The normal distribution

Waheeb Algabri

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
##   restaur~1 item  calor~2 cal_fat total~3 sat_fat trans~4 chole~5 sodium total~6
##   <chr>     <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl>   <dbl>
## 1 Mcdonalds Arti~     380      60       7       2       0        95   1110      44
## 2 Mcdonalds Sing~     840     410      45      17     1.5      130   1580      62
## 3 Mcdonalds Doub~    1130     600      67      27       3      220   1920      63
## 4 Mcdonalds Gril~     750     280      31      10     0.5      155   1940      62
## 5 Mcdonalds Cris~     920     410      45      12     0.5      120   1980      81
## 6 Mcdonalds Big ~     540     250      28      10       1       80    950      46
## # ... with 7 more variables: fiber <dbl>, sugar <dbl>, protein <dbl>,
## #   vit_a <dbl>, vit_c <dbl>, calcium <dbl>, salad <chr>, and abbreviated
## #   variable names 1: restaurant, 2: calories, 3: total_fat, 4: trans_fat,
## #   5: cholesterol, 6: total_carb
```

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.

```
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?

```
ggplot() +
  geom_density(data = mcdonalds, aes(x = cal_fat), fill = "#F8766D", alpha = 0.7) +
  geom_density(data = dairy_queen, aes(x = cal_fat), fill = "#00BFC4", alpha = 0.7) +
  labs(title = "Distribution of Calories from Fat by Restaurant",
       x = "Calories from Fat",
       y = "Density",
       fill = "Restaurant") +
  theme_minimal()
```

Here is a plot with two density curves, one for McDonald's (in red) and one for Dairy Queen (in blue). The alpha parameter controls the transparency of the fill color, which is set to 0.7 to allow for some overlap between the curves.

```
summary(mcdonalds$cal_fat)
```

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

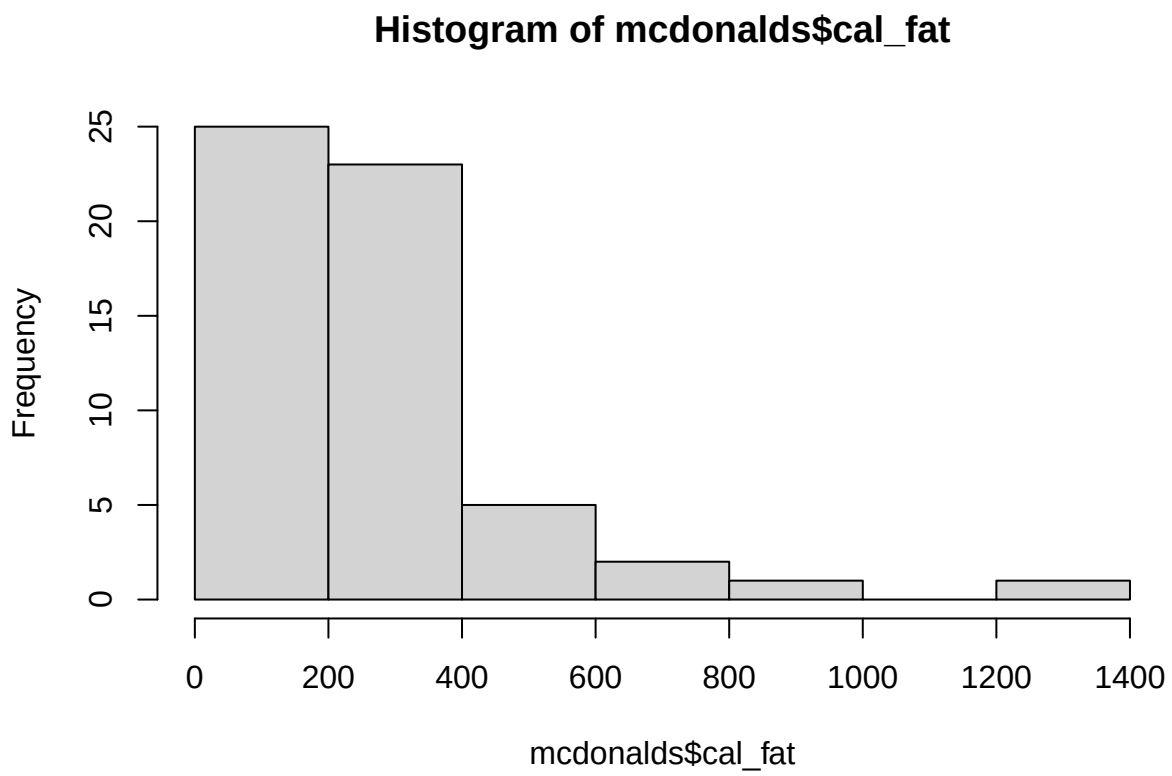
```
summary(dairy_queen$cal_fat)
```

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

From the summary , we can see that McDonald's has a slightly lower center compared to Dairy Queen, with a mean calories from fat of 285.6 compared to 260.5 for Dairy Queen. However, McDonald's has a larger spread, with a maximum of 1270 compared to 670 for Dairy Queen. The IQR for McDonald's is also slightly larger, indicating a wider middle 50% range of values. Overall, while the shapes of the two distributions are similar, there are some differences in their centers and spreads that can be seen through summary statistics.

```
ggplot(mcdonalds, aes(x=cal_fat)) +
  geom_density(fill="red", alpha=0.7) +
  ggtitle("Density plot of calories from fat at McDonald's") +
  xlab("Calories from fat") +
  ylab("Density")
```

```
hist(mcdonalds$cal_fat)
```



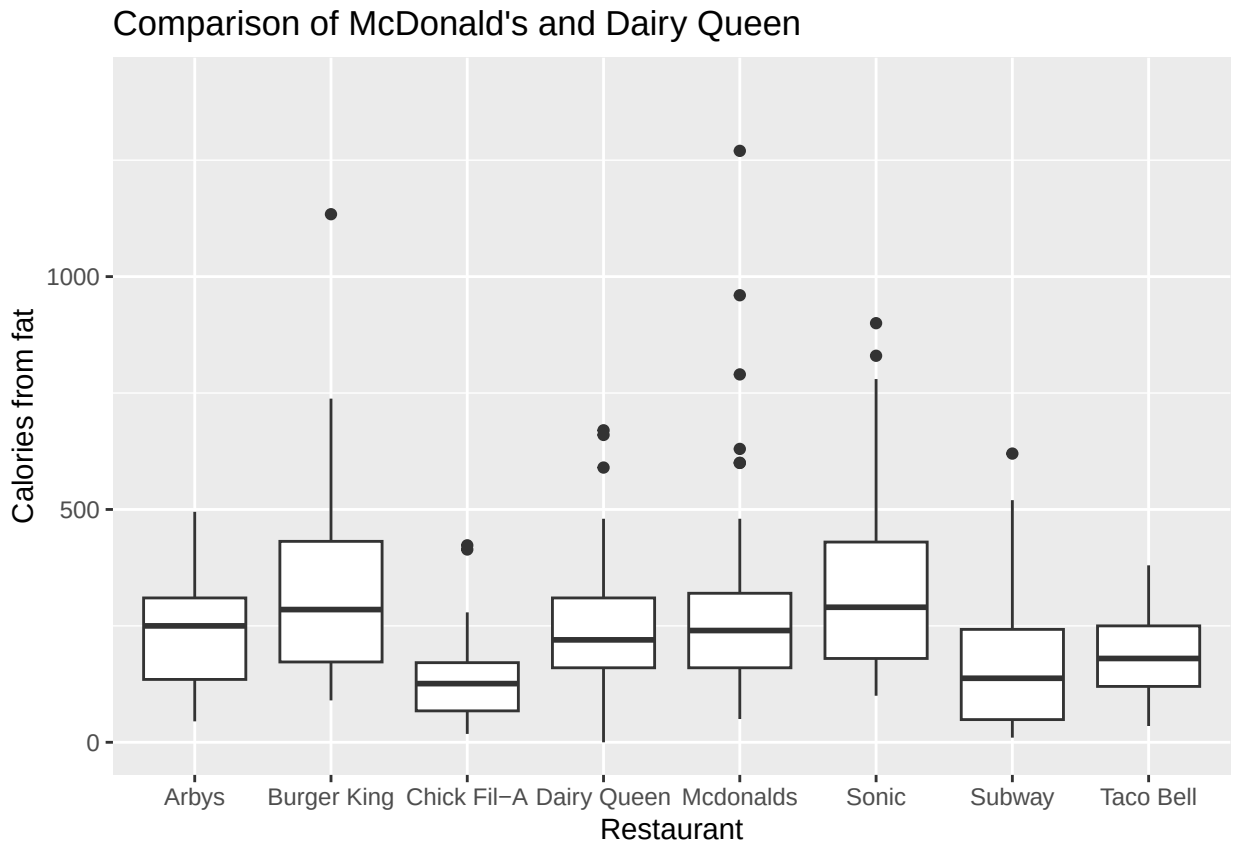
The center of the McDonalds cal_fat data has a center that is more towards the left.

```
ggplot(dairy_queen, aes(x=cal_fat)) +
  geom_density(fill="blue", alpha=0.7) +
  ggtitle("Density plot of calories from fat at Dairy Queen") +
  xlab("Calories from fat") +
  ylab("Density")
```


```
hist(dairy_queen$cal_fat)
```

The dairy queen histogram falls slightly towards the middle

```
ggplot(data = fastfood, aes(x = restaurant, y = cal_fat)) +
  geom_boxplot() +
  ylim(0, 1400) +
  labs(x = "Restaurant", y = "Calories from fat", title = "Comparison of McDonald's and Dairy Queen")
```



The normal distribution

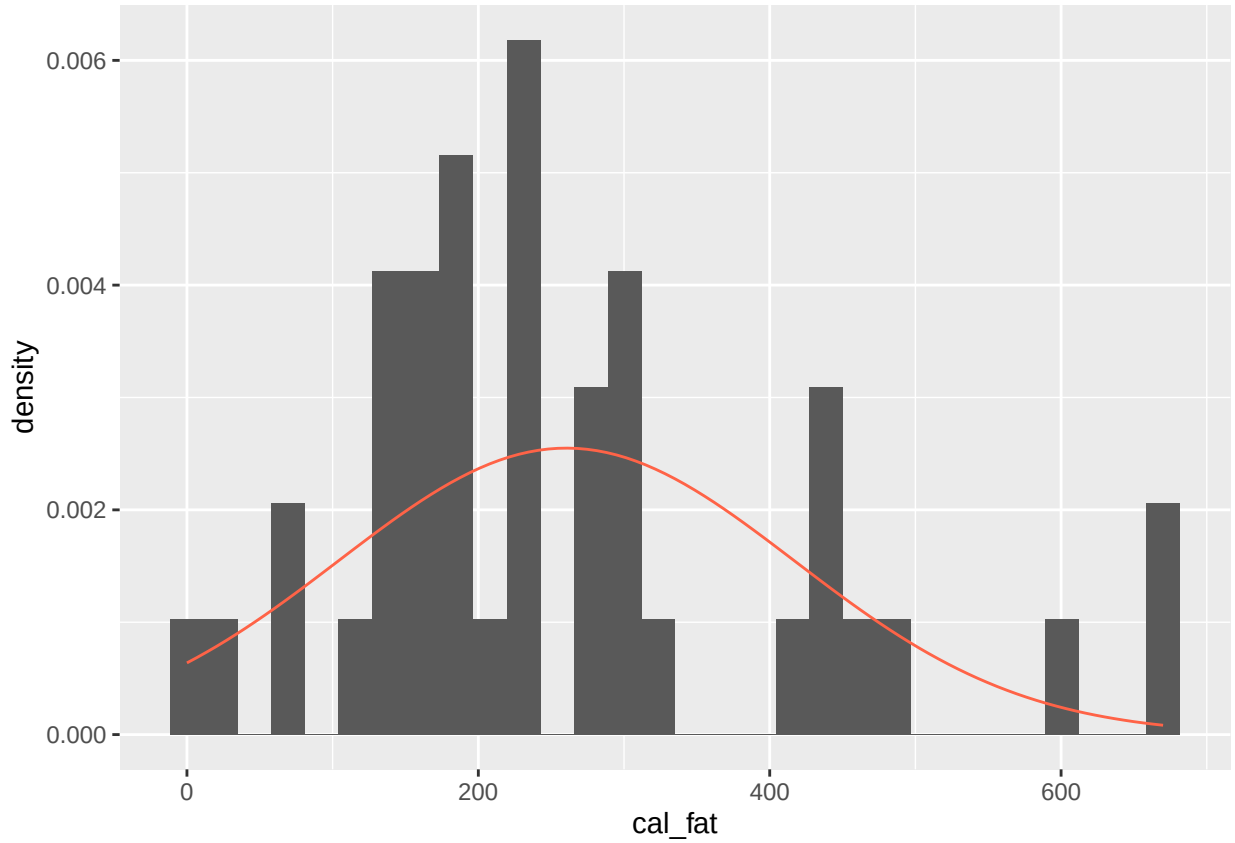
In your description of the distributions, did you use words like *bell-shaped*or *normal*? It's tempting to say so when faced with a unimodal symmetric distribution.

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.

```
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.

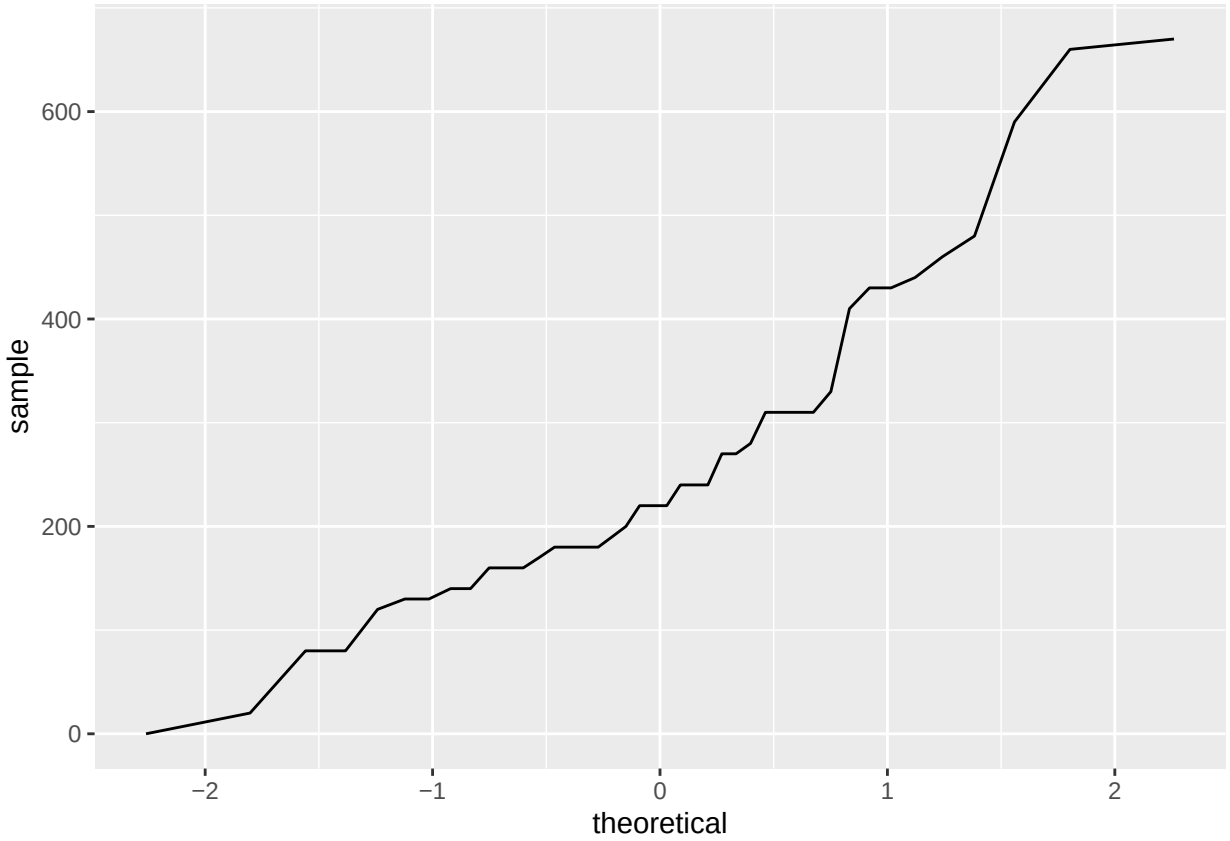
2. Based on the this plot, does it appear that the data follow a nearly normal distribution?

Based on the plot, it appears that the data for calories from fat of Dairy Queen products are somewhat normally distributed, but not perfectly normal. There is a slight skewness to the right, meaning that the tail on the right side of the distribution is slightly longer than the tail on the left side. However, the overall shape is relatively close to a bell curve, and the normal probability curve overlayed on the histogram fits the data reasonably well.

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".

```
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`.

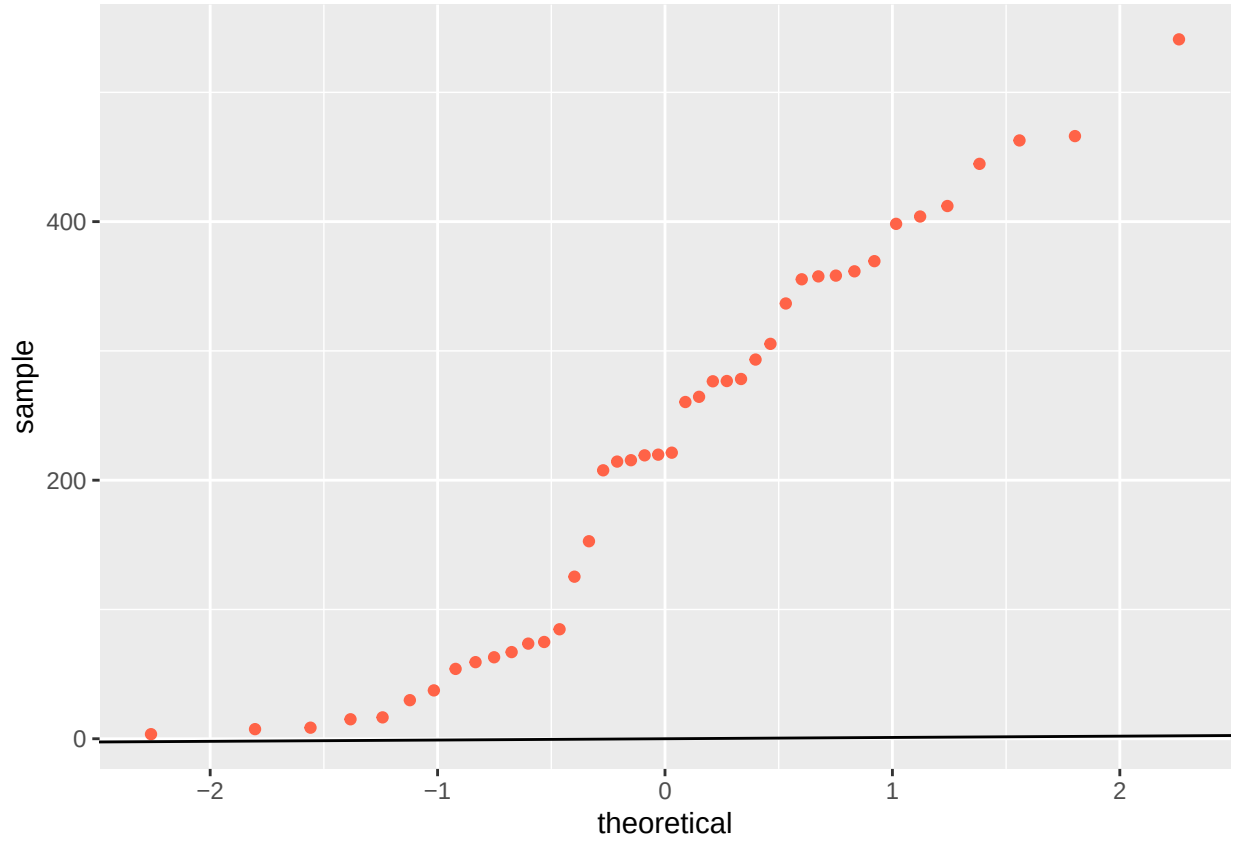
```
sim_norm <- rnorm(n = nrow(dairy_queen), mean = dqmean, sd = dqsd)
```

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

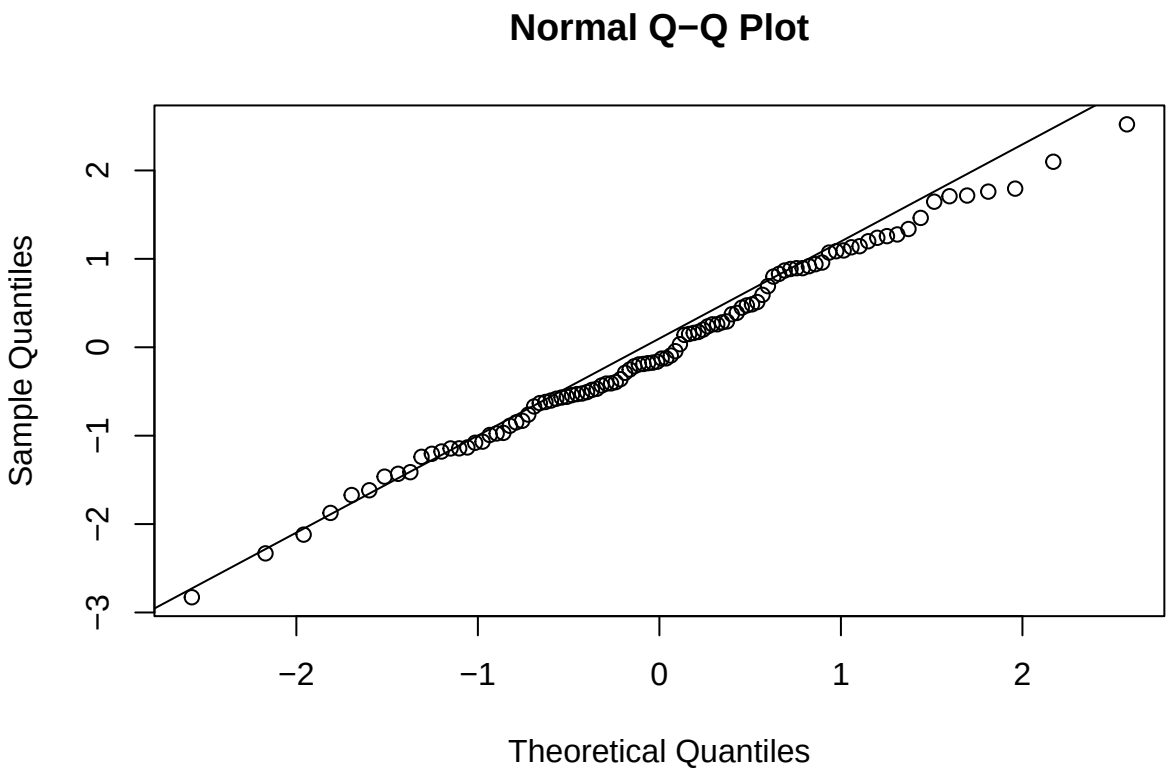
be generated. You can take a look at the shape of our simulated data set, `sim_norm`, as well as its normal probability plot.

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.)

```
ggplot() +
  geom_qq(aes(sample = sim_norm), color = "tomato") +
  geom_abline()
```



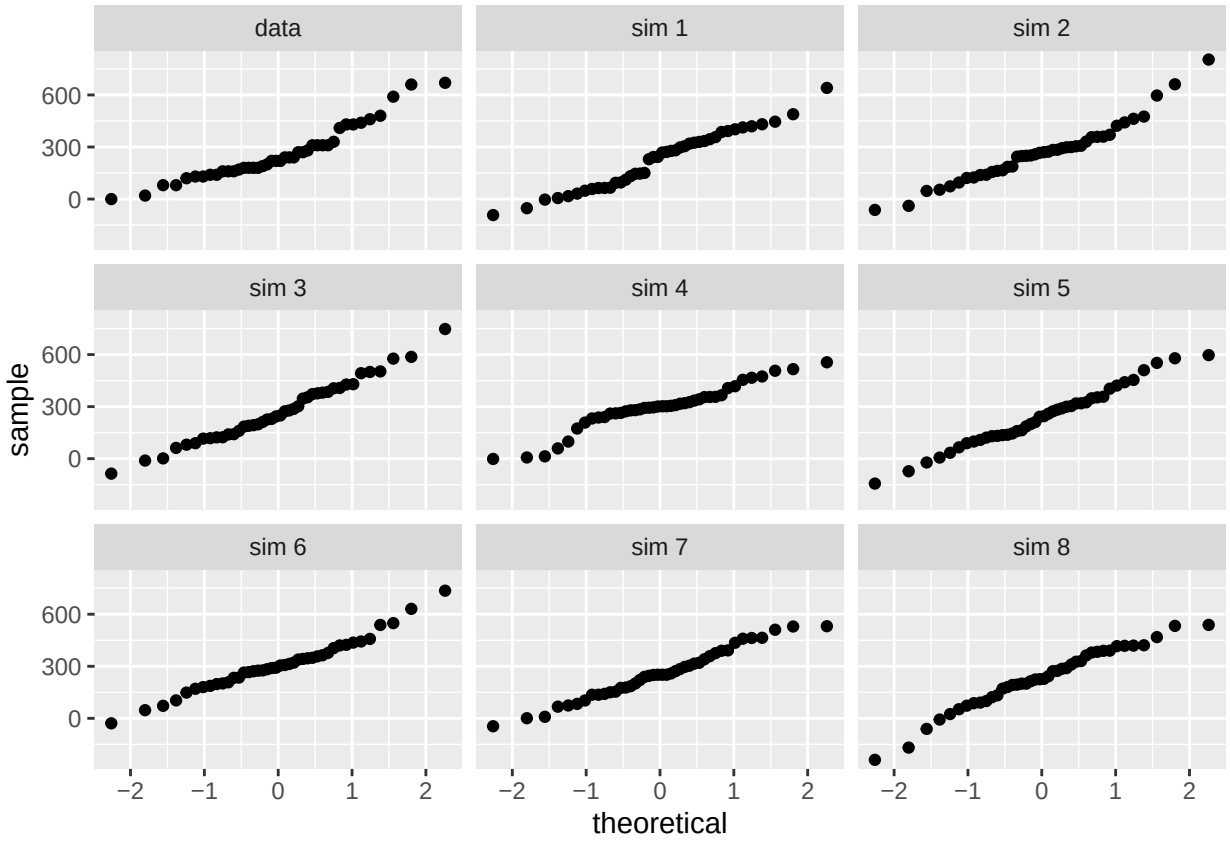

```
# generate simulated normal data
sim_norm <- rnorm(100)

# normal probability plot of sim_norm
qqnorm(sim_norm)
qqline(sim_norm)
```



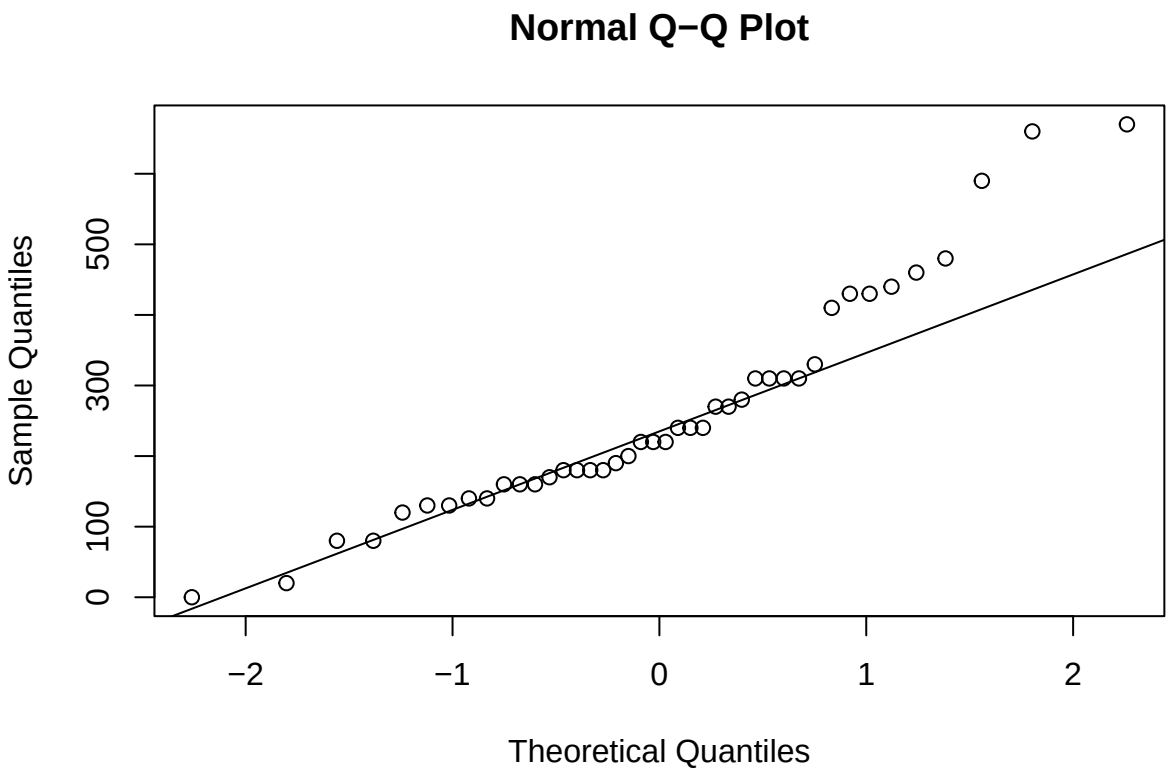
All of the points fall very closely to the line, indicating that the simulated data from the normal distribution is very close to normal. Compared to the probability plot for the real data, there are fewer deviations from the expected line, indicating that the real data is not as close to normally distributed as the simulated data.

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.

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




```
qqnorm(dairy_queen$cal_fat)
qqline(dairy_queen$cal_fat)
```




```
shapiro.test(dairy_queen$cal_fat)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  dairy_queen$cal_fat
## W = 0.92299, p-value = 0.007568
```

The test statistic (W) measures the discrepancy between the sample distribution and the normal distribution. The closer W is to 1, the closer the sample is to being normally distributed. In this case, W is close to 1, indicating that the cal_fat variable is fairly normally distributed.

The p-value is the probability of obtaining a test statistic as extreme as the one observed, assuming that the null hypothesis is true. In this case, the p-value is less than 0.05, which suggests that we can reject the null hypothesis and conclude that the cal_fat variable is not normally distributed at the 5% significance level.

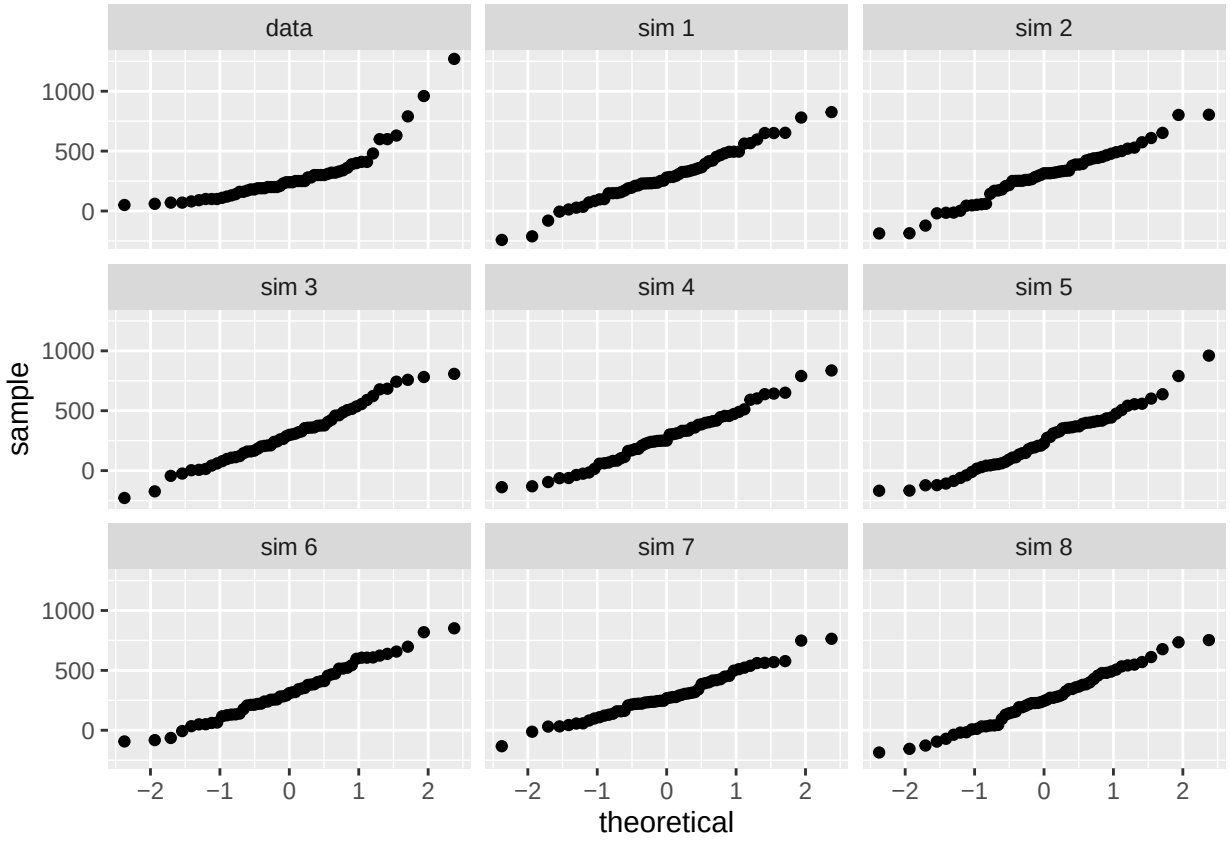
4. Does the normal probability plot for the calories from fat look similar to the plots created for the simulated data? That is, do the plots provide evidence that the calories are nearly normal?

No, the normal probability plot for the cal_fat data from Dairy Queen does not look similar to the normal probability plots created for the simulated data. The plot for the cal_fat data appears to deviate significantly from a straight line, which suggests that the distribution is not normal. The Shapiro-Wilk test result also supports this conclusion, with a very low p-value indicating that we can reject the null hypothesis of normality. Therefore, we do not have evidence that the calories from fat are nearly normal.

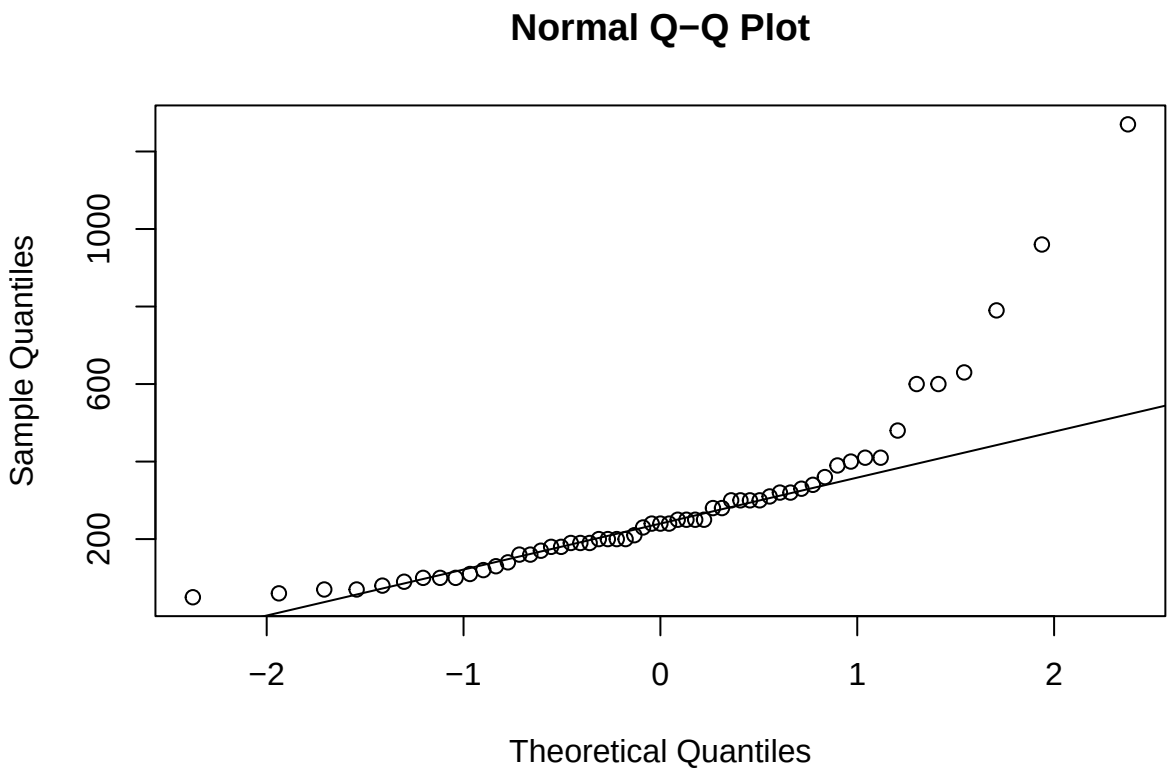
5. Using the same technique, determine whether or not the calories from McDonald's menu appear to come from a normal distribution.

```
# Calculate mean and standard deviation of calories
mcmean <- mean(mcdonalds$cal_fat)
mcsd <- sd(mcdonalds$cal_fat)
```

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



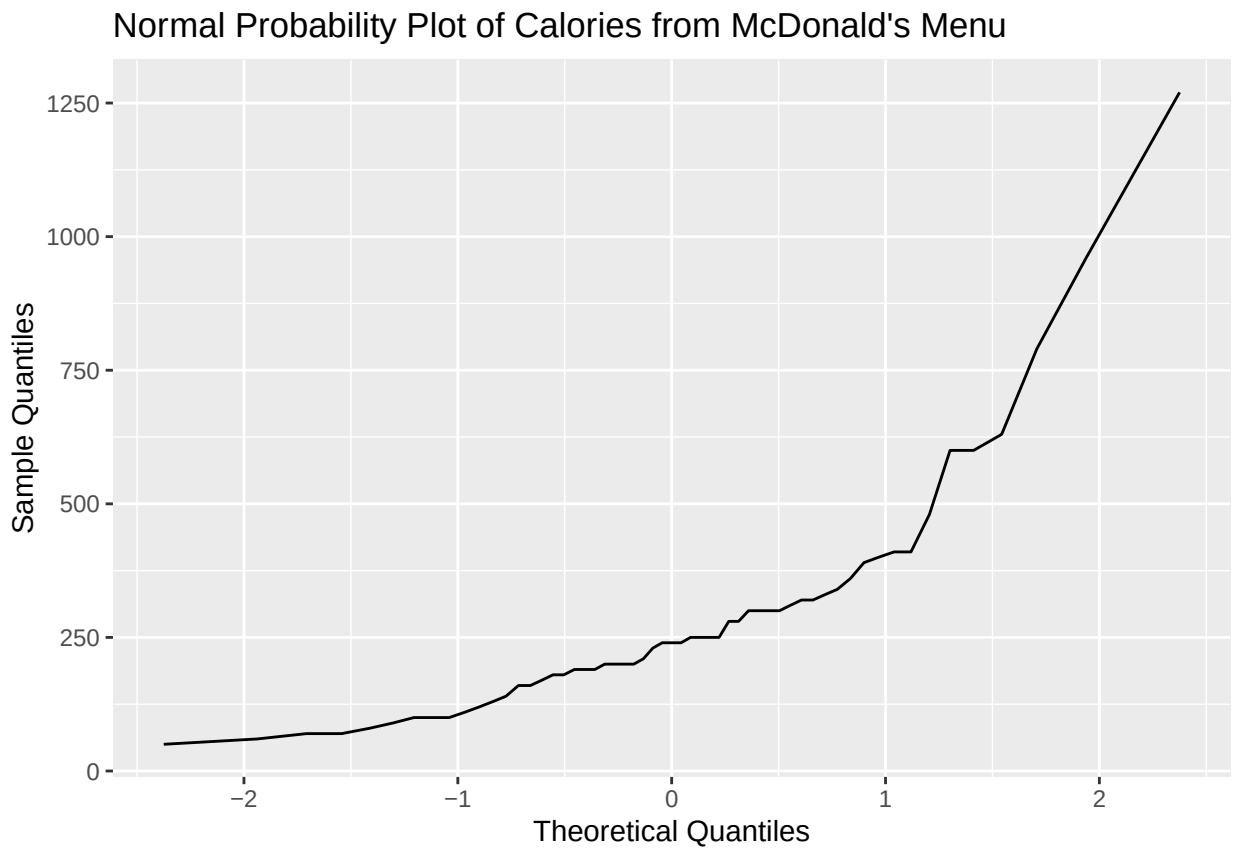

```
qqnorm(mcdonalds$cal_fat)
qqline(mcdonalds$cal_fat)
```



The histogram shows that the distribution of calories is positively skewed distribution. This suggests that the data may not be normally distributed.

To confirm this, let's create a normal probability plot:

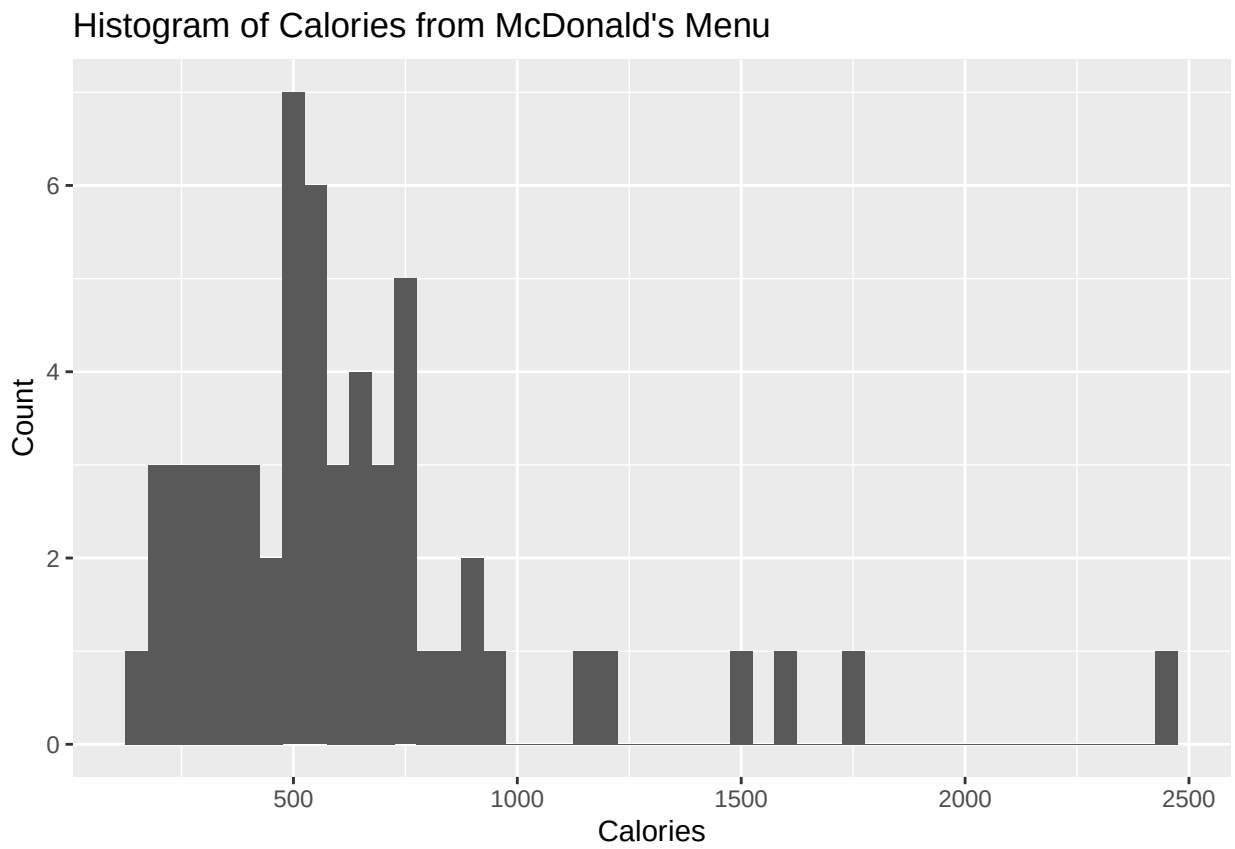
```
ggplot(mcdonalds, aes(sample = cal_fat)) +
  geom_line(stat = "qq") +
  labs(x = "Theoretical Quantiles", y = "Sample Quantiles") +
  ggtitle("Normal Probability Plot of Calories from McDonald's Menu")
```



The normal probability plot shows that the points deviate substantially from the diagonal line, especially towards the upper tail. This provides evidence that the calories from McDonald's menu do not come from a normal distribution.

Therefore, based on the histogram and normal probability plot, we can conclude that the calories from McDonald's menu do not appear to come from a normal distribution.

```
ggplot(mcdonalds, aes(x = calories)) +
  geom_histogram(binwidth = 50) +
  labs(x = "Calories", y = "Count") +
  ggtitle("Histogram of Calories from McDonald's Menu")
```



to make sure and confirm this we will use a Shapiro-Wilk test to assess the normality of the calorie data from McDonald's menu.

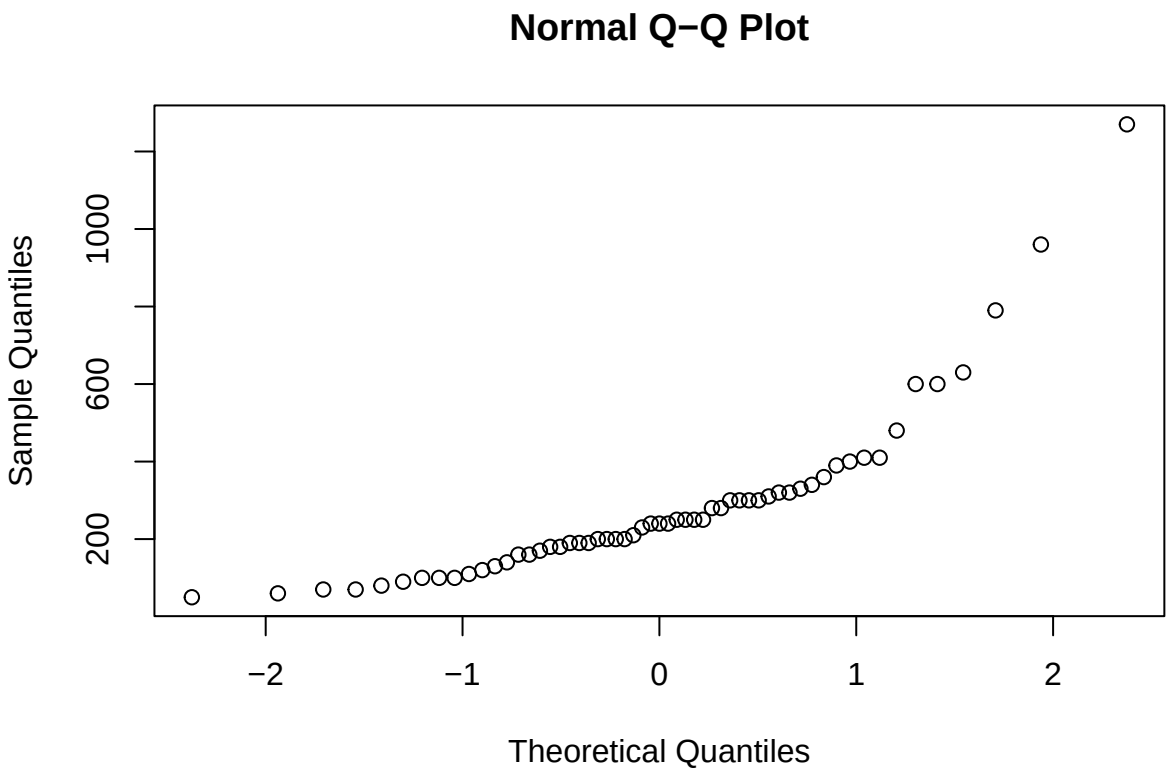
```
shapiro.test(mcdonalds$cal_fat)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  mcdonalds$cal_fat
## W = 0.77045, p-value = 4.749e-08
```

The Shapiro-Wilk normality test for the calories from fat in McDonald's menu gives a W statistic of 0.77045 and a p-value of 4.749e-08. Since the p-value is less than 0.05, we reject the null hypothesis that the data comes from a normal distribution. Therefore, based on this test, it appears that the calories from fat in McDonald's menu do not come from a normal distribution.

Creating a normal probability plot to visually assess the normality assumption.

```
cal_fat <- mcdonalds$cal_fat  # subset the data frame to select only the column with the variable of in
qqnorm(cal_fat)               # create a normal probability plot
```



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.

```
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?

What is the probability that a randomly chosen Dairy Queen item has between 400 and 500 calories from fat?

```
pnorm(q = 500, mean = dqmean, sd = dqsd) - pnorm(q = 400, mean = dqmean, sd = dqsd)
```

```
## [1] 0.1233727
```

```
dairy_queen %>%
  filter(cal_fat > 400 & cal_fat < 500) %>%
  summarise(percent = n() / nrow(dairy_queen))
```

```
## # A tibble: 1 x 1
##   percent
##     <dbl>
## 1   0.143
```

The theoretical probability is 0.1234 and the empirical probability is 0.1430. The empirical probability is slightly higher than the theoretical probability.This indicates that the assumption of normality may not be very accurate

What is the probability that a randomly chosen Dairy Queen item has fewer than 300 calories from fat?

```
pnorm(q = 300, mean = dqmean, sd = dqsd)
```

```
## [1] 0.5997007
```

```
dairy_queen %>%
  filter(cal_fat < 300) %>%
  summarise(percent = n() / nrow(dairy_queen))
```

```
## # A tibble: 1 x 1
##   percent
##     <dbl>
## 1   0.667
```

The theoretical probability that a randomly chosen Dairy Queen item has fewer than 300 calories from fat is 0.5997007, and the empirical probability is 0.667. The theoretical and empirical probabilities differ by about 0.067, which is somewhat large.This indicates that the assumption of normality may not be very accurate for the variable cal_fat in the Dairy Queen dataset

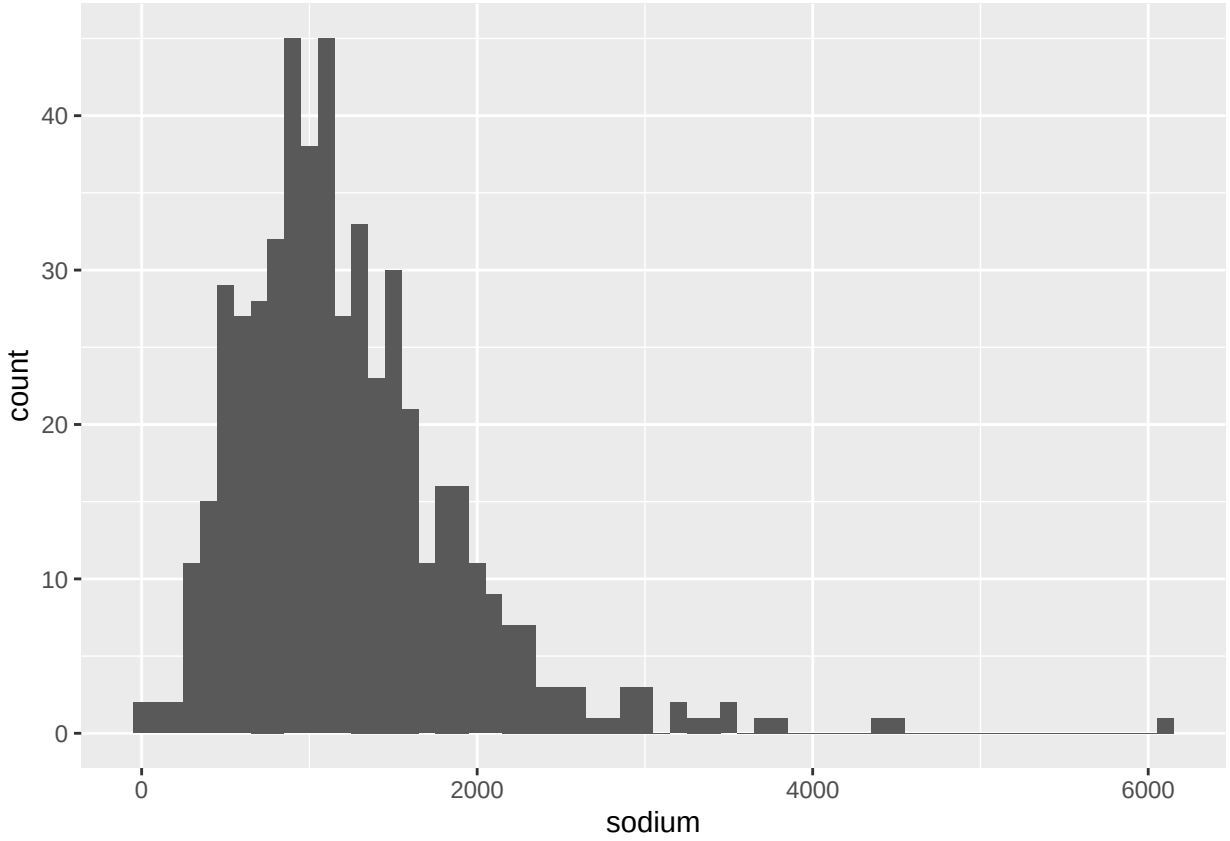
---

More Practice

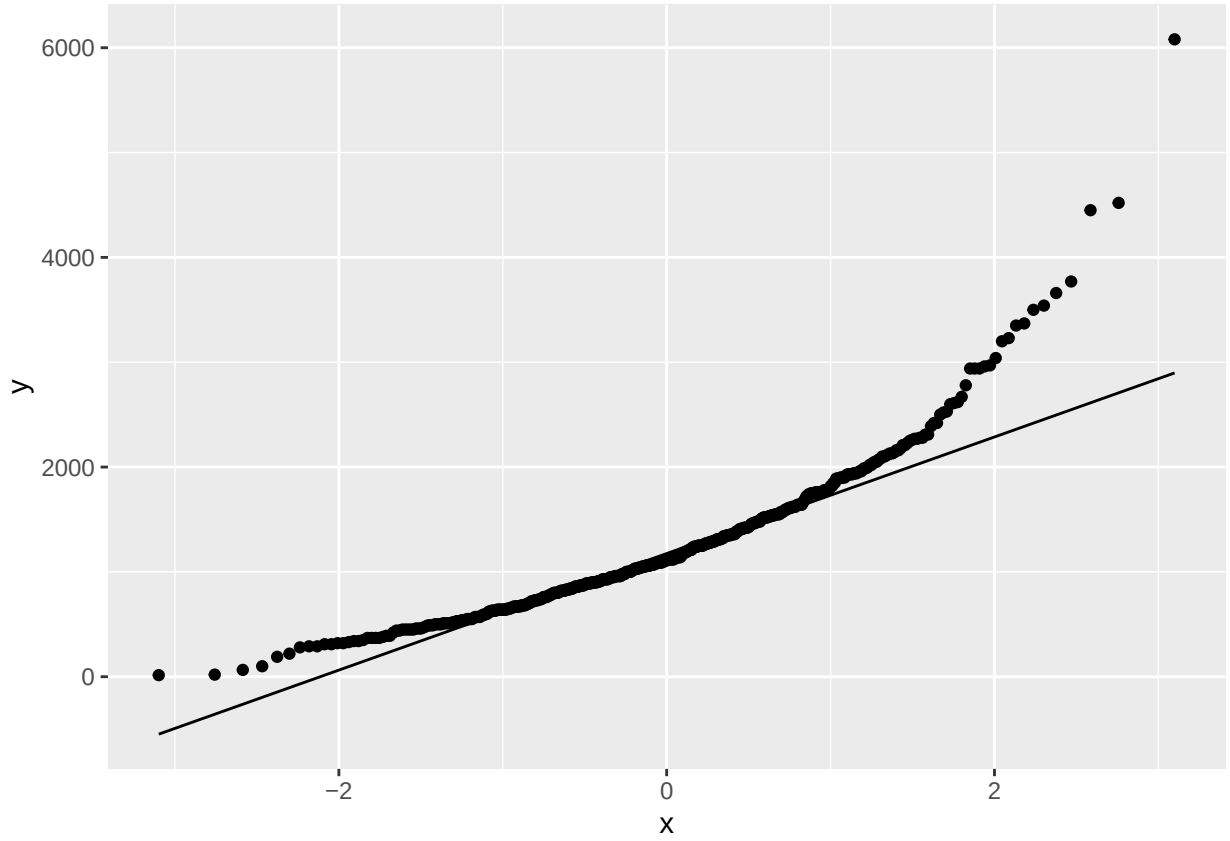
7. Now let's consider some of the other variables in the dataset. Out of all the different restaurants, which ones' distribution is the closest to normal for sodium?

To answer this question, we can first generate histograms and normal probability plots for the sodium variable of each restaurant using the same methods as before. Then we can visually inspect the plots to determine which one has the distribution that is closest to normal.

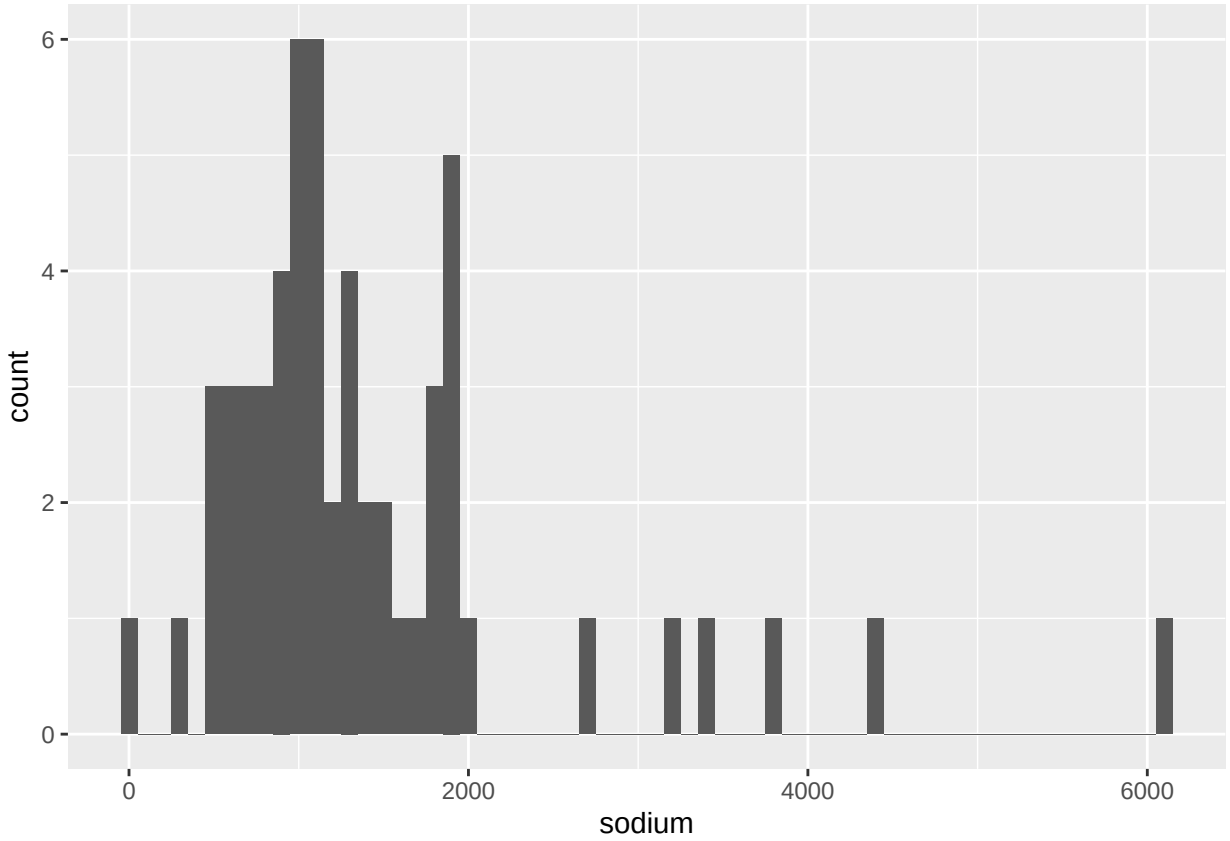
```
ggplot(data = fastfood, aes(x = sodium)) +
  geom_histogram(binwidth = 100)
```



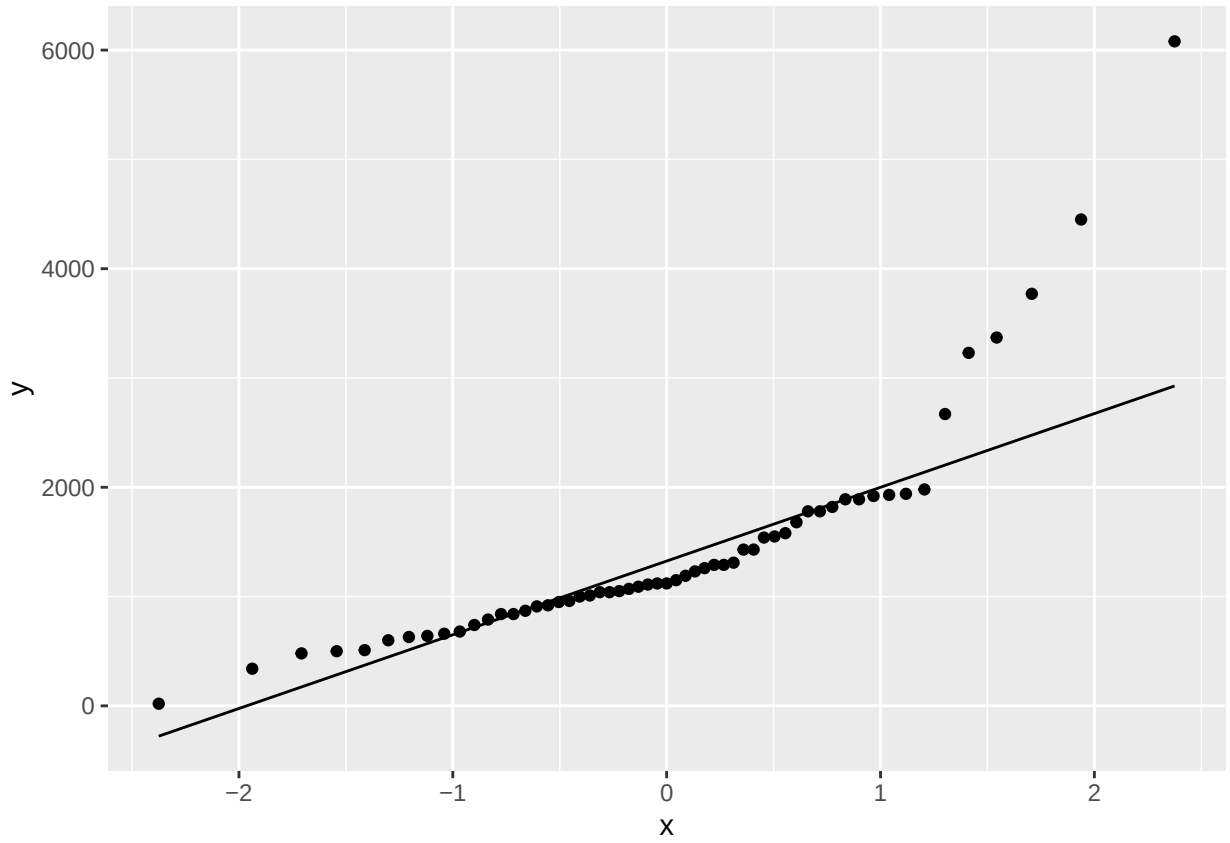

```
ggplot(data = fastfood, aes(sample = sodium)) +
  stat_qq() +
  stat_qq_line()
```



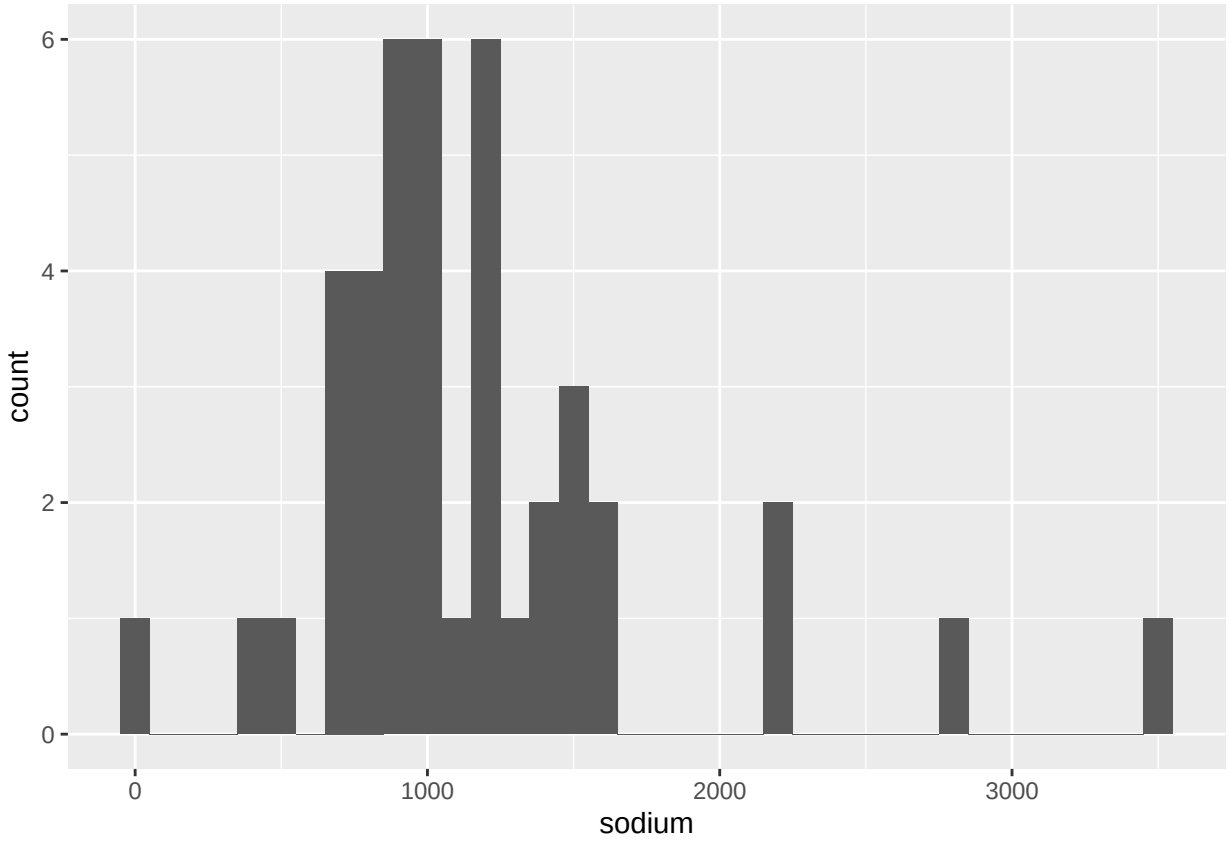

```
ggplot(data = mcdonalds, aes(x = sodium)) +
  geom_histogram(binwidth = 100)
```



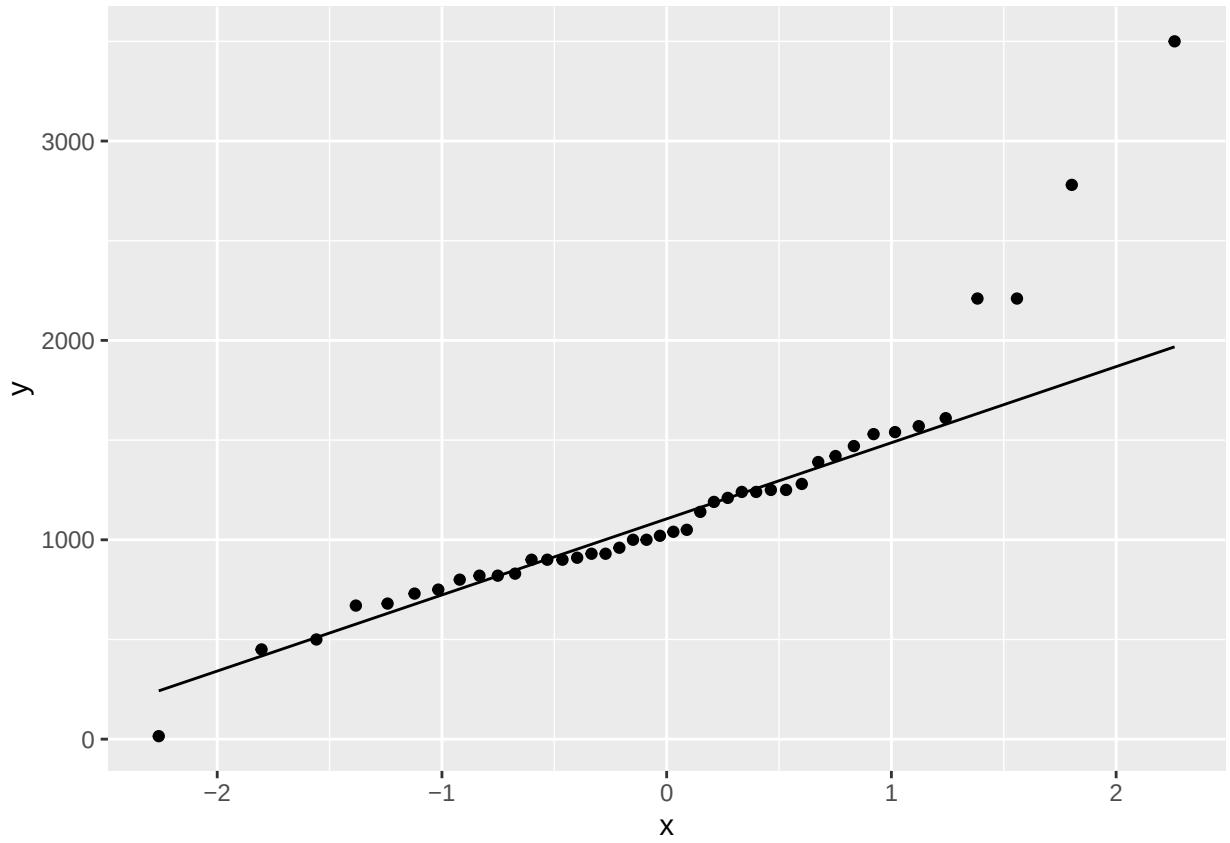

```
ggplot(data = mcdonalds, aes(sample = sodium)) +
  stat_qq() +
  stat_qq_line()
```




```
ggplot(data = dairy_queen, aes(x = sodium)) +
  geom_histogram(binwidth = 100)
```




```
ggplot(data = dairy_queen, aes(sample = sodium)) +
  stat_qq() +
  stat_qq_line()
```



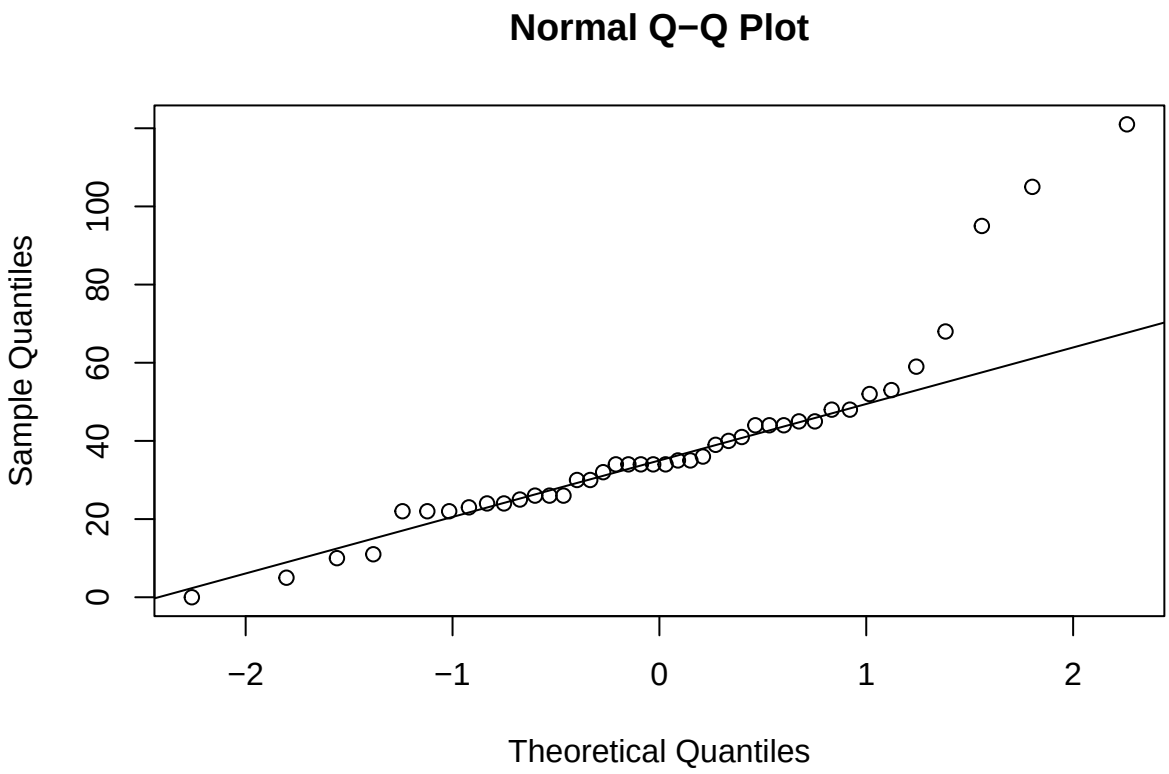
Based on visual inspection of the histograms and normal probability plots, it appears that the distribution of sodium in McDonald's menu items is the closest to normal.

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?

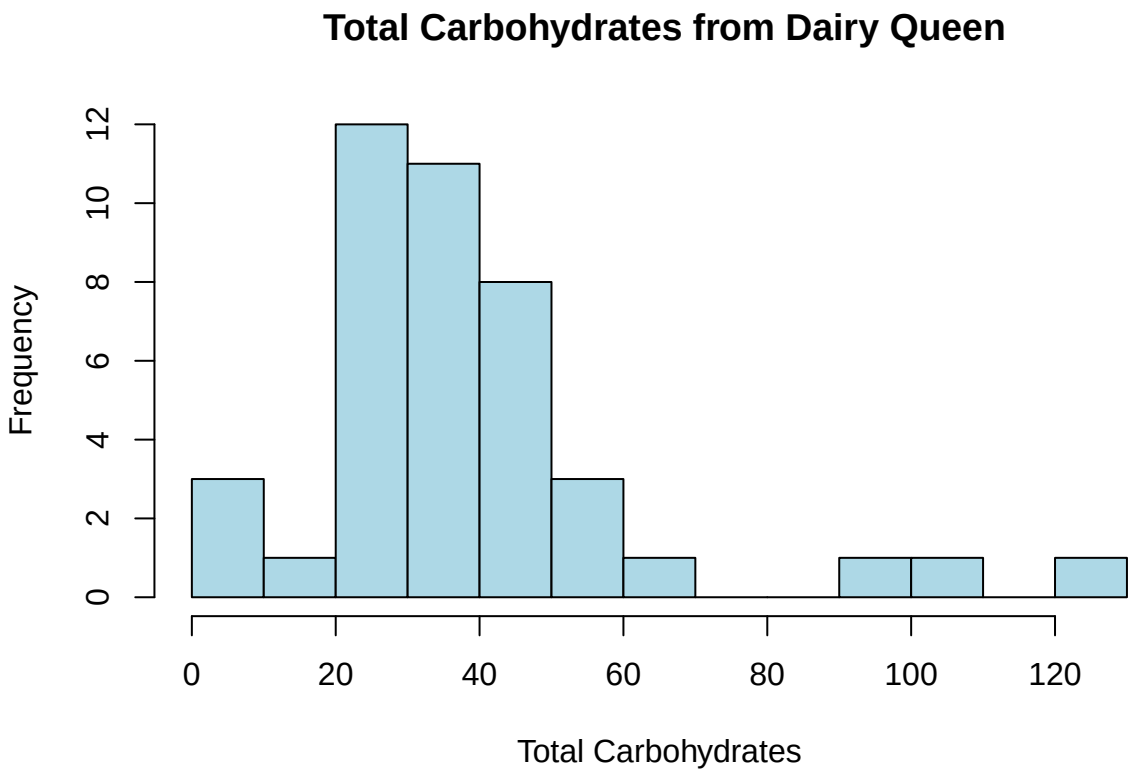
Maybe because the data is discrete rather than continuous.Also stepwise pattern seen in some normal probability plots for sodium distributions could be due to the fact that many food products contain a certain amount of salt (sodium chloride) as a preservative and flavor enhancer.

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.

```
# Creating a normal probability plot for total carbohydrates from Dairy Queen
carb_dq <- dairy_queen$total_carb
qqnorm(carb_dq)
qqline(carb_dq)
```




```r
# Creating a histogram for total carbohydrates from Dairy Queen
hist(carb_dq, breaks = 15, col = "lightblue ", xlab = "Total Carbohydrates", main = " Total Carbohydrat
```



Based on the normal probability plot and the histogram , the minimum value for total carbohydrates is 0, the median is 34, and the maximum is 121. This indicates that there may be some skewness in the data, as the minimum value is far from the median and mean. The histogram shows a longer tail to the right, and the normal probability plot shows that the points on the right side are above the expected line. This indicates that the data on the right side of the distribution is larger than expected if the distribution were normal.
