An Integrated Approach to Use Spreadsheet Programs for Enhanced Learning Benefits in Engineering Education

Naraig Manjikian
Department of Electrical and Computer Engineering
Queen’s University, Kingston, Ontario, Canada
naraig.manjikian@queensu.ca

Abstract
This paper presents an integrated approach to use spreadsheet programs for enhanced learning benefits in undergraduate engineering education. Within the context of an overall problem-solving procedure, development of spreadsheets is aided by a specification syntax to encourage methodical and logical thinking. Heavy emphasis is placed on the use of symbolic parameters. A template for spreadsheets with dynamic visualization is also used in this approach.

1 Introduction
Spreadsheet programs such as Microsoft Excel and OpenOffice Calc are used in plotting data points, performing regression, and tabulating numerical calculations. Although these applications are appropriate and important, there are opportunities for more imaginative use of spreadsheet programs in the context of undergraduate engineering education.

This paper outlines an integrated approach to use spreadsheet programs for enhanced learning benefits. The approach is comprised of a problem-solving procedure, a spreadsheet specification syntax that documents the problem-solving effort, heavy emphasis on using symbolic parameters, a template for spreadsheets with dynamic visualization, and an appropriate progression of instruction and practical laboratory activities.

The approach described in this paper stems from the author’s instruction of a first-year undergraduate engineering course, APSC 141 Personal Computers in Engineering consisting of two key technical modules: spreadsheet software (Microsoft Excel) and mathematical software (Maple). For Fall 2008 and Fall 2009, the author co-instructed APSC 141 with Steven Blostein. For credit, students were required to attend mandatory practical laboratory sessions every week, and a written final examination ensured proper assessment of individual student learning. Overall course administration and planning involved both instructors, but each was solely responsible for a module: N. Manjikian for spreadsheet software and S. Blostein for mathematical software. The remainder of this paper is devoted entirely to the approach developed and applied by the author for the spreadsheet module.

The approach taken by the author for the APSC 141 spreadsheet module is intended in part to address the variance in the aptitude and knowledge of students related to mathematics, computers and spreadsheet software, and most particularly problem-solving skills. The approach also considers other first-year courses. Effective application of spreadsheet software can reinforce concepts in the concurrent first-year physics course. Students also take a subsequent introductory course on computer programming in the C language. Given the often limited preparation of students in programming, some early exposure to language syntax and algorithmic thinking offers certain benefits.

All credit for APSC 141 is assigned to the category of engineering science, which the Canadian Engineering Accreditation Board defines as “the application of mathematics and natural science to practical problems” and “the development of mathematical or numerical techniques, modeling, simulation, and experimental procedures.” Imaginative and effective application of spreadsheet software as a technical engineering tool can address these aspects.

For contrast, the use spreadsheet software in engineering education is reflected in the comprehensive textbook of several hundred pages by Larsen on the use of Microsoft Excel in graphing, regression analysis, matrix operations, and numerical differentiation and integration. Such a thorough treatment may not necessarily be appropriate for first-year instruction as in the time-limited spreadsheet module for the APSC 141 course. Customized course material may be more suitable.

Structured problem-solving approaches are used in different disciplines. For example, a textbook by Mines...
and Lackey on environmental engineering describes the following procedure: identify and define the problem, identify given data and unknowns with relevant units, sketch the system, select the applicable theory, solve the problem, and check the solution. Other problem-solving approaches will have similar elements.

2 Adapting Previous APSC 141 Practice

In the spreadsheet module of the previous instance of the APSC 141 course prior to Fall 2008, emphasis was placed on preparing well-formatted spreadsheets with informative headers, structured data tables, and proper graphs, followed by regression analysis. This emphasis reflected expected use of spreadsheet software to produce output for inclusion in laboratory reports. Other details were also given some attention in previous instruction for the spreadsheet module. A suggestion was made for potential use of a symbolic parameter in a spreadsheet, utilizing the ability to give cells meaningful symbolic names as an alternative to the standard row/column identification. These symbolic names can then be used in formulas to refer to other cells. Previous instruction for the spreadsheet module also provided some limited background on the mathematics of regression analysis, but the primary emphasis was on the manner in which to formally report the results from regression analysis (i.e., using the 95% confidence interval).

After considering aspects from previous practice, the author of this paper preserved them, but also enhanced them in various ways. Symbolic parameter usage was required and heavily emphasized, not just suggested. For the benefit of other laboratory courses, the same spreadsheet header format was retained, as was the formal reporting for regression analysis. However, enhanced background for regression analysis was provided as an enhancement, including a brief historical outline and a dynamic spreadsheet visualization of residuals and the minimization of the sum-of-squared-residuals function for lecture presentation. In the laboratory, students created a custom spreadsheet to compute the formulas for regression analysis for comparison with the built-in regression tool of Microsoft Excel.

3 Problem-Solving Approach

As a further enhancement of the overall philosophy of the course to address the issues outlined in Section 1, the author developed an engineering approach for problem solving based on the following procedure:

- understanding and describing a technical problem
- establishing what is given or must be derived
- identifying appropriate theory for the problem
- designing specifications for a spreadsheet
- implementing a spreadsheet based on the design
- analyzing/solving the problem with the spreadsheet
- interpreting and documenting the results

When explaining this approach to students, it is noted that spreadsheet software is not involved until the fifth step so as to encourage a methodical and logical approach to technical problems, rather immediate creation of a spreadsheet in a haphazard manner. Furthermore, emphasis on specification and documentation highlights the importance of a traceable approach.

Exposition of this approach in course material and lectures progresses from simple examples to more complex ones. For self-study, the course material is complemented by take-home practice quizzes with solutions to enable students to gauge their grasp of the approach and its application. In the laboratory, students receive further instruction from a practical standpoint, and there is a progression from simpler to more complex examples.

4 Spreadsheet Specification Syntax

For methodical problem-solving with spreadsheets, a specification syntax has been developed by the author to express spreadsheet design in an algorithmic style. The syntax is derived from the C programming language, including C-style /*...*/ comments. It does not represent executable code; rather, it is a formal human-readable guide for spreadsheet implementation. It describes inputs and outputs, relevant calculations, and other spreadsheet content. The algorithmic basis provides an unambiguous sequence and logic for the implementation. It captures and documents the intellectual effort for problem solving, and it makes implementing the spreadsheet a routine task.

An example technical problem presented to students to initially illustrate the syntax and the overall problem-solving approach is the calculation of average power consumption (the unknown) for a laptop computer, based on battery capacity, nominal voltage, and measured running time (the given information), using the applicable theory related to electric power. Figure 1 provides a spreadsheet specification that consists of header and documentation, symbolic inputs and outputs with units indicated, and the calculation expressed symbolically. Figure 2 is the implementation of the specification in Figure 1. The inset portion of the figure highlights the use of symbolic names for cells.

4.1 A Structured Approach for Plotting Graphs of Theoretical Functions

In technical problem solving, plotting the graph of a theoretical function may be required to aid in analysis.
Average power consumption for a laptop computer

17 May 2010

Average_laptop_power.xlsx

Using battery capacity and running time to shutdown, this spreadsheet calculates average laptop power consumption.

Input:
- capacity: battery capacity in milliampere-hours
- voltage: nominal battery voltage in Volts
- running_time: measured running time in minutes

Output:
- average_power: computed average power in Watts

average_power = capacity * voltage / (running_time / 60) / 1000

Resistors connected in parallel

17 May 2010

parallel_resistance.xlsx

For this study, resistor R_fixed has a constant value. Another resistor R_var can be adjusted. The two resistors are in a parallel configuration whose equivalent resistance is R_eq.

Input:
- N: number of data points to plot
- R_fixed: value of fixed resistance (Ohms)
- R_var_min: minimum and maximum values of variable resistance (Ohms)
- R_var_max: variable resistance (Ohms)
- R_var_values[N]: values of independent variable (Ohms)
- R_eq_values[N]: values of dependent variable (Ohms)

Output:
- R_var_values[N]: values of independent variable (Ohms)
- R_eq_values[N]: values of dependent variable (Ohms)

Internal:
- i: loop index variable
- delta_R_var: separation of points (Ohms)

delta_R_var = (R_var_max - R_var_min) / (N - 1);

define arrays:
- R_var_values[N]: column of values of independent variable
- R_eq_values[N]: column of values of dependent variable

//-- construct table of data points to plot --->
for (i = 1; i <= N; i = i + 1) {
  R_var_values[i] = R_var_min + (i-1) * delta_R_var;
  R_eq_values[i] = (R_fixed * R_var_values[i]) / (R_fixed + R_var_values[i]);
}

//-- plot the data points --->
graph (x, y) : {R_var_values[], R_eq_values[]};
set x-axis title to "R_var (Ohms)";
set y-axis title to "R_eq (Ohms)";
set main title to "Resistors in parallel";
The theoretical expression for the equivalent resistance is 

\[ R_{eq} = \frac{(R_{\text{fixed}} \cdot R_{\text{var}})}{(R_{\text{fixed}} + R_{\text{var}})} \]

if \( R_{\text{var}} \) is treated as an independent variable, then \( R_{eq} \) can be viewed as a function that depends on \( R_{\text{var}} \). Given a number of points \( N \) to plot, and given minimum/maximum values for \( R_{\text{var}} \), the specification in Figure 3 allows for the parameterized implementation of the spreadsheet in Figure 4 with proper formatting. Each cell with a parameter value has the corresponding symbolic name from the specification. The cell for \( \Delta R_{\text{var}} \) also has a symbolic name. The table cells calculate \( R_{eq} \) values.

### 4.2 Macros to Control Parameter Values

Spreadsheet software permits the use of macros, which are small programs or recorded keystrokes that can be used to perform a sequence of operations on spreadsheet cells [2][6]. In the approach discussed in this paper, macros are adapted for the purpose of interactively controlling (i.e., incrementing or decrementing) the values of cells representing input parameters of interest. If various calculations are implemented in formulas that depend on parameter values, the result is a dynamic spreadsheet whose output changes according to user input. In this manner, the use of spreadsheet software can be made more interesting for students. Furthermore, when parameterized theoretical functions are plotted in a spreadsheet as described in Section 4.1, the spreadsheet software automatically redraws the graph for any change. The resulting dynamic spreadsheet becomes a powerful learning tool for the technical concepts reflected in the theoretical function of interest.

Students are provided straightforward instructions on how to represent simple macros to increment and decrement parameter values in a spreadsheet specification, how to record a simple sequence of keystrokes and mouse clicks that implement these operations, and then how to assign these macros to simple graphical objects (such as arrows) drawn in a spreadsheet. Clicking on a graphical object causes the assigned macro to be executed. All references to cells containing parameter values use symbolic names for those cells.

An example of an interesting and educational exercise with macros for students involves interpreting and implementing the spreadsheet specification shown in Figure 5. The specification is for a parameterized mathematical model of a two-link robot arm to be viewed graphically. With the aid of macros to interactively control the angle of each joint, students can dynamically change the view shown on the graph. The implementation of the specification is shown in Figure 6.

```plaintext
header | title = "Two-link robot arm visualization",
author = "Naraig Manjikian",
date = "17 May 2010",
filename = "two_link_robot_arm.xlsm"

documentation { "This spreadsheet provides a dynamic graphical visualization of a parameterized two-link robot arm." }

input { L_1, L_2, /-- link lengths (cm) --/
theta_1, theta_2, /-- joint angles (degrees) --/
}

output { theta_1, theta_2, /-- joint angles (degrees) --/
x[3], y[3], /-- coords for link endpoints --/
}

//-- convert angles from degrees to radians --/
ang_1 = theta_1 / 180 * pi();
ang_2 = theta_2 / 180 * pi();

//-- specification of graph to show links of robot arm --/
graph (x, y) : { x[1], y[1];
use points and lines;
set x-axis min/max to -50/50 /* permits combined length of 50 cm, */
set y-axis min/max to -50/50 /* and rotation through a full circle */
hide legend, main title, x-axis title, y-axis title; /* plain graph */

//-- specify graph to show links of robot arm --/
x[1] = L_1 *cos(ang_1); y[1] = L_1*sin(ang_1);

//-- construction of table of coords for graph --/
x[1] = theta_1 = 180; y[1] = 0;
x[2] = x[1] + L_1*cos(ang_1); y[2] = y[1] + L_1*sin(ang_1);

//-- macros for interactive control of joint angles --/
macro inc_theta_1 { theta_1 = theta_1 + 5; } /* assign to up/down */
macro inc_theta_2 { theta_2 = theta_2 + 5; } /* control of angles */
macro dec_theta_1 { theta_1 = theta_1 - 5; } /* spreadsheet */
macro dec_theta_2 { theta_2 = theta_2 - 5; } /* to update */
```

Figure 5: Specification for a two-link robot arm

Figure 6: The implementation for Figure 5

The macros at the bottom of Figure 5 are assigned to the arrows shown in Figure 6. Clicking on an arrow causes a corresponding joint angle to increase or decrease, and the graphs is automatically redrawn to reflect the change.

### 5 Spreadsheets with Automatic Dynamic Visualization

The use of macros to control parameter values as discussed in Section 4.2 allows for the creation of dynamic interactive spreadsheets. However, repeated user input (i.e., clicking on the mouse many times) is required to cause a parameter value to sweep through a range. A desirable enhancement is to allow for one or more parameters to sweep through a range automatically with a single mouse click. For this purpose, the author has developed a template spreadsheet for dynamic visualization.
Dim sweep_active '--- flag to indicate when sweep is active
Sub sweep_run() '--- macro for "play" button
Dim i
Range("sweep_value").Select '--- exit if sweep_value
i = ActiveCell.Value '--- is out of range
If (i < 0 Or i >= 100) Then
Exit Sub
End If
sweep_active = 1
Do While (0 = 0)
DoEvents '--- allow other tasks to run
Range("sweep_value").Select
ActiveCell.Value = i
i = i + 1
If (i > 100) Then '--- exit when 100 is reached
sweep_active = 0
Exit Sub
End If
Loop
End Sub
Sub sweep_reset() '--- macro for "reset" button
If (sweep_active = 1) Then
Exit Sub
End If
Range("sweep_value").Select
ActiveCell.Value = 0
End Sub

Figure 7: Macros for template spreadsheet
A cell that is given the symbolic name sweep_value has a value that ranges automatically from 0 to 100 after clicking on a “play” button. Clicking on a “reset” button causes the value to be set to 0 again. Two simple macros written in Visual Basic for Microsoft Excel are assigned to the graphical objects to implement these functions. The macros are shown in Figure 7.

The use of this template is illustrated in lectures with an example that is linked to the concepts that students are learning concurrently in their first-year physics course. A simulation of projectile motion is used. When a first-year physics instructor (J. Morelli) was contacted in Fall 2008 by the author of this paper to describe the intent, the response was positive and appreciative for the effort to reinforce the concept from the physics course in a dynamic visual manner. Figure 8 shows the essential parts of a specification for a spreadsheet based on the sweep_value cell to simulate projectile motion. For the implementation, Figure 9 provides a “time-lapse” illustration from superimposing multiple screen-capture images to show the progression of the simulation. For the initial angle and velocity, additional macros are used to control these parameter values in the manner discussed in Section 4.2.

For laboratory application of this approach, students are provided with the template spreadsheet containing the two buttons and the necessary macros. Students are then required to develop spreadsheet specifications for a given physics problem, and then implement a spreadsheet to simulate the problem using the given template. The problems include simulating the deceleration of an object on a horizontal surface with friction after sliding down a frictionless ramp or after being pushed by a compressed spring, simulating an object launched horizontally to then fall vertically and bounce, simulating a collision on a frictionless surface of two objects with different momentum, and simulating a catapult that launches an object horizontally to fall vertically to the ground. Such exercise requires students to apply problem-solving skills in translating the theory from physics into appropriate parameterized calculations in a dynamic spreadsheet. Students can then apply a similar approach in later courses to enhance their understanding of technical concepts.

6 Student Feedback
The University Survey of Student Assessment of Teaching (USAT) is the method established by Queen’s University for course and instructor evaluation. The USAT allows students to provide individual written feedback in response to the question “What did you especially like about this course?” In the Fall 2008 term, students completed the USAT evaluation near the end of the module of the course on spreadsheet software. More than 150 written responses to this question were provided, with many of those responses offering generally positive comments. The following sample of responses are provided verbatim (i.e., with all original spelling, punctuation, grammar, etc.).

I liked it when the course content was related to real engineering, such as the robot arm simulation.
Amazed at capabilities of spreadsheet software
I liked the learning on some of the more advanced tools and applications of excel.
The labs in this course are quite fun, and the course
provides a deep understanding of functions associated with 'Excel'.

So far the most enjoyable part of this course was learning how to use the more "hidden" features of Excel, such as macros.

The animation available through the use of macros. I liked that the lab were done in partners and the TAs and other students could interact when having difficulties.

This course showed me how to do things on Excel I didn’t know possible.

I liked the visualization on Excel.

The course introduced many features of Excel that I have never seen before, which makes it better to use.

The difficulty and complexity of the lab work got progressively more difficult, which was good because we already fostered an understanding in the previous weeks/labs.

These comments reflect the features of the spreadsheet software that are highlighted by the approach discussed in this paper. Students expressed their appreciation for technical applications to which they can easily relate, such as the interactive two-link robot arm visualization. The comments also indicate the positive perceptions of students regarding the manner in which the elaboration of concepts and practical techniques has explored more of the sophisticated features of spreadsheet software. Finally, one of the comments provides insight into student appreciation of the manner in which the laboratory activities are staged from week to week.

The written comments from students also provide a variety of individual suggestions for improvements, as well as some negative views. The following samples of such comments are again provided verbatim.

I think the relationship between the algorithmic specification and the actual spreadsheet needs to be explained in greater depth.

maybe teach us how to read the lab instructions

I think a little more time could be spent on how to use Excel because some people don’t have any experience with it.

Although some consideration can be given to the diversity of preparation for first-year students, an instructor in a professional, accredited program has a particular obligation to set an appropriately high standard, and students must strive to meet those standards. Nonetheless, considerable support is available for students: multiple teaching assistants in each laboratory session, additional learning material, and take-home practice quizzes with solutions for students to prepare for the final examination.

7 Conclusion

Imaginative use of spreadsheet programs in engineering education can make learning more effective and rewarding for students. To this end, an integrated approach is presented in this paper to encourage methodical and logical thinking when solving problems. The approach involves developing the design for a spreadsheet using a specification syntax, rather than immediately implementing a spreadsheet in a haphazard manner. Symbolic parameters are heavily emphasized for generality. A structured approach for tabulating data points for plotting theoretical functions uses symbolic parameters effectively. Macros and a template for dynamic visualization are also employed to enable interactivity in spreadsheets. Positive comments in student feedback indicate that there is appreciation for the proposed approach.

Acknowledgements

The author of this paper has appreciated the opportunity to work closely with Steven Blostein in the preparation and delivery of the APSC 141 course in Fall 2008 and Fall 2009. One of the previous instructors in the course from Fall 2007 and earlier, Gillian Woodruff, provided useful information and suggestions based on experience with the course in general and the spreadsheet module in particular.

References


[4] N. Manjikian and S. Blostein, APSC 141 Personal Computers in Engineering, Queen’s University, Campus Bookstore Custom Courseware, Fall 2009.
