STAT 425 Project

Mehmet Arslan

4/20/2022

Part A

```
set.seed(101)

m = 10000
n1 = c(25, 25, 50, 50)
n2 = c(25, 50, 25, 50)

hat.tt1 = c()
hat.tt2 = c()
hat.tt3 = c()
hat.wt1 = c()
hat.wt2 = c()
hat.wt3 = c()

for(i in 1:4){
  Ind.tt1 = c()
  Ind.tt2 = c()
  Ind.tt3 = c()
  Ind.wt1 = c()
  Ind.wt2 = c()
  Ind.wt3 = c()

  for(j in 1:m){
    x1 = rnorm(n1[i], 500, 100)
    y1 = rnorm(n2[i], 500, 100)

    x2 = rnorm(n1[i], 500, 100)
    y2 = rnorm(n2[i], 500, 125)

    x3 = rnorm(n1[i], 500, 100)
    y3 = rnorm(n2[i], 500, 150)

    ttest1 = t.test(x1, y1)
    ttest2 = t.test(x2, y2)
    ttest3 = t.test(x3, y3)

    welchtest1 = t.test(x1, y1, var.equal = T)
    welchtest2 = t.test(x2, y2, var.equal = T)
    welchtest3 = t.test(x3, y3, var.equal = T)

    Ind.tt1[j] = (ttest1$p.value < 0.05)
    Ind.tt2[j] = (ttest2$p.value < 0.05)
```

```r
    Ind.tt3[j] = (ttest3$p.value < 0.05)

    Ind.wt1[j] = (welchtest1$p.value < 0.05)
    Ind.wt2[j] = (welchtest2$p.value < 0.05)
    Ind.wt3[j] = (welchtest3$p.value < 0.05)


  }

  hat.tt1[i] = mean(Ind.tt1)
  hat.tt2[i] = mean(Ind.tt2)
  hat.tt3[i] = mean(Ind.tt3)

  hat.wt1[i] = mean(Ind.wt1)
  hat.wt2[i] = mean(Ind.wt2)
  hat.wt3[i] = mean(Ind.wt3)

  print(i)
  flush.console()
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
```

```r
(result = data.frame(n1, n2, hat.tt1, hat.wt1, hat.tt2, hat.wt2, hat.tt3, hat.wt3))
```

```
##   n1 n2 hat.tt1 hat.wt1 hat.tt2 hat.wt2 hat.tt3 hat.wt3
## 1 25 25  0.0545  0.0546  0.0486  0.0488  0.0516  0.0520
## 2 25 50  0.0474  0.0472  0.0496  0.0344  0.0505  0.0256
## 3 50 25  0.0472  0.0485  0.0477  0.0666  0.0449  0.0797
## 4 50 50  0.0495  0.0495  0.0489  0.0490  0.0480  0.0480
```

```r
#y has higher sd with higher observations for the second case.
# this is because the variance is not equal at all.

# The one with the largest sd has the smallest number which means that x dominates in this case.

# in the first case, the variances are indeed equal and they are getting really close to the 5% sig lev

# but in the case where we have a larger sample it causes severe underestimation.
```

Note that we got pretty close to our nominal significance level of $\alpha = 0.05$. This is because when we performed the t-test(s), even multiple times, the normality assumption was NOT violated. This is due to the fact that we generated our x's and our y's *from* the normal distribution. It is nice to see, however, that they did in fact get close to our nominal significance level. \

Overall the simulation results in each case were nothing short of remarkable. \

Part B

```
m = 10000
muy = seq(550, 600, 5)

n1 = c(25, 25, 50, 50)
n2 = c(25, 50, 25, 50)

p.tt1 = c()
p.tt2 = c()
p.tt3 = c()
p.wt1 = c()
p.wt2 = c()
p.wt3 = c()

for(k in 1:length(muy)){

  hat.tt1 = c()
  hat.tt2 = c()
  hat.tt3 = c()
  hat.wt1 = c()
  hat.wt2 = c()
  hat.wt3 = c()

  for(i in 1:4){
    Ind.tt1 = c()
    Ind.tt2 = c()
    Ind.tt3 = c()
    Ind.wt1 = c()
    Ind.wt2 = c()
    Ind.wt3 = c()

    for(j in 1:m){
      x1 = rnorm(n1[i], 500, 100)
      y1 = rnorm(n2[i], muy[k], 100)

      x2 = rnorm(n1[i], 500, 100)
      y2 = rnorm(n2[i], muy[k], 125)

      x3 = rnorm(n1[i], 500, 100)
      y3 = rnorm(n2[i], muy[k], 150)

      ttest1 = t.test(x1, y1)
      ttest2 = t.test(x2, y2)
      ttest3 = t.test(x3, y3)

      welchtest1 = t.test(x1, y1, var.equal = T)
      welchtest2 = t.test(x2, y2, var.equal = T)
      welchtest3 = t.test(x3, y3, var.equal = T)

      Ind.tt1[j] = (ttest1$p.value < 0.05)
      Ind.tt2[j] = (ttest2$p.value < 0.05)
      Ind.tt3[j] = (ttest3$p.value < 0.05)
```

```
      Ind.wt1[j] = (welchtest1$p.value < 0.05)
      Ind.wt2[j] = (welchtest2$p.value < 0.05)
      Ind.wt3[j] = (welchtest3$p.value < 0.05)


    }

    hat.tt1[i] = mean(Ind.tt1)
    hat.tt2[i] = mean(Ind.tt2)
    hat.tt3[i] = mean(Ind.tt3)

    hat.wt1[i] = mean(Ind.wt1)
    hat.wt2[i] = mean(Ind.wt2)
    hat.wt3[i] = mean(Ind.wt3)


}

 p.tt1 = rbind(p.tt1, hat.tt1)
 p.tt2 = rbind(p.tt2, hat.tt2)
 p.tt3 = rbind(p.tt3, hat.tt3)
 p.wt1 = rbind(p.wt1, hat.wt1)
 p.wt2 = rbind(p.wt2, hat.wt2)
 p.wt3 = rbind(p.wt3, hat.wt3)

  print(k)
  flush.console()

}
```

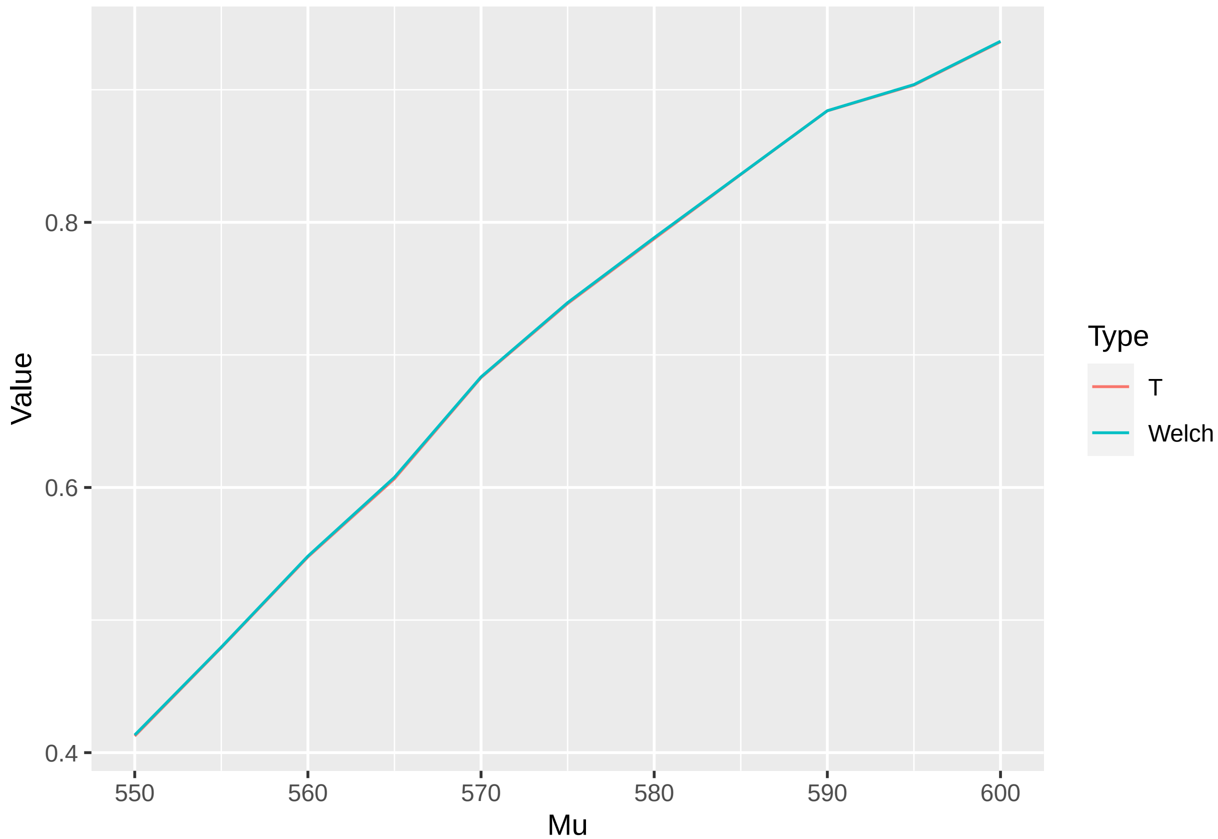
```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
## [1] 9
## [1] 10
## [1] 11
```

In the above chunk, I performed simlar tests. However, I decided to bind the vectors into a dataframe such that I could plot their power curves.

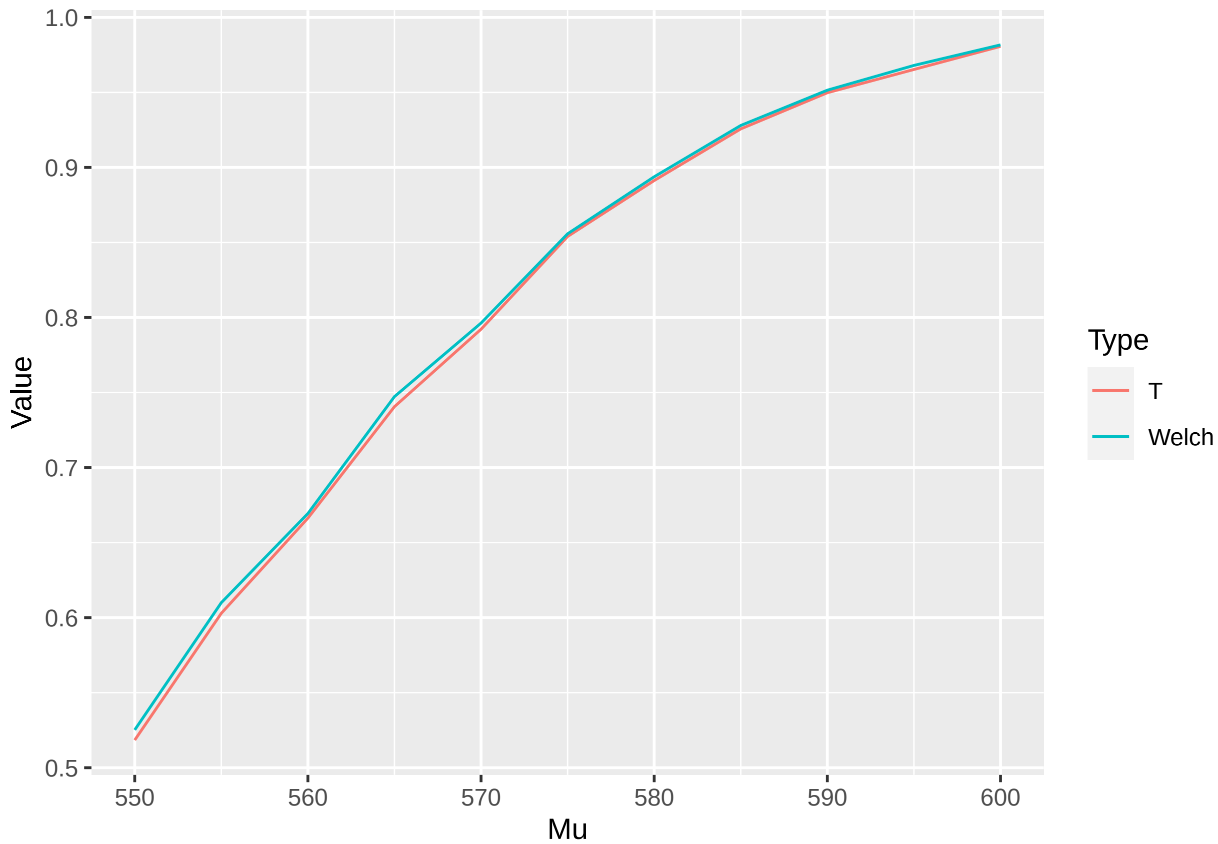
Interestingly, the power of the the samples seemed to increase and get arbitrarily close to 100% This was absolutely exciting to see!

```
library(tidyverse)

# plotting first n1, n2 combo (25, 25) and first variance group.

plot1 = data.frame(cbind("Mu" = muy, "T" = p.tt1[, 1], "Welch" = p.wt1[, 1]))

plot1 %>% gather(key = "Type", value = "Value", -Mu) %>%
```

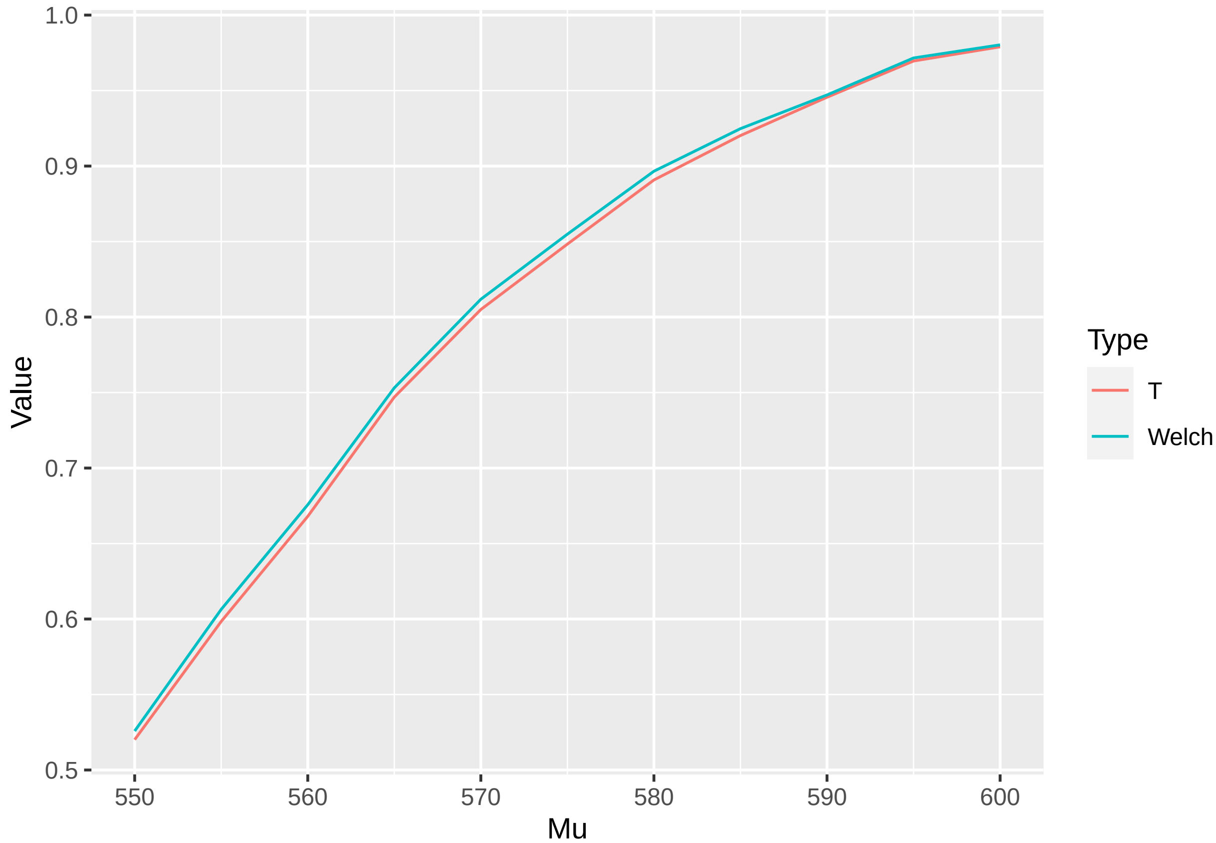
```r
  ggplot(aes(x = Mu, y = Value, col = Type)) +
  geom_line()
```



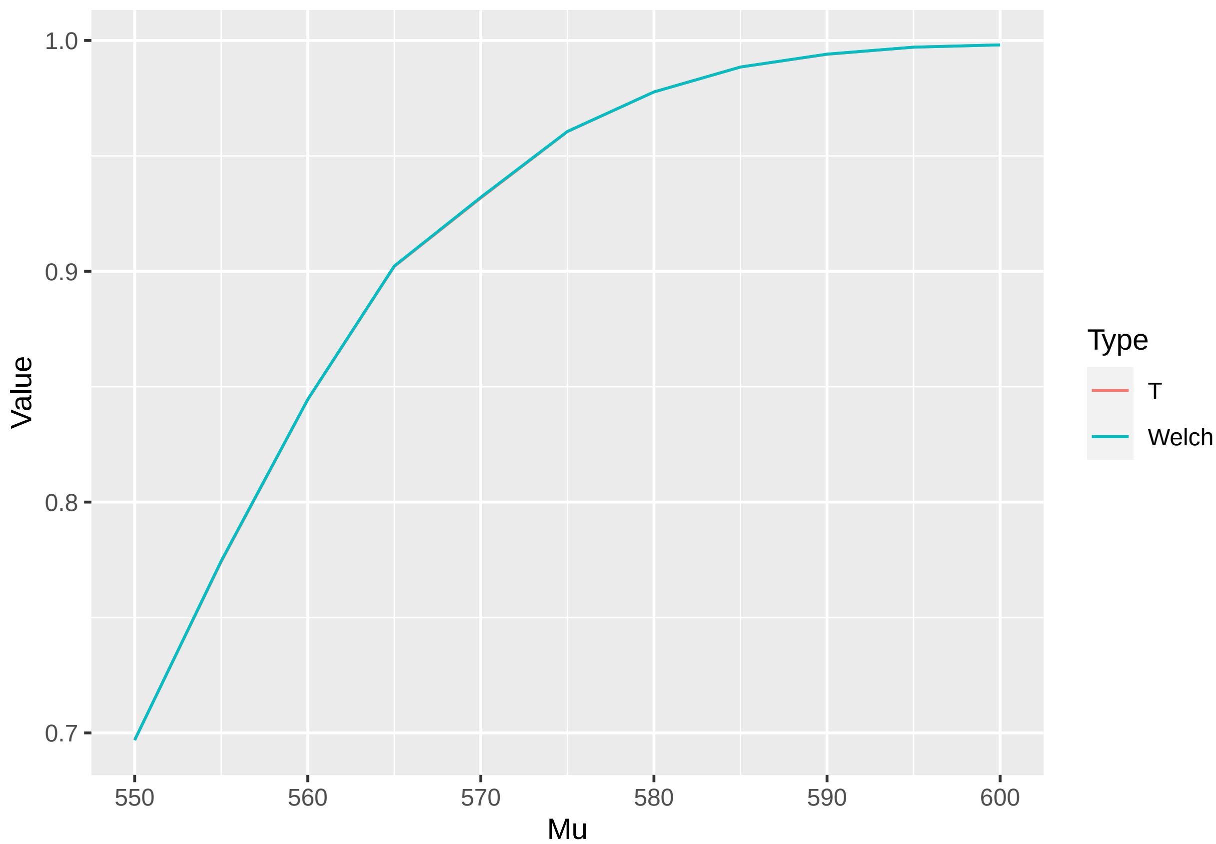

```r
# plotting second n1, n2 combo (25, 50) and first variance group.

plot2 = data.frame(cbind("Mu" = muy, "T" = p.tt1[, 2], "Welch" = p.wt1[, 2]))

plot2 %>% gather(key = "Type", value = "Value", -Mu) %>%
  ggplot(aes(x = Mu, y = Value, col = Type)) +
  geom_line()
```




```r
# plotting third n1, n2 combo (50, 25) and first variance group.

plot3 = data.frame(cbind("Mu" = muy, "T" = p.tt1[, 3], "Welch" = p.wt1[, 3]))

plot3 %>% gather(key = "Type", value = "Value", -Mu) %>%
  ggplot(aes(x = Mu, y = Value, col = Type)) +
  geom_line()
```

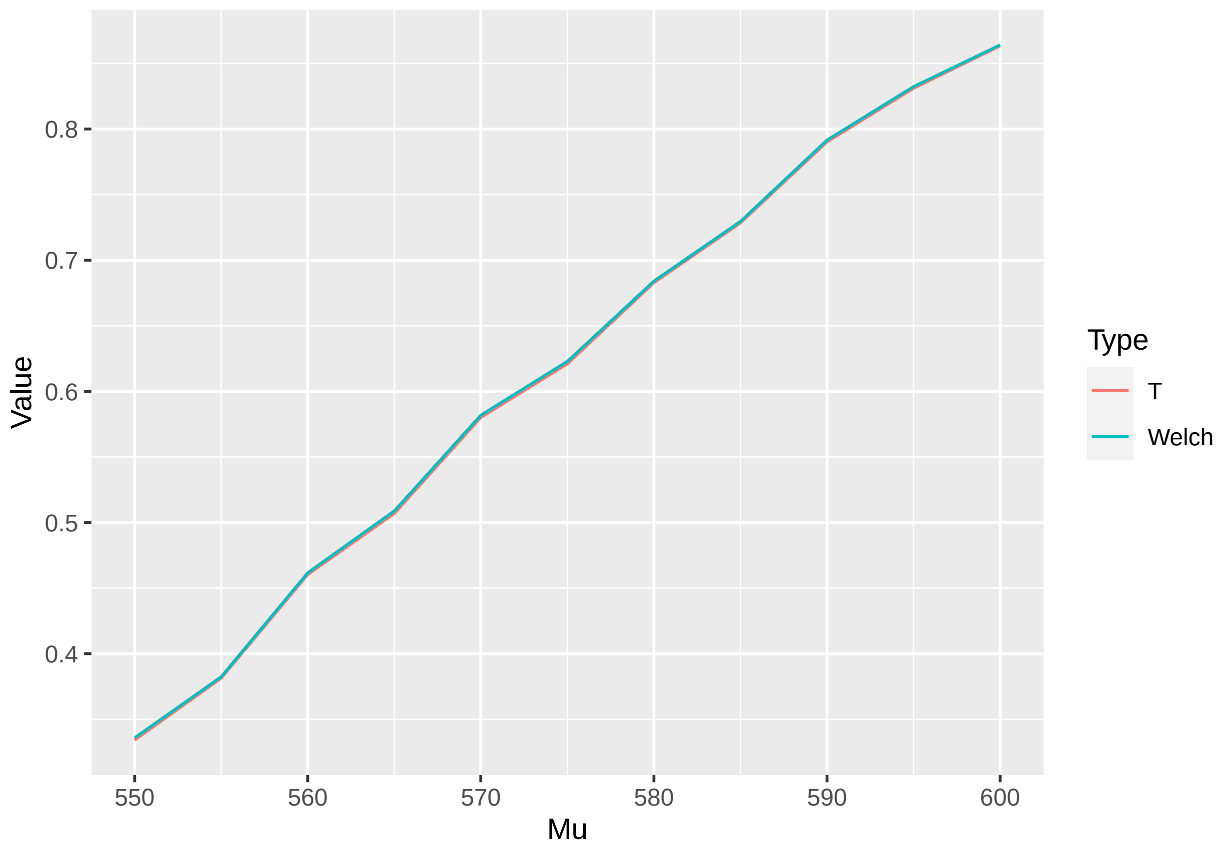



```r
# plotting fourth n1, n2 combo (50, 50) and first variance group.

plot4 = data.frame(cbind("Mu" = muy, "T" = p.tt1[, 4], "Welch" = p.wt1[, 4]))

plot4 %>% gather(key = "Type", value = "Value", -Mu) %>%
  ggplot(aes(x = Mu, y = Value, col = Type)) +
  geom_line()
```

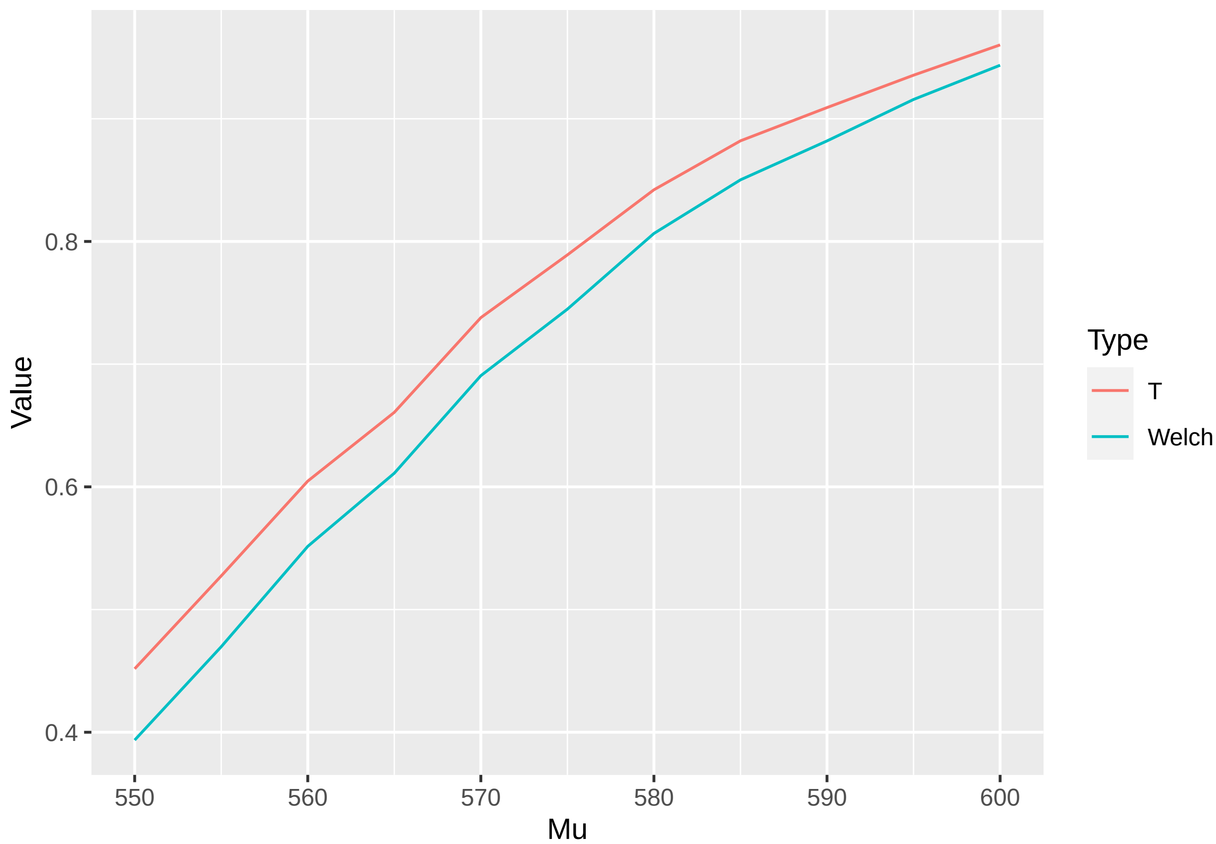


Note that the power of the samples were nearly identical in the first variance group! Also, note that the the $t - test$ seemed to have larger power overall for all each of the samples!

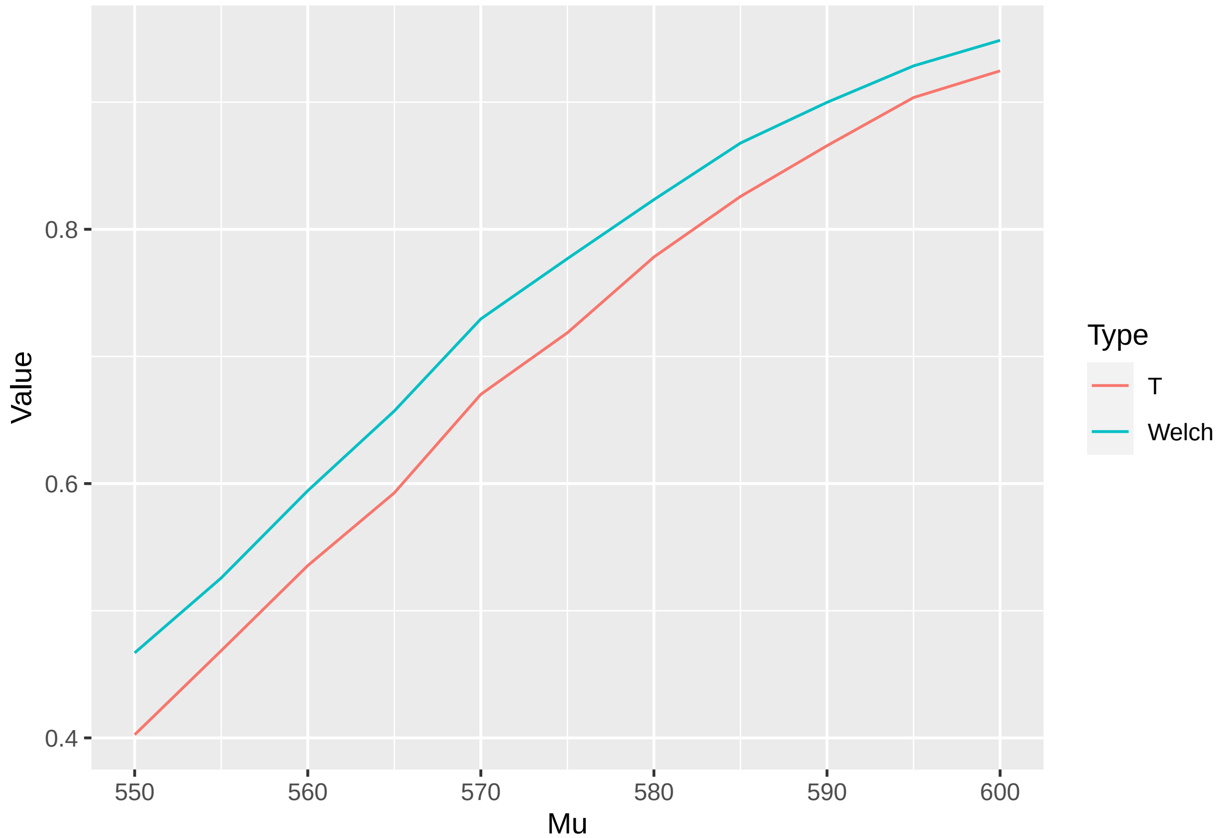
```
# plotting first n1, n2 combo (25, 25) and second variance group.

plot5 = data.frame(cbind("Mu" = muy, "T" = p.tt2[, 1], "Welch" = p.wt2[, 1]))

plot5 %>% gather(key = "Type", value = "Value", -Mu) %>%
  ggplot(aes(x = Mu, y = Value, col = Type)) +
  geom_line()
```



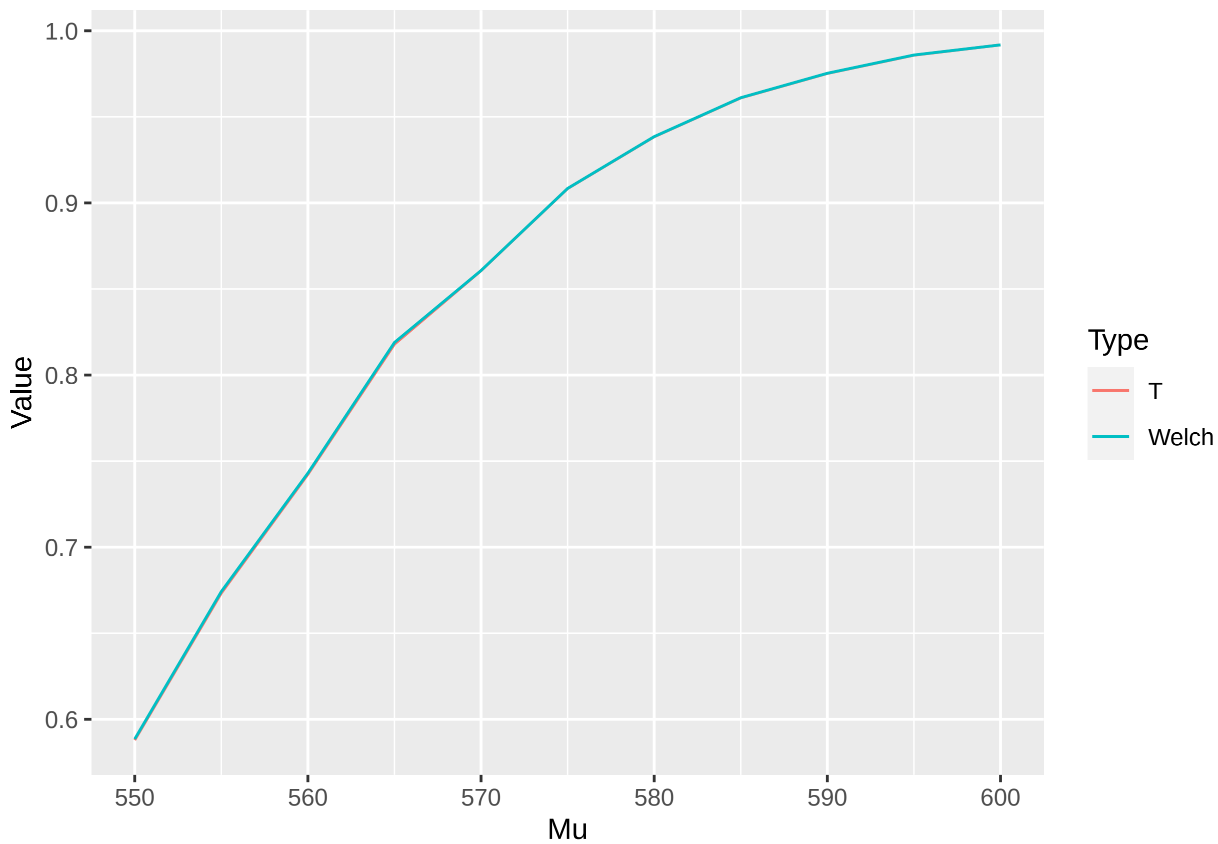

```r
# plotting second n1, n2 combo (25, 50) and second variance group.

plot6 = data.frame(cbind("Mu" = muy, "T" = p.tt2[, 2], "Welch" = p.wt2[, 2]))

plot6 %>% gather(key = "Type", value = "Value", -Mu) %>%
  ggplot(aes(x = Mu, y = Value, col = Type)) +
  geom_line()
```




```
# plotting third n1, n2 combo (50, 25) and second variance group.

plot7 = data.frame(cbind("Mu" = muy, "T" = p.tt2[, 3], "Welch" = p.wt2[, 3]))

plot7 %>% gather(key = "Type", value = "Value", -Mu) %>%
  ggplot(aes(x = Mu, y = Value, col = Type)) +
  geom_line()
```

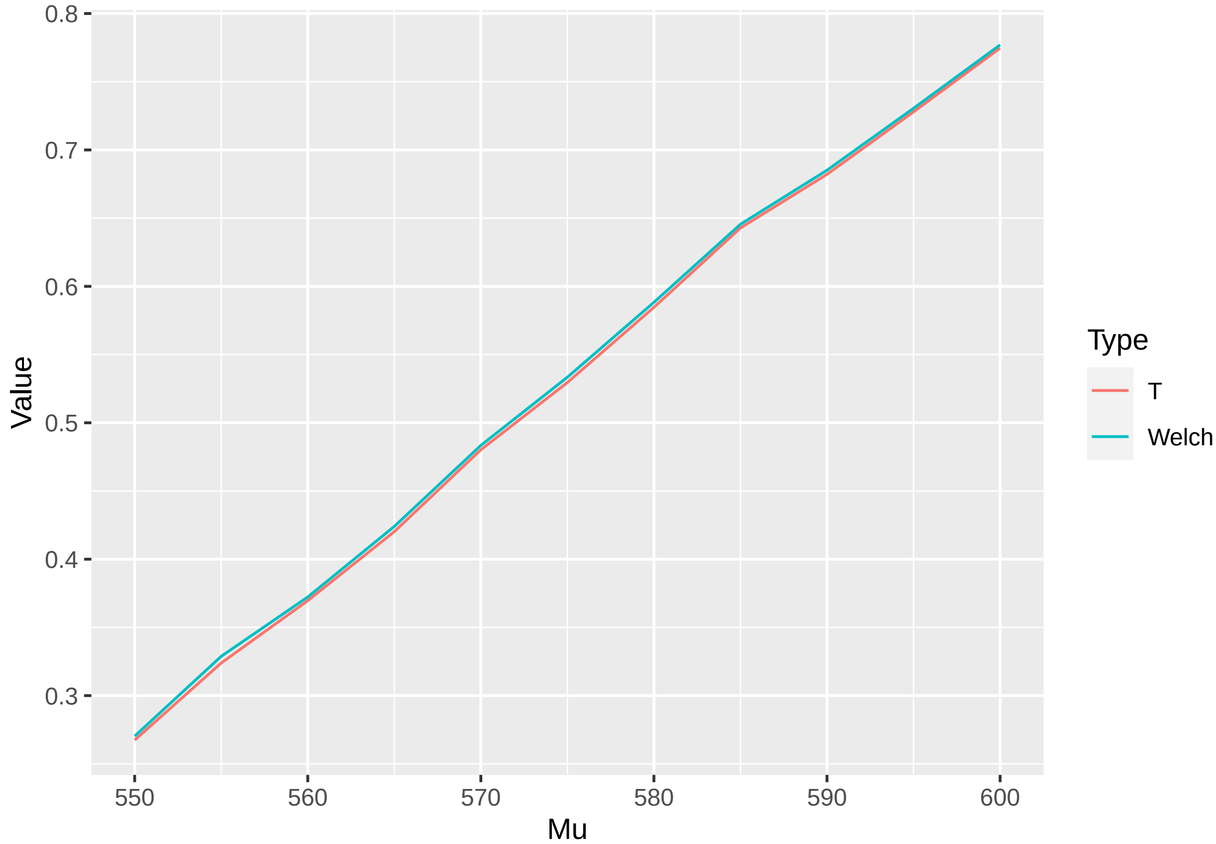



```r
# plotting fourth n1, n2 combo (50, 50) and second variance group.

plot8 = data.frame(cbind("Mu" = muy, "T" = p.tt2[, 4], "Welch" = p.wt2[, 4]))

plot8 %>% gather(key = "Type", value = "Value", -Mu) %>%
  ggplot(aes(x = Mu, y = Value, col = Type)) +
  geom_line()
```

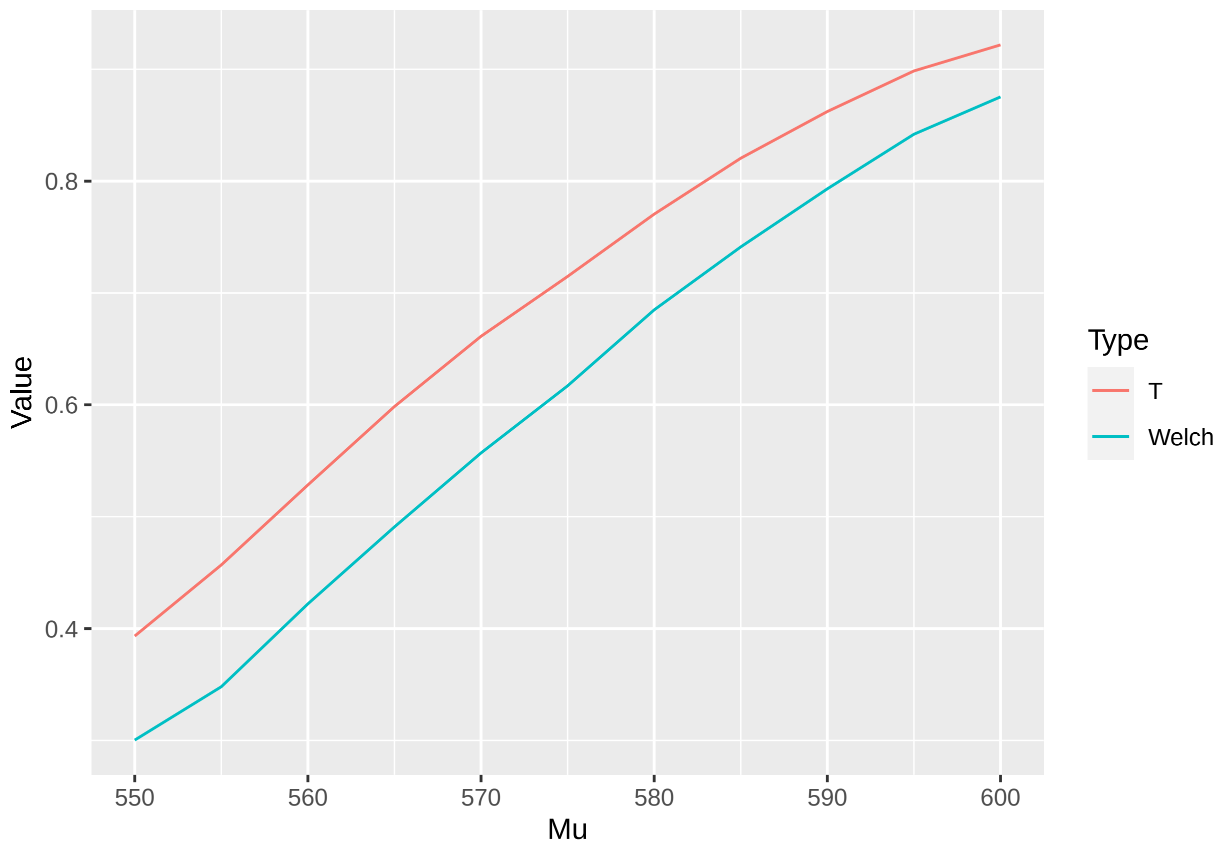


Interestingly, the power curves for the second variance group flipped! I speculate that this was due to the fact that we were sampling a larger spread for x and y respectively. As such, I noticed that the Welch's test does a better job overall when there is slightly more variance.

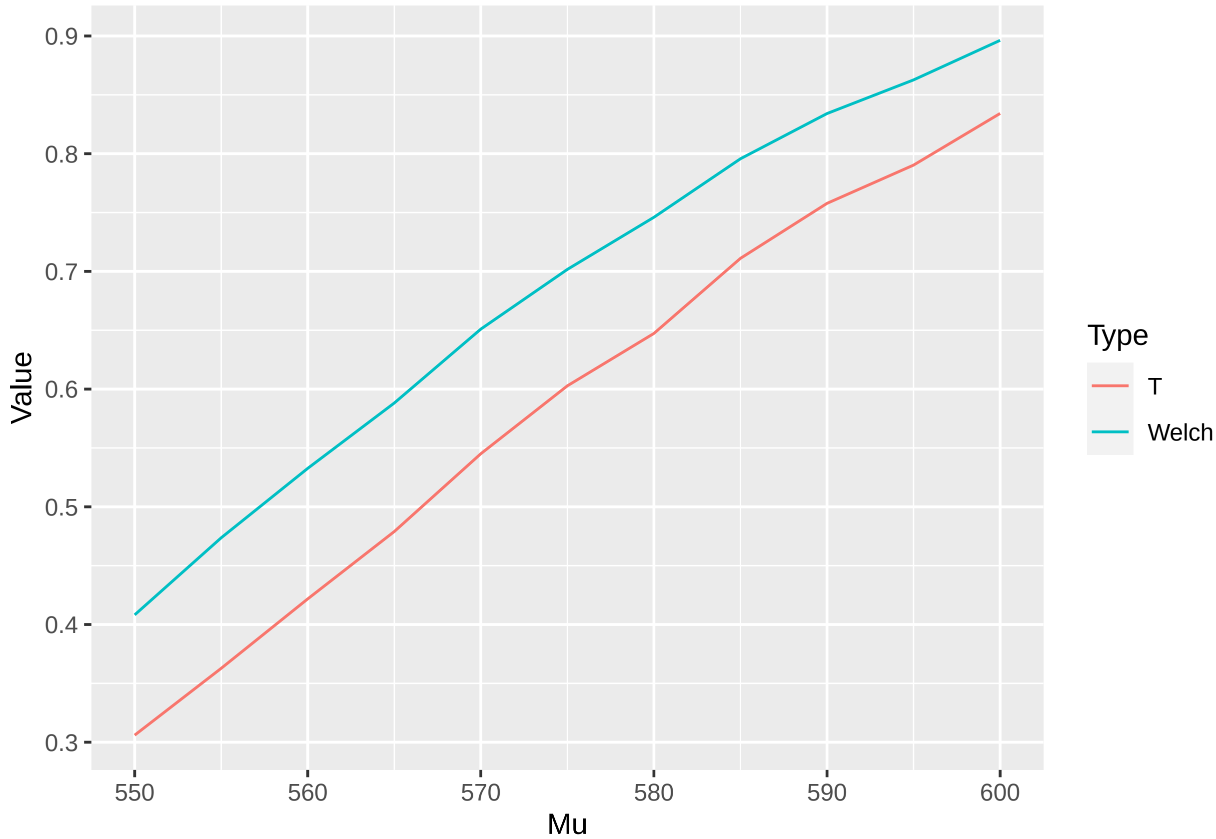
```
# plotting first n1, n2 combo (25, 25) and third variance group.

plot9 = data.frame(cbind("Mu" = muy, "T" = p.tt3[, 1], "Welch" = p.wt3[, 1]))

plot9 %>% gather(key = "Type", value = "Value", -Mu) %>%
  ggplot(aes(x = Mu, y = Value, col = Type)) +
  geom_line()
```



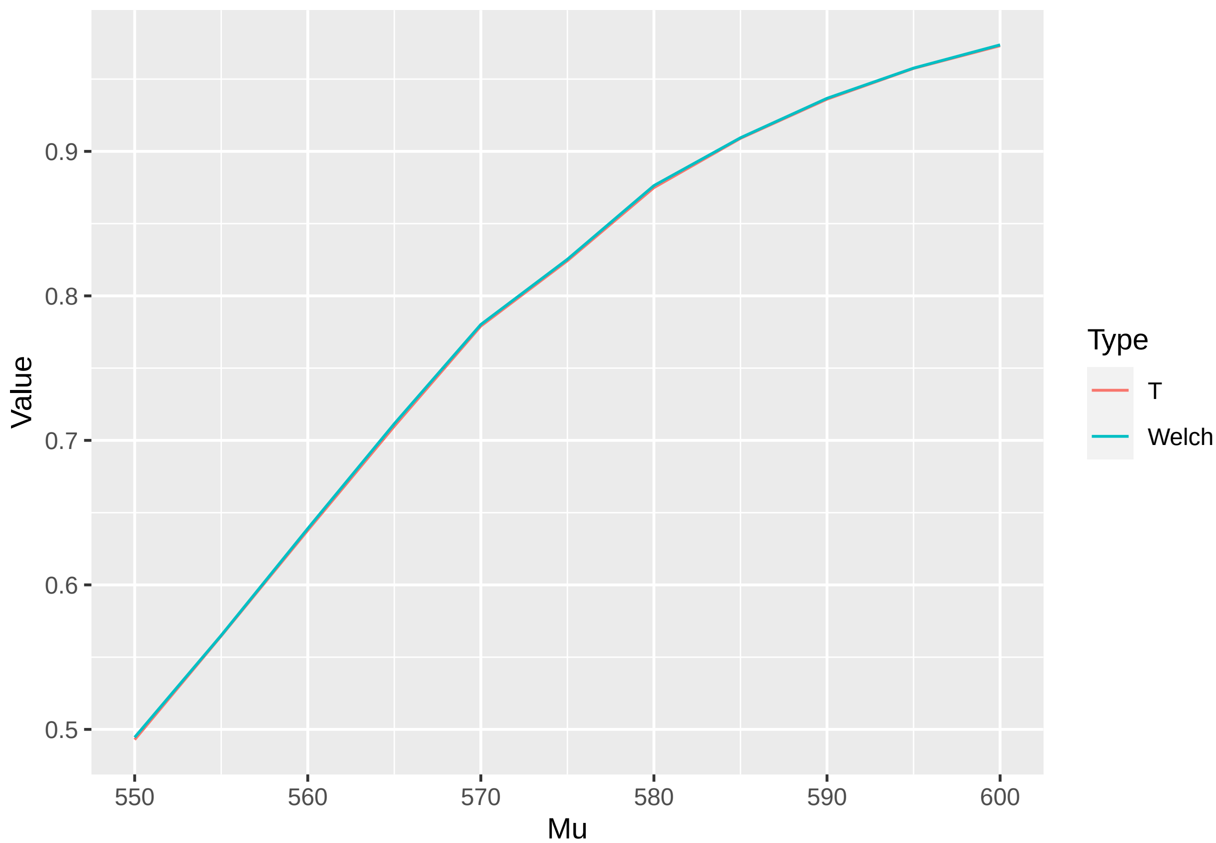

```
# plotting second n1, n2 combo (25, 50) and third variance group.

plot10 = data.frame(cbind("Mu" = muy, "T" = p.tt3[, 2], "Welch" = p.wt3[, 2]))

plot10 %>% gather(key = "Type", value = "Value", -Mu) %>%
  ggplot(aes(x = Mu, y = Value, col = Type)) +
  geom_line()
```




```
# plotting third n1, n2 combo (50, 25) and third variance group.

plot11 = data.frame(cbind("Mu" = muy, "T" = p.tt3[, 3], "Welch" = p.wt3[, 3]))

plot11 %>% gather(key = "Type", value = "Value", -Mu) %>%
  ggplot(aes(x = Mu, y = Value, col = Type)) +
  geom_line()
```




```
# plotting fourth n1, n2 combo (50, 50) and third variance group.

plot12 = data.frame(cbind("Mu" = muy, "T" = p.tt3[, 4], "Welch" = p.wt3[, 4]))

plot12 %>% gather(key = "Type", value = "Value", -Mu) %>%
  ggplot(aes(x = Mu, y = Value, col = Type)) +
  geom_line()
```



In the power curves for the third variance group were way more exaggerated in their differences. This is due to the nature of the $t - test$ itself. Also note that the second and third variance groups were *way* different. This of course implies that the variance groups directly impacted the performance of the model in terms of their power. Moreover, it proved the theory that the Welch's test does a better job when there is more variance, in certain cases but not in general.