# Coursework - Computational Statistics

CID: 02091191

November 22, 2021

**Abstract**

In this document we are presenting the work conducted during the second Computational Statistics Coursework of the year.

## 1 First question

Consider the density given by

$$f(x) = \frac{k}{1+|x-2|^3} \text{ for } x \in [0,5] \text{ and 0 otherwise}$$

where $k$ is a normalising (unknown at first) constant. Assume we want to sample from this density $f$ using Markov-Chain Monte-Carlo and especially Metropolis-Hastings sampling with a random walk proposal with additive noise following a normal distribution with standard deviation $\sigma$. The derivation of a such sampler is described in the Lecture Notes [1]. We implement this sampler directly in **R**.

### (a) $\sigma = 1$, estimations of $\mathbb{E}[X^3]$ and $\mathbb{P}[X < 1]$

First consider $\sigma = 1$ and run the sampler over 5000 iterations. We then generate data 5000 samples from a random variable $X$ with density $f$. We use a starting value of 1 and discard the first 100 samples as burn-in. We then estimate $\mathbb{E}[X^3]$ and $\mathbb{P}[X < 1]$ using Monte-Carlo integration:

$$\mathbb{E}[X^3] \simeq \frac{1}{n}\sum_{i=1}^{n} X_i^3 \text{ and } \mathbb{P}[X<1] \simeq \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{X_i<1}$$

We obtain the following estimates: $\mathbb{E}[X^3] \simeq 14.7740$ and $\mathbb{P}[X < 1] \simeq 0.1079$. We can compare these results to the "real" ones obtained using numerical integration in **R** for example using the `integrate()` function. Before proceeding to numerical integration in order to estimate $\mathbb{E}[X^3]$ and $\mathbb{P}[X < 1]$, we need to estimate the normalising constant $k$ as: $k = \left(\int_0^5 f(x)dx\right)^{-1}$. Hence, using numerical integration,

$$\mathbb{E}[X^3] = \int_0^5 x^3 f(x)dx \text{ and } \mathbb{P}[X<1] = \int_0^1 f(x)dx$$

We obtain in **R** the following values: $\mathbb{E}[X^3] = 14.1732$ and $\mathbb{P}[X < 1] = 0.1133$. Thus, the Metropolis-Hastings sampler seems to be correctly implemented as these results are similar to the ones obtained by the Markov-Chain Monte-Carlo procedure.

### (b) $\sigma = 0.05$, comparison of cdf

We now use $\sigma = 0.05$: hence, the additive noise has smaller variance and the jumps in the Monte-Carlo generation process should be smaller. We use the same Metropolis-Hastings sampler

described previously with $\sigma = 0.05$ for different starting values $1, 2, 3, 4$. We then plot in **R** the empirical cdf obtained from all generating processes and compare it to the real cdf $F$ obtained by numerical integration as

$$F(x) = \int_{-\infty}^{x} f(x)dx$$

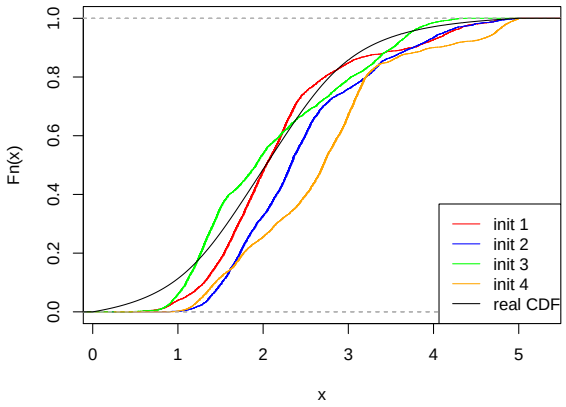



Figure 1.1: Comparison of empirical cdf from 4 different starting values with the real cdf

As shown in Figure 1.1, the empirical cdf (coloured) are significantly different from the expected theoretical cdf in black. This is the result of choosing a too small variance $\sigma$ in the random walk proposal. As $\sigma$ is low, the jumps in the Markov-Chain generating process are small and almost all accepted (high acceptance rate). Hence, the support of $f$ is not explored quickly and we should use a higher number of iterations. This illustrates the trade-off that has to be found between $\sigma$ and the number of iterations.

### (c) Efficiency of the Metropolis-Hastings sampler for $\sigma \in \{0.05, 1, 5\}$

To compare the sampling process with different values of $\sigma$, we may first focus on the acceptance rate obtained from all these different samplers with the same number of iterations $10^4$.

| | **Acceptance rate** |
|---|---|
| $\sigma = 0.05$ | 0.9827 |
| $\sigma = 1$ | 0.6688 |
| $\sigma = 5$ | 0.2132 |

Table 1: Acceptance rates obtained from Metropolis-Hastings samplers with $\sigma \in \{0.05, 1, 5\}$

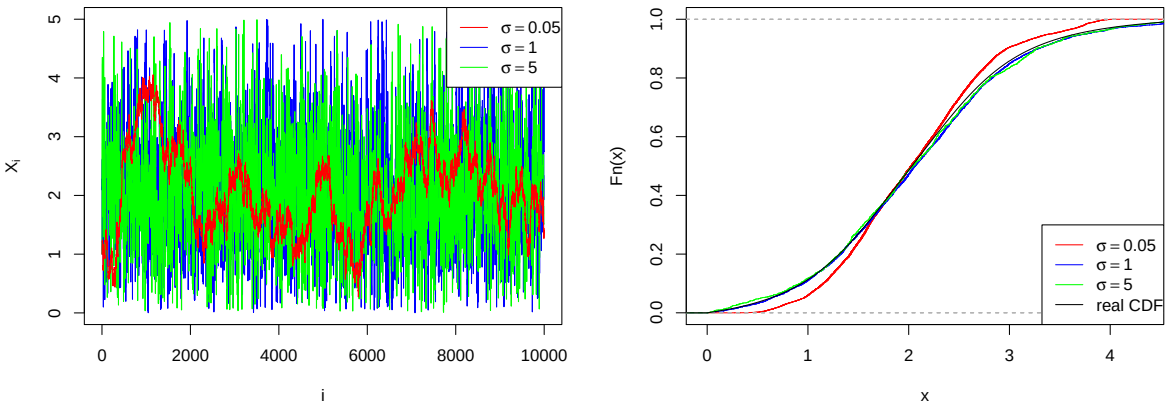
As expected and shown in Table 1, the acceptance rate for $\sigma = 0.05$ is much higher than the one of $\sigma = 5$. Thus, for $\sigma = 0.05$, the support of $f$ is not explored quickly and in the contrary for $\sigma = 5$ we are wasting a lot of simulations because most proposals are outside the range of $f$. A good trade-off would be $\sigma = 1$ as it has an acceptance rate close to 0.5 (heuristic).

We may moreover focus on the quality of distributions sampled for each value of $\sigma$. To do so, we could proceed to Kolmogorov-Smirnov tests in **R** in order to compare the samples (empirical) distributions to the theoretical one $F$. We would consider the null hypothesis $H_0$: the samples are coming from $f$ distribution against the alternative hypothesis $H_1$: the samples are not following $f$ distribution.

| | **p-value** |
|---|---|
| $\sigma = 0.05$ | $< 2.2 \cdot 10^{-16}$ |
| $\sigma = 1$ | 0.4549 |
| $\sigma = 5$ | $9.013 \cdot 10^{-9}$ |

Table 2: KS tests results for each value of $\sigma$

Table 2 shows that for $\sigma = 0.05$, we reject the null hypothesis that the samples are following $f$ distribution. However, for $\sigma = 1$, we fail to reject the null hypothesis and hence we should consider that the samples have the same distribution as $f$. Finally, for $\sigma = 5$, the $p$-value obtained is too low to accept the null hypothesis and therefore we would conclude that the samples are not coming from $f$ distribution even if it may be slightly better than for $\sigma = 0.05$.




(a) Random observations

(b) Empirical and real cdf for each value of $\sigma$

Figure 1.2: Random observations $X_i$ for each value of $\sigma$ and comparison of cdf

Finally, as shown in Figure 1.2b, the cdf of for $\sigma = 0.05$ is significantly different from the real cdf. Both cdf for $\sigma = 1, 5$ seem to be similar to the real cdf, so we may conclude that proceeding to only one graphical analysis may not have shown that the cdf of samples for $\sigma = 5$ are not significantly following $f$ distribution (purpose of KS tests). On Figure 1.2a, we may clearly see that the random walk proposal with $\sigma = 0.05$ is very different from the two other ones, and this results in a bad sampling process.

# 2 Second question

## (a) Confidence intervals for $\mathbb{E}[\cos(X)]$

Assume we want to compute a 95% confidence interval for $\mathbb{E}[cos(X)]$ using non-parametric bootstrap. We implement in **R** the computation of a standard 95% two-sided confidence interval and a Studentised 95% two-sided confidence interval. We obtain for the standard confidence interval $[0.0536, 0.6525]$ and for the Studentised confidence interval $[-0.1102, 0.6525]$.

## (b) Coverage probability

In order to examine the coverage probability of both methods, we generate 500 data sets $X_1, \ldots, X_n$ with $n = 15$ from a Student $t$-distribution with 4 degrees of freedom. Then, for each data set, we compute both confidence intervals, and check if the real value $\mathbb{E}[\cos(X)]$ when $X \sim t_4$ is inside the intervals or not. However, we do not know analytically $\mathbb{E}[\cos(X)]$. We

then may approximate it using numerical integration and Monte-Carlo integration. The estimates obtained are $\mathbb{E}[cos(X)] \simeq 0.5075$ for numerical integration and $\mathbb{E}[cos(X)] \simeq 0.4945$. As these values are similar, we may think that they are both close to the real value of $\mathbb{E}[\cos(X)]$. Therefore, in the following, we will assume that $\mathbb{E}[\cos(X)] = 0.5075$ (the numerical integration estimate). In **R**, we compute the coverage probability of each interval and obtain 0.844 for the standard confidence interval and 0.954 for the Studentised confidence interval. We may notice that the coverage probability for the standard confidence interval is less than the nominal coverage probability (95%) whereas the Studentised confidence interval has the same coverage probability as the nominal. Hence, the Studentised confidence interval performs much better.

### (c) Confidence interval for the median statistic

Suppose now we are interested in $T(X_1, \dots, X_n) = \text{median}(\cos(X_1), \dots, \cos(X_n))$. We may first use a standard confidence interval at 95% level in **R** just by replacing the mean statistic by the median statistic in our code. We obtain the following confidence interval: $[0.5769, 1.4989]$.

We also may compute a Studentised confidence interval for $T(X_1, \dots, X_n)$. However, things are slightly different here as we ignore the standard deviation of the median. To do so, we may use bootstrap. From $X = (X_1, \dots, X_n)$ samples, we sample with replacement and obtain $X^* = X_1^*, \dots, X_n^*$ samples, from which we can compute the bootstrap replications $T(X^*)$. Repeating this procedure $B$ times, we may estimate the standard deviation of the median as

$$\sqrt{\frac{1}{B}\sum_{b=1}^{B}(T(X^{*b}) - \bar{T})^2} \text{ where } \bar{T} = \frac{1}{B}\sum_{b=1}^{B} T(X^{*b})$$

This has to be done for every samples, and therefore increases the time complexity of the algorithm (a for loop in a for loop). We will then use only $B = 500$ in this case. After, the construction of the Studentised confidence interval is after the same as previously using

$$\mathbb{P}\left(c_1 \leq \frac{T(\hat{P}) - T(P)}{\sigma(\hat{P})} \leq c_2\right) = 1 - \alpha$$

Using the plug-in principle such that $T(\hat{P}) - T(P) \simeq T(P^*) - T(\hat{P})$, we then have

$$\mathbb{P}\left(c_1^* \leq \frac{T(P^*) - T(\hat{P})}{\sigma(P^*)} \leq c_2^*\right) = 1 - \alpha$$

which gives the approximate confidence interval

$$[T(\hat{P}) - c_2^*\sigma(\hat{P}), T(\hat{P}) - c_1^*\sigma(\hat{P})]$$

In **R**, we implement this procedure and obtain the confidence interval $[0.3531, 1.5707]$ which is wider than the standard confidence interval.

## 3 Third question

As my CID is 02091191, we will use $a_1 = 9$ and $a_2 = 1$, let $f$ be the density defined as

$$f(x, y) = k\sqrt{y}\exp\left\{-\frac{x^2}{2} - \frac{y}{2}\left[1 + (a_1 - x)^2 + (a_2 - x)^2\right]\right\} \text{ for } y \geq 0$$

To design a Gibbs sampler for a such multivariate distribution, we need to derive the conditional densities $f(x \mid y)$ and $f(y \mid x)$. First, notice that

$$f(y \mid x) \propto y^{1/2}\exp\left\{\frac{y}{2}(1 + (a_1 - x)^2 + (a_2 - x)^2)\right\}$$

Hence,

$$Y \mid X = y \sim \Gamma\left(\frac{3}{2},\ \frac{1}{2}(1+(a_1-x)^2+(a_2-x)^2)\right)$$

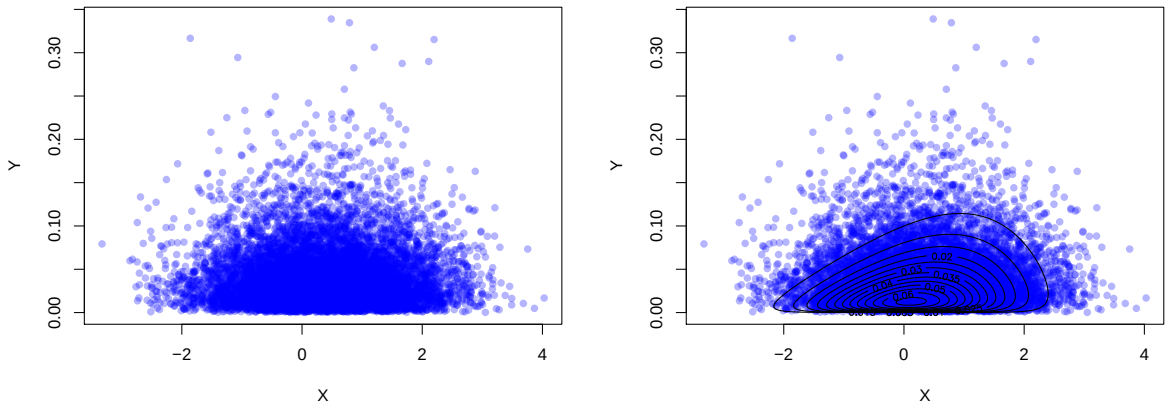
which denotes here a Gamma distribution with shape $\frac{3}{2}$ and rate $\frac{1}{2}(1+(a_1-x)^2+(a_2-x)^2)$. Moreover,

$$\begin{aligned}
f(x \mid y) &\propto \exp\left\{-\frac{x^2}{2} - \frac{y}{2}(1+(a_1-x)^2+(a_2-x)^2)\right\} \\
&\propto \exp\left\{-\frac{1}{2}\left[(1+2y)x^2 - 2(a_1+a_2)xy\right]\right\} \\
&\propto \exp\left\{-\frac{1+2y}{2}\left[x^2 - \frac{2y(a_1+a_2)}{1+2y}x\right]\right\} \\
&\propto \exp\left\{-\frac{1}{2\frac{1}{1+2y}}\left[x - \frac{(a_1+a_2)y}{1+2y}\right]^2\right\} \\
&\propto \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}
\end{aligned}$$

with $\sigma^2 = \frac{1}{1+2y}$ and $\mu = \frac{(a_1+a_2)y}{1+2y}$. Hence,

$$X \mid Y = x \sim \mathcal{N}\left(\frac{(a_1+a_2)y}{1+2y},\ \frac{1}{1+2y}\right)$$

Then, we implement in **R** the Gibbbs sampler which consists in iteratively sampling from both conditional distributions. We obtain the following scatter plot.




(a) Scatter plot of the approximate joint distribution

(b) Scatter plot and contour plot of the theoretical joint distribution $f$

Figure 3.1: Scatter plot of samples from Gibbs sampler for distribution $f$

Figure 3.1 shows the empirical distribution of samples obtained from Gibbs sampler in two dimensions. We can compare the samples to the theoretical distribution given by the contour plot on the right. It therefore seems that the samples have correctly been generated.

Now we can estimate $\mathbb{P}((X,Y) \in [-1,1] \times [0,0.5])$ and $\mathbb{E}[X^2]$ using Monte-Carlo integration with $n = 2000$ samples

$$\mathbb{E}[X^2] \simeq \frac{1}{n}\sum_{i=1}^{n} X_i^2 \text{ and } \mathbb{P}((X,Y) \in [-1,1] \times [0,0.5]) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{(X_i \in [-1,1] \cap Y_i \in [0,0.5])}$$

We obtain approximately $\mathbb{E}[X^2] = 1.1627$ and $\mathbb{P}((X,Y) \in [-1,1] \times [0,0.5]) = 0.6375$.

# 4 Fourth question

## (a) Permutation test

Consider the data set provided on BlackBoard containing observations for two groups: 50 observations in group 1 and 75 in group 2. Assume we are interested in testing the null hypothesis $H_0$ that the two groups have the same variance, against $H_1$ that the two groups have different variances. To perform such a test (see Lecture Notes [1]), we use the fact that the set of all observations is exchangeable under the null hypothesis. We implement in **R** this permutation test, and obtain $p$-value $= 0.0916$ (we can even compute a 95% confidence interval for the $p$-value, which is here $[0.0859, 0.0973]$). Therefore, we do not reject the null hypothesis at a 5% level.

## (b) Parametric bootstrap

**(i) Parametric bootstrap to estimate the distributions of sample variances of each group:** the idea is to assume that observations from group 1 and group 2 are coming from two different Gamma distributions with shape $k = 5$ and respectively scales $\theta_1$ and $\theta_2$ unknown for group 1 and group 2. For parametric bootstrap, we don't sample from the observed data directly but from a known distribution with estimated parameters using observed data, using for example the MLE of these parameters. Here,

$$\hat{\theta}_{MLE} = \frac{\bar{X}}{k}$$

Therefore, we can sample from Gamma distributions with parameters $k$ and $\theta_1, \theta_2$ for each group using the `rgamma()` function in **R**. We obtain observations from the assumed distribution of the same length as the initial group observations (50 for group 1, 75 for group 2), and then we can compute the empirical variance of the observations, and repeat this procedure a lot of times, say $p$ times. We thus obtain $p$ observations of sample variances for each group 1 and 2. We implement this sampling procedure in **R** and represent the empirical cdf obtained.

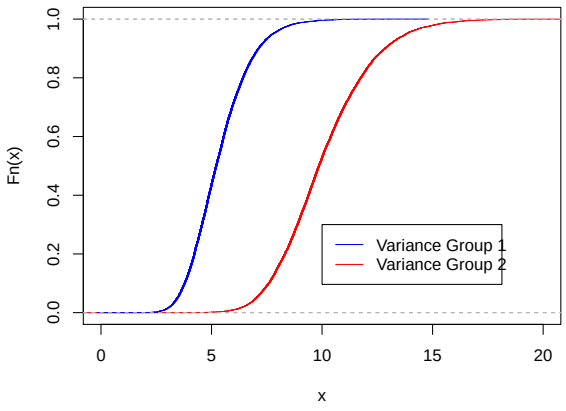



Figure 4.1: Empirical cdf obtained from parametric bootstrap sampling for each group 1 and 2

As shown on Figure 4.1, variance samples from group 1 and group 2 don't seem to have the same distribution.

**(ii) Parametric bootstrap test to estimate the distribution of the absolute difference of the sample variances:** the approach here is to proceed as for question (a) with the difference that here we sample from the parametric bootstrap distributions implemented at (b) (i) and not from permutation samples. We generate samples from the assumed distributions and

compute the test statistic $T$ of these observations where $T$ is the absolute difference between variances from 50s first elements and 75s last elements, then compare the $T$ value obtained to the $T$ obtained with observed data. We repeat this procedure a large number of times, and report the following $p$-value $= 0.0315$, which would lead here to reject the null hypothesis $H_0$ at a 5% level, i.e. that the two groups do not have the same variance.

Hence, this hypothesis testing leads to a different decision compared to permutation testings. Indeed, we should recall that the parametric bootstrap will be more accurate if the distribution and parametric assumptions are correct. Thus, we may look at observations from group 1 and 2 and check if they are following the assumed distributions. Assume these observations are coming from Gamma distributions with shape $k = 5$ and scale $\theta_1, \theta_2$ unknown. The MLE for $\theta_1$ and $\theta_2$ are

$$\hat{\theta}_1 = \frac{\bar{X}}{k} \simeq 1.0322 \text{ and } \hat{\theta}_2 = \frac{\bar{Y}}{k} \simeq 1.4194$$

where $X$ are observations from group 1 and $Y$ observations from group 2. Hence, we can now use these estimates in **R** as the real values of $\theta_1$ and $\theta_2$, and proceed to KS test on $X$ and $Y$, with $H_0$: $X$ is coming from a $\Gamma(k, \theta_1)$ and same for $Y$ with $\theta_2$. We obtain

| $H_0$ | **p-value** |
|---|---|
| $X \sim \Gamma(k, \theta_1)$ | $0.9651 > 0.05$ |
| $Y \sim \Gamma(k, \theta_2)$ | $0.511 > 0.05$ |

Table 3: Results of KS tests for samples from group 1 and 2

Table 3 shows that the $p$-values obtained allow us to assume that $X$ and $Y$ are coming from Gamma distributions with the specified parameters. Hence, the parametric bootstrap used for the second hypothesis test in (b) is more accurate than the one computed in (a) as the parametric assumptions are correct. Therefore, we would more trust the results of the parametric bootstrap test rather than the permutation test, and hence consider that the two groups do not have the same variance.

## References


[1] Dr Sarah Filippi (Autumn 2021) *MATH70093 - Computational Statistics lecture notes*, Imperial College London MSc Statistics resources

[2] Kenneth Lange (2010) *Numerical analysis for statisticians*, New York: Springer

# A R code

```
### Question 1
MH <- function(f, q, rq, n, init){
  ar <- 0
  xall <- rep(NA,n)
  x <- init
  xall[1] <- x
  for(t in seq(2,n)){
    y <- rq(x)
    if(runif(1)<=(f(y)*q(y,x))/(f(x)*q(x,y))){
      x <- y
      ar <- ar+1
    }
    xall[t] <- x
  }
  return(list(xall=xall, ar=ar/n))
}
f <- function(x) ifelse(x>=0 & x<=5,1/(1+abs(x-2)^3),0)

# a
set.seed(56823)
sigma <- 5
q <- function(x,y) dnorm(y,x,sigma)
rq <- function(x) rnorm(1,x,sigma)
n <- 5000
MHnrw <- MH(f,q,rq,n,1)
X <- MHnrw$xall
mean(X[100:n]^3)
mean(X[100:n]<1)
k <- 1/integrate(f, -Inf, Inf)$value
integrate(function(x) k*f(x)*x^3, -Inf, Inf)
integrate(function(x) k*f(x), -Inf, 1)

# b
set.seed(387)
sigma=0.05
q <- function(x,y) dnorm(y,x,sigma)
rq <- function(x) rnorm(1,x,sigma)
n <- 5000
X1 <- MH(f,q,rq,n,1)$xall[100:n]
X2 <- MH(f,q,rq,n,2)$xall[100:n]
X3 <- MH(f,q,rq,n,3)$xall[100:n]
X4 <- MH(f,q,rq,n,4)$xall[100:n]
F <- function(x) integrate(function(x) k*f(x), -Inf, x)$value
x <- seq(-3, 7, by=0.01)
plot(ecdf(X1), col="red", main="")
plot(ecdf(X2), col="blue", add=TRUE)
plot(ecdf(X3), col="green", add=TRUE)
plot(ecdf(X4), col="orange", add=TRUE)
lines(x, sapply(x, F))
legend("bottomright", legend=c("init 1", "init 2", "init 3", "init 4", "real CDF"),
```

```
       col=c("red", "blue", "green", "orange", "black"), lty=1:1)

# c
for(s in c(0.05, 1, 5)){
  q <- function(x,y) dnorm(y,x,s)
  rq <- function(x) rnorm(1,x,s)
  n <- 1e4
  mh <- MH(f,q,rq,n,1)
  ar <- mh$ar
  cat("Sigma", s, "\n")
  cat("Acceptance rate", ar, "\n")
}
FF <- Vectorize(F)
set.seed(578)
sigma <- 0.05
q <- function(x,y) dnorm(y,x,sigma)
rq <- function(x) rnorm(1,x,sigma)
n <- 1e4
MHnrw <- MH(f,q,rq,n,1)
X <- MHnrw$xall[100:n]
ks.test(X, "FF")

set.seed(578)
sigma <- 0.05
q <- function(x,y) dnorm(y,x,sigma)
rq <- function(x) rnorm(1,x,sigma)
n <- 1e4
MHnrw <- MH(f,q,rq,n,1)
X.05 <- MHnrw$xall
sigma <- 1
q <- function(x,y) dnorm(y,x,sigma)
rq <- function(x) rnorm(1,x,sigma)
n <- 1e4
MHnrw <- MH(f,q,rq,n,1)
X.1 <- MHnrw$xall
sigma <- 5
q <- function(x,y) dnorm(y,x,sigma)
rq <- function(x) rnorm(1,x,sigma)
n <- 1e4
MHnrw <- MH(f,q,rq,n,1)
X.5 <- MHnrw$xall
plot(X.1, type="l", col="blue", ylab=expression(X[i]), xlab="i")
points(X.5, type="l", col="green")
points(X.05, type="l", col="red")
legend("topright",
       legend=c(expression(sigma == 0.05),
                           expression(sigma == 1),
                           expression(sigma == 5)),
       col=c("red", "blue", "green"), lty=1:1)
plot(ecdf(X.05), col="red", main="")
plot(ecdf(X.1), col="blue", add=TRUE)
plot(ecdf(X.5), col="green", add=TRUE)
```

```
lines(x, sapply(x, F))
legend("bottomright", legend=c(expression(sigma == 0.05),
                               expression(sigma == 1),
                               expression(sigma == 5),
                               "real CDF"),
       col=c("red", "blue", "green", "black"), lty=1:1)


### Question 2
# a
set.seed(42)
X <- c(-0.71, -1.30, -0.13, -2.03, 1.62, 2.38, 0.48, 0.51, -0.69, -2.32,
       -2.02, 1.23, -0.25, 0.76, 0.65)
T <- function(x) mean(cos(x))
b <- 1000
# Two-sided interval
bootsamples <- replicate(b, {
  bX <- sample(X, replace=TRUE)
  T(bX) - T(X)
})
cstar <- quantile(bootsamples, c(0.025, 0.975))
cat(T(X) - cstar[2], T(X) - cstar[1], "\n")
# Studentised two-sided interval
sigma <- function(x) sd(cos(x))/sqrt(length(x))
bootsamples <- replicate(b,{
  bX <- sample(X, replace=TRUE)
  (T(bX) - T(X))/sigma(bX)
})
cstar <- quantile(bootsamples, c(0.025, 0.975))
cat(T(X) - sigma(X) * cstar[2], T(X) - sigma(X) * cstar[1], "\n")

# b
int.mean <- integrate(function(x) cos(x) * dt(x, 4), -Inf, Inf)$value
MC.mean <- mean(cos(rt(1e4, 4)))
# they are the same, let's take int.mean as reference
results <- c()
for(i in 1:500){
  n <- 15
  X <- rt(n, df=4)
  T <- function(x) mean(cos(x))
  sigma <- function(x) sd(cos(x))/sqrt(length(x))
  # Two-sided interval
  bootsamples <- replicate(b, {
    bX <- sample(X, replace=TRUE)
    T(bX) - T(X)
  })
  cstar <- quantile(bootsamples, c(0.025, 0.975))
  in.CI1 <- int.mean > T(X) - cstar[2] & int.mean < T(X) - cstar[1]
  # Studentised two-sided interval
  bootsamples <- replicate(b,{
    bX <- sample(X, replace=TRUE)
    (T(bX) - T(X))/sigma(bX)
```

```
  })
  cstar <- quantile(bootsamples, c(0.025, 0.975))
  in.CI2 <- int.mean > T(X) - sigma(X) * cstar[2] & int.mean < T(X) - sigma(X) * cstar[1]
  # Store the results
  results <- rbind(results, c(in.CI1, in.CI2))
}
mean(results[,1])
mean(results[,2])

# c
X <- c(-0.71, -1.30, -0.13, -2.03, 1.62, 2.38, 0.48, 0.51, -0.69, -2.32,
       -2.02, 1.23, -0.25, 0.76, 0.65)
T <- function(x) median(cos(x))
b <- 1000
# Two-sided interval
bootsamples <- replicate(b, {
  bX <- sample(X, replace=TRUE)
  T(bX) - T(X)
})
cstar <- quantile(bootsamples, c(0.025, 0.975))
cat(T(X) - cstar[2], T(X) - cstar[1], "\n")
# For studentized, we need to estimate sd(T(X))
b <- 500
genbootsample <- function(data){
  sample(data, length(data), replace=TRUE)
}
bootsd <- function(data, T, nrep){
  sd(replicate(nrep, T(genbootsample(data))))
}
bootsamples <- replicate(b,{
  bX <- sample(X, replace=TRUE)
  (T(bX) - T(X))/bootsd(bX, T, b)
})
cstar <- quantile(bootsamples, c(0.025, 0.975))
cat(T(X) - bootsd(X, T, b) * cstar[2], T(X) - bootsd(X, T, b) * cstar[1], "\n")


# Question 3
set.seed(42)
a1 <- 9
a2 <- 1
f <- function(x,y) sqrt(y) * exp(-x^2/2 - y/2 * (1 + (a1-x)^2 + (a2-x)^2))
it <- 2000
x <- rep(NA,it)
y <- rep(NA,it)
x[1] <- 0.1
y[1] <- 0.1
for (i in 2:it){
  x[i] <- rnorm(1, mean=(a1+a2)*y[i-1]/(1+2*y[i-1]), sd=sqrt(1/(1+2*y[i-1])))
  y[i] <- rgamma(1, shape=3/2, rate=0.5*(1 + (a1-x[i-1])^2 + (a2-x[i-1])^2))
}
mean(x^2)
```

```
mean(x>=-1 & x<=1 & y>=0 & y<=0.5)

X <- seq(-3, 3, length.out=1e3)
Y <- seq(0, 0.4, length.out=1e3)
Z <- matrix(data=NA, nrow=length(X), ncol=length(Y))
for(i in 1:length(X)){
  for(j in 1:length(Y)){
    Z[i,j] = f(X[i], Y[j])
  }
}
plot(x, y, pch=16, col=alpha("blue", 0.3), xlab="X", ylab="Y")
contour(X, Y, Z, add=TRUE)


### Question 4

# a
data <- read.csv("qu4.csv")
G1 <- data[data$group == 1,]$observations
G2 <- data[data$group == 2,]$observations
n <- length(G1)
m <- length(G2)
pooled <- c(G1, G2)
T <- function(data) abs(var(data[1:n]) - var(data[(n+1):(n+m)]))
t <- T(pooled)
Trep <- replicate(1e4, T(sample(pooled, length(pooled)))>=t)
pval <- mean(Trep)
pval + sd(Trep)/sqrt(length(Trep)) * qnorm(c(0.025, 0.975))

# b
k <- 5
genbootsample <- function(data){
  theta.hat <- mean(data)/k # the MLE estimator of the parameter
  rgamma(length(data), shape=5, scale=theta.hat)
}
dist.1 <- replicate(1e4, var(genbootsample(G1)))
dist.2 <- replicate(1e4, var(genbootsample(G2)))

plot(ecdf(dist.2), col="red", xlim=c(0, 20), main="")
plot(ecdf(dist.1), col="blue", add=TRUE)
legend("bottomright", legend=c("Variance Group 1", "Variance Group 2"),
       col=c("blue", "red"), lty=1:1)

# c
T <- function(data) abs(var(data[1:n]) - var(data[(n+1):(n+m)]))
pooled <- c(G1, G2)
t <- T(pooled)
boot <- function(data, T, nrep) replicate(nrep, genbootsample(data))
pval <- mean(boot(pooled, T, 1e4)>=t)

ks.test(G1, "pgamma", k, k/mean(G1))
ks.test(G2, "pgamma", k, k/mean(G2))
```