STA380 Term Test 1 Solutions

The grade will be calculated out of **35 points**.

Marking guidelines for the code portion:

- Allow leniency for isolated omissions of parentheses or braces, such as a missing {, }, `(`, or `)`.
- Minor deviations from perfect `R` syntax should be tolerated when the intended expression is clear. For instance, expressions such as

$$u < e^{-y^2/9+y}$$

or

$$x = \frac{1}{1-u[i]} - 1$$

may be accepted in place of the exact `R` syntax `exp(-(y^2/9) + y)` or `x <- 1/(1-u[i]) - 1`, even if the marking scheme specifies otherwise. HOWEVER, remark requests will be treated harshly if students have repeatedly made *this* mistake for their entire term test.

- Similarly, minor misuse of indexing brackets (e.g., using `[]` instead of { } or `()`) should not be heavily penalised. Similarly, remark requests will be treated harshly if students have repeatedly made *this* mistake for their entire term test.
- If they grow `R` objects, don't deduct marks for the first test but explicitly call them out. Growing `R` objects is when someone does the following:

```
x <- c() #initialize vector of size 0
for(i in 1:n){
    x <- c(x, i) # growing the vector
}
```

Substantial syntax errors or code that would not execute correctly in `R` should, however, be penalised appropriately, unless specified otherwise.

Question 1

Let X be a continuous random variable with the following probability density function:

$$f_X(x) = \frac{1}{(1+x)^2}, \quad x \geq 0.$$

(a) **[2 points]** Find $F_X(x)$, the cumulative distribution function of X. **Show all steps. Circle the final answer.**

(b) **[2 points]** Derive $F_X^{-1}(x)$. **Show all steps. Circle the final answer.**

(c) **[2 points]** Write an algorithm to generate a sample of size $n = 10^4$ from the distribution of X using the *inverse transform method*. **Be precise.**

(d) **[2 points]** Write out the `R` code for the algorithm.

Solution. This was the last question of the second quiz. Hence, I expect students to do very well in this question. **Also, since this was the last question of the second quiz, I did not give marks for a carry on mistake unless the mistake was very minor.**

(a) We know that for $x < 0$ that $F_X(x) = 0$. Now, for $x \geq 0$:

$$\begin{aligned}
F_X(x) &= \int_0^x \frac{1}{(1+t)^2} dt \\
&= \int_1^{1+x} u^{-2} du \quad (u = 1+t \text{ substitution}) \\
&= -u^{-1}\Big|_1^{1+x} \\
&= 1 - \frac{1}{1+x}.
\end{aligned}$$

(1 point for correctly finding $F_X(x)$ for $x \geq 0$). Hence, the CDF is:

$$F_X(x) = \begin{cases} 0 & x < 0, \\ 1 - \frac{1}{1+x}, & x \geq 0 \end{cases}$$

(1 point for correctly writing $F_X(x)$ including the entire support).

(b) For $x \geq 0$, Set $U = 1 - \frac{1}{1+x}$ and then solve for x:

$$U = 1 - \frac{1}{1+x} \quad \Rightarrow \quad -(U-1) = \frac{1}{1+x} \quad \Rightarrow \quad x = \frac{1}{1-U} - 1.$$

(1 point for correctly finding the inverse function).
You should get the following using inverse transform:

$$F_X^{-1}(x) = \frac{1}{1-x} - 1, \quad 0 \leq x < 1$$

Note that this function doesn't exist when $x = 1$.
(0.5 points for neatly writing out the inverse function, and another 0.5 points for writing the support correctly).

Some people might simplify:

$$\frac{1}{1-x} - 1 = \frac{x}{1-x}$$

(c) For each random variate required, **(0.5 points).**

- Generate a random variable $u \sim Uniform(0,1)$. **(0.5 points).**
- Deliver $x = \frac{1}{1-u} - 1$. **(1 point).**

(d)
```
n <- 10^4
u <- runif(n)
x <- 1/(1-u) -1
```

- **(0.5 points for initalizing n; technically, you could just plug this into `runif()`).**
- **(0.5 points from generating from a uniform distribution).**
- **(0.5 points for delivering x correctly).**
- **(0.5 points for perfect `r` syntax).**

Question 2

Consider the Weibull distribution:

$$f(x) = \frac{\alpha}{\beta}\left(\frac{x}{\beta}\right)^{\alpha-1} e^{-(x/\beta)^{\alpha}}, x \geq 0.$$

Here, $\alpha > 0$ represents the shape parameter and $\beta > 0$ represents the scale parameter (for both the Weibull and the Gamma distribution).

Use the acceptance-rejection method to generate a sample from the $Weibull(\alpha = 2, \beta = 3)$ distribution. Let the trial distribution be $Gamma(\alpha = 2, \beta = 1)$.

(a) [**3 points**] What is $c = \max\left\{\frac{f(x)}{g(x)}\right\}$, where $f(x)$ is the target distribution and $g(x)$ is the trial distribution? **You do <u>not</u> need to show that the value you found is a maximum.**

(b) [**2 points**] Write out the *acceptance-rejection* algorithm to generate a sample from X. **Be precise.**

(c) [**3 points**] Write out the `R` code for the algorithm. Assume $n = 10^4$. You should use `rgamma(n, shape, scale)`, where `n` represents the number of observations, and shape, scale represent the shape and scale parameters of the Gamma distribution.

(d) [**2 points**] If the first uniform in the acceptance-rejection algorithm was $U = 0.93$ and the first candidate draw was $Y = 1.21$. Would you accept or reject this candidate? **You must specify *accept* or *reject*. Show all steps.**

Solution.

(a) Note that,

$$f(x) = \frac{2}{3}\left(\frac{x}{3}\right)^{2-1} e^{-(x/3)^2} = \frac{2}{9}xe^{-(x/3)^2}, x \geq 0.$$

$$g(x) = \frac{1}{\Gamma(2)(1)^2}x^{2-1}e^{-x/1} = xe^{-x}, \quad x > 0$$

(1 point for correctly writing out the densities $f(x)$ and $g(x)$).

Then,

$$\begin{aligned}
\frac{f(x)}{g(x)} &= \frac{\frac{2}{9}xe^{-(x/3)^2}}{xe^{-x}} \\
&= \frac{2}{9}e^{-x^2/9+x} \\
\frac{d\left(\frac{f(x)}{g(x)}\right)}{dx} &= \frac{2}{9}e^{-x^2/9+x}\left(\frac{-2x}{9}+1\right) \\
0 &\overset{set}{=} \frac{2}{9}e^{-x^2/9+x}\left(\frac{-2x}{9}+1\right) \\
0 &= \left(\frac{-2x}{9}+1\right) \\
x &= 9/2
\end{aligned}$$

Thus we have that:

$$c = \frac{f(9/2)}{g(9/2)} = \frac{2}{9}e^{-(9/2)^2/9+(9/2)} \approx 2.11 \text{ or } 2.1084$$

- **(0.5 points for attempting to find the derivative.**
- **(0.5 points for correctly finding the maximum).**
- **(0.5 points for plugging the maximum value into c).**
- **(0.5 points for correctly finding the value of c.)**

(b) For each random variate required, **(0.5 points).**

(i) Generate $Y \sim Gamma(\alpha = 2, \beta = 1)$. **(0.5 points).**

(ii) Generate $U \sim Uniform(0, 1)$. **(0.5 points).**

(iii) If

$$u < \frac{2}{9}\frac{e^{-y^2/9+y}}{2.11} \approx 0.11e^{-y^2/9+y}$$

accept y and set $x = y$. Otherwise, reject y and generate a random variate again. **(0.5 points).**

(c)

```
n <- 10^4
accepted <- numeric(n)
u_accepted <- numeric(n)
i <- 0
iteration <- 0
while(i < n){
  y <- rgamma(n = 1, shape = 2, scale = 1) # candidate from g
  u <- runif(1) # u ~ uniform(0, 1)
  ftgt <- 0.11 * exp(-(y^2/9) + y) # f(x)/cg(x)

  if(u < ftgt){
    i <- i+1
    accepted[i] <- y
    u_accepted[i] <- u
  }
  iteration <- iteration + 1
}
```

There is a more efficient acceptance-rejection algorithm that students are allowed to use. I only discussed this in lecture, out of curiosity to see if students will use this method. This method also doesn't guarantee that there will be n amount of iterations:

```
n <- 10^4
c <- 2.11
m <- c*n
y <- rgamma(m, shape = 2, scale = 1)
u <- runif(m)
ftgt <- 0.11 * exp(-(y^2/9) + y)

accept <- (u < ftgt)
accepted <- y[accept]
u_accepted <- u[accept]
```

- **(1 point for the correct setup involving n, `accepted`, `u_accepted`, iteration, and preferably using a `while` loop correctly until getting the number of samples).**
- **(0.5 points from generating from a gamma distribution and the uniform distribution).**
- **(0.5 points for computing `ftgt`, or however they want to call it).**
- **(0.5 points for accepting or rejecting based on `u < ftgt`).**
- **(0.5 points for perfect `r` syntax).**

If you do not care to have a certain number of valid replicates, you could code this up using a for-loop instead. However, such method is not preferred for the rare case where you reject all samples. Nonetheless, you can award marks for this approach (as it's hard to code by hand).

(d) $0.11e^{-(1.21)^2/9+(1.21)} = 0.31$. Clearly, $u = 0.93 > 0.31$ and so we **reject**.

- **(0.5 points for plugging in $y = 1.21$ for $\frac{f(x)}{cg(x)}$). If plugging in but wrong value, then 0.25 points**
- **(0.5 points for comparing it to u).**
- **(1 point for correctly identifying that we reject).**
- **(1 point for wrong value, but correctly compare to u and correct conclusion based on wrong value).**

Question 3

[2 points] Let $Z_1, Z_2, \ldots, Z_n$ be independent and identically distributed random variables such that $Z_i \sim N(0, 1)$ for all $i = 1, 2, \ldots, n$. Then, the sum of their squares follows a chi-squared distribution with n degrees of freedom:

$$\sum_{i=1}^{n} Z_i^2 \sim \chi_n^2.$$

Use `rnorm()` and the property that a chi-squared random variable can be expressed as the sum of squared standard normal variables to generate $n = 10^4$ samples from the $\chi^2_{(30)}$ distribution.

Solution.

```
n <- 10^4
values <- numeric(n)
for(i in 1:n){
  values[i] <- sum(rnorm(30, 0, 1)^2)
}
```

- **(0.25 points for writing the sample size n).**
- **(0.25 points for initializing the vector and writing a for loop).**
- **(0.5 points for generating from `rnorm` correctly).**
- **(0.5 points for correctly squaring and taking the summation).**
- **(0.5 points for perfect `r` syntax).**

This code isn't the most efficient way to do it, but it's naturally the first solution one would come up with by hand. A more efficient way, which I'd be impressed (perhaps suspicious) if students use, is:

```
n <- 10^4
values <- rowSums(matrix(rnorm(n * 30, 0, 1)^2, nrow = n))

# Bonus; not graded, but if you were working outside of a test environment
# to double check the answer:
hist(values, prob = TRUE, ylim = c(0, 0.06))
curve(dchisq(x, df = 30), 0, 70, add = TRUE, col = "red", lwd = 4)
```

Question 4

Let $f_i(x)$ represent a valid density function for $i = 1, 2, \ldots, k$. Given k component distributions $f_1(x), ..., f_k(x)$, each with associated mixing probabilities (also known mixing weights) $\pi_1, \ldots, \pi_k$, a finite mixture model $f(x)$ is defined as:

$$f(x) := \sum_{i=1}^{k} \pi_i f_i(x),$$

where $0 \leq \pi_i \leq 1$ and $\sum_{i=1}^{k} \pi_i = 1$.

(a) **[2 points]** Show that $f(x)$ satisfies the properties of a density function. That is, show that $f(x)$ is non-negative and $\int_{-\infty}^{\infty} f(x)dx = 1$.

(b) **[4 points]** Suppose X_i has density f_i. Let $\mu_i := \mathbb{E}[X_i] = \int_{-\infty}^{\infty} x f_i(x)dx$ and let $\sigma_i^2 := \mathbb{V}[X_i]$ represent the standard deviation for the i-th component distribution. Show that:

$$\sigma^2 := \mathbb{V}[X] = \sum_{i=1}^{k} \pi_i \sigma_i^2 + \sum_{i=1}^{k} \pi_i (\mu_i - \mu)^2$$

Solution. This was done in lecture.

(a) By assumption, since we have that:

$$0 \leq \pi_i, \;\; 0 \leq f_i(x), \;\; \forall i \in \{1, 2, \ldots, n\},$$

$$\Rightarrow \;\; 0 \leq \pi_i f_i(x) \;\; \forall i \in \{1, 2, \ldots, n\},$$

$$\Rightarrow 0 \leq \pi_1 f_1(x) + \pi_2 f_2(x) + \cdots + \pi_n f_n(x) = \sum_{i=1}^{n} \pi_i f_i(x).$$

(1 point for correctly proving that $f(x)$ is non-negative).
(0.5 points for correctly specify the property of a valid density).
Now, to show that this integrates to 1:

$$\int_{-\infty}^{\infty} f(x)dx = \int_{-\infty}^{\infty} \sum_{i=1}^{n} \pi_i f_i(x)dx = \sum_{i=1}^{n} \int_{-\infty}^{\infty} \pi_i f_i(x)dx$$

Where the previous line is us applying the linearity property of the integral. **(0.5 points for writing up to this point; they do not need to specify the linearity property of the integral).**

$$\begin{aligned} \sum_{i=1}^{n} \int_{-\infty}^{\infty} \pi_i f_i(x)dx &= \sum_{i=1}^{n} \pi_i \underbrace{\int_{-\infty}^{\infty} f_i(x)dx}_{=1} \\ &= \sum_{i=1}^{n} \pi_i \\ &= 1 \quad \text{(By assumption.)} \end{aligned}$$

(0.5 points for writing up to this point; they should probably specify that we assumed that $\underbrace{\int_{-\infty}^{\infty} f_i(x)dx}_{=1}$ though).

(b)

$$
\begin{aligned}
\mathbb{V}[X] &= \mathbb{E}[(X-\mu)^2] \\
&= \int_{-\infty}^{\infty} (x-\mu)^2 \sum_{i=1}^{k} \pi_i f_i(x) dx \quad \textbf{(0.5 points).} \\
&= \sum_{i=1}^{k} \pi_i \int_{-\infty}^{\infty} (x^2 - 2x\mu + \mu^2) f_i(x) dx \quad \textbf{(0.5 points).} \\
&= \sum_{i=1}^{k} \pi_i \left[\int_{-\infty}^{\infty} x^2 f_i(x) dx - 2\mu \int_{-\infty}^{\infty} x f_i(x) dx + \mu^2 \int_{-\infty}^{\infty} f_i(x) dx \right] \quad \textbf{(0.5 points).} \\
&= \sum_{i=1}^{k} \pi_i \left[\mathbb{E}[X_i^2] - 2\mu\mu_i + \mu^2 \right] \quad \textbf{(0.5 points).} \\
&= \sum_{i=1}^{k} \pi_i \left[\mathbb{V}[X_i] + (\mathbb{E}[X_i])^2 - 2\mu\mu_i + \mu^2 \right] \quad \textbf{(0.5 points).} \\
&= \sum_{i=1}^{k} \pi_i \left[\sigma_i^2 + \mu_i^2 - 2\mu\mu_i + \mu^2 \right] \quad \textbf{(0.5 points).} \\
&= \sum_{i=1}^{k} \pi_i \left[\sigma_i^2 + (\mu_i - \mu)^2 \right] \quad \textbf{(0.5 points).} \\
&= \sum_{i=1}^{k} \pi_i \sigma_i^2 + \sum_{i=1}^{k} \pi_i (\mu_i - \mu)^2 \quad \textbf{(0.5 points).}
\end{aligned}
$$

Question 5

Convolutions and mixtures may appear similar in form, but they represent fundamentally different types of distributions. Let X_1, X_2, X_3 be independent random variables with

$$X_1 \sim N(\mu = -1, \sigma = 1), \quad X_2 \sim N(\mu = 3, \sigma = 2), \quad X_3 \sim N(\mu = 6, \sigma = 3).$$

(a) [**2 points**] Write `R` code to generate $n = 10^4$ samples from the following *convolution*:

$$S := 0.3X_1 + 0.3X_2 + 0.4X_3.$$

(b) [**2 points**] Write `R` code to generate $n = 10^4$ samples from the following *mixture distribution*:

$$F_X(x) := 0.3F_{X_1}(x) + 0.3F_{X_2}(x) + 0.4F_{X_3}(x),$$

where each F_{X_i} denotes the cumulative distribution function (CDF) of X_i.

Hint. You may find the following `R` function useful: `rnorm(n, mean = 0, sd = 1)` where `n` represents the number of observations, and the `mean` and `sd` represents the mean and standard deviation, respectively.

Solution.

```
n <- 10^4
x1 <- rnorm(n, mean = -1, sd = 1)
x2 <- rnorm(n, mean = 3, sd = 2)
x3 <- rnorm(n, mean = 6, sd = 3)

# Convolution version
s <- (0.3 * x1) + (0.3 * x2) + (0.4 * x3)

# Mixture version
k <- sample(1:3, size = n, replace = TRUE, prob = c(0.3, 0.3, 0.4))
x <- ifelse(k == 1, x1, ifelse(k == 2, x2, x3))
```

An alternative solution is the following (for the mixture version):

```
n <- 10^4
u < runif(n)
samp <- numeric(n)
for(i in 1:n){
  if(u[i] <= 0.3){
    samp[i] <- rnorm(1, mean = -1, sd = 1)
  } else if(u[i] <= 0.6){
    samp[i] <- rnorm(1, mean = 3, sd = 2)
  } else {
    samp[i] <- rnorm(1, mean = 6, sd = 3)
  }
}
```

I used the same x_1, x_2, x_3 for both the convolution version and the mixture version but this is not necessary.

(a)
- **(0.5 points for initializing n).**
- **(0.5 points for generating from x_1, x_2, x_3).**
- **(0.5 points for successfully writing the convolution).**
- **(0.5 points for perfect r syntax).**

(b)
- **(0.25 points for initializing n).**
- **(0.25 points for generating from x_1, x_2, x_3).**
- **(1 point for randomly sampling from one of the densities correctly).**
- **(0.5 points for perfect r syntax).**

If students do something similar to this instead:

```
x <- numeric(n)
for(i in 1:n){
    if(k[i] == 1){
        x[i] <- x1[i]
    } else if (k[i] == 2){
        x[i] <- x2[i]
    } else {
        x[i] <- x3[i]
    }
}
```

But they forgot to use `[i]` on either `x`, `k`, `x1`, `x2`, or `x3`, then feel free to award full marks (for this portion), even though it would not work in `R`.

Question 6

The probability density function (pdf) of the half-normal distribution is given by

$$g_X(x) = \frac{\sqrt{2}}{\sigma\sqrt{\pi}} \exp\left(-\frac{x^2}{2\sigma^2}\right), \quad x \geq 0.$$

Suppose $\sigma^2 = 4$.

(a) [**3 points**] Provide a detailed algorithm to obtain a Monte Carlo estimate of $\mathbb{P}(1 \leq X \leq 3)$ using a Monte Carlo sample size of $m = 10^4$. **Show all steps. Be precise.**

(b) [**2 points**] Write out the `R` code for the algorithm.

Solution.

(a) **(0.5 point for identifying the following):** Note that,

$$\mathbb{P}(1 \leq X \leq 3) = \int_1^3 \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{x^2}{8}\right\} dx.$$

(0.5 point for identifying the following): If $X \sim Uniform(a, b)$ then

$$f_X(x) = \frac{1}{3-1} = \frac{1}{2}, \quad 1 < x < 3, \quad \mathbb{E}[g(X)] = \int_1^3 \frac{1}{2}\left(\frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{x^2}{8}\right\}\right) dx$$

(0.5 point for identifying the following): Notice that:

$$\int_1^3 \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{x^2}{8}\right\} = 2\int_1^3 \frac{1}{2}\left(\frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{x^2}{8}\right\}\right) dx = 2\mathbb{E}[g(X)]$$

We'll use the SLLN to estimate $\mathbb{E}[g(X)]$. **(0.5 point for mentioning the SLLN. They can also mention WLLN or just law of large numbers.)** If they can correctly write the algorithm below without showing the work above, then feel free to award full marks. The above is mostly to give part marks for student with partial attempts.

(a) Generate $u_1, \ldots, u_m \sim Uniform(1, 3)$. **(0.25 points).**

(b) Compute $\widehat{\mathbb{E}[g(X)]} = \frac{1}{m}\sum_{i=1}^{m} \frac{1}{\sqrt{2\pi}} \exp\left\{-\frac{u_i^2}{8}\right\}$. **(0.5 points).**

(c) Deliver $\hat{\theta} = 2\widehat{\mathbb{E}[g(X)]}$. **(0.25 points).**

(b)
```
# Using Monte Carlo integration
n = 10^4
u <- runif(n, 1, 3)
2 *mean(1/sqrt(2 * pi) * exp(-u^2 / 8))
```

- **(0.5 points for initalizing n; technically, you could just plug this into `runif()`).**
- **(0.5 points from generating from a uniform distribution with the correct arguments).**
- **(0.5 points for calculating $\widehat{\mathbb{E}[g(X)]}$ using the mean; they can also brute force this).**
- **(0.5 points for perfect `r` syntax).**

Not needed for the test, but the answer I got from the previous question was 0.483398 (the answer does change per seed). If you wanted to compare the answer, this is how I would brute force this in `R` (and you can see Monte Carlo works well):

```
# Purely coding it
f <- function(x) {
  (1 / sqrt(2 * pi)) * exp(-x^2 / 8)
}
integrate(f, lower = 1, upper = 3)
# 0.4834607 with absolute error < 5.4e-15
```