Control Theory Intro: Home Assignment #1

October 15, 2021

Yogev Hadadi

Introduction

The purpose of this home assignment is to be familiar and remove rust from the student’s MATLAB capabilities.

You should submit a **.m** file. The first line should print your ID.

>> disp(’ID_STUDENT_1 ID_STUDENT_2’)

For clarity of the script, you can separate the different sections of the script with a **%%**. This will automatically create a block in your script. In order to run specifically this block of code press ’Ctrl+Enter’. To run the entire script press ’F5’.

1 Scalar arithmetic

Calculate the following

1. 3*2^4
2. (3*2)^4 % parentheses have highest priority
3. 3-2^4
4. 3^4-3
5. 8/2^4
6. 2^4\8 % same as previous! two different divisions, \ and /
7. 8^4/2

2 Vector

1. x = [3 4 7 11] % create a row vector (spaces)
2. x = 3:8 % colon generates list; default stride 1
3. x = 8:-1:0 % hstarti : hstridei : hstopi specifies list
4. xx = [8 7 6 5 4 3 2 1 0]; % same as last; semicolon suppresses output
5. xx % display contents
6. x = linspace(0,1,11) % generate vector automatically
7. x = 0:0.1:1 % same thing
8. y = linspace(0,1); % note semicolon!
9. length(x)

10. length(y)
11. size(x)
12. size(y)
13. y(3) % access single element
14. y(1:12) % access first twelve elements
15. y([3 6 9 12]) % access values specified in a vector!
16. x' % transpose
17. z = [1+2j 4-3j]
18. z'
19. z.' % note difference in transposes!
20. 3*[1 2 5] % factor replicated, multiplies each element

3 Matrix arithmetic

Assume,

$$A = \begin{bmatrix} 3 & 1 & 5 \\ 7 & 4 & 2 \\ 7 & 6 & 8 \end{bmatrix}, b = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, C = [1, 0, 0] \tag{1}$$

Calculate the following expressions:

1. A^{-1} - Inverse of A
2. A^T - Transpose of A
3. Ab
4. $b^T A$
5. $A^T b$
6. $Ct = \begin{bmatrix} b & Ab & A^2 b \end{bmatrix}$
7. $Ot = \begin{bmatrix} C \\ CA \\ CA^2 \end{bmatrix}$
8. e^A
9. $rank(Ot)$
10. $rank(Ct)$

11. Eigen-values of A
12. Eigen-vectors of A
13. Eigen-values of Ct
14. Eigen-vectors of Ct
15. Eigen-values of Ot
16. Eigen-vectors of Ot
17. Piecewise multiplication of Ot and Ct

4 Functions

4.1 Orthogonal matrix

Write a MATLAB function that given a squared matrix, A, the function returns '1' if the matrix is an orthogonal matrix, and '0' otherwise.

A matrix is orthogonal if its transpose is equal to its inverse.

4.2 Replace values in matrix

Given a matrix, A, a value, u and an additional value v, return a matrix that have values v in all indices that had a value of u in the original matrix.

For example, for $u = 1$, $v = 2$ and $A = \begin{bmatrix} 1 & 2 \\ 3 & 1 \end{bmatrix}$ a matrix $B = \begin{bmatrix} 2 & 2 \\ 3 & 2 \end{bmatrix}$ is returned.

5 Polynomials

Given the following polynomial,

$$f(x) = x^6 - 12x^4 + 39x^2 + 2x - 28 \tag{2}$$

1. Plot a graph of the function $f(x)$ for $x \in [-3, 3]$ in blue.
2. Find the roots of the function $f(x)$.
3. Find all of the minimas of the function $f(x)$ in the range $xin[-3, 3]$.
4. Plot, on the same graph plotted in 5.1, the minimas found in 5.3 in red 'x'.

6 System impulse response

Consider the following differential equation:

$$\ddot{x} = -5x - 2\dot{x} + u. \tag{3}$$

Assume the sampling time is $1000Hz$.

Plot the system response for a step input, $u(t) = 1\,\forall t > 0$, (initial conditions are $x = \dot{x} = 0$) in three different ways.

6.1 Iterative approach (for-loop)

1. Write a scripnt that iterates through time and calculates for each timestep the system states, namely, x, $\dot{x}$, and, $\ddot{x}$.
2. Plot the response in solid blue `plot(t,x,'b')`.

6.2 Built-in lsim command

1. Formulate the transfer function of the system in the Laplace domain ($G(s)$).
2. Use the `tf (...)` command to define the system as a variable.
3. Create a step signal.
4. Use the `lsim (...)` command to simulate the system's response.
5. Plot the response in dashed red on the same figure in 6.1 `plot(t,x_lsim,'r--')`.

6.3 Built-in step command

1. Formulate the transfer function of the system in the Laplace domain ($G(s)$).
2. Use the `tf (...)` command to define the system as a variable.
3. Use the `step(g)` command to simulate the system's step-response.
4. Plot the response in dot-dashed greed on the same figure in 6.1 `plot(t,x_step,'g.-')`.