Assignment 1

Bjørn Christian Weinbach

15th September, 2020

Clear R environment

```r
rm(list = ls())
```

Problem 1

a.) Verify that this is a proper probability density function.

We have the proposed pdf

$$f(x) = \frac{1}{2} - \frac{x}{4}, \quad -1 \leq x \leq 1 \tag{1}$$

which is 0 whenever $x \notin [-1, 1]$

To be qualified as a pdf the function need to satisfy the following:

1. $$\int_{-\infty}^{\infty} f(x)dx = 1 \tag{2}$$

2. $$f(x) \geq 0, \qquad \text{for any x} \tag{3}$$

Condition 2. can easily be validated by inspection. any x in the interval -1 to 1 will yield a positive value.

The condition 1, must be validated through integration. We do this analytically like so

$$\int_{-\infty}^{\infty} f(x)dx = \int_{-1}^{1} \frac{1}{2} - \frac{x}{4}dx = \left[\frac{x}{2} - \frac{x^2}{4}\right]_{-1}^{1} = 1 \tag{4}$$

a.) Find the mean $E(X)$ and the variance $Var(x)$.

Expectation is formally defined as

$$E[X] = \int_{\mathbb{R}} xf(x)dx \tag{5}$$

We do this analytically and get

$$E[X] = \int_{\mathbb{R}} xf(x)dx = \int_{-1}^{1} \frac{x}{2} - \frac{x^2}{4}dx = \left[\frac{x^2}{4} - \frac{x^3}{12}\right]_{-1}^{1} = -\frac{1}{6} \tag{6}$$

Variance is defined as

$$Var(X) = \int_{\mathbb{R}} x^2 f(x)\, dx - \mu^2 \tag{7}$$

Where $\mu = E[X]$

We do this analytically like so

$$Var(X) = \int_{\mathbb{R}} x^2 f(x)\, dx - \mu^2 = \int_{-1}^{1} \frac{x^2}{2} - \frac{x^3}{4} dx - (-\frac{1}{6})^2 = \left[\frac{x^3}{12} - \frac{x^4}{16}\right]_{-1}^{1} - \frac{1}{36} = \frac{11}{36} \tag{8}$$

b.) Find the cumulative distribution function $F(x)$ associated with X,

and use this to calculate $P(X < 0)$ and $P(-0.5 < X < 0.5)$.

$F_X(x)$ is formally defined as

$$F_X(x) = \int_{-\infty}^{x} f_X(t) dt \tag{9}$$

So in our case, the cumulative distribution function for f(x) is

$$F_X(x) = \int_{-\infty}^{x} f_X(t)\, dt = \int_{-1}^{x} \frac{1}{2} - \frac{x}{4} dt = \left[\frac{t}{2} - \frac{t^2}{4}\right]_{-1}^{x} = \frac{1}{8}(-x^2 + 4x + 5) \tag{10}$$

Since $F_X(x) = P(X < x)$, we can calculate the probabilities above. This gives us that $P(X < 0) = \frac{5}{8} = 0.652$ and $P(-0.5 < X < 0.5) = P(X < 0.5) - P(X < -0.5) = 0.84375 - 0.34375 = 0.5$

To find $F_X^{-1}(u)$ we set $U = \frac{1}{8}(-x^2 + 4x + 5)$ and solve for x which gives us

$$X = 2 \pm \sqrt{9 - 8u} \tag{11}$$

And we choose $X = 2 - \sqrt{9 - 8u}$ since we will use this together with the inverse transformation method to generate numbers X from the distribution above and this particular solution is the only one that will return values in the inverval $[-1, 1]$

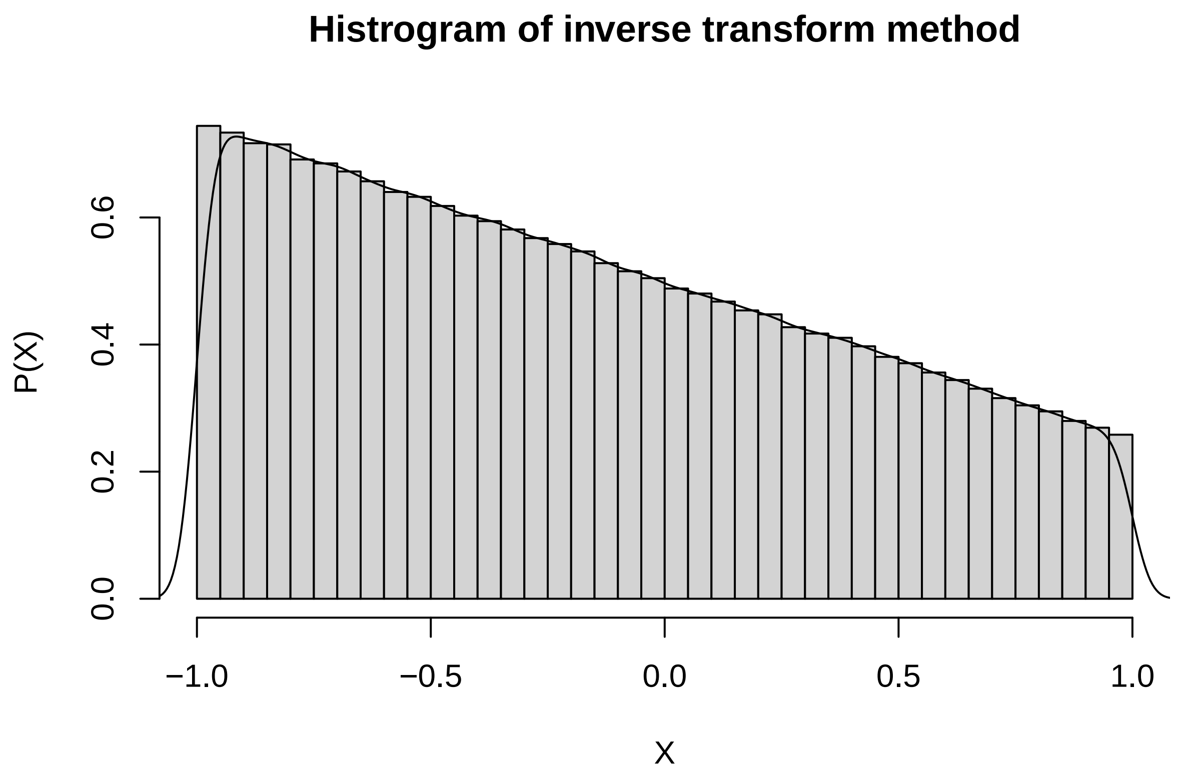
c.) Write a function (with single argument n) which produces n random numbers

with density (1)

```r
# itm (Inverse Transform Method) function for equation one in assignment 1
itm_equation1 <- function(n) {
  u <- runif(n)
  return (2 - sqrt(9 - 8*u))
}
```

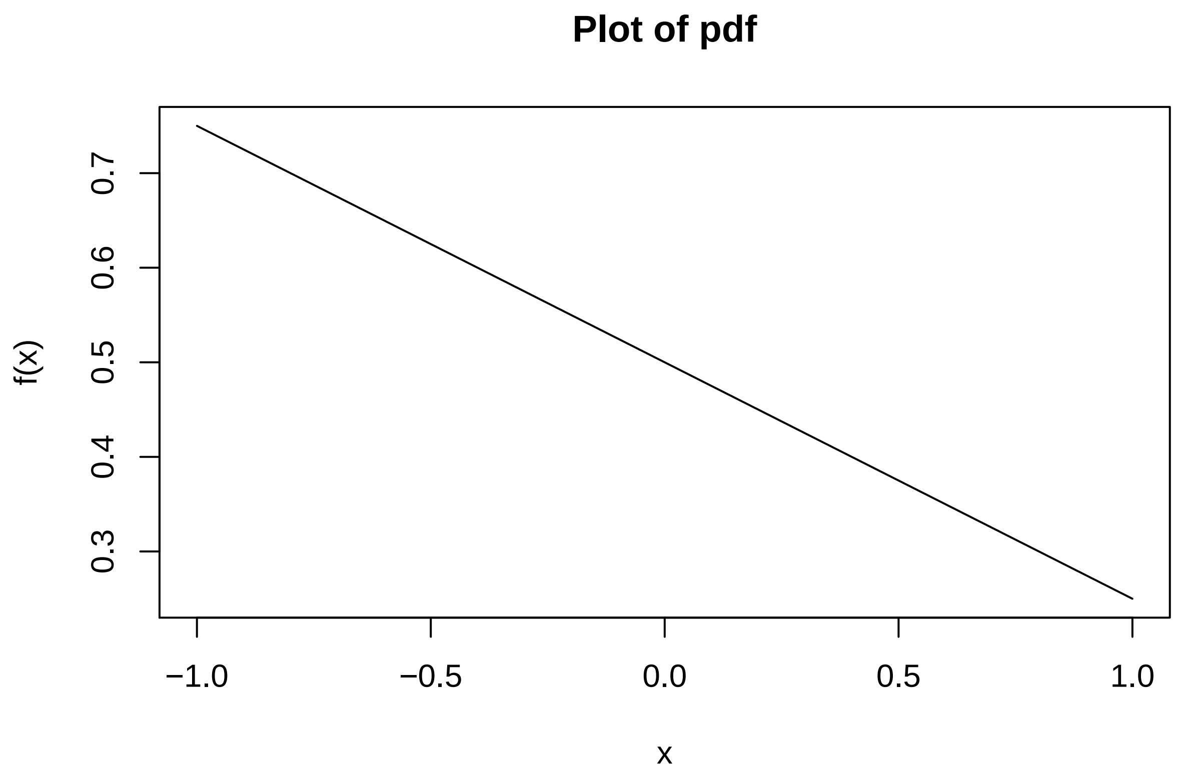
Check that your function produces correct results by comparing a histogram

```r
Nsim <- 1000000
itm_p <- itm_equation1(Nsim)

hist(itm_p, breaks = 30, prob = TRUE, xlab = "X", ylab = "P(X)",
     main="Histrogram of inverse transform method")
lines(density(itm_p))
```



And to see that this is correct, we plot the pdf provided by assignment 1 to see that this indeed generate numbers from the given distribution.

```
theoretical_pdf <- function(x) {
  return((1/2) - (x/4))
}

x <- seq(-1, 1, 0.01)
plot(x, theoretical_pdf(x), main = "Plot of pdf", xlab = "x", ylab = "f(x)",
     type ="l")
```



d.) Using a $uniform(-1,1)$ proposal distribution g, do the calculations required to obtain an accept-reject method for drawing random numbers with density (1)

The uniform distribution in interval $[a, b]$ has pdf

$$f_{u(a,b)}(x) = \frac{1}{b-a}, \qquad a \leq x \leq b \tag{12}$$

Which we will use later. The algorithm for acceptance-rejection algorithm is as follows:

1. generate y from g

where g is the proposal, in our case $uniform(-1,1)$.

2. generate U from $uniform(0,1)$
3. accept y if $u < \frac{f(y)}{cg(y)}$ and let $x = y$

to find the constant c. such that $\frac{f(t)}{g(t)} \leq c$ for all t where $f(t) > 0$. In our case, this means finding the maximum value of the equation

$$\frac{f(t)}{g(t)} = \frac{f_X(x)}{f_{u(-1,1)}(x)} = \frac{\frac{1}{2} - \frac{x}{4}}{\frac{1}{2}} \implies 1 - \frac{x}{2}, \qquad -1 \leq x \leq 1 \tag{13}$$

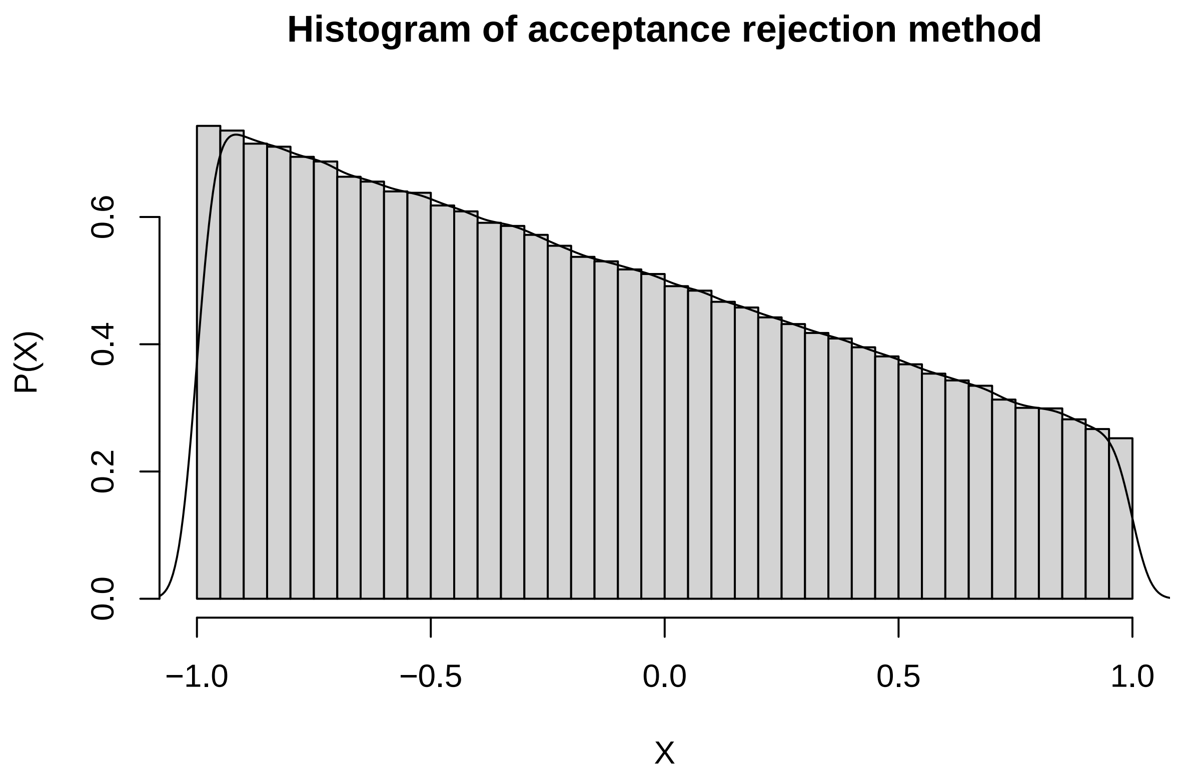
Which has maximum value at $x = -1$ which gives $f(-1) = 1 + \frac{1}{2} = 1.5$ which means our constant $c = 1.5$.

e.) Write a function (with single argument n) which produces n random numbers

with density (1) using the accept-reject method. Check the results in the same way as in point c.Which method would you prefer in this case - accept-reject or inverse transform? The function system.time is useful in this regard.

We apply the algorithm described above in R:

```r
# ar (acceptance rejection) for equation 1
ar_equation1 <- function(n) {
  f <- function(x) (1/2 - x/4)
  C = 1.5

   naccepts <- 0
   result.sample <- rep(NA, n)

   while (naccepts < n) {
    y <- runif(1, -1, 1)
    u <- runif(1)

    if ( u <= f(y) / (C*(1/2))) {
      naccepts <- naccepts + 1
      result.sample[naccepts] = y
    }
   }

   result.sample
}

Nsim <- 1000000
ar_p <- ar_equation1(Nsim)
hist(ar_p, probability = TRUE, main="Histogram of acceptance rejection method",
     xlab = "X", ylab = "P(X)", breaks=30)
lines(density(ar_p))
```



This function is alot slower than the inverse transform method, mainly because we iterate until n samples have been generated instead of using vectorized code. This slows down the code because of R being an interpereted language.

You can write a vectorized version which takes into account the probability of accepting a propasal and scaling up the number of simulations accordingly, but this does not guarantee that N samples is generated since the expected number of iterations needed is a geometric distrubution with expected value $E[X] = \frac{1}{p(accept)}$

f.) Check the answers you got for expectations, variance and probabilities in

points a) and b) above by simulation.

Let's begin with expectation. Analytically, the expectation is $E[X] = -\frac{1}{6}$. In R, the simulated smaple is:

```
# Mean of inverse transform method simulated values
mean(itm_p)
```

```
## [1] -0.1670969
```

```
#Mean of acceptance rejection simulated values
mean(ar_p)
```

```
## [1] -0.1673839
```

```
# Analytical mean
-1/6
```

```
## [1] -0.1666667
```

We se that both methods generate the correct mean that we have calculated.

Now let's check the variance:

```r
# Variance of inverse transform method simulated values
var(itm_p)
```

```
## [1] 0.305857
```

```r
# Variance of acceptance rejection simulated values
var(ar_p)
```

```
## [1] 0.3052262
```

```r
# Analytical variance
11/36
```

```
## [1] 0.3055556
```

To find the 95% confidence interval, we first take the sample mean of the data. We know that sample means are normally distributed, from the central limit theorem. We can then use the formula for confidence intervals using Z-scores

$$CI = \bar{X} \pm Z\frac{\sigma}{\sqrt{n}} \tag{14}$$

In R, we get:

```r
CI <- function(data) {
  xhat <- mean(data)
  sd <- sd(data)
  n <- length(data)
  lower <- xhat - 2*(sd / sqrt(n))
  upper <- xhat + 2*(sd / sqrt(n))
  return(c(lower, upper))
}

CI(itm_p)
```

```
## [1] -0.1682029 -0.1659908
```

Problem 2

a.) Which distribution does $A = \sum_{i=1}^{n} X_i$ have?

Since A is a convolution of iid normal distributions, A itself is a normal distribution with parameters $N(\sum_{i=1}^{n} \mu_{X_i}, \sum_{i=1}^{n} \sigma^2_{X_i})$

a.) Which distribution does $B = \frac{1}{n}\sum_{i=1}^{n} X_i$ have?

The same holds true for linear combinations of random variables and we get μ and σ^2 by using the formula for expectation and variance of linear combinations of random variables. Therefore B has the distribution $B \sim N(\mu_X, \frac{\sigma^2_X}{n})$

a.) Which distribution does $C = \frac{\sqrt{n}(B-\mu)}{\sigma}$ have?

By observation, we see that this resembles the formula for taking a normal deviate X to get a standard normal distribution Z. If we take the distribution B and translate it to a standard normal we get:

$$Z = \frac{X - \mu_X}{\frac{\sigma}{\sqrt{n}}} = \frac{\sqrt{n}(X - \mu_X)}{\sigma} \tag{15}$$

And therefore, the distribution of C is $N(0, 1)$

b.) Check the results you got in a).

We do this by simulating many independent replications of each of A, B and C. With $n = 100$, $\mu = 1$ and $\sigma = 2$.

R code below.

```r
# n simulations, k random variables and parameters
simulate_n_normal <- function(n, k, mu, sigma) {
  samples <- rnorm(n*k, mu, sigma)
  m <- matrix(samples, n, k)
  return(m)
}

Nsims <- 10000
Nrvs <- 100
mu <- 1
sigma <- 2

samples <- simulate_n_normal(Nsims, Nrvs, mu, sigma)
A <- rowSums(samples)
B <- A / Nrvs
C <- (sqrt(Nrvs)*(B - mean(samples)))/(sd(samples))

# Analytic expectation of A
mu*Nrvs
```

```
## [1] 100
```

```r
# Expectation by simulation
mean(A)
```

```
## [1] 100.3244
```

```r
# Analytic Standard deviation
sqrt(Nrvs)*sigma
```

```
## [1] 20
```

```r
# Standard deviation by simulation
sd(A)
```

```
## [1] 20.24087
```

```r
# Analytic expectation of B
mu
```

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

```r
# Expectation by simulation
mean(B)
```

```
## [1] 1.003244
```

```r
# Analytic standard deviation
sqrt(sigma**2 / Nrvs)
```

```
## [1] 0.2
```

```r
# Standard deviation by simulation
sd(B)
```

```
## [1] 0.2024087
```

```r
# Analytic expectation of C
0
```

```
## [1] 0
```

```r
# Expectation by simulation
mean(C)
```

```
## [1] -3.483675e-16
```

```r
# Analytic standard deviation
1
```

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

```r
# Standard deviation by simulation
sd(C)
```

```
## [1] 1.011599
```

c.) Find number of required simulations to get CI is not larger than 0.1

We use the following formula:

$$n = 4Var(X)/r^2 \tag{16}$$

Where $r = 0.1$ is the error. This gives us $n = 1600$

d.) For the n obtained in c.) and $\mu = 1$, simulate many outcomes of X.

We implement this in R with $N = 1600$ and $N = 10$

```r
outside_ci <- function(sims) {
  samples <- rnorm(sims, 1, 2)
  B <- mean(samples)
  S <- sd(samples)

  lower <- B - 1.96*(S/sqrt(sims))
  upper <- B + 1.96*(S/sqrt(sims))
  return(upper < 1 | lower > 1)
}

vector <- c()
for (i in 1:1000)
  vector[i] <- outside_ci(1600)

# Probability of mean outside range n = 1600
sum(vector)/length(vector)
```

```
## [1] 0.062
```

```
vector <- c()
for (i in 1:1000)
  vector[i] <- outside_ci(10)

# Probabiloity of mean outside range n = 10
sum(vector)/length(vector)
```

```
## [1] 0.091
```

We observe that the probability for getting a confidence interval that is outside the population parameter is about 5 percent when $N = 1600$ (which it should) but about 8% when $N = 10$.

Problem 3

a.) Import SEIR-model

```
source("seir_model.R")
f <- seir_sim()
```

a.) Find the number of persons in state S (susceptible) on March 15th 2021.

```
t <- which(f$dates=="2021-03-15")
print(f$ode[t,"S"])
```

```
## [1] 0.8690278
```

We have that 86.9% is susceptible on the 15th of march 2021.

a.) The maximum number of persons in critical care (CC) over the period, and also on which date this occurs.

```
max_index <- which.max(f$ode[,"CC"])
f$ode[max_index, "CC"]
```

```
## [1] 7.898301e-05
```

```
f$dates[max_index]
```

```
## [1] "2021-02-17"
```

We see that the maximum no of people in critical care is about $7.9 \times 10^{-3}\%$ and this occured on 17th February, 2021

a.) Find average number of people in critical care in 2021

```
t <- which(f$dates >= "2021-01-01" & f$dates <= "2021-12-31")
mean(f$ode[t,"CC"])
```

```
## [1] 3.934788e-05
```

We see that the average number of people in critical care predicted by the model is about $4 \times 10^{-3}\%$

b.) Run the model but now with no social disancing

```
f2 <- seir_sim(upper_thresh=10000)
```

b.) Find maximum number of people in critical care and on what date

```
max_index <- which.max(f2$ode[,"CC"])
f2$ode[max_index, "CC"]
```

```
## [1] 0.001163043
```

```
f2$dates[max_index]
```

```
## [1] "2020-06-01"
```

And we now find that the number of people in critical care with no social social distancing is 0.11% which more intuitively is 1392 times more than the maximum number in critical care in the same period with social distancing.

This would happen on 1st June, 2020 which is much earlier than with social distancing.

b.) Using the definition that an epidemic ends when fewer than 1 per 1 million people are in the E-state, how long does it take to reach the end of the epidemic?

```
t <- which(f2$ode[,"E"] < 1*10^-6 & f2$dates >= "2020-04-01")
f2$date[t[1]]
```

```
## [1] "2021-04-12"
```

Our model predicts that with no social distancing, the epidemic would end at 12th April, 2021.

b.) Which proportion of the population has been infected (i.e. in RR, RH or RC) when the epidemic ends in this model (herd immunity)?

```
sum(f2$ode[t[1], c("RR", "RH", "RC")])
```

```
## [1] 0.6056311
```

At this point, our model predicts that about 60.6% of the population is recovered or dead at the time of the predicted end of the epidemic.

c.) Simulate many (at least 100) SEIR models with different random values of

R0max and season ampliude drawn from uniform distributions.

```
nsim <- 200 # number of simulation replica
days <- 366 # 366 days in 2020.

t <- which(f2$dates<="2020-12-31" & f2$dates>="2020-01-01")
m <- matrix(,nsim, days)
r0s <- vector(mode="numeric",length=nsim)
sas <- vector(mode="numeric",length=nsim)

for(i in 1:nsim){
  R0max <- runif(1, min=1.8, max=3.0)
  r0s[i] <- R0max
  season_amplitude <- runif(1, min=0, max=0.4)
  sas[i] <- season_amplitude
  ff <- seir_sim(R0max = R0max,
                 season_amplitude = season_amplitude)
  m[i,] <- 10000*ff$ode[t,"CC"]
}
```

c.) The mean and standard deviation, the 90%, 95% and 99% quantiles of the maximum number of required critical care beds during 2020.

```
mean(m)
```

```
## [1] 0.4143066
```

```
sd(m)
```

```
## [1] 0.3465534
```

```
quantile(m, c(.90, .95, .99))
```

```
##       90%       95%       99%
## 0.8429838 1.0306407 1.4516038
```

Our simulated values are seen above, these are different each time the code block above is run due to random sampling for the parameters.

c.) Calculate the probability that more than 100 critical care beds per million i.e 1 per 10000 people are required.

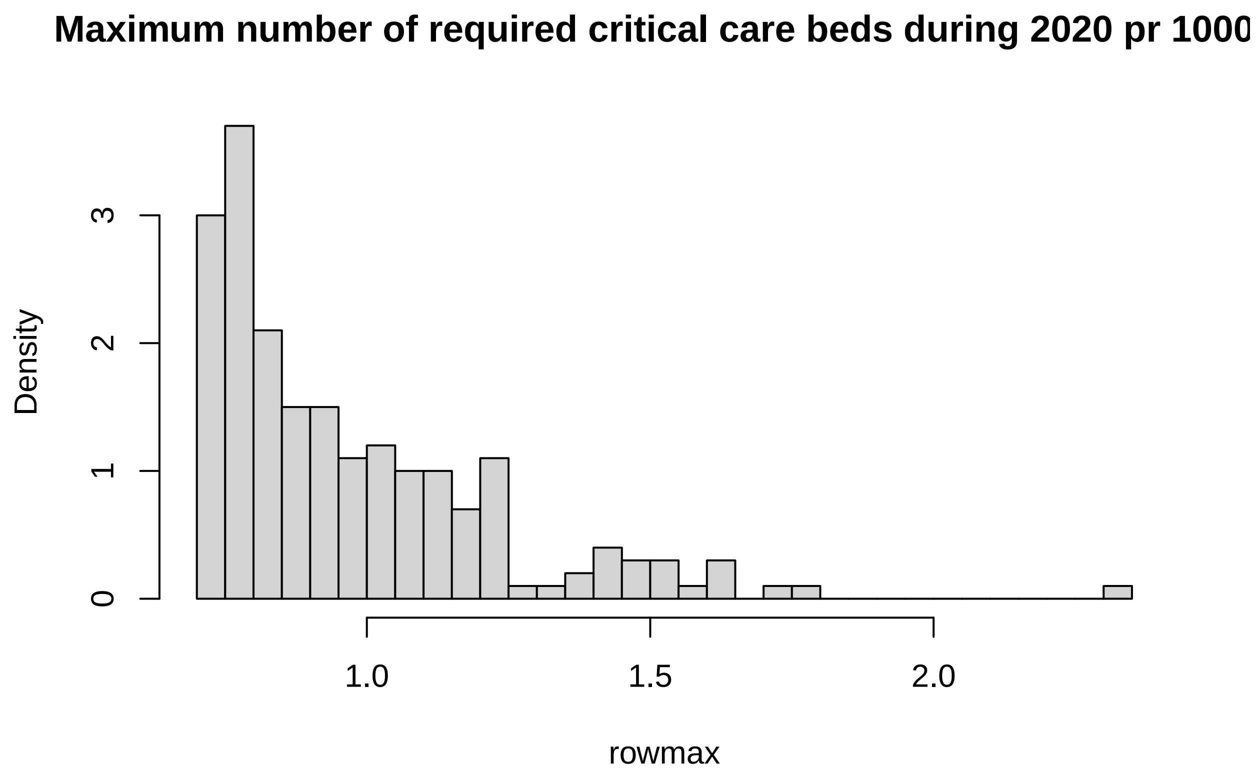
```
p <- sum(m >= 1) / length(m)
p
```

```
## [1] 0.0556694
```

Our simulated probability is seen above, this differs for each run due to random sampling.

c.) Make a graphical representation of the maximum number of required critical care beds during 2020.

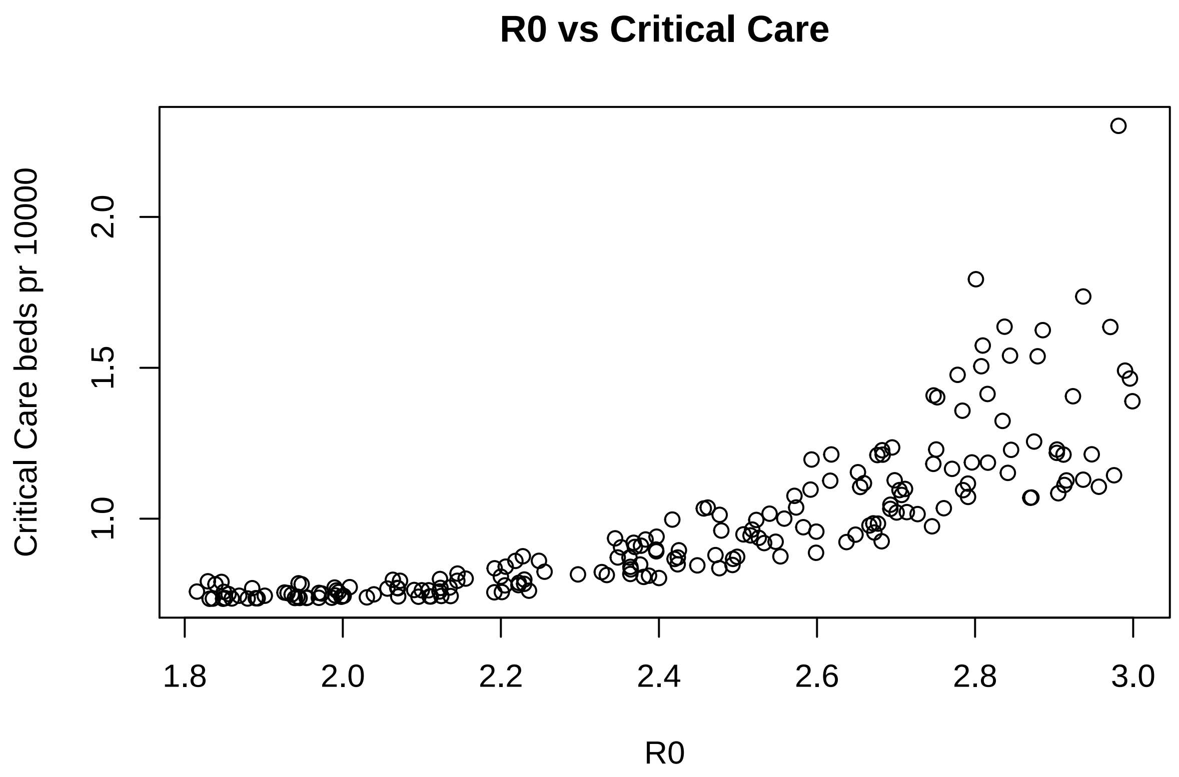
```
rowmax <- apply(m, 1, function(x) max(x))
hist(rowmax, probability = TRUE, breaks = 30,
     main="Maximum number of required critical care beds during 2020 pr 10000")
```



This is not well approximated by a normal distribution.

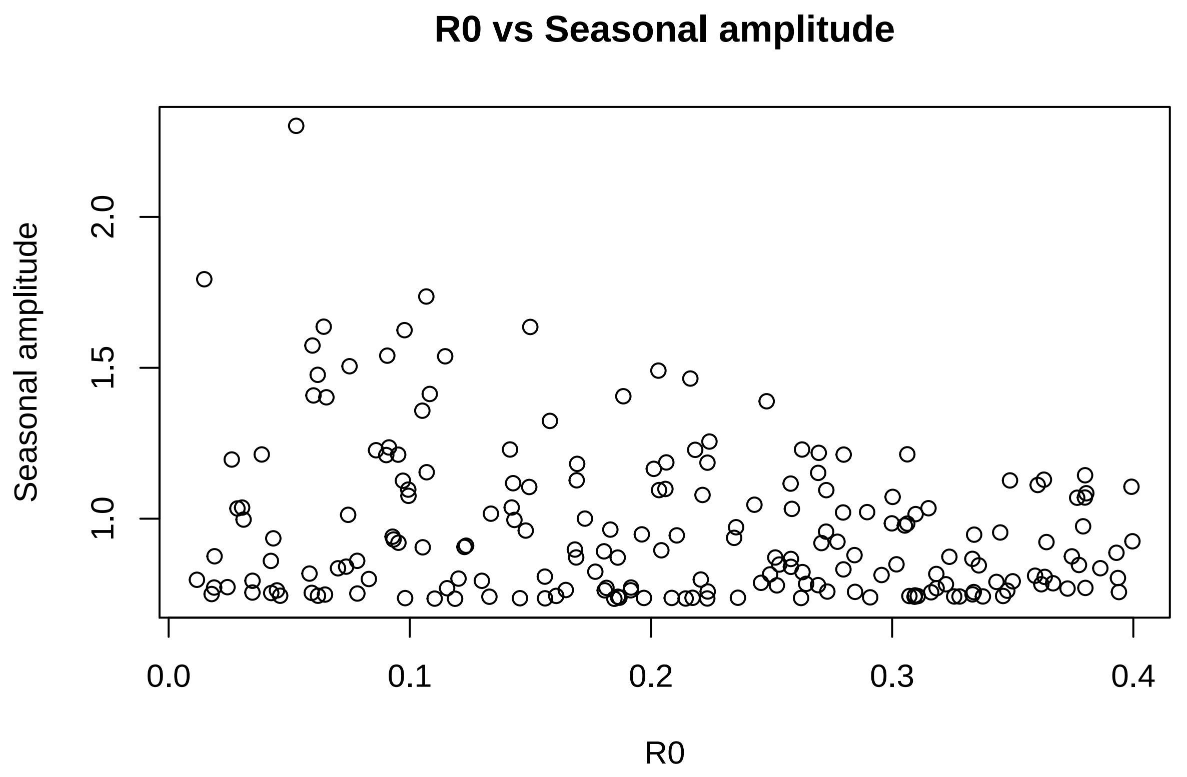
d.) Scatterplot of R0max

```
plot(x = r0s,
     y = rowmax,
     xlab = "R0",
     ylab = "Critical Care beds pr 10000",
     main = "R0 vs Critical Care"
)
```



d.) Scatterplot of season ampliude

```
plot(x = sas,
     y = rowmax,
     xlab = "R0",
     ylab = "Seasonal amplitude",
     main = "R0 vs Seasonal amplitude"
)
```



d.) Discussing linear relationship between critical care beds and the two parameters.

By visual inspection of the plots. We see that there is a linear relationship between R_0 and critical care beds needed. This is not the case for seasonal amplitude to the same extent and it seems that seasonal amplitude has a slight negative correlation with required critical care beds.

We clearly see a correlation between R_0 and critical care beds this suggests that R_0 influences the number of critical care beds. If one is to allow oneself to conclude a causal relationship in one direction. Which in this case is quite reasonable.

d.) Calculate the correlation between critical care beds and the two parameters

```r
# Correlation - R0 vs CC
cor(rowmax, r0s)
```

```
## [1] 0.8162049
```

```r
# Correlation - Seasonal Amplitude vs CC
cor(rowmax, sas)
```

```
## [1] -0.2527392
```

And this calculation confirms what we have observed by visual inspection of the plots.