

Problem Sheet 6

CMPE 58K, Bayesian Statistics and Machine Learning

Instructor: A. T. Cemgil

Due (Extended): 19 Jan 2009, Monday, 10:00.

This problem set **is** your final exam. You should get at least 20 points to pass this course.

Work out each problem clearly and explain your solution to get full credit. Most of the questions have a programming component.

Before 9 Jan, I will post on the course web page a set of test cases for questions 6.2 and 6.3. Make sure you report the results for these test cases.

A6.1 (50) (**AR(1) Model**) Consider the following model discussed in detail during the lectures.

$$\begin{aligned}A &\sim \mathcal{N}(A; 0, P) \\R &\sim \mathcal{IG}(R; \nu, \nu/\beta) \\x_k | x_{k-1}, A, R &\sim \mathcal{N}(x_k; Ax_{k-1}, R)\end{aligned}$$

where \mathcal{N} is a Gaussian and

$$\mathcal{IG}(R; a, b) = \exp\left(- (a + 1) \log R - \frac{b}{R} - \log \Gamma(a) + a \log b\right)$$

Caution: (This definition is different from the definition of \mathcal{IG} given in some of the earlier lectures.)

We are given the hyperparameters $\theta = (\nu, \beta, P)$

$$\begin{array}{lll} \nu & = & 0.4 \qquad \beta = 100 \qquad P = 1.2 \\ x_0 & = & 1 \qquad x_1 = -6 \end{array}$$

(a) (5 pts) Derive and implement an EM algorithm to find the MAP estimate

$$R^* = \operatorname{argmax}_R p(R | x_0, x_1, \theta)$$

(b) (5 pts) Derive and implement an EM algorithm to find the MAP estimate

$$A^* = \operatorname{argmax}_A p(A | x_0, x_1, \theta)$$

(c) (10 pts) Derive and implement an ICM (Iterative conditional modes) algorithm to find

$$(R^*, A^*) = \operatorname{argmax}_{A, R} p(A, R | x_0, x_1, \theta)$$

(d) (10 pts) Design an importance sampler to estimate the marginal likelihood

$$\log Z = \log p(x_1 = \hat{x}_1 | x_0 = \hat{x}_0, \theta)$$

and compute an estimate using $N = 100000$ samples.

(e) (20 pts) In the lectures, we have shown that the unnormalised posterior is

$$\begin{aligned} \phi &= p(A, R, x_1 = \hat{x}_1 | x_0 = \hat{x}_0, \theta) = \mathcal{N}(x_1; Ax_0, R) \mathcal{N}(A; 0, P) \mathcal{IG}(R; \nu, \nu/\beta) \\ &\propto \exp\left(-\frac{1}{2} \frac{x_1^2}{R} + x_0 x_1 \frac{A}{R} - \frac{1}{2} \frac{x_0^2 A^2}{R} - \frac{1}{2} \log 2\pi R\right) \\ &\quad \exp\left(-\frac{1}{2} \frac{A^2}{P} - \frac{1}{2} \log |2\pi P|\right) \\ &\quad \exp\left(-(\nu + 1) \log R - \frac{\nu}{\beta} \frac{1}{R} - \log \Gamma(\nu) + \nu \log(\nu/\beta)\right) \end{aligned}$$

We know also that the marginal log-likelihood

$$\log Z = \log p(x_1 = \hat{x}_1 | x_0 = \hat{x}_0, \theta)$$

is lower bounded by

$$\mathcal{B}_{VB} = \langle \log \phi \rangle_Q + H[Q]$$

where

$$\begin{aligned} Q &= q(A)q(R) \\ q(A) &= \mathcal{N}(A; m, \Sigma) \\ q(R) &= \mathcal{IG}(R; a, b) \end{aligned}$$

Extend the VB algorithm given in the slides so that you compute this bound at every iteration and plot the bound \mathcal{B} as a function of iterations. You should observe that the VB fixed point **monotonically** increases this lower bound. Restart your algorithm several times and compare the largest bound you find with the bound you find with importance sampling.

A6.2 (30) (**A Probability table**) The goal of this exercise is to investigate sampling algorithms and variational algorithms on a toy example and provide warm up for the following exercises.

Suppose we are given a $N_1 \times N_2$ table L where the entry at i 'th row and j 'th column denotes

$$L_{i,j} = \log p(x_1 = i, x_2 = j) + \log Z$$

where $p(x_1, x_2)$ is the joint distribution of x_1, x_2 with $x_1 \in \{1, \dots, N_1\}$ and $x_2 \in \{1, \dots, N_2\}$ and Z is an unspecified positive constant.

- (a) Write a program to compute the “variational marginals”, i.e., two distributions $q_1(x_1)$ and $q_2(x_2)$ such that

$$KL(q_1 q_2 || p)$$

is minimised. Compare the “variational marginals” with the exact marginals $p(x_1)$ and $p(x_2)$ given L .

A6.3 (70) (**A Chain**) Suppose we are given a probability distribution that factors according to

$$p(x_0, \dots, x_T) = \frac{1}{Z} \exp \left(\sum_{t=1}^T \psi(x_{t-1}, x_t) \right)$$

We know that $x_t \in \{1, \dots, N\}$

- (a) (10) Describe a procedure to compute Z and the marginals $p(x_t)$ for $t = 0, \dots, T$ exactly. Write a program to compute $\log Z$ and $\log p(x_t)$ for $t = 0, \dots, T$ given $\psi(x_{t-1} = i, x_t = j)$ as a $N \times N \times T$ array with `psi(i, j, t)`.

[Hint: Draw the factor graph and observe the similarities to the HMM derivation given in the lectures.]

- (b) (10) Derive an algorithm to compute the Viterbi path

$$x_{0:T}^* = \operatorname{argmax}_{x_{0:T}} p(x_0, \dots, x_T)$$

and write a program to compute it.

- (c) (15) Derive a simulated annealing (SA) algorithm to sample from

$$\frac{1}{Z_\beta} \exp \left(\beta \sum_{t=1}^T \psi(x_{t-1}, x_t) \right) = p_\beta(x_0, \dots, x_T)$$

where β is an inverse temperature variable. Design an annealing schedule $\beta \rightarrow \infty$ to compute the Viterbi path. Compare your solutions to the exact solution.

- (d) (15) Derive a variational Bayes algorithm that uses a fully factorised approximating distribution

$$Q = \prod_{t=0}^T q(x_t)$$

Derive the update equations and implement the algorithm. Compare the variational marginals to the true marginals.

- (e) (10) Derive the variational lower bound and write an algorithm to compute it. Compare to the exact $\log Z$ you have computed earlier. Show with a plot that it is strictly increasing during the iterations.
- (f) (10) The variational method can also be used to compute the Viterbi path by targeting p_β with $\beta \rightarrow \infty$. Using the same schedule as the SA, compare if you find better solutions with annealed VB. The solutions can be compared according to the number of mismatches with the true trajectory and the probability that the solution achieves.

A6.4 (50) (**Bayesian Estimation of Gaussians**) Consider the following model

$$\begin{aligned}\beta &\sim \mathcal{G}(\beta; \nu, 1) \\ \mu &\sim \mathcal{N}(\mu; 0, 1000) \\ x_i &\sim \mathcal{N}(x_i; \mu, \beta^{-1})\end{aligned}$$

for $i = 1 \dots N$. Suppose we are given the following dataset

$$x_{1:6} = \{-6, -1, -0.2, 0.1, 2, 4\} \equiv X$$

(a) (20) Derive and implement a Gibbs sampler to sample from

$$p(\mu, \beta | X, \nu = 0.1)$$

(b) (20) Derive and implement a variational Bayes algorithm to approximate $p(\mu, \beta | \nu = 0.1, X)$. Take as the approximating distribution a factorised distribution $Q = q_1 q_2$ where

$$\begin{aligned}q_1(\beta) &= \mathcal{G}(\beta; a, b) \\ q_2(\mu) &= \mathcal{N}(\mu; m, \Sigma)\end{aligned}$$

(c) (10) How would you find

$$\nu^* = \underset{\nu}{\operatorname{argmax}} p(x_{1:N} | \nu)$$

i.e., the ML estimate of ν ? Explain.

A6.5 (50) (**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) (10) Derive the expressions to compute the predictive density

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

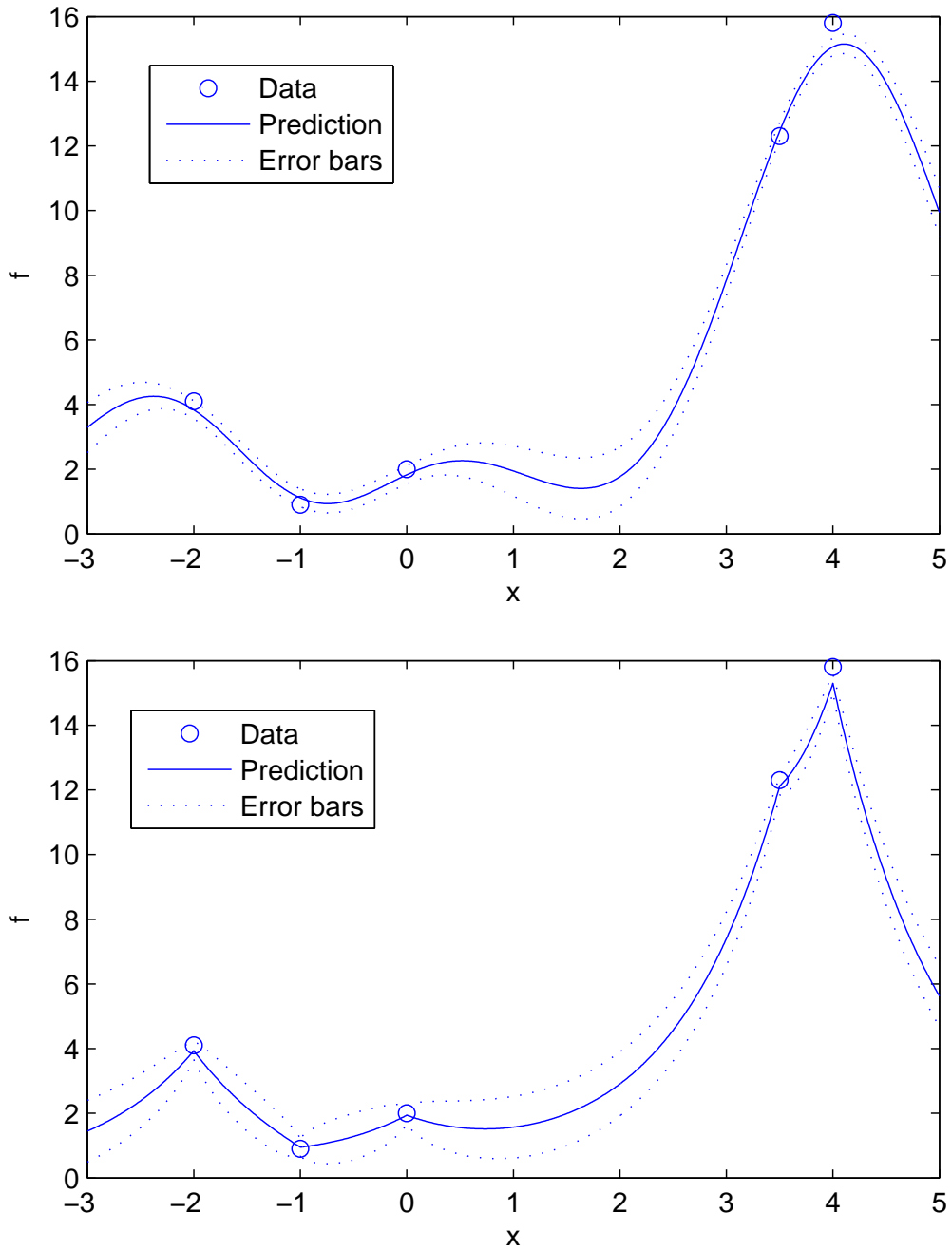


Figure 1: Gaussian Process Regression. Result obtained with a Bell shaped K_1 (Top) and Laplacian (Bottom) covariance function.

- (b) (20) 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 1 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).