LECTURE 6: SOLVING EQUATIONS NUMERICALY

The `solve` command is used for finding *symbolic* solutions to equations. This is not always possible, but *numerical* (approximate) solutions can usually be found. If `solve` is unable to find a symbolic solution, it will try to find a numerical solution.

For example, suppose we wish to solve the equation

$$\sin x = x^3 - 5x^2 + 4,$$

and that having plotted the graphs of $\sin x$ and $x^3 - 5x^2 + 4$ we know that there are three solutions, at approximately $x = -0.90$, 0.89 and 4.78 (these approximate solutions were obtained in Exercise 1.12).

To find these solution more precisely, we try `solve`:

```
>> clear
>> syms x
>> eqn = sin(x) == x^3 - 5*x^2 + 4;
>> solve(eqn, x)
```

This fails to find an exact symbolic solution but it finds a numerical solution

```
ans =
0.88543649189409090795806224578236
```

This solution is neither a `double` nor a `sym` class: we'll see in a moment why it is so accurate. But if you're going to *do something* else with the result, you probably want:

```
>> y = double(solve(eqn, x))
```

which converts it to a `double` quantity.

What about the other two solutions? The most common numerical approaches involve providing either an interval in which to search or an initial guess. Here we outline two possible approaches.

The `vpasolve` command The Symbolic Math Toolbox also has the command **vpasolve** which uses "variable-precision arithmetic" (see Appendix B.2 for more information.) One way to use this command is to provide a search interval:

```
>> vpasolve(eqn, x, [-1 0])
>> vpasolve(eqn, x, [0 1])
>> vpasolve(eqn, x, [4 5])

ans =
-0.90040020886787765320984918738742 6
ans =
0.88543649189409090795806224578236
ans =
4.7813983927628242128187791632837
```

Alternatively, you can provide an initial guess:

```
>> vpasolve(eqn, x, -.9)
>> vpasolve(eqn, x, .8)
>> y = double(vpasolve(eqn, x, 4.7))
```

(where the last example shows conversion to **double**, assuming the solution is to be used in further calculations.)

Note at least as of Releases 2013b, the **vpasolve** command does not respect assumptions on variables.

Alternative approach: `fzero` for numerical root finding The MATLAB command **fzero()** does root-finding: it searches for numerical solutions of $f(x) = 0$ where f is a function of one variable.

```
>> f = @(x)  sin(x) - ( x^3 - 5*x^2 + 4 );
>> format long
>> fzero(f, -1)

ans =
  -0.900400208867877
```

Some explanation: the first line creates a MATLAB "anonymous function" (see Appendix B.1), which is then passed to **fzero**, along with an initial guess. We can then find the third root:

```
>> fzero(f, 5)

ans =
   4.781398392762824
```

Note that `fzero` does not work with symbolic expressions or symbolic functions.

Having to enter the same expression again is not optimal and potentially error-prone. Here's one approach to convert the symbolic equation "`eqn`" from above into an appropriate function `f`:

```
>> % we want the left-hand-side and right-hand-side
>> LR = children(eqn)
>> f = matlabFunction(LR(1) - LR(2))
>> fzero(f, -1)

ans =
  -0.900400208867877
```

Exercise 2.5 Makes a plot showing the roots of $x^3 + 3x^2 - 2x + 1$. Then use `solve()` to find the roots. Use `double()` to find numerical approximations to the roots. □

Exercise 2.6 Find real solutions of the equation

$$x \sin x = \frac{1}{2}$$

for x in the following ranges: (i) $0 < x < 2$; (ii) $2 < x < 4$; (iii) $6 < x < 7$; (iv) $8 < x < 10$. Verify graphically that there are no solutions in the range $4 < x < 6$. □