APPLIED COMPUTATIONAL FLUID DYNAMICS

Edited by **Hyoung Woo Oh**

Applied Computational Fluid Dynamics
Edited by Hyoung Woo Oh

Notice
Statements and opinions expressed in the chapters are these of the individual contributors and not necessarily those of the editors or publisher. No responsibility is accepted for the accuracy of information contained in the published chapters. The publisher assumes no responsibility for any damage or injury to persons or property arising out of the use of any materials, instructions, methods or ideas contained in the book.

Contents

Preface

This book is intended for use as a reference text to help advanced scientists and research engineers solve numerous and diverse fluid flow problems using computational fluid dynamics (CFD). Each chapter has been contributed by the practiced experts in the field of fluid dynamics. This material covers a wide spectrum of CFD applications involving flow modeling in sedimentation tank, greenhouse environment, hypersonic aerodynamics, computational scheme, cooling systems design, photochemical reaction engineering, atmospheric reentry problem, fluid-structure interaction (FSI), atomization, hydraulic component design, air conditioning system, and industrial applications of CFD.

Dr. Hyoung Woo Oh
Department of Mechanical Engineering,
Chungju National University,
Chungju,
Korea

Computational Fluid Dynamics

Victor Udoewa[1] and Vinod Kumar[2]

[1]George Washington University,
USAID Development Engineer,
AAAS Science & Technology Policy Fellow, AAAS, 2009-2011,
[2]Mechanical Engineering,
University of Texas at El Paso,
USA

1. Introduction

Computational Fluid Dynamics (CFD) is the emerging field of fluid mechanics in which fluid flow problems are solved and analyzed using computational methods and numerical algorithms. In fluid mechanics, there are generally three routes of work in the field, three ways to conduct experiments. The first category is theoretical, or analytical, fluid mechanics. Theoretical fluid mechanics includes theorizing, manipulating and solving equations with pen and paper. The Navier-Stokes equation governing incompressible fluid flow is an example of theoretical fluid mechanics. Secondly, many engineers and physicist work in the area of experimental fluid mechanics. Experimental fluid mechanics involves conducting actual physical experiments and studying the flow and the effect of various disturbances, shapes, and stimuli on the flow. Examples include waves generated by pools, air flow studies in actual wind tunnels, flow through physical pipes, etc. Lastly, a growing number of engineers, mathematicians, computer scientists, and physicists work in the area of computational fluid dynamics (CFD). In CFD, you may still run an experiment of waves across water, an airplane in a wind tunnel, or flow through pipes, but now it is done through the computer Instead of actual, physical, 3D objects. A computer model is created, and computer programmers code the equations representing the physical laws that govern the flow of the molecules of fluid. Then the flow results (such as velocity and pressure) are output into files that can be visualized through pictures or animation so that you see the result just as you do with physical experiments.

In cases where an analytical, or theoretical, solution exists, CFD simulations and the mathematical models, which are coded in the computer program, are corroborated by comparison to the exact solutions. This comparative check is called validation. CFD is not yet to a point where solutions to problems are used without corroboration by existing, known, analytical or exact solutions when available. Validation is not to be confused with verification, however; validation is a check to make sure that the implemented, coded model accurately represents the conceptual, mathematical description and the solution intended to be modeled.

Still, there are many times when there is no analytical solution. In these cases, one often uses a computational approach. In such cases without a known solution, CFD is used to

approximate a solution. Most often, CFD is used when a computational solution is faster, cheaper, or more convenient. Convenience may be due to time or safety or another reason.

If we wanted to create a database of information about 3D flow around a cylinder for different types of fluids at varying speeds, CFD is quite helpful. It would take quite a long time to change the fluid in our flow container and clean the container for every type of fluid we desired. It would also take some time to change the speed of the flow. In this case, it's much faster to simulate it computationally. Then, anytime we wanted to change the speed of the flow, we simply would change one number in a computer input file. Or if we wanted to change viscosity and density for the fluid (representing a different fluid) we would just change the corresponding values in a computer input file. In this case, CFD is faster.

Now imagine if you were doing space defense work for a government, and you were asked to do fluid dynamics simulation of the combustion dynamics during an explosion or when a space shuttle launched (1). It would take immeasurable amounts of money to do test launches over and over as you studied the combustion dynamics of space shuttle thruster ignition. And it would take large amounts of money to test explosive devices, especially considering the damage they cause. In these cases, CFD is, again, quite helpful. The only costs in CFD are the time of creating a computer model, choosing the right mathematical model, coding it, and the power and computer time required to solve the equations. But it is plain to see that CFD is cheaper.

What if you were hired to help design material for the outfits of swimming athletes? Your company gave you the job of studying sharks and their agile ability to swim and maneuver through the water. So you start by trying to study the fluid dynamics around the shark skin (2). How convenient is it to locate sharks and place them in some type of testing container where you have probes and measuring devices located? How convenient is it to place probes on the body of the shark itself? How safe is it to work with the sharks in that manner? No, it's better to create a computer model of a shark and get the information for the shape, design, feel, and density of its skin and to use this information to run simulations. It is clear that CFD is more convenient in this situation. Sometimes a CFD simulation can be all three—safer, cheaper, and more convenient.

Imagine a situation in which two paratroopers, jumping from both side doors of a military cargo aircraft, always crash into each other down below (3). In order to analyze the fluid dynamics of the problem to see what air flow forces are affecting the paratrooper paths, you would need to perform test jumps with paratroopers. However, that is potentially injurious and not safe. You would also have to rent the plane, pay for the rental by hour, hire the test pilot, and pay for all the equipment for the jump. That is expensive. Lastly, the organization of the use of the military aircraft and personnel and equipment takes many months, and it can take from 6 to 12 months to plan the test. In this case a CFD experiment is more convenient: faster, cheaper, and safer.

Usually solving CFD problems involves three stages. First there is the pre-processing stage. In this stage, the geometric boundaries of the problem are defined. In 3D, a volume is created (in 2D, an area) over which the equation will be solved. This volume is broken into smaller units or cells creating a mesh (though there are meshless methods for computing CFD problems). This may be uniform or non-uniform. Along with constitutive equations, the particular equations are chosen for the problem in order to properly physically model the flow. These equations may be manipulated depending on the mathematical method being used. Boundary conditions are prescribed along the boundary. For time dependent problems, initial conditions are prescribed.

Secondly, the problem is solved numerically. At the least, we usually solve for velocity and pressure, but the list of unknowns can be longer depending on the mathematical model (equation) chosen for the physical situation. Other unknowns may include temperature, energy, and density. The numerical solution is usually computed iteratively for steady-state solutions. For time-dependent problems, a step in time is taken, and the equation is numerically solved again, eventually producing a solution for every time step.

The final step in the CFD process is post-processing. In this stage, the solution is analyzed usually with the help of visualization and possibly animation for dynamic, or time-dependent, problems. It is in this stage that CFD results are usually compared to any previous experimental results or known analytical, or theoretical, solutions. This comparison is usually called validation. Today, confidence in CFD is growing, but we have not arrived at the point of trusting CFD solutions without validation. Even if a particular model is validated, we still corroborate the results of a simulation with experimental or analytical results.

Because of the hybrid nature of CFD, advances in CFD are usually made in three areas: computational and applied mathematics, mechanical/chemical engineering, and computer science/electrical engineering. Some researchers work on new theoretical, mathematical models creating new discretization methods (ways to discretize the problem in order to numerically solve it over the discrete units or cells), or turbulence models. They might publish in applied mathematics or computational mathematics journals. Others work on computer architecture (such as different types of supercomputers or computer clusters), coding techniques such as parallel programming, or speeding up the computational processes through faster mesh generation and mesh reordering. They might publish their results in electrical engineering or computer science journals. Lastly others might use CFD to concentrate on new insights in the engineering aspect of the problem such as the mechanics of bird flight or sharkskin-inspired speedo design for less water resistance, or resistance to blood flow inherent to certain veins. They might publish their results in engineering journals next to experimental or theoretical engineering results. Their focus is on the application more so than the math or computer science.

2. Pre-processing

Pre-processing refers to the work that must be done prior to the actual computational experiment or simulation. This work can be reduced to four general areas: geometry definition, volume division, model choice and definition, and boundary condition definition. For the purposes of this article, we will ascribe the work of coding and the choice of computational implementation to the processing stage called simulation. Sometimes researchers refer to mesh generation as pre-processing in general because a mesh is generated when the volume is divided.

2.1 Geometry definition

The first step is to define the computational domain of the problem. The purpose of this definition is to confine the problem to a finite space and limit the computation. It is true that a plane flying in the air has some residual effects on air flow patterns 1,000 miles away, but because of the negligible nature of those effects, we are relatively safe in looking at the effect of the plane on air flow within a reasonable vicinity of the plane thereby limiting our

computational work and making the problem finite. It would be quite a task to compute the effect of the airplane on the air flow at a certain height around the entire earth.

To limit the flow in this way usually requires defining a domain inside which we will compute the flow, outside of which we will not compute the flow. Usually the geometry of the domain is chosen to be a box of some sort, usually a rectangular prism in 3D or a rectangle in 2D. However, any closed shape may be chosen as long as the shape closes off an inside computational domain from an outside space in which computations will not take place.

A closed space does not imply, however, an empty domain box. For example imagine that we are simulating intravenous blood flow (4) (5) around a cancerous growth. The vascular domain is modeled by a 3D cylindrical prism (our domain box), but we still have an object inside. In this case, our object is a semi-spherical cancerous growth on the surface of a wall of the vein. Traditionally, our domain must be totally closed, so the surface of the domain goes from the wall of the vein, joins the surface of the cancerous growth, and continues on the other side rejoining the wall of the vein creating a closed 3D space that does not go under the tumor but continues over the surface of it. Likewise, if we were calculating flow around a sphere (6), the domain box would be the outer half of the domain surface. The inner half of the surface would be the sphere inside the domain box. Just as in the cancer example, we are not calculating the flow inside the sphere, just as we were not calculating the blood flow in the tumor. But the tumor and sphere form part of the boundaries of the domain helping to close off the computational space in which we are interested in the velocity and pressure of the fluid. Remember, domain boxes may contain objects inside which no flow is calculated, but whose surface forms part of the surface of the domain helping to limit the computational space and better define where the fluid flows.

2.2 Volume division

The second step is volume division or mesh generation. Why we must divide the volume is not obvious until one remembers that it is easier to solve a flow problem over a smaller area or volume than a larger one. So dividing the volume into smaller units transforms the large problem over the entire domain into a large number of smaller problems over smaller sub-domains. However, the real reason we divide the volume is because we seek to find, for example, the velocity and pressure of the fluid at various points throughout the domain volume or area. In order to do this, we fill the inner domain volume or area with nodes—points at which we will calculate, in this instance, velocity and pressure. Once we have filled the inner volume or area with nodes, we connect the nodes with edges (and sides in 3D) creating smaller sub-volume or sub-area elements. For example, if our computational box is a rectangular prism and we fill it with nodes, we can connect the nodes to create quadrilaterals or tetrahedrals. If the domain is a rectangle and we fill the rectangle with nodes, we can connect all the nodes to create small rectangles or triangles. This network of rectangles/triangles and quadrilaterals/tetrahedrals creates a mesh of nodes; a mesh has been generated.

Remembering that the purpose of volume or area division is to create more manageable sub-volumes or sub-areas, it behooves us to evenly space out the distribution of nodes. If we do not, we may find that there are large spaces (volumes or areas) with no nodes. This is problematic because it means some of the sub-volume or sub-area elements will still be large; though our goal is to make them small.

Our second goal is to make each sub-element evenly shaped. In 2D, evenness in both directions means that 2D rectangles tend towards squares or that our triangles are equilateral. In 3D, if evenness is desired in all 3 directions, our quadrilaterals tend toward cubes and the tetrahedrals tend to be equilateral. Oblong and unevenly shaped sub-elements also create spaces (areas or volumes) with fewer nodes than parts of the domain with evenly shaped sub-elements. In these cases, our second goal serves a similar purpose as our first goal: to divide the domain into smaller, more manageable sub-domains with evenly distributed nodes.

An important aspect of mesh generation is choosing the appropriate size, or refinement, of the mesh sub-elements, such that important aspects of the flow are properly resolved. The general rule is that no fluid particle should advance through multiple sub-elements, cells, or units in one time step. Therefore our third goal is to increase the refinement (or the number of nodes) in areas or sub-volumes of increased fluid velocity or vorticity or any interesting fluid flow phenomena that you would like to capture computationally. This will allow us to visualize it later.

So far, the realm of geometry definition and mesh generation fall into the computer science side of CFD. When generating the mesh by defining a geometry and dividing that geometric area or volume, one must decide if one will use uniform sub-elements or non-uniform sub-elements. Definitely in parts of the domain with special flow requiring increased refinement, the elements in those parts will not match the refinement of elements elsewhere. But in the general flow one can still choose a uniform, structured mesh or a non-uniform, unstructured mesh. The ability to create non-uniform meshes is important in CFD because of the physical nature of fluids to occupy and fill any void left unoccupied. When dealing with complex geometries and small nooks, crannies, and crevices of an automobile or a model of a city block, it helps to have unstructured, non-uniform meshes that allow for the modeler to create the best shapes to fit the 2D or 3D space (7).

Likewise, the division of the volume or area in mesh generation requires the modeler to choose between quadrilaterals and rectangular prisms or triangles and tetrahedrals (there are other choices of shapes, as well, such as wedges and pyramids in 3D). Generally, quadrilaterals and rectangular prisms have a more accurate solution than triangles and triangular prisms, but there are ways to increase the accuracy of the latter. Because of the non-uniform and sharp geometries found in fluid problems by nature of fluids, triangles and triangular prisms work better geometrically for CFD applications. Normally, CFD researchers will utilize triangles and triangular prisms and then increase the number of interpolation points inside these elements so that no accuracy is lost. Interpolation points are points inside an element at which the solution is calculated. From these interpolation points, we can approximate the solution at any location inside an element.

All of the choices in dividing the volume and discretizing the mesh have the potential to introduce errors. Such errors, due to bad distribution of nodes or parts of the domain where the refinement is too low, are called discretization errors. These are errors that would disappear if we appropriately divided the volume or area or appropriately discretized the mesh.

2.3 Physical model definition

The first two steps dealt with the computer science side of CFD and there are many CFD engineers who work on geometric mesh discretization and mesh partitioning methods. Step three deals with the computational and applied mathematical side of CFD—choosing the

appropriate mathematical model. When preparing to model a certain fluid flow situation, one must decide which equation accurately describes the fluid flow one wishes to simulate. If no equation currently exists, the CFD engineer must do work in the theoretical side of CFD and formulate a new equation or a more specific equation for his or her specific fluid flow situation. If equations do exist, the CFD engineer must simply choose the correct equation for the fluid flow. This is not a trivial step as sometimes the same situation may require different mathematical models at different velocity regimes or different temperature regimes, for instance. So the specific parameters of the flow must be looked at in detail — velocity, viscosity, density, pressure, etc. — so that the correct equation is chosen.

A good example from fluid mechanics is the Navier-Stokes equation which is the basic or fundamental equation for fluid dynamics. If you remove viscosity from the equation, the Navier-Stokes equations become the Euler equations. Since all fluids have some amount of viscosity this approximation is important in flows in which the viscosity is negligible (8) such as sonic flows. A plane flying at sonic speeds will have air sliding past it, relatively, as if it had no viscosity. So the use of the Navier-Stokes equations also depends on the velocity of the flow, or more accurately the Reynolds number which governs the ratio of the kinematic forces to the inertial forces. You can still go further: if you remove vorticity from the Euler equations, you arrive at the full potential equations. The point in this illustration is that choosing the correct mathematical model is important, sometimes difficult, and always specific to the flow situation.

The choice of the mathematical model affects the unknown values you will compute. Some CFD simulations are really computational fluidothermodynamics because temperature and energy are calculated as well (9). For compressible flows, density is an unknown value and we would seek to solve for this value of density in the simulation. So the choice of mathematical modeling affects what unknowns we will compute. More accurately, the unknowns we want to compute in a given situation (along with other details about the flow situation) may help positively affect our choice of a mathematical model or the need to formulate a new one.

Remember that errors can be introduced at this step as well. If a inappropriate or poorly approximating governing equation is chosen, this affects the final solution. If a governing solution is chosen or formulated for which no analytical solution or experimental solution exists, we lose the opportunity for validation to reduce errors. Any simplification in the model or any untrue assumptions the mathematical model uses introduces errors as well. All of these types of errors can be classified as physical approximation errors because they deal with the physical, mathematical model (not the geometric model).

2.4 Boundary condition definition

After a mathematical model is chosen to model the physical phenomenon, usually boundary conditions must be chosen. This is the fourth step of the pre-processing stage, and this step falls on the applied math side of CFD. Usually we deal with boundary-value problems which require values to be assigned along the boundary of the domain of the problem in order to solve the problem throughout the 2D or 3D space.

For instance, in some problems, one may specify the value of the unknown on the boundary. Imagine prescribing the value of the velocity of the fluid on the boundary. Such a boundary condition is called a Dirichlet or a direct boundary condition because you are setting the

value of the unknown. If one specifies the value of a derivative of the unknown, the boundary condition is called a Neumann or natural boundary condition.

For example in hydrodynamic flow around a submerged rock with moss, a CFD researcher would usually place the boundary condition of free-stream velocity on the entrance side and exit side of the domain box assuming that the entrance side is sufficiently upstream from the rock so as to still be undisturbed, and the exit side is sufficiently downstream that the flow conditions have returned to free-stream conditions. The prescription of free-stream velocity would be Dirichlet or direct boundary conditions. The same researcher might assume stress (a derivative of velocity) to be zero in the direction perpendicular to the side surfaces of the domain box. When she prescribes stress in that direction she is setting Neumann or natural boundary conditions in the direction perpendicular to the side surfaces of the domain box.

Such boundaries are spatial boundaries. For time-dependent problems, there are temporal boundaries in a sense. Time dependent problems require an initial condition, prescribed values for the unknowns set at the temporal start of the simulation. In the same hydrodynamic example, let us say we want to simulate the flow when a rock, half the size of the stationary, mossy rock, was thrown into the river passing next to the mossy rock and hitting the riverbed. To start the simulation we need to have the steady flow of the river around the mossy, stationary rock without the 2nd rock thrown in. Once we have computed this flow, we can use the values of velocity and pressure from this flow as initial conditions for a simulation of a moving 2nd rock that is falling to the bottom of the river. From there the simulation will march in time and use the flow results from the previous time step as initial conditions for the next time step.

3. Simulation

The second phase of CFD work is the actual simulation or the "processing" work once the pre-processing work is completed. However, there are still some pre-simulation decisions to be made. CFD work is done through computers which not only decreases the time it takes to perform calculations, but also increases the amount of calculations that can be done in a given time period. As computing power has increased over the years, CFD has been used to solve larger and larger problems.

Large problems, however, were traditionally reserved for supercomputers. Supercomputers are large computers made up of multiple computers or CPUs. A desktop or laptop computer could only handle so many calculations due to hard drive limitations on different types of computer memory. As computers in general become more advanced, not only has the memory capacity of supercomputers increased, but so has the memory of desktops and laptops increased. This has created a cycle where problems solved by supercomputers today are solved by desktop computers and laptop computers tomorrow. And the problems solved by desktop and laptop computers today were only solved by supercomputers yesterday. For example, historically, a simulation of flow past an automobile was done on supercomputers (10) (11). Today one can create a model of an automobile and run a flow simulation of air flow past the automobile with one desktop or laptop machine. This example is one of many indicative of this ever-improving cycle.

Besides memory the other limitation on computing ability in today's world is clock speed. CFD workers are dependent upon computer scientist researchers to continue to increase the clock speed of microprocessors. In general, the faster computers become, the faster is the

Thank You for previewing this eBook

You can read the full version of this eBook in different formats:

> ➤ HTML (Free /Available to everyone)

> ➤ PDF / TXT (Available to V.I.P. members. Free Standard members can access up to 5 PDF/TXT eBooks per month each month)

> ➤ Epub & Mobipocket (Exclusive to V.I.P. members)

To download this full book, simply select the format you desire below