Statistical Computing HW 4

Mario Ibanez

January 30, 2016

Problem 4.3)

Give an efficient algorithm to simulate the value of a random variable X such that

$$P\{X = 1\} = 0.3$$
$$P\{X = 2\} = 0.2$$
$$P\{X = 3\} = 0.35$$
$$P\{X = 4\} = 0.15$$

Answer

Derivation

The CDF of the distribution is:

$$F(x) = \begin{cases} 0 & x < 1 \\ 0.3 & 1 \leq x < 2 \\ 0.5 & 2 \leq x < 3 \\ 0.85 & 3 \leq x < 4 \\ 1 & 4 \leq x \end{cases}$$

The can be used along with a standard uniform random variable to generate values of the random variable X.

Algorithm

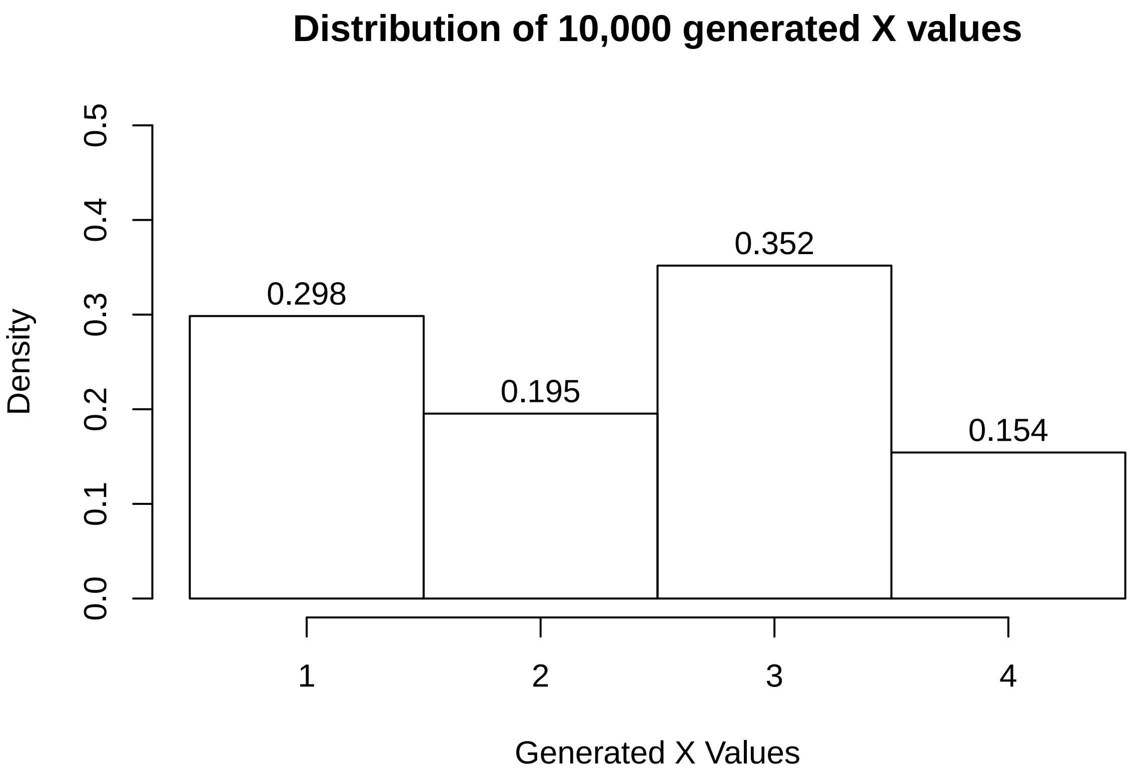
This is slightly different than the inverse transform method in the book but the outcome is the same. 1. Generate a number u from the standard uniform distribution 2. Set $x = F^{-1}(u) + 1$ 3. Repeat n times

Program

Below is a program and histogram using the derivation and algorithm. 10,000 values were generated. The distribution is quite good.

```
# F inverse function
F_inv <- function(u){
  x <- rep(0, length(u))
  if(u < 0.30)
    x <- 0 + 1
  else if((u >= 0.30) && (u < 0.50))
    x <- 1 + 1
  else if((u >= 0.50) && (u < 0.85))
    x <- 2 + 1
  else if((u >= 0.85) && (u < 1))
```

```r
    x <- 3 + 1
  return(x)
}
# This line is generates the x values
x <- sapply(array(runif(10^4)), FUN = F_inv)
```



Problem 4.14a)

Let X be a binomial random variable with parameters n and p. Suppose that we want to generate a random variable Y whose probability mass function is the same as the conditional mass function of X given that $X \geq k$, for some $k \leq n$. Let $\alpha = P\{X \geq k\}$ and suppose that the value of α has been computed. Give the inverse transform method for generating Y.

Answer

Derivation

Algorithm

Program

Analytical Result

Problem 4.15)

Give a method for simulating X, having the probability mass function $p_j,\ j = 5, 6, 7, ..., 14$, where

$$p_j = \begin{cases} 0.11 & \text{when } j \text{ is odd and } 5 \leq j \leq 13 \\ 0.09 & \text{when } j \text{ is even and } 6 \leq j \leq 14 \end{cases}$$

Use the text's random number sequence to generate X.

Answer

(What is the text's random number sequence?)
The method used will the composition approach.

Derivation

The composition approach works by finding α, $p_j^{(1)}$, and $p_j^{(2)}$ so that

$$P\{X = j\} = \alpha p_j^{(1)} + (1 - \alpha) p_j^{(2)}$$

Let $\alpha = 0.55$ and let

$$p_j^{(1)} = P\{X = j\} = \begin{cases} 0 & \text{for } x \text{ } \textit{even} \\ 0.2 & \text{for } x \text{ } \textit{odd} \end{cases}$$

and

$$p_j^{(2)} = P\{X = j\} = \begin{cases} 0 & \text{for } x \text{ } \textit{odd} \\ 0.2 & \text{for } x \text{ } \textit{even} \end{cases}$$

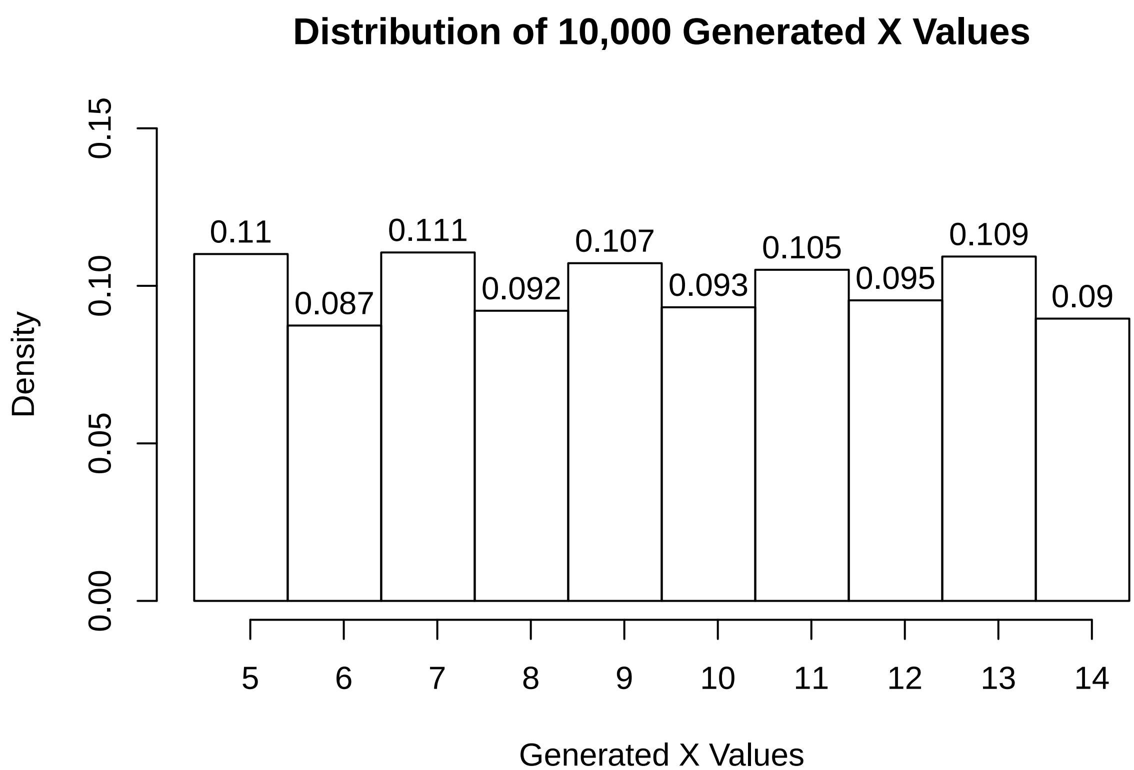
In other words, if j_0 is odd, then $P(X = j_0) = (0.55)(0.20) + (0.45)(0) = 0.11$ and if j_0 is even then $P(X = j_0) = (0.55)(0) + (0.45)(0.20) = 0.09$. This is the desired outcome.

Algorithm

1. Generate u_1 from standard uniform distribution
2. Generate u_2 from standard uniform distribution
3. If $u_1 < \alpha$, (odd case) set $x = 2(floor(5u_2)) + 5$
4. Otherwise (even case) set $x = 2(floor(5u_2)) + 6$
5. Go back to step 1, repeat n times

Program

```
# Initialize variables, generate uniform random numbers
N <- 10^4
alpha <- 0.55
uniform1 <- runif(N)
uniform2 <- runif(N)
x_dist <- c(rep(0, N))
# These two lines generate the x values
x_dist[uniform1 < 0.55] <- 2*floor(5*uniform2[uniform1 < 0.55])+5
x_dist[x_dist == 0] <- 2*floor(5*uniform2[uniform1 >= 0.55])+6
```



Problem 5.10) A casualty insurance company has 1000 policyholders, each of whom will independently present a claim in the next month with probability 0.05. Assuming that the amounts of the claims made are independent exponential random variables with mean $800, use simulation to estimate the probability that the sum of these claims exceeds $50,000.

Answer

Derivation

In order to generate random values from an exponential distribution with mean 800, we'll use the fact that the CDF of this distribution is

$$F(x) = 1 - e^{-x/800}$$

Given that $F(x)$ has a standard uniform distribution, then:

$$
\begin{aligned}
u &= 1 - e^{-x/800} \\
e^{-x/800} &= 1 - u \\
\frac{-x}{800} &= ln(1-u) \\
x &= (-800)ln(u)
\end{aligned}
$$

Note that the random variables U and $1 - U$ have the same distribution if U is standard uniform.

Also, since each policyholder has a 5% chance of making a claim, then the number of claims made out of 1000 has a binomial distribution with $n = 1000$ and $p = 0.05$. The random variable we are interested in is then

$$S = X_1 + X_2 + X_3 + ... + X_N$$

where the X_i are *iid* exponential with mean 800 and N is binomial with $n = 1000$ and $p = 0.05$. For simplicity in the program, values from the binomial distribution will be done using an R function, but the exponential values will be generated using values from a uniform distribution according to the derivation above.

Algorithm

- For each trial:
  - Generate a value n from Binomial(1000, 0.05)
  - Generate n values $u_1, u_2, ..., u_n$ from U(0,1)
  - Calculate each of the n claim amounts x by evaluating $x_i = (-800)ln(u_i)$
  - Find the sum of the n claims for that trial
- After the sum of each trial is calculated, then determine how many are greater than $50,000

The R code appears different than this algorithm, but in practice this is what is happening. The R language benefits from avoiding explicitly writing *for* loops, and instead using other methods like *apply*, *lapply*, and *sapply*.

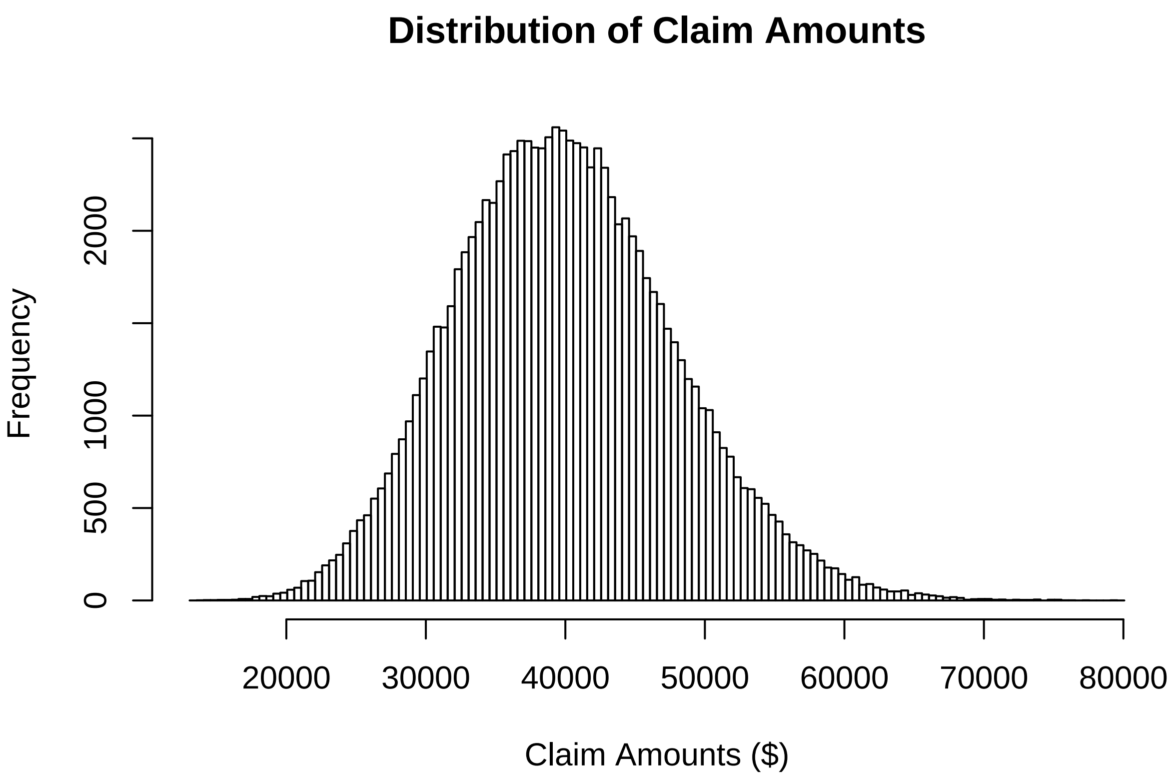
Program

The program uses the *generate_claims()* function to generate n claims from an exponential distribution, where n has a binomial distribution. 100,000 trials are run, and then the sum of the claims from each trial is calculated.

```r
# Set seed, set N = number of simulations
set.seed(1234)
N = 10^5
# Returns variable number claims from exponential distribution
generate_claims <- function(vector, holders = 1000, chance = 0.05, beta = 800){
  # Number of claims that will occur is binomial(holders, chance)
  num_of_claims <- rbinom(1, holders, chance)
  # Generate claim amounts from exponential distribution for each claimant
  claims <- -beta * log(runif(num_of_claims))
  # Return vector of claims
  return(claims)
}
```

```r
# Initialize claims list
claims <- rep(list(rep(0,100)), N)
# Simulate N times
claims <- lapply(claims, generate_claims)
# Find the sum for each trial
distribution <- sapply(claims, sum)
```

Below is a histogram of the results. The distribution is a sum of exponential random variables and thus is a continuous distribution. It appears to be slightly right tailed with a mean around $40,000.



The following R command returns the estimated probability that the sum of the claims is above $50,000:

```r
mean(distribution>50000)
```

```
## [1] 0.10661
```

Out of 100,000 trials, approximately 0.10661 trials resulted in an outcome where the total value of the claims exceeded $50,000.

Analytical Result

find out how to do this analytically