The normal distribution

In this lab, you'll investigate the probability distribution that is most central to statistics: the normal distribution. If you are confident that your data are nearly normal, that opens the door to many powerful statistical methods. Here we'll use the graphical tools of R to assess the normality of our data and also learn how to generate random numbers from a normal distribution.

Getting Started

Load packages

In this lab, we will explore and visualize the data using the **tidyverse** suite of packages as well as the **openintro** package.

Let's load the packages.

```
library(tidyverse)
library(openintro)
```

The data

This week you'll be working with fast food data. This data set contains data on 515 menu items from some of the most popular fast food restaurants worldwide. Let's take a quick peek at the first few rows of the data.

Either you can use `glimpse` like before, or `head` to do this.

```
library(tidyverse)
library(openintro)
data("fastfood", package='openintro')
head(fastfood)
```

```
## # A tibble: 6 x 17
##   restaurant item       calories cal_fat total_fat sat_fat trans_fat cholesterol
##   <chr>      <chr>         <dbl>   <dbl>     <dbl>   <dbl>     <dbl>       <dbl>
## 1 Mcdonalds  Artisan G~      380      60         7       2       0            95
## 2 Mcdonalds  Single Ba~      840     410        45      17       1.5         130
## 3 Mcdonalds  Double Ba~     1130     600        67      27       3           220
## 4 Mcdonalds  Grilled B~      750     280        31      10       0.5         155
## 5 Mcdonalds  Crispy Ba~      920     410        45      12       0.5         120
## 6 Mcdonalds  Big Mac         540     250        28      10       1            80
## # i 9 more variables: sodium <dbl>, total_carb <dbl>, fiber <dbl>, sugar <dbl>,
## #   protein <dbl>, vit_a <dbl>, vit_c <dbl>, calcium <dbl>, salad <chr>
```

You'll see that for every observation there are 17 measurements, many of which are nutritional facts.

You'll be focusing on just three columns to get started: restaurant, calories, calories from fat.

Let's first focus on just products from McDonalds and Dairy Queen.

```r
mcdonalds <- fastfood %>%
  filter(restaurant == "Mcdonalds")
dairy_queen <- fastfood %>%
  filter(restaurant == "Dairy Queen")
```

1. Make a plot (or plots) to visualize the distributions of the amount of calories from fat of the options from these two restaurants. How do their centers, shapes, and spreads compare?

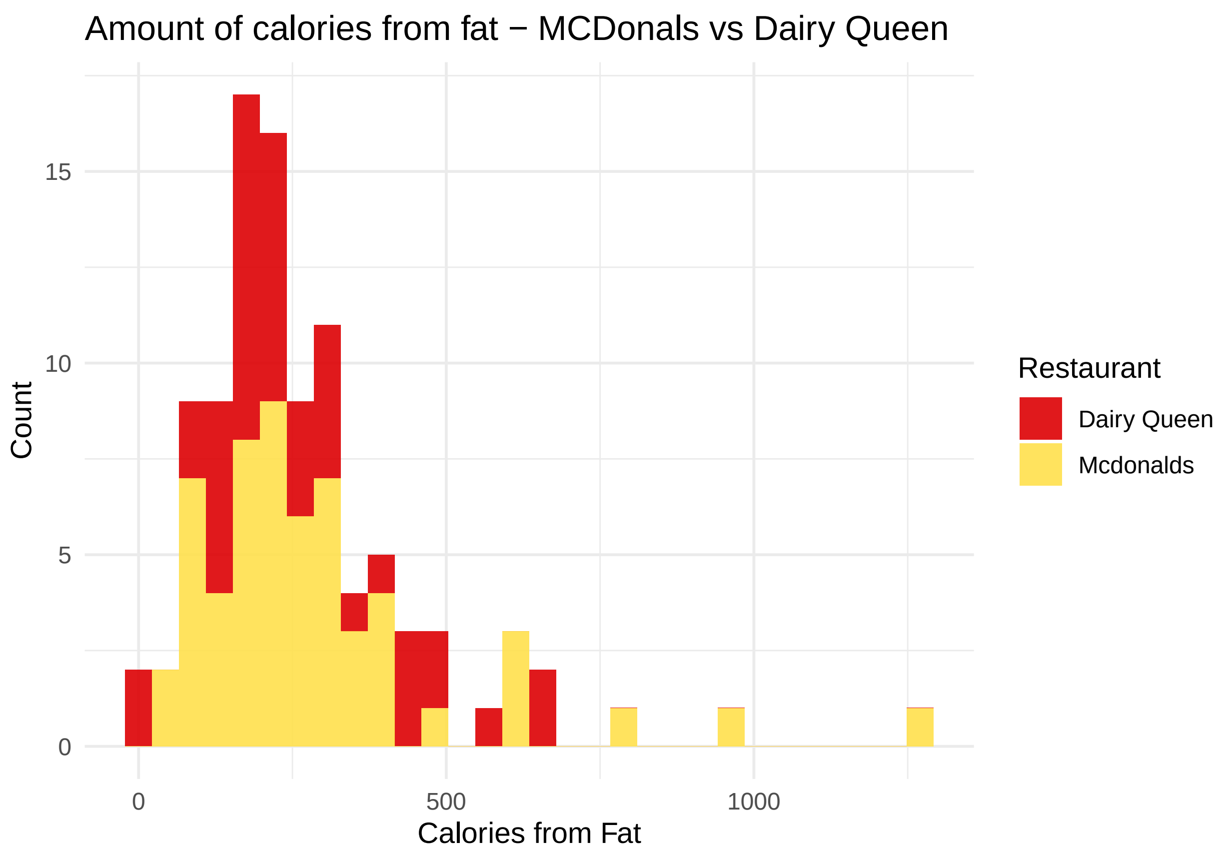
Insert your answer here

For McDonald we got a mean of 285.6 (dashed line), a median of 240 (straight line) and a standard deviation of 221. As observed in plot "Amount of calories from Fat - MCDonals", we can determinate that the shape of distribution is Skewed right because of the tail pulled toward to the right.

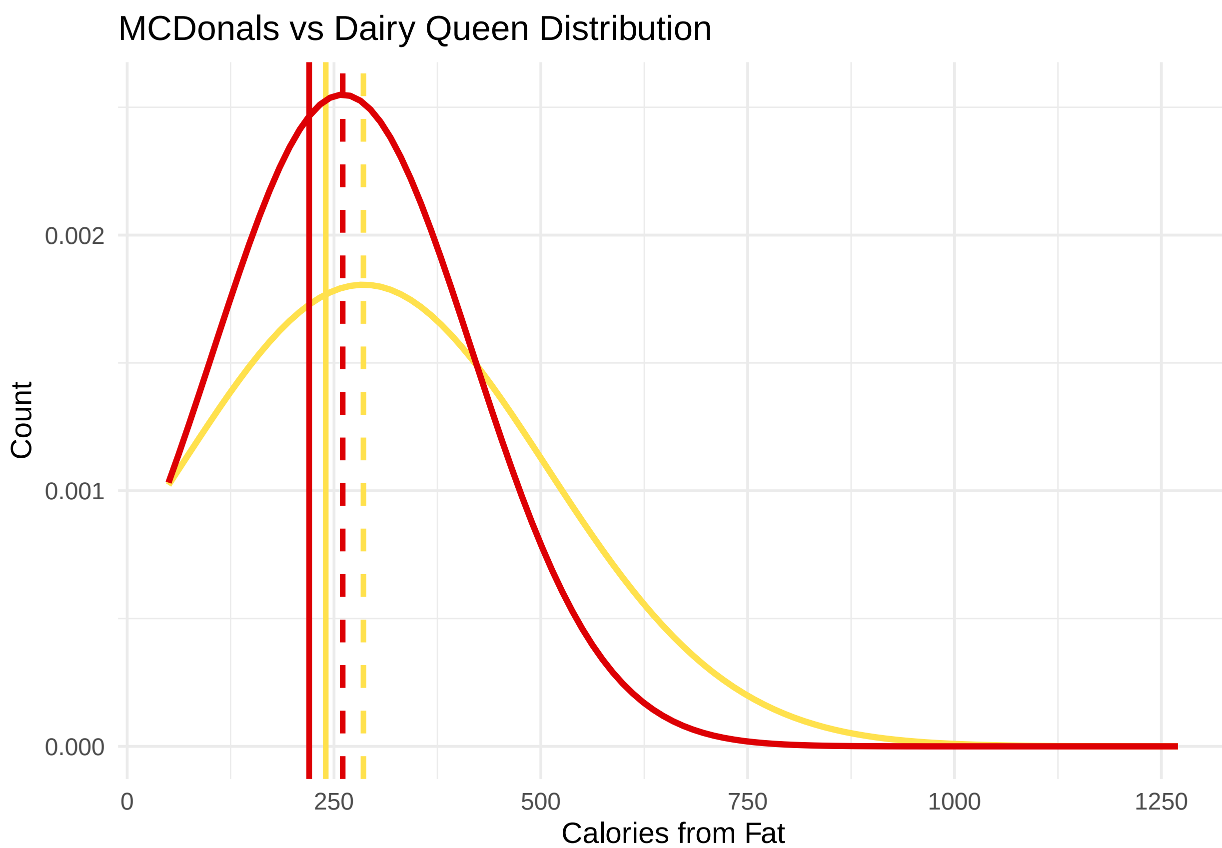
For Dairy Queen data set we got a mean of 260.5 (dashed line), a median of 220 (straight line) and a standard deviation of 156. As observed in plot "Amount of calories from Fat - Dairy Queen", the distribution is skewed right.

As displayed in plot "McDonals vs Dairy Queen Distribution", the data set for MCDonalds (yellow) has a higher standard deviation than Dairy Queen (red) distribution.

```r
#mcdonal vs Dairy Queen
fastfood_comp <- fastfood |>
            filter(restaurant == "Mcdonalds" | restaurant == "Dairy Queen")
ggplot(data = fastfood_comp, aes(fill=restaurant,x = cal_fat))+
  geom_histogram(alpha = 0.9)+
  scale_fill_manual('Restaurant', values=c('#DD0004','#FFE14D'))+
  ylab('Count')+
  xlab('Calories from Fat')+
  ggtitle('Amount of calories from fat - MCDonals vs Dairy Queen') +
  theme_minimal()
```



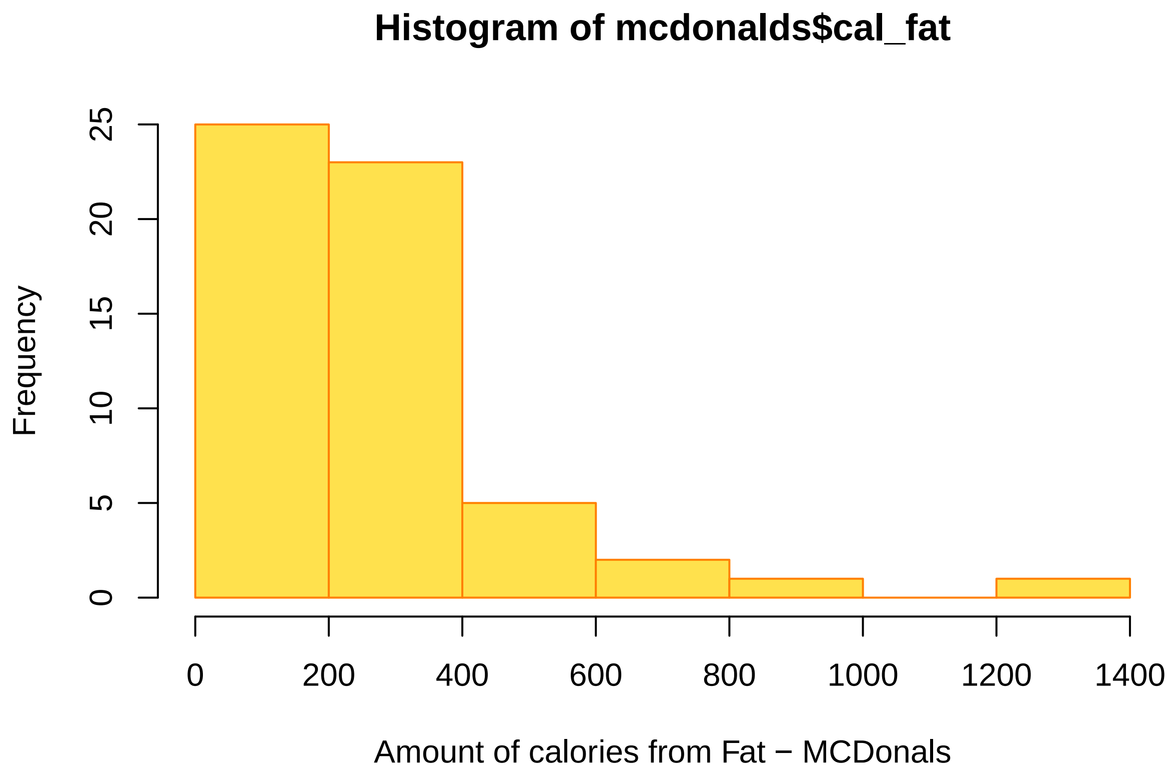

```r
#mcdonal vs Dairy Queen Distribution
ggplot(data = mcdonalds, aes(x = cal_fat))+
  stat_function(fun = dnorm,
                args = c(mean = mean(mcdonalds$cal_fat),
                         sd = sd(mcdonalds$cal_fat)), col = "#FFE14D", size=1.1)+
  geom_vline(xintercept=mean(mcdonalds$cal_fat), linetype="dashed", color = "#FFE14D", size=1)+
  geom_vline(xintercept=median(mcdonalds$cal_fat),  color = "#FFE14D", size=1)+
  stat_function(fun = dnorm,
                args = c(mean = mean(dairy_queen$cal_fat),
                         sd = sd(dairy_queen$cal_fat)), col = "#DD0004", size=1.1)+
  geom_vline(xintercept=mean(dairy_queen$cal_fat), linetype="dashed", color = "#DD0004", size=1)+
  geom_vline(xintercept=median(dairy_queen$cal_fat),  color = "#DD0004", size=1)+
  ggtitle('MCDonals vs Dairy Queen Distribution') +
  xlab('Calories from Fat')+
  ylab('Count')+
  theme_minimal()
```



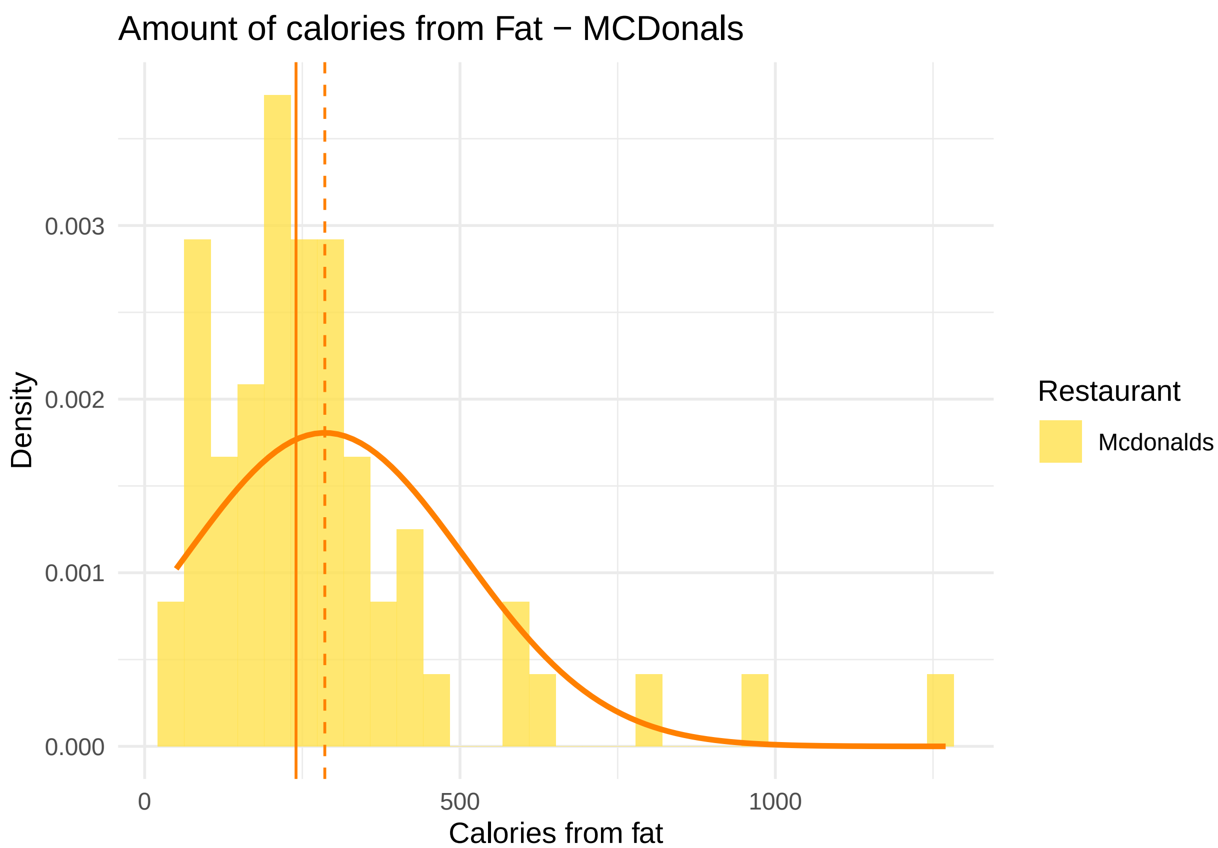

```r
#Q1mcdonal
summary(mcdonalds$cal_fat)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    50.0   160.0   240.0   285.6   320.0  1270.0
```

```r
mcdonalds_summary <- summary(mcdonalds$cal_fat)
hist(mcdonalds$cal_fat,
     xlab = "Amount of calories from Fat - MCDonals",
     col = "#FFE14D", border = "#ff8000")
```



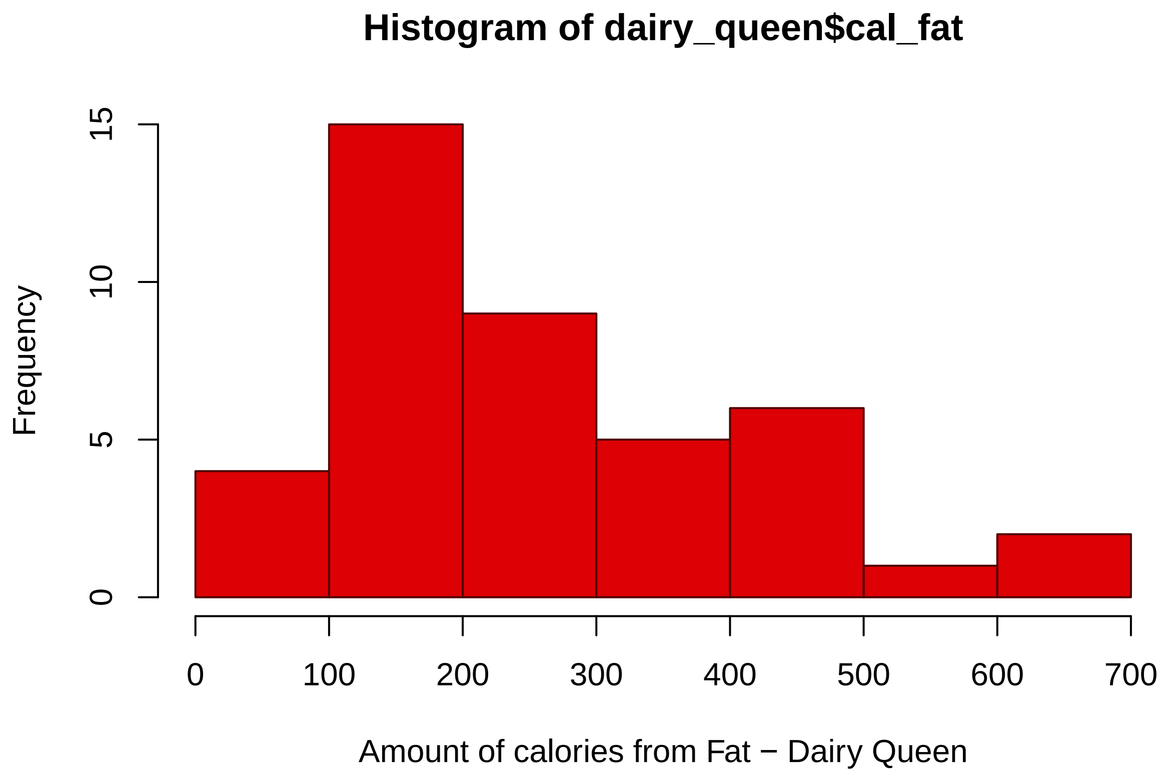

```r
ggplot(data = mcdonalds, aes(fill=restaurant, x = cal_fat)) +
  geom_blank() +
  geom_histogram(aes(y = after_stat(density)),alpha = 0.8) +
  stat_function(fun = dnorm,
                args = c(mean = mean(mcdonalds$cal_fat),
                sd = sd(mcdonalds$cal_fat)), col = "#ff8000", size=1)+
  geom_vline(xintercept=mean(mcdonalds$cal_fat), linetype="dashed", color = "#ff8000")+
  geom_vline(xintercept=median(mcdonalds$cal_fat),  color = "#ff8000")+
  scale_fill_manual('Restaurant', values=c('#FFE14D'))+
  theme_minimal()+
  ggtitle('Amount of calories from Fat - MCDonals') +
  xlab('Calories from fat')+
  ylab('Density')
```




```r
#Q2Dairy Queen
summary(dairy_queen$cal_fat)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   160.0   220.0   260.5   310.0   670.0
```

```r
dairy_queen_summary <- summary(dairy_queen$cal_fat)
hist(dairy_queen$cal_fat,
     xlab = "Amount of calories from Fat - Dairy Queen",
     col = "#DD0004", border = "#4e0000")
```




```
ggplot(data = dairy_queen, aes(fill=restaurant, x = cal_fat)) +
  geom_blank() +
  geom_histogram(aes(y = after_stat(density)),alpha = 0.7) +
  stat_function(fun = dnorm,
                args = c(mean = mean(dairy_queen$cal_fat),
                         sd = sd(dairy_queen$cal_fat)), col = "#4e0000", size=1)+
  geom_vline(xintercept=mean(dairy_queen$cal_fat), linetype="dashed", color = "#4e0000")+
  geom_vline(xintercept=median(dairy_queen$cal_fat),  color = "#4e0000")+
  scale_fill_manual('Restaurant', values=c('#DD0004'))+
  theme_minimal()+
  ggtitle('Amount of calories from Fat - Dairy Queen') +
  xlab('Calories from fat')+
  ylab('Density')
```

The normal distribution

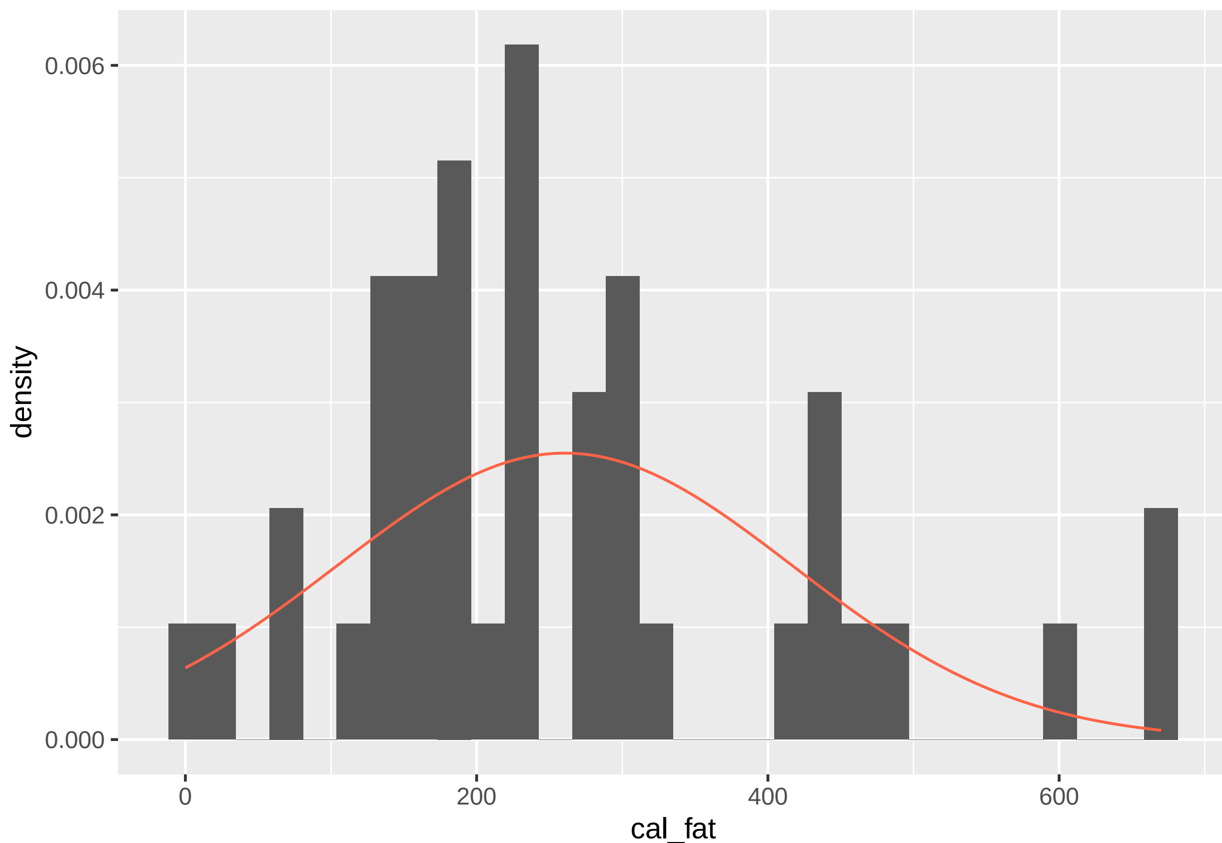
In your description of the distributions, did you use words like *bell-shaped*or *normal*? It's tempting to say so when faced with a unimodal symmetric distribution.

To see how accurate that description is, you can plot a normal distribution curve on top of a histogram to see how closely the data follow a normal distribution. This normal curve should have the same mean and standard deviation as the data. You'll be focusing on calories from fat from Dairy Queen products, so let's store them as a separate object and then calculate some statistics that will be referenced later.

```
dqmean <- mean(dairy_queen$cal_fat)
dqsd   <- sd(dairy_queen$cal_fat)
```

Next, you make a density histogram to use as the backdrop and use the `lines` function to overlay a normal probability curve. The difference between a frequency histogram and a density histogram is that while in a frequency histogram the *heights* of the bars add up to the total number of observations, in a density histogram the *areas* of the bars add up to 1. The area of each bar can be calculated as simply the height *times* the width of the bar. Using a density histogram allows us to properly overlay a normal distribution curve over the histogram since the curve is a normal probability density function that also has area under the curve of 1. Frequency and density histograms both display the same exact shape; they only differ in their y-axis. You can verify this by comparing the frequency histogram you constructed earlier and the density histogram created by the commands below.

```r
ggplot(data = dairy_queen, aes(x = cal_fat)) +
        geom_blank() +
        geom_histogram(aes(y = ..density..)) +
        stat_function(fun = dnorm, args = c(mean = dqmean, sd = dqsd), col = "tomato")
```



After initializing a blank plot with `geom_blank()`, the `ggplot2` package (within the `tidyverse`) allows us to add additional layers. The first layer is a density histogram. The second layer is a statistical function – the density of the normal curve, `dnorm`. We specify that we want the curve to have the same mean and standard deviation as the column of fat calories. The argument `col` simply sets the color for the line to be drawn. If we left it out, the line would be drawn in black.

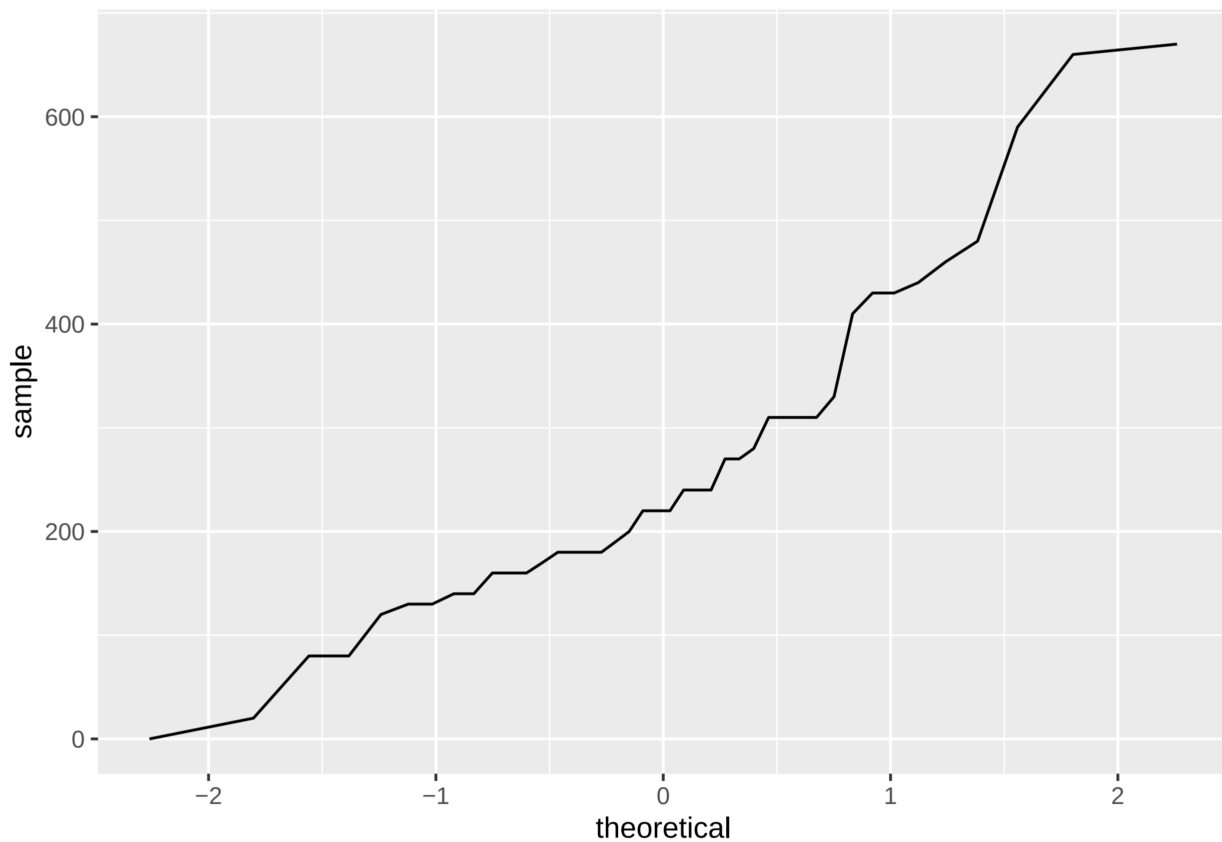
2. Based on the this plot, does it appear that the data follow a nearly normal distribution?

Insert your answer here Yes the plot appears that the data follow a nearly normal distribution because of shape. Although it has a small right tail.

Evaluating the normal distribution

Eyeballing the shape of the histogram is one way to determine if the data appear to be nearly normally distributed, but it can be frustrating to decide just how close the histogram is to the curve. An alternative approach involves constructing a normal probability plot, also called a normal Q-Q plot for "quantile-quantile".

```r
ggplot(data = dairy_queen, aes(sample = cal_fat)) +
  geom_line(stat = "qq")
```



This time, you can use the `geom_line()` layer, while specifying that you will be creating a Q-Q plot with the `stat` argument. It's important to note that here, instead of using `x` instead `aes()`, you need to use `sample`.

The x-axis values correspond to the quantiles of a theoretically normal curve with mean 0 and standard deviation 1 (i.e., the standard normal distribution). The y-axis values correspond to the quantiles of the original unstandardized sample data. However, even if we were to standardize the sample data values, the Q-Q plot would look identical. A data set that is nearly normal will result in a probability plot where the points closely follow a diagonal line. Any deviations from normality leads to deviations of these points from that line.

The plot for Dairy Queen's calories from fat shows points that tend to follow the line but with some errant points towards the upper tail. You're left with the same problem that we encountered with the histogram above: how close is close enough?

A useful way to address this question is to rephrase it as: what do probability plots look like for data that I *know* came from a normal distribution? We can answer this by simulating data from a normal distribution using `rnorm`.

```r
sim_norm <- rnorm(n = nrow(dairy_queen), mean = dqmean, sd = dqsd)
sim_norm
```

```
##  [1]  60.363462 396.206501  26.890582 136.175583 415.391376 160.673545
##  [7] 379.236345 150.137231 387.014067 -28.777146 274.597222 121.353311
## [13] 331.878497 227.788103 141.372924 230.779447 267.580264 374.978306
```

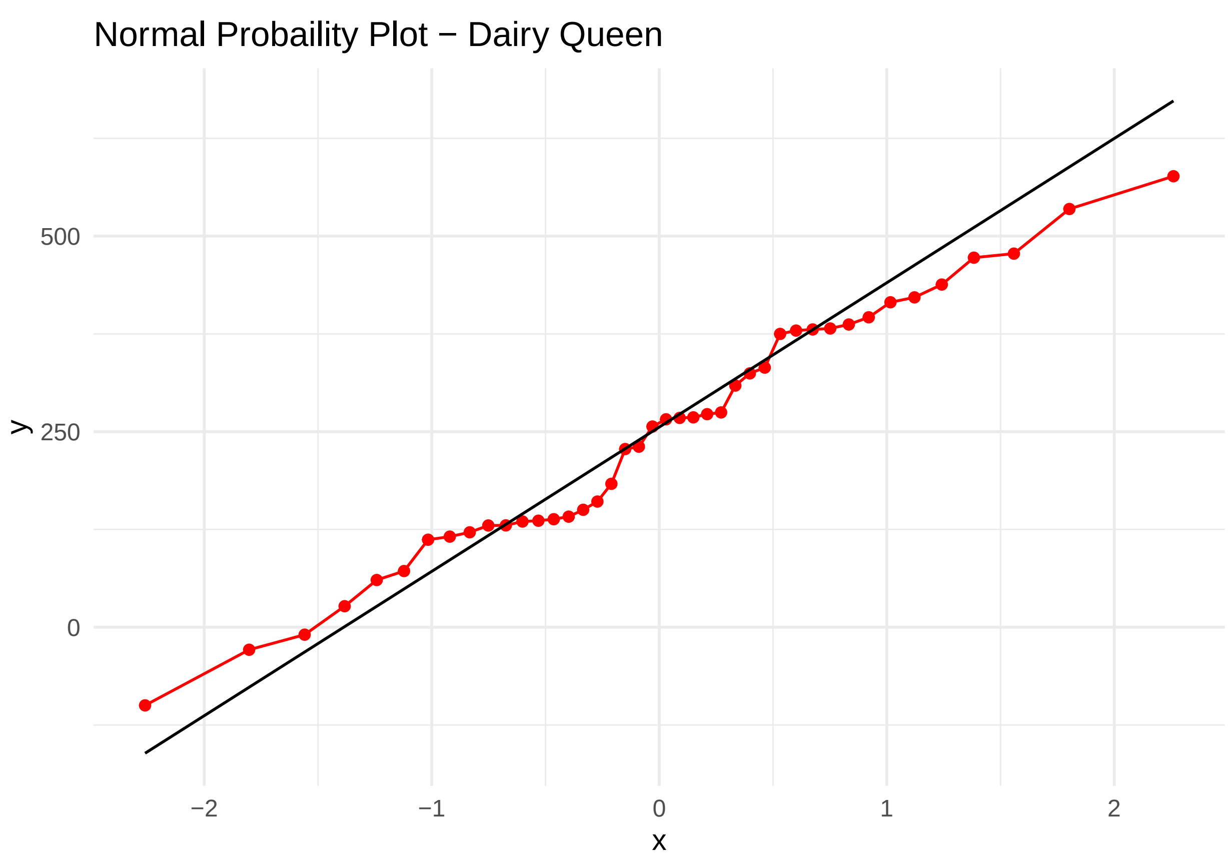
```
## [19] 138.087452 477.603429 111.899934 130.209920 130.056511 256.586889
## [25] 534.677888 308.974170 324.485525 268.297513 380.664838 -99.892468
## [31]  -9.518237 272.384046 576.527131 438.089472 421.679487  71.832559
## [37] 115.839393 381.945243 265.778719 183.354632 472.440227 135.148302
```

The first argument indicates how many numbers you'd like to generate, which we specify to be the same number of menu items in the `dairy_queen` data set using the `nrow()` function. The last two arguments determine the mean and standard deviation of the normal distribution from which the simulated sample will be generated. You can take a look at the shape of our simulated data set, `sim_norm`, as well as its normal probability plot.

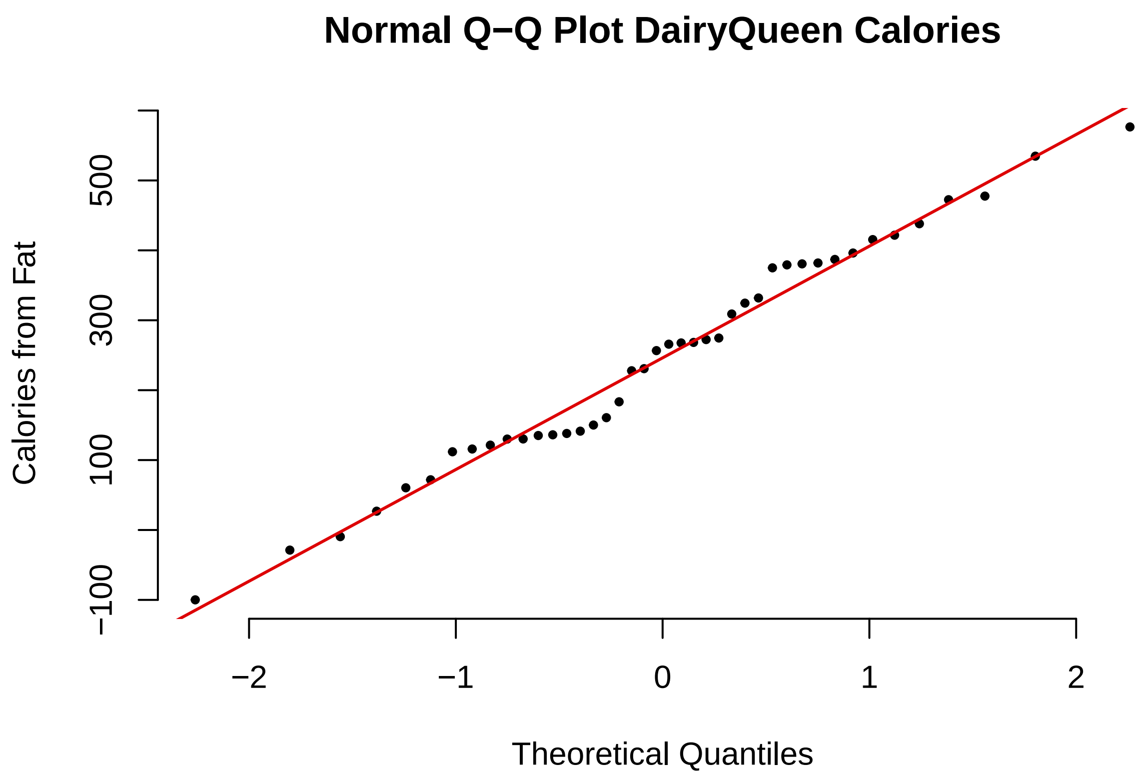
3. Make a normal probability plot of `sim_norm`. Do all of the points fall on the line? How does this plot compare to the probability plot for the real data? (Since `sim_norm` is not a data frame, it can be put directly into the `sample` argument and the `data` argument can be dropped.)

Insert your answer here The majority of the points fall on the line. Using a normal probability plot it helps to observe data values against the a normal distribution line. In other words, we can determinate if it has normal distribution by identifying points deviation from a straight line; QQ plot can be helpful to establish the distribution of a dataset if it can not be defined in a probability plot .

```
ggplot(data = dairy_queen, aes(sample = sim_norm)) +
  geom_line(stat = "qq",color = "red")+
  geom_qq(color = "red")+
  geom_qq_line() +
  theme_minimal()+
  ggtitle('Normal Probaility Plot - Dairy Queen')
```

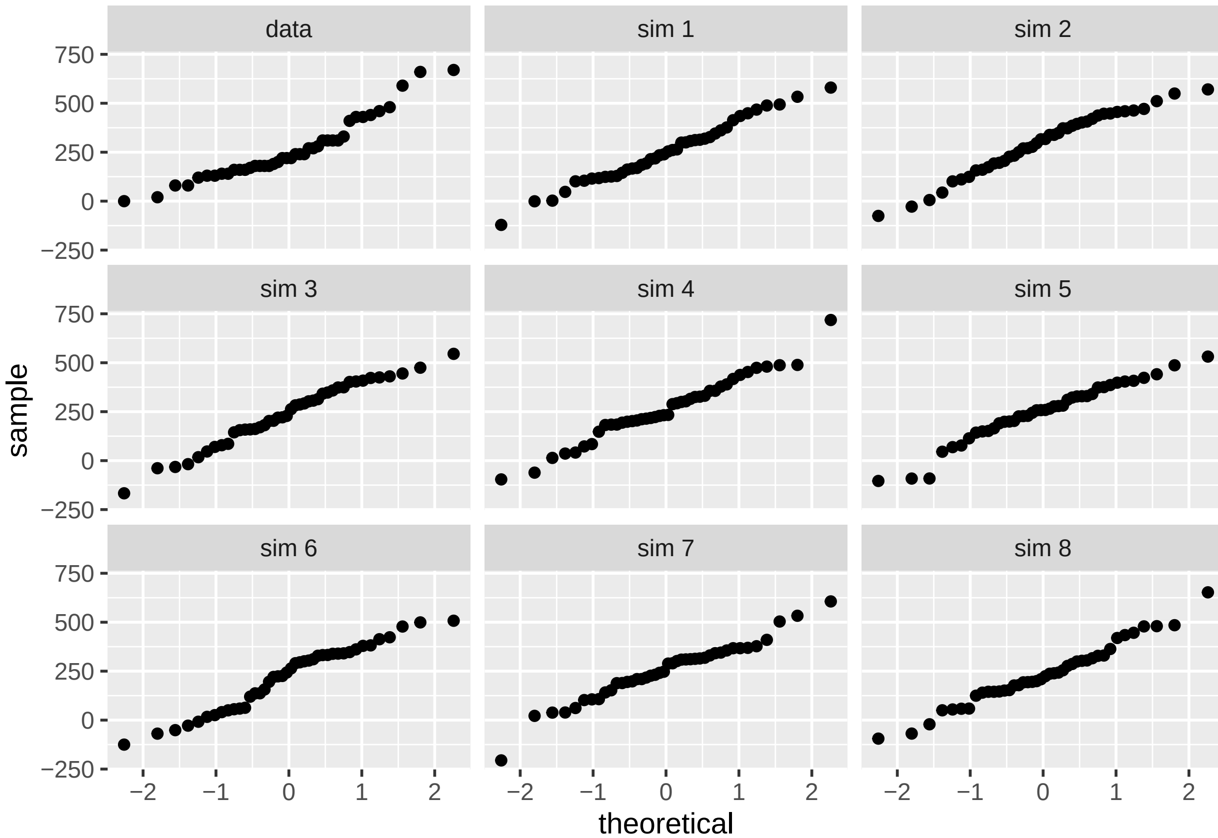


```r
qqnorm(sim_norm, cex=0.5,
       axes=F,
       main='Normal Q-Q Plot DairyQueen Calories',
       ylab='Calories from Fat', pch=19)
axis(1)
axis(2)
abline(mean(sim_norm), sd(sim_norm), col="#DD0004", lwd=1.5)
```



Even better than comparing the original plot to a single plot generated from a normal distribution is to compare it to many more plots using the following function. It shows the Q-Q plot corresponding to the original data in the top left corner, and the Q-Q plots of 8 different simulated normal data. It may be helpful to click the zoom button in the plot window.

```r
qqnormsim(sample = cal_fat, data = dairy_queen)
```



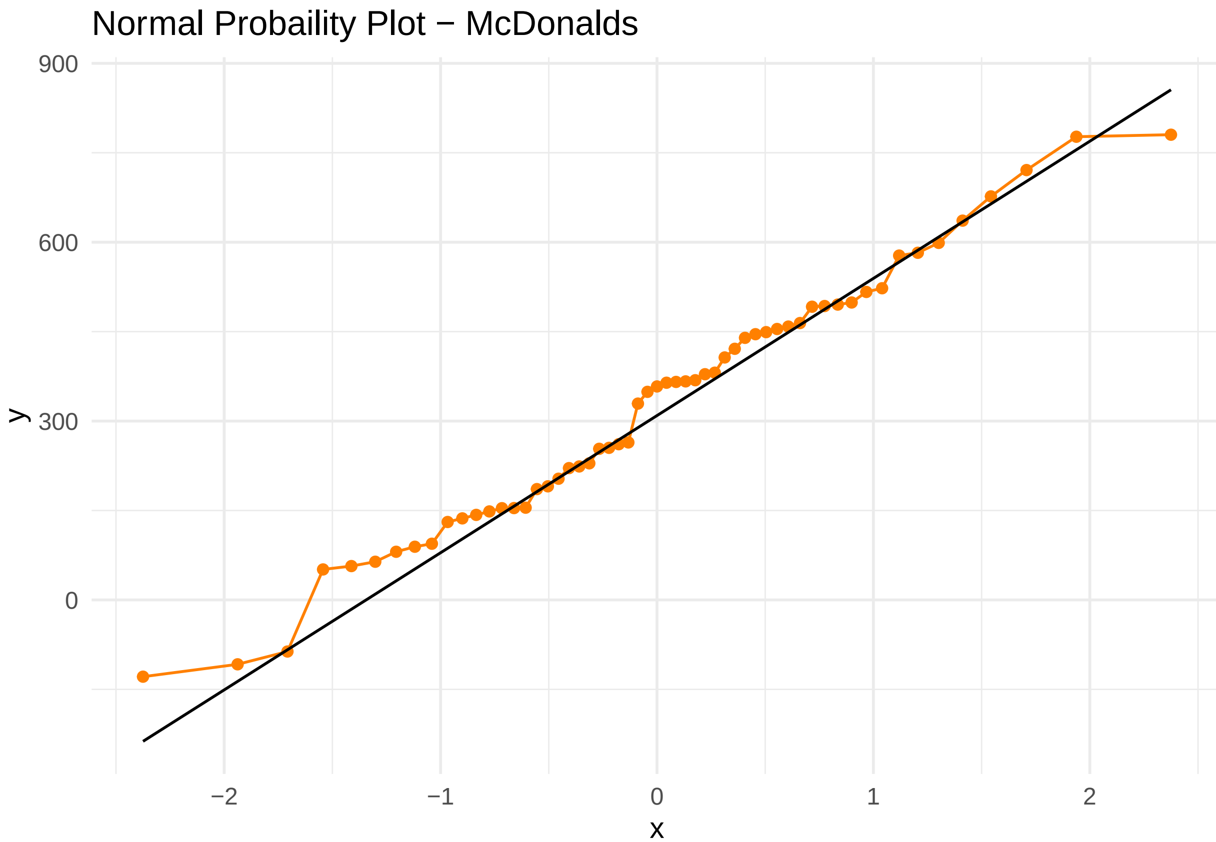
4. Does the normal probability plot for the calories from fat look similar to the plots created for the simulated data? That is, do the plots provide evidence that the calories are nearly normal?

Insert your answer here Yes, the plot generated with the original data look similar to the plots created for the simulated data. Therefore this is an evidence that the calories are nearly normal.

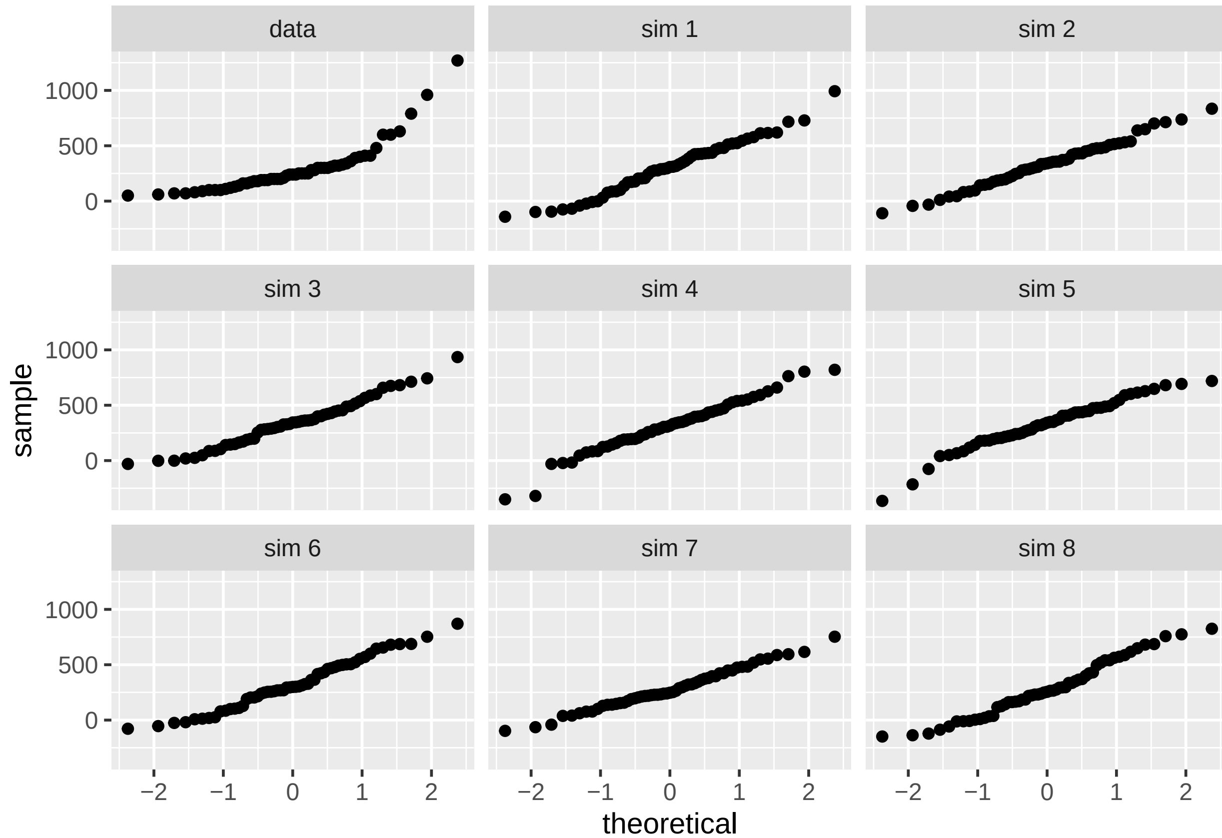
5. Using the same technique, determine whether or not the calories from McDonald's menu appear to come from a normal distribution.

Insert your answer here The points near the tail don't fall exactly along the straight line, the sample data appears to be right skewed.

```
mcmean <- mean(mcdonalds$cal_fat)
mcsd   <- sd(mcdonalds$cal_fat)

sim_normmc <- rnorm(n = nrow(mcdonalds), mean = mcmean, sd = mcsd)

ggplot(data = mcdonalds, aes(sample = sim_normmc)) +
  geom_line(stat = "qq",color = "#ff8000")+
  geom_qq(color = "#ff8000")+
  geom_qq_line() +
  theme_minimal()+
  ggtitle('Normal Probaility Plot - McDonalds')
```




```
qqnormsim(sample = cal_fat, data = mcdonalds)
```



Normal probabilities

Okay, so now you have a slew of tools to judge whether or not a variable is normally distributed. Why should you care?

It turns out that statisticians know a lot about the normal distribution. Once you decide that a random variable is approximately normal, you can answer all sorts of questions about that variable related to probability. Take, for example, the question of, "What is the probability that a randomly chosen Dairy Queen product has more than 600 calories from fat?"

If we assume that the calories from fat from Dairy Queen's menu are normally distributed (a very close approximation is also okay), we can find this probability by calculating a Z score and consulting a Z table (also called a normal probability table). In R, this is done in one step with the function `pnorm()`.

```
1 - pnorm(q = 600, mean = dqmean, sd = dqsd)
```

```
## [1] 0.01501523
```

Note that the function `pnorm()` gives the area under the normal curve below a given value, `q`, with a given mean and standard deviation. Since we're interested in the probability that a Dairy Queen item has more than 600 calories from fat, we have to take one minus that probability.

Assuming a normal distribution has allowed us to calculate a theoretical probability. If we want to calculate the probability empirically, we simply need to determine how many observations fall above 600 then divide this number by the total sample size.

```r
dairy_queen %>%
  filter(cal_fat > 600) %>%
  summarise(percent = n() / nrow(dairy_queen))
```

```
## # A tibble: 1 x 1
##   percent
##     <dbl>
## 1  0.0476
```

Although the probabilities are not exactly the same, they are reasonably close. The closer that your distribution is to being normal, the more accurate the theoretical probabilities will be.

6. Write out two probability questions that you would like to answer about any of the restaurants in this dataset. Calculate those probabilities using both the theoretical normal distribution as well as the empirical distribution (four probabilities in all). Which one had a closer agreement between the two methods?

Insert your answer here What is the probability of observations that have less than 500 calories for MCDonalds? Answer: theoretical distribution is 83.41% and empirical distribution is 89.5%

```r
pnorm(q = 500, mean = mean(mcdonalds$cal_fat), sd = sd(mcdonalds$cal_fat))
```

```
## [1] 0.834105
```

```r
mcdonalds %>%
  filter(cal_fat < 500) %>%
  summarise(percent = n() / nrow(mcdonalds))
```

```
## # A tibble: 1 x 1
##   percent
##     <dbl>
## 1   0.895
```

What is the probability of observations that have less than 500 calories for Dairy Queen? Answer: theoretical distribution is 93.70% and empirical distribution is 92.9% , In this case we have a closer agreement between both methods

```r
pnorm(q = 500, mean = mean(dairy_queen$cal_fat), sd = sd(dairy_queen$cal_fat))
```

```
## [1] 0.937072
```

```r
dairy_queen %>%
  filter(cal_fat < 500) %>%
  summarise(percent = n() / nrow(dairy_queen))
```

```
## # A tibble: 1 x 1
##   percent
##     <dbl>
## 1   0.929
```

More Practice

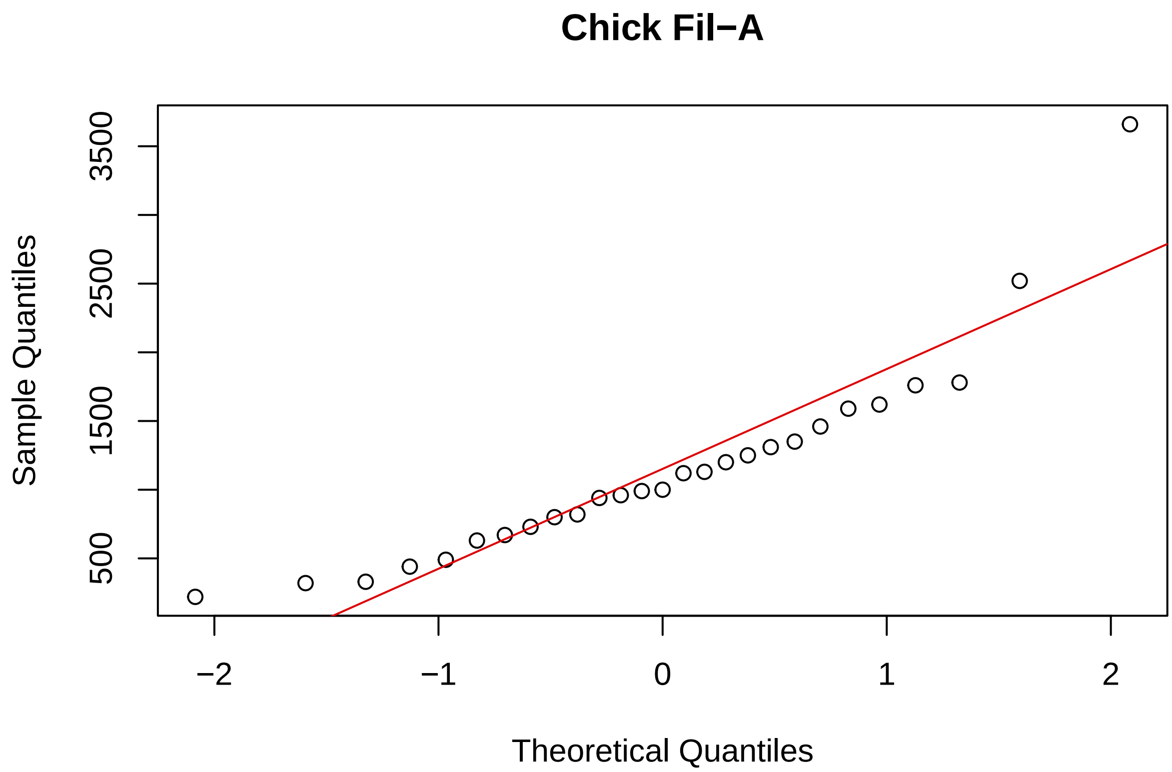
7. Now let's consider some of the other variables in the dataset. Out of all the different restaurants, which ones' distribution is the closest to normal for sodium?

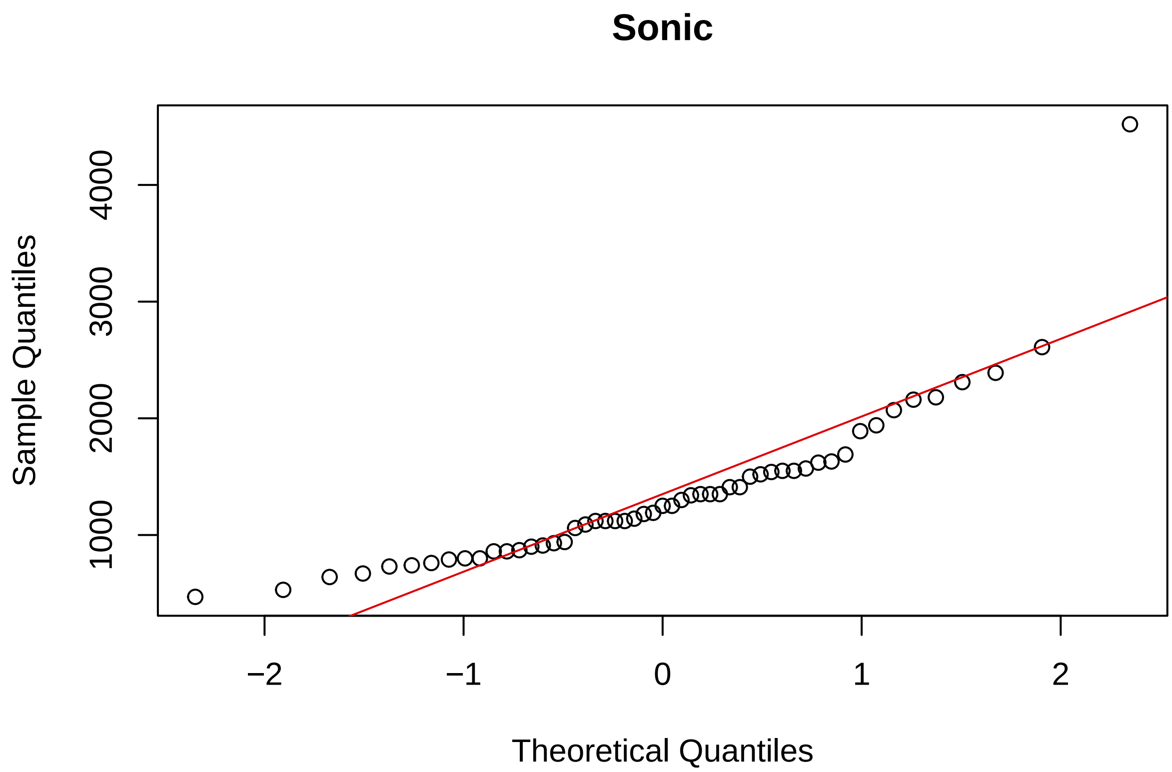
Insert your answer here Based on the plots generated, Burger King is the closest to normal for sodium. Arbys will also be one of the closest to normal for sodium.

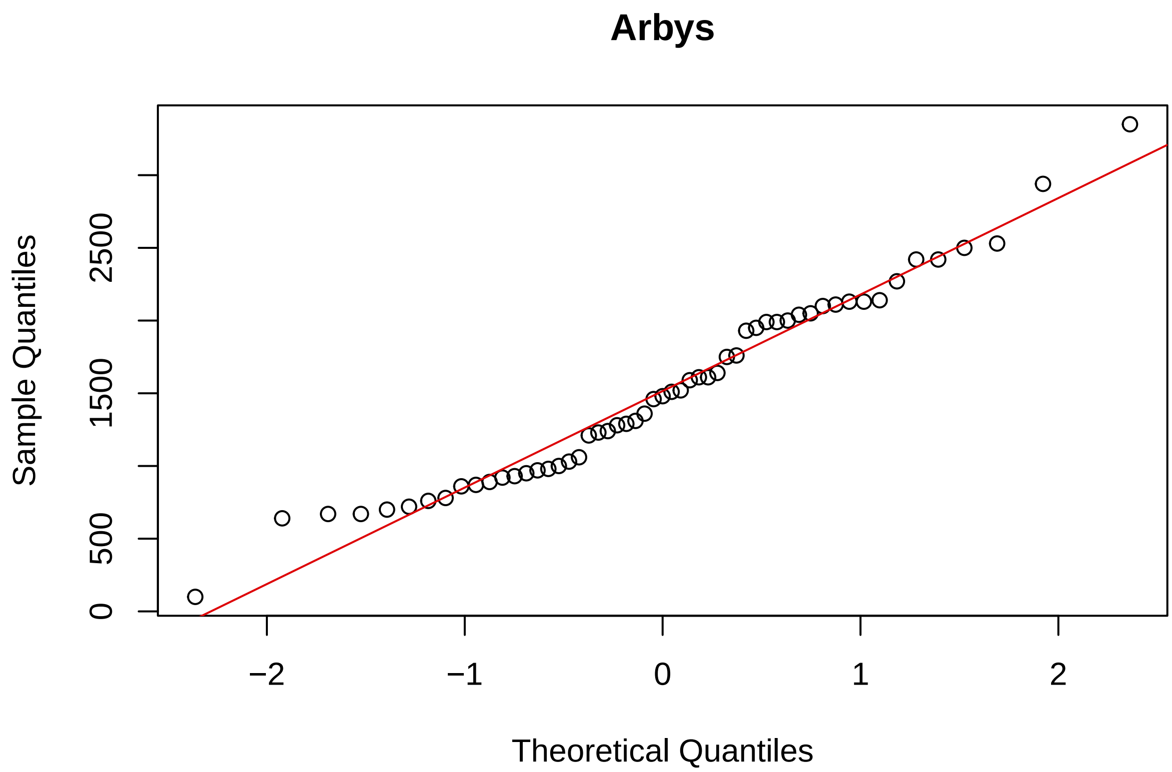
```
fastfood_f <- fastfood |>
  filter(restaurant != "Mcdonalds",
         restaurant != "Dairy Queen")

restaurant_name <- unique(fastfood_f$restaurant)
restaurant_name
```

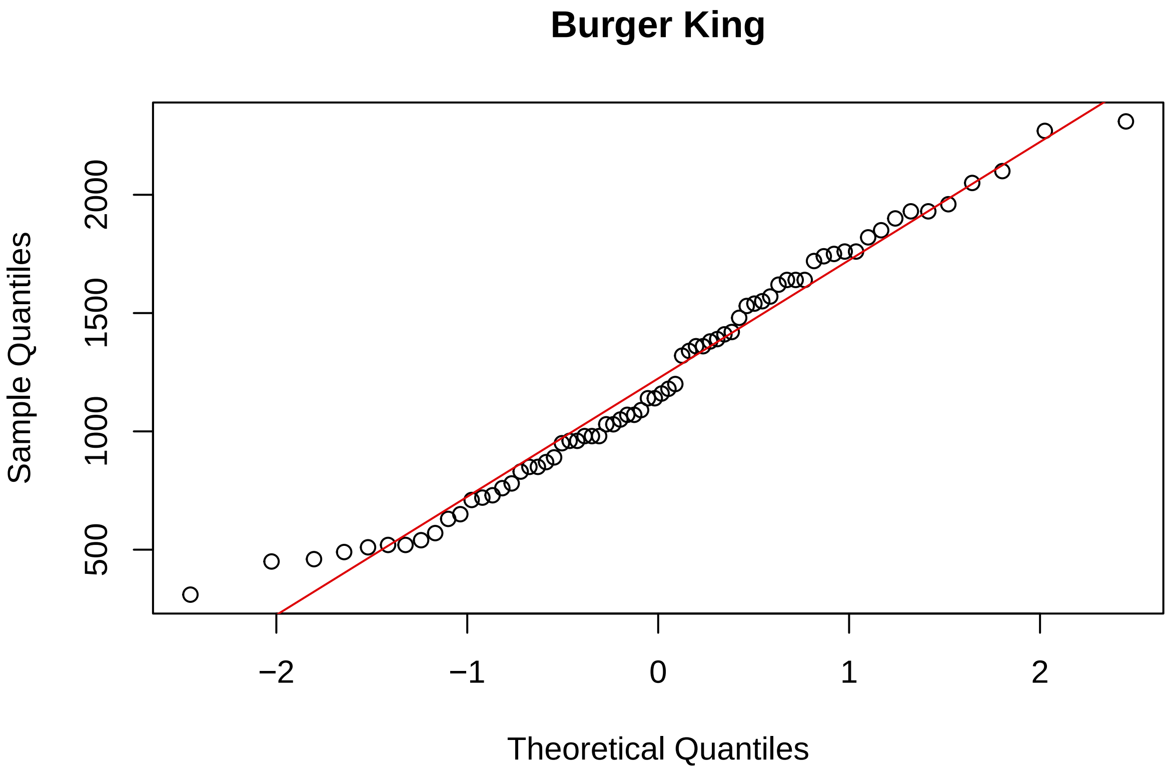
```
## [1] "Chick Fil-A" "Sonic"       "Arbys"       "Burger King" "Subway"
## [6] "Taco Bell"
```

```
for(i in 1:length(restaurant_name)){
  temp_name <- restaurant_name[i]
  temp_filter <- fastfood_f |>
  filter(restaurant == temp_name)
  #print(temp_name)
  qqnorm(temp_filter$sodium,
         main = temp_name)
  abline(mean(temp_filter$sodium), sd(temp_filter$sodium), col="#DD0004")

}
```

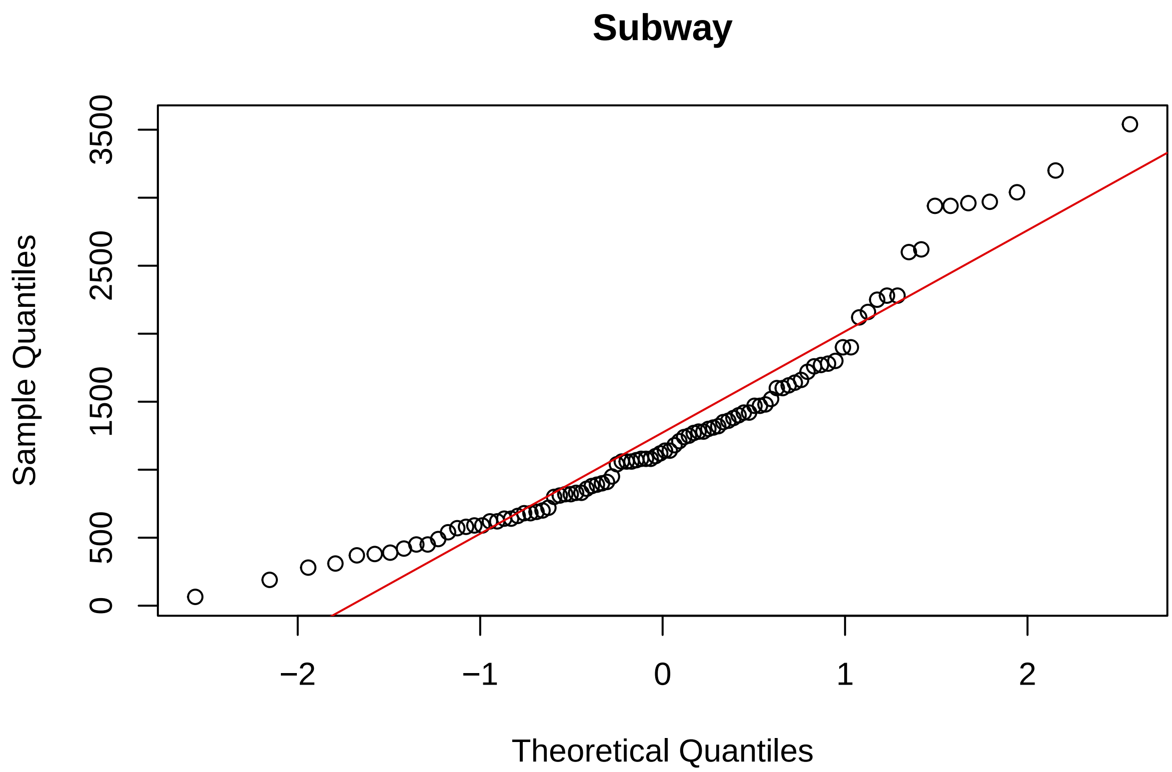

Chick Fil−A
Sample Quantiles
500
1500
2500
3500
−2
−1
0
1
2
Theoretical Quantiles

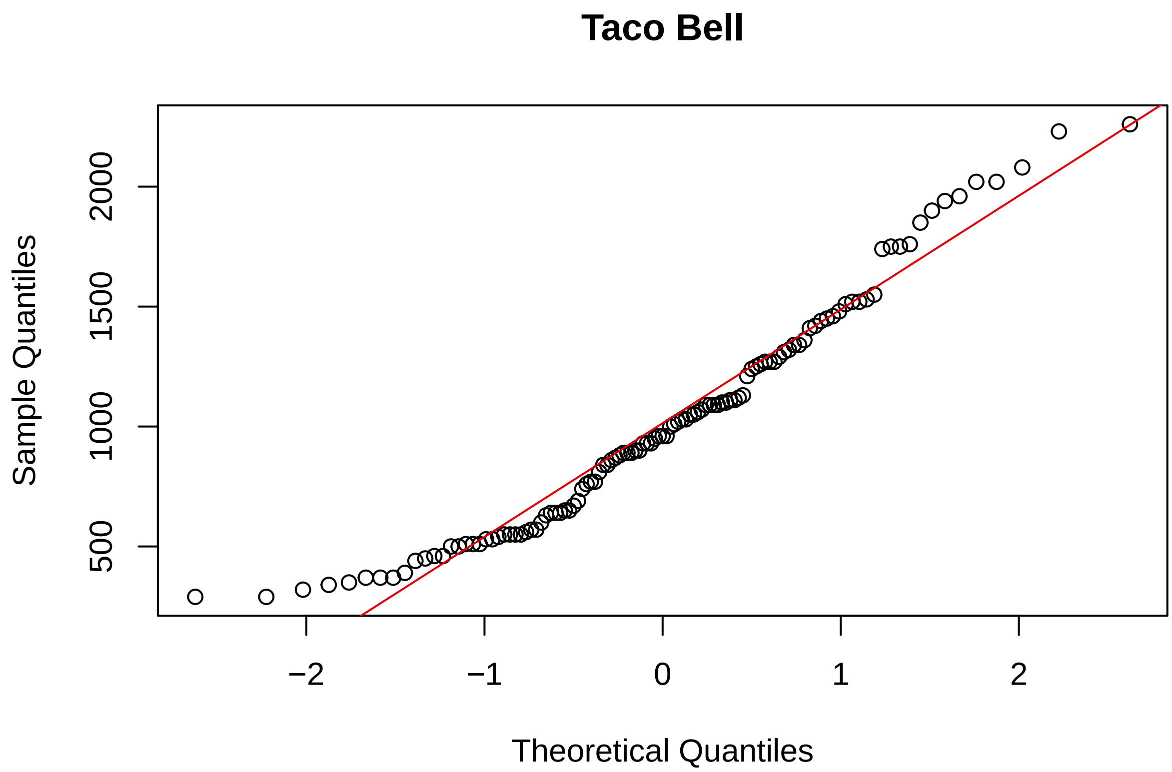

Sonic
Sample Quantiles
1000
2000
3000
4000
−2
−1
0
1
2
Theoretical Quantiles


Arbys
Sample Quantiles
0
500
1500
2500
−2
−1
0
1
2
Theoretical Quantiles



Burger King
Sample Quantiles
500
1000
1500
2000
−2
−1
0
1
2
Theoretical Quantiles


Subway
Sample Quantiles
0
500
1500
2500
3500
−2
−1
0
1
2
Theoretical Quantiles



8. Note that some of the normal probability plots for sodium distributions seem to have a stepwise pattern. why do you think this might be the case?

Insert your answer here It seems to me that the content of sodium for the items in the menu for each restaurant will determinate the distribution

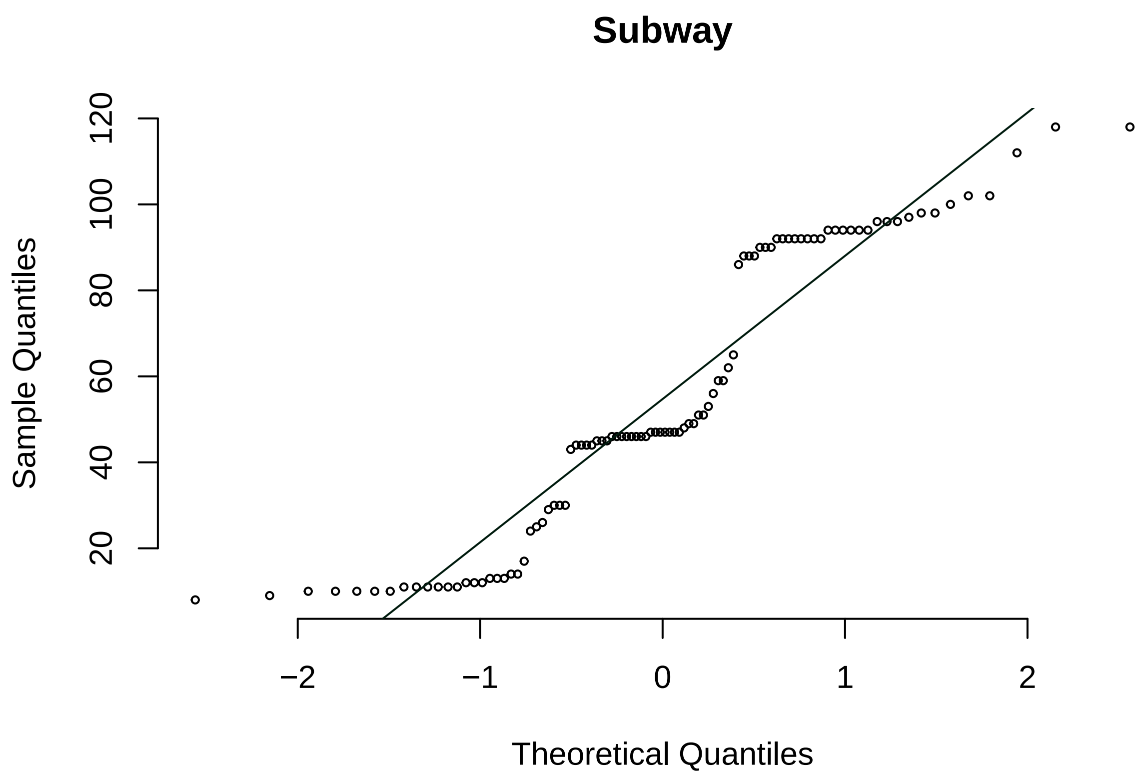
9. As you can see, normal probability plots can be used both to assess normality and visualize skewness. Make a normal probability plot for the total carbohydrates from a restaurant of your choice. Based on this normal probability plot, is this variable left skewed, symmetric, or right skewed? Use a histogram to confirm your findings.

Total carbohydrates for Subway is right skewed, the mean of 54.72 is greater than the median of 47. The tail of the graph is pulled toward to the right.

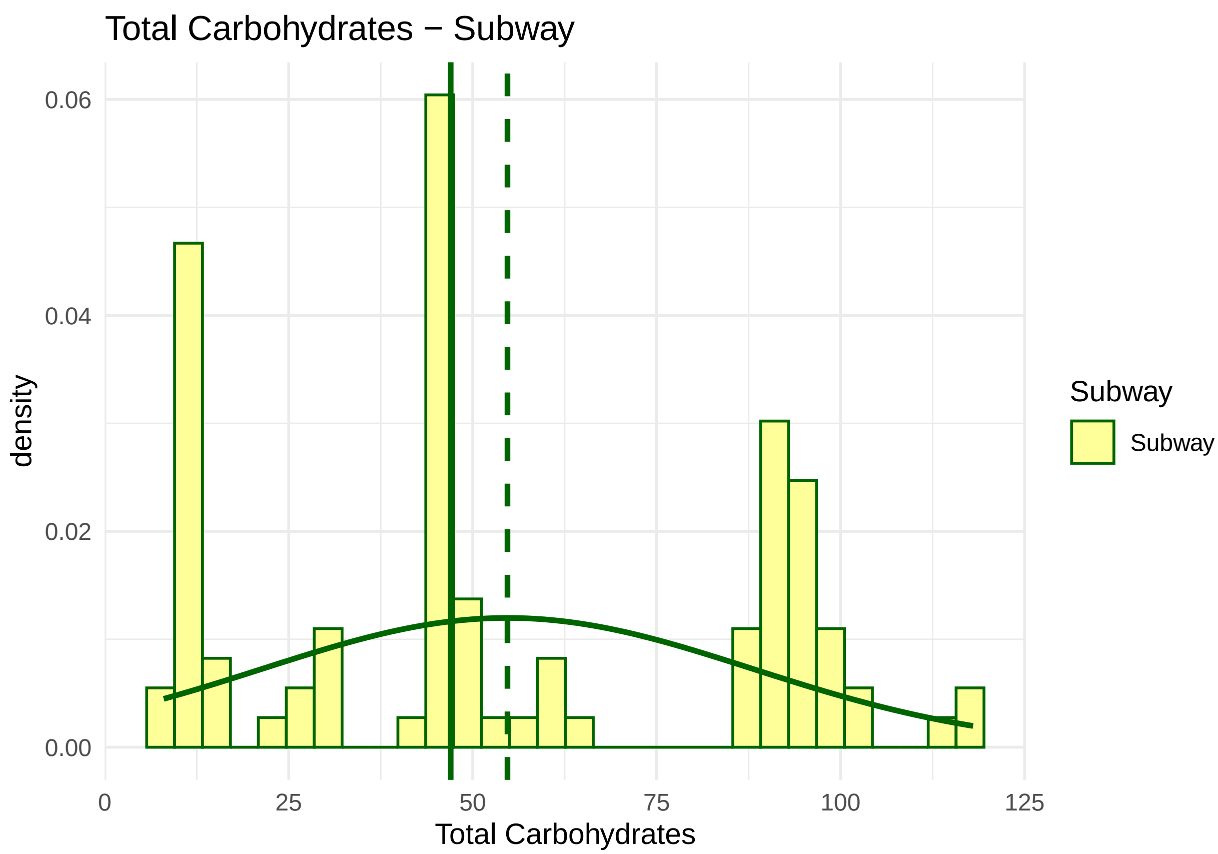
Insert your answer here

```
Subway <- fastfood %>%
  filter(restaurant == "Subway")

#ggplot(data = Subway, aes(sample = total_carb)) +
#  geom_line(stat = "qq")

qqnorm(Subway$total_carb, main = "Subway",cex=0.5,
       axes=F)
axis(1)
```

```
axis(2)
abline(mean(Subway$total_carb), sd(Subway$total_carb), col="#001b0e")
```




```
ggplot(data = Subway, aes(x = total_carb, fill="Subway")) +
  geom_blank() +
  geom_histogram(aes(y = ..density..), col = "darkgreen") +
  stat_function(fun = dnorm,
                args = c(mean = mean(Subway$total_carb),
                sd = sd(Subway$total_carb)),
                col = "darkgreen",size=1)+
  theme_minimal()+
  ggtitle('Total Carbohydrates - Subway')+
  scale_fill_manual('Subway', values=c('#ffff9a'))+
  xlab('Total Carbohydrates')+
  geom_vline(xintercept=mean(Subway$total_carb), linetype="dashed", color = "darkgreen", size=1)+
  geom_vline(xintercept=median(Subway$total_carb),  color = "darkgreen", size=1)
```


Total Carbohydrates − Subway
0.06
0.04
0.02
0.00
density
0
25
50
75
100
125
Total Carbohydrates
Subway
Subway