TOPIC 9
COMPUTER APPLICATIONS
FOR HANDLING SYMBOLIC EXPRESSION

Symbolic Objects

A symbolic object is a data structure that stores a string representation of the symbol. The actual computations involving symbolic objects are performed primarily by Maple, mathematical software developed by Waterloo Maple, Inc.

The following example illustrates the difference between a standard MATLAB data type, such as double, and the corresponding symbolic object.

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

Using symbolic object:

```
>>a = sqrt(sym(2))
ans=
2^(1/2)
```

The numerical value of a symbolic object can be obtained using the double command:

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

Create a fraction involving symbolic objects, MATLAB records the numerator and denominator:

```
>>sym(2)/sym(5)
ans =
        2/5
```

The Symbolic Math Toolbox enables you to perform a variety of symbolic calculations that arise in mathematics and science.

Creating Symbolic Variables and Expressions

In general, you can use sym or syms to create symbolic variables. We recommend you use syms because it requires less typing

Using symbolic variable to present the golden ratio $\rho = \frac{1+\sqrt{5}}{2}$

```
>>rho = sym('(1 + sqrt(5))/2')
```

Now you can perform various mathematical operations on rho.

```
>>f = rho^2 - rho - 1
f =
        (1/2+1/2*5^(1/2))^2-3/2-1/2*5^(1/2)
```

Simplify this answer by entering:

```
>>simplify(f)
```

Now suppose you want to study the quadratic function $f = ax^2 + bx + c$.
The command is:

```
>>f = sym('a*x^2 + b*x + c')
```

In this case, the Symbolic Math Toolbox does not create variables corresponding to the terms of the expression **a,b,c,** and **x** to perform further operation using the symbol.
A better alternative is to enter the commands

```
>>a = sym('a')
>>b = sym('b')
>>c = sym('c')
>>x = sym('x')
```

or simply: `>> syms a b c x`
Then enter : `>>f = a*x^2 + b*x + c` or `>>f = sym('a*x^2 + b*x + c')`

The subs Command

To substitute the value x = 2 in the symbolic expression, f = 2*x^2 - 3*x + 1

```
>>subs(f,2)
ans =
    3
```

If the expression contains more than one variable, specify the variable for which you want to make the substitution.

```
>> syms x y
>>f = x^2*y + 5*x*sqrt(y)
>> subs(f, x, 3)          %substitute the value x = 3
ans =
9*y+15*y^(1/2)
```

To substitute y = 3:

```
>>subs(f, y, 3)
ans =
3*x^2+5*x*3^(1/2)
```

The Default Symbolic Variable

For one-letter variables, default variable is the letter closest to x in the alphabet. If there are two letters equally close to x, default variable is the one that comes later in the alphabet.

In the preceding example, subs(f, 3) returns the same answer as subs(f, x, 3).

Use the findsym command to determine the default variable.

```
>>syms s t
>>g = s + t;
>>findsym(g,1)
```

returns the default variable:

```
ans =
        t
```

Using the Symbolic expressions, we can do the basic operations of calculus, i.e: Differentiation, Limits, Integration, and Symbolic Summation.

Differentiation

The command **diff(f)** differentiates f with respect to x.

Example:

```
>>syms x
>>f = sin(5*x)
ans =
5*cos(5*x)
```

Another example:

```
>> g = exp(x)*cos(x)  %exp(x) denotes e^x
>> diff(g)
ans =
        exp(x)*cos(x)-exp(x)*sin(x)
```

To take the second derivative of g, enter:

```
>>diff(g,2)
ans =
        -2*exp(x)*sin(x)
```

You can get the same result by taking the derivative twice:

```
>>diff(diff(g))
```

To take the derivative of a constant, you must first define the constant as a symbolic expression. For example, entering

```
>>c = sym('5');
>>diff(c)
```

returns ans =

0

If you just enter diff(5)

MATLAB returns ans =

[]

because 5 is not a symbolic expression.

Derivatives of Expressions with Several Variables

The diff command then calculates the partial derivative of the expression with respect to that variable. For example, given the symbolic expression

```
>>syms s t
>>f = sin(s*t)
ans =
        cos(s*t)*s
```

To differentiate f with respect to the variable s, enter >>diff(f,s)

If you do not specify a variable to differentiate with respect to, MATLAB chooses a default variable

To calculate the second derivative of f with respect to t, enter

```
>>diff(f,t,2)
ans =
        -sin(s*t)*s^2
```

Note that diff(f,2) returns the same answer because t is the default variable.

The diff function can also take a symbolic matrix as its input. In this case, the differentiation is done element-by-element. Consider the example:

```
>>syms a x
>>A = [cos(a*x),sin(a*x);-sin(a*x),cos(a*x)]
>>diff(A)
ans =
        [ -sin(a*x)*a,  cos(a*x)*a]
        [ -cos(a*x)*a, -sin(a*x)*a]
```

A table summarizing diff follows.

Mathematical Operator	MATLAB Command
$\frac{df}{dx}$	`diff(f)` or `diff(f,x)`
$\frac{df}{da}$	`diff(f,a)`
$\frac{d^2f}{db^2}$	`diff(f,b,2)`

Limits

The fundamental idea in calculus is to make calculations on functions as a variable "gets close to" or approaches a certain value. Recall that the definition of the derivative is given by a limit

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

provided this limit exists. The Symbolic Math Toolbox enables you to calculate the limits of functions directly. The commands

```
>>syms h n x
>>limit( (cos(x+h) - cos(x))/h,h,0 )
ans =
    -sin(x)

>>limit( (1 + x/n)^n,n,inf )
ans =
    exp(x)
```

One-Sided Limits

Calculate the limit of x/|x|, as x approaches 0 from the left or from the right.

To calculate the limit as x approaches 0 from the left, enter

```
>>limit(x/abs(x),x,0,'left')
ans =
         -1
```

To calculate the limit as x approaches 0 from the right, enter

```
>>limit(x/abs(x),x,0,'right')
ans =
         1
```

Since the limit from the left does not equal the limit from the right, the two- sided limit does not exist. In the case of undefined limits, MATLAB returns NaN (not a number). For example,

```
>>limit(x/abs(x),x,0)
ans =
         NaN
```

Observe that the default case, limit(f) is the same as limit(f,x,0).

This table present the options for the limit command, where f is a function of the symbolic object x.

Mathematical Operation	MATLAB Command
$\lim_{x \to 0} f(x)$	`limit(f)`
$\lim_{x \to a} f(x)$	`limit(f,x,a)` or `limit(f,a)`
$\lim_{x \to a-} f(x)$	`limit(f,x,a,'left')`
$\lim_{x \to a+} f(x)$	`limit(f,x,a,'right')`

Integration

If f is a symbolic expression, then

>> int(f)

attempts to find another symbolic expression, F, so that diff(F) = f.

That is, int(f) returns the indefinite integral or antiderivative of f (provided one exists in closed form). Similar to differentiation,

>>int(f,v)

uses the symbolic object v as the variable of integration, rather than the variable determined by findsym. See how int works by looking at this table.

Mathematical Operation	MATLAB Command
$\int x^n dx = \frac{x^{n+1}}{n+1}$	`int(x^n)` or `int(x^n,x)`
$\int_0^{\pi/2} \sin(2x)dx = 1$	`int(sin(2*x),0,pi/2)` or `int(sin(2*x),x,0,pi/2)`

$g = \cos(at+b)$
$\int g(t)dt = \sin(at+b)/a$
	`g = cos(a*t + b)`
	`int(g)` or
	`int(g,t)`

$\int J_1(z)dz = -J_0(z)$	`int(besselj(1,z))` or `int(besselj(1,z),z)`

Symbolic Summation

You can compute symbolic summations, when they exist, by using the symsum command. For example, the p-series $1 + \frac{1}{2^2} + \frac{1}{3^2} + \ldots$ sums to $\pi^2/6$, while the geometric

series $1 + x + x^2 + \ldots$ sums to $|x| < 1$, provided . Three summations are demonstrated below:

```
>> syms x k
>>s1 = symsum(1/k^2,1,inf)
>>s2 = symsum(x^k,k,0,inf)
s1 =
        1/6*pi^2
s2 =
        -1/(x-1)
```

Simplifications

There are several functions that simplify symbolic expressions and are used to perform symbolic substitutions, ie.: pretty, simplify, collect, expand.

- **pretty**

```
>>syms x
>>f = x^3-6*x^2+11*x-6
>>g = (x-1)*(x-2)*(x-3)
```

Here are their pretty printed forms, generated by pretty(f), pretty(g)

```
   3    2
x - 6 x  + 11 x - 6

(x - 1) (x - 2) (x - 3)
```

- **Collect**

The statement

```
>> collect(f)
```

views f as a polynomial in its symbolic variable, say x, and collects all the coefficients with the same power of x. A second argument can specify the variable in which to collect terms if there is more than one candidate. Here are a few examples.

f	collect(f)
(x-1)*(x-2)*(x-3)	x^3-6*x^2+11*x-6
x*(x*(x-6)+11)-6	x^3-6*x^2+11*x-6
(1+x)*t + x*t	2*x*t+t

- **Expand**

The statement

```
>>expand(f)
```

distributes products over sums and applies other identities involving functions of sums as shown in the examples below.

f	expand(f)
a*(x + y)	a*x + a*y
(x-1)*(x-2)*(x-3)	x^3-6*x^2+11*x-6
x*(x*(x-6)+11)-6	x^3-6*x^2+11*x-6
exp(a+b)	exp(a)*exp(b)
cos(x+y)	cos(x)*cos(y)-sin(x)*sin(y)
cos(3*acos(x))	4*x^3-3*x

Solving Algebraic Equations

If S is a symbolic expression,

>>solve(S)

attempts to find values of the symbolic variable in S (as determined by findsym) for which S is zero. For example,

```
>> syms a b c x
>>S = a*x^2 + b*x + c;
>>solve(S)
ans =
        [1/2/a*(-b+(b^2-4*a*c)^(1/2))]
        [1/2/a*(-b-(b^2-4*a*c)^(1/2))]
```

If you want to solve S for b, use the command

```
>>b = solve(S,b)
b =
        -(a*x^2+c)/x
```

Note that these examples assume equations of the form f(x)=0. If you need to solve equations of the form f(x)=q(x), you must use quoted strings. In particular, the command

```
>>s = solve('cos(2*x)+sin(x)=1')
s =
        [      0]
        [     pi]
        [ 1/6*pi]
        [ 5/6*pi]
```

Exercises:

Read more example in Matlab Help, Symbolic Math Toolbox.

Try the symbolic expression using maple.