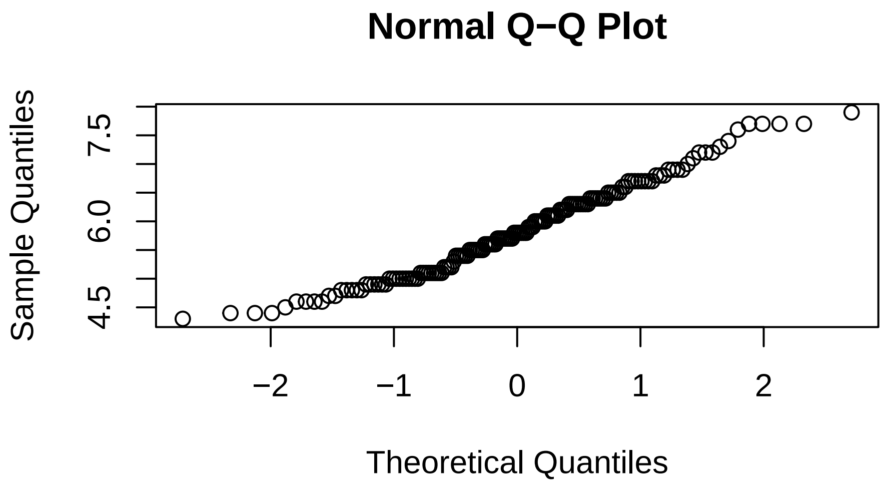
EPP 8273 Empirical Research in Theory and Practice

Q-Q Plots: Nuts and Bolts

1 Q-Q Plots

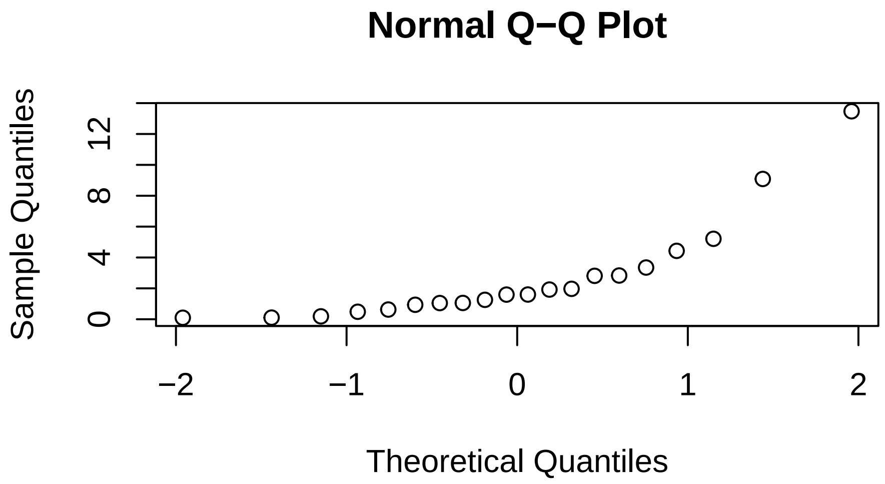
Q-Q Plots stands for Quantile-Quantile Plots. Quantiles are equivalent to percentiles and indicate a value below which a portion of your data are located. For example, the median is the 0.50 quantile and the 50th percentile: half of your data have values smaller than the median, half have larger values. To assess normality, we can plot the quantiles of your data (y-axis) vs. the quantiles of a theoretical sample we know is normally distributed (x-axis). The theoretical sample is comprised of the same number of observations as your actual sample, but it comes from a standard normal distribution for which the quantiles are exactly known. If the samples were distributed exactly alike, the data would fall on a perfectly straight line. Here is a decent looking Q-Q plot of `Sepal.Length` from the `iris` data:

```
qq_sepal <- qqnorm(iris$Sepal.Length)
```



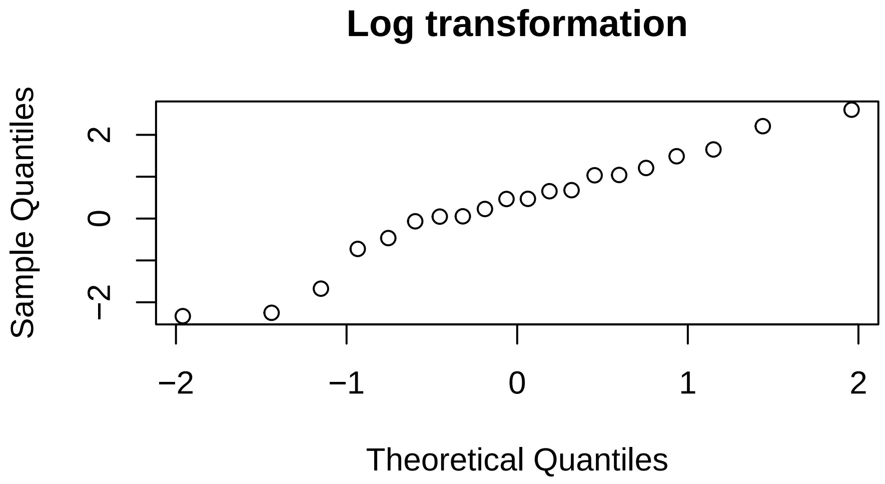
Here is an example of bad one, indicating the data are skewed

```
set.seed(123)
sample_data <- rexp(20,0.3)
qqnorm(sample_data)
```

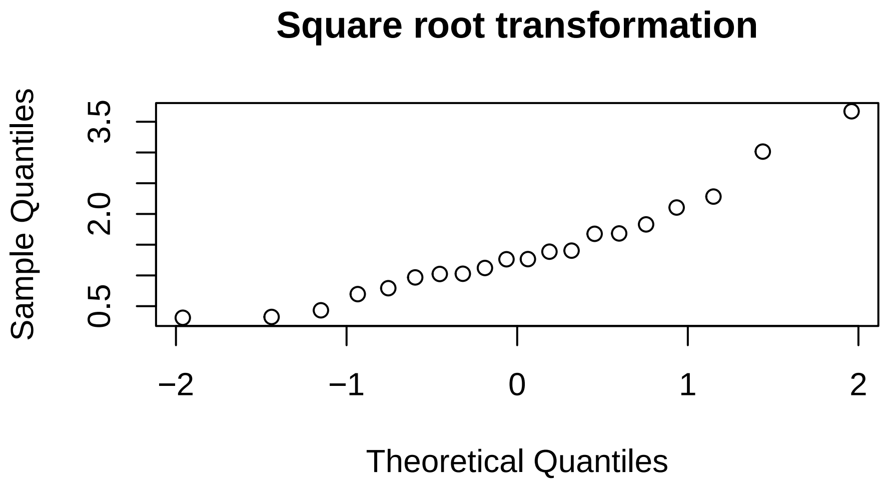


Either a log or square-root transformation helps though. There is some subjectivity here, but these look okay to me.

```r
qqnorm(log(sample_data), main="Log transformation")
```



```r
qqnorm(sqrt(sample_data), main="Square root transformation")
```



We will also look at Q-Q plots when we fit regressions. Therefore, it's worth understanding the nuts and bolts. Luckily, R has a function that does everything for us, but we will work through each component of a Q-Q plot in this section.

Quantiles are, by definition, between 0 and 1 (e.g., you cannot have -1% or 102% of the data). The first step is to sort the data (smallest to largest) and generate n=150 quantiles (we will calculate the quantile for each observation). These quantiles are obtained using `ppoints()` (which generates "`n`" equally spaced points between 0 and 1). Thus, our quantiles are equally spaced starting with 0 and ending at 1.

```
ordered_sample <- sort(iris$Sepal.Length)
head(ordered_sample)
```

```
## [1] 4.3 4.4 4.4 4.4 4.5 4.6
```

```
tail(ordered_sample)
```

```
## [1] 7.6 7.7 7.7 7.7 7.7 7.9
```

```
quantile_vec <- ppoints(n=length(ordered_sample))
head(quantile_vec)
```

```
## [1] 0.003333333 0.010000000 0.016666667 0.023333333 0.030000000 0.036666667
```

```
tail(quantile_vec)
```

```
## [1] 0.9633333 0.9700000 0.9766667 0.9833333 0.9900000 0.9966667
```

Then, we need to pair each observation in our sample with its corresponding quantile value. R has a useful command, `quantile()`, where `x` indicates a vector/string of numbers (our sample in this case) and `probs` indicates the quantile(s) we want to extract. For example, the median is the 0.50 (or 50%) quantile:

```
quantile(x=iris$Sepal.Length, probs=0.50)
```

```
## 50%
## 5.8
```

```r
median(iris$Sepal.Length)
```

```
## [1] 5.8
```

Since we have a string of 150 quantiles (`quantile_vec`), we will calculate the quantile for each observation:

```r
quantile_of_each_value <- quantile(x=iris$Sepal.Length, probs=quantile_vec)
head(quantile_of_each_value)
```

```
## 0.3333333% 1.0000000% 1.6666667% 2.3333333% 3.0000000% 3.6666667%
##   4.349667   4.400000   4.400000   4.447667   4.547000   4.600000
```

```r
tail(quantile_of_each_value)
```

```
## 96.3333333% 97.0000000% 97.6666667% 98.3333333% 99.0000000% 99.6666667%
##    7.507333    7.653000    7.700000    7.700000    7.700000    7.800667
```

Looking at the output above, this means that 3.67% of our data are smaller than 4.60 and 96.33% of our data are smaller than 7.5073. You could prove this "by hand" in R without much trouble:

```r
vals_smaller_than_7.5073 <- iris$Sepal.Length[iris$Sepal.Length < 7.5]
# n obs smaller than 7.5 / total obs in Sepal.Length
length(vals_smaller_than_7.5073)/length(iris$Sepal.Length)
```

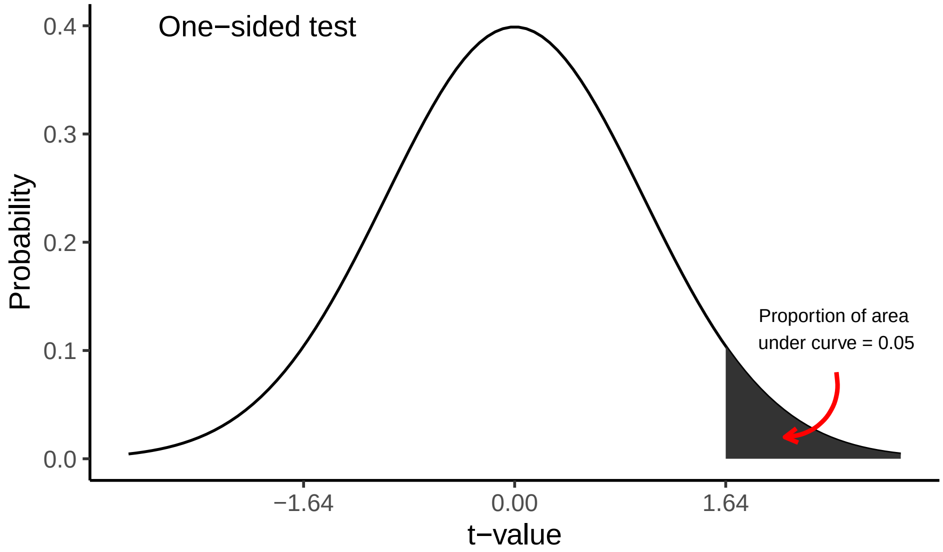
```
## [1] 0.96
```

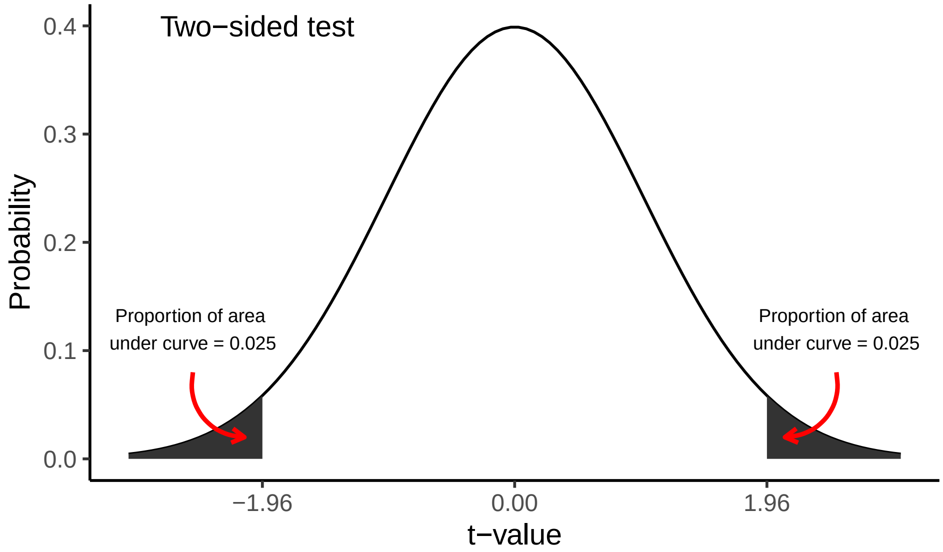
At this point, we have our sorted observations and their associated quantiles. We will now use those "associated quantiles" (`quantile_vec`) to obtain a theoretical sample from a standard normal distribution.

Think of it this way - in your undergrad stats course, you likely calculated a t-value and then searched through the back of a textbook for a t-table/t-distribution to find the corresponding p-value ("probability"-value). The quantiles of a t-distribution indicate the proportional area under the curve of a t-distribution… which is exactly what a p-value is! That is, a quantile value of 0.975 corresponds to t=~1.96 (assuming our sample size is large), meaning 97.5% of the area under a t-distribution is located between $-\infty$ and 1.96. If we conduct a t-test and get t=1.96, then our p-value is ~0.025. If our hypothesis is two-sided (e.g., sample A is "not equal" to sample B vs. "greater than" or "less than" than sample B), we need to multiply our p-value by 2 to account for that fact (`t.test()` does this for us when we specify `alternative="two.sided"`). SO, for two-sided hypotheses and large sample sizes, p-values $<$0.05 generally occur when t>1.96.

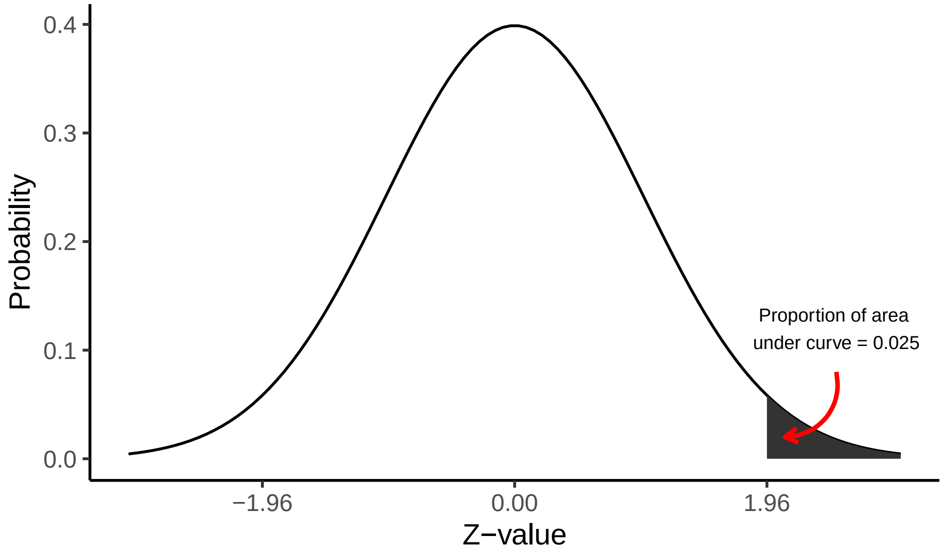
```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```





When you are calculating quantiles for your data, you are calculating the proportion of your data that occur below a certain value (we did this by hand above). This same logic and the logic of quantiles/p-values and t-values applies to standard normal distributions, from which our theoretical samples for Q-Q plots originate. The only difference is that standard normals use Z-values, and not t-values. Some of you may be wondering what the difference is between a t-distribution and the standard normal distribution. When the degrees of feedom approach ∞ for a t-distribution, the answer is absolutely nothing - they are the same. Here is a visual example of getting a Z-value from a quantile:



`qnorm()` gives the corresponding Z-values in a standard normal distribution for the specified quantiles. What follows a is (i) a proof of concept for the 0.975 quantile and (ii) the quantiles that correspond to the quantiles we estimated for our sample.

```
# proof of concept
round(qnorm(0.975),2)
```

```
## [1] 1.96
```

```
# quantiles corresponding to our sample
stand_norm_vals <- qnorm(quantile_vec)
head(stand_norm_vals)
```

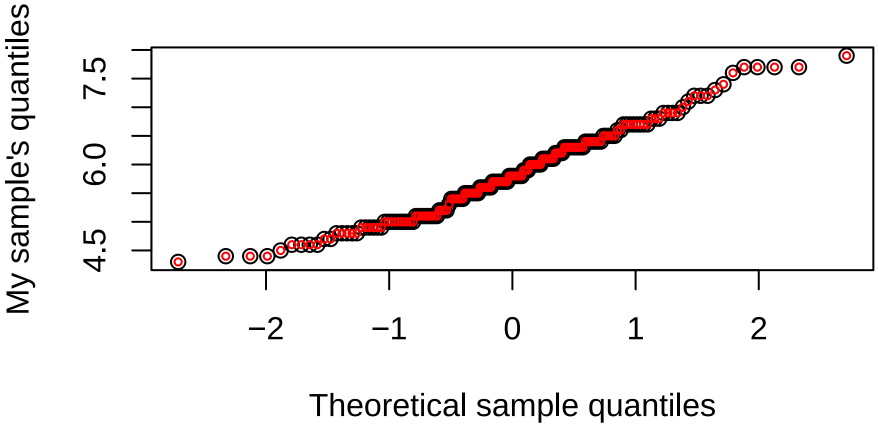
```
## [1] -2.713052 -2.326348 -2.128045 -1.989313 -1.880794 -1.790751
```

```
tail(stand_norm_vals)
```

```
## [1] 1.790751 1.880794 1.989313 2.128045 2.326348 2.713052
```

Now, compare the `qqnorm()` version (we created `qq_sepal` above) with our "by hand" version (red, smaller dots).

```
qq_sepal_x <- qq_sepal$x
qq_sepal_y <- qq_sepal$y
plot(qq_sepal_y~qq_sepal_x, xlab="Theoretical sample quantiles",
     ylab="My sample's quantiles")
points(ordered_sample~stand_norm_vals, col="red", cex=0.5)
```


My sample's quantiles
4.5
6.0
7.5
−2
−1
0
1
2
Theoretical sample quantiles