Lab 6 – Binomial Distribution and Distribution of Sample Means

This lab contains some numbered questions. You are required to submit:

1. One pdf document (Lab 6.pdf) that contains:
   - your written answers to the question and any charts produced
   - the commands you used to answer the question, when asked

2. One R script (Lab 6.R) that contains all the code used to complete the lab. The script must run without errors.

Submit your files to the Lab 6 folder by 11:59pm <u>two</u> school days from now.

Packages required: none

Lab Objectives

- calculate probabilities associated with a *binomial* random variable X
- investigate the *distribution of sample proportion* $\hat{p}$
- investigate the *distribution of sample means* $\bar{X}$ for samples from a uniform distribution

Binomial probabilities

Suppose X is a binomial random variable where:

- number of trials $= n$
- each trial is "success" or "failure"
- probability of success $= p$ (and probability of failure is $q = 1 - p$)
- trials are independent
- $X =$ number of "successes"

Then for any number $x = 0, 1, 2, \dots, n$, we can calculate the probability of x successes using:

$$P(X = x) = C(n, x)\, p^x \cdot q^{n-x}$$

Example Suppose you roll a die 10 times. Let X = number of times you roll a 1. Then X is a binomial variable with $n = 10, p = 1/6, q = 5/6$.

e.g. $P(X = 0) = C(10, 0)p^0 q^{10} = 1 \cdot 1 \cdot \left(\frac{5}{6}\right)^{10} = 0.1615$

We can also calculate binomial probabilities in R using the function **dbinom**.

Usage

```
dbinom(x, size, prob, log = FALSE)

pbinom(q, size, prob, lower.tail = TRUE, log.p = FALSE)

qbinom(p, size, prob, lower.tail = TRUE, log.p = FALSE)

rbinom(n, size, prob)
```

Arguments

`x, q`	vector of quantiles.
`p`	vector of probabilities.
`n`	number of observations. If `length(n) > 1`, the length is taken to be the number required.
`size`	number of trials (zero or more).
`prob`	probability of success on each trial.
`log, log.p`	logical; if TRUE, probabilities p are given as log(p).
`lower.tail`	logical; if TRUE (default), probabilities are *P[X ≤ x]*, otherwise, *P[X > x]*.

For example, we can find $P(X = 0)$ from the previous example using:

```
> dbinom(x=0, size=10, prob=1/6)
[1] 0.1615056

> dbinom(0, 10, 1/6)
[1] 0.1615056
```

If we pass in a vector of values, we can get the entire probability distribution of X at once.

```
> dbinom(1:10, 10, 1/6)
 [1] 3.230e-01 2.907e-01 1.550e-01 5.427e-02 1.302e-02 2.171e-03 2.481e-04
 [8] 1.861e-05 8.269e-07 1.654e-08
```

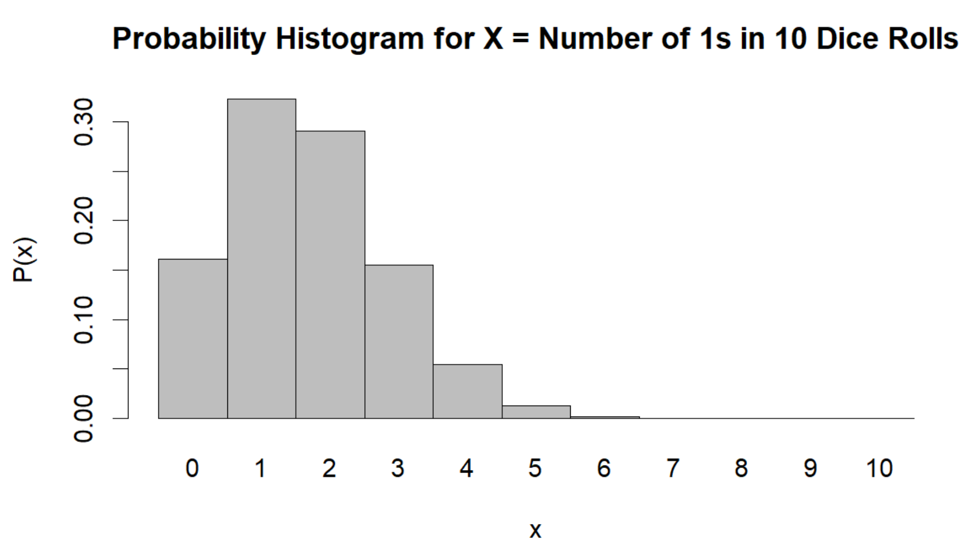
We can use **cbind** and **round** to produce a table showing the probability distribution in a readable way.

```
> cbind(1:10, round(dbinom(1:10, 10, 1/6), digits=4))
      [,1]   [,2]
 [1,]    1 0.3230
 [2,]    2 0.2907
 [3,]    3 0.1550
 [4,]    4 0.0543
 [5,]    5 0.0130
 [6,]    6 0.0022
 [7,]    7 0.0002
 [8,]    8 0.0000
 [9,]    9 0.0000
[10,]   10 0.0000
```

This tells us that, e.g., the probability of getting two 1s when we roll ten dice is 0.2907.

We can also create a probability histogram for these results using **barplot**. Note that we need to use the **names.arg** argument to get labels for each of the bars:

```
> barplot(dbinom(0:10, 10, 1/6), space=0,
        names.arg = 0:10,
        xlab="x", ylab="P(x)",
        main="Prob. Histogram for Number of 1s in 10 Dice Rolls")
```



The function **pbinom** is used to compute *cumulative probabilities*. For example, the following command will give the probability $P(X \leq 2)$, meaning getting two *or fewer* 1s when rolling 10 fair die:

```
> pbinom(2,10,1/6)
[1] 0.7752268
```

1. During one stage in the manufacture of integrated circuits, a coating must be applied. It is known from past experience that one third of chips do not receive a thick enough coating. Suppose 300 chips are randomly selected for testing. Let

$$X = \text{the number of chips that do not receive a thick enough coating}$$

Record the R commands to compute each probability. (Answers given to confirm code.)

   a. $X = 100$ (exactly 100 chips) Ans: 0.04881
   b. $X \leq 100$ (at most 100 chips) Ans: 0.5271
   c. $X < 100$ (fewer than 100) Ans: 0.4783
   d. $X \geq 110$ (at least 110) Ans: 0.1228
   e. $90 \leq X \leq 110$ (between 90 and 110, inclusive) Ans: 0.8017

Simulating Data for a Binomial Variable

R allows us to generate random numbers that follow a binomial distribution.

Example Suppose we roll 10 dice. Let $X =$ the number of times we get a 1. This X is a binomial random variable with $n = 10$ and $p = 1/6$. We can simulate one value of X using:

```
> rbinom(n=1, size=10, prob=1/6)
[1] 2
```

Here is a command that simulates rolling a set of 10 dice 100 times, and returns X (the number of 1s the come up) for each of those 100 experiments:

```
> X.vals <- rbinom(100, 10, 1/6)
> X.vals
  [1] 2 4 3 0 3 2 1 2 1 3 3 1 2 1 0 1 1 1 2 2 0 1 2 3 1 3 1 2 3 3 0 0 1 1 2
 [36] 3 0 2 0 1 0 3 1 0 6 1 1 0 2 2 1 2 0 0 1 1 4 1 2 2 2 2 1 4 0 0 2 0 7 4
 [71] 4 1 0 0 2 2 1 5 3 2 1 3 0 2 2 1 2 5 3 1 1 1 1 1 0 2 1 2 1 2
```

We can organize those results in a table:

```
> table(X.vals)
X.vals
 0  1  2  3  4  5  6  7
19 32 27 13  5  2  1  1
```

Alternatively, we can display the table vertically:

```
> cbind(table(X.vals))
  [,1]
0   19
1   32
2   27
3   13
4    5
5    2
6    1
7    1
```

We can also convert the frequency to *relative* frequency using the **proportions** function:

```
> proportions(table(X.vals))
X.vals
   0    1    2    3    4    5    6    7
0.19 0.32 0.27 0.13 0.05 0.02 0.01 0.01
```

2. In this question, you will generate the probability distribution for $X =$ the number of 1s obtained when rolling 10 dice. Do both of the following:

   a. Write a function **RollDice** that simulates rolling **n.dice** dice **r.reps** times using **sample**. Your function **RollDice** should output a table giving the relative frequencies of the different values of X along with a probability histogram of X. (Hint: use **histogram**.) Run your function for **n.dice**=10, **r.reps=**100000. Record the table and probability histogram.

   b. Write a function **RollDiceBinom** that simulates rolling **n.dice** dice **r.reps** times, using the **rbinom** function. As before, your function should output a table giving the relative frequencies, as well as a probability histogram. Run your function for **n.dice**=10, **r.reps**=100000 and provide a table and a probability histogram.

   (Note: both of your histograms should look very similar to the exact probability distribution obtained on pg. 3-4 of this lab using **dbinom**.)

3. Let's suppose that the true population proportion of left-handed students at BCIT is $p = 0.130$. Assuming you don't know this number, the best way to estimate it would be to take a random sample of students and calculate the proportion $\hat{p}$ for that sample.

   If you take a sample of $n = 200$ students and let $X =$ the number of left-handed students in the sample, then X is a binomial random variable with $n = 200$ and $p = 0.130$.

   a. Find the mean, μ, and the standard deviation, σ, of X.

   b. The sample proportion $\hat{p} = \frac{X}{n}$ is not a binomial variable (also it is closely related). Write a function **Left.Sample.Prop(n.students, p)** that generates a simulated random sample of students (with probability $\boldsymbol{p}$ of being "Left" and $\mathbf{1} - \boldsymbol{p}$ of being "Right"). Find $\hat{p}$ for the sample and return $\hat{p}$. Record one output where you run:
   ```
   > Left.Sample.Prop( 200, 0.130 )
   ```

   c. Write a function **Dist.Left.Sample.Prop(n.students, p, m.trials)** that simulates **m.trials** values of $\hat{p}$ assume a population proportion $\boldsymbol{p}$. Use these values to create a histogram showing the probability distribution of $\hat{p}$. (Let R choose the classes automatically.) Print the mean and standard deviation of $\hat{p}$ based on the simulated values. Record one output and histogram from running:
   ```
   > Dist.Left.Sample.Prop( 200, 0.130, 10^5 )
   ```

Distribution of Sample Mean ($\bar{X}$) for a Uniformly Distributed Population

We are going look at the distribution of $\bar{X}$ when the underlying population is uniform.

A good example of such a population is dice rolls. If you roll a fair die many times, the results 1, 2, 3, 4, 5, and 6 should occur with nearly equal frequency.

We will now investigate what we get when we take many samples of size n from a uniformly distributed population and calculate the mean $\bar{X}$ for each sample. From all these $\bar{X}$ values, we will compute the parameters

$\mu_{\bar{X}} =$ mean of all sample means

$\sigma_{\bar{X}} =$ standard deviation of all sample means

4. Create a function called **DiceMeans** where **DiceMeans(n.dice, m.trials)** simulates rolling **n.dice** dice **m.trials** times. For each trial, compute and store $\bar{X}$ for that sample. Your function should return the following:
    - The overall mean $\mu_{\bar{X}}$ of the **m.trials** sample means
    - The overall standard deviation $\sigma_{\bar{X}}$ of the **m.trials** sample means
    - A relative frequency histogram (drawn using **barplot** with **space=0**) of the **m.trials** sample means. Label the axes appropriately and create a title.

    Run your function for **m=10000** and for the following values of **n.dice**:
    - 1 (i.e, roll a single die)
    - 2 (roll 2 dice)
    - 10
    - 50
    - 100

    Submit five outputs (histograms + corresponding $\mu_{\bar{X}}$ and $\sigma_{\bar{X}}$). How do the means and standard deviations change as **n** increases? How do the shapes of the histograms compare?