometry affects them quite a lot. The body fitted mesh is also expected to give more accurate estimation of the quantities of interest and, thus shape optimization is expected to give the completely final design of the ducts. It should also be mentioned that the two ducts are handled as one during this step to avoid intersections and high proximity in the final design.

The morph shapes, required for the shape optimization described in the theory, are generated automatically using the flow field computed in the body fitted mesh. This way the two methodologies, topology and shape optimization described above are not disconnected but rather constitute the ingredients of a unified methodology. The flow velocity is the metric used to define optimally the morph shapes so that several types of morph shape can be formed as well as to ensure that the whole domain is appropriately space-filled and all the parts of the design are taken into consideration while performing the shape optimization step.

The automatically computed centers of the morph shapes are shown in Figure 55, left. In this case, the length scale of morph shape generator was equal to 0.2m which resulted in the generation of 22 control points in total. In Figure 55, right, the morph shape vectors are shown.

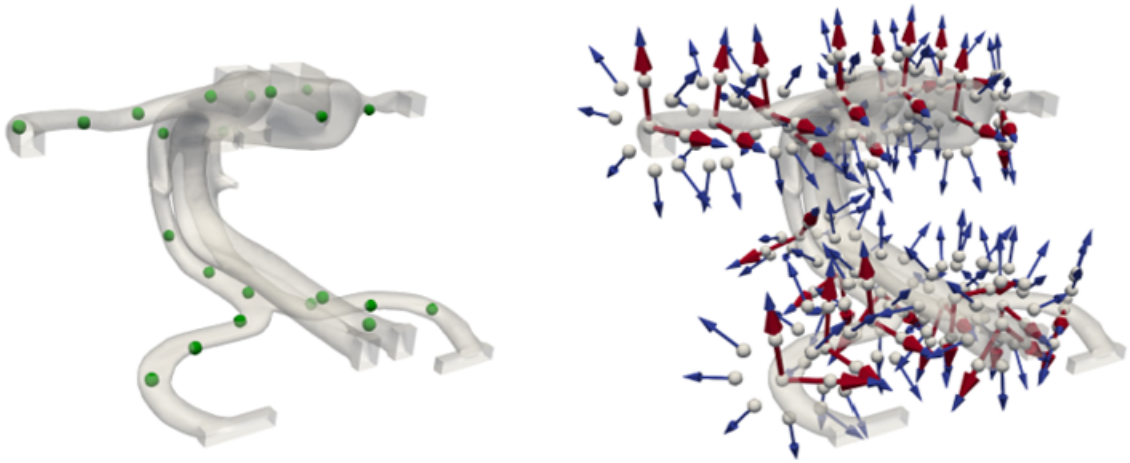


Figure 55: Bending and area morph shape centers and vectors.

Two different types of morph shapes are generated; the first one is depicted by the red vectors, and it allows for the bending of the duct during optimization. Two design variables per node are needed for this type of morph shape to cover all the design space. The second type of morph shapes is illustrated by the blue arrows and is used to increase or decrease the cross section of the duct geometry. One design variable per node is needed for this morph shape which includes four or eight control points with radial vectors that allow for the change in the cross section. The total number of morph shapes and thus the number of design variables for shape optimization is 66.

Based on the automatically computed morph shapes, shape optimization is performed further minimizing the power dissipation, with the same constraints on the mass flow rates at the exits and constraint on the volume of the design to be constant with a small tolerance. The initial and optimal geometries during this stage of optimization are shown in Figure 56. The distribution of the magnitude of the grid displacement that corresponds to the optimal geometry is also shown. The mass balance constraint at the exits is satisfied at this stage with a tolerance of 1%.

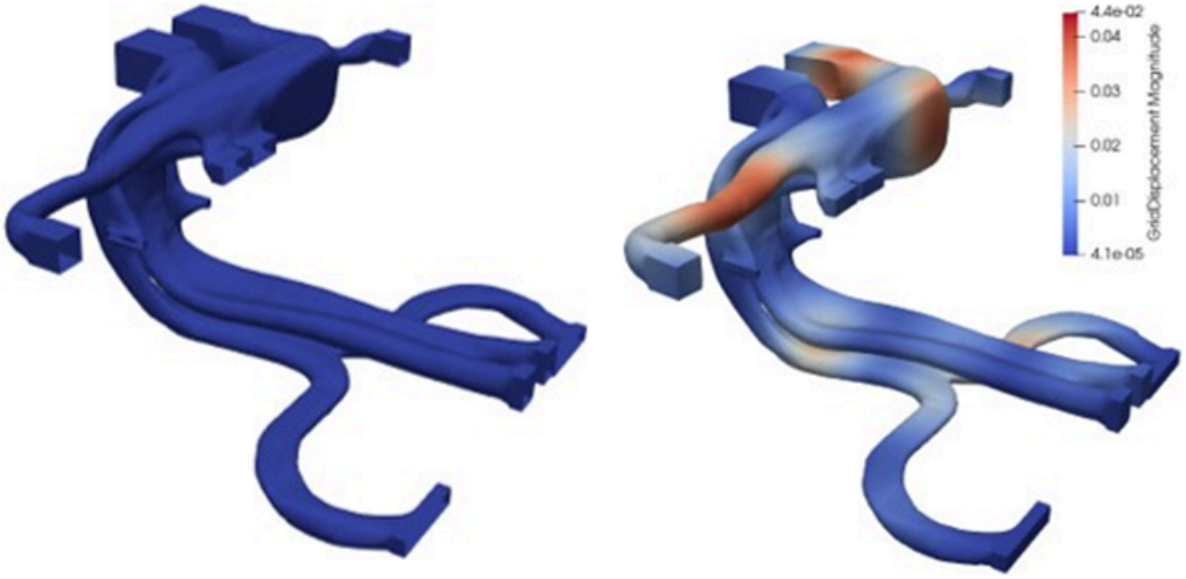


Figure 56: Shape optimization: initial and optimal shape

The optimal design of the HVAC with streamlines and velocity vectors is shown in Figure 57, together with the continuing velocity vectors inside the cabin.

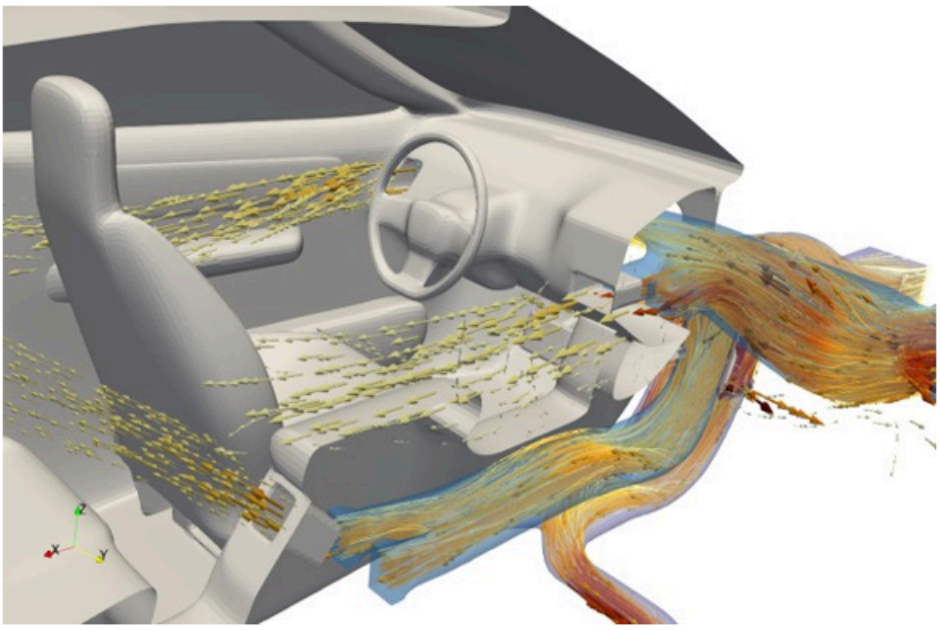


Figure 57: Optimal design

References

[1] Sandboge, R., Papadimitriou, D., and Reddy, M., "Shape and Topology Optimization in Computational Fluid Dynamics Including Heat Transfer Using Gaussian Processes and Adjoint Methods," 2021. https://doi.org/10.2514/6.2021-1892.

[2] Svanberg, K., "MMA and GCMMA-two methods for nonlinear optimization," vol, Vol. 1, 2007, pp. 1–15.

Solver Feature Guidelines

ALE (Arbitrary Lagrangian-Eulerian) CNF Parameters

The Arbitrary Lagrangian-Eulerian (ALE) method is a collection of technologies designed to simulate fluid flow with moving or deforming meshes, particularly useful for Fluid-Structure Interaction (FSI) problems. ALE involves key concepts such as:

- Modified Navier-Stokes equations with mesh velocity.
- Time integration methods that satisfy geometric conservation laws.
- Stable coupling between fluid forces acting on solids and the solid's response to these forces.
- Data exchange between fluid and solid meshes.
- Deformation of the fluid mesh.

In this context, mesh movement is the focus. Specifically, given boundary motion, either prescribed by you or determined by fluid forces, how to move the internal mesh nodes to maintain a valid mesh, avoiding negative-volume elements, sometimes called "inside-out" elements.

AcuSolve provides two mesh motion technologies:

Equation based mesh motion	This solves displacement equations for mesh nodes using methods such as hyper-elastic models and Mesh Quality Metric (MQM).
Interpolated Mesh Motion (MMI)	This approach skips solving equations, saving CPU time. Mesh node displacements are computed directly using simple algebraic interpolation.

In the input file (`.INP`), the command for mesh movement is:

```
EQUATION {
mesh = eulerian # arbitrary_lagrangian_eulerian, specified
}
```

- Eulerian means a stationary mesh.
- arbitrary_lagrangian_eulerian indicates equation-based mesh movement.
- Specified denotes mesh motion defined by algebraic relations (MMI) or user inputs, without solving additional equations.

ALE CNF Parameters

For fine control of equation-based ALE methods, the following configuration (CNF) parameters can be specified.

Mesh Poisson Ratio

parameter: -alenu <real>

default: 0.0

Relevant for hyperelastic models only. MQM does not use this parameter, although both models fall under ALE. This ratio affects the compressibility of the mesh material in hyperelastic models.

Mesh Reference Push Factor

parameter: -alefct <real>

default= 0

This parameter applies only to the Ogden. Ogden solves for the displacement of mesh nodes, which can be measured relative to the reference mesh in two ways:

- Relative to the initial mesh (when the parameter is set to 0).
- Relative to the mesh from the previous time step (when the parameter is set to 1).

When using option 2, the displacement magnitude is typically smaller as the mesh deforms progressively away from the initial configuration. This makes the equations easier to solve and allows for greater mesh deformation. However, because the reference mesh changes with each time step, the mesh cannot return to its original shape. This behavior is not ideal for cyclic movements, for example, pistons or valves, where restoring the original mesh configuration is important. However, for one-way motion, for example, a falling object or water collapse due to gravity, setting this parameter to 1 can help push the mesh deformation further. By default, the value is 0, meaning no reference push is applied.

Mesh Volume Absorption Factor

parameter: -alevaf <real>

default = 0

Applicable only to the Ogden hyperelastic model, this parameter introduces stiffness by having larger elements absorb more deformation. Typically set between 0.1 to 0.5, it helps preserve the shape of smaller elements and mimics some MQM effects.

MQM Reference Push Factor

parameter: -mqmpush <real>

default = 0

This controls the reference push factor for MQM-based mesh movement, influencing mesh stability during deformation.

Applying Mesh Parameters

To use mesh_volume_absorption_factor, add it to the `Acusim.cnf` file.

mesh_volume_absorption_factor =0.5

Enclosure Radiation Symmetry Plane

Symmetry planes may be modeled with *`num_symmetry_planes`*>0. This allows geometrical models that are a half, quarter, or eighth of the corresponding "ull"models. All planes must be mutually orthogonal. The radiation facets are reflected across each symmetry plane to create the full model. The view factors are computed on this model before being rescaled to the original model. A given surface facet may then

be able to see itself. Normally no `RADIATION_SURFACE` command is used on a surface lying in any of the symmetry planes, except if (partial) shielding is modeled. Specifying symmetry planes in this command only affects the calculation of view factors and has no effect on any other aspect of the simulation model.

Topology Optimization Guidelines

Pre-Processing, Solution Settings, and Post- Processing

The following steps are recommended to create and run a topology optimization problem in AcuSolve:

1. Create the model geometry in Parasolid format (other formats are also supported).
2. Import the Parasolid file in SimLab.
3. Create the mesh with tetrahedrons or hexahedrons. A boundary layer mesh is not required for the design space, but it may be used in other regions.
4. Create the flow setup with flow only, with or without turbulence modeling.
5. Create the topology optimization setup as required.
6. After setup the model can be run from optimization update.
7. Post-processing can be done using either direct `.log` file read or through a `.h3d` file format.

General Pre-Processing Guidelines

The following pre-processing guidelines are recommended before applying topology optimization.

- CAD preparation for the design space can be done using Inspire or another CAD tool.
- Simple and clean geometry is preferred. Complicated geometry can lead to large meshes and long run times.
- SimLab can be used to create simple geometries.
- The inlet and outlet regions are preferred to have non-design spaces or volumes. If only faces for inlet and outlet are exposed, it is preferred to extend them to obtain non-design regions. It is necessary to make sure that no back flow occurs in the outlet region.
- The design space is preferred to be at least two times larger than the intended connections between the inlet and the outlet regions.

Mesh Generation Guidelines

> **Note:** The following is a requirement for finite elements for topology optimization.
>
> - The finite elements must be tetrahedral or hexahedral with linear interpolation.

For the mesh setup the following guidelines are applicable.

- Boundary layer is not required for design space.
- At least minimum 6 or 8 nodes are recommended across the flow direction in the non-design space. For the design space the mesh resolution is preferable to be the same mesh size as the non-design space.
- For high Reynolds number denser mesh may be needed.

- It is good practice to evaluate the computed results to ensure the mesh resolution is adequate.

2D topology optimization is possible. SimLab will specify periodic conditions for all variables in the perpendicular direction. The same guidelines apply to 2D models as for 3D.

> **Note:** A 2D model in AcuSolve is always modeled using a 3D mesh, with a one element thick layer in the cross-flow direction. Here, the restriction to a hexahedron or tetrahedral mesh applies (up to version 2023.1), however future releases will support wedge elements.

Solution Settings

After pre-processing follow the solver setting rules given below.

- Auto Solution Strategy
  - Max cases are recommended to be 1,000 and can be higher for hard to converge problems.
  - Initial case time step should be at least 50 and may need to be increased to get better convergence from the initial conditions for the empty domain. During this cycle no topology optimization calculations are taking place.
  - The default relaxation factor is set to 0.3. Other values may be used as appropriate based on the flow solution requirements.
- Boundary conditions
  - Simple boundary and nodal boundary conditions for mass flow rate (volume flow rate) and velocity are supported.
  - Topology optimization does not support the pressure inlet boundary condition.
- Turbulence related Settings
  - Spalart-Allmaras is the only supported turbulence model.
  - For low Reynolds with steady flow, turbulence model can be turned off.
  - For moderate Reynolds numbers flows (unsteady flow or transition between laminar to Turbulent), the Spalart-Allmaras turbulence model is recommended.
  - For high Reynolds number flows, the Spalart-Allmaras turbulence model is used.

The topology optimization is a steady state feature. If there is transient behavior during the simulation, the flow rate can be reduced to get a steady flow in an initial run. A second run can then be done, using the target flow rate, based on the results from the first run, where the design is used as initial condition. The nodal initial condition files may be created using the acuProj command. It is also preferred to use an updated darcy factor value for the second run, in case automatic darcy value was used in the simulation (set by *`topology_auto_darcy`* = on in the `OPTIMIZATION` command, see also below). The updated darcy factor values are written to the `.log` file.

Objectives and Constraints

Topology optimization requires the specifications of objectives and constraints that are related to responses computed by the solution algorithm. Specific topology optimization responses are needed to define the objectives and constraints used by the topology optimization solution procedure, but other general responses may be defined to obtain other output of interest for post-processing.

- Topology optimization requires one main objective and optionally one or more constraints.

- For flow topology optimization the most common practice (the only option for objective in SimLab version 2022.3 and version 2023.1) is to minimize mechanical energy. This corresponds to minimizing the pressure drop for an incompressible fluid with inlet and outlet where mass flow rates are pre-defined.
- The additional option to minimize the occupied flow volume in the design domain can be included using *objective_min_flow_volume* = ON. The strength of this parameter can be controlled using *objective_min_flow_volume_mult*. The default value is 1, but the practical range for most problems is between 0.1 and 100. Typically, smaller values give smaller mechanical energy losses, while a larger number gives a smaller channel width.
- Instead of using the objective min flow option, design volume constraints can be used. This can be set using lower or upper bounds for the design volume fraction of the solid material. Thus, using a lower bound of 0.8 means that the solid material will occupy at least 80% of the design space.
- For problems with multiple outlets, it is possible to set mass flow constraints at the outlets. The target values can be set for each outlet and should total the inlet mass flow rate. The tolerance is generally set to 5-10% of the target value. This option may incur significant computation time (proportional to number of outlets). It may be best to make an initial run without mass balance and turn on the mass balance constraints in a second run using the solution from the first run as initial conditions. The two-stage approach has the additional advantage that the verification of the model setup is efficient.

The following solution settings are applicable.

- Automatic computation of the Darcy coefficient for the porosity model (representing solid material in the design domain) has been enabled as default.
- Initial Darcy can be larger (up to 1e6 to 1e8) for higher Reynolds number flows (beyond 10000 or higher) to speed up the convergence of the solution.
- Forchheimer coefficient is set to 0 for the design domain porosity model.
- Additional flow porosity is not recommended to be used (other than the porosity model for the design space).
- As part of the nodal output, the case output frequency can be adjusted as required. Currently, the default value is set to 20 in SimLab (*case_output_frequency* = 20 in the input file), but it can be changed under the optimization solver settings. Lowering the output frequency to below 20 can increase the usage of large disk space significantly. The parameter *output_initial_case* = ON specifies that the initial case should be output even if the case output frequency is greater than 1. Also make sure that *num_saved_states* = 0 to keep all results that are output. A value >0 can be used to limit the storage on the disk.

Post-Processing

The following post-processing guidelines are suggested.

- Direct read of AcuSolve `.log` file is available for post- processing within SimLab.
- Multiple types of outputs like h3d output, cgns and EnSight reader output can be used. These outputs are available from the acuTrans utility from the AcuSolve command line.
- Iso surface of level set at a value of 0 is recommended to get smoother output of the optimized shape.
- Iso surface of design volume at 0.5 can also be done to get the final optimized shape.

- The final optimized shape can be obtained using the deformed model under export results in stl format.
- Stl is imported into a new SimLab window to visualize the optimized design.
- Smoothing in SimLab is recommended if the surface appears uneven.
- Wrapping should be used only when necessary (when level set produces unclean geometry due to a coarse mesh).
- Use body fitted mesh (with boundary layers) when needed to obtain more accurate mechanical losses calculations compared to the values computed in the topology optimization solution.
- The response for mechanical losses (and other responses) may be used also for the run with a body fitted mesh making it easy to verify the results. The command is specified in the input file similar to the topology optimization run:

```
RESPONSE_VARIABLE( "mechanical energy" ) {
   type = topology_variable
   topology_variable = mechanical_energy
}
```

- AcuGetRsp can be used from the AcuSolve command line to get more detailed post-processing data for mechanical energy losses and other response output data. The first column gives the case identification number and the order of the remaining columns follows the order of the response commands in the AcuSolve input file.

Thermal-Electric Battery Simulation Setup and Guidelines

The battery solution is set up with a hierarchical connectivity structure of packs, modules and cells. The core unit is the battery cell, which will be characterized by its specific properties, for example, cathode chemistry and shape. These cell units are connected in series and parallel to form a module, that is, a group of cells, with a desired power output. Finally, the modules are assembled into a pack, with modules connected in series and parallel. These packs are where the overall electrical input/output is provided to the solution.

This section details the setup of the battery thermal electric simulation, including boundary conditions and typical electrical output variables. The sections are divided into:

- Battery Packs
- Battery Module
- Battery Model
- Electrical Output Variables
- Electrical Boundary Conditions
- Cell Electrical Connectivity, Cell Numbering, and Battery Components

Battery Packs

Battery packs provide two general inputs to a thermo-electric solution:

- How the modules in a pack are arranged in serial and parallel.
- The type of electrical input to the pack, for example, current, voltage, power or charging profile.

SimLab interface: The battery pack panel,shown in Figure 58, provides the module arrangement as well as the electrical input to the pack.

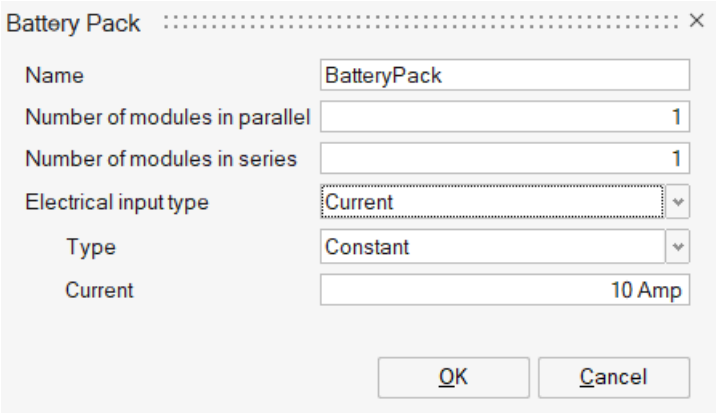


Figure 58: SimLab Battery Pack Panel

Module arrangement

The arrangement of the modules in serial and parallel determines how current will distribute through the pack. These are given nSmP arrangement, where n is the number of modules in series and m is the number of modules in parallel. As an example, a pack built from four serially connected modules, for example, 4S1P, is shown in Figure 59. An alternative pack with two parallel banks of four serially connected modules is shown in Figure 60.

Note: For consistency with the labeling at the module level, the module numbering is parallel first.

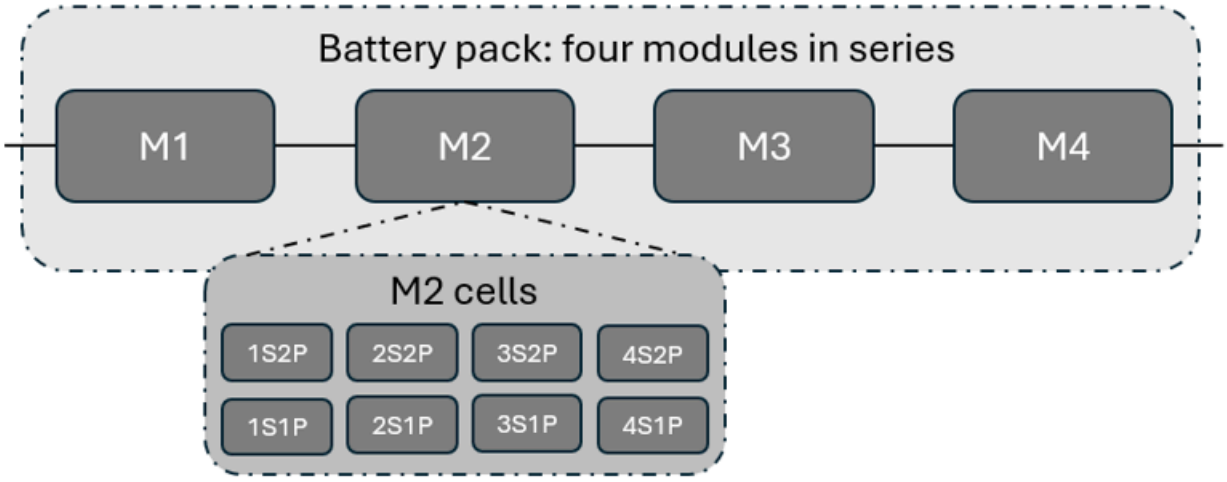


Figure 59: Battery pack of four modules connected in series. The insert ("M2 cells") shows the cells in an individual module.

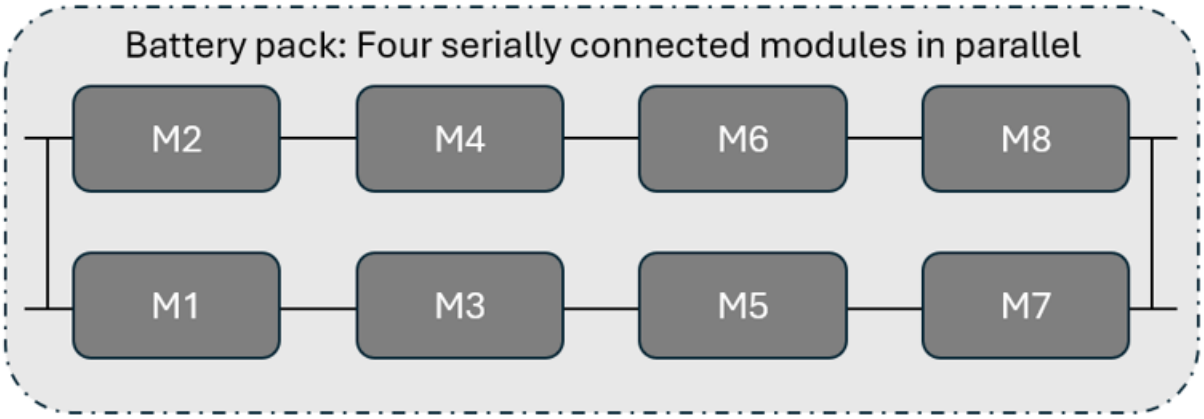


Figure 60: Battery pack of two parallel banks of four modules connected in series. The numbering is in the parallel direction first.

Electrical input type

Different types of electrical input can be included as an input to the battery pack using the `electrical_input_type` command. Although all inputs are done at a pack level these are then distributed to the cell level to ensure consistency of the electrical solution. These are summarized in the following sections.

Current

In general, when the electrical input is provided at the pack level this is required to be distributed to the cell level based on electrical connectivity as well as individual cell properties.

Overview of cell current calculation

If the pack current output I^k is provided as an input, this is required to be distributed to the cell level, since the cells are the source of the voltage that drives the current. To facilitate this distribution in a pack, consistency between the charge conservation solution and ECM models must be maintained. For a 2^{nd} order ECM model, the solution update for the polarization resistor current is given by,

$$I_{R_1}^{k+1} = \exp\left(\frac{-\Delta t}{R_1 C_1}\right) I_{R_1}^{k} + \left(1 - \exp\left(\frac{-\Delta t}{R_1 C_1}\right)\right) I_{\phi} \tag{400}$$

$$I_{R_2}^{k+1} = \exp\left(\frac{-\Delta t}{R_2 C_2}\right) I_{R_2}^{k} + \left(1 - \exp\left(\frac{-\Delta t}{R_2 C_2}\right)\right) I_{\phi} \tag{401}$$

Where I_ϕ is the current flux on the negative terminal of the battery (from the charge conservation PDE solution), R_1 and R_2 are the polarization resistance of the 1^{st} and 2^{nd} order RC pairs respectively, C_1 and C_2 are the polarization capacitance of the 1^{st} and 2^{nd} order RC pairs respectively, $I_{R_1}^k$, $I_{R_2}^k$,$I_{R_1}^{k+1}$, $I_{R_2}^{k+1}$ are diffusion resistor current at time step k and k+1, and Δt is the time step size.

For example, for a 2s2p pack the electrical circuit diagram would be as shown in Figure 59.

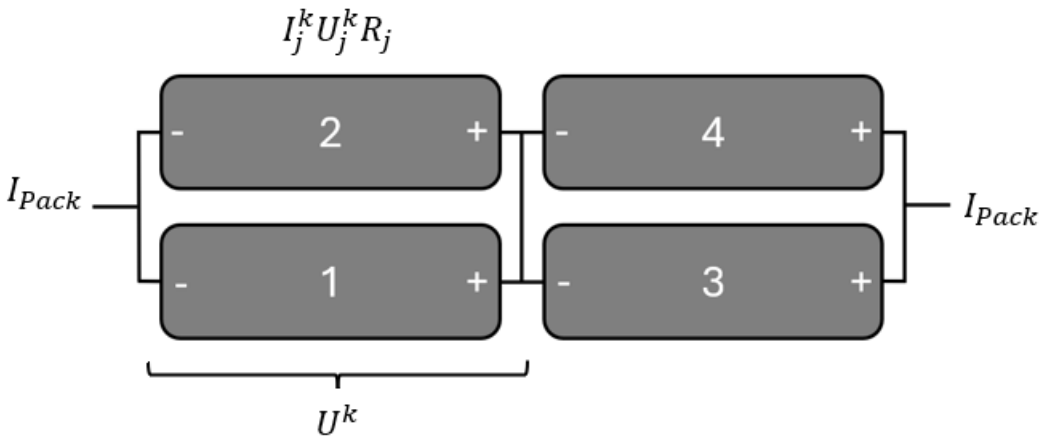


Figure 61: Battery pack cell current calculation information

Where I_j^k U_j^k, and R_j are the current, fixed voltage (for example, for second order: $U_j^k = V_{ocv} - V_1 - V_2$) and ohmic resistance in cell j, respectively. To determine the current, I_j^k the voltage, U^k, across a bank of parallel connected cells (PCU), can be determined based on the charging/discharging current (I^K) and the cell electrical properties, for example,

$$U^k = \frac{\sum_j \frac{U_j^k}{R_j} - I^k}{\sum_j \frac{1}{R_j}} \tag{402}$$

$$I_j^k = \frac{U_j^k - U_k}{\sum_j R_{s,j}} \tag{403}$$

The updated cell current, I_j^k, is then used as a boundary flux in the solution of the charge conservation equation for electrically conducting components connected to the negative terminal of the battery.

The battery solution supports both generic charging and discharging, including constant; piecewise linear; and cubic spline. Constant current discharge would be used for typical testing scenarios. Piecewise linear and cubic spline are used for experimental data such as a driving cycle, for example, NEDC or US06.

The current input, as well as all other electrical input types, are additionally subject to operating constraints defined in the `BATTERY_MODULE` command (see Battery Module). The constraints that control termination of the simulation are:

- maximum_cell_voltage
- minimum_cell_voltage
- minimum_cell_state_of_charge
- maximum_cell_state_of_charge

These are typically known from a battery specification sheet.

AcuSolve input

The following gives an example input for a current profile in the `BATTERY_PACK` command of the AcuSolve input file:

```
BATTERY_PACK( "4S2P battery module" ) {
    number_of_parallel_modules = 2
    number_of_serial_modules = 4
    electrical_input_type = current
    current_type = piecewise_linear
    current_curve_fit_values = Read( "time_current.txt" )
    current_curve_fit_variable = time
}
```

And `time_current.txt` is a two-column text file of time and current:

```
0               0.66809696
9.61104062      0.66809696
12.5842409      0.66809696
13.34691228     15.37136268
…
```

Figure 60 shows a typical NEDC driving profile with current input.

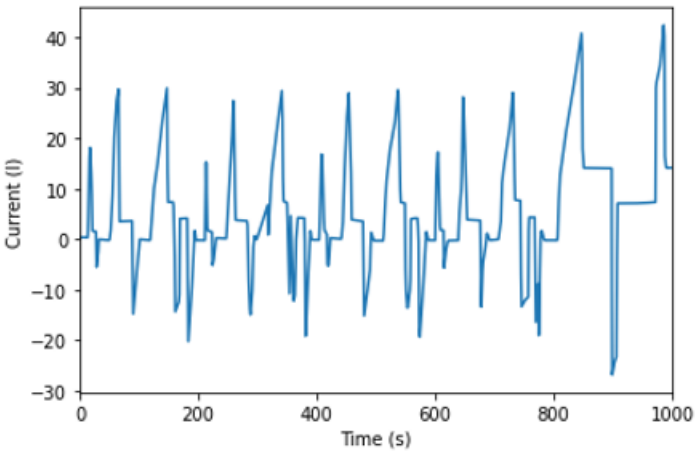


Figure 62: Current input for a NEDC driving cycle

Figure 61 shows a typical cell voltage and soc response form this current input applied to a battery pack.

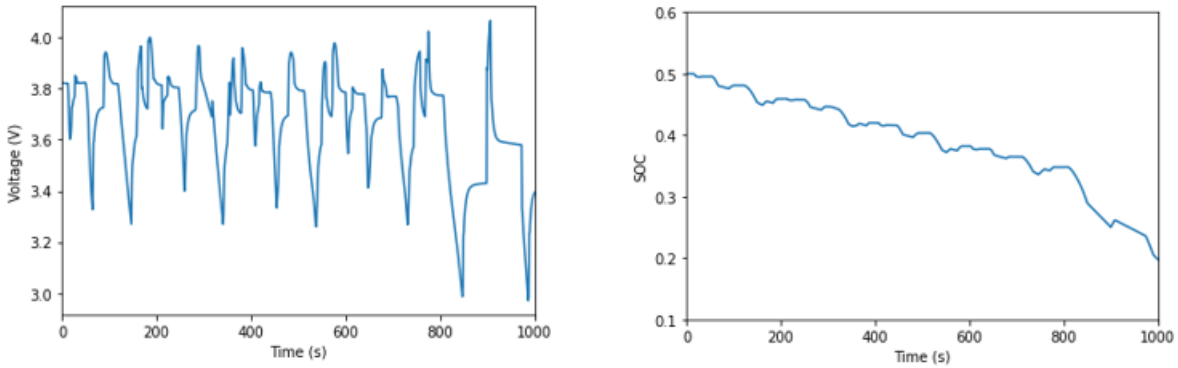


Figure 63: Voltage and soc output for a US06 driving profile

C-rate

Supports both generic charging and discharging, these include: constant; piecewise_linear; cubic_spline; and c-rate profiles. As with current, constant c-rate discharge would be used for typical testing scenarios. Piecewise linear and cubic spline are used for experimental data such as a driving cycle.

C-rate is the speed at which a battery is charged/discharged. For example, if you have a battery with a capacity of 3.5Ah, then you can calculate how many amps it can provide for different charging or discharging rates, some examples include:

- A battery with a capacity of 3.5Ah discharged at a c rate of 0.5C delivers 1.75A for 2 hours.
- A battery with a capacity of 3.5Ah discharged at a c rate of 1C delivers 3.5A for 1 hour.
- A battery with a capacity of 3.5Ah discharged at a c rate of 2C delivers 7A for 30 mins.

AcuSolve input

As an example, a 1C discharge would be defined in the AcuSolve `BATTERY_PACK` command by:

```
BATTERY_PACK( "4s2p battery module" ) {
    number_of_parallel_modules = 2
    number_of_serial_moudles = 4
    electrical_input_type = c_rate
    c_rate = 1
}
```

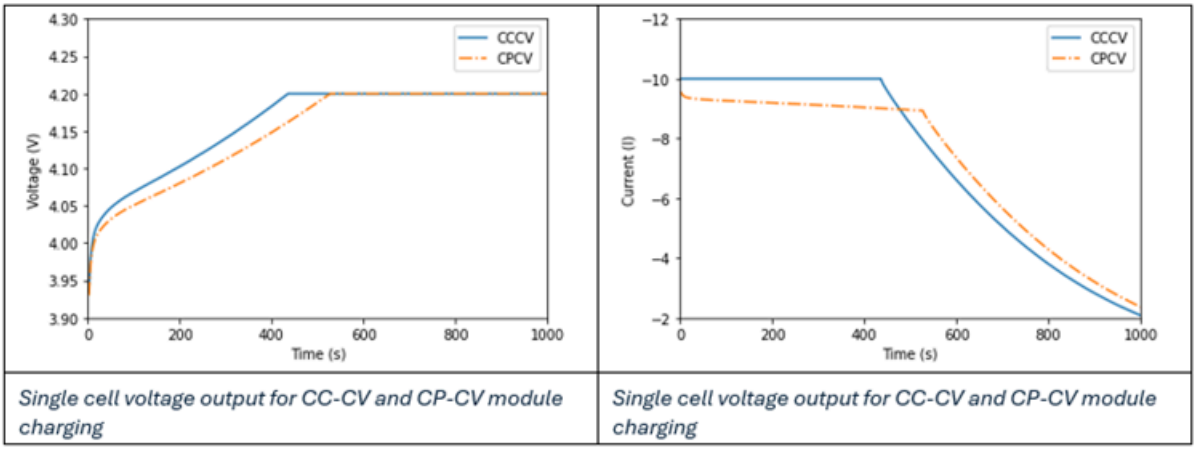
Charging profiles

Charging profiles include CC-CV (Constant Current-Constant Voltage) or CP-CV (Constant Power-Constant Voltage) methods.

In CC-CV charging, the process begins with a constant current (or c-rate) until reaching a specified voltage limit, and then switches to constant voltage charging. The simulation terminates when the current drops to a pre-defined level, for example, cut off current or maximum state of charge. In AcuSolve this is given as a fraction of the original input current. The switch between constant current and constant voltage is defined by the maximum voltage provided on the battery specification data sheet. This is set in the `BATTERY_MODULE` command (see Battery Module).

In CP-CV charging begins by applying a constant power, shifting to constant voltage once the voltage limit is reached. The termination criteria for CP-CV charging are the same as those for CC-CV.

A typical CC-CV or CP-CV charging current, voltage and soc curve are shown in Figure 62 and Figure 63.



Single cell voltage output for CC-CV and CP-CV module charging

Single cell voltage output for CC-CV and CP-CV module charging

Figure 64: Single cell voltage and current responses for CC-CV and CP-CV charging

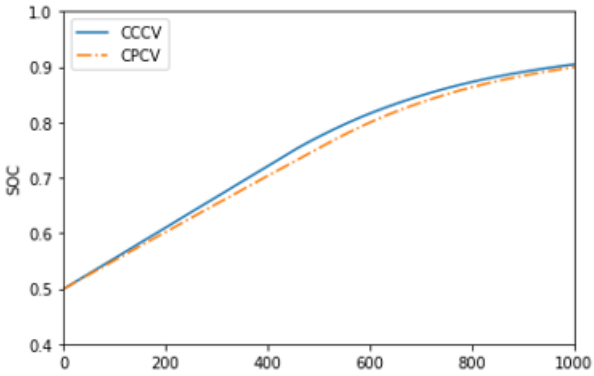


Figure 65: Single cell SOC response for CC-CV and CP-CV module charging

Sign convention

For charging profiles the current is always negative, since the sign convention in AcuSolve discharge current is positive.

SimLab input

An example of a typical CC-CV input is shown in Figure 64.

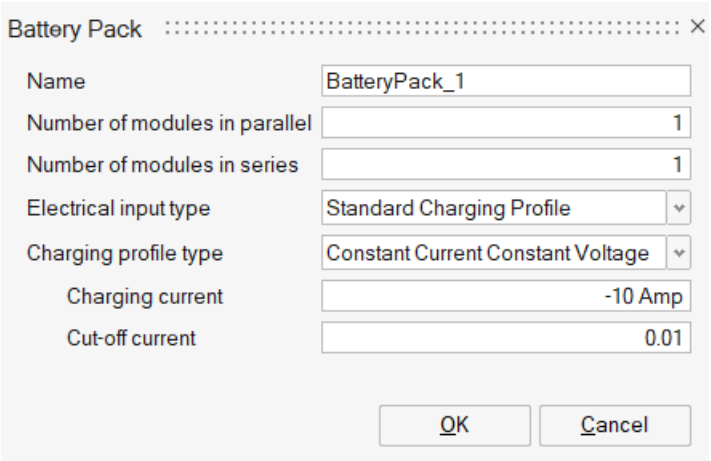


Figure 66: CC-CV input panel

AcuSolve input format

The AcuSolve input file for a CC-CV input requires the following parameters to be defined:

- *electrical_input_type* = standard_charging_profile
- *standard_charging_profile_type* = constant_current_constant_voltage
- *current*/*c_rate*

The following parameter can typically be taken to be default:

- *cut_off_current_percent*

```
BATTERY_PACK( "CCCV_pack" ) {
    number_of_parallel_modules = 1
    number_of_serial_modules = 4
    electrical_input_type = standard_charging_profile
    standard_charging_profile_type = constant_current_constant_voltage
    current_type = constant
    current = -20
    cut_off_current_percent = 0.01
}
```

The AcuSolve input file for a CP-CV input requires the following parameters to be defined:

- *electrical_input_type* = standard_charging_profile
- *standard_charging_profile_type* = constant_power_constant_voltage
- *power*

The following parameter can typically be taken to be default:

- *cut_off_current_percent*

```
BATTERY_PACK( "CPCV_pack" ) {
    number_of_parallel_modules = 2
    number_of_serial_modules = 4
    electrical_input_type = standard_charging_profile
    standard_charging_profile_type = constant_power_constant_voltage
    power_type = constant
    power = 300
    cut_off_current_percent = 0.01
```

```
}
```

Power

Power input can be provided as a constant power or a profile (piecewise linear and cubic spline). In this approach the power in the pack is calculated.

$$P(t) = U_T(t) i(t) \quad (404)$$

where $U_T(t)$ is the pack voltage (determined from the cell and module connectivity) and $i(t)$ is the pack current at time t. Using the above equation, it is possible to determine the cell current using an iterative approach based on the input power and known open circuit and polarization voltages.

AcuSolve input format

For a power input, the following parameters need to be defined in the AcuSolve input file:

- *electrical_input_type* = power
- *power_type* = constant/piecewise_linear/cubic_spline
- *power*

The following parameter can typically be taken to be default:

- *cut_off_current_percent*

```
BATTERY_PACK( "Power_pack" ) {
    number_of_parallel_modules = 7
    number_of_serial_moudles = 12
    electrical_input_type = power
    power_type = constant
    power = 1500
}
```

Constant Voltage

Constant voltage (CV) is charging with constant voltage. The input module voltage divided by the number of parallel cells should be between the minimum and maximum cell voltage.

AcuSolve input format

For a CV input, the following parameters need to be defined in the `BATTERY_PACK` command:

- *electrical_input_type* = voltage
- *voltage_type* = constant
- *voltage*

```
BATTERY_PACK( "Power_pack" ) {
    number_of_parallel_modules = 7
    number_of_serial_modules = 12
    electrical_input_type = voltage
    voltage_type = constant
    voltage = 16
}
```

Battery Module

The battery module is a hierarchical unit in the battery pack that is made up of cells arranged in serial and parallel. The command provides an avenue to identify electrical bodies that belong to the module as

well as the data sheet operating limits of the cell. The operating limits supported in the battery solutions are the maximum and minimum individual cell voltage as well as the SOC range, that is, maximum and minimum soc. A summary of the module inputs is given in the following.

Number of cells in parallel	Number of cells connected in parallel in a battery module for each series unit.
Number of cells in series	Number of parallel units connected in series.
Minimum cell voltage	Minimum allowed voltage of an individual battery cell.
Maximum cell voltage	Maximum allowed voltage of an individual battery cell. This avoids overcharging.
Minimum cell state of charge	Minimum state of charge at which a battery cell is allowed to operate.
Maximum cell state of charge	Maximum state of charge at which a battery cell is allowed to operate.
Parallel connected units	The number of parallel connected units (PCU) represents how cells are wired or connected in the module. This parameter only needs to be set when the state of charge differs between cells. More details on this parameter are given below and in the section Cell Electrical Connectivity, Cell Numbering, and Battery Components.

SimLab input

An example of the battery module input in SimLab is shown in Figure 65.

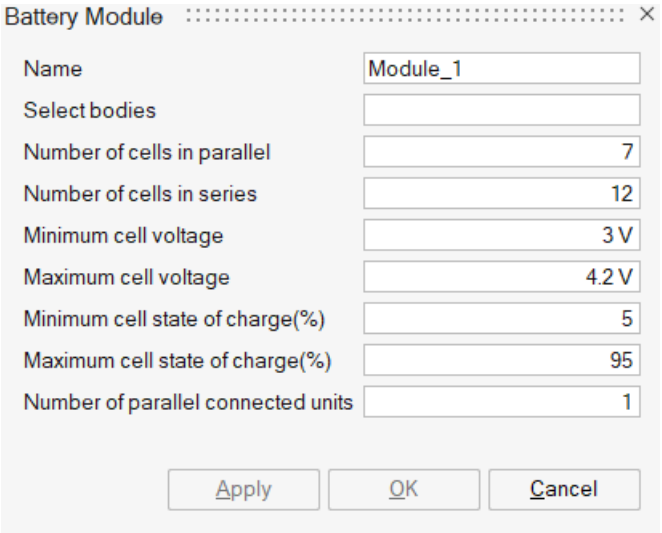


Figure 67: Battery Module input in SimLab

Select Bodies

The identification of the electrical bodies, for example, cells, allows for identification of a cell in a pack based on the module number and cell numbering. For example, if the label is M2S1P4, this means that

the cell in the pack belongs to module 2 and is the 4th cell in the first parallel bank connected in series. These bodies are added under a separate sub-assembly with a user-given name.

AcuSolve input format

For the battery module input, the following parameters need to be defined in the `BATTERY_MODULE` command:

- *number_of_parallel_cells* = 7
- *number_of_serial_cells* = 12
- *maximum_cell_voltage* = 4.2
- *minimum_cell_voltage* = 3
- *maximum_cell_state_of_charge* = 0.95
- *minimum_cell_state_of_charge* = 0.15
- *number_of_parallel_connected_units* = 1
- *battery_pack* = "BatteryPack_1"

```
BATTERY_MODULE( "M1" ) {
    number_of_parallel_cells = 7
    number_of_serial_cells = 12
    maximum_cell_voltage = 4.2
    minimum_cell_voltage = 3
    maximum_cell_state_of_charge = 0.95
    minimum_cell_state_of_charge = 0.15
    number_of_parallel_connected_units = 1
    battery_pack = "BatteryPack_1"
}
```

The number of `BATTERY_MODULE` commands depend on the number of modules defined in the `BATTERY_PACK` command (see Battery Packs).

The number_of_parallel_connected_units can generally be ignored as it has no effect on the solution if the initial state of charge of all the cells in a module is identical. At present, this feature is supported for current and power input. Details of this feature are outlined in Cell Electrical Connectivity, Cell Numbering, and Battery Components.

Battery Model

The `BATTERY_MODEL` specifies the ECM parameters for an individual battery. In the ECM approach, the electric behavior, for example, the voltage-current response, is modeled using a phenomenological electric circuit approach. The parameters for this electric circuit need to be provided, for example, as a function of soc and temperature, in order to model the heat generated from a battery. For example, a 2nd order ECM, that is, two RC pairs, requires the input of the following parameters: capacity, V_{OCV}, R_0, R_1, R_2, C_1, C_2 and optionally entropic heat.

Type

Three types of ECM models are supported: 1st, 2nd, and 3rd order. The choice of model order depends on the desired dynamics, such as dynamic current conditions. The 2nd and 3rd order models offer enhanced accuracy for pulse-discharge experiments and dynamic current scenarios. Typically, the 2nd order model is considered the optimal choice, striking a balance between accuracy and ease of parameter determination.

First Order ECM	ECM with a single ohmic resistance in series with one parallel RC pair to represent the dynamic voltage transients (or diffusion voltages).
Second Order ECM	ECM with a single ohmic resistance in series with two parallel RC pairs to represent the dynamic voltage transients (or diffusion voltages).
Third Order ECM	ECM with a single ohmic resistance in series with three parallel RC pairs to represent the dynamic voltage transients (or diffusion voltages).

Battery Capacity

The capacity of a battery cell. The coulomb value will be exported as Ah (Amp-hr).

Open Circuit Voltage

Open circuit voltage is the voltage established between positive and negative terminals when the current is zero, this is, the circuit is open. The type includes Constant, Linear and Bilinear, whereas Constant is a constant open circuit voltage,for Linear, a table can be created which defines the state of charge (soc) versus voltage plot values. For Bilinear, Voltage is specified as a function of both soc and temperature.

Ohmic Resistance

Ohmic resistance in the ECM model represents the internal resistance of battery components. The type includes Constant, Linear and Bilinear, whereas Constant is a constant ohmic resistance, for Linear, a table can be created which defines the state of charge (soc) versus resistance plot values. For Bilinear, resistance is specified as a function of both soc and temperature.

Polarization Inputs

Polarization resistance and capacitance in the ECM model describe the dynamic behavior of the battery, for example, ion transport and charge transfer.

Entropic Heat Coefficient

Entropic heat coefficient is the derivative of open circuit potential with respect to temperature. It represents reversible heat generation in the battery cell. The type includes Constant, Linear and Bilinear.

AcuSolve input format

```
BATTERY_MODEL( "CircuitModel" ) {
    battery_model_type = first_order_ecm
    ohmic_resistance_type = piecewise_linear
    ohmic_resistance_curve_fit_values = Read("SIMLAB.DIR/ro.fit" )
    ohmic_resistance_curve_fit_variable = soc
    open_circuit_voltage_type = piecewise_linear
    open_circuit_voltage_curve_fit_values = Read( "SIMLAB.DIR/ocv.fit" )
    open_circuit_voltage_curve_fit_variable = soc
    polarization_resistance1_type = piecewise_linear
    polarization_resistance1_curve_fit_values = Read( "SIMLAB.DIR/r1.fit" )
    polarization_resistance1_curve_fit_variable = soc
    polarization_capacitance1_type = piecewise_linear
    polarization_capacitance1_curve_fit_values = Read( "SIMLAB.DIR/c1.fit" )
    polarization_capacitance1_curve_fit_variable = soc
```

```
    battery_capacity = 2.5
}
```

SimLab interface

In SimLab, all of the above mentioned ECM parameters can be entered via the Circuit Model panel, shown in Figure 11.

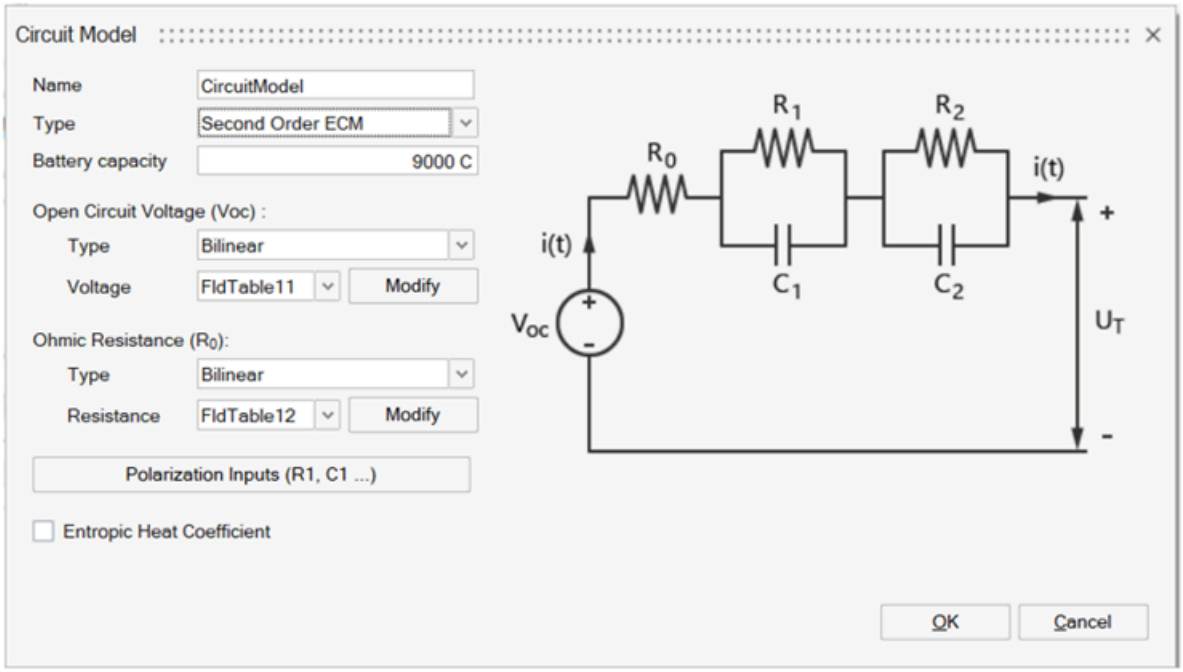


Figure 68: SimLab ECM Parameter Input Panel

Electrical Output Variables

In a battery thermos-electric simulation there are several output variables that are available as either a nodal or integrated value. For battery cells the following ECM-derived integrated values (element output) are available:

- State of charge (SOC)
- Cell current ($i(t)$ cell current determined based on input loads)
- Cell voltage (U_T, terminal voltage determined from the solution of the ECM model)

For electrically conducting components the following output variables are provided:

Electric potential — For battery packs the electric potential is determined based on the solution of charge conservation combined with the voltage generated by the cells.

Current density — The current density is taken directly from the solution of charge conservation and is given by

$$i = -\sigma \nabla \phi \qquad (405)$$

where σ is the thermal conductivity, representing the reciprocal of the resistivity provided to AcuSolve as an input.

Joule heat density

Joule heat density is the thermal energy generated due to the passage of current (electrical energy).

$$q_J = \sigma |\nabla \phi|^2 \qquad (406)$$

Electrical Boundary Conditions

For a well-defined problem, the charge conservation equation requires a reference voltage in the system. Following the sign convention where current is considered positive during discharge, a battery pack is arranged such that the reference voltage is a conductive pathway connected to the negative terminal of a battery. An illustration of this arrangement is depicted in Figure 66. In this configuration, the reference voltage (0V) is applied to the busbar connected to the negative terminal of the first battery pair (S1P1 & S1P2). The final busbar, linked to the positive terminals of the second battery pair (S2P1 & S2P2), automatically calculates the current based on charge conservation and the input provided to the battery pack (described in the Battery Packs section).

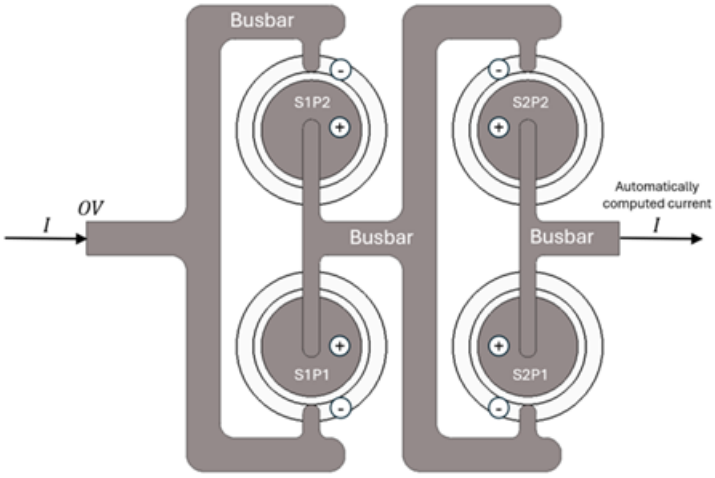


Figure 69: Electrical boundary condition setup

SimLab interface

The boundary conditions are defined in SimLab using the voltage and current panels (Figure 67):

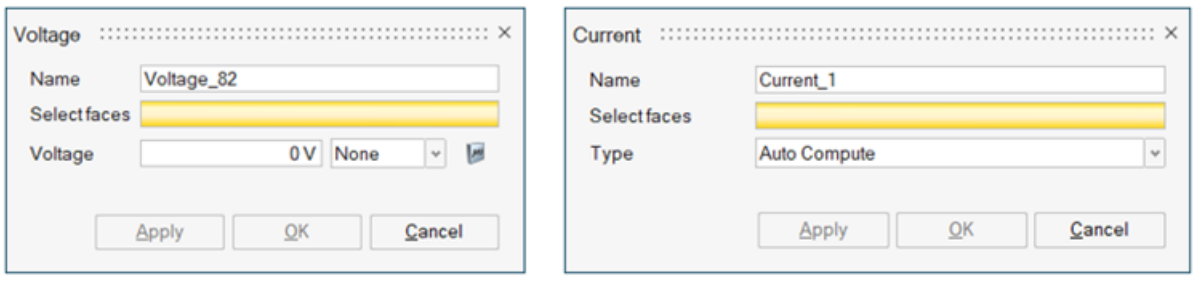


Figure 70: Voltage and current input in the SimLab interface

AcuSolve input format

The voltage and current conditions for the charge conservation equation are set under the `SIMPLE_BOUNDARY_CONDITION` section. The following snippets of the input file show these definitions.

Voltage:

```
SIMPLE_BOUNDARY_CONDITION(  "Voltage" ) {
    surface_sets = { "Voltage_PartBody.4_1_tet_tria3" }
    type = wall
    ...
    electric_potential_type = value
    electric_potential = 0.0
}
```

Current:

```
SIMPLE_BOUNDARY_CONDITION(  "Current" ) {
    surface_sets = { "Current_PartBody.3_1_tet_tria3" }
    type = wall
    ...
    electric_current_from_module = on
}
```

Cell Electrical Connectivity, Cell Numbering, and Battery Components

Typically, battery modules and packs have cell labeling in the format mSnP where m is the serial position and n is the parallel position. An example of a 2s2p configuration is given in Figure 69.

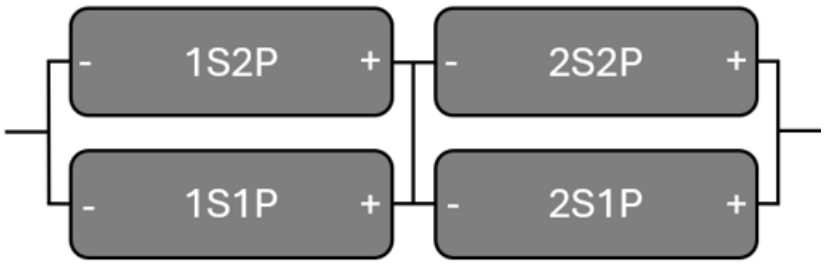


Figure 71: 2s2p module configuration

Cell numbering in AcuSolve

The number of parallel and serial cells, number_of_parallel_cells and number_of_serial_cells, respectively, are defined by their electrical connectivity first in the parallel direction and then in the serial direction. As an example, a pouch module made of 8 cells connected in a 4s2p configuration would have four pairs of parallel serially connected parallel cells: 1&2, 3&4, 5&6, and 7&8. This setup is shown in Figure 70.

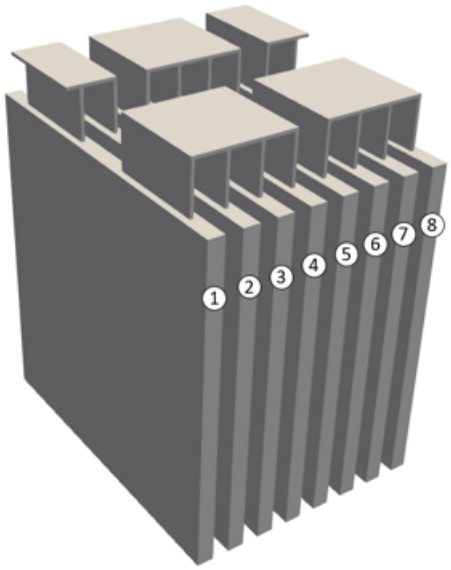


Figure 72: AcuSolve 4s2p battery module cell labeling

For a cylindrical module with 48 cells in an 8s6p configuration (shown in Figure 71), banks of six parallel connected cells, for example, 1-6 or 7-12, are connected in series. For example, cells 1-6 are connected in series to cells 7-12.

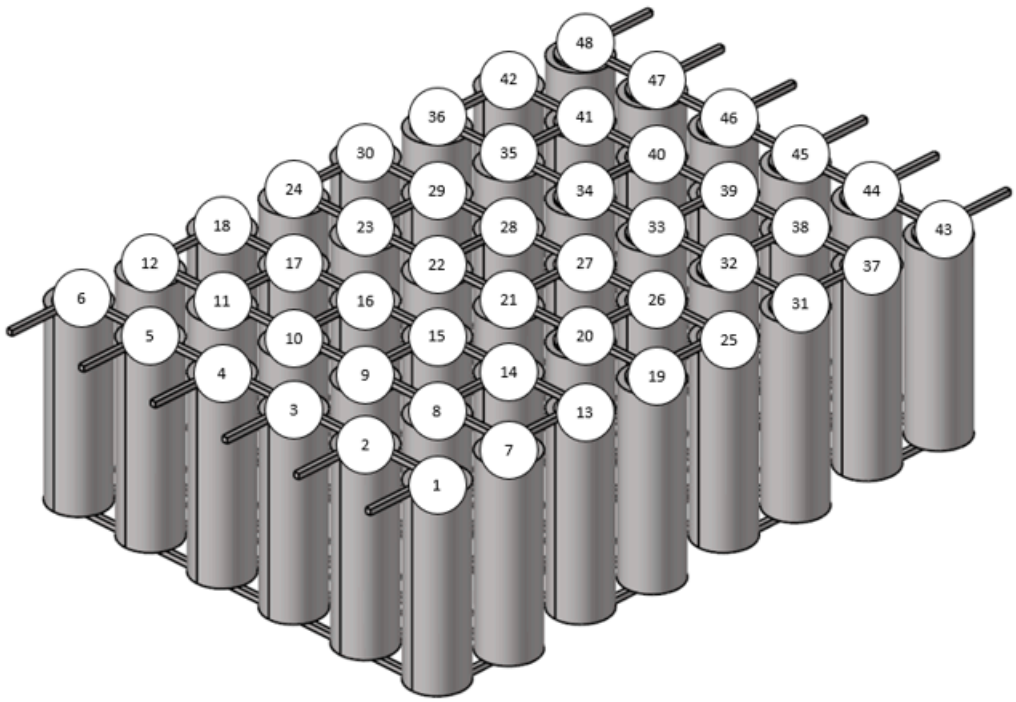


Figure 73: AcuSolve 8s6p battery module cell labeling

Moreover, each time a new module is created, the cell numbering restarts from one in the AcuSolve input file.

Component labeling in AcuSolve

A module in AcuSolve necessitates the following `BATTERY_COMPONENT_MODEL` types to be labeled:

battery_cell	Denotes the core of a detailed battery cell, with varying degrees of detail possible, or the battery itself for a homogenized cell with no detailed cell componentry.
busbar	The metal connectors linking cells together to carry current.
tab_positive	Represents the positive terminal or tab connected to the current collector. It can be depicted by either a solid volume connected to the jellyroll of the cell or a surface between a busbar and the cell.
tab_negative	Denotes the negative terminal or tab connected to the current collector. Similar to the positive tab, it can be depicted by either a solid volume connected to the jellyroll of the cell or a surface between a busbar and the cell.

The current collector is explicitly modeled only in the MSMD approach. In the single potential model, the jellyroll is a homogenized region representing the anode, cathode, separator, and current collector. The tab connects to this homogenized region in this modeling approach. Some examples cell structures are shown in Figure 72. The geometry can also include the fall details of the cell such as the current interrupt device (CID). A mock-up of such a geometry is shown in Figure 73. In this case the components would be defined as:

- The +tab set to tab_positive
- CID bottom disk, CID top disk and Terminal would be set to busbar
- Jellyroll is set to battery_cell

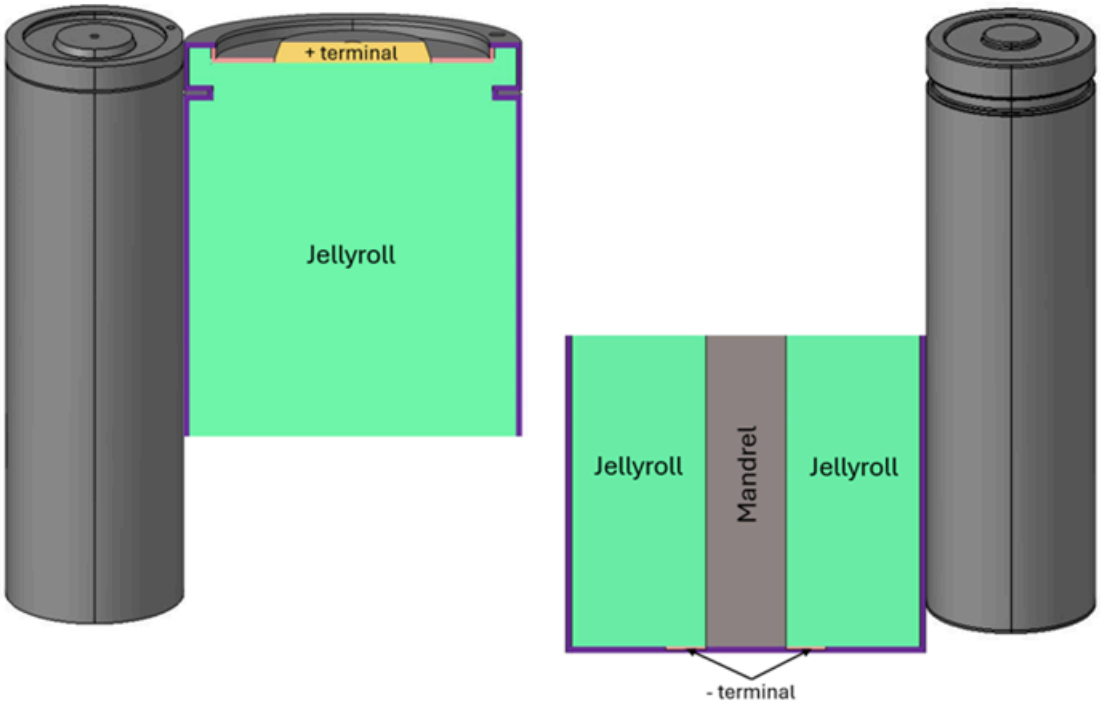


Figure 74: Example battery details for two 21700 cell designs

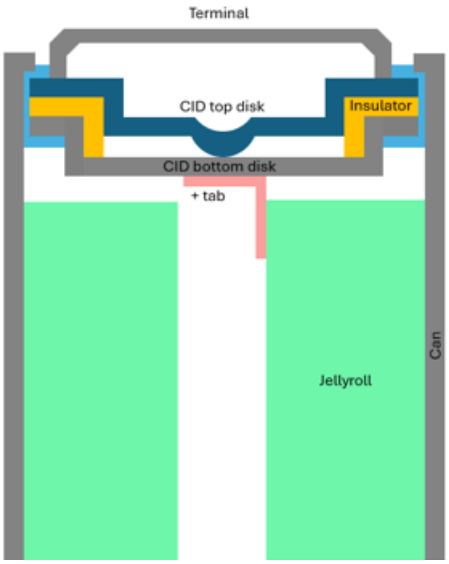


Figure 75: Detailed geometry that can be used to model the battery cell with more intricate detail. The +tab would be set to tab_positive. CID bottom disk, CID top disk and Terminal would be set to busbar; Jellyroll is set to battery_cell.

As an example, two 2s2p battery modules connected in series would be labeled as follows:

First module components:

```
BATTERY_COMPONENT_MODEL( "M1S1P1" ) {
    component_type = battery_cell
```

```
    cell_id = 1
    battery_module = "Module_1"
    isoc = 0.95
}
BATTERY_COMPONENT_MODEL( "M1S1P2" ) {
    component_type = battery_cell
    cell_id = 2
    battery_module = "Module_1"
    isoc = 0.95
}
BATTERY_COMPONENT_MODEL( "M1S2P1" ) {
    component_type = battery_cell
    cell_id = 3
    battery_module = "Module_1"
    isoc = 0.95
}
BATTERY_COMPONENT_MODEL( "M1S2P2" ) {
    component_type = battery_cell
    cell_id = 4
    battery_module = "Module_1"
    isoc = 0.95
}
```

Second module components:

```
BATTERY_COMPONENT_MODEL( "M2S1P1" ) {
    component_type = battery_cell
    cell_id = 1
    battery_module = "Module_2"
    isoc = 0.95
}
BATTERY_COMPONENT_MODEL( "M2S1P2" ) {
    component_type = battery_cell
    cell_id = 2
    battery_module = "Module_2"
    isoc = 0.95
}
BATTERY_COMPONENT_MODEL( "M2S2P1" ) {
    component_type = battery_cell
    cell_id = 3
    battery_module = "Module_2"
    isoc = 0.95
}
BATTERY_COMPONENT_MODEL( "M2S2P2" ) {
    component_type = battery_cell
    cell_id = 4
    battery_module = "Module_2"
    isoc = 0.95
}
```

SimLab interface

In the SimLab interface, the components within a battery pack are labeled using the Classify parts panel, as illustrated in Figure 74.

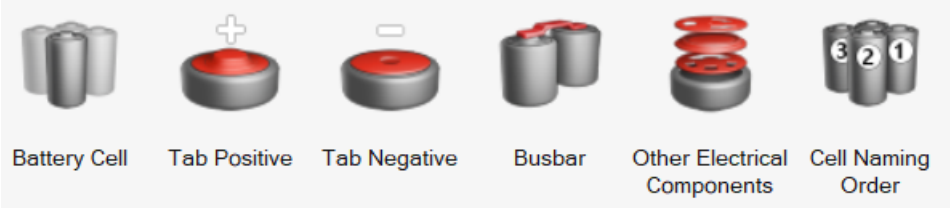


Figure 76: Panel to classify the different battery pack components

The last panel, Cell Naming Order, assigns labels to battery modules within a pack using the standard mSnP convention. Additionally, each module in the pack is identified similarly. For instance, in a battery pack with two modules, the location of the battery in the second parallel and serial position would be M2S2P2.

Cell connectivity within a module

The number_of_parallel_connected_unit is how many parallel connected unit (PCUs) you have in the configuration. Conceptually a PCU and a SCU is demonstrated in Figure 75.

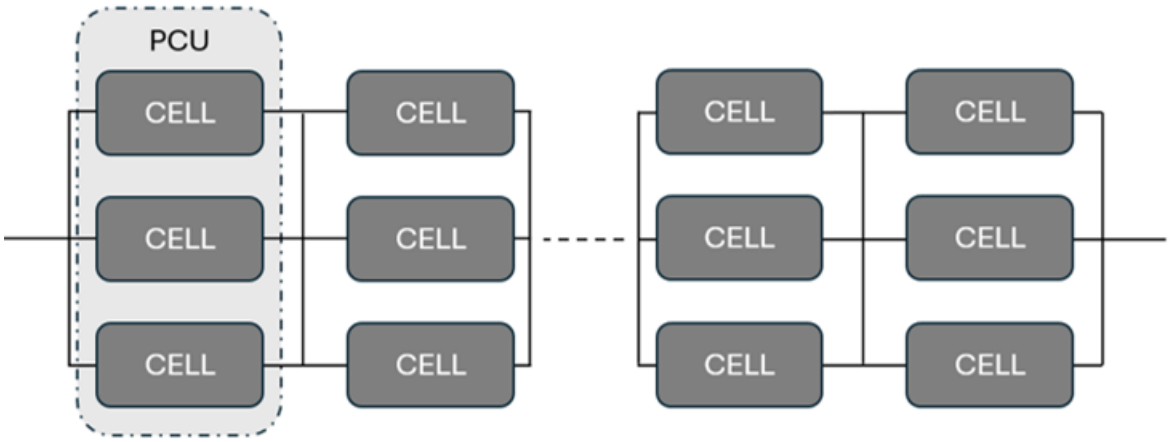


Figure 77: Battery module configurations

For example, in a 4s2p case, where the number_of_parallel_connected_units = 1, 2 or 4. The corresponding electrical connection of the battery cells would be represented by the circuits shown in Figure 76.

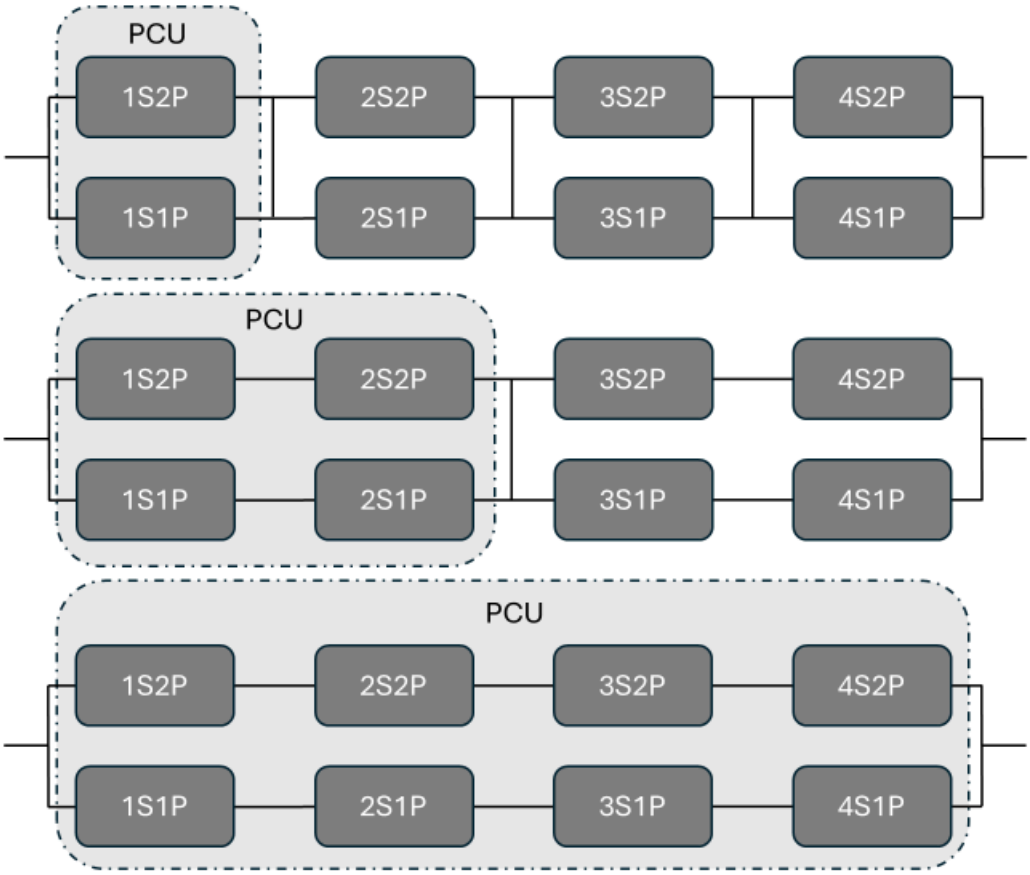


Figure 78: Example PCU configurations: Top: Four PCUs; Middle Two PCUs; and bottom: One PCU

The importance of understanding this connectivity is related to how current distributes through the battery pack when cells (and therefore modules) connected in parallel have cells of differing states. For instance, with a single PCU, it is essential for voltage to be uniform across each cell in the circuit to ensure conservation. This principle also applies to modules connected in parallel. In the AcuSolve battery solution, a rudimentary controller is integrated to balance cell voltages, thereby automatically calculating the current based on the solution derived from charge conservation in the busbars, tabs, and the properties of the cells.

Battery Thermal Runaway User Guide

The AcuSolve/SimLab battery solution for thermal runaway provides a virtual experimental platform to test pack designs for thermal safety, including the assessment of propagation characteristics, heat shield effectiveness, and general understanding of thermal abuse tolerance. Three models are available to simulate thermal runaway:

- NREL abuse model
- Arc reaction heat model
- Heat rate model

All models provide a heat-source (S_{TR}) that is applied to the energy equation. Further details of their physics and governing equations can be found in the Thermal Runaway Theory Manual.

NREL Abuse Model Formulation

The NREL model builds on previous models in literature ([1], [2], and [3]). The model solves four or five equations representing different material decomposition reactions, which are exothermic in nature, under thermal abuse conditions. These reactions can lead to uncontrollable heating and subsequently thermal runaway. The choice between four or five equations depends on whether the decomposition of the electrolyte is included as a decomposition reaction. Each of these decomposition reaction rates is governed by an Arrhenius equation. The heat generated is added as a source term in the energy equation based on the component mass and reaction enthalpy. To determine the type of model, the *type* command is set to NREL_abuse_model and *equation_type* is set to either four_equation or five_equation. For example:

```
BATTERY_THERMAL_RUNAWAY_MODEL( "NREL_model" ) {
    …
    equation_type = four_equations
    type = NREL_abuse_model
}
```

Details of the governing equations can be found in the Thermal Runaway Theory Manual.

The thermo-physical and chemical kinetic parameters required for each Arrhenius reaction equation and associated heat generation include:

- Mass of the cell components (anode, cathode, SEI (Solid Electrolyte Interphase), and electrolyte)
- Frequency factor
- Activation energy
- Reaction enthalpy

These parameters can be obtained from literature, experiments, for example, parameter fitting of DSC data, or by selecting pre-defined models (LCO (Lithium Cobalt Oxide, LiCoO2), NMC (Nickel Manganese Cobalt) and NCA (Nickel Cobalt Aluminium)) in SimLab. An example of LCO parameters for the cathode decomposition would be:

```
BATTERY_THERMAL_RUNAWAY_MODEL( "thermosphysical_chemical_kinetic" ) {
    …
    type = NREL_abuse_model
    cathode_decomposition_frequency_factor = 6.67000E+11
    cathode_decomposition_activation_energy = 2.03000E-19
    cathode_decomposition_reaction_enthalpy = 314000.0
    …
}
```

The mass of the individual cell components can be provided as either a total mass (kg) or a volume specific mass (kg/m^3). Details of this calculation are provided in the section Mass calculation (NREL model). To set the mass type the *mass_type* command is set to total for total mass or specific for volume specific mass. The masses of each of the components: *anode_mass*, *cathode_mass*, *electrolyte_mass*, and *sei_mass* are then set in either kg (for total mass) or kg/m^3 (for specific mass). For example:

```
BATTERY_THERMAL_RUNAWAY_MODEL( "total_mass_example" ) {
```

```
    …
    type = NREL_abuse_model
    mass_type = total
    anode_mass = 0.006
    cathode_mass = 0.012
    electrolyte_mass = 0.004
    sei_mass = 0.012
    …
}
```

An additional sixth Arrhenius equation, based on the work by [3], can be included to model an internal short circuit event. The equation requires the following parameter definitions:

- Internal short circuit temperature
- Frequency factor
- Activation energy
- Reaction enthalpy

The internal short circuit temperature is the trigger temperature that enables the solution of the sixth equation, that is, prior to the trigger temperature the reaction is deactivated. To enable an internal short circuit the command *internal_short_circuit* is set to on and the *internal_short_circuit_temperature* is provided. As an example:

```
BATTERY_THERMAL_RUNAWAY_MODEL( "internal_short_circuit ) {
    …
    type = NREL_abuse_model
    internal_short_circuit = on
    internal_short_circuit_temperature = 430.0
    …
}
```

A typical AcuSolve input for the NREL abuse model based on LCO chemistry for a five-equation model would be defined by

```
BATTERY_THERMAL_RUNAWAY_MODEL( "five_equation " ) {
    equation_type = five_equations
    formulation_type = lumped
    ecm_model = no
    type = NREL_abuse_model
    internal_short_circuit = off
    internal_short_circuit_temperature = 430.0
    mass_type = total
    anode_mass = 0.006
    cathode_mass = 0.012
    electrolyte_mass = 0.004
    sei_mass = 0.012
    anode_decomposition_frequency_factor = 2.50000E+13
    anode_decomposition_activation_energy = 2.24000E-19
    anode_decomposition_reaction_enthalpy = 1.714000E+6
    cathode_decomposition_frequency_factor = 6.67000E+11
    cathode_decomposition_activation_energy = 2.03000E-19
    cathode_decomposition_reaction_enthalpy = 314000.0
    sei_decomposition_frequency_factor = 1.67000E+15
    sei_decomposition_activation_energy = 2.24000E-19
    sei_decomposition_reaction_enthalpy = 257000.0
    electrolyte_decomposition_frequency_factor = 5.14000E+25
    electrolyte_decomposition_activation_energy = 4.54000E-19
    electrolyte_decomposition_reaction_enthalpy = 155000.0
```

```
    initial_anode_lithium_fraction = 0.75
    initial_sei_lithium_fraction = 0.15
    initial_sei_thickness = 0.033
    initial_degree_of_cathode_conversion = 0.04
    initial_electrolyte_fraction = 1.0
    sub_iteration_minimum_time_step_size = 0.0001
    sub_iteration_time_step_size_tolerance = 0.0001
}
```

The solution of the ODE for the Arrhenius reactions are stiff and require time-step adaptation for an accurate and efficient solution. The exact details of the adaptive time-stepping scheme are given in Adaptive time stepping (NREL and multistage model).

ARC Reaction Model

The ARC reaction model is a staged Arrhenius-based kinetic model that can be directly fitted to Accelerating Rate Calorimetry (ARC) data. An example of typical ARC data is shown in Figure 79.

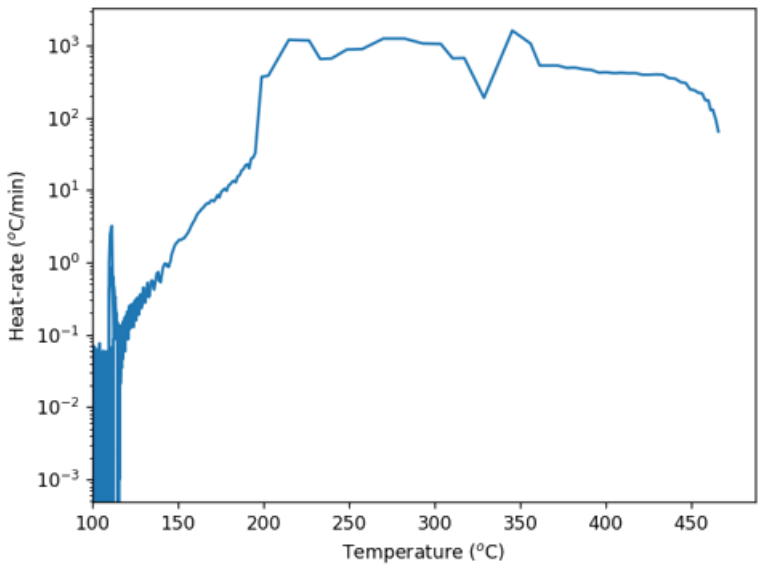


Figure 79: Heat-Rate Versus Temperature for a Typical ARC Test

To enable the arc reaction, model the type is set to arc_reaction_model.

```
BATTERY_THERMAL_RUNAWAY_MODEL( "ARC reaction model" ) {
    …
    type = arc_reaction_model
    …
}
```

This model requires the ARC data to be divided into stages or bands that loosely describe physical processes that occur during thermal runaway.

> **Note:** Since it is a data fitting approach some of the chemical processes can be lumped together. More details are given later.

The number of stages is determined by you during the data fitting phase. Figure 80 illustrates an example of dividing the ARC data into four stages based on temperature, where Ts represents the start

temperature and T_1, T_2, and T_3 represent the end temperatures of each stage one, two, and three, respectively. The actual stages are defined as: Stage 1: T_1-T_s; Stage 2: T_2-T_1; Stage 3: T_3-T_2; Stage 4: T_{max}-T_3. T_{max} is the maximum temperature recorded from the ARC test as is determined automatically from the data.

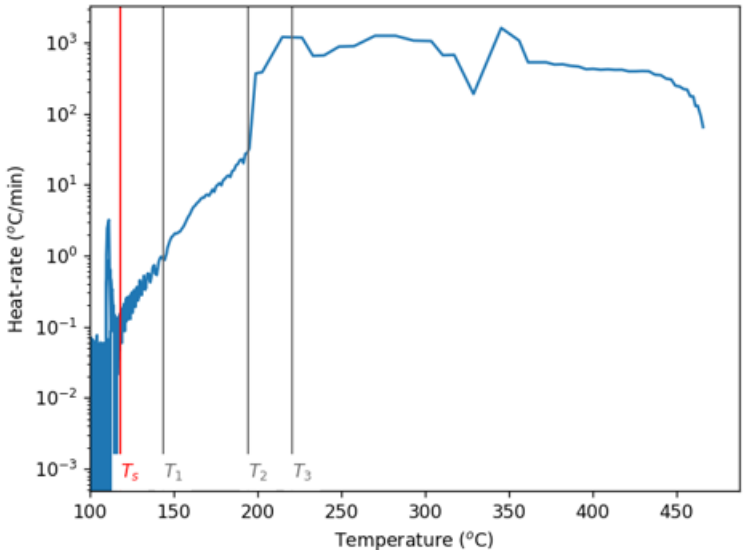


Figure 80: Four Stage Fit Showing T_s, T_1, T_2, T_3 and the Division of Stages

Although an ARC test does not individually identify the decomposition of the reacting components, it can still be represented by Arrhenius kinetics with physical meaning depending on the number of stages, for example, two stages may group together SEI, anode decomposition into the first reaction kinetic and cathode conversion and electrolyte decomposition into the second reaction kinetic. Using optimization methods, the unknown parameters can be determined for the following system of equations:

$$\frac{d\alpha_i}{dt} = \alpha_i^n (1-\alpha_i)^m \cdot A_{(a,i)} \cdot exp(-E_{(a,i)}/(k_b T)) \tag{407}$$

$$m c_p \frac{dT}{dt} = \sum m_i c_{p,i} \Delta T_i \eta_i \cdot \frac{d\alpha_i}{dt} \tag{408}$$

The unknown parameters for each stage (i) are:

- $A_{(a,i)}$: The frequency factor (units: 1/s).
- $E_{(a,i)}$: The activation energy (units: J).
- n, and m: The parameters that determine the reaction order and type, for example, auto-catalytic reaction when m>0.
- η_i: The reaction enthalpy adjustment factor.

The fitting is performed on a 0D model that ignores conduction. However, in the final simulation the right-hand side of the above equation is used as the heat-source in AcuSolve.

$$S_{TR} = \sum_i Q_i = \sum_i m_i c_{p,i} \Delta T_i \eta_i \frac{da_i}{dt} \tag{409}$$

The input for the multi-stage ARC reaction model is given as an array of stages and Arrhenius kinetic/material parameters.

As an example, a two-stage model would have the following structure.

Table 11:

Stages	**Enthalpy (h_i)**	**Activation energy ($E_{a,i}$)**	**Frequency factor ($A_{a,i}$)**	**Reaction order (n)**	**Reaction order (m)**	**Initial condition (α_i)**
1	176620.88	1.5E-19	-2.2551e6	1	0	1
2	1521147.329	2.9445e-19	2.31741e15	0	7.5	0

Note: Activation energy can be negative to account for a negative sign in the Arrhenius reaction equation.

The above in the `.inp` file would be given as follows:

```
BATTERY_THERMAL_RUNAWAY_MODEL( "two_stage_fit"){
    type = arc
    arc_type = multi_stage_fit
    multi_stage_model_parameters = { 176620.88, 1.5E-19, -2.2551E6, 1, 0, 1;
                                     1521147.329, 2.9445E-19, 2.31741E15, 0, 7.5, 0; }
}
```

Additionally, for a higher number of stages (three and four) to improve the accuracy of the final fit the heat produced by the final stage is not applied to the energy equation until above a specified temperature. The ODE solution is still solved throughout. The typical input, as an example for a four-stage fit, would be given by

```
BATTERY_THERMAL_RUNAWAY_MODEL( "four_stage_fit " ) {
    ecm_model = no
    type = arc_reaction_model
    formulation_type = lumped
    arc_type = multi_stage_fit
    parameter_fit_type = arrhenius_trigger
    multi_stage_model_parameters = Read( "ms.txt" )
    heat_rate_trigger_temperature = 500
}
```

heat_rate_trigger_temperature is the temperature at which the final stage heat is added to the energy equation. In this approach the reaction rate is still solved for the entire temperature range for the final stage. This is identified from the ARC data directly as the approximate temperature where the temperature jumps considerably. See temperature T_3 in Figure 80.

If a trigger temperature is included in the fitting process, it must be included in the simulation and vice versa.

The *multi_stage_model_parameters* can be read directly from a file, in which the file is formatted, as an example for `ms.txt`, as follows:

```
2.2244e+03 3.0693e-19 -1.2420e+20 1.0 0.0 1.0
2.1507e+03 3.8971e-19 -1.0e+25 1.0 0.0 1.0
…
```

Where each line of the file represents the parameters for a specific stage.

The ARC reaction model has three parameters related to the adaptive time-stepping scheme used to solve the governing thermal runaway equations. The exact details of the adaptive time-stepping scheme are given in Adaptive time stepping (NREL and multistage model).

Heat-Rate Model (Direct Reading of ARC Data)

A direct reading of thermal runaway ARC data is enabled via the following command:

```
BATTERY_THERMAL_RUNAWAY_MODEL( "Heat rate model" ) {
    …
    type = heat_rate
    …
}
```

In this approach a source term is calculated from the ARC data and applied directly in the energy equation given by:

$$S_{TR} = \rho_{cell} c_{p,cell} \frac{dT}{dt} \tag{410}$$

Where ρ_{cell} is the effective density of the cell, $c_{p,cell}$ is the effective specific heat of the cell. These two terms come from the `MATERIAL_MODEL` defined for the cell. dT/dt is the heat rate data in K/s from the ARC test and is read from a file. A typical input for direct reading ARC data would be given as follows:

```
BATTERY_THERMAL_RUNAWAY_MODEL( "my_arc_model"){
    type = heat_rate
    formulation_type = lumped
    heat_rate_type = linear
    heat_rate_curve_fit_values = Read( "dTdt.txt" )
    heat_rate_curve_fit_variable = temperature
    heat_rate_max_temperature = 740
    heat_rate_trigger_temperature = 490
    heat_rate_onset_temperature = 400
}
```

In the above example, *heat_rate_curve_fit_values* points to a file: `dTdt.txt`. This file is a two-column array of temperature and heat-rate values, for example:

```
…
402.752 0.006483333
402.978 0.0075
403.165 0.006233333
403.366 0.006633333
403.531 0.005516667
403.692 0.00535
403.853 0.005433333
403.993 0.004616667
404.149 0.005183333
```

...

The direct-read heat-rate model defines specific temperatures to improve accuracy and enable or disable the model. These parameters are: *heat_rate_max_temperature*, *heat_rate_trigger_temperature*, and *heat_rate_onset_temperature*.

heat_rate_max_temperature	This parameter provides the maximum temperature recorded during the ARC test.
heat_rate_trigger_temperature	This parameter provides the temperature at or around the point when thermal runaway occurs, for example, T_3 from the ARC data in Figure 80, and allows the ODE to sub-step at a time-step smaller than the PDE time-step of the energy equation.
heat_rate_onset_temperature	This parameter is the onset temperature for self-heating and can be directly identified from the ARC data.

Additional Model Information

Mass calculation (NREL model)

In the NREL thermal runaway model, the mass of individual cell components can be expressed either as total mass or specific mass. Below is a brief description of the theory.

The mass of the cathode, anode, and electrolyte can be computed using the formula: $m_T = m_s V$, where m_T is the total mass, m_S is the specific mass, and V the volume of the jellyroll. The jellyroll volume can be the whole cell in the case of a homogenized representation of the battery or the actual jellyroll geometry. For instance, a homogenized 18650 cell may have a jellyroll volume of $1.66\times10^{-5}m^3$, while a more precise estimation based on studies by [2] and [4] yield volumes of $1.05\times10^{-5}m^3$, and $1.29\times10^{-5}m^3$, respectively. Alternatively, if a detailed and accurate jellyroll geometry is available, it can be directly utilized.

Homogenized 18650 LCO cell	If the cell volume is $1.66\times10^{-5}m^3$ and the specific masses of the cathode and anode of the cell are 722.89 kg/m^3 and 351.45 kg/m^3, respectively, then the total mass of the cathode is 0.012 kg, and 0.006 kg for the anode.
Detailed Jellyroll 18650 LCO cell	If the volume of the jellyroll is $1.29\times10^{-5}m^3$ and the specific masses of the cathode and anode are again 722.89 kg/m^3 and 351.45 kg/m^3, respectively, then the total mass of the cathode is 0.0093 kg and 0.0047 kg for the anode.

Adaptive time stepping (NREL and multistage model)

Adaptive time stepping is necessary for the ODE-based thermal runaway models to increase computational efficiency. In the adaptive time-stepping algorithm, the ODE is sub stepped based on how the solution evolves over time. The algorithm updates the ODE sub stepping size once per PDE solution update. To determine the ODE time-step size, an estimate of how rapidly the solution is evolving is calculated using the formula:

$$e_n = \frac{y_n - y_{n-1}}{1 + \min(y_n, y_{n-1})} \tag{411}$$

where y_n is solution at the current time step, y_{n-1} is solution at the previous time step, e_n is calculated for each ODE solution variable for all cells, with the maximum value being selected among them. Based on control theory, it can be shown that the updated time step can be given by an integral controller:

$$\Delta t_I = \frac{Tol}{e_n} \cdot \Delta t_{n-1} \tag{412}$$

Here, *Tol* is a user-defined tolerance, which determines how sensitive the time step should be when solution is changing, Δt_{n-1} is the time step size at the previous time step. Once Δt_I is determined, it is bound between two user-defined limits: Δt_{min} and Δt_{max}.

Note: Δt_{max} cannot exceed Δt_{PDE}, which is the PDE time step size, that is:

$$\Delta t_n = \min(\max(\Delta t_{min}, \Delta t_I), \Delta t_{max}) \tag{413}$$

The adaptive time stepping commands for *Tol* and Δt_{min} are provided in the `BATTERY_THERMAL_RUNAWAY_MODEL` section of the `.inp` file and are given by the commands *`sub_iteration_minimum_time_step_size`* and *`sub_iteration_time_step_size_tolerance`*, respectively. An example input would be:

```
BATTERY_THERMAL_RUNAWAY_MODEL( "my_arc_model"){
    …
    sub_iteration_minimum_time_step_size = 0.0001
    sub_iteration_time_step_size_tolerance = 0.0001
}
```

References

[1] T. D. Hatchard, D. D. MacNeil, A. Basu and J. R. Dahn, "Thermal model of cylindrical and prismatic lithium-ion cells," Journal of The Electrochemical Society, p. A755, 2001.

[2] K. Gi-Heon, A. Pesaran and R. Spotnitz, "A three-dimensional thermal abuse model for lithium-ion cells," Journal of Power Sources, pp. 476-489, 2007.

[3] P. T. Coman, E. C. Darcy, C. T. Veje and R. E. White, "Modeling Li-ion cell thermal runaway triggered by an internal short circuit device using an efficiency factor and Arrhenius formulations, "Journal of The Electrochemical", p. A587, 2017.

[4] P. J. Bugryniec, J. N. Davidson and S. F. Brown, "Computational modeling of thermal runaway propagation potential in lithium iron phosphate battery packs," Energy Reports, pp. 189-197, 2020.

AS-EDEM Coupling Port Specification for AcuSolve 2024.1 and EDEM 2024 or Later Versions

Multi-Node Coupled Jobs	When running multi-node AcuSolve-EDEM coupled jobs (AcuSolve on one node and EDEM on another), a user-specified port number is required.
Multiple Coupled Jobs on the Same Machine	If you need to run multiple coupled jobs on the same machine, you must provide a unique port for each job. In this scenario, the host should be specified as: host = "localhost".
Port and Host Parameters	The *EXTERNAL_CODE { socket_host, socket_port }* parameters must be used to specify the host and port.
Socket Port Range	The socket_port must be within the range 32768 to 65535.
Version Compatibility	For all multi-node coupled jobs, you must provide both the host and port. AcuSolve 2024.1 must be used in conjunction with EDEM 2024. Using AcuSolve 2024.1 with an earlier version of EDEM will not work.

Run AcuSolve on Linux Servers

AcuSolve can be run on Linux servers in the following ways.

Execute AcuRun (servers without PBS/LSF/CCS)

On servers without PBS/LSF/CCS, AcuSolve can be run by using the acuRun executable directly.

With AcuSolve environment sourced

```
acuRun -pb <problem name> -np <number of processors> -nt <number of threads>
```

Without AcuSolve environment sourced

```
<Installation-Path>/altair/hwcfdsolvers/acusolve/linux64/bin/acuRun -pb <problem
name> -np <number of processors> -nt <number of threads>
```

Execute AcuRun (servers with PBS/LSF/CCS)

On servers with PBS/LSF/CCS, the PBS environment variables need to be set. They can be set in the command line or through a script. Here is an example of such a script.

```
#! /bin/sh -f

PBS -N <problem name>

PBS -o <problem name>.log.pbs

PBS -e <problem name>.error.pbs

PBS -l select=<np/ppn>:ncpus=<ppn>:mpiprocs=<ppn>

PBS -l walltime=00:2880:0
```

```
PBS -l place=scatter

PBS -l place=group=chassis

set -v

cd <working directory>

acuRun -pb <problem name> -np <number of processors> -nt _auto -do all -pbs_attach -pbs
```

Execute acuSub (servers with PBS/LSF/CCS)

On servers with PBS/LSF?CCS, acuSub is the most efficient way to run AcuSolve. Refer to the AcuSub section in the AcuSolve Programs Reference Manual for more information.

AcuSolve-MotionSolve Two-Way Coupling Job Submission on Linux Servers using Command Line Options

AcuSolve-MotionSolve two-way coupling can be executed through the terminal/command line or through shell scripts on Linux servers with various options.

Execute acuMSI

```
< Installation-Path > hwsolvers/motionsolve/bin/linux64/acuMSI -aport <APORT> -mport
<MPORT>
```

where

-aport <integer>
: Specifies the communication port number for communication between AcuSolve and acuMSI. The default is 48000. If you need to change the default port for communication between AcuSolve and acuMSI, in addition to changing this argument, you also have to specify the changed port number in the AcuSolve input file.

- mport<integer>
: Specifies the communication port number for communication between MotionSolve and acuMSI. The default is 94043. If you need to change the default port for communication between MotionSolve and acuMSI, in addition to changing this argument, you also have to specify the changed port number in an environment variable MS_AS_PORT. MotionSolve checks for this environment variable at the start of the simulation and changes its listening port accordingly.

Execute MotionSolve

Through hwsolver.tcl

```
< Installation-Path>/altair/hw/common/acc/scripts/hwsolver.tcl -solver MS -nt
<MotionSolve threads> -file <MotionSolve input file>
```

Through MotionSolve executable

```
<Installation-Path>/altair/scripts/motionsolve -file <MotionSolve input file>
```

Execute AcuRun

With AcuSolve environment sourced

```
acuRun -pb <problem name> -np <number of processors> -nt <number of threads>
```

Without AcuSolve environment sourced

```
<Installation-Path>/altair/hwcfdsolvers/acusolve/linux64/bin/acuRun -pb <problem
name> -np <number of processors> -nt <number of threads> -ecport <APORT>
```

You can also create a script with PBS environment variables set to execute all of these commands. Here is an example of such a script.

```
#! /bin/sh -f

#PBS -N <problem name>

#PBS -o <problem name>.log.pbs

#PBS -e <problem name>.error.pbs

#PBS -l select=<np/ppn>:ncpus=<ppn>:mpiprocs=<ppn>

#PBS -l walltime=00:2880:0

#PBS -l place=scatter

#PBS -l place=group=chassis

set -v

cd <working directory>

<installation-path>/hwsolvers/motionsolve/bin/linux64/acuMSI &

<installation-path>/altair/scripts/acc -solver MS -nt <number of threads> <MotionSolve
xml input file> &

acuRun -pb <problem name> -np <number of processors> -nt _auto -do all -pbs_attach -pbs
```

Guidelines for Quality CFD Modeling

5

This chapter covers the following:

Introduction to CFD Modeling Guidelines

The field of computational field dynamics (CFD) has seen tremendous advancement in recent decades, riding on the back of the phenomenal advances in computational science and power.

CFD evolved throughout the 20th century. In the pre-1950s, semi-analytical methods were developed to solve the complex problems that were still unsolved using analytical solution methods. During the same time period, numerical methods also advanced and became more mature with respect to numerical stability, speed and accuracy. Post 1950, with the availability of low cost computing power and advancement in computer science, researchers all over the world were able to develop complex CFD models and test them on a variety of applications, enabling the advancement of CFD at an accelerated scale. High-fidelity CFD solvers running on powerful computers today can accurately model complex fluid flows, solving the governing equations over billions of grid points, performing trillions of calculations at speeds unthinkable of a few years ago.

However, access to large amounts of computing power does not necessarily equate to high quality results from a CFD analysis. Modern CFD solvers are specialized and complex computer programs. The accuracy of the results provided by the solvers is subject to the accuracy of the inputs. It is necessary that a complete and correct set of inputs be provided to the solvers so that they can perform their task as correctly as possible. Some of the essential aspects of these inputs to be provided to the solvers are discussed in this section. You will be able to apply the basic yet important CFD analysis guidelines that are discussed to your own applications. Following these guidelines in most cases will ensure that the results obtained from the CFD analyses are reasonably accurate with only a small margin of error. However, it cannot be stressed enough that these guidelines should be treated as such, only guidelines, and not an authoritative set of instructions that will ensure correct analyses every time and for every application. You are both encouraged and solicited to apply sound engineering judgement to individual cases, as well as explore validation of your modeling methodology for each application of interest.

A CFD simulation involves solving of the complex Navier-Stokes equations, often coupled with other equations to determine distribution of flow, temperature and turbulence fields across a discretized simulation domain. The Navier-Stokes equations are highly nonlinear and are impossible to solve analytically with available methods. CFD solvers solve these equations numerically, often with a set of assumptions to simplify the solution and a set of parameters specific to the numerical method employed, to produce a stable solution. The numerical approach employed can vary from solver to solver. Also critical to the solution method is the set of constraints that you provide to the solver. This requires sound understanding of the problem definition and the objective of the simulation. At best, a CFD solver is a tool to simplify the task of solving the complex flow-field equations and must be applied carefully for the results to be a true and reliable representation of the actual physical processes. However, since even the best solver technique is based on a numerical approximation method, every CFD result has a degree of error and uncertainty introduced in it by these approximations.

Obtaining accurate results from a CFD simulation depends on many factors, such as how well you have modeled the conditions of interest and how well the CFD software has solved the equations necessary to model these conditions. In order to minimize the errors associated with a solution you must understand what these errors are. Some of these common error sources are listed below.

- Modeling error: This type of error occurs because of the difference between the true physical processes and the equations that are used to model the processes.

- Numerical or Discretization error: This type of error is induced because the model equations are not solved continuously but on a finite number of grid points in the domain.
- Iteration or Convergence error: This type of error is induced because the solution is iterated to a finite convergence level, which is not zero.
- Round-off error: This type of error is induced because the numbers are rounded off when stored in computer memory because of storage limitations.
- Input errors: These are the errors induced due to uncertainties in inputs, such as the choice of physical model pertinent to the application, boundary condition specification and inaccurate input.
- Code errors: These are the errors in the CFD solver code.

Ensuring accurate results requires extensive verification of the software and extensive validation of the modeling approach for the application of interest. Verification and validation are important to minimize the solver-side sources of error and uncertainty in the CFD analysis. Verification will ensure that the solver solves the equations correctly and validation tests the extent to which a chosen model accurately represents reality. Verification and validation are an important aspect of the CFD modeling process but will not be the focus of this discussion. Instead, this section focuses on general CFD quality assurance guidelines that you should follow to ensure that you can expect reasonable results from your analysis. The discussion is divided into several topic areas that represents an important aspect of modeling that addresses one of the error types discussed above. You need to be focused on these aspects, besides the user errors, to ensure high quality results. The following modeling aspects are discussed.

- Geometric sensitivity for geometric uncertainties
- Mesh sensitivity for discretization error
- Boundary condition sensitivity for errors in boundary conditions input
- Physical model sensitivity for modeling error
- Convergence tolerance sensitivity for convergence error

For many CFD applications the true solution to the equations is not always known. In the absence of an analytical solution, or high quality experimental data to compare against, it is difficult to judge the accuracy of a CFD simulation. However, you can still investigate the suitability of your modeling practices by performing sensitivity studies. This practice reveals aspects of your model that have a significant impact on your solution. Once these aspects are identified further work can be done to determine the most appropriate modeling practices for each of these aspects that will minimize the error associated with the individual aspects, ensuring good quality of results overall. It should be noted that performing a full suite of sensitivity studies to evaluate the solution is a costly and time consuming process. Since there are a number of aspects to be evaluated and all of them can impact the solution combining all of them in a single study is not advisable. It is best to study them one at a time. When performing a sensitivity study one aspect of the model that is being studied should alone be changed and everything else should be kept unchanged until a satisfactory solution is achieved. This process is then repeated for all of the aspects that you are interested in studying. In practice, many investigations are avoided and engineering judgment is used in its place. However, for applications where accuracy is of high importance a full set of sensitivity studies needs to be performed to ensure that modeling errors are minimized in the solution. It should be remembered that the sensitivity studies cannot take into account user and code errors and do not warrant that the solution achieved will be the true physical solution. The following discussion describes different sensitivity studies that are commonly performed when investigating the modeling practices to produce a quality solution.

As an exercise, and also to demonstrate to you the effect of the choices made for each of these aspects over the course of a sensitivity study, the discussion of the topics covered will be accompanied with the discussion of a simple but interesting case, the backward facing step case. For a detailed description of the case refer to the AcuSolve Validation manual in your AcuSolve installation directory. However, here you will not analyze the case with the objective of validating the model but with emphasis on educating you about guidelines for setting up the model correctly and identifying the appropriate modeling choices to make. A brief discussion of the case is as follows.

The problem consists of a fluid flowing in a channel with a sharp expansion. The step in the channel causes the flow to detach at the step and reattach some distance downstream of the step. The detached flow also causes recirculation in the region between the step and the reattachment point. The fluid properties are close to that of air. The Reynolds number at the inlet is 36,000, which is based on the step height. The flow is turbulent and you will pursue a steady state solution. The objective of the analysis is the accurate representation of the recirculation zone and determination of the reattachment point. The image below shows the schematic geometry of the case setup.

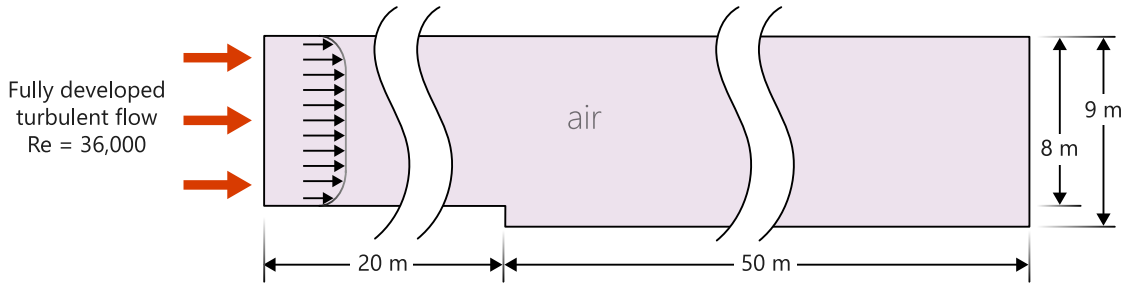


Figure 81: Schematic Geometry of the Backward Facing Step Case

This situation roughly mimics the behavior of flow exiting from a flow channel into a plenum of a plate type heat exchanger. An engineer is interested in placing a pressure tap along the lower wall to measure the total pressure drop through the system. In order to avoid spurious measurements from the pressure tap it is necessary to place the tap downstream of the recirculation region. Therefore, the location of the reattachment point must be known.

Geometric Sensitivity

This section discusses and illustrates the effects of the geometrical representation of the problem, and the discretization of the geometry on your CFD solution.

CFD simulation involves the creation of a discrete mathematical model of an object, assembly, region in space, and so on. In order to obtain accurate results from the CFD analysis, analysts must ensure that they are using an appropriate representation of the geometry that they are trying to simulate. When simulating flow involving complex geometric features, it is necessary to evaluate the degree to which the discretized representation of the geometry matches the actual component of interest. For example, it is common practice to remove small features such as small bolt holes and fasteners when simulating complex assemblies. These features may or may not have a significant impact on the results of the simulation. Analysts need to determine the appropriate level of detail required to achieve an accurate representation of the flow field. This often times requires a geometry sensitivity study to determine.

Another aspect of proper geometry modeling is how well the mesh matches the geometry. For curved geometries, it is necessary to have sufficient element density to resolve the curvature smoothly and provide an appropriate reproduction of the shapes. In addition to the mesh density, it is also necessary to ensure that the underlying CAD model provided appropriate resolution of the geometry. When constructing meshes from discrete CAD models you must pay particular attention to this to ensure that the discrete model is an appropriate representation of the true geometry.

The third geometric modeling aspect is the extent of the domain used for analysis. When modeling external flows, it is necessary to model a sufficiently large domain that minimizes the impact of the numerical constraints applied to the model. For example, when trying to simulate an airfoil in an infinite medium it is necessary to size the domain large enough so that the surrounding region is not constraining the flow around the airfoil in an unrealistic manner. Similarly, for internal flows, the boundaries of the domain should be located far away from the location of any phenomena of interest in the domain to minimize any interference in the results due to the presence of the boundaries.

A final aspect of the geometric modeling applies to the problems that have a rotating component within the domain. Examples of such flows are the flows in pumps and turbomachines where fluid interacts with a rotating part, adding swirl to the flow. The common approach while modeling such cases is splitting the domain into a rotating part which immediately surrounds the rotating component (rotor), and the static part that makes up the rest of the domain. While modeling such cases, it is very important to understand the sensitivity of the location of the interface which will separate the static and the rotating part. An interface located either too close or too far from the rotor reduces the accuracy of the solution. For an accurate solution it is advisable to test the sensitivity of the solution to the location of this interface. Unfortunately, there are no set guidelines to determine this, as the optimum location is a function of the dimensions of the domain and the rotor, the relative velocity between the fluid and the rotor, and so on.

The Backward Facing Step Case

A look at the schematic of the backward facing step case indicates that the outlet boundary of the domain has been modeled at a distance of 50 times the step height downstream from the step location. It is desired that the extent of the recirculation zone and the subsequent reattachment of the flow downstream of the step are not affected by the proximity to the boundary.

The extent to which the boundaries should be distanced from the region of interest should be evaluated on a case by case basis. It can be argued that using a distance of 50 times the step height is a very conservative estimate for the backward facing step (BFS) problem, which may be considered a relatively simple problem. Using such a conservative value is not always feasible especially as geometries and models become complex. The modeling aim should be to minimize the interference of boundaries on the results. Unless there is a constraint on the model that prevents shifting the boundaries sufficiently far from the region of interest, or you are interested in studying the effect of boundaries on the model, this practice should be followed.

Mesh Sensitivity

This section illustrates the various effects that mesh refinement, mesh quality and topology have on your CFD solution.

Mesh Refinement

The role of a CFD solver is to approximate the solution to a coupled set of partial differential equations.

This typically entails solving for the pressure, velocity and turbulence fields within a given domain. These fields can be both spatially and temporally varying. To obtain a solution to the governing equations, the simulation domain is discretized into small, finite regions, referred to as elements or cells, that are made up of faces that are connected by nodes or vertices. The solver then solves the governing equations over each of these finite regions to approximate the solution over the whole domain. In the case of a finite element solver, the solution is obtained and stored at the nodal locations. In other words, the solution is permitted to vary at the locations of the nodes in finite element solvers. The solution between the nodal locations is assumed to vary linearly, in most cases. With this in mind, it is clear that the mesh resolution thus must be sufficient so that all variations in the flow are sufficiently captured by an assumed linear profile between the locations where the solution is stored. Mesh resolution is a measure of the distance between two consecutive measurement points in the mesh.

A consistent numerical analysis will provide a result which approaches the actual result as the mesh resolution approaches zero. In this case the elements become infinitesimally small and the mesh behaves almost like a continuous representation of the domain. Thus, the discretized equations will approach the solution of the actual equations. One significant issue in numerical computations is what level of mesh resolution is appropriate. This is a function of the flow conditions, type of analysis, geometry and other variables. An insufficiently resolved mesh that has nodes far apart will not capture the gradients well enough when assuming linear variations between nodes and will lead to a less accurate numerical solution. An unnecessarily finely resolved mesh will lead to higher solution costs without any gain in accuracy. You are often left to start with a coarse mesh resolution and then conduct a series of mesh refinements to assess the effect of mesh resolution. This is known as a mesh refinement study. Strategic mesh refinement will call for you running a solution on the coarse mesh, studying the results and identifying the areas that require enhanced resolution to improve the solution. The refinement process should be continued until the results do not change with further refinement. Such a solution may be referred to as the grid converged solution.

CFD practitioners must recognize the distinction between a numerical result which approaches an asymptotic numerical value and one which approaches the true solution. It is hoped that as the mesh is refined and resolution improves the computed solution will not change much and approach an asymptotic grid converged value, that is, the true numerical solution. However, there still may be error between this asymptotic value and the true physical solution to the equations. This behavior is shown in the image below.

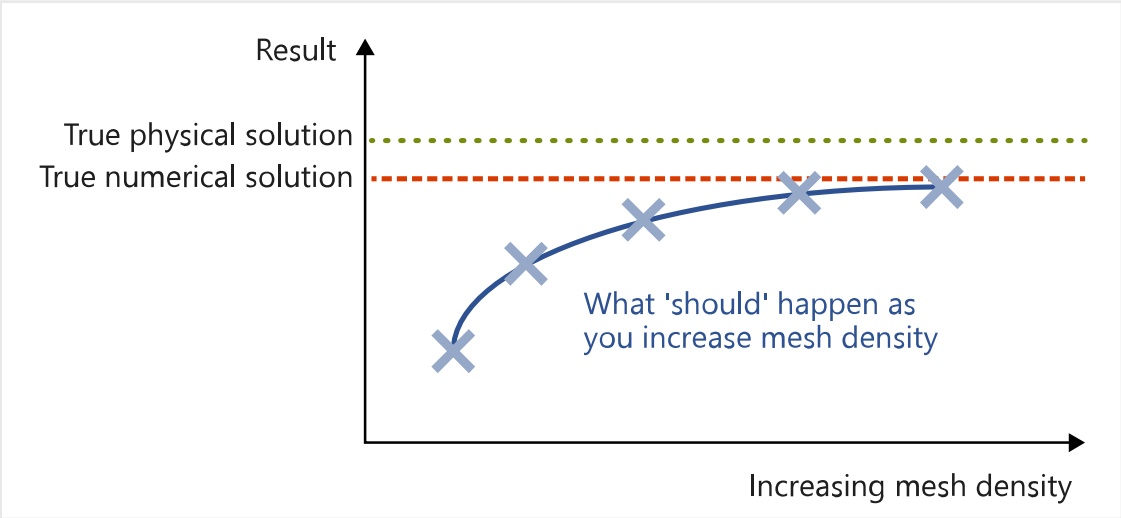


Figure 82: Effect of Increasing Mesh Density on Accuracy of the CFD Solution

Mesh Quality and Topology

Another possible sensitivity associated with the mesh used for a CFD analysis applies to the quality and type of elements that are used.

Each CFD solver has its own guidelines for what types of elements behave best from a stability, accuracy and convergence stand point. In addition to this, there are generally guidelines about the quality of each type of element that is suitable for use within the solver. Many finite volume solvers cite the aspect ratio of the elements, element skewness, orthogonality of the element faces and the element growth factor as critical metrics for determining their suitability for use. For this reason, it is sometimes recommended by code vendors that you use hexahedra elements for finite volume codes. Finite element solvers typically do not have this same constraint and will often recommend the use of tetrahedral

elements. You should consider the needs of your solver when constructing a mesh to determine the most suitable discretization for your application.

The Backward Facing Step Case

The following example uses the backward step facing case. Consider a simulation of this flow using a CFD solver. Assume that you are creating a very coarse mesh on the geometry and then you are running a simulation. The CFD solver produces a solution to the Reynolds-averaged Navier-Stokes (RANS) equations on this discretization that results in a seemingly realistic flow field. The flow is characterized by the large separation zone downstream of the step and the recovering boundary layer after reattachment. Representative results for such a simulation are shown in the image below.

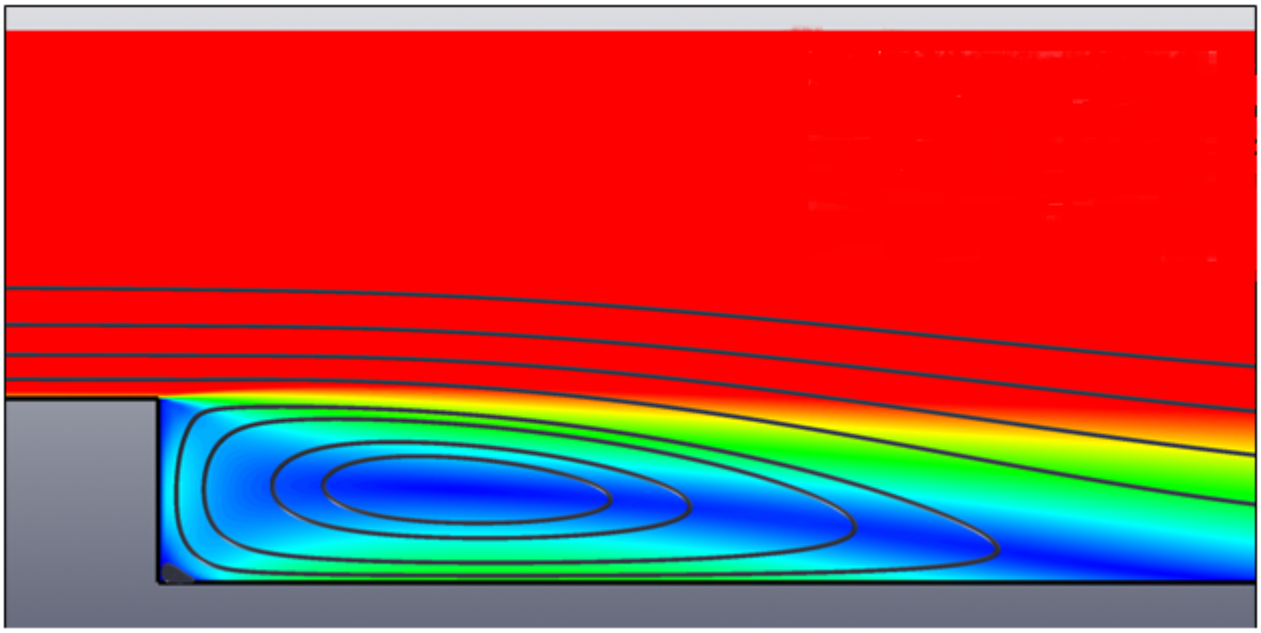
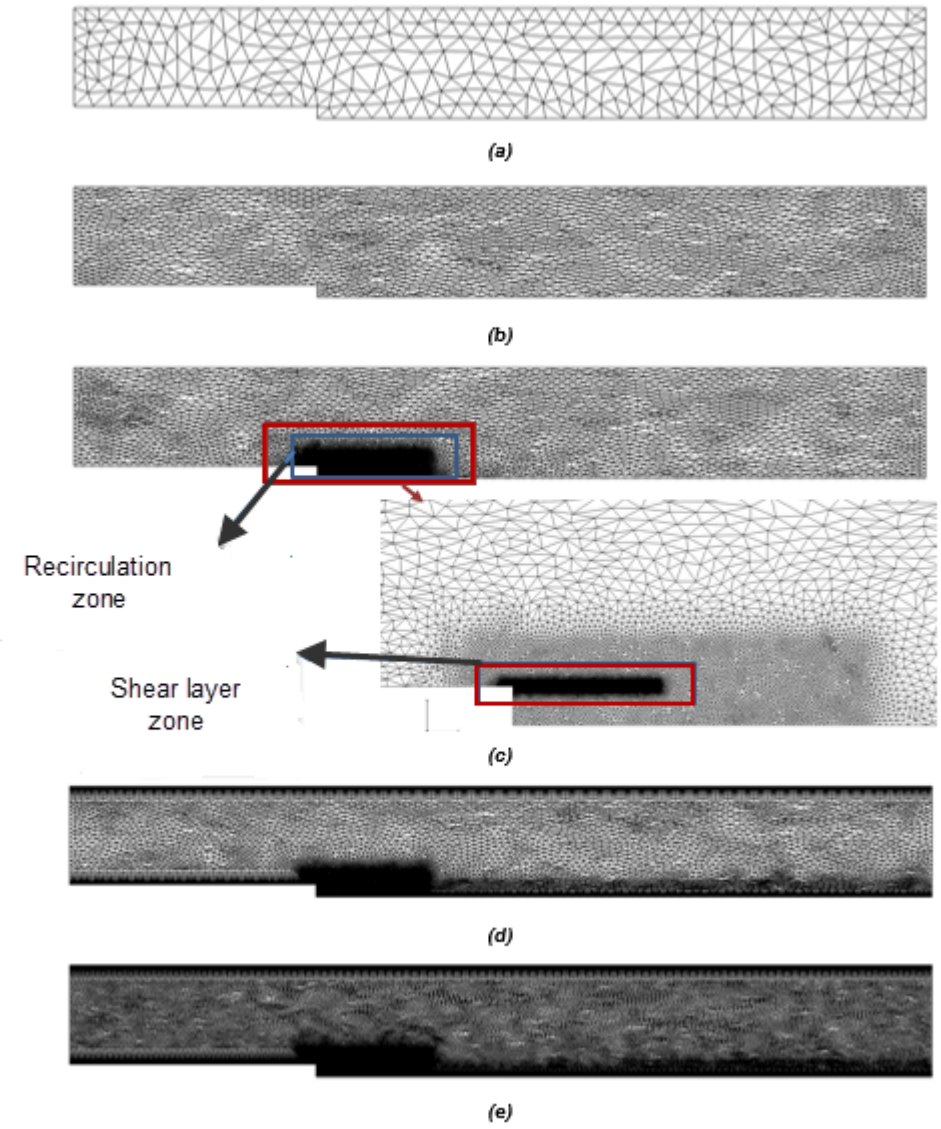


Figure 83: Streamlines Showing Separation Zone in Backward Facing Step Model

However, if there are insufficient points in the mesh to provide a good estimate of the gradients between the solution points the solver will not be able to resolve the flow features properly. Using this example, the shear layer that develops in the separation zone is under resolved and this leads to a poor representation of the reattachment point. In order to improve the solution it is necessary to refine the mesh in the locations of high velocity gradients. By doing this, the solver is better able to represent the local variations in the solution leading to a better approximation of the true solution to the governing equations. The image below shows five different meshes created for the backward facing step problem. Each mesh is progressively refined with respect to the previous mesh to illustrate the areas of refinement that must be paid careful attention to for this particular application.

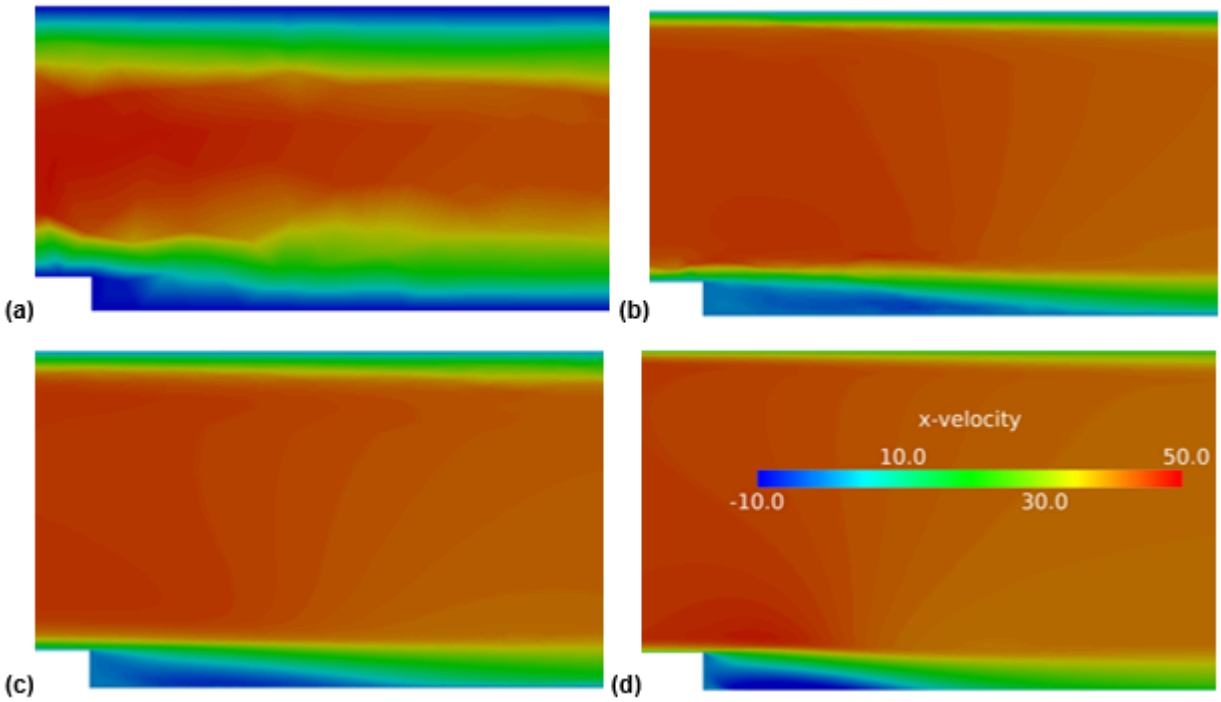


(a) Mesh 1; (b) Mesh 2; (c) Mesh 3; (d) Mesh 4; (e) Mesh5

Figure 84: Five Meshes Used in Backward Facing Step Mesh Sensitivity Study

The figure below shows the velocity profiles obtained from the solutions of each of the meshes using a consistent solver setup. Images (a) to (d) each correspond to the solution from mesh one through four, respectively. It is easy to see how the solution improves with each refinement in the mesh. A detailed discussion of these figures is covered in the mesh refinement process below.

Figures a-d each correspond to mesh one through four, respectively.



(a) Mesh 1; (b) Mesh 2; (c) Mesh 3; (d) Mesh 4

Figure 85: Velocity Profiles Downstream of the Step Depicting Mesh Sensitivity for the Backward Facing Step Case

The refinement process is briefly discussed below.

Mesh 1-image (a)

This is the first mesh created for this case. As mentioned earlier, you begin with a coarser mesh which is systematically refined to achieve a mesh suitable for the application.

Global absolute element size of 2m for each element is used for this mesh.

A look at the solution from Mesh 1 (image 5a) reveals thick boundary layers near the top and bottom walls and core flow velocity gradients in the vertical direction.

The reattachment location could not be obtained from this solution and the flow appears to never have made contact with the lower wall downstream of the step.

Mesh 2-image (b)

The previous solution had a poor representation of the core flow and boundary layers, indicating a mesh resolution insufficient to capture the gradients present.

It is advised to use the anisotropic mesh size setting for this case. This means that you will specify a different mesh setting for the horizontal and vertical direction for efficiency. Using anisotropic mesh sizing allows you to efficiently refine the mesh by targeting the specific direction where gradients are stronger. The mesh

displayed in image 4(b) is generated using the following global mesh size specification.

Mesh Size Type	Absolute Anisotropic
Element size in horizontal direction (x)	2 m
Element size in vertical direction (y)	0.4 m

This will give about five times more elements in vertical direction.

This solution (image 5b) shows much better resolution of the velocity field, but the shear layer and the recirculation zone are not properly represented in the solution as irregularities can be seen in the velocity profiles in these locations.

The reattachment point for this mesh is observed to be at a distance of 8.44 m downstream of the step.

Mesh 3 – image (c)

The shear layer zone and the recirculation zone must be properly resolved for proper representation.

Mesh can be refined in these zones using localized mesh refinement zones. The suggested mesh zones for this refinement are shown in the image. The mesh sizes used for these zones are as follows:

- Recirculation zone mesh size is 0.1 m.
- Shear layer zone mesh size is 0.01 m.

After refinement the shear layer and the recirculation zone are much better represented, as shown in image (c).

The reattachment point for this mesh is observed to be at a distance of 7.42 m downstream of the step.

Mesh 4-image (d)

Without proper resolution at the boundary it is difficult to visualize where exactly the recirculation zone ends and the flow reattaches to the lower wall. In this step boundary layers are added near the walls to remedy that.

The following settings are used for creating the boundary layers.

Wall Lower Downstream, Wall Lower Upstream, Wall Upper	**Wall - Step**
Mesh Type: Absolute Anisotropic	Mesh Type: Absolute
X, Y element size: 2 m, 0.4 m	Element size: 0.1 m
Boundary layer type: Full control	Boundary layer: No
Resolve type: Number of layers	
First element height: 0.00035 m	
Total layer height: 1.25 m, for Wall Lower Downstream: 0.625 m	
Layer growth rate: 1.3	
Boundary layer elements type: Tetrahedron	

Wall-Step does not need a boundary layer because it is not expected to have attached flow. The step edge has been resolved with a fine mesh because it is the location of separation.

Results from this mesh can be seen to have a proper representation of all of the features of interest for this case. The recirculation zone is properly represented and the location of the reattachment point can be easily identified.

The reattachment point for this mesh is observed to be at a distance of 5.73 m downstream of the step.

Mesh 5 – image (e)

Having refined all critical areas of the domain, further improvement in the solution may be achieved by using a smaller element size. Mesh topology does not need to be changed from this point forward.

For this study, the following settings were used to generate a fine mesh with similar topology as the Mesh 4:

- Global anisotropic mesh size: X, Y element size: 1 m, 0.2 m

- Recirculation zone mesh size: 0.05 m, Shear Layer Zone: 0.005 m

Wall Lower Downstream, Wall Lower Upstream, Wall Upper	**Wall - Step**
Mesh Type: Absolute Anisotropic	Mesh Type: Absolute
X, Y element size: 1 m, 0.2 m	Element size: 0.05 m
Boundary layer type: Full control	Boundary layer: No
Resolve type: Number of layers	
First element height: 0.0001 m	
Total layer height: 1.25 m, for Wall Lower Downstream: 0.625 m	
Layer growth rate: 1.2	
Boundary layer elements type: Tetrahedron	

The reattachment point for this mesh is observed to be at a distance of 5.73 m downstream of the step.

The table below gives a summary of the refinement process discussed above along with the mesh size generated in each case.

Table 12: Summary of the Mesh Refinement Process

Mesh Number	**Number of Nodes**	**Number of Elements**	**Refinement over previous mesh**
1	688	1,737	
2	4,508	12,981	Reduced element size
3	51,056	152,049	Refinement zone in recirculation zone

Mesh Number	Number of Nodes	Number of Elements	Refinement over previous mesh
4	68,014	202,329	Boundary layers on the walls
5	207,492	619,551	Reduced element size further

The table below shows the results achieved using five different meshes. Listed in the table is the variation of the location of the reattachment point for each mesh.

Table 13: Reattachment Point Location as Observed for Each of the Different Meshes

Mesh Number	Number of Nodes	Reattachment Point distance downstream of the step (m)
1	688	No reattachment happens
2	4,508	8.44
3	51,056	7.42
4	68,014	5.73
5	207,492	5.73

It is evident that refining the mesh in the shear layer, recirculation zone and the near wall area where the velocity gradients are highest leads to a change in the solution between meshes one through four. However, it is interesting to note that as the mesh is further refined between grids four and five there is no change seen in the solution. This implies that the solution error associated with grid refinement has been minimized and you have reached an asymptotically grid converged solution with the mesh density and topology utilized in mesh four. All of the critical regions have been refined at this stage and suitable mesh topology as well as sufficient mesh resolution has been achieved. Further refinement of the mesh did not change the solution quantity of interest. A review of the table above indicates that the separation location changed by almost 50 percent between mesh one and the final grid converged solution of mesh four. If no grid sensitivity study was performed the engineer could have made very poor conclusions about the location of the pressure tap.

Appropriate grid refinement is one of the most important aspects of obtaining accurate CFD results. Unfortunately, there is currently no reliable method of automatically generating appropriate meshes for all applications. Additionally, different numerical methods will require different levels of mesh refinement, that is, each code will be different. Ensuring the suitability of a mesh for a given application is your responsibility. The level of accuracy required from a CFD analysis depends on the desired use of the results. A conceptual design effort may be content with general depiction of the flow features, whereas a detailed design may require accurate determination of the flow field. Each quantity to be determined generally has its own accuracy requirement. Levels of effort invested into the CFD model

may vary according to the information required. It is often recommended that a minimum of three different meshes are constructed that cover a wide range of refinement and differ significantly in mesh density and resolution. For applications that require high levels of accuracy this number may be more depending on the number of mesh refinements it takes to achieve a grid converged solution. This allows you to evaluate the sensitivity of the solution quantity of interest to mesh density. Some CFD practitioners endorse the use of a Grid Convergence Index, or GCI, to estimate the level to which a simulation has reached a grid independent solution. More information about the GCI metric is available in the book by Patrick J. Roache, listed in reference [1].

Boundary Condition Sensitivity

This section discusses and illustrates the effects of the location of your model boundaries and the boundary conditions you apply on your CFD solution.

Boundary conditions specify the set of constraints on the solution of a CFD problem at the bounds of the modeled domain. There are many different types of boundary conditions that exist. The available set of constraints and conditions under which they should be used is specific to each CFD solver. However, you should consider the following points when applying boundary conditions.

Proximity of the boundary conditions with respect to the results of interest from the simulation.

In the discussion of the geometric sensitivity of the model it is mentioned that the boundaries must be modeled far away from the region of interest to prevent any interference on the results due to proximity to the boundary. To understand the possible reasons behind this interference it is important to understand that the boundary conditions impose a constraint on the solution. This constraint is an assumption about the behavior of the flow. If the boundary is placed too close to a location in which the results are of interest it is possible that the solution will be impacted drastically by the constraint applied at the boundary.

Realism of the constraints.

Determining appropriate boundary conditions to apply to CFD simulations is often very challenging. This is particularly true when considering inflow conditions. You should consider the suitability of the constraints that are applied to the simulation. For example, analysts need to evaluate the suitability of applying constant values for velocity and turbulence variables at inlet conditions as opposed to specifying boundary layer profiles. These types of decisions can have a significant impact on the realism of the model and introduce errors into the simulation. If there is a separation location downstream of the inlet, the thickness of the incoming boundary layer is a critical parameter and needs to be considered when assigning the location and constraints to apply.

Presence of reverse flow at outlet conditions.

When modeling complex flows it is important to take into consideration the behavior of the flow at the outlet. Most CFD codes do not apply any constraints at outflow conditions that force the flow to exit the domain. If the flow is recirculating and entering back into the domain it is indicative that the boundary is poorly located. If repositioning the boundary is not possible you need to ensure that any flow quantities re-entering the domain are properly bounded. For example, in the case of a turbulent thermal flow that is re-entering the domain at an outlet boundary, it is necessary to assign a constraint on the temperature and turbulence quantities that are being convected back into the domain. Each CFD solver may handle this situation differently, so it is up to you to determine if this situation is acceptable.

The Backward Facing Step Case

This section revisits each of the boundary condition related aspects discussed above in context of the backward facing step case.

- Proximity of the boundary conditions with respect to the results of interest from the simulation. If the boundaries in the backward step facing case are placed too close to the step it will have an adverse effect on the solution. If the outlet is too close to the step the presence of outlet constraints may interfere with the solution, preventing accurate representation of the recirculation zone and the reattachment point.
- Realism of the constraints. Two cases are presented to demonstrate the effect of inlet condition sensitivity on the solution of the backward facing step problem. The first case uses a constant value of velocity and turbulent eddy viscosity at the inlet. This case is derived from Mesh_4 from Mesh Sensitivity uses the same setup as mentioned in that section. The second case provides the inlet variables as a profile similar to the one used in an experimental study of the same problem. Figure 86 shows the profiles with boundary layer definition for the second case.

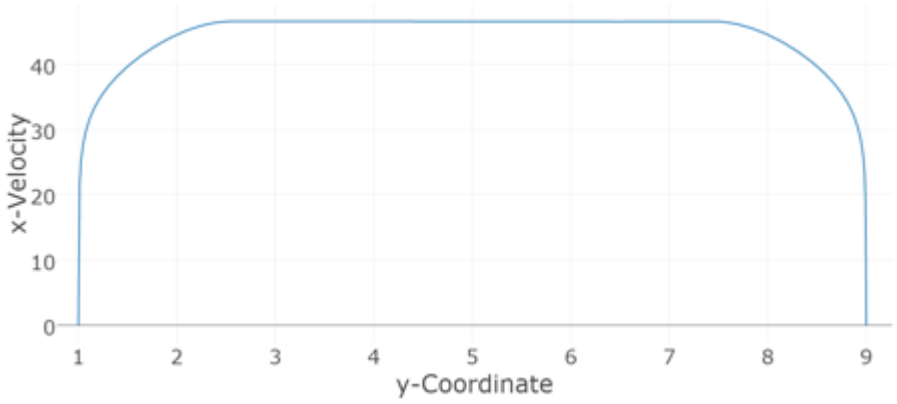


Figure 86: Inlet Velocity Profile

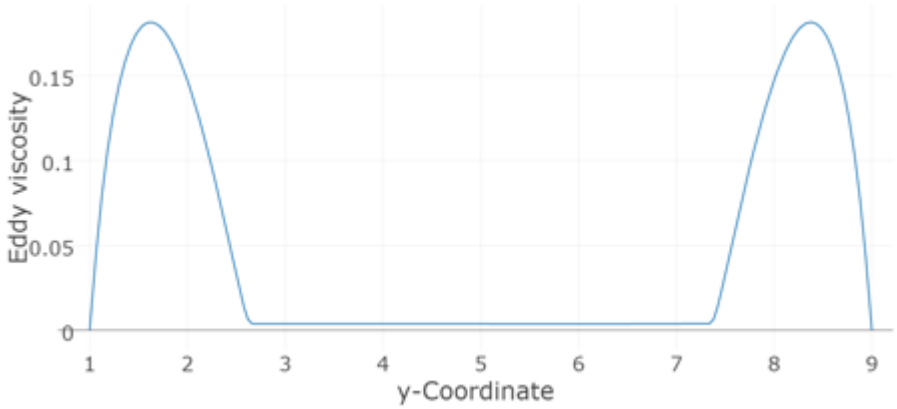


Figure 87: Inlet Turbulent Eddy Viscosity Profile for Inlet Boundary Condition Definition

The inlet velocity profile shows the presence of boundary layers at the edges while the eddy viscosity profile shows a concentrated presence of eddy viscosity in the boundary layer.

When comparing the results of the simulations run with the boundary conditions described above you can see that the profiled inlet condition reflects the experimental data much more accurately. The following plot compares the velocity profiles for both of the cases mentioned above with experimental measurements at the location four times of step height upstream of the step.

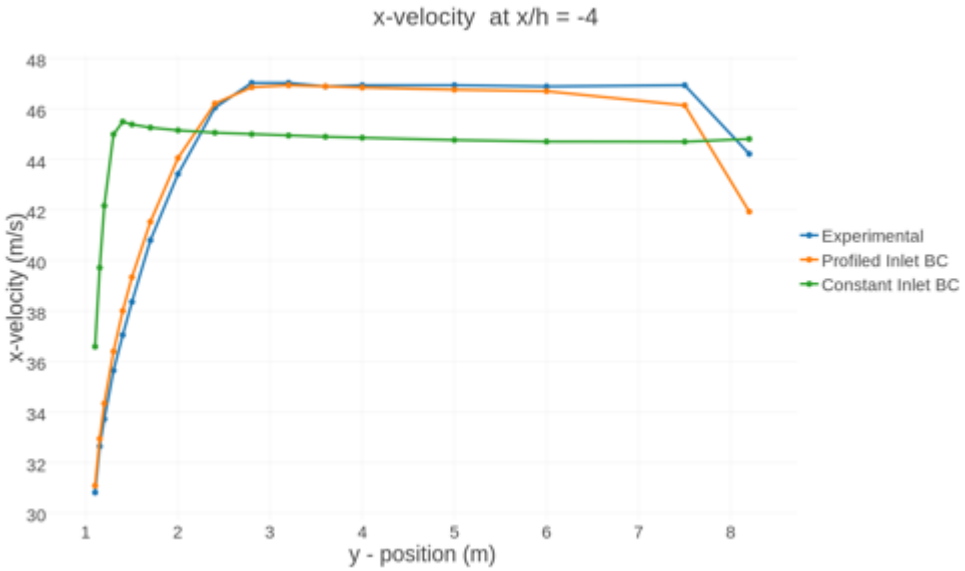


Figure 88: Velocity Profiles at x/h=-4 for Profiled and Constant Inlet Boundary Conditions Compared Against Experimental Data at the Same Location

You can see from the above comparison that using a boundary condition specification that resembles the real physics is much more likely to provide a correct solution to the problem.

The table below shows the difference between the reattachment length observed in the backward facing step problem when a profile similar to the one in the experiment is provided at the inlet, rather than using a constant value for the variables.

Table 14: Inlet Conditions Sensitivity for Backward Facing Step Simulation

Inlet condition type	**Reattachment Length (m)**
Constant	5.73
Velocity profile with boundary layer definition	5.96

Presence of reverse flow at outlet conditions. Figure 89 shows three candidate locations for an outlet boundary in the backward facing step model. The location represented by the white line falls in a region of recirculating flow and would cause inaccurate results as well as numerical difficulties due to the flow re-entering the domain. The location represented by the blue line falls outside of the recirculation zone, but there is still a significant amount of flow non-uniformity and falls fairly close to the recirculation region. To minimize the impact of the outflow conditions on the solution it should be moved further downstream, as shown by the green line.

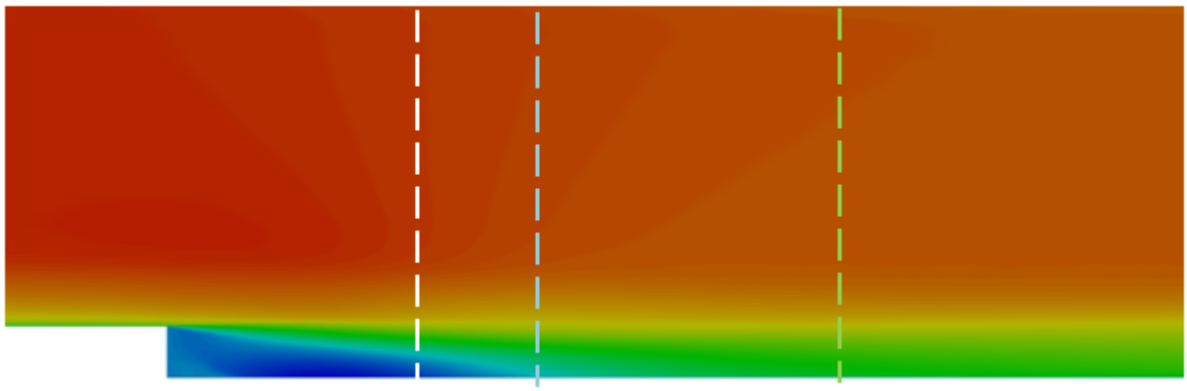


Figure 89: Candidate Locations for an Outlet Boundary on the Backward Facing Step Model

Physical Model Sensitivity

This section discusses and illustrates the effects of the mathematical models you choose to represent the problem physics, flow type, material properties, and so on, in your problem.

Setting up a CFD simulation requires the selection of a number of physical models. The purpose of the physical model is to mathematically represent the physics of a process that occurs in the simulation. One of the first steps in defining the simulation is the identification of the relevant physics that need to be modeled. Once this is done, a suitable model needs to be selected for each aspect of the simulation. Because various models exist for different characteristics of a simulation, it is important to investigate the sensitivity of the results to each of them. The following list mentions some of the possible physical models that need to be considered.

Flow Physics and Modeling

Flows can exhibit different physics under different conditions which must be carefully considered when setting up a simulation. A few variations of these are discussed below.

Steady and unsteady flows

In steady flows, flow properties, such as velocity and pressure, at a given point do not depend upon time.

In unsteady flows, flow properties at a point are functions of time.

Steady flow model is actually a simplification, and it is valid only when the time period over which the flow is observed is large. For modeling a steady flow, the time dependent terms in the Navier-Stokes governing equations are dropped.

Compressible and incompressible flows

Flows with Mach number up to 0.3 at room temperature are said to have weakly compressible behavior, and for the purpose of simulation can be considered incompressible flows, assuming the density to be constant.

Flows with Mach number greater than 0.3 are compressible where the fluid flow should account for the variation of fluid density.

Viscous and inviscid flows

Viscous flows are those in which frictional effects between fluid layers and fluid layers and surfaces are significant. All real flows are viscous flows.

Inviscid fluid flow model is a simplification that can be applied in cases when viscosity effects are expected to be small in comparison to inertial forces. High Reynolds number flows in large size domains where viscosity effects are limited to the small regions close to the boundaries can be considered as inviscid flows for bulk of the domain.

Single-phase and multi-phase flows

Single-phase flows are those in which the flow domain contains only one fluid material in a single state throughout the domain.

In case this condition is not satisfied the flow may be considered multiphase. Multiphase flows can contain a single material in different states or a mixture of two or more different materials in the same or different states. Examples of multiphase flows include solid or liquid particles immersed in a liquid medium or liquid droplets in gaseous medium.

Some common methods for dealing with multiphase flows are Volume of Fluid (VOF) method and Level Set method. Often the multiphase methods are not standalone methods to solve the complete flow field but track the interface between the existing phases.

Laminar and turbulent flows

Laminar flows are characterized by a highly ordered fluid flow where fluid flows in smooth non intersecting layers.

Turbulent flow is characterized by its irregular flow regime where fluid layers are not easily distinguished. Rapid changes are experienced in a turbulent flow regime. Instead of layers turbulent flows are made up of eddies of various sizes. These eddies constantly form and dissipate as they interact with each other in a rather chaotic manner.

The characterization between the laminar and turbulent nature of a flow is usually done on the basis of the Reynolds number (Re) for the flow. Pipe flows with Re < 2300 are laminar. Flows with Re > 4000 are considered turbulent. For flows with Re-value between these ranges the flow is said to be in a transition state.

If the problem is identified as a case of turbulent flow you must decide on the turbulence model to use for the problem. There are a number of choices available for this and not all models behave similarly in all situations. The following discussion is about making the choice of the turbulence model for the problem.

Turbulence Physics and Modeling

Turbulent flows have eddies of various scales. These eddies interact with each other causing rapid changes in flow properties. The highly chaotic nature of the flow makes modeling it a challenge.

The smallest turbulence eddies occurring in flows can have extremely small length scales. Direct simulation of these eddies requires mesh and temporal resolution which exceeds the computational capabilities of most modern computers. The computationally effective way is to model the behavior of the turbulent eddies in a time averaged flow field.

Turbulence models are used to predict the effects of turbulence in the fluid flow without resolving all of the small scales of motion that are associated with a turbulent flow. This reduces the amount of computing time necessary to obtain a solution to a fluid flow problem.

Most CFD solvers contain a broad selection of different turbulence models. The most commonly employed models in the industry are based on the Reynolds-averaged Navier-Stokes (RANS) equations.

Among RANS models there are eddy viscosity and Reynolds Stress formulations that need to be considered. Alternative approaches such as Detached Eddy Simulation (DES), Large Eddy Simulation (LES) and Direct Numerical Simulation (DNS) are also available in many CFD solvers. DNS approach, as the name suggests, is the approach that directly solves for eddies of all scales and does not involve any kind of modeling. It is mostly employed only for research purposes. Accuracy and computational requirements for these approaches is as follows.

RANS < DES < LES < DNS

When setting up a simulation it is your responsibility to select a suitable turbulence model for your application. Many models provide a reasonable level of accuracy for a broad range of applications while others are formulated specifically for certain flows. Among the most popular general purpose models are the one equation Spalart-Allmaras (SA) model and the two equation Shear Stress Transport (SST), and k-omega (k – ω) models.

Even while using general purpose turbulence models the choice of the model is important with respect to the problem being analyzed. Some models are better suited to some flows than others. The accuracy of a given turbulence model is highly case dependent and there are no established rules that can ensure correctness of the chosen model for a specific application. You must always proceed with caution and choose a turbulence model only after verifying its suitability to previously benchmarked problems of a similar nature.

In addition to the selection of a turbulence model it is also important to consider whether the flow is fully turbulent or transitional. If the flow is transitional it is important to include a transition model to properly account for this effect.

Wall modeling for turbulent flows	The use of a wall model is common practice when simulating high Reynolds Number viscous flows. The wall model relaxes the near wall meshing requirement to resolve the turbulent boundary layer and results in a reduction in required overall mesh count. Most wall models are formulated for simple boundary layer flows with no adverse pressure gradient. The application of these wall functions to non equilibrium boundary layers may impact accuracy.

Material Physics and Modeling

Setting up material properties is an important aspect of setting up a CFD simulation. Typically, CFD allows two types of materials, solids and fluids. Gases and liquids are both classified as fluids.

Setting up a solid model requires specification of density, specific heat and conductivity.

It is not required for a CFD model to have a solid material in the domain. These should be added only as required when an interaction between the fluid and the solid is present. Of the properties mentioned the specific heat and the conductivity are required only when energy equation is being solved.

Setting up a fluid model requires specification of the following properties:

- Density
- Viscosity
- Specific Heat
- Conductivity

- Viscoelastic properties, if fluid exhibits viscoelastic behavior
- Diffusivity for mass diffusion problems
- Porosity for porous materials

Of the properties mentioned above specific heat and the conductivity are required only when energy equation is being solved.

The material properties in a CFD simulation can be defined in a number of ways. The simplest definition is specifying the properties as a constant value. Once specified these constant values do not change during the simulation. For solids, the conductivity can also be specified as constant, but anisotropic. That is a different but constant conductivity in each normal direction.

For some cases, however, specifying a constant value of properties is not exactly representative of the actual material behavior. For example, for highly compressible flows specifying a constant density will lead to inaccurate results. For such flows it is prudent to use the ideal gas model for the fluid density, which uses the ideal gas law to calculate density at a given time step.

There are various options available to specify the different properties for the materials. You should choose the option that resembles the actual conditions as closely as possible. Some options are as follows:

- Linear or polynomial dependent on another variable (typically pressure, temperature or species)
- A user-defined function
- Ideal gas law for modeling density for compressible flows
- Boussinesq model for density in naturally convected flows in which small density variations encountered in such flows can be neglected in all but the buoyancy terms. This approach should not be used if large density variations are expected.

Heat Transfer Physics and Modeling

When temperature field variations are expected to have a significant impact on the flow properties, or the temperature distribution is of interest in the analysis, the energy equation must be included in the governing equations.

The heat transfer modes available are conduction, convection (natural and forced) and radiation.

For highly viscous flows viscous heating of the fluid must be taken into account. An example of flows where viscous heating can have a significant effect are the internal flows in microchannels.

For simulating processes where the fluid undergoes adiabatic compression compression heating of the fluid must be taken into account as well.

Prandtl number (Pr) is an important parameter when heat transfer within the boundary layer is being studied. Prandtl number can, in simple terms, be described as being proportional to the rate of growth of the velocity and thermal boundary layers. Smaller values of Pr indicate that thermal diffusion is more prominent than momentum diffusion. Thermal boundary layer grows faster than the velocity boundary layer.

When temperature effects are being studied and the flow is turbulent it is also critical to take into account the additional effect of turbulence on heat transfer properties. The turbulent Prandtl number is an important parameter in this case. Experiments show the average value to be 0.85 for most applications, but the actual value can vary between 0.7 and 0.9 depending on the specific problem.

If radiation effects are present in the problem you must choose the radiation parameters accordingly. In case of enclosure radiation view factor computation between the enclosure surfaces is critical. In case of external incident radiation, for example solar radiation, solar flux vector should be specified.

Chemistry Models

Selection of chemistry models can play a critical role in the results of simulations of combusting and reacting flows. When selecting a chemistry model you need to consider the rate of the chemical reaction in comparison to the times scales of other events in the flow. Some models are geared specifically towards turbulent flows where the turbulent time scale is much slower than the chemical reaction. Others are intended for the opposite scenario.

The Backward Facing Step Case

For the backward facing step case you will be solving the RANS equation, which includes terms for viscous effects. The flow field can be considered incompressible and single phase, as the Mach number is below 0.3 (it is actually about 0.13), and air is the only fluid present in the domain. The flow within the domain is turbulent, as the inlet Reynolds number is 36000. The flow can be solved for a steady state solution as the inlet conditions or the fluid properties are not time varying.

In the solver setups used in previous sections the backward facing step case was solved using the SA turbulence model. To illustrate the sensitivity of the solution to the selection of turbulence model, the results from two models, the one equation SA model and the two-equation SST turbulence model, are shown in the table below. For this application, experimental data shows a reattachment length of 6.26 m. The results show that the SST model has a lower deviation from the experimental observation indicating that it is more suitable for this flow than the SA model.

Table 15: Turbulence Model Sensitivity for Backward Facing Step Simulation

Turbulence model	Reattachment length (m)	Percent difference from experiment
SA	5.96	4.8 percent
SST	6.40	2.2 percent

Convergence Sensitivity

This section discusses and illustrates the effects of the level of convergence tolerance you provide to the solver, and also stresses on the need to look for other solution parameters before accepting a solution.

The convergence of the CFD solver gives an indication of how well it has solved the discretized equation set for the application of interest. Most CFD solvers measure convergence by monitoring the residuals, which represent the imbalance of terms in the governing equations at each iteration. As a general rule, the lower the residuals the lower the error due to imbalance in the terms in the equation. Some solvers also monitor the degree to which the solution is changing between iterations. As the simulation progresses, the residuals should fall and also the solutions to the equations should reach a stabilized value. It is a good practice to measure a solution quantity which may also capture the behavior of other solution variables to determine convergence. A converged solution means that the solver has done its job and produced the best solution it can for the given setup. However, that does not guarantee that the solution is correct in comparison to test or physics. The accuracy of the solution is a function of many other characteristics, some of which are described above.

CFD solver packages typically set default convergence criteria that are suitable for many applications. However, if high levels of accuracy are important it is necessary to monitor the impact of convergence on the solution for the application of interest. For steady state simulations this means monitoring the solution quantity of interest as the simulation is converged further. For transient applications it is also necessary to evaluate the convergence at each time step to see how it impacts the overall transient.

The Backward Facing Step Case

The backward facing step example is revisited to illustrate this point. Consider a steady state simulation of this flow. Figure 90 shows how the reattachment length estimated by the solver varies with each time step as the solver iterates over the equations to reach the desired residual levels. The target residual level has been set at 0.0001 for this solution. Using this criteria the converged solution for the location of reattachment point is determined to be 6.55 m downstream of the step.

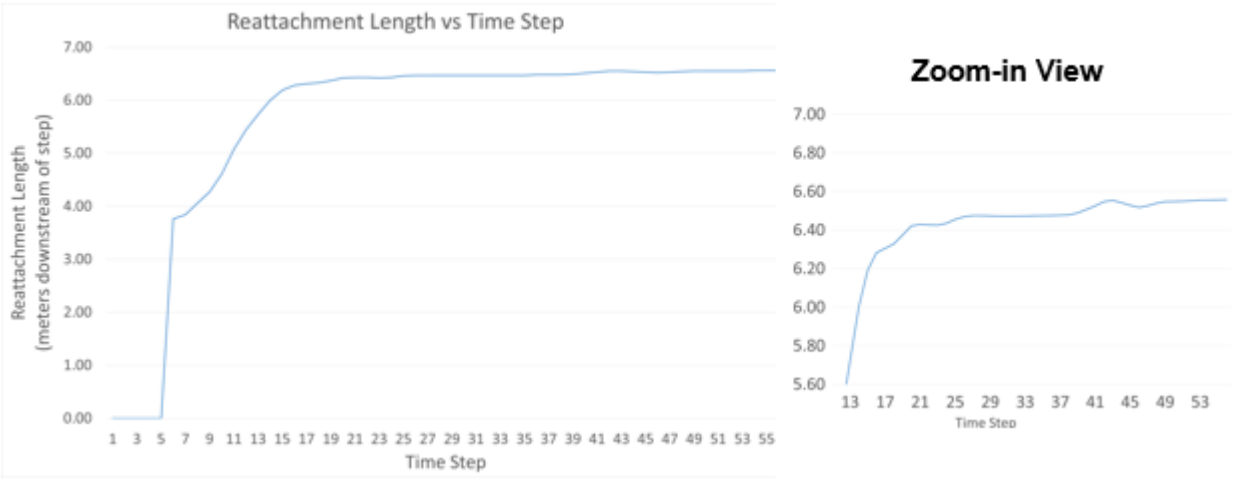


Figure 90: Convergence of Backward Facing Step Model

A closer inspection of the reattachment length convergence history in the second plot shows that the size of the separation zone does not stabilize until about time step 42. Therefore, the level of convergence necessary to obtain a high level of accuracy for the separation location corresponds to the levels corresponding to step 42 of the simulation. This value may or may not coincide with the solver's default convergence tolerance settings.

The solution can also be visualized to further inspect the convergence of the flow. The image below shows the solution at various time steps as the solution approaches convergence. Inspection of these figures indicates very little change in the flow after time step 20 when observed visually. However, as seen in the plot of reattachment length, the results change until stabilizing at time step 42. The decision of desired convergence tolerance thus depends on accuracy required from the CFD solution.

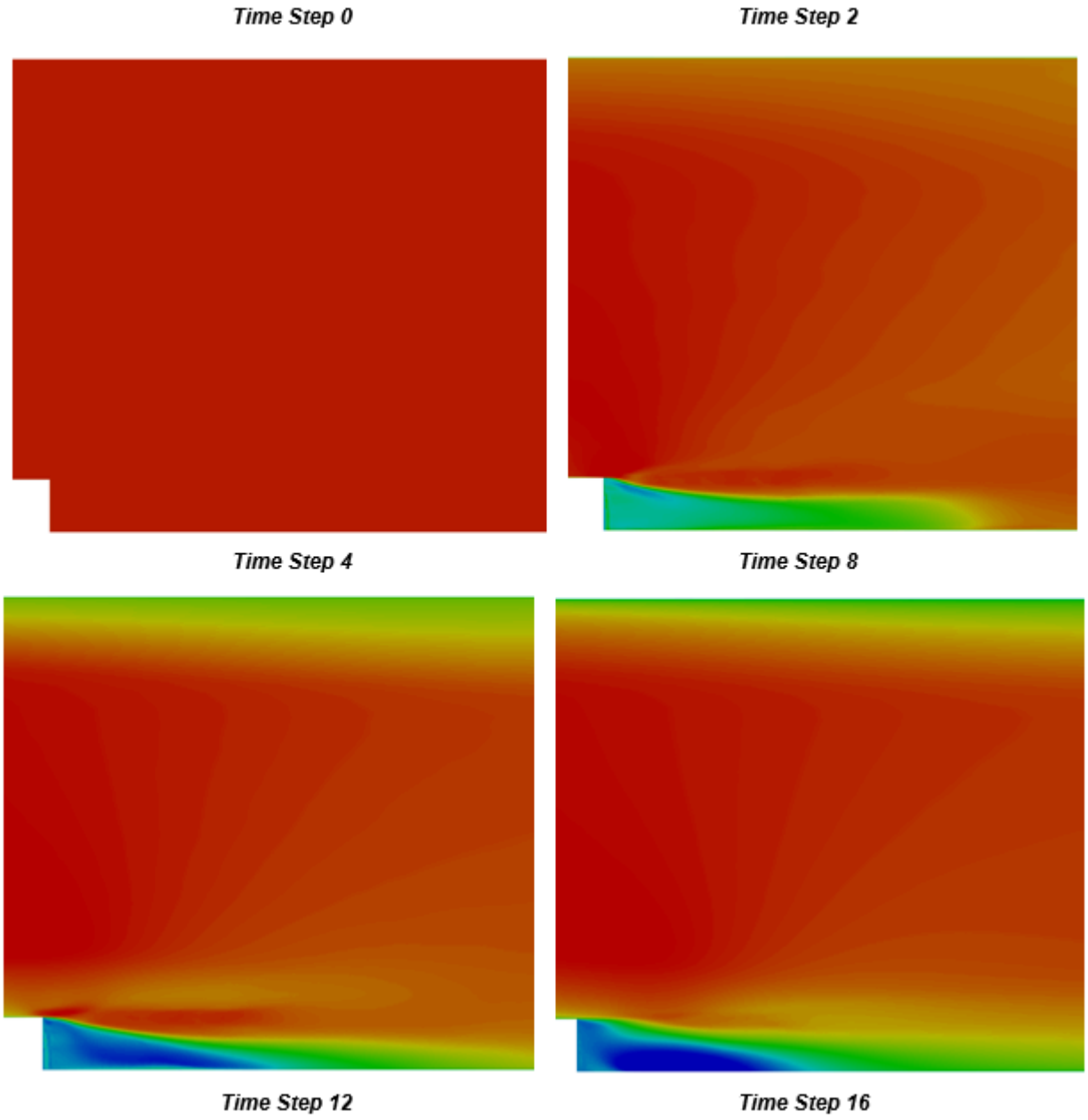


Figure 91: Convergence Behavior of the Backward Facing Step Simulation using the Shear Stress Transport (SST) Turbulence Model

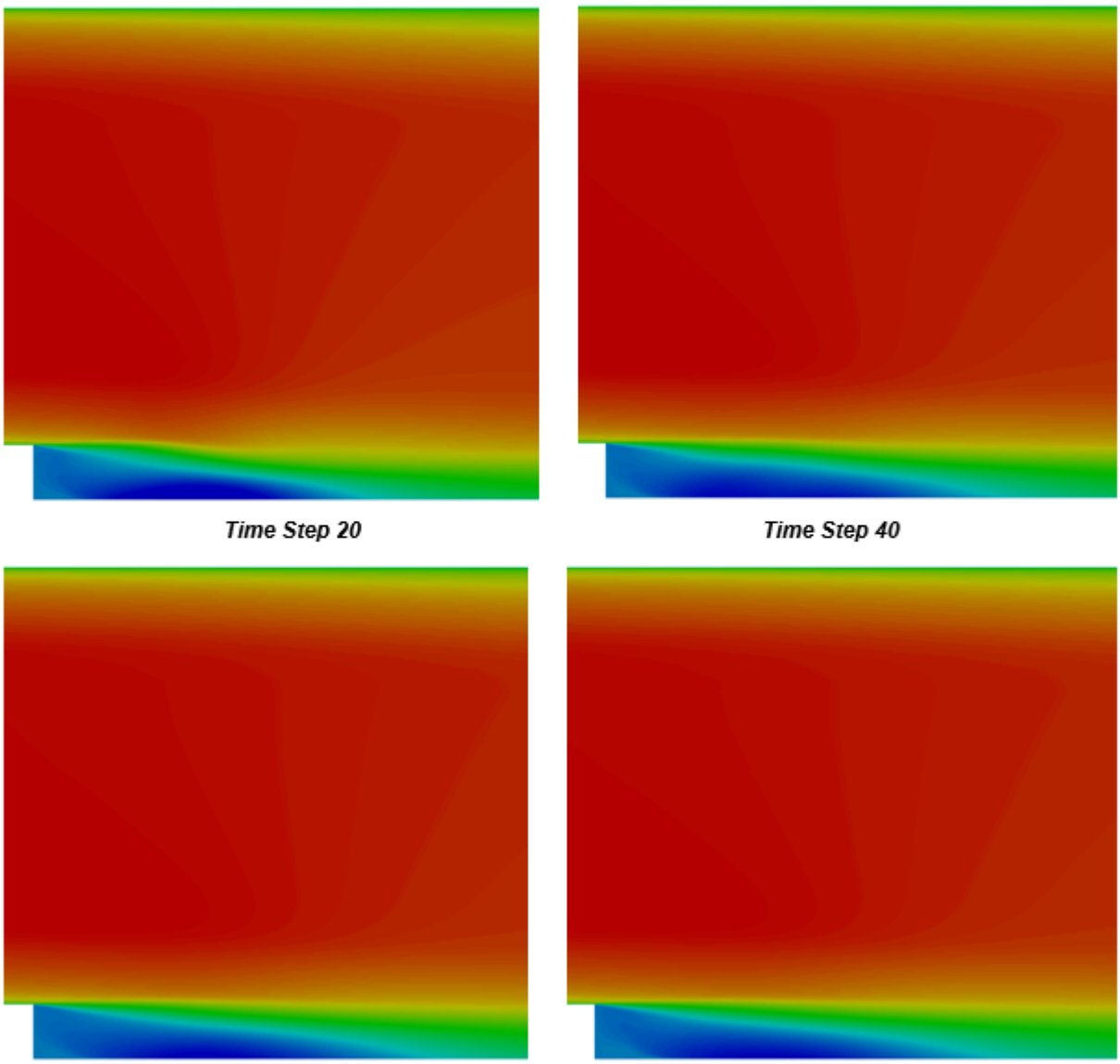


Figure 92: Convergence Behavior of the Backward Facing Step Simulation using the SST Turbulence Model

Each solver will be different and will perform differently on each application. It is up to you to investigate the proper convergence levels for your application. It should also be noted that convergence cannot be gauged by the residual metrics alone. Judging suitable convergence levels involves monitoring the convergence metrics, the solution quantity of interest, as well as looking at momentum and mass balances within the flow field when possible.

Conclusion

The modeling aspects described in this section are by no means complete for all of the flow simulations that can be imagined. This discussion is intended only to convey some of the common sources of variability among CFD model results.

A rigorous CFD analysis involves investigating all of the items described in this section. However, it is acknowledged that in practice it is necessary to rely on engineering judgment to reach a balance between producing results in a reasonable amount of time and performing a suitable level of quality assurance on the model. You should always consider the desired level of accuracy of the CFD model when deciding how much effort needs to be invested in quality assurance efforts.

An important observation to be made here is that this CFD solution may not be the true physical solution to the backward facing step problem. Over the course of this study, an attempt has been made to minimize the uncertainties by careful optimization of the problem setup with reference to each of the aspects discussed in the beginning of the section. It is meaningful now to go back to that discussion and recall that although these uncertainties can be minimized it is nearly impossible to eliminate them completely. At this stage, through optimization of all the critical aspects of your model, the uncertainties associated with these modeling aspects have been minimized. The solution at this stage can be deemed as the best possible CFD solution with respect to the aspects of the model discussed in this section. The difference observed between this numerical solution and the true physical solution can be attributed to the following factors:

- Mathematical modeling: The mathematical models which are used to represent the physical processes are not an exact representation of the physical phenomena. Even the best available models are based on certain approximations and simplifications.
- Input errors: These are the errors due to uncertainties of certain inputs such as the boundary condition specification. For example, even when the boundary conditions are specified as equivalent to the available experimental values there might be some interpolation errors when these values are projected on to the CFD domain.
- Uncertainties in experimental results and measurements: This discussion focused on a few guidelines to setup the CFD simulation model for a given problem. However, one key aspect that cannot be overlooked while comparing CFD results with experimental results of a problem is the uncertainties present in the experiment. An experiment, just like CFD analysis, is a controlled simulation of the problem. There are various sources of errors and uncertainties in the experiments such as uncertainties in the setup and environmental conditions in which the experiment is conducted, uncertanties in the measurement instruments used in the experiment and uncertainties in the post-processing equipment.

Due diligence is required to carry out an experiment with application of refined experimental methods to achieve a reliable set of results. A similar diligence, when applied to carrying out a CFD simulation, will help achieve a reliable set of results from CFD. An analysis completed with the above mentioned guidelines supporting the procedure will provide a result which for most purposes can be used as a substitute of the true value of the solution.

References

[1] Roache, P.J., Verification and Validation in Computational Science and Engineering, Hermosa Publishers, Albuquerque, New Mexico, 1998.

Intellectual Property Rights Notice

This Intellectual Property Rights Notice is exemplary, and therefore not exhaustive, of the intellectual property rights held by Altair Engineering Inc. or its affiliates. Software, other products, and materials of Altair Engineering Inc. or its affiliates are protected under laws of the United States and laws of other jurisdictions.

In addition to intellectual property rights indicated herein, such software, other products, and materials of Altair Engineering Inc. or its affiliates may be further protected by patents, additional copyrights, additional trademarks, trade secrets, and additional other intellectual property rights. For avoidance of doubt, copyright notice does not imply publication. Copyrights in the below are held by Altair Engineering Inc. or its affiliates. Additionally, all non-Altair marks are the property of their respective owners. If you have any questions regarding trademarks or registrations, please contact marketing and legal.

This Intellectual Property Rights Notice does not give you any right to any product, such as software, or underlying intellectual property rights of Altair Engineering Inc. or its affiliates. Usage, for example, of software of Altair Engineering Inc. or its affiliates is governed by and dependent on a valid license agreement.

Altair® HyperWorks®, a Design & Simulation Platform

Altair® AcuSolve® ©1997-2024

Altair® Activate® ©1989-2024

Altair® Automated Reporting Director™ ©2008-2022

Altair® Battery Damage Identifier™ ©2019-2024

Altair® CFD™ ©1990-2024

Altair Compose® ©2007-2024

Altair® ConnectMe™ ©2014-2024

Altair® DesignAI™ ©2022-2024

Altair® DSim™ ©2024

Altair® EDEM™ ©2005-2024

Altair® EEvision™ ©2018-2024

Altair® ElectroFlo™ ©1992-2024

Altair Embed® ©1989-2024

Altair Embed® SE ©1989-2024

Altair Embed®/Digital Power Designer ©2012-2024

Altair Embed®/eDrives ©2012-2024

Altair Embed® Viewer©1996-2024

Altair® e-Motor Director™©2019-2024

Altair® ESAComp® ©1992-2024

Altair® expertAI™ ©2020-2024

Altair® Feko® ©1999-2024

Altair® FlightStream® ©2017-2024

Altair® Flow Simulator™ ©2016-2024

Altair® Flux® ©1983-2024

Altair® FluxMotor® ©2017-2024

Altair® GateVision PRO™ ©2002-2024

Altair® Geomechanics Director™ ©2011-2022

Altair® HyperCrash® ©2001-2023

Altair® HyperGraph® ©1995-2024

Altair® HyperLife® ©1990-2024

Altair® HyperMesh® ©1990-2024

Altair® HyperMesh® CFD ©1990-2024

Altair® HyperMesh ® NVH ©1990-2024

Altair® HyperSpice™ ©2017-2024

Altair® HyperStudy® ©1999-2024

Altair® HyperView® ©1999-2024

Altair® HyperView Player® ©2022-2024

Altair® HyperWorks® ©1990-2024

Altair® HyperWorks® Design Explorer ©1990-2024

Altair® HyperXtrude® ©1999-2024

Altair® Impact Simulation Director™ ©2010-2022

Altair® Inspire™ ©2009-2024

Altair® Inspire™ Cast ©2011-2024

Altair® Inspire™ Extrude Metal ©1996-2024

Altair® Inspire™ Extrude Polymer ©1996-2024

Altair® Inspire™ Form ©1998-2024

Altair® Inspire™ Mold ©2009-2024

Altair® Inspire™ PolyFoam ©2009-2024

Altair® Inspire™ Print3D ©2021-2024

Altair® Inspire™ Render ©1993-2024

Altair® Inspire™ Studio ©1993-2024

Altair® Material Data Center™ ©2019-2024

Altair® Material Modeler™ ©2019-2024

Altair® Model Mesher Director™ ©2010-2024

Altair® MotionSolve® ©2002-2024

Altair® MotionView® ©1993-2024

Altair® Multi-Disciplinary Optimization Director™ ©2012-2024

Altair® Multiscale Designer® ©2011-2024

Altair® newFASANT™ ©2010-2020

Altair® nanoFluidX® ©2013-2024

Altair® NLVIEW® ©2018-2024

Altair® NVH Director™ ©2010-2024

Altair® NVH Full Vehicle™ ©2022-2024

Altair® NVH Standard™ ©2022-2024

Altair® OmniV™ ©2015-2024

Altair® OptiStruct® ©1996-2024

Altair® physicsAI™ ©2021-2024

Altair® PollEx™ ©2003-2024

Altair® PollEx™ for ECAD ©2003-2024

Altair® PSIM™ ©1994-2024

Altair® Pulse™ ©2020-2024

Altair® Radioss® ©1986-2024

Altair® romAI™ ©2022-2024

Altair® RTLvision PRO™ ©2002-2024

Altair® S-CALC™ ©1995-2024

Altair® S-CONCRETE™ ©1995-2024

Altair® S-FRAME® ©1995-2024

Altair® S-FOUNDATION™ ©1995-2024

Altair® S-LINE™ ©1995-2024

Altair® S-PAD™ ©1995-2024

Altair® S-STEEL™ ©1995-2024

Altair® S-TIMBER™ ©1995-2024

Altair® S-VIEW™ ©1995-2024

Altair® SEAM® ©1985-2024

Altair® shapeAI™ ©2021-2024

Altair® signalAI™ ©2020-2024

Altair® Silicon Debug Tools™ ©2018-2024

Altair® SimLab® ©2004-2024

Altair® SimLab® ST ©2019-2024

Altair® SimSolid® ©2015-2024

Altair® SpiceVision PRO™ ©2002-2024

Altair® Squeak and Rattle Director™ ©2012-2024

Altair® StarVision PRO™ ©2002-2024

Altair® Structural Office™ ©2022-2024

Altair® Sulis™ ©2018-2024

Altair® Twin Activate® ©1989-2024

Altair® UDE™ ©2015-2024

Altair® ultraFluidX® ©2010-2024

Altair® Virtual Gauge Director™ ©2012-2024

Altair® Virtual Wind Tunnel™ ©2012-2024

Altair® Weight Analytics™ ©2013-2022

Altair® Weld Certification Director™ ©2014-2024

Altair® WinProp™ ©2000-2024

Altair® WRAP™ ©1998-2024

Altair® HPCWorks®, a HPC & Cloud Platform

Altair® Allocator™ ©1995-2024

Altair® Access™ ©2008-2024

Altair® Accelerator™ ©1995-2024

Altair® Accelerator™ Plus ©1995-2024

Altair® Breeze™ ©2022-2024

Altair® Cassini™ ©2015-2024

Altair® Control™ ©2008-2024

Altair® Desktop Software Usage Analytics™ (DSUA) ©2022-2024

Altair® FlowTracer™ ©1995-2024

Altair® Grid Engine® ©2001, 2011-2024

Altair® InsightPro™ ©2023-2024

Altair® Hero™ ©1995-2024

Altair® Liquid Scheduling™ ©2023-2024

Altair® Mistral™ ©2022-2024

Altair® Monitor™ ©1995-2024

Altair® NavOps® ©2022-2024

Altair® PBS Professional® ©1994-2024

Altair® PBS Works™ ©2022-2024

Altair® Simulation Cloud Suite (SCS) ©2024

Altair® Software Asset Optimization (SAO) ©2007-2024

Altair® Unlimited™ ©2022-2024

Altair® Unlimited Data Analytics Appliance™ ©2022-2024

Altair® Unlimited Virtual Appliance™ ©2022-2024

Altair® RapidMiner®, a Data Analytics & AI Platform

Altair® AI Hub ©2023-2024

Altair® AI Edge™ ©2023-2024

Altair® AI Cloud ©2022-2024

Altair® AI Studio ©2023-2024

Altair® Analytics Workbench™ ©2002-2024

Altair® Graph Lakehouse™ ©2013-2024

Altair® Graph Studio™ ©2007-2024

Altair® Knowledge Hub™ ©2017-2024

Altair® Knowledge Studio® ©1994-2024

Altair® Knowledge Studio® for Apache Spark ©1994-2024

Altair® Knowledge Seeker™ ©1994-2024

Altair® IoT Studio™ ©2002-2024

Altair® Monarch® ©1996-2024

Altair® Monarch® Classic ©1996-2024

Altair® Monarch® Complete™ ©1996-2024

Altair® Monarch® Data Prep Studio ©2015-2024

Altair® Monarch Server™ ©1996-2024

Altair® Panopticon™ ©2004-2024

Altair® Panopticon™ BI ©2011-2024

Altair® SLC™ ©2002-2024

Altair® SLC Hub™ ©2002-2024

October 7, 2024

Technical Support

Altair's support resources include engaging learning materials, vibrant community forums, intuitive online help resources, how-to guides, and a user-friendly support portal.

Visit Customer Support to learn more about our support offerings.

Index

Special Characters

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

V

W