Cambridge University Press
978-1-107-05374-8 - Applied Computational Aerodynamics: A Modern Engineering Approach
Russell M. Cummings, William H. Mason, Scott A. Morton and David R. McDaniel
Excerpt
More information

1 Introduction to Computational Aerodynamics

Codes do not produce results, people produce results using codes.

—Dave Whitfield,
a longtime CFD code developer and user[1]

CFD simulation of a modern commercial transport (courtesy of Airbus; a full color version of this image is available in the color insert pages of this text).

LEARNING OBJECTIVE QUESTIONS

After reading this chapter, you should know the answers to the following questions:

- What is computational aerodynamics?
- What is computational fluid dynamics?
- What are the major goals of computational aerodynamic simulation efforts?
- What does it mean to be an intelligent user of computational aerodynamics?
- How have computational aerodynamic simulation capabilities changed during the past fifty years? Why?
- Can you list five ways that computational aerodynamics could be used on the design of a new airplane?
- Why is computational aerodynamics needed? Why not just determine aerodynamic characteristics with flight testing or wind tunnel testing?
- What is the difference between a real flow and a model of a real flow?
- What are the four steps of the computational fluid dynamics process?
- What is one of the largest sources of error in computational aerodynamics?

1.1 Introduction

Welcome to the world of computational aerodynamics! If you want to learn to apply what you know about fluid dynamics and aerodynamics to computationally simulate the flow around real aircraft, such as the F-16 shown in Fig. 1.1, please read on. Having the ability to predict the aerodynamics of a real aircraft can be exciting, and we hope you decide to join us as we learn the information required to perform such a simulation.

Not too many years ago, *computational aerodynamics* (CA) was reserved for those who performed numerical methods research or perhaps were program developers, or even for those developing new ways to apply computational aerodynamics to real-world analysis and design. Aerodynamicists, numerical methods researchers, and graduate students spent countless hours developing programs to apply computational aerodynamics to real-world analysis and design problems. While you may do some programming to arrive at numerical solutions in the course of your career, chances are you will be applying the techniques, programs, and models developed by others to new and interesting aircraft and aerodynamic concepts, like the X-31 flying at a high angle of attack as shown in Fig. 1.2. The vortices developing on the aircraft's canards, strake, and wing are clearly seen in this simulation.

Figure 1.1 F-16 fighter in formation flight with a computational aerodynamic simulation showing strake vortices, surface pressures, and the exhaust of the engine (courtesy of Stefan Görtz and the USAFA High Performance Computing Research Center; a full color version of this image is available in the color insert pages of this text as well as on the website: www.cambridge.org/aerodynamics).

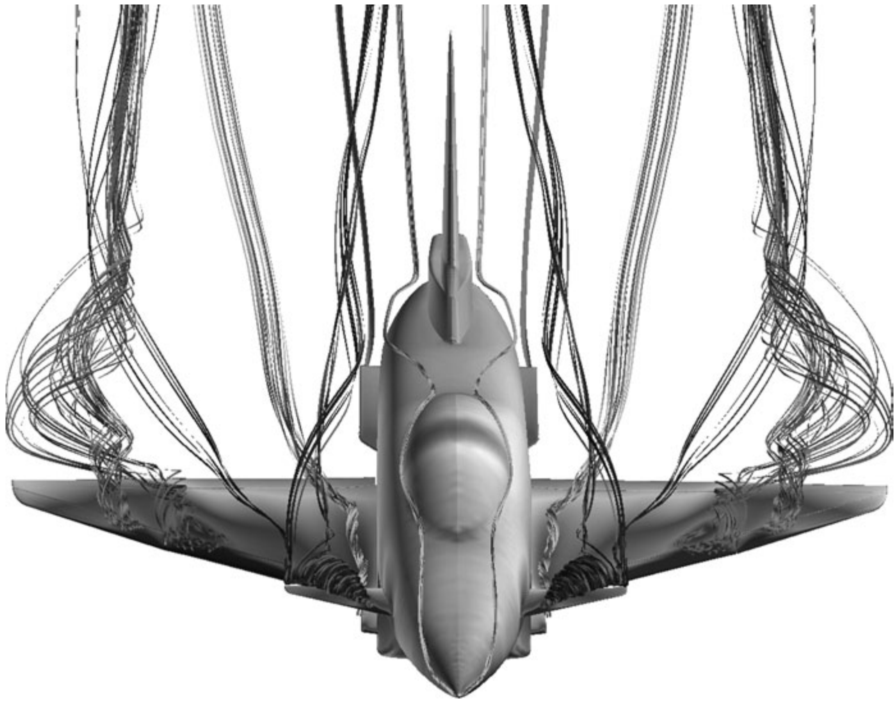

Figure 1.2 X-31 simulation at $\alpha = 16°$ with particle traces showing the various vortices on the aircraft (courtesy of Andreas Schütte of DLR; a full color version of this image is available in the color insert pages of this text as well as on the website: www.cambridge.org/aerodynamics).

Being able to adequately predict these vortices is essential to understanding how the airplane flies.

The ready and affordable availability of fast and efficient CA programs and the ever-increasing speed and memory of modern computers have made CA an integral component of modern aerospace design and analysis. You may perform applied research in your career, but it is more likely that you will be called on to analyze and design real and proposed aircraft components using the tools of computational aerodynamics. For example, perhaps you are designing an aircraft like the AV-8B Harrier that is required to hover above the ground, and you use CA tools to better understand how the vehicle will fly in this situation, such as for the simulation shown in Fig. 1.3. Having a detailed understanding of how the hovering vehicle interacts with the ground is essential to being able to design and control the aircraft in this situation, and CA tools can be extremely useful in determining these characteristics. Performing this type of simulation requires you to be able to properly understand and utilize the tools of computational aerodynamics. Since you are among the first undergraduate engineers to embark upon this world as practitioners of what has only recently become a well-established field of engineering, you have a special responsibility to take CA beyond where it has been.

The goal of this book is to make you an *intelligent user* of computational aerodynamics. What kind of user will you be? Will you know how to use the

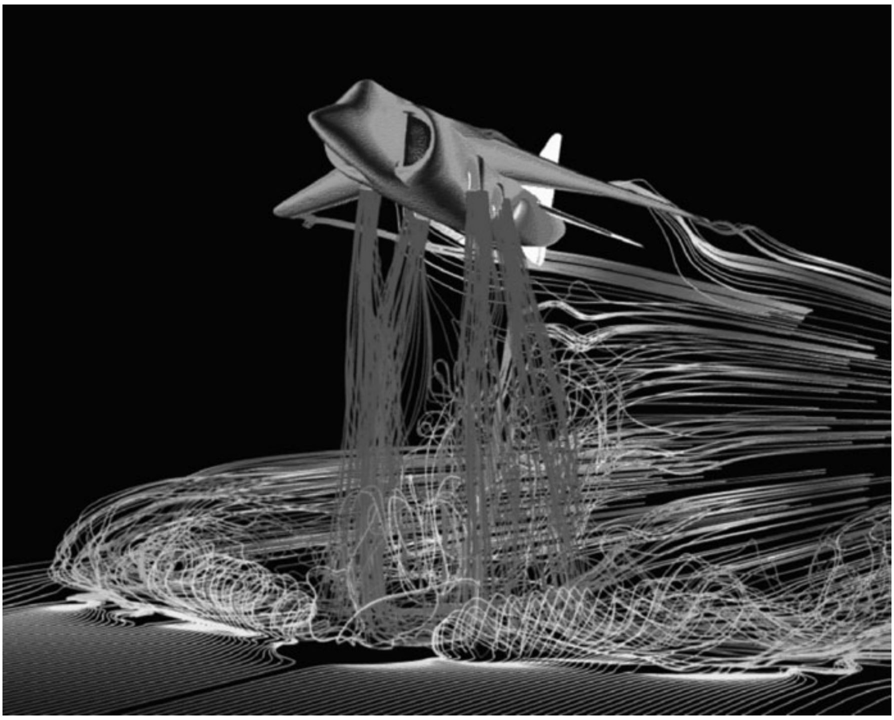

Figure 1.3 AV-8B Harrier hover simulation showing interaction of various jets with the ground plane; a full color version of this image is available in the color insert pages of this text as well as on the website www.cambridge.org/aerodynamics.

tools of CA? Will you know which tools are appropriate for which applications, or will you blindly input numbers into a program and get results that may not be meaningful? As an intelligent user, you will want to have enough knowledge and experience to understand what you are doing and why you are doing it. As someone learning to become an intelligent user, we hope you enjoy finding out about, and using, CA to accomplish aerodynamic analysis and design.

If you are going to learn about CA, you should probably know what CA is. Different people may have different opinions about a formal definition of computational aerodynamics, but a simple working definition should serve us well here: whenever you use a computer to model an aerodynamic flowfield, you are using some form of computational aerodynamics. Whether you are solving a potential flow problem using computer algebra software on a laptop, or you are using the most powerful supercomputer to perform a direct numerical simulation of the Navier-Stokes equations, you are using computational aerodynamics.

In order to better understand what all of this means, take a look at the F-16 fighter in flight as shown in Fig. 1.1 and 1.4. Fig. 1.1 shows the F-16 cruising at a constant speed and altitude, and Fig. 1.4 shows the same airplane while maneuvering. The differing colors shown on the solid surfaces of the airplane indicate the variation of pressure over the surfaces at a specific Mach number and angle of attack – these images are the result of a computational

Figure 1.4 F-16 fighter with surface colored to show the pressures plus surfaces of constant vorticity showing various vortex flow structures (courtesy of Stefan Görtz and the USAFA High Performance Computing Research Center; a full color version of this image is available in the color insert pages of this text as well as on the website: www.cambridge.org/aerodynamics).

aerodynamics simulation. Notice that the vortices and separated flow regions are visualized by surfaces of constant vorticity, which also show the shear layer created by the engine exhaust. Each one of these flow features makes an important contribution to the airplane's ability to fly well, and understanding each of these features is the job of an aerodynamicist. Because of the great power of such simulations, giving the aeronautical engineer the ability to simulate various aircraft components and the performance of the entire airplane, computational aerodynamics plays a critical role in current aerodynamic analysis and design, especially in the development process for advanced vehicles. That is why it is important to have a working knowledge of the appropriate roles and limitations of CA and to understand how and when CA can be used.

Another important distinction is how computational aerodynamics differs from *computational fluid dynamics*, and how you can tell the difference between the two. Computational fluid dynamics (CFD) involves using numerical methods and computers to solve the governing equations of fluid dynamics directly (the governing equations are a set of coupled nonlinear, second-order, partial differential equations that we will derive and discuss in Chapter 3). While CFD is a partial subset of computational aerodynamics, computational aerodynamics is broader and more encompassing than CFD, and certain applications of CFD are not directly related to aerodynamics. In the Venn diagram shown in Fig. 1.5, CFD is shown as an overlapping set with computational aerodynamics. Some of the CFD space lies outside of

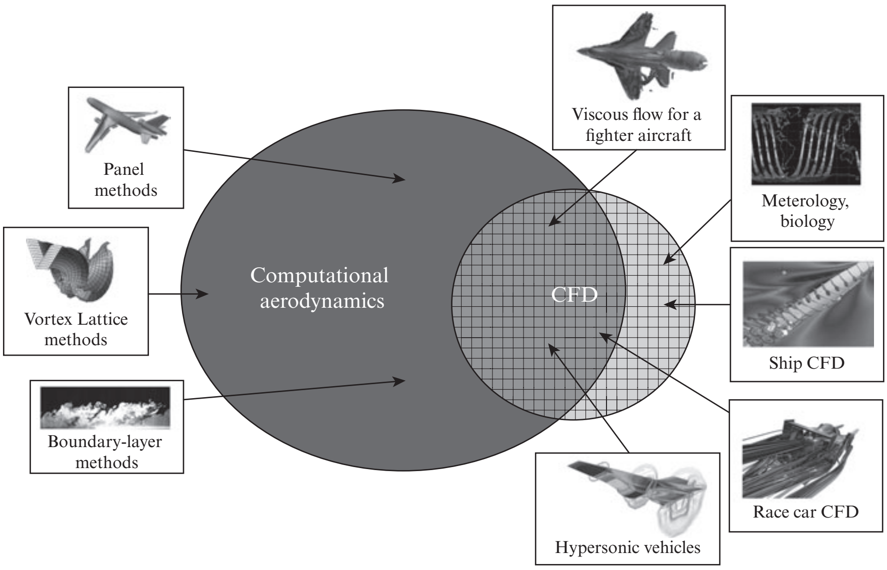

Figure 1.5 What is the difference between Computational Aerodynamics and CFD? (Don't worry if some of the concepts mentioned are not familiar – we will discuss most of them in later chapters.) Boundary layer image reprinted with permission © 1977 AIP Publishing, LLC; vortex lattice methods image courtesy of Phil Beran of the Air Force Research Laboratory; MD-11 courtesy of Analytical Methods, Inc.; F-16 courtesy of U.S. Air Force Academy; meteorology image courtesy of the U.S. Air Force; ship CFD courtesy of Fred Sterns of the University of Iowa; race car CFD courtesy of Richard Smith of Symscape © 2013; hypersonic vehicle courtesy of NASA Dryden Flight Research Center.

the domain of computational aerodynamics (including CFD applied to biological applications such as blood flow,[2] or geophysical applications such as weather prediction,[3] or for the design of ships and myriad other applications). But much of the CFD world lies inside the larger domain of computational aerodynamics – using computers to solve aerodynamic problems (including flow over airplanes, hypersonic vehicles, and even sports applications such as race car aerodynamics[4]). Computational aerodynamics also includes non-CFD applications, such as the so-called panel methods and vortex lattice methods (which will be developed in Chapter 5) used to predict flowfields assuming the flow is inviscid and irrotational, and boundary layer methods, among a host of other approximate but still valuable techniques.

1.2 The Goals of Computational Aerodynamics

Understanding the difference among these various CA approaches to analyzing problems, and knowing their strengths and weaknesses, is crucial to being a good user of CA methods. For example, the design and analysis of airplanes using CA is quite pervasive today, but the analysis can use a wide variety of tools, from low-fidelity (more approximate) models such as panel methods to high-fidelity models such as those that can predict the aerodynamics for full aircraft including viscous effects. Understanding how and why CA is being used is an important aspect of knowing how to use CA correctly. Some typical CA usages for airplanes include:

- Airframe design and analysis, i.e., development of airfoils, wings, and entire configurations, including the determination of loads for structural design
- Aircraft/propulsion system integration, i.e., the design of inlets, diffusers, and nozzles
- Performance prediction: estimation of lift and drag characteristics for determination of takeoff, climb, cruise, descent, and landing performance of aircraft
- Vehicle stability, control, and handling characteristics, i.e., to provide the mathematical model for flight simulation, and use of simulations to resolve any related flight problems
- Aeroelastic analysis, including flutter and divergence – requiring coupling with structural analysis and control system design analysis methodology
- Multidisciplinary optimization, i.e., simultaneously computing and optimizing multiple facets of an aircraft design, such as aerodynamics, structures, propulsion, performance, and mission objectives

When performing these types of calculations, it will become apparent that there are many differences between CFD and CA, which leads to the need to be able to intelligently choose which computational aerodynamics tool to use for a given task. Do you need to use a full Navier-Stokes code[i] to solve for the pressure distribution over an airfoil at low angles of attack (where the flow is attached) and low speeds (where the flow is essentially incompressible)? Can you use a potential flow code to analyze a jet fighter that is flying at very high angles of attack? The answers to these questions forms the purpose of this textbook – anyone using computational aerodynamics should be an *intelligent user* of the programs and methods available, as we will describe.

1.3 The Intelligent User

Skilled engineers can obtain valuable results using computational aerodynamics with knowledge, ingenuity, and judgment. The computer power available to every engineer today is greater than the total computing power available to the engineers who put men on the moon in the Apollo program in the 1960s, and even to those who designed the Space Shuttle during the 1970s. Significant responsibility accompanies the use of this computational capability! Unfortunately, it is possible for an engineer using this large computational power to make errors and not catch them, especially when that engineer puts too much faith and not enough skepticism in the untested results

[i] A *code* (or *source code*) is a collection of statements written to run on a computer.

of a massive computation. As Stan Lee, the creator of Spider-Man, said, "With great power there must also come great responsibility."[5] We believe that becoming an "intelligent user" of CA is very important and comes with responsibility; we therefore want to spend some time helping you to understand what is involved in using the great power of computers.

While numerous books are available that describe CA, guidance on the application of the methods and approaches for CA is often scarce. Many books concentrate on how to develop CA tools. It is more likely, however, that you, along with the vast majority of engineers working in computational aerodynamics today, will be *applying* existing methods and tools, not *developing* new ones. Successful practitioners must, however, understand the underlying algorithms and assumptions employed in CA methods to be *effective users*. The ability to approach aerodynamics problems using computational methods, assess the results, and make good decisions is a critical engineering skill.

An analogy may help you understand what it means to be an intelligent user of computational aerodynamics software. Traditionally there are at least three "levels" of carpenters: the apprentice, the journeyman, and the master – each level requiring increasingly more sophisticated abilities and tools. When building a house a variety of skills are required which use various tools, including hand tools and power tools. The apprentice carpenter, new to the trade, learns the basics of carpentry: using tools, joinery, planning, etc. A journeyman carpenter has mastered all of the manual skills required to perform basic carpentry and is capable of working with others to complete every aspect of a project. The master carpenter has gone beyond the level of the journeyman by showing that all aspects of using tools and building requirements are known and understood. A master carpenter has many more tools on his tool belt than an apprentice carpenter and a great deal more knowledge about how to use those tools. However, just knowing how to use a tool does not make you a master carpenter; it is the ability to apply that knowledge toward solving a problem (such as working with an architect or contractor to suggest more economical, efficient, or structurally superior methods of building a house) that makes one a skilled carpenter. Being a master carpenter means knowing how to use all the tools, knowing the right tool to use for each task, and seeing where each task fits within the project as a whole.

How does all of that relate to computational aerodynamics? Most of the history of computational aerodynamics has been devoted to developing the tools that are needed to perform computations (numerical algorithms, grid generators, flowfield visualization tools, etc., all of which will be explained in detail in later chapters). This would be equivalent to a modern carpenter spending time making his own tools, which is something that carpenters had to be able to do a long time ago. But carpenters today no longer have to make their own tools; there are many well-designed and affordable tools already available. Computational aerodynamics has reached a level of maturity where analysis can be performed by those of us who do not want to develop tools, but want to apply the already available tools. We want to build the house! This book is designed to help you become an intelligent user of computational aerodynamics tools, rather than a developer of computational aerodynamic tools. If you want to learn to develop the tools, there are numerous excellent books available, including those by Blazek,[6] Pletcher, Tannehill, and Anderson,[7] Hoffmann and Chiang,[8] and Löhner.[9] However, if your goal is to learn how to use the existing tools effectively, then please continue reading.

To become a "master user" of computational aerodynamics tools, you will find that you need to have at least a rudimentary understanding of aerodynamics, numerical methods, experimental procedures, and computer systems. This "entry-level" knowledge has traditionally forced computational aerodynamics into the realm of graduate studies at universities, since the average undergraduate student would not be able to become "expert" enough in all these areas to make any meaningful headway. Now that computer capabilities have advanced significantly, and various codes are becoming mature, undergraduate students should be able to take well-developed CA tools and apply them. The danger is that these easy-to-use codes can be used by people who do not understand enough about them, leading to errors that may be costly (or even dangerous). Knowing how much you need to know, however, is quite difficult (especially if you do not know what you need to know!). Finding the balance between the two is one of the important goals of this book, which will show you what you need to know about the basic physics and numerical methods involved in CA, code usage (other than how to make it run), and computer-related issues that you should understand in order to use the code efficiently and effectively. Also, we will make you aware of experimental and theoretical aspects of aerodynamics. Maybe you will not have to know all of these things in your career, but we believe having a basic knowledge is essential to being an intelligent user. These are the types of issues you should resolve before you "blindly" begin using computational aerodynamics programs.

Computational Aerodynamics Concept Box

The Garbage In/Garbage Out (GIGO) Syndrome of Computer Usage

"On two occasions I have been asked, 'Pray, Mr. Babbage, if you put into the machine wrong figures, will the right answers come out?' … I am not able rightly to apprehend the kind of confusion of ideas that could provoke such a question."

– Charles Babbage, *Life of a Philosopher*

Actually, we have heard this question from our students for years! We all have seemingly developed a belief that computers are smart and capable of doing anything. This had led the average person to somehow believe that computers are smarter than the people who program them or the codes they run. Simply put, however, computers are really quite dumb, since they only do *exactly* what you tell them to do. If you put Garbage In you will get Garbage Out, which is known as the GIGO syndrome.

This not only applies to using computers, it also applies to programming computers. Students and practicing engineers alike may believe that the computer knows everything, even things that they have not "told" the computer (including correct input information or unknown information not supplied by the user). Over the years, the syndrome has spread to include those who make decisions based on the computer results. Supervisors, managers, accountants, and politicians have all been guilty of basing decisions on faulty, incomplete, or imprecise data. These perceptions are often so pervasive that a new GIGO syndrome has developed called Garbage In, Gospel Out. This is based on our mistaken tendency to believe and trust computerized results and blindly accept whatever the computer says. Computers are very good on certain things, such as processing large amounts of data very quickly, but they are not very good at knowing what we want unless we tell them completely and correctly.

A very troubling example of GIGO happened to Wall Street bankers during the financial collapse of 2008. Federal regulations required that each bank and investment firm have computers perform risk management analysis for every investment. These computer programs are supposed to raise a red flag when the bank is over-exposed to certain risky investments, and then the bank is supposed to reduce the amount that is invested in the risky enterprise. "In other words, the computer is supposed to monitor the temperature of the party and drain the punch bowl as things get hot," according to *New York Times* reporter Saul Hansell.[10] Unfortunately, "the people who ran the financial firms chose to program their risk-management systems with overly optimistic assumptions and to feed them oversimplified data. This kept them from sounding the alarm early enough.... Lying to your risk management computer is like lying to your doctor. You just aren't going to get the help you really need." Bad data were put into the computers (Garbage In) and the risk management programs did not warn the bankers that they were in trouble (Garbage Out).

1.4 A Bit of Computational Aerodynamics History

Computational aerodynamics is a relatively young field of engineering. While computers have been around for many decades, the capacity of the computers to perform high-level calculations (the kind required for many CA applications) has only recently become sufficient to perform the calculations and display the results for full aircraft of interest. In fact, it would be safe to say that in the past CA has helped to push the development of larger, faster computers. A fuller understanding of this will be evident once you work your way through the next few chapters, but for now we can see the progress that CA has made by looking at some examples from the past; this should help to put things in perspective.

The original use of computers for CA predictions occurred during the 1960s primarily for various potential flow methods. These methods included the panel method and the vortex lattice method, both of which will be described in greater detail in Chapters 4 and 5. Panel methods allowed researchers to represent an airplane by a series of panels, where each panel