Table of Contents

Examples of using Matlab's Symbolic Toolbox

```
% ReactorLab.net
```

ROOTS of nonlinear equation with symbolic variables

```
clear; clc
syms x t k1 k2
f = (k1 + k2*x^2) * t == 0
r = solve(f,x);
s = length(r);
fprintf('there are %i roots of f(x) \n',s)
r
```

```
f =

t*(k2*x^2 + k1) == 0

there are 2 roots of f(x)

r =

  (-k1)^(1/2)/k2^(1/2)
 -(-k1)^(1/2)/k2^(1/2)
```

ROOTS of nonlinear equation with constants

```
clear; clc
syms x
k1 = 0.004; % class(k1) = double
k2 = 0.25;
f = (k1*x^0.5 + k2*x^2) * 10 == 0
xSolns = solve(f,x);
s = length(xSolns);
fprintf('there are %i solutions for x, these are: \n',s)
```

```
xSolns

% to convert symbolic answer to floating point there are
% three alternative built-in functions: vpa, double, cast
xSolns1 = double(xSolns)
xSolns2 = cast(xSolns,'double')
xSolns3 = vpa(xSolns)
```

```
f =

(5*x^2)/2 + x^(1/2)/25 == 0

there are 3 solutions for x, these are:

xSolns =

                                                  0
                           ((-1)^(2/3)*2^(2/3))/25
 ((-1)^(2/3)*2^(2/3)*((3^(1/2)*1i)/2 + 1/2)^2)/25


xSolns1 =

   0.0000 + 0.0000i
  -0.0317 + 0.0550i
  -0.0317 - 0.0550i


xSolns2 =

   0.0000 + 0.0000i
  -0.0317 + 0.0550i
  -0.0317 - 0.0550i


xSolns3 =


      0
 - 0.031748021039363989495034112785446 +
 0.054989185479944105055339167787544i
 - 0.031748021039363989495034112785446 -
 0.054989185479944105055339167787544i
```

DERIVATIVE with symbolic variables

```
clear; clc
syms x t k1 k2 k3
f = k2*x/(k1*x^0.5+k3*x^2)
dfdx = diff(f,x)
```

```
f =

(k2*x)/(k1*x^(1/2) + k3*x^2)


dfdx =

k2/(k1*x^(1/2) + k3*x^2) - (k2*x*(2*k3*x + k1/(2*x^(1/2))))/
(k1*x^(1/2) + k3*x^2)^2
```

DERIVATIVE with constants

```
clear; clc
syms x
k1 = 0.3; % class(k1) = double
k2 = 0.25;
f = k1*x/(k1*x^0.5+k2*x^2)
dfdx = diff(f,x)


f =

(3*x)/(10*(x^2/4 + (3*x^(1/2))/10))


dfdx =

3/(10*(x^2/4 + (3*x^(1/2))/10)) - (3*x*(x/2 + 3/(20*x^(1/2))))/
(10*(x^2/4 + (3*x^(1/2))/10)^2)
```

DERIVATIVE with constants and evaluate at specified value

```
clear; clc
syms x
k1 = 0.3; % class(k1) = double
k2 = 0.25;
f = k1*x/(k1*x^0.5+k2*x^2)
dfdx = diff(f)
% use built-in function subs (substitute) to substitute value(s)
% for x into symbolic result dfdx
disp('evaluate df/dx at x = 0.5')
dfdx = subs(dfdx,x,0.5)

% to convert symbolic answer to floating point there are
% three alternative built-in functions: vpa, double, cast
dfdx = double(dfdx)


f =
```

```
(3*x)/(10*(x^2/4 + (3*x^(1/2))/10))


dfdx =

3/(10*(x^2/4 + (3*x^(1/2))/10)) - (3*x*(x/2 + 3/(20*x^(1/2))))/
(10*(x^2/4 + (3*x^(1/2))/10)^2)

evaluate df/dx at x = 0.5

dfdx =

3/(10*((3*2^(1/2))/20 + 1/16)) - (3*((3*2^(1/2))/20 + 1/4))/
(20*((3*2^(1/2))/20 + 1/16)^2)


dfdx =

    0.1733
```

INTEGRAL, indefinite

```
clear; clc
syms x
f = 1/(1-x) % leave off the dx in f = dx * 1/(1-x)
disp('integrate f(x)dx')
fint = int(f,x)


f =

-1/(x - 1)

integrate f(x)dx

fint =

-log(x - 1)
```

INTEGRAL, definite

```
clear; clc
syms x
f = 1/(1-x) % leave off the dx in f = dx * 1/(1-x)
disp('integrate f(x)dx from x = 0 to x = 0.5')
fint = int(f, x, 0, 0.5)

% to convert symbolic answer to floating point there are
% three alternative built-in functions: vpa, double, cast
fint = double(fint)
```

```
f =

-1/(x - 1)

integrate f(x)dx from x = 0 to x = 0.5

fint =

log(2)


fint =

    0.6931
```

Published with MATLAB® R2016a