Stat3001 Assignment 2

Dominic Scocchera

April 2023

Q1

a)

Algorithm 1 Non-Parametric Bootstrap

Require: sample $x_1, x_2, ..., x_n$ from a distribution with density f and mean μ

Repeat this K times where in the k_{th} iteration:

(1) Resample $x^*_{1k}, x^*_{2k}, ..., x^*_{nk}$ from the original data $x_1, x_2, ..., x_n$

(2) Compute $\bar{x}^*_k - \bar{x}$, where $\bar{x}^*_k = \frac{1}{n}\sum_{i=1}^{n} x^*_{ik}$ is the sample mean of the k_{th} bootstrap sample

Then $\mathbb{P}(|\bar{X} - \mu| > 2)$ can be approximated by $\frac{1}{K}\sum_{i=1}^{K} |\bar{x}^*_k - \bar{x}|\mathbf{1}_{|\bar{x}^*_k - \bar{x}|>2}$

---

b)

Algorithm 2 Parametric Bootstrap

Require: sample $x_1, x_2, ..., x_n$

Estimate $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $\bar{\sigma}^2 = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$

Repeat this K times where in the k_{th} iteration:

(1) Draw a sample $x^*_{1k}, x^*_{2k}, ..., x^*_{nk}$ from the distribution $N(\bar{x}, \bar{\sigma}^2)$

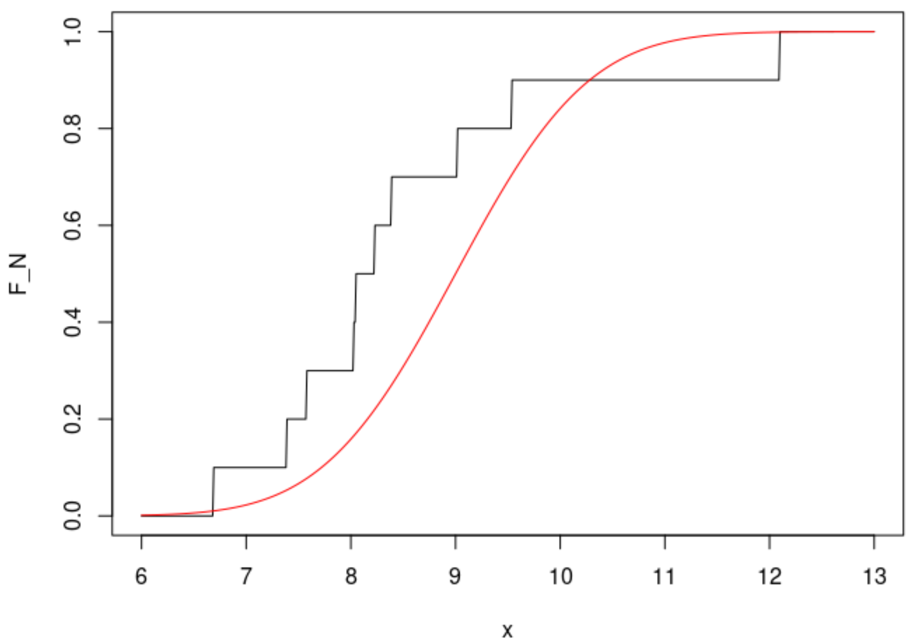
(2) Compute $\bar{x}^*_k - \bar{x}$, where $\bar{x}^*_k = \frac{1}{n}\sum_{i=1}^{n} x^*_{ik}$ is the sample mean of the k_{th} bootstrap sample

Then $\mathbb{P}(|\bar{X} - \mu| > 2)$ can be approximated by $\frac{1}{K}\sum_{i=1}^{K} |\bar{x}^*_k - \bar{x}|\mathbf{1}_{|\bar{x}^*_k - \bar{x}|>2}$

---

c)

We have the following CDF and ECDF:



We perform the Kolmogorov-Smirnov test:

$$\mathbb{H}_0 : \text{The data follows a N(9,1) distribution}$$
$$\mathbb{H}_1 : \text{The data does not follow a N(9,1) distribution}$$

The test stastic is:

$$D_n = \max_{1 \leq i \leq N} \left[F(x_i) - \frac{i-1}{N}, \frac{i}{N} - F(x_i) \right] = 0.4290691$$

We let our significance level $\alpha = 0.05$. The p-value is 0.03494. This is below the significance level so we reject the null hypothesis. Below is the code used.

```
#data
x = c(8.23, 7.58, 7.39, 9.02, 6.69, 8.05, 8.38, 8.03, 9.54, 12.10)
n = length(x)
#calculate Fn and F
xgrid = seq(floor(min(x)), ceiling(max(x)), by=0.01)
Fn = c()
for(i in 1:length(xgrid)) Fn[i] = mean(x <= xgrid[i])
```

```
plot(xgrid, Fn, type="n", xlab="x", ylab="F_N")
lines(xgrid, Fn)

#add the true cdf
cdf <- pnorm(sort(xgrid), mean=9, sd=1)
lines(xgrid, cdf, col="red")

#simulate Dn
dn = NULL; K = 100000;
for(k in 1:K) {
  u = runif(n, min=0, max=1);   #random uniform sample
  i = 1:n;                      #index
  u.sorted = sort(u);           #sort u from smallest to largest
  dn[k] = max( max(abs(u.sorted-i/n)), max(abs(u.sorted-(i-1)/n)) )
}
#test statistic
dn.max = max(abs(Fn-cdf)); dn.max
#p-value
p.value= mean(dn >= dn.max); p.value
```

Q2

Recall in Tutorial Sheet 4 Question 4, we implemented a Gibbs sampler to draw samples $\mathbf{X} = (X, Y)^T$ from the bivariate normal distribution $N(\mathbf{0}, \Sigma)$ where $\mathbf{0} = (0, 0)^T$ and Σ is given below. We describe a random walk Metropolis-Hastings algorithm to draw samples from this bivariate distribution, using the proposal density $N(\mathbf{0}, \Phi)$, where

$$\Sigma = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}, \Phi = \begin{pmatrix} a^2 & 0 \\ 0 & b^2 \end{pmatrix}$$

Here a and b are constants.

Algorithm 3 Metropolis-Hastings random walk

Require: starting state $\mathbf{X}_0 = \mathbf{0}$

Repeat N times to get N sets of sample from $N(\mathbf{0}, \Sigma)$ where in the i_{th} iteration:

1) Generate $\mathbf{Z}$ from $N(\mathbf{0}, \Phi)$, where $\mathbf{Z} = (z_1, z_2)^T$

2) Calculate the proposal $\mathbf{Y} = \mathbf{X}_n + \mathbf{Z}$, here $\mathbf{X}_n$ is the current state

3) Evaluate the acceptance probability $\alpha(\mathbf{X}_n, \mathbf{Y}) = \min\left\{\frac{f(\mathbf{Y})}{f(\mathbf{X}_n)}, 1\right\}$, here f is the joint pdf of the $N(\mathbf{0}, \Sigma)$

4) Generate $U \sim \mathbf{U}(0, 1)$ and calculate $\mathbf{X}_{n+1} = \begin{cases} \mathbf{Y} & \text{if } U \leq \alpha(\mathbf{X}_n, \mathbf{Y}) \\ \mathbf{X}_n & \text{else} \end{cases}$

return $\mathbf{X}_n$

Q3

We have the density:

$$f(x,y,z) \propto \binom{z}{y} x^{\alpha+y-1}(1-x)^{\beta+z-y-1}\frac{\gamma^z}{z!}$$

For $0 < x < 1$, $y = 0, 1, 2, ..., z$, and $z = 0, 1, 2, ...$, and where $\alpha > 0$, $\beta > 0$ and $\gamma > 0$ are constants.

We want to derive a Gibbs sampler.

The conditional for x is:

$$\begin{aligned}
f_1(x|y,z) &= \frac{f(x,y,z)}{f(y,z)} \\
&\propto f(x,y,z) \\
&\propto x^{\alpha+y-1}(1-x)^{\beta+z-y-1} \\
&\propto \frac{\Gamma(\alpha+\beta+z)}{\Gamma(\alpha+y)\Gamma(\beta+z-y)} x^{\alpha+y-1}(1-x)^{\beta+z-y-1}
\end{aligned}$$

The conditional for y is:

$$\begin{aligned}
f_1(y|x,z) &= \frac{f(x,y,z)}{f(x,z)} \\
&\propto f(x,y,z) \\
&\propto \binom{z}{y} x^{y}(1-x)^{z-y}
\end{aligned}$$

The conditional for z is:

$$\begin{aligned}
f_1(z|x,y) &= \frac{f(x,y,z)}{f(x,y)} \\
&\propto f(x,y,z) \\
&\propto \binom{z}{y} (1-x)^{z}\frac{\gamma^z}{z!} \\
&= \frac{z!}{y!(z-y)!}\frac{((1-x)\gamma)^z}{z!} \\
&\propto \frac{((1-x)\gamma)^z e^{-(1-x)\gamma}}{z!}
\end{aligned}$$

So our Gibbs sampler is:

Algorithm 4 Gibbs Sampler

Require: $\alpha, \beta, \gamma > 0$, $0 < x_0 < 1$, $y_0 = 0, ..., z_0$, $z_0 = 0, 1, 2, ...$

$x_{t+1} \sim \text{Beta}(\alpha + y_t, \beta + z_t - y_t)$

$y_{t+1} \sim \text{Bin}(z_t, x_{t+1})$

$z_{t+1} \sim \text{Poi}((1 - x_{t+1})\gamma)$