Course Content and Assignment 2 for Biom612 (Week 2) v2

Deanne Taylor, Pichai Raman, Joe Dybas, with some edits by Yuanchao Zhang

Jan 22nd, 2018

Contents

1 Starting Up

Packages to install this week:

- ggplot2 (CRAN)
- nortest (CRAN)
- pwr

Load data from last week:

```
tcga_luad <- read.csv("TCGA_LUAD_clinical.csv", na.string = "NA", stringsAsFactors = FALSE,
    row.names = 1)
```

2 The Normal Distribution

R provides several functions to allow you to generate normal distributions, for testing and for statistical use.

The Normal package in stats give several functions useful for utilizing statistics and observations from a normal distribution.

The four functions included with the normal distribution are:

- rnorm(n, mean = 0, sd = 1)
- dnorm(x, mean = 0, sd = 1, log = FALSE)
- pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
- qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)

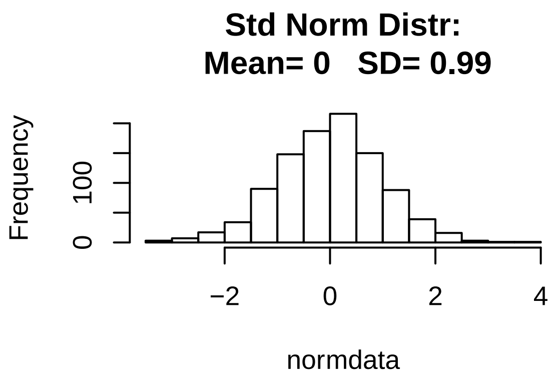
Note that all the default parameters for these functions assume the standard distribution ($\sigma = 1, \mu$=0$)

2.1 rnorm(): generating normal distribution data

You can use the help facility and type "?rnorm" to start reading about the Normal methods in the stats package.

Let's generate a normal distribution using rnorm().

```r
par(mfrow = c(2, 2))
# split your graphing window into a 2x2 (row,column). You may need to open up
# your plotting window a bit wider.

sampling <- 1000
# Pull observations from the standard normal distribution
normdata <- rnorm(sampling, mean = 0, sd = 1)

sdnorm <- sd(normdata)
meannorm <- mean(normdata)
hist(normdata, main = paste("Std Norm Distr:\n Mean=", round(meannorm, 2), "  SD=",
    round(sdnorm, 2)))

# re-run the lines above three more times (from sampling downwards) to fill up
# your 2x2 using sampling=10 repeatedly.  Something should be obvious to the eye
# in the histogram and in what you expect from the mean and standard deviation.

# At the end of your run, reset your graphics window, if you don't you'll be
# stuck with a 2x2 divide until you remove all plots from your plot window.

par(mfrow = c(1, 1))
```



Now, in the chunk above, increase the sampling variable to 100, then 1000. Rerun each 100 and 1000 four times like you did with sampling=10. This gives you a feel for the law of large numbers in operation.

Observe the sd() and mean() in the histogram titles when sampling=10, 100 and 1000. How does this relate to the expected mean and standard deviation in a standard normal distribution?

2.2 dnorm(): the probability density function on a normal distribution.

Recall that the dnorm() function listed at the top of this document is:

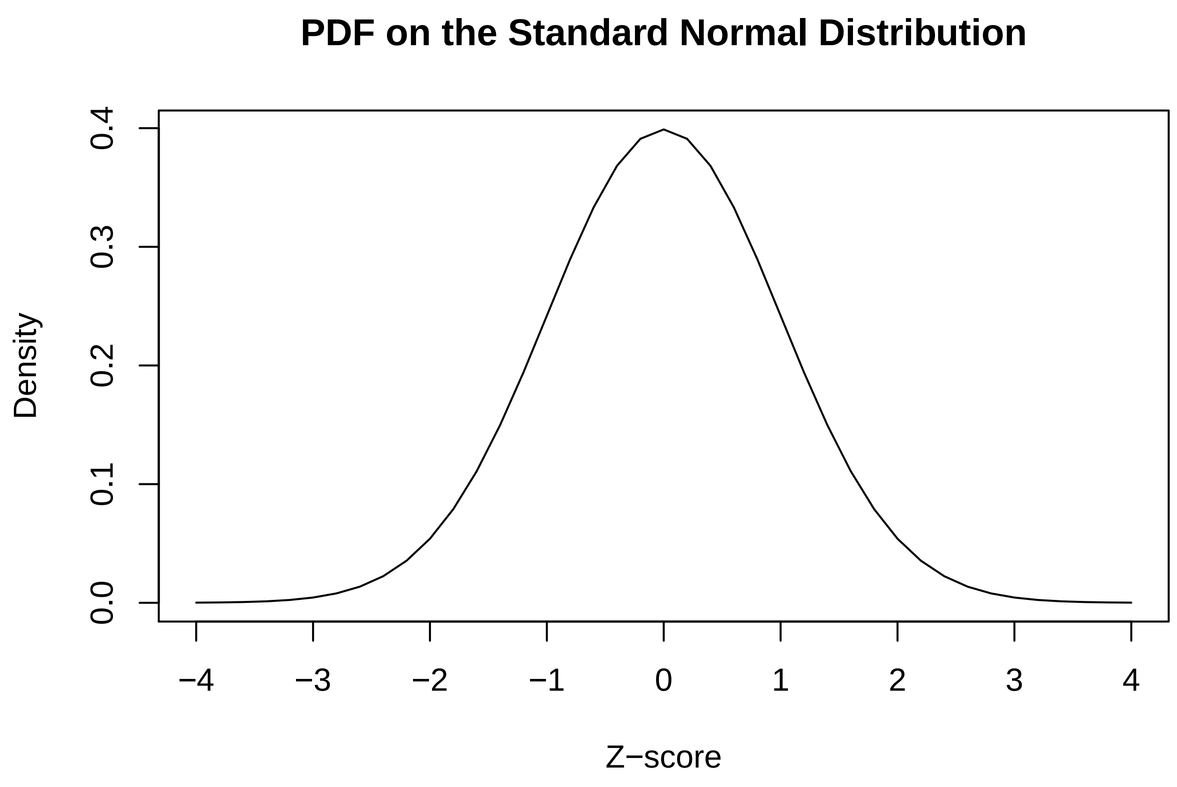
dnorm(x, mean = 0, sd = 1, log = FALSE)

The dnorm() function can be thought of an easy way to access the height of the normal curve at any point (x) along the curve.

This represents the probability density function (PDF), a function that represents the likelihood that the value of a random variable would be equal to a particular observation. First, if we only have a numerical observation on the distribution, we transform the value into a zscore, as the variable 'x' in the dnorm() function requires a z-score.

Let's take a look at how dnorm() works.

```
# Let's create a range of z-scores to test.
z_score_range <- seq(-4, 4, by = 0.2)  #Range of zscores to test using seq()

# Calculate the height of the curve at each location. These results are
# equivalent to the height of the density curve (or histogram) at each position.
# This also is equivalent of the 'probabilty density' at that point:
den_zscores <- dnorm(z_score_range)

# We plot without the axis, so we can put in tickmarks with equivalent z-scores
# on the density function.

plot(den_zscores, type = "l", main = "PDF on the Standard Normal Distribution", xlab = "Z-score",
    ylab = "Density", xaxt = "n")

den_zscore_sigmas <- c(dnorm(4), dnorm(3), dnorm(2), dnorm(1), dnorm(0), dnorm(1),
    dnorm(2), dnorm(3), dnorm(4))  #in order of the sigma scores.

den_score_labels <- c(-4, -3, -2, -1, 0, 1, 2, 3, 4)  #labels

# axis adds an axis of your choice to the plot.

axis(1, at = which(den_zscores %in% den_zscore_sigmas), labels = den_score_labels)
```



2.3 pnorm() and the Cumulative Distribution Function (CDF)

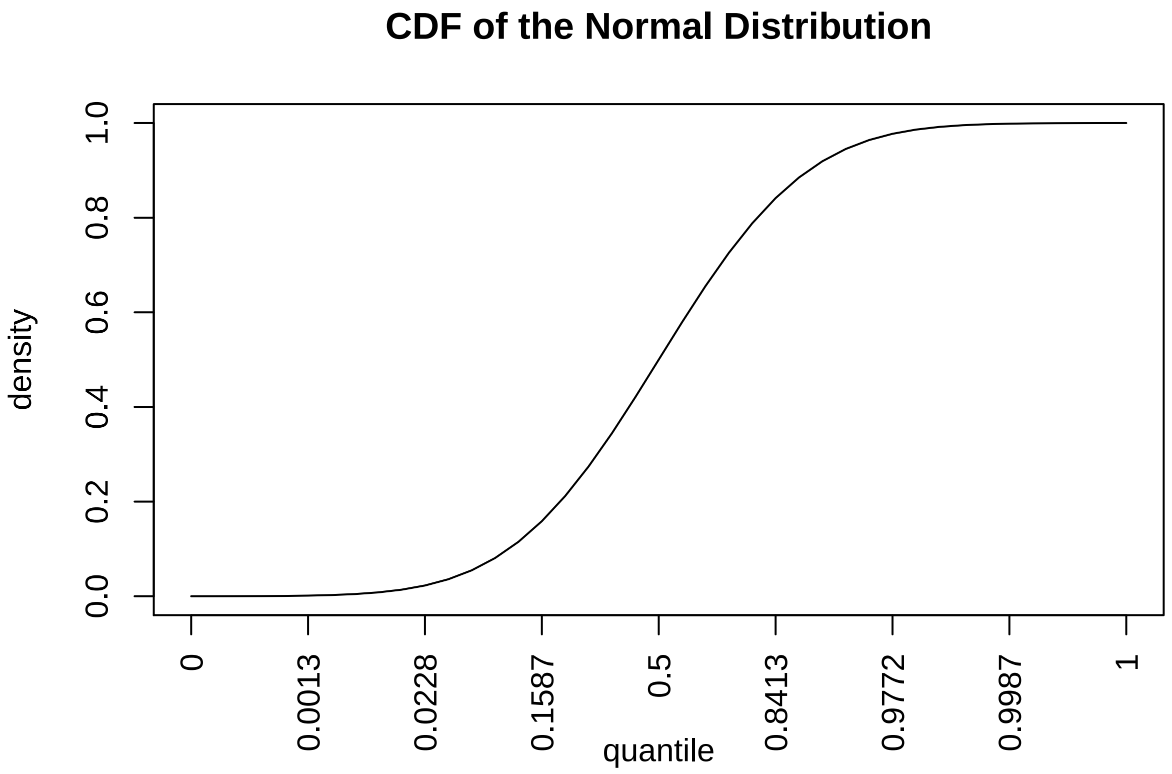
The function pnorm() provides the total amount of data available UP TO a certain z-score value representing a point on the normal distribution:

```
# How much data is represented if we integrate all data under the normal curve up
# to a z-score of 1.96? That is, what is the cumulative amount of data by the
# time we move through the normal curve to the Z-score of 1.96?

pnorm(1.96)
```

```
## [1] 0.9750021
```

```
# Does this make sense? Do you expect this value for the amount of data by the
# time you reach z-1.96 on the right of the plot? See Figure 3 in the course
# notes.

# using the z-scores calculated for dnorm() above, let's make a plot of the data
# accumulation along our selected points of the normal distribution:

pnorm_scores <- pnorm(z_score_range)

plot(pnorm_scores, type = "l", main = "CDF of the Normal Distribution", xlab = "quantile",
    ylab = "density", xaxt = "n")

pnorm_values <- c(pnorm(-4), pnorm(-3), pnorm(-2), pnorm(-1), pnorm(0), pnorm(1),
    pnorm(2), pnorm(3), pnorm(4))

# axis adds an axis of your choice to the plot. Bonus: vertical numbering with
# the las parameter.
```

```r
axis(1, at = which(pnorm_scores %in% pnorm_values), labels = round(pnorm_values,
    4), las = 3)
```



2.4 qnorm: Getting back the p-value from the cumulative density

Suppose you want to get the inverse of pnorm(): what z-score represents a probability of 0.99 that an event has occurred? You can use the qnorm() function for this.

```r
# what is the z-score where we have encountered the 99% percentile of the normal
# distribution?
qnorm(0.99)
```

```
## [1] 2.326348
```

```r
# How about the 99.99% percentile?
qnorm(0.9999)
```

```
## [1] 3.719016
```

```r
# what does lower.tail do?

qnorm(0.9999, lower.tail = FALSE)
```

```
## [1] -3.719016
```

```r
# let's try inverting pnorm and qnorm:

pnorm(qnorm(0.9999))
```

```
## [1] 0.9999
```

```
# we get back 0.9999.
```

2.5 Central Limit Theorem example

Central limit theorem states:

Let $X_1, X_2, ..., X_n$ be a sequence of identically distributed (i.i.d.) random variables with mean $E[X_i] = \mu$ and finite variance $Var(X_i) = \sigma^2$. Define

$$S_n = \frac{1}{n}\sum_i X_i$$

Then, as $n \to \infty$,

$$S_n \xrightarrow{\mathcal{D}} \mathcal{N}\left(\mu, \sigma^2/n\right)$$

Central limit theorem is difficult to prove algebraically, but it is quite easy to demonstrate with simulation.

Simulation procedure:

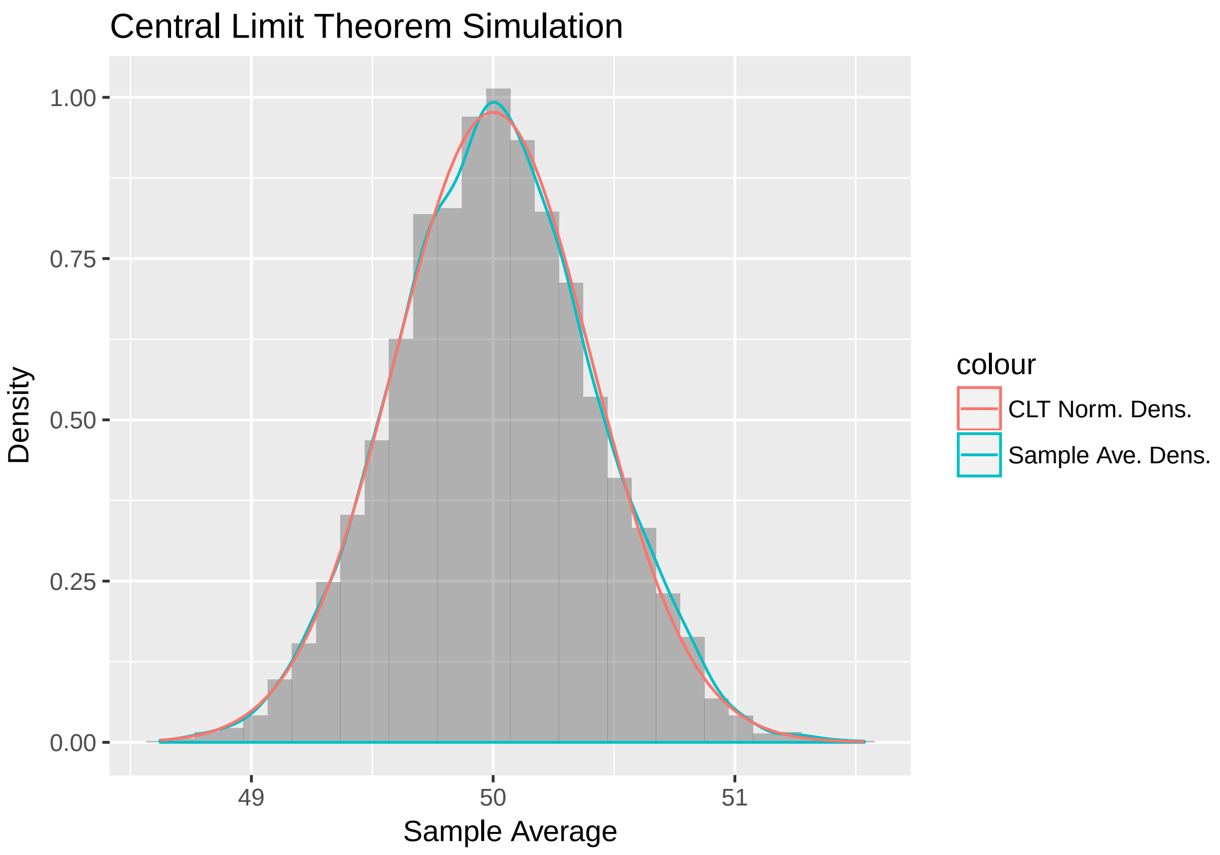
1. Let $X_1, X_2, ..., X_n$ be a sequence of i.i.d. uniformly distributed random variables.
2. Calculate sample everage S_n.
3. Repeat m times.
4. Plot the empirical PDF of S_n and normal PDF stated by the theorem.

```
# Let X1, X2, ..., X10 be a sequence of i.i.d. random variables uniformly
# distributed from 0 to 1.
runif(n = 10, min = 0, max = 1)
```

```
##  [1] 0.12169118 0.03253928 0.02867899 0.54709960 0.80169440 0.13809104
##  [7] 0.23243222 0.87570667 0.49938996 0.91393429
```

```
# We use "replicate"" to build a vector of length "number_of_samples" by repeatedly
# executing the following expression surrounded by {}.
# In the expression, we sum the binomial sampling of "rolls" times.
number_of_samples <- 5000
size_of_each_sample <- 5000
unif_min <- 0
unif_max <- 100
# Calculate the uniform distribution mean and variance using equations from
# https://en.wikipedia.org/wiki/Uniform_distribution_(continuous)
unif_mean <- (unif_max - unif_min) / 2
unif_sd <- (((unif_max - unif_min) ^ 2) / 12) ^ 0.5
clt_norm_mean <- unif_mean
clt_norm_sd <- unif_sd / (size_of_each_sample ^ 0.5)


sample_average_vector <- replicate(number_of_samples, {
  mean(runif(n = size_of_each_sample, min = unif_min, max = unif_max))
})

ggplot(data = data.frame(x = sample_average_vector), mapping = aes(x = x)) +
  geom_histogram(mapping = aes(y = ..density..), alpha = 0.4) +
  geom_density(mapping = aes(color = 'Sample Ave. Dens.')) +
  stat_function(fun = dnorm,
                # Parameters of the normal distribution specified by the central
                # limit theorem.
                args = list(mean=clt_norm_mean,
                            sd=clt_norm_sd),
                aes(colour = 'CLT Norm. Dens.')) +
  labs(x = 'Sample Average', y = 'Density') +
  ggtitle("Central Limit Theorem Simulation")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
# Feel free to change the "number_of_samples" and "size_of_each_sample" of the simulation.
```

3 Testing for normality

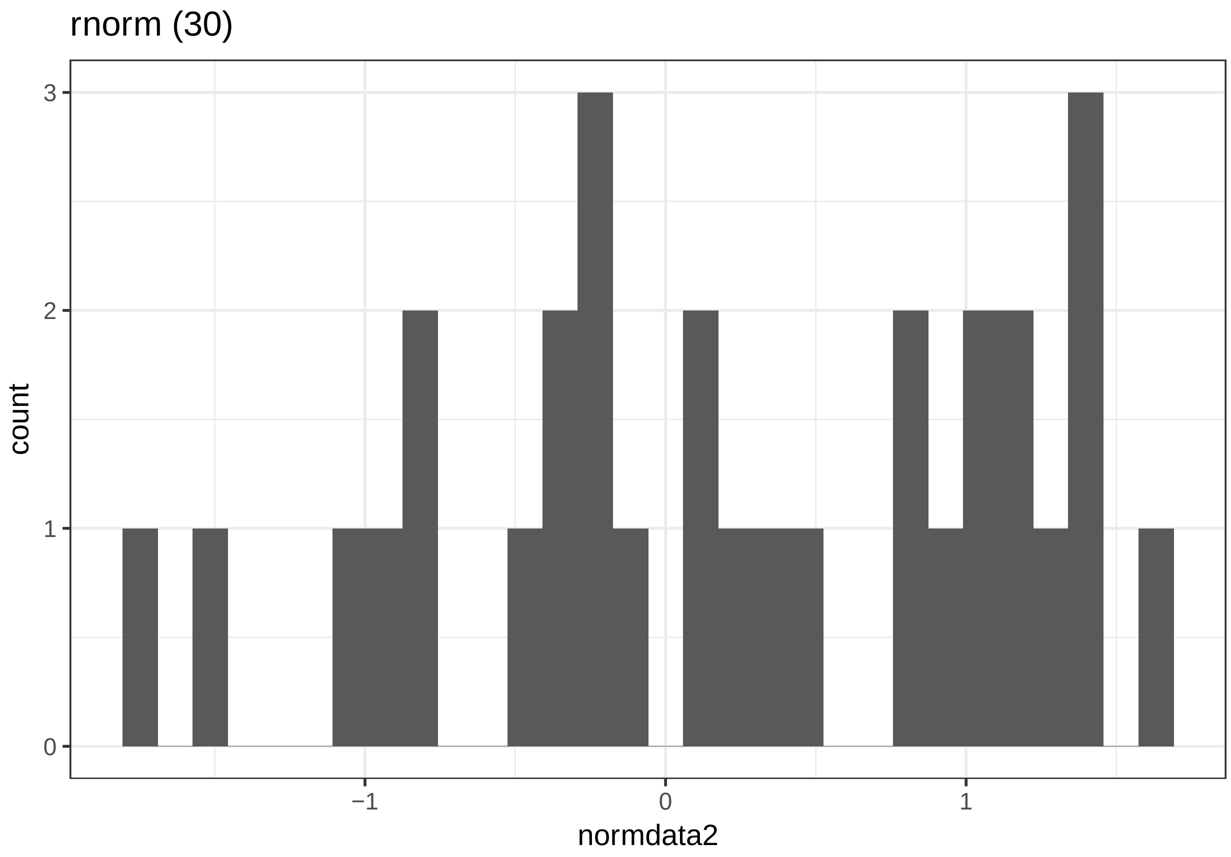
Important to do, as many statistical tests require that data being tested is normally distributed.

3.1 Plotting data, Quantile-quantile plots.

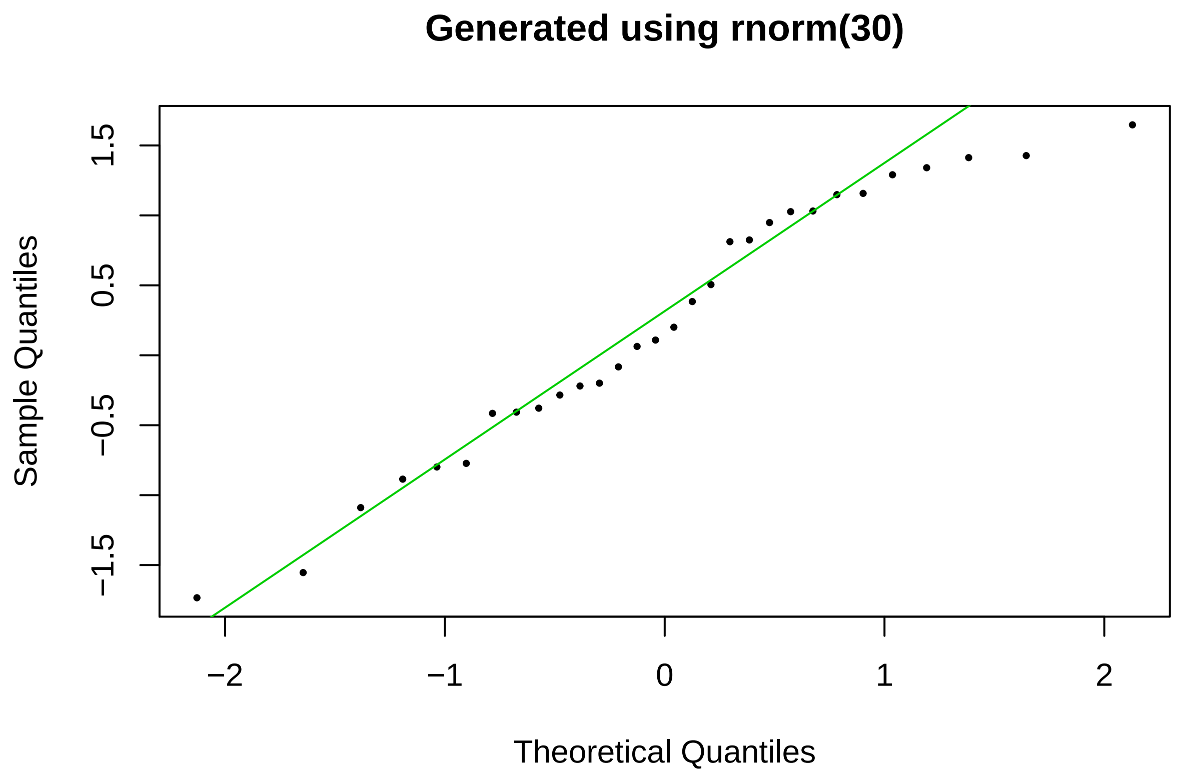
```
# Testing for normal: First we will test a qq-plot on a truly normal distribution
# using qqnorm()

num_points <- 30
normdata2 <- rnorm(num_points)

# Histogram using ggplot2. Another way to do it -- we discuss more about ggplot2
# later.
```

```r
qplot(normdata2) + geom_histogram() + theme_bw() + ggtitle(paste("rnorm (", num_points,
    ")", sep = ""))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

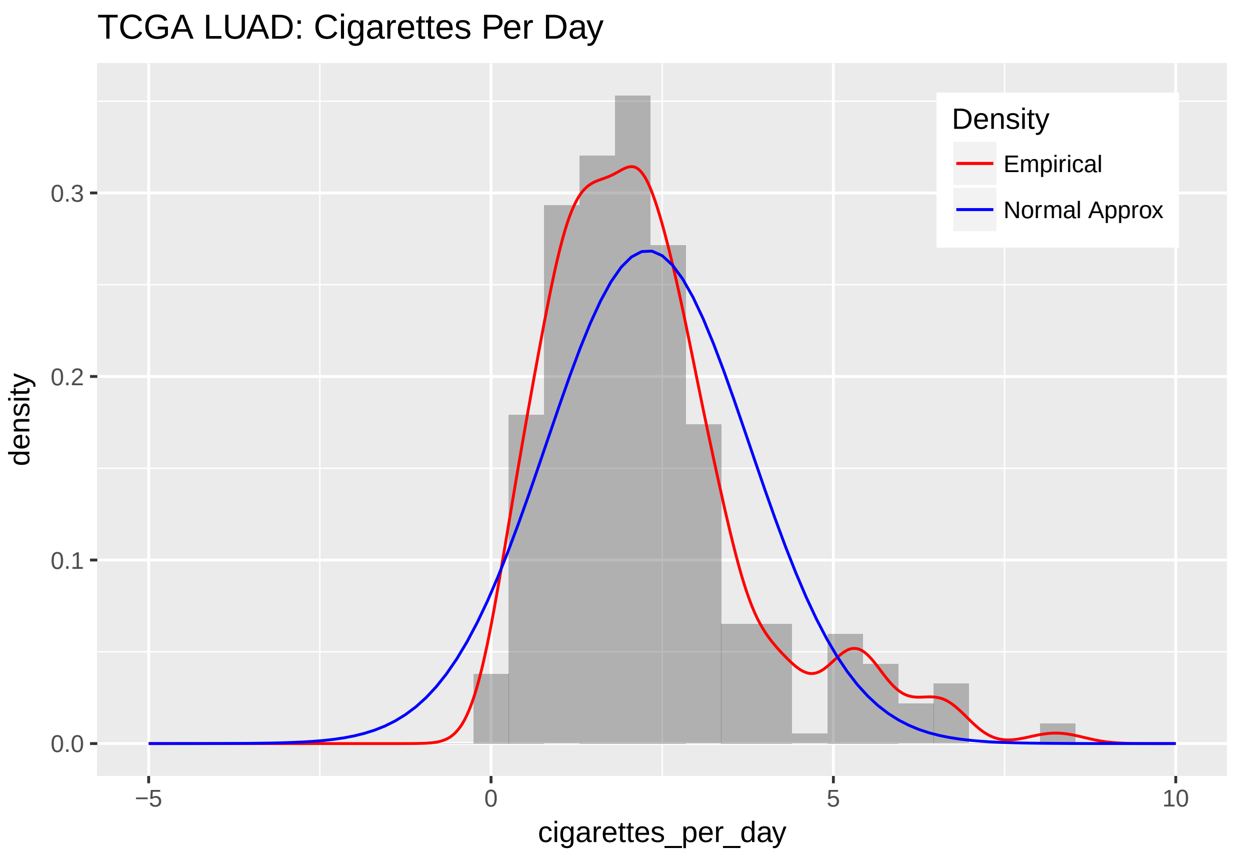


```r
# ignore any binwidth errors today.

# Let's look at this distribution in normdata2 versus an idealized normal
# distribution (theoretical quantiles) This is Q-Q plot:

qqnorm(normdata2, pch = 16, cex = 0.5, main = paste("Generated using rnorm(", num_points,
    ")", sep = ""))
qqline(normdata2, col = 3)
```



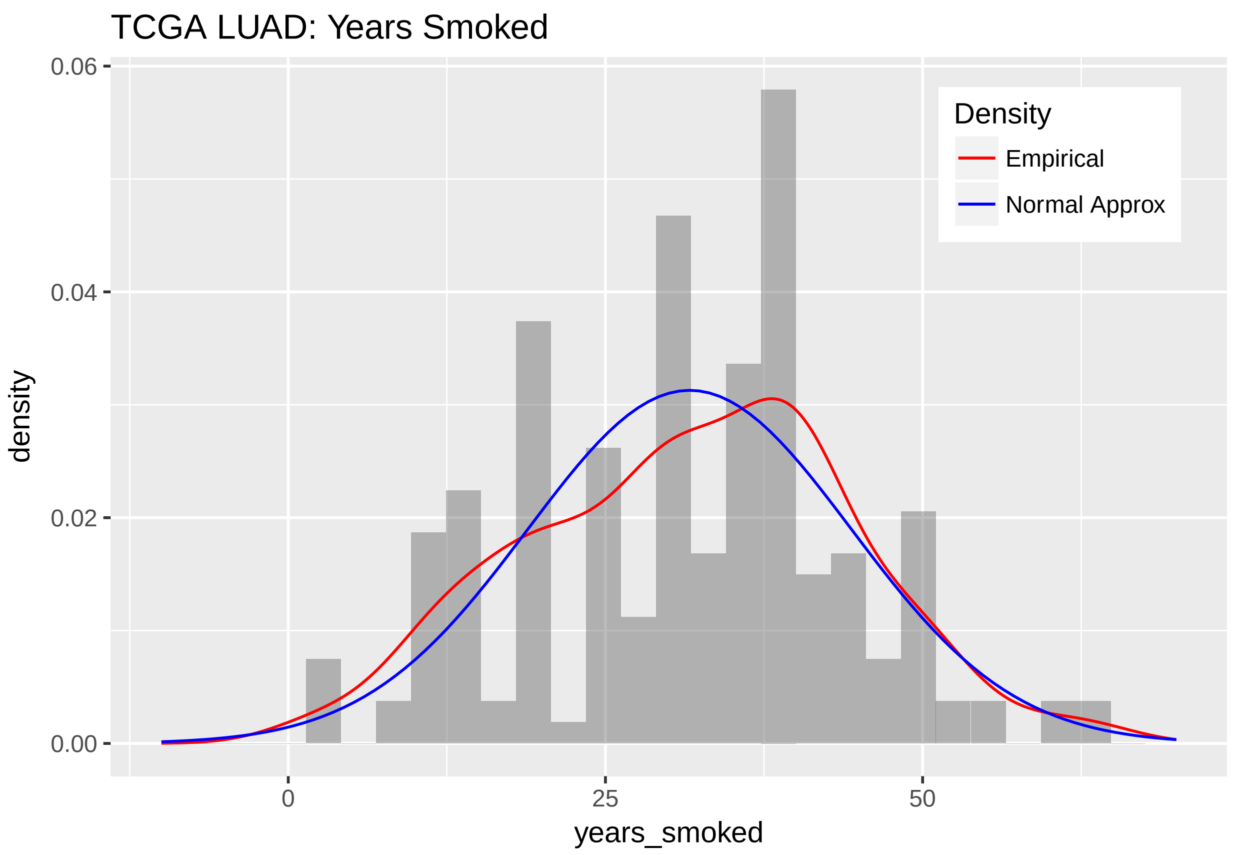
```r
# Now, above, increase the number of num_points to a much higher number and
# repeat. What do you observe?

# Let's use real data: Plotting the distributions from TCGA-LUAD from Week 1.
# Cigarettes per day. mean and sd:

luad_cpd_sd <- sd(tcga_luad$cigarettes_per_day, na.rm = TRUE)
luad_cpd_mean <- mean(tcga_luad$cigarettes_per_day, na.rm = TRUE)

# Using ggplot2, graphing the cigarettes per day -- first with the histogram,
# then overlay a blue normal approximation, then a red empirical density plot
# (same data as histogram). Note that each call to ggplot2 is separated by a '+'
# sign.  (Ignore warnings on non-finite values that come out -- this is normal
# for columns with missing data.)

qplot(x = cigarettes_per_day, xlim = c(-5, 10), data = tcga_luad, geom = "blank") +
    geom_histogram(aes(y = ..density..), alpha = 0.4) + geom_line(aes(y = ..density..,
    colour = "Empirical"), stat = "density") + stat_function(fun = dnorm, args = list(mean = luad_cpd_me
    sd = luad_cpd_sd), aes(colour = "Normal Approx")) + scale_colour_manual(name = "Density",
    values = c("red", "blue")) + theme(legend.position = c(0.85, 0.85)) + ggtitle("TCGA LUAD: Cigarettes
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 166 rows containing non-finite values (stat_bin).

## Warning: Removed 166 rows containing non-finite values (stat_density).

## Warning: Removed 1 rows containing missing values (geom_bar).
```

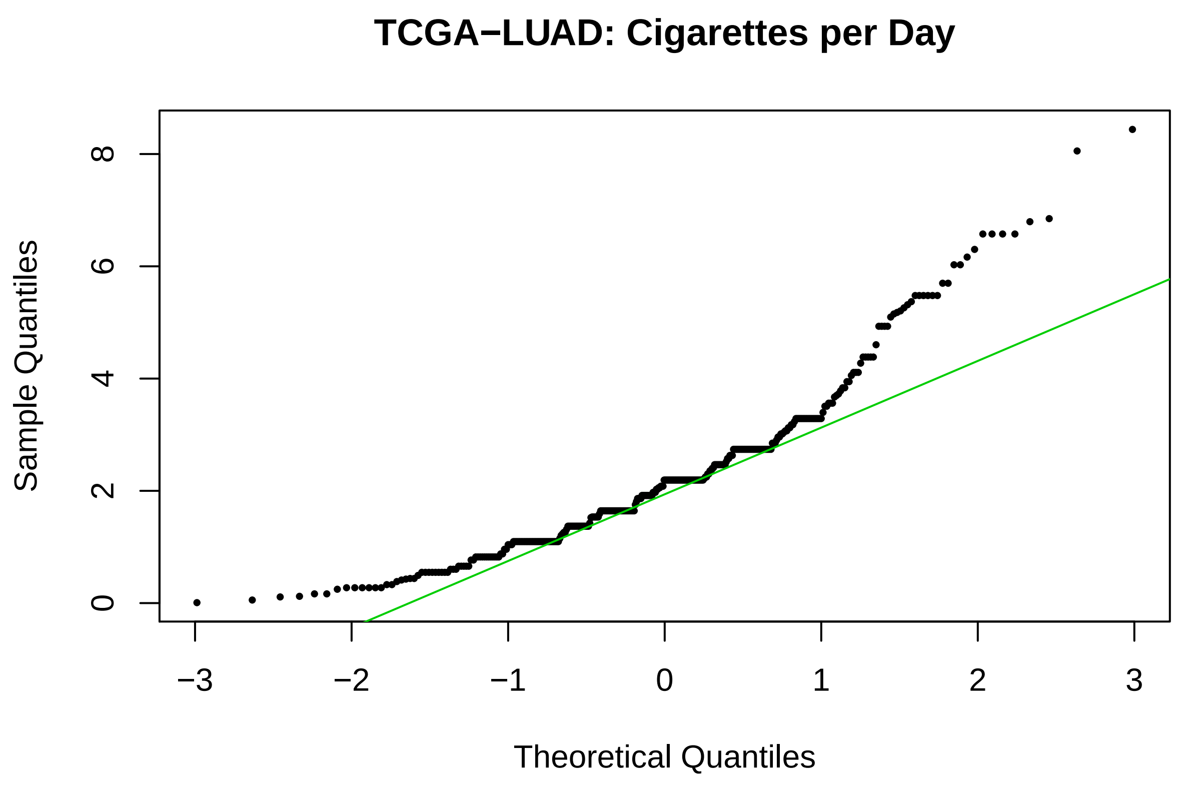


```r
# Repeat this for years_smoked below:
luad_ys_sd <- sd(tcga_luad$years_smoked, na.rm = TRUE)
luad_ys_mean <- mean(tcga_luad$years_smoked, na.rm = TRUE)

qplot(x = years_smoked, data = tcga_luad, xlim = c(-10, 70), geom = "blank") + geom_histogram(aes(y = .
    alpha = 0.4) + geom_line(aes(y = ..density.., colour = "Empirical"), stat = "density") +
    stat_function(fun = dnorm, args = list(mean = luad_ys_mean, sd = luad_ys_sd),
        aes(colour = "Normal Approx")) + scale_colour_manual(name = "Density", values = c("red",
    "blue")) + theme(legend.position = c(0.85, 0.85)) + ggtitle("TCGA LUAD: Years Smoked")
```

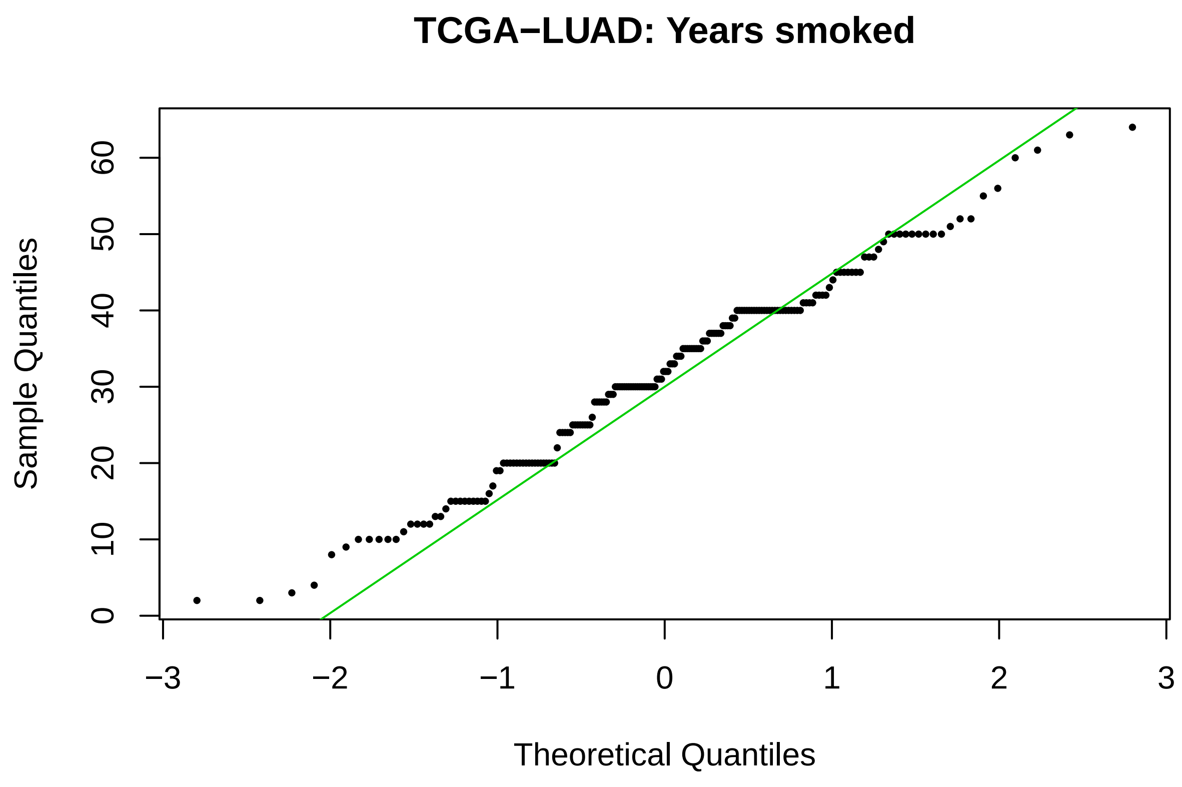
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## Warning: Removed 328 rows containing non-finite values (stat_bin).
## Warning: Removed 328 rows containing non-finite values (stat_density).
## Warning: Removed 1 rows containing missing values (geom_bar).
```



```r
# So basically both of these plots make the data look like they might be normal
# based on just the general distribution (blue ideal versus red empirical
# densities). Another helpful plot is the qqplot, where the approximate normal
# distribution is on the x-axis 'Theoretical Quantiles' and the y-axis shows the
# actual data organized by distribution as 'Sample Quantiles'.

qqnorm(tcga_luad$cigarettes_per_day, pch = 16, cex = 0.5, main = "TCGA-LUAD: Cigarettes per Day")

# now add a line approximating a linear 'fit' to the majority of the data:

qqline(tcga_luad$cigarettes_per_day, col = 3)
```



```
# Interpret the interesting shape of the points above the line -- check out the
# cigarettes_per_day histogram.

qqnorm(tcga_luad$years_smoked, pch = 16, cex = 0.5, main = "TCGA-LUAD: Years smoked")
qqline(tcga_luad$years_smoked, col = 3)  #adds a trendline
```



In the qqplots (qqnorm), you can now clearly note that the cigarettes per day distribution deviates quite a bit from normal whereas the “years smoked”" distribution is probably more normally distributed. You can tell this by how far the sample and theoretical quantiles deviate from one-another (as evidenced by how far off the line they are). However, even if you see some points far off the line they are NOT outliers. A qqplot does not determine outliers! There are other methods for that, which we will discuss later in the semester.

3.2 Statistical tests of normality

Plots are all well and good, but what is the statistical evidence for deviations from normality?

The next thing you may do is perform a statistical test to measure the deviation from normal.

At this point you will download the package ‘nortest’ so you can run some of tests in this package. Then it is quite straightforward to run the test

```
# Testing the theoretical normal distribution

nortest_points <- 100
normdata_nortest <- rnorm(nortest_points)

# Sharpio-Wilk test:
shapiro.test(normdata_nortest)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  normdata_nortest
## W = 0.98923, p-value = 0.603
```

```
# Let's run the Anderson-darling test on normal data:
ad.test(normdata_nortest)
```

```
##
##  Anderson-Darling normality test
##
## data:  normdata_nortest
## A = 0.35523, p-value = 0.4534
```

```
shapiro.test(tcga_luad$cigarettes_per_day)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  tcga_luad$cigarettes_per_day
## W = 0.90998, p-value = 9.873e-14
```

```
shapiro.test(tcga_luad$years_smoked)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  tcga_luad$years_smoked
## W = 0.98572, p-value = 0.04686
```

This test verifies what was gleaned from the QQ-plot.

3.3 One-sample tests of proportion

As we learned in the lecture, the null hypothesis of a one-sided test of proportion can be rejected if z significantly differs from z0 in the chosen tail, where z0 is z-value obtained from the 100(1 - α) percentile of the standard normal distribution.

Let's take an example:

Suppose we are studying 'heavy smokers' – people who smoke more than 5 cigarettes per day.

Suppose we observe that a proportion – approximately 10% – of patients from the TCGA lung adenocarcinoma study smoked more than 5 cigarettes per day.

A new study is performed where 22 out of 200 patients smoked more than 5 cigarettes per day. We would like to check if this group of patients has a statistically greater proportion of heavy smokers compared to the TCGA study.

At the α=.05 significance level, what is the strength of the evidence that that the proportion of heavy smokers in the new study (p) is greater than the proportion of heavy smokers in the TCGA lung adenocarcinoma study (p_0) ? This is a right-tailed test (see Table 2.3 from your assigned reading in Sabo and Boone), so to reject the null hypothesis, we will test if $z \geq z_0$ (value of z greater than the critical value, z_0)

```
# What is the critical value of Z for an alpha of 0.05? See also sections 2.6.1
# and 2.7 in Sabo and Boone

# ONE TAILED TEST: Greater Than.
alpha = 0.05
z0 = qnorm(1 - alpha)  #critical value on one tail.
print(z0)  #what is it?
```

```
## [1] 1.644854
```

```
# What is the test statistic (z) for proportion #2?
prop2 = 22/200  # second study patient proportion
p0 = 0.1  # TCGA LUAD proportion
n = 200  # second study sample size

# eq 2.2 in Sabo/Boone:
z = (prop2 - p0)/sqrt(p0 * (1 - p0)/n)
print(z)
```

```
## [1] 0.4714045
```

```
# Note that when prop2 goes lower, say 13/200, z is negative. What does this
# mean? Does this impact your interpretation at all?

# Do we support the null hypothesis that the second study has a significantly
# greater number of heavy smokers given an alpha of 0.05? If the null hypothesis
# is upheld, then TRUE, if not, then FALSE.
z >= z0
```

```
## [1] FALSE
```

```
# Change the numbers, now. Move to proportions in the new study being (say)
# 18/200,20/200, 50/200...is it what you expect?
```

3.4 Using prop.test

Our other alternative is to use a χ^2 based proportion test built into R, called prop.test(). It does not use the same methods as the z statistic (equation 2.2 in Sabo and Boone). We discuss χ^2 statistics later in the semester.

```r
# See sections 2.6.1 and 2.7 in Sabo and Boone for this background.

alpha = 0.05
z0 = qnorm(1 - alpha)
print(z0)  #critical value we're testing against.
```

```
## [1] 1.644854
```

```r
# What is our test statistic (z) for our test?
hs = 22  # second study heavy smokers
p0 = 0.1  # TCGA LUAD proportion
n2 = 200  #Number of tests

propresult <- prop.test(hs, n2, p = p0, correct = FALSE, alternative = "greater")
# no Yates correction, select for p>=p0

print(propresult)
```

```
##
##  1-sample proportions test without continuity correction
##
## data:  hs out of n2, null probability p0
## X-squared = 0.22222, df = 1, p-value = 0.3187
## alternative hypothesis: true p is greater than 0.1
## 95 percent confidence interval:
##  0.0786844 1.0000000
## sample estimates:
##    p
## 0.11
```

```r
# Comparing prop.test to z-score method.
Xstat <- propresult$statistic[[1]]  # X-statistic
print(sqrt(Xstat))  #for comparison to z
```

```
## [1] 0.4714045
```

```r
# Is prop.test's X-squared statistics close to the z-statisic?
z = (hs/n2 - p0)/sqrt(p0 * (1 - p0)/n2)
print(abs(z))
```

```
## [1] 0.4714045
```

3.5 One sided test of proportions, single tail

```r
# See sections 2.6.1 and 2.7 in Sabo and Boone for this background.

hs = 22  # second study heavy smokers
p0 = 0.1  # TCGA LUAD proportion
n2 = 200  #Number of tests
```

```r
propresult <- prop.test(hs, n2, p = p0, correct = FALSE, alternative = "greater")
# no Yates correction

print(propresult)
```

```
##
##  1-sample proportions test without continuity correction
##
## data:  hs out of n2, null probability p0
## X-squared = 0.22222, df = 1, p-value = 0.3187
## alternative hypothesis: true p is greater than 0.1
## 95 percent confidence interval:
##  0.0786844 1.0000000
## sample estimates:
##    p
## 0.11
```

```r
# Comparing prop.test to z-score method.
Xstat <- propresult$statistic[[1]]  # X-statistic
print(sqrt(Xstat))  #for comparison to z
```

```
## [1] 0.4714045
```

```r
# Is prop.test's X-squared statistics close to the z-statisic?
z = (hs/n2 - p0)/sqrt(p0 * (1 - p0)/n2)
print(abs(z))
```

```
## [1] 0.4714045
```

```r
# Probability of difference (of rejecting the null hypothesis) can be calculated
# with a p-value derived from the Z-score. We use the pnorm() function. However,
# in pnorm(), avoid any problems with the sign of the z-score we use the negative
# absolute value of the z-score.

# See Figure 3 in the Week 2 Course Notes for the relationship between the Z
# score and the normal distribution, and where alpha=0.05, z=1.96

# In the case of a one-sided test:
pvalue_z_onesided <- pnorm(-abs(z))
pvalue_z_onesided
```

```
## [1] 0.3186759
```

```r
# and from the prop.test:
propresult$p.value
```

```
## [1] 0.3186759
```

```r
# both are identical, though prop.test does use a different method.
```

3.6 One sided test of proportions, two-tailed

```r
# this is the Z-statistic test for the two-tailed case.

alpha = 0.05
```

```
z0 = qnorm(1 - alpha/2)
print(z0)  #critical value we're testing against.
```

```
## [1] 1.959964
```

```
# TWO-TAILED TEST for Z statistic

hs = 30  # second study patient proportion
p0 = 0.1  # TCGA LUAD proportion
n2 = 200  # second study sample size

# eq 2.2 in Sabo/Boone:
z_twotail = (hs/n2 - p0)/sqrt(p0 * (1 - p0)/n)

abs(z_twotail) <= z0  # false==reject null of no difference
```

```
## [1] FALSE
```

```
# Two-tailed p-value: must multiply the equivalent single-tail probability by 2.
# This p-value is 1-P where P is the probability that the null can be rejected.
# Thus the smaller the p-value, the greater the probability of rejecting the
# null.

pval_z_twotail = 2 * pnorm(z_twotail, lower.tail = FALSE)
# pvalue from the two-tailed z-score analysis:
pval_z_twotail
```

```
## [1] 0.01842213
```

```
# For both two sample and one sample proportion tests, you can specify
# alternative='two.sided', 'less', or 'greater' to indicate a two-tailed, or
# one-tailed test. A two sided test is the default.

proptest_twotail <- prop.test(hs, n2, p = p0, correct = FALSE, alternative = "two.sided")  #no Yates co
# pvalue from the two-tail proptest analysis:
proptest_twotail$p.value
```

```
## [1] 0.01842213
```

4 Power analysis for one-sample test of proportion

There are four main elements in calculating the power of a test in no particular order:

1. Effect size, or the difference in means between two groups
2. Sample size: N
3. Significance, as the acceptance probability of false positives, which you have learned is α (often 0.05)
4. Power, which is defined as the ability to detect a true positive, which is also 1-P where P = probability of a false negative. That is, how well can we actually detect the true effect?

Usually we calculate one of the four by using the other three.

In the coming weeks we will be touching on the concept of power several times. The power of an experiment is the ability to detect an effect of a given size with certain degree of confidence. It can also help us calculate the number of samples needed to perform an experiment given other parameters.

For now, we'll use these measures to calculate the power of a one-sample test of proportion using the pwr package in R.

```r
# see also
# https://cran.r-project.org/web/packages/pwr/vignettes/pwr-vignette.html

# h=effect size. For now use 0.5 which is a 'medium' effect size. We discuss next
# week.  See https://en.wikipedia.org/wiki/Cohen%27s_h

# here we use the function pwr.p.test(). We must leave the one we wish to
# calculate set to NULL or just leave it out of the parameters entirely.

# what's the power of the test? The 'ideal' for power is somewhere > 0.90.

pwr.p.test(h = 0.5, n = 55, sig.level = 0.05, power = NULL)  #calculate power
```

```
##
##      proportion power calculation for binomial distribution (arcsine transformation)
##
##               h = 0.5
##               n = 55
##       sig.level = 0.05
##           power = 0.9597797
##     alternative = two.sided
```

```r
# or just leave power out of the parameters, which defaults 'power' to NULL:
pwr.p.test(h = 0.5, n = 55, sig.level = 0.05)
```

```
##
##      proportion power calculation for binomial distribution (arcsine transformation)
##
##               h = 0.5
##               n = 55
##       sig.level = 0.05
##           power = 0.9597797
##     alternative = two.sided
```

```r
pwr.p.test(h = 0.5, n = NULL, sig.level = 0.05, power = 0.9)  #how many samples would you need for this
```

```
##
##      proportion power calculation for binomial distribution (arcsine transformation)
##
##               h = 0.5
##               n = 42.02968
##       sig.level = 0.05
##           power = 0.9
##     alternative = two.sided
```

```r
pwr.p.test(h = NULL, n = 100, sig.level = 0.05, power = 0.9)  #What effect size could you calculate wit
```

```
##
##      proportion power calculation for binomial distribution (arcsine transformation)
##
##               h = 0.3241514
##               n = 100
##       sig.level = 0.05
##           power = 0.9
##     alternative = two.sided
```

```r
# In the pwr package functions, one function is ES.h() which allows us to
# establish an approximate effect size h for our test. The vignette for pwr (link
# above) gives an example of a one-sided test of proportion as follows:

# How many times should we flip the coin to have a high probability (or power),
# say 0.80, of correctly rejecting the null of a fair coin (probability of heads
# = 0.5) if our coin is indeed loaded instead to land heads 75% of the time? Note
# that in this function, the larger number of the two (p1, p2) must be p1.

pwr.p.test(h = ES.h(p1 = 0.75, p2 = 0.5), sig.level = 0.05, n = NULL, power = 0.8,
    alternative = "greater")
```

```
##
##      proportion power calculation for binomial distribution (arcsine transformation)
##
##               h = 0.5235988
##               n = 22.55126
##       sig.level = 0.05
##           power = 0.8
##     alternative = greater
```

```r
# How many times should we flip the coin to have a high probability (or power),
# say 0.95, of correctly rejecting the null of a fair coin (probability of heads
# = 0.5) if our coin is indeed loaded instead to land heads 51% of the time?
# 50.5% of the time?

pwr.p.test(h = ES.h(p1 = 0.51, p2 = 0.5), sig.level = 0.05, n = NULL, power = 0.95,
    alternative = "greater")
```

```
##
##      proportion power calculation for binomial distribution (arcsine transformation)
##
##               h = 0.02000133
##               n = 27051.83
##       sig.level = 0.05
##           power = 0.95
##     alternative = greater
```

Assignment 2

Create a new R Markdown file for submitting this assignment. You should submit both your knit file in HTML and the original Rmd file to the Week 1 homework submission link in CANVAS. This homework is due 1/29/18 at 11:59 PM (evening).

For this assignment, you will be Using the TCGA data provided for you in the Week 2 Module:

TCGA_LUAD TCGA_HNSC TCGA_SCKM

For this assignment, please produce the markdown file and your knit file (pdf, html or doc), by submitting your files in the Assignment 2 link in the Week 2 Module.

Using the TCGA data, please answer the following:

Question 1

Using Q-Q plots and tests for normality, determine if the height of patients from the TCGA_LIHC and TCGA_SKCM data sets are normally distributed.

Question 2

For patients noted as smoking more than 40 years (years_smoked), is the proportion of TCGA_HNSC patients significantly different than the proportion of 40+ year smoking patients from the TCGA_LUAD patients? Show your calculations with comments.

Question 3

What is the probability that the null model on a normal distribution can explain the difference in proportions between patients with height greater than 170 in the TCGA_SKCM data and TCGA_LIHC?

Question 4

How many samples would you need from TCGA_SKCM to detect a difference in proportions between patients with height greater than 170 as compared to those with height > 170 TCGA_LIHC with 95% power and α=0.05?