Norwegian University of Science and Technology
Department of Mathematical Sciences

TMA4215 Numerical mathematics
Autumn 2015

Project 1

This project counts for 10% of the final grade in the course.
Due date: September 20, 23:59.

Some hints:

- Before starting, make sure that you master MATLAB on an elementary level. See e.g. the slides at the homepage. You can also consult the *Getting started page* at the MATLAB documentation, there are plenty of tips there.

  You should know:

  - functions in MATLAB, see: `function, nargin, nargout.` Remember that functions in MATLAB can use function handles, vectors, matrices, etc. as arguments.
  - control sentences: `if-else, for, while, break.`
  - Vector and matrix computations. For example: the linear system $\mathbf{Ax} = \mathbf{b}$ is solved by `x=A\b.` Do not mix row- and column vectors.
  - Other useful commands: `norm, sprintf, fprintf, disp.`

- Start as simple as possible. Do one thing at the time, and check that it is correct before proceeding. Compare with hand calculations on simple problems, if you find this easier.

- Make sure you understand how Newtons method for systems of equations works, see Definition 4.5 and example 4.5 in S&M.

Note: For problem 1 you should hand in a written answer to all the questions. You are free to choose whether you want to write your answers by hand or use LaTeX. It is sufficient to just answer the questions. Do not write a traditional report.

$\boxed{1}$ Given the function

$$f(x) = x^3 + x^2 - 3x - 3, \qquad x \in [1, 2].$$

a) Prove that f has at least one root in the interval $[1, 2]$.

b) Plot $f(x)$ by using MATLAB, and locate the root(s). Are there more than one?

c) Show that the equation $f(x) = 0$ can be reformulated in either of the forms

$$x = \sqrt[3]{-x^2 + 3x + 3} = g_1(x), \tag{1}$$

$$x = \frac{-x^2 + 3x + 3}{x^2} = g_2(x), \tag{2}$$

$$x = \sqrt{\frac{-x^2 + 3x + 3}{x}} = g_3(x), \tag{3}$$

Convince yourself that $\sqrt{3}$ is the root of f, as well as the fixed point of each of the right hand side functions above.

d) For each $g_i(x)$ above, use the iteration scheme $x_{k+1} = g_i(x_k)$. Start with $x_0 = 1.5$ and do 50 iterations with each of these schemes. Describe what you observe, and explain it.

e) The experiment above leaves two potentially useful iteration schemes. For those two, use the contraction mapping theorem to prove that the fixed point iterations will converge for all starting values $x_0 \in [1, 2]$.

f) Consider again the potentially useful iteration schemes. Using $x_0 = 1.0$, try to find an upper bound on the maximum number of iterations k required to ensure that

$$|x_k - \xi| < 10^{-10},$$

assuming you do not know the fixed point ξ.

Verify your result with a numerical experiment.

Note: For problem 2 we only want you to hand in the two MATLAB functions `testProblem` and `newton`, written according to the given specifications.

$\boxed{2}$ You are supposed to write a library routine for solving a system of non-linear equations

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}, \qquad \mathbf{f} : \mathbb{R}^m \to \mathbb{R}^m,$$

using Newtons method.

Specifications:

- *Arguments in:*
  - A handle to a function with a vector $\mathbf{x}$ as input parameter, returning $\mathbf{f}(\mathbf{x})$ and $J(\mathbf{x})$.
  - A vector with starting values $\mathbf{x}^{(0)}$.
  - An error tolerance ϵ.
  - The maximum allowed number of iterations `nMax`
- *Arguments out:*
  - The solution `x`.
  - The number of iterations `nIt`.
  - A flag, `iFlag` telling whether the iterations were successful or not.
- *And:*
  - Make a help text (a user should be able to use the routine from the information given by `help`).
  - Stop the iterations when $\max_i |f_i(x)| \le \epsilon$ (success) or `nIt` $\ge$ `nMax` (fiasco).
  - Write a warning message if the iterations do not converge.
  - The code should be self-documented, with a reasonable amount of comments in the code.

As a testproblem, use the equation

$$\begin{aligned} f_1(x_1, x_2) &= x_1^2 - 2x_1 - x_2 + 0.5 = 0, \\ f_2(x_1, x_2) &= x_1^2 + 4x_2^2 - 4 = 0. \end{aligned} \tag{4}$$

a) (Preparation)
Find the Jacobian $J_f(\mathbf{x})$ of $\mathbf{f}(\mathbf{x})$ given in (4).
Do one Newton iteration by hand, starting with $\mathbf{x}^{(0)} = (1, 1)^T$.

b) Write a MATLAB-function, `testProblem`, taking a vector `x` as an input argument, and returning the vector $\mathbf{f}(\mathbf{x})$ and the Jacobian $J_f(\mathbf{x})$ for the problem (4). The first line of this function will typically be

```
function [f,J] = testProblem(x)
```

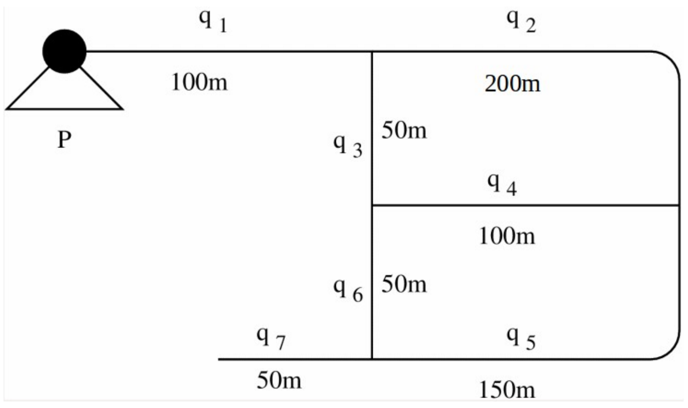
c) Write the library function by the specifications given above. The first line will typically be

```
[x,nIt,iFlag] = newton(f,x0,tol,nMax)
```

d) Use the function to solve the problem (4).

Application

Water flows through a pipe network as shown in the picture below.



We would like to know the distribution of the water flow, that is the flow rate q_i, $i = 1, 2 \ldots, 7$, through each pipe in the net. The pump produce an outlet gauge pressure of $4.1 \cdot 10^5$ Pa.

The following controls the flow:

- In each junction the rate of water that enters the junction is equal to the rate of water that leaves. For the junction up in the middle, that gives

$$q_1 - q_2 - q_3 = 0.$$

  Similar equations can be obtained for the remaining junctions.

- In each of the pipes, the pressure is reduced due to friction. The pressure drop is given by

$$\Delta P = \frac{8f\rho L}{\pi^2 d^5} q^2$$

  where f is a friction factor, ρ is the density of water, L is the length of the pipe, q is the flow rate and d is the inside diameter of the flow. For the current network:

$$f = 0.00225, \quad \rho = 998\text{kg/m}^3, \quad d = 0.15\text{m},$$

- The sum of the pressure drops around a loop in the network equals zero. Remember to include the *outer* loop (the one involving the gauge pressure).

Note: For problem 3 we want you to hand in the MATLAB-function described below and the script in which the function is called and the output presented. Also include a brief derivation of the system of equations in your written answer.

$\boxed{3}$ Consider the pipe network problem depicted above. Find seven equations that determine the flow rates q_i (in m^3/s), $i = 1, 2 \ldots, 7$, in the pipes. These can be written as a system of nonlinear equations $\mathbf{f}(\mathbf{q}) = \mathbf{0}$ with $\mathbf{q} = (q_1, q_2, \ldots, q_7)^T$. Create a MATLAB-function called `pipeProblem` (similar to `testProblem`), taking a vector $\mathbf{q}$ as an input argument, and returning the vector $\mathbf{f}(\mathbf{q})$ and the Jacobian $J_f(\mathbf{q})$, for these flow network equations. Solve the equations using your function `newton`.