AC 2012-4402: IMPROVEMENTS IN COMPUTATIONAL METHODS COURSES IN CHEMICAL ENGINEERING

Dr. Joshua A. Enszer, University of Maryland Baltimore County

Joshua Enszer is a full-time lecturer in chemical engineering at the University of Maryland, Baltimore County. He has taught core and elective courses across the curriculum, from introduction to engineering science and material and energy balances to process control and modeling of chemical and environmental systems. His research interests include technology and learning in various incarnations: electronic portfolios as a means for assessment and professional development, implementation of computational tools across the chemical engineering curriculum, and game-based learning.

Dr. Victoria E. Goodrich, University of Notre Dame
Dr. Rachel B. Getman, Clemson University

Improvements in Computational Methods Courses in Chemical Engineering

Abstract

As more core courses in the undergraduate curriculum require significant ability using computer skills, we see a need for improved methods of instruction in computer methods courses required in the chemical engineering curriculum. It is important to provide students with a series of approaches and activities that ensure (1) that students are applying algorithmic thinking and not just learning how to operate in a single computing environment, (2) that students are able to develop their abilities to formulate problems in a computational context, and (3) that students are applying numerical algorithms in meaningful ways and not just following a template. To that end, we discuss here overviews of our computational methods courses and implementation that encourages behavior independent of choice of computer software.

We also share here our plans to evaluate student abilities and perceptions in courses following computer methods to assess the effectiveness of the courses. We share results here from preliminary self-assessment surveys plus plans for implementation for the Spring 2012 semester.

Introduction

Likely motivated by significant improvements in functionality and user-friendliness, computational software has become ubiquitous in engineering education. This has undeniably enhanced the quality of education, as class time that was once spent teaching numerical methods and computer syntax can now be spent using software to illustrate examples and explain complex phenomena. [1, 2] Seemingly, an advantage to this transition is that engineering curricula can focus more specifically on the “science,” as a student with even a novice-level understanding of numerical methods and proper programming can use software to solve and analyze a variety of engineering problems.

Many engineering programs incorporate courses in computer methods early in the curricula. A primary goal of these courses is to train students how to use software to solve problems that they will encounter later in the curricula. While different across disciplines and schools, in general, these classes have a variety of common themes, i.e., linear regression, statistics, ordinary differential equations, etc., and use similar software, i.e., Excel (http://office.microsoft.com/en-us/excel/), MatLab (http://www.mathworks.com/products/matlab/), MathCAD (http://www.ptc.com/products/mathcad/), Maple (http://www.maplesoft.com/products/maple/), Mathematica (http://www.wolfram.com/mathematica/), etc. Over time, these courses have shifted focus almost entirely to training students on how to use these software packages, replacing the numerical methods courses that predated the ubiquity of user-friendly computational software.

When engineering graduates enter the workforce, they will solve a variety of problems requiring a variety of software, which may be quite different than the problems they encountered during their baccalaureate. In fact, a recent study performed by Vengara et al. [3] indicates that most employers seek workers who are highly proficient using Excel, but that there is no general preference for any other specific software or type of software, with the exception of some sort of computer aided design (CAD). Instead, employers seek workers who can teach themselves how to use new software when necessary. Lowe et al.[4] point out that teaching students how to use specific codes to solve specific problems, as can be the scenario in today's computer methods courses, is of little value in such situations, i.e., when other types of problems need to be solved or other software needs to be used. Thus contemporary computer methods courses may not teach students the flexibility they will ultimately need in their careers.

In this paper, we describe the efforts taken in three different introductory courses at three different universities in different parts of the United States to address the need for students to develop skills independent of specific computer software. We detail here common initiatives in surveying student abilities and perceptions as they relate to computer methods.

Methods

The University of Notre Dame is a medium sized, Midwestern, private institution with a traditional student composition, i.e. the vast majority of students complete their undergraduate studies in four years and are in the age range of 18-22. The overall student body is 53% male and 47% female, while the College of Engineering is approximately 75% male and 25% female. In chemical engineering the student population is 55% male and 45% female with 21% minorities.

In terms of institutional structure, first-year students are admitted to the separate First-Year of Studies program regardless of their intended future major. Students select their major (whether engineering or something else) near the end of their first-year when they register for classes for the upcoming fall semester. With few exceptions, students that are considering an academic pathway within engineering complete a standard first-year curriculum, including the two-semester course sequence "Introduction to Engineering." They then pass into the college of their selection in their sophomore year. Beginning in their sophomore year and until they graduate, students are institutionally recognized by their college, which, in the case of this study, is the College of Engineering; and by their specific engineering discipline within. However, beyond admission / selection into the university as a whole, there are no admission or selection criteria for entering any of the disciplines of engineering; rather, it is based on student interest alone.

The chemical engineering computer methods course at Notre Dame is a three-credit, lecture-based course that is taken in the spring of the sophomore year. The course includes major semester projects that motivate the selection and timing of computational topics covered in the course, which include statistics, differential equations, and optimization. Regular homework assignments and exams are used to give students more practice and exposure to computational techniques. Applications from junior- and senior-level courses are distilled to their mathematical component for examples.

The University of Maryland, Baltimore County is a medium-sized, Northeastern, public institution whose student body is comprised of approximately 75% full time and 25% part time students. The overall student population is 53% male and 47% female and is diverse with about 40% of students representing minority populations. In chemical engineering, the student population is 63% male and 37% female with 46% minorities.

First-year students may designate chemical engineering as their major upon entry, provided they are prepared to enroll in the required math, science, and engineering courses (the "academic gateway"). Each engineering department has slightly different admission criteria regarding completion of the gateway requirements. Upon completing the academic gateway with at least C's, students are permitted to take chemical engineering courses.

The computational methods course at UMBC is a required course typically taken in the sophomore year (its only official prerequisite is material and energy balances). It is a four credit course that meets three times a week for problem-solving sessions and once a week for a two-hour period in either a computer lab or experimental lab. The meetings are accompanied with short problems outside class with attention paid to frequency of recall instead of time on task. [5] The course is designed in part with "just-in-time" teaching strategies to solve engineering problems using statistical and programming ideas in the first half of the semester, and with an emphasis on retrieval in the second half of the semester, where students are expected to apply similar ideas in team settings as they conduct experiments of their own design. Each week of lab brings feedback from the previous week and a new assignment for more practice building on experimental design, statistical analysis, and communication skills. These skills are further tested in the senior level laboratory and design courses, which require the same reference texts.

Clemson University is a medium-sized, Southeastern, public land-grant state institution with a unique governance system: of the 13 members on the Board of Trustees, seven are life trustees who select their successors, and six are appointed by the State Legislature. The overall student population is 54% male and 46% female. Minorities comprise approximately 12% of the undergraduate population. The undergraduate population studying engineering is 80% male and 20% female, and the same is true in Chemical Engineering.

All students who desire to major in engineering are initially admitted into the General Engineering program, and they take a general engineering course in their first semester. Students must declare specific majors by the end of the spring semester of their first year. However, they can choose a specific discipline in the fall semester of their first year. Beyond the requirements for admission into the Institution as a whole, there are no specific requirements to get into Chemical Engineering. Those choosing Chemical Engineering in the fall take an introductory "Chemical Engineering Tools" course in the spring. Some of the other engineering departments have analogous classes. Students who do not declare a specific discipline in the fall, as well as those who declare disciplines without a specific "Engineering Tools" course, take a general engineering tools course in the spring. Students follow a traditional chemical engineering sequence in the sophomore year and beyond.

Clemson administers a two-credit Chemical Engineering Tools course in the spring semester of the freshman year. It is comprised of a one-hour lecture component and a two-hour lab component, each which meet once weekly. The lab involves experimental and computational tasks. In the beginning of the semester, very fundamental tools, such as measurements, units and conversions, are introduced. As the semester progresses, these evolve into engineering computations, graphing techniques, and statistical analysis. The primary objective of the course is to prepare students for future courses in the chemical engineering curriculum, which is accomplished by having students solve a variety of canonical chemical and engineering problems.

In order to better assess the methodologies used across institutions, a two-part assessment was shared for each institution. Students first completed a survey form self-assessment to describe their confidence in and perceived value of basic engineering computation. Additionally, while instructors at each institution have varying strategies to assess student ability through the semester, they have agreed to a set of common learning goals in order to compare and contrast results across courses:

- Convert chemical engineering problems into numerical problems.
- Apply an appropriate computational tool to solve a numerical problem.
- Describe a set of data using appropriate statistical measures.
- Interpret the results of data analysis.
- Evaluate the accuracy of a numerical claim.
- Communicate the results of computations in meaningful and effective ways.

A brief survey including these learning goals is used at the beginning and end of the semester to gauge student perceptions. The assessment is broken down into two parts: student ratings of their personal abilities in the learning goals and student perceptions of the importance of these skills to their career goals. The instrument consisted of Likert items using a scale of 0 (no ability or not important) to 5 (excellent ability or very important). A complete list of the survey items can be found in the Appendix. Surveys were completed as a part of a homework grade via Google Docs within the first two weeks of the semester and will be conducted again at the completion of the semester. Identifying information was recorded only so that the resulting changes in the individual student perceptions and self-assessed ability could be compared pairwise.

To measure student performance, the courses at all three institutions assigned a statistics-based project with an emphasis on creating and implementing algorithms. Each project has a basis in concepts of central tendency, curve fitting, and error analysis; however, projects were unique at each institution to account for large differences in classroom structure and expectations. A common rubric, which was shared between the courses, assessed students based on the same six learning goals.

Students were graded on a four point scale (novice, developing, satisfactory, and exemplary) for each ability. The rubric is in the Appendix. Results from the rubric assessment will be compared thematically for strengths and weaknesses after project completion.

Results and Discussion

In the Spring of 2011, a version of the survey was administered at Notre Dame to discern changes in student perceptions of their own abilities and the importance of skills related to engineering and computational methods. Based on the feedback from this survey, the survey was streamlined to the version that is implemented at all three institutions for the Spring of 2012.

In the 2011 version of the survey, there were several more survey questions, but four that matched the learning goals of this project. Data analysis and interpretation were bundled into a single survey item. The results from the relevant comparable questions are shown in Table 1 below.

Table 1: Mean Survey Results (Before and After Computer Methods) from University of Notre Dame in Spring 2011

Goal	Before		After	
	Ability	Importance	Ability	Importance
Formulate engineering problems	3.92	**4.30**	3.87	**3.87**
Use computer software to solve a problem	3.73	4.35	3.68	4.10
Communicate results and implications of computations	3.97	4.56	4.10	4.51
Perform analysis on experimental data and interpret results	4.00	**4.30**	3.78	**3.73**

Of the items above, by pairwise comparison of individual responses, only two exhibit a statistically significant (α=0.01) change; they are bolded in the table. In the case of both formulation of engineering problems and experimental data analysis, student perception of the importance of these items actually decreased when comparing the beginning and end of the semester.

The results of the survey administered at the start of the courses are shown in Table 2 below.

Table 2: Mean Survey Results for Student Perceptions Before Computer Methods, by Institution

	Notre Dame		UMBC		Clemson	
Goal	Ability	Importance	Ability	Importance	Ability	Importance
Formulate engineering problems	3.19	4.24	2.96	4.28	3.24	4.49
Use computer software to solve a problem	2.80	4.32	2.70	4.52	3.29	4.37
Communicate results and implications of computations	3.57	4.76	3.64	4.82	3.79	4.78
Perform analysis on experimental data	3.57	4.59	3.56	4.68	3.56	4.58
Interpret the results of data analysis	3.72	4.80	3.64	4.78	3.76	4.71
Evaluate the accuracy of a numerical claim	2.96	4.52	3.50	4.58	3.68	4.69

At the end of the semester, the survey will be administered again and pairwise comparisons will be made to track individual student perceptions and self-assessment results. This will also be compared against instructor direct assessment of student abilities to evaluate the alignment between the two assessment methods. The findings of surveys and direct assessment results will be compared among the institutions to inform future changes to the individual courses in the cases where student attitudes, opinions, and abilities significantly vary.

Conclusions

We discuss here the demographics and differences among approaches to teaching computational methods to chemical engineering undergraduates. We have prepared two instruments for assessing student ability and student self-assessment as they relate to computational thinking independent of specific computer program. Preliminary results from one institution show that student perception of the importance of computational ability actually decreased in the sophomore year. We will compare the approaches and attitudes of students at three institutions once data is available at the end of the spring 2012 semester.

References

1. Marchioro II, T. L.; Landau, R. H. Web-Based Education in Computational Science and Engineering. *IEEE Computational Science & Engineering, Based on presentations at the IEEE Computer Society Workshop on Computational Science & Engineering, Oct. 1996, Purdue University,* April-June 1997.
2. Thornton, L.; Nola, S.; E., G. R.; Asta, M.; B., O. G. Computational Materials Science and Engineering Education: A Survey of Trends and Needs. *Journal of Metals, Computational Materials Education* **2009,** *61* (12).
3. Vergara, C. E.; Urban-Lurain, M.; Dresen, C.; Coxen, T.; T., M.; Frazier, K.; Briedis, D.; N., B.; Esfahanian, A.; Paquette, L.; Sticklen, J.; LaPrad, J.; Wolff, T. F. Aligning Computing Education with Engineering Workforce Computational Needs: New Curricular Directions to Improve Computational Thinking in Engineering Graduates. *39th ASEE/IEEE Frontiers in Education Conference*, San Antonio, TX, October 18 – 21, 2009.
4. Lowe, D. B.; Scott, C. A.; Bagia, R. A Skills Development Framework for Learning Computing Tools in the Context of Engineering Practice. *European Journal of Engineering Education* **2000,** *25* (45).
5. Mastascusa, E. J. . S. W. J. . H. B. S. *Effective Instruction for STEM Disciplines;* Jossey-Bass: San Francisco, 2011.

Appendix

Table A 1. Common Rubric for Proficiency in Computational Methods for Chemical Engineers

Outcome	Novice (0 points)	Developing (1 point)	Satisfactory (2 points)	Exemplary (3 points)
Convert chemical engineering problems into numerical problems.	Unable to consistently draw appropriate process diagrams. Difficulty in starting a solution procedure.	Able to draw a process diagram with some error. No systematic problem solving behavior exhibited.	Able to draw an appropriate process diagram with little or no error and follow a template for a similar problem or follow a general problem solving algorithm.	Routinely able to draw an appropriate process diagram, perform degree of freedom analysis, and clearly present the solution procedure for solving a problem.
Apply an appropriate computational tool to solve a numerical problem.	Unable to determine how to start solving a problem numerically; requires explicit instruction in completing a task.	Able to modify existing computer programs to obtain a solution to a similar problem.	Able to enact a numerical method exclusively in one computer program but perhaps struggles to describe the method more universally.	Able to clearly communicate a numerical method independent of programming software; able to enact the method using at least one computer program.
Describe a set of data using measures of central tendency and variation when appropriate.	Cannot consistently complete statistical computations.	Able to compute some statistics for a data set as explicitly instructed.	Able to perform computational methods to produce statistics but may not select the most appropriate measures to characterize a data set.	Able to select appropriate measures to characterize a data set as completely as necessary.
Communicate results of computations in meaningful and effective ways.	Does not create legible and logical figures, tables, or equations.	Able to create rudimentary figures, tables, or equations but with no indication of reasoning as to the choice of data presentation.	Creates figures, tables, and equations to communicate results, but with some error, ambiguity, or obstacle toward clear understanding.	Selects and formats figures, tables, and equations as appropriate to clearly communicate a result.
Interpret the results from data analysis.	Does not perform statistical analysis or make any attempt to interpret a result.	Able to compute some but not all necessary statistical measures or tests to explain a result.	Performs appropriate statistical measurements and tests but may have difficulty explaining the result.	Performs appropriate statistical measurements and tests to make statistical claims and explains the implications of the result.
Evaluate the accuracy of a numerical claim.	Cannot provide relevant sources of experimental or computational error in solving a problem.	Provides mostly trivial sources of experimental or computational error. Does not attempt to quantify error.	Explains sources of experimental or computational error in analysis but perhaps includes trivial sources. Able to numerically estimate effects of error when prompted.	Explains the sources of experimental and/or computational error made in experimental analysis. Able to numerically estimate the effects of such error.

Survey of Student Self-Assessment and Perceptions

Which of the following career trajectories are you currently considering upon completion of your B.S. in chemical engineering? Please select all that you are considering.
Pharmaceuticals
Petrochemicals
Energy
Biotechnology
Environmental
Food Processing
Consumer Products
Industrial Management
Academia
Medicine
Law
Other (please specify)

For each of the following skills, please rate your perceived level of proficiency on a scale of 0 (no ability) to 5 (excellent ability).

Formulate engineering problems
Solve mathematical problems by hand
Solve mathematical problems using a computer
Select an appropriate piece of computer software to solve a given problem
Use a piece of computer software to solve a given problem
Communicate results and implications of computations
Perform data analysis on experimental data
Interpret the results of data analysis
Evaluate the accuracy of a numerical solution

For each of the following skills, please rate your perception of how important this skill will be for your career on a scale of 0 (not important at all) to 5 (very important).

Formulate engineering problems
Solve mathematical problems by hand
Solve mathematical problems using a computer
Select an appropriate piece of computer software to solve a given problem
Use a piece of computer software to solve a given problem
Communicate results and implications of computations
Perform data analysis on experimental data
Interpret the results of data analysis
Evaluate the accuracy of a numerical solution