# Statistical Machine Learning 2018

Assignment 3
Deadline: 25th of November 2018

**Instructions:**

- You can work **alone or in pairs** (= max 2 people). **Write the full name and S/U-number of all team members on the first page of the report.**
- Write a **self-contained report** with the answers to each question, **including** comments, derivations, explanations, graphs, etc. This means that the elements and/or intermediate steps required to derive the answer have to be in the report. (Answers like 'No' or 'x=27.2' by themselves are not sufficient, even when they are the result of running your code.)
- If an exercise specifically asks for code, put **essential code snippets** in your answer to the question in the report, and explain briefly what the code does. In addition, hand in **complete (working and documented) source code** (MATLAB recommended, other languages are allowed but not "supported").
- In order to avoid extremely verbose or poorly formatted reports, we impose a **maximum page limit** of 20 pages, including plots and code, with the following formatting: fixed **font size** of 11pt on an **A4 paper**; **margins** fixed to 2cm on all sides. All figures should have axis labels and a caption or title that states to which exercise (and part) they belong.
- Upload reports to **Blackboard** as a **single pdf** file: '`SML_A3_<Namestudent(s)>.pdf`' and one zip-file with the executable source/data files (e.g. matlab m-files). For those working in pairs, only one team member should upload the solutions.
- Assignment 3 consists of 3 exercises, weighted as follows: 5 points, 2 points, and 3 points. The **grading** will be based solely on the report pdf file. The source files are considered supplementary material (e.g. to verify that you indeed did the coding).
- For any problems or questions, send us an email, or just ask.
  Email addresses: `tomc@cs.ru.nl` and `b.kappen@science.ru.nl`

# Exercise 1 – The faulty lighthouse (weight 5)

A lighthouse is somewhere off a piece of straight coastline at a position $\alpha$ along the shore and a distance $\beta$ out to sea. Due to a technical fault, as it rotates the light source only occasionally and briefly flickers on and off. As a result it emits short, highly focused beams of light at random intervals. These pulses are intercepted on the coast by photo-detectors that record only the fact that a flash has occurred, but not the angle from which it came. So far, $N$ flashes have been recorded at positions $\mathcal{D} = \{x_1, \ldots, x_N\}$. Where is the lighthouse?

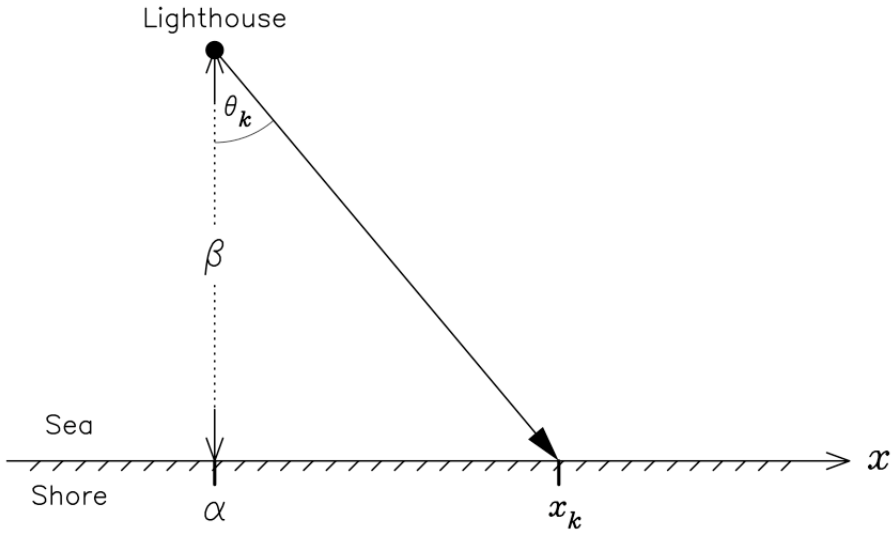


Figure 1: Geometry of the lighthouse problem.

**Part 1 – Constructing the model**

1. Let $\theta_k$ be the (unknown) angle for the $k$-th recorded flash, see fig.1. Argue why

$$p(\theta_k|\alpha, \beta) = \frac{1}{\pi} \tag{1}$$

would be a reasonable distribution over $\theta_k$ between $\pm\pi/2$ (zero otherwise).

We only have the position $x_k$ of the detector that recorded flash $k$, but we can relate this to the unknown $\theta_k$ via elementary geometry as

$$\beta \tan(\theta_k) = x_k - \alpha \tag{2}$$

2. Show that the expected distribution over $x$ given $\alpha$ and $\beta$ can be written as

$$p(x_k|\alpha, \beta) = \frac{\beta}{\pi\,[\beta^2 + (x_k - \alpha)^2]} \tag{3}$$

by using (2) to substitute variable $x_k$ for $\theta_k$ in the distribution (1). Plot the distribution for $\beta = 1$ and a particular value of $\alpha$.
Hint: use the Jacobian $\left|\frac{\mathrm{d}\theta}{\mathrm{d}x}\right|$ (Bishop,p.18) and the fact that $(\tan^{-1} x)' = \frac{1}{1+x^2}$.

Inferring the position of the lighthouse corresponds to estimating $\alpha$ and $\beta$ from the data $\mathcal{D}$. This is still quite difficult, but if we assume that $\beta$ is known, then from Bayes' theorem we know that $p(\alpha|\mathcal{D}, \beta) \propto p(\mathcal{D}|\alpha, \beta)\, p(\alpha|\beta)$. We have no a priori knowledge about the position $\alpha$ along the coast other than that it should not depend on the distance out at sea.

3. Show that with these assumptions the log of the posterior density can be written as

$$L = \ln\left(p(\alpha|\mathcal{D},\beta)\right) = constant - \sum_{k=1}^{N} \ln\left[\beta^2 + (x_k - \alpha)^2\right] \tag{4}$$

and give an expression for the value $\hat{\alpha}$ that maximizes this posterior density.

Suppose we have a data set (in km) of $\mathcal{D} = \{3.6, 7.7, -2.6, 4.9, -2.3, 0.2, -7.3, 4.4, 7.3, -5.7\}$. We also assume that the distance $\beta$ from the shore is known to be 2 km. As it is difficult to find a simple expression for the value of $\hat{\alpha}$ that maximizes (4), we try an alternative approach instead.

4. [MATLAB] - Plot $p(\alpha|\mathcal{D}, \beta = 2)$ as a function of $\alpha$ over the interval $[-10, 10]$. What is your most likely estimate for $\hat{\alpha}$ based on this graph? Compare with the mean estimate of the dataset. Can you explain the difference?

**Part 2 – Generate the lighthouse data**

We will try to solve the original problem by letting MATLAB find the lighthouse for us. For that we first need a data set.

1. [MATLAB] - Sample a random position $(\alpha_t, \beta_t)$ from a uniform distribution over an interval of 10 km along the coast and between 2 and 4 km out to sea.
2. [MATLAB] - From this position generate a data set $\mathcal{D} = \{x_1, \ldots, x_N\}$ of 500 flashes in random directions that have been registered by a detector at point $x_i$ along the coast. Assume that the flashes are i.i.d. according to (1).
3. [MATLAB] - Make a plot of the mean of the data set as a function of the number of points. Compare with the true position of the lighthouse $\alpha_t$. How many points do you expect to need to obtain a reasonable estimate of $\alpha_t$ from the mean? Explain.

**Part 3 – Find the lighthouse**

From the analysis in the first part we know that trying to find a maximum likelihood estimate in the usual way is possible (compute gradient, set equal to zero and solve), but that this does not result in a 'nice' closed-form expression for the solution, even when one of the parameters is assumed to be known. As we want to find estimates of both $\alpha$ and $\beta$ from the data, we will try a different approach instead.

1. Use (3) to get an expression for the loglikelihood of the data $\mathcal{D}$ as a function of $\alpha$ and $\beta$.

We can see how this likelihood (as a function of $\alpha$ and $\beta$) changes, as data points come in.

2. [MATLAB] - Process your data set $\mathcal{D}$ one by one and make a plot of the (log)likelihood after one, two, three, and 20 points have arrived, respectively. Explain what happens.

   *Hint:* Create a function that calculates the (log)likelihood at a specific point $(\alpha, \beta)$ after the first $k$ data points $\{x_1, \ldots, x_k\}$ have come in. Use this with the MATLAB `meshgrid` and `surf` functions to make plots over the interval $[-10 \leq \alpha \leq +10] \times [0 \leq \beta \leq 5]$. Decide if/when it makes more sense to use the likelihood directly or the log of the likelihood.

We can make a reasonable (visual) estimate of the most probable position of the lighthouse from the graph, after a few data points have been observed. However, as we are working with a computer, we will let MATLAB do the dirty work for us.

3. [MATLAB] - Create a function that uses MATLAB function `fminsearch` to compute the values of $\alpha$ and $\beta$ that maximize the likelihood for a data set of $k$ points, and plot these as a function of the number of points. Use $[0, 1]$ as the initial starting value for `fminsearch` (see examples in MATLAB-help). Compare your final estimate with the true values $(\alpha_t, \beta_t)$.[1]

[1] If you use OCTAVE, you need to install the `optim` package, which provides an implementation of `fminsearch` (which has a different interface than the MATLAB `fminsearch` function, by the way).

## Exercise 2 – Bayesian linear regression (weight 2)

This exercise builds on exercise 2, week 7, "Fitting a straight line to data". For a detailed description (and explanation) see EXERCISES AND ANSWERS, WEEK 7 in Brightspace. The final part of that exercise computed the predictive distribution after a single data point was observed. Here we consider a new data set, consisting of no less than *two* points: $\{x_1, t_1\} = (0.4, 0.1)$ and $\{x_2, t_2\} = (0.6, -0.4)$.

1. Assume $\alpha = 1$ and $\beta = 15$. Compute the predictive distribution $p(t|x, \mathbf{t}, \mathbf{x}, \alpha, \beta)$ after these two points are observed.
2. Plot the mean of the predictive Gaussian distribution and one standard deviation on both sides as a function of $x$ over the interval $[0, 1]$. Plot the data in the same figure. See `a009plotideas.m` in Brightspace for some plotting hints. Compare your plot with Figure 3.8b (Bishop, p.157) and explain the difference.
3. Sample five functions $y(x, \mathbf{w})$ from the posterior distribution over $\mathbf{w}$ for this data set and plot them in the same graph (i.e. with the predictive distribution). You may use the Matlab function `mvnrnd`. See again `a009plotideas.m` for some plotting hints.

## Exercise 3 – Gradient descent revisited (weight 3)

In this exercise, we will have a closer look at the gradient descent algorithm for function minimization. When the function to be minimized is $E(\mathbf{x})$, the gradient descent iteration is

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \eta \nabla E(\mathbf{x}_n) \tag{5}$$

where $\eta > 0$ is the so-called learning-rate.

1. Consider the function $f(x) = \frac{\lambda}{2}(x - a)^2$ with parameters $\lambda > 0$, and $a$ arbitrary.
   (a) Write down the gradient descent iteration rule. Verify that the minimum of $f$ is a fixed point[2] of the gradient descent iteration rule.
   (b) Find $\eta$ for which convergence is the fastest (actually in one step).
   (c) We will investigate the convergence properties for different values of $\eta$. For this we look at the ratio of the distance to the fixed point after the step and the distance before the step:
   $$r_n = \frac{|x_{n+1} - x^*|}{|x_n - x^*|}. \tag{6}$$
      i. What does it mean if all $r_n < c < 1$ (i.e. all ratio's are smaller than a certain number below 1). Give for this case the (optimal) upper bound of the distance $|x_n - x^*|$ in terms of $|x_0 - x^*|$, $c$ and $n$.
      ii. What is the consequence if all $r_n > c > 1$? Give for this case the optimal lower bound of the distance $|x_n - x^*|$ in terms of $|x_0 - x^*|$, $c$ and $n$.
   (d) Show that in our case, $r_n = |1 - \eta\lambda| \equiv r$, independent of $n$ (we refer to $r$ as the convergence rate). For which $\eta$ is the algorithm convergent?
2. Consider the function $g(x, y) = \frac{\lambda_1}{2}(x - a_1)^2 + \frac{\lambda_2}{2}(y - a_2)^2$ with parameters $0 < \lambda_1 \leq \lambda_2$, and $a_i$ arbitrary.
   (a) Write down the gradient descent iteration rule. Verify that the minimum of $f$ is a fixed point.

[2] A fixed point $x^*$ of an iteration $x_{n+1} = F(x_n)$ satisfies $x^* = F(x^*)$.

(b) Show that $\eta = \dfrac{2}{\lambda_2 + \lambda_1}$ minimizes the larger ratio between $r_{n,x} = \dfrac{|x_{n+1} - x^*|}{|x_n - x^*|}$ and $r_{n,y} = \dfrac{|y_{n+1} - y^*|}{|y_n - y^*|}$, i.e., $\arg\min_{\eta}\{\max\{r_{n,x}, r_{n,y}\}\} = \dfrac{2}{\lambda_2 + \lambda_1}$. What happens if $\eta$ is smaller than this optimal value? What happens if it is larger?

(c) What is the convergence rate for this $\eta$? What does this say about the applicability of gradient descent to functions with steep hills and flat valleys (i.e., if $\lambda_2 \gg \lambda_1$)?

(d) Implement the gradient descent algorithm for $g$ in MATLAB.

(e) Make some plots of the trajectories $\{(x_n, y_n)\}_{n=0}^{N}$ for different values of $\lambda_i$ and $\eta$ ($\eta$ optimal, larger than optimal, and smaller than optimal) to illustrate what is going on. Plot these trajectories on top of a contour plot of $g$. Monitor the convergence rates. Explain what happens.