6.3100: Intro to Matlab™

Basic Syntax

```
% DT Transfer functions
dT = 1.0e-3;
z = tf([1 0], 1, dT); % Define z as the transform variable
R = 1/z;

% CT Transfer functions
s = tf('s')

K = 50; % some controller transfer function
H = 1000/((s+1)*(s+10)*(s+100)); % some plant transfer functions
Hc = 10*(s+20)/(s+200)

% some open loop transfer function (no feedback, just forward pa
G = K*H*Hc



% Find the closed loop transfer function through Black's Formula
% Note: only use if the control block diagram is applicable witl
G_cl = feedback(G, 1)
```

```
% Plot the step response of a transfer function for 5 seconds
step(G_cl, 5)
```

Useful Functions

```
% Formatting Settings
sympref('FloatingPointOutput',true) % makes everything print as
format short g % simplifies exponential notation
```

Symbolic Variables:

- This is hugely helpful in solving theoretical problems (and the later prelabs)
- For example, if you need to find the gains K to satisfy some natural frequency requirement, you can have MATLAB solve for the natural frequencies in terms of K. Then you can have MATLAB solve the equation that the natural frequencies are equal to the specified value, and it will return what gains K satisfy this.
- Further reading: https://www.mathworks.com/help/symbolic/create-symbolic-numbers-variables-and-expressions.html
- Example: find stable Kp range for discrete system

```
syms Kp real % create real symbolic variable lambda


DT = 0.01;
lambda = 1 - DT * Kp; % express lambda in terms of Kp
% for one variable equation:
% ans = solve(equation, variable to solve for)
Kp_min = solve(lambda==1, Kp)
Kp_max = solve(lambda==-1, Kp)

% for system of equations:
% solutions = solve([equations], [variables to solve for])
```

Common Mistakes

- using feedback when Black's formula should not be applied

Matlab also has their own guide with some basic functions to get you started. The first page is included for your convenience

- https://www.mathworks.com/content/dam/mathworks/fact-sheet/matlab-basic-functions-reference.pdf

MathWorks®

MATLAB® Basic Functions Reference

MATLAB Environment

`clc`	Clear command window
`help fun`	Display in-line help for `fun`
`doc fun`	Open documentation for `fun`
`load("filename","vars")`	Load variables from `.mat` file
`uiimport("filename")`	Open interactive import tool
`save("filename","vars")`	Save variables to file
`clear item`	Remove items from workspace
`examplescript`	Run the script file named `examplescript`
`format style`	Set output display format
`ver`	Get list of installed toolboxes
`tic, toc`	Start and stop timer
`Ctrl+C`	Abort the current calculation

Operators and Special Characters

`+, -, *, /`	Matrix math operations
`.*, ./`	Array multiplication and division (element-wise operations)
`^, .^`	Matrix and array power
`\`	Left division or linear optimization
`.', '`	Normal and complex conjugate transpose
`==, ~=, <, >, <=, >=`	Relational operators
`&&, ||, ~, xor`	Logical operations (AND, NOT, OR, XOR)
`;`	Suppress output display
`...`	Connect lines (with break)
`% Description`	Comment
`'Hello'`	Definition of a character vector
`"This is a string"`	Definition of a string
`str1 + str2`	Append strings

Special Variables and Constants

`ans`	Most recent answer
`pi`	π=3.141592654...
`i, j, 1i, 1j`	Imaginary unit
`NaN, nan`	Not a number (i.e., division by zero)
`Inf, inf`	Infinity
`eps`	Floating-point relative accuracy

Defining and Changing Array Variables

`a = 5`	Define variable a with value 5

`A = [1 2 3; 4 5 6]`
`A = [1 2 3`
`4 5 6]`
	Define A as a 2x3 matrix
	"space" separates columns
	";" or new line separates rows

`[A,B]`	Concatenate arrays horizontally
`[A;B]`	Concatenate arrays vertically
`x(4) = 7`	Change 4th element of x to 7
`A(1,3) = 5`	Change A(1,3) to 5
`x(5:10)`	Get 5th to 10th elements of x
`x(1:2:end)`	Get every 2nd element of x (1st to last)
`x(x>6)`	List elements greater than 6
`x(x==10)=1`	Change elements using condition
`A(4,:)`	Get 4th row of A
`A(:,3)`	Get 3rd column of A
`A(6, 2:5)`	Get 2nd to 5th element in 6th row of A
`A(:,[1 7])=A(:,[7 1])`	Swap the 1st and 7th column
`a:b`	[a, a+1, a+2, ..., a+n] with a+n≤b
`a:ds:b`	Create regularly spaced vector with spacing ds
`linspace(a,b,n)`	Create vector of n equally spaced values
`logspace(a,b,n)`	Create vector of n logarithmically spaced values
`zeros(m,n)`	Create m x n matrix of zeros
`ones(m,n)`	Create m x n matrix of ones
`eye(n)`	Create a n x n identity matrix
`A=diag(x)`	Create diagonal matrix from vector
`x=diag(A)`	Get diagonal elements of matrix
`meshgrid(x,y)`	Create 2D and 3D grids
`rand(m,n), randi`	Create uniformly distributed random numbers or integers
`randn(m,n)`	Create normally distributed random numbers

Complex Numbers

`i, j, 1i, 1j`	Imaginary unit
`real(z)`	Real part of complex number
`imag(z)`	Imaginary part of complex number
`angle(z)`	Phase angle in radians
`conj(z)`	Element-wise complex conjugate
`isreal(z)`	Determine whether array is real

mathworks.com/help/matlab