Distributions Related to Bernoulli Trials

Chapter 3, Lab 3: Solutions

OpenIntro Biostatistics

Topics

- Geometric distribution
- Negative binomial distribution
- Hypergeometric distribution

This lab discusses other distributions derived from a series of repeated Bernoulli trials: the geometric, negative binomial, and hypergeometric distributions. The simulations in this lab require additional control structures: the `repeat` and `while` loops.

The material in this lab corresponds to Section 3.5 of *OpenIntro Biostatistics*.

Geometric distribution

Let X represent the number of trials required to obtain a success in a series of independent Bernoulli trials, including the (last) successful trial. If the probability of success in one trial is p and the probability of failure is $1-p$, then the probability of obtaining the first success in the k^{th} trial is given by

$$P(X = k) = (1-p)^{k-1}p.$$

For a geometric random variable, $E(X) = \frac{1}{p}$ and $Var(X) = \frac{1-p}{p^2}$.

The other convention for defining the geometric distribution models the number of trials required to obtain the first success; i.e., the number of failures until the first success. Formally, let Y be the number of failures until the first success. $Y = X - 1$, where Y can take on values $\{0, 1, 2, \dots\}$ and X can take on values $\{1, 2, 3, \dots\}$.[1]

The text and the labs use the former convention, while R calculates geometric probabilities using the latter definition. so care should be taken when using `dgeom()` and `pgeom()`.

The following code shows how to calculate $P(X = 5)$, $P(X \leq 5)$, and $P(X > 5)$ for $X \sim \text{Geom}(0.35)$, where X includes the successful trial.

```
#probability X equals 5
dgeom(5 - 1, 0.35)
```

```
## [1] 0.06247719
```

```
#probability X is less than or equal to 5
pgeom(5 - 1, 0.35)
```

```
## [1] 0.8839709
```

[1] For reference, $E(Y) = \frac{1-p}{p}$ and $Var(Y) = \frac{1-p}{p^2}$.

```r
#probability X is greater than 5
pgeom(5 - 1, 0.35, lower.tail = FALSE)
```

```
## [1] 0.1160291
```

The following code shows how to simulate a geometric random variable for a given success probability p and number of replicates.

```r
# GEOMETRIC RANDOM VARIABLE SIMULATION #

#define parameters
p = 0.35
replicates = 10

#create empty vector to store results
trials.for.first.success = vector("numeric", replicates)

#set the seed for pseudo-random sampling
set.seed(2018)

#simulate the experiment
for(k in 1:replicates){

  trial.number = 1

  repeat {

    outcome.individual = sample(c(0,1), size = 1,
                                prob = c(1 - p, p), replace = TRUE)

    if(outcome.individual == 0){

      trial.number = trial.number + 1

    } else {

      trials.for.first.success[k] = trial.number

      break
    }

  }
}
```

1. This question uses the scenario in Example 3.38 of the text. In the Milgram shock experiments, the probability of a person refusing to give the most severe shock is $p = 0.35$. Suppose that participants are tested one at a time until one person refuses.

   Run a simulation with 10 replicates, with seed set to 2018.

There are several layers to the simulation code. The outer layer with the `for` loop replicates the experiment, just as in simulations shown in previous labs. The sampling is nested within a `repeat` loop, which will run indefinitely until it encounters the `break` statement. The `repeat` loop contains an `if-else` statement that contains the `break` statement.

a) Explain the code used to generate `outcome.individual`.

The variable `outcome.individual` contains either a `0` or `1`, sampled with probabilities $1 - p$ and p, respectively. A `1` represents a successful trial and a `0` represents an unsuccessful trial.

b) The `if-else` statement is closely related to the `if` statement and has the basic structure `if( condition ) { statement 1 } else { statement 2 }`. If the condition is satisfied, the first condition is carried out; else, the second statement is carried out.

Examine the condition in the `if-else` statement and explain how the condition specifies when statement 1 should be carried out versus statement 2. In the context of the experiment, is statement 1 or statement 2 carried out when a success occurs?

The condition specifies that when `outcome.individual` is `0`, statement 1 should be carried out; i.e., when a failure occurs, statement 1 is carried out. Thus, when a success occurs, statement 2 is executed.

c) The variable `trial.number` keeps track of how many trials have occurred in the current replicate. The variable is set at `1` at the beginning of the loop.

Why is `trial.number` only recorded in `trials.for.first.success` (at the k^{th} position) when `outcome.individual` is not `0`?

If `outcome.individual` is `0`, then a success has not yet occurred in the series of trials (for a particular replicate). It is only when the success has occurred that `trial.number` reflects the number of trials required to observe a first success.

d) Why is a `repeat` loop necessary to model the geometric distribution but not the binomial distribution?

The number of trials is not fixed with the geometric distribution, which requires the simulation to continue indefinitely until a condition is met. In binomial sampling, the number of trials is fixed.

e) In which of these 10 replicates did the first participant to refuse to administer the most severe shock?

In replicates 3, 4, 5, and 7, the first participant refused to administer the most severe shock.

```
which(trials.for.first.success == 1)
```

```
## [1] 3 4 5 7
```

f) Set the number of replicates to 10,000 and re-run the simulation. What is the estimated probability that the first success occurs within the first four participants tested?

The estimated probability that the first success occurs within the first four participants is 0.817.

```
table(trials.for.first.success)
```

```
## trials.for.first.success
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15
## 3438 2302 1502  925  613  417  291  176  128   73   51   27   20   11   10
##   16   17   19   20   21   23
##    9    3    1    1    1    1
```

```
#estimate probability
sum(trials.for.first.success <= 4)/replicates
```

```
## [1] 0.8167
```

g) Use pgeom() to calculate the probability estimated in part f).

The probability of the first success occuring within the first four participants tested is 0.821.

```
pgeom(4 - 1, 0.35)
```

```
## [1] 0.8214937
```

2. When patients receive blood transfusions, it is critical that the blood type of the donor is compatible with the patients, or else an immune system response will be triggered. For example, a patient with Type O- blood can only receive Type O- blood, but a patient with Type O+ blood can receive either TYpe O+ blood or Type O-. Furthermore, if a blood donor and recipient are of the same ethnic background, the chance of an adverse reaction may be reduced.

   According to data collected from blood donors, 0.37 of white, non-Hispanic donors are Type O+ and 0.08 are Type O-. For the following questions, assume that only white, non-Hispanic donors are being tested.

   a) On average, how many donors would need to be randomly sampled for a Type O+ donor to be identified? With what standard deviation?

On average, 2.70 (about 3) donors would need to be randomly sampled for a Type O+ donor to be identified, with standard deviation 2.15.

```
p = 0.37

mean = 1/p; mean
```

```
## [1] 2.702703
```

```
variance = (1-p)/p^2
sd = sqrt(variance); sd
```

```
## [1] 2.145204
```

b) What is the probability that 4 donors must be sampled to identify a Type O+ donor?

The probability that 4 donors must be sampled to identify a Type O+ donor is 0.093.

```
dgeom(4 - 1, 0.37)
```

```
## [1] 0.09251739
```

c) What is the probability that more than 4 donors must be sampled to identify a Type O+ donor?

The probability that more than 4 donors must be sampled to identify a Type O+ donor is 0.158.

```
pgeom(4 - 1, 0.37, lower.tail = FALSE)
```

```
## [1] 0.1575296
```

d) What is the probability of the first Type O- donor being found within the first 4 people?

The probability of the first Type O- donor being found within the first 4 people is 0.284.

```
pgeom(4 - 1, 0.08)
```

```
## [1] 0.283607
```

e) What is the probability that fewer than 4 donors must be tested before a Type O- donor is found?

The probability that fewer than 4 donors must be tested before a Type O- donor is found is 0.221.

```
pgeom(3 - 1, 0.08)
```

```
## [1] 0.221312
```

Negative binomial distribution

The negative binomial distribution is a generalization of the geometric distribution. While the geometric distribution describes the probability of observing the first success on the k^{th} trial, the negative binomial describes the probability of observing the r^{th} success on the k^{th} trial.

Let X represent the number of trials required to obtain r successes in a series of independent Bernoulli trials, including the last (successful) trial. If the probability of success in one trial is p and the probability of failure is $1-p$, then the probability of obtaining the r^{th} success in the k^{th} trial is given by

$$P(X=k) = \binom{k-1}{r-1} p^r (1-p)^{k-r}.$$

For a negative binomial random variable, $E(X) = \frac{r}{p}$ and $Var(X) = \frac{r(1-p)}{p^2}$.

Just as with the geometric distribution, the negative binomial is alternatively defined as the number of failures before the r^{th} success. Formally, let Y be the number of failures until the first success. $Y = X - r$, where Y can take on values $\{0, 1, 2, \dots\}$ and X can take on values $\{r, r+1, r+2, \dots\}$.

R calculates negative binomial probabilities using the latter definition. The following code shows how to calculate $P(X = 5)$, $P(X \leq 5)$, and $P(X > 5)$ for $X \sim \text{NB}(4, 0.8)$.

```
#probability X equals 5
dnbinom(5 - 4, 4, 0.8)
```

```
## [1] 0.32768
```

```
#probability X is less than or equal to 5
pnbinom(5 - 4, 4, 0.8)
```

```
## [1] 0.73728
```

```
#probability X is greater than 5
pnbinom(5 - 4, 4, 0.8, lower.tail = FALSE)
```

```
## [1] 0.26272
```

The following code shows how to simulate a negative binomial random variable for a given success probability p, number of successes r, and number of replicates.

```
# NEGATIVE BINOMIAL RANDOM VARIABLE SIMULATION#

#define parameters
p = 0.35
r = 4
replicates = 20

#create empty vector to store results
trials.for.rth.success = vector("numeric", replicates)

#set the seed for pseudo-random sampling
set.seed(2018)

#simulate the experiment
for (k in 1:replicates){

  trial.number = 1
  successes = 0

  while(successes < r){
    outcome.individual = sample(c(0,1), size = 1,
                                prob = c(1 - p, p), replace = TRUE)

    if(outcome.individual == 1){

      successes = successes + 1

      }

    trial.number = trial.number + 1

    }

  trials.for.rth.success[k] = trial.number - 1

}

#view the results
trials.for.rth.success
table(trials.for.rth.success)
```

3. In the Milgram shock experiments, what is the probability that five participants must be tested in order to observe four participants who refuse to administer the most severe shock? In other words, what is the probability of a fourth success on the fifth trial?

   Run a simulation with 20 replicates, with seed set to `2018`.

   This simulation nests the sampling within a `while` loop, which will run as long as the specified condition is no longer satisfied. The `while` loop contains an `if` statement that keeps track of the number of successes that have been observed.

   a) The variable successes keeps track of how many successes have occurred in the current replicate. The variable is set to `0` at the beginning of the loop.

      Explain what happens when the condition of the `if` statement is satisfied.

   When the condition of the `if` statement is satisfied, then `1` is added to sucesses.

   b) Is a `1` added to `trial.number` only when a success occurs?

   No, the line adding `1` to `trial.number` is outside of the `if` statement and will occur regardless of whether a success occurs.

   c) What is the condition required for the `while` loop to continue running?

   The `while` loop runs as long as successes is less than `r`, where `r` is specified at the beginning of the simulation.

   d) Once a value can be recorded in `trials.for.rth.success` for the k^{th} replicate of the experiment, why is it necessary to subtract `1` from `trial.number`? (*Hint*: It may help to think about the answer to parts b) and c).)

   A `1` is added to `trial.number` each time the loop runs, just before the loop ends, in anticipation of the loop running again. On the last run of the loop, however, right before successes equals `r`, there will be not be another trial. It is necessary to subtract `1` before recording the number of trials to reflect that the loop stops at that iteration.

   e) In the first of these 20 replicates, how many trials did it take to observe four successes?

   In the first of these 20 replicates, it took 14 trials to observe four successes.

```
trials.for.rth.success[1]
```

```
## [1] 14
```

   f) Set the number of replicates to 1,000 and re-run the simulation. What is the estimated probability that five participants must be tested in order to observe four who refuse to administer the most severe shock?

   The estimated probability that five participants must be tested in order to observe four who refuse to administer the most severe shock is 0.034.

```
table(trials.for.rth.success)
```

```
## trials.for.rth.success
##   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21
##   8  34  66  71  81 103  89  83  91  79  65  51  38  28  27  19  15  11
##  22  23  24  25  26  27  29  32  34
```

```
##  14   6   4   7   2   4   2   1   1
```

```
sum(trials.for.rth.success == 5)/replicates
```

```
## [1] 0.034
```

g) Use pnbinom() to calculate the probability estimated in part f).

The probability estimated in part f) is 0.054.

```
pnbinom(5 - 4, 4, 0.35)
```

```
## [1] 0.0540225
```

4. Cilantro leaves are widely used in many world cuisines. While some people enjoy it, others claim that it has a soapy, pungent aroma. A recent study conducted on participants of European ancestry identified a genetic variant that is associated with soapy-taste detection. In the initial questionnaire, 1,994 respondents out of 14,604 reported that they thought cilantro tasted like soap. Suppose that participants are randomly selected one by one.

   a) Find the probability that the first soapy-taste detector is the fourth person selected.

The probability that the first soapy-taste detector is the fourth person selected is 0.0879. This can be calculated using either the negative binomial distribution or the geometric, since the negative binomial simplifies to the geometric for $r = 1$.

```
dnbinom(4 - 1, 1, 1994/14604)
```

```
## [1] 0.08789883
```

```
dgeom(4 - 1, 1994/14604)
```

```
## [1] 0.08789883
```

   b) Find the probability that the second soapy-taste detector is identified before ten people are tested.

The probability that the second soapy-taste detector is identified before ten people are tested is 0.405.

```
pnbinom(10 - 2, 2, 1994/14604)
```

```
## [1] 0.4053403
```

   c) Find the probability that no more than two people out of ten are soapy-taste detectors.

The probability that no more than two people out of ten are soapy-taste detectors is 0.854. Note that this is a binomial probability, since the number of trials is fixed.

```
pbinom(2, 10, 1994/14604)
```

```
## [1] 0.8538778
```

   d) What is the probability that two soapy-taste detectors are identified from testing ten people?

The probability that two soapy-taste detectors are identified from testing ten people is 0.259. This is also a binomial probability.

```r
dbinom(2, 10, 1994/14604)
```

```
## [1] 0.2592181
```

e) Calculate the mean and standard deviation of the number of people that must be tested if the goal is to identify two soapy-taste detectors.

The mean and standard deviation of a negative binomial random variable with $r = 2$ and $p = 1994/14604$ are 14.65 and 9.62, respectively.

```r
r = 2
p = 1994/14604

mean = r/p; mean
```

```
## [1] 14.64794
```

```r
variance = (r*(1-p))/p^2
sd = sqrt(variance); sd
```

```
## [1] 9.624614
```

Hypergeometric distribution

Let X represent the number of successes in a series of repeated Bernoulli trials, where sampling is done without replacement. Suppose that in the population of size N, there are m total successes. The probability of observing exactly k successses in a sample of size n (i.e., n dependent trials) is given by

$$P(X = k) = \frac{\binom{m}{k}\binom{N-m}{n-k}}{\binom{N}{n}}.$$

For a hypergeometric random variable, $E(X) = np$ and $Var(X) = np(1-p)\frac{N-n}{N-1}$, where $p = m/N$.

Hypergeometric probabilities are calculated in R with the use of `dhyper()` and `phyper()`. The following code shows how to calculate $P(X = 5)$, $P(X \leq 5)$, and $P(X > 5)$ for $X \sim \text{HGeom}(10, 15, 8)$, where $m = 10$, $N - m = 15$, and $n = 8$.

```
#probability X equals 5
dhyper(5, 10, 15, 8)
```

```
## [1] 0.1060121
```

```
#probability X is less than or equal to 5
phyper(5, 10, 15, 8)
```

```
## [1] 0.9779072
```

```
#probability X is greater than 5
phyper(5, 10, 15, 8, lower.tail = FALSE)
```

```
## [1] 0.02209278
```

The following code shows how to simulate a hypergeometric random variable for a given population size N, total number of successes m, and sample size n.

```
# HYPERGEOMETRIC RANDOM VARIABLE SIMULATION #

#define parameters
N = 450
m = 100
n = 50
replicates = 1000

#create empty vectors to store results
outcome.replicate = vector("numeric", n)
outcomes = vector("numeric", replicates)

#set the seed for pseudo-random sampling
set.seed(2018)

#simulate the experiment
for(k in 1:replicates){

  outcome.replicate = sample(rep(c(0,1), c(N - m, m)), size = n,
```

```
                              replace = FALSE)

  outcomes[k] = sum(outcome.replicate)

}
```

5. Section 3.5.3 in *OI Biostat* introduces the capture-recapture method as an example of the hypergeometric distribution. There are N deer in a population, with m deer that are initially captured and marked. The hypergeometric distribution models the probability of observing k marked deer (i.e., recaptured deer) in a sample of size n .

   Suppose there are 450 deer in a population. Last year, 100 deer were captured and marked.

   a) Use simulation to estimate the probability of observing more than 15 marked deer in a sample of size 50.

   The estimated probability of observing more than 15 marked deer in a sample of size 50 is 0.065.

```
table(outcomes)
```

```
## outcomes
##   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
##   1   3   1  11  20  48  99 112 126 125 136 116  75  62  37  19   4   4
##  20
##   1
```

```
sum(outcomes > 15)/replicates
```

```
## [1] 0.065
```

   b) Examine the code used to generate `outcome.replicate`. How does this differ from the code used to simulate binomial sampling?

   The `sample()` command is run with `replace = FALSE`. The sample is drawn from a vector of several `0`'s and `1`'s, where there are $N - m$ `0`'s and m `1`'s. It is not necessary to specify a probability vector, since by default R will draw from the remaining elements with equal probability.

   c) Use simulation to estimate the probability of observing more than 15 marked deer in a sample of size 50 if the deer are sampled with replacement.

   The estimated probability of observing more than 15 marked deer in a sample of size 50 if the deer are sampled with replacement is 0.073.

```
#define parameters
N = 450
m = 100
n = 50
p = m/N
replicates = 1000

#create empty vectors to store results
```

```
outcome.replicate = vector("numeric", n)
outcomes = vector("numeric", replicates)

#set the seed for pseudo-random sampling
set.seed(2018)

#simulate the experiment
for(k in 1:replicates){

  outcome.replicate = sample(c(0,1), prob = c(1 - p, p), size = n,
                             replace = TRUE)

  outcomes[k] = sum(outcome.replicate)

}

#estimate probability
sum(outcomes > 15)/replicates
```

```
## [1] 0.073
```

6. Recall that in a questionnaire, 1,994 respondents out of 14,604 reported that they thought cilantro tasted like soap. Suppose a random sample of 150 individuals are selected for further study.

   a) What is the mean and variance of the number of people sampled that are soapy-taste detectors?

The mean and variance of the number of people sampled that are soapy-taste detectors are 20.48 and 17.50, respectively.

```
N = 14604
n = 150
p = 1994/14604

mean = n*p; mean
```

```
## [1] 20.48069
```

```
variance = n*p*(1 - p)*((N - n)/(N - 1)); variance
```

```
## [1] 17.50386
```

   b) Calculate the probability that 20 of the people sampled are soapy-taste detectors.

The probability that 20 of the people sampled are soapy-taste detectors is 0.0953.

```
dhyper(20, 1994, 14604-1994, 150)
```

```
## [1] 0.09526515
```

   c) Calculate the probability that 20 or more of the people sampled are soapy-taste detectors.

The probability that 20 or more of the people sampled are soapy-taste detectors is 0.582.

```
phyper(19, 1994, 14604-1994, 150, lower.tail = FALSE)
```

```
## [1] 0.5820395
```

d) Suppose that the 150 individuals were sampled with replacement. What is the probability of selecting 20 soapy-taste detectors?

The probability of selecting 20 soapy-taste detectors is 0.0948.

```
dbinom(20, 150, 1994/14604)
```

```
## [1] 0.09479083
```

e) Compare the answers from part b) and part d). Explain why the answers are essentially identical.

With a large sample size, sampling with replacement is highly unlikely to result in any particular individual being sampled again. In this case, the hypergeometric and binomial distributions will produce equal probabilities.