AC 2009-509: A HANDS-ON APPROACH TO COMPUTATIONAL METHODS IN ENGINEERING

Michael Gustafson, Duke University
MICHAEL R. GUSTAFSON II is an Assistant Professor of the Practice in the Department of Electrical and Computer Engineering at Duke University. His research interests include linear and non-linear control systems as well as curriculum development. He received his Ph.D. in Mechanical Engineering from Duke University.

Rebecca Simmons, Duke University
REBECCA SIMMONS is an Adjunct Assistant Professor with the Pratt School of Engineering at Duke University. Her research interests include computational modeling and experimental analysis of dynamic bubble systems. She received her Ph.D. in Mechanical Engineering from Duke University.

W. Neal Simmons, Duke University
W. NEAL SIMMONS is an Assistant Research Professor in the Department of Mechanical Engineering and Materials Science at Duke University. His research interests include failure analysis. He received his Ph.D. in Mechanical Engineering from Duke University.

Michael Ehrenfried, Kent Denver School
MICHAEL J. EHRENFRIED, JR. is Chair of Computer Studies at the Kent Denver School. He received his BSE in Electrical and Computer Engineering with a dual degree in Computer Science from Duke University.

Tod Laursen, Duke University
TOD A. LAURSEN is Professor and Chair of the Department of Mechanical Engineering and Materials Science at Duke University. His research interests include structural and solid mechanics, inelastic material modeling, large deformation kinematics, and finite-element concepts. He received his Ph.D. in Mechanical Engineering from Stanford University.

A Hands-On Approach to Computational Methods in Engineering

Abstract

The Pratt School of Engineering at Duke University has been actively focusing on the development of a variety of mechanisms to provide undergraduate engineering students with an earlier, more practical, experience with engineering concepts. Through these mechanisms, students are now exposed to elements from each of our four departments, which gives the students a clearer understanding of the field of engineering that they want to pursue. As a part of this ongoing effort, all engineering undergraduate students are now required, during their first year, to take a course on computational methods in engineering, EGR 53L. Because of the newfound prominence of EGR 53L in the curriculum, as well as the wide spectrum of student interests and backgrounds, the course has undergone several significant changes in the past five years aimed at improving the student experience and exposure to engineering. This paper outlines several key facets of our approach to redesigning the course and also reports on outcomes and student assessments of the hands-on portion of the revised experience.

Introduction

Undergraduate engineering students at Duke have long been required to take a full-credit course in either computational methods or computer programming. A full-credit lab course at Duke, denoted by the "L" after the course number, is the equivalent of four credit hours. As recently as five years ago, there were several different offerings that would fulfill this requirement. Over the past 20 years, several programming languages have been used in these courses – Fortran, Pascal, C, C++, and Java – and students could take the courses through the computer science department or the school of engineering. Furthermore, students could use AP credit to satisfy the requirement. This model led to a large disparity in the experiences of undergraduates – sometimes even those students pursuing the same field of engineering. Also, given the foundational requirements of chemistry, mathematics, and physics, undergraduate engineering students would often take no courses in engineering until their sophomore year. As a result, in many cases students would drop out of the school of engineering to pursue degrees in the social sciences, natural sciences, or humanities, without ever having experienced what engineering could be.

Beyond that, instructors in future classes could not count on any uniformity of programming skill, experience, or even language among students. A faculty member who wanted students to complete an assignment with a computational tool often had to find time in their course to instruct students on how to use that tool, reducing the amount of time spent on analyzing the primary material in the course. Furthermore, in addition to the traditional emphasis on programming and simulation in such courses, we felt that a true introduction to computation should devote attention to other, equally important, computational issues in engineering, for example, computerized data acquisition and actuation and control of elementary devices. All of these factors led to a comprehensive redesign of the first-year computational methods course offered in the engineering school. Each department then changed its curriculum requirements to mandate that all engineering students take the new course - neither computer science courses nor

AP credits can be used to replace it. What follows is a discussion of the redesign of the course, information about the addition of hands-on laboratories, and an assessment of the success of those laboratory experiences.

Redesign

As a first step in the redesign process, eight years ago, the associate dean for the school formed a committee to evaluate the curriculum of the freshman EGR 53L class. Faculty members were polled as to the future usability of the language being taught in EGR 53L. At the time, students would learn either C or C++. While the faculty generally agreed that students planning to pursue graduate school in most engineering fields would need to know a compiled language, they also felt that a programming package such as MATLAB would be more globally useful during their undergraduate career – reflecting the conclusions of Bjedov and Anderson[1] as well as those of Azemi and Pauley in a more recent study[2]. Furthermore, given the vast differences in programming experience of entering freshman students, a language like MATLAB was seen as presenting similar educational benefits to a compiled language, without as steep a learning curve. Thus, the course has been completely revised to utilize MATLAB. One benefit of this approach has been the inclusion or expansion of MATLAB use in several other engineering courses[3,4] as well as in math and physics courses. This vertical integration of the foundations of computational methods with upper level offerings allows students to continue using the computational skills throughout their undergraduate career while also providing instructors in other courses the opportunity to leverage the skills learned in EGR 53L.

Second, a weekly recitation was added to give students an opportunity to work through shorter programming assignments with assistance from undergraduate teaching assistants. This change was motivated by surveys that asked students about their overall educational experience in EGR 53L. While the results of those surveys were *generally* positive, several students indicated that they did not receive enough directed instruction regarding the use of individual programming tools, structures, and methods in the course. At the time, homework assignments were typically centered on solving a problem that focused on a single engineering or science concept and required a particular method to be used in only one specific way to solve a problem. In end-of-year surveys, students indicated that they wanted to be exposed to a broader range of uses for the various computational methods. With the inclusion of the current recitation sections, students are learning more about the range of problems for which a particular method may prove helpful and expanding their intuition and their aptitude in using that method. Along with this change, two textbooks were identified for the course – *Applied Numerical Methods with MATLAB*[5] by Steven C. Chapra and *Introduction to MATLAB 7 for Engineers*[6] by William J. Palm III. While the addition of a recitation increased the amount of time per week students spend in class, students taking EGR 53L still receive one credit only. This is consistent with other courses at Duke that have lectures, laboratories, and recitation sections.

Finally, from teacher-course evaluations and anecdotal information, it was clear that students were interested in how computational methods are useful in the real world. Rather than simply solving problems on the computer, they wanted to analyze and control actual devices. To accommodate this desire, several hands-on experiments were added to the laboratory assignments that capitalize on the tight integration between MATLAB programming and its Data

Acquisition Toolbox. The new laboratories allow students to control voltages or take voltage data from various devices, process the data using various computational methods covered in lecture, and characterize real systems such as light emitting diodes and pressure sensors. This gives the students practical experience, combining the methods they have learned with analysis of actual data from physical devices. Below is a discussion of the first three of these hands-on experiences, followed by survey data supporting the notion that adding hands-on experiences to our first-year engineering course has improved student reception.

Data Acquisition Labs

Currently, there are five data acquisition labs in the course –

- DAQ 1: Introduction to Data Acquisition: Digital I/O
- DAQ 2: Introduction to Data Acquisition: Analog I/O
- DAQ 3: Calibration of a Pressure Sensor
- DAQ 4: Introduction to Data Acquisition: Synchronous I/O
- DAQ 5: Aliasing and Frequency Space

This paper will look at the first three; a future work will present the latter two.

Students work in groups of two – or at most three – at stations equipped with two PCs running Windows. At each of these stations, one of the two computers has a multifunction data acquisition card (DAQ)[7] that has eight programmable digital I/O lines, two analog output channels, and 16 single-ended analog input channels. The five labs are meant to introduce students to each of these functions and then to apply them in a way that is relevant to the material in the course.

For DAQ 1, the digital I/O lab, students learn how to use MATLAB to control the data acquisition card. They also get an introduction to basic electronics – such as how breadboards work, what resistors are, how nominal resistance is read, and what LEDs are. The lab comes at a time in the course when students have just finished learning about the binary system and how integers and floating-point numbers are stored in MATLAB. The lab manual describes how to build a circuit using three of the digital I/O lines to turn three LEDs on independently while three other digital I/O lines are set to measure the voltages. For the assignment, students are required to add an additional light and measurement to the breadboard and then alter the program to control the light. They must then use the four-light system to create a binary-to-decimal conversion game where the system will pick a random number between 0 and 15 and display it on the lights. Their program must solicit a guess from the user and then report if the answer is correct. Through this exercise, students not only begin to see directly that MATLAB can be used to affect real-world devices but also have their basic programming skills reinforced – specifically, using logical operators, building iterative structures, and understanding number systems.

DAQ 2 and DAQ 3 are closely related in that they use the DAQ card's analog channels to asynchronously set and measure voltages. DAQ 2 is also an extension of DAQ 1 in that students are again examining how LEDs work. This time, however, instead of merely turning lights on and off, students write programs that sweep through various voltages to see how they alter the brightness of the LED. Students are also required to determine graphically the turn-on voltage of between four and six LEDs – differentiated by color, size, and shape – and then draw conclusions as to the determining factor or factors for the turn-on voltage. For this lab, not only do students discover the inherently nonlinear relationship between voltage and current that a diode possesses, they also reinforce iterative programming, data extraction and routing, and the power of using plots to graphically determine solutions. In addition, they learn that MATLAB, in combination with the Data Acquisition Toolbox, can be used to characterize physical devices such as sensors – a conclusion that leads directly to the third DAQ lab.

DAQ 3 involves calibrating a pressure sensor so that MATLAB can be used to determine the volume of fluid in a graduated cylinder. The sensor – a PX 26 differential pressure sensor from Omega[8] - can be purchased in several configurations; our students use the PX26-005GV which, when powered with 10 V, measures 10 mV per psi up to 5 psi. The graduated cylinders are 1 L and made of plastic. The graduated cylinders have a port drilled at the 100 mL mark and this port is where the pressure sensor is positioned. During this lab, students are required to accomplish several tasks:

- Determine the appropriateness of the sensor for the task at hand, including finding the height of fluid that could be measured with a 5 psi gage and comparing it to the height of their cylinder.
- Determine if the gage used is the *most* appropriate gage for the task. This question is focused on the notions of *range* versus *precision* in the experiment.
- Complete a MATLAB program to take two sets of voltage measurements across the sensor for various fluid heights. As a part of this exercise, students must write a loop to obtain multiple readings and average them to eliminate noise. One set is taken over a wide range of volumes while the other is in a narrow range.
- Complete a MATLAB program to use least-squares-fit to obtain coefficients for volume as a function of voltage and save these coefficients for both sets of voltage measurements. Because of the order of the fit and the size of the volume offset versus the volume deviation in the second data set, MATLAB will report that the fit is ill-conditioned, reinforcing lectures where students are shown the potential pitfalls of polynomial fitting, specifically with respect to narrowly spaced data relatively far from the origin.
- Use a MATLAB program that reads their coefficients and presents a real-time assessment of the fluid in the cylinder. Generally, the wide-range coefficients present good readings over the entire cylinder while the narrow range is accurate only within the confines of the calibration run.
- Finally, complete a MATLAB program to use least-squares-fit to obtain coefficients for voltage as a function of pressure and compare these coefficients to the manufacturer's specifications of 10 mV per psi.

Assessment

During the past five years the students have been surveyed on how well the course has satisfied the following learning objectives, on a scale of 1-7 with 1 meaning "very poorly" and 7 meaning "very well":

1) Interpret engineering problem statements and solve them using basic MATLAB programs,

2) Model physical systems and optimize parameters using iterative structures,
3) Solve engineering design problems using numerical integration, roots of equations, simultaneous equation solving, finite difference methods, matrix analysis, linear programming, dynamic programming, and heuristic solutions, and
4) Prepare documentation of engineering design solutions using a document preparation system,

The academic-year averages to these questions are:

Year	02-03	03-04	04-05	05-06	06-07	07-08
		DAQs Added in Fall, 2003				
Enrollment	203	210	289	392	334	365
Responses	175	186	252	320	293	300
Objective 1	5.23	5.87	5.93	5.90	5.84	5.99
Objective 2	4.72	5.42	5.25	5.50	5.34	5.63
Objective 3	4.85	5.44	5.77	5.75	5.76	5.87
Objective 4	5.97	5.91	5.99	6.05	5.58	5.80

These results show a major step-change in students' assessment of modeling physical systems and of solving engineering design problems. With respect to the Data Acquisition Labs, students report that the labs generally improve their perceptions of the course. Specifically, on a survey with questions ranging from their background in math and physics to their interest in being a teaching assistant for the course, the response averages for the question "How did the DAQ labs affect your impression of the course?" for the past five years' of survey data – totaling over 1400 students – are:

- 37.43%: Made it much better
- 43.24%: Made it somewhat better
- 12.12%: No impact
- 5.75%: Made it somewhat worse
- 0.84%: Made it much worse
- 0.62%: No response

For the question "How well did the DAQ labs accomplish the goal of exposing you to basic DAQ concepts with MATLAB?" the average grades given the course are:

- 43.31%: A
- 42.12%: B
- 11.78%: C
- 2.03%: D
- 0.28%: F
- 0.48%: No response

Though there is clearly room for improvement, the surveys indicate that the work done thus far has led to an increase in students' satisfaction with the course as well as their self-rated abilities with computational methods.

Conclusions

In this work, the evolution of the curriculum and the practical experience provided by our undergraduate course in computational methods have been described. The class has evolved from a hodgepodge of several course options in different languages to a single unified course teaching one practical language, MATLAB. As a part of this change, we have designed a series of hands-on laboratory exercises to give students practical experience in writing and working with programs that interact with different physical devices. The inclusion of data acquisition and analysis laboratories has clearly improved students' perception of what they are able to accomplish upon leaving the class and their overall interest in engineering. There has also been a marked improvement in student satisfaction with this first year computational class.

Acknowledgments

We would like to acknowledge the support of the Office of the Dean of the Pratt School of Engineering in making the additions to this course possible. We would also like to recognize the invaluable contributions of the undergraduate teaching assistants. Finally, we would like to thank the reviewers for their helpful comments and Mary Lindblad for her editorial advice.

---

[1] Bjedov, G. and Anderson, P.K., *Should Freshman Engineering Students Be Taught a Programming Language?*, Proceedings of the 26th Frontiers in Education Conference, 1996, pp. 90-92.
[2] Azemi, A. and Pauley, L.L., *Teaching the Introductory Computer Programming Course for Engineers Using Matlab*, Proceedings of the 38th ASEE/IEEE Frontiers in Education Conference, 2008, pp. T3B-18—21.
[3] Huettel, L.G. and Collins, L.M., *A vertically-integrated application-driven signal processing laboratory*, ASEE Annual Conference and Exposition, Conference Proceedings (2005), pp. 15613 – 15623.
[4] Huettel, L.G., et al., *Work in Progress: Theme-Based Redesign of an Electrical and Computer Engineering Curriculum*, Proceedings of the 34th ASEE/IEEE Frontiers in Education Conference, 2004, pp. S2C-1—2.
[5] Chapra, S.C., *Applied Numerical Methods with MATLAB for Engineers and Scientists.* McGraw-Hill, 2nd ed., 2008.
[6] Palm, W.J. III, *Introduction to MATLAB 7 for Engineers*. McGraw-Hill, 2005.
[7] National Instruments, *BSeries Multifunction DAQ, 16-Bit, 200 kS/s, 16 Analog Inputs*, specifications online at http://www.ni.com/pdf/products/us/4daqsc208-209_212-213_230.pdf.
[8] Omega, *PX 26 Low Cost Wet/Wet Differential Pressure Sensor*, specifications online at http://www.omega.com/Pressure/pdf/PX26.pdf.