Investigations Concerning Pedagogical Strategies Promoting Engineering Students' Ability to Solve Open-ended Problems

Christian Hipp[1], Veronica Addison[2]

Abstract - This paper describes an on-going, NSF funded, exploratory research project in the Course, Curriculum, and Laboratory Improvement Program (CCLI). Though the project currently is still collecting and analyzing data, this paper seeks to relate the project's justification, goals, and methods. The research objective is to determine the best ways to introduce computing into early undergraduate mechanical engineering curriculum, focusing particularly on numerical methods and analysis. Given the importance of computing in professional engineering practice, this project seeks to improve students' facility with computers while moving away from 'cookbook' approaches which emphasize software-specific skills at the expense of more fundamental mathematical and conceptual knowledge.

Keywords: open-ended problems, computer, numerical methods

INTRODUCTION

When students first learn to use computers in the engineering classroom, they often learn to use the computer in a stepwise manner. However, this approach may hinder their abilities to think about the kinds of open-ended problems they will face as professional engineers. Students expecting such a 'plug-and-chug' approach are lost in the workplace – they do not solve problems well because they do not know how to start. This research project focuses on finding ways to efficiently develop skills that will enable students to solve more complex problems as well as get students to think more creatively about computer-aided approaches while seeing the problems with 'cookbook' approaches for themselves.

To this end, this project's investigations are structured by four central research questions: (1) What computer experiences (STEM -- Science, Technology, Engineering, Mathematics -- or otherwise) do students have when they enter college-level engineering class? (2) In what ways does varying the timing of the introduction of computer techniques affect students' expectations and creative use of these methods? (3) In what kinds of problems does the computer specifically enhance understanding? In what kinds of problems does the computer act as an obstacle to understanding? (4) How can we emphasize the importance of setting up problems for computer-aided solutions instead of emphasizing the results of the process? These questions are investigated through a sophomore course in numerical analysis. All four areas are the source of several experiments using different teaching strategies and methods.

---

[1] Christian Hipp, Department of Philosophy, University of South Carolina, Columbia, SC 29028. hipp_c@bellsouth.net.

[2] Veronica Addison, Department of Mechanical Engineering, University of South Carolina, Columbia, SC 29208. ADDISONV@engr.sc.edu.

CURRENT CONTEXT

The Department of Mechanical Engineering at the University of South Carolina includes about 300 undergraduate students. Since 85% of our undergraduate students directly enter the workforce, our program's primary purpose is on preparing students for engineering practice. Practicing mechanical engineers use computers extensively (*e.g.*, design processes, manufacturing equipment, robotics, analysis of heat transfer problems, etc.). Over the past 20 years, computational methods have been gradually integrated into the curriculum. Unfortunately, little consideration was given as to how to structure the learning environment for students to efficiently and effectively gain the skills they will need in practice. Parts of our curriculum often overemphasize rote solution techniques for deterministic problems, which caricatures the complexities of problems engineers face when they enter the workforce.

Today, students have lived their whole lives in an era of 'everyday computing.' Thus, when they take college engineering classes, professors are hardly 'introducing' them to computers; rather, they are showing them new applications. Engineering educators need to understand the pre-existing experiences and assumptions students have about computing and exploit those tendencies for better comprehension of engineering concepts whether computer-aided or not. The primary objective of this project is to better understand the processes by which engineering students encounter and learn to use computers for engineering problem solving.

RESEARCH PLAN

The platform for the four research questions noted in the introduction is Dr. Sarah Baxter's sophomore Numerical Methods course (EMCH 201). Along with a freshman course in Graphics and Visualization, Numerical Methods constitutes one of the two engineering courses in which beginning students first use computers for engineering applications. The technical platform in EMCH 201 is a symbolic manipulator, currently Mathcad.[3]

Students' prior computer experiences as they enter college-level engineering classes

The first step in this research project was to determine what computer experiences (STEM and otherwise) students have when they enter college-level engineering classes. We surveyed students in both freshman and sophomore engineering to gauge both their previous experiences with computers and their assumptions about how to use them, what computers do well, and when, whether, and why they trust the results provided by computers. The surveys focused on three issues. We queried students about their computer experience for STEM applications and more generally. We determined the assumptions students have about the ways computers are used and the kind of problems they can solve. Finally, the surveys helped establish the extent to which students place unreasonable trust in computer-generated solutions.[4]

[3] Our research questions are not software specific and would be equally served by MatLab or Mathematica. Also, the concepts introduced include consideration of programming languages independent of these packages (*e.g.*, Visual Basic, Fortran, or C).

[4] Others have argued that unwarranted trust in the machine is problematic (Clough 1990). If instructors have some degree of clarity about the extent of unwarranted trust, then particular pedagogical strategies may help instructors counter it.

Timing and Order of Introducing Computer Techniques

The timing and order of presentation of concepts and skills is a critical component in engineering classes. Two approaches are common in teaching numerical methods: 1) traditional lecture, including theory, examples, and homework to be carried out "by hand," and 2) computer based courses, where problems are introduced immediately on the computer (Merino & Abel 2003). This project focuses on the design of the timing component of this experiment, specifically how to plan the course in order to study how varying the timing of the introduction of computer techniques affects student expectations and creative use of the methods.

In this project, multiple steps for each conceptual topic were established that correspond to the ideas of "by hand" first or "computer first." The class was split into two equal and representative halves. For one group, solving non-linear equations (e.g., root-finding) was initially presented by chalkboard lectures, followed by written homework stressing problem set-up and the iterative process and concluded with a larger scale project that could be implemented on the computer. For the alternate group, non-linear equations were initially introduced graphically on the computer, and illustrated as a root problem.

Assessment of student skills was evaluated through several avenues. Graduate students were used as participant-observers taking each of the classes in question. They keep class-by-class records of the students' classroom questions and reactions, as well as their own impressions of the student class and computer laboratory experiences and responses to instructional choices. Student skills are also assessed through analysis of computer-based worksheets, homework, and written tests. We hope use these assessments to piece together a picture of the cognitive effects of varying instructional methods.

Comprehension through Problem-solving: Computer Enhanced or Hindered?

Certain types of problems, when solved using computer-aided methods can lead students to a higher conceptual understanding (Hall & Obregon 2002, Burton, *et al* 2004).[5] However, sometimes students fail to learn to visualize three dimensions in their mind because computer use eclipsed this skill (Ferguson 1994). These are likewise our intuitions, reinforced with experience.

At the sophomore level, the most significant contribution computational software makes to understanding is the ability to visualize problems and solutions. Engineering students need to visualize mathematical entities that have specific physical meaning. Computational software also facilitates understanding by demonstrating somewhat less tangible theoretical concepts – *e.g.*, limits, convergence, divergence, or oscillation – (Burton, *et al* 2004). Yet, many ideas do not register with students when they are only presented with the computer-aided version. For example, iterative processes typically do not make sense until a student has to work through a couple of iterations by hand. Similarly, an understanding of the programming constructs of loops and conditional statements, initially demand working out by hand.

This project will compare outcomes regarding the kinds of problems computers specifically enhance as well as problems in which the computer acts as an obstacle to understanding. This consideration will be assisted via student discussion as to their perceived level of understanding.

[5] Some examples include Fourier series calculation, root finding, and finite element approach to trusses.

There will also be a comparison of answers to a question examined by two different groups within the same class who can self-assess when understanding of a particular issue came. Finally, observations by participant research assistants who note the kinds of questions and answers prompted by the students will further assist us in answering this research question.

Emphasis of Problem Design for Computer-aided Solutions, not Process Result

Ideally, students should be able to recognize components of an engineering problem that could be reformulated or approached in multiple ways and, subsequently select the best approach to fit the context and resources at hand. However, to expect students to gain this flexibility and creativity as a by-product of engineering instruction is naïve (MacNeal 1987). Facility with open-ended problems must be constructed as an instructional goal. Thus, we examine the effect of emphasizing problem set-up, in particular with more open-ended problems, rather than the actual answer.

Traditional instructional approaches often work by rote, focusing on presenting multiple examples. This approach builds a ‘problem-type’ base, prompting them to identify the patterns which exist between problems of a similar type. In this project, we ask whether instruction in computer-aided methods produces the necessary reflexivity to allow this type of pedagogy to work effectively. In order to emphasize the difference between formulating a problem (the set-up) and producing a numerical solution (*i.e.*, entering variables into the computer to produce a number), we propose to re-establish the priority of problem set-up and to develop appropriate sophomore-level open-ended problems that require focus on the problem set-up procedure.

Two approaches were applied to emphasize set up over results. In one approach, both the lecture and the homework will be non-computer based. A series of problems will be set-up to the point that they could be solved using the computer, but the work assessed only on the set-up. An alternative approach will be to present more open-ended problems – *i.e*., problems not identified by type, not presented in similar format, perhaps embedded in more complex topics or problems, etc. This approach will force students to use the problem-solving procedures to determine type and recognize problems as similar.

ASSESSMENT METHOD

To provide a strong evidentiary base to support sustained improvement in the use of computers in engineering education through open-ended problems, we used multiple assessment instruments to tack the experiences and outcomes of participating students – student scores, surveys, focus groups, interviews, and observations. The following paragraphs describe the data collection techniques and how they will be implemented to ensure valid, reliable, and generalizable results are found.

Student Scores

A comparison of student scores on homework, tests, and projects, will help to determine the validity and extent to which the interventions described have an effect upon the specific intended domain of content.

Surveys

Surveys were used to collect data on multiple research questions and at multiple times throughout the project. Surveys are used because they contribute to the reliability of results by presenting all subjects with a standardized stimulus. Additionally, the validity of the survey results will be established in part by interviewing a sample of the students surveyed.

Participant Observers (PO)

Two graduate students supported by the project are the primary participant observers. The POs attend all lectures in the numerical methods course, survey assignments and tests, and engage with the students enrolled in course. The POs will keep short notes of their observation during class, expanded notes that they will record in weekly journals, and provisional running records of analysis and interpretation that will form the basis of their research. They will reflect upon their observations of student performance and stumbling blocks, as well as their own experiences. They will also participate in the development of survey, interview, and focus group questions to both improve the generalizability of the research and enhance their own professional development.

Interview and Focus Groups

Interviews and focus groups of students will be conducted to collect data that will help explore causal relationships between instructional approaches and student learning outcomes. Since there may be multiple explanations for why student performance on particular topics can vary in addition to the intervention being studied, alternatives will be explored in these interviews and groups and the results compared with other assessment measures. The groups were formed from student participants randomly selected to collect data and to vet the findings and conclusions of the participant observers and the results of the student surveys.

CONCLUSION

The intellectual merit of this project stems from the research itself. Arguably, the computer has rendered a countless number of previously intractable engineering problems solvable. Yet, learning how to use the computer in engineering pedagogy has lagged behind learning to make use of it in professional practice (Jones 1998). By undertaking some focused research on undergraduate teaching and learning, we hope to both challenge and confirm some of our intuitions about how computers either improved or hinder conceptual understanding.

Furthermore, this research represents not only collaboration between researchers in the history and philosophy of technology and mechanical engineering but also a partnership between two different colleges at the University of South Carolina. This alliance is promising for the two graduate students (one from philosophy, the other from mechanical engineering) who will learn firsthand about such interdisciplinary collaboration. Additionally, the project models interdisciplinary teamwork to undergraduates. Finally, this research will be disseminated through at least two distinctly different literatures – engineering education and science and technology studies. The gender distribution of the research team also broadens the participation of women in engineering.

The impact this research has on students is significant in at least two ways. First, the research on students' previous computer experience will help to increase awareness of potential engineering student geographical location, by tailoring engineering education to meet the needs of students from under-resourced school districts.[6] Second, the broadest impact of this project will come from the greater flexibility and facility that our students will possess as a result of this research and its longer term outcomes. Students with greater facility in setting up problems for computer-aided solutions will also have a greater facility in learning new software, as they will be less wedded to the idiosyncratic operations of any single software package. Companies employing these engineers will have a work force that offers greater flexibility in making choices about the use of software in their firms.

REFERENCES

Burton, Leone, Falk Lennert and Sven Jarner. "Too Much, too seldom," *International Journal of Mathematical Education in Science and Technology*, 2004, 219-226.

Ferguson, Eugene. *Engineering and the Mind's Eye*, Cambridge, MIT Press, 1994.

Hall, Kevin W. and Rafael Obregon "Applications and Tools for Design and Visualization," *The Technology Teacher*, 61:7, 2002, 7-11.

Jones, J.B. "The Non-Use of Computers in Undergraduate Engineering Science Courses," *Journal of Engineering Education*, 1998, 11-14.

MacNeal, Richard H. "Zen and the Art of Finite Element Design," *Finite Elements in Analysis and Design*, 1987, 85-91.

Merino, Donald N. and Kate Abel. "Evaluating the Effectives of Computer Tutorials versus Traditional Lecturing in Accounting Topics," *Journal of Engineering Education*, 2003, 189-194.

[6] This phenomenon occurs in South Carolina because many students are minority students and from rural school districts.