

Problem Sheet 3

CMPE 58K, Bayesian Statistics and Machine Learning

Instructor: A. T. Cemgil

Due: 17 Dec 2009, 10:00.

Exercises are labelled with greek characters α, β, γ . Each label denotes the type of the question and roughly corresponds to its difficulty with α the hardest. A π denotes questions that have a programming component.

While some of the exercises are originals, many of the exercises are verbatim copies or slight modified versions of exercises from MacKay, Bishop or last years course. Often, the solutions can be found on the web or from "other" sources. It is OK to lookup the public domain solutions but resisting the temptation and attempting them first yourself would be a lot more useful for your understanding.

- **What to submit**

- A front page with your name, date of submissions and the list of the questions that you have solved. An example front page is given as the last page of this assignment sheet. Obeying the format would save my time.
- Don't send any executables, just the source code and an example run output is sufficient. However, write your programs clearly as some of these will be used as subroutines in later exercises.

- **How and when to submit**

- Hand in before the class on the day of the deadline.
- We accept printed or handwritten solutions. Use preferably both sides of pages. Electronic submissions are accepted only in extraordinary circumstances as it is harder for me to keep track of them (there is no TA for this course).
- Comment your source code. Avoid submitting pages of irrelevant computer code (driver routines, plotting), only focus on the 'gist'.

- **Grading**

- Each question is graded over 10. You get separate grades for each question.

A4.1 (α, π) (**Self Localisation on prime numbers, part 2**) A robot is moving across a circular corridor. We assume that the possible positions of the robot are elements of a discrete set with N locations, numbered as $i = 1, \dots, N$. The exact initial position of the robot is unknown but it is known to be located on one of the non-prime locations. At each step k , the robot stays where it is with probability ϵ , moves to the next point in counterclock direction with probability $(1 - \epsilon)/2$ or moves to the next point in clock direction with probability $(1 - \epsilon)/2$. At each step k , the robot can observe the color of the tile it is on, independently from previous readings. The corridor is designed such that the tiles with prime numbered locations $i = 2, 3, 5, 7, 11, \dots$ are white, others are blue. Due to the noise present at the visual sensor, the true color is observed only with probability δ , with probability $1 - \delta$ a white (blue) tile is observed as blue (white).

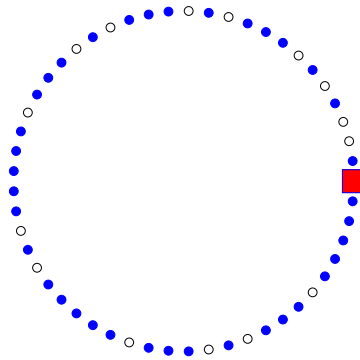


Figure 1: Robot (Square) moving in a circular corridor. Small circles denote the possible N tiles. Prime numbered tiles are white, others are blue. The robot can sense the color of the tile it is on.

- Implement a program that simulates this scenario; i.e., generates realisations from the movements of the robot and the associated sensor readings.
[Hint: You can use the `randgen` function you wrote earlier. Simulate a scenario for $k = 1, 2, \dots, K$ for $K = 500, N = 50, \epsilon = 0.3, \delta = 0.9$]
- (Filter)** Implement a program that computes, for each time step, the posterior distribution over the robots position, given the sensor readings so far.
- (Fixed lag smoother)** Implement a program that computes, for each time step k , the posterior distribution over the robots past L positions $p(x_l | y_{1:k}), k - L + 1 \leq l \leq k$.
- (Viterbi Path)** Implement a program that computes the most likely state trajectory, given all the observations.
- (Interpolation)** Implement a program that computes the smoothed state estimates, given observations $y_{1:L}$ and $y_{K-L+1:K}$ for any L such that $1 < L < K/2$.
- For all the tasks above run your program with at least two different ϵ and δ settings, and create figures similar to the ones shown in the lecture slides. Comment on self localisation performance. In particular comment how ϵ and δ effect it. Discuss if the prime numbers are special in some respect. Could we get the same performance with coloring, say, odd numbers ?

- (g) (**Parameter Estimation**) The goal of this exercise is to see if model parameters can be estimated from data as well. Let us denote the true parameters by $\theta_{\text{true}} = (\epsilon, \delta)$. Generate data from a model with $K = 500$ and $N = 50, \epsilon = 0.3, \delta = 0.9$. Compute the evidence $\mathcal{L}(\theta) = p(y_{1:K}|\theta)$ where θ is varied on a sufficiently dense grid on the unit square $[0, 1]^2$. Generate a contour plot of $\mathcal{L}(\theta)$ and compare its peak with θ_{true} .

A4.2 (α) (**Adding Poisson RV's**) An important property of Poisson random variables is that the sum is also Poisson distributed.

$$\begin{aligned}x_1 &\sim p_1(x_1) = \mathcal{PO}(x_1; \mu_1) \\x_2 &\sim p_2(x_2) = \mathcal{PO}(x_2; \mu_2) \\y &= x_1 + x_2\end{aligned}$$

- (a) Show that

$$p(y) = \sum_x p_1(x)p_2(y-x)$$

(That is the convolution of p_1 and p_2).

- (b) A probability generating function of a nonnegative discrete random variable is defined as

$$G_x(z) = \sum_{t=0}^{\infty} p(x=t)z^t$$

Show that when two pdf's are convolved, the resulting pdf has a probability generating function that is the product of the individual generating functions.

- (c) Derive the probability generating function for a Poisson random variable with density $\mathcal{PO}(x; \mu)$.
- (d) Using the above results, show that y has a Poisson distribution. Find the mean and variance of y .
- (e) Find the posterior distribution $p(x_1, x_2|y)$.
- (f) Find the posterior marginal $p(x_1|y)$.

A4.3 (γ) (**Woodbury Formula**) A very useful result from linear algebra is the *Woodbury* matrix inversion formula, also known as the Matrix inversion lemma, given by

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}$$

- (a) Verify the correctness of this formula by multiplying both sides by $(A + BCD)$.
- (b) Using the matrix inversion lemma, verify the following where I denotes identity matrices (not necessarily the same size)

$$\begin{aligned}(A - BCD)^{-1} &= A^{-1} + A^{-1}B(C^{-1} - DA^{-1}B)^{-1}DA^{-1} \\(I + B^{\top}DB)^{-1} &= I - B^{\top}(D^{-1} + BB^{\top})^{-1}B \\(A^{-1} + B^{\top}B)^{-1} &= A - AB^{\top}(I + BAB^{\top})^{-1}BA \\(A - C^{-1})^{-1} &= A^{-1} + A^{-1}(C - A^{-1})^{-1}A^{-1}\end{aligned}$$

- (c) Woodbury formula is particularly useful for reducing the computational load and improving stability in matrix computations. Assume D is a $N \times N$ **diagonal** matrix and v is a N vector. Assuming that inversion of a matrix is $O(N^3)$, estimate approximately the computational requirement for a direct evaluation of

$$G = (D + vv^\top)^{-1}$$

- (d) The inverse of a diagonal matrix is easy to compute in $O(N)$. Using the Woodbury formula, rewrite G to exploit this fact. Estimate the computational requirement and compare to the naive method.
- (e) (Optional) Implement both equations in matlab and verify your result. Compare the execution time for both implementations for $N = 100, 1000, 10000$.

A4.4 (γ) (**Partitioned Inverse Equations**) We partition a matrix Z as

$$Z = \begin{pmatrix} A & B \\ D & C \end{pmatrix}$$

and define the Schur complements of Z with respect to this partitioning as

$$\begin{aligned} M &= (A - BC^{-1}D) \\ N &= (C - DA^{-1}B) \end{aligned}$$

Verify the following

(a)

$$Z^{-1} = \begin{pmatrix} M^{-1} & -M^{-1}BC^{-1} \\ -C^{-1}DM^{-1} & C^{-1} + C^{-1}DM^{-1}BC^{-1} \end{pmatrix}$$

(b)

$$Z^{-1} = \begin{pmatrix} A^{-1} + A^{-1}BN^{-1}DA^{-1} & -A^{-1}BN^{-1} \\ -N^{-1}DA^{-1} & N^{-1} \end{pmatrix}$$

A4.5 (α) (**Multivariate Gaussian Distribution**) A convenient property of the multivariate Gaussian distribution is that its marginals and conditionals are Gaussians, hence can be expressed by a mean and a covariance parameter. In this exercise, we will investigate these important properties.

Consider the joint distribution over the variable

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

where the joint distribution is Gaussian $p(x) = \mathcal{N}(x; \mu, \Sigma)$ where

$$\begin{aligned} \mu &= \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} \\ \Sigma &= \begin{pmatrix} \Sigma_1 & \Sigma_{12} \\ \Sigma_{12}^\top & \Sigma_2 \end{pmatrix} \end{aligned}$$

(a) (Conditionals) Find the following

i. $p(x_1|x_2)$

ii. $p(x_2|x_1)$

(b) (Marginals) Find

i. $p(x_1)$

ii. $p(x_2)$

[Hint: Using the partitioned inverse equations, you need to rearrange

$$p(x_1, x_2) \propto \exp \left(-\frac{1}{2} \begin{pmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \end{pmatrix}^\top \begin{pmatrix} \Sigma_1 & \Sigma_{12} \\ \Sigma_{12}^\top & \Sigma_2 \end{pmatrix}^{-1} \begin{pmatrix} x_1 - \mu_1 \\ x_2 - \mu_2 \end{pmatrix} \right)$$

bring the expression in form of $p(x_1)p(x_2|x_1)$ (or $p(x_2)p(x_1|x_2)$) where the marginal and conditional can be easily identified. See also Bishop, section 2.3.]

This leads to a factorisation of form $p(x_2)p(x_1|x_2)$. The second decomposition will lead to the other factorisation $p(x_1)p(x_2|x_1)$.

A4.6 (β) (**Prediction and Update Equations**) Consider the following model:

$$\begin{aligned} x_0 &\sim \mathcal{N}(x_0; \mu_0, \Sigma_0) \\ x_1|x_0 &\sim \mathcal{N}(x_1; Ax_0, Q) \\ y_1|x_1 &\sim \mathcal{N}(y_1; Cx_1, R) \end{aligned}$$

(a) Draw the graphical model

(b) Find the following quantities and express them as Gaussian Distributions in moment parametrisation

i. $p(x_1)$

ii. $p(x_0, x_1)$

iii. $p(y_1)$

iv. $p(x_1|y_1)$

v. $p(x_0, x_1|y_1)$

A4.7 (γ) (**Multiplication of Gaussian Kernels**) Express

$$G(x) = K_1(x)K_2(x)$$

in form $e^\alpha \mathcal{N}(x; \mu, \Sigma)$ where for $i = 1, 2$

(a) $K_i = \mathcal{N}(x; 0, 1)$

(b) $K_i = \mathcal{N}(x; 0, P_i)$

(c) $K_i = \mathcal{N}(x; \mu_i, 1)$

(d) $K_i = \mathcal{N}(x; \mu_i, P_i)$

A4.8 (β) (**Log-partition function and its derivatives**) We call a probability distribution with density $p(x; \theta)$ on a set $\mathcal{X}^n \subset \mathbb{R}^n$ an exponential family¹ if it has the following functional form

$$\begin{aligned} p(x; \theta) &= \exp(\theta^\top \phi(x) - A(\theta)) \\ \int_{\mathcal{X}^n} dx p(x; \theta) &= \exp(-A(\theta)) \int_{\mathcal{X}^n} dx \exp(\theta^\top \phi(x)) = 1 \\ A(\theta) &= \log \int_{\mathcal{X}^n} dx \exp(\theta^\top \phi(x)) \end{aligned}$$

- The elements of the vector θ are known as *exponential* or *canonical* parameters.
- The functions $\phi(x)$ are the *sufficient statistics*.
- The function $A(\theta)$ is known as the *log partition function* or the *cumulant generating function* and ensures that the distribution normalizes to one. The log-partition function is defined through an integral. Hence we have to ensure that this integral exists (i.e. is finite). We define the set of valid parameters as $\Theta \equiv \{\theta | A(\theta) < \infty\}$

We already know that many well-known distributions belong to an exponential family. The derivatives of $A(\theta)$ provide the cumulants of the distribution².

(a) Show that

$$\frac{\partial}{\partial \theta} A(\theta) = \langle \phi(x) \rangle_{p(x; \theta)}$$

i.e., the derivative of the log-partition function gives the expected sufficient statistics

(b) Show that the Hessian (the matrix of second derivatives) is given as

$$\frac{\partial^2}{\partial \theta \partial \theta^\top} A(\theta) = \langle \phi(x) \phi(x)^\top \rangle - \langle \phi(x) \rangle \langle \phi(x) \rangle^\top$$

[Hint: This latter equation, by Jensen's inequality shows that the Hessian is always positive definite, hence $A(\theta)$ is convex.]

(c) Express the following distributions as an exponential family, identify the log partition function $A(\theta)$ and the canonical parameters θ , and by calculating the derivatives of $A(\theta)$, find the expected sufficient statistics

- Gaussian $\mathcal{N}(x; \mu, \Sigma)$
- Gamma $\mathcal{G}(x; a, b)$
- Beta $\mathcal{B}(x; \alpha, \beta)$
- Bernoulli $\mathcal{BE}(x; \pi)$
- Poisson $\mathcal{P}(x; \lambda)$

¹Note that in the lecture we gave a slightly more general definition of an exponential family including a scaling function $h(x)$.

²The mean is the first cumulant, the covariance is the second cumulant e.t.c. The term "cumulant" come from the fact that when two independent random variables are added, their cumulants are added (accumulated), too. Note the subtle difference between a moment generating function which is defined as $\langle \exp(\theta x) \rangle_{p(x)}$.

A4.9 (α) (**Gaussian Process Regression**) The goal of this exercise is to test your understanding of manipulations associated with multivariate Gaussians. You may also find it helpful to read Bishop 6.4.

In Bayesian machine learning, a frequent problem that pops up is the regression problem where we are given a pairs of inputs $x_i \in \mathbb{R}^N$ and associated noisy outputs $y_i \in \mathbb{R}$. We assume the following model

$$y_i \sim \mathcal{N}(y_i; f(x_i), R)$$

The interesting thing about a Gaussian process is that the function f is not specified in close form, but we assume that the function values

$$f_i = f(x_i)$$

are jointly Gaussian distributed as

$$\begin{pmatrix} f_1 \\ \vdots \\ f_L \end{pmatrix} = f_{1:L} \sim \mathcal{N}(f_{1:L}; 0, \Sigma(x_{1:L}))$$

Here, we define the entries of the covariance matrix $\Sigma(x_{1:L})$ as

$$\Sigma_{i,j} = K(x_i, x_j)$$

for $i, j \in \{1, \dots, N\}$. Here, K is a given *covariance function*. Now, if we wish to predict the value of \hat{f} for a new \hat{x} , we simply form the following joint distribution:

$$\begin{pmatrix} f_1 \\ \vdots \\ f_L \\ \hat{f} \end{pmatrix} \sim \mathcal{N}((f_{1:L}, \hat{f}); 0, \Sigma(x_{1:L}, \hat{x}))$$

Here, $\Sigma(x_{1:L}, \hat{x})$ is a $L + 1$ by $L + 1$ covariance matrix with entries

$$\Sigma_{i,L+1} = \Sigma_{L+1,i} = K(x_i, \hat{x}) = K(\hat{x}, x_i)$$

Popular choices of covariance functions to generate smooth regression functions include a Bell shaped one

$$K_1(x_i, x_j) = \exp\left(-\frac{1}{2}\|x_i - x_j\|^2\right)$$

and a Laplacian

$$K_2(x_i, x_j) = \exp\left(-\frac{1}{2}\|x_i - x_j\|\right)$$

(a) Derive the expressions to compute the predictive density

$$p(\hat{y}|y_{1:L}, x_{1:L}, \hat{x})$$

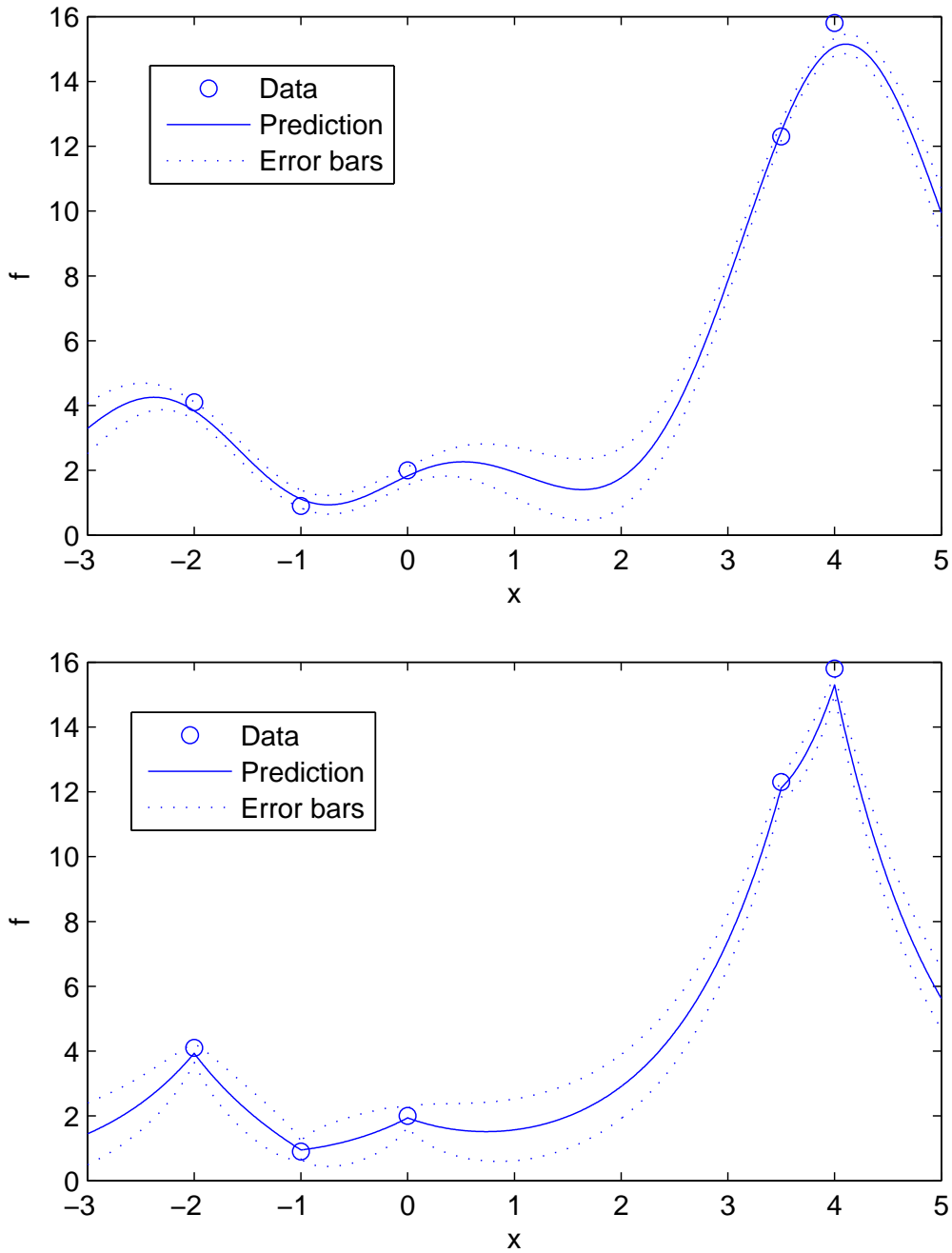


Figure 2: Gaussian Process Regression. Result obtained with a Bell shaped K_1 (Top) and Laplacian (Bottom) covariance function.

- (b) Write a program to compute the mean and covariance of $p(\hat{y}|y_{1:L}, x_{1:L}, \hat{x})$ to generate figures like Figure 2 for the following data:

$$\begin{aligned} \mathbf{x} &= [-2 \ -1 \ 0 \ 3.5 \ 4]'; \\ \mathbf{y} &= [4.1 \ 0.9 \ 2 \ 12.3 \ 15.8]'; \end{aligned}$$

Try different covariance functions and observation noise covariances R and comment on the nature of the approximation.

- (c) (20) Suppose we are using a covariance function parameterised by

$$K_{\beta}(x_i, x_j) = \exp\left(-\frac{1}{\beta}\|x_i - x_j\|^2\right)$$

Find the optimum regularisation parameter $\beta^*(R)$ as a function of observation noise variance via maximisation of the marginal likelihood, i.e.

$$\beta^* = p(y_{1:N}|x_{1:N}, \beta, R)$$

Generate a plot of $b^*(R)$ for $R = 0.01, 0.02, \dots, 1$ for the dataset given in (b).