Assignment 1

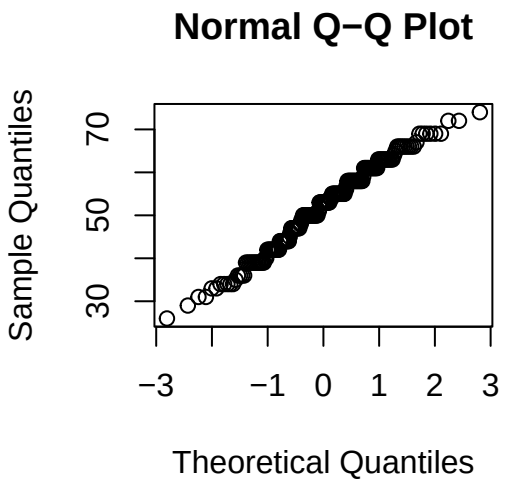
Martynas Vaznonois, Andrei Puchkov, Carlo Peron (Group 1)

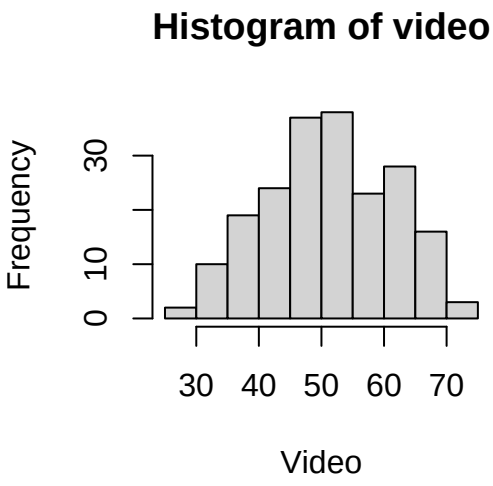
23 February 2024

Exercise 1

a) The histogram and the qq plot look fairly indicative of normality but the Shapiro-Wilk test provides evidence for the contrary.

```
par(mfrow=c(1, 2)); qqnorm(data$video)
hist(data$video, freq=T, main="Histogram of video", xlab="Video")
```






```
shapiro.test(data$video)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data$video
## W = 1, p-value = 0.03
```

Assuming normality, especially with a sample size $n > 30$, we use the z-score instead of the t_{n-1}-score to compute the 97% confidence interval. Rearranging the formula, we find the number of participants needed to make the CI at most 3. Finally, we compute the bootstrap CI by simulating sampling from the same distribution and calculating the mean test statistics.

```r
# CI
alpha = 0.03
ci <- qnorm(1 - alpha/2) * sd(data$video) / sqrt(length(data$video))
c(mean(data$video) - ci, mean(data$video) + ci)
```

```
## [1] 50.3 53.4
```

```r
# Participants
n <- (qnorm(1 - alpha/2) * sd(data$video) / (3/2))^2; ceiling(n)
```

```
## [1] 206
```

```r
# Bootstrap CI
B = 100000
Tstar = numeric(B)
for(i in 1:B)
  Tstar[i] = mean(sample(data$video, replace=T))
c(2*mean(Tstar) - quantile(Tstar, 1 - alpha/2), 2*mean(Tstar) - quantile(Tstar, alpha/2))
```

```
## 98.5%  1.5%
##  50.3  53.4
```

b) The first test below shows that the H_0 is rejected ($p < 0.001$). The confidence interval for the true mean is $(50.69; \infty)$. This shows that the true mean can be found within that interval with 95% certainty. Because the mean of 50 under H_0 is not in that interval, the probability of H_0 being true is less than 5%, leading to its rejection. Under the second test, the μ_0 is within that same interval, which is why the p-value is above 0.05 and why H_0 cannot be rejected.

```r
t.test(data$video, mu=50, alternative="greater")
```

```
##
##  One Sample t-test
##
## data:  data$video
## t = 3, df = 199, p-value = 0.004
## alternative hypothesis: true mean is greater than 50
## 95 percent confidence interval:
##  50.7  Inf
## sample estimates:
## mean of x
##      51.9
```

```r
t.test(data$video, mu=51, alternative="greater")
```

```
##
##  One Sample t-test
##
## data:  data$video
## t = 1, df = 199, p-value = 0.1
## alternative hypothesis: true mean is greater than 51
```

```
## 95 percent confidence interval:
##  50.7  Inf
## sample estimates:
## mean of x
##      51.9
```

c) The first test below is the sign (binomial) test. It assumes $H_0 : m = 50$ to be the median and if the H_0 is correct, then there should be approximately the same number of observations to the right and left of it. The Wilcoxon test also considers ranks to make stronger predictions. Comparing to **b)**, the t-tests require the data to be normally distributed which is unclear in our case. The two latter tests do not make such an assumption which may make them more applicable. On the other hand, they throw away more information, and in the case of the sign test, so much information has been lost that H_0 cannot be rejected, like it was with the t-test. The Wilcoxon test only assumes symmetrical distribution, which seems likely. And it, like the t-test, provides enough evidence to reject H_0

```
binom.test(sum(data$video>50), length(data$video), alternative='g')
```

```
##
##  Exact binomial test
##
## data:  sum(data$video > 50) and length(data$video)
## number of successes = 108, number of trials = 200, p-value = 0.1
## alternative hypothesis: true probability of success is greater than 0.5
## 95 percent confidence interval:
##  0.479 1.000
## sample estimates:
## probability of success
##                   0.54
```

```
wilcox.test(data$video, mu=50, alternative='g')
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  data$video
## V = 9836, p-value = 0.005
## alternative hypothesis: true location is greater than 50
```

Testing whether less than 25% of the data falls below 42 comes down to calculating the fraction of the data below 42 and checking if it is less than 0.25.

```
sum(data$video<42)/length(data$video) <= 0.25
```

```
## [1] TRUE
```

d) The bootstrap test allows us to check if a particular sample is expected from some distribution. By calculating some statistic T from many samples of some distribution F, we can estimate the likelihood of that statistic appearing in the distribution. Then, if that statistic of our sample $T(X_1, ..., X_N)$ is unexpected compared to the simulated statistics, we can conclude that our sample did not come from distribution F. This can been seen in the code below which tests 101 normal distributions with different means to test which ones were plausible to yield our sample.

```r
# Bootstrap test
B = 10000; t = min(data$video); Tstar = numeric(B); means = NULL
for(m in 0:100){
  for(i in 1:B)
    Tstar[i] = min(rnorm(length(data$video), mean=m, sd=10))
  if(2*min(sum(Tstar<t)/B, sum(Tstar>t)/B) > 0.05)
    means = c(means, m)
}
range(means)
```

```
## [1] 48 62
```

The Kolmogorov-Smirnov test can show whether two distributions are the same. It can estimate the distribution of our sample by computing the empirical distribution function and compare it to some distribution F. Therefore it is also applicable in this situation. We run it once to test the integer values of possible Gaussians for our data and then again to get a more accurate range.

```r
# Kolmogorov-Smirnov test, to roughly estimate the means which fit
means = NULL
for(m in 0:100){
  if(ks.test(data$video, pnorm, m, 10)[[2]] > 0.05)
    means = c(means, m)
}
range(means)
```

```
## [1] 52 53
```

```r
# More precise KS test
means = NULL
for(m in seq(51, 54, by=0.001)){
  if(ks.test(data$video, pnorm, m, 10)[[2]] > 0.05)
    means = c(means, m)
}
range(means)
```

```
## [1] 51.3 53.4
```

e)

Here we create separate vectors for male scores and female ones. Then we run the 3 tests on the data to check if male scores are indeed higher than female. All tests provide support for the expert's claim.

```r
fvideo = data$video[data$female == 1]; mvideo = data$video[data$female == 0]

t.test(mvideo, fvideo, alternative='g')
```

```
##
##  Welch Two Sample t-test
##
## data:  mvideo and fvideo
## t = 2, df = 177, p-value = 0.04
```

```
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.186   Inf
## sample estimates:
## mean of x mean of y
##      53.2      50.7
```

```
wilcox.test(mvideo, fvideo, alternative='g')
```

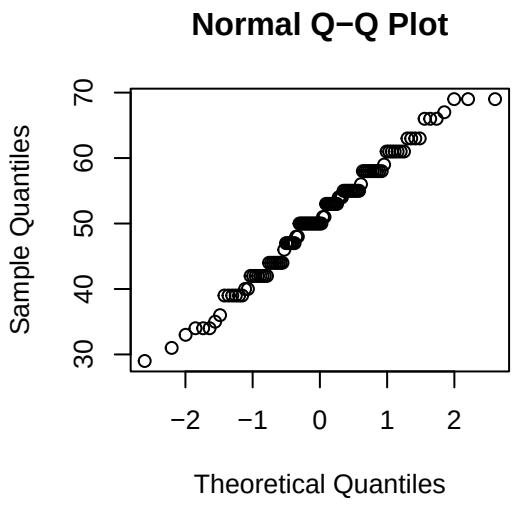
```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  mvideo and fvideo
## W = 5748, p-value = 0.03
## alternative hypothesis: true location shift is greater than 0
```

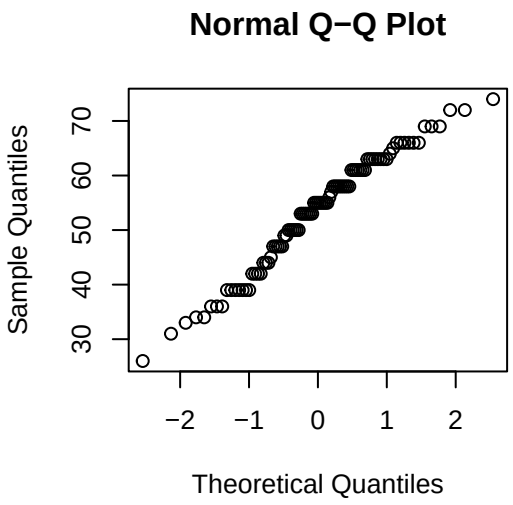
```
ks.test(mvideo, fvideo, alternative='l')
```

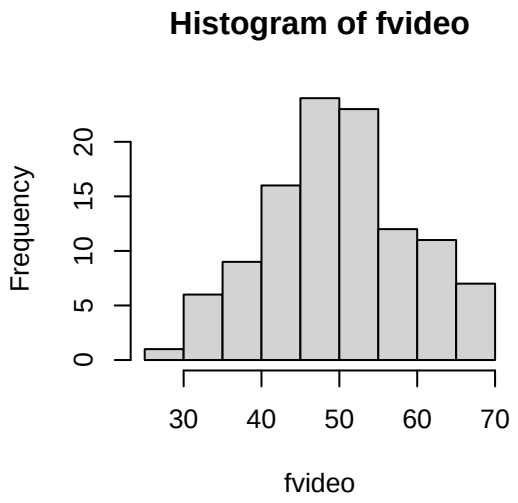
```
##
##  Exact two-sample Kolmogorov-Smirnov test
##
## data:  mvideo and fvideo
## D^- = 0.2, p-value = 0.04
## alternative hypothesis: the CDF of x lies below that of y
```

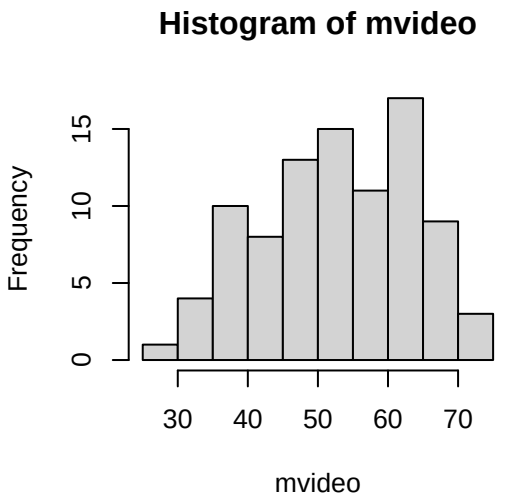
To confirm the tests' applicability, we need to check the distribution of data. For t-tests, it needs to follow a normal distribution. The histograms and qq plots, as well as the Shapiro-Wilk tests, likely confirm this. We do not show it here, but the boxplots also indicate normality. Therefore, the t-test is applicable. The Wilcoxon rank sum test only assumes symmetry which also seems likely. The Kolmogorov-Smirnov test does not assume a particular distribution and so is also applicable. The permutation test on the other hand is not applicable because these are independent samples.

```
par(mfrow=c(2, 2)); qqnorm(fvideo); qqnorm(mvideo); hist(fvideo); hist(mvideo)
```










```
shapiro.test(fvideo); shapiro.test(mvideo)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  fvideo
## W = 1, p-value = 0.3

##
##  Shapiro-Wilk normality test
##
## data:  mvideo
## W = 1, p-value = 0.06
```

f) Here we first apply Pearson's correlation test which rejects $H_0 : r = 0$. This test assumes normality so we tested for it as in **e)**, and found that the data is not normal. Then we ran the Spearman's correlation test,

which does not assume normality and found that the correlation was still there, meaning that $H_0 : r = 0$ can be rejected.

```
cor.test(data$video, data$puzzle)
```

```
##
##  Pearson's product-moment correlation
##
## data:  data$video and data$puzzle
## t = 7, df = 198, p-value = 4e-12
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.349 0.567
## sample estimates:
##   cor
## 0.465
```

```
cor.test(data$video, data$puzzle, method='spearman')
```

```
##
##  Spearman's rank correlation rho
##
## data:  data$video and data$puzzle
## S = 7e+05, p-value = 6e-13
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##   rho
## 0.481
```

To test the hypothesis that the score on puzzles is higher than on video games, we ran a paired t-test. this found no difference. Unfortunately, `data$puzzle - data$video` is not normally distributed so a different test is needed. We then tried the Wilcoxon paired test, but it similarly did not find a significant difference.

```
t.test(data$puzzle, data$video, paired=T, alternative='g')
```

```
##
##  Paired t-test
##
## data:  data$puzzle and data$video
## t = 0.7, df = 199, p-value = 0.2
## alternative hypothesis: true mean difference is greater than 0
## 95 percent confidence interval:
##  -0.695    Inf
## sample estimates:
## mean difference
##           0.555
```

```
wilcox.test(data$puzzle, data$video, paired=T, alternative='g')
```

```
##
##  Wilcoxon signed rank test with continuity correction
```

```
##
## data:  data$puzzle and data$video
## V = 10292, p-value = 0.07
## alternative hypothesis: true location shift is greater than 0
```

Exercise 2

a) We create a factorial design and distribute N random fishes to each combination of the factors.

```
I=4; J=2; N=10
data.frame(fish=sample(1:(N*I*J)), rate=rep(1:I, each=N*J), method=rep(1:J, N*I))
```

b) Here we first see that there is no interaction between the two factors. Next, we see that **method** does not have a significant impact on hemoglobin levels at all, while **rate** does.

```
anova(lm(hemoglobin~rate*method, data=data))
```

```
## Analysis of Variance Table
##
## Response: hemoglobin
##             Df Sum Sq Mean Sq F value  Pr(>F)
## rate         3   90.6   30.19   19.47 2.4e-09 ***
## method       1    2.4    2.42    1.56    0.22
## rate:method  3    4.9    1.62    1.05    0.38
## Residuals   72  111.6    1.55
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

c) This is not a particularly good question with the previous ANOVA test because it is unclear how a factor may influence the dependent variable through the interaction. Once we remove the interaction and consider only the additive model, we see that indeed **rate** has the greater impact and that **method** is not significant at all. Rate 2 leads to the significantly highest hemoglobin levels. Method B may be a little larger than method A but this cannot be ascertained with adequate statistical power. Therefore the highest mean hemoglobin combination is with rate 2 and probably method B.

```
anova(lm(hemoglobin~rate+method, data=data))
```

```
## Analysis of Variance Table
##
## Response: hemoglobin
##           Df Sum Sq Mean Sq F value Pr(>F)
## rate       3   90.6   30.19   19.43  2e-09 ***
## method     1    2.4    2.42    1.55   0.22
## Residuals 75  116.5    1.55
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(lm(hemoglobin~rate+method, data=data))
```

```
##
```

```
## Call:
## lm(formula = hemoglobin ~ rate + method, data = data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -3.454 -0.888  0.005  0.841  2.339
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    6.801      0.312   21.83  < 2e-16 ***
## rate2          2.760      0.394    7.00  9.2e-10 ***
## rate3          2.405      0.394    6.10  4.2e-08 ***
## rate4          1.880      0.394    4.77  8.9e-06 ***
## methodB        0.348      0.279    1.25     0.22
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.25 on 75 degrees of freedom
## Multiple R-squared:  0.444,  Adjusted R-squared:  0.414
## F-statistic:   15 on 4 and 75 DF,  p-value: 4.92e-09
```

Below can be seen the mean hemoglobin when using rate 3 and method A. Also, as seen before, rate 2 leads to the highest hemoglobin values.

```
mean(data$hemoglobin[data$rate==3 & data$method=='A'])
```

```
## [1] 9.03
```

```
which.max(aggregate(hemoglobin ~ rate, data=data, mean)$hemoglobin)
```

```
## [1] 2
```

d) The one-way ANOVA test shows that there is a difference in some means among the **rate** factor levels. In this dataset and after our initial analysis, we suspect that **method** does not influence the hemoglobin. Therefore, a block design is not warranted, and the one-way ANOVA is no more wrong or less useful than the two-way ANOVA.

```
hemoaov = lm(hemoglobin ~ rate, data=data); anova(hemoaov)
```

```
## Analysis of Variance Table
##
## Response: hemoglobin
##           Df Sum Sq Mean Sq F value  Pr(>F)
## rate       3   90.6   30.19    19.3 2.1e-09 ***
## Residuals 76  118.9    1.56
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Point estimates for hemoglobin values for each rate
aggregate(hemoglobin ~ rate, data=data, mean)
```

```
##   rate hemoglobin
## 1    1       6.97
## 2    2       9.73
## 3    3       9.38
## 4    4       8.86
```

e) ANOVA makes several assumptions while the Kruskal-Wallis is the generalization of the Wilcoxon test. ANOVA assumes normally distributed data, equal variance for all levels of all factors and their combinations, and that the residuals follow a normal distribution. With the small sample size, it is hard to know whether these conditions are met. On the other hand, the non-parametric Kruskal-Wallis test can be employed regardless. In our case, it similarly rejects H_0.

```
kruskal.test(hemoglobin ~ rate, data=data)
```

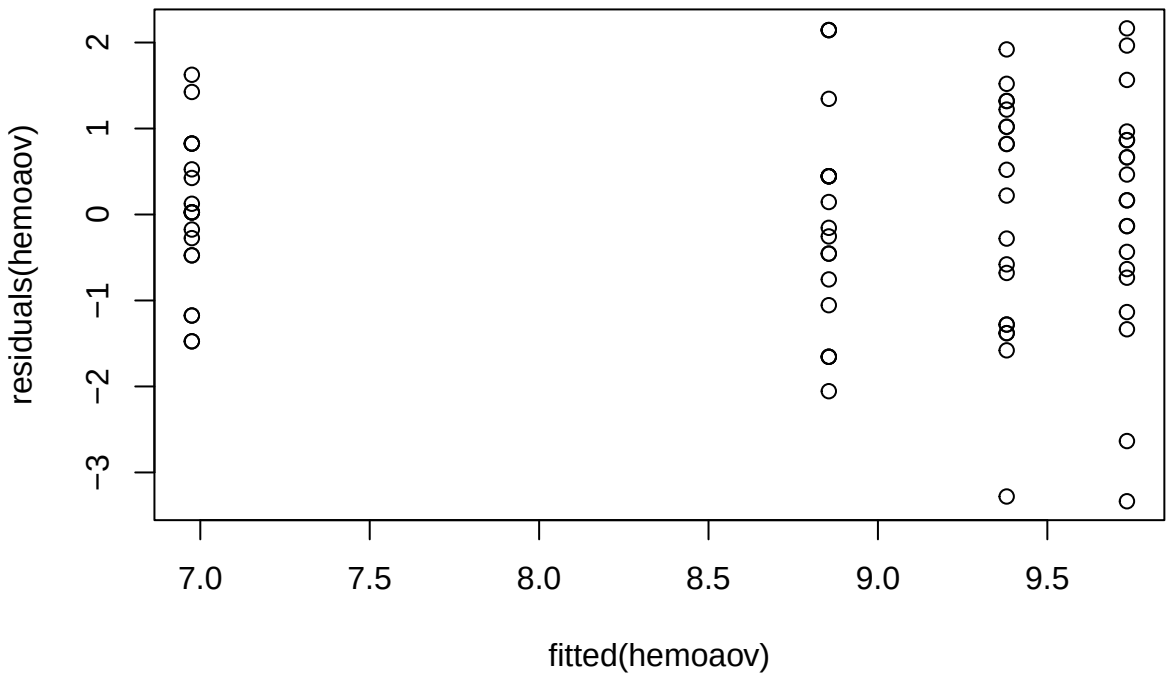
```
##
##  Kruskal-Wallis rank sum test
##
## data:  hemoglobin by rate
## Kruskal-Wallis chi-squared = 34, df = 3, p-value = 2e-07
```

However, there are several things we can do to check if using ANOVA is appropriate before the need for the Kruskal-Wallis test. The first is to check if the variance is identical. It us unclear if the variance is identical here. We can also check the residuals. Plotting the residuals against the fitted parameters should yield no structure. In our case ANOVA is probably fine but if we need to be completely certain, the Kruskal-Wallis may be preferable.

```
# Checking the variance
aggregate(hemoglobin ~ rate:method, data=data, sd)
```

```
##   rate method hemoglobin
## 1    1      A      1.019
## 2    2      A      1.717
## 3    3      A      1.135
## 4    4      A      1.000
## 5    1      B      0.707
## 6    2      B      0.887
## 7    3      B      1.559
## 8    4      B      1.553
```

```
# Checking the residuals
par(mfrow=c(1, 1)); plot(fitted(hemoaov), residuals(hemoaov))
```



Exercise 3

a) We ran an additive ANOVA model and found that the **starter** indeed have a strong effect on the **acidity**. Out of the two blocks, the **position** had no effect but the **batch** did.

The following summary shows that there is no statistically significant difference in effects of starters 1 and 2. The only starter that seems to deviate from the rest and yield significantly higher acidity is starter 4.

```
aciditylm = lm(acidity ~ batch + position + starter, data); summary(aciditylm)
```

```
##
## Call:
## lm(formula = acidity ~ batch + position + starter, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.2836 -0.2336  0.0384  0.3584  1.0204
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    8.662      0.533   16.26  1.5e-09 ***
## batch2        -1.348      0.467   -2.88    0.014 *
## batch3         0.276      0.467    0.59    0.566
## batch4         1.368      0.467    2.93    0.013 *
## batch5         0.200      0.467    0.43    0.676
```

```
## position2     -0.618      0.467   -1.32    0.211
## position3     -0.038      0.467   -0.08    0.937
## position4     -0.764      0.467   -1.63    0.128
## position5     -0.264      0.467   -0.56    0.583
## starter2      -0.150      0.467   -0.32    0.754
## starter3      -0.980      0.467   -2.10    0.058 .
## starter4       2.810      0.467    6.01  6.1e-05 ***
## starter5      -0.484      0.467   -1.04    0.321
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.739 on 12 degrees of freedom
## Multiple R-squared:  0.909,  Adjusted R-squared:  0.818
## F-statistic: 9.96 on 12 and 12 DF,  p-value: 0.000178
```

b) Since position had no effect on acidity, we removed it. Our findings remain identical as in **a)**. Here we see that starter 4 has the biggest effect on **acidity**. From the summary, we can see that α_4 is the only one that is positive, and the only one that differs significantly from α_1. This shows not only that starter 4 has the strongest positive effect on the acidity of the sour cream, but also that the reference starter, starter 1, has the second strongest, though insignificantly different from the rest of the lower acidity starters.

```
aciditylm = lm(acidity ~ starter + batch, data); anova(aciditylm)
```

```
## Analysis of Variance Table
##
## Response: acidity
##           Df Sum Sq Mean Sq F value  Pr(>F)
## starter    4   44.1   11.03   19.84 4.8e-06 ***
## batch      4   18.8    4.69    8.44 0.00073 ***
## Residuals 16    8.9    0.56
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(aciditylm)
```

```
##
## Call:
## lm(formula = acidity ~ starter + batch, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.5648 -0.2548 -0.0548  0.3592  1.1352
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    8.325      0.447   18.60  2.9e-12 ***
## starter2      -0.150      0.472   -0.32    0.755
## starter3      -0.980      0.472   -2.08    0.054 .
## starter4       2.810      0.472    5.96  2.0e-05 ***
## starter5      -0.484      0.472   -1.03    0.320
## batch2        -1.348      0.472   -2.86    0.011 *
## batch3         0.276      0.472    0.59    0.567
```

```
## batch4         1.368      0.472    2.90    0.010 *
## batch5         0.200      0.472    0.42    0.677
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.746 on 16 degrees of freedom
## Multiple R-squared:  0.876,  Adjusted R-squared:  0.814
## F-statistic: 14.1 on 8 and 16 DF,  p-value: 6.47e-06
```

c) Friedman test is also applicable here, as it is similar to 2-way ANOVA but does not assume normality in the data distribution. Therefore, it can be seen like a non-parametric counterpart to 2-way ANOVA. Here it leads to the same result that the starter does indeed have an effect on the acidity.

```
friedman.test(acidity ~ starter | batch, data)
```

```
##
##  Friedman rank sum test
##
## data:  acidity and starter and batch
## Friedman chi-squared = 13, df = 4, p-value = 0.01
```

d) We fitted two linear mixed effects models on our data. One with the **starter** factor as a fixed effect and one without it. Then we ran an ANOVA to see if the 2 models were different, and indeed they were, as the previous experiments would have suggested. Looking into `summary(lm1)` (not shown here), we found identical alphas for the starter, but we also saw how much variance the random effect factors explained. Finally we plotted the residuals of the fitted model to ensure that ANOVA was appropriate here.

```
lm1 = lmer(acidity ~ starter + (1|batch) + (1|position), data)
lm2 = lmer(acidity ~ (1|batch) + (1|position), data)
```

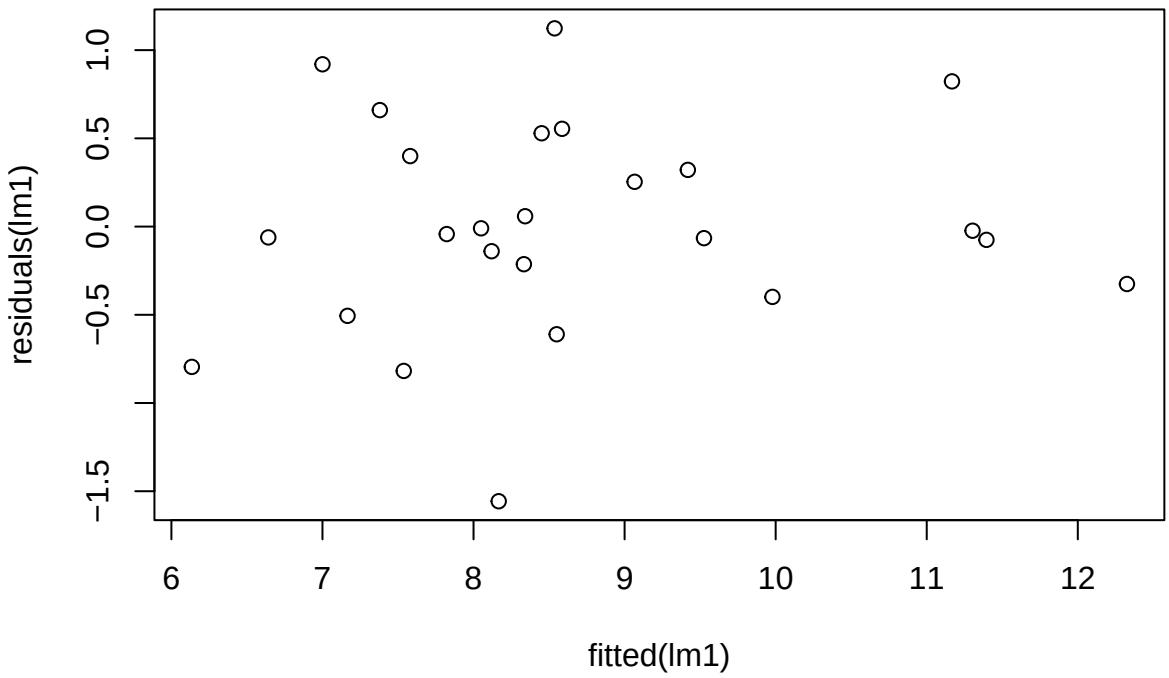
```
## boundary (singular) fit: see help('isSingular')
```

```
anova(lm1, lm2)
```

```
## refitting model(s) with ML (instead of REML)

## Data: data
## Models:
## lm2: acidity ~ (1 | batch) + (1 | position)
## lm1: acidity ~ starter + (1 | batch) + (1 | position)
##     npar   AIC   BIC logLik deviance Chisq Df Pr(>Chisq)
## lm2    4 105.1 109.9  -48.5     97.1
## lm1    8  77.2  86.9  -30.6     61.2  35.9  4    3.1e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(fitted(lm1), residuals(lm1))
```


residuals(lm1)
1.0
0.5
0.0
−0.5
−1.5
6
7
8
9
10
11
12
fitted(lm1)