The University of British Columbia
I.K. Barber Faculty of Science

COSC 405/DATA 405/DATA 505/COSC 505 Modelling and Simulation

Practice Term Test 1 Solutions

1. (4 marks) Write out the R code required to

(a) calculate the standard deviation of the following sample:

```
##  [1]  27  48  72 101  98  37  22  55  41  79  58  44  61
```

```r
x <- c(27, 48, 72, 101, 98, 37, 22, 55, 41, 79, 58, 44, 61)
sd(x)
```

(b) produce a boxplot of the sample above.

```r
boxplot(x)
```

(c) produce a normal QQ-plot of the sample above with reference line.

```r
qqnorm(x)
qqline(x)
```

(d) test whether the mean of the sample differs from 50. and constructs a 99% confidence interval for the mean of the population from which the sample was taken.

```r
t.test(x, mu = 50, conf.level=.99)
```

2. (10 marks) Consider the following simulated data and analysis. The simulation is to emulate a paper airplane throwing experiment involving a number of throws for each of a number of different types of paper.

```
papergroup <- factor(rep(1:5, 7))
distances <- rnorm(35, mean = 9, sd = 3)
```

(a) How many different types of paper are being simulated?

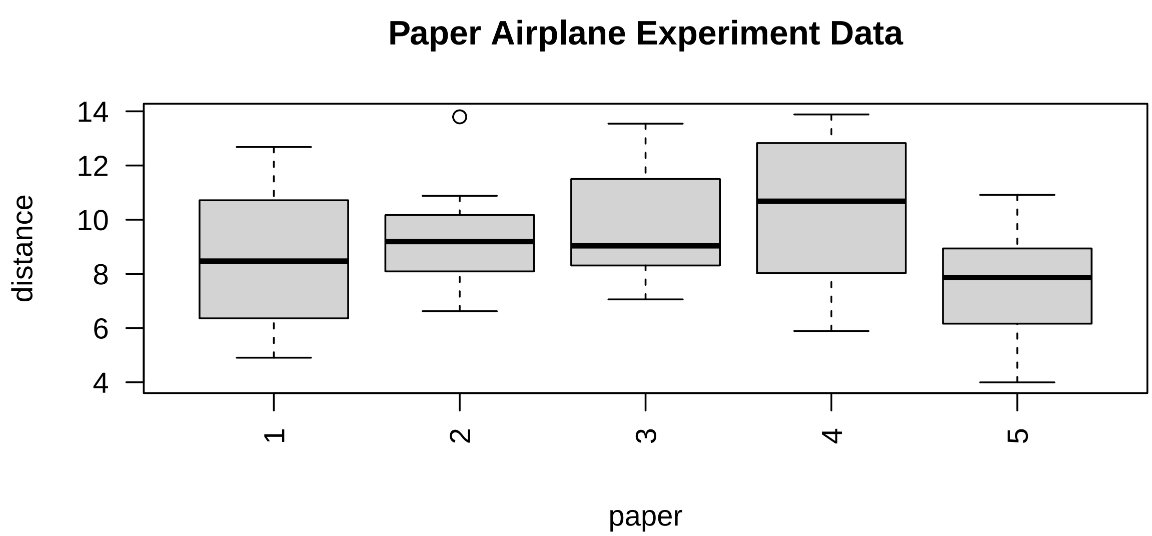
Since there are 5 levels in the paper factor, there must be 5 different types paper.

(b) Is there a true difference in the mean distance travelled by paper airplanes in the different groups?

The mean for all 35 measurements is 9; therefore, all simulated paper airplanes in all of the groups have the same mean. There is no difference.

(c) Provide the R code to obtain the following plot

```
boxplot(distances ~ papergroup, las = 2, xlab = "paper", ylab = "distance")
title("Paper Airplane Experiment Data")
```



(d) Fill in the blanks in the code below in order to get the following output.

```
_________(lm(____________ ~ _______________))
```

```
anova(lm(distances ~ papergroup))
```

```
## Analysis of Variance Table
##
## Response: distances
##            Df Sum Sq Mean Sq F value Pr(>F)
## papergroup  4  33.50  8.3751   1.199 0.3316
## Residuals  30 209.55  6.9849
```

(e) Based on the output from part (d), what can you conclude about the simulated paper airplanes? Briefly support your conclusions.

The p-value is .3316 which is not small. Therefore, there is no evidence that the mean distance travelled by the simulated paper airplanes are different for the different groups.

3. (6 marks) Consider the gas mileage data in `table.b3` of the *MPV* package.

(a) Fit a multiple regression model to estimate mean gas mileage y for cars with x_7 number of transmission speeds and having weight x_{10}.

```
library(MPV)
b3.lm <- lm(y ~ x7 + x10, data = table.b3)
coef(b3.lm)
```

```
##  (Intercept)           x7          x10
## 30.374638574  2.047521052 -0.004739167
```

```
summary(b3.lm)$sigma
```

```
## [1] 3.157661
```

The fitted model is

$$\widehat{y} = 30.375 + 2.048x_7 - 0.005x_{10}$$

where the error has mean 0 and an estimated standard deviation of 3.16.

(b) Use the model to estimate mean gas mileage for cars having weight 5000 pounds and 4 transmission speeds.

```
predict(b3.lm, newdata = data.frame(x7 = 4, x10 = 5000),
    interval="confidence")
```

```
##        fit      lwr      upr
## 1 14.86889 10.83812 18.89965
```

The 95% confidence interval for gas mileage for such cars is (10.8, 18.9).

4. (5 marks) Consider a data frame, such as the `women` object built into R, for which the heights could be taken as x values and the weights could be taken as y values.

   Write an R function called TukeySmooth which outputs a new data frame consisting of a column of equally spaced x values and a column of corresponding local medians, and which takes the following arguments

   - `x`: the vector of x values
   - `y`: the vector of y values
   - `x.min`: a constant which specifies the left boundary of the plotted curve
   - `x.max`: a constant which specifies the right boundary of the plotted curve
   - `window`: a constant which specifies the range of the x values used to calculate each of the moving medians.

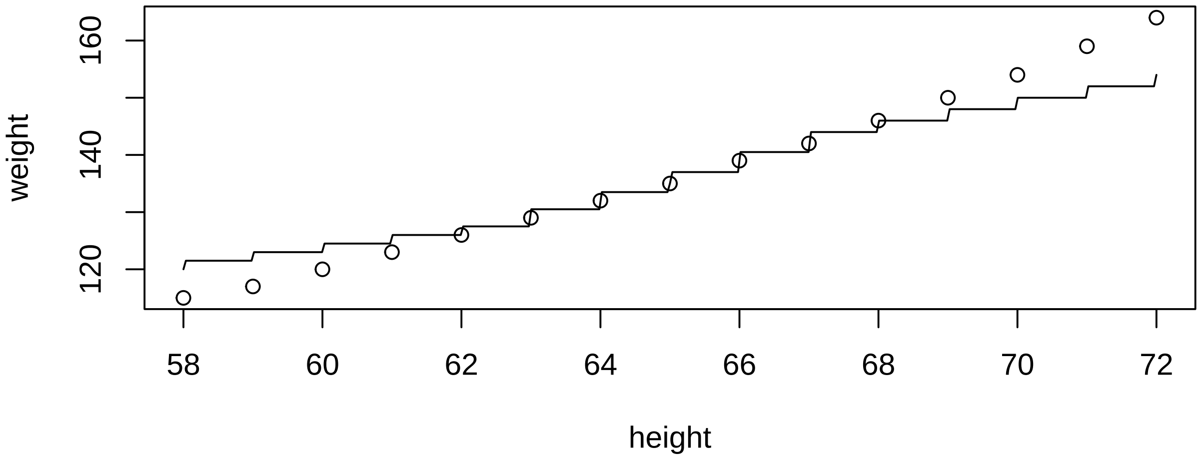
   The output for this function will be a data frame with 2 columns: `x` and `y`, which will correspond to the y-medians and the corresponding x locations where the medians are taken.

   This is very similar to the `smoother()` function described in the textbook, with median in place of mean.

```
TukeySmooth <- function(x, y, x.min, x.max, window=1) {
     xpoints <- seq(x.min, x.max, len=401)
     ymedians<- numeric(401)
     for (i in 1:length(xpoints)) {
         indices <- which(abs(x - xpoints[i]) < window)
         if (length(indices) < 1) {
              stop("Your choice of window width is too small.")
         } else {
              ymedians[i] <- median(y[indices])
        }
     }
     data.frame(x = xpoints, y = ymedians)
}
```

5. (2 marks) Apply the function obtained in the previous question, using a window width of 5, to the data in `women`, plotting the data, and overlaying the smooth curve.

```
women.TS <- TukeySmooth(women$height, women$weight, x.min = 58, x.max=72, window=5)
plot(weight ~ height, data = women)
lines(women.TS)
```



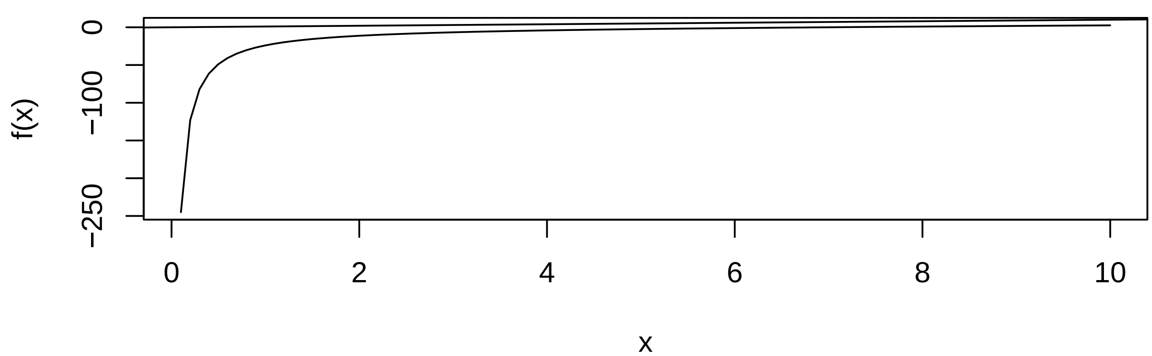
6. (5 marks) Consider the following iteration scheme:

$$x_{n+1} = f(x_n) := \frac{x_n}{2} - \frac{24.5}{x_n}$$

where x_0 is the initial value, say, $x_0 = 0.5$.

(a) Plot the graph of the function $f(x)$, for $x \in [0.1, 10]$ and overlay the straight line with intercept 0 and slope 1. Does $f(x)$ have a fixed point?

```
f <- function(x) x/2 - 24.5/x
curve(f(x), 0.1, 10)
abline(0,1)
```



Since the function and the line do not intersect, the function does not have a fixed point.

(b) Using a `for` loop in R, run 10 steps of the proposed scheme, printing out the value of x_n at each step.

```
x <- 25
for (i in 1:10) {
    x <- f(x)
    print(x)
}

## [1] 11.52
## [1] 3.633264
## [1] -4.926616
## [1] 2.509679
## [1] -8.507364
## [1] -1.373824
## [1] 17.14652
## [1] 7.144399
## [1] 0.1429396
## [1] -171.3296
```

(c) Based on your analysis, could the proposed scheme lead to a useful pseudorandom number generator?

On the basis of the analysis so far, we would not rule out this function as a possible starting point to produce a useful generator, but a lot of additional testing would be needed to ensure that it produced approximately independent and uniformly distributed numbers.

7. (4 marks) Which of the following linear congruential pseudorandom number generators have maximal cycle length?

(a) $a = 1025, c = 27, m = 2^{31}$ *Yes*

(b) $a = 1025, c = 54, m = 2^{31}$ *No, since c and m are both multiples of 2.*

(c) $a = 1025, c = 375, m = 2^{31}$ *Yes*

(d) $a = 10241, c = 375, m = 2^{31}$ *Yes*

8. (8 marks) Write a function called `myrbinom()` to compute `N` independent binomial random variates which have parameter `n` and `p`. The function should use the `runif()` function in a single `for()` loop to simulate `N` vectors of `n` Bernoulli(`p`) random variates which are then added to obtain the binomial numbers:

$$X = \sum_{i=1}^{n} B_i$$

is a binomial (n, p) random variable if $B_1, \ldots, B_n$ are independent Bernoulli (p) random variables.

Following the instructions of the question directly, we have:

```
myrbinom <- function(N, n, p) {
    X <- numeric(N)
    for (j in 1:N) {
        X[j] <- sum(runif(n) <= p)
    }
    X
}
```