MA2501 Numerical
Methods
Spring 2014

Norwegian University of Science
and Technology
Department of Mathematics

Exercise set 5

1 Assume that the *regula falsi* method (without modifications) is used for finding the
solution of the equation $x^3 = 0$.

a) Show that during all the iterations one endpoint of the solution interval remains
unchanged, unless the iteration finds the solution after the first step.

b) Denote by c_k the result of the method at the k-th step. Show that

$$\lim_{k \to \infty} \frac{|c_{k+1}|}{|c_k|} = 1. \tag{1}$$

Note: It can be proven that the method of false position converges to the root,
so you can assume $\lim_{k \to \infty} c_k = 0$.

Remark: A sequence $(c_k)_{k \in \mathbb{N}}$ converging to 0 and satisfying (1) is said to converge
sublinearly. Sublinear converge of a sequence means that the number of iterations it
takes to come closer to the limit by even one digit will become arbitrarily large. Thus
numerical methods that have such sequences as output should usually be avoided.

2 Apply fixed point iteration to the solution of the equation $\cos(x) = \frac{1}{2}\sin(x)$ using
the mapping

$$\Phi(x) = x + \cos(x) - \frac{1}{2}\sin(x).$$

a) Show that the mapping Φ is a contraction on $[0, \pi/2]$ (note that you also have
to show that $\Phi(x) \in [0, \pi/2]$ for all $x \in [0, \pi/2]$).

b) Compute the first five iterates of the fixed point iteration using the starting
value $x^{(0)} = 0$.

c) Provide an estimate of the accuracy of the outcome of the method after the 5th
iteration. How many iterations will be needed to obtain a result with an error
smaller than 10^{-12}?

3 Define the function $G: \mathbb{R} \to \mathbb{R}$ by

$$G(x) = \begin{cases} x + e^{-x} & \text{if } x \geq 0, \\ 1 & \text{if } x \leq 0. \end{cases}$$

Show that this function satisfies $|G(x) - G(y)| < |x - y|$ for all $x \neq y \in \mathbb{R}$, but has no
fixed point. Why is this example no contradiction to Banach's fixed point theorem?
What happens if you apply fixed point iteration with this function G?

$\boxed{4}$ Show that $\mathbf{G}\colon \mathbb{R}^3 \to \mathbb{R}^3$, defined by

$$\mathbf{G}(x_1, x_2, x_3) = \left[\frac{1}{3}\cos(x_2 x_3) + \frac{1}{6}, \frac{1}{25}\sqrt{x_1^2 + \frac{5}{16}} - \frac{3}{100}, -\frac{1}{20}e^{-x_1 x_2} - \frac{10\pi - 3}{60}\right]^T,$$

has a unique fixed point in $D = \left\{[x_1, x_2, x_3]^T \mid -1 \le x_i \le 1, i = 1, 2, 3\right\}$. Use $\|\cdot\|_\infty$. Approximate the solution to within 10^{-5} in $\|\cdot\|_\infty$ by applying fixed point iteration.

$\boxed{5}$ Find an approximation of a solution of the equation $x^3 - 2x - 5 = 0$ by applying three steps of:

   a) the bisection method with starting interval $[2, 3]$,

   b) the secant method with starting values $x^{(0)} = 3$ and $x^{(1)} = 3.5$,

   c) the Newton method with starting value $x^{(0)} = 3$.

$\boxed{6}$ Compute the first three steps of the Newton method for the solution of the system of equations
$$-5x + 2\sin(x) + \cos(y) = 0,$$
$$4\cos(x) + 2\sin(y) - 5y = 0$$

with initial value $(x^{(0)}, y^{(0)}) = (0, 0)$ (you may want to use MATLAB for solving the linear systems).

$\boxed{7}$ Implement in MATLAB Newton's method for the solution of an equation $f(x) = 0$ with $f\colon \mathbb{R} \to \mathbb{R}$. Your function should take as an input the function f, its derivative f', and a starting value $x^{(0)}$. Test your implementation on the function in example $\boxed{5}$ of this exercise set.

MATLAB-comments

There are basically two methods to pass functions as arguments to another function. The first possibility is to pass the *name* of the function as a *string* (that is, the name written in single quotes). Then the command `feval` can be used for evaluating the function. Alternatively, you can pass a *function handle*, which is an argument of the form `@f`, where `f` is the function. Then you can evaluate the function either directly with `f(x)` or, again, using `feval`.

Consider for example the function

```
function y = myfun(f)
y = feval(f,1);
end
```

If you then call `myfun('sin')`, you obtain as result sin(1). The same result will be obtained using the function call `myfun(@sin)`.