MA2501 Numerical Methods

Spring 2009

Project 1.

- *Announced:* Tuesday January 27.
- *Hand-in:* MATLAB-code + a *short* report as a PDF file. Give your student number at the front page, but *no names.*

  The report should have two parts:

  - The results of some tests, using one of the test problems given below by your own choice. Maybe you could look for one than more solution..
    NB. It should be possible to reproduce the results, thus all relevant data has to be given.
  - A description on how you solved the "application". And the result.
- *Deadline:* Friday February 13. Send the MATLAB-code and the report to the teacher by e-mail. Each group will demonstrate the use of their Newton solver for the teacher afterwards.
- *Groups:* 1-2 students in each.
- *Supervision:* During the exercises in week 5 and 7. From February 6, whenever needed (within the office hours).
- *Evaluation:* The implementation of Newtons method counts for 70%, the solution of the application 30%. Emphasis will be given to a readable code, and good documentation. For full score, the first two points under *possible improvements* have to be implemented.

The exercise

You are supposed to write a library routine for solving a system of non-linear equations

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

using Newtons method. It should then be applied to a nontrivial problem (application).

Specifications:

- *Arguments in:*
  - A handle to a function with a vector x as input parameter, returning $F(x)$. If necessary, also the Jacobi-matrix $J(x)$.
  - A vector with starting values x_0.
  - An error tolerance tol.
  - The maximum allowed number of iterations $Nmax$
- *Arguments out:*
  - The solution x.
  - The number of iterations nit.
  - A flag 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 $\max_i |f_i(x)| \leq tol$ (success) or $nit \geq Nmax$ (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.
- *Possible improvements (lower priority):*
  - Write a function to compute a numerical approximation to the Jacobian. Let the user choose wether she prefer an analytic or an approximated Jacobian.
  - Give tol and $Nmax$ default values. But it should also be possible for the user to set them.
  - Do the iterations diverge? If the difference of two succeeding iterates increases, this is probably the case.
  - Make an option for nice output during the iterations.
  - Etc.

Some hints:

- Before starting, make sure that you master MATLAB on an elementary level. The MATLAB functions on the web-page might give you some ideas. You should master:
  - 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 $Ax = b$ is solved by `x=A\b.` Do not mix row- and column vectors.

  Other useful functions (see `doc`)

  - `norm, sprintf, disp.`
- Start as simple as possible. Choose a system of two equations, if you know the solution it is even better. Do a couple of iterations by hand. Write the MATLAB function computing $F(x)$ and $J(x)$ for the given system. Test it. Write a loop, doing a few Newton iteration. Test it. Do you get the same result as with your hand calculations. Same result? If yes, you are on the track. If not, spot exactly where the discrepancy occur.

  You can also test on a linear system. Newton should then converge in one iteration.

  Add more functionality, one by one. Test each thing separately. Do the program behave as expected? What did you expect?
- Learn how to master the debugger in MATLAB.

Testproblems:

P1:

$$\begin{aligned} x_1^2 - 2x_1 - x_2 + 0.5 &= 0 \\ x_1^2 + 4x_2^2 - 4 &= 0 \end{aligned}$$

P2:

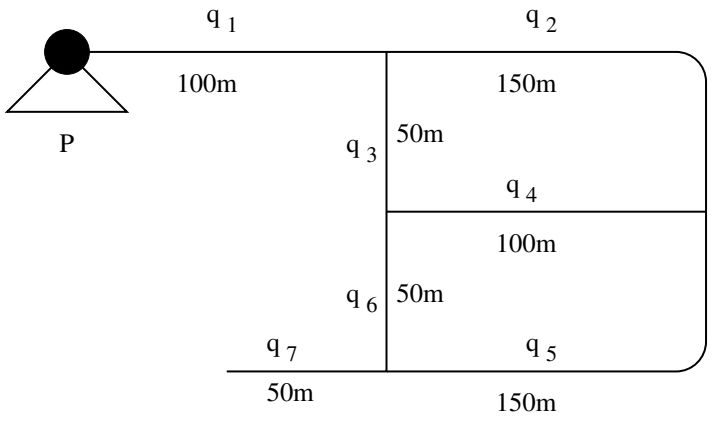
$$\begin{aligned} 5x_1^2 - x_2^2 &= 0 \\ x_2 - 0.25\left(\sin x_1 + \cos x_2\right) &= 0 \end{aligned}$$

P3:

$$\begin{aligned} 3x_1 - \cos(x_2 x_3) - 0.5 &= 0 \\ x_1^2 - 81(x_2 + 0.1)^2 + \sin x_3 + 1.06 &= 0 \\ e^{-x_1 x_2} + 20x_3 + \frac{10\pi - 3}{3} &= 0 \end{aligned}$$

Application:

(Bradie -06) 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 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

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

  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.

- The sum of the pressure drops around a loop in the network equal 0.

Find the seven equations that describes the flow rates in the pipes. Use

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

and compute the flow rates q_i, $i = 1, 2 \ldots, 7$ (in m^3/s).