ENGR-183: PROGRAMMING WITH MATLAB FOR ENGINEERS AND SCIENTISTS

Effective Term
Fall 2024

BOT Approval Date
12/14/2023

Discipline
ENGR - Engineering Technology

Course Number
183

Course Title
Programming with MATLAB for Engineers and Scientists

Short Title
Programming w/MATLAB ENGR&SCI

Credit Status (CB 04)
D - Credit - Degree applicable

Units

Lecture Units
3.00

Lab Units
1.00

Total Units
4

Hours

Lecture Contact Hours
48-54

Out of Class Hours Lecture
96-108

Lab Contact Hours
48-54

Out of Class Hours Lab
0.00

Total Contact Hours
96 - 108

Total Out of Class Hours
96 - 108

Total Student Learning Hours
192 - 216

Is this an existing Extensive Lab?
No

Do you want to propose this for Extensive Lab?

No

Distance Education

Yes

Distance Education

Distance Education Type

Both Fully Online and Hybrid Online

Fully Online Delivery Requirements:
a. Students must be notified via the college schedule of classes and the syllabus for the class if proctored tests are required for this course
b. Any planned face-to-face meetings, such as an orientation or study session, must be optional
c. The MSJC curriculum committee requires the use of accessible, asynchronous discussion as a component of every fully online course

Are you using publisher content?

No

Regular Substantive Interaction

Instructor-to-student, student-to-student, and student-to-content contact shall be distributed in a manner that will ensure regular substantive interaction (RSI) is maintained not only weekly but also for the duration of the course. Instructor interactions should be equivalent in time, quality and consistency to the engagement students would receive in a face-to-face course—for example, through announcements, discussion participation, personalized feedback, video messages, or other instructional communications. Instructors are expected to be responsive to student inquiries within a 24-72 hour time frame excluding holidays. RSI will be maintained through a variety of methods: • Orientation - Students must be oriented to the online and face-to-face portions at the start of the course. • Announcements - Announcements using the course management system must be posted at least weekly to keep students current on course events, due dates, materials, etc. • Discussion Forums - Content-related discussion forums within the designated CMS will include appropriate instructor participation and will be initiated and maintained, with timely feedback provided. Instructors are required to monitor the discussions and related activities to provide guidance and direction. Open-ended discussion boards may be used to further learning and interaction. • MSJC Communication – Instructor will use MSJC-administered communication tools (such as email or the designated CMS communication tools) to regularly communicate with and provide ease of access to students. • Timely Feedback - Timely feedback on student work will be provided by the instructor, which may include comments and/or rubrics, within the Grades area of the designated CMS. • Scheduled Meetings – Hybrid and synchronous courses will meet at regularly scheduled times. ADA Compliance - All course interactions and content must be designed with accessibility in mind and meet ADA requirements.

C-ID (Admin Only)

C-ID ENGR 120

Catalog Description

This course teaches computer programming using MATLAB's language, focusing on syntax, control, and data structures to solve engineering problems. It covers numerical methods like root-finding and interpolation, along with data analysis and visualization for scientific and engineering use. The course also includes linear and nonlinear optimization techniques.

Requisites

Prerequisite(s)

Prerequisite(s) (must be taken before)
MATH-211 (with a grade of C or better).

Codes

TOP Code (CB 03)

0901.00 - Engineering, General (requires Calculus) (Transfer)

CIP Code
14.0102 - Pre-Engineering.

Repeatability Code
No

Grading Option
Letter grade OR P/NP

Learning Objectives

Learning Objectives

Learning Objective
1. Understand the principles of structured programming, including the roles of variables, data types, and control structures.
2. Understand the key concepts of control structures like if, for, and while loops in MATLAB programming.
3. Apply numerical methods for solving ordinary differential equations (ODEs) in engineering problems.
4. Analyze the efficiency and limitations of different root-finding techniques such as Bisection and Newton-Raphson methods.
5. Create customized plots and data visualizations to better convey scientific information.
6. Analyze the contributions of specialized MATLAB toolboxes in solving domain-specific engineering problems.
7. Apply best practices for code commenting and documentation to improve maintainability.
8. Evaluate the computational efficiency of different numerical methods for solving ordinary differential equations in MATLAB.
9. Create a MATLAB program that employs numerical methods like Euler's method or Runge-Kutta techniques to solve and visualize ordinary differential equations in engineering applications.
10. Recognize that diversity, equity, and inclusion in engineering is an ongoing journey. Stay updated with emerging research, technologies, and best practices that promote inclusivity in engineering solutions and in the broader professional environment.

Learning Outcome Information

Course Learning Outcomes

Learning Outcome
1. Students will demonstrate proficiency in MATLAB programming by efficiently writing, debugging, and executing scripts and functions to solve complex engineering problems.
2. Students will apply mathematical concepts, such as calculus, linear algebra, and numerical analysis, using MATLAB to model, analyze, and simulate real-world engineering systems and processes.
3. Students will effectively utilize MATLAB's built-in functions and specialized toolboxes to create comprehensive and visually appealing plots, optimize designs, process signals, and tackle various engineering challenges.
4. Students will effectively communicate their engineering solutions and MATLAB-based analyses through well-organized reports and presentations, demonstrating a clear understanding of the problem, methodology, results, and conclusions.

Program Learning Outcomes

Learning Outcome
Students will demonstrate technical skills used in the industry for engineering technology disciplines such as advanced manufacturing, civil and architectural, and electronics.
Students will use the principles of engineering design to develop a solution, device, or process that resolves or improve an existing solution.
Students will solve technical problems in engineering technology using a combination of mathematics, computer software, scientific principles, and relevant skills.

Institutional Learning Outcomes

Communication: The student will effectively express and exchange ideas through listening, speaking, reading, writing, visual works, and other modes of interpersonal expression.
Critical Thinking: The student will analyze problems, gather and synthesize relevant information, evaluate ideas, information, and evaluate alternative points of view to create, innovate and implement effective solutions.

Information and Technology Literacy: The student will access, interpret, evaluate, and apply relevant information sources and digital media effectively, and in an ethical and legal manner.
Scientific Awareness: The student will apply the scientific method of inquiry to assess potential solutions for real-life challenges by employing science-based knowledge and methodologies in daily life.

Content

Course Lecture Content

1. Introduction to MATLAB and Course Overview
   a. MATLAB environment and interface
   b. Basic operations and commands
   c. Variables and data types
   d. Vectors and matrices
   e. Importance of diversity, equity, and inclusion in engineering
2. MATLAB Programming Fundamentals
   a. Control structures (if, for, while)
   b. Functions and scripts
   c. Debugging and error handling
   d. File input and output
3. Advanced Programming Concepts
   a. Recursion
   b. Data structures (cell arrays, structures, and classes)
   c. Error and exception handling
   d. Code organization and modularity
4. Plotting and Visualization
   a. Basic plotting commands
   b. Advanced plotting techniques
   c. 2D and 3D visualization
   d. Customizing plots and figures
5. Calculus in MATLAB: Limits, Differentiation, and Integration
   a. Limits and continuity
   b. Differentiation
   c. Integration
   d. Multivariable calculus
6. Calculus in MATLAB: Ordinary Differential Equations (ODEs)
   a. ODE theory and classification
   b. Numerical solution techniques
   c. Real-world applications and DEI considerations:
      i. Designing and optimizing renewable energy systems to promote energy equity. ODEs can be used to model and simulate the performance of renewable energy technologies, such as solar panels and wind turbines. By optimizing these systems, engineers can make clean energy more accessible and affordable, particularly for underprivileged communities that may be disproportionately affected by energy poverty or pollution from conventional energy sources.
7. Linear Algebra and Matrix Computations
   a. Matrix operations and properties
   b. Systems of linear equations
   c. Eigenvalues and eigenvectors
   d. Matrix factorizations
   e. Real-world application and DEI considerations:
      i. E.G. Designing inclusive infrastructure and transportation systems. Linear algebra techniques can be applied to model and analyze the efficiency and accessibility of transportation networks, ensuring that they cater to people with diverse abilities, socio-economic backgrounds, and mobility needs.
8. Numerical Analysis Techniques: Root-finding and Interpolation
   a. Root-finding methods (Bisection, Newton-Raphson)
   b. Interpolation and curve fitting (Lagrange, Splines)
   c. Real-world engineering applications and DEI considerations:
      i. DEI Connection: Modeling the environmental impact of engineering projects on underrepresented communities. Numerical analysis techniques can be used to simulate the effects of pollution, water resource allocation, and other environmental factors, helping to minimize negative consequences for marginalized populations.

9. Numerical Analysis Techniques: Differentiation, Integration, and ODEs
    a. Numerical differentiation and integration
    b. Numerical solutions to ODEs (Euler, Runge-Kutta)
        i. Connect content to African American Computer Katherine G. Johnson
10. Real-world engineering applications and DEI considerations
11. Optimization and Nonlinear Equations
    a. Unconstrained and constrained optimization
    b. Linear and nonlinear programming
    c. Multi-objective optimization
    d. Real-world engineering applications and DEI considerations:
        i. E.G. Developing fair algorithms for hiring and promotion in engineering firms. Optimization techniques can be employed to create unbiased algorithms that evaluate job candidates and employees fairly, ensuring that underrepresented groups have equal opportunities for career advancement.
12. Signal Processing and Data Analysis
    a. Discrete Fourier Transform (DFT) and Fast Fourier Transform (FFT)
    b. Digital filters (FIR, IIR)
    c. Time-frequency analysis
    d. Statistical analysis and hypothesis testing
    e. Real-world engineering applications and DEI considerations:
        i. E.G. DEI Connection: Analyzing speech and language patterns to develop more inclusive voice recognition systems. Signal processing techniques can help improve the performance of voice recognition technology to better understand and respond to diverse accents and dialects, promoting equity in access to technology.
13. Modeling, Simulation, and Control
    a. Model development and validation
    b. Parameter estimation and sensitivity analysis
    c. Monte Carlo simulation
    d. System dynamics and control
14. Specialized MATLAB Toolboxes (continued)
    a. Application-specific toolboxes (Control System, Signal Processing, Optimization, Symbolic Math)
    b. Domain-specific toolboxes (Image Processing, Neural Network, Computer Vision, Robotics)
    c. Integrating toolboxes into engineering projects and DEI considerations
15. Best Practices and Advanced Techniques
    a. Code optimization and vectorization
    b. Parallel and distributed computing
    c. Integration with other programming languages (C, C++, Python)
    d. Simulink and Model-Based Design
    e. Developing custom toolboxes and functions
16. Final Project, Course Wrap-Up, and DEI Reflections
    a. Project proposal and planning
    b. Implementation and testing
    c. Documentation and presentation
    d. Project evaluation and feedback
    e. Reflection on DEI in engineering programming and its impact on the course

Course Lab Content

1. MATLAB Basics and Interface
    a. Navigating the MATLAB environment
    b. Creating and manipulating variables
    c. Working with vectors and matrices
2. Basic Programming Concepts
    a. Writing and executing simple MATLAB scripts
    b. Implementing if statements
    c. Using for loops in calculations
3. Advanced Programming Concepts
    a. Implementing while loops and switch-case statements
    b. Writing and using custom functions
    c. File input and output operations

4. Plotting and Visualization
    a. Creating basic 2D and 3D plots
    b. Customizing plot appearance and annotations
    c. Saving and exporting plot images
5. Calculus Applications in MATLAB
    a. Computing limits, derivatives, and integrals
    b. Solving ordinary differential equations
    c. Visualizing multivariable functions
6. Linear Algebra and Matrix Computations
    a. Matrix operations and properties
    b. Solving systems of linear equations
    c. Eigenvalue and eigenvector calculations
7. Numerical Analysis: Root-finding and Interpolation
    a. Implementing root-finding algorithms
    b. Performing polynomial and spline interpolation
    c. Curve fitting and data smoothing
8. Numerical Analysis: Differentiation, Integration, and ODEs
    a. Numerical differentiation and integration techniques
    b. Implementing Euler and Runge-Kutta methods for ODEs
    c. Analyzing and comparing numerical methods
9. Optimization and Nonlinear Equations
    a. Unconstrained and constrained optimization problems
    b. Linear and nonlinear programming applications
    c. Multi-objective optimization case studies
10. Signal Processing and Data Analysis
    a. Discrete Fourier Transform and Fast Fourier Transform applications
    b. Designing and applying digital filters
    c. Time-frequency analysis and spectrograms
11. Modeling, Simulation, and Control
    a. Developing and validating mathematical models
    b. Parameter estimation and sensitivity analysis
    c. Implementing control algorithms
12. Specialized MATLAB Toolboxes
    a. Exploring and using application-specific toolboxes
    b. Integrating toolbox functions into engineering projects
    c. Assessing the benefits of using specialized toolboxes
13. Best Practices and Advanced Techniques
    a. Code optimization and vectorization exercises
    b. Parallel and distributed computing tasks
    c. Integrating MATLAB with other programming languages
14. Final Project
    a. Selecting and planning a final project
    b. Implementing, testing, and refining the project
    c. Presenting and evaluating the final project

Methods of Instruction

Method

In-class Exercises

Integration

In-class activities are strategically designed to align with the Learning Objectives of the course, offering immediate practice and reinforcement of concepts introduced in lectures. These in-class activities vary in form, from short programming tasks aimed at mastering basic MATLAB operations to quizzes designed to assess understanding of advanced programming concepts. On certain occasions, students will engage in group problem-solving activities that apply MATLAB functions and operations to engineering problems. These activities provide a dynamic, hands-on learning experience, enabling students to internalize and apply key concepts and techniques.

DE Adaptations for MOI

In-class activities will be adapted into self-paced online exercises hosted on the course's learning management system. Students will access a curated set of exercises that they can complete either independently or in collaboration with peers through the virtual group functions of the learning management system. To facilitate peer feedback and community learning, discussion boards within the system will be utilized. Here, students can post their solutions or approaches and respond to each other's submissions. Deadlines for completing these exercises will be clearly marked in the course calendar within the learning management system to ensure students remain engaged and on schedule.

Method

Lecture

Integration

Lectures will introduce and elaborate on key concepts, principles of MATLAB syntax, and essential techniques for problem-solving. Each lecture can also incorporate live coding examples to demonstrate the practical applications of MATLAB functions, as well as discussions on best practices and the implications of these skills in real-world engineering and scientific scenarios.

DE Adaptations for MOI

Lectures will be pre-recorded and segmented into manageable, topic-focused videos that can be accessed through the course's learning management system. To encourage active learning and engagement, each video lecture will be supplemented with interactive elements such as embedded quizzes or reflection questions. These elements will be hosted on the same learning management system where the lecture videos are housed. This approach allows students to engage with the material at their own pace, while also providing structured opportunities for assessment and reflection.

Method

Homework

Integration

Homework tasks will be tailored to assess and reinforce students' understanding of MATLAB syntax, programming techniques, and problem-solving skills introduced during lectures and in-class exercises. Assignments may consist of coding exercises, analytical problem-solving, and data analysis projects using MATLAB. Each assignment will be evaluated based on coding efficiency, correctness, and the quality of the analysis provided.

DE Adaptations for MOI

Assignments will be made available and submitted through the course's learning management system, where clear deadlines and submission guidelines will be stated. A dedicated online discussion board will be available for students to post questions, share insights, and seek clarification on homework tasks. Virtual office hours will also be scheduled to provide one-on-one support. When feasible, automated grading tools will be employed to offer timely and constructive feedback, aiding in the learning process.

Method

Lab Activities

Integration

Lab activities are designed to solidify students' understanding of MATLAB programming and its application to engineering problems. These labs will incrementally increase in complexity, starting from basic programming exercises to more advanced simulations and data analytics tasks. The labs will be assessed based on the completeness, accuracy, and ingenuity of the student's solution.

DE Adaptations for MOI

Virtual lab activities will be made accessible via the course's learning management system. Each lab will come with detailed instructions, example code snippets, and necessary datasets for students to work through independently. A discussion board specifically dedicated to lab activities will be established for peer-to-peer interaction, where students can post questions, share

solutions, and provide peer reviews. Virtual lab sessions or office hours will be scheduled for instructor-led guidance and support, ensuring students stay engaged and succeed in mastering the lab objectives.

Method

Projects

Integration

Projects serve as a capstone experience, where students are required to synthesize the concepts, techniques, and skills acquired throughout the course to address real-world engineering challenges. These long-term assignments, whether individual or group-based, will be assessed on the quality of the MATLAB code, the efficacy of the solution, as well as the clarity and depth of accompanying documentation or presentation.

DE Adaptations for MOI

Assignments and project expectations will be thoroughly detailed in the course's learning management system, including deadlines and required deliverables. A repository of supporting resources like tutorials and exemplar projects will be available to guide students. A dedicated discussion board will facilitate ongoing communication, and students will be encouraged to post regular progress updates or questions. Scheduled virtual meetings or presentations will serve as checkpoints where students can share their work-in-progress, allowing for timely feedback and adjustments.

Methods of Evaluation

Method

Homework

Integration

Homework assignments are designed to assess students' ability to apply MATLAB programming techniques, problem-solving skills, and data analysis methods. These assignments will primarily focus on hands-on coding exercises.

Example Standard for Evaluation:
Assignments will be evaluated based on several criteria, including but not limited to:
-Code Correctness: Does the program execute without errors and produce the expected output?
-Code Efficiency: Are the most efficient algorithms and methods used?
-Problem-solving: Is the chosen approach to solving the problem appropriate and effective?
-Data Analysis: Is the data interpreted and used correctly?
-Code Documentation: Are comments and documentation included to explain the code's functionality?

DE Adaptations for MOE

Clear deadlines and submission guidelines will be set for homework assignments. Online resources and virtual office hours or a discussion board will be available for remote students to ask questions and seek help. Automated grading tools may be utilized to provide prompt, objective feedback on coding exercises.

Method

Exams/Tests

Integration

Exams or tests will be scheduled periodically throughout the course to assess students' understanding of the material and their ability to apply MATLAB concepts and techniques to solve engineering problems.

DE Adaptations for MOE

Design online exams or tests with a mix of multiple-choice, short-answer, and coding questions. Utilize Canvas to administer timed exams with clear start and end times. Employ automated grading tools and plagiarism detection software to ensure academic integrity and provide timely feedback.

Method

Projects

Integration

Projects will be assigned to students, either individually or in groups, to apply their knowledge to real-world engineering problems. Projects can span multiple weeks, allowing students to develop, test, and refine their MATLAB-based solutions.

DE Adaptations for MOE

Clearly outline project expectations, deadlines, and deliverables. Provide resources, such as tutorials and examples, to help students get started. Encourage regular progress updates and communication through online discussion boards or group collaboration tools. Schedule virtual check-ins or presentations for students to share their progress and receive feedback.

Method

Class Work

Integration

Class work will include in-class exercises, group problem-solving activities, and participation in discussions. These activities allow students to actively engage with the course material and apply their learning in real-time.

DE Adaptations for MOE

Design asynchronous class work in the form of self-paced activities, such as online quizzes, interactive tutorials, or peer-reviewed coding exercises. Encourage students to participate in online discussion boards, where they can ask questions, share insights, and collaborate with their peers. Track and assess student participation to ensure engagement and understanding of the course content.

Assignments

Assignment Type

Writing

In-class and/or outside of class

In-class
Outside of class

Formative or Summative

Summative

Assignment

Project Assignment: Rocket Trajectory Plotting and Visualization

Objective:
The goal of this project is to create a MATLAB script that models the dynamics of a rocket and plots its trajectory using the provided function. Students will then analyze and explain the generated plot in a brief report.

Instructions:

1. Use the provided rocket dynamics function to model the rocket's motion:

MATLAB Code Given:

```
function [x_dot, y_dot, v_x_dot, v_y_dot] = rocket_dynamics(t, x, y, v_x, v_y, m_rocket, F_thrust, g, k_drag)
%ROCKET_DYNAMICS Returns the time derivatives of position and velocity for a rocket
% t: time (s)
% x, y: position (m)
% v_x, v_y: velocity (m/s)
% m_rocket: rocket mass (kg)
% F_thrust: rocket thrust force (N)
% g: gravitational acceleration (m/s^2)
```

```
% k_drag: drag coefficient

% Calculate drag force
F_drag_x = -k_drag * v_x * sqrt(v_x^2 + v_y^2);
F_drag_y = -k_drag * v_y * sqrt(v_x^2 + v_y^2);

% Calculate net force
F_net_x = F_thrust + F_drag_x;
F_net_y = -m_rocket * g + F_drag_y;

% Calculate acceleration
a_x = F_net_x / m_rocket;
a_y = F_net_y / m_rocket;

% Return time derivatives
x_dot = v_x;
y_dot = v_y;
v_x_dot = a_x;
v_y_dot = a_y;
end
```

---

2. Implement a MATLAB script that integrates the rocket dynamics function over time to compute the rocket's trajectory. You can use any suitable numerical integration method, such as ode45 or ode23.

3. Plot the rocket's trajectory in a 2D graph, with the x-axis representing horizontal distance and the y-axis representing altitude. Customize the plot's appearance (e.g., line style, markers, colors) for improved clarity and readability.

4. Write a brief report (1-2 pages) explaining the rocket's trajectory plot. Your report should include:

- A brief description of the rocket dynamics function and its parameters.
- An explanation of the numerical integration method used to compute the rocket's trajectory.
- A detailed analysis of the rocket's trajectory, including any key features or patterns observed in the plot (e.g., maximum altitude, range, flight time).
- Discussion of any potential improvements or modifications to the rocket's design or flight path to optimize its performance.

Submission:
Submit your MATLAB script, the generated rocket trajectory plot, and your written report to Canvas by the specified deadline.

Grading Criteria:
Your project will be graded based on the following criteria:

Correct implementation of the rocket dynamics function and numerical integration method.
Quality and clarity of the rocket trajectory plot.
Depth and accuracy of the analysis provided in the written report.
Overall organization and presentation of the submitted materials.

How assignment will be adapted in online format

For distance education, all project instructions, code templates, and deadlines will be clearly outlined in the course's learning management system. Students can collaborate on the project virtually, and they're encouraged to share progress updates or seek clarification through a dedicated discussion board. Virtual check-ins or presentations will be scheduled to ensure students are on the right track and to provide real-time feedback.

Assignment Type

Other

In-class and/or outside of class

In-class

Formative or Summative

Formative

Assignment

This is a short activity designed to gauge a student's understanding of numerical integration:

Objective:
The purpose of this formative assessment is to gauge students' understanding of numerical integration concepts and their ability to implement numerical integration techniques using MATLAB.

Instructions:

1. Implement the following mathematical function in MATLAB:

```
function y = func(x)
%FUNC Returns the value of the mathematical function for a given input x
% x: Input value

y = exp(-x^2);
end
```
---

2. Use the trapezoidal rule to approximate the integral of the given function func over the interval [0, 2]. Write a MATLAB script that computes the integral using the trapezoidal rule with N equally spaced subintervals. Test your implementation with N = 10 and N = 100. Report your results and comment on the differences in the approximation values.

3. Implement the same integral using MATLAB's built-in integral function. Compare your trapezoidal rule approximations with the result obtained from the integral function. Discuss the advantages and limitations of each method.

Submission:
Submit your MATLAB script and a brief written response discussing your results and observations in Canvasl.

Grading Criteria:
This formative assessment will be graded based on the following criteria:

- Correct implementation of the trapezoidal rule for numerical integration.
- Proper use of MATLAB's built-in integral function.
- Quality of the written response, including the comparison of methods and discussion of advantages and limitations.

Feedback:

Feedback will be provided on the submitted work, with an emphasis on identifying areas of improvement or misconceptions to help guide students' learning and understanding of numerical integration concepts.

How assignment will be adapted in online format

For the distance education version of this formative assessment, students will be provided with all instructions, function templates, and deadlines via the course's learning management system. Instead of submitting the work in a physical class setting, students will upload their MATLAB scripts and written responses. Students can ask questions and seek clarifications on a specialized discussion board focused on this assignment. Feedback will be returned electronically, allowing for a timely review and understanding of the concepts.

Assignment Type

Reading

In-class and/or outside of class

In-class
Outside of class

Formative or Summative

Formative

Assignment

Exploring MATLAB Help Documentation

Objective:
The purpose of this formative assessment is to gauge students' ability to locate and use MATLAB's help documentation to solve a given problem. This will help students develop essential skills for self-directed learning and problem-solving in MATLAB.

Instructions:

1. Locate the MATLAB help documentation for the fsolve function, which is used to find the roots of a system of nonlinear equations. Read through the documentation to understand the function's syntax, input parameters, and usage.

2. Define the following system of nonlinear equations:

x^2 + y^2 - 1 = 0
x^2 - y^2 - 0.5 = 0

3. Implement a MATLAB script that uses the fsolve function to find the roots of the given system of nonlinear equations. Use the knowledge gained from reading the help documentation to ensure proper usage of the function. Provide an initial guess of [0.5, 0.5] for the solver.

4. Write a brief explanation of the fsolve function and its key parameters, based on your understanding from the help documentation. Discuss how you used the function to find the roots of the given system of nonlinear equations.

Submission:
Submit your MATLAB script and a brief written response explaining your understanding and usage of the fsolve function to Canvas.

Grading Criteria:
This formative assessment will be graded based on the following criteria:

- Correct use of the fsolve function to find the roots of the given system of nonlinear equations.
- Quality of the written response, including the explanation of the fsolve function and its key parameters.
- Demonstrated ability to locate and understand MATLAB help documentation.

Feedback:
Feedback will be provided on the submitted work, with an emphasis on identifying areas of improvement or misconceptions to help guide students' learning and understanding of using MATLAB help documentation effectively.

How assignment will be adapted in online format

For the distance education adaptation, all assignment materials, including problem statements and due dates, will be uploaded to the course's learning management system. Students will submit their MATLAB scripts and brief written responses to a designated submission folder within the platform. A specialized discussion board will be available for students to ask questions, share insights, and seek clarifications about the assignment. All submitted work will receive electronic feedback focused on areas of improvement or misconceptions, allowing for timely comprehension and skill development.

Assignment Type

Reading

In-class and/or outside of class

In-class

Formative or Summative

Formative

Assignment

DEI Reading Assignment: Ethical Considerations in Numerical Modeling for Environmental Impact
Objective:
The purpose of this formative assignment is to foster a deeper understanding of the ethical dimensions of numerical modeling, especially as it pertains to environmental impacts on marginalized communities. Students will read a selected article and engage in an in-class discussion to reflect on the ethical considerations discussed.

Reading Component:
Read the assigned article on the ethical considerations when using numerical methods to model the environmental impact of engineering projects on marginalized communities.

https://www.epa.gov/newsreleases/epa-report-shows-disproportionate-impacts-climate-change-socially-vulnerable

Preparation for In-Class Discussion:
Take notes on key points made in the article, particularly those related to fairness and the impact on underrepresented communities.
Prepare at least two questions or discussion points to bring up during the in-class discussion.

In-Class Discussion:
Participate actively in the in-class discussion about the article, focusing on its ethical implications and fairness criteria. Listen carefully to peers' insights and offer your own perspectives, supporting them with evidence from the article or your own reasoning.

Submission:
No formal submission required; participation in the in-class discussion is essential.

Grading Criteria:
This formative assignment will be graded based on the following criteria:

Preparedness for the in-class discussion, evidenced by thoughtful questions or discussion points.
Active and respectful participation in the discussion.
Depth of engagement with the ethical and DEI considerations in the article.
Feedback:
Feedback will be provided during the in-class discussion, emphasizing the quality and depth of students' engagement with the ethical and DEI aspects of the reading.

How assignment will be adapted in online format

In a Distance Education environment, the in-class discussion on the ethical considerations in numerical modeling for environmental impact will be adapted to an asynchronous format using Canvas's discussion boards. Students will be required to post their reflections and responses to the reading on the designated discussion board by a certain deadline. They will also be expected to engage with their peers by responding to at least two classmates' posts, facilitating a rich, asynchronous discussion on the topic. This allows for the same level of engagement and learning outcomes, while accommodating the needs of distance learners.

Course Materials

Textbooks

MATLAB for Engineering Applications, 5th Edition (2023)
ISBN10: 1264144040 | ISBN13: 9781264144044

Matlab : A Practical Introduction To Programming And Problem Solving (2017)
ISBN: 9780128154793

Other Resources

Optional: License of MATLAB + Simulink Student Suite

Class Size Information

Class Size

32

Class Size Category

G - Large lab 32

Diversity and Inclusion

Explain how diversity and inclusion(e.g., gender, culture/race, sexuality, etc.) is infused into the course.

The course aims to promote diversity, equity, and inclusion in engineering, as stated in section 1e of the Introduction to MATLAB and Course Overview. This is at the beginning of all the engineering courses that are currently being designed/revised as it has been shown time and again that diverse engineering teams are more innovative.

Real-world engineering applications and DEI considerations are integrated throughout the course, as seen in section 6c on Calculus in MATLAB and section 7e on Linear Algebra and Matrix Computations.

The course also incorporates the achievements of African American computer scientist Katherine G. Johnson, connecting content on numerical solutions to ODEs to her contributions to space exploration, as highlighted in section 9b on Numerical Analysis Techniques: Differentiation, Integration, and ODEs.

Describe how assignments, instruction, evaluation, course content, and interactions are contextualized for diverse learners promoting an equitable course.

Assignments are designed to be accessible to learners with diverse backgrounds and experiences. For example, in the calculus section (Section 5), learners may be given real-world problems to solve using calculus concepts, such as modeling the spread of a disease or optimizing a renewable energy system. These problems are contextualized in ways that are relevant to diverse learners and their communities. The assignments have also been designed to allow for multiple approaches to problem-solving and encourage creativity and exploration of different methods.

Instruction is designed to be inclusive and accessible to learners with different learning styles and needs. For example, the course includes a variety of instruction methods, such as video lectures, written materials, and interactive MATLAB exercises. The course also provides opportunities for learners to work in groups and collaborate with others, which can help to build a sense of community and support among diverse learners.

Evaluation criteria will be clearly communicated and designed to allow for diverse perspectives and approaches to problem-solving.

Interactions

Explain how the course will incorporate student-to-student interaction.

This course incorporates student-to-student interactions through group projects and class discussions.

Group projects provide opportunities for students to work collaboratively, share ideas, and learn from each other. The course will also have discussions (in person, or online through Canvas) where students can ask and answer questions, share their experiences, and provide feedback to each other.

Additionally, this course requires peer review of assignments, which is an industry-standard practice, allowing students to provide constructive criticism and learn from each other's work.

Explain how the course will incorporate instructor-to-student interaction.

The course will incorporate several instructor-to-student interactions to support students in their MATLAB programming skills. These interactions will include live lectures, office hours, and one-on-one consultations to provide students with personalized feedback and guidance. In addition, the instructor will facilitate online discussion forums and group projects, where students can collaborate, share ideas, and provide feedback to each other. The instructor will also provide regular feedback on students' assignments and projects, offering constructive criticism and suggestions for improvement. These interactions will ensure that students receive the support they need to succeed in the course and develop their MATLAB programming skills.

Explain how the course will incorporate student-to-content interaction.

The course will incorporate student-to-content interaction through various means. For example, students will have access to interactive MATLAB tutorials, code samples, and examples that they can engage with on their own time. These resources will be

designed to provide a comprehensive understanding of the course content and allow students to practice what they have learned. Additionally, there will be assignments that require students to apply their knowledge to specific problems, encouraging them to engage with the content on a deeper level. The course will also feature quizzes and assessments that give students the opportunity to evaluate their understanding of the material and receive feedback from the instructor.

Explain how the course will incorporate student-to-self interaction (metacognitive).

Student-to-self interaction in this course refers to the ability of students to reflect on their own learning and programming processes. This course will encourage metacognitive practices through regular self-assessment exercises and opportunities for self-reflection. For example, students may be asked to reflect on their problem-solving strategies and evaluate their effectiveness in tackling programming challenges. They will also be encouraged to seek out additional resources or practice problems on their own to reinforce their learning. Additionally, the course will provide opportunities for students to set their own programming goals and track their progress toward achieving them. Through these activities, students can develop a deeper understanding of their own learning process and become more effective and efficient programmers.

Key: 3041