Module 7 Lab

In this lab we will explore, through simulations, the robustness of the t-test to departure from assumptions. Namely, we examine the impact of small sample sizes and non-Normality.

Rejection Rates and Uniform p-values

First, we use a familiar case—a one-sample t-test for Normal data—to review the relationship between Uniform p-values, rejection rates, and valid testing procedures. We will write a short function that randomly generates a sample of Normal data, runs a t-test on the data, and extracts the p-value.

```
pval <- function(n = 25){ # Default sample size of 25
  x <- rnorm(n, mean = 0, sd = 1) # Generate N(0,1) sample
  test <- t.test(x, mu = 0) # Run t-test on sample, with null hypoth. H_0: mu = 0
  test$p.value # Extract and return p-value
}

pval() # Run function to confirm it is working
```

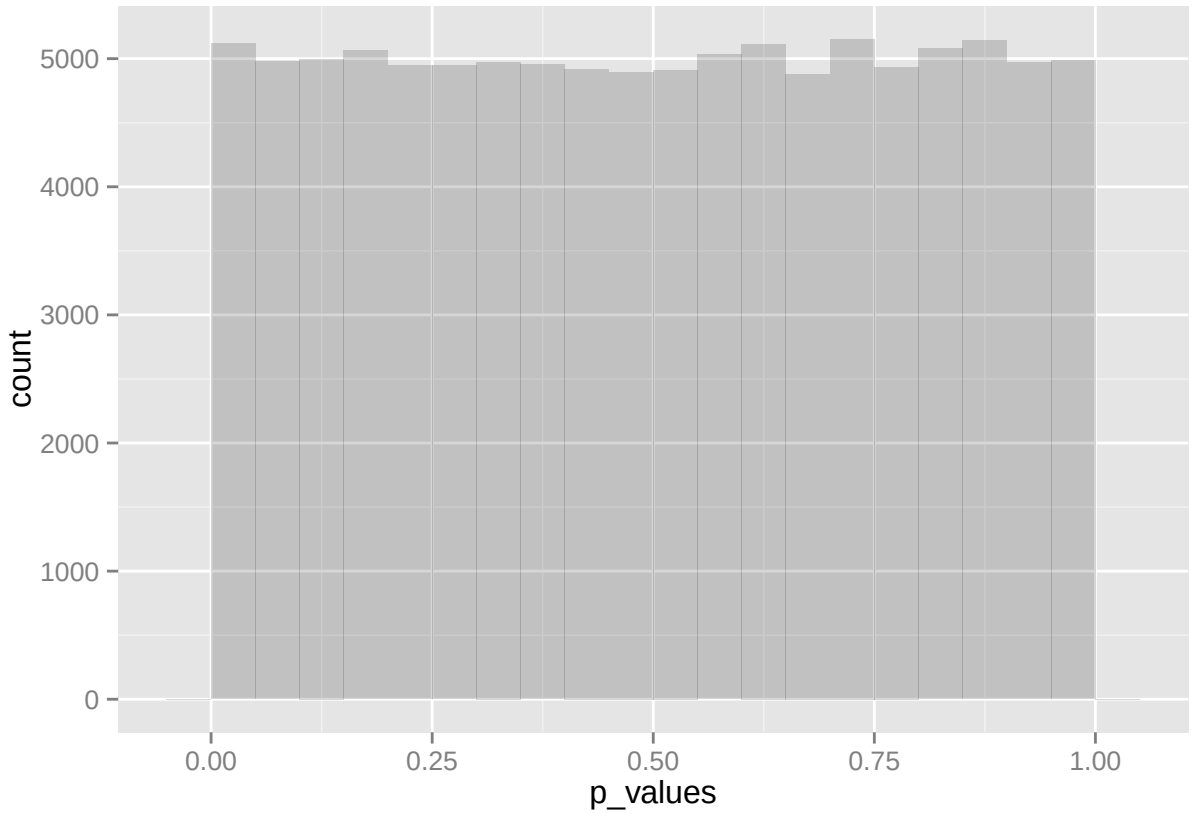
```
## [1] 0.1270278
```

Now that we have our function, we can `replicate()` and see if we get Uniform p-values. Remember, we should, because this test is exact in this case; all assumptions are met. See Module 7, Lecture 1 for more details.

```
library(ggplot2) # load ggplot2 package, for histogram
p_values <- replicate(100000, pval()) # 100,00 replications
mean(p_values < 0.05) # What proportion are less than 0.05?
```

```
## [1] 0.05117
```

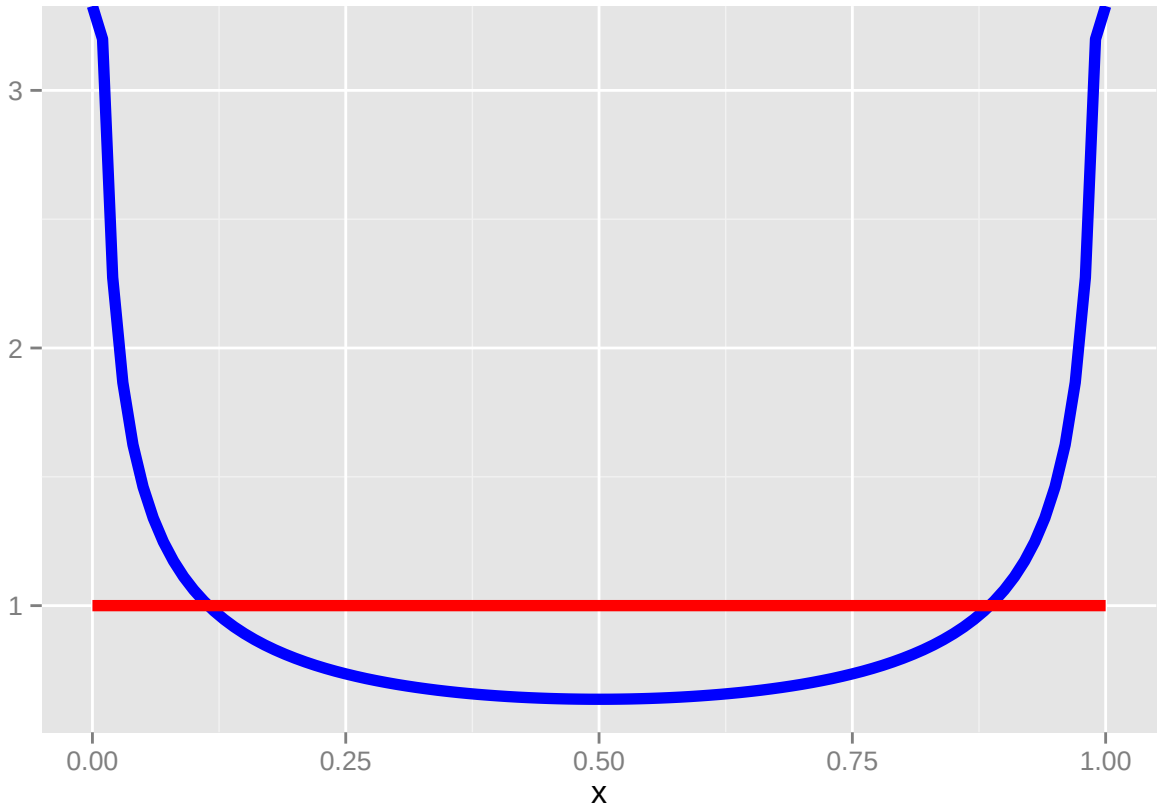
```
qplot(p_values, binwidth = 0.05, alpha = I(0.2)) # Test Validity roadmap
```



Looks pretty Uniform! Now we can use this histogram, and this collection of p-values, to ask questions about rejection rates for any level test. For example, if we run a $\alpha = 0.05$ level test, we can confirm that under the null hypothesis (which we know is true, because we simulated the data) we rejected approximately 5% of the time. This is exactly what `mean(p_values < 0.05)` does, and we see it is rejecting at a rate very close to 0.05.

Violating Assumptions: Non-Normality

Is the two-sample t-test valid when the populations are non-normal, and sample sizes are small and unequal? Let's find out! Consider below the blue Beta(0.5, 0.5) distribution, and the red Uniform(0,1) distribution. These are definitely non-Normal!



How will our testing procedure work? First, let's run one test to find out. We generate two random samples, and use `t.test()` to perform the t-test.

```
# 2-sample t-test, both samples are size = 10.
x <- rbeta(10, shape1 = 0.5, shape2 = 0.5) # Beta(0.5, 0.5) sample
y <- runif(10, 0, 1) # Uniform(0,1) sample
t.test(x, y, mu = 0, paired = FALSE, var.equal = FALSE)
```

```
##
##  Welch Two Sample t-test
##
## data:  x and y
## t = 0.9356, df = 17.726, p-value = 0.3621
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.1763933  0.4590471
## sample estimates:
## mean of x mean of y
## 0.7195065 0.5781796
```

We get all the usual information in our output. Now, let's write another function that automates this process, and extracts the p-value.

```
pval2 <- function(n1 = 25, n2 = 25){ # Arguments are sample sizes.
  x <- rbeta(n1, shape1 = 0.5, shape2 = 0.5) # Draw size = n1 sample from Beta(0.5, 0.5)
  y <- runif(n2, min = 0, max = 1) # Draw size = n2 sample from Uniform(0,1)
  test <- t.test(x, y, mu = 0, paired = FALSE, var.equal = FALSE) # Run test on samples
```

```
  test$p.value # Extract p-value
}
```

The arguments, n1 and n2, allow us to change the Beta and Uniform sample sizes easily. The next two lines draw the samples, and the final two lines conduct the test and extract the p-value. All that is left to evaluate the performace, is to replicate the process for varying sample sizes to see how it performs.

We will start with a sample size of five from both distributions.

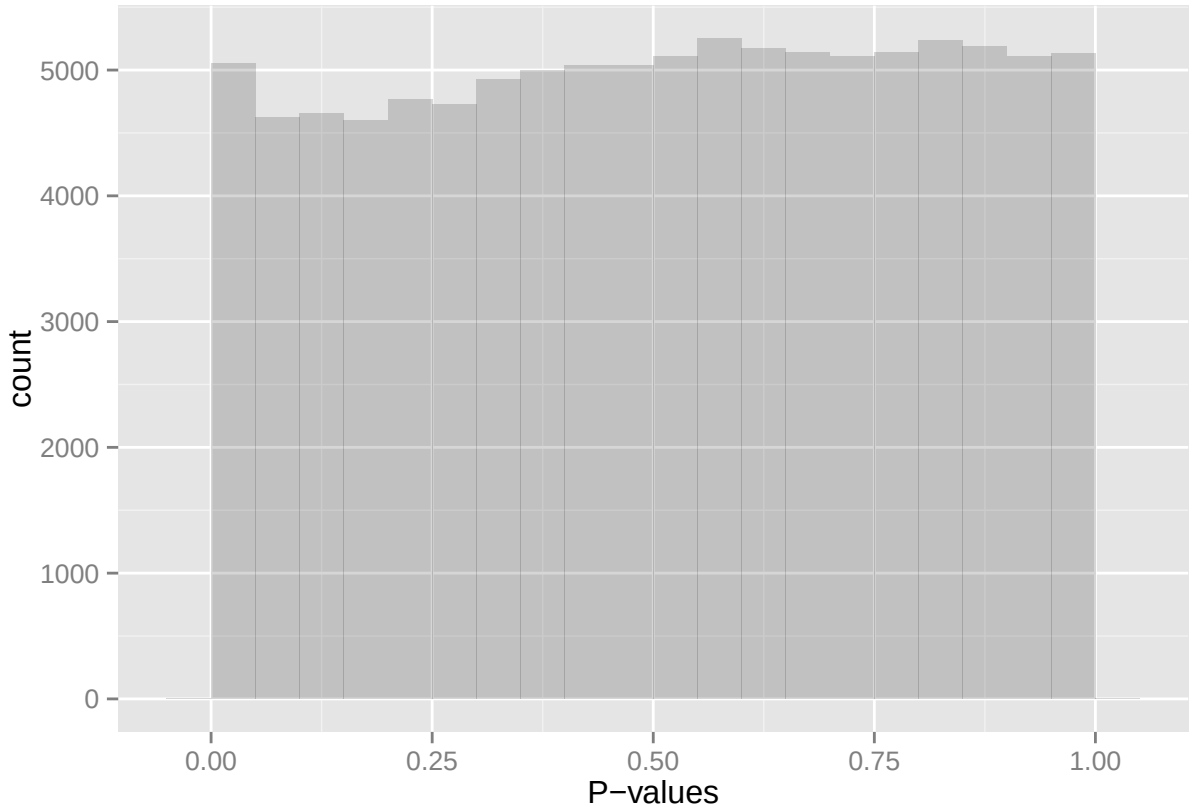
```
set.seed(1770) # Pierre-Simon Laplace calculated p-values in the 1770s!
sim <- replicate(100000, pval2(5, 5)) # Sample size of 5 from each distribution, each replication
mean(sim < 0.05) # Rejection rate for level alpha = 0.05
```

```
## [1] 0.05049
```

```
mean(sim < 0.3) # Do we have 30% of p-vals less than 0.3?
```

```
## [1] 0.28416
```

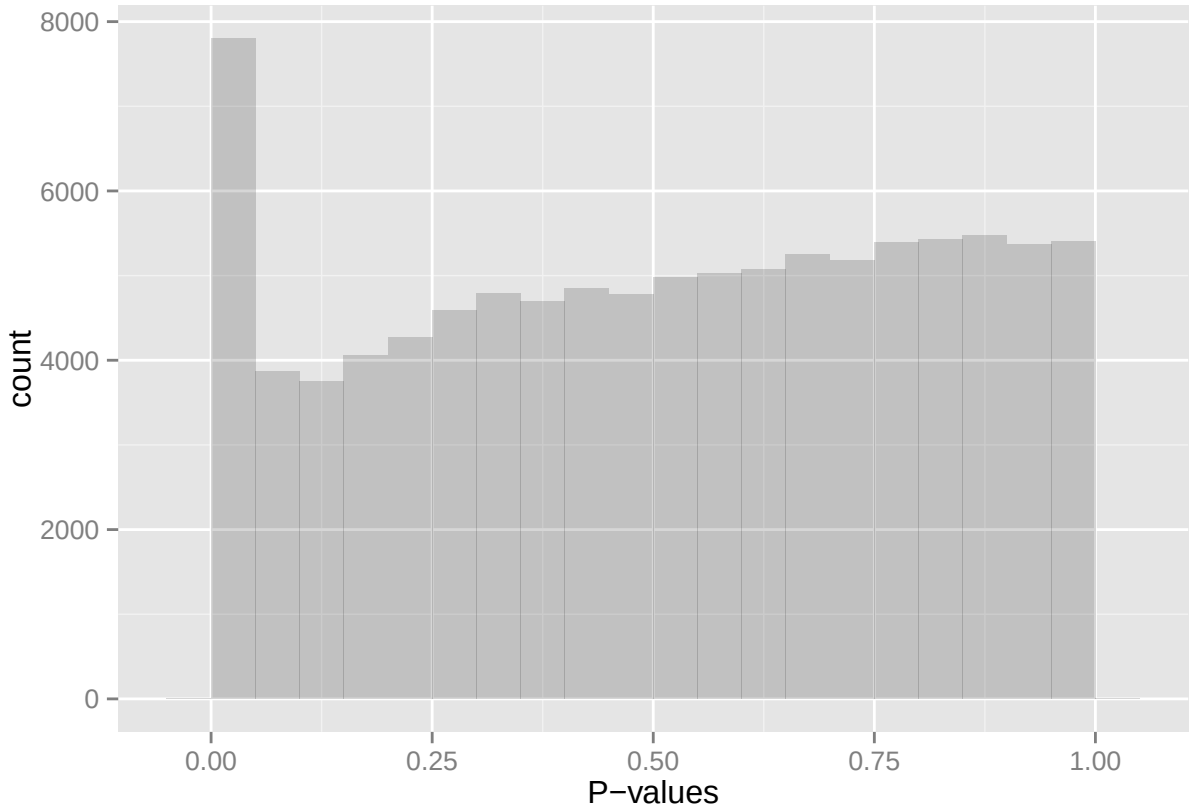
```
qplot(sim, binwidth = 0.05, alpha = I(0.2)) + xlab("P-values")
```



The histogram looks pretty Uniform, though not quite perfectly Uniform. The cumulative area (as a proprotion of total area) of the histogram less than p = 0.30, is 0.28416. This indicates that 28.4% of outcomes have a p-value between 0 and 0.30; it would be closer to 30% if the assumptions were met. However, this is quite close for such small sample sizes, so the robustness of the t-test here is still quite impressive.

What happens if one of the sample sizes increases, but not the other?

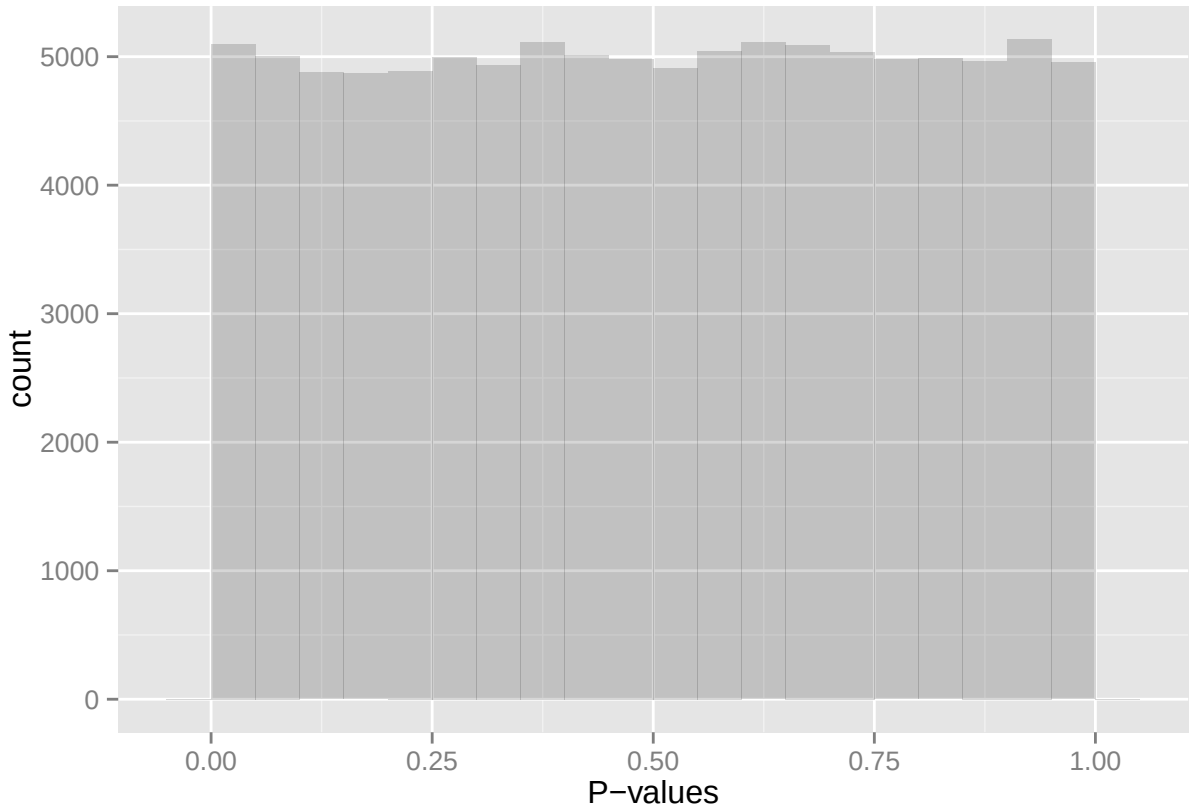
```
sim <- replicate(100000, pval2(5, 50)) # Now 50 draws from U(0,1) in each replication
qplot(sim, binwidth = 0.05, alpha = I(0.2)) + xlab("P-values")
```



The test performed worse! Now the test is rejecting far too often, as indicated by the large number p-values between 0 and 0.05. Is it the same if you reverse the sample sizes, and let n1 = 50, and n2 = 5?

What about if both samples are size 25?

```
sim <- replicate(100000, pval2(25, 25)) # Now 50 draws from U(0,1) in each replication
qplot(sim, binwidth = 0.05, alpha = I(0.2)) + xlab("P-values")
```



It looks great! The p-values are uniformly distributed, so the test is valid at sample sizes n1 = n2 = 25, even though the underlying populations are non-Normal. We could now say with confidence that the t-test is robust to non-Normality, at sample sizes of 25, for the case of data from Uniform(0,1) and Beta(0.5, 0.5) distributions.

With the procedure we just followed, and the sequence of simulations, we explored the robustness of the t-test to one particular violation of assumptions, at varying sample sizes. However, we could use the same procedure to investigate the robustness of the t-test to other departures from t-test assumptions.

Also, note that we drew from populations with differing standard deviations, and the test performed well. This is a nice feature of Welch's two-sample t-test, which `t.test()` uses.