4. Roots of algebraic and transcendental equations

- Roots of algebraic (polynomial) equations
- User-defined functions
- Roots of transcendental equations
- Symbolic computation

Roots of polynomial equations: roots

- To find the roots of a 2nd order polynomial equation $x^2-x-2=(x-2)(x+1)=0$, type as follows:

```
>> C=[1,-1,-2];
>> roots(C)
ans =
    2
   -1
```

- Roots of a 3rd order equation $x^3+1=0$ are calculated as follows:

```
>> C=[1,0,0,1];
>> roots(C)
ans =
   -1.00000 + 0.00000i
    0.50000 + 0.86603i
    0.50000 - 0.86603i
```

User-defined functions

- You can define an arbitrary function by writing a script of the form:

```
function [y1,...,yN] = myfun(x1,...,xM)
y1 = ...
...
endfunction
```

  - Save the following script into, say, “myfun.m”

```
#myfun.m
function y = myfun(x)
  y = x^2+sin(x)-1;
endfunction
```

  - You can call it as a function in the following ways:

```
>> myfun(0)
ans = -1
>> myfun(1)
ans = 0.84147
```

Remark: These commands must be run in the same directory (folder) as myfun.m was saved. Or you can add the directory where myfun.m exists to Octave’s load path; type “help path” for details.

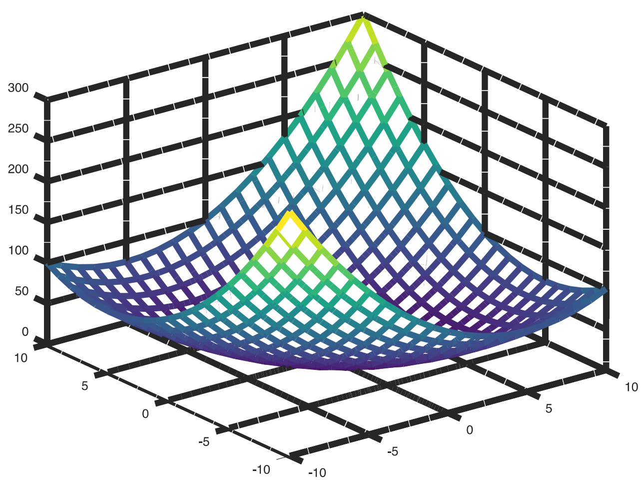
Anonymous function

- You can use *anonymous function*, which is another way of creating a user-defined function

```
>> myfun1 = @(x) (x^2+sin(x)-1);
>> myfun1(1)
ans = 0.84147
```

  - An example of functions with two (and more) variables:

```
>> myfun2 = @(x,y) (x.^2+y.^2+x.*y);
>> [X,Y] = meshgrid(-10:10);
>> mesh(X,Y,myfun2(X,Y))
```

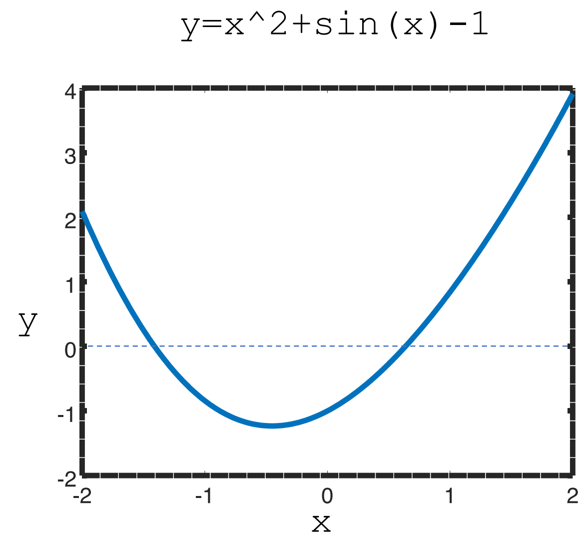


Remark: The use of x.^2 instead of x^2 above makes it possible to deal with the case when x is a matrix (or a vector or even a tensor).

Roots of transcendental equation: fsolve

- To find roots of $x^2+\sin(x)-1=0$, type as follows:

```
>> fsolve(@(x) x^2+sin(x)-1, 1.0)
ans =  0.63673
>> fsolve(@(x) x^2+sin(x)-1, -1.0)
ans = -1.4096
```



  - fsolve tries to find a root starting from given initial value
  - It can fail to find any root; the success depends on the equation and the provided initial values

20.1 Solvers From https://www.gnu.org/software/octave/doc/

Octave can solve sets of nonlinear equations of the form

```
F (x) = 0
```

using the function *fsolve*, which is based on the MINPACK subroutine *hybrd*. This is an iterative technique so a starting point must be provided. This also has the consequence that convergence is not guaranteed even if a solution exists.

Function File: **fsolve** *(fcn, x0, options)*
Function File: *[x, fvec, info, output, fjac]* = **fsolve** *(fcn, …)*

Solve a system of nonlinear equations defined by the function *fcn*.

fcn should accept a vector (array) defining the unknown variables, and return a vector of left-hand sides of the equations. Right-hand sides are defined to be zeros. In other words, this function attempts to determine a vector *x* such that *fcn (x)* gives (approximately) all zeros.

x0 determines a starting guess. The shape of *x0* is preserved in all calls to *fcn*, but otherwise it is treated as a column vector.

Symbolic package

- Extends Octave to enable symbolic computation
  - Function `solve` in MATLAB has not been implemented as of today
- To install `symbolic` package, visit https://github.com/cbm755/octsympy and follow the instruction.
- To use this package, type the following in Command Window:

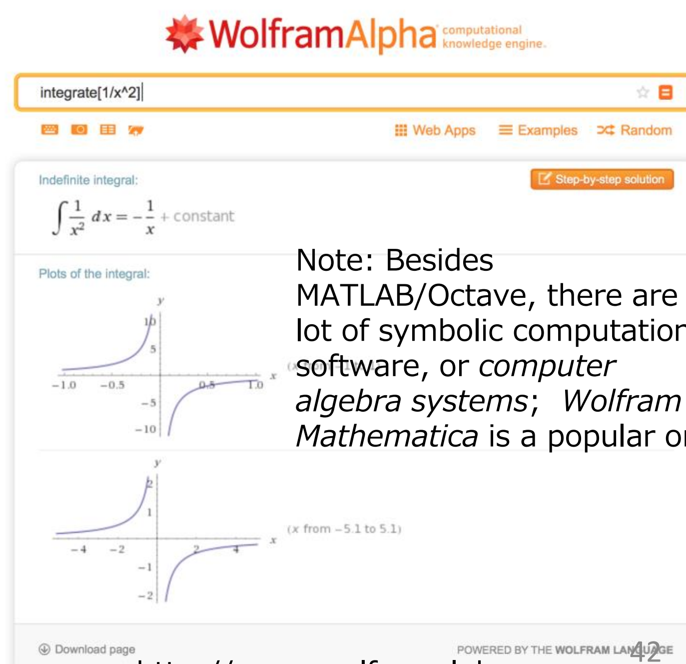
```
>> pkg load symbolic
```

- To start symbolic computation, you must first declare a symbolic variable by syms

```
>> syms x
```

- A symbolic representation of a function:

```
>> x^2+sin(x)-1
ans = (sym)
    2
   x  + sin(x) - 1
```



Note: Besides MATLAB/Octave, there are a lot of symbolic computation software, or *computer algebra systems*; *Wolfram Mathematica* is a popular one

http://www.wolframalpha.com

Symbolic package: factorization

- Factorization of a polynomial: factor

```
>> syms x
>> f=x^3+13*x^2-105*x+171;
>> factor(f)
ans = (sym)
        2
 (x - 3) *(x + 19)
```

```
>> syms x y
>> f=x^3*y-3*x^3-4*x^2*y+12*x^2-3*x*y+9*x+18*y-54;
>> factor(f)
ans = (sym)
        2
 (x - 3) *(x + 2)*(y - 3)
```

Symbolic package: differential

- Symbolic differential: diff

```
>> diff(x^2+sin(x)-1)
ans = (sym) 2*x + cos(x)
```

```
>> diff(exp(-x*sin(x)))
ans = (sym)
                                 -x*sin(x)
  (-x*cos(x) - sin(x))*e
```

Remark: If some special characters such as e^x or √ are not displayed properly, try the "sympref display ascii" command to switch to ascii mode.

Symbolic package: indefinite integral

- Indefinite integral : int

```
>> int(x^2+sin(x)-1)
ans = (sym)

    3
   x
   -- - x - cos(x)
   3
```

```
>> int(sin(log(x)))
ans = (sym)

   x*sin(log(x))   x*cos(log(x))
   ------------- - -------------
         2               2
```

Exercises 4.1

- Find all the roots to the following equation

$$10 \cdot sin^2(Ax) \cdot \exp\left(-\frac{Bx}{2}\right) + 0.01(C + D)x - 0.3 = 0, (0 \leq x \leq 5)$$

  - A, B, C, D are constant value, which is identified by your student number.
  - If your student number is ‘C6TB1234’, A=1, B=2, C=3, and D=4.

e.g.) **C 6 T B $\underline{1}$ $\underline{2}$ $\underline{3}$ $\underline{4}$**

$$\begin{array}{cccc} \| & \| & \| & \| \\ A & B & C & D \end{array}$$

  - Hint: You must specify good initial values to use `fsolve`. To do so, plot the function y=f(x) in the interval [0,5] as follows and make guesses of possible roots.

```
>> x=0:0.01:5;
>> y=10*sin(A*x).^2.*exp(-B*x/2) + 0.01*(C+D)-0.3;
>> y0=zeros(1,length(x));
>> plot(x,y,x,y0)
```