Statistical Machine Learning 2018

Assignment and answers 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 **Brightspace** 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 α along the shore and a distance β 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?

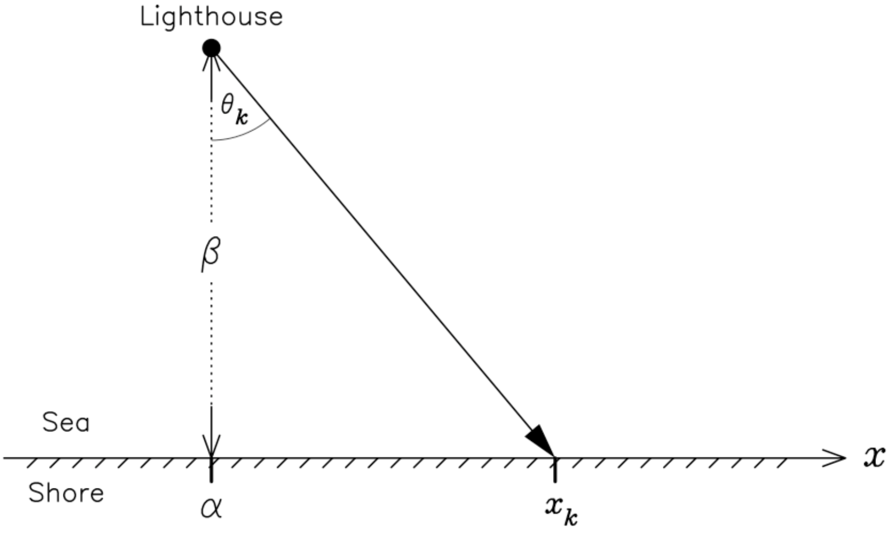

Figure 1: Geometry of the lighthouse problem.

Part 1 – Constructing the model

1. Let θ_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 θ_k between $\pm\pi/2$ (zero otherwise).

ANSWER: Random intervals imply random directions, so a uniform distribution over the angles between $\pm\pi/2$ seems appropriate. Angles $|\theta_k| > \pi/2$ are out to sea and cannot be detected along the coast. Normalized properly this gives (1). (Implicitly assuming an infinitely long coastline, densely packed with tiny detectors).

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

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

2. Show that the expected distribution over x given α and β can be written as

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

by using (2) to substitute variable x_k for θ_k in the distribution (1). Plot the distribution for $\beta = 1$ and a particular value of α.

Hint: use the Jacobian $\left|\frac{\mathrm{d}\theta}{\mathrm{d}x}\right|$ (Bishop,p.18) and the fact that $(\tan^{-1} x)' = \frac{1}{1+x^2}$.

ANSWER: From Bishop, p.18 we know that for a probability distribution a change in variables involves a 'correction factor' by multiplying with the Jacobian of the transformation (similar to a change of variables for integration). In this case we want to transform the very simple distribution over unknown quantity θ_k to the much more relevant distribution over measured variable x_k. To do so we need to substitute x_k for θ_k in (1) and multiply with the Jacobian $\left|\frac{\mathrm{d}\theta_k}{\mathrm{d}x_k}\right|$. The first step involves no action as θ_k does not actually appear in (1). That leaves just multiplication with the Jacobian. From (2) we have

$$\theta_k = \arctan\left(\frac{x_k - \alpha}{\beta}\right) \tag{4}$$

and so for the Jacobian (using the chain rule and the hint for the derivative)

$$\frac{\mathrm{d}\theta_k}{\mathrm{d}x_k} = \frac{1}{\beta\left[1 + \left(\frac{x_k-\alpha}{\beta}\right)^2\right]} \tag{5}$$

which, after multiplication with (1) and a little bit of rearranging, gives the result (3). Figure 2 illustrates the probability densities.

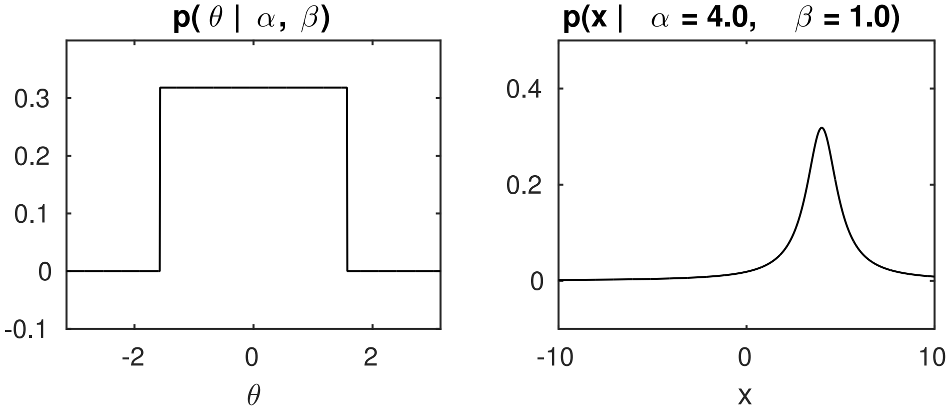

Figure 2: Probability densities $p(\theta_k|\alpha,\beta)$ (left) and $p(x|\alpha = 4, \beta = 1)$ (right).

Inferring the position of the lighthouse corresponds to estimating α and β from the data $\mathcal{D}$. This is still quite difficult, but if we assume that β 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 α 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

$$L = \ln\left(p(\alpha|\mathcal{D},\beta)\right) = \textit{constant} - \sum_{k=1}^{N} \ln\left[\beta^2 + (x_k - \alpha)^2\right] \tag{6}$$

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

ANSWER: The log posterior density of the distance α out to sea given a dataset $\mathcal{D}$ and prior $p(\alpha|\beta)$ is determined as

$$\begin{aligned} L(\alpha) &= \ln(p(\alpha|\mathcal{D},\beta)) \\ &= \ln\left(c_1 \cdot p(\mathcal{D}|\alpha,\beta) \cdot p(\alpha|\beta)\right) \\ &= \ln(c_1) + \ln\left(p(\mathcal{D}|\alpha,\beta)\right) + \ln\left(p(\alpha|\beta)\right) \end{aligned}$$

with c_1 the constant of proportionality.
The prior over α reduces to

$$p(\alpha|\beta) = p(\alpha) = c_2 \tag{7}$$

over some large enough interval, and is therefore also constant in the loglikelihood.
The measurements of the flashes do not influence one another, so the likelihood function for the data set $\mathcal{D}$ is just the product of N independent observations of a random variable with distribution (3):

$$\begin{aligned}
\ln(p(\mathcal{D}|\alpha,\beta)) &= \prod_{k=1}^{N} p(x_k|\alpha,\beta) \\
&= \ln\left(\prod_{k=1}^{N} \frac{\beta}{\pi\left[\beta^2 + (x_k-\alpha)^2\right]}\right) \\
&= \sum_{k=1}^{N}\left(\ln\left(\frac{\beta}{\pi}\right) - \ln\left[\beta^2 + (x_k-\alpha)^2\right]\right) \\
&= c_3 - \sum_{k=1}^{N} \ln\left[\beta^2 + (x_k-\alpha)^2\right]
\end{aligned}$$

with c_3 the sum over $\ln\left(\frac{\beta}{\pi}\right)$, and the minus sign appears as a result of the division in (3). Grouping all (irrelevant) constant terms $c_1 + c_2 + c_3 = \textit{constant}$ then gives (6).
Maximizing w.r.t. α implies differentiating L w.r.t. α, setting equal to zero and solving

$$\left.\frac{\mathrm{d}L}{\mathrm{d}\alpha}\right|_{\hat{\alpha}} = 2\sum_{k=1}^{N} \frac{x_k - \hat{\alpha}}{\beta^2 + (x_k - \hat{\alpha})^2} = 0 \tag{8}$$

Unfortunately, apart form the trivial case for $N = 1$ and the reasonably straightforward case for $N = 2$, this cannot be reduced much further for $N > 2$.

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 β 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 (6), we try an alternative approach instead.

4. [MATLAB] - Plot $p(\alpha|\mathcal{D}, \beta = 2)$ as a function of α 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?

ANSWER:

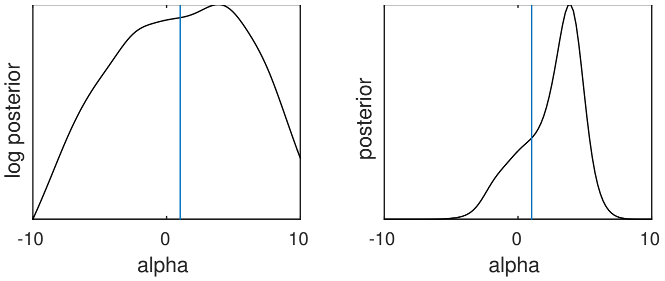

Figure 3: The (log-)posterior density for the position α of the lighthouse.

The left-hand side of Figure 3 depicts the log posterior density of α, the right-hand side the posterior density. The top of the curve corresponds to the MAP estimate $\hat{\alpha} \approx 3.87$. The mean $\bar{\alpha} = \frac{1}{N}\sum_{k=1}^{N} x_k$ of the data set, depicted as a vertical blue line, lies significantly below this estimate at $\bar{\alpha} = 1.02$. The reason for this is that the mean and the expectation values are typically different for the Cauchy distribution (3). In fact: the mean will *never* converge to a fixed value, no matter how many points you obtain! The heavy tails of this distribution (for example in comparison to a Gaussian) make that the model is very insensitive to outliers: further lowering the value of the last point in the data set will actually *increase* the most likely value $\hat{\alpha}$, as it becomes more likely that this was an unrepresentative point, somewhere far out in the tail, resulting in more relative weight for the other points in determining the best estimate.

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 (α_t, β_t) from a uniform distribution over an interval of 10 km along the coast and between 2 and 4 km out to sea.

   ANSWER: The following piece of code will give $\alpha_t = 2.7132$ and $\beta_t = 2.0415$.

   ```
   rng(10); % fix seed for reproducible results
   a_t = rand*10 - 5; % between [-5, +5] km along coast
   b_t = rand*2 + 2;  % between [2, 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).

   ANSWER: Generate flashes in random direction from uniform $\theta \in [-\frac{\pi}{2}, +\frac{\pi}{2}]$, and transform to positions on the coast using the tan-equation (2). You can also recognize the resulting Cauchy distribution as Student's t-distribution with 1 degree of freedom, and use the MATLAB function `trnd` to sample it directly. Example code is provided in `a008.m`.

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 α_t. How many points do you expect to need to obtain a reasonable estimate of α_t from the mean? Explain.

   ANSWER: This is a bit of a trick question, since the Cauchy distribution is an example of a distribution for which the expected value is *not defined.* This means that the sample mean will never converge to a single value, no matter how many points you sample (see Figure 4). This becomes more apparent when you sample a few thousand points and plot the mean: it may seem to converge but always new 'jumps' appear when a very large value occurs. (These are *not* outliers, but an essential part of the distribution!).

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 α and β 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 α and β.

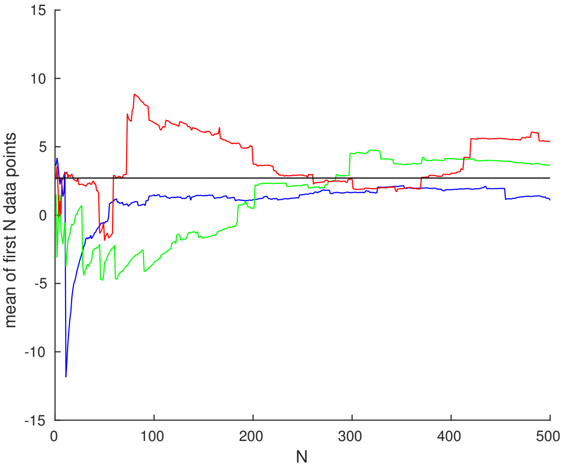

Figure 4: Mean of the data $x_1, \ldots, x_N$ as a function of N, the number of data points. The three colors correspond with three different random data sets.

ANSWER: Similar to eq.(6), except that β is now not a constant:

$$\begin{aligned} L(\mathcal{D}|\alpha,\beta) &= \prod_{k=1}^{N} p(x_k|\alpha,\beta) \\ &= \prod_{k=1}^{N} \frac{\beta}{\pi\left[\beta^2 + (x_k-\alpha)^2\right]} \end{aligned}$$

Taking the log then gives

$$\log L(\mathcal{D}|\alpha,\beta) = N \log\beta - \sum_{k=1}^{N} \log(\beta^2 + (x_k-\alpha)^2)$$

where we ignored the contribution of constant factors.

We can see how this likelihood (as a function of α and β) 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 (α, β) 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.

   ANSWER: See example implementation in `a008.m` and Figure 5 for the generated plots. Note: in cases where there are many points the product of small terms may fall below the threshold in MATLAB, which makes summing over many such values unreliable or pointless. In such cases it is better to convert to logs first, do the calculations and then convert back to the actual value.

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 α and β that maximize the likelihood for a data set of k points, and plot these as a

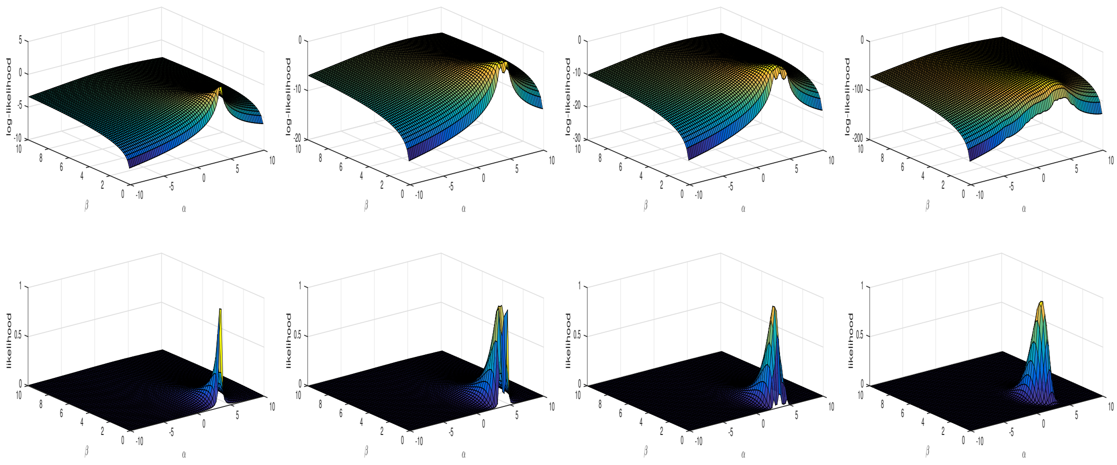

Figure 5: Top: log-likelihoods after N data points have arrived; Bottom: likelihoods after N data points have arrived. From left to right: $N = 1$, $N = 2$, $N = 3$, $N = 20$.

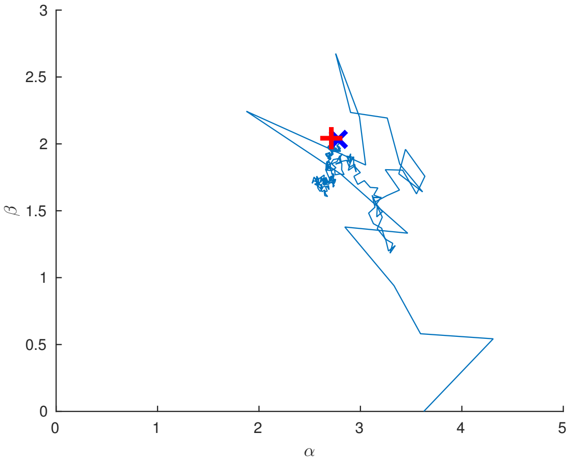

Figure 6: Trajectory of consecutive maximum-likelihood position estimates as the number of data points increases. The final point in the trajectory is marked with a blue ‘x’. The true location is marked with a red cross.

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 (α_t, β_t).[1]

ANSWER: See `a008.m` and Figures 6 and 7. Note that this is not a form of sequential learning, since each time the entire data set is used to find the maximum likelihood.

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

[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).

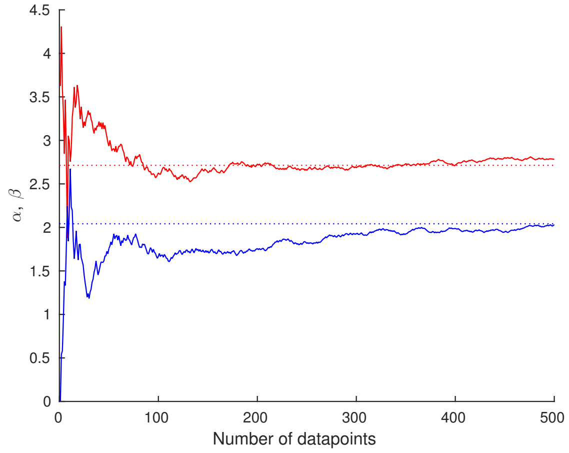

Figure 7: Maximum-likelihood position estimates versus number of flashes (α is red and β is blue). The true location is indicated by dotted lines.

$\{x_2, t_2\} = (0.6, -0.4)$.

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.

   ANSWER: As before, computing the predictive distribution implies calculating

$$\begin{aligned}
&p(t|x, \mathbf{x}, \mathbf{t}) = \mathcal{N}(t|m(x), s^2(x)); \\
&\qquad m(x) = \boldsymbol{\phi}(x)^T \mathbf{m}_N = N\beta \begin{pmatrix} 1 & x \end{pmatrix} \mathbf{S}_N \begin{pmatrix} \bar{\mu}_t \\ \bar{\mu}_{xt} \end{pmatrix}; \\
&\qquad s^2(x) = \beta^{-1} + \boldsymbol{\phi}(x)^T \mathbf{S}_N \boldsymbol{\phi}(x) = \beta^{-1} + \begin{pmatrix} 1 & x \end{pmatrix} \mathbf{S}_N \begin{pmatrix} 1 \\ x \end{pmatrix}; \\
&\qquad \mathbf{S}_N^{-1} = \begin{pmatrix} \alpha & 0 \\ 0 & \alpha \end{pmatrix} + N\beta \begin{pmatrix} 1 & \bar{\mu}_x \\ \bar{\mu}_x & \bar{\mu}_{xx} \end{pmatrix}.
\end{aligned}$$

   For the case of two data points, $\{x_1, t_1\} = (0.4, 0.1)$ and $\{x_2, t_2\} = (0.6, -0.4)$, we have

$$\bar{\mu}_t = \frac{1}{N} \sum_n t_n = -0.15 \qquad \bar{\mu}_{xt} = \frac{1}{N} \sum_n x_n t_n = -0.1$$

$$\bar{\mu}_x = \frac{1}{N} \sum_n x_n = 0.5 \qquad \bar{\mu}_{xx} = \frac{1}{N} \sum_n x_n^2 = 0.26$$

   which gives, with $N = 2$, $\alpha = 1$ and $\beta = 15$,

$$\begin{aligned}
\mathbf{S}_N^{-1} &= \begin{pmatrix} 31 & 15 \\ 15 & 8.8 \end{pmatrix} \implies \mathbf{S}_N \simeq \begin{pmatrix} 0.1841 & -0.3138 \\ -0.3138 & 0.6485 \end{pmatrix}; \\
\mathbf{m}_N &= N\beta \mathbf{S}_N \begin{pmatrix} \bar{\mu}_t \\ \bar{\mu}_{xt} \end{pmatrix} \simeq \begin{pmatrix} 0.113 \\ -0.5335 \end{pmatrix}.
\end{aligned}$$

   Substituting $\mathbf{S}_N$ into the equations for the mean $m(x)$ and the variance $s^2(x)$ results in

$$\begin{aligned}
m(x) &= 0.113 - 0.5335x; \\
s^2(x) &= 0.2508 - 0.6276x + 0.6485x^2.
\end{aligned}$$

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.

   ANSWER: The standard deviation needed in the plot below is given by $s(x) = \sqrt{s^2(x)}$, where $s^2(x)$ is computed in the previous exercise. See implementation in `a009.m`. The graph should look like the one in Figure 8.

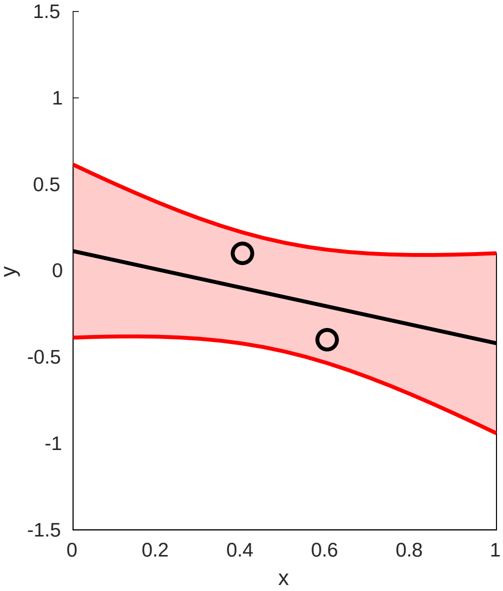

Figure 8: Mean function (black) ± standard deviation function (red)

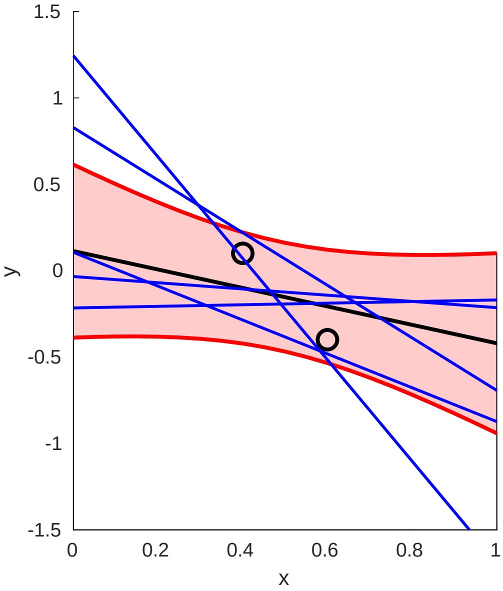

Figure 9: Mean function (black) ± standard deviation (red) and 5 samples (blue)

   The graph in the Matlab file resembles Figure 3.8b in Bishop. One difference is that the standard deviation takes its minimum *between* the two data points instead of at the points. Another difference is that the standard deviation increases much less as x moves further away from the data. The reason for this smaller increase is that our model is much more restrictive (linear) than the one used in the example from Figure 3.8 (high order polynomial), which makes possible regression functions much less flexible, so two points also give much less information on the overall behavior. Note that the fitted straight line $m(x)$ does not go through either point due to the prior on α.

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.

   ANSWER: See implementation in `a009.m`. You should obtain five straight lines that lie mostly in the shaded region, like in Figure 9.

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

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \eta \nabla E(\mathbf{x}_n) \tag{9}$$

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.

   ANSWER:
   $$x_{n+1} = x_n - \eta\lambda(x_n - a) = (1-\eta\lambda)x_n + \eta\lambda a$$
   The minimum of f is $x^* = a$. Fill in $a = (1-\eta\lambda)a + \eta\lambda a = a$.

   (b) Find η for which convergence is the fastest (actually in one step).

   ANSWER: With $\eta = 1/\lambda$,
   $$x_{n+1} = x_n - (x_n - a) = a$$
   So $x_1 = a$ for any x_0.

   (c) We will investigate the convergence properties for different values of η. For this we look at the ratio of the distance to the fixed point after the step and the distance before the step:
   $$r_n = \frac{|x_{n+1} - x^*|}{|x_n - x^*|}. \tag{10}$$

   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.

   ANSWER: Note that
   $$\frac{|x_n - x^*|}{|x_0 - x^*|} = \frac{|x_n - x^*|}{|x_{n-1} - x^*|}\frac{|x_{n-1} - x^*|}{|x_{n-2} - x^*|}\cdots\frac{|x_1 - x^*|}{|x_0 - x^*|} = \prod_{i=1}^{n}\frac{|x_{n-i+1} - x^*|}{|x_{n-i} - x^*|}$$
   Therefore: $r_n < c < 1$: convergent, $|x_n - x^*| < c^n|x_0 - x^*|$. $c < 1$, so $c^n \to 0$.
   $r_n > c > 1$: divergent, $|x_n - x^*| > c^n|x_0 - x^*|$. $c > 1$, so $c^n \to \infty$.

   (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 η is the algorithm convergent?

   ANSWER:
   $$\frac{|x_{n+1} - x^*|}{|x_n - x^*|} = \frac{|(1-\eta\lambda)x_n - (1-\eta\lambda)a|}{|x_n - a|} = |1 - \eta\lambda|$$
   For convergence, η must be such that $|1 - \eta\lambda| < 1$, so $0 < \eta\lambda < 2$. Now since both η and λ are larger than zero, only the upper bound is of interest:
   $$\eta < \frac{2}{\lambda}.$$

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

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 g is a fixed point.

ANSWER:

$$\begin{aligned} x_{n+1} &= (1-\eta\lambda_1)x_n + \eta\lambda_1 a_1 && (11)\\ y_{n+1} &= (1-\eta\lambda_2)y_n + \eta\lambda_2 a_2 && (12) \end{aligned}$$

The minimum of g is (a_1, a_2). The two equations are decoupled. Same as previous.

(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 η is smaller than this optimal value? What happens if it is larger?

ANSWER: Note that both iterations (i.e. in x and y) are independent, and can be treated so. For given η the convergence rate $r_{n,x}$ of the x equation is $r_{n,x} = |1-\eta\lambda_1|$, while the convergence rate $r_{n,y}$ of the y equation is $r_{n,y} = |1-\eta\lambda_2|$. Since the absolute value is a piecewise-defined function, we have to analyze two separate cases: $\eta \leq \dfrac{2}{\lambda_1+\lambda_2}$ and $\eta \geq \dfrac{2}{\lambda_1+\lambda_2}$.

- $\eta \leq \dfrac{2}{\lambda_1+\lambda_2}$: We first show that, under this constraint, $\max\{r_{n,x}, r_{n,y}\} = r_{n,x}$.

$$\begin{aligned} r_{n,x} \geq r_{n,y} &\iff |1-\eta\lambda_1| \geq |1-\eta\lambda_2| \\ &\iff |1-\eta\lambda_2| \leq 1-\eta\lambda_1 \ (1-\eta\lambda_1 \geq 0 \text{ when } \eta \leq \frac{2}{\lambda_1+\lambda_2}\ ) \\ &\iff \eta\lambda_1 - 1 \leq 1-\eta\lambda_2 \leq 1-\eta\lambda_1 \\ &\iff \eta(\lambda_1+\lambda_2) \leq 2 \leq 2+\eta(\lambda_2-\lambda_1) \end{aligned}$$

  In the last row, the first inequality holds because $\eta \leq \dfrac{2}{\lambda_1+\lambda_2}$, while the second holds because $\lambda_2 \geq \lambda_1$. Thus, we have proved that $r_{n,x} \geq r_{n,y}$ when $\eta \leq \dfrac{2}{\lambda_1+\lambda_2}$. Because $r_{n,x} = 1-\eta\lambda_1$ is decreasing as a function of η in this interval, the minimum is obtained at $\eta = \dfrac{2}{\lambda_1+\lambda_2}$, for which $r_{n,x} = \max\{r_{n,x}, r_{n,y}\} = \dfrac{\lambda_2-\lambda_1}{\lambda_1+\lambda_2}$.

- $\eta \geq \dfrac{2}{\lambda_1+\lambda_2}$: We first show that, under this constraint, $\max\{r_{n,x}, r_{n,y}\} = r_{n,y}$.

$$\begin{aligned} r_{n,x} \leq r_{n,y} &\iff |1-\eta\lambda_1| \leq |1-\eta\lambda_2| \\ &\iff |1-\eta\lambda_1| \leq \eta\lambda_2 - 1 \ (1-\eta\lambda_2 \leq 0 \text{ when } \eta \geq \frac{2}{\lambda_1+\lambda_2}\ ) \\ &\iff 1-\eta\lambda_2 \leq 1-\eta\lambda_1 \leq \eta\lambda_2 - 1 \\ &\iff 2+\eta(\lambda_1-\lambda_2) \leq 2 \leq \eta(\lambda_1+\lambda_2) \end{aligned}$$

  In the last row, the first inequality holds because $\lambda_2 \geq \lambda_1$, while the second holds because $\eta \geq \dfrac{2}{\lambda_1+\lambda_2}$. Thus, we have proved that $r_{n,x} \leq r_{n,y}$ when $\eta \geq \dfrac{2}{\lambda_1+\lambda_2}$. Because $r_{n,y} = \eta\lambda_2 - 1$ is decreasing as a function of η in this interval, the minimum is obtained at $\eta = \dfrac{2}{\lambda_1+\lambda_2}$, for which $r_{n,y} = \max\{r_{n,x}, r_{n,y}\} = \dfrac{\lambda_2-\lambda_1}{\lambda_1+\lambda_2}$.

We notice that in both intervals considered we arrived at the same solution. We can now conclude that the minimum of $\max\{r_{n,x}, r_{n,y}\}$ across the entire space of η is obtained at $\eta = \dfrac{2}{\lambda_2 + \lambda_1}$, i.e., $\arg\min_{\eta}\{\max\{r_{n,x}, r_{n,y}\}\} = \dfrac{2}{\lambda_2 + \lambda_1}$. □

If η is smaller than the optimal value, then gradient descent slows down in the flat direction. If η is larger than the optimal value, then gradient descent will overshoot more in the steep direction, which also results in a slower convergence, if $\eta < 2/\lambda_2$. If $\eta > 2/\lambda_2$, then $r_{n,y} > 1$, which means that gradient descent will diverge.

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

ANSWER:

$$r_{n,x} = r_{n,y} = r = \left|1 - 2\frac{\lambda_1}{\lambda_2 + \lambda_1}\right| = \left|\frac{\lambda_2 - \lambda_1}{\lambda_2 + \lambda_1}\right|$$

If $\lambda_2 \gg \lambda_1$, then r will be close to 1, which means that gradient descent will converge only very slowly.

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

ANSWER: See the MATLAB file a015.m

(e) Make some plots of the trajectories $\{(x_n, y_n)\}_{n=0}^{N}$ for different values of λ_i and η (η 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.

ANSWER: See Figure 10. If $\lambda_1 = \lambda_2 = \lambda$, then gradient descent converges in one step for the optimal value of $\eta = \frac{1}{\lambda}$. If η is in the interval $\left(0, \frac{1}{\lambda}\right)$, then the algorithm will converge more slowly (in more steps) to the minimum. If η is in the interval $\left(\frac{1}{\lambda}, \frac{2}{\lambda}\right)$, the algorithm will overshoot the minimum, but still converge at a slower rate. If $\eta > \frac{2}{\lambda}$, then gradient descent will diverge.

If $\lambda_1 \neq \lambda_2$, then the fastest convergence (according to our criterion) is for $\eta = \frac{2}{\lambda_1 + \lambda_2}$, as previously determined. If η is in the interval $\left(0, \frac{2}{\lambda_1 + \lambda_2}\right)$, then the algorithm slows down in the flat direction, which makes convergence slower. If η is in the interval $\left(\frac{2}{\lambda_1 + \lambda_2}, \frac{2}{\lambda_2}\right)$, then the algorithm overshoots in the steep direction, but still converges at a slower rate. If $\eta > \frac{2}{\lambda_2}$, then gradient descent diverges in the y-dimension, and if $\eta > \frac{2}{\lambda_1}$, then gradient descent diverges in the x-dimension.

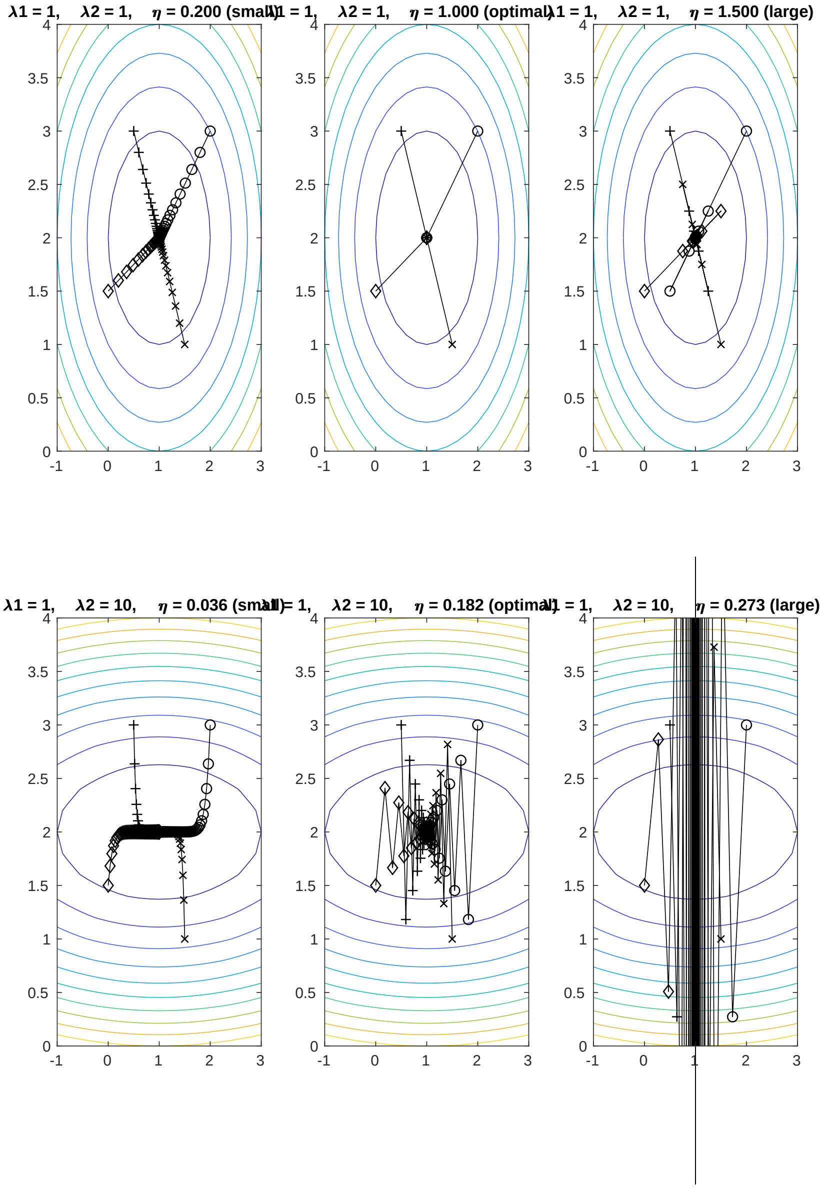

Figure 10: For six combinations of learning rate η and function parameters λ_1, λ_2: four different gradient descent trajectories, each with a different starting point, have been plotted on top of a contour plot of the function $g(x, y) = \frac{\lambda_1}{2}(x - a_1)^2 + \frac{\lambda_2}{2}(y - a_2)^2$.