Designing, Running, and Analyzing Experiments

Ashish Markanday

11/05/2018

Common distributions

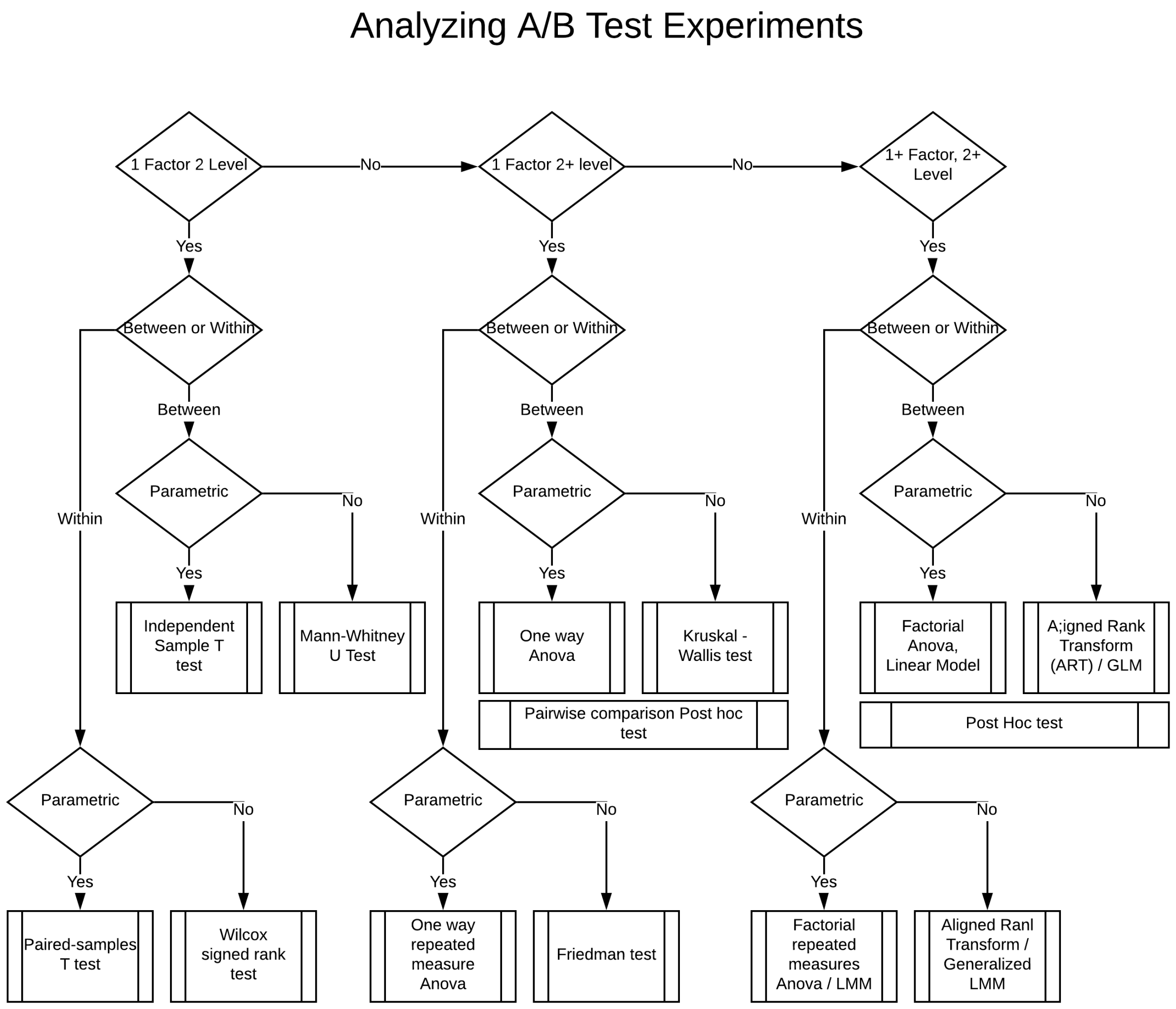
- Normal: Most common for distribution, commonly called a Bell curve.
- Log normal: Time taken to complete a task
- Exponential distributions : Wealth
- Poisson: Count data such as counts of rare events. This is only integer response.
- Binomial distribution: Coin toss
- Power law distribution: Distribution of friends in a social network
- Beta distribution

Parametric Vs Non parametric tests

- Parametric assume a distribution of data such a Normal, exponential, Poisson etc.
- Non parametric tests do not make those assumptions. They typically work by comparing the rank of each item in a variant with each item in the other variants

---

General workflow


Analyzing A/B Test Experiments
1 Factor 2 Level
No
1 Factor 2+ level
No
1+ Factor, 2+ Level
Yes
Between or Within
Between
Within
Parametric
Yes
No
Independent Sample T test
Mann-Whitney U Test
One way Anova
Kruskal - Wallis test
Pairwise comparison Post hoc test
Factorial Anova, Linear Model
A;igned Rank Transform (ART) / GLM
Post Hoc test
Paired-samples T test
Wilcox signed rank test
One way repeated measure Anova
Friedman test
Factorial repeated measures Anova / LMM
Aligned Ranl Transform / Generalized LMM

Factor 2 Level test: Independent Sample T Test

Example: You are launching a new design for Apollo home, and you want to see if that has an impact on user retention. There is only one redesign (one variant) and you will be comparing that with existing home page.

Background:

- Lets call the new redesign as "New" and old design as "Old"
- You have done your due diligence and selected a metric that will track retention. Lets call this metric as App logins in the next 11 days
- You have done your sample size calculations and decided to send 50% of new users to the new design and 50% to the older design

Analysis Descriptive test

- Summary statistics such as mean, median, quantile etc.

- Plot the distribution / histogram of the response variable
- Plot box plots
- Visual examination of the plots
- Do you think the responses look different
    - Run statistical test

```r
## Independent-samples t-test

# read in a data file with page views from an A/B test
logins11Day = read.csv("logins11Day.csv")
head(logins11Day)
```

Read input file

```
##   Subject Site Logins
## 1       1  New      2
## 2       2  New      6
## 3       3  Old      5
## 4       4  New      7
## 5       5  Old      3
## 6       6  New      2
```

```r
logins11Day$Site = factor(logins11Day$Site) # convert to nominal factor
summary(logins11Day)
```

```
##     Subject       Site         Logins
##  Min.   :  1.0   New:255   Min.   : 1.000
##  1st Qu.:125.8   Old:245   1st Qu.: 3.000
##  Median :250.5             Median : 4.000
##  Mean   :250.5             Mean   : 3.958
##  3rd Qu.:375.2             3rd Qu.: 5.000
##  Max.   :500.0             Max.   :11.000
```

```r
# descriptive statistics by Site

ddply(logins11Day, ~ Site, function(data) summary(data$Logins))
```

Run descriptive statistics

```
##   Site Min. 1st Qu. Median     Mean 3rd Qu. Max.
## 1  New    1       3      4 4.490196       6   11
## 2  Old    1       3      3 3.404082       4    6
```

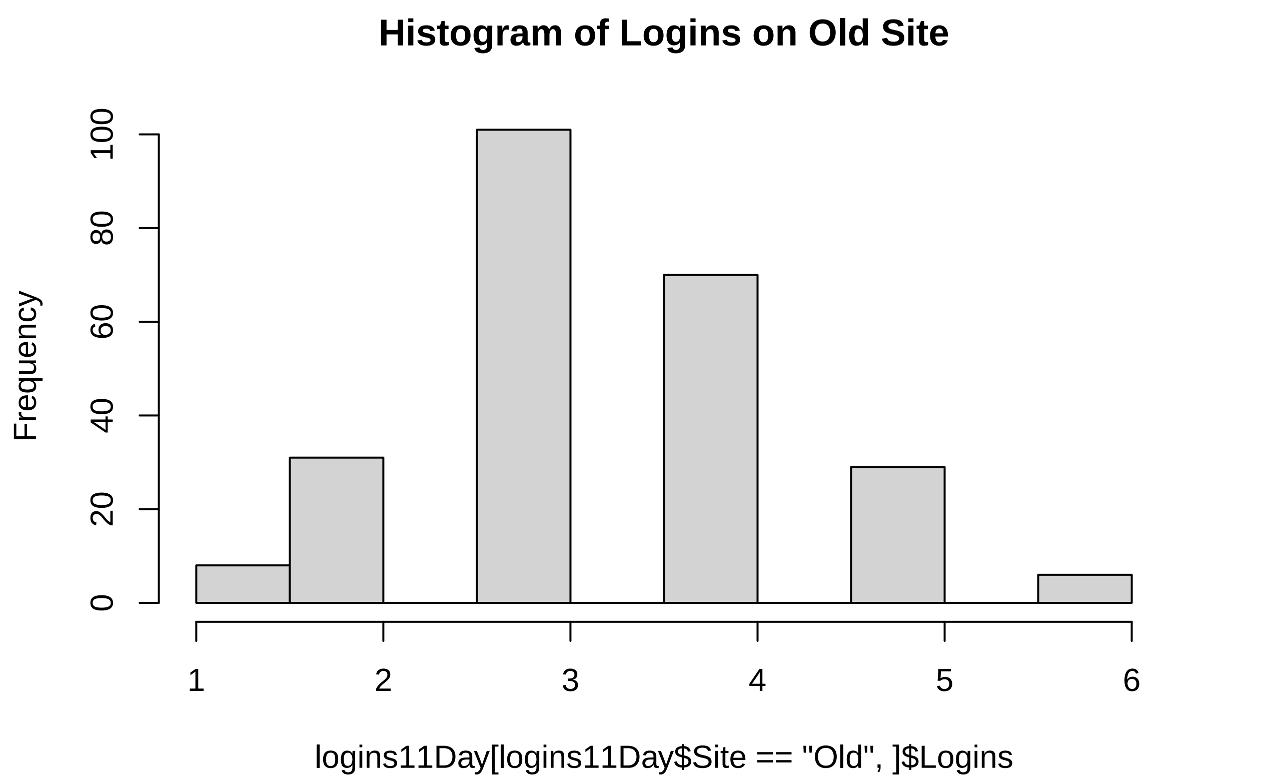
```r
ddply(logins11Day, ~ Site, summarise, Pages.mean=mean(Logins), Pages.sd=sd(Logins))
```

```
##   Site Pages.mean Pages.sd
## 1  New   4.490196 2.127552
## 2  Old   3.404082 1.038197
```

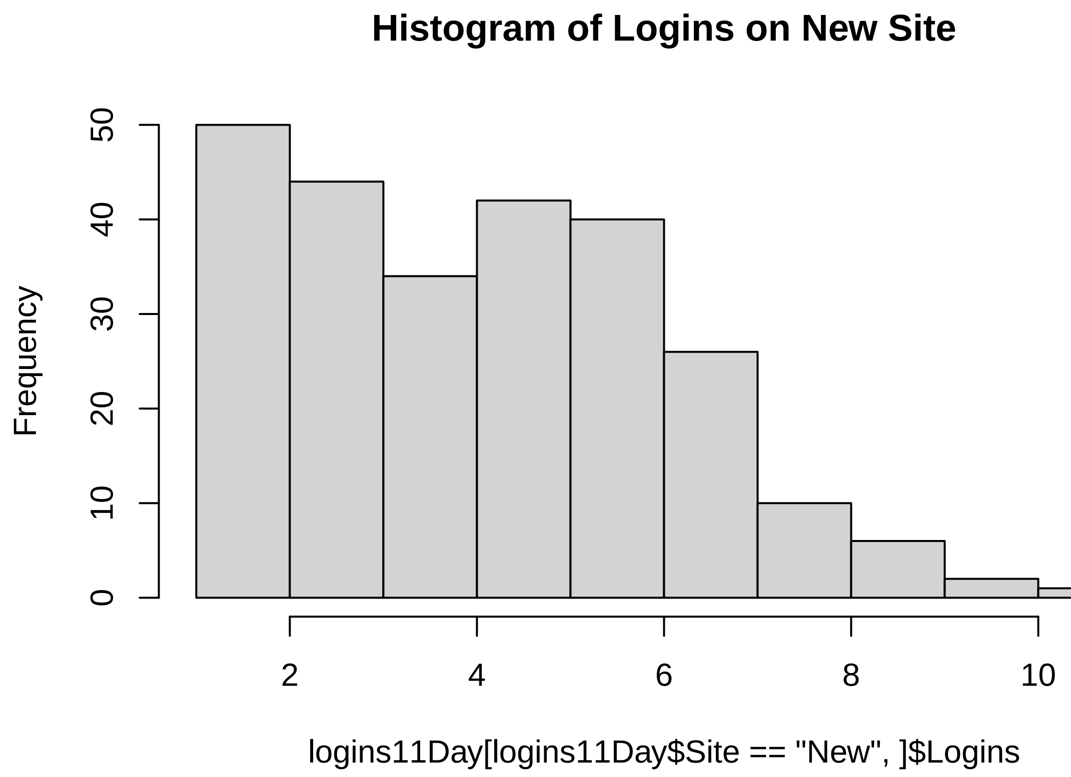
Plot distribution of data

- Notice that the distribution for the second graph is not quite normal
    - We will ignore this for now, and revisit this in the next example

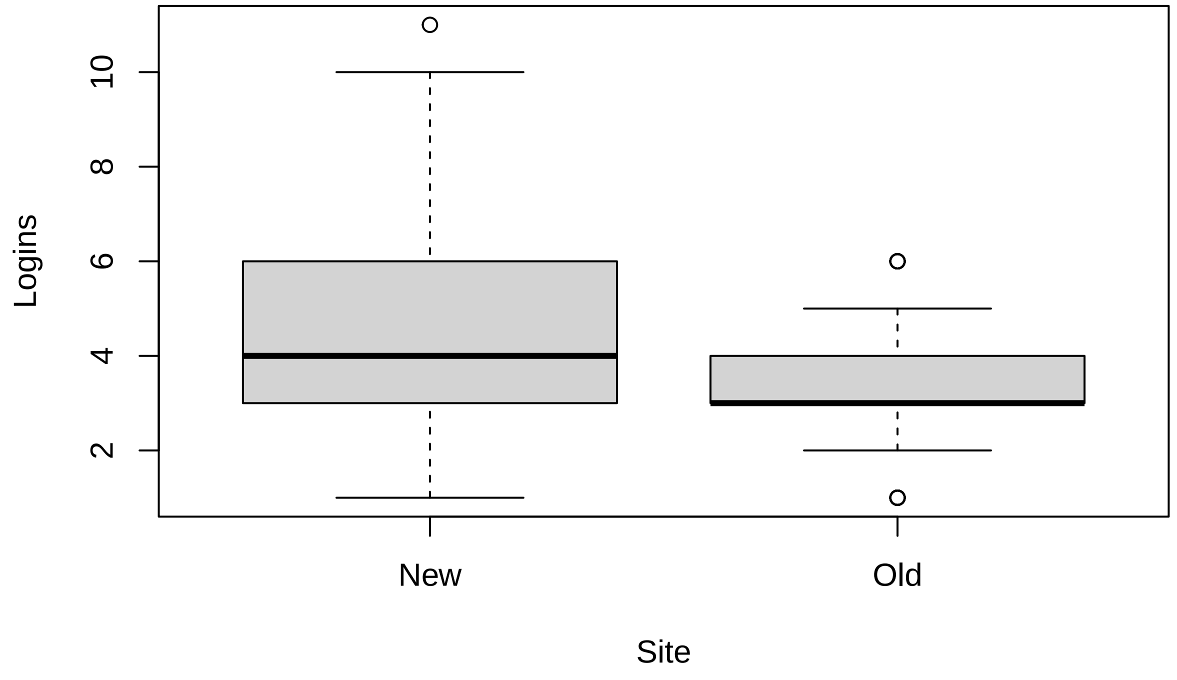
```r
# graph histograms and a boxplot
hist(logins11Day[logins11Day$Site == "Old",]$Logins,
     main = "Histogram of Logins on Old Site")
```



```r
hist(logins11Day[logins11Day$Site == "New",]$Logins,
     main = "Histogram of Logins on New Site")
```



```r
plot(Logins ~ Site, data=logins11Day)
```



Run t-test

- We are running the default t-test assuming that variances are equal
  - I am highlighting this example because this is not a prescribed way of running t-tests
  - We have not checked any Anova assumptions

```r
# independent-samples t-test
t.test(Logins ~ Site, data=logins11Day, var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
## data:  Logins by Site
## t = 7.2083, df = 498, p-value = 2.115e-12
## alternative hypothesis: true difference in means between group New and group Old is not equal to 0
## 95 percent confidence interval:
##  0.7900745 1.3821544
## sample estimates:
## mean in group New mean in group Old
##          4.490196          3.404082
```

Test says that the 2 groups are **different**, but can we trust it ?

T-test done the right way : 1 Factor 2 Level test

Example: You have assigned users tasks (say completion of a survey or a time taken to complete an adventure) and you want to see if there is a difference between time taken each for task variant

Background:

- Let us compare the time taken to complete 2 tasks and call these task 1 and task 2. This is similar to the experiment in the previous example. We will add more tasks later.
- For now just consider 1 factor, 2 variant test
- You have done your sample size calculations and decided to send 50% of new users to task 1 and 50% to task 2

Import data

```r
# read in a data file with task completion times (min) for 2 different tasks
anova1 = read.csv("anova1.csv")
head(anova1)
```

```
##   Subject  TASK Time
## 1       1 Task1  341
## 2       2 Task1  291
## 3       3 Task1  283
## 4       4 Task1  155
## 5       5 Task1  271
## 6       6 Task1  270
```

```r
anova1$Subject = as.factor(anova1$Subject) # convert to nominal factor
anova1$TASK = as.factor(anova1$TASK) # convert to nominal factor
summary(anova1)
```

```
##     Subject      TASK         Time
##  1      : 1   Task1:20   Min.   :155.0
##  2      : 1   Task2:20   1st Qu.:271.8
##  3      : 1              Median :313.5
##  4      : 1              Mean   :385.1
##  5      : 1              3rd Qu.:422.0
##  6      : 1              Max.   :952.0
##  (Other):34
```

Run descriptive tests

Summary stats suggest that time taken to complete the 2 tasks is different

```
# view descriptive statistics by TASK

ddply(anova1, ~ TASK, function(data) summary(data$Time))
```

```
##    TASK Min. 1st Qu. Median   Mean 3rd Qu. Max.
## 1 Task1  155  246.50  287.0 302.10  335.25  632
## 2 Task2  232  294.75  393.5 468.15  585.50  952
```

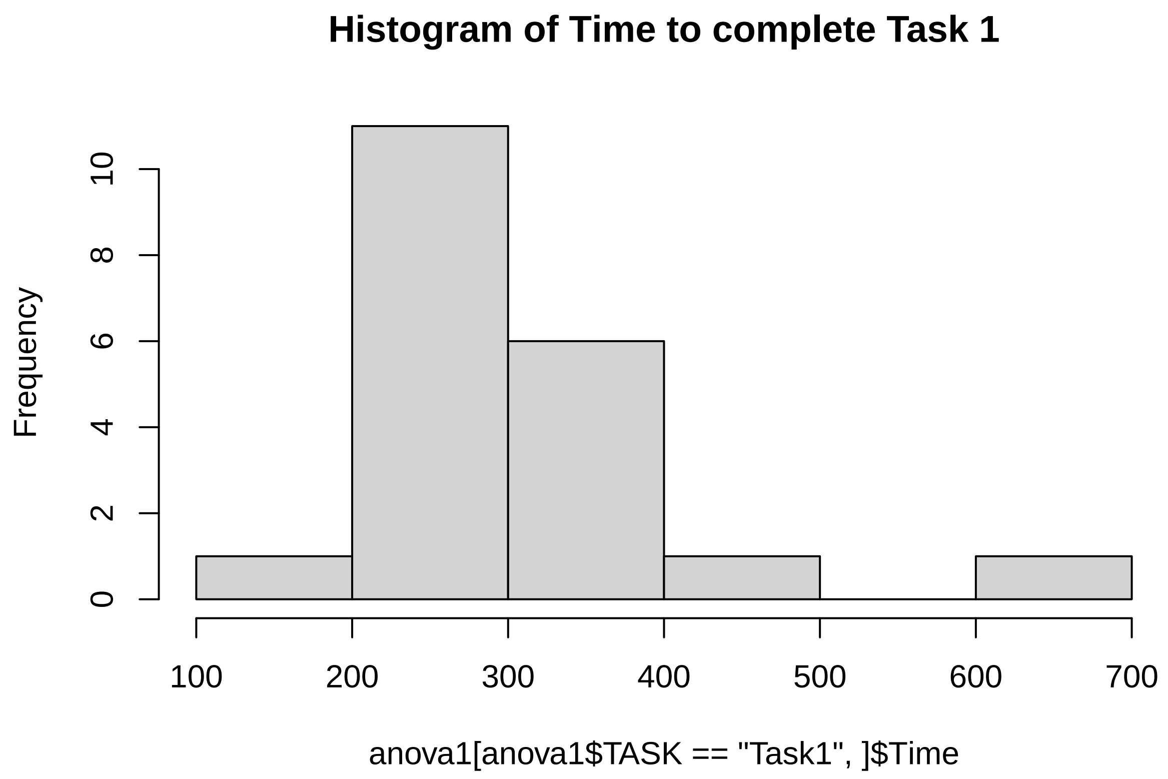
```
ddply(anova1, ~ TASK, summarise, Time.mean=mean(Time), Time.sd=sd(Time))
```

```
##    TASK Time.mean  Time.sd
## 1 Task1    302.10 101.0778
## 2 Task2    468.15 218.1241
```

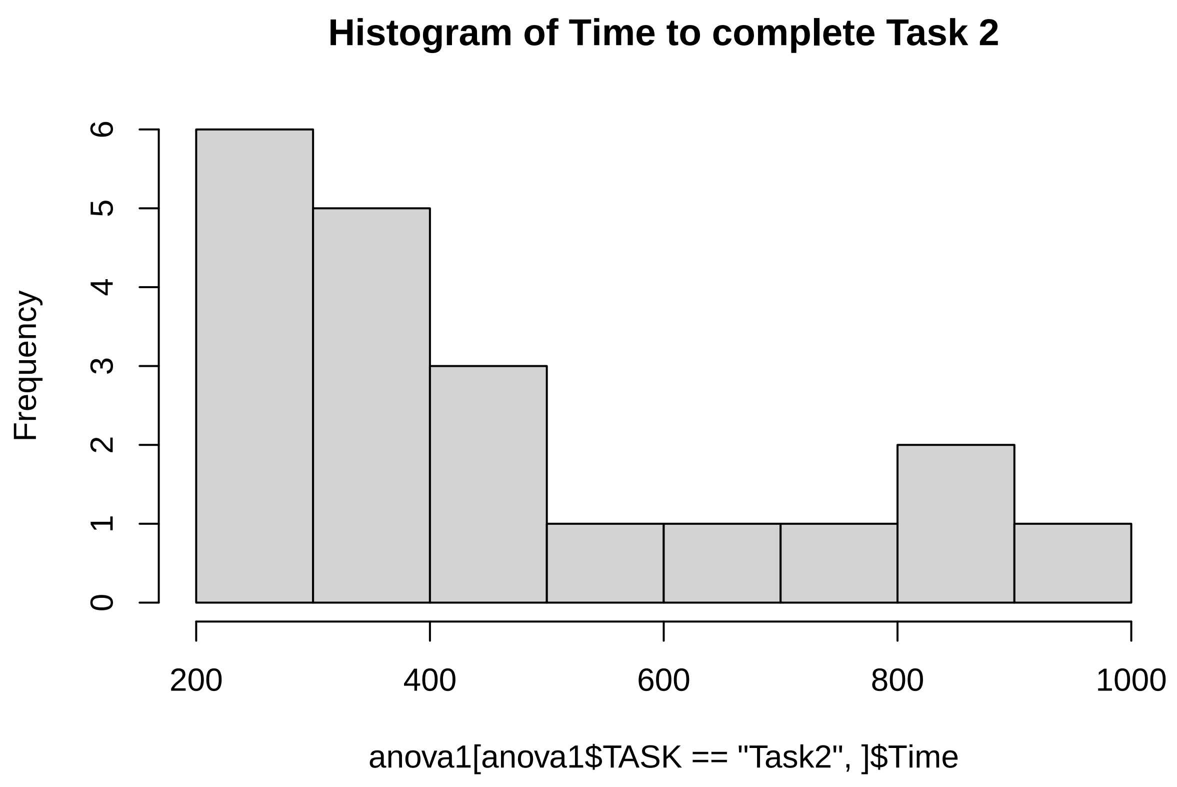
Graph histograms and box plots

Histogram of the data is not really normal

```
# graph histograms and a boxplot
hist(anova1[anova1$TASK == "Task1",]$Time,
     main = "Histogram of Time to complete Task 1") # histogram
```



```
hist(anova1[anova1$TASK == "Task2",]$Time,
     main = "Histogram of Time to complete Task 2") # histogram
```

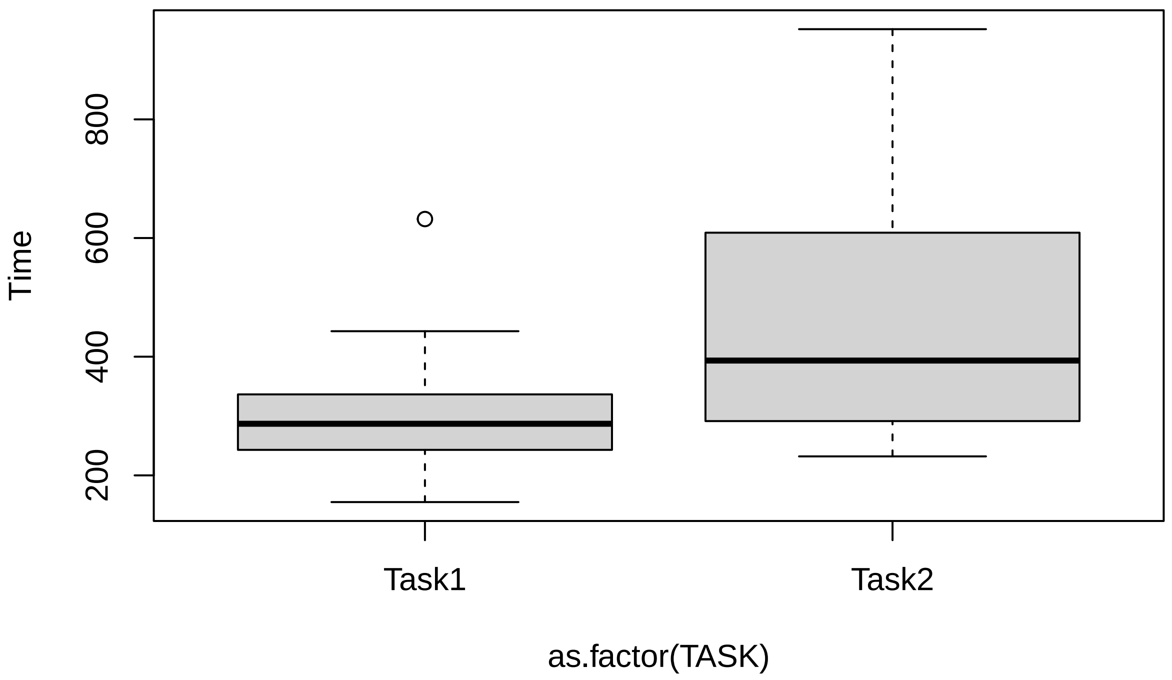


Box plots suggest that there may be a difference between time taken

```r
#head(anova1)
glimpse(anova1)
```

```
## Rows: 40
## Columns: 3
## $ Subject <fct> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,~
## $ TASK    <fct> Task1, Task1, Task1, Task1, Task1, Task1, Task1, Task1, Task1,~
## $ Time    <int> 341, 291, 283, 155, 271, 270, 250, 272, 209, 236, 295, 214, 44~
```

```r
plot(Time ~ as.factor(TASK), data=anova1) # boxplot
```



Running an independent-samples t-test is not suitable because we have not tested Anova assumptions

Testing ANOVA assumptions

- We will use Shapiro Wilk test to test assumptions for normality

```
# Shapiro-Wilk normality test on response
shapiro.test(anova1[anova1$TASK == "Task1",]$Time)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  anova1[anova1$TASK == "Task1", ]$Time
## W = 0.84372, p-value = 0.004191
```

```
shapiro.test(anova1[anova1$TASK == "Task2",]$Time)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  anova1[anova1$TASK == "Task2", ]$Time
## W = 0.87213, p-value = 0.01281
```

Notice that the P values are less than 5% for both test. This means that Time is not normally distributed. We have violated the first assumption for Anova that the data should be normally distributed

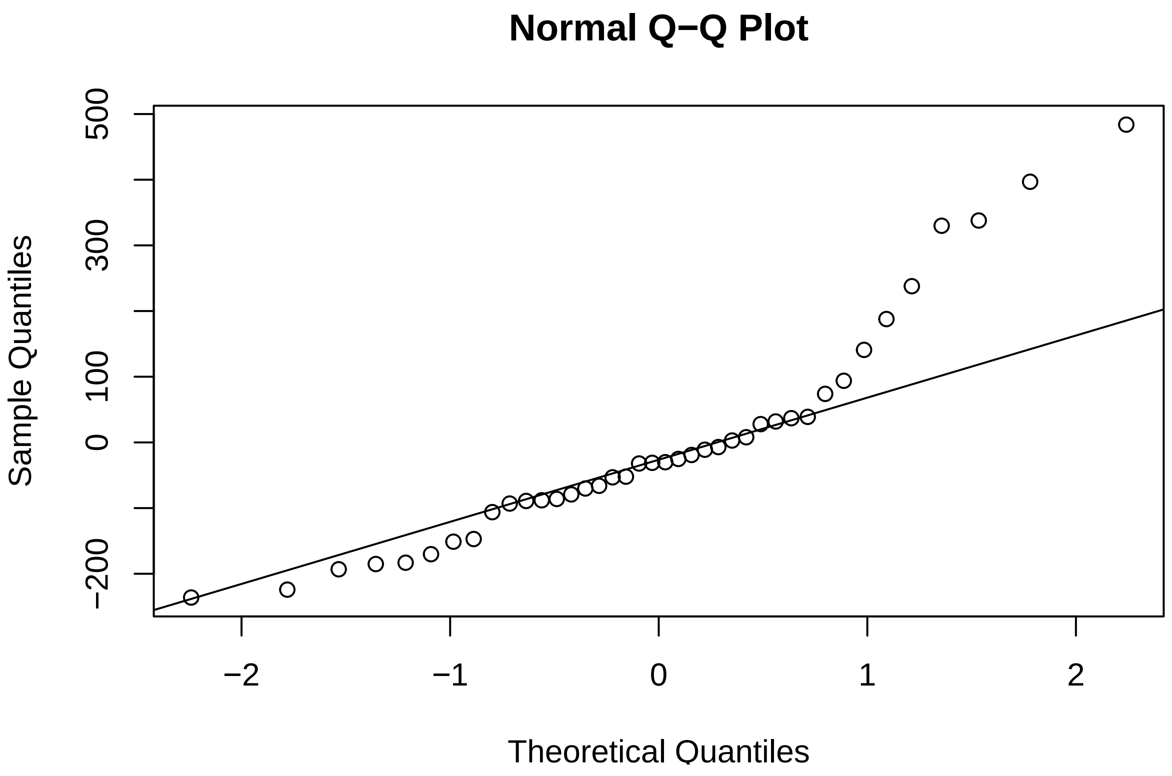
Really what we are looking for is that the residuals are normally distributed

```
m = aov(Time ~ TASK, data=anova1) # fit model
shapiro.test(residuals(m)) # test residuals
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(m)
## W = 0.894, p-value = 0.001285
```

We can also plot the qq plot to see if the residuals are normally distributed

```r
qqnorm(residuals(m));
qqline(residuals(m)) # plot residuals
```



qq plot show that the data is not normally distributed

Since the residuals are not normally distributed, you have 2 options

- Use data transformations to make data normal and use parametric tests
- Use a non parametric test. Non parametric tests do not assume a distribution

Continuing with a parametric test

```
Typically when one is looking at task completion times, lognormal is a good candidate distribution to st
Intutive reasoning behind using a log normal distribution is that the "Log Normal" is the log of the
normal distribution.
When users are asked to complete a task, majority will complete it with a certain time and a
small minority will take much longer to complete the same task. That distribution generally follows a
log normal distribution.
If log normal does not work, other distribution to try would be a poisson distribution, or a beta distri
```

T Test and Anova are generally robust and will work even if the data in not quite normal

- Kolmogorov-Smirnov test for log-normality
- Fit the distribution to a lognormal to estimate fit parameters
- Then supply those to a K-S test with the lognormal distribution fn

```r
fit = fitdistr(anova1[anova1$TASK == "Task1",]$Time, "lognormal")$estimate #Fit a lognormal distributio
ks.test(anova1[anova1$TASK == "Task1",]$Time, "plnorm", meanlog=fit[1], sdlog=fit[2], exact=TRUE) # Tes
```

```
##
##  Exact one-sample Kolmogorov-Smirnov test
##
```

```
## data:  anova1[anova1$TASK == "Task1", ]$Time
## D = 0.13421, p-value = 0.8181
## alternative hypothesis: two-sided
```

```
fit = fitdistr(anova1[anova1$TASK == "Task2",]$Time, "lognormal")$estimate #Fit a lognormal distributio
ks.test(anova1[anova1$TASK == "Task2",]$Time, "plnorm", meanlog=fit[1], sdlog=fit[2], exact=TRUE) # Tes
```

```
##
##  Exact one-sample Kolmogorov-Smirnov test
##
## data:  anova1[anova1$TASK == "Task2", ]$Time
## D = 0.12583, p-value = 0.871
## alternative hypothesis: two-sided
```

P values greater than 5% indicates that the new fit does not violate the log normal assumptions

Testing for the 2nd Anova assumption: Homoscedasticity (homogeneity of variance)

```
Homogeneity of variance means that the variance does not increase as the size of the variable increases
It is mostly seen in time series where the value in the this time period depends on value in the previo
time period. In that case, errors from previous value get added to the current value and compounded.
Other places where you would see that is where the relationship with the dependent variable is logarith
In those cases, box-cox transformation can help you alleviate heteroscedasticity
```

A recommended and a generic approach is to use GLM models and specify the distributiosn while modeling the data

We will touch on GLM models later in the document

```
# tests for homoscedasticity (homogeneity of variance)

leveneTest(Time ~ TASK, data=anova1, center=mean) # Levene's test
```

```
## Levene's Test for Homogeneity of Variance (center = mean)
##       Df F value   Pr(>F)
## group  1  11.959 0.001356 **
##       38
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
leveneTest(Time ~ TASK, data=anova1, center=median) # Brown-Forsythe test
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value  Pr(>F)
## group  1  5.9144 0.01984 *
##       38
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We run tests for both Mean and Median. Since the P value is less than 5%, this suggests that it does not satisfy homogeneity of variance assumption

You can use Welch's T test if the homoscedasticity assumption is violated

```
# Welch t-test for unequal variances handles
# the violation of homoscedasticity. but not
# the violation of normality.
t.test(Time ~ TASK, data=anova1, var.equal=FALSE) # Welch t-test
```

```
##
```

```
##  Welch Two Sample t-test
##
## data:  Time by TASK
## t = -3.0889, df = 26.8, p-value = 0.004639
## alternative hypothesis: true difference in means between group Task1 and group Task2 is not equal to
## 95 percent confidence interval:
##  -276.38735  -55.71265
## sample estimates:
## mean in group Task1 mean in group Task2
##              302.10              468.15
```

Running the T test again, after controlling for Anova assumptions

Steps * Transform variables * Test for normailty * Test for homoscedasticity * Run appropriate T test

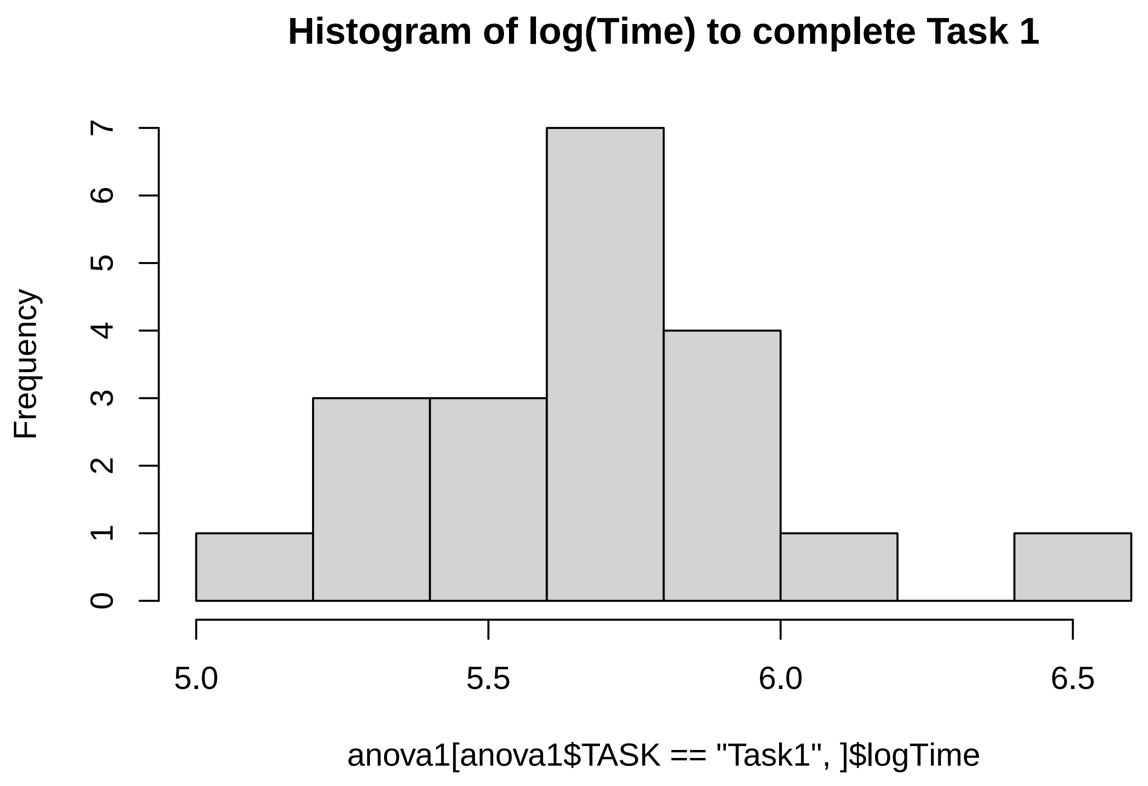
Data transformation

```r
# create a new column in anova1 defined as log(Time)
anova1$logTime = log(anova1$Time) # log transform
head(anova1) # verify
```

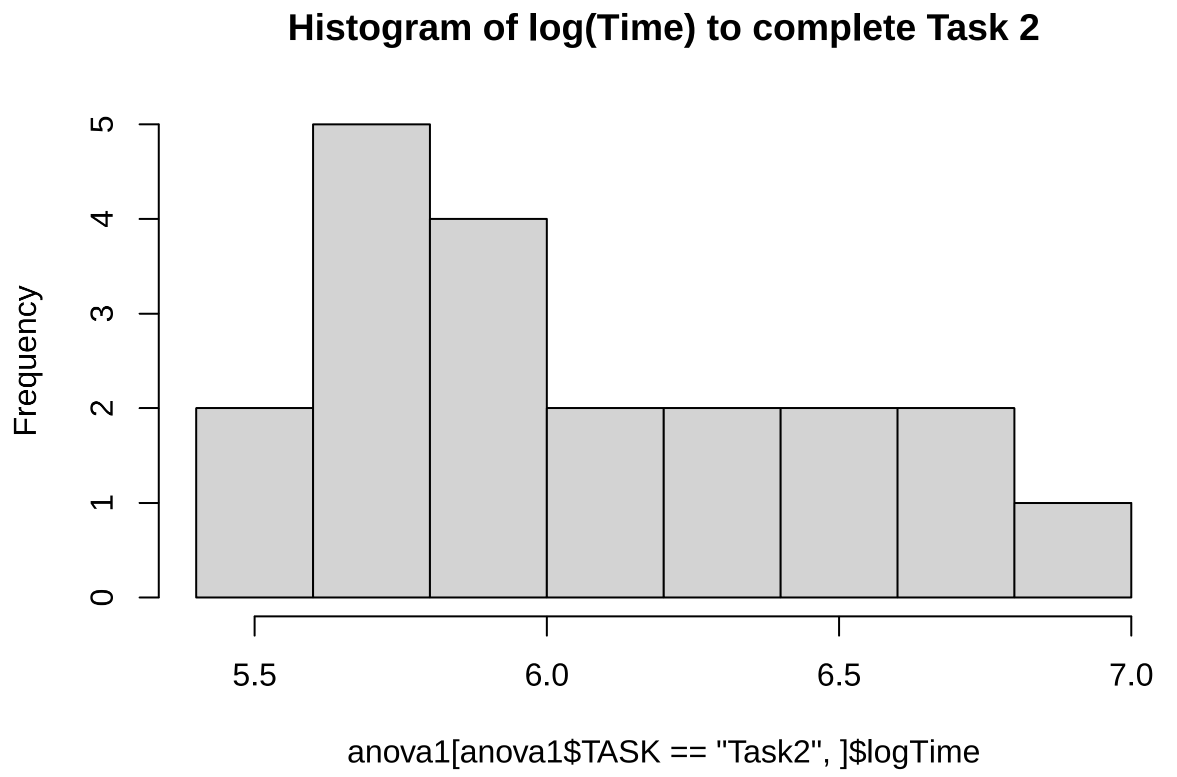
```
##   Subject  TASK Time  logTime
## 1       1 Task1  341 5.831882
## 2       2 Task1  291 5.673323
## 3       3 Task1  283 5.645447
## 4       4 Task1  155 5.043425
## 5       5 Task1  271 5.602119
## 6       6 Task1  270 5.598422
```

Visually examine transformed data

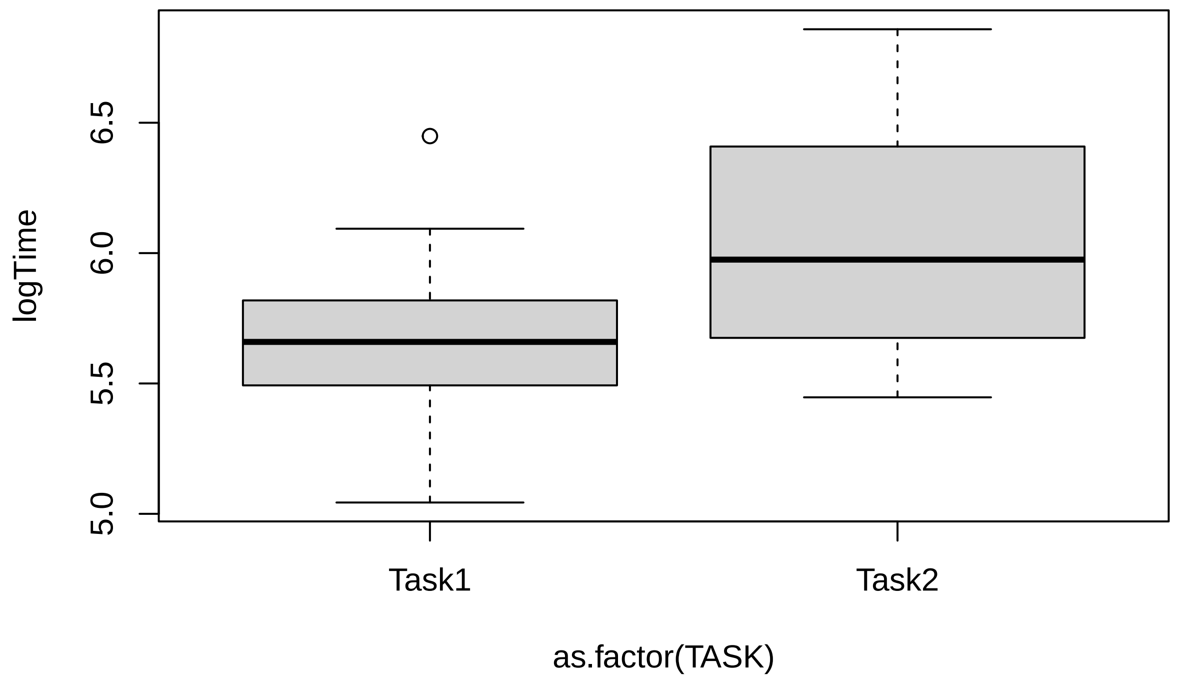
```r
hist(anova1[anova1$TASK == "Task1",]$logTime,
     main = "Histogram of log(Time) to complete Task 1") # histogram # histogram
```



```r
hist(anova1[anova1$TASK == "Task2",]$logTime,
     main = "Histogram of log(Time) to complete Task 2") # histogram
```



```r
plot(logTime ~ as.factor(TASK), data=anova1) # boxplot
```



Re-test for normality

```r
shapiro.test(anova1[anova1$TASK == "Task1",]$logTime)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  anova1[anova1$TASK == "Task1", ]$logTime
## W = 0.95825, p-value = 0.5094
```

```r
shapiro.test(anova1[anova1$TASK == "Task2",]$logTime)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  anova1[anova1$TASK == "Task2", ]$logTime
## W = 0.93905, p-value = 0.23
```

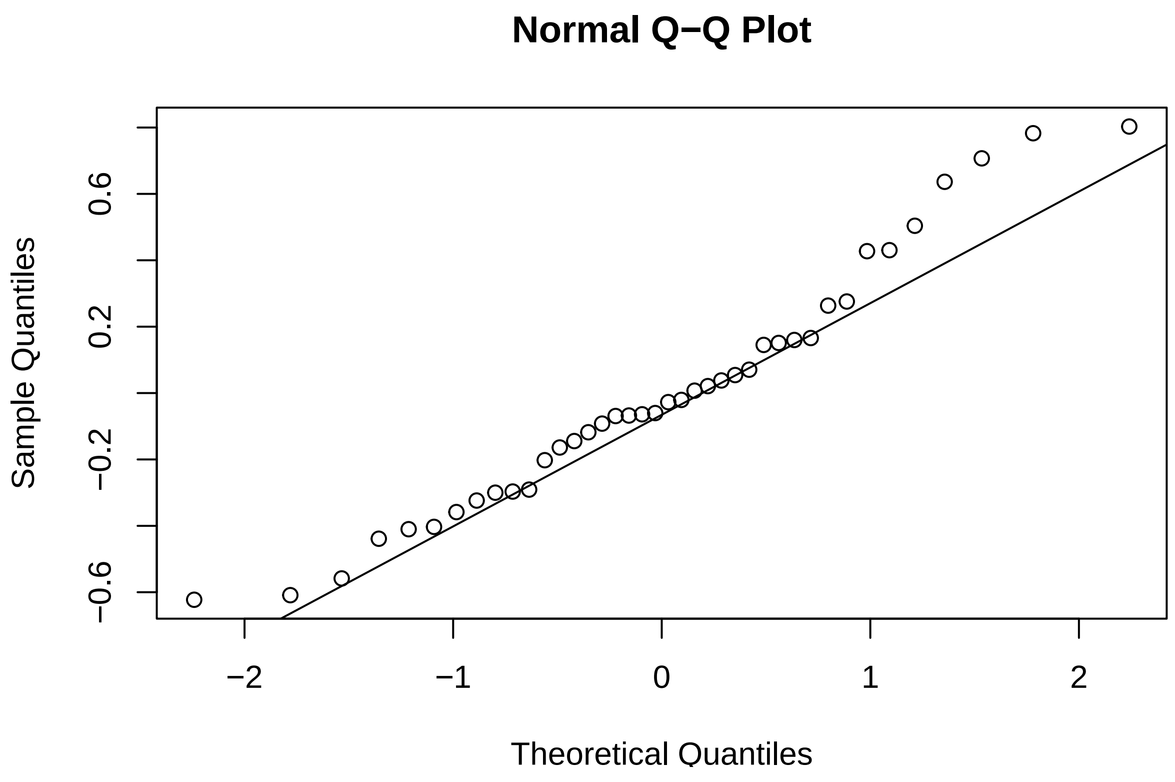
Re-test for normality of residuals

```r
m = aov(logTime ~ TASK, data=anova1) # fit model
shapiro.test(residuals(m)) # test residuals
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(m)
## W = 0.96218, p-value = 0.1987
```

Visually examine residuals

```r
qqnorm(residuals(m)); qqline(residuals(m)) # plot residuals
```



Re-test for homoscedasticity

```r
leveneTest(logTime ~ TASK, data=anova1, center=median) # Brown-Forsythe test
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value  Pr(>F)
## group  1  3.2638 0.07875 .
##       38
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Finally: Independent-samples t-test (now suitable for logTime)

```r
t.test(logTime ~ TASK, data=anova1, var.equal=TRUE)
```

```
##
##  Two Sample t-test
##
## data:  logTime by TASK
## t = -3.3121, df = 38, p-value = 0.002039
## alternative hypothesis: true difference in means between group Task1 and group Task2 is not equal to
## 95 percent confidence interval:
##  -0.6276133 -0.1514416
## sample estimates:
## mean in group Task1 mean in group Task2
##            5.666118            6.055645
```

Running a non-parametric test

If the underlying distribution is not normal, and none of the transformations that you have tried to make the data normal seem to work, it is time to run a non-paramteric test

```
Non paramteric tests, do not assume an underlying distribution
Instead, they compare compare and rank each row element is the first
group with other row elemets in second group.

When reporting results for non-parametric tests, you should report the median
and not the mean
```

```r
wilcox_test(Time ~ as.factor(TASK), data=anova1, distribution="exact")
```

```
##
##  Exact Wilcoxon-Mann-Whitney Test
##
## data:  Time by as.factor(TASK) (Task1, Task2)
## Z = -2.9487, p-value = 0.002577
## alternative hypothesis: true mu is not equal to 0
```

```r
wilcox_test(logTime ~ as.factor(TASK), data=anova1, distribution="exact") # note: same result
```

```
##
##  Exact Wilcoxon-Mann-Whitney Test
##
## data:  logTime by as.factor(TASK) (Task1, Task2)
## Z = -2.9487, p-value = 0.002577
## alternative hypothesis: true mu is not equal to 0
```

This completes the section for running a one factor two level t test In the next section we explore how to run one factor multiple levels Anova test

---

One factor multiple levels parametric one way Anova between populations

We will add one more task to the previous dataset so that now we are comparing 3 tasks

One-way ANOVA

- Read in a data file with task completion times (min) now from 3 tasks

```
anova2 = read.csv("anova2.csv")
head(anova2)
```

```
##   Subject  TASK Time
## 1       1 Task1  341
## 2       2 Task1  291
## 3       3 Task1  283
## 4       4 Task1  155
## 5       5 Task1  271
## 6       6 Task1  270
```

```
anova2$TASK = factor(anova2$TASK) # convert to nominal factor
summary(anova2)
```

```
##     Subject         TASK         Time
##  Min.   : 1.00   Task1:20   Min.   :143.0
##  1st Qu.:15.75   Task2:20   1st Qu.:248.8
##  Median :30.50   Task3:20   Median :295.0
##  Mean   :30.50              Mean   :353.9
##  3rd Qu.:45.25              3rd Qu.:391.2
##  Max.   :60.00              Max.   :952.0
```

- View descriptive statistics by TASK

```
ddply(anova2, ~ TASK, function(data) summary(data$Time))
```

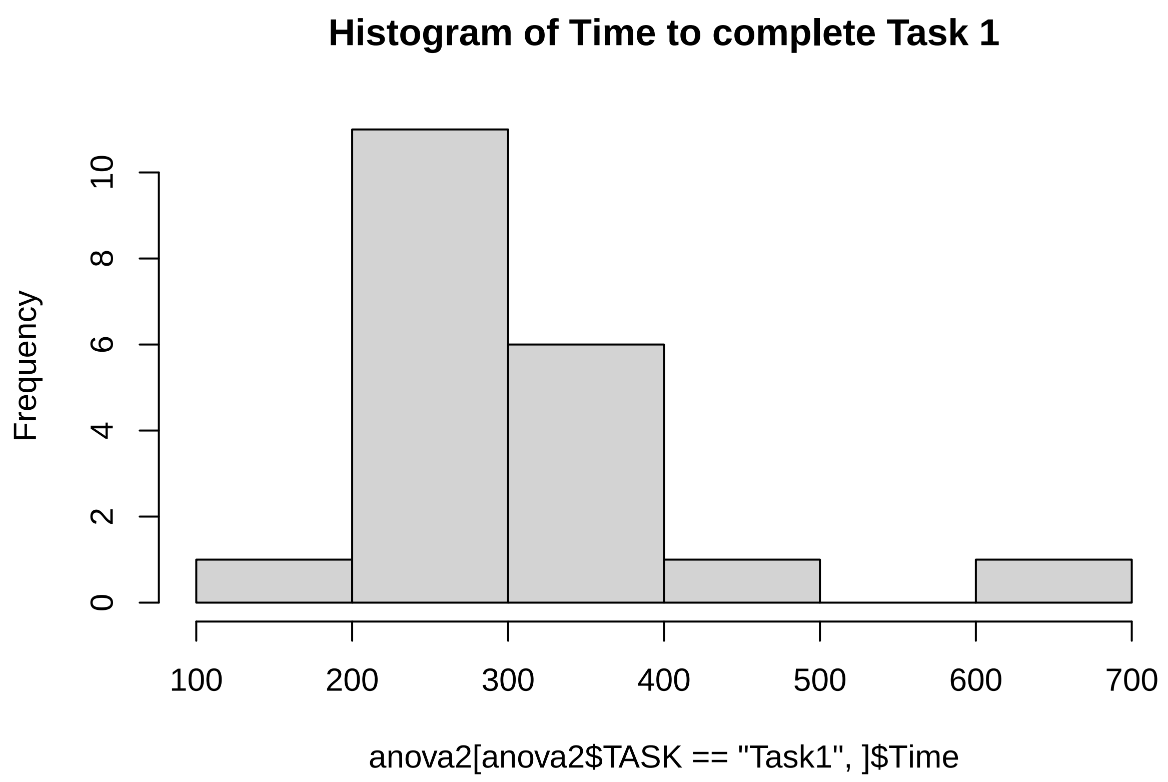
```
##    TASK Min. 1st Qu. Median   Mean 3rd Qu. Max.
## 1 Task1  155  246.50  287.0 302.10  335.25  632
## 2 Task2  232  294.75  393.5 468.15  585.50  952
## 3 Task3  143  232.25  279.5 291.45  300.00  572
```

```
ddply(anova2, ~ TASK, summarise, Time.mean=mean(Time), Time.sd=sd(Time))
```

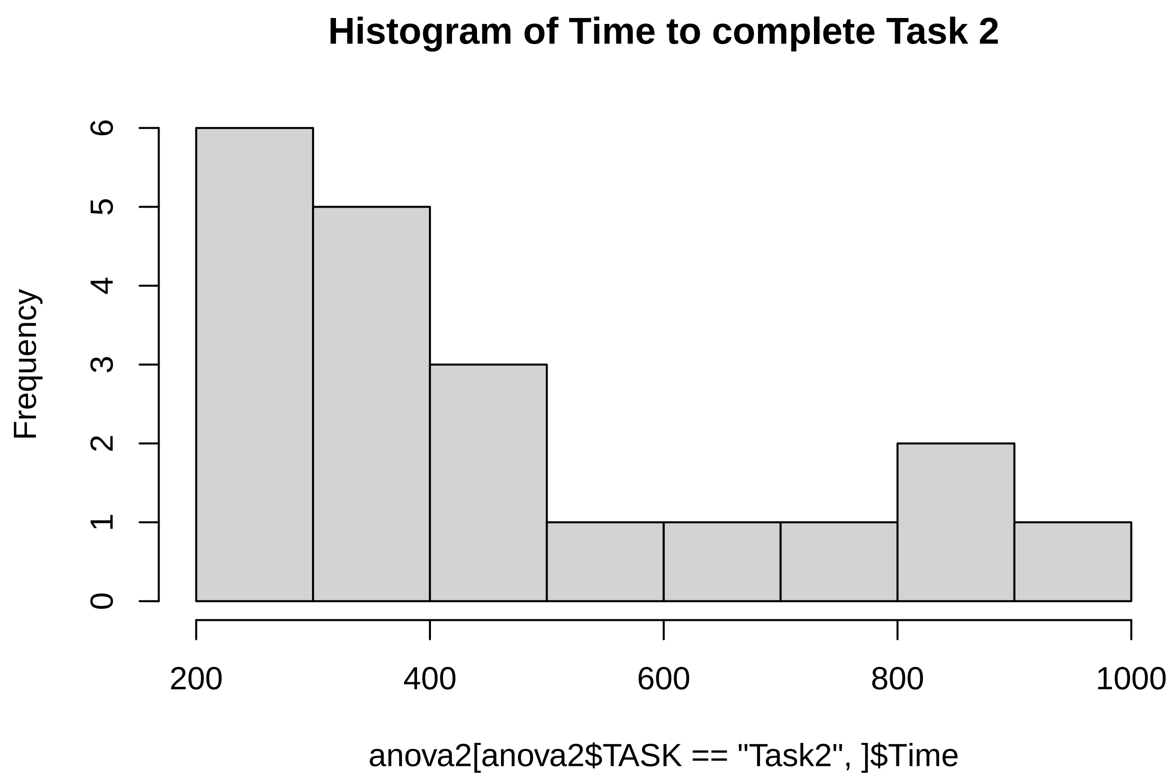
```
##    TASK Time.mean  Time.sd
## 1 Task1    302.10 101.0778
## 2 Task2    468.15 218.1241
## 3 Task3    291.45 106.8922
```

- Explore new response distribution

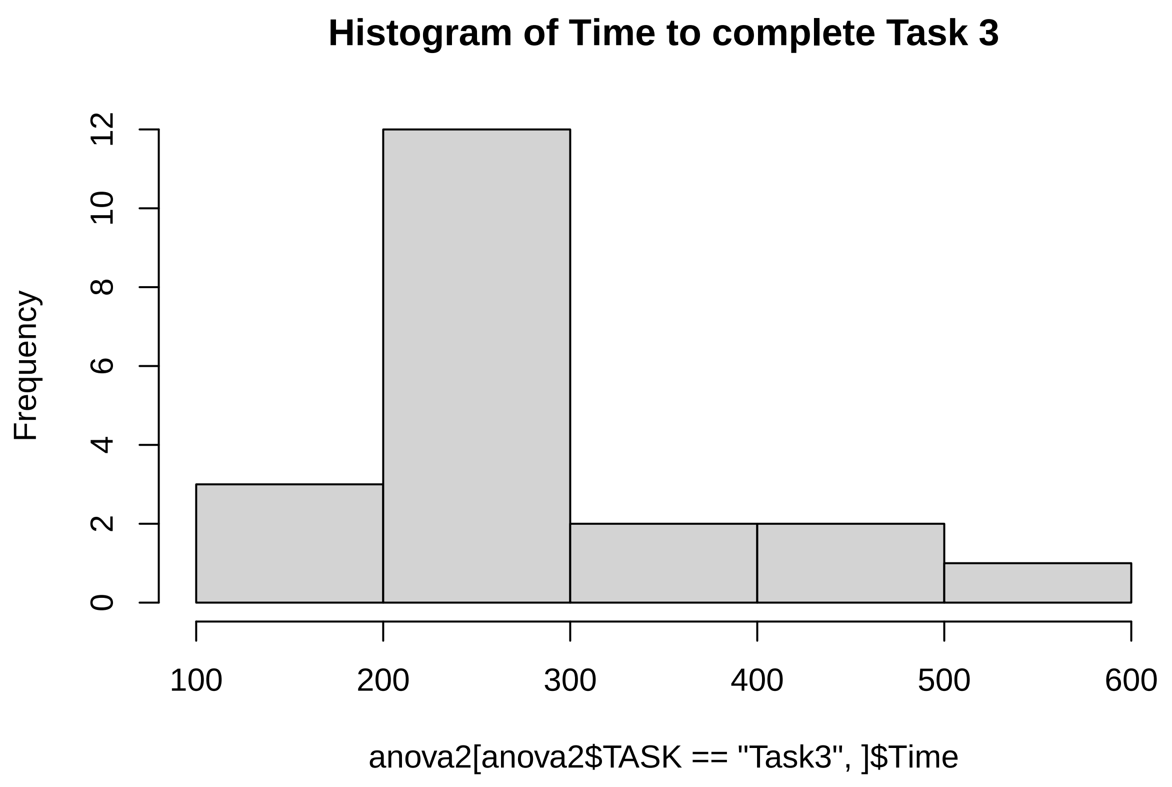
```
hist(anova2[anova2$TASK == "Task1",]$Time,
     main = "Histogram of Time to complete Task 1") # histogram
```



```r
hist(anova2[anova2$TASK == "Task2",]$Time,
     main = "Histogram of Time to complete Task 2") # histogram
```



```r
hist(anova2[anova2$TASK == "Task3",]$Time,
     main = "Histogram of Time to complete Task 3") # histogram # new one
```



```
#plot(Time ~ TASK, data=anova2) # boxplot
```

- Test normality for new TASK

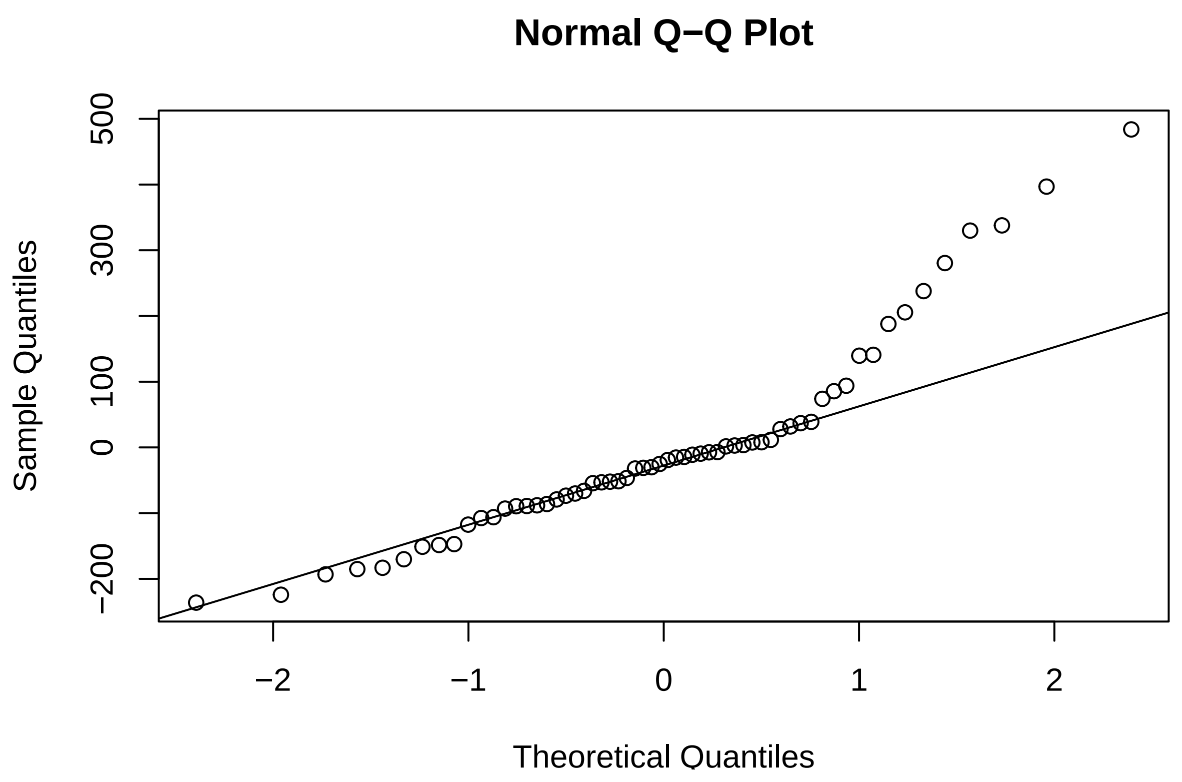
```
shapiro.test(anova2[anova2$TASK == "Task3",]$Time)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  anova2[anova2$TASK == "Task3", ]$Time
## W = 0.88623, p-value = 0.02294
```

```
m = aov(Time ~ TASK, data=anova2) # fit model
shapiro.test(residuals(m)) # test residuals
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(m)
## W = 0.89706, p-value = 0.000103
```

```
qqnorm(residuals(m)); qqline(residuals(m)) # plot residuals
```



- Test log-normality of new TASK

```
fit = fitdistr(anova2[anova2$TASK == "Task3",]$Time, "lognormal")$estimate
ks.test(anova2[anova2$TASK == "Task3",]$Time, "plnorm", meanlog=fit[1], sdlog=fit[2], exact=TRUE) # log
```

```
##
##  Exact one-sample Kolmogorov-Smirnov test
##
## data:  anova2[anova2$TASK == "Task3", ]$Time
## D = 0.1864, p-value = 0.4377
## alternative hypothesis: two-sided
```

- Compute new log(Time) column and re-test

```
anova2$logTime = log(anova2$Time) # add new column
head(anova2) # verify
```

```
##   Subject  TASK Time  logTime
## 1       1 Task1  341 5.831882
## 2       2 Task1  291 5.673323
## 3       3 Task1  283 5.645447
## 4       4 Task1  155 5.043425
## 5       5 Task1  271 5.602119
## 6       6 Task1  270 5.598422
```

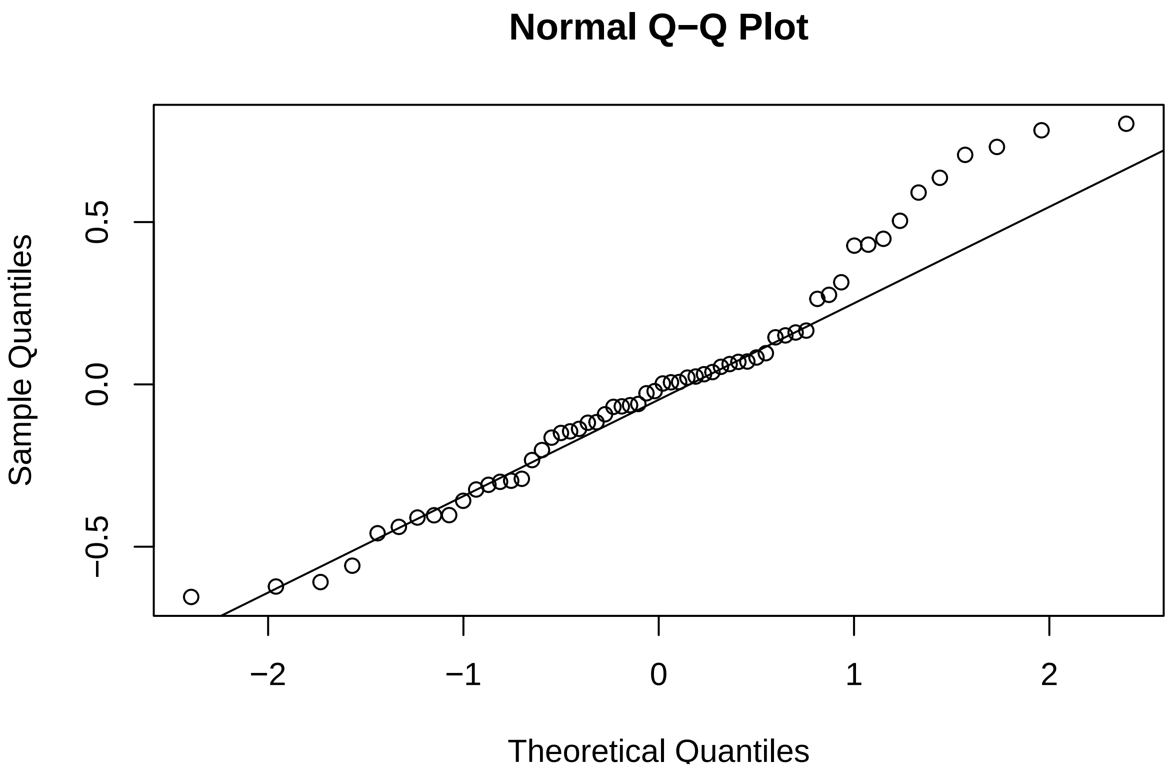
```
shapiro.test(anova2[anova2$TASK == "Task3",]$logTime)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  anova2[anova2$TASK == "Task3", ]$logTime
## W = 0.96579, p-value = 0.6648
```

```
m = aov(logTime ~ TASK, data=anova2) # fit model
shapiro.test(residuals(m)) # test residuals
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(m)
## W = 0.96563, p-value = 0.08893
```

```
qqnorm(residuals(m)); qqline(residuals(m)) # plot residuals
```



- Test homoscedasticity

```
leveneTest(logTime ~ TASK, data=anova2, center=median) # Brown-Forsythe test
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value Pr(>F)
## group  2  1.7797 0.1779
##       57
```

One-way ANOVA, suitable now to logTime

```
m = aov(logTime ~ TASK, data=anova2) # fit model
anova(m) # report anova
```

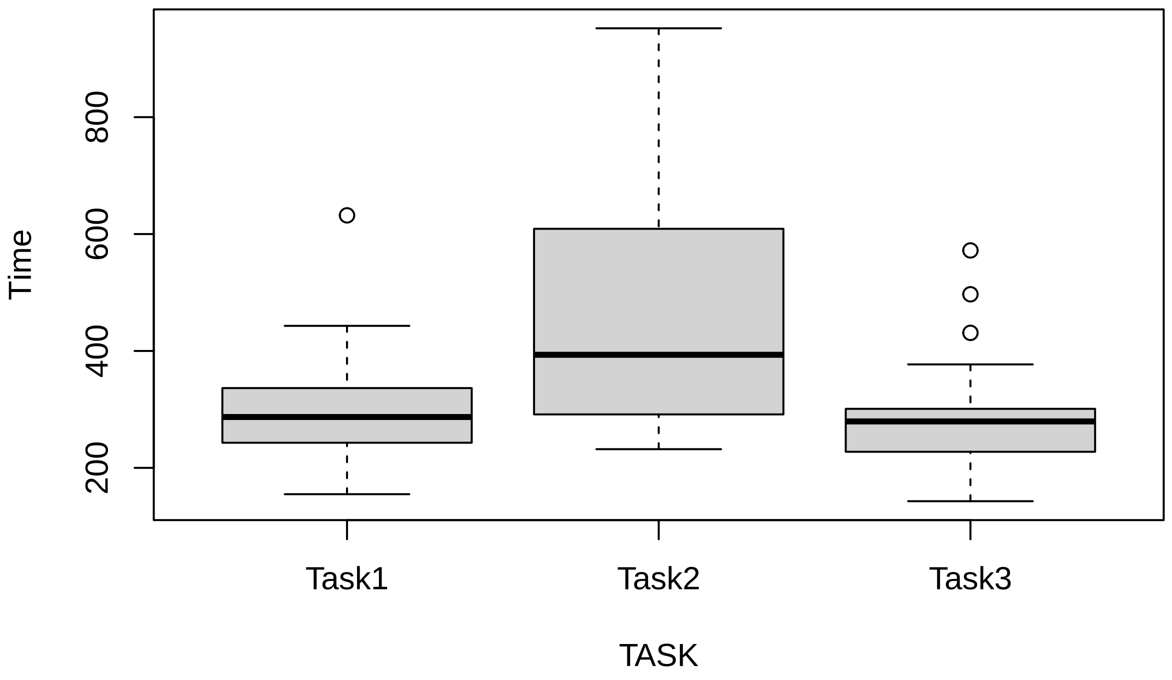
```
## Analysis of Variance Table
##
## Response: logTime
##           Df Sum Sq Mean Sq F value    Pr(>F)
## TASK       2 2.3064  1.1532   8.796 0.0004685 ***
## Residuals 57 7.4729  0.1311
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
How to analyze the output?
What is a F score?
```

Results of Anova tell us that atleast one of the task time is different than the rest, however it does not tell which task time is different

Next step is to do a pair-wise comparison of task times

```
plot(Time ~ TASK, data=anova2) # for convenience
```



```
summary(glht(m, mcp(TASK="Tukey")), test=adjusted(type="holm")) # Tukey means compare all pairs
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: aov(formula = logTime ~ TASK, data = anova2)
##
## Linear Hypotheses:
##                      Estimate Std. Error t value Pr(>|t|)
## Task2 - Task1 == 0    0.3895     0.1145   3.402 0.002458 **
## Task3 - Task1 == 0   -0.0485     0.1145  -0.424 0.673438
## Task3 - Task2 == 0   -0.4380     0.1145  -3.826 0.000978 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- holm method)
```

```
# adjusted(type="holm") is a corection applied while doing a pairwise comparison
```

- Task2 is different from Task1 and Task3
- Task1 and Task3 are similar

```
# Kruskal-Wallis test
PMCMRplus::kwAllPairsConoverTest(Time ~ as.factor(TASK), data=anova2, distribution="asymptotic") # can'
```

Non parametric anova

```
## Warning in kwAllPairsConoverTest.default(c(341L, 291L, 283L, 155L, 271L, : Ties
## are present. Quantiles were corrected for ties.
```

```
##
##  Pairwise comparisons using Conover's all-pairs test
```

```
## data: Time by as.factor(TASK)
```

```
##       Task1  Task2
## Task2 0.0085 -
## Task3 0.8992 0.0024
```

```
##
## P value adjustment method: single-step
```

```
PMCMRplus::kwAllPairsConoverTest(logTime ~ as.factor(TASK), data=anova2, distribution="asymptotic") # n
```

```
## Warning in kwAllPairsConoverTest.default(c(5.83188247728352, 5.67332326717149,
## : Ties are present. Quantiles were corrected for ties.
```

```
##
##  Pairwise comparisons using Conover's all-pairs test
```

```
## data: logTime by as.factor(TASK)
```

```
##       Task1  Task2
## Task2 0.0085 -
## Task3 0.8992 0.0024
```

```
##
## P value adjustment method: single-step
```

```
# for reporting Kruskal-Wallis as chi-square, we can get N with nrow(anova2)

# manual post hoc Mann-Whitney U pairwise comparisons
# note: wilcox_test we used above doesn't take two data vectors, so use wilcox.test
vs.ec = wilcox.test(anova2[anova2$TASK == "Task1",]$Time, anova2[anova2$TASK == "Task2",]$Time, exact=F
vs.py = wilcox.test(anova2[anova2$TASK == "Task1",]$Time, anova2[anova2$TASK == "Task3",]$Time, exact=F
ec.py = wilcox.test(anova2[anova2$TASK == "Task2",]$Time, anova2[anova2$TASK == "Task3",]$Time, exact=F
p.adjust(c(vs.ec$p.value, vs.py$p.value, ec.py$p.value), method="holm")
```

```
## [1] 0.007681846 0.588488864 0.007681846
```

```
# alternative approach is using PMCMR for nonparam pairwise comparisons

#posthoc.kruskal.conover.test(Time ~ TASK, data=anova2, p.adjust.method="holm") # deprecated
PMCMRplus::kwAllPairsConoverTest(Time ~ TASK, data=anova2, p.adjust.method="holm") # Conover & Iman (19
```

```
## Warning in kwAllPairsConoverTest.default(c(341L, 291L, 283L, 155L, 271L, : Ties
## are present. Quantiles were corrected for ties.
```

```
##
##  Pairwise comparisons using Conover's all-pairs test
```

```
## data: Time by TASK
##       Task1  Task2
## Task2 0.0062 -
## Task3 0.6620 0.0025
##
## P value adjustment method: holm
```

Factorial Anova (Crossed factor design)

> Suppose you want to understand the impact of pricing on retention for exiting users. Also, you want to test it across different user engagement levels

---

- There are 3 pricing levels - low (7.99) , medium (10.99) and high (12.99)
- There are 2 levels of user engagement for existing users - low and high. For this example, we assume the engagement levels definition is already established

---

Note: If you wanted to run an experiment that includes new users also, that would just be a separate "One factor [Price] multiple levels [low, medium and high] one way Anova test between populations", since we don't yet know the engagement level of new users

What questions are you trying to answer

- Does pricing affect retention ?
- Does user engagement level affect retention ?
- Is there an interaction affect between pricing and user engagment level? For example: Are highly engaged users insensitive to price?

```
pricing = read.csv("pricing.csv")
pricing$Engagement  <- factor( pricing$Engagement , levels = c("Low","High"))
pricing$Price <- factor( pricing$Price , levels = c("Low (7.99)","Medium (9.99)","High (12.99)"))

head(pricing)
```

```
##   X Engagement      Price Days
## 1 1       High Low (7.99)    6
## 2 2       High Low (7.99)    3
## 3 3       High Low (7.99)    6
## 4 4       High Low (7.99)    6
## 5 5       High Low (7.99)    5
## 6 6       High Low (7.99)    5
```

```
str(pricing)
```

```
## 'data.frame':    1440 obs. of  4 variables:
##  $ X         : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Engagement: Factor w/ 2 levels "Low","High": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Price     : Factor w/ 3 levels "Low (7.99)","Medium (9.99)",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Days      : int  6 3 6 6 5 5 6 3 4 4 ...
```

```
summary(pricing)
```

```
##        X           Engagement            Price          Days
##  Min.   :   1.0   Low :720   Low (7.99)   :480   Min.   :0.000
##  1st Qu.: 360.8   High:720   Medium (9.99):480   1st Qu.:3.000
##  Median : 720.5              High (12.99) :480   Median :5.000
##  Mean   : 720.5                                  Mean   :4.242
##   3rd Qu.:1080.2                                  3rd Qu.:6.000
##   Max.   :1440.0                                  Max.   :7.000
```

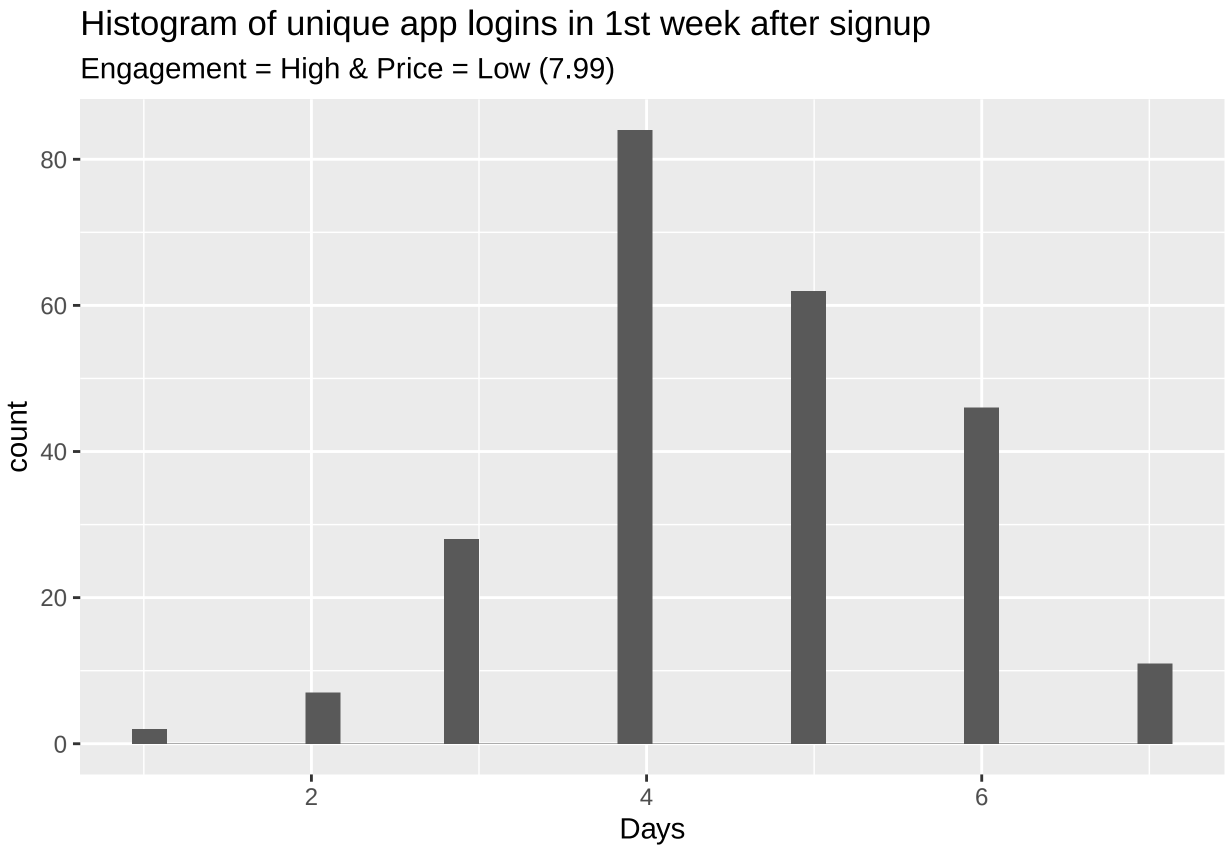
```
ddply(pricing, ~ Engagement * Price, function(data) summary(data$Days))
```

```
##   Engagement         Price Min. 1st Qu. Median     Mean 3rd Qu. Max.
## 1        Low     Low (7.99)    3       5    5.0 5.437500       6    7
## 2        Low Medium (9.99)    1       4    5.0 4.533333       5    7
## 3        Low  High (12.99)    0       0    1.0 1.666667       3    7
## 4       High     Low (7.99)    1       4    4.0 4.579167       5    7
## 5       High Medium (9.99)    0       4    5.5 5.337500       6    7
## 6       High  High (12.99)    0       3    4.0 3.895833       5    7
```

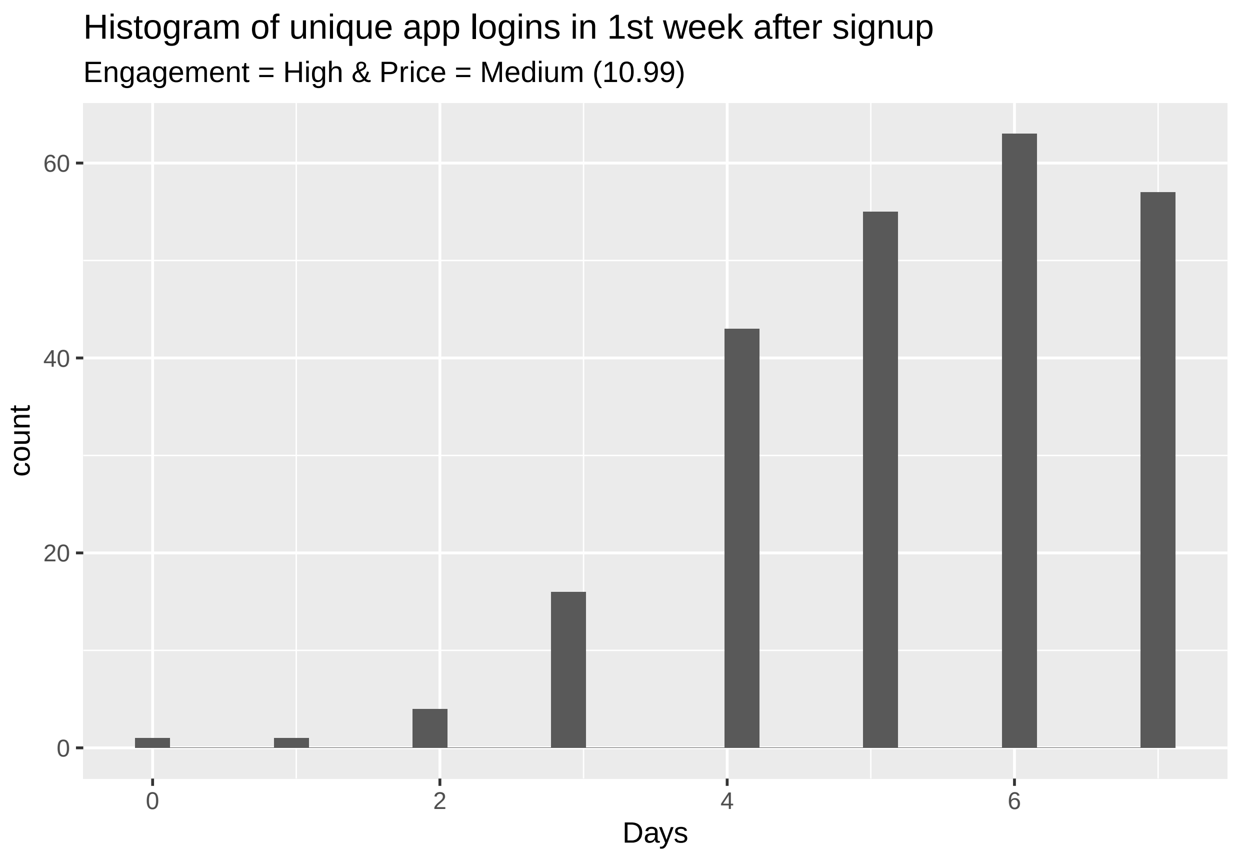
```
ddply(pricing, ~ Engagement * Price, summarise, Days.mean=mean(Days), Days.sd=sd(Days))
```

```
##   Engagement         Price Days.mean   Days.sd
## 1        Low     Low (7.99)  5.437500 0.9034163
## 2        Low Medium (9.99)  4.533333 1.1310256
## 3        Low  High (12.99)  1.666667 1.4910238
## 4       High     Low (7.99)  4.579167 1.1900843
## 5       High Medium (9.99)  5.337500 1.3685996
## 6       High  High (12.99)  3.895833 1.6622763
```

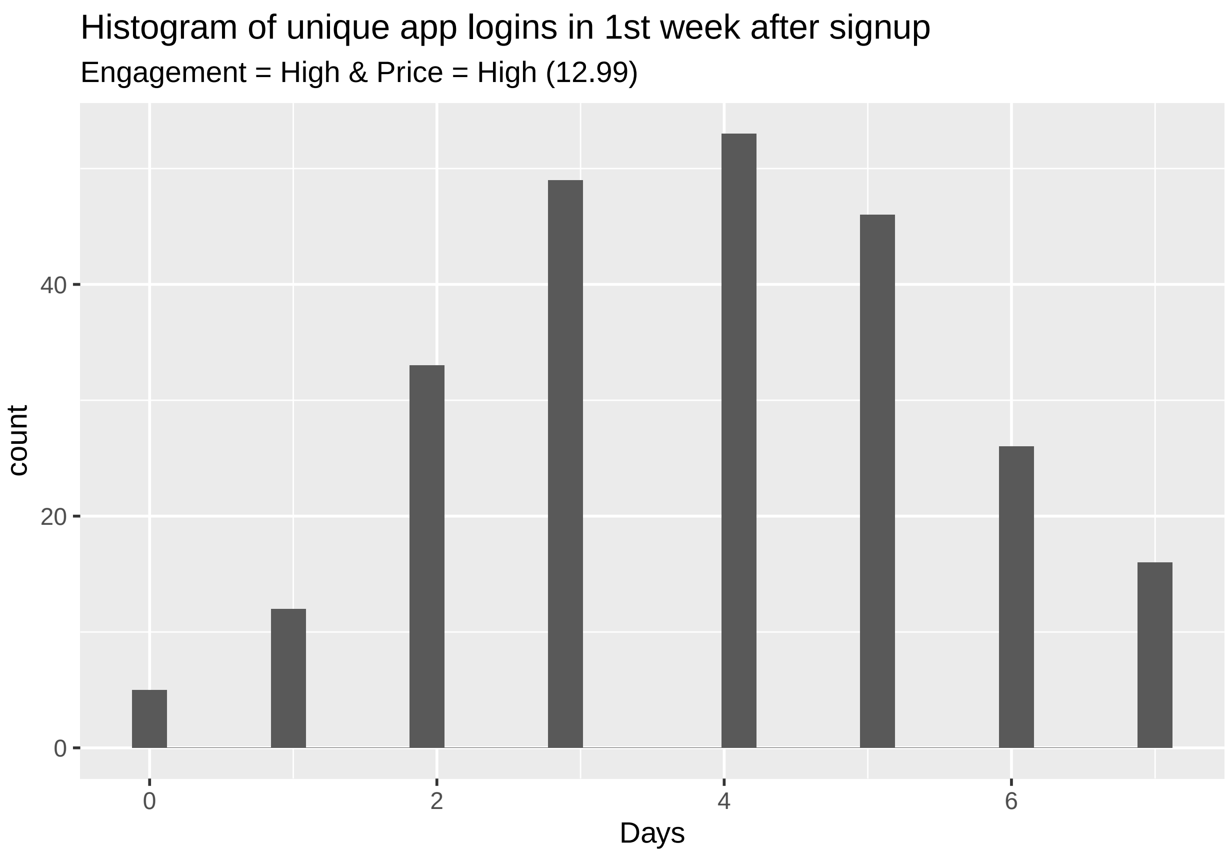
```
ggplot(data = pricing[pricing$Engagement == "High" & pricing$Price == "Low (7.99)",], aes(x=Days) ) + g
  labs(title = "Histogram of unique app logins in 1st week after signup", subtitle = "Engagement = High
```



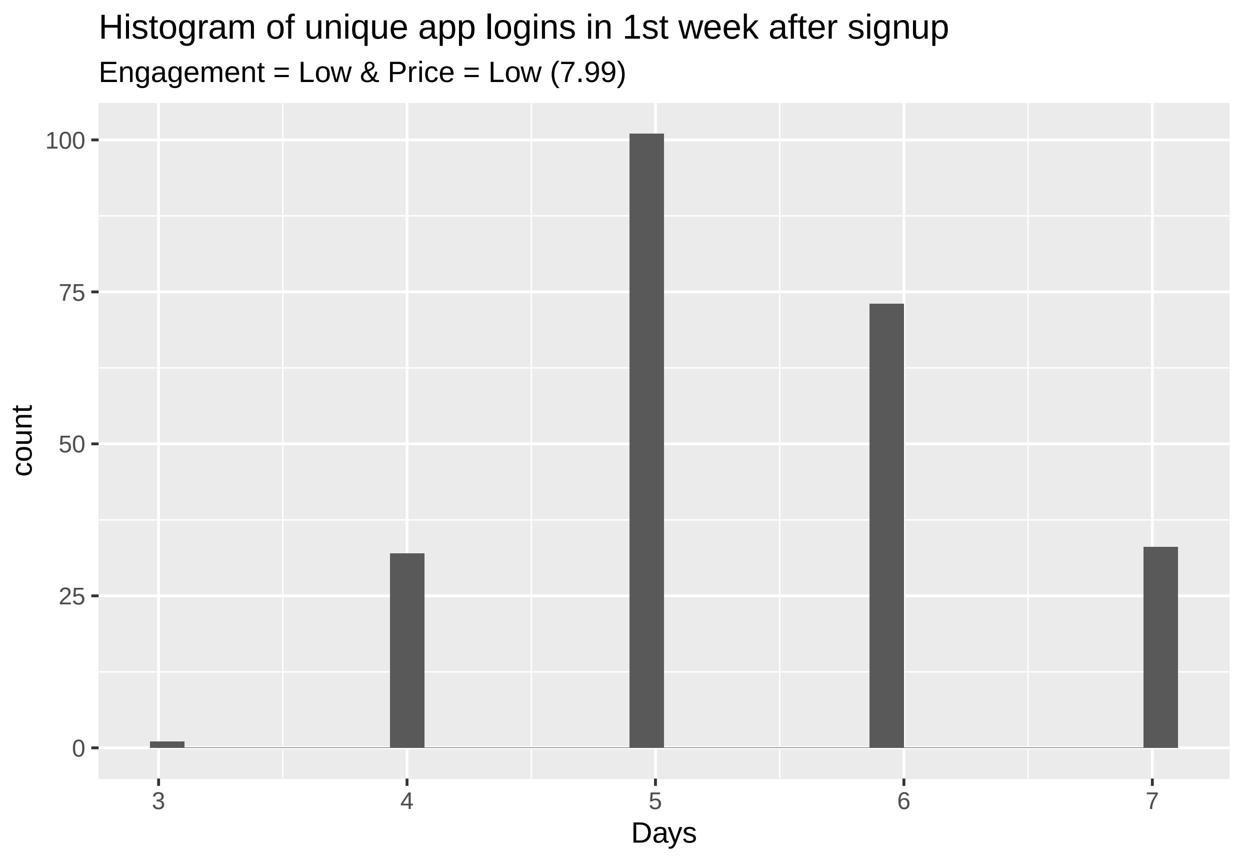
```
ggplot(data = pricing[pricing$Engagement == "High" & pricing$Price == "Medium (9.99)",], aes(x=Days) ) 
  labs(title = "Histogram of unique app logins in 1st week after signup", subtitle = "Engagement = High
```



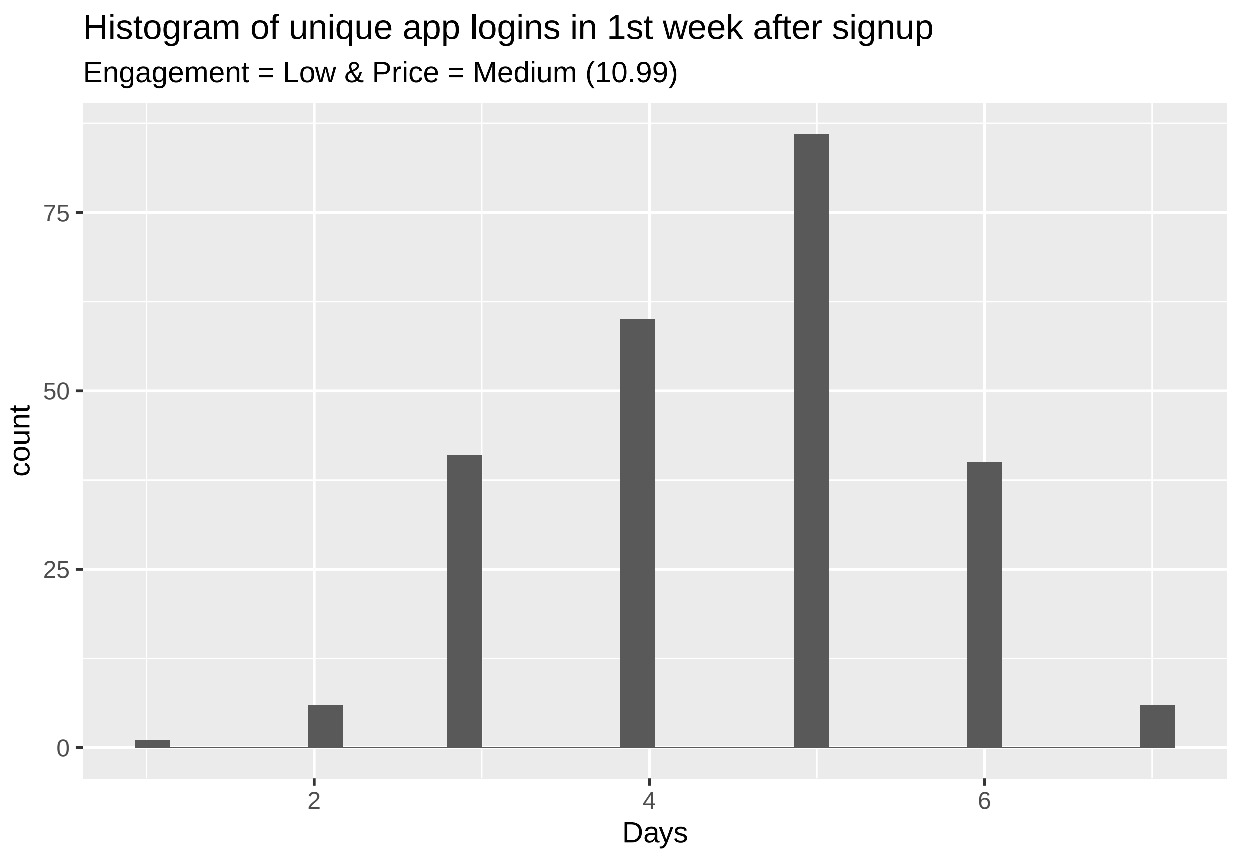
```
ggplot(data = pricing[pricing$Engagement == "High" & pricing$Price == "High (12.99)",], aes(x=Days) ) +
  labs(title = "Histogram of unique app logins in 1st week after signup", subtitle = "Engagement = High
```



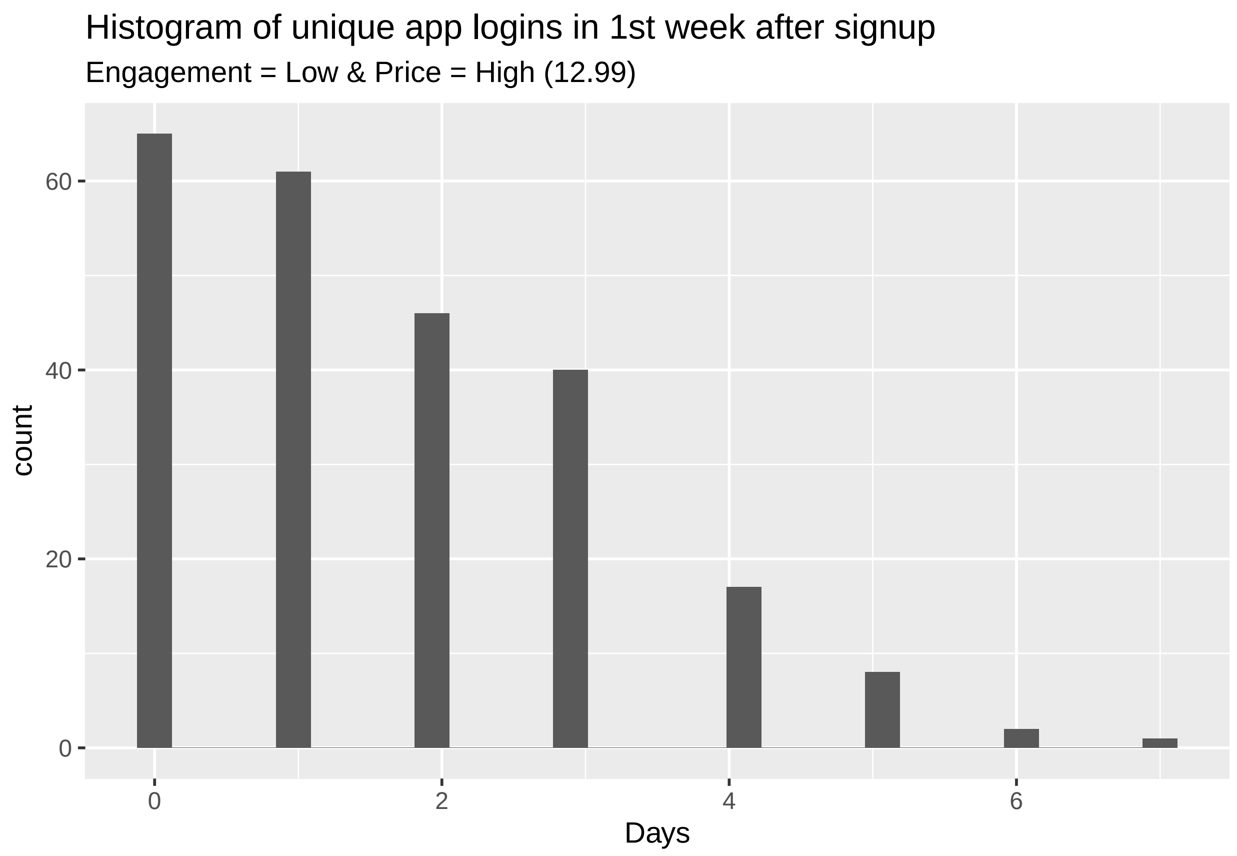
```
ggplot(data = pricing[pricing$Engagement == "Low" & pricing$Price == "Low (7.99)",], aes(x=Days) ) + geo
  labs(title = "Histogram of unique app logins in 1st week after signup", subtitle = "Engagement = Low &
```



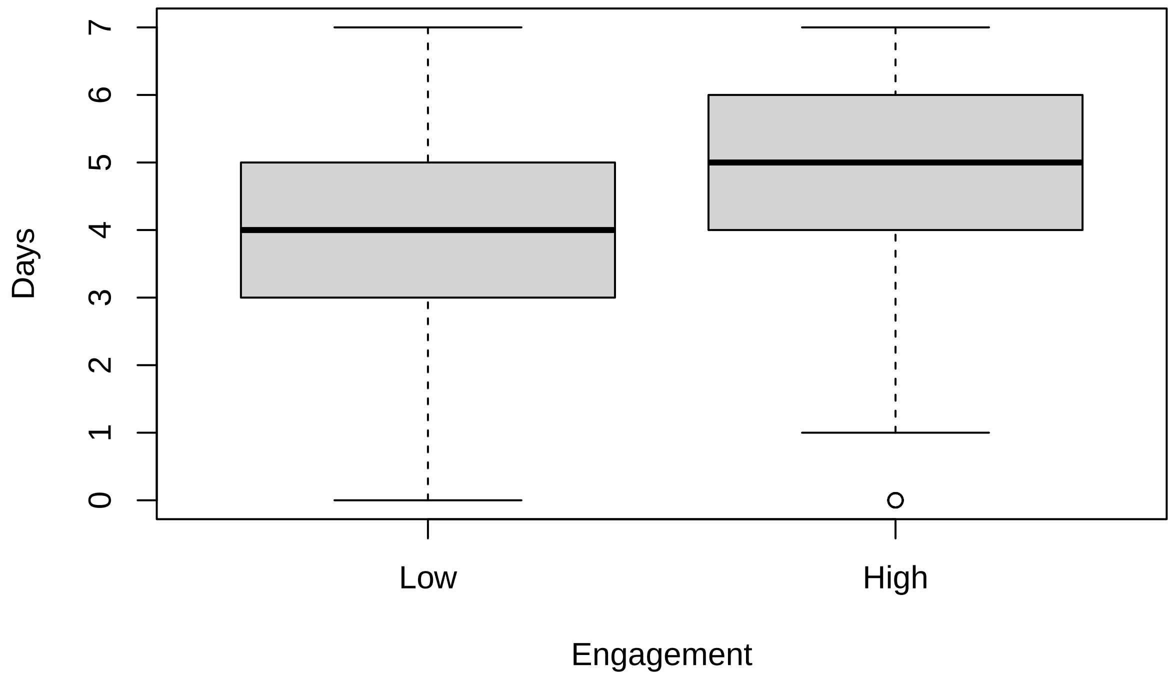
```r
ggplot(data = pricing[pricing$Engagement == "Low" & pricing$Price == "Medium (9.99)",], aes(x=Days) ) +
  labs(title = "Histogram of unique app logins in 1st week after signup", subtitle = "Engagement = Low &
```



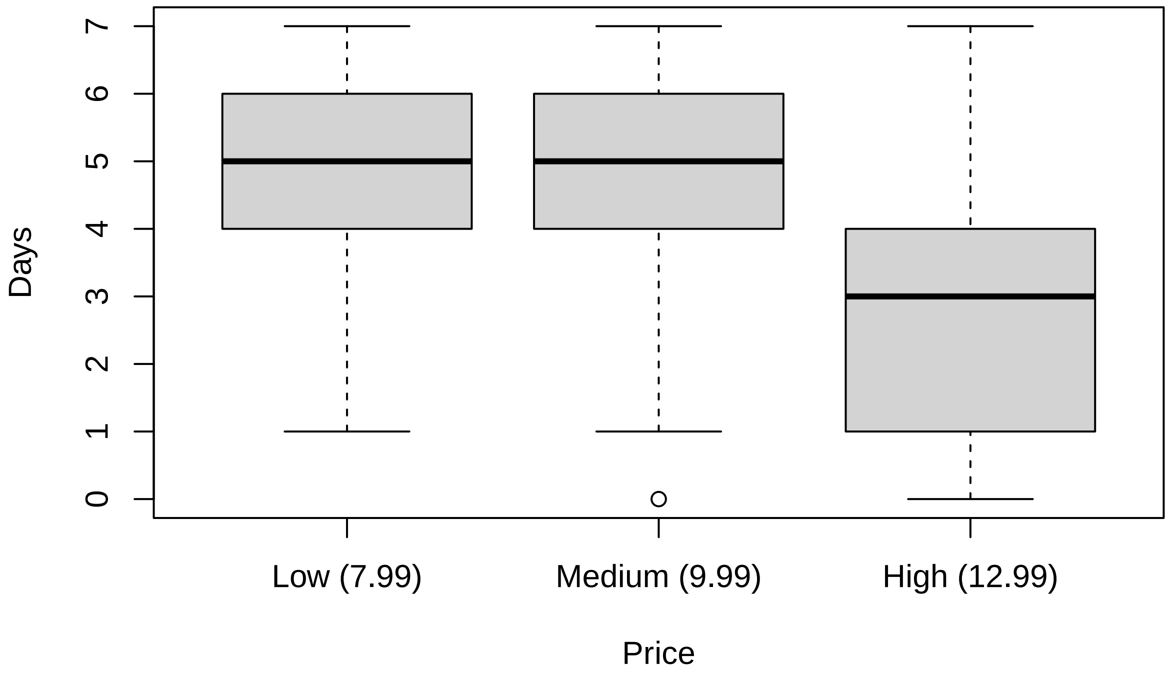
```
ggplot(data = pricing[pricing$Engagement == "Low" & pricing$Price == "High (12.99)",], aes(x=Days) ) + g
  labs(title = "Histogram of unique app logins in 1st week after signup", subtitle = "Engagement = Low &
```



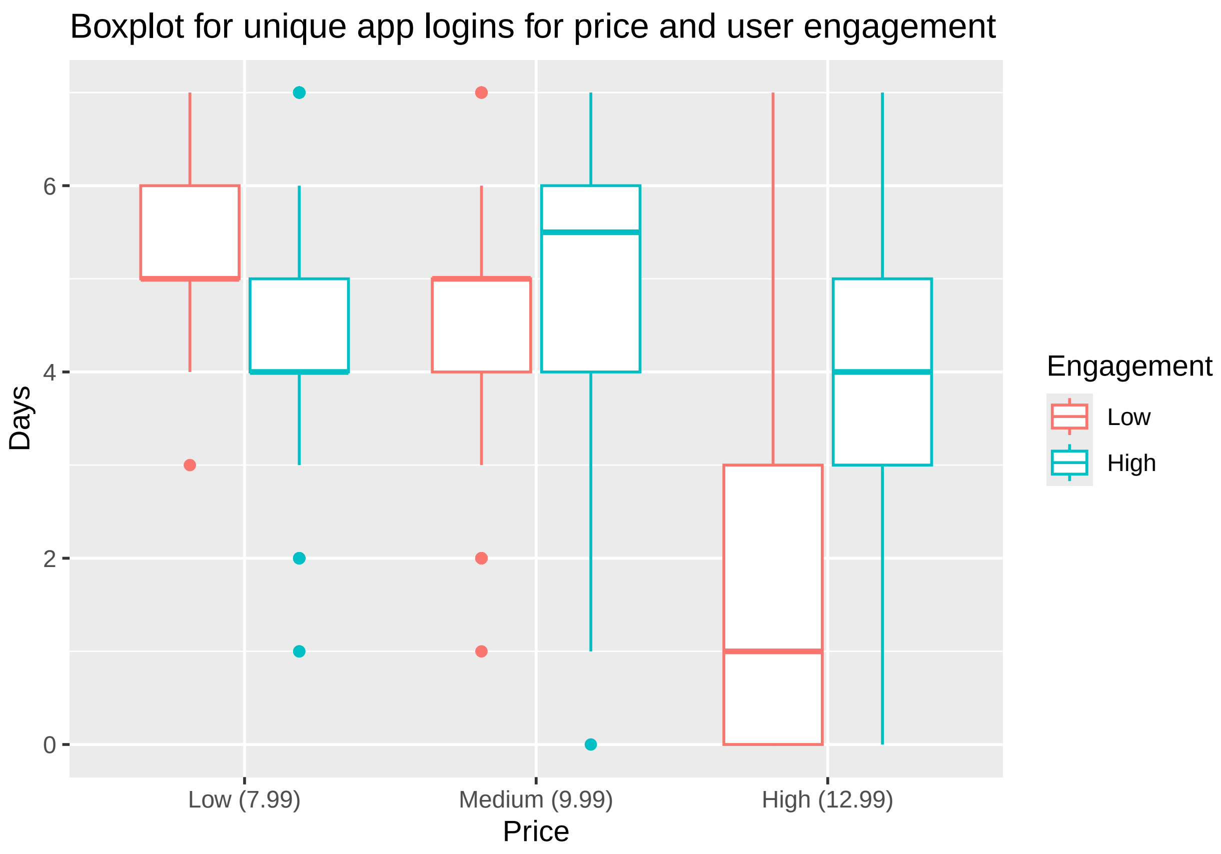
```r
boxplot(Days ~ Engagement , data=pricing, xlab="Engagement", ylab="Days") # boxplots
```



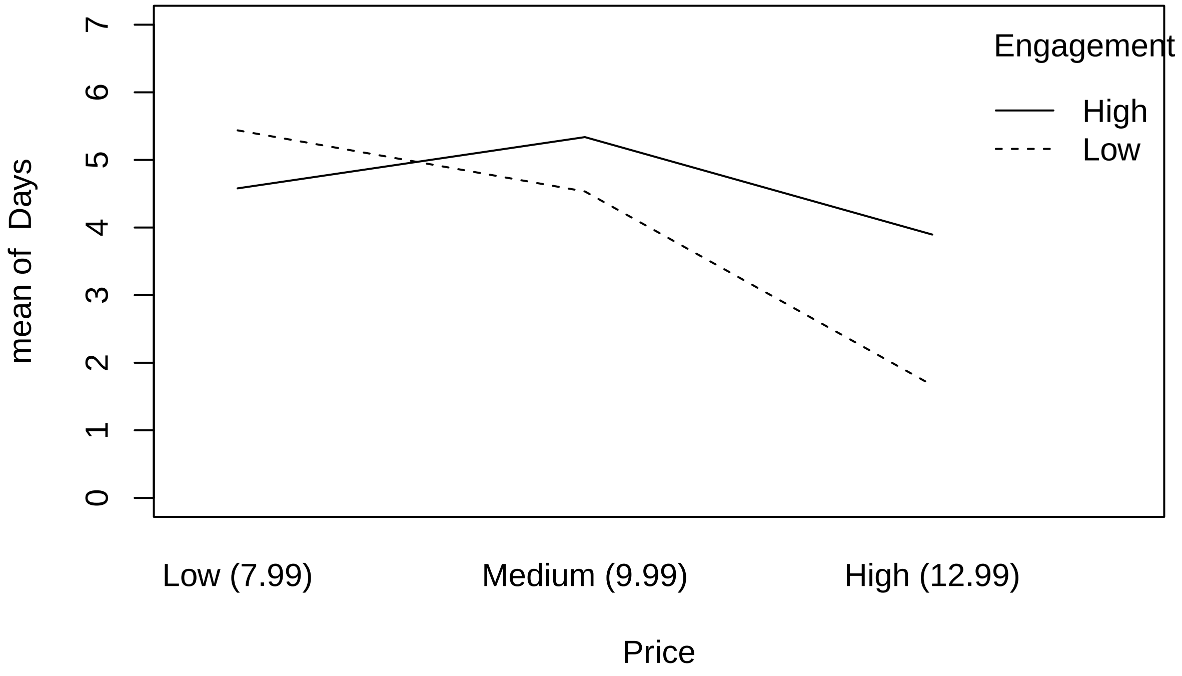
```r
boxplot(Days ~  Price, data=pricing, xlab="Price", ylab="Days") # boxplots
```



```r
ggplot(data = pricing ) + geom_boxplot(aes(x=Price, y = Days , color = Engagement)) +
  labs(title = "Boxplot for unique app logins for price and user engagement")
```



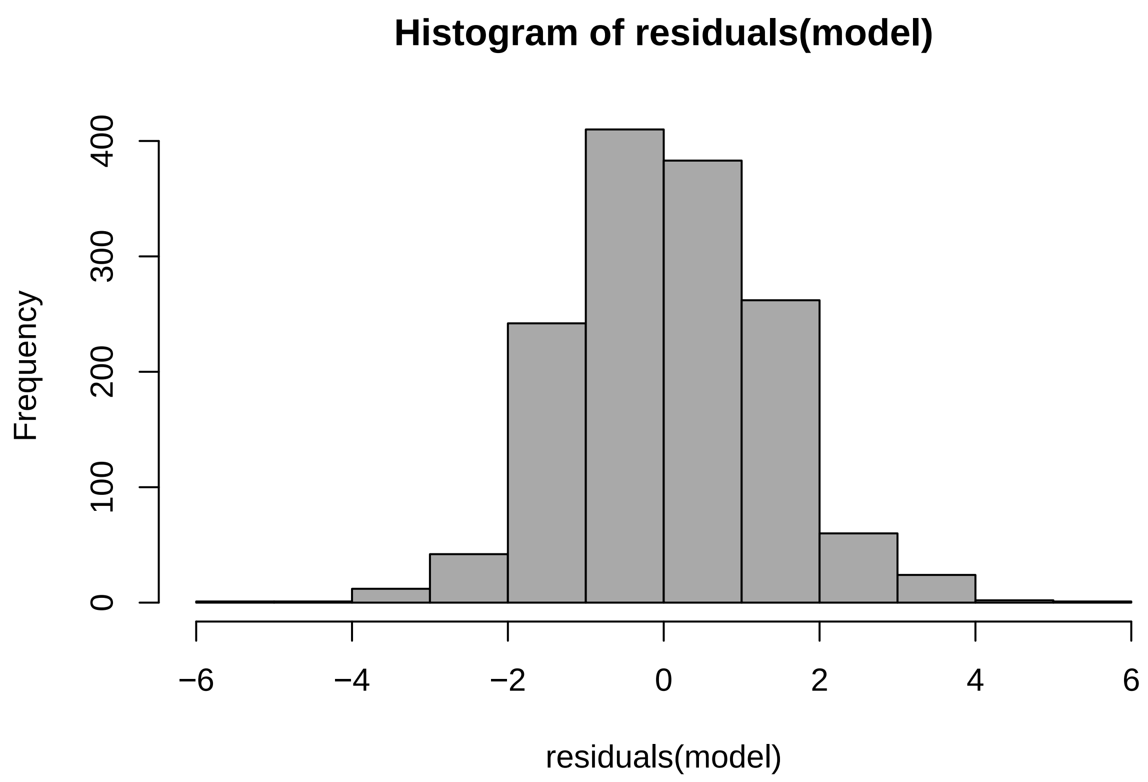
```r
with(pricing, interaction.plot(Price, Engagement, Days, ylim=c(0, max(pricing$Days)))) # interaction pl
```



```r
model = lm(Days ~ Price + Engagement + Price:Engagement,
           data=pricing)

Anova(model, type="II")
```

```
## Anova Table (Type II tests)
##
## Response: Days
##                    Sum Sq   Df F value    Pr(>F)
## Price             1536.90    2  444.57 < 2.2e-16 ***
## Engagement         189.22    1  109.47 < 2.2e-16 ***
## Price:Engagement   573.09    2  165.78 < 2.2e-16 ***
## Residuals         2478.68 1434
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
hist(residuals(model), col="darkgray")
```



```
shapiro.test(residuals(model))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(model)
## W = 0.98576, p-value = 1.069e-10
```

```
lsmeans(model, pairwise ~ Engagement, adjust="tukey")
```

```
## $lsmeans
##  Engagement lsmean    SE   df lower.CL upper.CL
##  Low          3.88 0.049 1434     3.78     3.98
##  High         4.60 0.049 1434     4.51     4.70
##
## Results are averaged over the levels of: Price
## Confidence level used: 0.95
##
## $contrasts
##  contrast   estimate     SE   df t.ratio p.value
##  Low - High   -0.725 0.0693 1434 -10.463  <.0001
##
## Results are averaged over the levels of: Price
```

```
lsmeans(model, pairwise ~ Price, adjust="tukey")
```

```
## $lsmeans
##  Price          lsmean   SE   df lower.CL upper.CL
##  Low (7.99)       5.01 0.06 1434     4.89     5.13
##  Medium (9.99)    4.94 0.06 1434     4.82     5.05
##  High (12.99)     2.78 0.06 1434     2.66     2.90
```

```
##
## Results are averaged over the levels of: Engagement
## Confidence level used: 0.95
##
## $contrasts
##  contrast                     estimate     SE   df t.ratio p.value
##  Low (7.99) - Medium (9.99)     0.0729 0.0849 1434   0.859  0.6661
##  Low (7.99) - High (12.99)      2.2271 0.0849 1434  26.243  <.0001
##  Medium (9.99) - High (12.99)   2.1542 0.0849 1434  25.383  <.0001
##
## Results are averaged over the levels of: Engagement
## P value adjustment: tukey method for comparing a family of 3 estimates
```

```
lsmeans(model, pairwise ~ Engagement:Price, adjust="tukey")
```

```
## $lsmeans
##  Engagement Price         lsmean     SE   df lower.CL upper.CL
##  Low        Low (7.99)      5.44 0.0849 1434     5.27     5.60
##  High       Low (7.99)      4.58 0.0849 1434     4.41     4.75
##  Low        Medium (9.99)   4.53 0.0849 1434     4.37     4.70
##  High       Medium (9.99)   5.34 0.0849 1434     5.17     5.50
##  Low        High (12.99)    1.67 0.0849 1434     1.50     1.83
##  High       High (12.99)    3.90 0.0849 1434     3.73     4.06
##
## Confidence level used: 0.95
##
## $contrasts
##  contrast                               estimate   SE   df t.ratio p.value
##  Low Low (7.99) - High Low (7.99)         0.8583 0.12 1434   7.152  <.0001
##  Low Low (7.99) - Low Medium (9.99)       0.9042 0.12 1434   7.534  <.0001
##  Low Low (7.99) - High Medium (9.99)      0.1000 0.12 1434   0.833  0.9614
##  Low Low (7.99) - Low High (12.99)        3.7708 0.12 1434  31.419  <.0001
##  Low Low (7.99) - High High (12.99)       1.5417 0.12 1434  12.845  <.0001
##  High Low (7.99) - Low Medium (9.99)      0.0458 0.12 1434   0.382  0.9989
##  High Low (7.99) - High Medium (9.99)    -0.7583 0.12 1434  -6.319  <.0001
##  High Low (7.99) - Low High (12.99)       2.9125 0.12 1434  24.267  <.0001
##  High Low (7.99) - High High (12.99)      0.6833 0.12 1434   5.694  <.0001
##  Low Medium (9.99) - High Medium (9.99)  -0.8042 0.12 1434  -6.700  <.0001
##  Low Medium (9.99) - Low High (12.99)     2.8667 0.12 1434  23.885  <.0001
##  Low Medium (9.99) - High High (12.99)    0.6375 0.12 1434   5.312  <.0001
##  High Medium (9.99) - Low High (12.99)    3.6708 0.12 1434  30.586  <.0001
##  High Medium (9.99) - High High (12.99)   1.4417 0.12 1434  12.012  <.0001
##  Low High (12.99) - High High (12.99)    -2.2292 0.12 1434 -18.574  <.0001
##
## P value adjustment: tukey method for comparing a family of 6 estimates
```

Factorial Anova (Crossed factor design): Non parametric test Now we run the above test using a non-parametic approach

`Note: Non parametric tests typically suffer from Type 1 error for interaction terms`

```
model = art(Days ~ Price * Engagement, data=pricing) # uses LMM
anova(model)
```

```
## Analysis of Variance of Aligned Rank Transformed Data
##
```

```
## Table Type: Anova Table (Type III tests)
## Model: No Repeated Measures (lm)
## Response: art(Days)
##
##                      Df Df.res F value     Pr(>F)
## 1 Price               2   1434  397.04 < 2.22e-16 ***
## 2 Engagement          1   1434  102.61 < 2.22e-16 ***
## 3 Price:Engagement    2   1434  164.21 < 2.22e-16 ***
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#Omnibus test

for (engagement in c("High","Low") ) {
  #print(str_c("engagement = " ,engagement ))
  if ( engagement == "High") {
    highly_engaged_7.99_9.99 <- (wilcox.test(pricing[pricing$Price == "Low (7.99)" & pricing$Engagement
    highly_engaged_7.99_12.99 <-  (wilcox.test(pricing[pricing$Price == "Low (7.99)" & pricing$Engagemen
    highly_engaged_7.99_12.99 <- (wilcox.test(pricing[pricing$Price == "High (12.99)" & pricing$Engagem
  }

  if ( engagement == "Low") {
    low_engaged_7.99_9.99 <- (wilcox.test(pricing[pricing$Price == "Low (7.99)" & pricing$Engagement ==
    low_engaged_7.99_12.99 <-  (wilcox.test(pricing[pricing$Price == "Low (7.99)" & pricing$Engagement
    low_engaged_7.99_12.99 <- (wilcox.test(pricing[pricing$Price == "High (12.99)" & pricing$Engagement
  }

}

p.adjust(c(highly_engaged_7.99_9.99$p.value, highly_engaged_7.99_12.99$p.value, highly_engaged_7.99_12.
```

```
## [1] 6.213742e-11 2.028246e-20 2.028246e-20 3.143035e-17 4.894782e-58
## [6] 4.894782e-58
```

Generalized Linear Models

Generalized Linear Models (GLM) extend Linear Models (LM) for studies with between factors to acommodate nominal (incl. binomial) or ordinal responses, or with non-normal response distributions (e.g., Poisson, exponential, gamma). All GLMs have a distribution and a link fn relating their factors to their response. The GLM generalizes the LM, which is a GLM with a normal distribution and "identity" link fn. See, e.g., http://en.wikipedia.org/wiki/Generalized_linear_model

Case study: Analyze user preference by Sex with multinomial logistic regression

Data: Table with user_id (Subject), Design preference (Pref) and Sex (M|F)

Questions to answer * Do users prefer one design over the other * Is the preference different by Gender

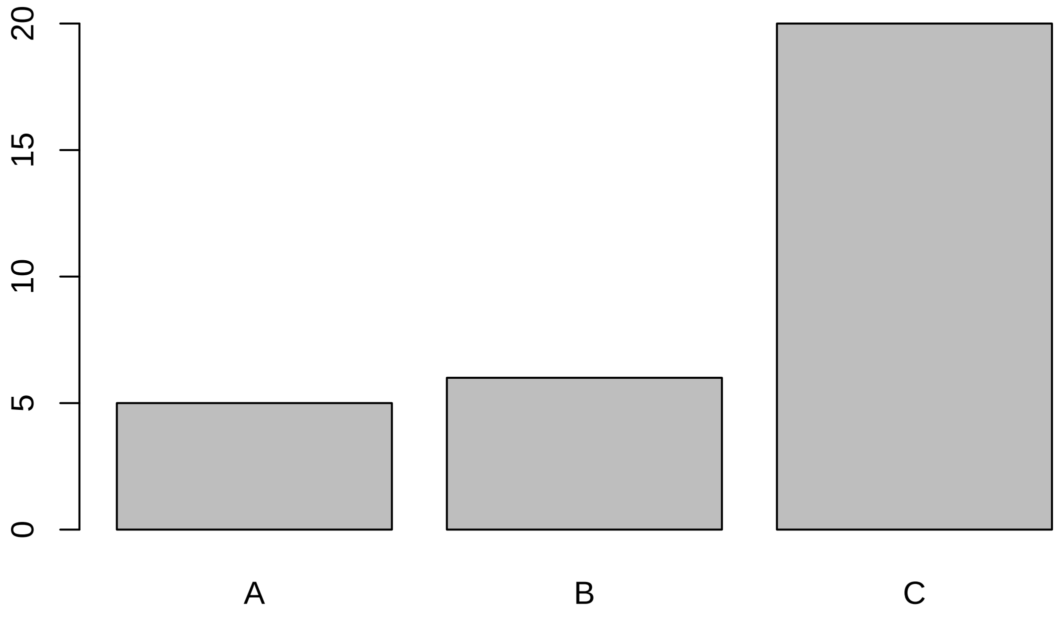
```
prefsABCsex.2 = read.csv("prefsABCsex.csv") # revisiting so add ".2"
head(prefsABCsex.2)
```

```
##   Subject Pref Sex
## 1       1    C   F
## 2       2    C   M
## 3       3    B   M
## 4       4    C   M
## 5       5    C   M
## 6       6    B   F
```

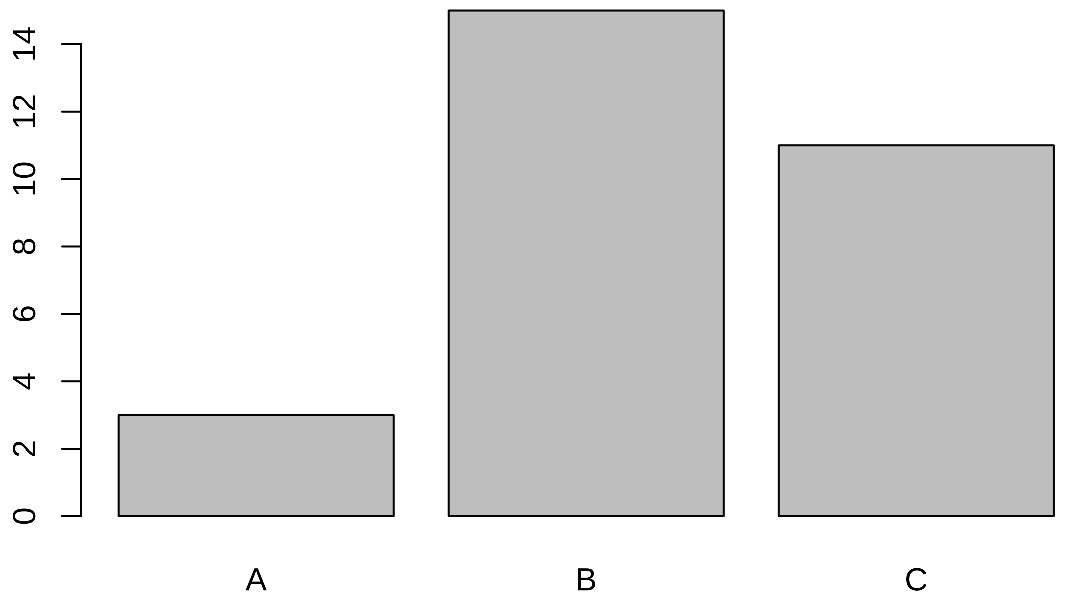
```
prefsABCsex.2$Subject = factor(prefsABCsex.2$Subject) # convert to nominal factor
prefsABCsex.2$Sex = as.factor(prefsABCsex.2$Sex)
summary(prefsABCsex.2)
```

```
##     Subject       Pref              Sex
##  1      : 1   Length:60          F:29
##  2      : 1   Class :character   M:31
##  3      : 1   Mode  :character
##  4      : 1
##  5      : 1
##  6      : 1
##  (Other):54
```

```
plot(as.factor(prefsABCsex.2[prefsABCsex.2$Sex == "M",]$Pref))
```



```
plot(as.factor(prefsABCsex.2[prefsABCsex.2$Sex == "F",]$Pref))
```



```
# set sum-to-zero contrasts for the Anova call
contrasts(prefsABCsex.2$Sex) <- "contr.sum"
m = multinom(Pref ~ Sex, data=prefsABCsex.2) # multinomial logistic
```

```
## # weights:  9 (4 variable)
## initial  value 65.916737
## iter  10 value 55.099353
## iter  10 value 55.099353
## final  value 55.099353
## converged
```

```
Anova(m, type=3) # note: not "anova" from stats pkg
```

```
## Analysis of Deviance Table (Type III tests)
##
## Response: Pref
##     LR Chisq Df Pr(>Chisq)
## Sex   7.0744  2    0.02909 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# note: if Pref had only had two response categories, we might use
# binomial regression, which uses the same syntax as Poisson regression
# below, but with family=binomial.
```

- Anova test shows that the preferences are different.
- Next steps is to do a pairwise comparsion to understand the difference in preference for males

```
ma = binom.test(sum(prefsABCsex.2[prefsABCsex.2$Sex == "M",]$Pref == "A"), nrow(prefsABCsex.2[prefsABCs
mb = binom.test(sum(prefsABCsex.2[prefsABCsex.2$Sex == "M",]$Pref == "B"), nrow(prefsABCsex.2[prefsABCs
mc = binom.test(sum(prefsABCsex.2[prefsABCsex.2$Sex == "M",]$Pref == "C"), nrow(prefsABCsex.2[prefsABCs
p.adjust(c(ma$p.value, mb$p.value, mc$p.value), method="holm") # correct for multiple comparisons
```

```
## [1] 0.109473564 0.126622172 0.001296754
```

Males preferred C over A and B

- Understanding the difference in preference for females

```
fa = binom.test(sum(prefsABCsex.2[prefsABCsex.2$Sex == "F",]$Pref == "A"), nrow(prefsABCsex.2[prefsABCs
fb = binom.test(sum(prefsABCsex.2[prefsABCsex.2$Sex == "F",]$Pref == "B"), nrow(prefsABCsex.2[prefsABCs
fc = binom.test(sum(prefsABCsex.2[prefsABCsex.2$Sex == "F",]$Pref == "C"), nrow(prefsABCsex.2[prefsABCs
p.adjust(c(fa$p.value, fb$p.value, fc$p.value), method="holm") # correct for multiple comparisons
```

```
## [1] 0.02703274 0.09447821 0.69396951
```

Females do not prefer A

Using GLM for poisson distributed data   When the underlying data does not follow a normal distribution

Previously, we had run Anova tests on Normally distributed data Using GLM, you can run Anova on data that is not normally distributed

In the example below, we will be looking at eror count data that follows a poisson distribution

Case study: You have rolled 2 different versions of the user profile page and you want to see how many users have errors entering their details on the page. We will compare results of the 2 new versions with the existing version

```
Poisson distribution is widely seen for count data.

The Poisson distribution may be useful to model events such as
* The number of meteorites greater than 1 meter diameter that strike Earth in a year
* The number of patients arriving in an emergency room between 10 and 11 pm
* In a laser the number of photons hitting a detector in a particular time interval will follow Poisson

In our example, poisson is a candidate distribution because
* Errors are discrete
* We assume that occurrence of one error does not affect the probability that a second error will occur
* Users cannot make 2 errors at the same time

https://en.wikipedia.org/wiki/Poisson_distribution
```

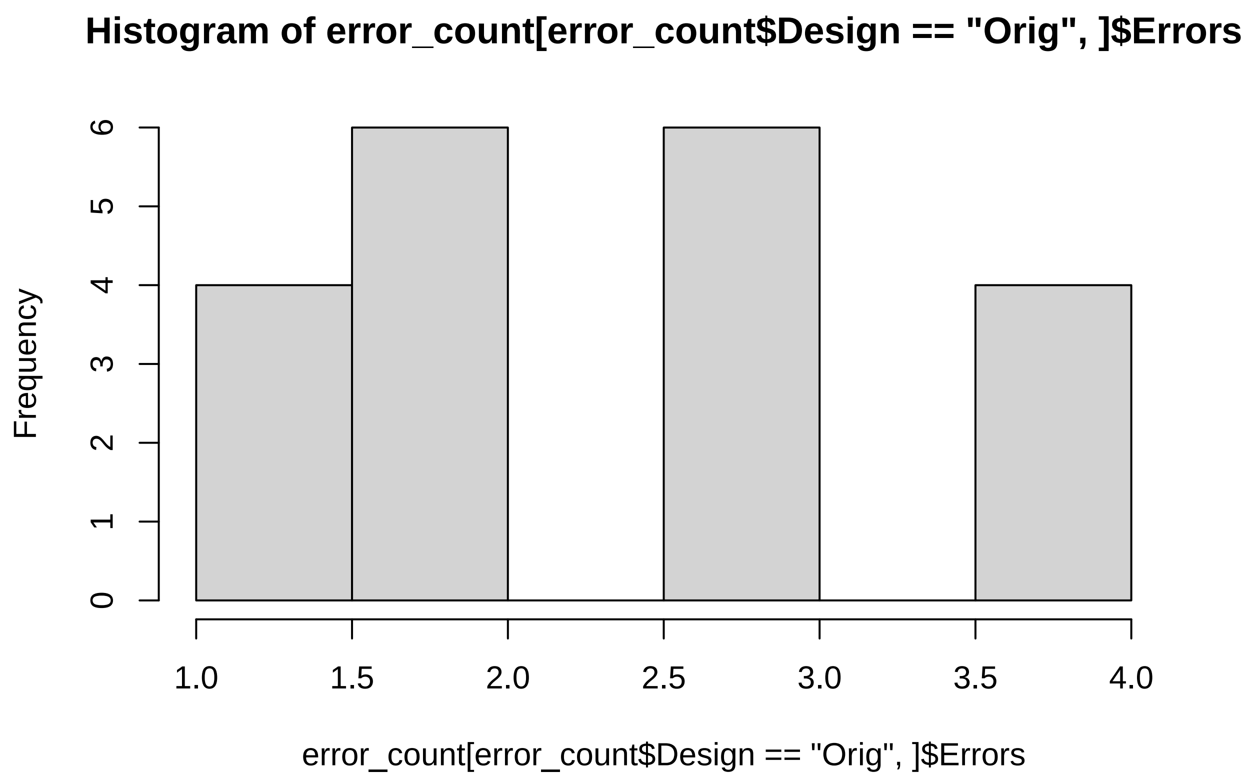
```
error_count <- read.csv("error_count.csv")
error_count$Design <- factor(error_count$Design, levels=c("Orig","New1","New2"))

ddply(error_count, ~ Design, function(data) summary(data$Errors))
```

```
##   Design Min. 1st Qu. Median Mean 3rd Qu. Max.
## 1   Orig    1    2.00    2.5 2.50    3.00    4
## 2   New1    0    0.00    0.5 0.70    1.00    2
## 3   New2    2    3.75    5.0 5.05    6.25    9
```

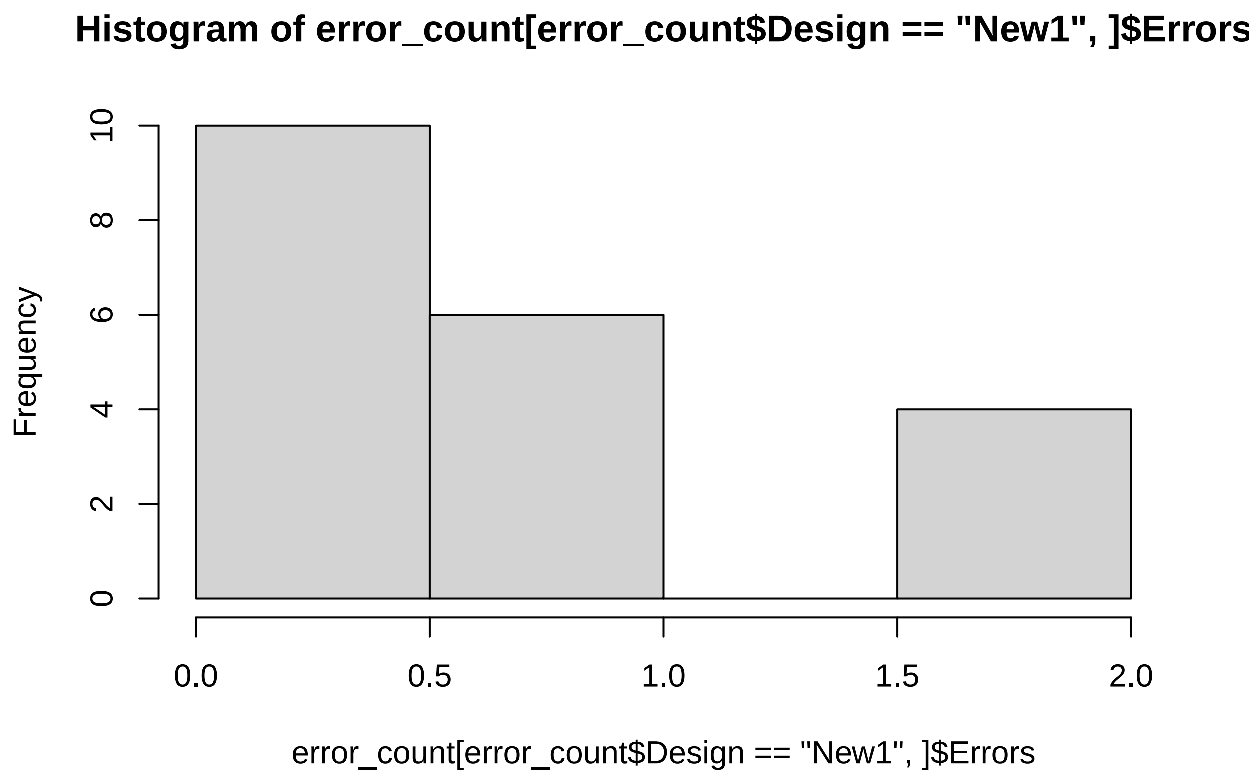
```
ddply(error_count, ~ Design, summarise, Errors.mean=mean(Errors), Errors.sd=sd(Errors))
```

```
##   Design Errors.mean Errors.sd
## 1   Orig        2.50 1.0513150
## 2   New1        0.70 0.8013147
## 3   New2        5.05 1.9049796
```

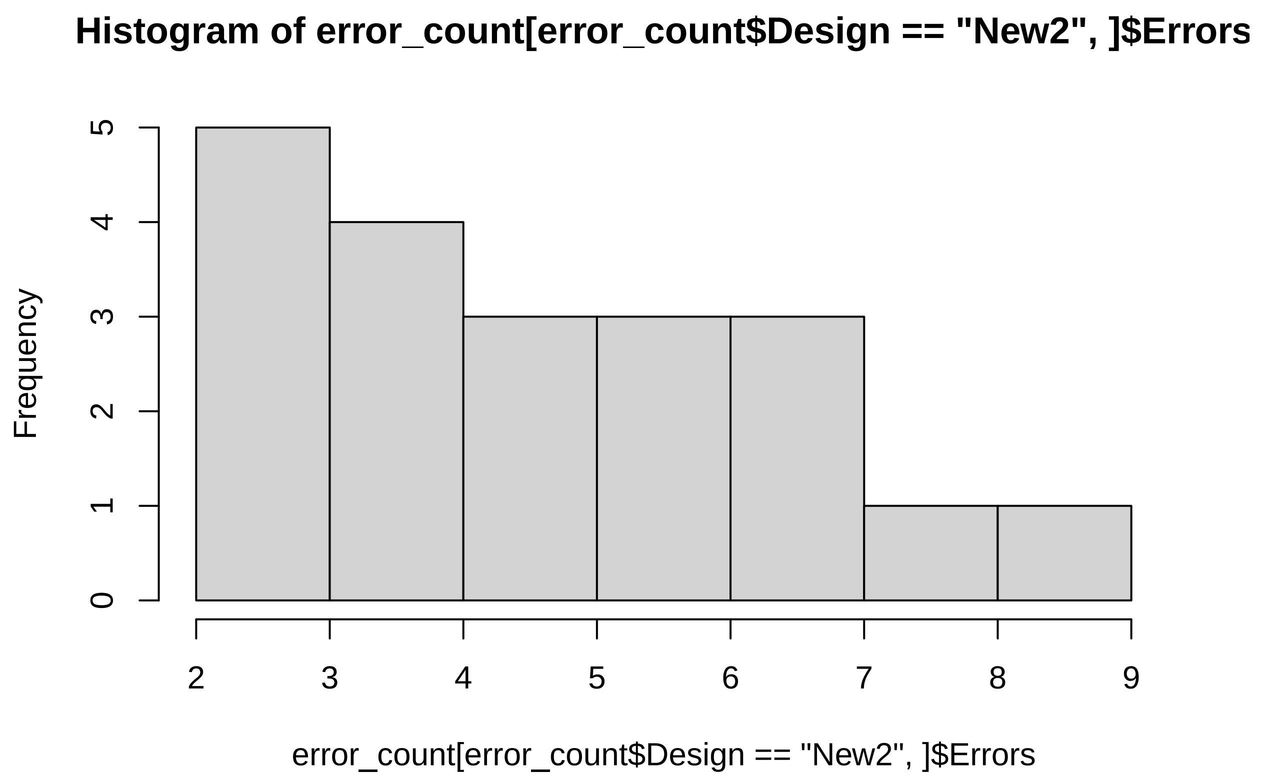
```
hist(error_count[error_count$Design == "Orig",]$Errors)
```



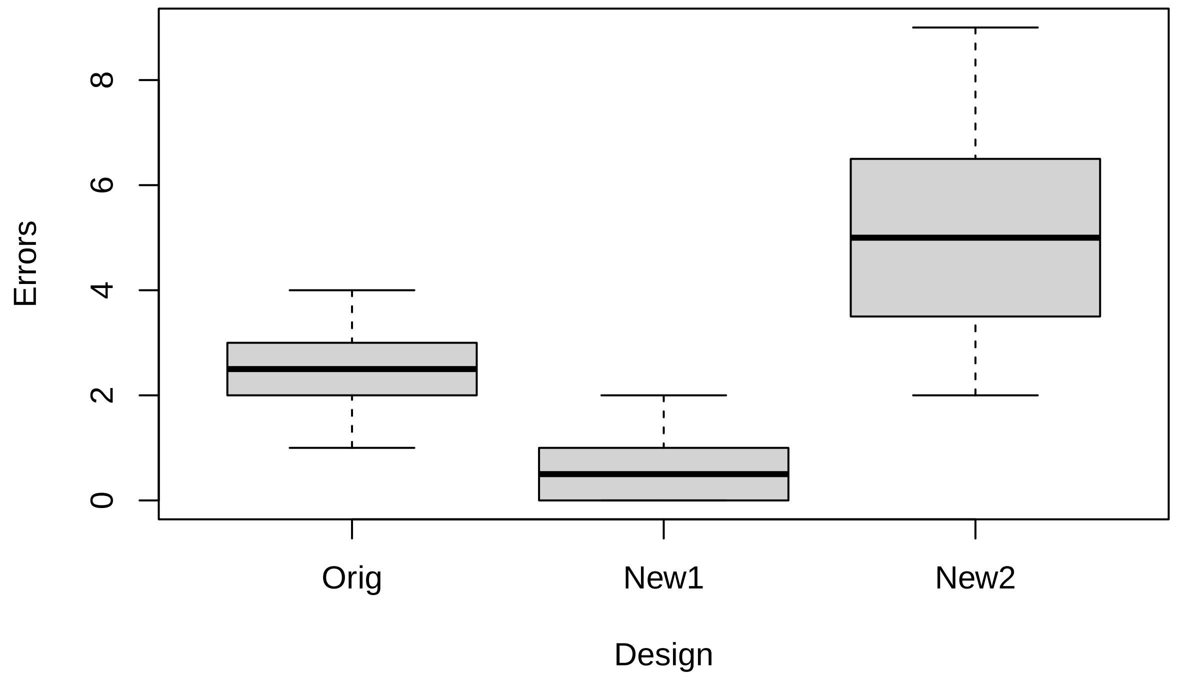
```
hist(error_count[error_count$Design == "New1",]$Errors)
```



```
hist(error_count[error_count$Design == "New2",]$Errors)
```



```r
plot(Errors ~ Design, data=error_count) # boxplot
```



```r
# re-verify that these data are Poisson-distributed
library(fitdistrplus)
fit = fitdist(error_count[error_count$Design == "Orig",]$Errors, "pois", discrete=TRUE)
gofstat(fit) # goodness-of-fit test
```

```
## Chi-squared statistic:  1.522232
## Degree of freedom of the Chi-squared distribution:  2
## Chi-squared p-value:  0.4671449
##    the p-value may be wrong with some theoretical counts < 5
## Chi-squared table:
##      obscounts theocounts
## <= 1  4.000000   5.745950
## <= 2  6.000000   5.130312
## <= 3  6.000000   4.275260
## > 3   4.000000   4.848477
##
## Goodness-of-fit criteria
##                                 1-mle-pois
## Akaike's Information Criterion   65.61424
## Bayesian Information Criterion   66.60997
```

```
fit = fitdist(error_count[error_count$Design == "New1",]$Errors, "pois", discrete=TRUE)
gofstat(fit) # goodness-of-fit test
```

```
## Chi-squared statistic:  0.3816087
## Degree of freedom of the Chi-squared distribution:  1
## Chi-squared p-value:  0.5367435
##    the p-value may be wrong with some theoretical counts < 5
## Chi-squared table:
##      obscounts theocounts
## <= 0 10.000000   9.931706
## <= 1  6.000000   6.952194
## > 1   4.000000   3.116100
##
## Goodness-of-fit criteria
##                                 1-mle-pois
## Akaike's Information Criterion   45.53208
## Bayesian Information Criterion   46.52781
```

```
fit = fitdist(error_count[error_count$Design == "New2",]$Errors, "pois", discrete=TRUE)
gofstat(fit) # goodness-of-fit test
```

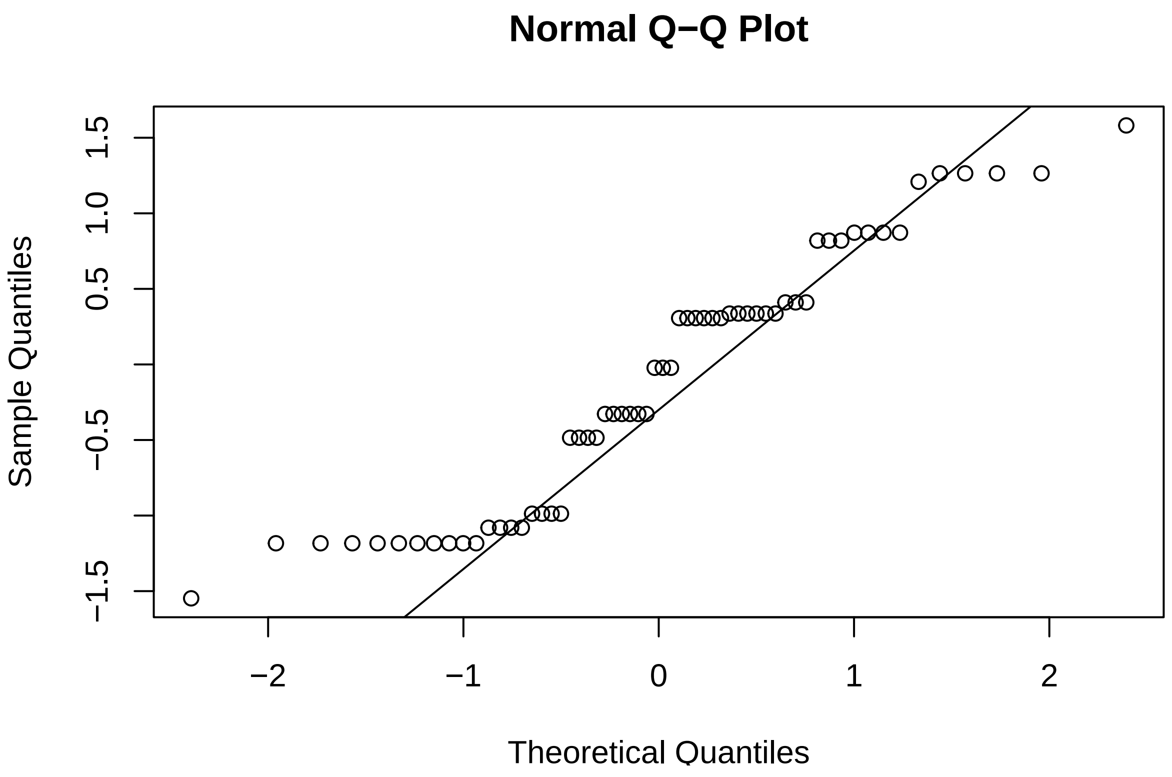
```
## Chi-squared statistic:  0.1611327
## Degree of freedom of the Chi-squared distribution:  3
## Chi-squared p-value:  0.9836055
##    the p-value may be wrong with some theoretical counts < 5
## Chi-squared table:
##      obscounts theocounts
## <= 3  5.000000   5.161546
## <= 4  4.000000   3.473739
## <= 5  3.000000   3.508476
## <= 6  3.000000   2.952967
## > 6   5.000000   4.903272
##
## Goodness-of-fit criteria
##                                 1-mle-pois
## Akaike's Information Criterion   84.19266
## Bayesian Information Criterion   85.18839
```

```
# analyze using Poisson regression
# set sum-to-zero contrasts for the Anova call
contrasts(error_count$Design) <- "contr.sum"
```

```r
# family parameter identifies both distribution and link fn
m = glm(Errors ~ Design, data=error_count, family=poisson)
Anova(m, type=3)
```

```
## Analysis of Deviance Table (Type III tests)
##
## Response: Errors
##        LR Chisq Df Pr(>Chisq)
## Design    74.93  2  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
qqnorm(residuals(m)); qqline(residuals(m)) # s'ok! Poisson regression makes no normality assumption
```



```r
# conduct pairwise comparisons among levels of Design
library(multcomp)
summary(glht(m, mcp(Design="Tukey")), test=adjusted(type="holm")) # Tukey means compare all pairs
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: glm(formula = Errors ~ Design, family = poisson, data = error_count)
##
## Linear Hypotheses:
##                  Estimate Std. Error z value Pr(>|z|)
## New1 - Orig == 0  -1.2730     0.3024  -4.210 5.11e-05 ***
## New2 - Orig == 0   0.7031     0.1729   4.066 5.11e-05 ***
## New2 - New1 == 0   1.9761     0.2852   6.929 1.27e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- holm method)
```