Tyler Grimes
Feb. 2, 2016
Homework 4

Ch. 4 #3, 14a, and 15; Ch. 5 #10.

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

$$P(X = 1) = 0.3, P(X = 2) = 0.2, P(X = 3) = 0.35, P(X = 4) = 0.15$$

Derivation

We will use a slight variation of the discrete inverse transform method to generate X. Rather than use the cdf, we will organize the values of X from most probable to least. The probabilities are accumulated in this order, creating the function

$$F(x) = \begin{cases} 0.35 & , x = 3 \\ 0.65 & , x = 1 \\ 0.85 & , x = 2 \\ 1 & , x = 4 \end{cases}.$$

The notation $F(x)$ is used for this function, but note it is not a cdf. However, by ordering from largest probabilities to smallest, we will need to perform less comparisons (on average) when running the algorithm.

Algorithm

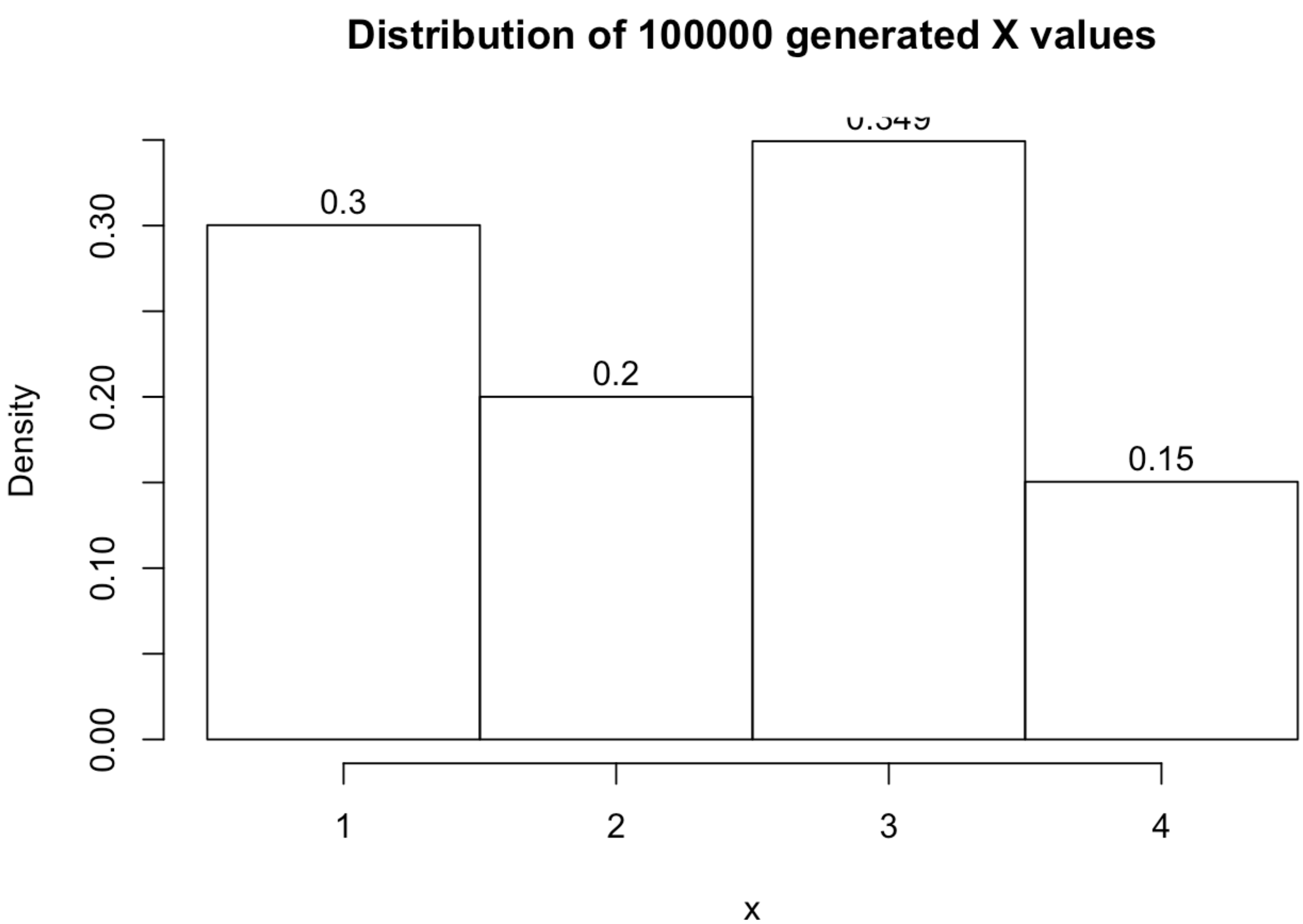
To generate a variable from the distribution,

1. Generate $u \sim Unif(0, 1)$.
2. If $u < 0.35$ set $X = 3$
   Else if $u < 0.65$ set $X = 1$
   Else if $u < 0.85$ set $X = 2$
   Else set $X = 4$

Simulation

```
set.seed(12)
rX <- function() {
  u <- runif(1, 0, 1)
  if(u < 0.35) return(3)
  if(u < 0.65) return(1)
  if(u < 0.85) return(2)
  return(4)
}

x <- numeric(100000)
for(i in 1:100000) { x[i] <- rX() }
hist(x, breaks = c(0.5, 1.5, 2.5, 3.5, 4.5), labels = T, freq = F,
     main = "Distribution of 100000 generated X values")
```



14.) 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.

(a) Give the inverse transform method for generating Y.

Derivation

Let Y_k denote the random variable with the pmf

$$P(Y_k = y) = \frac{P(X = y|y \geq k)}{P(X \geq k)} = \begin{cases} \frac{1}{\alpha}\binom{n}{y}p^y(1-p)^{n-y} & , y = k, k+1, \ldots n \\ 0 & , \text{otherwise} \end{cases}.$$

The cdf of Y_k is

$$P(Y_k \leq y) = \frac{1}{\alpha}\sum_{x=k}^{y}\binom{n}{x}p^x(1-p)^{n-x} = F_k(y)$$

Algorithm

We assume p, n, k, and α are provided. To generate a variable from the distribution,

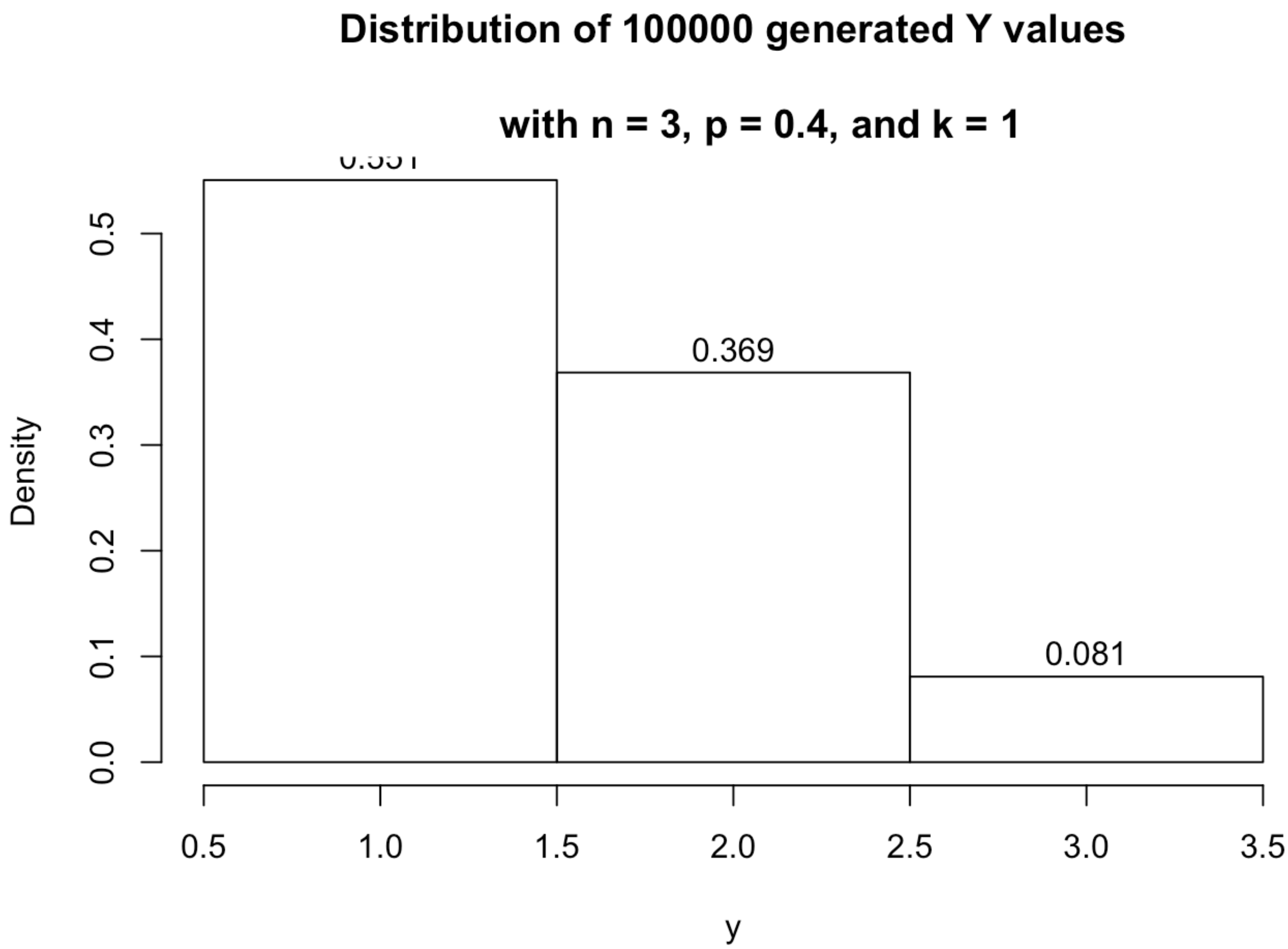
1. Generate $u \sim Unif(0, 1)$.
2. For y from k to n
   If $F_k(y-1) < y < F_k(y)$, set Y = y.

Simulation

```
#Cdf of Y.
FY <- function(y, n, p, k) {
  alpha <- 1 - pbinom((k-1), n, p)
  #sum(dbinom(seq(k, y, 1), n, p))/alpha
  (pbinom(y, n, p) - pbinom((k-1), n, p))/alpha
}

#Generate random y.
rY <- function(n, p, k) {
  u <- runif(1, 0, 1)
  upperBounds <- numeric(n - k + 1)
  for(j in k:n) {
    upperBounds[j - k + 1] <- FY(j, n, p, k)
  }
  y <- k + sum(u > upperBounds)
  y
}

n = 3
p = 0.4
k = 1
y <- numeric(100000)
for(i in 1:100000) {
  y[i] <- rY(n, p, k)
}
hist(y, seq(k - 0.5, n + 0.5, 1), labels = T, freq = F,
     main = "Distribution of 100000 generated Y values\n
     with n = 3, p = 0.4, and k = 1")
```



Analytical

In the simulation we used $n = 3, p = 0.4$, and $k = 1$. For this distribution, $\alpha = P(Y \geq 1) = 1 - P(Y \leq 0) = 1 - (1 - 0.4)^3 = 0.784$. The distribution of Y_k is

$$P(Y = y) = \begin{cases} \frac{1}{0.784}\binom{3}{1}(0.4)^1(1-0.4)^{3-1} \approx 0.55102 & , y = 1 \\ \frac{1}{0.784}\binom{3}{2}(0.4)^2(1-0.4)^{3-2} \approx 0.36735 & , y = 2 \\ \frac{1}{0.784}\binom{3}{3}(0.4)^3(1-0.4)^{3-3} \approx 0.08163 & , y = 3 \\ 0 & , \text{otherwise} \end{cases}.$$

The distribution of the simulated sample agrees with the expected distribution to two decimal places.

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

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

Use the text's random number sequence to generate X

Derivation

Using the composition method, we can break up the pmf into two pieces

$$p_j = \frac{11}{20} p_{1j} + \frac{9}{20} p_{2j},$$

where $p_{1j} = 1/5$ for $j = 5, 7, \ldots 13, 0$ elsewhere, and $p_{2j} = 1/5$ for $j = 6, 8, \ldots 14, 0$ elsewhere. We can easily verify that these are valid pmfs; they are nonnegative, and both sum to 1 over their support.

Algorithm

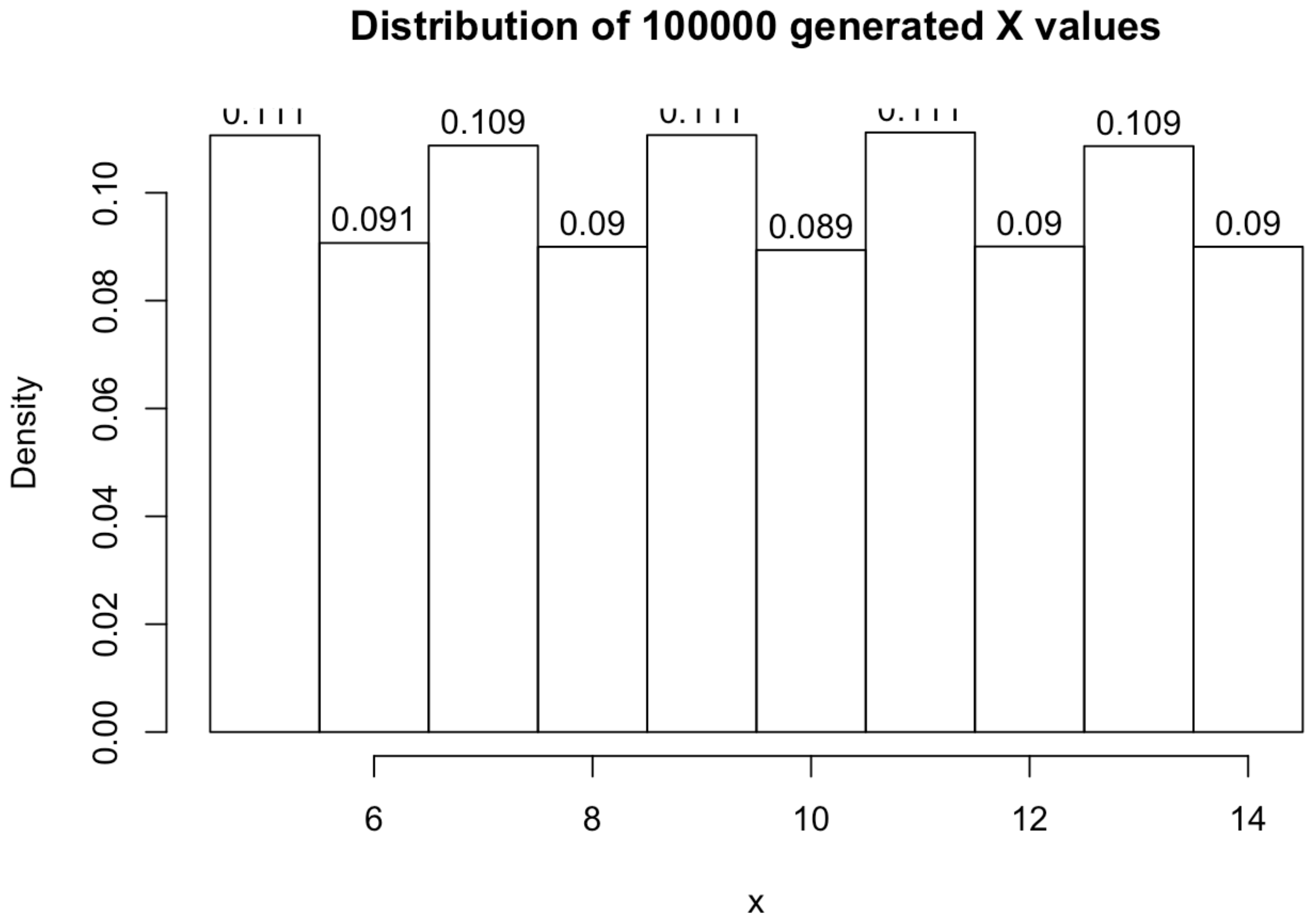
To generate a variable from the distribution,

1. Generate $u_1, u_2 \sim Unif(0, 1)$
2. If $u_1 < \frac{11}{20}$, $x = 2\lceil 5u_2 \rceil + 3$
   Else $x = 2\lceil 5u_2 \rceil + 4$

Simulation

```
rX <- function(n = 1) {
  u1 <- runif(n, 0, 1)
  u2 <- runif(n, 0, 1)
  x <- numeric(n)
  cond1 <- u1 < 11/20
  x[cond1] <- 2*ceiling(5*u2[cond1]) + 3
  x[!cond1] <- 2*ceiling(5*u2[!cond1]) + 4
  x
}

x <- rX(100000)
hist(x, breaks = c(seq(4.5, 14.5, 1)), labels = T, freq = F,
     main = "Distribution of 100000 generated X values")
```


Distribution of 100000 generated X values
0.111
0.091
0.109
0.09
0.111
0.089
0.111
0.09
0.109
0.09
Density
0.00
0.02
0.04
0.06
0.08
0.10
6
8
10
12
14
x

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 $\$50,000$.

Derivation

To generate an obsevation from this population of claims, we need to first generate a number k from a Binomial($1000, 0.05$) distribution. This will be the number of claims made in the month. Then, we generate k numbers, $x_1, \ldots, x_k$ from an Exponential($\lambda = \frac{1}{800}$) and take their sum, S.

Algorithm

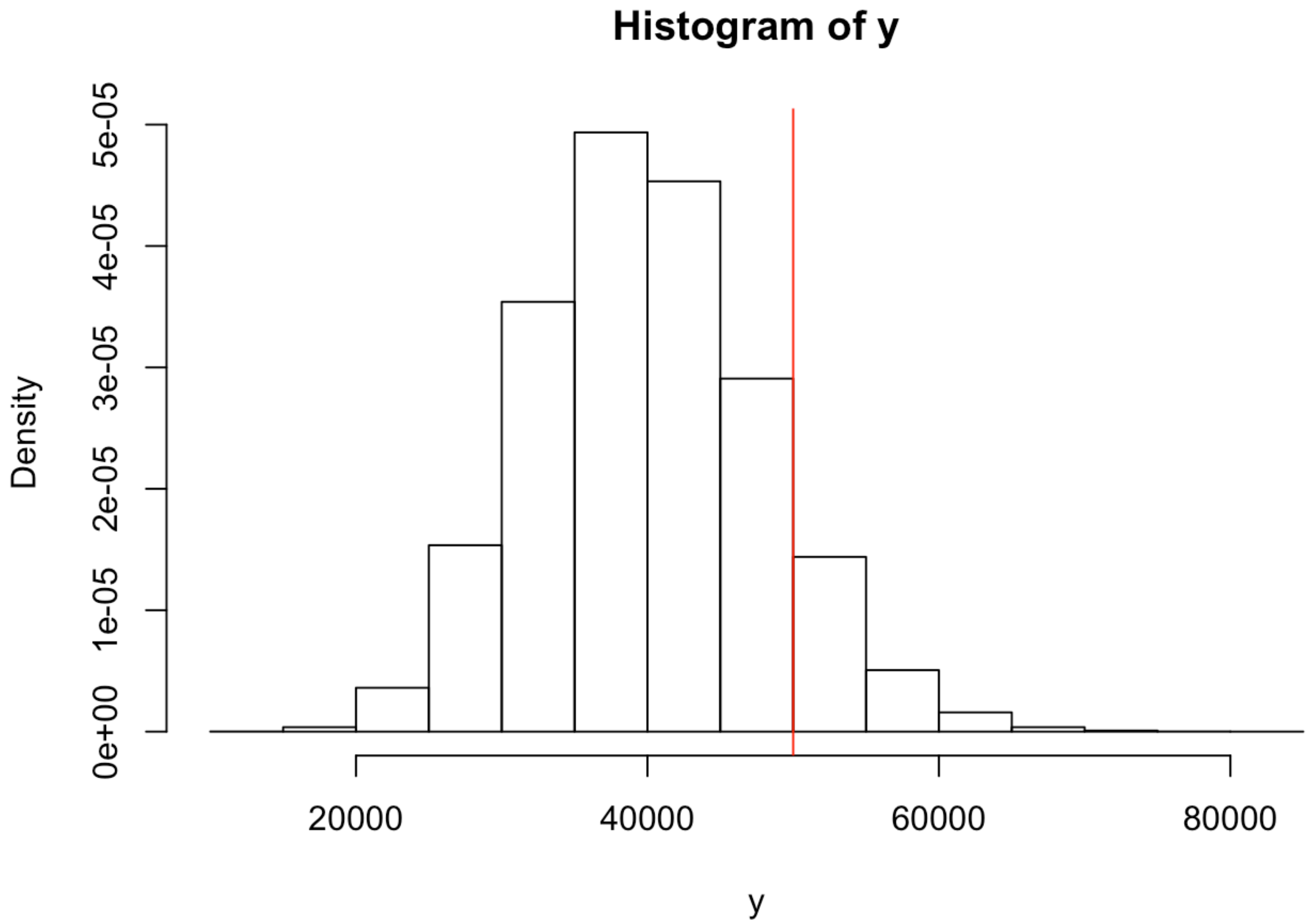
Assuming we have a way to generate random numbers from an exponential and binomial, to generate a random S,

1. Generate a random $k \sim Bin(1000, 0.05$.
2. Generate $x_1, \ldots, x_k \sim \text{iid } Exp(1/800)$.
3. Evaluate $S = \sum x_i$.

Simulation

```
rep <- 100000
p <- 0.05
n <- 1000
mean <- 800
y <- numeric(rep)
for(i in 1:rep) {
  k <- rbinom(1, n, p)
  if(k == 0) {
    y[i] = 0
  }
  else {
    y[i] <- sum(rexp(k, 1/800))
  }
}

hist(y, freq = F)
abline(v = 50000, col = "red", main = "Distirbution of 100000 monthly total claims")
```




```
mean(y > 50000)
```

```
## [1] 0.1075
```

Analytical

Let $K \sim binom(n, p)$, $x_i \sim$ iid $exp(\lambda)$, and $S_k = \sum_1^k x_i$. But then, since S_k is a sum of independent exponential random variables, its distribution is Gamma($\alpha = k, \lambda$). Now consider the joint pdf of S_k and K,

$$\begin{aligned} P(S_k = s, K = k) &= P(S_k = s|K = k)P(K = k) \\ &= \frac{1}{\lambda^k \Gamma(k)} s^{k-1} e^{\frac{-s}{\lambda}} \binom{n}{k} (p)^k (1-p)^{n-k}, \text{for k} = 0,\ 1,\ ...,\ \text{n}. \end{aligned}$$

Since our main interest is on $P(S_k > s)$, we need the marginal distirbution of S_k. Since k is an integer, we can obtain the conditional cdf of S_k by using integration by parts k times:

$$P(S_k \leq s, |K = k) = \int_0^s \frac{1}{\lambda^k \Gamma(k)} x^{k-1} e^{\frac{-x}{\lambda}} dx$$
$$= 1 - \sum_{n=0}^{(} k-1) \frac{e^{\frac{-s}{\lambda}}}{n!} \left(\frac{s}{\lambda}\right)^n .$$

Now to obtain $P(S_k > s)$, we sum the joint distribution over the support of K

$$P(S_k > s) = \sum_{k=0}^{n} P(S_k > s, K = k)$$
$$= \sum_{k=0}^{n} \sum_{m=0}^{(} k-1) \left[\frac{e^{\frac{-s}{\lambda}}}{m!} \left(\frac{s}{\lambda}\right)^m \right] \binom{n}{k} (p)^k (1-p)^{n-k}$$

Without any obvious ways to simplify this expression, we will leave the expression here. If we were to perform this calculation using $n = 1000$ we would need to compute $1000!$. So instead, we will use an approximation and only sum from $n = 0$ to 90. While this is only a portion of the entire support of k, the probability that it is outside of this range is less than 10^-7.

$$P(S_k > 50000) \approx \sum_{k=0}^{90} \sum_{m=0}^{(} k-1) \left[\frac{e^{\frac{-50000}{800}}}{m!} \left(\frac{50000}{800}\right)^m \right] \binom{1000}{k} (0.05)^k (1-0.05)^{1000-k}$$

```
s <- 50000
n <- 1000
p <- 0.05
lambda <- 800
pSk <- function(s, n, p, lambda) {
  approx <- 90
  terms <- numeric(approx)
  for(i in 1:approx) {
    terms[i] <- sum(exp(-s/lambda)/factorial(0:(i-1))*(s/lambda)^(0:(i-1)))*dbinom(i,
n, p)
  }
  sum(terms)
}

pSk(s, n, p, lambda)
```

```
## [1] 0.1070977
```

This approximation and our simulated approximatoin agree to three decimal places.