The University of British Columbia

Computer Science/Data Science 405/505 Modelling and Simulation

Assignment 3 Solutions

Exercises

1. Suppose V has cdf $F_V(x) = 1 - e^{-x^2}$ when $x > 0$, and is 0, otherwise.

   (a) Find the quantile function for V, and write an R function called `rmyV` which takes `n` as an argument and returns a vector containing `n` random variates from the distribution of V.

$$F_V^{-1}(p) = \sqrt{-\log(1-p)}, \quad p \in [0,1]$$

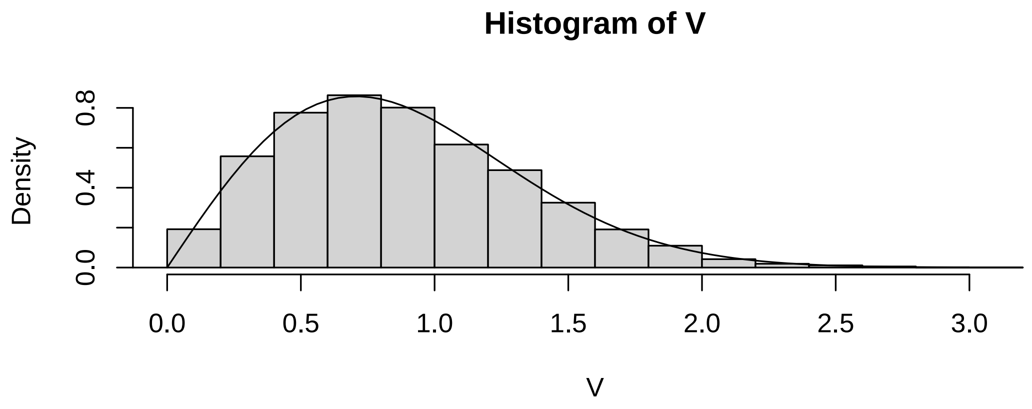
```
rmyV <- function(n) {
    U <- runif(n)
    X <- sqrt(-log(1-U))
X
}
```

   (b) Simulate 10000 values from the distribution of V and display the values in a relative frequency histogram with overlaid pdf curve.

$$f_V(x) = 2xe^{-x^2}, \quad x \geq 0,$$

and 0, otherwise.

```
V <- rmyV(10000)
hist(V, freq = FALSE)
fV <- function(x) 2*x*exp(-x^2)*(x>=0)
curve(fV(x), -1, 3, add = TRUE)
```



The histogram gives a good approximation to the density curve.

2. Suppose X has pdf $f_X(x) = 3x^2$, for $x \in [0,1]$, and 0, otherwise.

(a) Determine the cumulative distribution function of X.

$$F_X(x) = x^3, \qquad \text{for } x \in [0, 1]$$

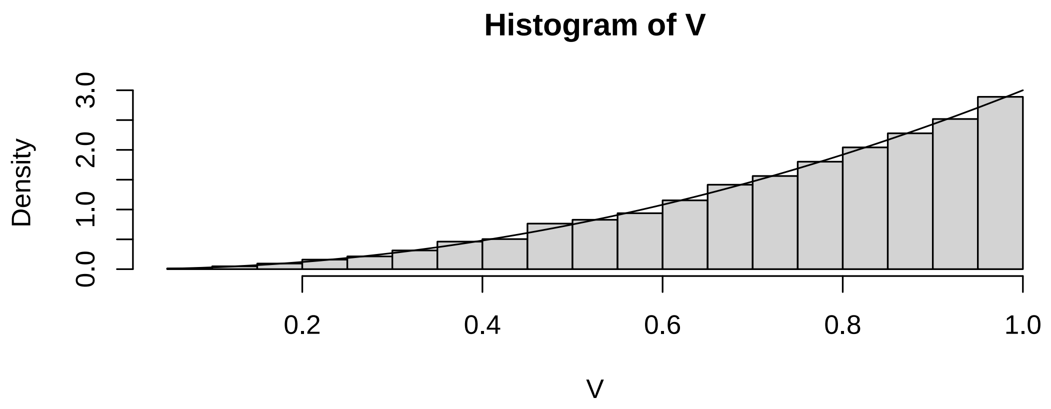
and 0, for $x < 0$ and 1, for $x > 1$.

(b) Find the quantile function for X, and write an R function called `rmyX` which takes `n` as an argument and returns a vector containing `n` random variates from the distribution of X.

$$F_X^{-1}(p) = (p)^{1/3}.$$

```
rmyX <- function(n) {
    U <- runif(n)
    X <- (U)^(1/3)
X
}
```

(c)
```
V <- rmyX(10000)
hist(V, freq=FALSE)
curve(3*x^2, add = TRUE)
```



3. Suppose p is a real number in the interval $(0, 1)$, and a random variable Y has pdf

$$g(y) = pf_V(y) + (1 - p)f_X(y)$$

where f_V and f_X are defined in questions 1 and 2.

(a) Determine the cumulative distribution function of Y.

$$G_Y(y) = pF_V(y) + (1 - p)F_X(y)$$

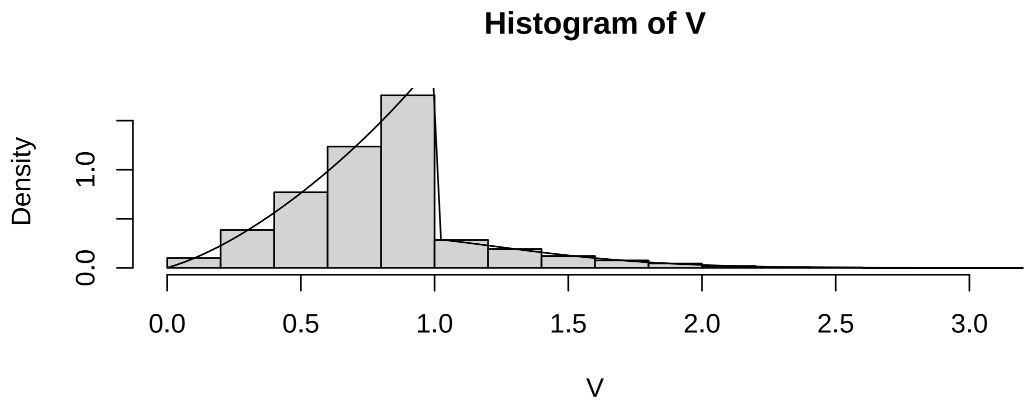
where F_V and F_X were defined in the first two questions.

(b) Write an R function called `rmyY` which takes `n` and `p` as arguments and returns a vector containing `n` random variates from the distribution of Y. (For this purpose, you will need to also use the `rbinom()` function, and the functions created in the previous exercises.)

```
rmyY <- function(n, p) {
    P <- rbinom(n, 1, p) # this is 1 w.p. p and 0 w.p. (1-p)
    Y <- P*rmyV(n) + (1-P)*rmyX(n)
    Y
}
```

(c) Simulate 10000 values from the distribution of Y, for the case where $p = 0.4$. and construct a relative frequency histogram with the graph of the pdf overlaid.

```
V <- rmyY(10000, p = .4)
hist(V, freq=FALSE)
fX <- function(x) 3*x^2*(x>=0 & x<=1)
curve(.4*fV(x) + .6*fX(x), add = TRUE)
```



4. Consider the pdf $h(x) = |x|\mathrm{e}^{-x^2}$, and suppose W has pdf $f_W(x) = ph(x-a) + (1-p)h(x-b)$ for real constants a, b and $p \in (0,1)$. Write a function called `rmyW` that takes arguments `a`, `b`, `p` and `n` and returns a vector of `n` random variates from the distribution of W. Obtain samples of 10000 Ws for the cases where $a = 1$, $b = 3$, and $p = .5$, and where $a = 1$, $b = 0.5$ and $p = .3$. Plot histograms with pdf curves overlaid.

Start with the simpler problem of simulating X from the distribution with PDF $f_X(x) = 2xe^{-x^2}$, since this X with a random sign will have PDF $h(x)$. The CDF of X is $F_X(x) = 1 - e^{-x^2}$ for $x \geq 0, and the quantile function is F_X^{-1}(p) = \sqrt{-\log(1-p)}$, so we can simulate X using the following function:

```
rX <- function(n) {
    U <- runif(n)
    X <- sqrt(-log(1-U))
    X
}
```

By multiplying by B where $B = 1$ with probability 0.5 and $B = -1$ with probability 0.5, we can simulate variates that follow the distribution with PDF $h(x)$.

```r
rh <- function(n) {
    U <- runif(n)
    X <- sqrt(-log(1-U))
    B <- 1-2*rbinom(n, size = 1, prob = 0.5)
    X*B
}
```

Simulating a random variate X from the distribution with PDF $h(x-a)$ is the same as simulating Y from $h(x)$ and adding a: that is, $X = Y + a$. We modify the above function to handle this change in location as follows:

```r
rh <- function(n, a) {
    U <- runif(n)
    X <- sqrt(-log(1-U))
    B <- 1-2*rbinom(n, size = 1, prob = 0.5)
    X*B + a
}
```

We can now write the function rmyW() as

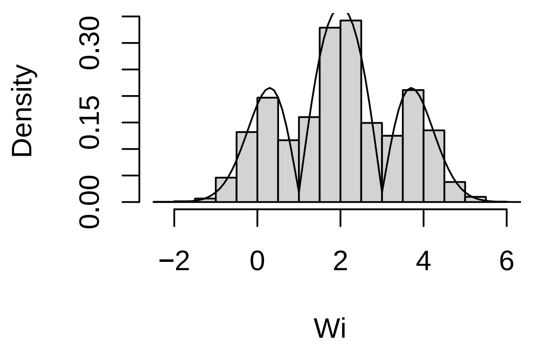
```r
rmyW <- function(n, a, b, p) {
    X1 <- rh(n, a)
    X2 <- rh(n, b)
    B <- rbinom(n, size = 1, prob = p)
    B*X1 + (1-B)*X2
}
```

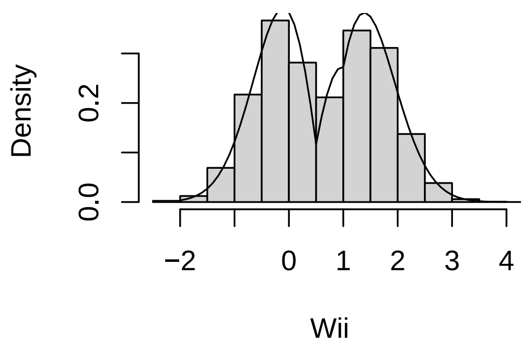
```r
Wi <- rmyW(10000, 1, 3, 0.5)
Wii  <- rmyW(10000, 1, 0.5, 0.3)
```

To plot the density curve, we need a function to compute the PDF:

```r
dh <- function(x, a) abs(x-a)*exp(-(x-a)^2)
dmyW <- function(x, a, b, p) {
      p*dh(x, a) + (1-p)*dh(x, b)
}
```

```r
par(mfrow=c(1,2))
hist(Wi, freq = FALSE, main="")
curve(dmyW(x, 1, 3, 0.5), -2, 8, add=TRUE)
hist(Wii, freq=FALSE, main="")
curve(dmyW(x, 1, 0.5, 0.3), -2, 8, add=TRUE)
```





5. Consider the following scenario. A sample of $2n$ patients with a particular disease are registered in a clinical trial for a new drug therapy. The patients have been randomly assigned to two equal groups of size n: a placebo group and a treatment group. The recovery time for each patient can be modelled with a lognormal distribution with parameters μ_i and σ_i, for $i = 1, 2$, depending on which group the patient has been assigned to. All patients are recruited to the trial at the same time and the trial ends at time T, at which point, the results would be analyzed. The recovery time for any patient who has not recovered before time T would not be known; this is an example of *censoring*.

   (a) Write a function called `rClinicalTrial` which takes `n`, `mu` (2-vector) and `sigma` (2-vector) as arguments and returns a data frame consisting of 3 columns: a column indicating the treatment group (1 or 2), a column of recovery times (some of which will not be known and should be simply recorded as T) and a column indicating whether the recovery time was censored (1) or not (0).

```r
rClinicalTrial <- function(n, mu, sigma, T) {
    trt.grp <- rep(c(1,2), each=n)
    recovery.times <- rlnorm(2*n, mu[trt.grp], sigma[trt.grp])
    censored <- recovery.times >= T
    recovery.times[censored] <- T
    trt.grp <- factor(trt.grp) # optional, for plot labels below
    levels(trt.grp) <- c("placebo", "drug") # optional
    data.frame(trt.grp, recovery.times, censored)
}
```

   (b) Simulate a clinical trial which should take 2 years, involving a total of 100 patients where under the placebo the parameter values are $\mu = 0.5$ and $\sigma = 1$, and under the drug treatment, the parameter values are $\mu = 0.5$ and $\sigma = 0.1$.

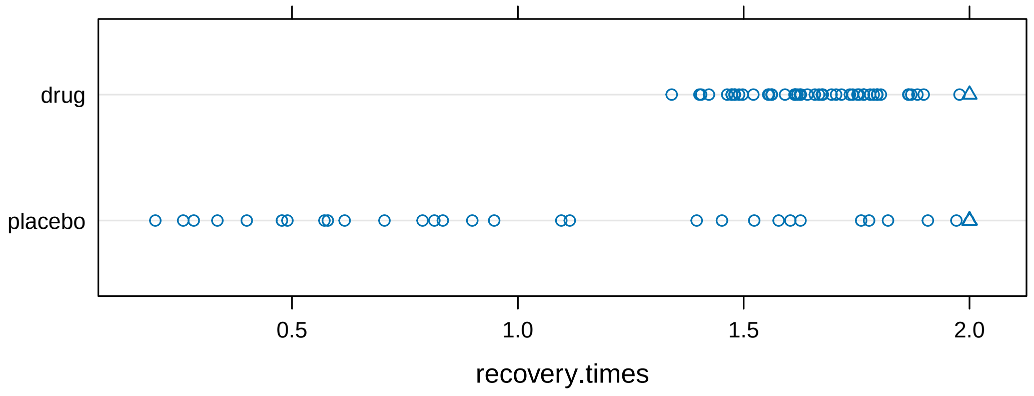
```r
simulatedClinicalTrial <- rClinicalTrial(50, mu=c(.5, .5),
    sigma=c(1, .1), T = 2)
```

   (c) Construct side-by-side dot plots of the two groups of simulated data, highlighting the censored observations with a different plotting character from the other observations.

```
par(mar=c(3, 3, .75, .5))
library(lattice)
names(simulatedClinicalTrial)

## [1] "trt.grp"        "recovery.times" "censored"

dotplot(trt.grp ~ recovery.times,  data = simulatedClinicalTrial,
    pch=1+simulatedClinicalTrial$censored)
```



6. Repeat the previous question, but this time, under the assumption that patients are recruited to the study at different times - modelled as a gamma random variable with shape and scale parameters α and β. The function `rClinicalTrial` will now need additional arguments called `alpha` and `beta` but will return the same kind of data frame, where the censoring times are now the length of time the subject was in the study at time T. Run the simulation with $\alpha = 2$ and $\beta = .2$.

```
rClinicalTrial <- function(n, mu, sigma, T, alpha, beta) {
    trt.grp <- rep(c(1,2), each=n)
    start.times <- rgamma(2*n, shape=alpha, scale=beta)
    recovery.times <- rlnorm(2*n, mu[trt.grp], sigma[trt.grp])
    censored <- (start.times + recovery.times) >= T
    recovery.times[censored] <- T - start.times[censored]
    trt.grp <- factor(trt.grp) # optional, for plot labels below
    levels(trt.grp) <- c("placebo", "drug") # optional
    data.frame(trt.grp, recovery.times, censored)
}
```

New simulated data:

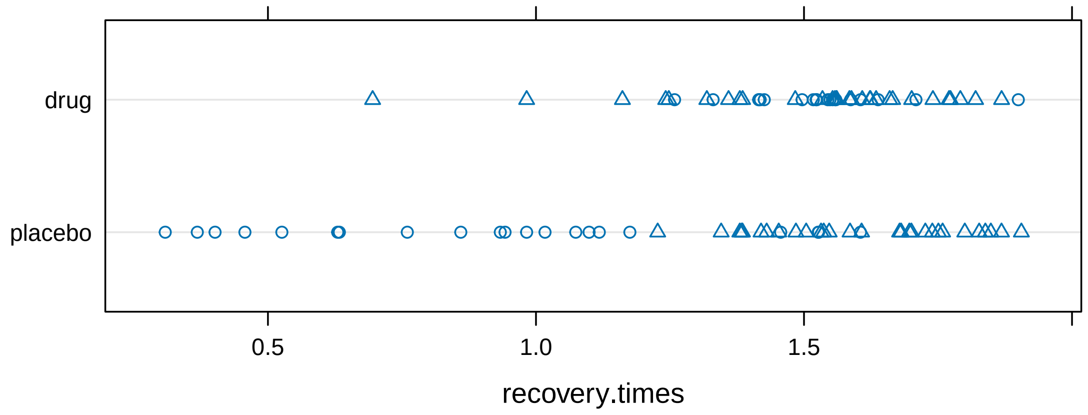
```
simulatedClinicalTrial <- rClinicalTrial(50, mu=c(.5, .5),
    sigma=c(1, .1), T = 2, alpha = 2, beta = .2)
```

Side by side dot plots of new simulated data:

```r
par(mar=c(3, 3, .75, .5))
library(lattice)
names(simulatedClinicalTrial)

## [1] "trt.grp"        "recovery.times" "censored"

dotplot(trt.grp ~ recovery.times,  data = simulatedClinicalTrial,
    pch=1+simulatedClinicalTrial$censored)
```



7. Consider the Pareto distributions of Examples 6.28 and 6.29, and suppose X is a random variable with PDF

$$f_X(x) = \frac{(k-1)}{2(1+|x|)^k}$$

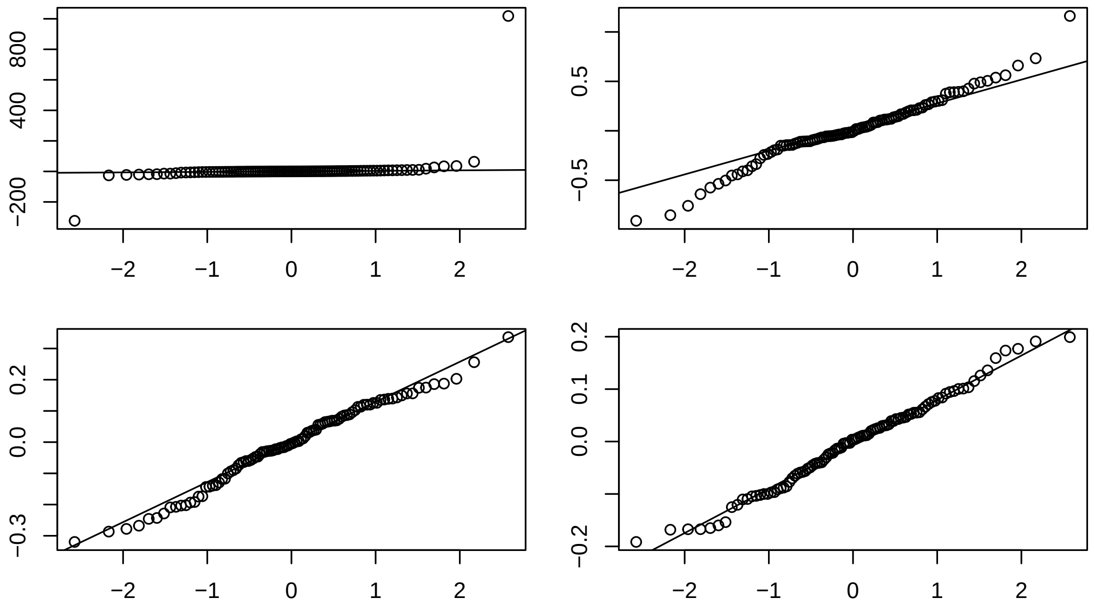
where $k \in \{2, 3, \ldots\}$.

(a) Write a function which takes `n` and `k` as arguments and returns a vector of length `n` containing simulated values from this distribution. The simplest way to do this will be to use the `rbinom()` function and the Pareto simulator to randomly assign positive or negative signs to the variates.

```r
rpareto2 <- function(n, k) {
    U <- runif(n)
    X <- (1-U)^(-1/(k-1)) - 1
    X*(1-2*rbinom(n, 1, 0.5))
}
```

(b) For $k = 2, 3, 4$ and $k = 5$, simulate 100 samples of size 50, calculating the averages in each case. (To do this step, you should use a `for()` loop.) Construct normal QQ-plots of the 100 averages for each value of k.

```r
n <- 50 # sample size
xbar <- numeric(100) # this will hold the averages
par(mfrow=c(2, 2)) # we want a 2 by 2 layout of QQ-plots
for (k in 2:5) {
    for (j in 1:100) {
        xbar[j] <- mean(rpareto2(n, k))
    }
    qqnorm(xbar) # normal QQ-plot
    qqline(xbar) # reference line for QQ
}
```



(c) For which values of k does the central limit theorem appear to hold? What condition of the central limit theorem is violated in the other cases? *The QQ-plots give fairly straight lines when $k = 4$ and $k = 5$. This means that the averages are approximately normally distributed in those cases. Therefore, the central limit theorem appears to hold when $k = 4$ and $k = 5$. When $k = 2$ and $k = 3$, the variance of the X's is not finite, which violates the conditions of the central limit theorem.*