

THE USE OF MATLAB TO INTRODUCE HIGH SCHOOL STUDENTS  
TO COMPUTER PROGRAMMING AND  
PROBLEM SOLVING

by

Kiersten Purves

An Essay submitted to the Faculty of the Graduate School,  
Marquette University,  
in Partial Fulfillment of the Requirements for  
the Degree of Master of Science

Milwaukee, Wisconsin

May 2012

## Introduction

In this time of a down economy, falling national ranking in test scores for math and science, and an ever increasing dropout rate of American students from degrees in the fields of mathematics, engineering, and computer science, there is an overwhelming movement to restructure the way in which STEM (Science, Technology, Engineering, and Mathematics) courses are taught. The purpose of this paper is to outline ways in which MATLAB (*matrix laboratory*, a programming language primarily used for numerical computing) can be integrated into the high school mathematics classroom through a selection of suggested activities. (Note: The activities outlined in this paper are mere suggestions and are not fully developed lessons. Please refer to the Activities Appendix for in depth examples of how to use the suggested activities to support mathematical thinking, STEM learning and the Core Standards for Mathematical Practice). The intention is to introduce a wider scope of students to some basic components of computer programming, identify ways in which mathematics and computer programming are interdisciplinary and can be applied to real world problems, and strengthen students' problem solving skills through the study of the underlying algorithm of a program. In preparation for my proposed restructuring at a personal classroom level, this paper looks at some current movements in STEM and computer science education and the reasons behind them. Additionally, this paper will discuss the state of STEM education and computer science at the high school in which I teach, and reasons for integrating programming into the mathematics classroom, specifically the choice of MATLAB.

In September of 2010, as a component of his campaign "Educate to Innovate", President Obama unveiled *Change the Equation*, a non-profit organization of business leaders and CEO'S motivated to improve STEM education through efforts such as increasing student opportunities

in engineering and robotics competitions, and professional development for teachers in math and the sciences. During his announcement of *Change the Equation*, President Obama made the following statement (Sabochik, 2010):

*“We’re here for a simple reason: Everybody in this room understands that our nation’s success depends on strengthening America’s role as the world’s engine of discovery and innovation. And all the CEOs who are here today understand that their company’s future depends on their ability to harness the creativity and dynamism and insight of a new generation. And that leadership tomorrow depends on how we educate our students today -- especially in science, technology, engineering and math.”*

This statement clearly outlines a current concern and movement in today’s education system.

This sentiment was echoed in the December, 2011 release of *Building a Science, Technology, Engineering, and Math Education Agenda* by the National Governors Association (Thomasian, 2011). This agenda, similar to Obama’s *Change the Equation*, focuses on strengthening STEM education, and encouraged actions for improvement such as increasing the rigor in STEM courses, and expanding math and science learning beyond the classroom.

The high school in which I teach is currently experiencing a dichotomy in its efforts to strengthen STEM education. On one hand, the school promotes STEM education through its involvement with Project Lead the Way (PLTW), offering four (4) rigorous courses in varying engineering fields, and striving to connect the students and teachers in these courses with community members who work in STEM fields. This attempt to create connections between students, teachers, community members and local businesses models *Change the Equation*’s effort to have American companies take a stake in the education of their future leaders. On the other hand, the school has limited offerings in the computer sciences and has cut all instructor led computer programming courses for the next year. The only opportunity a student has to be

exposed to computer programming is through an online course offered through the *eAchieve Academy – Wisconsin Online Opportunities*. As an educator of mathematics, engineering and life sciences, I feel it is important to offer cross-curricular opportunities in which students have the chance to experience what computer programming is and how it can be applied to problem solving in mathematics, science, and engineering.

The choice of implementing programming in the high school mathematics classroom may serve to kill two birds with one stone. First, it can offer an explanation to the age old mathematics question of “who uses this stuff?” and garner interest through hands on learning by allowing students a glimpse into the ways in which mathematics can be applied to different disciplines through the use of computers. Secondly, it allows for a wide portion of the student body to be introduced to some of the basics of computer science and computer programming. In the 2009 Program for International Student Assessment, fifteen year olds from the United States ranked 25<sup>th</sup> out of 34 countries in their mathematics scores (Hechinger, 2010). Similar to mathematics, the United States has fallen behind in producing students that are prepared to enter and persist in computer science programs at the college level. In *The New Educational Imperative*, (Stephenson, 2006, p.20) the CSTA (Computer Science Teachers Association) noted the unsuccessful efforts of the U.S education system to implement a standardized study of computer science:

*“While other countries have designed and implemented national computer science education programs in order to better prepare their students for the increasingly competitive global economy, attempts to bring coherent computer science education into U.S. high schools failed to address the need to instill a fluent understanding of algorithmic thinking and problem solving, and to provide exposure to software development using programming skills in U.S. students.”*

This same report went on to outline guiding principles that are essential to high school computer science. Of these principles, the following could be addressed by the process of introducing basic programming in the high school mathematics classroom. These principles are as follows: general capabilities and skills, problem solving and algorithmic thinking, integrative and interdisciplinary knowledge, comprehensive programming, and gender differences and minority issues. More specifically, the CSTA has defined general capabilities and skills to be the development of practical skills independent of specific technology that foster lifelong learning and adaptation, including written and oral communication. This principle could be addressed by the inclusion of write-ups and/or presentations accompanying a student's program, as well as the student-to-student discussions had in the development of an algorithm for a program. The algorithmic nature of mathematics lends itself to an algorithmic approach to programming, which is also the focus of comprehensive programming where the emphasis is not only on the coding, but on the underlying algorithm and its efficiency. The integration of programming as a means to apply mathematics to different disciplines gives student an interdisciplinary view of mathematics as well as programming, and since three years of math are required for graduation, classes tend to be mixed in terms of gender and ethnicity as opposed to predominantly white males.

In order to offer opportunities for programming, a programming language must be chosen. In the case of this paper the language of choice is MATLAB. One reason for this is that there are multiple easily accessible ways to get help with MATLAB. MathWorks ([mathworks.com](http://mathworks.com)) is an excellent online source for help and tutorials, and the *doc* command in MATLAB itself can be very helpful. Secondly MATLAB can easily interface with other main programming languages such as Java and C++. It is widely used in academia, research and

industry, including the colleges and universities most often attended by students from this high school. Lastly, MATLAB is frequently used in engineering courses and fields, so for a school that strives to prepare students for college degrees in various engineering fields through PLTW, it seems to be the programming language of choice.

### **MATLAB Activities for the High School Classroom**

The first section of activities is meant to accompany a junior level high school course in advanced algebra and is comprised of two main divisions. The first division includes topics on systems of linear equations, matrices, coded messages and transformations/translations and ways in which these mathematical ideas can be explored by introducing the plot function and basic matrix functions in MATLAB. The second division covers the natural number  $e$  and the golden ratio,  $\varphi$ . These sections include a bit of historical background as it is likely that students are unfamiliar with both  $e$  and  $\varphi$ , and then use these topics to introduce the programming concept of loops. The second section of activities is geared toward a course in AP Calculus offering a programming challenge to students in approximations of derivatives and integrals. The final section is a brief discussion of activities from Cleve Moler's online text *Experiments with MATLAB* which integrate mathematics and programming with the life sciences, and engineering.

#### **Systems of Equations:**

The plot function could be included in any unit at any time and may be an excellent way to introduce students to MATLAB since plotting data points and graphing functions are concrete topics that most high school juniors are familiar with and MATLAB outputs easy to understand visual data. The early portion of a junior year algebra course covers review topics from freshman algebra such as linear functions and solving systems of linear equations. In addition to

having students solve systems of linear equations algebraically and graphically by hand, the plot function in MATLAB could be introduced and used to reinforce graphical solutions. As students become familiar with the plot function and how to plot multiple functions on the same graph, they could be introduced to various commands that can be used to enhance their plots such as how to turn the grid on, add a title, change colors of individual lines, and adjust the axes of the plot to obtain a better approximation of an intersection point.

The examples below use the following equations:

$$2x + y_1 = 9$$

$$3x - y_2 = 16$$

$$2x + y_3 = 3$$

$$4x + 2y_4 = 18$$

Through choice pairing we can demonstrate an independent system ( $y_1$  v.  $y_2$ ), an inconsistent system ( $y_1$  v.  $y_3$ ), and a dependent system ( $y_1$  v.  $y_4$ ) with these equations. Notice that all figures below demonstrate the use of the title command and selective coloring of individual functions, while figure (1) also makes use of a grid and a scaling of the axes in order to pinpoint the solution  $(5, -1)$  (please see MATLAB Code Appendix (1) for matching code).

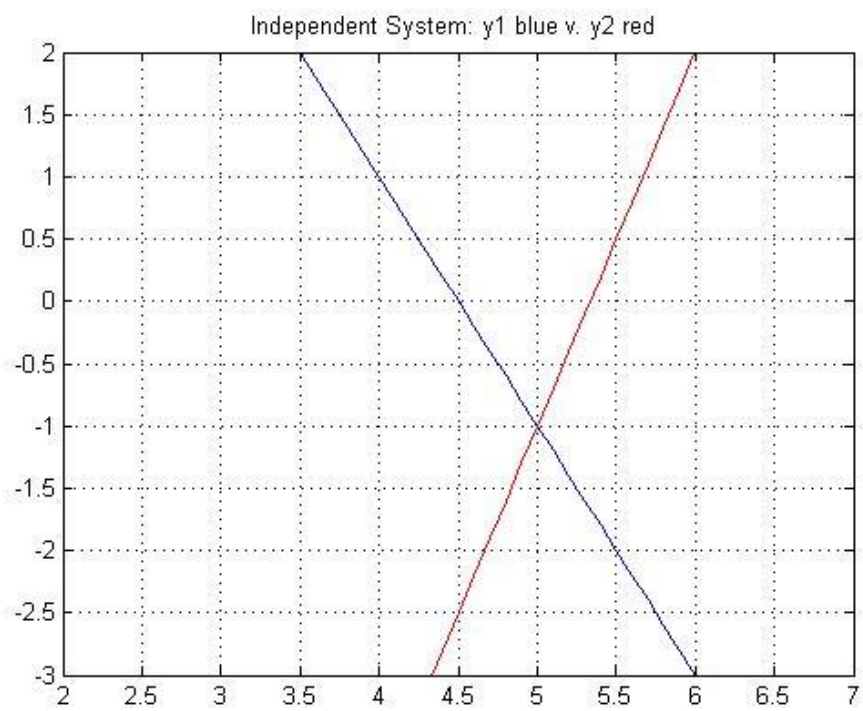


Figure (1)

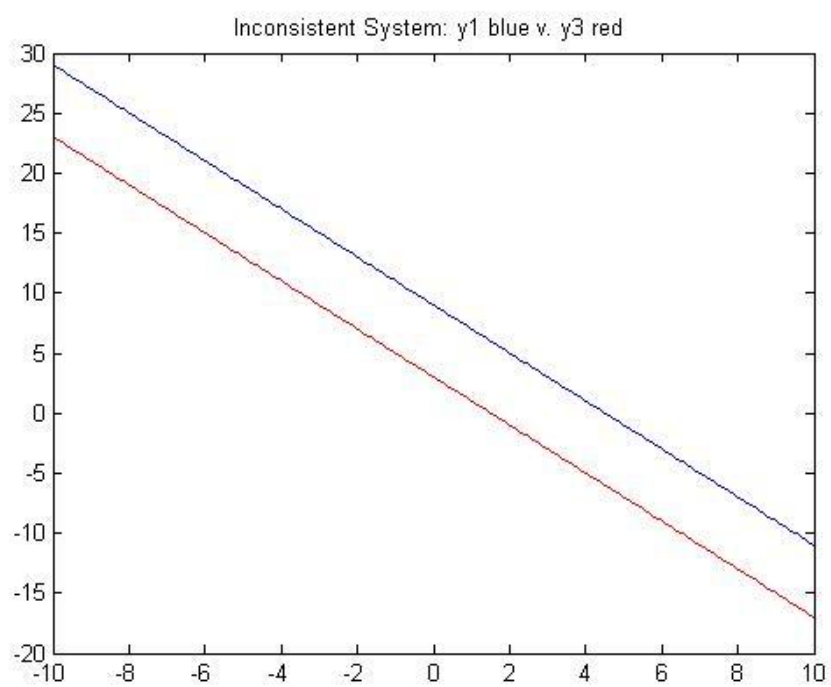


Figure (2)



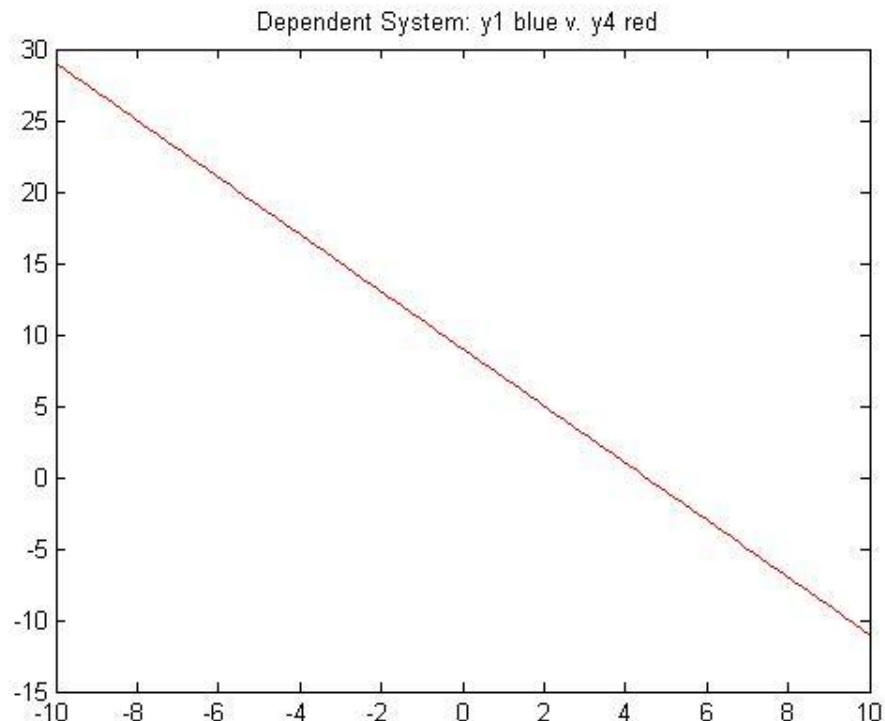


Figure (3)

**Matrices:**

Another way to solve systems of linear equations is through the use of matrices. Typically, most juniors in high school have had limited prior experience with matrices. At most they may have learned that a matrix is a rectangular array of elements, where those elements have been numerical values and some may have had some experience with adding and subtracting matrices, but their familiarity usually ends there. Since MATLAB was originally developed as a matrix calculator it can be a good format for students to explore basic properties of matrices such as addition, subtraction, scalar multiplication and matrix multiplication. Explorations might involve questioning what must be true of matrices that can be added or

subtracted, how to add or subtract matrices, when can two matrices be multiplied together, and what matrix operations are commutative.

For example, given the matrices listed below, students could be asked to perform various calculations in MATLAB to discover answers to the above explorations.

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, B = \begin{bmatrix} 6 & 7 \\ 8 & 9 \\ 1 & 2 \end{bmatrix}, C = \begin{bmatrix} 0 & 3 \\ 4 & 2 \end{bmatrix}, D = \begin{bmatrix} 3 & 1 & 1 \\ -1 & 2 & -3 \end{bmatrix}$$

To discover that matrices need to be of the same dimension in order to perform addition and subtraction, and that the process is performed element wise, students could be asked to use MATLAB to add and subtract different combinations of the above matrices.

```
% Enter Given Matrices
A=[1 2;3 4];
B=[6 7;8 9;1 2];
C=[0 3;4 2];
D=[3 1 1;-1 2 3];

% Explore Addition/Subtraction
ac=A+C
ad=A+D
```

When  $ac = A + C$  is calculated, students should be able to see that matrix addition is element wise. Likewise, when  $ad = A + D$  is attempted, students will be given the following warning from MATLAB. “*??? Error using ==> plus Matrix dimensions must agree.*” This should lead them to discover that in order to be able to perform matrix addition (as well as subtraction) the dimensions of the matrices must match. Similar explorations could be done to answer the other questions posed above (please see MATLAB Code Appendix (2) for additional code regarding other explorations). It is important to note that there is a distinction in MATLAB coding between  $A * C$  and  $A.*C$ . This may be another area to have students explore. The first coding is for traditional matrix multiplication

$$A * C = \begin{bmatrix} 8 & 7 \\ 16 & 17 \end{bmatrix}.$$

While the second coding performs element wise multiplication

$$A.* C = \begin{bmatrix} 0 & 6 \\ 12 & 8 \end{bmatrix}.$$

Once students have mastered a basic understanding of matrices and matrix operations, matrices can then be used to solve systems of linear equations. Previously graphing calculators such as the TI-83 or 84 have been used to aid in solving matrix equations. In addition to, or perhaps in place of the graphing calculator, MATLAB could now be used. Taking the same four linear equations used in the *system of equations* section above the following code and discussion outlines ways in which MATLAB can be used.

The first system was the independent system

$$2u + v = 9$$

$$3u - v = 16.$$

Written as the matrix equation  $Ax = b$ , we find  $\begin{bmatrix} 2 & 1 \\ 3 & -1 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 9 \\ 16 \end{bmatrix}$ . Entering this into

MATLAB we can solve this equation in a number of ways, two are illustrated below. The first uses the idea of finding the inverse of a matrix  $A$ , often written as  $A^{-1}$ , and defined in MATLAB as `inv(A)`. The inverse of a square matrix  $A$  is a matrix such that  $AA^{-1} = A^{-1}A = I$  where  $I$  is the identity matrix, a matrix comprised of zeros with the exception of ones (1) down the main diagonal. Students can investigate this property,  $AA^{-1} = A^{-1}A = I$ , in MATLAB and see the outcome of  $A^{-1}Ax = A^{-1}b$ . In the activity below, *Coded Messages*, students will again be asked to use this idea of solving a matrix equation using inverses, this time finding the inverse of a

two-by-two matrix by hand. This model, although computationally less efficient, matches concepts discussed in Algebra and Trigonometry regarding inverses and identities, and could lead to a more correct understanding of a way in which matrix equations can be solved. The second method, which can be computationally more efficient, may lead to the misconception that matrix equations behave in the same manner as a linear equation and can be solved by dividing both sides of the equation by matrix  $A$  in order to find  $x$ .

```
A=[2 1;3 -1]
b=[9;16]

% Find the Inverse of A
A1 = inv(A)

% Check A * inv(A) = inv(A)*A = I
A*inv (A)
inv(A)*A

% Solve Ax=b using inv(A)
x=inv(A)*b

% Solve Ax=b using matrix left division (A\b)
xx=A\b
```

Both equations lead to the solution  $(5, -1)$  as seen earlier when this system was solved graphically. The problem arises when attempting to solve inconsistent or dependent systems in these ways. Both situations lead to the MATLAB warning “*Warning: Matrix is singular to working precision.*” In much the same way the TI-83/84 outputs “*ERR: Singular Mat*” in both cases. Either way, students will be left to determine if the system has no solution, or infinitely many. Perhaps this is when they can rely on their earlier experiences with the plot function to determine which case is which.

### Coded Messages: Steganography:

To close out a study of matrices, an interesting activity that can be done with matrices is to code messages into the pixels of a picture, a process known as steganography. Texas Instruments has created a number of mathematical activities that accompany episodes of the television show NUMB3RS. The activity *Coded Messages* (Flynn, 2009) accompanies the episode “The Mole.” In this activity students are introduced to steganography by learning how to use a matrix to encrypt a message, and its inverse to decode a message. As a final project, they are asked to create a picture of their own and encrypt it. Then they are asked to send the encrypted matrix to another student to decode. An example of how this works can be seen by completing the practice exercise number 5 from the activity. Here we are given the following intercepted message B and the encryption key K:

$$B = \begin{pmatrix} 29 & 28 & 32 & 37 & 25 & 29 & 20 \\ 44 & 58 & 47 & 62 & 45 & 44 & 25 \\ 34 & 38 & 42 & 47 & 35 & 39 & 25 \\ 44 & 58 & 47 & 62 & 45 & 44 & 25 \\ 29 & 28 & 32 & 37 & 25 & 29 & 20 \end{pmatrix}$$

$$K = \begin{pmatrix} 0 & 1 & 2 & 0 & 0 & 1 & 1 \\ 1 & 2 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 2 & 1 & 0 & 1 & 2 \\ 2 & 4 & 1 & 3 & 2 & 1 & 0 \\ 0 & 0 & 2 & 1 & 1 & 2 & 0 \end{pmatrix}$$

Decoding the message with the following code ( $\text{Message} = B * \text{inv}(K)$ ) and using the given color values for each number (4 Dark Blue, 9 Orange-Red) we find the following:

```

4.0000  4.0000  4.0000  9.0000  4.0000  4.0000  4.0000
4.0000  9.0000  4.0000  9.0000  4.0000  9.0000  9.0000
4.0000  9.0000  4.0000  9.0000  4.0000  4.0000  9.0000
4.0000  9.0000  4.0000  9.0000  4.0000  9.0000  9.0000
4.0000  4.0000  4.0000  9.0000  4.0000  4.0000  4.0000

```

Lastly, coloring in a grid to represent each pixel, we would obtain a picture similar to the one seen below that shows the letters OE.

4.0000	4.0000	4.0000	9.0000	4.0000	4.0000	4.0000
4.0000	9.0000	4.0000	9.0000	4.0000	9.0000	9.0000
4.0000	9.0000	4.0000	9.0000	4.0000	4.0000	4.0000
4.0000	9.0000	4.0000	9.0000	4.0000	9.0000	9.0000
4.0000	4.0000	4.0000	9.0000	4.0000	4.0000	4.0000

### Transformations/Translations:

While the plot function may have been introduced in an earlier unit studying systems of equations as outlined above, the study of transformations could take this idea to another level introducing students to other plot functions such as subplot, and ways in which to add in the  $x$  and  $y$  axes. For the purpose of this paper transformations are explored on basic exponential functions base  $e$  or otherwise. Figure (4) demonstrates a reflection across the  $y$ -axis by comparing the basic exponential equation  $y = e^x$  and  $y = e^{-x}$ . Figure (5) demonstrates the vertical translations in the equations  $y = e^x + 1$  and  $y = e^x - 2$  alongside the parent function  $y = e^x$ . Lastly, figure (6) shows the use of the subplot function to compare multiple plots, and demonstrates a way to include the  $x$  and  $y$  axes, and the grid command seen earlier (please see the MATLAB Code Appendix).

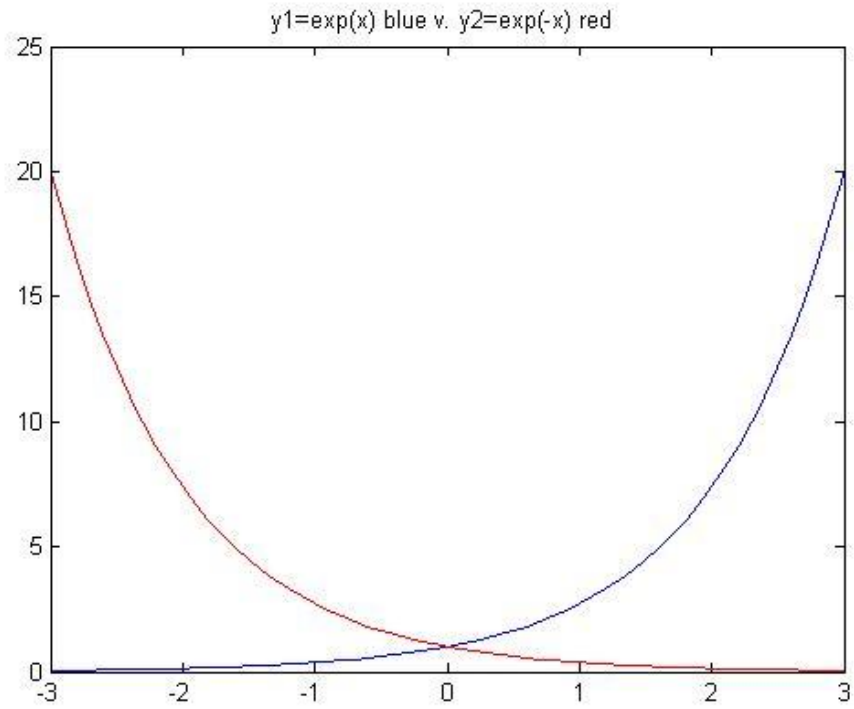


Figure (4)

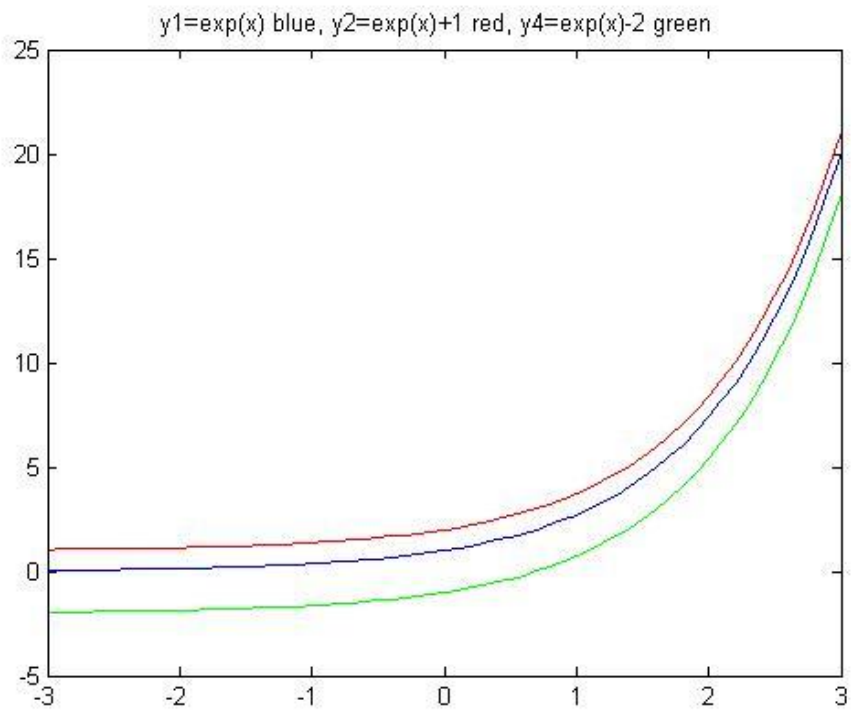


Figure (5)

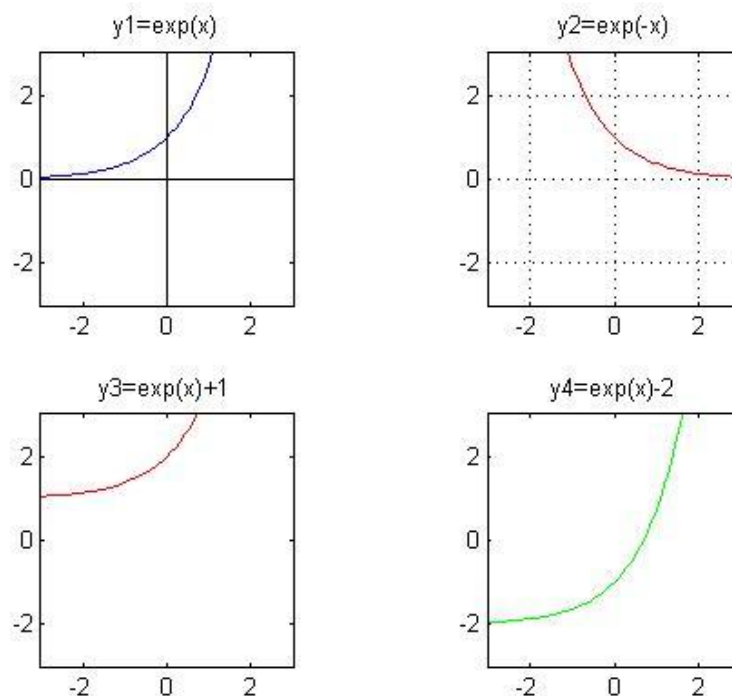


Figure (6)

### The Number $e$ :

Throughout much of the 1600's mathematicians such as the likes of Napier, Briggs and Huygens danced around the idea of  $e$  in their studies of logarithms and rectangular hyperbola. Jacob Bernoulli is credited with finding, in 1683, the first approximation of  $e$  through his study of compound interest. He approximated  $e$  to lie somewhere between 2 and 3 when he searched to find the limit of  $(1 + 1/n)^n$  as  $n$  approached infinity. At this time, the notation  $e$  was not in use. Like much of the current notation used in mathematics, the "e" notation for the number  $e$  is attributed to Euler. It first appeared in a letter from Euler to Goldbach in 1731. A few years later Euler published *Introductio in Analysin infinitorum*, a publication in which he devoted considerable work to the number  $e$ , including revisiting Bernoulli's earlier limit problem with compound interest, to obtain the following approximation of  $e$ .



$$\lim_{n \rightarrow \infty} \left(1 + \frac{1}{n}\right)^n \approx e \approx 2.71828$$

It is this limit that we will first use to approximate  $e$ .

Chapter 1 of Cleve Moler's online text, *Experiments with MATLAB*, (Moler, 2011) looks to introduce the concepts of an assignment statement, **for** and **while** loops, and the plot function using the golden ratio. In a similar manner, these concepts can be introduced along with the number  $e$ . (In a later chapter, Moler explores the exponential function  $e^t$  through differential equations and the power series for the exponential equation in an approach suited to higher level mathematic students such as those in an AP Calculus course.) It is important to note that the equal sign is an assignment operator in most programming languages. This means that it instructs the program to compute the right hand side and store the value as the variable on the left hand side replacing whatever the previous value of that variable was. In order to approximate the value of  $e$ , students must be able to write a loop. A loop is a sequence of code that is repeated for a given number of times, as in the case of a **for** loop, or until a certain condition is met, as is the case with a **while** loop.

Using a **for** loop, like the one shown below, students can experiment with the accuracy of their approximation by using different values of  $n$  such as  $n = 10, 100, 1,000, 10,000$ .

```
for n=1:10000
e=(1+1/n)^n;
end
e
```

For  $n = 10,000$  we find that  $e = 2.7181$ , which is accurate to three decimal places.

A **while** loop can add a different perspective to the question. Instead of having students experiment with different values of  $n$ , and their effect on the approximation of  $e$ , they can be asked to find the value of  $n$  that yields an approximation that is correct to four decimal places (2.7182). See the example code below:

```
n=1;
e=0;
while e<2.7182
e=(1+1/n)^n;
n=n+1;
end
n
```

In this case we find that  $n = 16,610$  will approximate  $e$  to four decimal places.

Another way in which the value of the natural number  $e$  can be explored would be through the use of the Maclaurin Series

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots = \sum_0^{\infty} \frac{x^n}{n!}$$

This can be calculated as both the infinite series that it is, or as a definite sum. It can be quite interesting to compare the speed at which the Maclaurin Series converges in comparison to Euler's limit above. In the case of the Maclaurin Series, an approximation for  $e$  accurate to four decimal places happens for values of  $n < 20$ , for Euler's limit  $n > 16,000$ . If included, students would need to be introduced to the code for summations and how to force MATLAB to use a symbolic approach in its calculations. Sample code for both the summation of the infinite series and a definite sum can be seen below.

```
% Definite Sum
a=0
b=10
syms n;
```

```

symsum(1^n/sym('n!'), n, a, b)

% Infinite Sum
a=0
b=Inf
syms n x;
symsum(x^n/sym('n!'), n, a, b)

```

## The Golden Ratio, a Programming Challenge

Seemingly found everywhere you look, romanticized in the books you read, and disguised by numerous names, the Golden Ratio ( $\varphi \approx 1.618034$ ) is a fascinating number with a long history in many disciplines. The first undisputed definition of the Golden Ratio is attributed to Euclid. At its onset, the Golden Ratio was known as Euclid's "extreme and mean ratio." While mentioned in Book II of the *Elements*, the definition of the proportion commonly thought of as the "extreme and mean ratio" with respect to a segment occurs in Book VI. Described by Euclid (Livio, 2002, p.3) "A straight line is said to have been cut in extreme mean ratio when, as the whole line is to the greater segment, so is the greater to the lesser." Taking a closer look at this statement and, with the use of the figure below, one can recreate Euclid's "extreme and mean ratio".



By Euclid's definition, segment AB cut by point C is in extreme mean ratio when the following proportion holds:

$$\frac{AB}{AC} = \frac{AC}{CB}$$

Euclid uses this ratio in other books of the *Elements* to aid in the construction of a pentagon, icosahedron, and dodecahedron.

With some algebra, it is possible to calculate the value of the Golden Ratio from Euclid's "extreme and mean ratio." First, let the measure of segment CB equal 1 unit in length and segment AC be equal to Phi ( $\varphi$ ). Then, using Euclid's extreme mean ratio, substitute in the length values for each segment to obtain the proportion below. A few basic algebra steps then lead to the shown quadratic equation.

$$\frac{AB}{AC} = \frac{AC}{CB} \rightarrow \rightarrow \frac{\varphi + 1}{\varphi} = \frac{\varphi}{1} \rightarrow \rightarrow \varphi^2 - \varphi - 1 = 0$$

Lastly, using the quadratic formula the Golden Ratio  $\varphi$ , is found to be equal to  $\frac{1}{2}(1 + \sqrt{5}) \approx 1.618034$ . According to Euclid's definition  $\varphi$  is the greater of the two sections of segment AB. By this definition  $\varphi > 1$ , so the second solution to the quadratic equation  $\frac{1}{2}(1 - \sqrt{5}) \approx -0.618034$  is invalid.

Perhaps one of the most intriguing things about the Golden Ratio is its relation to Fibonacci's sequence. Leonardo of Pisa, also known as Leonardo Fibonacci published *Liber abaci* (Book of the abacus) in 1202. In it he put forth the well-known rabbit problem in which a man starts with a pair of immature rabbits that produce a pair of offspring each month, and with two months growth, each offspring pair also reproduces monthly (Atalay, 2004). Tracking the

number of adult rabbit pairs leads to the Fibonacci sequence 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89... where each successive term (starting with the third term) is obtained by adding the previous two terms together. This sequence, named by Edouard Lucas (1842 – 1891) for Fibonacci himself, can be seen in a wide variety of natural occurrences other than *Liber abaci*'s idyllic reproduction of rabbits: optics of light rays, the arrangement of leaves on a stem, the scales on a pineapple, or the flowers of an artichoke. In 1611 Johannes Kepler is credited with discovering that the ratio of a term in the Fibonacci sequence to term immediately prior to it approached the Golden Ratio as the number of the terms becomes larger. Mathematically, if we call the  $n^{\text{th}}$  term in the Fibonacci sequence  $X_n$ , Kepler discovered that  $\frac{X_{n+1}}{X_n} \approx \varphi$  for large values of  $n$ . Out of this discovery comes an interesting mathematical “trick.” Picking any two numbers to start with creates a sequence in the same way as the Fibonacci, by adding the previous two terms to obtain the next term in the sequence. Much like Kepler’s discovery, the ratio of the  $\frac{X_{n+1}}{X_n}$  converges to  $\varphi$ . For example, my birthday is July 29<sup>th</sup>. Starting with the values 7 and 29 the following sequence is obtained: 7, 29, 36, 65, 101, 166, 267... 86,115, 139,337... From this sequence the 19<sup>th</sup> and 20<sup>th</sup> terms are 86,115, and 139, 337 respectively. If, like Kepler, we take the ratio of the 20<sup>th</sup> term to the 19<sup>th</sup> term we find the following:

$$\frac{X_{n+1}}{X_n} = \frac{139,337}{86,115} \approx 1.618034$$

The programming challenge for the Golden Ratio is to write a program, including a **for** loop, that creates the sequence for the mathematical “trick” described above to approximate phi, each student using their own birthday. An example of this code is shown below:

```
% Define two numbers
a = 7
b = 29
n = 20
```

```

% create series (n = 20) similar to
Fibonacci
s = zeros(n,1);
s(1,1) = a;
s(2,1) = b;

for i = 3:n
    s(i) = s(i-1) + s(i-2);
end

Sequence = s

% Use Kepler's Ratio to find phi
phi = s(n)/s(n-1)

```

In this case we find  $\varphi = 1.6180$  which is accurate to four decimal places.

### AP Calculus, a Programming Challenge

Moving into a higher level of mathematics, the following programming challenges are intended to be used in an Advanced Placement Calculus course. This first challenge is to write a numerical approximation for differentiation, the second is to do the same thing for the integration of a definite integral. The definition of a derivative for a function  $f$  makes use of what is often referred to as Fermat's difference quotient. Using this we find that the derivative of  $f$  is the function  $f'$  such that

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}.$$

From this definition students can write a program to numerically approximate the derivative of a given differentiable function by using the slope equation and adjusting the step size of  $\Delta y = h$ .

The example code and plots below show an approximation for the function  $f(x) = x^3$ . An excellent way to demonstrate the effect of the limit as  $h \rightarrow 0$  can be seen in the plots below.

Figure (7) shows a step size  $h = 0.1$  while figure (8) shows a step size  $h = 0.01$ . For both cases the actual value,  $f'(x) = 3x^2$ , is shown in blue, while the numerical approximation is

shown in red. It is quite clear that decreasing the step size from 0.1 to 0.01 leads to a much closer approximation of the true value.

```
f='x^3'
h=0.01
xpts=(-1:h:1)'% x values from -1 to 1 increments of size h
fpts=(xpts).^3 % values for f(x) for given x

% Use the Slope Formula to Approximate the Derivative
slope=zeros(length(xpts)-1,1)
for count=1:length(xpts)-1
    slope(count,1)=...
        (fpts(count+1,1)-fpts(count,1))/(xpts(count+1,1)-xpts(count,1))
end

fprime='3*x^2'

% Plot Compares Numerical Approximation to Actual Value
figure(1)
fplot(fprime,[-1 1], 'b') % Actual Value
hold on
plot(xpts(1:length(xpts)-1,1), slope, 'r') % Approximation
```

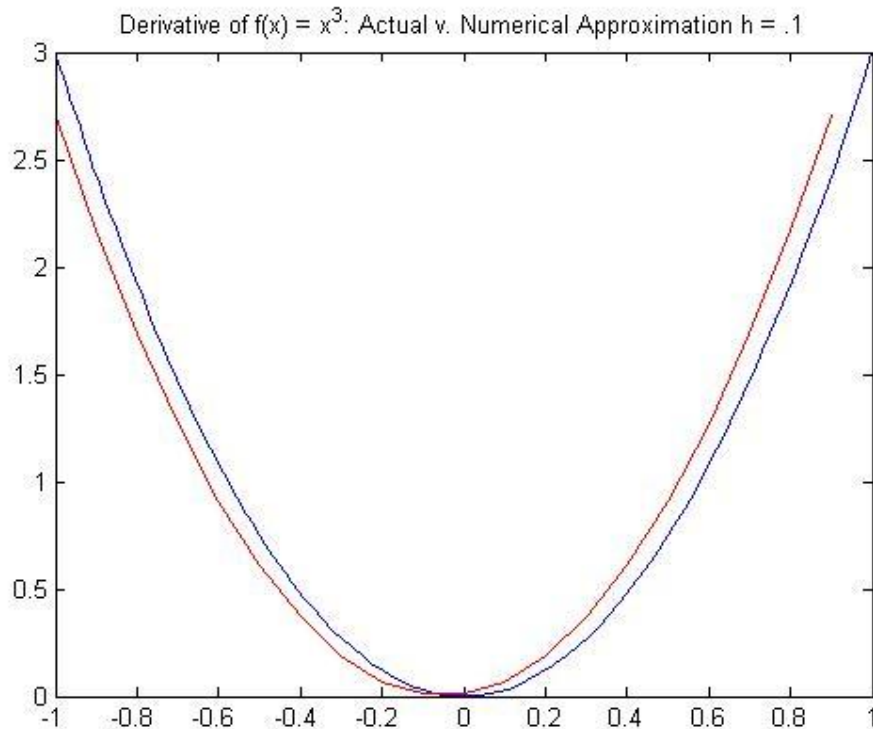


Figure (7)

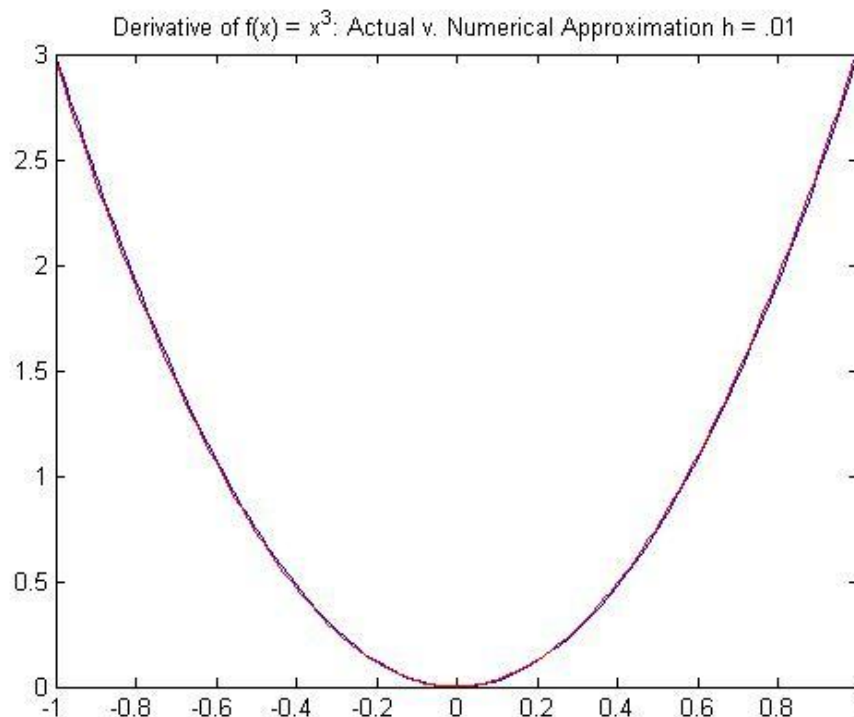


Figure (8)

The second programming challenge is to create a program that yields a numerical approximation to a definite integral by approximating the area under the curve. This can be done similar to Riemann Sums by using rectangles in a number of ways. For the purpose of programming, the height of the rectangles could either be right or left justified, or, as in the case of the example program below, by using the midpoints of each rectangle's increment to determine the height. Students could also use the trapezoidal method to find a numerical approximation to a definite integral. In the case of the example below we once again are using the function  $f(x) = x^3$ . Here we want to approximate the following definite integral

$$\int_{-1}^1 x^3 dx$$

and compare it to its actual value



$$\int_{-1}^1 x^3 dx = \frac{x^4}{4} \Big|_{-1}^1 = \frac{1}{4} - \frac{1}{4} = 0$$

The code below is for approximating the definite integral above using 100 divisions. This yields an approximate area of 9.3675e-017 which is very close to the actual value of zero.

```
f='x^3'
fint='(x^4)/4'
a=-1;
b=1;
n=100;    % n = number of divisions
delta=(b-a)/n    % delta is delta x
xpts=(a:delta:b);

midpts=zeros(n,1);
for count=1:n
    midpts(count,1)=a+(delta/2)+(count-1)*delta; %finds midpoints
end

height=zeros(n,1);
for count=1:n
    height(count,1)=(midpts(count,1)).^3;    %finds height at mdpt.
end

area=zeros(n,1);
for count=1:n
    area(count,1)=height(count,1).*delta; % finds area of rects.
end

A=sum(area)    % finds the summation of all rectangles
```

### Other Areas of Interest: Engineering and Life Sciences

In the text *Experiments with MATLAB*, Moler (2011) offers some interesting connections between mathematics, programming and other areas such as engineering and the life sciences. In the exercises listed for Chapter 5: *Linear Equations*, Moler sets up systems of linear equations that can be used to solve a truss system in static equilibrium (exercise 5.6). Truss problems are a common part of the PLTW class Principles of Engineering (POE). This exercise uses programming to link the use of matrices in solving linear equations learned in math class to the truss problems faced in engineering. Likewise, exercise 5.7 uses matrices to solve an electronic circuit. Students who have taken the PLTW course in Digital Electronic, or AP Physics have had

to solve many circuits by hand. Here Moler demonstrates two different ways in which the circuit can be solved by setting up systems of linear equations in matrix form.

Another interesting topic covered is covered in Chapter 16: *Predator-Prey Models*. This chapter starts by looking at a single population and comparing the difference between exponential growth of a population and a more realistic model, logistic growth. The more realistic model is obtained by the introduction of carrying capacity, limiting factors on a population such as availability of nourishment and habitat. As the chapter progresses the effect of predators is also added to population model. At this point the math behind the program moves outside of the realm of the high school level, and uses one of MATLAB's ordinary differential equation solvers, but resulting plots are quite interesting and serve to show a use for the study of higher level mathematics.

**Conclusion:**

The ability to connect the study of mathematics to other disciplines and demonstrate the power of mathematical modeling can be taken to a new level with the introduction of computer programming to the math classroom. Through the use of MATLAB students are able to hone their problem solving skills working through the algorithms needed to write a functional program, and can explore uses for mathematics that extend beyond the traditional story problem. This can open a new world of interest in the classroom as students can get a truer sense of how mathematicians, engineers, scientists and computer programmers uses computers to model and explore data while being introduced to some of the basics of computer programming in the process. Additionally, the use of MATLAB in a high school math class can act as an alternative way to introduce more students to the computer sciences and computer programming as it would

address several principles considered important to the learning of computer science by the CSTA: general capabilities and skills, problem solving and algorithmic thinking, integrative and interdisciplinary knowledge, comprehensive programming, and gender differences and minority issues. Finally, it is possible that an introduction to computer programming in high school will make it more likely for students to successfully complete college degrees in areas like computer science, engineering, and the sciences.

## REFERENCES

- Atalay, B. (2004). *Math and the Mona Lisa: The art and science of Leonardo da Vinci*. Washington, D.C: Smithsonian Books.
- Common Core State Standard Initiative: Standards for Mathematical Practice. (2011). Retrieved from <http://www.corestandards.org/the-standards/mathematics/introduction/standards-for-mathematical-practice/>
- Deek, F., Jones, J., McCowan, D., Stephenson, C., and Verno, A. (2004). *A Model Curriculum for K - 12 Computer Science: Final Report of the ACM Task Force Curriculum Committee*. Retrieved from [http://www.acm.org/education/education/curric\\_vols/k12final1022.pdf](http://www.acm.org/education/education/curric_vols/k12final1022.pdf)
- Dick, T. & Hollebrands, K. (2011). *Focus on High School Mathematics: Technology to Support Reasoning and Sense Making*. Retrieved from <http://www.nctm.org/catalog/product.aspx?ID=14315>
- Flynn, P. (2009). *NUMB3RS ACTIVITY: Coded Messages*. Retrieved from <http://education.ti.com/calculators/downloads/US/Activities/Detail?id=7509>

- Hechinger, J. (2010). U.S. Teens Lag as China Soars on International Test. Retrieved from <http://www.bloomberg.com/news/2010-12-07/teens-in-u-s-rank-25th-on-math-test-trail-in-science-reading.html>
- Livio, M. (2002). *The golden ratio: The story of phi, the world's most astonishing number*. New York: Broadway Books.
- Moler, C. (2011). *Experiments with MATLAB*. [ pdf ]. Retrieved from <http://www.mathworks.com/moler>.
- Morrison, J. (2006). *TIES STEM Education Monograph Series Attributes of STEM Education The Student The School The Classroom*. Retrieved from [http://www.tiesteach.org/documents/Jans%20pdf%20Attributes\\_of\\_STEM\\_Education-1.pdf](http://www.tiesteach.org/documents/Jans%20pdf%20Attributes_of_STEM_Education-1.pdf)
- Sabochik, K. (2010). Changing the Equation in STEM Education [Blog post]. Retrieved from [http:// www.whitehouse.gov/blog/2010/09/16/changing-equation-stem-education](http://www.whitehouse.gov/blog/2010/09/16/changing-equation-stem-education)
- Schollmeyer, M. (1996). Computer Programming in High School vs. College. *ACM SIGCSE Bulletin*, 28. doi:10.1145/236452.236584

Stephenson, C., Gal-Ezer, J., Haberman, B., & Verno, A. (2006). *The New Educational Imperative: Improving High School Computer Science Education*. Retrieved from [http://csta.acm.org/Communications/sub/.../White\\_Paper07\\_06.pdf](http://csta.acm.org/Communications/sub/.../White_Paper07_06.pdf)

Thomasian, J. (2011). *Building a Science, Technology, Engineering, and Math Education Agenda*. Retrieved from <http://www.nga.org/files/live/sites/NGA/files/pdf/1112STEMGUIDE.PDF>

## ACTIVITIES APPENDIX

It is the purpose of this appendix to offer an example of a way in which activities outlined in this paper could be utilized in accordance with the Common Core State Standards Initiative in Mathematics and the attributes of a STEM education to promote student engagement, mathematical reasoning, problem-solving, communication, and understanding. According to Morrison (2006), a school and classroom supportive of STEM education should be “active and student-centered” (p.5), allowing students to be responsible for their own learning through activities that demand investigation and analysis. Additionally, there should be an emphasis on the design process as a model way in which to approach problem-solving. Since an underlying goal of this paper is an attempt to tie the mathematics classroom to the PLTW engineering classroom, the design process will be discussed as parts of the 12 step PLTW design process. The steps are as follows: 1. Define the Problem, 2. Brainstorm, 3. Research and Generate Ideas, 4. Identify Criteria and Constraints, 5. Explore Possibilities, 6. Select an Approach, 7. Develop a Design Proposal, 8. Model or Prototype, 9. Test and Evaluate. 10. Refine, 11. Create or Make, 12. Communicate Results.

Morrison (2006) suggests that STEM educated students be problem-solvers, innovators, inventors, self-reliant, logical thinkers who are technologically literate. She defines a problem solver as “able to frame problems as puzzles and then able to apply understanding and learning to these novel situations” (p.2). Innovators are able to use the design process to formulate an investigation, and then, in the role of an inventor, create and implement a solution. These three attributes reflect the PLTW design process. The problem-solver models step one by defining a problem. The innovator uses steps 2 through 6 in which students brainstorm, research, explore

and select an approach to problem-solving. Lastly, the inventor works through steps 7 – 12 which include the development of a design and model, testing, refinement, and finally communication of the solution. The idea of using MATLAB in the math classroom is to help create an active, student-centered learning environment in which students work through a design process taking on the roles of problem-solvers, innovators and inventors while increasing their technological literacy.

While the process of writing a program utilizes the design process and requires an understanding of the mathematical algorithm to be used, manipulation of a program to investigate and test conjectures is also possible. In general, all of the suggested activities require the use of the design process in order to create and write a desired program. For example, when completing the activity *The Golden Ratio, a Programming Challenge*, students will have to work through the design process that is central to STEM education. First, wearing the hat of a problem-solver, they will need to define the problem and then begin to apply their prior knowledge of MATLAB coding and Fibonacci's sequence. At this time, they may also need to research new MATLAB commands or Fibonacci's sequence in order to proceed. As they move toward the role of an inventor, students will have to develop code, test it, refine it and eventually arrive at a solution. Having to code a program also requires a high level of communication. First, it is imperative that the coding is precise and correctly commands the actions of the computer. Secondly, it is customary to document code so that others can readily use and manipulate it. The discussion of activities that follow, Systems of Equations and Transformations/Translations, focus on the manipulation of a program to investigate and test conjectures.



The first activity, Systems of Equations, makes use of MATLAB's plot function to quickly produce neat and accurate graphical solutions to systems of linear equations. This ability to quickly and accurately produce graphical representations of symbolic algebraic equations is an advantage of the use of technology. It allows students to create many more graphical representations than they could by hand and use these representations for investigative purposes. In the case of programming using MATLAB, students must understand more of the underlying functions being performed by the program to create the graphs than if they were to use a graphing calculator such as the TI – 84. When plotting an equation in MATLAB the programmer not only needs to determine the range and scale needed for the x and y-axes, but also the domain of x-values over which to plot the graph. This requires more involvement in creating a graph that highlights crucial aspects of a function than blindly zooming in and out on a graphing calculator, and eventually losing sight of the scale of the x and y-axis. The following paragraphs will discuss ways in which students can use MATLAB and systems of linear equations to investigate the cognitive fidelity of a solution, and the process of solving a system of linear equations by hand using substitution.

The NCTM (National Council of Teachers of Mathematics) in their publication, *Focus on High School Mathematics: Technology to Support Reasoning and Sense Making* defines cognitive fidelity to be “faithfulness of a machine’s representation of a mathematical object to the individual’s perception of the mathematical object being represented” (Dick, 2011, p.63). Some minor changes to the first equation from the Systems of Equations activity can offer an exploration into cognitive fidelity, and making sense of computer results. Modifying the

equation  $y_1 = 2x + 9$  and keeping the equation  $y_2 = 3x - 16$  a programmer could obtain the results seen below on the left side of figure (1). Posing the question, what would happen if the scales of the x and y axis were changed to only show a small part of the graph of  $(y_1 \text{ v. } y_2)$ , students would have to make a prediction and then investigate their claim. The graph on the right of figure (1) is also of  $(y_1 \text{ v. } y_2)$ , but on a different scale.

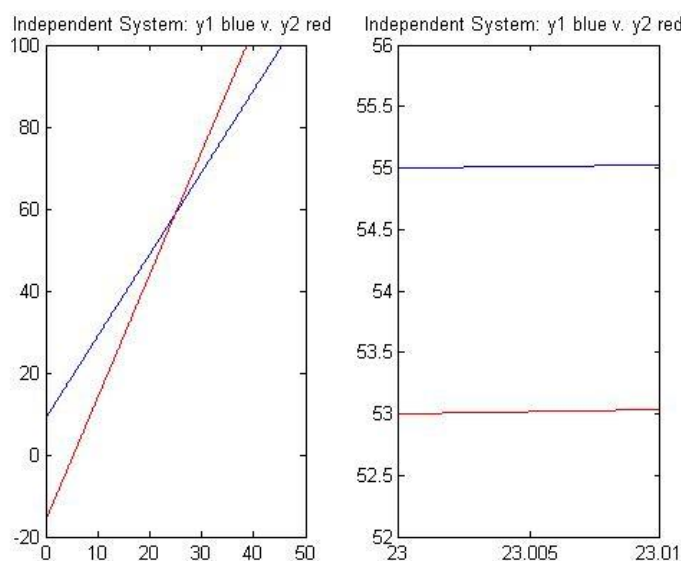


Figure (1)

Here it may be hard for a student to see that the system of equations on the right is an independent system since the lines, at a glance, may be mistaken as parallel. This is an example of cognitive fidelity, where the results given by the computer program do not match the true mathematical answer. Students could be asked to justify, using explanations of slope and y-intercept, how they would know that the given system of equations is independent, not inconsistent as it seems to be in the right side of figure (1). As a follow up activity, students could be asked to find a way in which an inconsistent system might be graphically mistaken for a dependent system (zooming out by creating large scales for the x and y-axis could make parallel line look as if they were the same line). Using the design process students would have to work

through the jobs of problem-solver, innovator and inventor to arrive at a solution. In addition to supporting the design process affiliated with STEM education and making learning student - centered through investigation, solving systems of equations meets the following common core standards: A-REI.6., A-REI.10., A-REI.11., F-IF.7., and N-Q.1. Full definitions of all standards mentioned in this paper are listed at the end of this appendix.

Another way in which to use the equations and programs given in the Systems of Equations would be to use the first pairing of equations,  $y_1$  and  $y_2$ , and have students solve this system by hand using substitution. Starting with linear equation  $3x - 16 = -2x + 9$ , students could complete exercise 2.1 in *Focus on High School Mathematics: Technology to Support Reasoning and Sense Making* (Dick, 2011, p.20). This exercise suggests having students graphically represent each algebraic step made by hand in an attempt to link the graphical and symbolic approach to solving linear equations, as well as their relation to the properties of equality and the properties of number and operations. For example, figure (2) below shows the original two functions in red and blue, and one choice for a first step in solving  $3x - 16 = -2x + 9$ , adding  $2x$  to both sides to obtain  $5x - 16 = 9$ ,  $y = 9$  in black and  $y = 5x - 16$  in green.

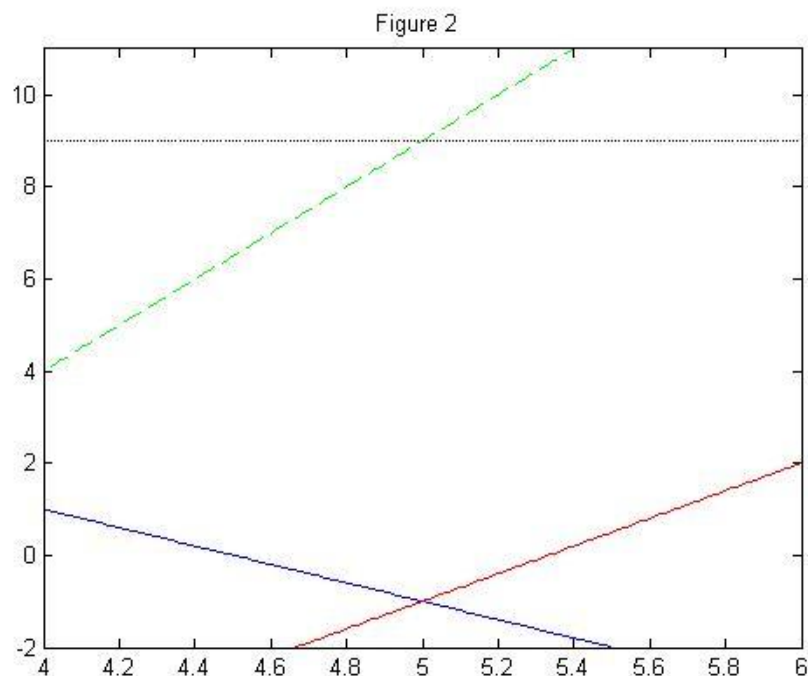


Figure (2)

Through this activity students are encouraged to make observations and conjectures about the effect of applications of properties of equality and properties of number and operations and make connections between the steps made in solving by hand and their graphical representations. For example, in the step shown above, adding  $2x$  to both side of the equation is a property of equality. Students may notice that a property of equality graphically preserves the value of  $x$  and the  $y$ -intercepts, but changes the slopes of the graph pairs. Further investigation could look at the effect of adding or multiplying a constant to both sides again relating the graphical representation to the ideas of slope and intercepts. This exercise coincides with the common core standards A-REI.1 and A-REI.3

Additionally, Plot could be used explore transformations on a variety of functions and create generalizations about the effects of different parameters on multiple functions. While the

example used in the body of this paper was limited to exponential functions, the programming would be the same for other families of functions. Example 2.3 Primitive Parameter Exploration from *Focus on High School Mathematics: Technology to Support Reasoning and Sense Making* (Dick, 2011) uses this idea assigning teams of students each a different function family to explore. Sticking to the example from the body of the paper, a team may be asked to explore the function family  $f(x) = a * e^{(b*x-c)} + d$ . Having been introduced to transformations earlier in the year while studying linear and quadratic equations students may recall, for example, that the value of  $d$  moves a graph up and down. The purpose of the programming is to create a base program in which students can easily manipulate the parameters to test conjectures and create generalizations through the creation of multiple graphs. One way to program this is shown below.

```
x=-3:.1:3;
a=1;
b=1;
c=1;
d=1;
y1=a*exp(b*x-c)+d;

figure(3)
plot(x,y1,'b')
hold on
title('y1=a*exp(b*x-c)+d')
```

Students will have to use problem-solving and the design process to find ways in which to manipulate their program to show multiple graphs in one figure. This way the effects of a parameter can easily be seen, tested and compared. They will also need to utilize their programs to draw generalizations and conclusions about the effects of a given parameter. Comparing and contrasting the generalizations made by different groups for different function families can lead to a whole class discussion linking the symbolic expressions of a function to a graphical

representation. This exercise meets common core standards under the heading of analyzing functions using different representations.

**Common Core State Standards Initiative in Mathematics Identified in Appendix:**

**A-REI.1.** Explain each step in solving a simple equation as following from the equality of numbers asserted at the previous step, starting from the assumption that the original equation has a solution. Construct a viable argument to justify a solution method.

**A-REI.3.** Solve linear equations and inequalities in one variable, including equations with coefficients represented by letters.

**A-REI.6.** Solve systems of linear equations exactly and approximately (e.g., with graphs), focusing on pairs of linear equations in two variables.

**A-REI.10.** Understand that the graph of an equation in two variables is the set of all its solutions plotted in the coordinate plane, often forming a curve (which could be a line).

**A-REI.11.** Explain why the  $x$ -coordinates of the points where the graphs of the equations  $y = f(x)$  and  $y = g(x)$  intersect are the solutions of the equation  $f(x) = g(x)$ ; find the solutions approximately, e.g., using technology to graph the functions, make tables of values, or find successive approximations. Include cases where  $f(x)$  and/or  $g(x)$  are linear, polynomial, rational, absolute value, exponential, and logarithmic functions.★

**F-IF.7.** Graph functions expressed symbolically and show key features of the graph, by hand in simple cases and using technology for more complicated cases.★

**N-Q.1.** Use units as a way to understand problems and to guide the solution of multi-step problems; choose and interpret units consistently in formulas; choose and interpret the scale and the origin in graphs and data displays.

## MATLAB CODE APPENDIX

**1. Systems of Equations & Plot Function**

```
clear all
close all
clc

%% Plot Function

x=-10:.1:10;
y1=-2*x+9;
y2=3*x-16;
y3=-2*x+3;
y4=(-4*x+18)/2;

figure(1)
plot(x,y1,'b')
hold on
plot(x,y2,'r')
title('Independent System: y1 blue v. y2 red ')
grid on
axis([2 7 -3 2])

figure(2)
plot(x,y1,'b')
hold on
plot(x,y3,'r')
title('Inconsistent System: y1 blue v. y3 red ')

figure(3)
plot(x,y1,'b')
hold on
plot(x,y4,'r')
title('Dependent System: y1 blue v. y4 red ')
```

**2. Matrices**

```
clear all
close all
clc

% Enter Given Matrices
A=[1 2;3 4];
B=[6 7;8 9;1 2];
C=[0 3;4 2];
D=[3 1 1;-1 2 3];

% Explore Addition/Subtraction
ac=A+C
%ad=A+D

% Explore Scalar Multiplication
```



```

x=3;
xa=x*A

% Explore Matrix Multiplication
ac=A*C
ca=C*A
ad=A*D
%ab=A*B

% Explore Matrix Multiplication A*C v. A.*C
AstarC=A*C
AdotstarC=A.*C

```

### 3. Transformations & Subplots

```

clear all
close all
clc

%% Plot Function

x=-3:.1:3;
y1=exp(x);
y2=exp(-x);

figure(1)
plot(x,y1,'b')
hold on
plot(x,y2,'r')
title('y1=exp(x) blue v. y2=exp(-x) red ')

y3=exp(x)+1;
y4=exp(x)-2;

figure(2)
plot(x,y1,'b')
hold on
plot(x,y3,'r')
hold on
plot(x,y4,'g')
title('y1=exp(x) blue, y2=exp(x)+1 red, y4=exp(x)-2 green ')

%% Graphing

figure(3)
subplot(2,2,1)
plot(x,y1,'b')
title('y1=exp(x)')
axis([-3 3 -3 3])
axis square
% An example of a way to plot x = 0 and y = 0 only.

```

```
grid1 = [-10:.1:10];
xgrid = zeros([size(grid1)]);
ygrid = zeros([size(grid1)]);
hold on;
plot(xgrid,grid1,'black')
plot(grid1,ygrid,'black')

subplot(2,2,2)
plot(x,y2,'r')
title('y2=exp(-x)')
axis([-3 3 -3 3])
axis square
% An example of a plot with the grid.
grid on

subplot(2,2,3)
plot(x,y3,'r')
title('y3=exp(x)+1')
axis([-3 3 -3 3])
axis square

subplot(2,2,4)
plot(x,y4,'g')
title('y4=exp(x)-2')
axis([-3 3 -3 3])
axis square
```