دانشکده روانشناسی و علوم تربیتی

پژوهشگاه دانش‌های بنیادی IPM

MATLAB for Brain and Cognitive Psychology (Programing)

Presented by:

Ehsan Rezayat, Ph.D.

Faculty of Psychology and Education, University of Tehran.

Institute for Research in Fundamental Sciences (IPM), School of Cognitive Sciences,

emails: rezayat@ut.ac.ir, rezayat@ipm.ir, erezayat.er@gmail.com

Working with files

- Saving and loading Matlab variables to and from .mat files does not require any special file handling, just use save() and load()

- However, if you want to read another kind of file, or create a file that can be read outside of Matlab, you must deal with files directly

Working with files

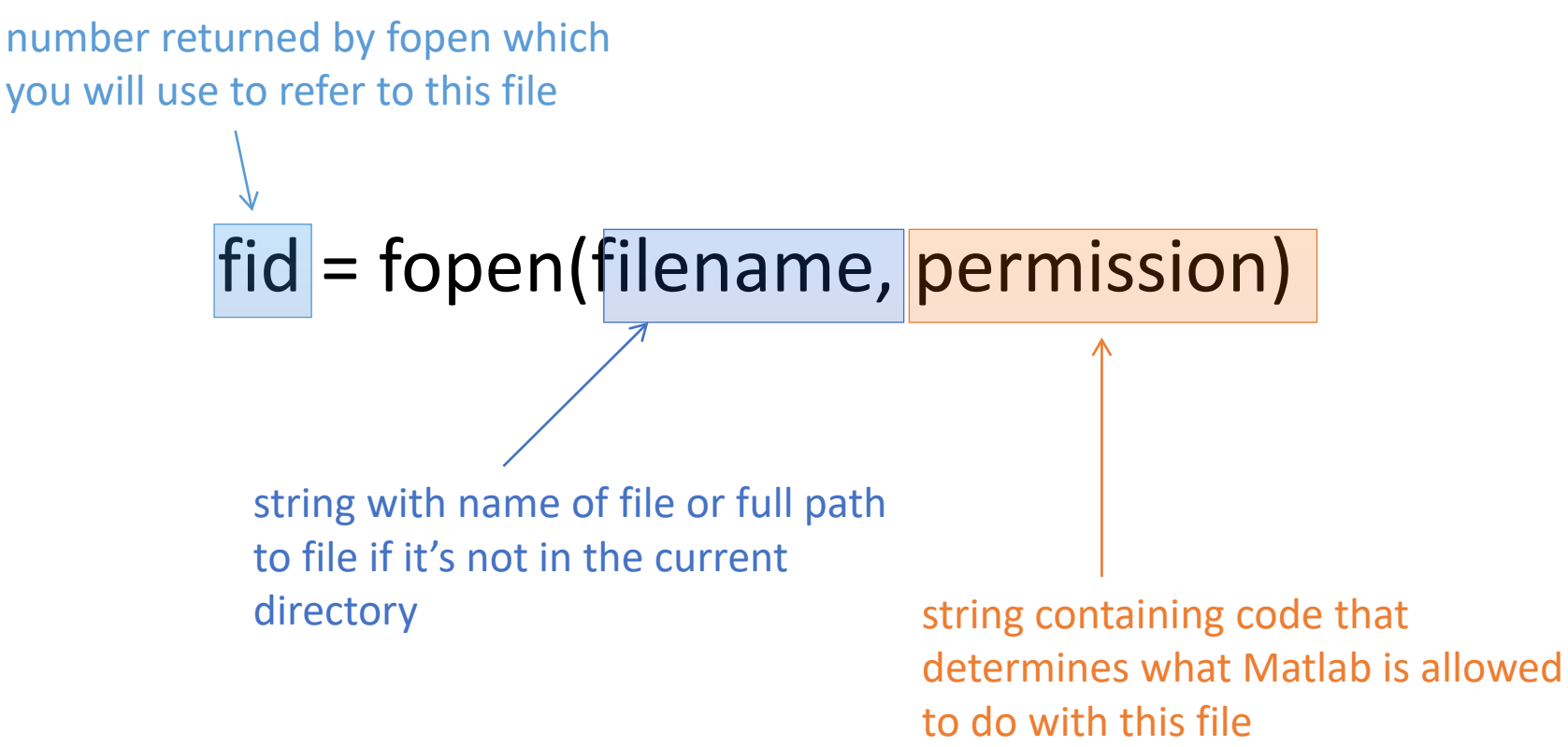
- Introducing fopen() and fclose()
- General plan for working with files:

fopen()

<read from file or write to file>

fclose()

Opening files



Opening files

- Permission codes
  - ‘r’ open file for reading
  - ‘w’ open file for writing (will create or overwrite)
  - ‘a’ append data to file (will create if doesn’t already exist)

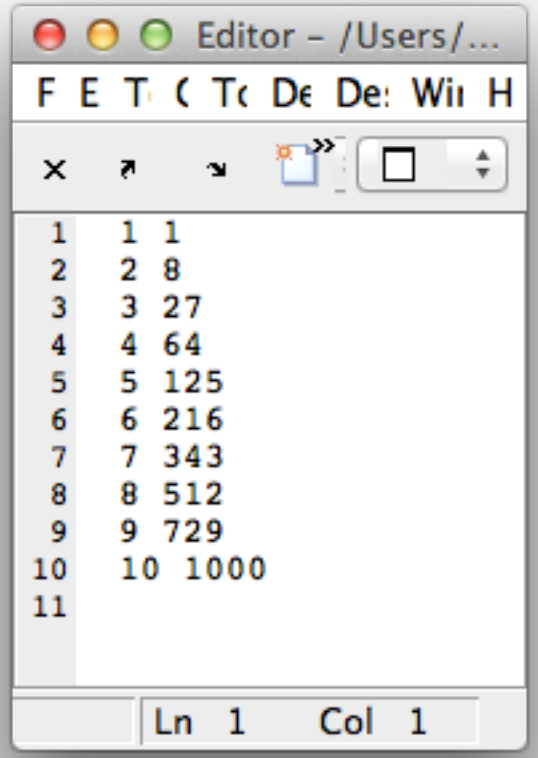
Writing to files

```
>> myFileID = fopen('testfile.txt','w')
myFileID =

     3
>> x = 100;
>> fprintf(myFileID,'X is equal to %d\n',x);
>> fclose(myFileID);


>> fopen('/usr/bin/test.txt','w')

ans =

    -1
```

If fopen() returns -1 then it has failed to open the file

Writing to files

```
>> x = [1:10];
>> y = x .^3;
>> myExponentsFile = fopen('e.txt','w');
>> fprintf(myExponentsFile,'%d %d\n',[x;y]);
>> fclose(myExponentsFile);
```

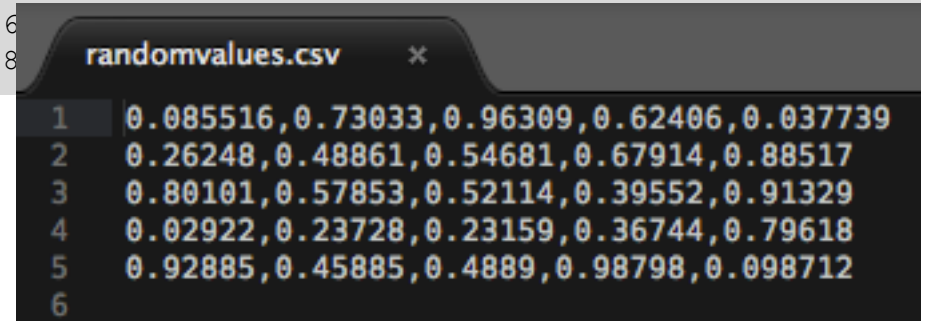


Writing to files

- Other ways to write to files:
  - csvwrite()
  - dlmwrite()

Writing to files

```
>> x = rand(5)

x =

    0.0855    0.7303    0.9631    0.6241    0.0377
    0.2625    0.4886    0.5468    0.6791    0.8852
    0.8010    0.5785    0.5211    0.3955    0.9133
    0.0292    0.2373    0.2316    0.3674    0.7962
    0.9289    0.4588    0.4889    0.9880    0.0987

>> csvwrite('randomvalues.csv',x)
>> clear all
>> x = csvread('randomvalues.csv')
x =

    0.0855    0.7303    0.9631    0.6241    0.0377
    0.2625    0.4886    0.5468    0.6791    0.8852
    0.8010    0.5785    0.5211    0.3955    0.9133
    0.0292    0.2373    0.2316    0.3674    0.796
    0.9288    0.4588    0.4889    0.9880    0.098
```



Reading from text files

- textscan()
- fgetl()
- dlmread()
- csvread()

Reading from text files

```
>> help textscan

textscan Read formatted data from text file or string.
    C = textscan(FID,'FORMAT') reads data from an open text file identified
    by FID into cell array C. Use FOPEN to open the file and obtain FID.
    The FORMAT is a string of conversion specifiers enclosed in single
    quotation marks. The number of specifiers determines the number of
    cells in the cell array C.  For more information, see "Format Options."
```

Reading from text files

Contents of "log.txt":

```
SM01  -0.12 0.29 0.51 -0.15 0.30 0.38
SM02  -0.04 0.04 0.08 -0.13 -0.18 -0.07
SM03  -0.18 0.08 0.35 -0.10 0.08 -0.04
SM04  0.15 0.06 -0.27 -0.58 -0.10 0.22
SM05  0.15 0.22 0.25 0.20 0.09 0.29
SM06  -0.18 -0.63 -0.82 -0.52 -0.28 -0.53
SM07  0.24 0.37 0.22 0.29 0.33 0.53
SM08  0.05 0.37 0.23 -0.09 -0.04 0.07
SM09  0.37 0.54 0.67 0.58 0.74 0.55
```

```
>> logFID = fopen('log.txt');
>> data = textscan(logFID,'%s %f %f %f %f %f %f')
data =

  Columns 1 through 5

    {9x1 cell}    [9x1 double]    [9x1 double]    [9x1 double]    [9x1 double]

  Columns 6 through 7

    [9x1 double]    [9x1 double]
```

Reading from text files

```
>> subjectcodes = data{1}
subjectcodes =

    'SM01'
    'SM02'
    'SM03'
    'SM04'
    'SM05'
    'SM06'
    'SM07'
    'SM08'
    'SM09'

>> fclose(logFID);
>>
```

Saving variables

- save() to save the workspace to disk
- load() to load a .mat file that contains variables from disk
- clear() to remove a variable from memory

Saving variables

```
>> who

Your variables are:

age        employee  mycell    patient   score

>> whos
  Name           Size              Bytes  Class
Attributes

  age            1x1                   8  double
  employee       1x4                   8  char
  mycell         1x2                 304  cell
  patient        1x2                1056  struct
  score          1x1                   8  double
```

Saving variables

```
>> save('matlabclass1')
>> clear
>> who
>>
>> load('matlabclass1')
>> who

Your variables are:

age        employee  mycell    patient   score

>> save('onevar','patient')
>> clear
>> who
>> load('onevar')
>> who

Your variables are:

patient
```

Working with files

```
>> load ch08filename.mat
>> datafile     = load ('ch08filename.mat');
%the ch08filename.mat contains three variables: frex, timevec,and tf_data.

>> data         = cell(5,1);
>> for fi       = 1:5
    data{fi}    = load(['data_rat' num2str(fi) '.mat']);
   end
```

```
>>  files1       = dir('D:/Workshop-psycho/code/supplementary/*rat*.mat');
```

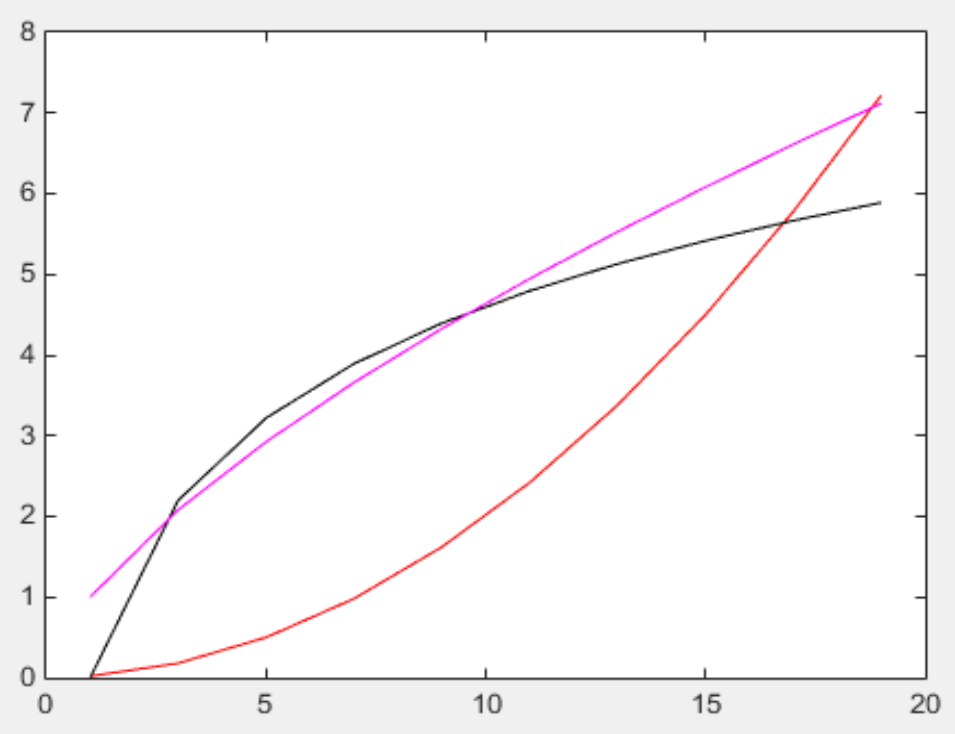
```
>> filedir      = 'D:/Workshop-psycho/code/supplementary/';
>> files        = dir([ filedir '*rat*.mat' ]);
>> data         = cell(size(files));
>> for fi       = 1:length(files)
    data{fi}    = load([ filedir files(fi).name ]);
   end
```

Plotting

- Plot
- Bar
- Errorbar
- Scatter
- Image
- Surf

Plot lines

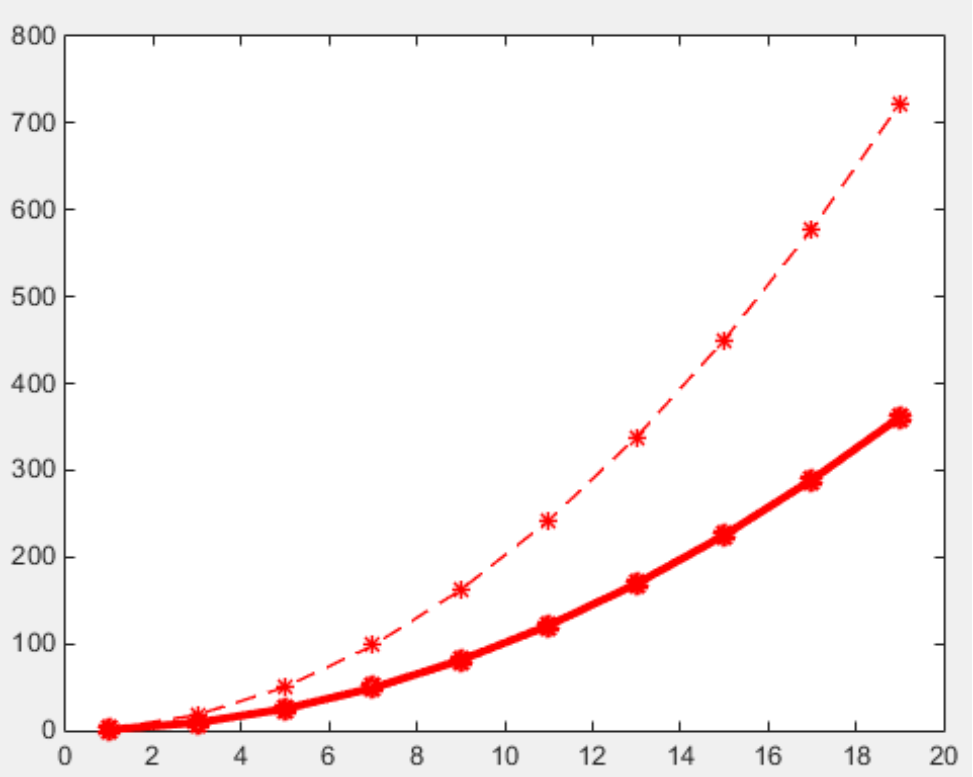
```
>> cla;
>> plot(x,y/50,'r');
>> hold on
>> plot(x,log(y),'k'); % log is the natural log
>> plot(x,y.^(1/3), 'm');
```



Plot lines

linewidth

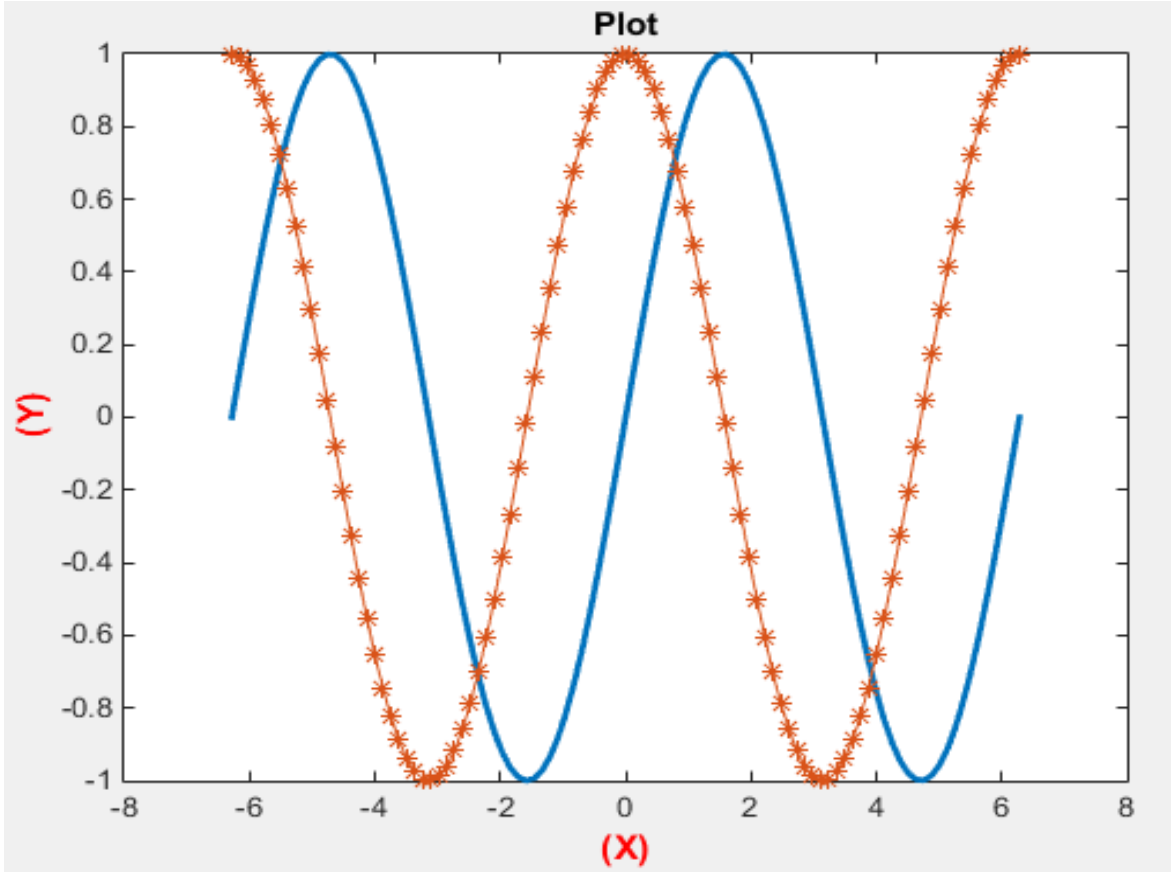
```
>> plot(x,y,'ro-','linewidth',3)
>> hold on
>> plot(x,2*y,'r*--','linewidth',1) % default with is 1
```



Plot lines

X-label , y-label

```
>> x               = linspace(-2*pi,2*pi);
>> y1              = sin(x);
>> y2              = cos(x);
>> p               = plot(x,y1,x,y2);
>> p(1).LineWidth = 2;
>> p(2).Marker     = '*';
>> title('Plot')
>> xlabel('(X)','FontSize',12,...
       'FontWeight','bold','Color','r');

>> ylabel('(Y)','FontSize',12,...
       'FontWeight','bold','Color','r');
```



Bar plot

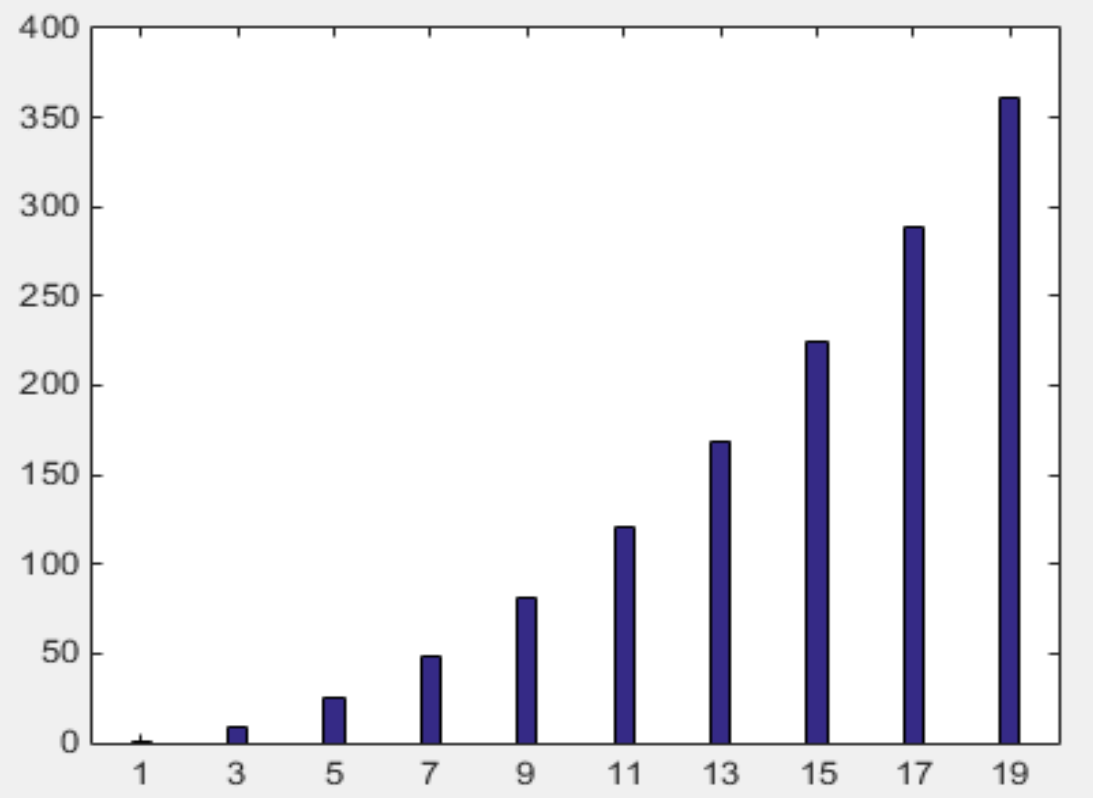
bar(Y)

bar(X,Y)

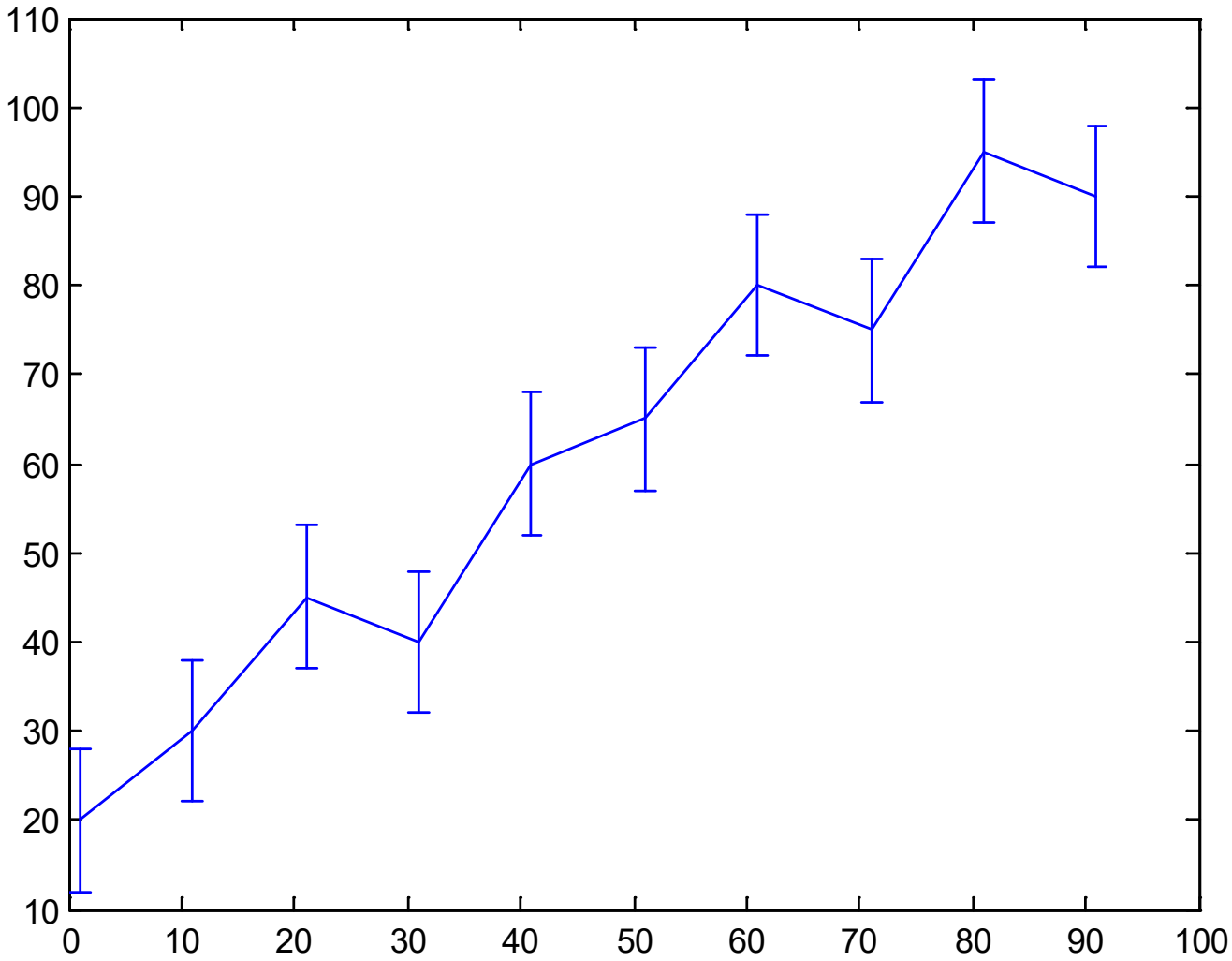
bar(---,width)

bar(---,Name, Value)

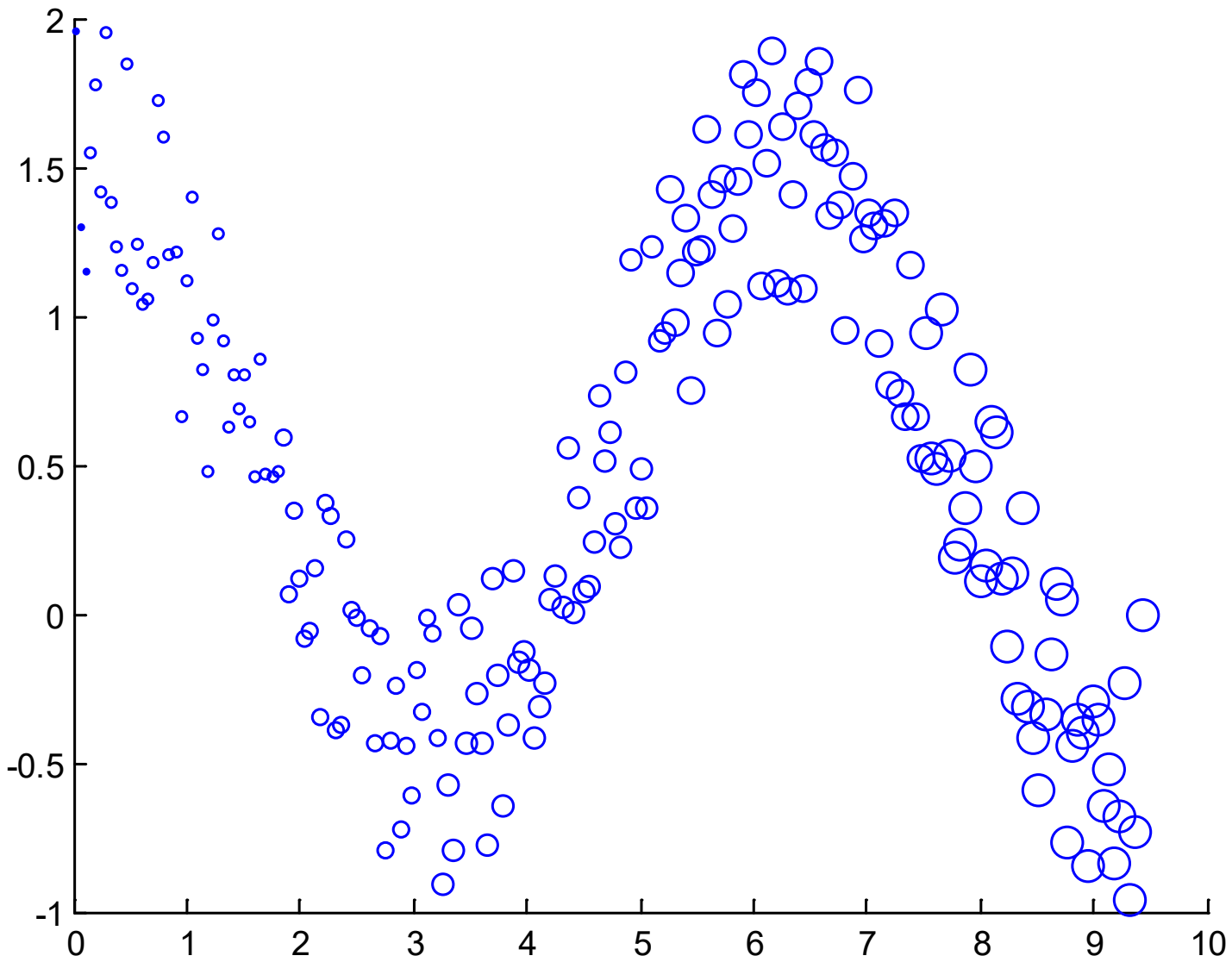
```
>> bar(x,y,.2);
```



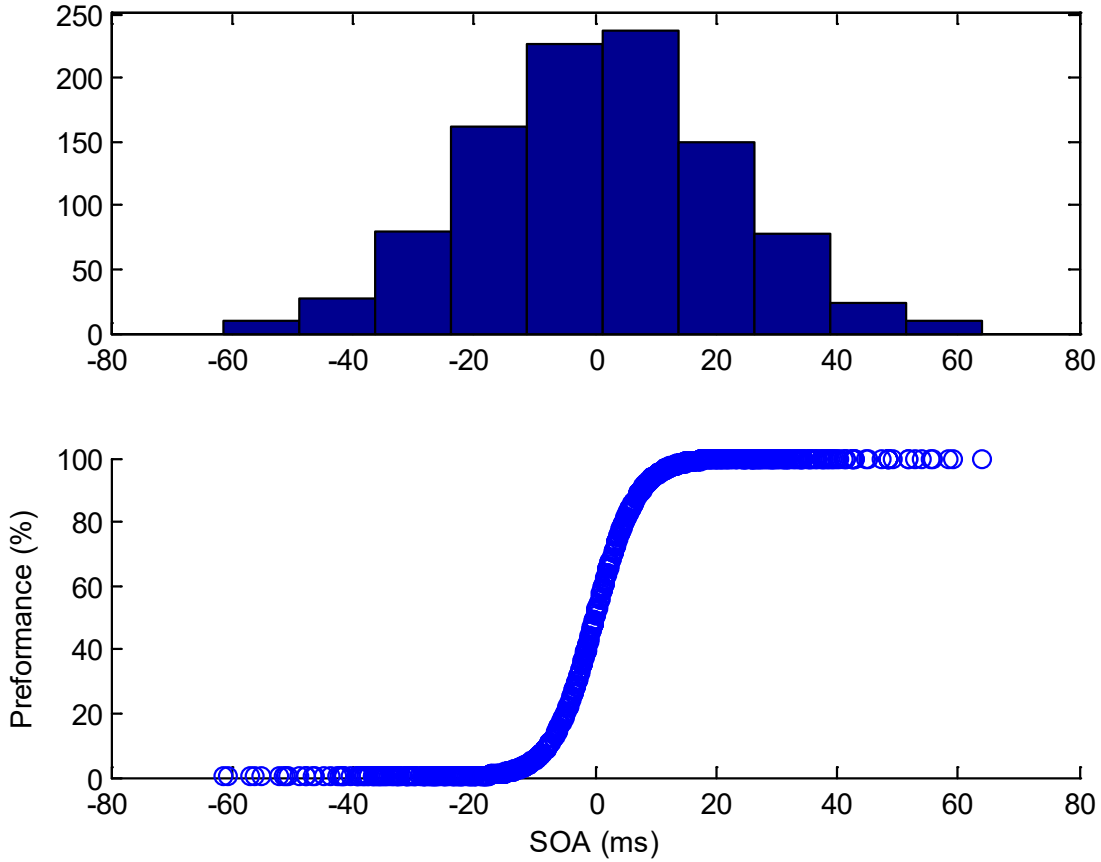
Errorbar



Scatter

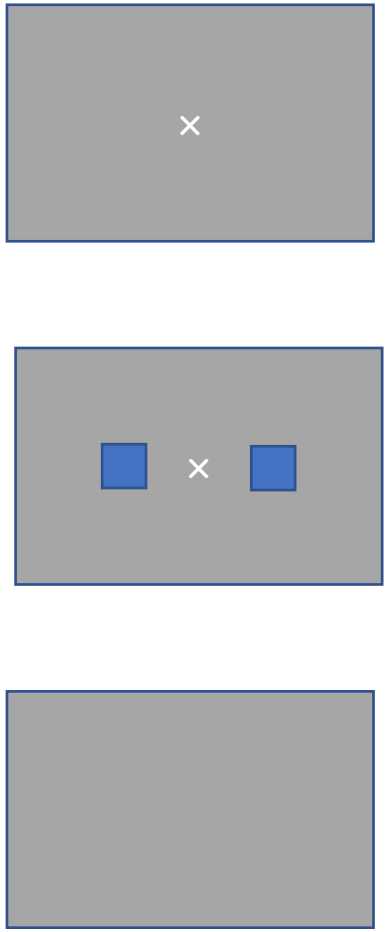


Example



SOA
-50 :2: 50 ms

Saccades
Left or Right



Coding style

- Your code needs to be readable by humans as well as by machines
- Never trust yourself to remember anything. You will always forget. Just because something appears obvious now does not mean it will in the future, or to someone else.
- Use comments to explain what you are doing in English.

Coding style

- In a collaborative laboratory setting your code is not just for you:
  - you need to write to allow other people to update and change your code for their purposes
  - you need to write your code to be as flexible as possible. this means we will expect the code to be transported to other machines, and other environments

Coding style

```
ist= 10;
sst= 4;
r= [1,2];

ist = ist/fr;
sst = sst/fr;
```

```
%set up standard presentation parameters
instructionScreenTime = 10;      %how long the instructions will stay on, in seconds
stimulusScreenTime = 4;          %how long the stimulus will stay on, in seconds
acceptableResponses = [1,2];     %which responses buttons the subject may press

%convert times from seconds into frames
instructionScreenTime = instructionScreenTime/frameRate;
stimulusScreenTime = stimulusScreenTime/frameRate;
```

Coding style

Make your comments *informative*

```
%add two to the instruction screen time
instructionScreenTime = instructionScreenTime + 2;     %here we are adding two
```

```
%add time to the instruction screen time to account for
%the additional time needed by this subject population
instructionScreenTime = instructionScreenTime + 2;
```

Coding style

- Matlab does not enforce spacing and indentation, but you should for code readability
- Keep blocks of code clearly demarcated

Coding style

```
function reviewSubjects(subjectList)
%Function to go through each subject in a list and perform review

    for i = 1:length(subjectList)

        fprintf('Reviewing subject %d',subjectList(i));
        x = reviewData(subjectList(i));

        if x
            fprintf('Review successful\n');
        else
            fprintf('Review unsuccessful\n');
        end

    end

return
```

```
function reviewSubjects(subjectList)
%Function to go through each subject in a list and perform review

for i = 1:length(subjectList)

fprintf('Reviewing subject %d',subjectList(i));
x = reviewData(subjectList(i));

if x
fprintf('Review successful\n');
else
fprintf('Review unsuccessful\n');
end

end

return
```

Assignment session # 4

Write a function called yourInitials_session4()

The function should take no input:

1) make a random variable ‘x_data’ 1*200

2) Save your variable in memory

3) Plot your variable in line plot

4) make a random variable ‘y_data’ 1*200 and plot with x_data in scatter plot