14. Using Computational Aerodynamics: Review and Reinforcement

It is a well-known phenomena that computer programs, left to themselves, will cease to work. This is known as "Bit Rot". Programs NOT left to themselves also cease to work. This is known as "Enhancement".

The FLOPS Motto, Arnie McCulllers, ViGYAN, Inc.

To complete the course, we present a quick review of some of the key aspects of computational aerodynamics from the point of view of someone needing to apply these methods in aerodynamics. We include:

• Code Development
• Code Validation
• Code Selection
• Problem Solving with Computational Aerodynamics

14.1 Code development:

Recall the procedure described in Chapter 3 to develop computational aerodynamics codes. This is the list. It is based on recommendations from Roache:[1]

• Start Simple
• Debug and test on a coarse mesh first (reduces cycle time)
• Print out "enough" information:
    - some at each step
    - lots sometimes
    - print good diagnostic functionals
• Use graphics to look at the *whole* field
• Always check on the finest mesh possible before releasing code
• Test convergence to machine accuracy
• Try to check all option combinations in a production program
• Check convergence/stability over the widest possible range of parameters
• Test accuracy against:
    - exact solutions
    - approximate solutions
    - experimental data
• Avoid unnecessary hardware dependence

The final issue is the problem of meeting schedules for code development. If the reader has done the exercises at the end of each chapter, than he or she knows that it always takes longer than you expect to do computational work. It is important to plan for this. In particular, code developers often quote the time it takes them to think of how to do something, not the time it takes to actually implement and validate a piece of work. This is vitally important if you are responsible for meeting the budget and schedule.

14.2 Code Validation Procedure:

Once the initial code development is finished, it must be validated. Although originally not recognized as a key step in the computational aerodynamics process, it has begun to be realized by managers that confidence (and credibility) must be established before results from codes can be used to make engineering decisions involving financial and time schedule risks. In fact, an entire AGARD meeting was devoted to the subject. Two reports have been issued as a result.[2,3] Thus the situation has begun to follow the tradition of wind tunnel testing, where a significant effort is made to validate the results. To develop a code to this level of maturity requires special planning. The procedures shown in Figure 14-1, first defined by Bobbitt,[4] identify the steps required.

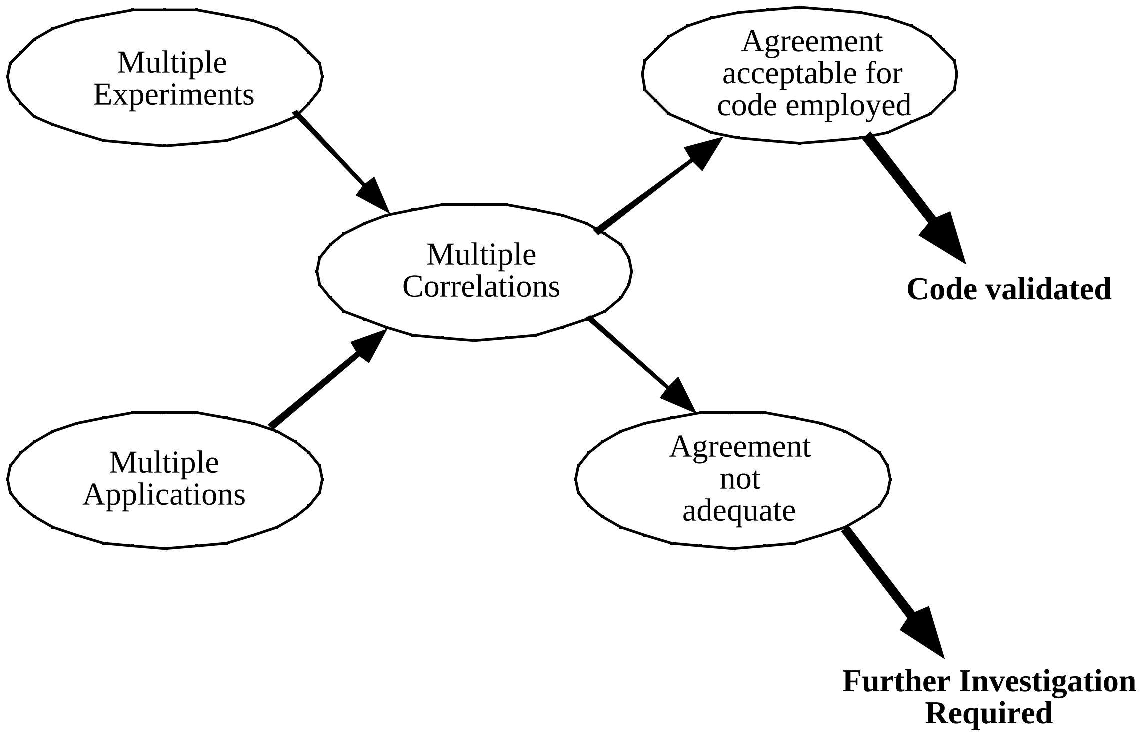

Figure 14-1. Required code development/validation steps needed before real application

In 1994 several AIAA meeting sessions were devoted to code validation, and in fact some new definitions to define code status were proposed. The concept of a certified code was introduced. This has been done in other software areas. The interest in doing this in CFD implies that CFD is beginning to reach a stage of development where standards can be established to define code accuracy. The basis for code certification was described in a key paper by Melnik, et al[5]. Two other papers identified the problem and discussed approaches to validation.[6],[7] We can expect this activity to increase as industry begins to rely more heavily on code results to make decisions.

14.2.1 Issues for CFD codes and Experiment

After comparing experiment and computations, we must attempt to understand any poor agreement found between computed and experimental results. This can be very difficult, and requires considerable knowledge of both CFD and experimental aerodynamic methodology. A list of items to consider in the explanation of disagreement between code and experiment was also developed by Bobbitt. It included:[4]

Code:

- Are incremental effects better predicted than absolute values?
- Is the math model adequate?
- Can the solution technique be improved?
- Is there a higher order code available?
- Has the grid been refined sufficiently where large changes occur?

Experiment:

- Is the instrumentation adequate/accurate?
- Can another tunnel/model support be used?
- Is flow quality/uniformity or transition a factor?
- Has the wall-interference been evaluated?

14.2.2 Error sources in CFD codes and in Wind Tunnel Data

In trying always to determine if we have the right answer we need to make sure we can understand what the possible sources of error are, and how to evaluate them.

Error sources in CFD codes:

- math model/equation set
- artificial viscosity/dissipation
- geometry representations
- solution not converged
- round off error/truncation error
- bugs
- solution algorithm
- boundary conditions
- grid resolution
- turbulence model
- Reynolds number

Error sources in wind tunnel data

- sting effects (model mounting)
- Reynolds number
- transition
- geometry definition and accuracy
- flow uniformity
- wall effects
- flow quality/noise
- instrumentation
- aeroelasticity
- surface finish

14.3 Code Selection Considerations

Will you use a code to do engineering work and make judgments that affect your future? If so, make sure you consider:

- capability
- problem setup time/grid generation
- turnaround time
- availability
- cost
- confidence

What is required to use a code?

- CFD code validation
  - CFD validation experiments may be required
- Validated CFD Code
  - CFD code "calibration"

There is one major problem. Algorithm developers produce codes faster than the validation process takes. If you want to use the latest methods they will probably never be completely validated, and in fact you may find bugs, and have to work with the source code to handle exceptional cases. Nevertheless, any time that I've been involved with projects that tried to use CFD codes without first examining the code accuracy in detail we regretted it before the project was finished. The recent paper by Knill, *et al*[8] demonstrates the effort required to develop confidence in CFD code predictions. This is especially true when codes are used in optimization projects, where thousands of code executions occur and the results of each execution are not evaluated individually (until a problem occurs).

14.4 Problem solving using CA[9]

To solve a specific problem, the following steps must be followed:

- What do you want from the computation
  - the decision here is the key: should you or shouldn't you?
- Model the physics as a properly posed problem (BC's)
- Select the appropriate code (previously validated)
- Define the geometry, the mesh and the input
- Check the input (visual displays are essential)
- Run the program
- Display and interpret the results!

In addition, just like the code development time problem, it always takes longer to make the analysis than anticipated. At the minimum, the procedures should be defined and a "dry run" made using approximate information while waiting for final data. Inevitably, an unforeseen problem will arise. As a final bit of worthwhile reading, look at the article by Dick Bradley,[10] who has been both a code developer, user, and is now a manager. In this capacity he has considerable perspective on the problems and possibilities of computational aerodynamics. He provides an overview of CFD issues in an engineering environment that are not available from CFD experts.

14.5 That's It!

• You now have tools for aerodynamic design using computational methods

• Goals for the course:

- Knowledge
- Attitude (perspective)
- Skill

We've made an effort to achieve each of these goals. Good luck!

14.6 References

[1] Roache, Patrick J., Computational Fluid Dynamics, Hermosa Press, 1972

[2] AGARD CP-437, Validation of CFD, May 1988. (published in two volumes)

[3] Sacher, P.W., edited by R.G. Bradley, Jr., and W. Schmidt, "Technical Evaluation Report on the Fluid Dynamics Panel Symposium on Validation of Computational Fluid Dynamics, AGARD-AR-257, May 1989.

[4] Bobbitt, P. "The Pro's and Con's of Code Validation," AIAA Paper No. 88-2535, 1988.

[5] Melnik, R. E., Siclari, M. J., Barber, T., and Verhoff, A., "A Process for Industry Certification of Physical Simulation Codes," AIAA Paper 94-2235, June 1994.

[6] Oberkampf, "A Proposed Framework for Computational Fluid Dynamics Code Calibration/Validation," AIAA 94-2540, June 1994.

[7] Bussoletti, J. E., "CFD Calibration and Validation: The Challenges of Correlating Computational Model Results With Test Data," AIAA Paper 94-2542, June 1994.

[8] Knill, D. L., Grossman, B., Mason, W. H., and Haftka, R. T., "Certification of an Euler code for High-Speed Civil Transport Optimization," AIAA 34th Aerospace Sciences Meeting and Exhibit, AIAA Paper 96-0330, Jan., 1996.

[9] Rubbert, Paul, AIAA Short Course for Computational Aerodynamics, 1978.

[10] Bradley, Richard G., "Future Directions for Applied Computational Fluid Dynamics," in *Applied Computational Aerodynamics*, P. Henne, ed., AIAA, Washington, 1990. pp. 889-901.