Symbolic Mathematics

Dr. Mihail

November 20, 2018

Overview

Symbolic

So far in this course we dealt with MATLAB variables that were placeholders for numeric types (e.g., scalars, vectors, matrices), with one exception, anonymous functions: `f = @(x) ...`

Overview

Symbolic

So far in this course we dealt with MATLAB variables that were placeholders for numeric types (e.g., scalars, vectors, matrices), with one exception, anonymous functions: `f = @(x) ...`

We will now introduce the **symbolic** MATLAB data type. This is a non-numeric data type, used by the MATLAB Symbolic Math Toolbox to solve equations analytically, integrate and differentiate.

Symbolic Math

Symbolic Variables

To create three symbolic variables x, y and z, the following syntax is used:

```
>> syms x y z
```

Notice the lack of commas.

```
>> whos
  Name        Size              Bytes  Class     Attributes
  x           1x1                 112  sym
  y           1x1                 112  sym
  z           1x1                 112  sym
```

Symbolic Expressions

Symbolic expressions are created using symbolic variables. For example:

```
>> syms x y z
>> f = x.^2 + y - z


f =


x^2 + y - z
```

It can also be created using the `sym` function:

```
f = sym('x.^2 + y - z')
```

Utilities

Substitution

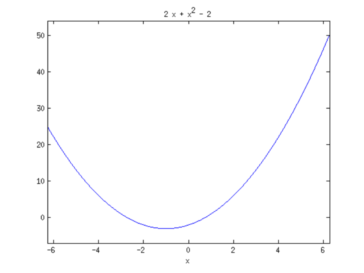
Symbolic expressions can be changed. One useful operation is substitution. The MATLAB function `subs` does that. The syntax is as follows:
`subs(S, old, new)`.
For example:

```
>> f = sym('x^2 + y - z');
>> subs(f, 'x', 'a')


ans =


a^2 + y - z
```

Plotting

MATLAB symbolic toolbox provides a function to plot symbolic expressions of one variable: `ezplot(S)`, where `S` is the symbolic expression. Example:

```
>> f = sym('x^2 + 2*x - 2');
>> ezplot(f)
```



Utilities

Expansion

MATLAB symbolic toolbox provides functions to manipulate algebraic expressions. For example `expand(S)`:

```
>> f = sym('(x + 2) * (x + 1)');
>> expand(f)

ans =

x^2 + 3*x + 2

```

performs an expansion of `f`.

Utilities

Factorization

`factor(S)`:

```
>> f = sym('x^2 + 3*x + 2');
factor(f)


ans =


(x + 2)*(x + 1)
```

performs the factorization of `f`.

Utilities

Simplification

`factor(S)`:

```
>> syms  x a b c
>> simplify(exp(c*log(sqrt(a+b))))

ans =

(a + b)^(c/2)
```

performs the simplification of `f`.

Utilities

Pretty

factor(S):

```
>> syms  x a b c
>> S = simplify(exp(c*log(sqrt(a+b))))
S =

(a + b)^(c/2)

>> pretty(S)
ans =
        c/2
(a + b)
>> S = sym('2*x^2 + 3*x - 2');
>> pretty(S)
   2
2 x  + 3 x - 2
```

Equation Solving

The `solve` function is used to solve equations. For example:

```
 >> S = sym('x^2 + 2 = 0');
>> solve(S)

ans =

  i
 -i
```

Two complex solutions.

Equation Solving

```
>> S = sym('sin(x) = 2*pi');
>> solve(S)

ans =

      asin(2*pi)
 pi - asin(2*pi)
```

Infinite number of solutions, since $a \in \mathcal{R}$.

Utilities

Differentiation

The `diff` function performs analytic differentiation.

```
>> S = sym('sin(x)');
>> diff(S)

ans =

cos(x)
```

Utilities

Differentiation

Another example:

```
>> S = sym('sin(x) + cos(x) - 2*x^2 + 2');
>> diff(S)

ans =

cos(x) - 4*x - sin(x)
```

Utilities

Integration

The `int(S)` function returns the indefinite integral of a symbolic expression `S`.

```
>> S = sym('cos(x)');
>> int(S)

ans =

sin(x)
```

$\int cos(x) = sin(x)$

Utilities

Integration

The `int(S, 1, 2)` function returns the definite integral of a symbolic expression `S`, evaluated in the range $[1, 2]$.

```
>> S = sym('cos(x)');
>> int(S)

ans =

sin(x)
```

$\int cos(x, 1, 2)|_1^2 = sin(2) - sin(1)$