Bil 108

Introduction to the Scientific and Engineering Computing with MATLAB

Lecture 10

F. Aylin Konuklar   Lale T. Ergene

The basic differences between numerical and symbolic computation are:

Numerical	Symbolic
Variables represent numbers	Variables
Answers can only be numbers	Answers can contain variables and functions
Numeric computation can be done unsing standard programming languages	Symbolic computations are not similar to standard programming languages – they use symbolic manipulations

Symbolic Manipulation

- The computational engine underlying the toolboxes is the kernel of Maple®,
- Available in the "Symbolic Math Toolbox" – which is completely installed in the professional version
- The Symbolic Math Toolbox defines a new MATLAB data type called a *symbolic object*.

Properties

Facility	Covers
Calculus	Differentiation, integration, limits, summation, and Taylor series
Linear Algebra	Inverses, determinants, eigenvalues, singular value decomposition, and canonical forms of symbolic matrices
Simplification	Methods of simplifying algebraic expressions
Solution of Equations	Symbolic and numerical solutions to algebraic and differential equations
Variable-Precision Arithmetic	Numerical evaluation of mathematical expressions to any specified accuracy
Transforms	Fourier, Laplace, *z*-transform, and corresponding inverse transforms

Example

- ```
  >>sqrt(2)
      ans =     1.4142
  ```

returns a floating-point decimal number.

- On the other hand, if you convert 2 to a symbolic object using the sym command, and then take its square root by entering

  ```
  >>a = sqrt(sym(2))    %the result is
      a = 2^(1/2)
  ```

- You can always obtain the numerical value of a symbolic object with the double command:

  ```
  >>double(a)
      ans = 1.4142
  ```
  ```

Symbolic Algebra

- Used regularly in math, engineering and science classes
- Often preferable to manipulate equations symbolically before you substitute in values for variables

$$y = \frac{2*(x+3)^2}{x^2+6x+9}$$

Consider this equation

Symbolic Algebra

$$y = \frac{2*(x+3)^2}{x^2+6x+9}$$

This looks like a fairly complicated function of x

$$y = \frac{2*(x+3)^2}{x^2+6x+9} = \frac{2*(x^2+6x+9)}{(x^2+6x+9)} = 2$$

If you expand it, it simplifies dramatically

Symbolic Algebra

However, when you simplify you may lose information

$$y = \frac{2*(x+3)^2}{x^2+6x+9}$$

Let x equal -3

$$y = \frac{2*(-3+3)^2}{9-18+9} = 2*\frac{0}{0} = undefined$$

When x is equal to -3, the equation is undefined

- MATLAB's symbolic capability allows you perform the simplification, or to manipulate the numerator and denominator separately

Symbolic Variables

- Two approaches
  - Use the **sym** command to create
    - Single variable
    - Expression
    - Equation
  - Use the **syms** command to create
    - Single variables
    - Compose expressions and equations from the variables you’ve defined

Examples

1) Define x as a symbolic variable

>>x=sym('x') or

>>syms x

Use x to create a more complicated expression

>>y = 2*(x+3)^2/(x^2+6*x+9)

2) >>S=sym('x^3 – 2*y^2 + 3*a')

NOTE: In the symbolic math mode, variables are not matrices

Do not use .*, .^, or ./ operators

Examples

- create multiple variables

  >>**syms Q R T D0**

- Use these variables to create another symbolic variables

  >>**D=D0*exp(-Q/(R*T))**

- Create an entire expression with the sym command

  >> **E=sym('m*c^2')**

- create an entire equation, and give it a name

  >>ideal_gas_law=sym('P*V=n*R*Temp')

The findsym Command

- To determine what symbolic variables are present in an expression

```
>>syms a b n t x z
>>f = x^n; g = sin(a*t + b);
```

- You can find the symbolic variables in f by entering

```
>>findsym(f)
   ans = n, x
```

The subs Command

- To evaluate an expression, use the subs command
  - subs(S, old, new)
- When your expression contains more than one variable, you can specify the variable for which you want to make the substitution. For example, to substitute the value $x = 3$ in the symbolic expression,
  - syms x y
  - f = x^2*y + 5*x*sqrt(y)
- subs(f, x, 3)
  - ans = 9*y+15*y^(1/2)
- subs(f, x, z) %Can substitute a new variable
  - ans = (z)^2*y + 5*(z)*sqrt(y)

Symbolic Functions

•All four of these functions can be used with either equations or expressions

collect(S)	Collects coefficients of S
collect(S,'v')	...with respect to the coefficient v
expand(S)	Expands S
factor (S)	Factors the expression

When used with equations, each side is treated separately

Example

```
>>expand((x-2)*(x-4))
ans= x^2-6*x+8
>>expand(cos(x+y))
ans= cos(x)*cos(y)-sin(x)*sin(y)

>>factor(cos(x)*cos(y)-sin(x)*sin(y) )
ans=cos(x+y)
```

poly2sym and sym2poly

- creates a polynomial from a vector

  poly2sym(u)

```
>> a=[1,3,2]
a =
     1     3     2
>> b=poly2sym(a)
b =
x^2+3*x+2
>>
```

- Extract the coefficients from a polynomial,

  sym2poly (v)

```
>> c=sym('5*x^2+3*x-2')
c =
5*x^2+3*x-2
>> d=sym2poly(c)
d =
     5     3    -2
>>
```

Equation Solving

- **solve(f)**
  - Assumes f is equal to 0
  - Solves for the symbolic variable
- **solve(f1,f2,f3…)**
  - Solves a system of equations
  - Can solve linear systems, as well as nonlinear systems

Symbolic Equation Solving

% Solve the system of equations $x^2+y^2=1$, $x+y = 1$

```
>>[x, y] = solve('x^2+y^2 = 1', 'x+y=1')

x =

   [ 1]
   [ 0]

y =

   [ 0]
   [ 1]
```

% the two solutions are:

```
>>s1 = [x(1), y(1)],  s2 = [x(2), y(2)]

s1 =

   [ 1, 0]

s2 =

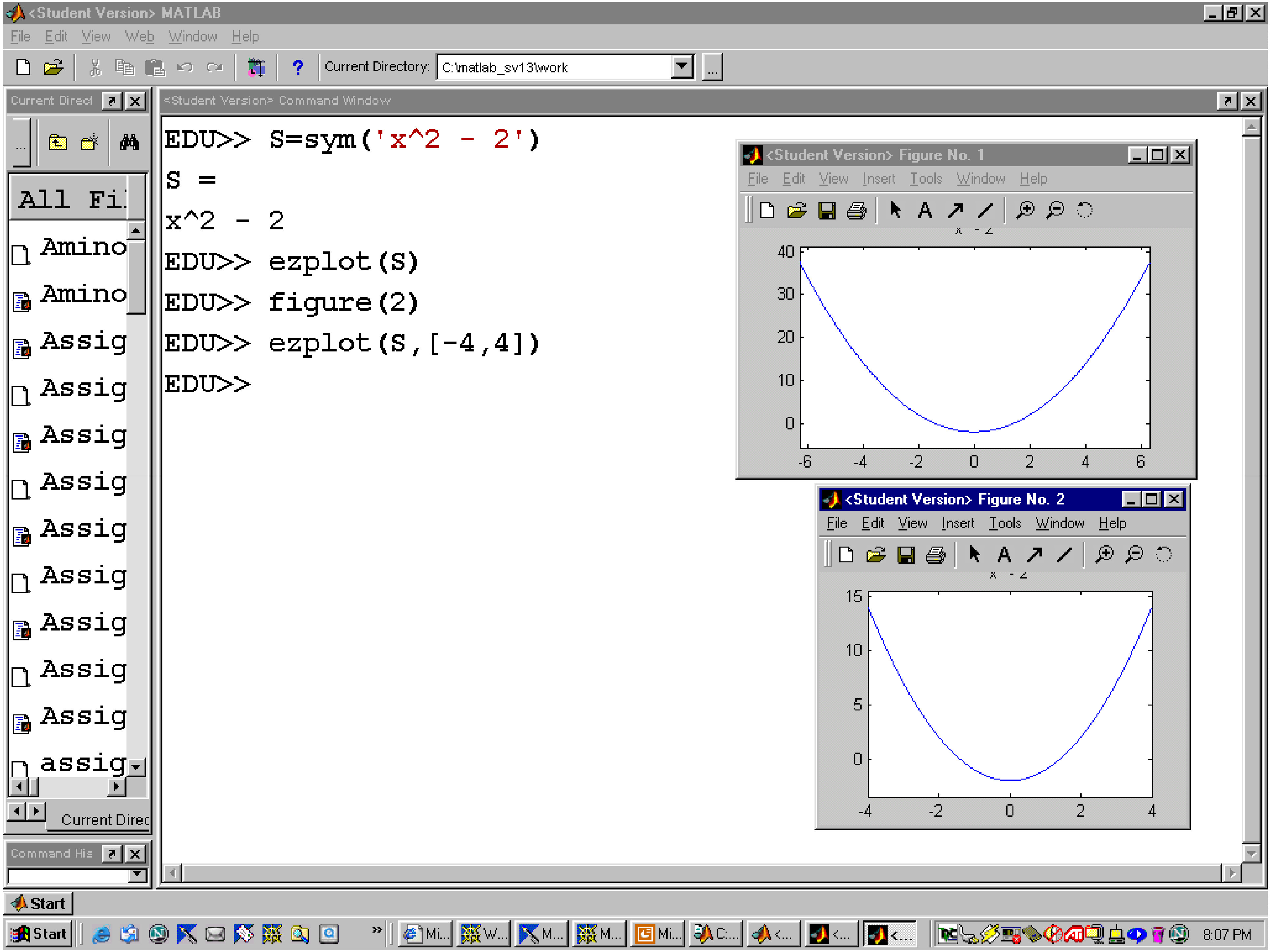
   [ 0, 1]
```

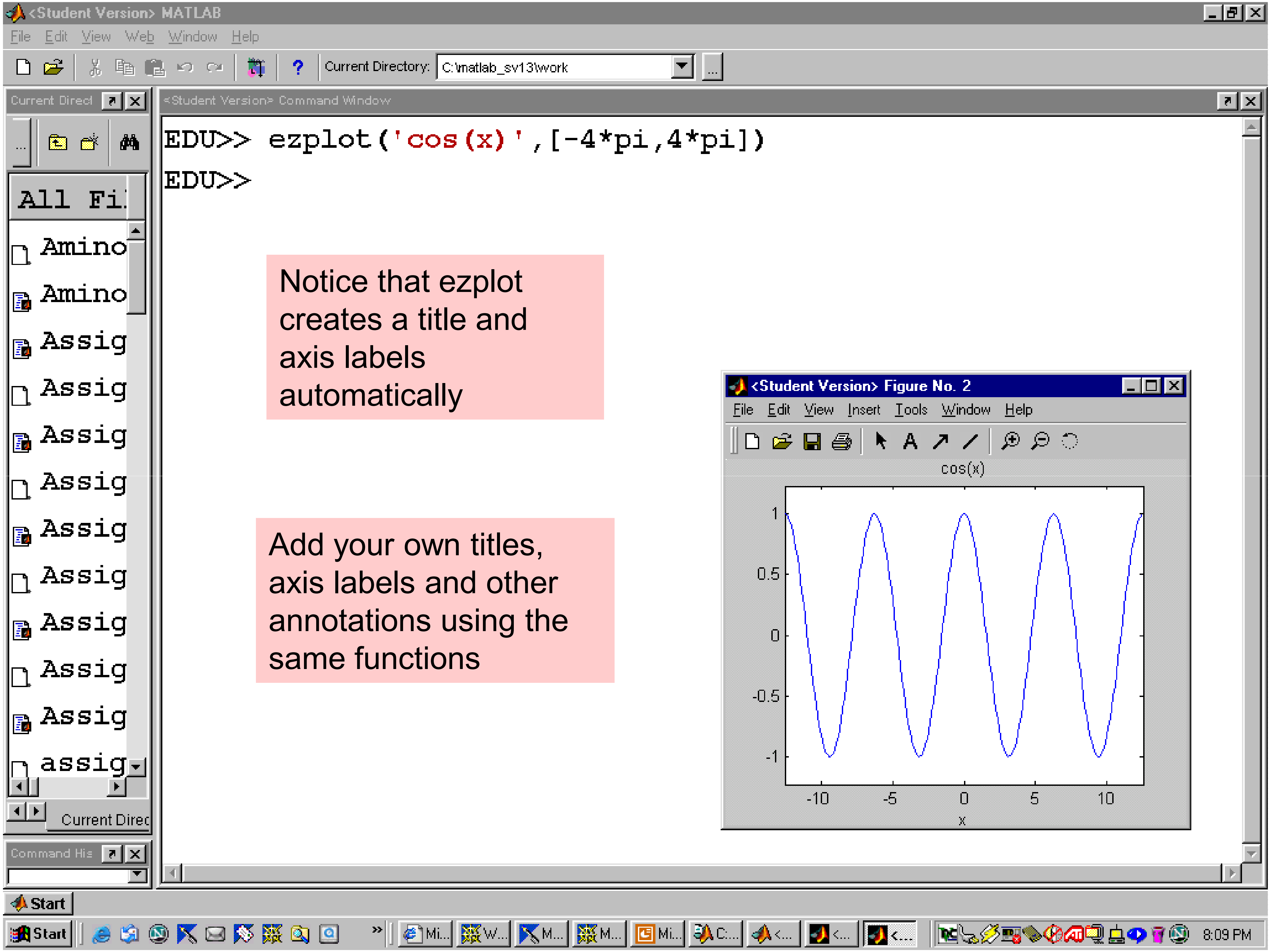
EZPlot

- Allows you to plot symbolic expressions

  ezplot(S)

  - Defaults to a range of -2π to +2π
- ezplot(S, [xmin, xmax])


<Student Version> MATLAB
File Edit View Web Window Help
Current Directory: C:\matlab_sv13\work
<Student Version> Command Window
EDU>> S=sym('x^2 - 2')
S =
x^2 - 2
EDU>> ezplot(S)
EDU>> figure(2)
EDU>> ezplot(S,[-4,4])
EDU>>
<Student Version> Figure No. 1
<Student Version> Figure No. 2
File Edit View Insert Tools Window Help
40
30
20
10
0
-6
-4
-2
0
2
4
6
15
10
5
0
-4
-2
0
2
4
Start
8:07 PM


<Student Version> MATLAB
File Edit View Web Window Help
Current Directory: C:\matlab_sv13\work
<Student Version> Command Window
EDU>> ezplot('cos(x)',[-4*pi,4*pi])
EDU>>
Notice that ezplot creates a title and axis labels automatically
Add your own titles, axis labels and other annotations using the same functions
<Student Version> Figure No. 2
File Edit View Insert Tools Window Help
cos(x)
1
0.5
0
-0.5
-1
-10
-5
0
5
10
x
8:09 PM

Derivative

- **>> syms x**
  >> f=x^3-cos(x);
  >> g=diff(f)
- Matlab returns:
- **g =**
  3*x^2+sin(x)
- Note that the command "**diff**" was used to obtain the derivative of function *f*.

- If there are more than one independent variable in a function, you should include the "intended" variable in the following format:
- **diff(f, x)** where *x* is the "intended" variable.
- **>> syms x y**
  >> f=x^2+(y+5)^3;
  >> diff(f,y)
- Matlab returns:
- **ans =**
  3*(y+5)^2

Integral

- To integrate function f(x,y) in the previous slide,
- **>> int(f,x)**
- Matlab returns:
- **ans =**

  1/3*x^3+(y+5)^3*x

- If we wish to perform the following definite integral: $$\int_0^{10} f(x,y)\,dy$$
- Matlab command entry:
- **>> int(f,y,0,10)**
- Matlab returns:
- **ans =**
  12500+10*x^2

Matrix Symbolic Calculation

```
>> syms a b c d e f g h
```

- Matrix **A** is then defined as:

```
>> A=[a b; c d]
>> B=[e f;g h]
>> C=A+B
   C =
   [ a+e, b+f]
   [ c+g, d+h]
```

- >> D=A*B
  D =
  [a*e+b*g, a*f+b*h]
  [c*e+d*g, c*f+d*h]
- >> a=1;b=2;c=3;d=4;e=5;f=6;e=7;f=8;g=9;h=0;
- >> eval(A)
- ans =
- 1 2
  3 4

>> eval(B)
ans =
7 8
9 0

- The inverse of A can be expressed symbolically:

```
>> D=inv(A)
 D =
 [ d/(a*d-b*c), -b/(a*d-b*c)]
 [ -c/(a*d-b*c), a/(a*d-b*c)]
```

- Numerically, D is expressed by

```
>> Dn=eval(inv(A))
Dn =
-2.0000 1.0000
1.5000 -0.5000
```

Summations:

- The “symsum” command is used Let us consider , where f(k) is a function of an integer k. The syntax to calculate this using MATLAB is syms k, symsum(f,k,m,n).

$$\sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{(2k+2)!}$$

- **>> clear, syms k x;**
- symsum((-1)^k*x^(2*k+1)/ sym('(2*k+1)!'),k,0,Inf)

Taylor series

- **>> clear, syms x; Tay_expx = taylor(exp(x),5,x,3)**

- This is exactly correct only at x = 3.
- **>> ezplot(Tay_expx), hold on, ezplot(exp(x)), hold off**

Single Differential Equation

dsolve

r = dsolve('eq1,eq2,...', 'cond1,cond2,...', 'v')

r = dsolve('eq1','eq2',...,'cond1','cond2',...,'v') symbolically solves the ordinary differential equation(s) specified by eq1, eq2,... using v as the independent variable and the boundary and/or initial condition(s) specified by cond1,cond2,....

$$\frac{dy}{dx} = x + y$$

In order to solve such an equation in Matlab,

>>dsolve(‘Dy=y+x’,x)

- **dsolve('D2y+a*y=0')**
- **ans = C1*cos(a^(1/2)*t)+C2*sin(a^(1/2)*t)**

More about Symbolic

Symbolic integral transforms:

Matlab can also obtain Fourier and Laplace transforms using these commands:

fourier(f(u),u,v) - Fourier transform of f(u) to F(v)

Laplace(f(t),t,s) - Laplace transform of f(t) to L(s)

ifourier - Inverse Fourier transform.

ilaplace - Inverse Laplace transform.

More About Plotting
(Double y-axis plots)

- The plotting command **plotyy** allows one to plot two different data sets (with the same range in the dependent variable) on the same plot but with different scales for the -axes.

```
% Double y axes plot; Also demonstrate subplots
x = -10:.1:10; % define points for independent variable
y1 = (sin(x)./x).^2; % "dot" operator squares individual elements
                     % and not the whole array
y2 = 3*sin(abs(x));
subplot(2,1,1); % Divide figure into 1x2 array of plots and start with #1
plot(x,y1,x,y2); % Plot both data sets an same scale
subplot(2,1,2); % Put next plot in second slot
plotyy(x,y1,x,y2)
```

Histograms

- creates histograms

  hist

```
% Histogram plot clear all; close all;
x = -3:.2:3;
y = randn(1000,1);
% Generate some random (Gaussian) data
hist(y,x)
```

- MATLAB

Plots with error bars

- Often when we are plotting data we can estimate the error in each point and the magnitude of error may vary from point to point.

```
% Plotting with error bars
close all; clear all;
x = linspace(0,pi,30);
y = 1-x.^2/2+x.^4/(4*3*2);
error = cos(x)-y;
errorbar(x,y,error)
```

- MATLAB

Polar Plots

- polar[theta,radius,’line specifiers’]
- (optional)

Vectors

Eq. $r = 3\cos^2(0.5\theta) + \theta \text{ for } 0 \le \theta \le 2\pi$

```
t=linspace(0,2*pi,200),
r=3*cos(0.5*t).^2+t;
polar(t,r)
```

Plots with special graphics

- Vertical Bar Plot

eg. yr=[1988:1994];

sle=[8 10 20 22 18 24 27]

bar(yr,sle,’r’)

xlabel(‘year’)

ylabel(‘Sales’)

Horizontal Bar Plot

```
yr=[1988:1994];
sle=[8 10 20 22 18 24 27]
barh(yr,sle)
xlabel('Sales')
ylabel('year')
```

Stairs Plot

```
yr=[1988:1994];
sle=[8 10 20 22 18 24 27]
Stairs(yr,sle)
```

Stem Plot

```
yr=[1988:1994];
sle=[8 10 20 22 18 24 27]
stem(yr,sle)
```

Pie Plot

```
grd=[11,18,26,9,5];
Pie(grd)
Title('class grades')
```