CME 192: Introduction to Matlab

Lecture 3

Over 600 Rhode Islands could fit inside Alaska

Lecture 2 Topics

- Scripts and functions
- Control flow
- Debugging

Lecture 3 Topics

- Data Structures
  - Cell arrays
  - Struct
- Plotting

Data Structs

- A structure array is a data type that groups related data using data containers called fields. Each field can contain any type of data.

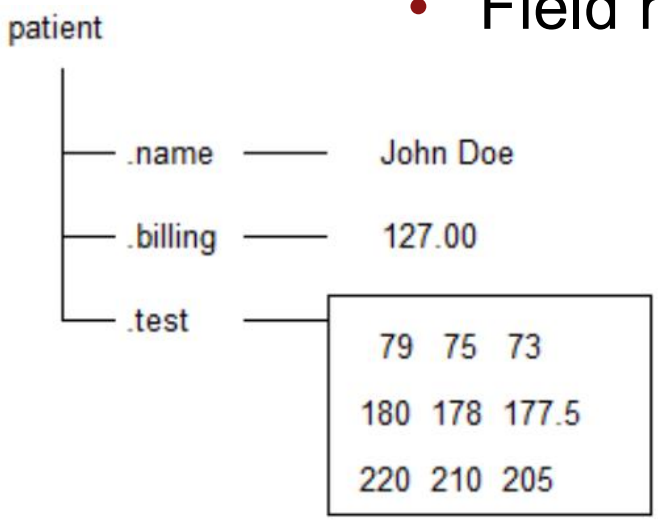
- Access data in a structure using dot notation of the form structName.fieldName.

Data Struct Applications

- Very useful for organizing big sets of data.
- Saving time series and analysis on it.
- Common way of saving research data.

Struct Example

- Struct name is ‘patient’
- Field names are ‘name’, ‘billing’, and ‘test’.




```
patient(1).name = 'John Doe';
patient(1).billing = 127.00;
patient(1).test = [79, 75, 73; 180, 178, 177.5; 220, 210, 205];
patient
```

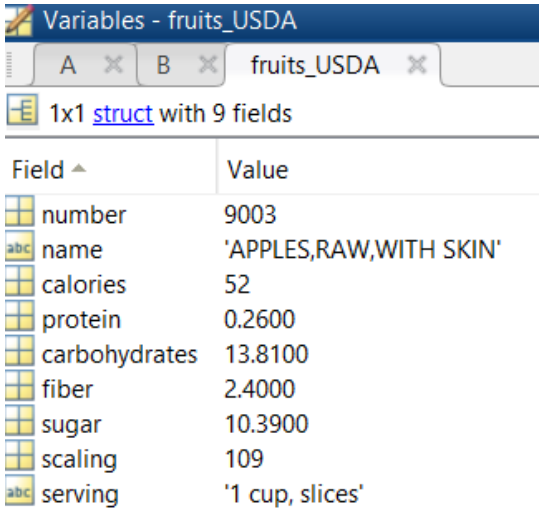
Creating Data Structs

Struct name

Alternating field name and field value

```
% Example entry for using structs
%Single struct
fruits_USDA = struct('number', 09003,'name', 'APPLES,RAW,WITH SKIN',...
                     'calories', 52, 'protein', 0.26, ...
                     'carbohydrates', 13.81, 'fiber', 2.4, 'sugar', 10.39,...
                     'scaling', 109, 'serving', '1 cup, slices');
```

The fields of the struct



Appending to Structs

We perform the same process but we add a second element to our struct.

```
%Appending to struct
fruits_USDA(2) = struct('number', 09038,'name', 'AVOCADOS,RAW,CALIFORNIA',...
                        'calories', 167, 'protein', 1.96, ...
                        'carbohydrates', 8.64, 'fiber', 6.8, 'sugar', 0.3,...
                        'scaling', 136, 'serving', '1 fruit, without skin and seed');
```

The same field names must be used. In the variable editor each row is a different fruit.

Variables - fruits_USDA

Editor - cellArrayExample.m

A | B | fruits_USDA

1x2 struct with 9 fields

Fields	number	name	calories	protein	carbohydrates	fiber	sugar	scaling	serving
1	9003	'APPLES,...	52	0.2600	13.8100	2.4000	10.3900	109	'1 cup, sli...
2	9038	'AVOCAD...	167	1.9600	8.6400	6.8000	0.3000	136	'1 fruit, wi...

Creating a Large Array of Struct

Each field name is given and the respective values are given in “{ }”.

```
fruits_USDA = struct(...
    'number', {9003, 9038, 9050, 9070, 9148, 9176, 9191, 9316},...
    'name', {'APPLES,RAW,WITH SKIN','AVOCADOS,RAW,CALIFORNIA',...
             'BLUEBERRIES,RAW','CHERRIES,SWEET,RAW','KIWIFRUIT,GRN,RAW',...
             'MANGOS,RAW','NECTARINES,RAW','STRAWBERRIES,RAW'},...
    'calories', {52   167    57    63    61    60    44    32},...
    'protein', {0.2600 1.9600 0.7400 1.0600 1.1400 0.8200 1.0600 0.6700},...
    'carbs', {13.8100 8.6400 14.4900 16.0100 14.6600 14.9800 10.5500 7.6800},...
    'fiber', {2.4000 6.8000 2.4000 2.1000 3.0000 1.6000 1.7000 2.0000},...
    'sugar', {10.3900 0.3000 9.9600 12.8200 8.9900 13.6600 7.8900 4.8900},...
    'scaling', {109   136    68   138    69   336   129   152},...
    'serving', {'1 cup, slices','1 fruit, without skin and seed',...
                '50 berries','1 cup, with pits yields','1 fruit,  (2" dia)',...
                '1 fruit, without refuse','1 small,  (2-1/3" dia)',...
                '1 cup,  halves'});
```

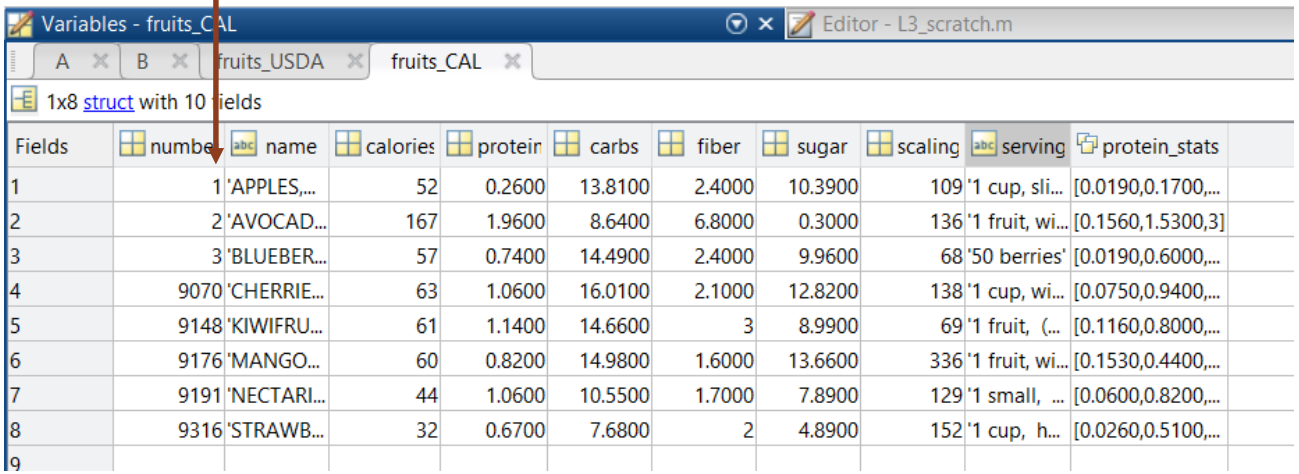
Copying and Modifying a Struct

```
fruits_CAL = fruits_USDA
```

Makes a hard copy of the struct. Changes to one struct will not be updated in the other.

```
fruits_CAL(1).number = 1;
fruits_CAL(2).number = 2;
fruits_CAL(2).number = 3;
```

Updates can be made



Variables - fruits_CAL

A | B | fruits_USDA | fruits_CAL

1x8 struct with 10 fields

Fields	number	name	calories	protein	carbs	fiber	sugar	scaling	serving	protein_stats
1	1	'APPLES,...	52	0.2600	13.8100	2.4000	10.3900	109	'1 cup, sli...	[0.0190,0.1700,...
2	2	'AVOCAD...	167	1.9600	8.6400	6.8000	0.3000	136	'1 fruit, wi...	[0.1560,1.5300,3]
3	3	'BLUEBER...	57	0.7400	14.4900	2.4000	9.9600	68	'50 berries'	[0.0190,0.6000,...
4	9070	'CHERRIE...	63	1.0600	16.0100	2.1000	12.8200	138	'1 cup, wi...	[0.0750,0.9400,...
5	9148	'KIWIFRU...	61	1.1400	14.6600	3	8.9900	69	'1 fruit, (...	[0.1160,0.8000,...
6	9176	'MANGO...	60	0.8200	14.9800	1.6000	13.6600	336	'1 fruit, wi...	[0.1530,0.4400,...
7	9191	'NECTARI...	44	1.0600	10.5500	1.7000	7.8900	129	'1 small, ...	[0.0600,0.8200,...
8	9316	'STRAWB...	32	0.6700	7.6800	2	4.8900	152	'1 cup, h...	[0.0260,0.5100,...
9										

Plotting in Matlab

- When using a Matlab plot function a new window will automatically come up.
- You can define new figure windows by using the command ‘figure’
- Figures can be numbered such as `figure(1); figure(2); figure(3);`

Plotting Options

- Matlab will plot over whichever figure is open by default. To plot a second graph over that one use ‘hold on’.
- ‘xlablel’, ‘ylabel’, and ‘zlabel’ are all used to label the axes.
- A legend can be added by using ‘legend’ and listing all the domains.

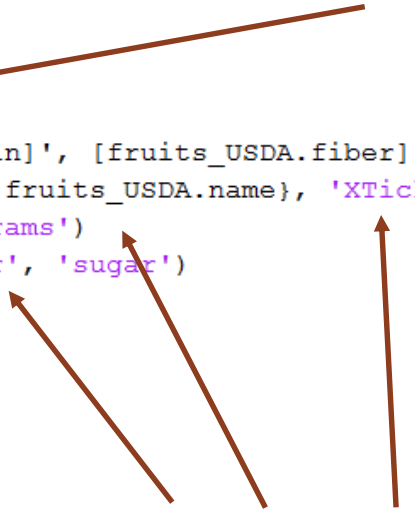
Types of Plots

- Bar
- Line
- Scatter
- 3D

Bar Plot

Creates the bar plot and plots the values from the designated fields.

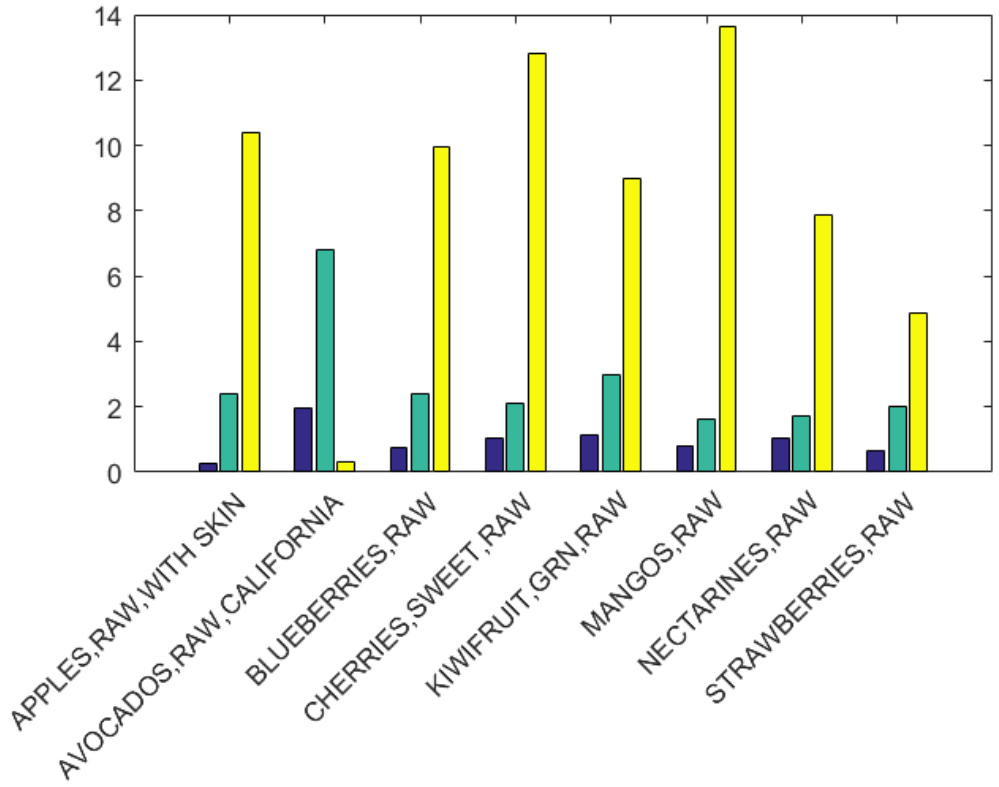
```
%% Basic bar plot

bar([[fruits_USDA.protein]', [fruits_USDA.fiber]', [fruits_USDA.sugar]' ])
set(gca, 'XTickLabel', {fruits_USDA.name}, 'XTickLabelRotation', 45)
ylabel('Grams per 100 grams')
legend('protein', 'fiber', 'sugar')
```



- Creates the labels for the plot.
- gca stands for “get current axes.”

Plot Output

```
%% Basic bar plot

bar([[fruits_USDA.protein]', [fruits_USDA.fiber]', [fruits_USDA.sugar]' ])
set(gca, 'XTickLabel', {fruits_USDA.name}, 'XTickLabelRotation', 45)
ylabel('Grams per 100 grams')
legend('protein', 'fiber', 'sugar')
```



Stacking the Bars

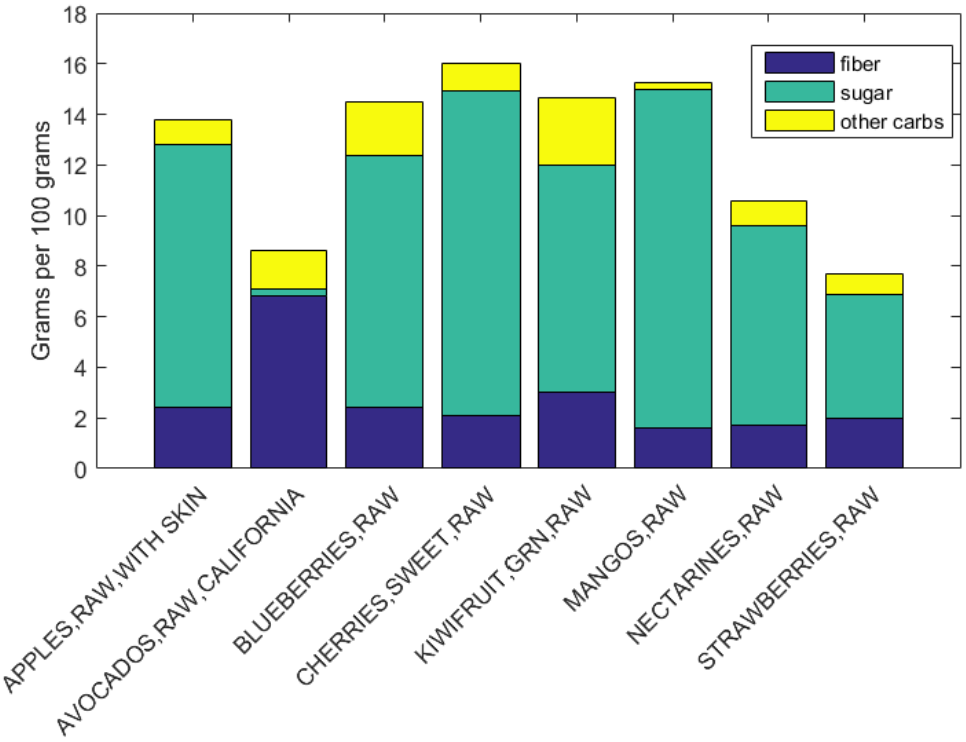
Create new variable

Stack the fields

```
%% Stacked bar plot

other_carbs = [fruits_USDA.carbs]'-...
    [fruits_USDA.fiber]'-[fruits_USDA.sugar]';
bar([[fruits_USDA.fiber]', [fruits_USDA.sugar]',...
    other_carbs], 'stacked')
set(gca, 'XTickLabel', {fruits_USDA.name}, 'XTickLabelRotation', 45)
ylabel('Grams per 100 grams')
legend( 'fiber', 'sugar', 'other carbs')
```

Stacking the Bars

```
%% Stacked bar plot

other_carbs = [fruits_USDA.carbs]'-...
    [fruits_USDA.fiber]'-[fruits_USDA.sugar]';
bar([[fruits_USDA.fiber]', [fruits_USDA.sugar]',...
    other_carbs], 'stacked')
set(gca, 'XTickLabel', {fruits_USDA.name}, 'XTickLabelRotation', 45)
ylabel('Grams per 100 grams')
legend( 'fiber', 'sugar', 'other carbs')
```



3D Bar Graph

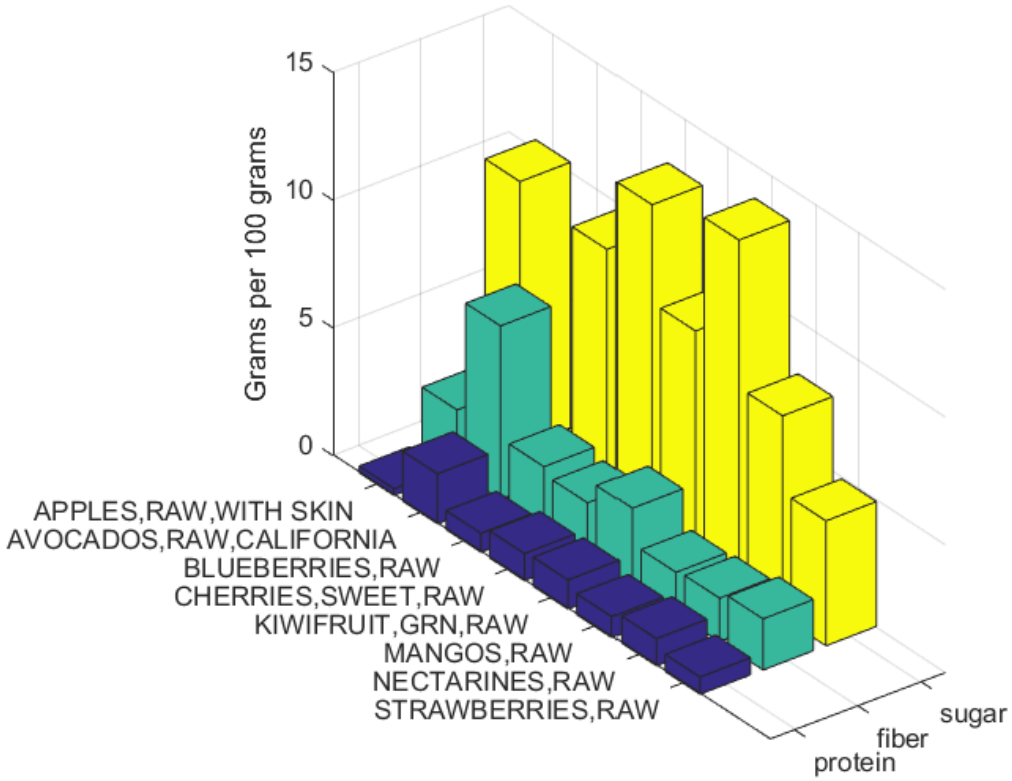
We now use the function ‘bar3’

```
%% 3D bar plot

bar3([[fruits_USDA.protein]', [fruits_USDA.fiber]', [fruits_USDA.sugar]' ])
set(gca, 'YTickLabel', {fruits_USDA.name})
set(gca, 'XTickLabel',{'protein', 'fiber', 'sugar'})
zlabel('Grams per 100 grams')
```

- The x, y, and z axes are labeled.
- In y we give the fruit names.
- In x we give the calorie type.
- In z we give the grams.

3D Bar Graph

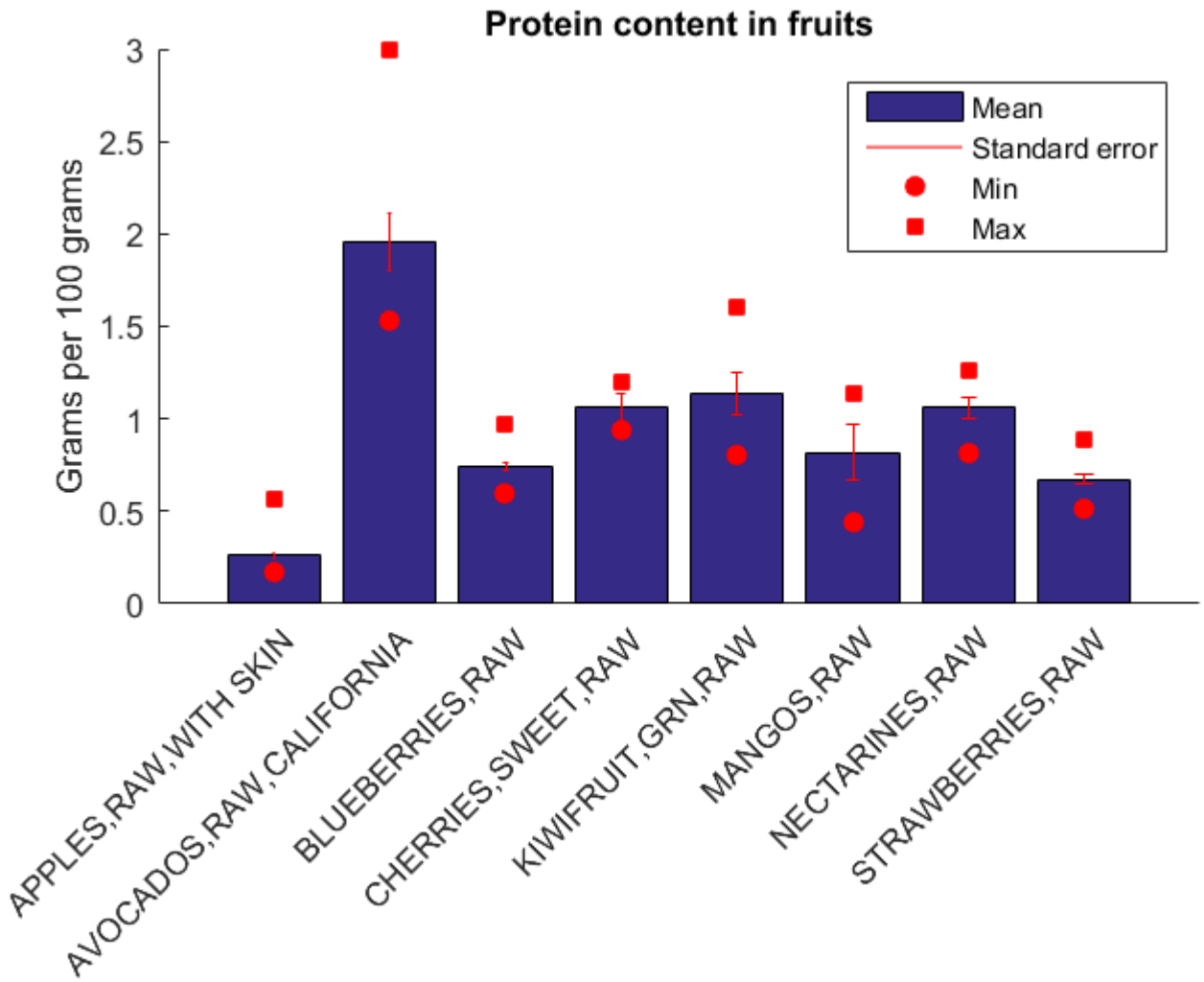
```
%% 3D bar plot

bar3([[fruits_USDA.protein]', [fruits_USDA.fiber]', [fruits_USDA.sugar]' ])
set(gca, 'YTickLabel', {fruits_USDA.name})
set(gca, 'XTickLabel',{'protein', 'fiber', 'sugar'})
zlabel('Grams per 100 grams')
```



Plotting Additional Data

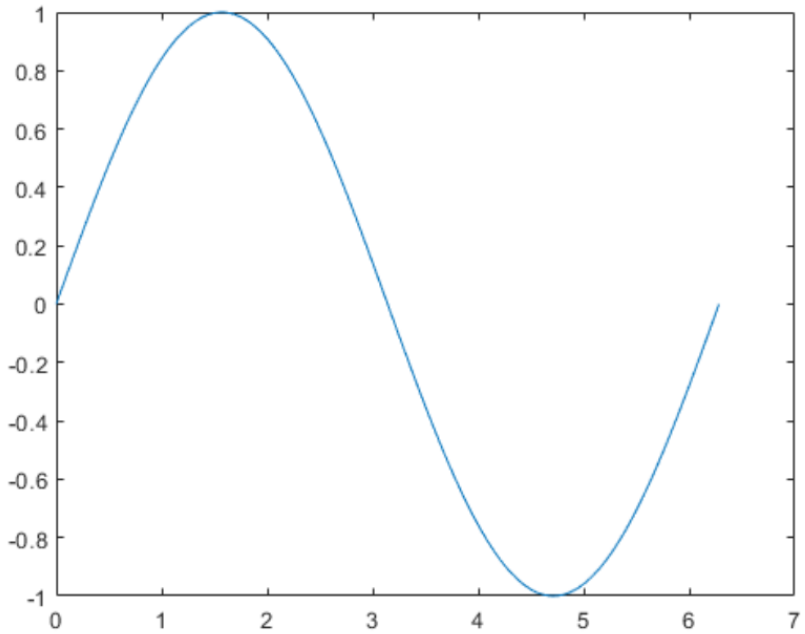
```
%% Bar plot with errorbars and max/min values

figure()
hold on
h1 = bar([fruits_USDA.protein]);
for ifruit = 1:size(fruits_USDA,2)
    %Plot standard deviation as error bar
    h2 = errorbar(ifruit, fruits_USDA(ifruit).protein, ...
        fruits_USDA(ifruit).protein_stats(1),'r');
    %Plot min and max measurements
    h3 = plot(ifruit, fruits_USDA(ifruit).protein_stats(2), 'ro',...
        'MarkerFaceColor', 'r');
    h4 = plot(ifruit, fruits_USDA(ifruit).protein_stats(3), 'rs',...
        'MarkerFaceColor', 'r');
end
title('Protein content in fruits')
set(gca, 'XTick', 1:8, 'XTickLabel', {fruits_USDA.name}, 'XTickLabelRotation', 45)
ylabel('Grams per 100 grams')
legend([h1 h2 h3 h4], {'Mean', 'Standard error', 'Min', 'Max'})
```

Plotting Additional Data



Plotting Numerical Functions

```
x = 0:pi/100:2*pi;
y = sin(x);
plot(x,y)
```

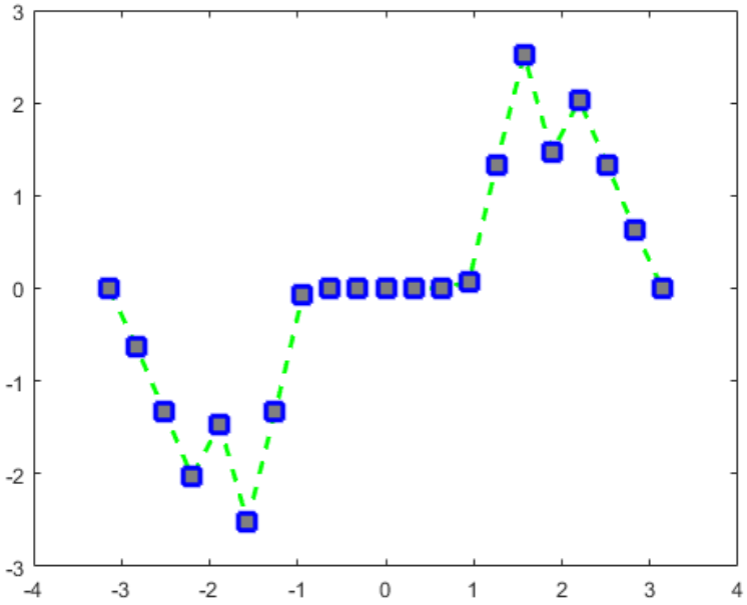


https://www.mathworks.com/help/matlab/ref/plot.html

Plotting Options

```
x = -pi:pi/10:pi;
y = tan(sin(x)) - sin(tan(x));

figure
plot(x,y,'--gs',...
    'LineWidth',2,...
    'MarkerSize',10,...
    'MarkerEdgeColor','b',...
    'MarkerFaceColor',[0.5,0.5,0.5])
```

Markers and line parameters can be modified.



Plotting Options

```
plot(X,Y)
plot(X,Y,LineSpec)
plot(X1,Y1,...,Xn,Yn)
plot(X1,Y1,LineSpec1,...,Xn,Yn,LineSpecn)
```

Line

expand all

`LineStyle` — Line style
`'-'` (default) | `'--'` | `':'` | `'-.'` | `'none'`

`LineWidth` — Line width
`0.5` (default) | positive value

`Color` — Line color
`[0 0 0]` (default) | RGB triplet | `'r'` | `'g'` | `'b'` | ...

`LineJoin` — Style of line corners
`'round'` (default) | `'miter'` | `'chamfer'`

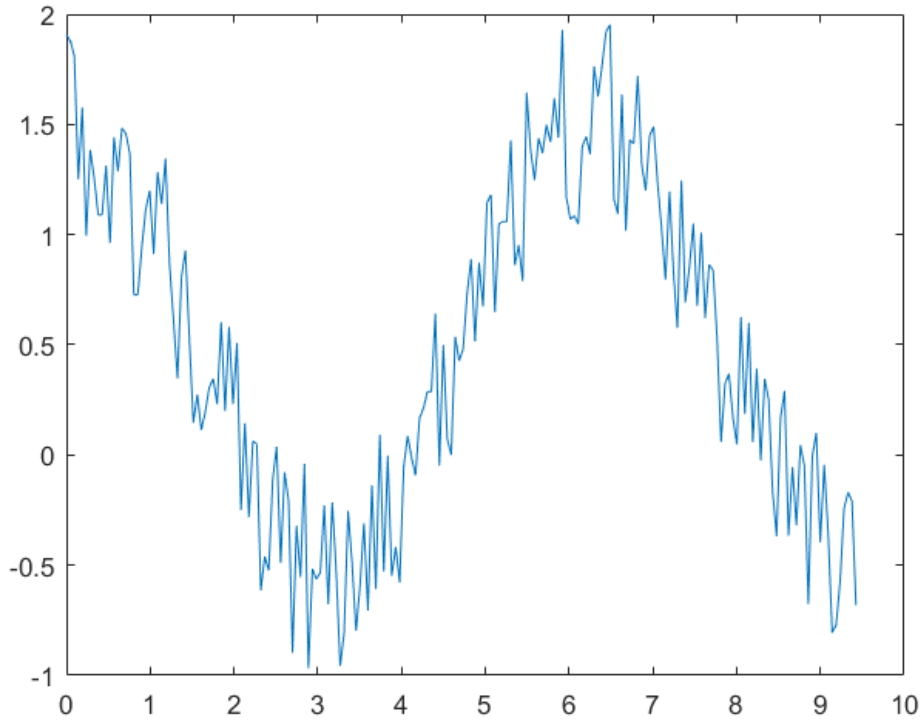
`AlignVertexCenters` — Sharp vertical and horizontal lines
`'off'` (default) | `'on'`

https://www.mathworks.com/help/matlab/ref/matlab.graphics.chart.primitive.line-properties.html
https://www.mathworks.com/help/matlab/ref/plot.html

Another Way of Viewing Data?

```
x = linspace(0,3*pi,200);
y = cos(x) + rand(1,200);
plot(x, y)
```

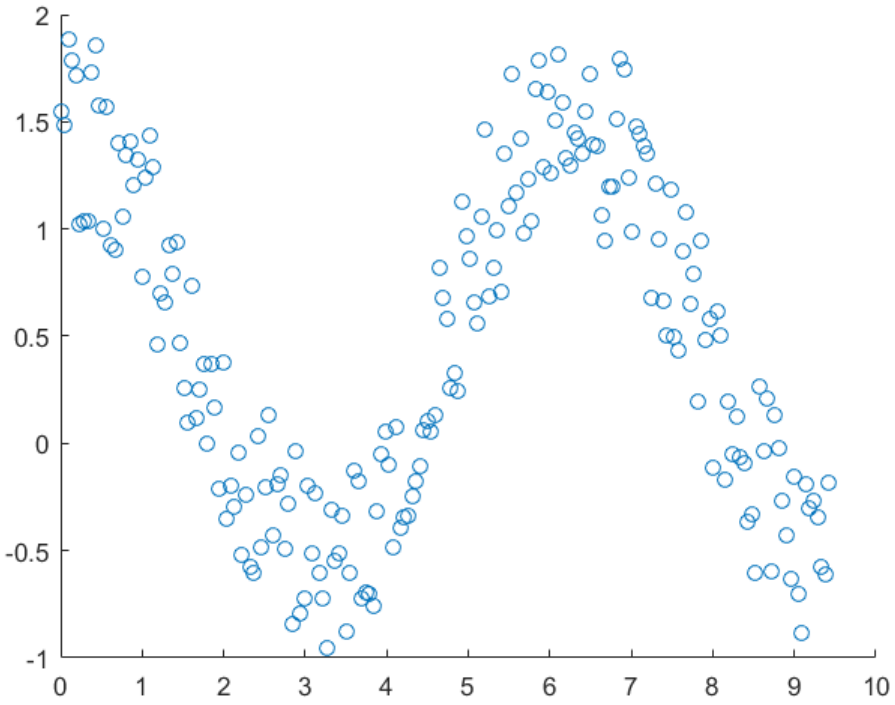
Line plots may not always be the best method for viewing data



Scatter Plot

```
x = linspace(0,3*pi,200);
y = cos(x) + rand(1,200);
scatter(x, y)
```

Places data points but does not connect them.

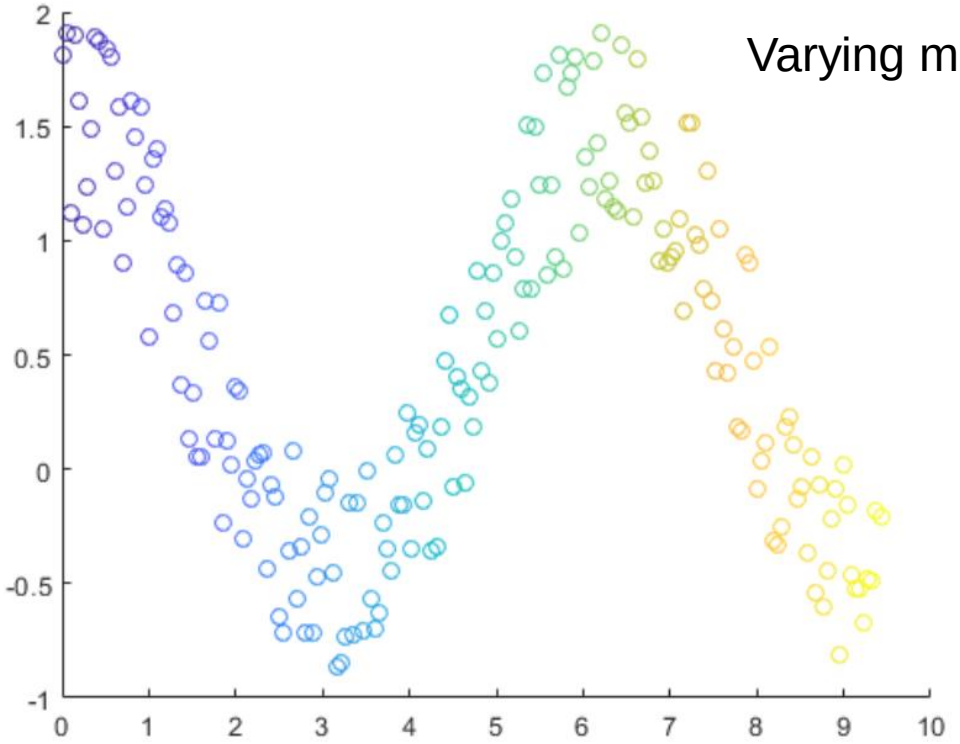


Scatter Plots

```
x = linspace(0,3*pi,200);
y = cos(x) + rand(1,200);
c = linspace(1,10,length(x));
scatter(x,y,[],c)
```

color

size



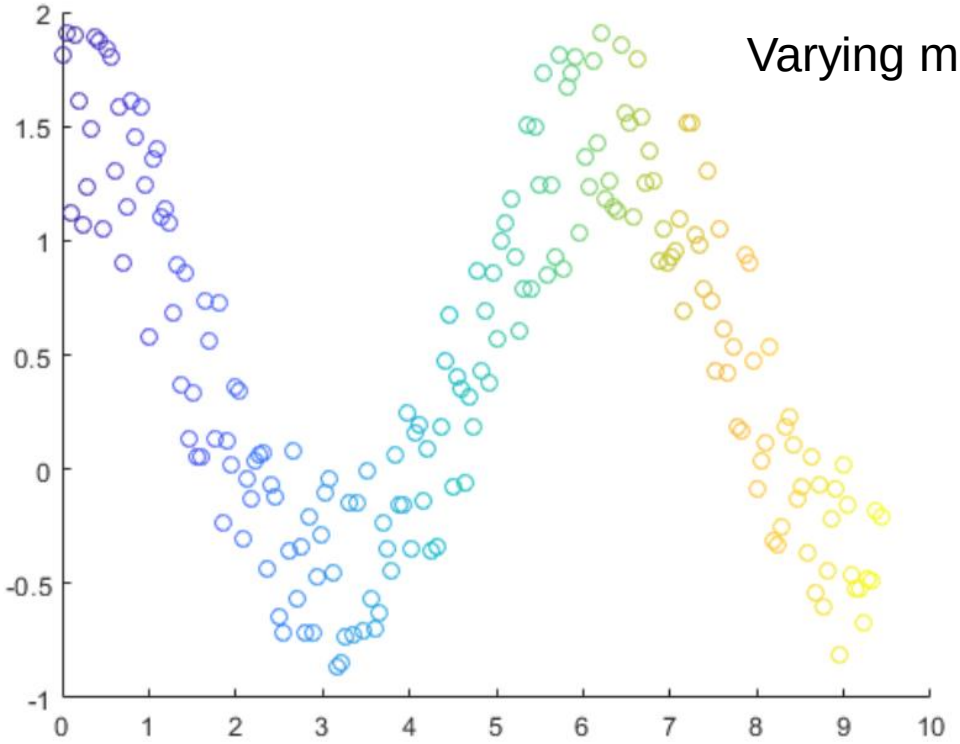
Varying marker color.

Scatter Plots

```
x = linspace(0,3*pi,200);
y = cos(x) + rand(1,200);
c = linspace(1,10,length(x));
scatter(x,y,[],c)
```

color

size



Varying marker color.