MA1710: Key points in week 6 Matlab session

Anonymous or one line functions

Suppose that we have the following functions.

$$
\begin{aligned}
f_1(x) &= \sin(x/6) - 1/2, \\
f_2(x) &= \tan(x/4) - 1, \\
f_3(x) &= \cos(x/3) - 1/2
\end{aligned}
$$

In Matlab we can set these up and plot them with the following statements which uses anonymous functions.

```
f1 =@(x) sin(x/6)-0.5;
f2 =@(x) tan(x/4)-1;
f3 =@(x) cos(x/3)-0.5;

x=linspace(2*pi/3, 4*pi/3, 201);
figure(2)
plot(x, f1(x), '--', x, f2(x), x, f3(x), '-.')
```


A function m-file for $\tan(x/4) - 1$

```
f2 =@(x) tan(x/4)-1;
```

A function file version of `f2` is to have a file called `f4.m` which contains the following 2 lines.

```
function y = f4(x)
y = tan(x/4)-1;
```


A function m-file solving $z^n = \zeta$

All the solutions to

$$z^n = \zeta = \rho(\cos(\alpha) + i\sin(\alpha))$$

are given by

$$z_k = \rho^{1/n}\left(\cos\left(\frac{\alpha}{n} + \frac{2k\pi}{n}\right) + i\sin\left(\frac{\alpha}{n} + \frac{2k\pi}{n}\right)\right), \quad k = 0, 1, \ldots, n-1.$$

A function to implement this in the file `all_nth_roots.m` can be as follows.

```
function z = all_nth_roots(zeta, n)
r = abs(zeta);
t = angle(zeta);
s = t+2*pi*(0:(n-1));
s = s/n;
z = r^(1/n)*(cos(s)+1i*sin(s));
```


A function m-file solving a quadratic equation

$$ax^2 + bx + c = 0, \quad x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

A function m-file in the file `solve_quadratic.m` can be as follows.

```
function [x1, x2] = solve_quadratic(a, b, c)
d = b*b-4*a*c;
s = sqrt(d);
x1 = (-b-s)/(2*a);
x2 = (-b+s)/(2*a);
```

Adding help comment lines

```
function [x1, x2] = solve_quadratic(a, b, c)
%%   [x1, x2] = solve_quadratic(a, b, c)
%   Given a~=0, b and c the function generates the
%   roots x1 and x2 of the quadratic  a*x^2+b*x*c

d = b*b-4*a*c;
s = sqrt(d);
x1 = (-b-s)/(2*a);
x2 = (-b+s)/(2*a);
```

The comments are displayed when we type

```
help solve_quadratic
```


The function header syntax

The first executable line in `myfun.m` has the following form.

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

1. Communication with other parts of a program are through the input and output arguments. All other quantities are local to the function.
2. We use the function with a statement of the form

   `[b1, ..., bN] = myfun(a1, ..., aM)`

3. With only one function per file the statements are executed until a `return` statement is reached or until the end of the file is reached.

Solving $f(x) = 0$ by the bisection method

Suppose $f : [a, b] \to \mathbb{R}$ is continuous and $f(a)f(b) < 0$. This implies that there exists $x \in (a, b)$ such that $f(x) = 0$.

We want a function which we can use as follows.

```
f1 =@(x) sin(x/6)-0.5;
f2 =@(x) tan(x/4)-1;
f3 =@(x) cos(x/3)-0.5;

[a1, b1] = bisec_meth(f1, 2, 4)
[a2, b2] = bisec_meth(f2, 2, 4)
[a4, b4] = bisec_meth(@f4, 2, 4)
[a3, b3] = bisec_meth(f3, 2, 4)
```


A function implementing the bisection method

A candidate function file called `bisec_meth.m` is as follows.

```
function [a, b]=bisec_meth(f, a, b)

fa=f(a);
fb=f(b);

for k=1:200
    c=0.5*(a+b);
    fc=f(c);
    if fa*fc<=0
        b=c;
        fb=fc;
    else
        a=c;
        fa=fc;
    end
end
```

You should add statements to improve it.