Symbolic Math in Matlab

Matlab has a powerful symbolic math ability. Rather than making calculations on **known** numbers, we can make calculations on symbolic expressions. For example, what is the limit as x approaches inf of 1 + 1/2^1 + 1/2^2 + 1/2^3...+1/2^n ? Matlab can tell us. What is the integral of x^3 for any x? Matlab can tell us.

Symbolic Math in Matlab

Matlab allows you to create symbolic math expressions. This is useful when you don't want to immediately compute an answer, or when you have a math "formula" to work on but don't know how to "process" it.

Matlab allows symbolic operations several areas including:

- Calculus
- Linear Algebra
- Algebraic and Differential Equations
- Transforms (Fourier, Laplace, etc)

The key function in Matlab to create a symbolic representation of data is: **sym()** or **syms** if you have multiple symbols to make.

Below is an example of creating some symbolic fractions and square roots:

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

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

>> 2 / 5
ans =
  0.4

>> 2/5 + 1/3
ans =

0.7333

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

>> sym(2) / sym(5) + sym(1) / sym(3)
ans =
11/15
```

Defining Symbolic Expressions

We can define symbolic functions using the sym command and syms command. Here is an example of creating a symbolic function for (a*X^2) + (b*x) + c:

```
>> syms a b c x % define symbolic math variables

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

From now on we can use the f symbol to represent the given function.

Evaluation of Symbolic Expressions

The keyfunction **subs** (which stands for substitute) is used to replace symbolic variables with either new symbolic variables or with acutal values. The syntax for the function is: subs(symbolic_function, list_of_symbols, list_of_values). Here is an example:

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

>> subs(f,x,5)

ans =
25 * a + 5 * b + c

>> subs(f,[x a b c],[5 1 2 3])
ans =
   38
```

Plotting Symbolic Function

In Matlab, we can plot a symbolic function over one variable by using the **ezplot** function. Here is an example:

```
>> y = sin(x)

y =

sin(x)

>> ezplot(y)
```

If you want to see something cool, try:

```
>> f = sin(x);
>> ezsurf(f);
```

Now try:

```
>> f = sin(x);
>> g = cos(y);
>> ezsurf(f+g);
```

Or really cool!

```
>> ezsurf( 'real(atan(x+i*y))' );
```

To set the bounding values of the variables, you can use:

```
>> ezplot(y, [ -5, 10 ]);  % from -5 < x < 10
>> ezsurf(z,[[1 2] [5 7]); % x from 1 to 2, y from 5 to 7
```

Or plotting a polynomial equation:

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

        >> ezplot(subs(f,[a b c],[1 2 3]));
```

Integration and Derivation

Matlab can also compute many integrals and derivatives that you might find in Calculus or many advanced engineering courses.

The key functions are **int** for integration and **diff** for derivation.

Differentiation

```
>> syms x;
f = sin(5*x)

>> f =
sin(5*x)

>>diff(f)

ans =
5*cos(5*x)
```

2nd Derivative

To take the 2nd (or greater) derivative of an equation, we use:

```
>> f = x^3

f =

x^3


>> diff(f) % 1st derivative

ans =

3*x^2


>> diff(f,2) % 2nd derivative

ans =

6*x


>> diff(f,3) % 3rd derivative

ans =

6

>> diff(f,4) % 4th derivative

ans =

0
```

Partial Differential Equations

Sometimes you have multiple variables in an expression. If we want to compute the derivative of the function with respect to one variable we can use a second parameter to the diff function:

```
>> syms x t;
f = sin( x * t )

>>diff(f,t)  % derivative of f with respect to t

ans =
cos(x*t)*t
```

Integration

By using the **"int"** function, in the same way we use the **diff** function, we can ask Matlab to do symbolic integration for us.

Warning: *Do not confuse the int function in Matlab with the integer (int) data type in C or the int8, int16, int32 data types in Matlab.*

```
>> syms x t;
f =  x * x ;

>>int(f)

ans =
1/3*x^3
```

Definite Integrals

As you (should) know, a definite integral represents the area under a curve from a given start point to a given end point. What if we want to know how much area is under the curve f(x) = x^3; from 0 to 10, from -10 to 10, from -10 to 0?

```
>> int(x^3,0,10)

ans =

2500


>> int(x^3,-10,10)

ans =

0
```

```
>> int(x^3,-10,0)

ans =

-2500
```

Summations

You can use Matlab to tell you the sum of a series of equations, such as:

1 + 1/2^2 + 1/3^2 + 1/4^2 + ... + 1/N^2

From N = 1 to inf (or from value1 to value2)

```
>> syms x k
>> s1 = symsum(1/k^2,1,inf)

s1 =

1/6*pi^2


>> s2 = symsum(x^k,k,0,inf)

s2 =

-1/(x-1)
```

Mathematical Limits

Every wonder what happens if you divide infinity by infinity? Well depending on how those values were created, you can get some interesting results. For example, what is the limit of x/ x*x as X approaches infinity?

```
>> limit( 1 / x )  % with no params, by definition, as x approaches 0

ans =

NaN

>> limit(x / x^2, inf) % as x approaches inf

ans =

0

>> limit(sin(x) / x) % as x approaches 0
```

```
ans =

????
```

Expand Function

The expand function will "expand" a formula by doing basic symbolic math where possible.

```
syms x;

>> f1 = (x+5)*(x+5);

f1 =

(x+5)^2


>>  expand(f1)

ans =

x^2+10*x+25
```

Simplify Function

When you have a complex evaluated **symbolic**expression, such as: (sin(x)^2 + cos(x)^2), you can use the **simplify** function to ask matlab to try and simplify it to a less complex term:

```
simplify(sin(x)^2 + cos(x)^2)
ans =
   1
```

"Pretty" Printing Symbolic Functions

When you want to print a symbolic function to make it easier for the user of the program to read, you can use the "pretty" function. Here is an example:

```
     f = sin(x)^2 + cos(x)^2;

     pretty( f )

                            2         2
                      sin(x)  + cos(x)
```

"Taylor" Command

If you would like to create a taylor series, you can use the "taylor" function.

```
  >> f = taylor(log(1+x))

  f =

  x-1/2*x^2+1/3*x^3-1/4*x^4+1/5*x^5
```

Known Bad Variable Names

A few years ago Matlab "upgraded" their symbolic library. When they did so they, "broke" the ability to use any arbitrary variable name. For example, the symbol D (capitol D) is invalid in some cases. For example:

```
int('A*x^3+B*x^2+C*x+D')
Warning: Explicit integral could not be found.

ans =

int(C*x + A*x^3 + B*x^2 + D, x)

% BUT

syms A B C D x
int(A*x^3+B*x^2+C*x+D)

ans =

(A*x^4)/4 + (B*x^3)/3 + (C*x^2)/2 + D*x
```

To fix this problem, append an _ (underscore) to the variable

The following symbols are known

Back to Topics List