Application of the Studio Model to Teaching Heat Transfer

Robert J. Ribando, Timothy C. Scott, Gerald W. O’Leary
University of Virginia

Abstract

Over the past five years we have transformed our undergraduate heat transfer course from a strictly lecture format (with an associated lab the following semester) by replacing one lecture a week with a two-hour “studio” session. These sessions are held in a classroom equipped with a computer for each pair of students. Much of the studio work revolves around a set of locally developed, research-based numerical algorithms that solve in real time the governing ordinary and partial-differential equations describing heat and fluid flow. Several of the modules may be considered “virtual” laboratories, that is, they allow students to take data from the computer screen for post-processing — much as if they were working in a real, extremely well-equipped laboratory. Others give the option of performing dozens of "what if” calculations rapidly, thus inviting their use in the design process. Some merely replace the table and chart lookups that are so commonly used in the study and practice of heat transfer. In the studio projects, students are exposed to modern computational techniques while seeing them applied to fundamental problems. With the complete field solution available from the numerical model and not just a “bottom-line” result, most of the modules are designed to be visually rich, and thus allow the students to understand the physics behind the often-confusing convection correlations. Since many of the problems we work on do have analytical solutions, they gain experience in the verification of results. In several projects students are exposed to spreadsheet programming using macros. For a number of these modules we have developed an accompanying desktop experiment to enhance still further the hands-on nature of the studio and to allow comparison with our numerical models.

Introduction

In recent years we have transformed our undergraduate heat transfer course from a lecture course (with an associated thermal sciences laboratory the following semester) to include what we call a “studio” session. The latter, a two-hour “hands-on” session held in a room containing a computer for each pair of students, supplements the two lectures each week that are held in a room having a computer and projection system just for the instructor. Much of the studio session centers on a set of modules that we have developed locally for use in our undergraduate and graduate heat transfer courses. While it would probably be possible to take the next step and teach this entire course in the “studio” mode, as has been done in many courses at RPI and several other universities [1-4], we have not taken so drastic a step as yet.

The software we use and the implementation of the studio experience in this particular course has come about largely as a confluence of the authors' industry, research and teaching experiences. Each of the three authors worked previously in an industry that makes extensive use of computational modeling (the nuclear, automotive and aerospace industries, respectively), and thus began this work with a perspective that might differ from that of a purer academic. Those past experiences, coupled with discussions with engineers currently practicing in industry, indicate that many recent engineering graduates are not as well versed as desired in using the modern computer-based tools in their own disciplines. Virtually all do use word processors, spreadsheets and presentation and search tools proficiently, but then so do today's liberal arts graduates. What we mean here includes the ability to use a computer on a problem of the sort they might solve in their math, science and engineering courses on paper, but making profitable use of the speed, versatility and visualization capabilities afforded by modern engineering software tools. In addition we include expertise with the full range of problem solving and design capabilities made possible only recently by the advent of ubiquitous computing and inexpensive software packages. Finally, and ultimately, we mean the wisdom to know when a simple "slide rule" calculation is sufficient for the engineering job at hand and when a rigorous, thorough computational analysis would be appropriate.

To address the problem of little computer use between a first year programming course and the canned "design" packages commonly employed toward the end of the four-year program [5], some departments have added a numerical methods course somewhere in the curriculum. With an already overcrowded curriculum undergoing credit hour reduction under a state mandate, the addition of a new course did not seem a viable option for us. To our way of thinking the heat transfer course that is taught in the sixth semester or thereabouts of the undergraduate mechanical engineering curriculum seemed a likely candidate for a major infusion of computational methods. First of all, at least in the thermal sciences stem of the ME curriculum, it is the last engineering science course that students will take. Students are ready to see how the principles they have covered in a variety of courses come together, and heat transfer provides plenty of good examples. Principles and techniques they have covered in ordinary and partial differential equations, physics, thermodynamics and fluid mechanics are all used in heat transfer and moreover, are applied to the design of practical devices such as heat sinks and heat exchangers. Unfortunately, however, the field is so highly mature that many of the topics have become very "cookbook." The science of convection heat transfer includes "magical" formulae containing non-dimensional numbers raised to strange powers and affording little, if any, physical insight. Transient conduction involves computing non-dimensional numbers and then looking up other non-dimensional numbers on charts made in the 1940's specifically because no one had a computer sitting on their desk! Heat exchanger design and analysis again uses charts "canned" in the first half of the 20th century, because no one other than a few bright academicians could solve the problems at the time. The result is a course that may be boring for the students, especially the better ones. For some instructors teaching heat transfer is a frustrating experience; while others may jump at the chance to teach a course for which their lecture notes are "timeless."

Origins of the Teaching Modules

While some instructors seeking to introduce modern computational modeling into mainline undergraduate courses have tried using existing commercial Computational Fluid Dynamics (CFD) packages, we avoided that approach. Our prior experience on a research project with at least one such package indicated that its use would divert too much course time to learning that particular product. We wanted “learning the software” to be synonymous with learning the course material! For that reason we used as the basis of our teaching modules a collection of Fortran programs that the first author had created over the years. These programs had been developed as student homework projects and in-class demonstrations for a variety of courses, both graduate and undergraduate. The early incentive for their development was teaching in our televised Master of Engineering degree program. The format of courses taught through this outreach program is two live classes per week. Classes last the full 75 minutes; most instructors feel that making much small talk would be inappropriate. Much information that for a normal on-campus course would be conveyed orally is instead distributed in print (via the Web in recent years) in order to avoid barrages of phone calls and e-mails - thus meaning more class time to fill! Interaction is difficult to generate and sustain (although somewhat easier now that the delivery mode has been switched from one-way video, two-way audio satellite-based to a fully-interactive videoconferencing format). Unless the instructor takes proactive steps, then for nearly the full class period, the camera is either on his face or on his hand writing with a blue felt tip pen on a blue pad. The result is probably close to ten times as much “face time” per week as a highly paid network news anchorperson! Under such conditions there is considerable cause to enliven one’s classes by use of whatever visual aids one can collect and develop.

Until the 1995-96 academic year when a university program aimed at helping faculty willing to use technology in their undergraduate instruction was established, this software remained pretty much “batch” programs that were used by the instructor for demonstration purposes. Each program did, however, include integrated graphics using the primitives provided with the Watfor-77 instructional Fortran package. Thus for about five years they were employed (in a projection system-equipped classroom as well as on TV) for motivating lectures, demonstrating the effects of parameters, etc. With the support of that university program, we created Visual Basic interfaces for some eight Fortran programs, thus for the first time giving us modules that students could use readily themselves.

Watcom Fortran 77 was used for the intensive numerical computations and, along with the MS Windows API’s, for generation of all graphical output. For each module, the applicable Fortran routines are combined into a single dynamic link library (DLL), which is then available for function calls from a tailored executable written in Microsoft Visual Basic. The VB available at the time (Version 3) being an interpreted, rather than compiled, language was much too slow for the intense computations required of several of our numerical models; indeed, it was not fast enough even for most of the graphical presentations. With the currently available Version 6 of VB, which is a compiled language, any new development will not require the previous very complicated, mixed-language programming.

Goals and Outcomes

Rather than discuss the technical details of individual software modules in any detail here (many have been reported thoroughly in other venues [6,7]), we focus here on goals and outcomes of their use in our class. Full-color screen shots of the interfaces of each of the modules are available on the first author's website, the URL of which is given in the references.

Using Modern Computational Methods

All of these modules use modern, research-based algorithms operating behind the scenes. As an example we cite the module developed for transient conduction in a one-dimensional body. This topic is a mainstay of all courses and textbooks, virtually all of which present charts that were published in 1947 based on a one-term approximation of the solution found by separation of variables. Our module, which uses the finite-volume numerical method rather than an analytical solution, allows the students to watch the actual transient in progress on their screen. All inputs are the same as needed in order to use the charts, and thus the "stretch" to using a numerical solution is not that great, since, with the normal progression of the course, they will be doing such solutions themselves in the following week or two anyway.

Understanding the Physics behind the Correlations

For many students convection heat transfer seems to consist of "a bunch of equations involving dimensionless variables raised to strange powers." While certainly convenient, the conventional approach clearly tends to obfuscate the underlying physics. Our flat plate module [6] solves the boundary layer equations quickly for both laminar, and with a simple turbulence model included, turbulent flows. Students are able to observe the effect of Reynolds and Prandtl numbers in an instant (See Figure 1) and to experiment with various thermal boundary conditions. A virtual probe even allows them to take local velocity measurements. The module is set up so that they can take data and dump it into a spreadsheet for processing, much as if they were working in an extraordinarily well-equipped laboratory.

Comparing with Experiments

For several of the modules we have built a real physical experiment that we bring into the studio and take data from to compare with the computed predictions. One particularly successful one involves the use of an electrical analog for a two-dimensional, steady-state conduction problem. Students set up the nodal equations for a two-dimensional fin and then are able to make a one-to-one comparison between their numerical model and a physical model of the problem consisting of several hundred resistors.

Returning to Fundamentals

One particularly onerous area to teach in a heat transfer course is heat exchanger analysis and design. Because the solution of a coupled set of ordinary or partial differential heat balance equations by analytical means is challenging to say the least, the solutions were graphed during the first half of the 20th century and have been used ever since. Now it is relatively trivial to solve the discretized versions of the same heat balance equations in real time [7]. With the entire solution available on the screen, the student designer can actually "see" what is happening inside what previously had been a "black box." Now instead of getting a single "right" answer to an end-of-chapter problem, students actually ask how they can use the tool to create a better design.

Testing Parameters and Using Spreadsheet Macros

Originally we had hoped to have enough of the "canned" VB/Fortran modules to cover all the major topics in heat transfer and all 14 weeks of the semester, but that simply was not feasible. For other weeks we have developed projects for which the students have to create their own solutions, more than likely on a spreadsheet. One example involves the frequently used model of a human runner approximated as a cylinder in cross-flow. For this project we created Visual Basic for Applications (VBA) macros [8] for all the properties of air and water needed in the calculation. Students develop a spreadsheet where they can assess the importance of the ambient temperature, relative humidity and runner speed on getting rid of waste metabolic heat. Their cell formulae access the necessary property macros so that a range of parameters can be run easily. Hypotheses can be tested readily, and conclusions can be reached and presented graphically. This quick introduction to VBA gives students an additional skill that they can use in other courses and eventually on the job.

Verifying Output and Input

Certain of our modules and spreadsheet workbooks actually allow the user to visually check their input data. The viewfactor spreadsheet is a good example. Once the user inputs the appropriate dimensions, they get a scaled drawing of their geometry on the screen as well as the numerical results for the viewfactor (Figure 2). Seeing good human-computer interface design practiced in their own classes cannot help but be good training for young engineers who will soon be designing products and processes themselves involving such interaction.

Lessons Learned and Conclusion

Over the last half-dozen years of software development, use and refinement, we have learned a number of lessons. First of all, because of the inordinate amount of time involved in developing a foolproof piece of software, it does not make sense to try to "computerize" a topic that is already covered well in print media. Furthermore, software that is "user-friendly" for industry is probably not appropriate for instruction. Practicing engineers are already supposed to know the fundamentals; students do not or they would not be taking the course. In constructing modules of the sort we have done here, interface design and pedagogy is just as important as the science and is more time-consuming. One very important finding from this exercise is that with well-developed software (and thus a very limited learning curve) and the intent to limit the amount of class time devoted to the exposition of pre-computer analysis techniques, there is more time left to cover engineering design. In fact, good software invites the user to test ideas and explore hunches because that has been made so easy.

References

1. Wilson, J.M., "Institution-wide Reform of Undergraduate Education in Science, Mathematics, Engineering and Technology," Proceedings of the Frontiers in Education Conference, Nov. 6-9, 1996.

2. Lahey, jr., R.T. and Gabriele, G.A., "Curriculum Reform at Rensselaer," Proceedings of the Frontiers in Education Conference, Nov. 6-9, 1996.

3. Harris, J.M. and Fleishon, N., "The Excellence in Mathematics, Science and Engineering (EMSE) Project at Cal Poly," Proceedings of the Frontiers in Education Conference, Nov. 6-9, 1996.

4. Carlson, L.E., Peterson, L.D., Lund, W.S. and Schwartz, T.L., "Facilitating Interdisciplinary Hands-on Learning Using LabStations," Proceedings of the ASEE Annual Conference, June 28 – July 1, 1998.

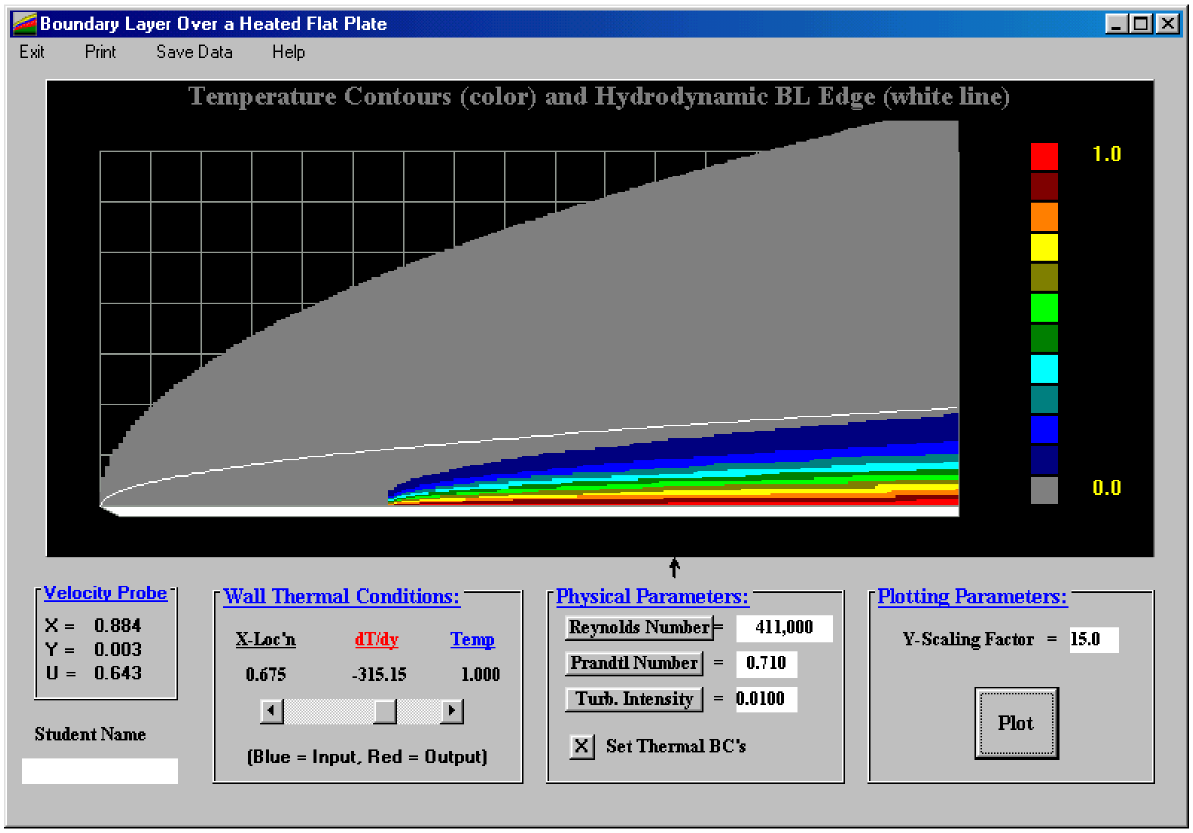
5. Jones, J.B., "The Non-Use of Computers in Undergraduate Engineering Science Courses," *J. Engineering Education*, Vol. 87, no. 1, 1998, pp. 11-14.

6. Ribando, R.J., Coyne, K.A., and O'Leary, G.W., "Teaching Module for Laminar and Turbulent Forced Convection on a Flat Plate," *Computer Applications in Engineering Education*, Vol. 6, No.2, pp. 115-125,1998.

7. Ribando, R.J., O'Leary, G.W, and Carlson, S.E., "A General, Numerical Scheme for Heat Exchanger Thermal Analysis and Design," *Computer Applications in Engineering Education,* Vol. 5, pp. 231-242, 1997.

8. Ribando, R.J., "An Excel/Visual Basic for Applications (VBA) Primer," *Computers in Education Journal,* Vol. VIII, No. 2, April-June 1998, pp. 38-43.

A complete description of each module, full-color screenshots of each interface and references to journal articles describing many of the modules may be found at the author's website: http://www.people.virginia.edu/~rjr/modules


Boundary Layer Over a Heated Flat Plate
Exit
Print
Save Data
Help
Temperature Contours (color) and Hydrodynamic BL Edge (white line)
1.0
0.0
Plot

Velocity Probe

X = 0.884
Y = 0.003
U = 0.643

Student Name

Wall Thermal Conditions:

X-Loc'n	dT/dy	Temp
0.675	-315.15	1.000

(Blue = Input, Red = Output)

Physical Parameters:

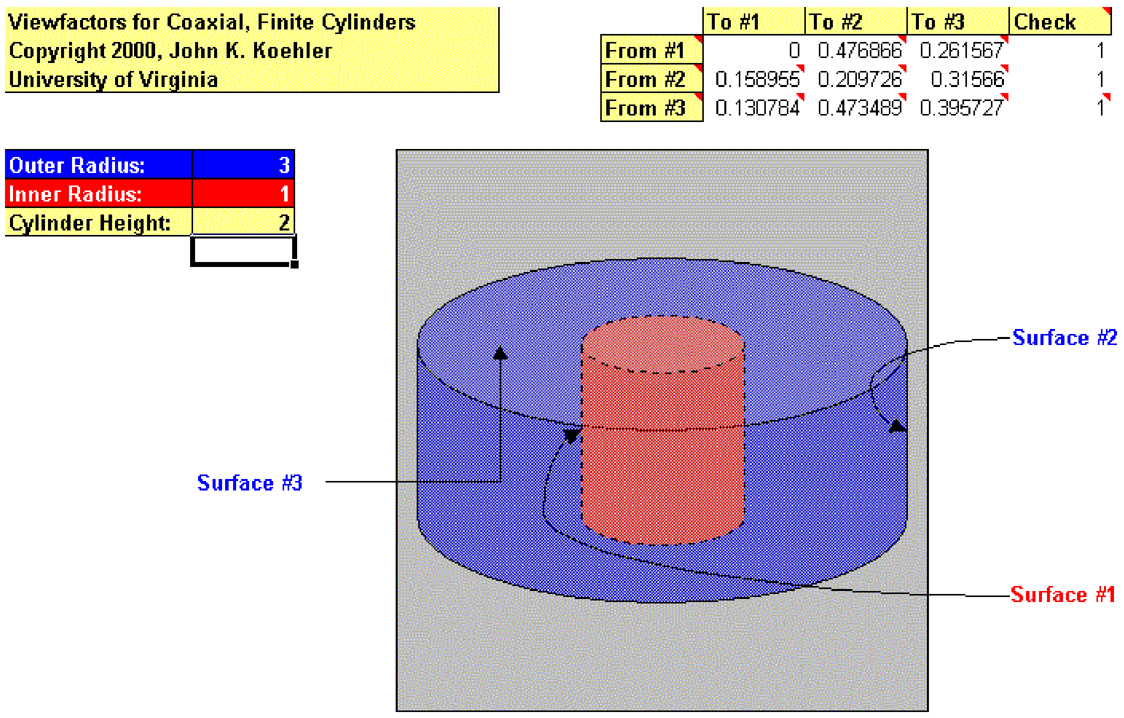
Reynolds Number =	411,000
Prandtl Number =	0.710
Turb. Intensity =	0.0100

[✓ Set Thermal BC's]

Plotting Parameters:

Y-Scaling Factor = 15.0

Figure 1. Interface for Forced Convection over a Flat Plate Module



Viewfactors for Coaxial, Finite Cylinders
Copyright 2000, John K. Koehler
University of Virginia

	To #1	To #2	To #3	Check
From #1	0	0.476866	0.261567	1
From #2	0.158955	0.209726	0.31566	1
From #3	0.130784	0.473489	0.395727	1

Outer Radius:	3
Inner Radius:	1
Cylinder Height:	2

Figure 2. Interface for Viewfactors Spreadsheet

Biographies

ROBERT J. RIBANDO
Robert J. Ribando is Director for Academic Outreach Programs in the School of Engineering and Applied Science and an associate professor in the Mechanical and Aerospace Engineering Department at the University of Virginia. In the former role he is in charge of our off-grounds master of engineering degree program, which is part of Virginia's Commonwealth Graduate Engineering Program. Over the last decade or so has had an active role in the development of infrastructure for making use of technology in instruction at the University.

TIMOTHY C. SCOTT
Timothy C. Scott is Instructional Laboratory Director and an associate professor in the Mechanical and Aerospace Engineering Department at the University of Virginia, where his primary interests are in the thermal sciences area. All three degrees are from the University of Michigan. Dr. Scott has 15 years experience in the automotive industry and continues to consult with his former employer.

GERALD W. O'LEARY
Gerald W. O'Leary is a Computer Systems Engineer in the Mechanical and Aerospace Engineering Department and the developer of all our Visual Basic interfaces. His degrees include a BS from the University of Notre Dame and an ME degree from U.Va. He has worked as a field engineer in the aerospace industry and as a systems engineer in the defense industry.