Lecture Overview

- Introduction and History
- Matlab architecture
- Operation basics
- Visualization
- Programming
- Other useful information
Introduction and History

- Invented by Cleve Moler in late 1970s to give students access to LINPACK and EISPACK without having to learn Fortran.
- Together with Jack Little and Steve Bangert they founded Mathworks in 1984 and created Matlab.
- The current version is 7.
- Interpreted-code based system in which the fundamental element is a matrix.
- Available for a large number of platforms:
  - Windows, Linux, HP-UX, Mac OS X and Solaris
Architecture

- Currently, the Matlab core contains a mixture of routines in C and Java
- New routines can be obtained by acquiring toolboxes
  - Control systems, signal processing, image processing, neural networks, fuzzy logics, among many others.
- Simulink is one of their major toolboxes
Simulink is a software package for making modeling and simulation easy and *fun*. It is all based on block diagrams.
Architecture (3)

- Toolbox
  - Signal Processing
- Toolbox
  - Control Systems
- SIMULINK

MATLAB
(Code in C & Java)
Operation Basic

- Interface:
  - Workspace
  - Launch Pad
  - Command History or Current Directory
  - Command Window
Operations Basics (2)

- 4 basic variable types:
  - Double precision matrices
  - Strings
  - Cell arrays
  - Structures

- Variable assignment
  - Single value: \( a = 1 + 7 \)
  - Matrix: \( m = \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} \)

\[ a = 8 \]
\[ m = \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} \]
Operations Basics (3)

- Using variables in expressions
  - \( b = \sqrt{a \times 10 + 1}/10 \)
  - \( b = 0.9 \)

- Using functions
  - Single outputs
    - \( y = \sqrt{x} \)
  - Multiple outputs:
    - \([r,p,k] = \text{residue}(\text{num,den})\)
      - Partial-fraction expansion
Example function: Partial-fraction expansion

\[
\frac{B(s)}{A(s)} = \frac{s^4 + 8s^3 + 16s^2 + 9s + 6}{s^3 + 6s^2 + 11s + 6} = s + 2 + \frac{-6}{s + 3} + \frac{-4}{s + 2} + \frac{3}{s + 1}
\]

\[
\begin{align*}
\text{num} &= [1 \ 8 \ 16 \ 9 \ 6] \\
\text{den} &= [1 \ 6 \ 11 \ 6] \\
[r,p,k] &= \text{residue}(\text{num},\text{den})
\end{align*}
\]
Operations Basics – Matrices

\[
\begin{align*}
\text{>> } v &= [1, 2, 3, 4, 5] \\
\text{>> } v' \\
\text{ans} &= 1 \\
&\quad 2 \\
&\quad 3 \\
&\quad 4 \\
&\quad 5
\end{align*}
\]

1x5 matrix

transpose(v) gives 5x1 matrix

\[
\begin{align*}
\text{>> } v(4) \\
\text{ans} &= 4
\end{align*}
\]

4th element
Matrix multiplication

\[
\begin{bmatrix}
1 & 2 \\
3 & 4
\end{bmatrix} \times \begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix}
\]

\[
\text{ans} = \\
1 & 2 \\
3 & 4
\]

Dot multiplication

\[
\begin{bmatrix}
1 & 2 \\
3 & 4
\end{bmatrix} \cdot \begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix}
\]

\[
\text{ans} = \\
1 & 0 \\
0 & 4
\]

Dot division (./), Kronecker tensor product (kron(M,N))
Other important matrix functions:
- Inverse: inv(M)
- Rank: rank(M)
- Determinant: det(M)
- null matrix: M = []
- n by m matrix of zeros: M = zeros(n,m)
- n by m matrix of ones: M = ones(n,m)
- Identity matrix n by n: M = eye(n)
Visualization

- Matlab is great for plotting functions.
  
- \( x = [0:0.01:2\pi]; \)
  
- \( y = \cos(x); \)
  
- \( \text{plot}(x,y) \)

The cosine of a matrix is a matrix of the cos of each x-value.

Do not show me the result on screen.
Add titles to graphs and axis

- `title('this is the title')`
- `ylabel('y')`
- `xlabel('x')`
Visualization (3)

- Multiple plots
  - `plot(x,y1,'b-', x,y2,'r:')`
  - `legend('y1','y2')`
Visualization (4)

- Matlab can also do 3D plots, such as mesh surfaces:
  - \([X,Y] = \text{meshgrid}(-2:.2:2, -2:.2:2);\)
  - \(Z = X.*\exp(-X.^2 - Y.^2);\)
  - \text{mesh}(X,Y,Z)\)
Visualization (5)

- Or simple 3D plots:
  - \( x = -2:.02:2; \)
  - \( y = x; \)
  - \( z = x .* \exp(-x.^2 - y.^2); \)
  - \( \text{Plot3}(x,y,z,).'.' \)
Visualization (6)

- All calls to plot, plot3, mesh, etc. will overwrite on the “current figure”. Make sure that you want to erase the “current figure”.

- To create a new empty figure, use the command `figure`
Adding figures to reports:

- Print the figure directly
- Save it to a TIFF file and add to the report (File → Export...)
- Copy to clipboard and paste to the report (Edit → Copy Figure)
  - The background is copied too! By default it is gray. To change the background color use:
    - `set(gcf,'color','white')`
Programming

- Programming in Matlab is done by creating "m" files.
  - File → New → M-File
  - edit

- Programs can be divided into two categories:
  - Script programs
  - Functions
Programming (2)

- Script M-files
  - Storing a sequence of functions
  - All variables are **global**
  - Call the program by writing the name of the file where it is saved
  - “%” can be used for commenting
Programming (3)

- Functions M-files
  - It has to start declaring the name of the function, its inputs and outputs
    - E.g. `function [a, b, c] = foo(in1, in2)`
  - The file name has to be the same name of the function (`foo.m`)
  - The variables are **local**
  - Call it by calling the function name with inputs and outputs
To create a new script M-file:

Go to FILE -> NEW -> M-file

Save file as filename.m

To run the file - In the Command Window, type filename
Programming (5)

- Matlab accepts if-then-else constructs

```matlab
if <condition>
    <commands>
elseif <condition>
    <commands>
else
    <commands>
end
```
Programming (6)

- For loops

```latex
\begin{verbatim}
for k = <array>
   <statements>
end
\end{verbatim}
```

The commands are executed repeatedly and the variable $k$ is given each value in the loop-vector.
Example of using “if-elseif-then” and “for” command

```matlab
a = randn(1, 5);
for i = 1:length(a)
    if a(i) > 0,
        s(i) = 1;
    elseif a(i) == 0,
        s(i) = 0;
    else
        s(i) = -1;
    end
end
```

```
a = 1.1909 1.1892 -0.0376 0.3273 0.1746
>> s = 1   1   -1   1   1
```
Other important Control Flow functions:

- **if** statement
- **for** statement
- **while** statement
- No **goto** statement!
- **break** exists from **for** and **while** loops
- **switch case otherwise end** combination
- **try catch end** combination
- **end** is to end control statements above
- **return** is used in functions in the same way as Fortran
Other useful information

- To run programs, they have to be in the current active directory or in a directory in the path (File → Set path...)
- `help <function name>` displays the help for the function
  - If you create a function, the first things commented (%) after the function declaration are considered the help information
- `Helpdesk` brings up a GUI for browsing very comprehensive help documents
Other useful information (2)

- `save <filename>` saves everything in the workspace (or the function variable space) to `<filename>.mat`. If no `<filename>` is given, saves to `matlab.mat`.
- `load <filename>` loads the file.