Teaching Programming Skills with MATLAB

Marc E. Herniter, David R. Scott
Northern Arizona University, Flagstaff, Arizona
Rakesh Pangasa
Arizona Western College, Yuma, Arizona

Abstract

A major challenge in contemporary engineering education is how to include and reinforce computer programming thinking skills throughout the curriculum without trivializing the problems to be solved. With all of the application specific computer programs available to solve engineering problems, engineering schools do a disservice to students if they solve trivial problems by writing programs in a high level language rather then using application specific programs. As an example, engineers would not write a program in C to solve a four transistor circuit when they could have solved the problem in a few minutes with SPICE. With the proliferation of application specific programs, instructors can assign non-trivial problems that can be easily solved with application specific programs but are difficult to solve in a reasonable amount of time with a high level programming language. The result is that most engineering curricula teach a high level programming class in the freshmen year and most students are seldom required to use the language again. The Electrical Engineering department at NAU has attempted to solve this problem by teaching programming in MATLAB and then requiring the use of MATLAB throughout the curriculum. MATLAB has enough programming constructs to teach an introductory programming course along with built-in functions to solve non-trivial problems in most high-level courses. This facilitates not only the learning of valuable programming thinking skills but also their reinforcement in the solution of nontrivial application problems throughout the curriculum.

I. Introduction

A computer programming course is required in most engineering curriculums. One reason for this requirement is to teach problem solving skills and the process of instructing someone else (in this case a computer) to solve a problem. Typically, these courses are taught in the freshman or sophomore year and use either Fortran, C, C++ or JAVA as their programming language. Because these programming languages are difficult to use when solving problems in other engineering, mathematics, and science courses, the student often does not reinforce those skills and loses a potentially valuable educational experience. MATLAB, on the other hand, is a programming language that not only retains the basic programming constructs but also features a host of advanced application-specific functions, the ability to create graphical user interfaces, an optional command line interaction, debugging tools, and symbolic mathematics. All these features allow students the opportunity to be more productive and go beyond the knowledge gained in the introductory programming course. By employing a full-featured higher-level programming language such as MATLAB, students have the opportunity to reinforce those skills and be more productive in a wide variety of their courses. Side benefits of using MATLAB as the programming language include easy plotting, advanced visualization, and powerful matrix mathematics that result in simpler programs and in the development of advanced mathematics skills.

II. MATLAB Programming Course

To solve the problem inherent in the use of traditional programming languages, a novel programming course has been developed where MATLAB is used as the programming language. MATLAB is a high level language that not only retains the basic programming constructs, but also includes many features that allow students to be more productive in their math, science and engineering classes. The course follows a traditional outline of a first semester programming course except for using a non-traditional programming language. The goal of this course is to teach programming skills and to give the students an introduction to a tool that will be useful in their college courses and throughout their careers.

II.1. Course Philosophy

The philosophy of the course is to teach students the same programming skills and techniques they would learn in a first semester programming course, and introduce the students to a tool that can be used in later courses to reinforce those skills. The emphasis is on algorithm development using structured programming constructs rather than on writing extensive programming code and getting lost in syntax errors. An example would be generating the characteristic curve of a MOSFET. A circuits professor would usually be apprehensive about assigning a problem of this type if it were to be solved with Fortran or C. With MATLAB, the problem can be solved with a few lines of code using a FOR loop and a few of MATLAB's plotting functions (see example in section III). This solves a useful problem as well as reinforces the programming techniques covered in the introductory MATLAB programming course.

II.2. Course Syllabus

1. MATLAB Fundamentals, Environment and Matrix Mathematics
2. Control Flow
    a. IF Statement
    b. Switch-Case Statement
    c. FOR Loops
    d. While Loops
3. Functions
    a. Structure of Functions
    b. Scope of Variables
    c. Passing Parameters
    d. Global Variables
    e. Recursive Functions
4. Structured Data Types
    a. Static Arrays and Applications
    b. Dynamic Arrays
    c. Designing MATLAB Functions with Array Inputs

    d. Cell Arrays
    e. Structures
5. File Input and Output
    a. Opening and Closing Files
    b. Writing to and Reading From Data Files
    c. Binary Files
    d. Exchanging Data with Other Programs through Comma Separated Files
6. Plotting in MATLAB
    a. Setting Up Basic Two-Dimensional Plots
    b. Manipulating Plot Characteristics
    c. Various Two-Dimensional Plots (Log-Log, Semi-Log, Polar, Etc.)
    d. Handle Graphics
7. MATLAB Applications
    a. Curve Fitting
    b. Solving Equations
    c. Numerical Integration
    d. Solving Initial-Value and Boundary-Value Differential Equations

II.3. Example of Programs Covered in the MATLAB Programming Course

Two types of programming problems are assigned in the class as homework. The first type are traditional programs that would be covered in a conventional programming class. The idea is to teach a specific programming techniques. An example is the ask_q function below:

```
function answer=ask_q(valid_answers, output_string);
% function answer=ask_q(valid_answers, output_string);
%
% This function asks a question of the user and loops indefinitely
% until the user enters one of the correct responses. The question asked
% is contained in the string variable output_string.
% The valid responses are contained in string valid_answers. All
% correct responses can only be 1 character in length.
% The return value, answer, is the position of the correct response in the
% string of valid answers. For example, if we use the command:
%
% xx=ask_q('abcde', 'Enter your command:')
%
% If the user types the letter a, function ask_q will return the numerical value 1.
% If the user types the letter b, function ask_q will return the numerical value 2.
% If the user types the letter c, function ask_q will return the numerical value 3.
% If the user types the letter d, function ask_q will return the numerical value 4.
% If the user types the letter e, function ask_q will return the numerical value 5.
% If the user types anything else, it will respond “Error – Input one of the following: a,b,c,d,e”

good_input=0;
while ~good_input
        in_string=input(output_string,'s');
        for i = 1:length(valid_answers)
                if strcmp(in_string,valid_answers(i))
                        good_input=1;
                        answer=i;
        end
    end
    if ~good_input
        fprintf('Error - Input one of the following: ');
        for i = 1:(length(valid_answers)-1)
            fprintf(valid_answers(i));
            fprintf(',');
        end
        fprintf([valid_answers(length(valid_answers)),'.\n']);
    end
end
```

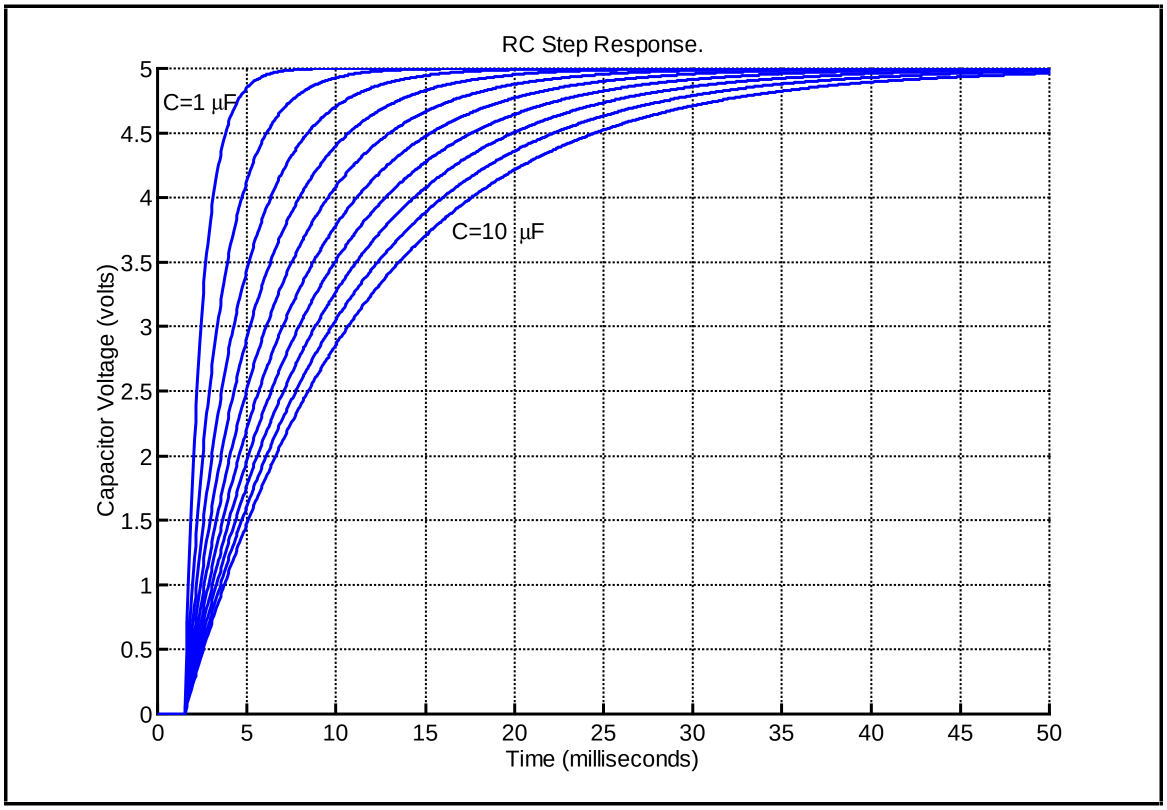
This function asks a question of the user and checks if the user response is valid. When a valid response is entered, it returns a number corresponding to the answer to the calling program. This illustrates the benefits of modularizing a task and the benefits of writing functions. If a program has to ask the user several questions, the programmer is better off writing a general function to ask a question and then call the function when needed, rather than repeatedly write the same code segment over and over. This function does not illustrate a specific engineering application, but does illustrate good programming practice. Once this function is correctly developed, students are encouraged to use this function in future assignments when an error checked response is required from the user.

The second type of problem solves a specific engineering application, emphasizes programming skills and shows how MATLAB can be used to solve engineering problems of interest to the student. A problem of this type would not usually be covered in a traditional programming course. An example is the program below that displays the step response of an RC circuit for different values of the capacitor.

```
% Example calculation of the response of an RC circuit with a switch.
% We would like to evaluate the equation.
%       Vc = Vci    for t < to
%       Vc = Vci + (Vci-Vcf*exp[-(t-to)/RC]) for tf >t >= to
% where to is the time the switch closes and tf is the ending time of the simulation.
% We would like to evaluate the equation for 1000 points of data.
%
% This script file calculates the response for ten values of C from 1 to 10 microfarads

Vcf = 5; R = 1000;  to=0.0015; tf = 0.05; Vci = 0;
t=linspace(0,tf,1000);
Vc = zeros(1,1000);
hold on

for j = 1:10
    C=j*1e-6;
    for i = 1:1000
        if t(i) < to
            Vc(i) = Vci;
        else
            Vc(i) = Vcf + (Vci -Vcf*exp(-(t(i)-to)/(R*C)));
        end
    end
    plot(t*1000,Vc);
```

```
end

title('RC Step Response.');
ylabel('Capacitor Voltage (volts)');
xlabel('Time (milliseconds)')
text(16.5,3.75,'C=10  \muF');
text(.3,4.75,'C=1 \muF');
grid
hold off
```

This program is used to teach FOR Loops and IF statements, and also relates the programming techniques to circuit applications. Students may be taking the course that solves the RC circuit at the same time they are taking this programming course. Students can immediately see how programming with MATLAB can be used to help solve problems in their other courses. The plot of the above program is shown below:



III. Other Courses Reinforcing the use of MATLAB

To maximize the benefits of using MATLAB as a programming language, MATLAB should be used in many higher level courses to reinforce and further develop programming techniques. MATLAB is already heavily used in control and signal processing courses, so the challenge is to incorporate MATLAB into other courses where programming is not traditionally used. In these courses, the use of MATLAB should not be a force-fit where the application is just a programming exercise. Instead it should be one that an engineering student would need to solve, and MATLAB would be a valid way to solve it.

III.1. List of Courses and Applications

Below is a list of non-traditional courses in which we have integrated MATLAB.

1. Electrical Engineering II - This is an electric circuits course covering transient response, Laplace transforms, frequency response, and op-amp analog filtering. MATLAB has been used throughout the course in the following applications:
    a. Solving ordinary differential equations arising from RL, RC and RLC circuits.
    b. Measuring time constants from data collected in the laboratory and graphed using MATLAB.
    c. Measuring frequency content of signals filtered with a resonant RLC circuit by employing the Fast Fourier Transform.
    d. Root finding and pole zero mapping of transfer functions in the s domain.
    e. Laplace and inverse Laplace transforms using symbolic mathematics
    f. Partial fraction expansion of s domain functions.
    g. Frequency response Bode plotting.
    h. Filter design including choice of filter order for Butterworth, Chebychev and elliptic analog filters and choice of circuit component values to implement first and second order op amp filter stages.

The variety and number of nontrivial applications of MATLAB in this sophomore level course clearly illustrates that a student is not only able to reinforce programming skills but also achieve a high level of productivity and visualization. This is achieved by MATLAB's combination of basic programming constructs and the advanced features and functions available with MATLAB.

2. Analog Circuits I – This is a first course in electronics covering diode and zener circuit, power supplies, non-linear Op-AMP circuit, and NMOS and CMOS digital circuits.
    a. In the saturation region, the equation for a MOSFET is $I_D = K(V_{GS} - V_T)^2$. The student measures I-V data points for a MOSFET in the lab and then uses a curve fit to find numerical values of K and V_T. The original data and the curve fit are plotted to judge the validity of the fit. This problem can be easily solved with the MATLAB polyfit function and a few lines of code.
    b. To observe the switching speed of an NMOS logic gate, a complex differential equation must be solved. This differential equation is non-linear and changes with time and voltage. This problem can be solved with MATLAB by writing a function that returns the value of the derivative, and writing a short main program to call the differential equation solver and plot the results.
    c. The transfer curve (V_{OUT} versus V_{IN}) of an NMOS inverter may have up to three regions where the equation for V_{OUT} as a function of V_{IN} changes. MATLAB can be used to solve the equations in the three different regions and plot the transfer curve.
3. Analog Circuits 2 - This is a course in analog electronics covering all phases of amplifier design, including biasing, gain calculations, and frequency response.
    a. The frequency response can be plotted for both measured and theoretical data. Here we compare frequency response data generated by PSpice to data measured

in the lab. This requires that we save the PSpice data in a file and have MATLAB read the data file. The measured data can be read from a data file or entered as part of the MATLAB program.

III.2. Example MATLAB Programs Used in Higher Level Courses

As indicated earlier, MATLAB has traditionally been used in higher level courses such as signal processing and control theory. The goal in these courses is to continue to use MATLAB, but modify the exercises in a way that continues to help the understanding of the course material and also requires some degree of structured programming to maintain and develop a student's programming skills.

Many other courses such as analog and digital electronics, electromagnetics, and semiconductor physics would not traditionally use MATLAB. Here the challenge is to include MATLAB exercises that are relevant to the course and also emphasize structured programs. The exercises should be a problem that an engineer might really solve with MATLAB.

An example of an exercise that could be used in a first level circuits course is the analysis of a voltage divider using resistors with tolerance. In this example, we ask the question, if each resistor in the voltage divider has a 5% Gaussian distribution, what does the distribution of the voltage divider look like? The code section below answers this question:

```
gain=zeros(1,16384);
for i = 1:16384
        R1=1000*(1 + randn(1)/80);
        R2=1000*(1 + randn(1)/80);
        gain(i)=R2/(R1+R2);
end
[n,x]=hist(gain,50);
bar(x,n);
grid;
xlabel('Voltage Gain');
ylabel('Number of Occurrences');
```

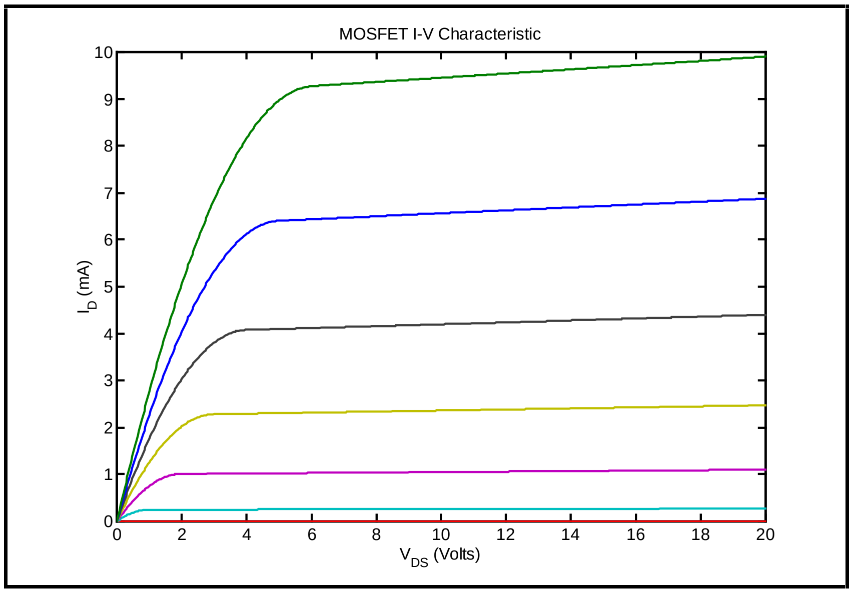
This program can also be solved more succinctly using the array properties of MATLAB:

```
R1=1000*(1 + randn(1,16384)/80);
R2=1000*(1 + randn(1,16384)/80);
gain=R2./(R1+R2);
[N,X]=hist(gain,25);
Npct=100*(N/sum(N));
bar(X,Npct);
grid;
xlabel('Voltage Divider Gain');
ylabel('Percent of Trials')
```

The first solution requires the student to use some structured programming by using a FOR loop. The second solution is a sequence of simple MATLAB statements, but it does require the understanding of how MATLAB uses arrays.

A second example is a program that plots the current-voltage characteristic curve of a MOSET:

```
K = 0.00025; Vt=2; Lambda = 0.005; Vgs = 5;
Vds = linspace(0,20,500)';
Id = zeros(500,7);
for Vgs = 2:8
    for i = 1:500
        if Vgs < Vt
            Id(i,Vgs+1) = 0;
        elseif Vds(i) >= Vgs-Vt
            Id(i,Vgs+1) = K*(Vgs-Vt)*(Vgs-Vt)*(1+Lambda*Vds(i));
        else
            Id(i,Vgs+1) = K*(2*(Vgs-Vt)*Vds(i) - Vds(i)*Vds(i))* (1+Lambda*Vds(i));
        end
    end
end
plot(Vds,1000*Id);
xlabel('V_{DS} (Volts)');
ylabel('I_D (mA)');
title('MOSFET I-V Characteristic');
```



This is useful example in analog electronics and also requires a fair amount of programming logic.

IV. Industrial Relevance and Motivation

The widespread applicability and the use of MATLAB to analyze very large data sets and perform many engineering computations has been extensively documented on the World Wide Web. A

MATLAB training workshop was recently developed for a group of engineers from the U.S. Army Yuma Proving Ground, a general-purpose facility with experience in testing weapon systems.

In discussions of the content of the workshop, it was observed that the data-analysis and computation tools being used included FORTRAN, EXCEL, and, to a limited extent, MATLAB. Recognizing the analytic capabilities of MATLAB in combination with standard programming practices, the Yuma Proving Grounds initiated efforts to train engineers in MATLAB. The following topics were emphasized:

- Review of basic matrix algebra
- Overview of MATLAB workspace and scripting
- Matrix operations in MATLAB
- Numeric calculations in MATLAB – functions, curve fitting etc.
- Statistics in MATLAB – mean, standard deviation, probability distributions, empirical modeling, etc.
- Visualization – 2D/3D plots, line plots, surface plots, etc.
- Programming techniques using MATLAB.

V. Conclusion

Although the example MATLAB programs are fairy simple, they do reinforce the programming techniques covered in the introductory programming course. The examples show students that programming using MATLAB gives them the following:

1. A powerful problem solving tool that can be applied to many different types of problems.
2. Development and maintenance of their programming and thinking skills.
3. Valuable experience with an effective industry standard tool that is useful throughout their schooling and career.

MARC E. HERNITER
Marc E. Herniter is currently an Associate Professor at Northern Arizona University. He received a B.S. in Electrical Engineering and B.A. in Physics from Boston University in 1983, and a Ph.D. from the EECS Department at the University of Michigan in 1989. He is the author of several textbooks on PSpice, MATLAB, and introductory usage of computers.

DAVID R. SCOTT
David Scott is currently Interim Chair of the Electrical Engineering Department at Northern Arizona University. He received his Ph.D. from New Mexico State University in 1990. His interests include digital signal processing, image processing, engineering design and distance education.

RAKESH PANGASA
Rakesh Pangasa is currently a Consultant and Professor. He teaches Mathematics and Engineering at the Arizona Western College, and Operations management and Business Information Systems at the Webster University and Northern Arizona University Yuma campus. He received a B.Sc. in Chemical Engineering from Punjab University in 1971, and a Ph.D. in Industrial Engineering from the Indian Institute of Technology in 1985.