Statistical Testing

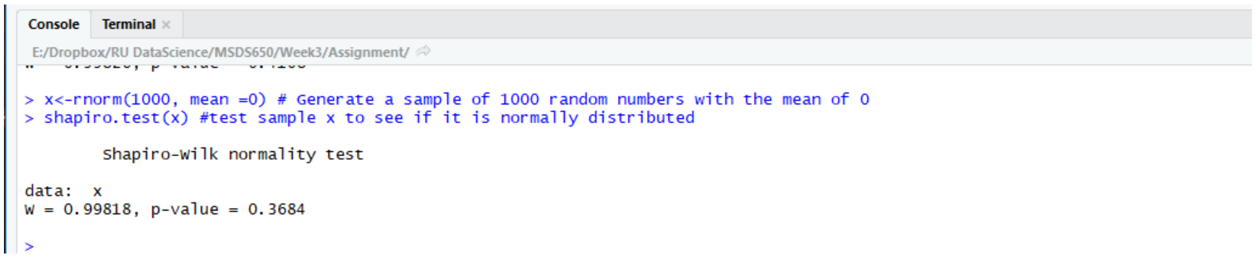
For this assignment I first performed exercises showing how Shapiro-Wilk test is implemented in R using the shapiro.test() function.

Exercise 1:

I generated a sample of 1,000 random numbers and applied Shapiro-Wilk test to it.

Console window output:

```
> x<-rnorm(1000, mean =0) # Generate a sample of 1000 random numbers with the mean of 0
> shapiro.test(x) #test sample x to see if it is normally distributed

         Shapiro-Wilk normality test

data:  x
W = 0.99818, p-value = 0.3684
```



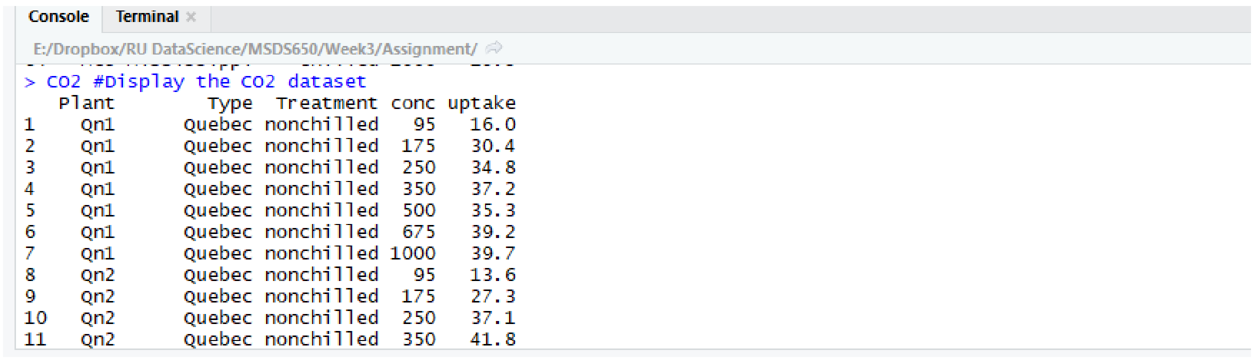
```
> x<-rnorm(1000, mean =0) # Generate a sample of 1000 random numbers with the mean of 0
> shapiro.test(x) #test sample x to see if it is normally distributed

        Shapiro-Wilk normality test

data:  x
W = 0.99818, p-value = 0.3684

>
```

This code yielded a p-value equal to 0.3684, which is greater than the significance level of 0.05, and therefor the null hypothesis is not rejected.

Exercise 2:

Then second exercised used the CO@ dataset, which comes with R, and describes experiments conducted on grass species. First, I displayed the dataset to visually inspect the data and see what variable to use for Shapiro-Wilk test.

```
> CO2 #Display the CO2 dataset
   Plant        Type  Treatment conc uptake
1    Qn1      Quebec nonchilled   95   16.0
2    Qn1      Quebec nonchilled  175   30.4
3    Qn1      Quebec nonchilled  250   34.8
4    Qn1      Quebec nonchilled  350   37.2
5    Qn1      Quebec nonchilled  500   35.3
6    Qn1      Quebec nonchilled  675   39.2
```

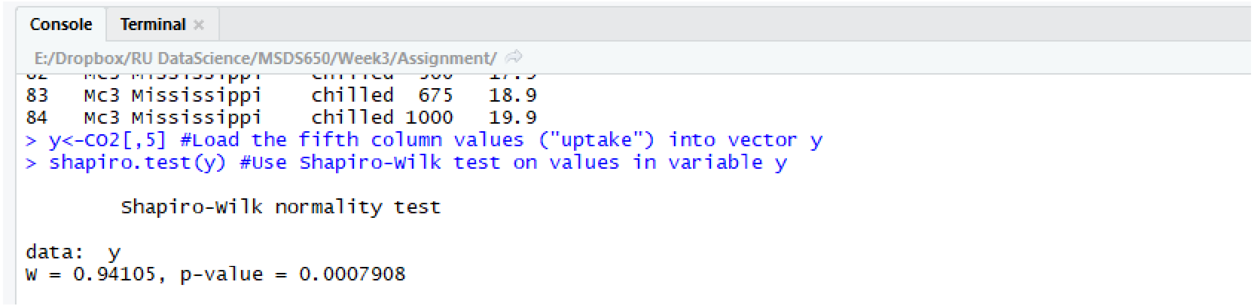

Console Terminal

E:/Dropbox/RU DataScience/MSDS650/Week3/Assignment/
```
> CO2 #Display the CO2 dataset
   Plant        Type  Treatment conc uptake
1    Qn1      Quebec nonchilled   95   16.0
2    Qn1      Quebec nonchilled  175   30.4
3    Qn1      Quebec nonchilled  250   34.8
4    Qn1      Quebec nonchilled  350   37.2
5    Qn1      Quebec nonchilled  500   35.3
6    Qn1      Quebec nonchilled  675   39.2
7    Qn1      Quebec nonchilled 1000   39.7
8    Qn2      Quebec nonchilled   95   13.6
9    Qn2      Quebec nonchilled  175   27.3
10   Qn2      Quebec nonchilled  250   37.1
11   Qn2      Quebec nonchilled  350   41.8
```

Column 5 – “uptake” – contains a numeric vector of carbon dioxide uptake rates (umol/m^2 sec) See https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/zCO2.html .

Load the content of the fifth column into a variable y and run the Shapiro-Wilk test on it using the following commands:

```
> y<-CO2[,5] #Load the fifth column values ("uptake") into vector y
> shapiro.test(y) #Use Shapiro-Wilk test on values in variable y

        Shapiro-Wilk normality test

data:  y
W = 0.94105, p-value = 0.0007908
```


Console Terminal

E:/Dropbox/RU DataScience/MSDS650/Week3/Assignment/
```
83   Mc3 Mississippi    chilled  675    18.9
84   Mc3 Mississippi    chilled 1000    19.9
> y<-CO2[,5] #Load the fifth column values ("uptake") into vector y
> shapiro.test(y) #Use Shapiro-Wilk test on values in variable y

        Shapiro-Wilk normality test

data:  y
W = 0.94105, p-value = 0.0007908
```

As the above output shows, the p-value is much smaller than 0.05 (p= 0.0007908) which provides us enough evidence to reject the null hypothesis in favor of the alternative hypothesis. It means that CO2 uptake rates are not normally distributed.

Exercise 3. Normality and Testing for Normality. Based on article by Thomas Hopper (https://www.r-bloggers.com/normality-and-testing-for-normality/)

Statistical Testing

In this exercised I explored relationship between the size of the sample, shape of the distribution and results of Shapiro-Wilk normality test.

First, I loaded ggplot2 and reshape2 libraries and the code for user-defined function that is used to perform multiple Shapiro-Wilk tests for three sample sizes (5, 10 and 1,000) drawn from the same data.

Console window output:

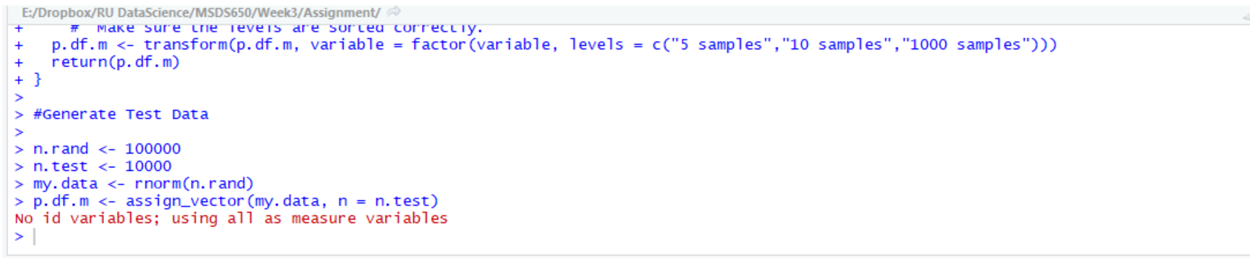
```
> library(ggplot2)
> library(reshape2)
> #Variables:
> #' @name assign_vector
> #' @param data A vector of data to perform the t-test on.
> #' @param n An integer indicating the number of t-tests to perform. Default is 1000
> #' @return A data frame in "tall" format
>
> assign_vector <- function(data, n = 1000) {
+   # replicate the call to shapiro.test n times to build up a vector of p-values
+   p.5 <- replicate(n=n, expr=shapiro.test(sample(my.data, 5, replace=TRUE))$p.value)
+   p.10 <- replicate(n=n, expr=shapiro.test(sample(my.data, 10, replace=TRUE))$p.value)
+   p.1000 <- replicate(n=n, expr=shapiro.test(sample(my.data, 1000, replace=TRUE))$p.value)
+   #' Combine the data into a data frame,
+   #' one column for each number of samples tested.
+   p.df <- cbind(p.5, p.10, p.1000)
+   p.df <- as.data.frame(p.df)
+   colnames(p.df) <- c("5 samples","10 samples","1000 samples")
+   #' Put the data in "tall" format, one column for number of samples
+   #' and one column for the p-value.
+   p.df.m <- melt(p.df)
+   #melt function from reshape2 package stacks several groups into the same column and
+   #creates a factor variable to indicate which group of variables it corresponds to
+     #' Make sure the levels are sorted correctly.
+   p.df.m <- transform(p.df.m, variable = factor(variable, levels = c("5 samples","10 samples","
1000 samples")))
+   return(p.df.m)
+ }
```

Note: **melt()** function from the reshape2 package that stacks several groups into the same column and creates a factor variable to indicate which group of variables it corresponds to.

Next, I generated the test data using the following code:

```
> #Generate Test Data
>
> n.rand <- 100000
> n.test <- 10000
> my.data <- rnorm(n.rand)
> p.df.m <- assign_vector(my.data, n = n.test)
```

It displayed a warning (see screen shot below) on the console window but had no effect on any of the results.

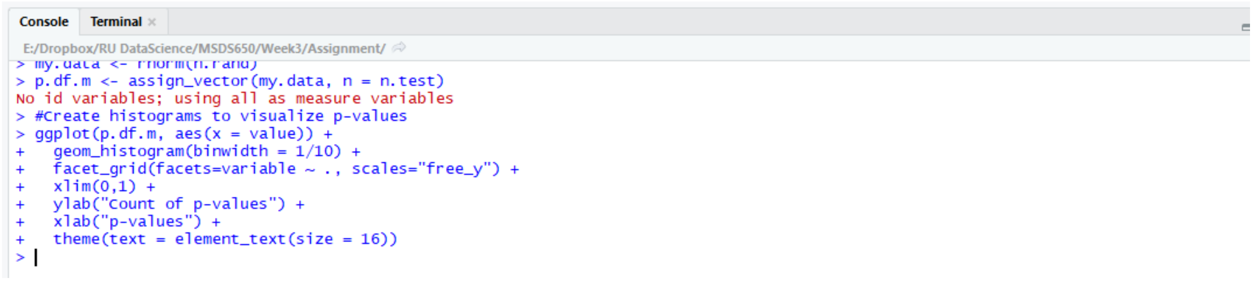


```
E:/Dropbox/RU DataScience/MSDS650/Week3/Assignment/
+     # make sure the levels are sorted correctly.
+   p.df.m <- transform(p.df.m, variable = factor(variable, levels = c("5 samples","10 samples","1000 samples")))
+   return(p.df.m)
+ }
>
> #Generate Test Data
>
> n.rand <- 100000
> n.test <- 10000
> my.data <- rnorm(n.rand)
> p.df.m <- assign_vector(my.data, n = n.test)
No id variables; using all as measure variables
> |
```

Next, I used histograms to visualize probabilities for any given p-values, which are expected to be approximately equal in case of **normal data**.

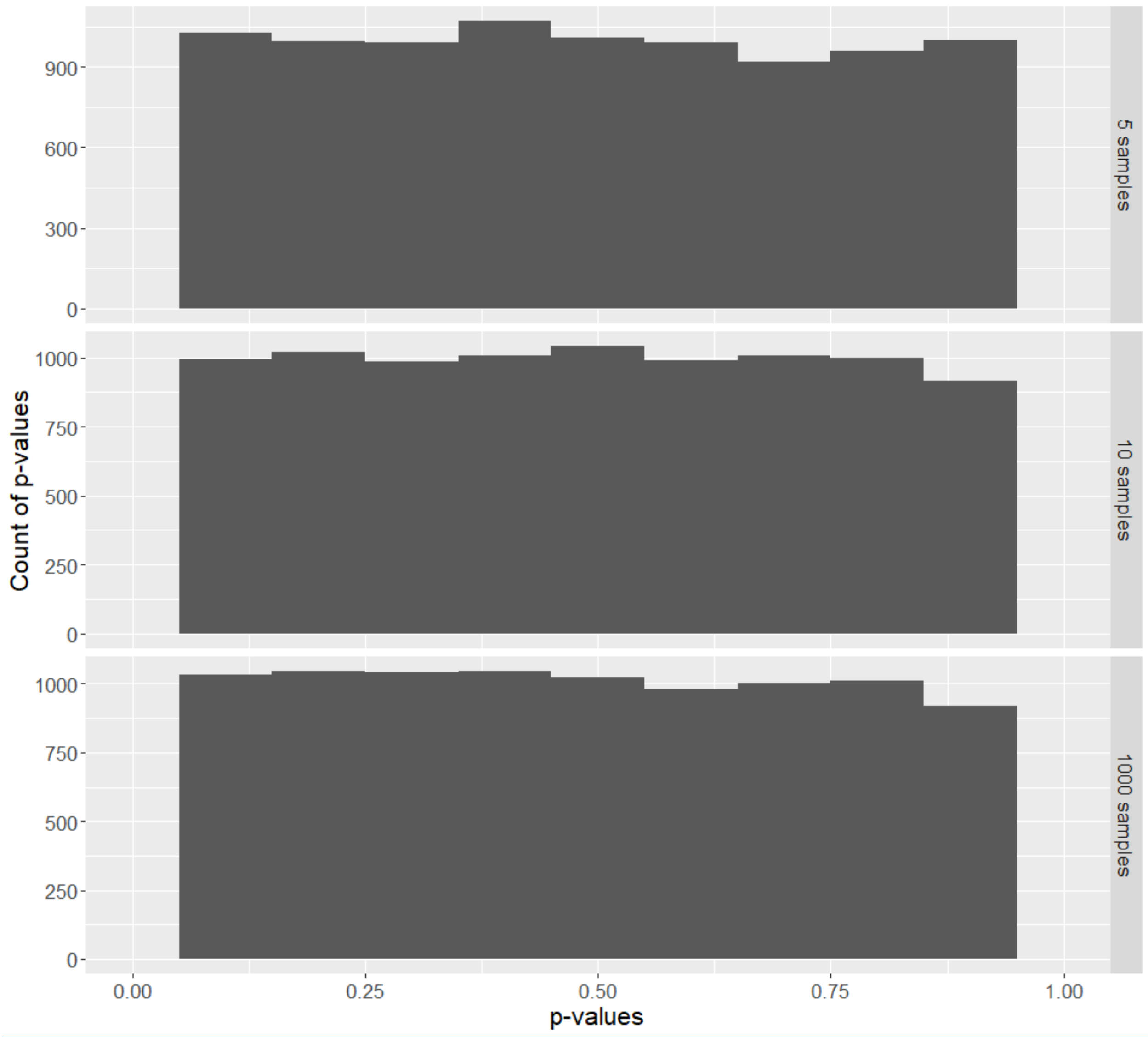
```
> #Create histograms to visualize p-values
> ggplot(p.df.m, aes(x = value)) +
+   geom_histogram(binwidth = 1/10) +
+   facet_grid(facets=variable ~ ., scales="free_y") +
+   xlim(0,1) +
+   ylab("Count of p-values") +
+   xlab("p-values") +
+   theme(text = element_text(size = 16))

>
```

Console window output:



```
Console   Terminal
E:/Dropbox/RU DataScience/MSDS650/Week3/Assignment/
> my.data <- rnorm(n.rand)
> p.df.m <- assign_vector(my.data, n = n.test)
No id variables; using all as measure variables
> #Create histograms to visualize p-values
> ggplot(p.df.m, aes(x = value)) +
+   geom_histogram(binwidth = 1/10) +
+   facet_grid(facets=variable ~ ., scales="free_y") +
+   xlim(0,1) +
+   ylab("Count of p-values") +
+   xlab("p-values") +
+   theme(text = element_text(size = 16))
> |
```

The resulting graphs are shown below:



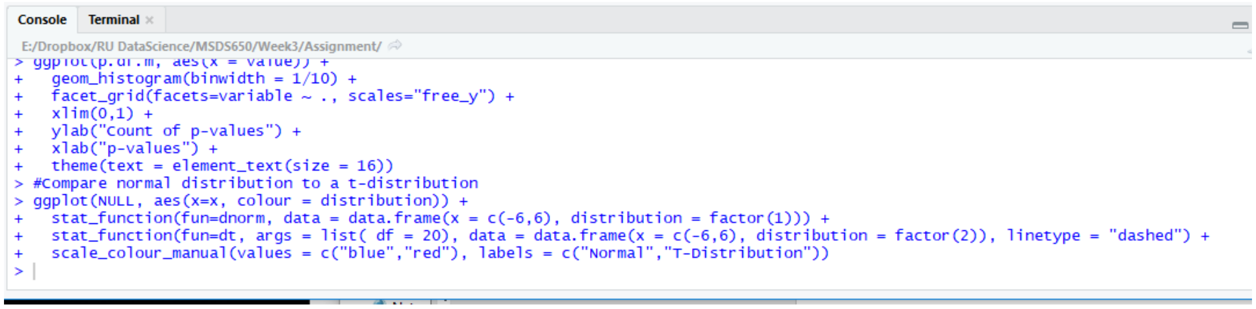
Conclusions:

As expected, the resulting probabilities for p-values are approximately equal for all three samples.

In the next step, I compared normal and t-distribution which look very close during visual inspection.

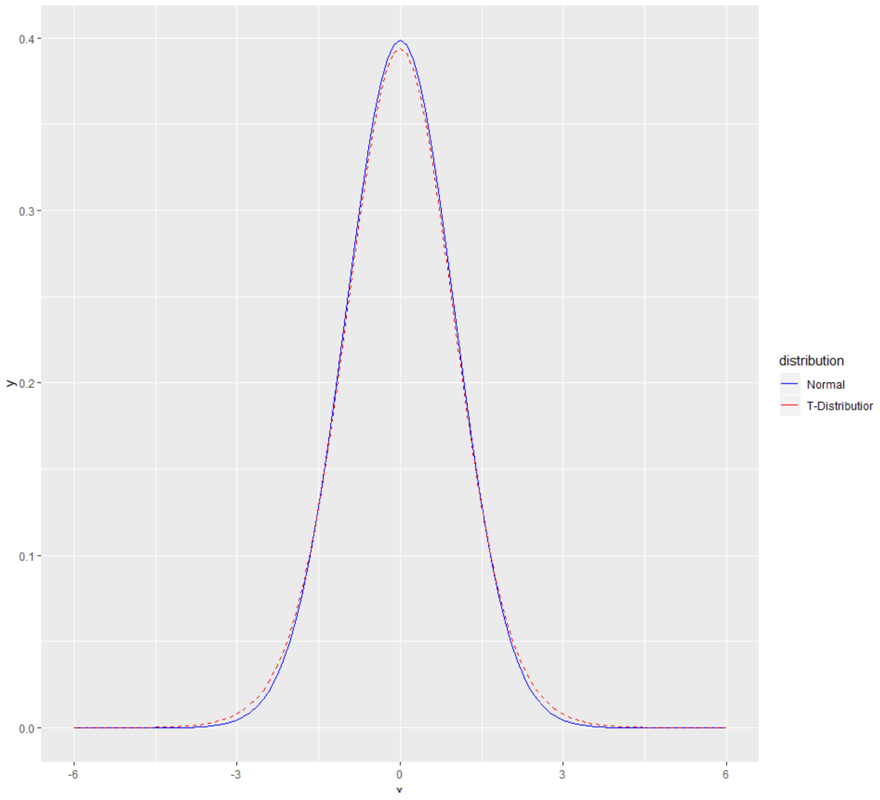
The following code was used to compare two graphs:

```
> #Compare normal distribution to a t-distribution
> ggplot(NULL, aes(x=x, colour = distribution)) +
+   stat_function(fun=dnorm, data = data.frame(x = c(-6,6), distribution = factor(1))) +
+   stat_function(fun=dt, args = list( df = 20), data = data.frame(x = c(-6,6), distribution = fa
ctor(2)), linetype = "dashed") +
+   scale_colour_manual(values = c("blue","red"), labels = c("Normal","T-Distribution"))
```



```
> ggplot(p.df.m, aes(x = value)) +
+   geom_histogram(binwidth = 1/10) +
+   facet_grid(facets=variable ~ ., scales="free_y") +
+   xlim(0,1) +
+   ylab("Count of p-values") +
+   xlab("p-values") +
+   theme(text = element_text(size = 16))
> #Compare normal distribution to a t-distribution
> ggplot(NULL, aes(x=x, colour = distribution)) +
+   stat_function(fun=dnorm, data = data.frame(x = c(-6,6), distribution = factor(1))) +
+   stat_function(fun=dt, args = list( df = 20), data = data.frame(x = c(-6,6), distribution = factor(2)), linetype = "dashed") +
+   scale_colour_manual(values = c("blue","red"), labels = c("Normal","T-Distribution"))
> |
```

The two resulting distributions are almost indistinguishable visually:



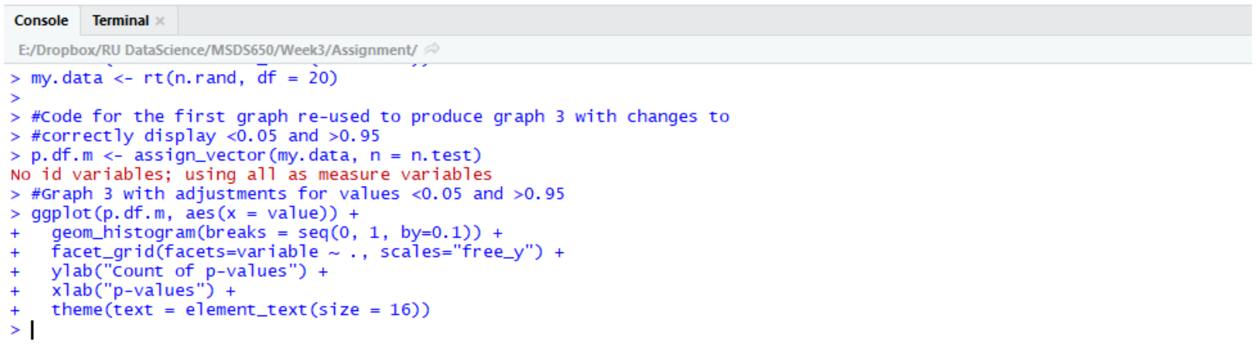
Next, collect a random sample from the data generated using **t-distribution:**

```
> my.data <- rt(n.rand, df = 20)
>
```

My first attempt to graphically present probabilities resulting in the histograms that omitted values at the both extremities. So, I adjusted the code as suggested in the assignment tips.

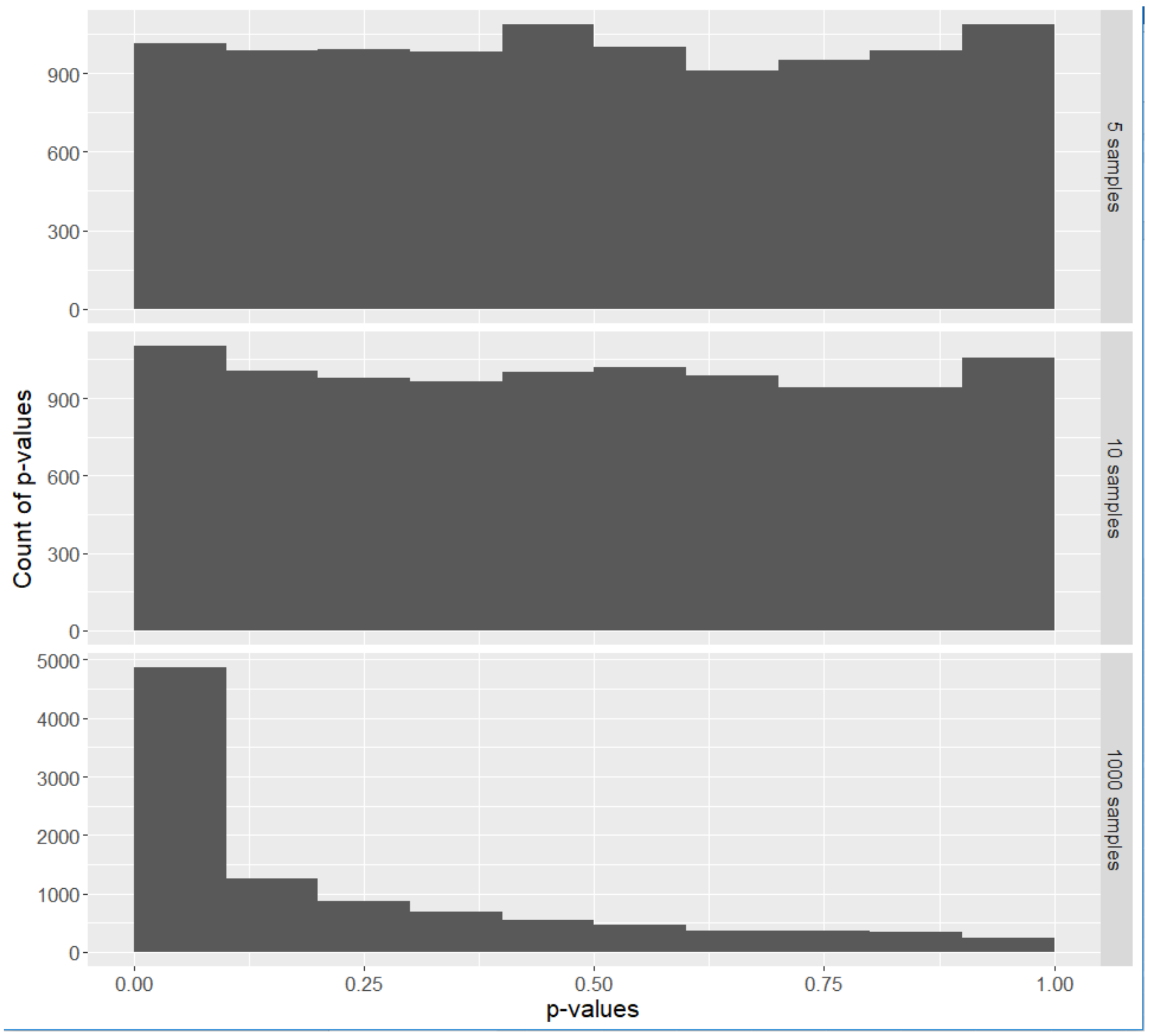
```
> #Code for the first graph re-used to produce graph 3 with changes to
> #correctly display <0.05 and >0.95
> p.df.m <- assign_vector(my.data, n = n.test)
No id variables; using all as measure variables
> #Graph 3 with adjustments for values <0.05 and >0.95
> ggplot(p.df.m, aes(x = value)) +
+   geom_histogram(breaks = seq(0, 1, by=0.1)) +
+   facet_grid(facets=variable ~ ., scales="free_y") +
+   ylab("Count of p-values") +
+   xlab("p-values") +
+   theme(text = element_text(size = 16))
```

Console window output:



```
E:/Dropbox/RU DataScience/MSDS650/Week3/Assignment/
> my.data <- rt(n.rand, df = 20)
>
> #Code for the first graph re-used to produce graph 3 with changes to
> #correctly display <0.05 and >0.95
> p.df.m <- assign_vector(my.data, n = n.test)
No id variables; using all as measure variables
> #Graph 3 with adjustments for values <0.05 and >0.95
> ggplot(p.df.m, aes(x = value)) +
+   geom_histogram(breaks = seq(0, 1, by=0.1)) +
+   facet_grid(facets=variable ~ ., scales="free_y") +
+   ylab("Count of p-values") +
+   xlab("p-values") +
+   theme(text = element_text(size = 16))
> |
```

The resulting graphs are displayed below:



Conclusions:

For small sample sizes (5 and 10) the histograms of p-values for t-distribution look very close to similar graphs for normal distribution. It suggests, that Shapiro-Wilk test is not sensitive enough to detect departure from normality in care of small sample sizes.

Even in case of bigger sample sizes (1,000) drawn from a t-distribution, for p values below 0.05 Shapiro-Wilk test fails to reject normality hypothesis in about 50% of cases.

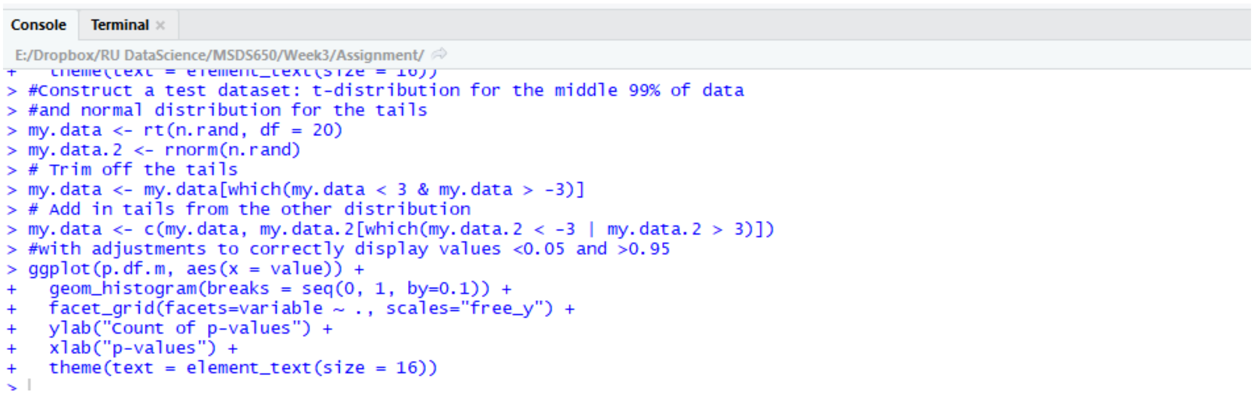
Next, I explored possible differences in the ways main part and tails of the distribution influence the results of Shapiro-Wilk tests.

t-distribution in the middle and normal tails

First, I constructed a test data set containing 99% percent of data in the middle coming from a t-distribution and 1% in extremities coming from a normal distribution and displayed the results as histograms of p-values using the following code.

```
> #Construct a test dataset: t-distribution for the middle 99% of data
> #and normal distribution for the tails
> my.data <- rt(n.rand, df = 20)
> my.data.2 <- rnorm(n.rand)
> # Trim off the tails
> my.data <- my.data[which(my.data < 3 & my.data > -3)]
> # Add in tails from the other distribution
> my.data <- c(my.data, my.data.2[which(my.data.2 < -3 | my.data.2 > 3)])
> #with adjustments to correctly display values <0.05 and >0.95
> ggplot(p.df.m, aes(x = value)) +
+   geom_histogram(breaks = seq(0, 1, by=0.1)) +
+   facet_grid(facets=variable ~ ., scales="free_y") +
+   ylab("Count of p-values") +
+   xlab("p-values") +
+   theme(text = element_text(size = 16))
```

Console widow output:

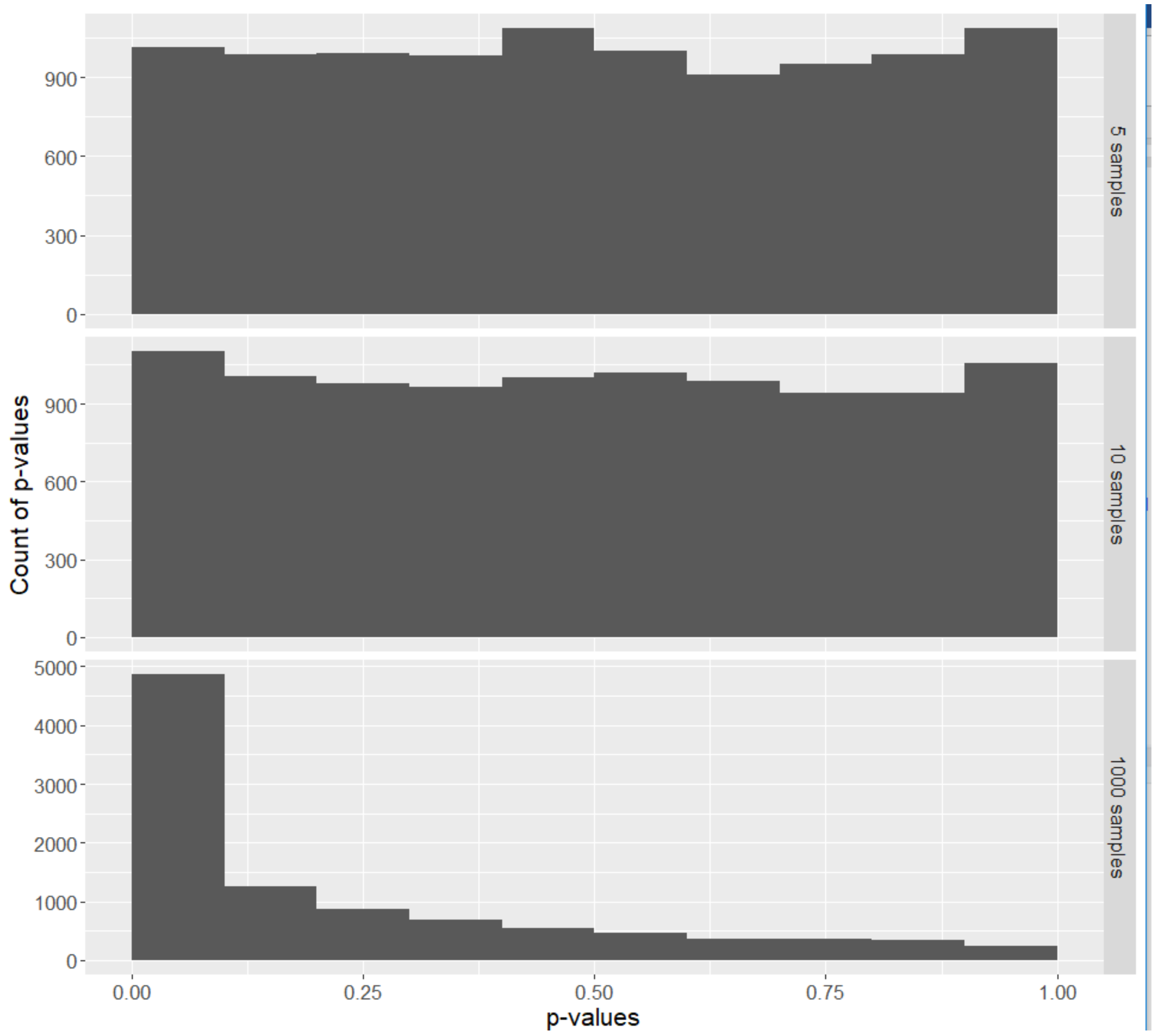


Console Terminal

E:/Dropbox/RU DataScience/MSDS650/Week3/Assignment/

```
+   theme(text = element_text(size = 16))
> #Construct a test dataset: t-distribution for the middle 99% of data
> #and normal distribution for the tails
> my.data <- rt(n.rand, df = 20)
> my.data.2 <- rnorm(n.rand)
> # Trim off the tails
> my.data <- my.data[which(my.data < 3 & my.data > -3)]
> # Add in tails from the other distribution
> my.data <- c(my.data, my.data.2[which(my.data.2 < -3 | my.data.2 > 3)])
> #with adjustments to correctly display values <0.05 and >0.95
> ggplot(p.df.m, aes(x = value)) +
+   geom_histogram(breaks = seq(0, 1, by=0.1)) +
+   facet_grid(facets=variable ~ ., scales="free_y") +
+   ylab("Count of p-values") +
+   xlab("p-values") +
+   theme(text = element_text(size = 16))
> 
```

Graphical output:



Conclusions:

The resulting histograms show that Shapiro-Wilk test failed to provide enough proof to reject the normality hypothesis despite the fact that 99% percent of data comes from a t-distribution. Histograms for 5 and 10 sample sizes look almost identical to the normal distribution.

Tails of the distribution might have disproportionately bigger impact on Shapiro-Wilk normality test compared to the main body of the distribution.

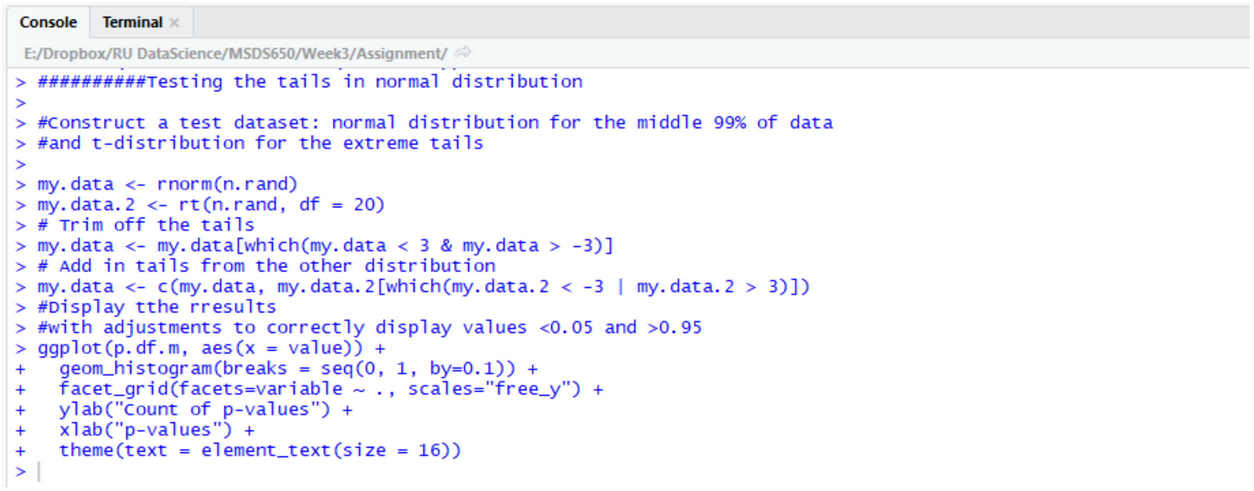
Next step -testing the opposite: 99% of data in the middle come from a normal distribution, with only 1% of extreme tails coming from a t-distribution.

Statistical Testing

After constructing a sample, I followed a similar testing procedure by using the following code:

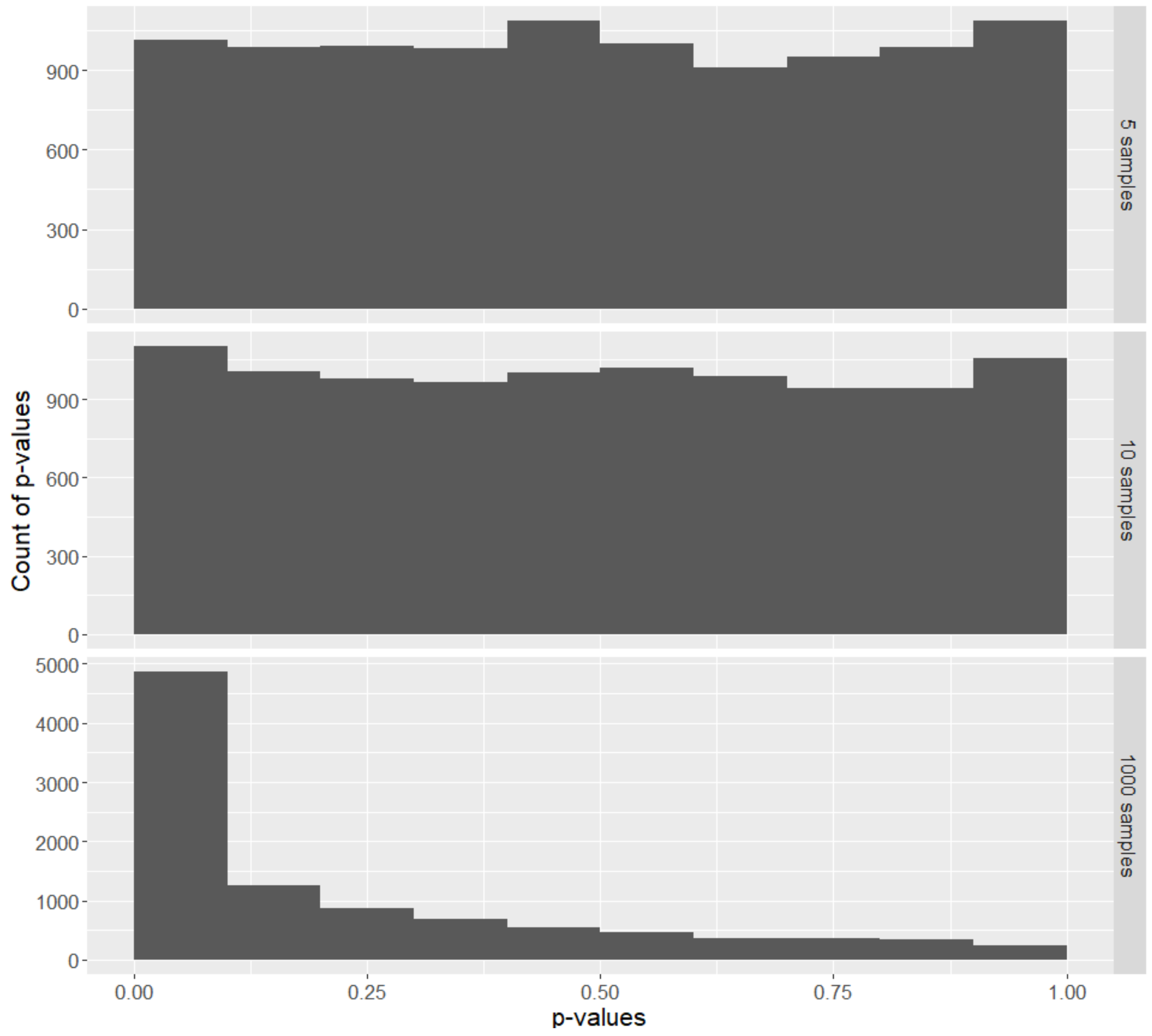
```
##########Testing the tails in normal distribution
>
> #Construct a test dataset: normal distribution for the middle 99% of data
> #and t-distribution for the extreme tails
>
> my.data <- rnorm(n.rand)
> my.data.2 <- rt(n.rand, df = 20)
> # Trim off the tails
> my.data <- my.data[which(my.data < 3 & my.data > -3)]
> # Add in tails from the other distribution
> my.data <- c(my.data, my.data.2[which(my.data.2 < -3 | my.data.2 > 3)])
> #Display tthe rresults
> #with adjustments to correctly display values <0.05 and >0.95
> ggplot(p.df.m, aes(x = value)) +
+   geom_histogram(breaks = seq(0, 1, by=0.1)) +
+   facet_grid(facets=variable ~ ., scales="free_y") +
+   ylab("Count of p-values") +
+   xlab("p-values") +
+   theme(text = element_text(size = 16))
```

Console window output:



```
> ##########Testing the tails in normal distribution
>
> #Construct a test dataset: normal distribution for the middle 99% of data
> #and t-distribution for the extreme tails
>
> my.data <- rnorm(n.rand)
> my.data.2 <- rt(n.rand, df = 20)
> # Trim off the tails
> my.data <- my.data[which(my.data < 3 & my.data > -3)]
> # Add in tails from the other distribution
> my.data <- c(my.data, my.data.2[which(my.data.2 < -3 | my.data.2 > 3)])
> #Display tthe rresults
> #with adjustments to correctly display values <0.05 and >0.95
> ggplot(p.df.m, aes(x = value)) +
+   geom_histogram(breaks = seq(0, 1, by=0.1)) +
+   facet_grid(facets=variable ~ ., scales="free_y") +
+   ylab("Count of p-values") +
+   xlab("p-values") +
+   theme(text = element_text(size = 16))
> |
```

This yielded the following results:



Conclusions:

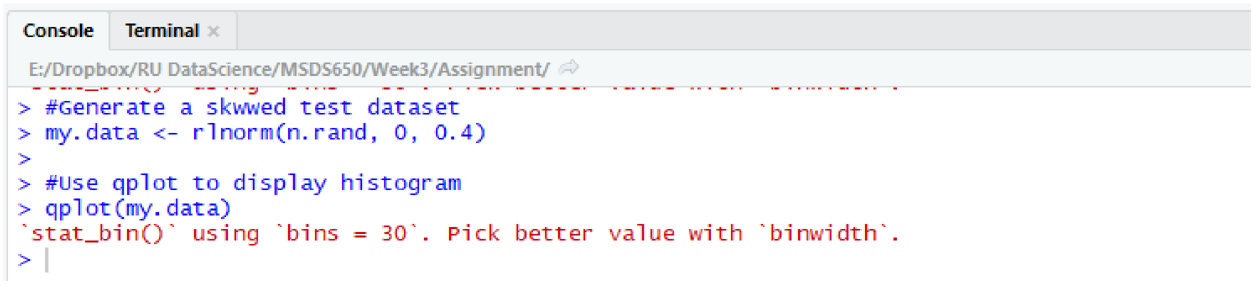
Histograms for this composite sample look very similar to the results provided by applying Shapiro-Wilk test to the samples drawn from t-distributed data, especially for small sample sizes. For the 1,000 sample the probability of p value <=0.05 is only slightly less that for the t-distribution.

Testing highly skewed data

In order to examine a possible effect of skewness on the results of Shapiro-Wilk normality test, I applied similar procedures to a sample with highly skewed data. To create a sample I used the following:

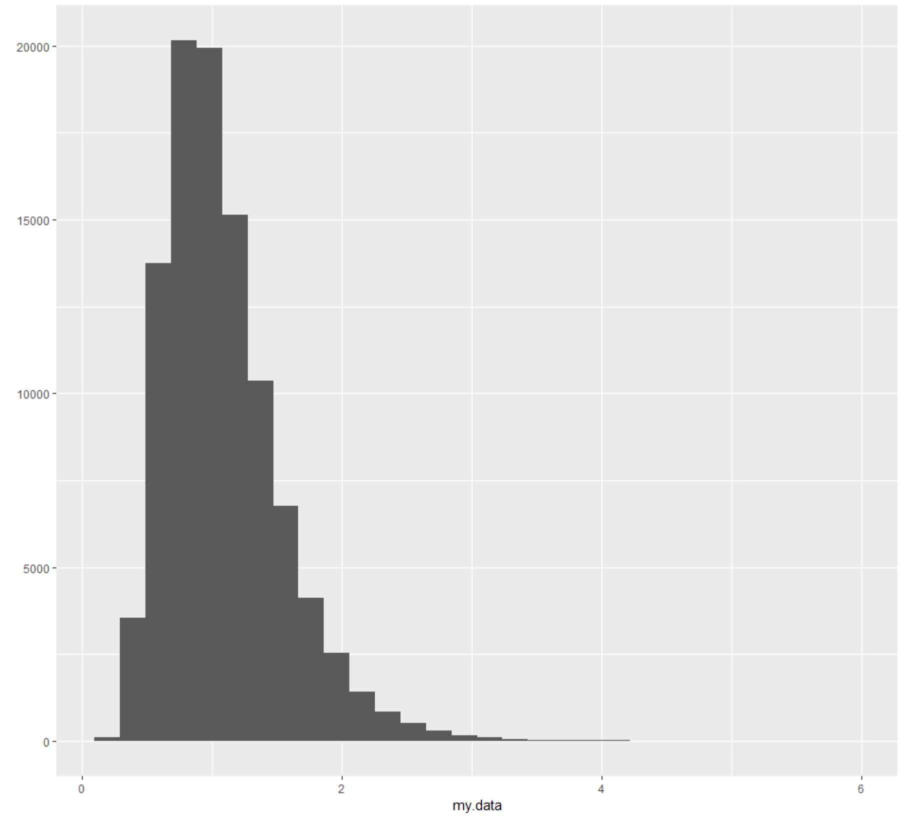
```
> my.data <- rlnorm(n.rand, 0, 0.4)
> qplot(my.data)
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

I used quick plot function **qplot()** to inspect the sample:



```
> #Generate a skwwed test dataset
> my.data <- rlnorm(n.rand, 0, 0.4)
>
> #Use qplot to display histogram
> qplot(my.data)
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
>
```

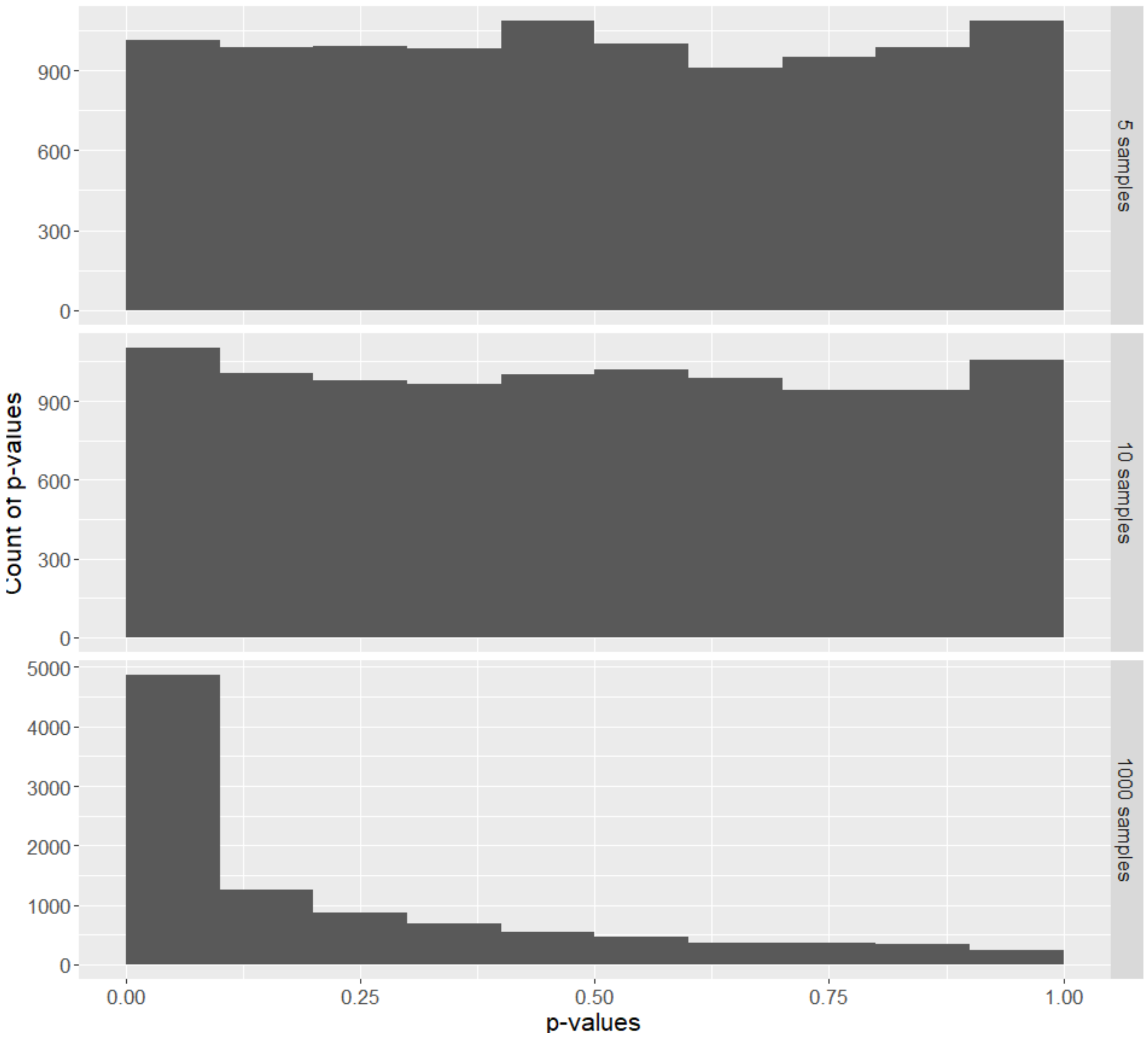
The resulting histogram looked like this:



Next, as in previous test cases, I used the following code to compare p-value frequencies for the three sample sizes:

```
> #Display the results
> #with adjustments to correctly display values <0.05 and >0.95
> ggplot(p.df.m, aes(x = value)) +
+   geom_histogram(breaks = seq(0, 1, by=0.1)) +
+   facet_grid(facets=variable ~ ., scales="free_y") +
+   ylab("Count of p-values") +
+   xlab("p-values") +
+   theme(text = element_text(size = 16))
```

The resulting histograms are presented below:



Conclusions:

For small sample sizes (5 and 10) p-values distributions look normal. Highly skewed data using small sample sizes is very likely to pass the Shapiro-Wilk normality test.

Overall Conclusions:

- The Shapiro-Wilk normality test is sensitive to the sample size. It can not provide reliable results in case of small sample sizes.

- The normality test is very sensitive to the distribution of data in the extremities (tails), even if the main body of the data might have a different distribution.
- Neither visual tools or normality testing alone can provide reliable results in determining normality of the underlying distribution. They should be used together.