TMA4300 Spring 2014 – Solution

Problem 1

a) The Box-Muller algorithm can be used to generate random samples from a normal distribution with mean equal to zero and variance equal to one. The algorithm produces from two uniformly distributed random variables on $(0, 1)$, two standard normal distributed samples. The algorithm proceeds as follows:

1. Generate two random variables u_1 and u_2 from $\mathcal{U}(0, 1)$.
2. Compute

$$x_1 = \sqrt{-2 \log u_1} \cos(2 \cdot \pi \cdot u_2)$$
$$x_2 = \sqrt{-2 \log u_1} \sin(2 \cdot \pi \cdot u_2)$$

3. Then x_1 and x_2 are $\mathcal{N}(0, 1)$ distributed.

To transform the samples so that they are $\mathcal{N}(\mu, \sigma^2)$ distributed, proceed as follows:

$$y_1 = \mu + \sigma \cdot x_1$$
$$y_2 = \mu + \sigma \cdot x_2$$

If only one sample or an odd number of samples is required, we return only one of y_1 and y_2.

b) To sample from a mixture of two normal distributions we use the methods based on mixtures. That means we first decide in which component of the mixture we are, i.e. we sample a random variable $u \sim \mathcal{U}(0, 1)$ and compare with the mixture weight w. If $u < w$ we are in the first component, and otherwise in the second component. Then conditioned on the component we proceed as follows: If in the first component, sample $y \sim \mathcal{N}(\mu_1, \sigma_1^2)$ using the method detailed in 1a) otherwise sample $y \sim \mathcal{N}(\mu_2, \sigma_2^2)$. This way y follows the provided mixture density.

Problem 2

a) Let $\pi(\boldsymbol{x})$ be the distribution of interest with $\boldsymbol{x} = (x^1, \ldots, x^p)$. Gibbs sampling is an MCMC scheme where we sequentially sample from the full-conditional distributions $\pi(x_i|\boldsymbol{x}_{-i})$. Here, $\boldsymbol{x}_{-i}$ refers to vector $\boldsymbol{x}$ but without the i-th element. That means to obtain samples $\boldsymbol{x}_i$ from the joint target distribution $\pi(\boldsymbol{x})$ do the following:

1. Select starting values $\boldsymbol{x}_0$ and set $i = 0$.
2. Repeatedly:

$$\begin{aligned}
&\text{Sample} \quad x_{i+1}^1|\cdot \sim \pi(x^1|x_i^2, \ldots, x_i^p) \\
&\text{Sample} \quad x_{i+1}^2|\cdot \sim \pi(x^2|x_{i+1}^1, x_i^3, \ldots, x_i^p) \\
&\vdots \\
&\text{Sample} \quad x_{i+1}^{p-1}|\cdot \sim \pi(x^{p-1}|x_{i+1}^1, x_{i+1}^2, \ldots, x_{i+1}^{p-2}, x_i^p) \\
&\text{Sample} \quad x_{i+1}^p|\cdot \sim \pi(x^p|x_{i+1}^1, \ldots, x_{i+1}^{p-1})
\end{aligned}$$

   where $|\cdot$ denotes conditioning on the most recent updates of all other elements of $\boldsymbol{x}$.
3. Increment i and go to step 2.

Of course, the full-conditionals must be known and ideally it is easy to sample from them. If for some components we need a Metropolis-Hastings step, we talk about Metropolis-within-Gibbs. Note, it is possible to sample blocks of the x^j together.

b) The full-conditional distributions can be derived from the posterior distribution:

$$p(\boldsymbol{\eta}, \boldsymbol{\theta}, \kappa_z, \kappa_\theta|\boldsymbol{y}) \propto \prod_{i,j} p(y_{ij}|\eta_{ij}) \cdot p(\eta_{ij}|\theta_j, \kappa_z) \cdot p(\boldsymbol{\theta}|\kappa_\theta) \cdot p(\kappa_z) \cdot p(\kappa_\theta)$$

The full-conditionals follow by omitting all terms of the posterior distribution that do not depend on the parameter(s) of interest. Thus we get that:

$$\begin{aligned}
p(\kappa_z|.) &\propto \prod_{i,j} p(\eta_{ij}|\theta_j, \kappa_z) \cdot p(\kappa_z) \\
&\propto \prod_{i=1}^{I} \prod_{j=1}^{J} \kappa^{1/2} \exp(-\frac{\kappa_z}{2}(\eta_{ij} - \theta_j)^2) \cdot \kappa_z^{\alpha_z - 1} \exp(-\beta_z \kappa_z) \\
&\propto \kappa_z^{\frac{I \cdot J}{2} + \alpha_z - 1} \exp(-\kappa_z(\beta_z + \frac{1}{2} \sum_{i,j} (\eta_{ij} - \theta_j)^2)
\end{aligned}$$

Thus, $\kappa_z|. \sim \text{Gamma}(\frac{I \cdot J}{2} + \alpha_z, \beta_z + \frac{1}{2} \sum_{i,j} (\eta_{ij} - \theta_j)^2)$.

$$
\begin{aligned}
p(\kappa_\theta|.) &\propto p(\boldsymbol{\theta}|\kappa_\theta)p(\kappa_\theta) \\
&\propto \kappa_\theta^{\frac{J-1}{2}} \exp(-\frac{\kappa_\theta}{2}\boldsymbol{\theta}^\top \mathbf{R}\boldsymbol{\theta}) \cdot \kappa_\theta^{\alpha_\theta - 1} \exp(-\beta_\theta \kappa_\theta) \\
&\propto \kappa_\theta^{\frac{J-1}{2}+\alpha_\theta-1} \exp(-\kappa_\theta(\beta_\theta + \frac{1}{2}\boldsymbol{\theta}^\top \mathbf{R}\boldsymbol{\theta}))
\end{aligned}
$$

Thus, $\kappa_\theta|. \sim \text{Gamma}(\frac{J-1}{2} + \alpha_\theta, \beta_\theta + \frac{1}{2}\boldsymbol{\theta}^\top \mathbf{R}\boldsymbol{\theta})$.

$$
\begin{aligned}
p(\eta_{ij}|.) &\propto p(y_{ij}|\eta_{ij})p(\eta_{ij}|\theta_j, \kappa_z) \\
&\propto \exp(\eta_{ij})^{y_{ij}} \exp(-E_{ij}\exp(\eta_{ij})) \exp(-\frac{\kappa_z}{2}(\eta_{ij}-\theta_j)^2) \\
&\propto \exp(-\frac{\kappa_z}{2}(\eta_{ij}-\theta_j)^2 + y_{ij}\eta_{ij} - E_{ij}\exp(\eta_{ij}))
\end{aligned}
$$

This is no standard distribution.

$$
\begin{aligned}
p(\boldsymbol{\theta}|.) &\propto \prod_i \prod_j \exp\left(-\frac{\kappa_z}{2}(\eta_{ij}-\theta_j)^2\right) \cdot \exp\left(-\frac{\kappa_\theta}{2}\boldsymbol{\theta}^\top \mathbf{R}\boldsymbol{\theta}\right) \\
&= \exp\left(-\frac{\kappa_z}{2}\sum_i \sum_j (\eta_{ij}-\theta_j)^2 - \frac{\kappa_\theta}{2}\boldsymbol{\theta}^\top \mathbf{R}\boldsymbol{\theta}\right) \\
&= \exp\left(-\frac{\kappa_z}{2}\sum_i \begin{pmatrix} \theta_1 - \eta_{i1} \\ \vdots \\ \theta_J - \eta_{iJ} \end{pmatrix}^\top \begin{pmatrix} \theta_1 - \eta_{i1} \\ \vdots \\ \theta_J - \eta_{iJ} \end{pmatrix} - \frac{\kappa_\theta}{2}\boldsymbol{\theta}^\top \mathbf{R}\boldsymbol{\theta}\right)
\end{aligned}
$$

Thus, $\boldsymbol{\theta}|. \sim \mathcal{N}(\mathbf{Q}^{-1}\boldsymbol{m}, \mathbf{Q}^{-1})$, with

$$
\mathbf{Q} = \kappa_\theta \mathbf{R} + \kappa_z \cdot I\mathbf{I} \qquad\qquad \boldsymbol{m} = \kappa_z \cdot \left(\sum_{i=1}^I \eta_{i1}, \ldots, \sum_{i=1}^I \eta_{iJ}\right)^\top
$$

Problem 3

a) – The "burn-in" period refers to the iterations the sampler needs to converge to the stationary/target distribution. The first iterations are strongly influenced by the starting value of the Markov chain. Including these samples would make the estimation of characteristics of the target distribution less accurate.

- We have here a random walk proposal. A new value $x^\star$ is sampled from a normal distribution centered around the current value x. As the proposal distribution is symmetric i.e. $\pi(x|x^\star) = \pi(x^\star|x)$ we obtain that the proposal ratio is equal to one and thus the log-proposal ratio is equal to zero.

b)
- According to the traceplots, the acceptance rate will be high for this sampler, as only very small changes are proposed in each iteration and essentially all proposals are accepted.
- The parameter `sd` is a tuning parameter for the algorithm, and will determine how often values $x^\star$ are accepted or rejected. To explore the parameter space more efficiently, a higher value of `sd` should be used. Then newly proposed values, $x^\star$, will be a bit more far away from the current value, x. Therefore the proposal will be accepted less often.

c)
- It means, that the estimated number of independent samples needed to obtain an estimate with the same precision as the MCMC based on 1000 samples is 23. Thus, the algorithm is not very efficient as samples are highly correlated.
- The quantity $q = P(X > 0)$ can be estimated by

$$\frac{1}{n}\sum_{i=1}^{n} I(x_i > 0),$$

where x_i refers to the i-th, $i = 1, \ldots, n$, out of n posterior samples and $I(.)$ denotes the indicator function.

Problem 4

a) Cross-validation can be use to estimate the misclassification rate of a statistical classification method. k-fold cross-validation involves randomly dividing the set of observations into k groups, or folds, $A_1, \ldots, A_k$ of approximately equal size. For the j-th fold (test set), we fit the model to the other $k-1$ folds (training set) of the data, and count the number of misclassifications of the fitted model when predicting the j-th part of the data. We do this for $j = 1, 2, \ldots, k$ and combine the k estimates into the misclassification rate as follows:

$$\frac{1}{n}\sum_{j=1}^{k}\left[\sum_{i\in A_j} 1(y_i \neq \hat{y}_{-A_j}(x_i))\right].$$

b) Leave-one-out cross-validation (LOOCV) is a special case of k-fold cross-validation in which k is set equal to n.

The most obvious advantage of k-fold cross-validation with $k < n$ is computational. LOOCV requires fitting the statistical learning method n times, whereas say 10-fold cross-validation requires fitting the classifier only ten times.

LOOCV will give approximately unbiased estimates for the true misclassification rate, since each training set contains $n - 1$ observations, which is almost as many as the number of observations in the full data set. Since statistical methods tend to perform worse when trained on fewer observations, this suggests that LOOCV is almost unbiased. And performing k-fold CV for, say, $k = 5$ or $k = 10$ will lead to an intermediate level of bias, since each training set contains $(k - 1)n/k$ observations. Therefore, from the perspective of bias reduction, LOOCV is to be preferred to k-fold CV.

However, the LOOCV can have high variance as the n training sets are so similar to one another. When we perform LOOCV, we are in effect averaging the outputs of n fitted models, each of which is trained on an almost identical set of observations; therefore, these outputs are highly (positively) correlated with each other. In contrast, when we perform k-fold CV with $k < n$, we are averaging the outputs of k fitted models that are somewhat less correlated with each other, since the overlap between the training sets in each model is smaller. Since the mean of many highly correlated quantities has higher variance than does the mean of many quantities that are not as highly correlated, the test error estimate resulting from LOOCV tends to have higher variance than does the test error estimate resulting from k-fold CV.