Exercise 4 (27.10.2014)

1) Read chapter 3 from Haykin's book (2nd edition), starting from section 3.1 to 3.7. Summarize or sketch your insights in mind-map or an outline or a summary.
2) Consider training a two-input perceptron. Give an upper bound on the number of training examples sufficient to assure with 90% confidence that the learned perceptron will have true error of at most 5%. Does this bound seem realistic?
3) Solve Ex 3.1 and 3.4 from Haykin's book(2^{nd} edition).

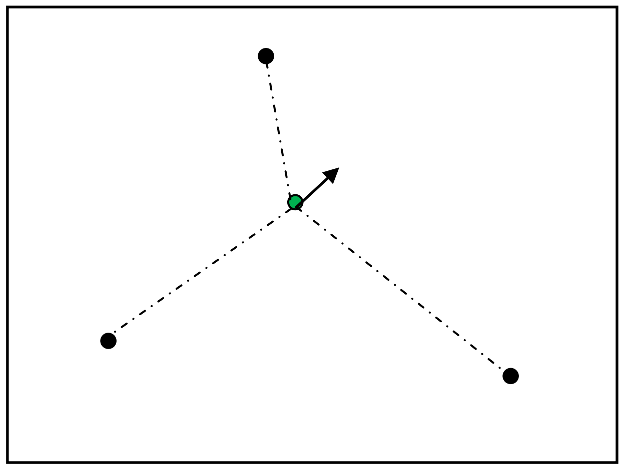

Fig 1. Ship (green dot), 3 light-houses (black dots)

4) A sample MATLAB-code has been attached with the question's file. This is an application of Non-Linear Gauss Newton. Your task will be first to study the sample code.
   PGP_pose_from_distance.m -> MATLAB function for solving Gauss-Newton.
   test_pose_from_distance2.m -> Test file (run it for results)

   Basically what it does is this:
   Solves a set of three non-linear equations (here: the distance to 3 light houses) to build the [x,y] of a ship.
   The idea behind the sample code is:
   - linearize the eq's (i.e.buildt Jacobian) at current pose
   - build residues
   - LMS solve for pose_correction
   - pose = pose + pose_correction
   - iterate till error is small

Now this implementation doesn't have the bearing involved. Your task will be to determine Pxint from **measured** bearing of the ship w.r.to. each lighthouse.

In order to do this:

a) Assume ground truths. Ship and light houses' coordinates, ship's bearing (a direction).
b) Take measured bearing as something close to true bearings (w.r.to. each lighthouses)
c) replace the equations
   (distance_from eq1= sqrt(....))
   by the bearing to
   atan2((y-yi),(x-xi)) - alpha_i = residu_i
   with alpha_i = measured distance to lighthouse i at [xi,yi]

Output of the task:

Find the region of convergence i.e. from which Pxinit will the algorithm still converge to the right answer and comment on your findings.

5) Please upload 3 questions and their brief answers on the reading material in a separate txt file.

- w.r.to. => with respect to
- "**Bearing**" is a term used in navigation; simply to refer to direction of motion. "Bearing of ship w.r.to. a lighthouse" is the angle between ship's forward direction, and the direction from ship to an lighthouse.
- In order to take measured bearing, first calculate true bearing w.r.to each lighthouse using atan2 formula and take some value close to it (adding some distortion).
- In case you want to use OCTAVE, please use the symbolic toolbox for it (http://octave.sourceforge.net/symbolic/)
- You can also use Python. sympy is a good option for it.