Using R for Probability and Statistical Analysis – Part 1
Week 12 Notes
STAT 330

Outline of Notes:

1. Descriptive Statistics	6. The Normal Distribution
2. Contingency Tables	7. Other Continuous Distributions
3. The Binomial Distribution	8. Graphing Distributions
4. The Hypergeometric Distribution	9. QQ Plot for Normality
5. Other Discrete Distributions	

Descriptive Statistics

- The `summary()` function gives some useful statistics for vectors, matrices, factors, and data frames.
  - Example with a numeric vector:

```
> y<-c(3,3,5,8,3,13,7,5,12)  #THIS OBJECT IS USED THROUGHOUT THESE NOTES
> summary(y)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  3.000   3.000   5.000   6.556   8.000  13.000
```

  - Example with a numeric matrix (works column by column):

```
> mymat<-matrix(1:50,10,5)  #THIS OBJECT IS USED THROUGHOUT THESE NOTES
> summary(mymat)
       V1              V2              V3              V4       
 Min.   : 1.00   Min.   :11.00   Min.   :21.00   Min.   :31.00  
 1st Qu.: 3.25   1st Qu.:13.25   1st Qu.:23.25   1st Qu.:33.25  
 Median : 5.50   Median :15.50   Median :25.50   Median :35.50  
 Mean   : 5.50   Mean   :15.50   Mean   :25.50   Mean   :35.50  
 3rd Qu.: 7.75   3rd Qu.:17.75   3rd Qu.:27.75   3rd Qu.:37.75  
 Max.   :10.00   Max.   :20.00   Max.   :30.00   Max.   :40.00  
       V5       
 Min.   :41.00  
 1st Qu.:43.25  
 Median :45.50  
 Mean   :45.50  
 3rd Qu.:47.75  
 Max.   :50.00  
```

  - Example with a factor (gives the frequency for each level):

```
> data(Orange)   #built-in R dataset
> summary(Orange$Tree)
3 1 5 2 4
7 7 7 7 7
```

  - Example with a data frame (works column by column):

```
> summary(Orange)
Tree        age          circumference  
 3:7   Min.   : 118.0   Min.   : 30.0  
 1:7   1st Qu.: 484.0   1st Qu.: 65.5  
 5:7   Median :1004.0   Median :115.0  
 2:7   Mean   : 922.1   Mean   :115.9  
 4:7   3rd Qu.:1372.0   3rd Qu.:161.5  
       Max.   :1582.0   Max.   :214.0  
```

- There are many functions for basic descriptive statistics.

Function	Purpose
`mean()`	calculate the arithmetic mean
`median()`	calculate the median
`sd()`	calculate the sample standard deviation
`var()`	calculate the sample variance
`min()`	returns the minimum value
`max()`	returns the maximum value
`range()`	returns the minimum and maximum values
`fivenum()`	returns the five-number summary (min, quartiles, max)
`quantile(x, p)`	returns the p^{th} percentile for object x ($0 \le p \le 1$)

- Examples with a numeric **vector**:

```
> mean(y)
[1] 6.555556
> median(y)
[1] 5
> sd(y)
[1] 3.811532
> var(y)
[1] 14.52778
> min(y)
[1] 3
> max(y)
[1] 13
> range(y)
[1]  3 13
> fivenum(y)
[1]  3  3  5  8 13
> quantile(y,.75)
75%
8
```

- **<u>Note</u>:** Most of these functions are picky about values that are missing or not available (`NA`). Even one `NA` value in the data can cause any of these functions to return `NA` or to even give an error message.
  - Examples:

```
> x<-c(30,16,1,16,28,12,NA,25,2,3)
> mean(x)
[1] NA
> quantile(x,.75)
Error in quantile.default(x, .75) :
  missing values and NaN's not allowed if 'na.rm' is FALSE
```

  - To fix this problem, you can add an `na.rm=TRUE` argument to the function. This tells R to ignore the `NA` values. (Note that by default, `na.rm=FALSE`.)

```
> mean(x,na.rm=TRUE)
[1] 14.77778
```

```
> quantile(x,.75,na.rm=T)
75% 
 25 
```

- The `mean()`, `median()`, `sd()`, `max()`, `min()`, `range()`, `fivenum()`, and `quantile()` functions treat a numeric **matrix** as a *single set of data* (i.e., a single variable), which is not always desired/applicable.
  - Examples:

```
> mean(mymat)
[1] 25.5
> range(mymat)
[1]  1 50
```

- When the `var()` function is applied to a numeric **matrix** or **data frame**, it will return something called a covariance matrix. The diagonal values of this matrix will contain the variances of the respective column variables. The off-diagonal values contain the covariance between the row and column variables (you are not expected to be familiar with covariance).
  - Examples:

```
> data(cars)
> var(cars)
          speed      dist
speed  27.95918 109.9469
dist  109.94694 664.0608
```

variance for speed

covariance between distance and speed

variance for distance

- Some of these functions are "smart" about numerical data frames; they will treat each column as a separate variable. However, some of them are not.

- **Example 1:** Use the iris **data frame** (which contains numeric *and* categorical variables) with each of the functions from the table on the previous page. Report your findings about how they each treat the data frame.

- **Example 2:** Use the cars **data frame** (which contains *only* numeric variables) with each of the functions from the table on the page 2. Report your findings about how they each treat the data frame.

- To obtain the mode, the number that is most frequently used in a dataset (e.g., mode=3 in vector `y` created on page 1), you need to use the function called `Mode()` in the package called '**prettyR**'.
  - To do this, go to the *Packages* menu in R, choose *Set CRAN Mirror* and select a mirror from the list that is close to you. I generally select *USA (TN)*.
  - Go back to the *Packages* menu and select *Install Package(s)*, pick the package named *prettyR* from the list of packages, and click *OK*.
  - Go to the *Packages* menu, select *Load Package*, pick the package named *prettyR* from the list, and click *OK*. Alternatively, you can type `library(prettyR)` at the command prompt.

  - Information about the prettyR package can be found at: http://cran.r-project.org/web/packages/prettyR/prettyR.pdf

- **Example 3:**
  1. What is the difference between typing `mode(y)` and `Mode(y)`?

  2. Compute the value which represents the 90th percentile of the numbers in the `y` vector.

3. Use the `freq()` function on the `y` vector. What does this function do?

4. Create a vector called `y2` using the following code:

```
> y2<-c(y,NA)
```

   a. What does this vector look like?

   b. Use the `freq()` function on the `y2` vector. Comment on how the results differ from #3.

Contingency Tables

- The `table()` function can be used to create contingency tables.

- If the object is a vector, a one-way contingency table is created.
  - Examples:

```
> age<-c(3,5,7,5,3,2,6,8,5,6,9,4,5,7,3,4)
> table(age)
age
2 3 4 5 6 7 8 9
1 3 2 4 2 2 1 1

> gender<-c("female","male","male","male","female","female","female",
+ "male","female","female","male","male","female","female","female",
+ "female")
> table(gender)
gender
female   male
    10      6
```

- If the object is a data frame, then a two-way contingency table is created.
  - Example:

```
> info<-data.frame(age,gender)
> table(info)
   gender
age female male
  2      1    0
  3      3    0
  4      1    1
  5      2    2
  6      2    0
  7      1    1
```

```
  8       0     1
  9       0     1
```

- Alternatively, you can give the two vectors to the `table()` function to create a two-way contingency table.

```
> table(age,gender)
   gender
age female male
  2       1     0
  3       3     0
  4       1     1
  5       2     2
  6       2     0
  7       1     1
  8       0     1
  9       0     1
```

- If three variables are given to the `table()` function, then a separate two-way table (of the first two variables) would be created for each value of the third variable.
  - Example:

```
> eyes<-rep(c("brown","blue","green","hazel"),4)
> table(age,gender,eyes)
, , eyes = blue

   gender
age female male
  2       1     0
  3       0     0
  4       0     0
  5       0     1
  6       1     0
  7       1     0
  8       0     0
  9       0     0

, , eyes = brown

   gender
age female male
  2       0     0
  3       2     0
  4       0     0
  5       2     0
  6       0     0
  7       0     0
  8       0     0
  9       0     0

, , eyes = green

   gender
age female male
  2       0     0
  3       1     0
  4       0     0
```

```
  5      0    0
  6      1    0
  7      0    1
  8      0    0
  9      0    1
, , eyes = hazel

   gender
age female male
  2      0    0
  3      0    0
  4      1    1
  5      0    1
  6      0    0
  7      0    0
  8      0    1
  9      0    0
```

- Note: `table(cbind(info,eyes))` would result in the same output.

- The `ftable()` function can be used to create a "flat" contingency table.
  - Example:

```
> ftable(age,gender,eyes)
           eyes blue brown green hazel
age gender
2   female         1     0     0     0
    male           0     0     0     0
3   female         0     2     1     0
    male           0     0     0     0
4   female         0     0     0     1
    male           0     0     0     1
5   female         0     2     0     0
    male           1     0     0     1
6   female         1     0     1     0
    male           0     0     0     0
7   female         1     0     0     0
    male           0     0     1     0
8   female         0     0     0     0
    male           0     0     0     1
9   female         0     0     0     0
    male           0     0     1     0
```

- To add row and column totals (sums) to a table, you can use the `addmargins()` function.
  - Example:

```
> addmargins(table(age,gender))
    gender
age   female male Sum
  2        1    0   1
  3        3    0   3
  4        1    1   2
  5        2    2   4
  6        2    0   2
  7        1    1   2
```

```
8          0    1    1
9          0    1    1
Sum       10    6   16
```

The Binomial Distribution

- The binomial distribution deals with observing the number of successes in a fixed number of trials, *n* (called `size` in R), where each trial has the same probability of success, *p* (called `prob` in R). The trials must be identical and independent. The random variable (X) is the number of successes observed.

- We can use R to find probabilities of a binomial random variable (i.e., the probability of a certain number of successes). To do this, we use the `dbinom()` function.
  - General format: To find P(X = x), use `dbinom(x, size, prob)`
  - Example: Suppose we are rolling a fair die 10 times and wish to observe the number of times a five is rolled (let X = number of fives rolled). Here X has the binomial distribution with n = 10 and p = 1/6. Calculate the probability that three fives are rolled; i.e., calculate P(X = 3).

```
> dbinom(3,10,1/6)
[1] 0.1550454
```

- We can also calculate cumulative probabilities, P(X ≤ q) for a given value of q. To do this, use the `pbinom()` function.
  - General format: To find P(X ≤ q), use `pbinom(q, size, prob)`
  - Dice Example: Calculate the probability that *at most* three fives are rolled; i.e., calculate P(X ≤ 3).

```
> pbinom(3,10,1/6)
[1] 0.9302722
```

- To find P(X > q) for a given value of q, use the `lower.tail=FALSE` argument in the `pbinom()` function.
  - Dice Example: Calculate the probability that *more* than three fives are rolled; i.e., calculate P(X > 3).

```
> pbinom(3,10,1/6,lower.tail=F)
[1] 0.06972784
> 1-pbinom(3,10,1/6)        #using the complement rule to find same answer
[1] 0.06972784
```

- You can also give the `dbinom()` and `pbinom()` functions a vector of values for `x`. The function will return the probability for each value in the vector (in the same order in which it was given).

- Dice Example: The probabilities and cumulative probabilities using x-values from 0 to 10.

```
> dbinom(0:10, 10, 1/6)
 [1] 1.615056e-01 3.230112e-01 2.907100e-01 1.550454e-01 5.426588e-02
 [6] 1.302381e-02 2.170635e-03 2.480726e-04 1.860544e-05 8.269086e-07
[11] 1.653817e-08

> pbinom(0:10, 10, 1/6)
 [1] 0.1615056 0.4845167 0.7752268 0.9302722 0.9845380 0.9975618 0.9997325
 [8] 0.9999806 0.9999992 1.0000000 1.0000000
```

Notice that the answers from the `dbinom()` function were given in scientific notation (the e that is used is *not* Euler's number). For example, P(X = 0) = 1.615056 x 10^{-1}.

- To find quantiles from the binomial distribution, the `qbinom()` function can be used. For a given probability, p (0 ≤ p ≤ 1), the function will return the smallest value of x such that P(X ≤ x) ≥ p.
  - General format: To find the smallest x such that P(X ≤ x) ≥ p, use `qbinom(p, size, prob)`
  - Dice Example: Find the minimum value of x such that the probability of rolling at most x fives is at least 90%.

```
> qbinom(0.9,10,1/6)
[1] 3
```

- We can generate binomial random variables using the `rbinom()` function.
  - General format: To generate `n` random variables from the binomial distribution with `size` trials and probability of success `prob`, use `rbinom(n, size, prob)`
  - Dice Example: Simulate rolling a die 10 times and observing the number of fives rolled.

```
> rbinom(1,10,1/6)    # results will vary
[1] 2
```

  - Dice Example: Now simulate doing this 100 times (with each time consisting of 10 die rolls and observing the number of fives rolled).

```
> rbinom(100,10,1/6)    # results will vary; not shown here
```

The Hypergeometric Distribution

- The hypergeometric distribution deals with observing the number of successes when drawing a certain number of items (`k`) without replacement from a finite population with a certain number of successes (`m`) and failures (`n`).
  - To use the terminology in the R help file for the hypergeometric functions, consider drawing a sample of **k** balls from an urn that contains **m** white balls and **n** black balls. We are interested in observing the *number of white balls* in a sample drawn from the urn without replacement.

- We can use R to find probabilities of a hypergeometric random variable (i.e., the probability of a certain number of successes). To do this, we use the `dhyper()` function.
  - General format: To find P(X = x), use `dhyper(x, m, n, k)`
  - Example: Suppose we are drawing 5 cards from a standard deck. Calculate the probability that three face cards are drawn (let X = number of face cards drawn); i.e., find P(X = 3) where m = 12 (total number of face cards in the deck), n = 40 (total number of non-face cards in the deck), and k = 5 (number of cards being drawn).

```
> dhyper(3,12,40,5)
[1] 0.06602641
```

- We can also calculate cumulative probabilities, $P(X \leq q)$ for a given value of q. To do this, use the `phyper()` function.
  - General format: To find $P(X \leq q)$, use `phyper(q, m, n, k)`
  - Cards Example: Calculate the probability that *at most* three face cards are drawn; i.e., calculate $P(X \leq 3)$.

```
> phyper(3,12,40,5)
[1] 0.9920768
```

- To find $P(X > q)$ for a given value of q, use the `lower.tail=FALSE` argument in the `phyper()` function.
  - Cards Example: Calculate the probability that *more* than three face cards are drawn; i.e., calculate $P(X > 3)$.

```
> phyper(3,12,40,5,lower.tail=F)
[1] 0.007923169

> 1-phyper(3,12,40,5)       #using the complement rule to find same answer
[1] 0.007923169
```

- You can also give the `dhyper()` and `phyper()` functions a vector of values for `x`. The function will return the probability for each value in the vector (in the same order in which it was given).
  - Cards Example: The probabilities and cumulative probabilities using x-values from 0 to 5.

```
> dhyper(0:5,12,40,5)
[1] 0.2531812725 0.4219687875 0.2509003601 0.0660264106 0.0076184320
[6] 0.0003047373
> phyper(0:5,12,40,5)
[1] 0.2531813 0.6751501 0.9260504 0.9920768 0.9996953 1.0000000
```

- To find quantiles from the hypergeometric distribution, the `qhyper()` function can be used. For a given probability, p ($0 \leq p \leq 1$), the function will return the smallest value of x such that $P(X \leq x) \geq p$.

- General format: To find the smallest x such that $P(X \le x) \ge p$, use `qhyper(p, m, n, k)`
- Cards Example: Find the minimum value of x such that the probability of getting at most x face cards is at least 80%.

```
> qhyper(0.8,12,40,5)
[1] 2
```

- We can generate hypergeometric random variables using the `rhyper()` function.
  - General format: To generate `nn` random variables from the hypergeometric distribution, use `rhyper(nn, m, n, k)`
  - Cards Example: Simulate drawing five cards without replacement from a standard deck and observing the number of face cards obtained.

```
> rhyper(1,12,40,5)    # results will vary
[1] 3
```

  - Cards Example: Now simulate doing this 100 times (dealing five-card hands from a standard deck and observing the number of face cards – consider each *hand* as coming from a new deck).

```
> rhyper(100,12,40,5)    # results will vary; not shown here
```

Other Discrete Distributions

- R also has functions for the Poisson, geometric, and negative binomial distributions, among others.

Poisson Distribution	
The Poisson random variable deals with counting the number of successes/events in a given area of opportunity (time/distance/area/volume/etc.), where the mean number of successes/events per area of opportunity is λ (`lambda`). X = number of successes/events in the area of opportunity	
Function	**Purpose**
`dpois(x, lambda)`	Calculate $P(X = x)$
`ppois(q, lambda)`	Calculate $P(X \le q)$
`qpois(p, lambda)`	Find smallest x such that $P(X \le x) \ge p$ $(0 \le p \le 1)$
`rpois(n, lambda)`	Generate n Poisson random variables

- Poisson Examples: A help desk receives an average of two calls per minute. Here $\lambda = 2$ per minute and X = the number of calls in a minute.
  - Find the probability that three calls arrive in a minute; i.e., $P(X = 3)$.

```
> dpois(3,2)
[1] 0.180447
```

- Find the probability that at most four calls arrive in a minute; i.e., P(X ≤ 4).

```
> ppois(4,2)
[1] 0.947347
```

- Find the probability that more than four calls arrive in a minute; i.e., P(X > 4).

```
> 1-ppois(4,2)
[1] 0.05265302
> ppois(4,2,lower.tail=F)
[1] 0.05265302
```

- Simulate the number of calls that arrive in a minute for 60 randomly selected minutes.

```
> rpois(60,2)   # results will vary
```

<table>
<tr><th colspan="2">Geometric Distribution</th></tr>
<tr><td colspan="2">The geometric random variable deals with observing the number of failures before the first success occurs. Each trial is independent and identical with the same probability of success (<code>prob</code>).

X = number of failures before the first success occurs

Note: Many sources (including me in STAT 301) will define X as the number of <u>trials</u> until and including the first success, but that is not how it is done in R.</td></tr>
<tr><th>Function</th><th>Purpose</th></tr>
<tr><td><code>dgeom(x, prob)</code></td><td>Calculate P(X = x)</td></tr>
<tr><td><code>pgeom(q, prob)</code></td><td>Calculate P(X ≤ q)</td></tr>
<tr><td><code>qgeom(p, prob)</code></td><td>Find smallest x such that P(X ≤ x) ≥ p (0 ≤ p ≤ 1)</td></tr>
<tr><td><code>rgeom(n, prob)</code></td><td>Generate n geometric random variables</td></tr>
</table>

- Geometric Examples: Consider rolling a fair 6-sided die until the first six is rolled. Here, the probability of success (rolling a six) for each roll is 1/6 and X = the number of rolls before obtaining the first six.

  - Find the probability that the first six occurs on the fifth roll; i.e., P(X = 4).

```
> dgeom(4,1/6)
[1] 0.0779651
```

  - Find the probability that the first six occurs within the first five rolls; i.e., P(X ≤ 4).

```
> pgeom(4,1/6)
[1] 0.5981224
```

  - Find the probability that the first six occurs after the fifth roll; i.e., P(X > 4).

```
> 1-pgeom(4,1/6)
[1] 0.4018776
> pgeom(4,1/6,lower.tail=F)
[1] 0.4018776
```

- Simulate rolling a die and observing the number of rolls before the first six occurs. Do this 15 times.

```
> rgeom(15,1/6)      # results will vary
```

Negative Binomial Distribution	
The negative binomial random variable deals with observing the number of *failures* before the r^{th} (r is `size` in R) success occurs. Each trial is identical and independent with the same probability of success (`prob`). X = number of *failures* before the r^{th} success occurs **Note:** Many sources (including me in STAT 301) will define X as the number of **<u>trials</u> until *and including* the r^{th} success**, but that is ***not*** how it is done in R.	
Function	**Purpose**
`dnbinom(x, size, prob)`	Calculate $P(X = x)$
`pnbinom(q, size, prob)`	Calculate $P(X \le x)$
`qnbinom(p, size, prob)`	Find smallest x such that $P(X \le x) \ge p$ $(0 \le p \le 1)$
`rnbinom(n, size, prob)`	Generate n negative binomial random variables

- Negative Binomial Examples: Consider rolling a fair die until the 3^{rd} six is rolled. Here, the probability of success (rolling a six) for each roll is 1/6 and X = the number of failures (non-sixes) that occur before the 3^{rd} six.

  - Find the probability that the 3^{rd} six occurs on the 10^{th} roll; i.e., $P(X = 7)$ since 7 non-sixes would have to happen before the 3^{rd} six happens on the 10^{th} roll.

```
> dnbinom(7,3,1/6)
[1] 0.04651361
```

  - Find the probability that the 3^{rd} six occurs within the first 10 rolls; i.e., $P(X \le 7)$.

```
> pnbinom(7,3,1/6)
[1] 0.2247732
```

  - Find the probability that the 3^{rd} six occurs after the 10^{th} roll; i.e., $P(X > 7)$.

```
> 1-pnbinom(7,3,1/6)
[1] 0.7752268
> pnbinom(7,3,1/6,lower.tail=F)
[1] 0.7752268
```

  - Simulate rolling a die and observing the number of non-sixes rolled before the 3^{rd} six occurs. Do this 20 times.

```
> rnbinom(20,3,1/6)  # results will vary
```

The Normal Distribution

- Recall that a normal distribution is a bell-shaped, symmetric, continuous distribution with mean μ and standard deviation σ.
- The standard normal distribution (often denoted by Z) is the normal distribution with μ = 0 and σ = 1.
- Note: Recall that for a continuous random variable, we only find probabilities for *intervals*.

- To find normal probabilities, we use the `pnorm()` function.
  - If Z is standard normally distributed then:
    - P(Z ≤ q) for some number q is given by `pnorm(q)`
    - P(Z > q) is given by `pnorm(q, lower.tail=FALSE)`
  - If X is normally distributed with mean μ and standard deviation σ, then:
    - P(X ≤ q) for some number q is given by `pnorm(q, mean=μ, sd=σ)`
    - P(X > q) is given by `pnorm(q, mean=μ, sd=σ, lower.tail=FALSE)`
    - Note that you must give the values of μ and σ.

- Examples: Suppose that the daily high January temperatures on a tropical island are approximately normally distributed with mean 82°F and standard deviation 2.5°F. Let X = the high temperature for a January day on this island.
  - Find the probability that a randomly selected January day has a high temperature that is less than 80°F; i.e., P(X < 80).

```
> pnorm(80,mean=82,sd=2.5)
[1] 0.2118554
```

  - Find the probability that a randomly selected January day has a high temperature that is higher than 85°F; i.e., P(X > 85).

```
> 1-pnorm(85,82,2.5)        #notice that "mean=" and "sd=" can be left out
[1] 0.1150697
> pnorm(85,82,2.5,lower.tail=F)
[1] 0.1150697
```

- To find quantiles from the normal distribution, the `qnorm()` function can be used. For a given probability, p (0 ≤ p ≤ 1), the function will return the value of x such that P(X ≤ x) = p.
  - If Z is standard normally distributed then:
    - To find z such that P(Z ≤ z) = p for some probability p, use `qnorm(p)`
    - To find z such that P(Z > z) = p, use `qnorm(p, lower.tail=FALSE)`
  - If X is normally distributed with mean μ and standard deviation σ, then:
    - To find x such that P(X ≤ x) = p for some probability p, use `qnorm(p, mean=μ, sd=σ)`
    - To find x such that P(X > x) = p, use `qnorm(p, mean=μ, sd=σ, lower.tail=FALSE)`

- Example: Consider the tropical island.
  - Find the 75th percentile of daily high January temperatures; i.e., find x such that $P(X \le x) = 0.75$.

```
> qnorm(.75,82,2.5)
[1] 83.68622
```

  - Find the temperature such that 80% of January days have a high temperature that is higher than this value; i.e., find x such that $P(X > x) = 0.80$.

```
> qnorm(.80,82,2.5,lower.tail=F)
[1] 79.89595
> qnorm(.2,82,2.5)
[1] 79.89595
```

- We have already seen the use of the `rnorm()` function to generate random variables from the normal distribution. The general form (with default values for the standard normal distribution) is `rnorm(n, mean=0, sd=1)`.

- Example: Simulate the high temperature for 5 randomly selected January days on the tropical island.

```
> rnorm(5, 82.5, 2.5)          # results will vary; not shown here
```

- Examples in the context of inferential statistics:
  - Consider testing H_0: $p = 0.75$ vs. H_a: $p > 0.75$, and obtaining a test statistic of $z^* = 2.5$. For this one-tailed test, the p-value is calculated as $P(Z \ge 2.5)$.

```
> pnorm(2.5,lower.tail=F)
[1] 0.006209665
> 1-pnorm(2.5)
[1] 0.006209665
```

  - Suppose you were testing the two-tailed alternative hypothesis $p \ne 0.75$. Then the p-value is calculated as $2 \cdot P(Z \ge |2.5|)$. There are multiple ways you could do this with R.

```
> 2*pnorm(2.5,lower.tail=F)
[1] 0.01241933
> 2*(1-pnorm(2.5))
[1] 0.01241933
> 2*pnorm(abs(2.5),lower.tail=F)
[1] 0.01241933
```

  - Suppose you wish to calculate a 95% confidence interval for the proportion. You need the value from the standard normal distribution such that $P(Z \ge z) = 0.025$. [Recall that when you have a confidence level of $1 - \alpha$, you need the z-value with $\alpha/2$ to the right. In this case, $1 - \alpha = 0.95$, so $\alpha/2 = 0.025$.]

```
> qnorm(0.025,lower.tail=F)
[1] 1.959964
```

Other Continuous Distributions

- R also has functions for the t, Chi-square, and F distributions (among others). The `pdist()` function is useful in the calculation of p-values in hypothesis testing, and the `qdist()` function is useful in the calculation of a critical value for confidence intervals. (Note: *dist* is replaced with the appropriate term for the desired distribution.)

- The examples will illustrate using these functions in the context of hypothesis testing and confidence intervals. Depending on your previous statistics class(es), you may not have seen some of these tests. ***That is OK!*** You are not expected to remember all of the types of tests and confidence intervals and all of the specifics. However, if you are given a probability statement such as $P(t_{24} \leq -1.28)$, you should be able to use R to evaluate the expression.

The t Distribution	
The t distribution is characterized by its degrees of freedom (`df`). The degrees of freedom is often indicated as a subscript, t_{df}.	
Function	**Purpose**
`pt(q, df)`	Calculate $P(t_{df} \leq q)$
`pt(q, df, lower.tail=F)`	Calculate $P(t_{df} > q)$
`qt(p, df)`	Find the value from the t_{df} distribution (t) such that $P(t_{df} \leq t) = p$
`qt(p, df, lower.tail=F)`	Find the value from the t_{df} distribution (t) such that $P(t_{df} \geq t) = p$
`rt(n, df)`	Generate n random variables from the t_{df} distribution.

- p-value examples with t distribution:
  - Consider testing H_0: $\mu = 100$ vs. H_a: $\mu < 100$, and obtaining a test statistic of $t^* = -1.28$ with df = 24. The p-value for this one-tailed test is calculated as $P(t_{24} \leq -1.28)$.

```
> pt(-1.28,24)
[1] 0.1063899
```

  - Suppose you were testing the two-tailed alternative hypothesis $\mu \neq 100$. Then the p-value is calculated as $2 \cdot P(t_{24} \geq |-1.28|)$. There are multiple ways you could do this with R.

```
> 2*pt(-1.28,24)
[1] 0.2127798
> 2*pt(1.28,24,lower.tail=F)
[1] 0.2127798
> 2*pt(abs(-1.28),24,lower.tail=F)
[1] 0.2127798
```

- Critical value example with the t distribution:
  - Suppose you wish to calculate a 99% confidence interval for the population mean (μ) based on a sample of size 30 (so you have 29 degrees of freedom). You need the t value such that $P(t_{29} \geq t) = 0.005$. [Recall that when you have a confidence level of $1 - \alpha$, you need the t-value with $\alpha/2$ to the right. In this case, $1 - \alpha = 0.99$, so $\alpha/2 = 0.005$.]

```
> qt(0.005,29,lower.tail=F)
[1] 2.756386
```

The Chi-Square Distribution	
The Chi-square distribution (χ^2) is characterized by its degrees of freedom (`df`). The degrees of freedom is often indicated as a subscript, χ^2_{df}.	
Function	**Purpose**
`pchisq(q, df)`	Calculate P(χ^2_{df} ≤ q)
`pchisq(q, df, lower.tail=F)`	Calculate P(χ^2_{df} > q)
`qchisq(p, df)`	Find the value from the χ^2_{df} distribution (c) such that P(χ^2_{df} ≤ c) = p
`qchisq(p, df, lower.tail=F)`	Find the value from the χ^2_{df} distribution (c) such that P(χ^2_{df} ≥ c) = p
`rchisq(n, df)`	Generate n random variables from the χ^2_{df} distribution.

- p-value examples with χ^2 distribution:
  - Consider testing H_0: σ = 1.5 vs. H_a: σ > 1.5, and obtaining a test statistic of c* = 38.28 with df = 20. The p-value for this one-tailed test is calculated as P(χ^2_{20} ≥ 38.28).

```
> pchisq(38.28,20,lower.tail=F)
[1] 0.008183203
```

  - Suppose you were testing the two-tailed alternative hypothesis σ ≠ 1.5. Then the p-value is calculated as 2·min{P(χ^2_{20} ≤ 38.28), P(χ^2_{20} ≥ 38.28)}.

```
> 2*min(pchisq(38.28,20),pchisq(38.28,20,lower.tail=F))
[1] 0.01636641
```

- Critical value example with the χ^2 distribution:
  - Suppose you need to find the critical values from the χ^2 distribution with 35 degrees of freedom for a 95% confidence interval. This would require you to find two values: c_1 such that P(χ^2_{35} ≥ c_1) = 0.025 and c_2 such that P(χ^2_{35} ≥ c_2) = 0.975.

```
> qchisq(0.025,35,lower.tail=F)
[1] 53.20335
> qchisq(0.975,35,lower.tail=F)
[1] 20.56938
```

The F Distribution	
The F distribution is characterized by its numerator degrees of freedom (`df1`) and denominator degrees of freedom (`df2`). The degrees of freedom is often indicated as a subscript, $F_{df1, df2}$.	
Function	**Purpose**
`pf(q, df1, df2)`	Calculate P($F_{df1, df2}$ ≤ q)
`pf(q, df1, df2, lower.tail=F)`	Calculate P($F_{df1, df2}$ > q)
`qf(p, df1, df2)`	Find the value from the $F_{df1, df2}$ distribution (f) such that P($F_{df1, df2}$ ≤ f) = p
`qf(p, df1, df2, lower.tail=F)`	Find the value from the $F_{df1, df2}$ distribution (f) such that P($F_{df1, df2f}$ ≥ f) = p
`rf(n, df1, df2)`	Generate n random variables from the $F_{df1, df2}$ distribution.

- p-value example with F distribution:
  - Consider testing H_0: $\mu_1 = \mu_2 = \mu_3$ vs. H_a: not all μ are equal, and obtaining a test statistic of f* = 2 with 12 numerator degrees of freedom and 6 denominator degrees of freedom. The p-value for this ANOVA test is calculated as $P(F_{12,\,6} \geq 2)$.

```
> pf(2,12,6,lower.tail=F)
[1] 0.2030822
> 1-pf(2,12,6)
[1] 0.2030822
```

- Critical value example with the F distribution:
  - Suppose you need to find the critical values from the F distribution with 15 numerator degrees of freedom and 20 denominator degrees of freedom for a 90% confidence interval. This would require you to find two values: f_1 such that $P(F_{15,\,20} \geq f_1) = 0.05$ and f_2 such that $P(F_{15,\,20} \geq f_2) = 0.95$.

```
> qf(0.05,15,20,lower.tail=F)
[1] 2.203274
> qf(0.95,15,20,lower.tail=F)
[1] 0.4296391
```

- Note: There are also functions for other continuous distributions, such as the uniform, gamma, and exponential.

Graphing Distributions

- To graph the probability density function of a **CONTINUOUS** random variable, the `d`*`dist`*`()` function can be used (where *`dist`* is replaced with the appropriate term for the desired distribution).
  - Create a fine grid of x-values (that are appropriate for that distribution) using the `seq()` function.
  - Plot x vs. the `d`*`dist`*`()` function evaluated at the values in the x vector. **Use `type='l'`**.

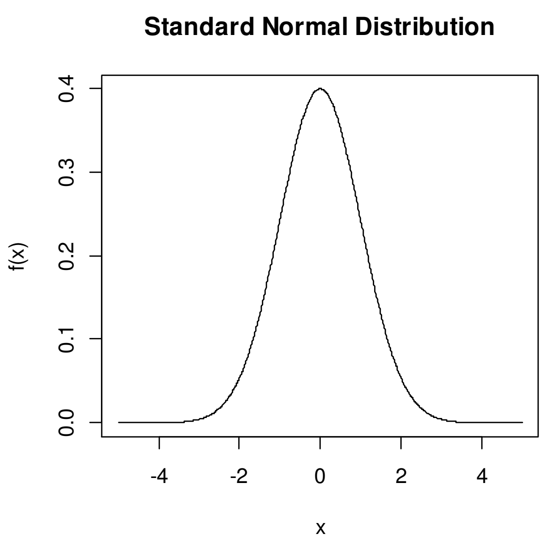
- Example: standard normal distribution (shown at top left on next page)

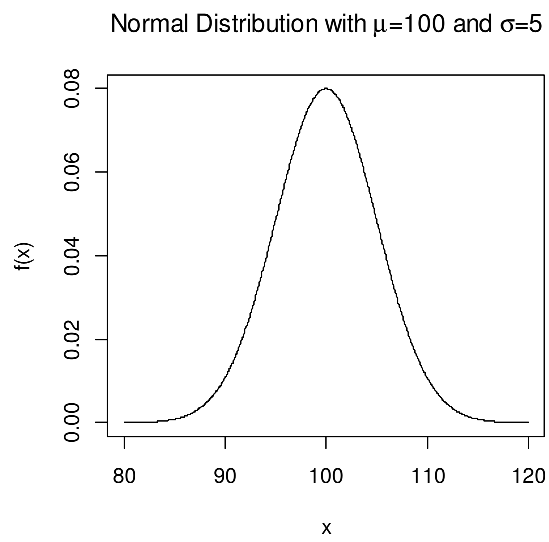
```
> x<-seq(-5,5,length=1000)
> plot(x,dnorm(x),type='l',ylab="f(x)",main="Standard Normal Distribution")
```

- Example: normal distribution with μ = 100 and σ = 5 (shown at top right of next page)

```
> x<-seq(80,120,length=1000)
> plot(x,dnorm(x,100,5),type='l',ylab="f(x)",main=
+ expression(paste("Normal Distribution with ",mu,"=100 and ",sigma,"=5")))
```

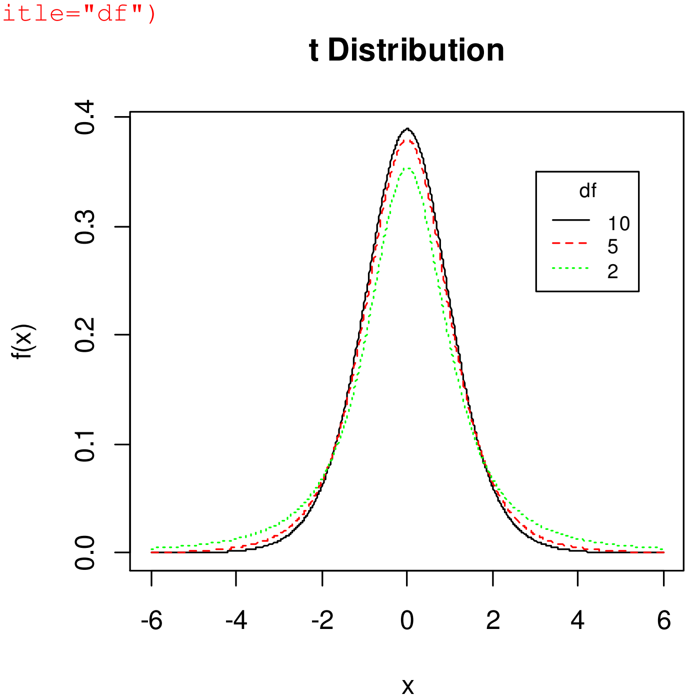
  - Notice that I got fancy by putting Greek symbols in the title!!! ☺





- Example: t distribution with varying degrees of freedom

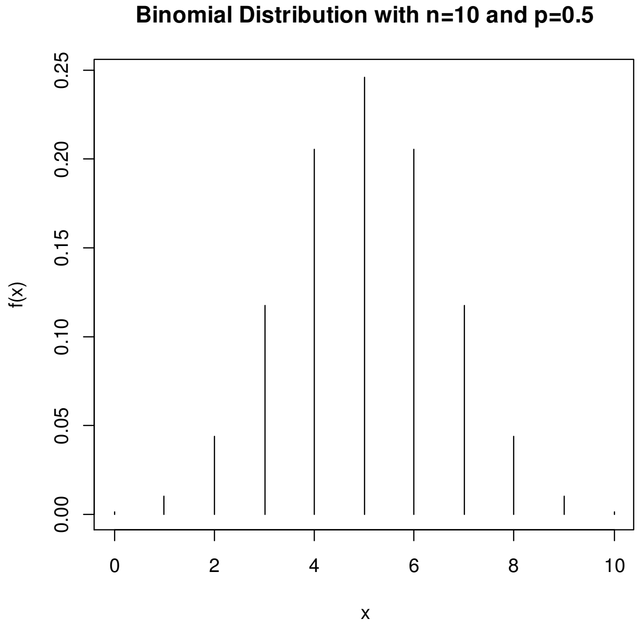
```
> x<-seq(-6,6,length=1000)
> plot(x,dt(x,10),type='l',main="t Distribution",ylab="f(x)")
> lines(x,dt(x,5),col="red",lty=2)
> lines(x,dt(x,2),col="green",lty=3)
> legend(3,.35,c("10","5","2"),lty=1:3,col=c("black","red","green"),
+ cex=0.75,title="df")
```



- To <u>graph the probability mass function of a **DISCRETE** random variable</u>, the `ddist()` function can be used (where `dist` is replaced with the appropriate term for the desired distribution).
  - Create a vector of <u>*all possible x-values for **that** distribution*</u> (if the distribution has an infinite number of possibilities, then just include a large enough number of values).
  - Plot x vs. the `ddist()` function evaluated at the values in the x vector. **Use `type='h'`**.

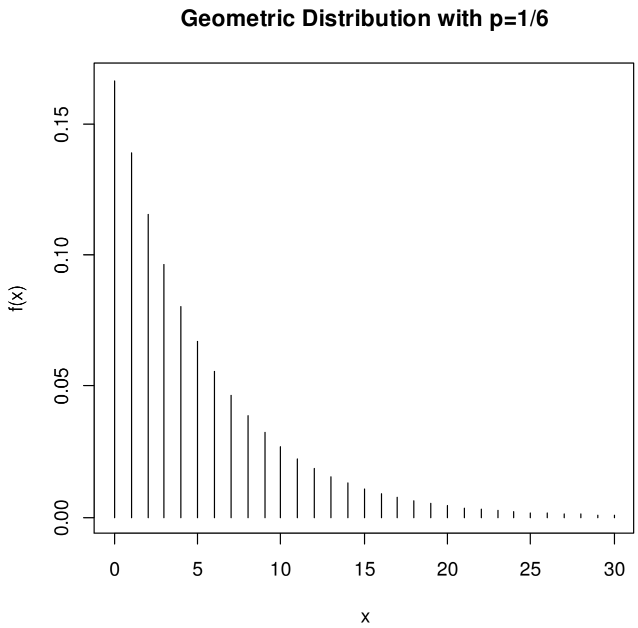
- Example: binomial distribution with 10 trials and probability of success 0.5

```
> x<-0:10 #possible values of x are 0 to 10
> plot(x,dbinom(x,10,.5),type='h',ylab='f(x)',main=
+ 'Binomial Distribution with n=10 and p=0.5')
```



- Example: geometric distribution with probability of success 1/6

```
> x<-0:30 #possible values of x are 0 to infinity
> plot(x,dgeom(x,1/6),type='h',ylab='f(x)',main=
+ "Geometric Distribution with p=1/6")
```



QQ Plot for Normality

- A visual tool to determine if a dataset comes from a reasonably normal population is the quantile-quantile plot (or QQ plot). If the plot shows an approximate straight line, then it can be assumed that the data likely comes from a normal population.

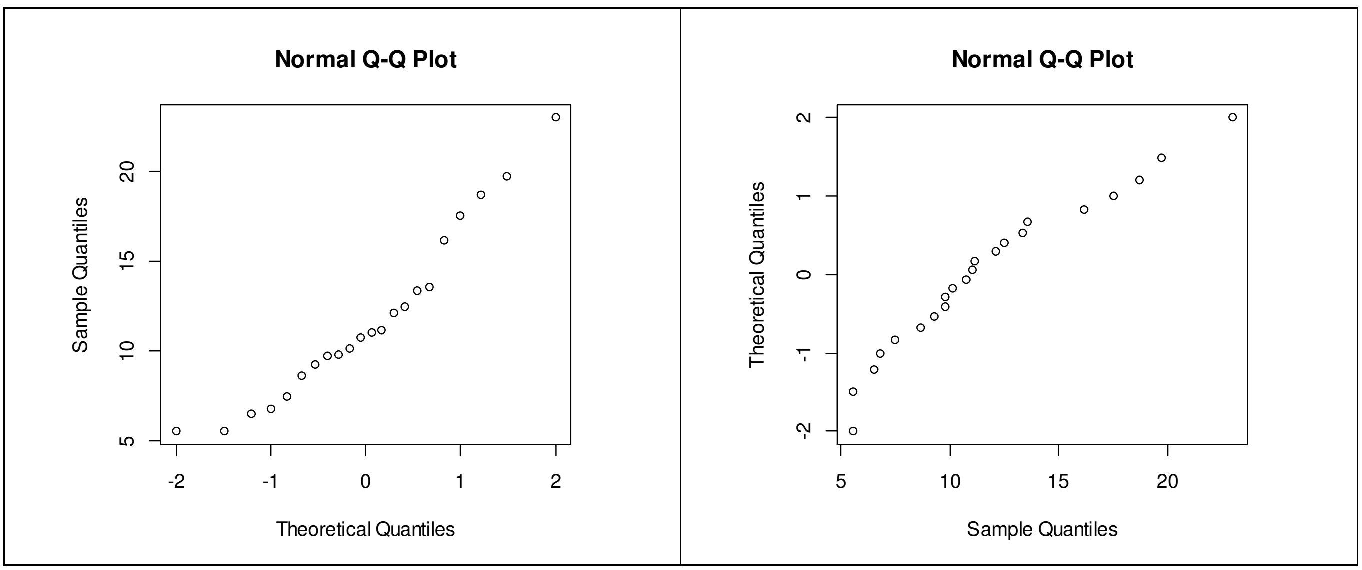
- To create a QQ plot based on the normal distribution, we use the `qqnorm()` function.
  - General format: `qqnorm(x)` where the vector x contains the data

- Example: Consider the weights of 22 randomly selected pumpkins. Create a QQ plot and determine if pumpkin weights are normally distributed (shown in *left* figure below).

```
> weights<-c(5.52,5.53,6.52,6.80,7.44,8.63,9.28,9.76,9.79,10.14,10.77,11.01,
+ 11.14,12.12,12.50,13.36,13.57,16.19,17.55,18.73,19.74,23.01)
> qqnorm(weights)
```

  - It is often the case with QQ plots that the data is plotted on the x-axis. To do this, you can use the `datax=TRUE` argument in the `qqnorm()` function (shown in *right* figure below).

```
> qqnorm(weights,datax=T)
```

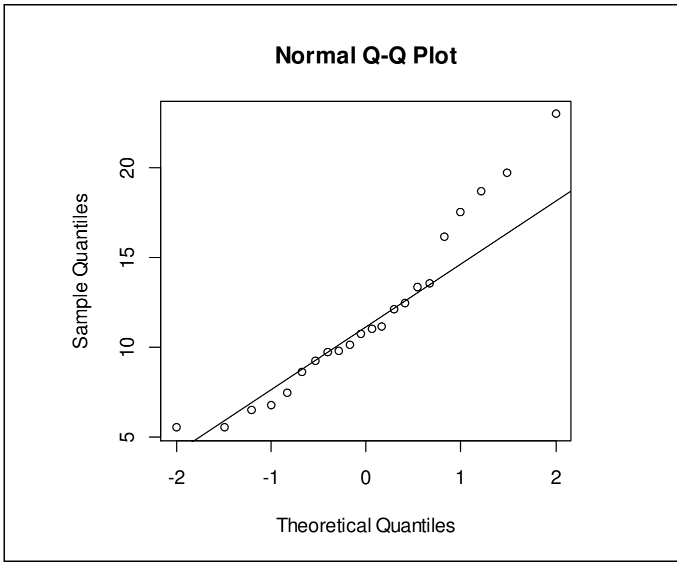


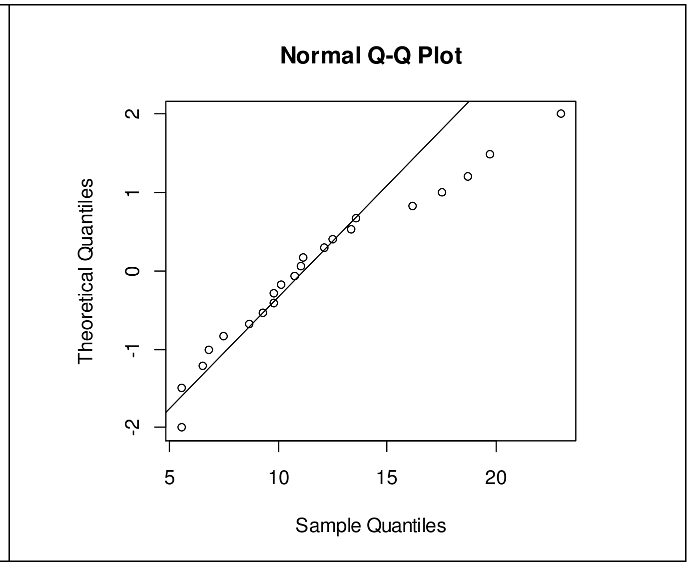
  - Since neither plot shows an approximately straight line, pumpkin weights ***cannot*** be assumed to be normally distributed.

  - If you'd like to add a straight line (for reference) to the plot, use the `qqline()` function. Make sure that if you used the `datax=T` argument in `qqnorm()`, you also use it in `qqline()`.

```
> qqnorm(weights)
> qqline(weights)          #left graph on next page

> qqnorm(weights,datax=T)
> qqline(weights,datax=T)  #right graph on next page
```


Normal Q-Q Plot
Sample Quantiles
Theoretical Quantiles
5
10
15
20
-2
-1
0
1
2


Normal Q-Q Plot
Theoretical Quantiles
Sample Quantiles
-2
-1
0
1
2
5
10
15
20