Appendices

Appendix A: CFD procedures

In general, a CFD analysis is typically composed of three phases: pre-processing, solving and post-processing. Pre-processing consists of importing the building geometry from CAD, creating a computational domain, mesh generation, defining boundaries and initial conditions, and setting numerical controls. Once the computational conditions have been set, the analysis will proceed using commercial CFD codes such as STAR-CD and STAR CCM+ *(CD-Adapco)*, FLUENT and CFX *(ANSYS, Inc)*, OpenFOAM, or COVERFLOW. The results can take the form of color plots, contour plots, and numerical reports in the post-processing phase. The specific process involved in performing a CFD analysis is outlined in detail in *NPARC Alliance CFD Verification and Validation* (2012):

1) Formulate the Flow Problem
2) Model the Geometry and Flow Domain
3) Establish the Boundary and Initial Conditions
4) Generate the Grid
5) Establish the Simulation Strategy
6) Establish the Input Parameters and Files
7) Perform the Simulation
8) Monitor the Simulation for Completion
9) Post-Process the Simulation to get the Results
10) Make Comparisons of the Results
11) Repeat the Process to Examine Sensitivities
12) Document

Appendix B: CFD fundamentals

1. Governing equations

According to the book *Computational Fluid Dynamics: The Basics With Applications* (Anderson, 1995), CFD utilizes the fundamental governing equations of fluid dynamics as represented by the continuity, momentum, and the energy equations. These are based on three fundamental physical principals:

(1) Mass is conserved;

(2) F = ma (Newton's second law); and

(3) Energy is conserved (Anderson, 1995, p.38).

$$\frac{\partial u_i}{\partial x_i} = 0$$

Equation 1- The conservation of mass

$$\frac{\partial(\rho u_i)}{\partial t} + \frac{\partial(\rho u_j u_j)}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\mu \frac{\partial u_j}{\partial x_j}\right) + S_i$$

Equation 2 - A momentum equation

These are summarized in the Navier–Stokes equations for incompressible flow:

where x_i (i = 1, 2, 3) are the three coordinate directions, u_i (i = 1, 2, 3) are the velocities in these directions, p is pressure, ✓ is density and ⚲ is viscosity. Equation 1 expresses the conservation of mass, while Equation 2 is a momentum equation where the first term on the left-hand side represents the variation with time and the second is the convection term; on the right-hand side the terms are the pressure gradient, a diffusion term and the source term, respectively (Wainwright & Mulligan, 2004). In general, the Navier-Stokes equations comprise a system of nonlinear partial differential equations and provide the highest level approximations for the flow physics approaching the continuum-mechanics based flow regime (Veress & Rohڑcs, 2012).

2. Initial conditions

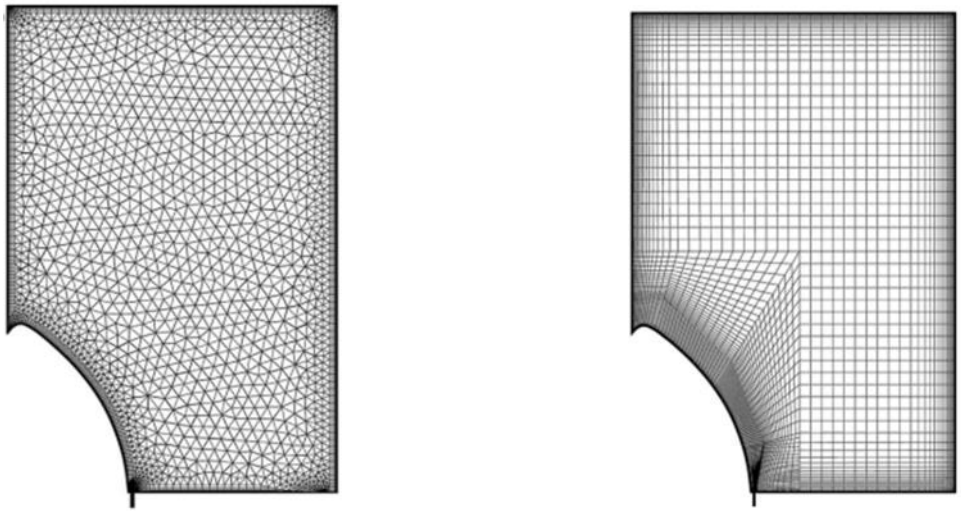
At an early stage of the CFD process, the initial conditions must be determined. These conditions are sensitive to the selection of the turbulence model to be utilized. The number of iterations is set up for steady flow problems, but for the case of unsteady flow problems, time steps are needed. A more detailed explanation of how the turbulence model is selected is provided below in Section 5.

3. Grid generation

The geometric domain to be modeled using CFD must be divided up or *discretized* into cells. The combination of all the cells is regarded as the grid or computational domain. For example, three dimensional cells are usually hexahedral or tetrahedral in shape (Scott et al., 2006). The process of determining the grid is referred to as *grid*, or *mesh generation*. Anderson (1995) pointed out that the grid generation is a significant consideration in the CFD process and an appropriate determination of the type of grid is essential if a valid numerical solution is to be obtained. This process has a significant impact on the efficiency and accuracy of CFD, as the element or cell shape and size influences both the computation speed and numerical accuracy (Kortelainen, 2009).

Based upon the connectivity of the mesh or on the type of elements two types of meshes, structured and unstructured, have been defined as shown in Figure 1. According to Bern and Plassmann (2000), all the interior vertices of a structured mesh are topologically alike and the vertices of an unstructured mesh may be arbitrarily different for local neighborhoods. They also noted that a hybrid mesh can be made up of a number of small structured meshes in an overall unstructured pattern, as shown in Figure 2. While a structural mesh is simple and easy to access, an unstructured mesh generally provides a better fit for complicated domains and offers more convenient mesh adaptivity (Oliveira et al., 2008). A high-quality hybrid mesh can take advantage of both these approaches. In addition, both structured and unstructured meshes must

remove unnecessary details in the geometry to ensure robustness and mesh quality (Oliveira et al., 2008).



a. Unstructured mesh b. Structured mesh

Figure 1 - The two different mesh types (Bosbach et al. ,2006)

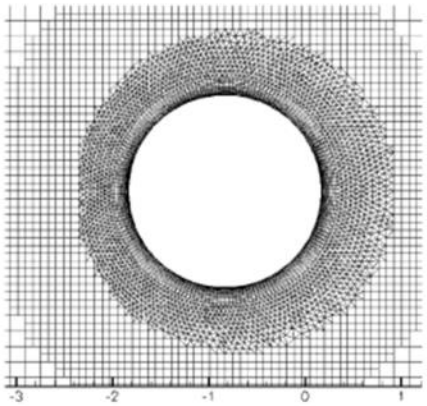


Figure 2 - Close-up view of a hybrid computational grid (Zhang & Wang, 2004)

4. Discretization

Once the cell types have been determined, the next step is to divide the flow domain into small computational volumes called *elements* or *cells* that cover the domain completely but do not

overlap one another (Kortelainen, 2009). Depending upon the type of discretization method utilized, computational grids can be either structured or unstructured. Anderson defined *discretization* as:

the process by which a closed-form mathematical expression, such as a function or a differential or integral equation involving functions, all of which are viewed as having an infinite continuum of values throughout some domain, is approximated by analogous (but different) expressions which prescribe values at only a finite number of discrete points or volumes in the domain (1995, p.125).

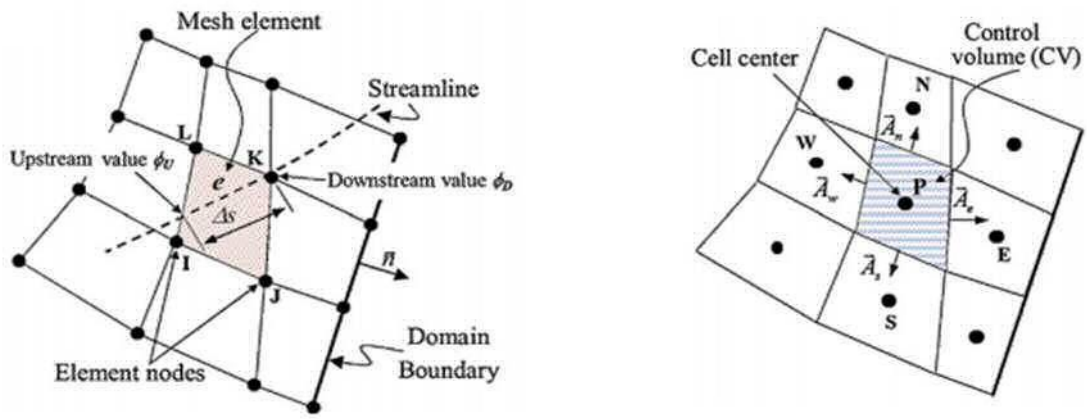
Various numerical discretization methods have been developed for the solution of non-linear Partial Differential Equations (Uddin, 2008). According to Veress and Rohȷcs (2012), three discretization methods are normally used in commercial CFD codes: the Finite Difference method, the Finite Element method and the Finite Volume method. Other discretization methods such as the Spectral, Boundary element, Vorticity type and Lattice gas or Lattice Boltzmann methods have been rarely, if ever, used for this purpose.

The characteristics of the three most frequently used methods can be summarized as follows (Udoewa & Kumar, 2012; Veress & Rohȷcs, 2012):

1) The *Finite Difference* method (FDM) is the most traditional and oldest method of the three and is only suitable for structured grids. The mesh size and properties such as stretch ratio, aspect ratio and skewness have a significantly impact on its accuracy. This method is not widely used because of the geometric limitations it imposes on applications, but it is the easiest discretization method of the three.
2) The *Finite Element* method (FEM) provides greater flexibility for dealing with complex geometries than the Finite Difference method. The mesh in the Finite Element method can be either structured or unstructured. While this method is more stable than the Finite Volume method, it requires more computing memory than the Finite Volume method.

3) The *Finite Volume* method (FVM) calculates the partial differential equations over finite volumes created as parts of one or more cells that can be either overlapped or non-overlapped. In this method, it is possible to modify the shape and location of the control volumes to allow greater freedom in the choice of the functional representation of the flow field. The Finite Volume Method permits the use of irregular grids while retaining computational simplicity.

As shown in Figure 3, while FEM separates the continuum region of interest into a number of simply shaped regions, FVM subdivides the domain of interest into a finite number of contiguous control volumes based on a grid (Molina-Aiz et al., 2010).



a. FEM b. FVM

Figure 3 - Structured meshes for the two main discretization methods (Molina-Aiz et al., 2010)

5. Turbulence models

A fundamental phenomenon in this field of study is turbulence. Generally, turbulence is a complicated phenomenon that occurs over a wide range of time scales and length scales. Most CFD researchers deal with this by selecting a certain length scale and time scale for the range of turbulence anticipated since turbulence acts over such a wide range of length scales and its periodicity manifests over similarly wide ranges of time periods (Udoewa & Kumar, 2012). Udoewa and Kumar explained that resolving turbulence means computing and calculating

turbulent quantities such as vorticity, velocities and pressures in turbulent regions and noted that modeling turbulence creates the effect of fully resolved turbulence on unknown values such as velocity and pressure based on the effect of the equation to which an expression is added. Turbulence modeling is difficult and represents the most important challenge facing CFD (Wainwright & Mulligan, 2004) and successfully modeling turbulence greatly enhances the quality of numerical simulations (Sodja, 2007).

There are three popular approaches for the analysis of a turbulent flow problem, namely *Direct Numerical Simulation* (DNS), *Large Eddy Simulation* (LES) and models based on *Reynolds Averaged Navier-Stokes* (RANS) equations (Udim, 2008). The characteristics of these three approaches are summarized by Wainwright & Mulligan in their book *Environmental Modeling: Finding Simplicity in Complexity* as follows:

For DNS, it is possible to predict all the eddy structures from the large ones down to the smallest but it requires significant computing resources for practical flows. In LES approach, it uses a length scale to differentiate between larger and smaller eddies. The larger eddies are resolved directly but the smaller eddies are not predicted directly. The smaller eddies are accounted for through a subgrid-scale model. When applying this approach, appropriate filter or grid size should be considered in order to achieve accurate results. However it is less expensive than DNS, LES still requires high computing power and needs fine grids for practical flows. On the other hand, the most widely used approach is the turbulence models based on RANS equations. The most popular option is the k - model which is usually the default option in CFD software. *k* represents the kinetic energy in the turbulent fluctuations and represents the rate of dissipation of *k* (2004, pp.338).

In addition, Udim (2008) mentioned that the RANS approach predicts only the mean flow field and can be applied to achieve a sufficiently accurate prediction in many cases. However, this does fail to represent the true flow physics.

In sum, although both the DNS and LES approaches provide better results, they require far greater computer resources than turbulence models based on RANS methods. Furthermore, there are a number of different versions of turbulence models based RANS methods now available commercially (Udoewa & Kumar, 2012).

6. Solution algorithm

CFD simulations are constructed around numerical algorithms that can deal with fluid flow problems. For example, consider the numerical algorithms used to resolve the coupling that arises in the solution of Navier-Stokes equations between velocity, density and pressure (Moukalled & Darwish, 2000). The SIMPLE (Semi-Implicit Method for Pressure Linked Equations) type of algorithm is commonly used in CFD studies (Patankar & Spalding, 1972), but over the years, a number of modified versions of the SIMPLE algorithm have been suggested including the SIMPLER, SIMPLEC and PISO algorithms (Moukalled & Darwish, 2000).

Versteeg and Malalasekera (1995) compared those four algorithms in their book *An Introduction to Computational Fluid Dynamics: The Finite Volume Method*. The SIMPLE algorithm was originally proposed by Patankar and Spalding (1972) and is essentially a guess-and-correct procedure for the calculation of pressure on a staggered grid arrangement. This method is relatively straightforward and has been successfully used in numerous CFD procedures. The SIMPLER (SIMPLE-Revised) algorithm is an improved version of the SIMPLE algorithm that utilizes pressure corrections only to correct velocity fields. Consequently, this algorithm is very effective in calculating the pressure field correctly and solving the momentum equations while at the same time reducing the computing resources required. The book also introduces the SIMPLEC and PISO algorithms. The SIMPLEC (SIMPLE-Consistent) algorithm follows the same steps as the SIMPLE algorithm but defines the coefficients in the pressure-correlation equation differently. The PISO algorithm, where the acronym PISO stands for Pressure Implicit with Splitting of Operators, and implements a pressure℘velocity calculation procedure for the non-iterative computation of unsteady compressible flows. Versteeg and Malalasekera consider

that although the SIMPLEC and PISO algorithms have proved that they are as efficient as the

SIMPLER algorithm for certain types of flows, it is not clear whether they are better overall than the SIMPLER algorithm.

7. Boundary conditions

After choosing a suitable mathematical model for the physical phenomenon, boundary conditions must be chosen. For boundary-value problems, it is essential to assign values along the boundary of the domain that will permit the problem to be solved throughout the entire 2D or 3D space. There are two types of boundary conditions, namely *Dirichlet* and *Neumann* boundary conditions. For a Dirichlet boundary condition, the fixed value of the unknown on the boundary is specified for some flow problems; this is referred to as a direct boundary condition. In contrast, when the value of a derivative of the unknown is specified, this is called a Neumann or natural boundary condition. These boundaries are spatial boundaries (Udoewa & Kumar, 2012).

According to Wainwright & Mulligan (2004), it is common for the velocity to be fixed at an inlet and outlet. For an inlet, turbulence quantities must be correctly specified and inlet profiles consistent with the definition of roughness along the boundaries of the domain. For the outlet condition, careful consideration must be given to the question of whether the outflow should equal the inflow or whether the flow profile should be uniform along the stream direction in order to avoid poor convergence or solutions that are not physically feasible.

8. Numerical parameters for controlling the calculation

Numerical parameters must be specified in order to control the calculation and reduce the computational resources needed. These parameters include the relaxation factor, monitoring residuals, different numerical schemes, the number of iterations for steady flow, number of time steps for unsteady flow, and the choice between single and double precis

Appendix C: Definitions of several CFD terms

CFD is a computer based mathematical modeling tool that is based upon the fundamental governing equations of fluid dynamics. However, terms within CFD are often not obvious to non-experts to that field (Souza, 2005). Thus, it is important to introduce and describe CFD terms to them. While the most commonly used parameters of CFD will be introduced in this study, several terms of CFD simulation will be introduced briefly in this section.

1. Aspect ratio

This is ratio of longest edge length to shortest edge length and it is a measure of quality for a computational grid. In order to avoid convergence problems, the cell aspect ratios should be kept as small as possible (Souza, 2005; STAR-CCM+ user guide 8.04).

2. Boundary conditions

The spatial or temporal specification of variable values or behavior to produce an unique solution (Souza, 2005).

3. Convergence criteria

Through the iterations, the magnitude of the velocity divergence is reduced below some absolute numerical value that tends to a single answer (Souza, 2005)

4. Divergence

The progression of a numerical scheme away from any single solution (Souza, 2005).

5. Grid / mesh

This is the outcome of discretizing the computational domain into a number of elements or cells defining the discrete points at which the numerical solution is computed (Souza, 2005).

6. Grid density

This term means the number of cells in a given volume. A higher grid density should be applied in regions of interest where the variables change rapidly so that their gradients can be computed and represented accurately. In the case of lower grid density, it is generally used where the solution is changing less in order to reduce the computational resources (Souza, 2005).

7. Invalid cell

When each cell interior face belongs to exactly two cells, this is considered as valid cell. If the arrays storing the cell connectivity and vertices have entries that make reference to faces or vertices that fall outside the expected range of values, invalid cells occur (STAR-CCM+ user guide 8.04).

8. No-slip wall

This term means that the fluid velocity at the wall equals the wall velocity. The wall can rest or a tangential wall velocity in the form of a velocity vector with respect to the laboratory coordinate system can be specified (STAR-CCM+ user guide 8.04).

9. Schmidt number

A dimensionless number that is the ratio of kinematic viscosity to diffusivity (STAR-CCM+ user guide 8.04).

10. Skewness

This term is an important measure of cell quality and it is designed to reflect whether the cells on either side of a face are formed in such a way as to permit diffusion of quantities without these quantities becoming unbounded (STAR-CCM+ user guide 8.04).

11. Slip wall

This term means that the fluid velocity at the wall is different with the wall velocity (STAR-CCM+ user guide 8.04).

12. Structured grid

This grid forms a regular pattern. Its grid lines are continuous across the domain and are usually aligned with the co-ordinate directions. For this grid, hexahedra in three dimensions or quadrilaterals in two dimensions are included (Souza, 2005).

13. Unclosed cell

Generally a topologically closed volume cell consists of edges that are connected to exactly two faces, and faces with normals pointing outwards. When a cell is missing a face, or the outward normals are inconsistent, it is regarded as unclosed mesh (STAR-CCM+ user guide 8.04).

14. Unstructured grid

Unlike structured grid, this grid forms an irregular pattern. This grid allows highly complex geometries to be modeled with relative ease compared to structured grids (Souza, 2005).

15. Wall functions

This function has been used to describe the effects of turbulent boundary layers in the region adjacent to a wall, without resolving details of the near wall flow and eliminating the need for high grid resolution in the viscous sub-layer (Souza,

Appendix D: The commercial CFD codes

Many commercial CFD codes such as STAR-CD and STAR CCM+ *(CD-Adapco)*, FLUENT and CFX *(ANSYS, Inc.)* or OpenFOAM have been used to simulate physical processes. These commercial CFD packages include pre-processors to construct grids, and apply initial conditions, processing algorithms that permit segregated or coupled, steady or unsteady state flows and a variety of turbulence models, and post-processors that display results in a manner that enhance interpretation. Among commercial codes, the characteristics of FLUENT and STAR-CCM+ have been reviewed.

1. FLUENT

FLUENT is a most commonly used commercial CFD code. FLUENT solves the conservative form of the Navier-Stokes equations using the finite volume method on an unstructured, non-orthogonal, curvilinear coordinate grid system. Combinations of elements in a variety of shapes are permitted, such as quadrilaterals and triangles for 2-D simulations and hexahedra, tetrahedra, polyhedra, prisms and pyramids for 3-D simulations. Meshes can be created using mesh generators (GAMBIT) or by several third-party CAD packages. FLUENT allows the following file formats for importing are summarized in table 1 (*ANSYS* FLUENT, 12.0 User's Guide).

Table 1 - File formats for importing into FLUENT (*ANSYS* FLUENT, 12.0 User's Guide)

ABAQUS .inp, .fil, and .odb files
Mechanical APDL .inp, .cdb, .rst, .rmg, and .rfl files.
ANSYS CFX .def and .res files
CGNS files
EnSight files
ANSYS FIDAP Neutral files
GAMBIT files
HYPERMESH ASCII files.
NASTRAN Bulk Data files
PATRAN Neutral files.
PLOT3D mesh and result files.

Turbulent flow models including several popular k–epsilon and k–omega models, and the Reynolds stress model (RSM) are available. Moreover, large eddy simulation (LES) and the more economical detached eddy simulation (DES) turbulence models can be employed (*ANSYS* FLUENT, 12.0 User's Guide).

2. STAR-CCM+

STAR-CCM+ includes the general characteristics of FLUENT. In addition, STAR-CCM+ has a single environment for whole CFD process such as solid modeling, mesh generation, post-processing, etc. Moreover, STAR-CCM+ has a comprehensive suite of geometry creation and preparation tools. For meshing, STAR-CCM+ employs the technology of hexahedral, dodecahedral and arbitrary polyhedral cells and produces boundary trimmed meshes that conform to geometry, as well as high quality prism layer meshes for accuracy. The following file formats that can be imported into STAR-CCM+ are summarized in table 2 (STAR-CCM+ user guide 8.04