TMA4300 - Spring 2021 - Solution sketch (not all details are given)

Direct Sampling

Part a)

The cdf of the standard Weibull distribution is

$$F_0(x) = \int_0^x \alpha u^{\alpha-1} e^{-u^\alpha} du = 1 - e^{-x^\alpha}$$

we then need to invert the cdf and find $F_0^{-1}(u)$ as

$$\begin{aligned} 1 - e^{-x^\alpha} &= u \\ e^{-x^\alpha} &= 1 - u \\ x &= (-\log(1-u))^{(1/\alpha)} \end{aligned}$$

To simulate $x \sim f_0(x)$ we can use the following algorithm

- Simulate $U \sim \text{Unif}(0,1)$
- Compute $X = F_0^{-1}(u)$
- Return X

Part b)

We want to show that if $Y \sim N(0,1)$ then $X = F_0^{-1}(\Phi(Y))$ is standard Weibull distributed.

$$\begin{aligned} P(X < x) &= P(F_0^{-1}(\Phi(Y)) < x) \\ &= P(\Phi(Y) < F_0(x)) \\ &= P(Y < \Phi^{-1}(F_0(x))) = \Phi(\Phi^{-1}(F_0(x))) = F_0(x) \end{aligned}$$

To simulate two dependent and standard Weibull distributed random variables (X_1, X_2) one can use the following algorithm

- Simulate $(Y_1, Y_2) \sim N(0, \Sigma)$
- Compute $X_1 = F_0^{-1}(\Phi(Y_1))$ and $X_2 = F_0^{-1}(\Phi(Y_2))$
- Return X_1 and X_2

MCMC

Part a)

A sufficient condition for convergence is that the chain fullfills the detailed balance condition, that is

$$\pi(y)P(y,x) = \pi(x)P(x,y)$$

where $P(x,y)$ is the transition probability. In the given algorithm the transition probability is given by

$$\begin{aligned} P(x,y) &= Q(x,y)\alpha_1(x,y) && \text{if } x \neq y \\ P(x,y) &= Q(x,y)\alpha_1(x,y) + \int_{x'} Q(x',y)(1-\alpha_1(x',y))dx' && \text{if } x = y \end{aligned}$$

If $x = y$ the detailed balance condition is obviously fulfilled. If $x \neq y$ we have that

$$\begin{aligned}\pi(x)P(x,y) &= \pi(x)Q(x,y)\alpha_1(x,y)\\ &= \pi(x)Q(x,y)\frac{\pi(y)Q(y,x)}{\pi(y)Q(x,y)+\pi(x)Q(y,x)} = \\ &= \pi(y)Q(y,x)\frac{\pi(x)Q(x,y)}{\pi(y)Q(x,y)+\pi(x)Q(y,x)} = \pi(y)P(y,x)\end{aligned}$$

Part b) The acceptance probability for the MH agrithm is

$$\alpha_2(x,y) = \min\left\{1, \frac{\pi(y)Q(y,x)}{\pi(x)Q(x,y)}\right\}$$

If $\pi(y)Q(y,x) > \pi(x)Q(x,y)$ then $\alpha_2(x,y) = 1 \geq \alpha_1(x,y)$.

If $\pi(y)Q(y,x) < \pi(x)Q(x,y)$ then $\alpha_2(x,y) = \frac{\pi(y)Q(y,x)}{\pi(x)Q(x,y)} \geq \frac{\pi(y)Q(y,x)}{\pi(x)Q(x,y)+\pi(y)Q(y,x)} = \alpha_1(x,y)$

This means that, given the same proposal, the MH algorithm has a higher acceptance rate. According to the theorem, moreover, we have that the variance of the estimator from the MH algorith is smaller than the one of the estimator from the given algorithm.

Part b)

We simulate $\mathbf{x}^t = (x_1^t, x_2^t, x_3^t, x_4^t)$, $t = 1, \dots, N$ from the MCMC algorithm. After discarding the first L samples as burn-in we can estimated q as:

$$\hat{q} = \frac{1}{N-L}\sum_{i=L+1}^{N} I(\frac{x_1^i + x_3^i}{x_4^i})$$

where $I()$ is an indicator function.

INLA

Part a)

In order for the model to be amenable to INLA we need to have

- $\pi(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} \pi(y_i|x_i)$
- $\pi(\mathbf{x}|\theta) \sim N(0, Q^{-1}(\theta))$ where $Q^{-1}(\theta)$ is a sparse precision matrix

Part b)

Advantages: - INLA is fast and deterministic (no need to wait for convergence and for convercenge diagnostic) - INLA is already implemented in the R-INLA library - INLA gives often estimates that are more accurate than MCMC algorithms

Disadvantages: - INLA can only be applied to Latent Gaussian models with a sparse precision matrix governing the latent Gaussian field - INLA only provides estimates for posterior marginals while MCMC gives estimates for the whole joint posterior distribution.

Rejection Sampling

Part a)

The proposal $X = x$ is accepted

$$\begin{aligned}
P(X \text{ is accepted}) &= P\{U \leq \frac{h(X)}{K\tilde{g}(x)}\} + P\{U > \frac{h(X)}{K\tilde{g}(x)}, V \leq \frac{\tilde{\pi}(X) - h(X)}{K\tilde{g}(X) - h(X)}\} \\
&= P\{U \leq \frac{h(X)}{K\tilde{g}(x)}\} + P\{U > \frac{h(X)}{K\tilde{g}(x)}\} P\{V \leq \frac{\tilde{\pi}(X) - h(X)}{K\tilde{g}(X) - h(X)}\} \\
&= \frac{h(X)}{K\tilde{g}(x)} + (1 - \frac{h(X)}{K\tilde{g}(x)}) \frac{\tilde{\pi}(X) - h(X)}{K\tilde{g}(X) - h(X)} \\
&= \frac{h(X)}{K\tilde{g}(x)} + \frac{K\tilde{g}(X) - h(X)}{K\tilde{g}(x)} \frac{\tilde{\pi}(X) - h(X)}{K\tilde{g}(X) - h(X)} \\
&= \frac{h(X)}{K\tilde{g}(x)} + \frac{\tilde{\pi}(X) - h(X)}{K\tilde{g}(x)} = \frac{\tilde{\pi}(X)}{K\tilde{g}(x)}
\end{aligned}$$

We then have to find the distribution of the samples accepted by the algorithm. We have that

$$F(x) = P(X < x | x \text{ is accepted}) = \frac{P(X < x \text{ and } x \text{ is accepted})}{x \text{ is accepted}}$$

$$= \frac{\int_{-\infty}^{x} \frac{\tilde{\pi}(u)}{K\tilde{g}(u)} g(u) du}{\int_{-\infty}^{\infty} \frac{\tilde{\pi}(u)}{K\tilde{g}(u)} g(u) du}$$

$$= \frac{\int_{-\infty}^{x} \frac{Z_\pi \pi(u)}{Z_g g(u)} g(u) du}{\int_{-\infty}^{\infty} \frac{Z_\pi \pi(u)}{K Z_g g(u)} g(u) du}$$

$$= \frac{\int_{-\infty}^{x} \pi(u) du}{\int_{-\infty}^{\infty} \pi(u) du} = \int_{-\infty}^{x} \pi(u) du$$

Part b)

We have that

$$\begin{aligned}
P(\text{Carry out step 3}) &= 1 - P(\text{Accept in step 2}) \\
&= 1 - \int P(\text{Accept } x \text{ in step 2} | X = x) g(x) dx \\
&= 1 - \int \frac{h(u)}{K\tilde{g}(u)} g(x) dx \\
&= 1 - \int \frac{h(u)}{K Z_g g(u)} g(x) dx \\
&= 1 - \int \frac{h(u)}{K Z_g} dx = \frac{\int h(u)}{K Z_g}
\end{aligned}$$

Such algorithm can be convenient in cases when computing $\pi(x)$ is computationally heavy.

Bootstrap

Part b)

We regard the bootstrap sample as a set of observations, i.e. ordering is ignored. Since all entries are distinct, we can regard them as n different classes. Generating a bootstrap sample can be regarded as sampling from an urn with n different colored balls with replacement. This is equivalent to sampling from a multinomial distribution. Thus each bootstrap sample has probability:

$$\frac{n!}{z_1!\dots z_n!}(\frac{1}{n})^{z_1+\dots+z_n} = \frac{n!}{z_1!\dots z_n!}(\frac{1}{n})^n$$

Where, z_i represents the number of times the i-th observation is picked. The probability gets largest, if $z_1 = \dots = z_n = 1$ that means each entry appears once, i.e. we get the original data set. The probability gets smallest, if one of the z_i is equal n while all the others are 0, that means that only one of the element of the original dataset is picked up n times.