Matlab’s Symbolic Toolbox

ChEn 1703
Some Capabilities of the Symbolic Toolbox

- Algebra - manipulate expressions
- Solve systems of linear equations
  - Other linear algebra functions also.
- Solve systems of nonlinear equations
- Calculus - integration & differentiation of expressions
- Solve differential equations
- Fourier & Laplace transforms
- “help symbolic”

NOTE: The symbolic toolbox is included in the student version. It must be purchased separately in general.
Building Symbolic Variables and Expressions

- Normal variables in MATLAB have a value (number, string, etc)
- Symbolic variables are special, and are declared differently

```matlab
syms a b c x;
```

- Symbolic expressions may also be created:

```matlab
syms x y; % declare symbolic variables.
x=y^2; % build a symbolic expression.
x=sym('y^2');
```

- You can do arithmetic with symbolic expressions

```matlab
syms a b c d x;
y1=a*x^2+b*x+c;
y2=d*x^2;
y3=y1+y2;
```

```matlab
y1=sym('a*x^2+b*x+c');
y2=sym('sin(y1)');
```
Some Useful Commands

<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>syms</code></td>
<td>Declare a symbolic variable</td>
<td><code>syms a b c x;</code></td>
</tr>
<tr>
<td><code>pretty(f)</code></td>
<td>Print out the symbolic expression <code>f</code> with nice formatting</td>
<td><code>pretty( sqrt(a*x)*exp(c) );</code></td>
</tr>
<tr>
<td><code>collect(f,v)</code></td>
<td>Collect coefficients in <code>f</code> in terms of powers of <code>v</code>.</td>
<td><code>syms a b c x; collect(a*x^2+c+x^2+b*x+b+x/2);</code></td>
</tr>
<tr>
<td><code>expand(f)</code></td>
<td>Expands polynomials, trig functions, exponential functions.</td>
<td><code>syms a b c x; expand( exp(a+b) ); expand( (a*x+c)^2 );</code></td>
</tr>
</tbody>
</table>
| `factor(f)` | Obtain the prime factors of a given number. If `f` is a polynomial, this factors the polynomial. | `factor(90);`  
`factor(x^3-6*x^2+11*x-6)` |
| `simplify(f)` | Attempts to simplify the expression. Not always very useful. | `syms x; simplify( sin(x)^2+cos(x)^2 )` |
| `subs(f,t,v)` | Substitute `v` for `t` in the expression `f`. | `syms x a b c; subs( sin(pi*x), x, (a+b)/c )` |

Some Useful Commands

- `exp(a + b) → exp(a) exp(b)`
- `(ax^1 + c)^2 → ax^2 + 2acx + c^2`
- `(x - 1)(x - 2)(x - 3)`
- `sin^2(x) + cos^2(x) → 1`
- `sin(πx) → sin \left( \frac{π(a + b)}{c} \right)`
Simplification

\[ y = x^3 - 6x^2 + 11x - 6 \]
\[ = (x - 1)(x - 2)(x - 3) \]
\[ = x(x(x - 6) + 11) - 6 \]

Which form of “y” do you want?
It depends on the situation.

```
syms x;
f = x^3-6*x^2+11*x-6;
g = (x-1)*(x-2)*(x-3);
h = x*(x*(x-6)+11)-6;
pretty( collect(f) );
pretty( collect(g) );
pretty( collect(h) );
```

```
syms x;
f = x^3-6*x^2+11*x-6;
g = (x-1)*(x-2)*(x-3);
h = x*(x*(x-6)+11)-6;
pretty( factor(f) );
pretty( factor(g) );
pretty( factor(h) );
```

```
syms x;
f = x^3-6*x^2+11*x-6;
g = (x-1)*(x-2)*(x-3);
h = x*(x*(x-6)+11)-6;
pretty( horner(f) );
pretty( horner(g) );
pretty( horner(h) );
```

“Horner” form is also referred to as “nested”

`simple` - tries to find the shortest representation of the expression.
Substitutions: subexpr

\[ A x = b \]

\[ A^{-1} A x = A^{-1} b \]

\[ I x = A^{-1} b \]

\[ x = A^{-1} b \]

Linear system

Multiply by the “inverse” of \( A \).

\[ A^{-1} A = I \]

\[ I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \]

Solution for \( x \).

\[
A = \begin{bmatrix}
a_{11} & a_{12} & a_{13} \\
a_{21} & a_{22} & a_{23} \\
a_{31} & a_{32} & a_{33}
\end{bmatrix}
\]

subexpr will replace the \%1 generated by pretty.

\[
A^{-1} = \begin{bmatrix}
a_{22} a_{33} - a_{23} a_{32} & - a_{12} a_{33} + a_{13} a_{32} & a_{12} a_{23} - a_{13} a_{22} \\
- a_{21} a_{33} + a_{23} a_{31} & a_{11} a_{33} - a_{13} a_{31} & - a_{11} a_{23} + a_{13} a_{21} \\
a_{21} a_{32} - a_{22} a_{31} & - a_{11} a_{32} + a_{12} a_{31} & - a_{11} a_{22} + a_{12} a_{21}
\end{bmatrix}
\]

\[ s = a_{11} a_{22} a_{33} - a_{11} a_{23} a_{32} - a_{21} a_{12} a_{33} + a_{21} a_{13} a_{32} + a_{31} a_{12} a_{23} - a_{31} a_{13} a_{22} \]
Create a matrix “B” and solve for its inverse.

\[
\text{BI} = \text{inv}(B);
\]

\[
\text{B} = \text{sym}('[b_{11} \ b_{12}; b_{21} \ b_{22}]');
\]

\[
\text{s} = \text{sym}('b_{11}b_{22} - b_{12}b_{21}');
\]

Substitute the symbol ‘s’ for the denominator in each element of \(B^{-1}\).
More with **subs**

```matlab
syms a b c x;
y = a*x^2+b*x+c;
xsoln = solve(y,x);
pretty(xsoln);

a=2; b=-5; c=3;
subs(xsoln);
```

\[ y = ax^2 + bx + c \]

\[ x = -\frac{b \pm \sqrt{b^2 - 4ac}}{2a} \]

\[ y = (2x - 3)(x + 2) \]

\[ x = -\frac{b \pm \sqrt{b^2 - 4\pi c}}{2\pi} \]
Solving Equations

\[
\begin{align*}
\text{syms } & a \ b \ c \ x; \\
y & = a*x^2+b*x+c; \quad \% \text{ quadratic eqn.} \\
xroot & = \text{solve}(y,x); \quad \% \text{ solve for } x. \\
\text{asolv} & = \text{solve}(y,a); \quad \% \text{ solve for } a.
\end{align*}
\]

\[
\begin{align*}
\text{solve('}y=a*x^2+b*x+c', 'x') \\
\text{ans} & = \\
-1/2*(b-(b^2+4*a*y-4*a*c)^(1/2))/a \\
-1/2*(b+(b^2+4*a*y-4*a*c)^(1/2))/a
\end{align*}
\]

\[
\begin{align*}
\text{solve('}y=a*x^2+b*x+c', 'a'); \\
\text{ans} & = \\
-(y+b*x+c)/x^2
\end{align*}
\]

NOTE: here we don’t need to use “sym” since solve expects a symbolic expression. Quotes do the trick. Of course, this also works with a sym expression (see above).
Solving Multiple Equations

Example:

\[ y_1 = a_1 x_1 + b_1 \]
\[ y_2 = a_2 x_2 + b_2 \]

\[ S = \text{solve}(\ 'y_1=a_1*x_1+b_1', \ 'y_2=a_2*x_2+b_2') ; \]

Returns a “structure” with the solution variables, \( x_1, x_2 \).

\[ S.x_1 = (y_1-b_1)/a_1 \]
\[ S.x_2 = -(y_2-b_2)/a_2 \]

Example:

\[ x = \sqrt{\frac{y - 3}{2}} \]
\[ y = x^3 + x^2 - 6x + 3 \]

\[ S = \text{solve}(\ 'x=sqrt((y-3)/2)', \ 'y=x^3+x^2-6*x+3') ; \]
\[ \text{fprintf}(\ 'x=\n'); \text{disp}(S.x) ; \]
\[ \text{fprintf}(\ 'y=\n'); \text{disp}(S.y) ; \]

\[ x= 0 \ 3 \ y= 3 \ 21 \]

Recall we solved this system numerically...

A generalized version:

\[ S = \text{solve}(\ 'x=sqrt((y-e)/f)', \ldots \ 'y=a*x^3+b*x^2+c*x+d', x, y) \]
\[ \text{disp}(S.x) ; \]

Warning: ugly solution...