CE 300 SUMMER PRACTISE
INTRODUCTION TO MATLAB

MATLAB TOOLBOXES

Beliz Uğurhan
K6-108

What is a Matlab Toolbox?

Toolboxes are libraries of extra routines.

Example Toolboxes Civil Engineers Use:

- Aerospace Toolbox
- Curve Fitting Toolbox
- Financial Toolbox
- Image Processing Toolbox
- Mapping Toolbox
- Optimization Toolbox
- Paralle Computing Toolbox
- Signal Processing Toolbox
- Statistics Toolbox
- Symbolic Math Toolbox
- System Identification Toolbox
- Simulink

OUTLINE

- Symbolic Math Toolbox
- Curve Fitting Toolbox
- Statistics Toolbox

OBJECTIVE:

Introduce you 3 toolboxes of MATLAB.

Get familiar with the basics of these toolboxes.

Symbolic Math Toolbox

- Perform symbolic computations.
- Basic tasks:
  - Differentiation
  - Integration
  - Linear algebra
  - Simplification
  - Equation solving

- You can access from **Matlab Command Line** or from MuPAD Notebook.

Symbolic Toolbox Basics

- **x=sym('x')** defines x as a symbolic object
- **x=sym('w','x','y','z')** defines w,x,y,z as symbolic objects
- **syms w x y z**
- **a=sym(3/5)** a is defined as a symbolic object and assigned to 3/5.
- **M=[w x; y z]** M is a symbolic matrix since w,x,y,z are symbolic objects.
- **f=5*x^2+3*x-2** By assigning different values to x, you can obtain the correspoding values of f by: **eval(f)**

OR

subs(f, 1) Substitutes 1 as x in f.

```
>> syms x
>> f=5*x^2+3*x-2;
>> x=1;
>> eval(f)

ans =

     6

>> x=2;
>> eval(f)

ans =

    24

>> subs(f,1)

ans =

     6

>>
```

Symbolic Expressions

Given a function f(x)

- **simplify(f)** Simplifies the symbolic expression
- **expand(f)** Groups similar terms, multiplies parenthesis to rewrite a polynomial in a standart form
- **factor(f)** Shows the polynomial roots

```
>> syms x
>> f=(x+3)*(x^2+5*x+2)+(x^2+1)*(x+3)
>> expand(f)

ans =

2*x^3 + 11*x^2 + 18*x + 9

>> factor(f)

ans =

(2*x + 3)*(x + 3)*(x + 1)

>>
```

Differentiation and Integration

diff and **int** are the two functions used for differentiation and integration, respectively.

```
>> syms x
>> f=atan(x);
>> diff(f)

ans =

1/(x^2 + 1)

>> syms y
>> f=sin(x)*exp(2*y);
>> diff(f,y)

ans =

2*exp(2*y)*sin(x)

>>
```

Differentiate f with respect to y

```
>> syms x
>> f=x^2+x+3;
>> y=int(f)

y =

(x*(2*x^2 + 3*x + 18))/6

>> expand(y)

ans =

x^3/3 + x^2/2 + 3*x

>> syms y
>> f=x^4+y^3;
>> int(f,y)

ans =

x^4*y + y^4/4

>> int(f,y,1,10)

ans =

9*x^4 + 9999/4

>>
```

INDEFINITE INTEGRAL
Integrate f with respect to y

DEFINITE INTEGRAL
Integrate f with respect to y between 1 and 10.

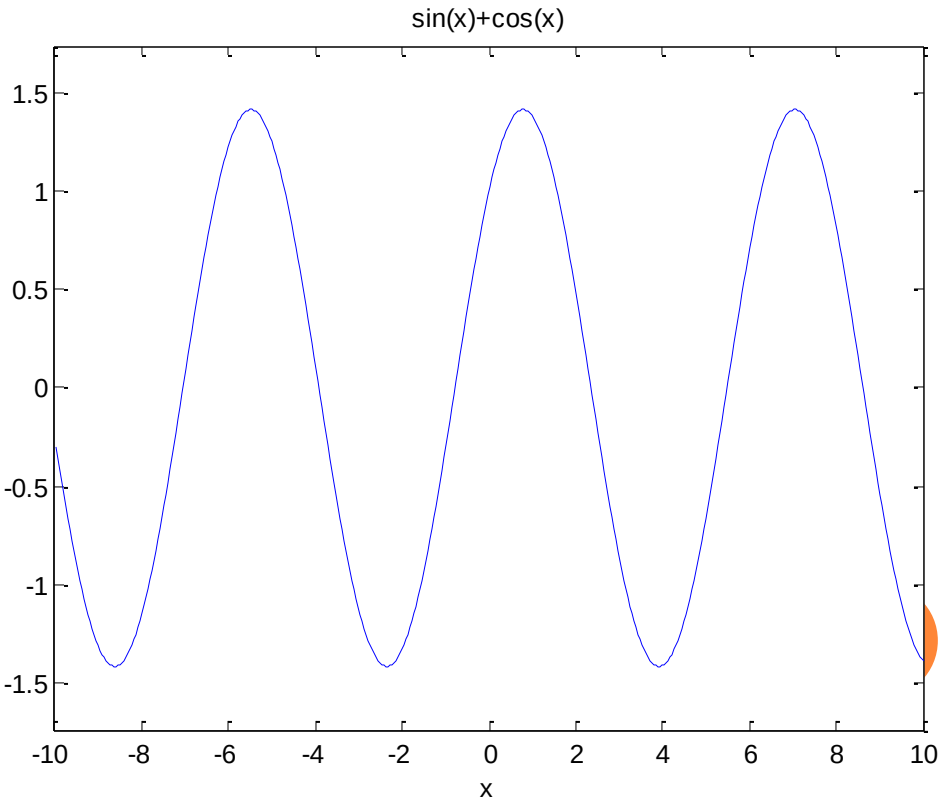
Equation Solving & Plotting

solve is used to obtain the solution of system equations.

```
>> syms x x1 x2 x3
>> eq='x^2-6*x-7';
>> solve(eq)

ans =

 -1
  7

>> eq1='x1+2*x2+2*x3=0';
>> eq2='2*x1+3*x2+4*x3=1';
>> eq3='-x1+x2+x3=-6';
>> sol=solve(eq1,eq2,eq3);
>> sol.x1

ans =

4

>> sol.x2

ans =

-1

>> sol.x3

ans =

-1
```

ezplot is used to obtain simple plots of equations.

```
>> syms x
>> f='sin(x)+cos(x)';
>> ezplot(f,[-10,10]);
>>
```

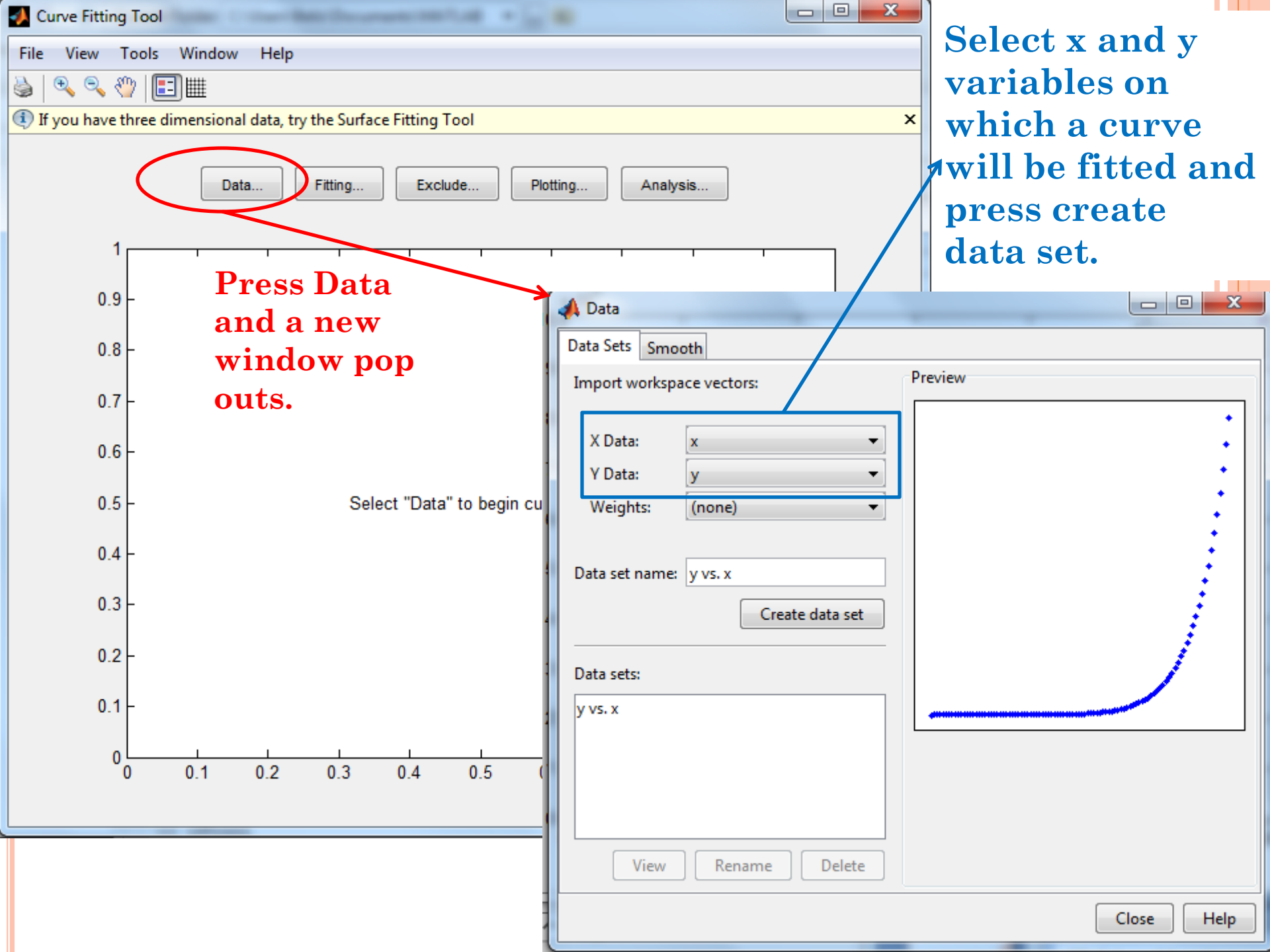


Curve Fitting Toolbox

- Fitting methods for linear least squares, nonlinear least squares, weighted least squares, constrained least squares, and robust fitting are available
- Data and fit statistics to assist you in analyzing your models

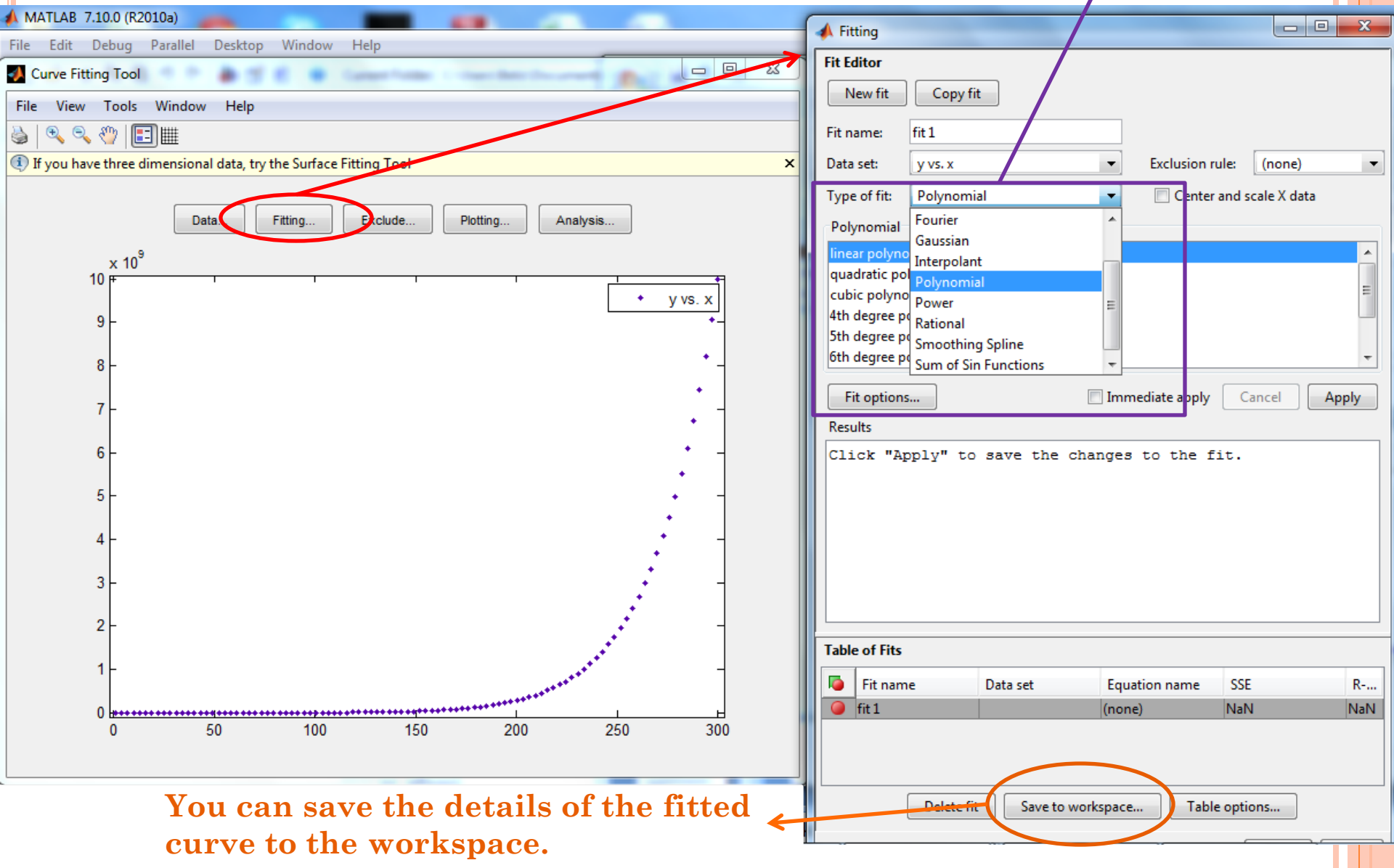
- In the command window, type:
  - **cftool** to open **Curve Fitting Toolbox**
  - **sftool** to open Surface Fitting Toolbox

- We will work on Curve Fitting Toolbox.

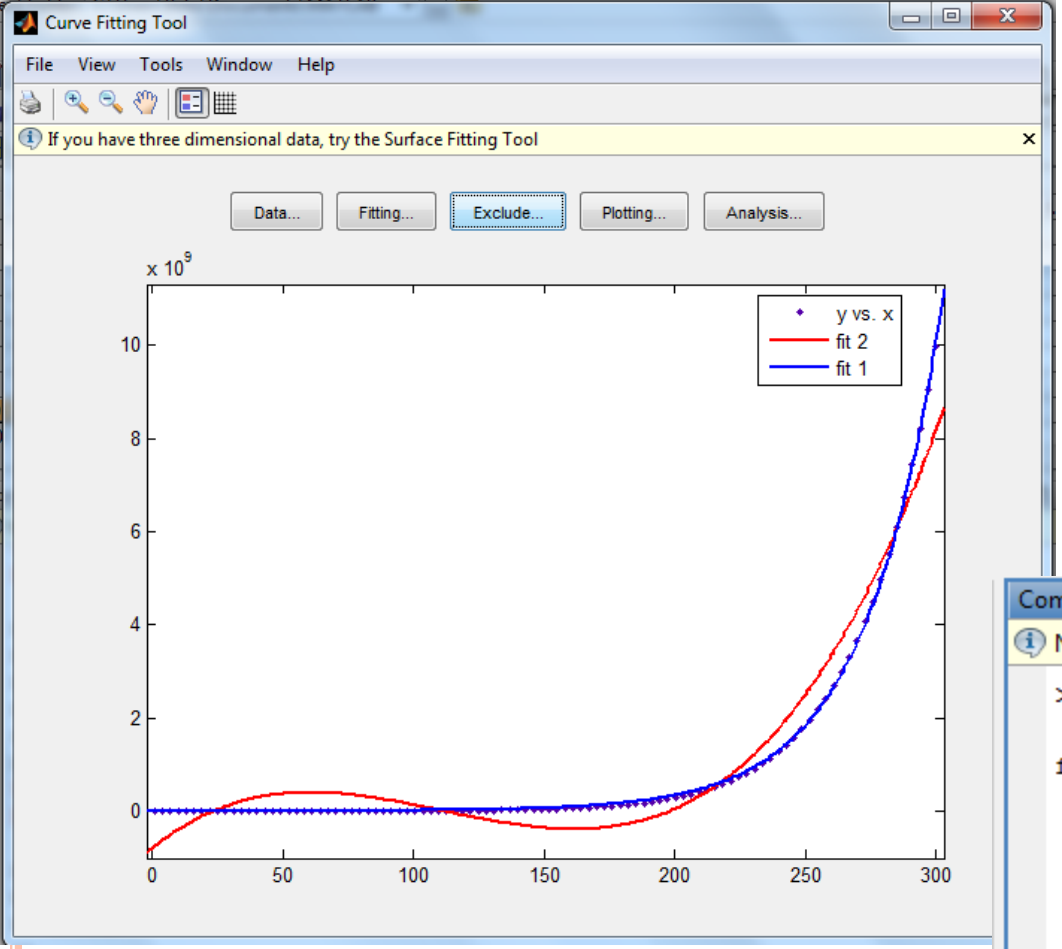

Curve Fitting Tool
File View Tools Window Help
If you have three dimensional data, try the Surface Fitting Tool
Data... Fitting... Exclude... Plotting... Analysis...
Press Data and a new window pop outs.
Select "Data" to begin cu
Select x and y variables on which a curve will be fitted and press create data set.
Data
Data Sets Smooth
Import workspace vectors:
X Data: x
Y Data: y
Weights: (none)
Data set name: y vs. x
Create data set
Data sets:
y vs. x
View Rename Delete
Preview
Close Help

After having created the data set, press Fitting button.

Select the type of fit.



You can save the details of the fitted curve to the workspace.



- In the curve fitting tool, you can visualize the fitted models with respect to the dataset.

- You can see the properties of the fitted plot from the command window.

Command Window

New to MATLAB? Watch this Video, see Demos, or read Getting Started.

```
>> fittedmodel1

fittedmodel1 =

     General model Gauss1:
     fittedmodel1(x) =  a1*exp(-((x-b1)/c1)^2)
     Coefficients (with 95% confidence bounds):
       a1 =  1.912e+032  (-2.389e+037, 2.389e+037)
       b1 = -4.296e+004  (-7.72e+007, 7.711e+007)
       c1 =        5153  (-4.617e+006, 4.627e+006)
>> fittedmodel2

fittedmodel2 =

     General model Exp1:
     fittedmodel2(x) = a*exp(b*x)
     Coefficients (with 95% confidence bounds):
       a =  3.701e+005  (3.481e+005, 3.921e+005)
       b =     0.03405  (0.03384, 0.03426)

fx >>
```

Curve Fitting From the Command Line

- **options = fitoptions('Method',*method*)**

Method	The fitting method. A complete list of supported fitting methods is given below. The default is 'None'.	
	'NearestInterpolant'	Nearest neighbor interpolation
	'LinearInterpolant'	Linear interpolation
	'PchipInterpolant'	Piecewise cubic Hermite interpolation (curves only)
	'CubicSplineInterpolant'	Cubic spline interpolation
	'BiharmonicInterpolant'	Biharmonic surface interpolation
	'SmoothingSpline'	Smoothing spline
	'LowessFit'	Lowess smoothing (surfaces only)
	'LinearLeastSquares'	Linear least squares
	'NonlinearLeastSquares'	Nonlinear least squares

- **ffun = fittype(libname)** ⟹

libname	Description
'poly1'	Linear polynomial curve
'poly11'	Linear polynomial surface
'poly2'	Quadratic polynomial curve
'linearinterp'	Piecewise linear interpolation
'cubicinterp'	Piecewise cubic interpolation
'smoothingspline'	Smoothing spline (curve)
'lowess'	Local linear regression (surface)

- **ffun = fittype(expr)**

⇩

```
>> g = fittype('a*x^2+b*x+c','coeff',{'a','b','c'})

g =

     General model:
     g(a,b,c,x) = a*x^2+b*x+c
>>
```

Curve Fitting From the Command Line

- **[fitobject, gof] = fit(x,y,fitType)** fits the data in x and y with the library model, anonymous function or fittype object specified by fitType.

- It returns the goodness-of-fit statistics to the structure gof which are summarized in the below table.

Field	Value
sse	Sum of squares due to error
R2	Coefficient of determination
adjustedR2	Degree-of-freedom adjusted coefficient of determination
stdError	Root mean squared error (standard error)

Statistics Toolbox

- Aim is to assess and understand data

- Functions for modeling data, analyzing historical trends, simulating systems, developing statistical algorithms.

Key Features

- Data organization and management
- Descriptive statistics
- Statistical plotting and data visualization
- Probability distributions
- Analysis of variance (ANOVA)
- Linear and nonlinear modeling
- Multivariate statistics
- Design of Experiments (DOE)
- Hypothesis testing
- Statistical Process Control (SPC)

Measure of Central Tendency

Function Name	Description
geomean	Geometric mean
harmmean	Harmonic mean
mean	Arithmetic average
median	50th percentile
mode	Most frequent value
trimmean	Trimmed mean

Note that;

mean(x) is the average of the data x

median(x) is the middle value of the data x sorted by value

mode (x) is the most common value in the dataset x

STATISTICAL ANALYSIS

```
>> a=[1 2 3;1 3 4;2 2 5]

a =

     1     2     3
     1     3     4
     2     2     5
```

```
>> mean(a)

ans =

    1.3333    2.3333    4.0000

>> median(a)

ans =

     1     2     4

>> mode(a)

ans =

     1     2     3
```

Measures of Scale (Maximum and Minimum)

max *(x):* find the largest value in x

[y,k]=max*(x):* find y and k that are the maximum value of x, indices of first maximum.

max*(x,y)*: compares x and y and report the minimum

min*(x):* find the smallest value in x

[y,k]=min*(x):* find y and k that are the minimum value of x, indices of first minimum.

min*(x,y):* compares x and y and report the minimum

STATISTICAL ANALYSIS

```
>> a=[1 2 3;4 5 6;7 8 9];
b=[7 8 9;1 2 3;4 5 6];
```

```
>> max(a)
```

```
>> [y,k]=max(a)
```

```
>> min(a,b)
```

```
>> max(1,5)
```

Measures of scale (Sum and Product)

sum *(x):* sum of elements

prod *(x):* product of elements

cumsum *(x):* cumulative sum of elements

cumprod *(x):* cumulative product of the elements

STATISTICAL ANALYSIS

```
>> A=[1 2 3;4 5 6;7 8 9];
```

```
>> sum(A)

ans =

    12    15    18
```

```
>> cumsum(A)

ans =

     1     2     3
     5     7     9
    12    15    18
```

```
>> prod(A)

ans =

    28    80   162
```

```
>> cumprod(A)

ans =

     1     2     3
     4    10    18
    28    80   162
```

Measure of dispersion

Function Name	Description
iqr	Interquartile range
mad	Mean absolute deviation
moment	Central moment of all orders
range	Range
std	Standard deviation
var	Variance

Note that;

standard deviation shows how much variation there is from the data average

variance shows how far a set of numbers are spread out from each other.

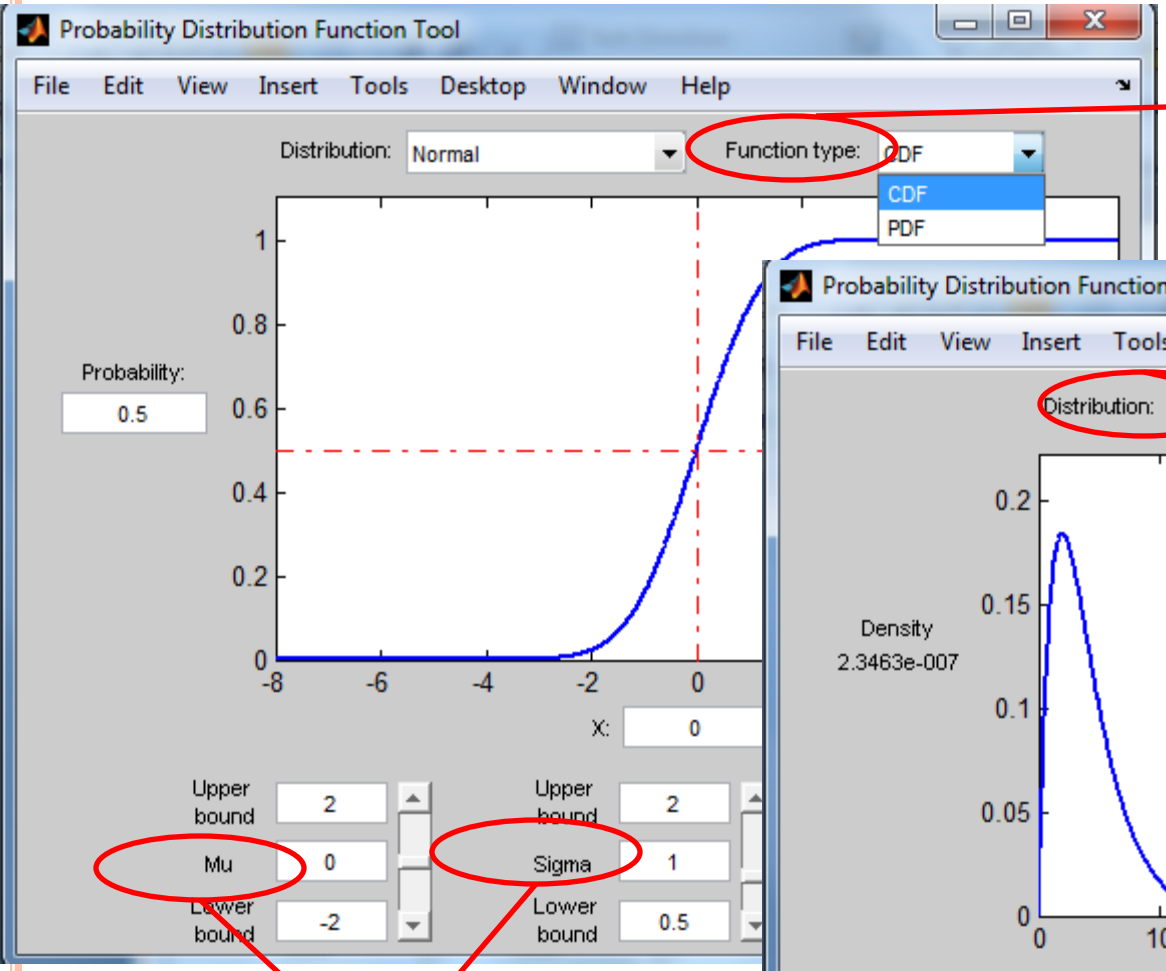
Random number generation

- **randn([m n])** returns an mxn matrix containing numbers that obey standart normal distribution

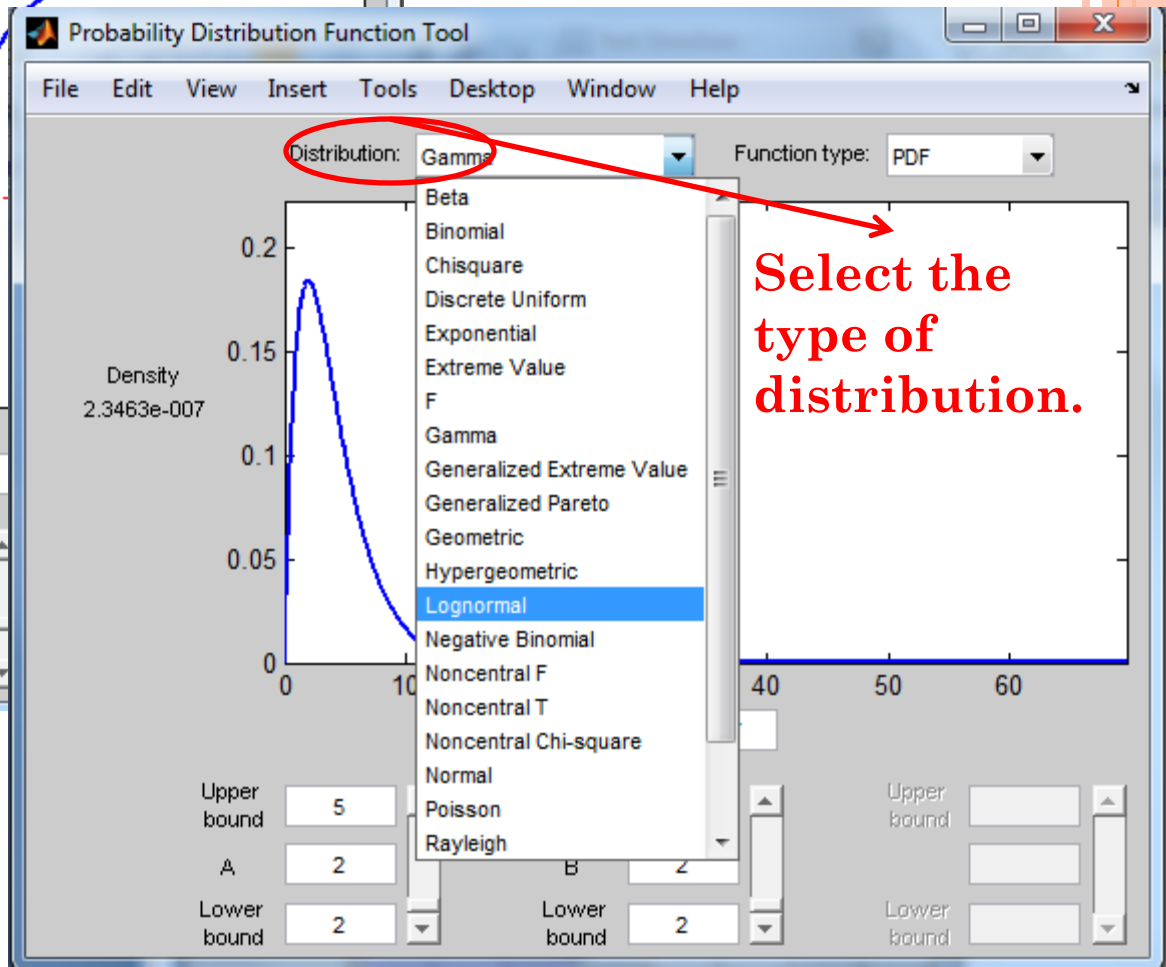
- **lognrnd (mu,sigma,m,n)** returns an mxn matrix containing numbers that obey lognormal distribution that have mu and sigma as the mean and stdev of the corresponding normal dist.

Probability Distribution

- Type **disttool** to the command window



You can select the function type, either a PDF or a CDF.



Select the type of distribution.

μ and σ, which are the identifiers of a distribution are defined.

QUESTIONS??