MAST30027: Modern Applied Statistics

Week 10 Lab Sheet

1. **Metropolis-Hastings**

   Recall that $\mathbf{X} = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, with $\boldsymbol{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}$ and $\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{pmatrix}$, iff $\mathbf{X}$ has joint density

   $$f_{\boldsymbol{\mu},\boldsymbol{\Sigma}}(\mathbf{x}) = \frac{1}{2\pi|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right),$$

   where $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$.

   (a) Write an R function that evaluates the density of a bivariate normal distribution. The function should take as input the point $\mathbf{x}$, the mean $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{\Sigma}$.

   You will find the functions `solve` and `det` useful.

   Solution:

```
> dbinorm <- function(x, mu, Si) {
+   # x and mu are vectors length 2 and Si a 2x2 matric
+   # returns the density at x of a bivariate normal mean mu var Si
+   exp(-t(x - mu)%*%solve(Si, x - mu)/2)/2/pi/sqrt(det(Si))
+ }
```

   You can check that it is working by noting that `dbinorm(c(1,1), c(0,0), matrix(c(1,0,0,1),2,2))` gives the same answer as `dnorm(1)^2`.

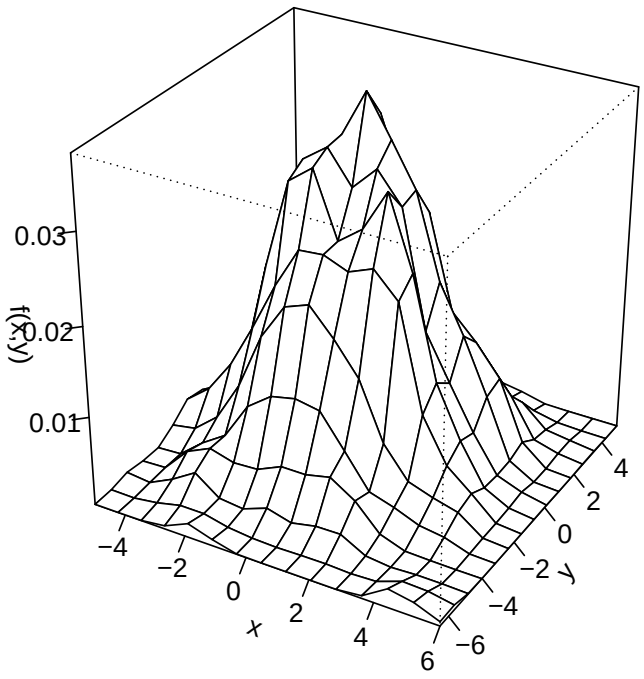
   (b) Write a program in R that uses the Metropolis-Hastings algorithm to generate a sample of size $n = 1000$ from the $N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 4 & 1 \\ 1 & 4 \end{pmatrix}\right)$ distribution. Use the symmetric random walk proposal distribution $N\left(\mathbf{x}, \sigma^2 I\right)$ with $\sigma = 2.5$.

   Use $\mathbf{X}(0) = \begin{pmatrix} 6 \\ -6 \end{pmatrix}$ as your initial state. Report the proportion of accepted values.

   Solution: The acceptance rate was 46% (this will vary a little every time you run the program). To check that the output looks ok, I have plotted (a kernel density estimate of) the joint density. Easier than plotting a joint density would be to plot histograms/densities of the marginal samples, using `hist` or `density`.

```
> # Metropolis-Hastings simulation of a bivariate normal
> # inputs
> mu <- c(0, 0)                      # mean
> Si <- matrix(c(4, 1, 1, 4), 2, 2) # variance
> iterations <- 1000      # sample size
> startvalue <- c(6, -6)  # initial value
> sd <- 2.5               # std-dev for proposal chain
> # main loop
> chain <- matrix(nrow = iterations+1, ncol = 2)
> chain[1,] <- startvalue
> accepted <- 0 # counts num accepted proposals
> for (i in 1:iterations){
+   proposal <- rnorm(2, chain[i,], sd)
+   prob <- dbinorm(proposal, mu, Si)/dbinorm(chain[i,], mu, Si)
+   if (is.nan(prob)) prob <- 0
+   if (runif(1) < prob) {
+     chain[i+1,] <- proposal
+     accepted <- accepted + 1
+   } else {
```

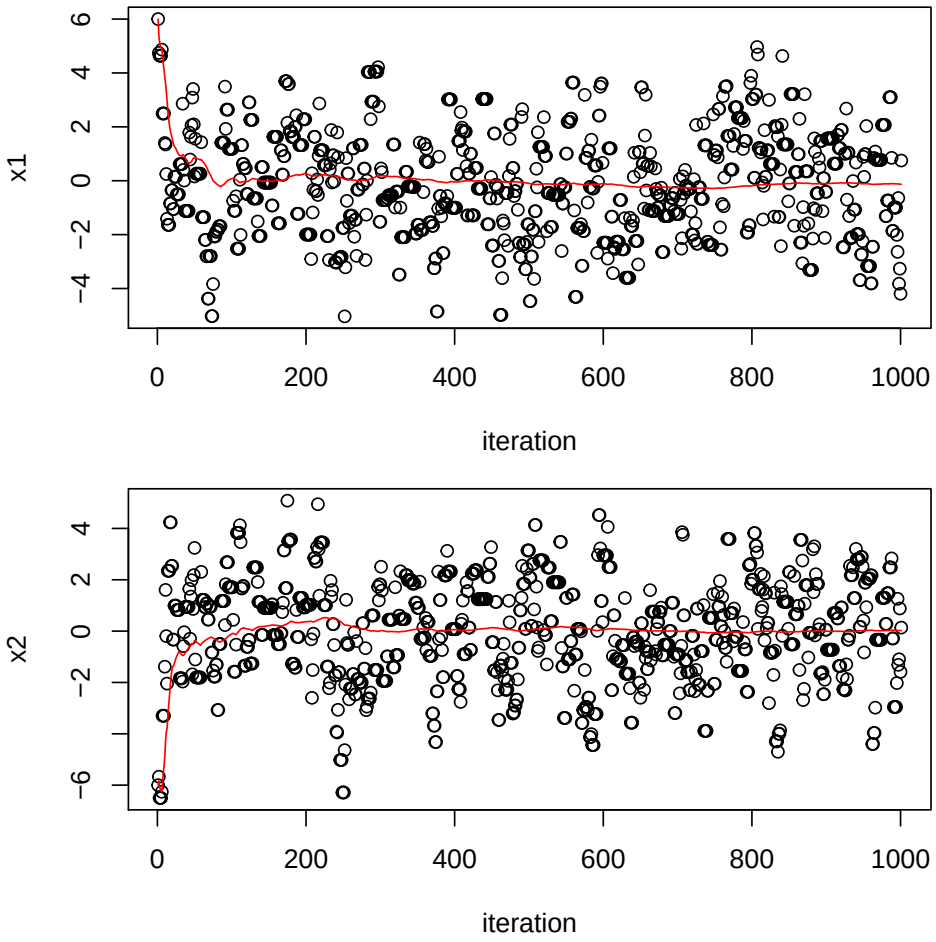
```
+     chain[i+1,] <- chain[i,]
+   }
+ }
> # acceptance rate
> accepted/iterations
[1] 0.457
> # 2D density plot
> library(MASS)
> chain_density <- kde2d(chain[,1], chain[,2], n = 15)
> persp(chain_density, phi = 30, theta = 30, d = 5,
+       xlab = "x", ylab = "y", zlab = "f(x,y)",
+       ticktype = "detailed")
```



(c) Let $\mathbf{X}(n)$ be the n-th sample point. Plot $X_i(n)$ and the cumulative averages $\overline{X}_i(n) = n^{-1}\sum_{j=1}^{n} X_i(j)$, for $i = 1, 2$. The cumulative averages should give a rough idea of how quickly the $\mathbf{X}(n)$ converge in distribution.

Solution:

```
> # cumulative averages
> par(mfrow=c(2,1), mar=c(4,4,1,1))
> plot(chain[,1], xlab = "iteration", ylab = "x1")
> cumavg1 <- cumsum(chain[,1])/1:(iterations + 1)
> lines(1:(iterations + 1), cumavg1, col = "red")
> plot(chain[,2], xlab = "iteration", ylab = "x2")
> cumavg2 <- cumsum(chain[,2])/1:(iterations + 1)
> lines(1:(iterations + 1), cumavg2, col = "red")
>
>
```



2. Suppose that $\mathbf{X} = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, with $\boldsymbol{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}$ and $\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{pmatrix}$.

   (a) Show that the conditional distribution of $X_1|X_2 = x_2$ is normal with mean $\mu_1 + (x_2 - \mu_2)\sigma_{12}/\sigma_2^2$ and variance $\sigma_1^2 - \sigma_{12}/\sigma_2^2$.

   Solution: Let $\boldsymbol{\Sigma}^{-1} = \begin{pmatrix} a & b \\ b & c \end{pmatrix}$, then the condional marginal distribution of X_1 given $X_2 = x_2$ is

$$\begin{aligned} \frac{f(x_1, x_2)}{f(x_2)} &\propto f(x_1, x_2) \\ &\propto \exp\{-\tfrac{1}{2}[(x_1 - \mu_1)^2 a + 2(x_1 - \mu_2)(x_2 - \mu_2)b + (x_2 - \mu_2)^2 c]\} \\ &\propto \exp\{-\tfrac{1}{2}[x_1^2 a - 2x_1(\mu_1 a - (x_2 - \mu_2)b]\} \\ &\propto \exp\{-\tfrac{1}{2}[x_1 - (\mu_1 - (x_2 - \mu_2)b/a)]^2 a\} \end{aligned}$$

   Thus $X_1|X_2 = x_2 \sim N(\mu_1 - (x_2 - \mu_2)b/a, 1/a)$, where $a = \sigma_2^2/(\sigma_1^2\sigma_2^2 - \sigma_{12}^2)$ and $b = -\sigma_{12}/(\sigma_1^2\sigma_2^2 - \sigma_{12}^2)$, and thus $b/a = -\sigma_{12}/\sigma_2^2$ and $1/a = \sigma_1^2 - \sigma_{12}^2/\sigma_2^2$.

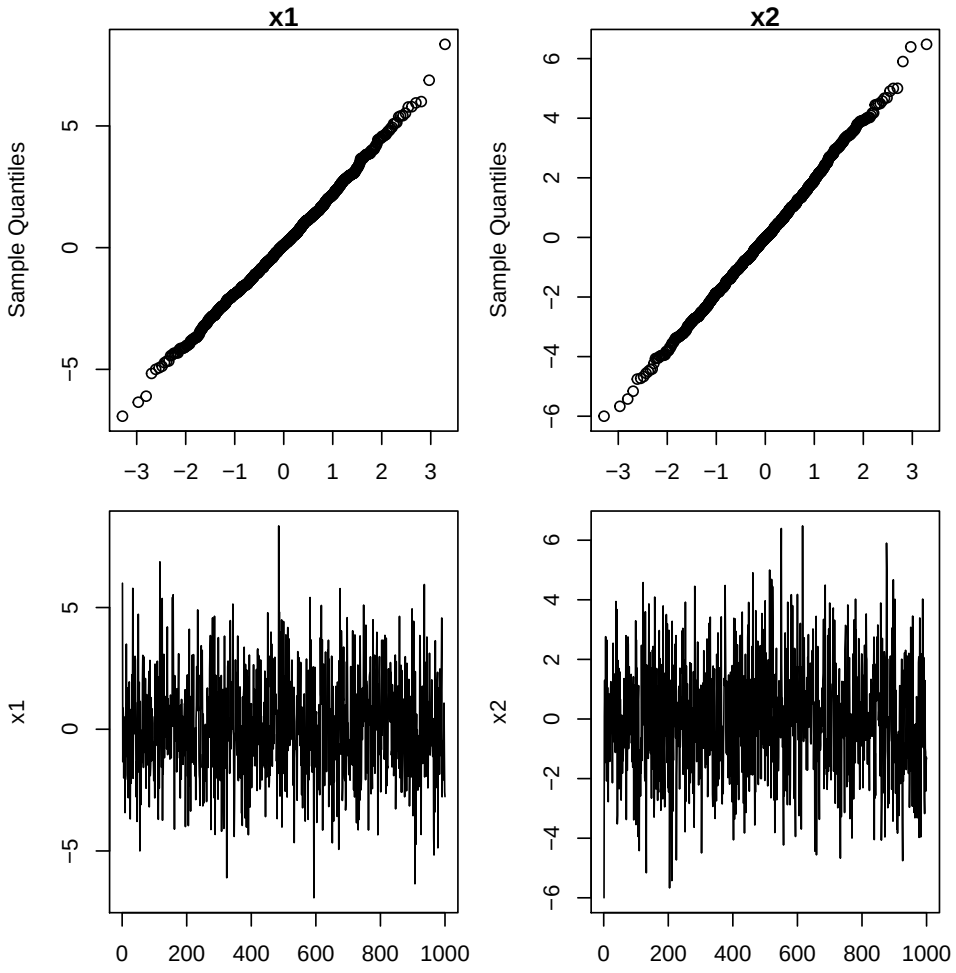
   (b) Write an R function that uses the Gibbs sampler to generate a sample of size $n = 1000$ from the $N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 4 & 1 \\ 1 & 4 \end{pmatrix}\right)$ distribution.

   Plot traces of X_1 and X_2.

   Solution: To test the simulator we do a normal probability plot for each marginal, and both look good. The traces show pretty good mixing.

```
> set.seed(200)
> # params
```

```
> mu1 <- 0
> mu2 <- 0
> s11 <- 4
> s12 <- 1
> s22 <- 4
> # initial values
> x1 <- 6
> x2 <- -6
> # sample size
> nreps <- 1000
> Gsamples <- matrix(nrow=nreps, ncol=2)
> Gsamples[1,] <- c(x1, x2)
> # main loop
> for (i in 2:nreps) {
+   x1 <- rnorm(1, mu1 + (x2 - mu2)*s12/s22, sqrt(s11 - s12/s22))
+   x2 <- rnorm(1, mu2 + (x1 - mu1)*s12/s11, sqrt(s22 - s12/s11))
+   Gsamples[i,] <- c(x1, x2)
+ }
> # output
> par(mfrow=c(2,2), mar=c(2,4,1,1))
> qqnorm(Gsamples[,1], main="x1")
> qqnorm(Gsamples[,2], main="x2")
> plot(Gsamples[,1], type="l", xlab="iteration", ylab="x1")
> plot(Gsamples[,2], type="l", xlab="iteration", ylab="x2")
```



(c) Use your simulator to estimate $\mathbb{P}(X_1 \geq 0, X_2 \geq 0)$. To get a feel for the convergence rate, calculate the estimate using samples $\{1, \ldots, k\}$, for $k = 1, \ldots, n$, and then plot the estimates against n.

Solution: The plot appears after part (d).

```
> par(mfrow=c(1,1))
> success <- apply(Gsamples, 1, function(x) (x[1] > 0)&(x[2] > 0))
> mean(success)
```

```
[1] 0.296
```

```
> plot(1:nreps, cumsum(success)/(1:nreps), type="l", xlab="k", ylab="prob", ylim=c(0,1))
```

(d) Now change $\mathbf{\Sigma}$ to $\begin{pmatrix} 4 & 2.8 \\ 2.8 & 4 \end{pmatrix}$ and generate another sample of size 1000.

What do the traces/estimates look like now?

Solution: We put `s12 <- 2.8` then re-run the code above, getting a different `Gsamples`. We plot the cumlative estimates on top of the previous graph using `lines`. The cumulative estimates are more volatile in the second case, reflecting the stronger autocorrelation in the Markov chain, caused by the stronger correlation between X_1 and X_2.

```
> success <- apply(Gsamples, 1, function(x) (x[1] > 0)&(x[2] > 0))
> mean(success)
```

```
[1] 0.38
```

```
> lines(1:nreps, cumsum(success)/(1:nreps), col="red")
```