Classical Solutions Are Like Classical Music: Both Pass the Test of Time

Robert J. Houghtalen, P.E., Michael A. Robinson, P.E.
Department of Civil Engineering
Rose-Hulman Institute of Technology

Abstract:
Classical solutions to civil engineering problems used to be commonplace in engineering practice. A decade ago, classical solutions could be found in most appendices of any engineering report. This is no longer true. Engineering practice is dominated by the use of design or analysis software, and engineering reports are replete with software-generated solutions to traditional civil engineering problems. In short, classical solution techniques have all but disappeared. Since computer solutions to engineering problems have become so dominant in engineering practice, it is fair to ask if classical solutions should continue to be taught in engineering classes.

This paper presents a three-theme approach to engineering education; exposure to background principles (theory), experience with classical solutions, and introduction to design (or analysis) software. The authors have used this technique for many years in undergraduate and graduate engineering education, as well as in ASCE continuing education classes. The rationale for the process is supported by the education literature and the interdependence of the three components is discussed. In addition, an ethical argument is made for the approach when contrasted with an educational approach dominated by exposure to design software. Finally, the results of two surveys are presented. One survey was given to undergraduate engineering students on the value of the three components of this teaching philosophy. The second survey, very similar to the first, was given to practicing engineers attending an ASCE continuing education seminar. The survey results appear to support the importance of teaching theory and classical solutions in order to appropriately use engineering software.

Introduction
There is little debate that engineering software has changed the face of engineering practice over the last few decades. It is also changing the face of engineering education. Personal computers (PCs) began to appear in design offices in the mid 1980s, and powerful engineering software for PCs began to appear by the early 1990s. Let's trace a little of the history of these changes through the eyes of the authors.

The senior author received his bachelor's degree from North Carolina State University in 1973. He worked a few years for the Corps of Engineers before returning to the same institution to receive his master's degree in 1977. While in school, the only computer training he received was a 2-credit course in elementary computer programming. FORTRAN was the computer language of choice at the time. No instruction was given in the use of computer software for the practice of civil engineering. Of course, very little computer software had been written at the time. All of the available classroom time was spent on theory and classical solutions.

In the senior author's work experiences, both before and after his master's degree, the only software he was exposed to in the practice of civil engineering was HEC-1 and HEC-2. These are well-know software packages developed by the U.S. Army Corps of Engineers for hydrologic analysis of watersheds and water surface profile determinations. The software was used on an IBM mainframe computer and required considerable training to use them properly. However, most of the engineering work he accomplished relied upon classical engineering solutions using equations and calculators.

Now fast forward a decade. The junior author received his bachelor's degree in chemical engineering from West Virginia Institute of Technology in 1985. He received his master's degree in environmental engineering from Virginia Tech in 1990. His exposure to computer programming and engineering software was more extensive. During his undergraduate study, he took a full semester of FORTRAN programming and had some exposure to spreadsheet software in his engineering labs. During his master's work, he used MathCAD extensively and off-the-shelf water quality modeling software. However, most of his classroom time was still spent on theory and classical solutions.

The junior author's work experience before and immediately after his master's degree relied upon engineering software and self-programmed solutions, particularly in MathCAD. The engineering software he used included air quality modeling on task-specific personal computers (PCs) and groundwater quality models. Some of the work he accomplished relied upon classical engineering solutions with equations and calculators.

Both authors had additional work experiences in the mid-1990s and saw the trend continue toward the use of engineering software and away from classical solutions. In addition, both taught engineering classes in 1990s and up through the present. There is no doubt that there is increasing pressure to teach students how to use modern engineering software. It comes from seeing first hand its role in professional practice and from outside sources. For example, the Accreditation Board for Engineering and Technology (ABET) requires engineering graduates to possess competency in the use of modern engineering tools in criterion 3k[1]. Our own Civil Engineering Board of Advisors and prospective employers would also like to see our students trained in the use of modern engineering software.

To make matters worse, the demands for classroom time have never been greater. ABET and ASCE are pushing communication, teamwork, and leadership skills along with a more well-rounded education. Those same organizations and industry are demanding that graduates be trained in the use of modern engineering software. To complicate the issue, state legislators are cutting back on the number of credit hours that state schools can require in the procurement of a BS degree in engineering. It is certainly fair to ask the questions; "What do we educators take out?" and "Should we spend less time or no time on engineering theory and classical solutions?"

The Three Theme Approach

Over the years, the authors and many other educators have developed a three theme approach in many engineering classes. The teaching themes in this approach include:

- introduction to basic principles (or theory),
- experience with classical solutions, and
- exposure to modern engineering software.

The next few paragraphs describe the three teaching themes, the logic of the approach, and the classroom activities that support the approach. The first teaching theme is an introduction to basic principles. This is not new and was described in the authors' engineering education in the 1970s and 1980s. It has always seemed important to engineering educators to build the foundation upon which design and analysis equations rest. In fact, this is merely an extension of the basic mathematics and science foundation that all engineering students are required to build upon. It is often referred to as engineering theory, although we often avoid this term since it can have a bad connotation with students.

An introduction of basic principles (or theory) usually precedes the development of equations and techniques that constitute classical solutions to engineering problems. This introduction accomplishes a number of things:

- Students are confident that the classical solutions rest upon a sound foundation of scientific and mathematical principles.
- Students begin to recognize how various principles and concepts can be tied together to solve complex engineering problems.
- Students are exposed to underlying assumptions in the equations and solution techniques.

Introducing students to theory does not need to be tedious. The authors have found a number of ways to accomplish this in an active learning environment. One method is to give students a mini-proof (shortened from what is in the textbook), require them to review it in small groups, and develop one question about the proof. A second method is to give them a mini-proof interspersed with questions or blanks to fill in.

The second teaching theme is experience with classical solutions. Again this is not new. The equations resulting from basic principles are ordered in such a way as to solve real engineering problems. Students need exposure to the equations and procedures to understand how to apply them, work within their limitations, and recognize appropriate and inappropriate applications. This often requires showing them worked examples, but preferably by engaging them in classroom exercises and homework problems.

The third teaching theme, exposing students to modern engineering software, is a more recent development. Some educators argue that it is the least important. They argue that a solid foundation in the first two teaching themes is the charge of engineering education and students will be able to learn to use software quickly when they enter the workforce. They also argue that there are so many competing software packages, many of which are proprietary and expensive, that it wastes valuable classroom time to select one package to teach.

However, many educators feel that exposure to engineering software is important in the educational process. The authors argue that some exposures is necessary to meet ABET mandates, make graduates more desirable to prospective employers, and to ease the transition into engineering practice. Fortunately, the price of engineering software has become much more affordable in recent years and educational discounts abound. In addition, some software packages have become dominant in the marketplace easing the question about which package to teach. In areas of civil engineering where this has not happened, many of the packages are beginning to look alike, which is what happened to spreadsheet software years ago.

Teaching students to use engineering software is not difficult, nor does it need to take a lot of time. As software packages have matured, many have become very user friendly. Once students understand classical solutions, they easily learn to use the software. One method that the authors use to expose students to engineering software is to have them check their classical solutions (i.e., homework) with appropriate software. Another instructional method is to have them use engineering software for class projects. This is particularly apropos in the capstone design experience.

What Does the Educational Literature Say?

Now let's go to the recent educational literature to briefly examine if there is support for all three teaching themes. First, let's examine the importance of introducing students to theory (basic principles). Clearly the definition of "engineering" rests on obtaining specialized knowledge in science and mathematics, as well as the engineering sciences.[2] The Accreditation Board for Engineering and Technology (ABET) also supports this concept. Indeed, "engineering judgment" comes through the study, application, and practice of principles that rest on the foundation of mathematics and the natural sciences.[1] Lawson enthusiastically supports the concept.[3] He states that "theory promotes understanding, and understanding enables engineers to develop the practical expression of judgment and intuition vital to the engineering profession." Most engineering educators recognize the importance of providing their students with a solid foundation in the basic principles, and the only thing we wrestle with is finding the most effective way to deliver it.

Next let's look at providing students with experience in classical solutions. In his defense of providing students with a solid background in theory, Lawson provides an argument for exposure to classical solutions[3]. He supports exposure to theory precisely because it provides the link to the real world, and that true understanding requires theory and application. In his paper on the lack of computer usage in engineering science classes, Jones argues for more effective use of computer software, but never against the effectiveness of teaching classical solutions.[4] Indeed, many papers have been written on the proper balance between computer software and classical solutions, but we did not come across any arguing to drop the latter.[5,6,7]

Finally, let's examine the literature for support of exposing engineering students to modern engineering software. The literature is replete with testimonials as to its importance. Whiteman and Nygren suggest that the use of mathematical software packages in engineering classes produces higher quality learning and enhanced critical thinking.[5] Kohler et. al. extol exposure to the tools of the profession, but warn against a "black box" approach.[7] In fact, they support the conjunctive use of theory and classical solution checks on the software results. Bining goes so far as to suggest hydraulics can be taught without water by the extensive use of computer software.[8] While we do not go quite that far, we do favor the judicious use of modern application software in engineering education.

It was enlightening to review the recent literature concerning the three teaching themes presented: theory, classical solutions, and engineering software. Certainly there are ample examples in support of teaching theory and the exposure to modern computer software. Few articles were found in support of teaching classical solutions. It wasn't that the support for this was lacking. In fact, support for the other two themes was always hinged on a better understanding of engineering applications, i.e., classical solutions. It was almost understood that this component of engineering educations is sacrosanct and did not need to be stated or proved.

An Ethical Argument

The practice of civil engineering has become heavily, indeed almost exclusively, dependent on computer software. The technical appendices of engineering reports rarely contain classical solutions, rather they contain page after page of computer generated solutions. These solutions take the form of orderly tables and impressive graphs, often 3-D models. This represents the state of engineering practice!

However, the competent and ethical engineer must ask whether the impressive computer generated solutions are correct. Puri reported the interesting results of an ASCE task committee related to computer-related errors in engineering practice.[9] Of the 52 cases of computer-related errors investigated, 13% were attributed to faulty hardware, 25% were attributed to faulty software, and 58% to "faulty" users. It might be easy at this point to say we need to teach engineers how to better use engineering software. The authors take a different view. While user mistakes are regrettable and often avoidable, any mistake can lead to an engineering failure, and some may be catastrophic. And that is what must be avoided at all cost!

In the ASCE Code of Ethics, Article 1 of the Fundamental Cannons states that "Engineers shall hold paramount the safety, health, and welfare of the public …"[10] It is our ethical obligation to avoid making mistakes in our engineering designs. The fact that we are in the computer age does not mean that design mistakes will not be made. So how do we avoid them? We would argue that a solid engineering education is the key; an education that is rich is theory, classical solutions, and exposure to engineering software. An engineer who is astute in the basic principles (theory) will be a wise user of engineering software. One who is well-trained in the use of the software is not likely to make as many mistakes. Moreover, an engineer who has a solid background in the classic solutions of engineering will be able to put pencil to paper to check computer generated solutions for accuracy and reasonableness. Engineering software certainly saves us time, so let's at least act ethically and make sure the solutions generated are correct and optimal.

What do the Students Think of the Approach?

The authors prepared two surveys to gage learner receptiveness to the three teaching themes: basic principles (theory), classical solutions, and engineering software. The surveys were given to college juniors (n = 28) studying pressure pipe networks in a hydraulics class and a group of practicing engineers (n = 26) taking ASCE continuing education classes on the use of a hydrologic/hydraulic software package. Each group was asked if they found it useful to go over the theory on the topic, perform calculations in the form of a classical solution scheme, and utilize engineering software to solve a problem. In addition, they were asked how much time should be allocated to each theme to enhance their learning.

Let's examine the college students first, found in Table 1. It was anticipated that engineering students would not find the classical solution of pressure pipe networks useful and would prefer to spend little time on the classical solution. It was also anticipated that they would not be enamored with basic principles (i.e., theory).

Table 1. Results of Survey Given to College Juniors. First entry corresponds to usefulness of theme, second entry in parenthesis amount of time to spend on theme

Theme	Very Useful --------------- (Spend a Lot of Time)	Useful ------------------ (Spend Mod. Am't of Time)	Somewhat Useful -------------- (Spend Some Time)	Not Very Useful ---------------------- (Spend Very Little Time)
Basic Principles (Theory)	**43% (7%)**	**50% (50%)**	**7% (36%)**	**0% (7%)**
Classical Solution	**32% (22%)**	**43% (46%)**	**18% (32%)**	**7% (0%)**
Engineering Software	**60% (0%)**	**36% (61%)**	**0% (21%)**	**4% (18%)**

Now let's examine the responses from the practicing engineers found in Table 2. It was anticipated that practicing engineers would have more of an appreciation for basic principles and the classical solution since they would understand the need for both in effectively using engineering software. However, that could be offset by the fact that they were specifically being trained in the use of engineering software.

Table 2. Results of Survey Given to Practicing Engineers. First entry corresponds to usefulness of theme, second entry in parenthesis amount of time to spend on theme

Theme	Very Useful --------------- (Spend a Lot of Time)	Useful ------------------ (Spend Mod. Am't of Time)	Somewhat Useful ---------------------- (Spend Some Time)	Not Very Useful ---------------------- (Spend Very Little Time)
Basic Principles (Theory)	**62% (15%)**	**38% (62%)**	**0% (23%)**	**0% (0%)**
Classical Solution	**38% (15%)**	**50% (50%)**	**4% (27%)**	**8% (8%)**
Engineering Software	**73% (42%)**	**27% (50%)**	**0% (8%)**	**0% (0%)**

It is easy to see that both students and practicing engineers have some appreciation for the three theme approach to engineering educations and appreciate the importance of basic principles and of classical solutions.

Although the survey was given to a limited group of students, several interesting observations are worth noting. Basic principles (theory) were considered useful to very useful by 93 percent of the students, although only 57 percent of them favored allocating a lot to a moderate amount of time on them. We believe that the students viewed basic principles as fundamental to understanding the classical solution and not as useless proofs quickly to be forgotten. However, the discrepancy between their stated importance and the amount of time they would allocate can not be explained.

Interestingly, basic principles were also considered useful to very useful by 100 percent of the practicing engineers and classical solution useful to very useful by 88 percent. Slightly smaller percentages, 77 percent and 65 percent respectively, favored spending a lot of time to a moderate amount of time on basic principles and classic solution. This confirmed our initial expectation that practicing engineers would recognize the importance of basic principles and the classic solution despite taking a course specifically designed for engineering software.

Classical solutions were considered useful to very useful by 75 percent of the students, the lowest ranking of the three themes. In close agreement, 68 percent favored allocating a lot to moderate amount of time to them. The classical solution solved by the students was a ten pipe network solved with the Hardy-Cross method. Although sufficient to teach the concepts supporting the Hardy-Cross method, the problem was not computationally intensive. The same problem was then assigned to be solved using the engineering software.

Students tended to favor the engineering software (96 percent very useful or useful) over classical solutions (75 percent very useful or useful). The same trend was expressed by the practicing engineers (100 percent and 88 percent respectively), although such a response may be expected as they were enrolled in a course to specifically learn the engineering software. However, despite a 96 percent useful or very useful rating, 39 percent of the students preferred to spend some or very little time on the engineering software. In addition, not one student wanted to spend a lot of time on the software. User-friendly software may explain this observation. Students were able to quickly solve the assigned problem, a ten pipe network. They recognized the computational advantage of the software over the classical solution and related solving the problem to mastery of the software. Thus there was no need to spend more time on the software.

Conclusions

Our experience, the literature, students, and practicing engineers agree that classical solutions are a necessary part of the education process. We believe a three theme approach of basic principles (theory), a classical solution, and finally engineering software is a useful approach. It introduces students to engineering software but still emphasizes the importance of basic principles and the classical solution. Engineering software will continue to be an important part of the education process. It should never replace the classical solution but ease the computational burdens of the classical solution.

The student well trained in theory and classical solutions will be a much wiser user of engineering software and a better engineer. We conclude with a quote from a survey completed by a practicing engineer: "Having been away from academia for 20 years now, I've lost theory and classical solutions ability. The refresher was valuable. With those in mind, the computer solutions fall together and speak for themselves."

Bibliography

1. Accreditation Board for Engineering and Technology (ABET). (2003). "Criteria for Accrediting Engineering Programs," (http://www.abet.org/criteria.html).
2. National Society of Professional Engineers (NSPE). (1997). *Engineering Licensure Laws: A state-by-state summary and analysis*, NSPE, Alexandria, Va.
3. Lawson, W. D. (2002). "In Defense of a Little Theory." *Journal of Professional Issues in Engineering Education and Practice*, 128(4), 206-11.
4. Jones, J. B. (1998). "The Non-Use of Computer in Undergraduate Engineering Science Courses." *Journal of Engineering Education*, January, 11-23.
5. Whiteman, Col. W. E., and Col. K. P. Nygen. (2000). Achieving the Right Balance: Properly Integrating Mathematical Software Packages into Engineering Education. *Journal of Engineering Education*, July, 331-36.
6. Jewell, T. K. (2001). "Teaching Hydraulic Design Using Equation Solvers." *Journal of Hydraulic Engineering*, 127(12), 1013-21.
7. Kohler, R. L., D. A. Sabotini, and L. D. Fink. (2002). "Laptops in the Classroom: Do They Make a Difference?" *Journal of Engineering Education*, October, 397-401.
8. Binning, C. D. (2003). "Teaching Hydraulics without Water." Proceedings of the World Water and Environmental Water Resources Congress 2003. Philadelphia, Pennsylvania, June 24-26, 2003.
9. Puri, S. P. S. (1998). "Editorial: Avoiding Engineering Failures Caused by Computer Related Errors." *Journal of Computing in Civil Engineering*, 12(4), 170-2.
10. American Society of Civil Engineers (ASCE). (2003) "Code of Ethics – Fundamental Canons," (http://www.asce.org/professional/ethics).

Biographical Information

ROBERT HOUGHTALEN is a Civil Engineer and Department Head and Professor of Civil Engineering at Rose-Hulman Institute of Technology.

MICHAEL ROBINSON is an Environmental Engineer and Assistant Professor of Civil and Environmental Engineering at Rose-Hulman Institute of Technology.