Mt. San Antonio College
ENGR 7 – MATLAB for Engineers – Reference # 41964
Course Syllabus – Spring 2018

Basic Course Information	
Days/Times/Location	Integrated Lecture/Lab: MW 12:00 – 3:10 pm, Bldg 11, Room 2107
Duration	February 26th, 2018 – June 8th, 2018 (16 weeks)
Units	4; CSU/UC transferrable; Letter Grade Only; 54 Lecture Hours and 54 Lab Hours
Course Web Site	We will use Canvas for online activities and grade documentation. To access this Canvas, log into the Mt. SAC portal (http://inside.mtsac.edu) using your Mt. SAC username and password, then click the 'sign on to Mt. SAC Canvas' in the eLearning Resources tab.

Instructor Information	
Professor	Eugene L. D. Mahmoud
E-mail Address	emahmoud@mtsac.edu
Office Phone	909.274.5745
Office Location	Building 60 – Room 1407
Office Hours	MW: 11:00am – 12:00 pm, Bldg 60-1407 or 11-2107 M: 9:40pm – 10:10 pm, 11-2107 Tu: 1:45pm – 2:15pm pm, Bldg 60-1407 or 11-2304 W: 3:10m – 4:10pm, Bldg 60 – 1407 or 11-2107 And by appointment

Course Description

Programming Applications for Engineers is an integrated, lecture-lab, semester-long course that introduces lower-division engineering and physics students to numerical analysis and computational problem solving. Students will use applied mathematical techniques, plotting, logic operations, data acquisition, and graphical user interfaces to design, test, and debug numerical algorithms. The assessment of student learning outcomes is evaluated primarily by each student's documentation of laboratory activities, development of authentic computational models and project solutions with industry standard software. *PREREQUISITE:* Math 180

Required Materials

MATLAB Student *and* the Symbolic Math Toolbox by Mathworks (~$60)

To use your MATLAB license, you will need access to an internet-connected, MATLAB-ready computer outside of class, AND a USB memory stick, smartphone, or "cloud-based" drive to digitally back up all work.

Strongly Recommend Materials:

Textbook: Moore, Holly, *Matlab for Engineers*, 4th Edition
This text was designed for freshmen engineering students in introductory programming courses at two-year colleges. If you do not have a computer programming reference text or significant programming experience, you should obtain this text.

Course Measurable Objectives

Students will:

1. Identify, formulate, and solve computational problems using a methodical approach. Design algorithms and flowcharts to facilitate programming and problem solution.
2. Create, test, and debug computer programs using procedural and object-oriented approaches.
3. Apply numeric techniques and computer simulations to solve engineering-related computational problems.
4. Create and apply MATLAB computer programs to analyze data and to generate tables, charts, and graphs.
5. Communicate analytical approaches and results according to standard engineering practices.
6. Design and document computer programs in a careful and complete manner so as to facilitate analysis and debugging by another programmer, and to anticipate and resolve user errors.
7. Create embedded programs to run on a microcontroller.

Professor Expectations

Each of you in this course intends to be either a scientist or an engineer, and as such you must develop a set of skills that are unique to these fields. In this course, you will use your knowledge to make things work in a real-world context. You will calculate, compute, solve, troubleshoot, design, analyze, propose, design, build, present and experience some failure as part of your process. We hope you will explore everything the class has to offer, make new friends, embrace many new challenges, and have some fun along the way. Be Respectful. Be Prepared. Be Productive. Be Kind. I expect you to:

- Access the web site(s) daily for information on this course.
- Participate in class by asking questions, answering questions, participating appropriately in all discussion, laboratory and collaborative activities.
- Review class activities, your lecture notes, and supplemental materials before class and after class, paying attention as you go, documenting your progress in your laboratory posts, until you understand everything that was presented in class.
- Complete all of the assigned challenge activities and submit them by the due date.
- Come to office hours, SI sessions, and study with other classmates to clarify what is not clear and to work on additional activities.
- Complete your work on all activities and assessments and projects neatly and clearly.
- Practice not until you are good at something, but until you do not make mistakes in the context of your process.

Professor Expectations

I will listen to you, treat you with respect, and work with you to support your success in this class. I will not intentionally mislead you. Please trust that I have a plan for this course.

I will make every effort to respond directly to student questions and to e-mail messages in a timely manner. I will make every effort to grade your work in a timely manner.

I will make every effort to arrive on time. I will notify the class via Canvas if I expect to be late. I will notify the class and the division office if I expect to be absent.
I will use the laboratory portion of the course to provide lectures, demonstrations and assistance.

Challenges

Here, we describe the assignments in general. We'll call them challenges, not just to be cute, but because we want you to think of them as challenges you want to try. These challenges reflect the different roles engineers take on: mathematical modeling, technical writing, peer evaluation, computational modeling, debugging and program design. Some challenges are optional. Some challenges will need part one complete to unlock part two. Some challenges are collaborative.

For challenges completed using MATLAB, submissions of your work to the course website should be in the form of a published document file (*.pdf) including your actual code, outputs, and any comments you choose to make. All submissions should be checked for errors. Please note that code requiring user inputs will not produce any meaningful outputs. Submit images of your outputs as captioned *.png files and captioned .

Laboratory posts should be able to clearly communicate what you learned to someone who is not enrolled in the class. For each day of class, a complete journal entry should be submitted by midnight and include the following: 1) The date, 2) A description of the activities that the class explored that day, 3) Answer the question "What did I learn?" Be specific and include documentation (i.e. skills learned, vocabulary, description of steps taken, photographs of data or fabrications, calculations, sketches, screenshots, and animations) 4) Answer the question "What questions do I have?" or "What questions did I have during this lesson?". Please see the appendix for examples of forum posts.

Integrated Lecture/Lab:

Laboratory activities (Crafting) are conducted under the active, continuing presence of the professor. The instructor will grade laboratory activities outside of the class.

Assessments

The purpose of the Celebration of Knowledge (Final Exam) is to demonstrate your programming and problem solving abilities in a time constrained setting. It is scheduled for Monday, June 11th, 2018 and Wednesday June 13th, 2018 from 10:30 am – 1:00 pm.

This is a project-based course rather than a topic-based course, and you will learn the computational and problem solving skills needed to complete your Epic Quests (projects) rather than progressing through a typical programming text. You will be responsible for figuring out what you need to learn and for teaching yourself and each other those things. I will help in your approach. All projects will be graded using rubrics directly related to the course measurable objectives. The following is a tentative overview of each project.

a. Epic Quest 1:
    i. Team size: 3-4
    ii. Project description: Make measurements of a real one dimensional system, create a Matlab model and then compare that model to actual experiment.
    iii. Deliverable: A 10 page group report and program.
b. Epic Quest 2:
    i. Team size: 1
    ii. Project description: Make measurements of a real three dimensional system, create a Matlab model and then compare that model to actual experiment.
    iii. Deliverable: A 3 page individual report and program.
c. Epic Quest 3:
    i. Team size: 2
    ii. Project description: Create a closed loop control system that makes measurements of a physical system using LabVIEW and then control actuators to adjust that system within some design tolerances.
    iii. Deliverable: A group presentation and program.

Tentative Grading Scale

The grading format of the course is modeled as an experience point system in a role-playing game. The different point scales are named after various computer languages and are associated with different letter grades. You will begin on the first day of class with zero experience points (XP). You gain XP by completing challenges and quests, dominating stage bosses, and demonstrating proficiency in the final celebration. There is no curve.

Grading Rationale / Point Scale Nomenclature			
F	ENIAC: 0 – 399	F	Fortran: 400 – 799
F	COBOL: 800 – 1199	F	BASIC: 1200 – 1599
F	B: 1600 – 2199	F	Pascal: 2200 – 2799
F	C: 2800 – 3399	F	MATLAB:3400 – 4199
F	C++: 4200 – 4999	F	LabVIEW: 5000 – 5999
D	Mathematica: 6000 – 6999	C	Python: 7000 – 7999
B	Java: 8000 – 8999	A	Scratch: ≥ 9000

Category Point Totals	
Celebration of Knowledge	1500
Challenges	2500
Crafting	1000
Epic Quests	5000

Participation

As per the Mt. SAC catalog and state law, attendance is mandatory in all classes. If you are unavoidably late or absent, you are responsible for obtaining the information from a classmate. Be sure to contact me in advance if you are not able to attend lab. I cannot guarantee that you will be re-assessed on any class activities missed. If you must come in late to class without prior notice, please do so quietly and look embarrassed. I may drop a student if he/she is either not attending class regularly or not participating in the class. Keep in mind that I can only drop a student within the allowed time for dropping a class and cannot drop a student after the last date to drop a class has passed.

Student Equity and Academic Accommodations

My interest is to promote student success and help you to be successful in my class. If you have a specific need that I can address to assist you to be successful in my class, please discuss it with me. If you have a disability, a medical condition or if you believe there is a factor in your life that might prevent you from doing your best, please see me privately to discuss your needs. Students with disabilities should check with the Mt. SAC Disabled Student Programs & Services available. If you find a feature of the course inaccessible to you, please contact me or DSPS or the ADA/504 Compliance Officer at (909) 275-4225.

Late *Policy*

Probably the most difficult thing about college is that “life goes on.” Sadly, friends and family don’t wait until we’re done with our studies to get sick, have problems, or otherwise impact our lives. Since we’re all adults, I do not feel I have the authority to decide whether a reason for missing assignments or classes is “good enough” – all reasons are good enough; if you say there was a good reason for you to be late, or skip some homework, I believe you. However, the point of this class is to come away with a certain level of knowledge in a specific amount of time, not just be enrolled through it. If life interferes with your ability to successfully complete the assignments in a timely manner, you are advised to drop the class until you have the opportunity to commit your personal resources appropriately. Ultimately, you decide how much you are able to study and I’ll assign a grade as the syllabus dictates. The bottom line is that tardy, incomplete, missing or un-attempted coursework is not acceptable unless written, verifiable proof of an emergency, such as severe personal illness is provided by a doctor. I am sympathetic about emergencies and I may make exceptions.

Collaboration and Academic Integrity

Collaboration is an important part of your engineering education because supporting others' learning helps you understand the subject better, working in teams will prepare you for your future careers, and crediting others develops your ethical practices. I encourage you to work together in solving homework problems and coming up with answers to questions! Your colleagues in the class can be your most useful allies in your mutual success in this course. Put together a study group at the beginning of the semester and meet regularly.

However, using someone else's work or ideas without proper citation is academically dishonest. It is important that you read and understand the Mt. SAC Cheating and Plagiarism policy from the Catalog and let me know if there is anything that you do not understand regarding this policy. If you are having difficulty in understanding an assignment or activity, it is your responsibility to get help from me. You should be prepared to explain how you completed any work you submit. If you receive help from another student or reference, cite your source in writing.

Responsible Use Policy

This is a Bring Your Own Device (BYOD) class. You are encouraged to bring your cell phones, iPads/pods, laptops, kindles, etc. When using any internet connected technology, you are agreeing to school site policies and my Responsible Use Policy and described below:

- I will use technology for education purposes and activities.
- Keep my personal information, and that of others, private.
- Show respect for myself and others when using technology, especially social media.
- Properly acknowledge others for their ideas and work.
- Report inappropriate use of technology immediately.
- Never leave my device unattended or myself logged in when not using it.
- I will know the limitation of my data plan and use my device within those limits.

Class Conduct

Due to the presence of computers, electrical cables, and other technical equipment in the classroom; only natural foods and water may be consumed in class away from the lab equipment. Please do not bring processed food, carbonated drinks, refined sugar or artificial sweeteners into our class. If you make a mess, clean it up. Please silence your cell phones and electronic devices during class. Please perform personal hygiene, and make all phone calls and texts outside of the classroom. Students are responsible to protect, save and backup all evidence of their work. All problems with digital media production are the responsibility of the student and are a not an acceptable excuse for late, incomplete or poor quality work. Do not remove or plug in any USB peripherals (mice, keyboards, device chargers, etc.) or other electrical cords into computers or outlets without my permission.

Tentative Scope and Sequence

Week	Discussion	Lab Activity	Week	Discussion	Lab Activity
1	Engineering Problem Methodology	Engineering Design	**6**	Matrixes	Matrix Algebra
	Introduction to MATLAB	MATLAB Interface	**7**	Solving Simultaneous Equations	Engineering Applications
	Simple Operations	Script Files	**8**	Symbolic Mathematics	Matrix Data Types
	Output Options		**9**	Numerical Techniques	Interpolation and Curve Fitting
2	Functions	Diary function		Introduce Project 2	Project 2
	User Defined Functions		**10**		Project 2
3	x-y plots	User Inputs	**11**	Intro to LabVIEW	LabVIEW Interface
	3D plots		**12**	LabView Programming	Logger Pro Interface
	Relational and Logical Operators		**13**	LabView Plotting & Data I/O	Project 3
4	Selection Structures	Project 1	**14**	Memorial Day	Project 3
	Loops	Distribution Game		Electronics hardware	Project 3
5	Mathematical Modelling	Project 1	**15**		Project 3

Appendix 1: Forum Post Student Exemplar

Lab/Tues:
In this laboratory, we learned about matrix multiplication while furthering our knowledge of matrix manipulation.

Learning Outcomes:

dot-product:

- "dot(A,B)"

- Essentially the sum of a vector multiplication.

- i.e. A = <a,b>, B = <c,d>. Dot product = ac+bd (scalar value).

Summarily, dot-product is vector multiplication followed by a sum of elements, for a 1xN vector (where N is an arbitrary length). The MatLab “dot(A,B)” (where A and B are matrices of size MxN is likely founded on two other MatLab processes; namely, vector multiplication (A.*B) and the sum(A) function.

We recently learned that the one-dimensional dot product of two 2xN array will output a vector with size 1xN. In doing this it vector multiplies the rows of the two matrices and adds the columns up. i.e. A = [a b;e f], B=[c d;g h]. Dot product = [ac+eg,bd+fh] (vector value). So, using a dot product of dimension = 1 (dot(A,B,1)), we can find the dot products for multiple vectors by making a matrix of the target vectors in parallel columns. i.e. we want the dot product of arrays A.B, C.D, and E.F in one line of code. Succinctly: dot([A',C',E'],[B',D',F'] = [A.B C.D E.F]. Pretty neat. But there's a slightly *easier* way without the transposition of matrices. Why? I suspect this function is built on the foundation of the "sum()" MatLab function.
We know that vector multiplication is the multiplication of a matrix along their parallel rows (same row index). We also

understand that the sum function sums a 1xN vector by adding each element in the row; however, when an array of size MxN is processed in the sum() function, we see that the columns' elements are added instead, outputting a vector of size 1xN. We see this happening when arrays of size MxN are run through the dot(A,B) function. However, the use of the sum, like the dot, function is overloaded to have the possible input of a dimensional parameter. i.e. sum([a b;c d],1) = [a+c, b+d] (same without the dimensional parameter: "1"); and, sum([a b;c d],2) = [a+b;c+d]. We see that this holds true for the dot function as well. The importance in this is the elimination of a tedious transposition and makes the inputting of the vectors much more intuitive since we are used to seeing single vectors in rows in lower mathematics.

Note: maybe it would be better to grow accustomed to the standard input format as this method will give different outcomes for fields
Note: the dimensional parameter doesn't seem to be associated with the actual size of the arrays, such as: sum(A,3+) = A (the same as the input).

Determinants:

- Found by subtracting the product of the major diagonal of a matrix by the product of the diagonal of the same, flipped (left-right) matrix (for a 2x2 matrix).

- MatLab does this with the function det(A), where A is the matrix in question.

I learned that there are multiple ways to find the determinant of an array. I'm especially unfamiliar with Professor Mahmoud's diagonal-product-sum minus inverse-diagonal-product-sum method. I am more comfortable seeing determinants calculated as the screenshot I uploaded titled "simple determinant hand-calculation".

Cross Products: (null)

There is no section for cross products because I consider them the same as determinants.

Comments:

I forget how to solve determinants in a 4x4 matrix, and, for that, I am glad to have MatLab pick up the slack. I don't find it necessary that the students have a significant understanding of matrices to succeed in this class; however, I do very much appreciate all uploaded handouts. I think they come as a benefit for anyone who does look through them. Thank you.

Some students, myself included, have asked the instructor on how to go about concatenating different data types, like strings, into a matrix output. I understand this isn't necessarily MatLab's focus, as there are plenty of apt programs built with such intentions in mind. Until we touched upon the topic of matrix input, manipulation, processing, and output, I didn't find much interest in MatLab compared to other popular programs. I am excited about the upcoming lectures and warn that these lab posts may grow long.

Uploads:

practialExercises.m

- the examples of lecture topics done during lab in a single script file

- each of the functions run when uncommented

Center of Gravity Plot.png

- plot from "center of gravity/mass" exercise

- red lines ending with an "x" are position vectors of the various point masses (i.e. screw,bolt)

- blue line ending with "o" is the position vector of the center of mass

- all vectors are relative to origin <0,0,0>.

simple determinant hand-calculation.jpg

- photo of my take on how to solve a determinant

Appendix 2: Forum Post Student Exemplar

1. October 6, 2016
2. Today we learned how to use geometric constraint tools more like the fixed tool, horizontal, vertical, tangent, what it means to be fully constrained, parametric equations, hole tool, embossing and creating drawing files.
3. In the first 2 hours the professor walked around the classroom to check our homework on parametric modeling fundamentals and out inventor net. While he was doing that I was able to learn from him how to change the colors of lines and change line types. To do that we right click the line we want to change and click properties and then we should be able to change the color and the line type and the thickness of the line. While the professor was walking around he had us create 3D models and 2D sketches in Inventor. I learned to create the 3D model from looking at a multiview sketch and creating the 2D sketch was a little more difficult because of the circles. After the professor was done we started playing around with the geometric constraint tools.

We created a triangle and constrained it with the horizontal and vertical constraint tools. After we constrained and dimensioned the triangle this means that the last side of the triangle is fixed at that dimension and can no longer be changed unless the dimension of the other sides of the triangle are changed. This is why the object is noted as "Fully constrained" at the bottom right. We also learned about the fix tool (lock shape) that will fix the object to the plane you are drawing on. We learned about unnecessary dimensions. We tried to dimension the last side of the triangle and when we tried to do that this box pops up saying that this dimension will over constrain the sketch meaning this is a unnecessary dimension. As we learned to be a doubling dimension in our guideline.

In the third hour we learned about using parametric equations to help scale objects easier. We do this by changing the dimension to reference the first dimension with created and from then we reference all other dimensions from that so that when you change the dimension of the first dimension made it will change all the other dimensions. And we are able to see the functions we created under the manage tab->parameters. We also learned that we can change the functions from here too. We also reviewed what we learned last time about finding what the volume of our object is by right clicking the part we are on in the browser to the left and click properties and go under the physical tab. And this is where we will find the volume and and find the center of mass and change the object material. We also find out that even though we create the same object with the same dimensions the volume could come out different. This is because order matters when we create our model it does alter the value in volume. After that we learned about Project Geometry under the Sketch tab. This tool helps us to create offsets because it allows us to select the entire object then offset it.

During the fourth hour we had a DUN DUN DUN.. **STAGE BOSS!** We had 30 minutes to defeat the stage boss by creating 2 3D sketches. We learned more about the hole tool after we had our stage boss because we had to know it in order to complete the stage.

In the fifth hour we talked about the assignments due. We have nothing due on tuesday and parametric constraints due on Thursday. We also learned how to emboss text onto an object. (This changes the value of volume so remember to get the value before embossing) Lastly we learned about creating drawing files. we go from 3D dimension to flat paper surface. We have to have saved part files in order to create drawing files. When we open a new drawing file we open a part file on it and we click base and this dialog box pops up. This allows us to scale the object to be bigger or smaller on the paper. And we want it to show hidden lines. Then we can create the different views by moving the mouse above the object then click create the view then to the side of the object to create the side view and at an angle to create an isometric view.

With our final time left we had open lab!

4. Questions

So how does the layers work in inventor? Are we going to be using it? What the new project about?! Have a great weekend Professor :D! See you Tuesday!~~