Grapevine

Naitong Chen, Shirley Cui, Shannon Edie

Data description

Chardonnay

There are 40 samples in each of the chardonnay2019 and chardonnay2020 data set. The 40 samples consist of 8 samples from each of the 5 blocks. Among the 8 samples within each block, 4 samples of each treatment are included.

Both datasets contain 3 measurements of each sample with no missing data.

```
## [1] ï..sample.ID           row                    treatment
## [4] block                  subsample              X..of.clusters
## [7] yield..kg.             av.cluster.weight..kg.
## <0 rows> (or 0-length row.names)
```

Merlot

There are 60 samples in each of the merlot2019 and merlot2020 data set. The 60 samples consist of 10 samples from each of the 6 blocks. Among the 10 samples within each block, 5 samples of each treatment are included.

There are 10 measurements that are common between the two merlot data sets. In addition, merlot2020 contains measures of fruitfulness, bloom, and brown seed color. merlot2019 contains measures of pruning weight and ravaz index.

In the merlot2019 data set, there are missing values from the measurement of veraison. In the merlot2020 data set, there are missing vales from the measurements of veraison, bloom, yield, cluster weight, berries/cluster, berry weight, berry Brix, berry TA, berry pH, and brown seed color.

```
##  [1] ï..Key          block           treatment       row
##  [5] subsample       SPAD            veraison        cluster.number
##  [9] yield           cluster.weight  berries.cluster berry.weight
## [13] berry.TA        berry.pH        berry.Brix      pruning.weight
## [17] Ravaz.index
## <0 rows> (or 0-length row.names)
```

```
##  [1] ï..Key           block            treatment        row
##  [5] subsample        fruitfulness     bloom            spad
##  [9] veraison         cluster.number   yield            cluster.weight
## [13] berries.cluster  berry.weight     berry.Brix       berry.TA
## [17] berry.pH         brown.seed.color
## <0 rows> (or 0-length row.names)
```

Missing data in merlot2019 and merlot2020

Veraison

We know from the client that the veraison column indicates the number days into August in each respective year when 50% of the berries on a given vine had changed to red. The missing entries are a result of the vines reaching 50% veraison on the first day of measurement. They are missing not at random (MNAR) as whether this value is missing for each vine is dependent on this particular value. More specifically, if any of the missing values were recorded, it must be negative.

While imputation seems like a reasonable approach, since we know these values must be negative, we can take the maximum between each imputed value and zero to produce complete data sets?

Percentage of missing for the veraison measure of merlot2019 and merlot2020 are shown below.

```
## [1] 0.4333333 0.1333333
```

Bloom

Similar to veraison, the bloom column includes values that indicate the number of days into June in each respective year when 50% of the flowers on a given vine had started to bloom. The missing entries are a result of the vines reaching 50% bloom on the first day of measurement. They are also missing not at random (MNAR).

Again, we can consider taking the maximum between each imputed value and zero to produce complete data sets.

Percentage of missing for the bloom measure of merlot2020 is shown below.

```
## [1] 0.1166667
```

Yield related measures

For all columns that contain missing values (yield, cluster weight, berries/cluster, berry weight), the reason for missing is that the vine had no yield. Even though these missing values depend on the measures of cluster number of their corresponding vines being zero, these yield related values are missing precisely because there are no yield. This is then also a case of missing not at random (MNAR).

Assigning these missing entries the value of zero seems plausible.

Percentage of missing for the yield related measures of merlot2020 is shown below.

```
## [1] 0.1166667
```

Berry related measures

Similarly, for all columns that contain missing values (berry Brix, berry TA, berry pH, brown seed color), the reason for missing is that the vine had no yield. These values are missing simply because there were no grapes produced to have these measurements recorded. Depending on the relationship between berry quality and yield, this is either a case of missing at random (MAR, in the absence of an association between berry quality and yield) or missing completely at random (MCAR, in the presence of an association between berry quality and yield).

Due to the structure of the study, particularly the existence of blocking and replication, imputation seems to be the most reasonable approach to fill in the missing values.

Percentage of missing for the berry related measures of merlot2020 is shown below.

```
## [1] 0.1333333
```

Question

In the merlot2020 data set, most of the vines that had no entries for bloom or veraison (implying earlier ripening) also had no yield. This is against my intuition. If anything, as observed in the other three data sets, early ripening may be associated with higher yield? What was the weather event that caused some of the vines to have no yield?

Constructing complete data sets

We construct complete data sets using the approaches described above. Once these complete data sets are constructed, analysis can be done separately on each of the imputed data sets. Although we need to look into the necessity and methods to aggregate these results. We should also look into the order or columns to impute, as well as whether all responses should be used to predict the missing values.

For now, I have imputed with and without the yield related measures. We may consider more variants such as using only responses that are highly correlated.

Linear regerssions are used for imputation. There are a lot more available options specified on page 76 of https://cran.r-project.org/web/packages/mice/mice.pdf.

Note that the the columns of key, row and subsample are excluded from all imputations below.

merlot2019

The imputed values of veraison for merlot2019 from the first imputed data set are shown below. Note they are all above zero, they will all be filled in with zero before the analysis.

```
##  [1] 21.715994 18.125289 16.773634 14.942121 19.471914 21.589641 21.486780
##  [8] 15.115459  8.349683 18.339226 20.047271 13.725877 21.528018  7.691104
## [15] 18.035914 20.088900 22.080232 11.933042 13.388718  2.451018  8.835080
## [22] 24.841542 25.660091 20.944922  9.503893 11.462480
```

merlot2020

The first set of imputed values for merlot2020 using all responses are shown below.

```
## [1] "veraison"

## [1] 30.67912 30.34631 31.56788 30.22332 25.51688 28.10610 27.17114 28.72719

## [1] "berry Brix"

## [1] 27.28276 27.34771 25.86445 27.62986 28.48672 28.04230 27.84948 28.06373

## [1] "berry pH"

## [1] 3.519315 3.523890 3.232155 3.496757 3.630889 3.655843 3.511203 3.484332

## [1] "berry TA"

## [1] 6.349962 6.422805 8.139608 6.498603 5.441846 4.693712 6.532854 6.638227

## [1] "brown seed color"
## [1] 63.62264 63.88353 75.09464 64.83935 72.78476 73.82176 68.46240 68.89437
```

The first set of imputed values for merlot2020 not using yield related measures are shown below.

```
## [1] "veraison"
```

```
## [1] 32.35104 31.94097 31.91933 31.39454 27.36092 29.21802 25.75810 30.42609
```

```
## [1] "berry Brix"
```

```
## [1] 26.22747 26.31896 25.91935 26.55039 27.50730 26.92197 26.82877 27.06777
```

```
## [1] "berry pH"
```

```
## [1] 3.290631 3.298345 3.231982 3.272862 3.408987 3.421944 3.281947 3.258690
```

```
## [1] "berry TA"
```

```
## [1] 7.357714 7.404429 8.037524 7.580077 6.348228 5.812225 7.558876 7.710729
```

```
## [1] "brown seed color"
```

```
## [1] 79.30215 79.52354 76.95949 78.53604 89.29272 88.31270 84.29497 83.44413
```

Again the veraison values are all above zero, and so zero values will assigned before conducting the analysis.

We therefore won't need to do imputation on veraison altogether, and so can proceed for now using the following three sets of data (merlot2019, merlot2020 imputed with filled-in zeros, merlot2020 imputed without filled-in zeros).

Check out the imputed data sets in the code blocks!

data preprocessing - PCA

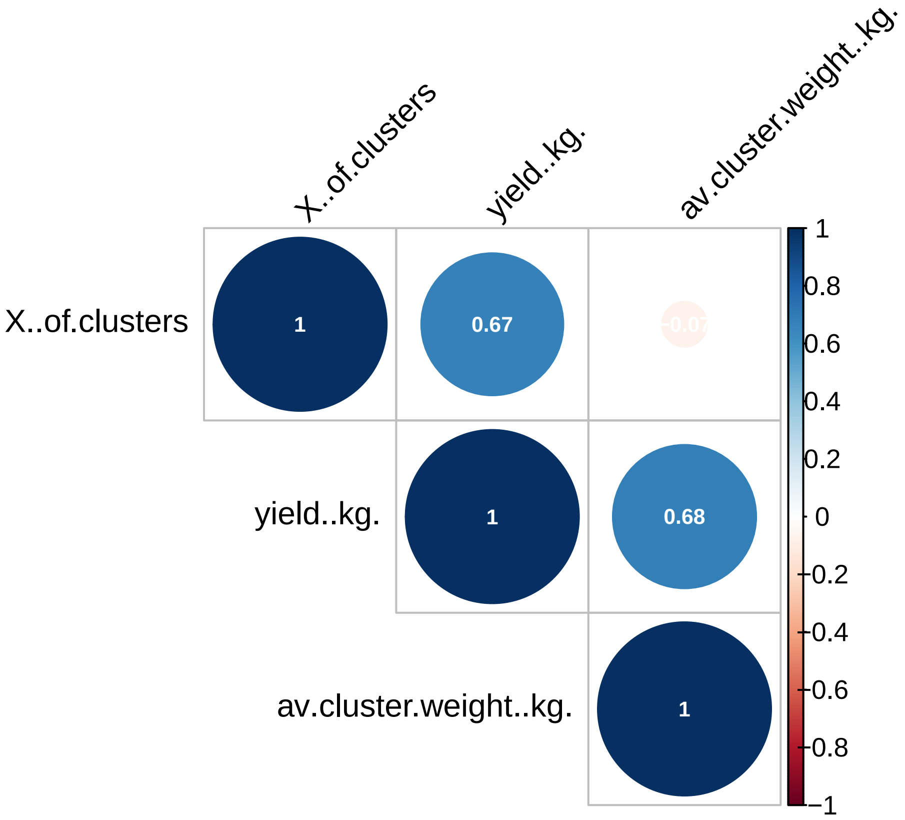
Basically, we have reduced the dimentions of the dataset down to four/three dimensions for merlot 2013/2014 using PCA.

Check for correlations

Chardonnay dataset

chardonnay 2019

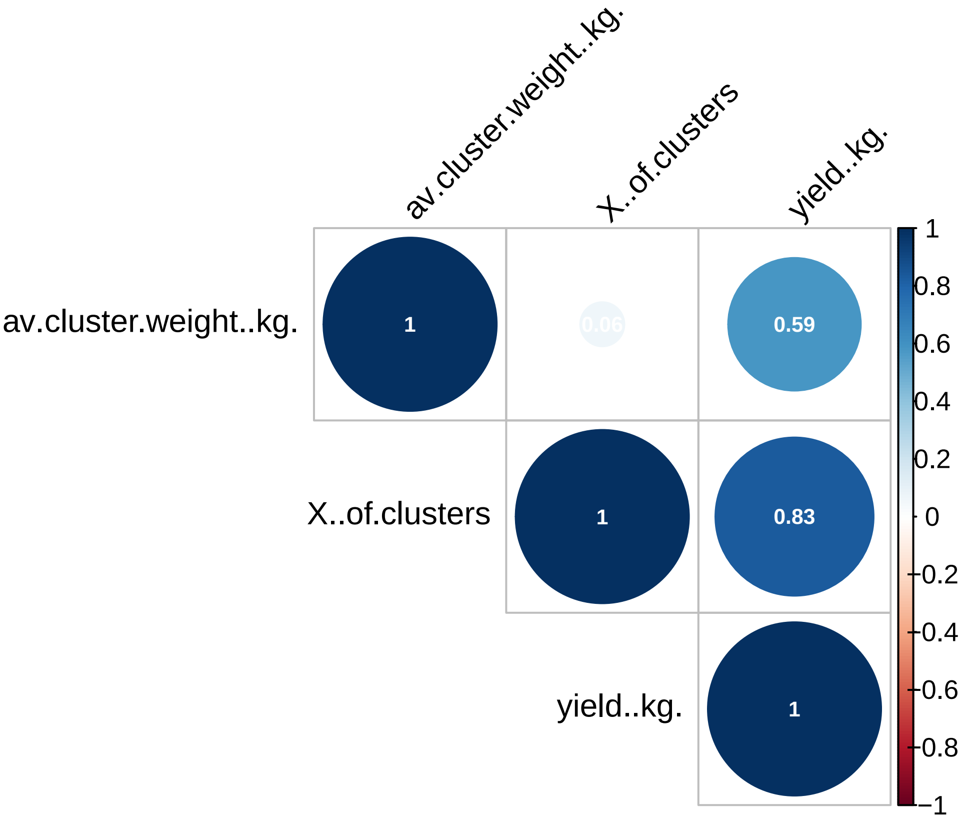
```
chardonnay2019 %>%
  select(-1:-5) %>%
  cor() %>%
  corrplot(type = "upper", order = "hclust", tl.col = "black", tl.srt = 45, addCoef.col="white", number
```



From the output of correlation function, there seem no correlations between response variables.

chardonnay 2020

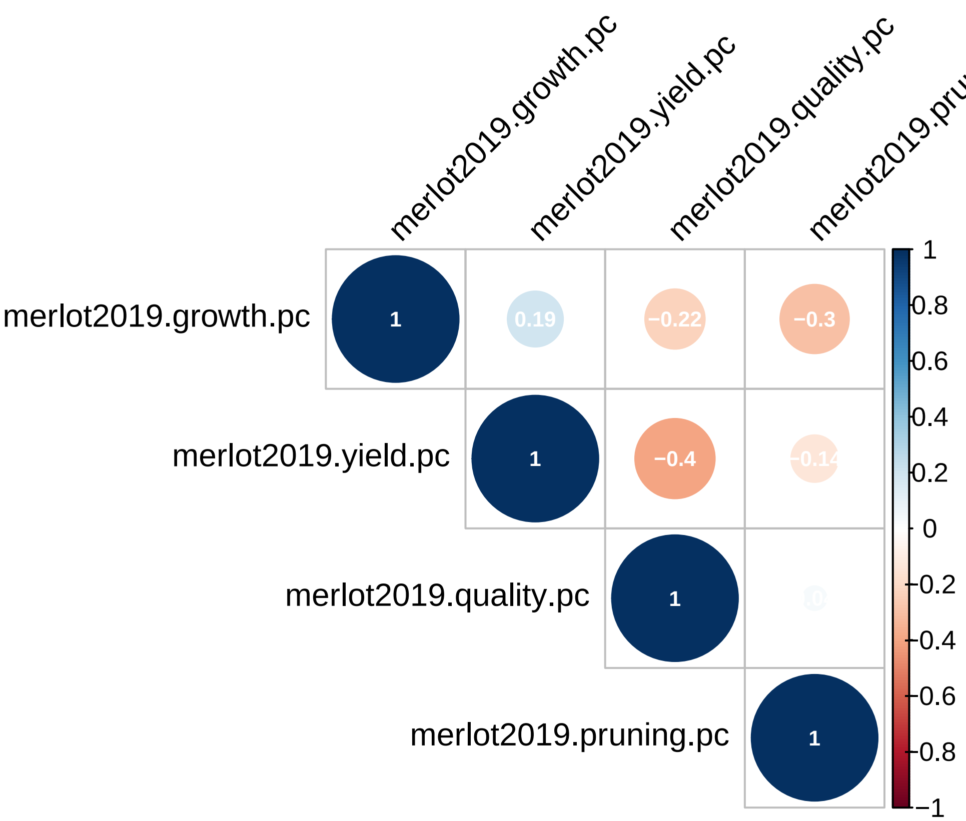
```
chardonnay2020 %>%
  select(-1:-5) %>%
  cor() %>%
  corrplot(type = "upper", order = "hclust", tl.col = "black", tl.srt = 45, addCoef.col="white", number
```



Yield is found to be positively correlated with number of clusters.

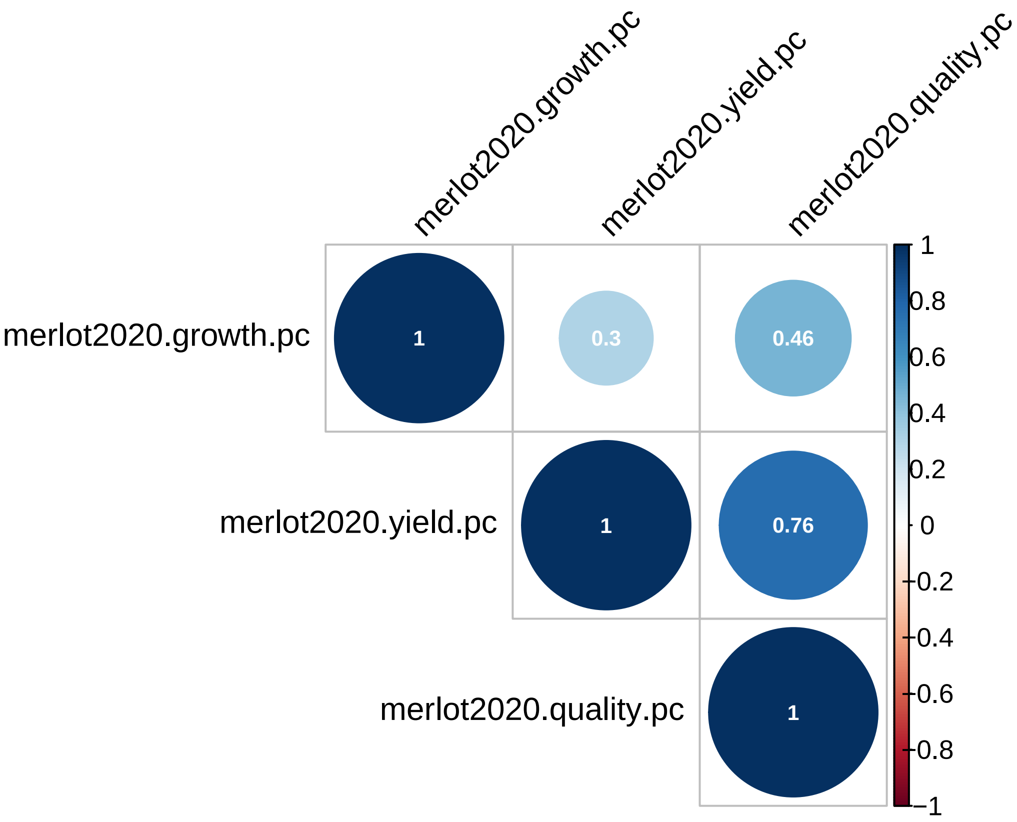
merlot 2019

```
merlot2019.pca %>%
  select(-1:-5) %>%
  cor() %>%
  corrplot(type = "upper", order = "hclust", tl.col = "black", tl.srt = 45, addCoef.col="white", number
```



### merlot 2020

```
merlot2020.pca %>%
  select(-1:-5) %>%
  cor() %>%
  corrplot(type = "upper", order = "hclust", tl.col = "black", tl.srt = 45, addCoef.col="white", number
```



From the correlation matrix plots, there seem to be no obvious correlations among the response variables in these four datasets. We are good to continue with normality and equal variance assumptions check.

Check for assumption of ANOVA test: normality and equal variance

Chardonnay 2019

1. table of mean and variance under control and treatment for each response variable

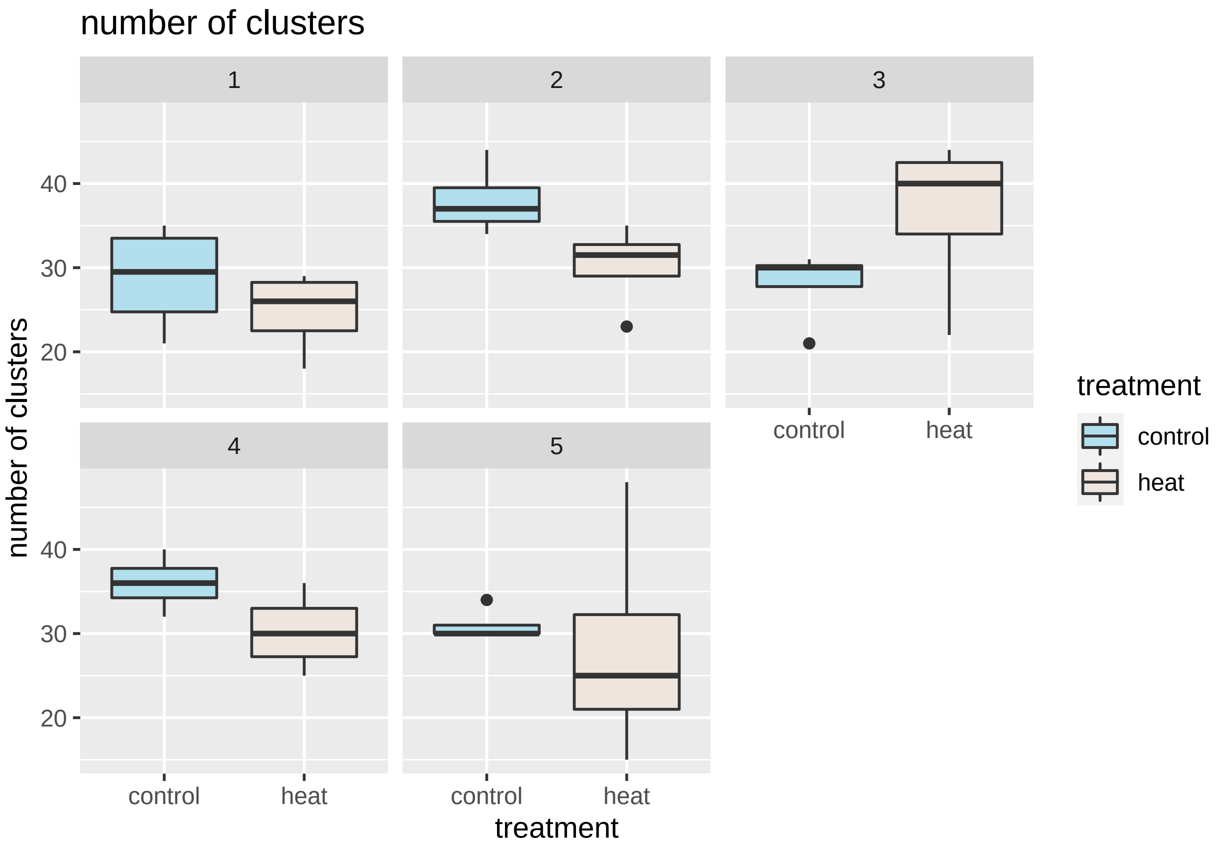
```
chardonnay2019 %>%
  group_by(block, treatment) %>%
  summarise(var.num_of_cluster=var(X..of.clusters),
            mean.num_of_cluster=mean(X..of.clusters),
            var.yield=var(yield..kg.),
            mean.yield=mean(yield..kg.),
            var.cluster_weight=var(av.cluster.weight..kg.),
            mean.cluster_weight=mean(av.cluster.weight..kg.))
```

```
## `summarise()` has grouped output by 'block'. You can override using the `.groups` argument.
```

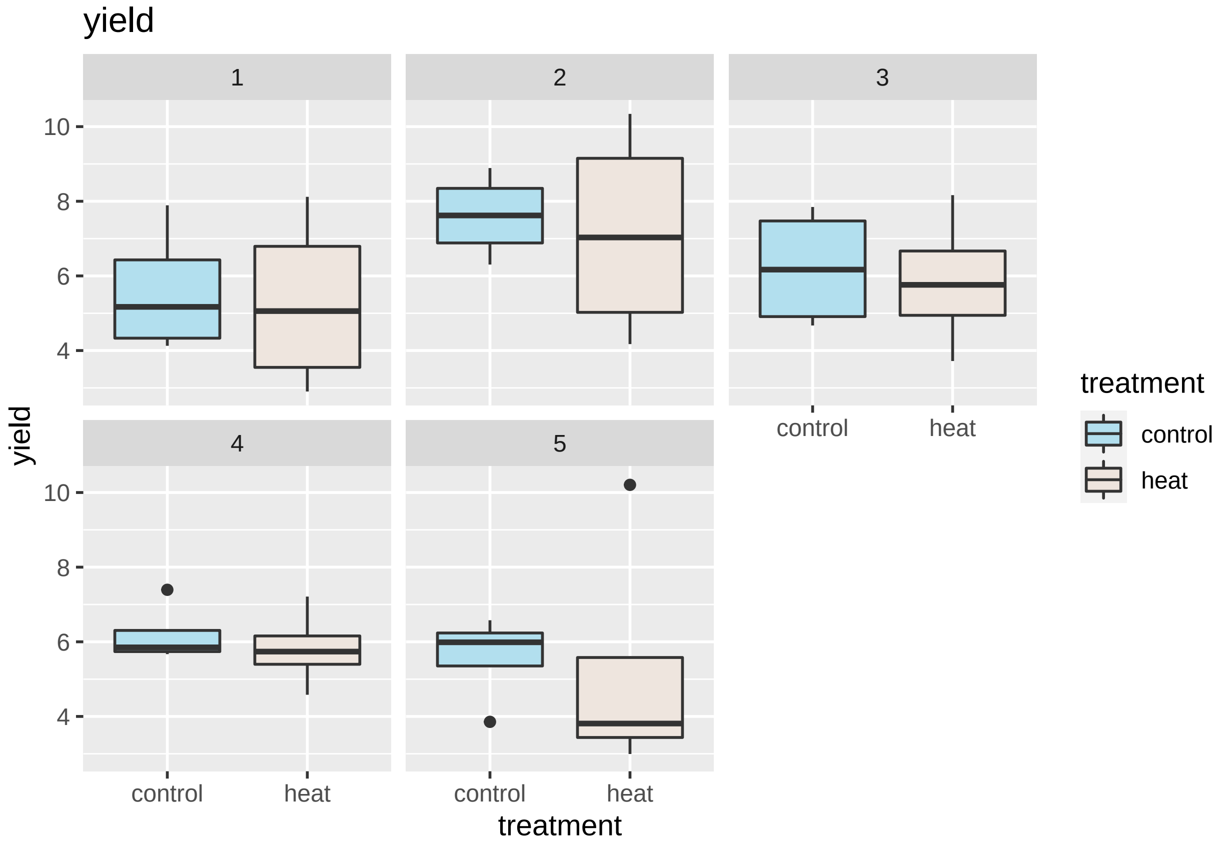
```
## # A tibble: 10 x 8
## # Groups:   block [5]
##    block treatment var.num_of_clus~ mean.num_of_clu~ var.yield mean.yield
##    <fct> <fct>                <dbl>            <dbl>     <dbl>      <dbl>
##  1 1     control               41.6             28.8      2.99       5.59
##  2 1     heat                  24.9             24.8      5.72       5.28
##  3 2     control               18.7             38        1.31       7.61
##   4 2     heat                    26.2             30.2      8.34        7.14
##   5 3     control                 22               28        2.61        6.21
##   6 3     heat                    99.7             36.5      3.42        5.85
##   7 4     control                 11.3             36        0.655       6.19
##   8 4     heat                    22.9             30.2      1.17        5.82
##   9 5     control                  4               31        1.45        5.60
##  10 5     heat                   198.              28.2     11.3         5.20
## # ... with 2 more variables: var.cluster_weight <dbl>,
## #   mean.cluster_weight <dbl>
```

2. Visualization (side by side boxplots for each response variable)

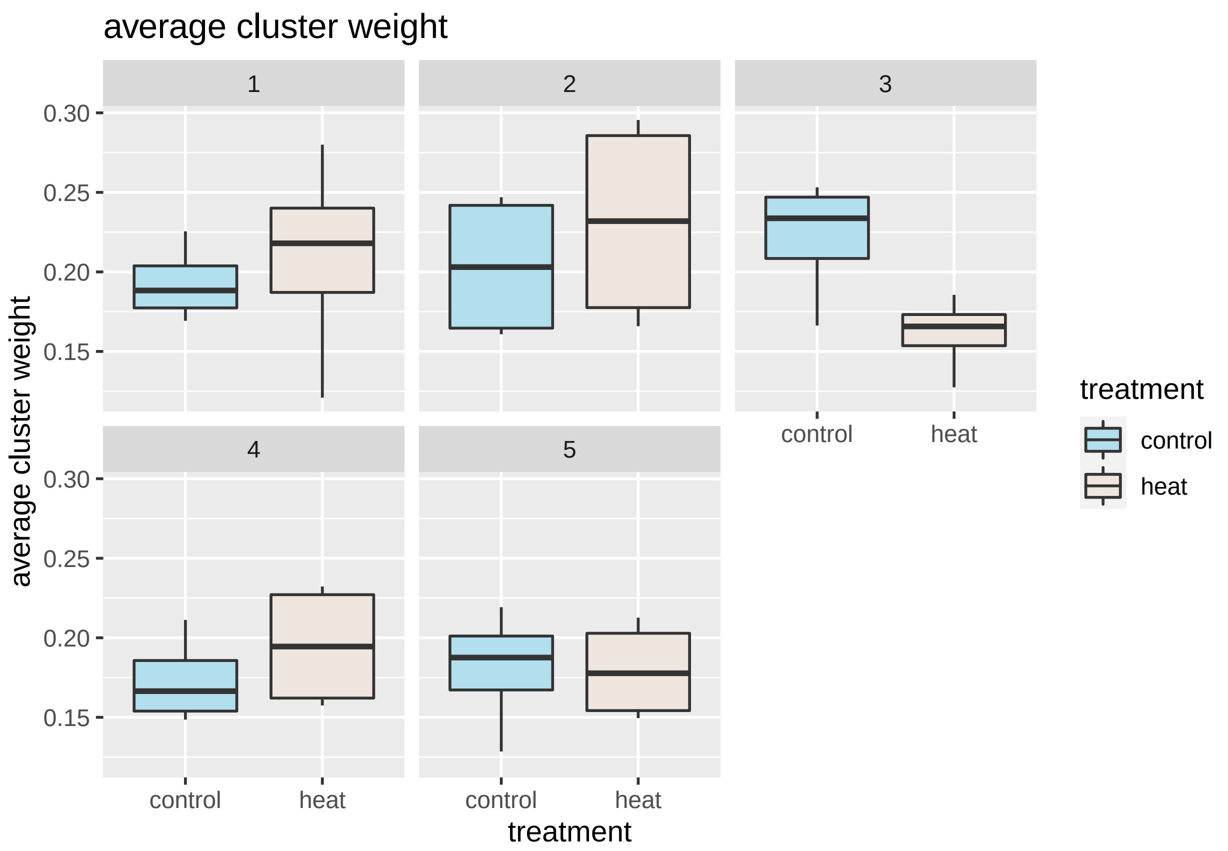
```
## check assumptions of number of clusters
chardonnay2019 %>%
  ggplot(aes(x=treatment, y = X..of.clusters, fill=treatment))+
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "number of clusters") +
  ylab("number of clusters")+
  facet_wrap(~block)
```



```
## check assumptions of yield
chardonnay2019 %>%
  ggplot(aes(x=treatment, y = yield..kg., fill=treatment)) +
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "yield") +
  ylab("yield")+
  facet_wrap(~block)
```



```
## check assumptions for average cluster weight
chardonnay2019 %>%
  ggplot(aes(x=treatment, y = av.cluster.weight..kg., fill=treatment)) +
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "average cluster weight") +
  ylab("average cluster weight")+
  facet_wrap(~block)
```



### Chardonnay 2020 1. table of mean and variance under control and treatment for each response variable

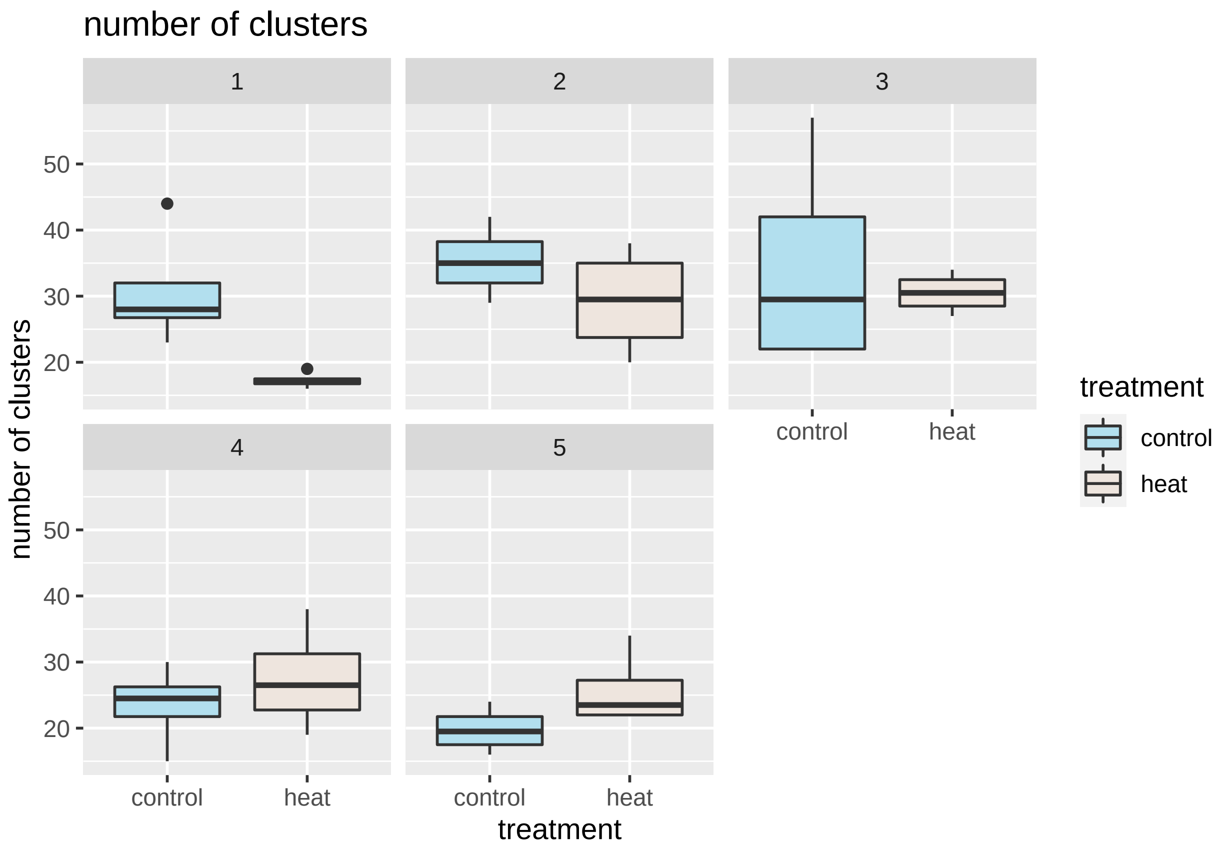
```
chardonnay2020 %>%
  group_by(block, treatment) %>%
  summarise(var.num_of_cluster=var(X..of.clusters),
            mean.num_of_cluster=mean(X..of.clusters),
            var.yield=var(yield..kg.),
            mean.yield=mean(yield..kg.),
            var.cluster_weight=var(av.cluster.weight..kg.),
            mean.cluster_weight=mean(av.cluster.weight..kg.))
```

```
## `summarise()` has grouped output by 'block'. You can override using the `.groups` argument.
```

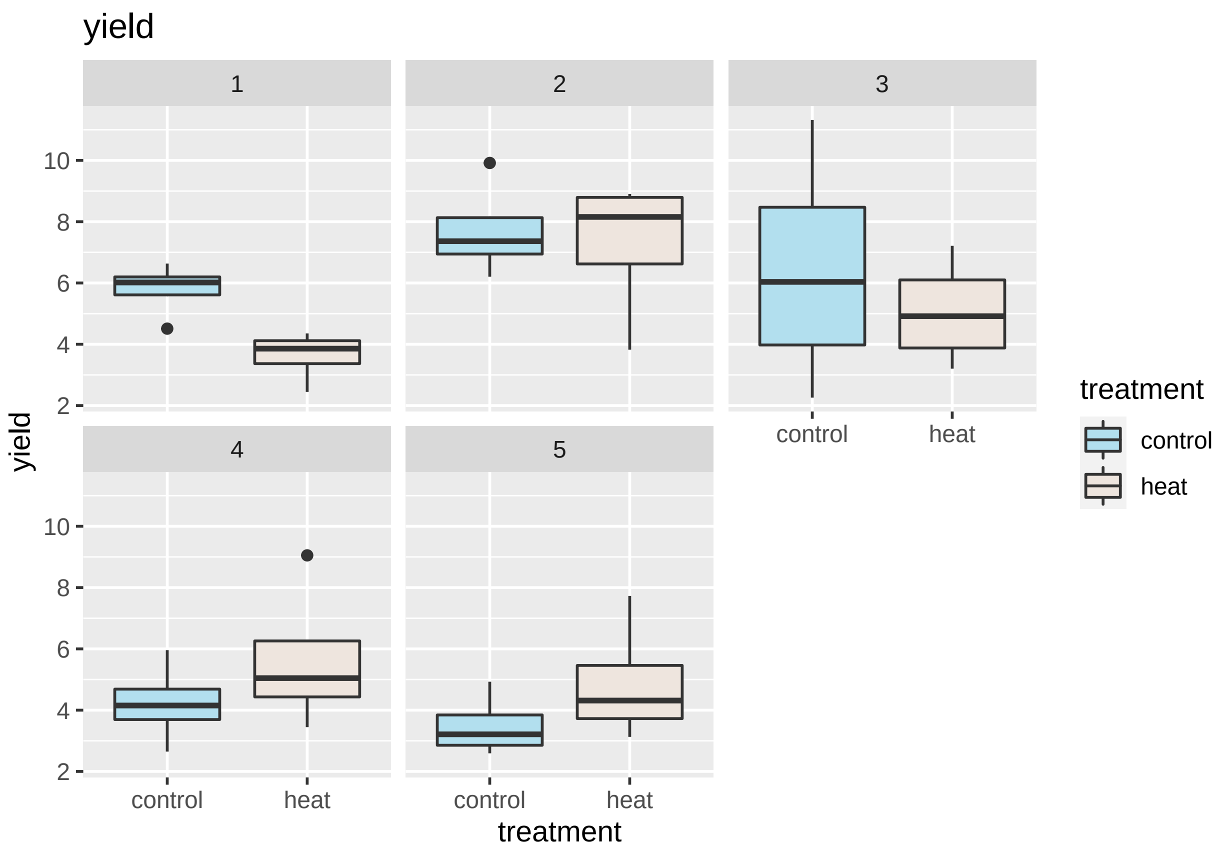
```
## # A tibble: 10 x 8
## # Groups:   block [5]
##    block treatment var.num_of_clus~ mean.num_of_clu~ var.yield mean.yield
##    <fct> <fct>                <dbl>            <dbl>     <dbl>      <dbl>
##  1 1     control              83.6              30.8     0.818       5.79
##  2 1     heat                  1.58             17.2     0.699       3.63
##  3 2     control              30.9              35.2     2.48        7.71
##  4 2     heat                 67.6              29.2     5.61        7.26
##  5 3     control             275                34.5    15.3         6.41
##  6 3     heat                  9.67             30.5     3.15        5.06
##  7 4     control              39                23.5     1.84        4.23
##  8 4     heat                 65.7              27.5     5.77        5.65
##  9 5     control              12.2              19.8     1.06        3.49
## 10 5     heat                 32.2              25.8     4.04        4.87
## # ... with 2 more variables: var.cluster_weight <dbl>,
## #   mean.cluster_weight <dbl>
```

2. Visualization (side by side boxplots for each response variable)

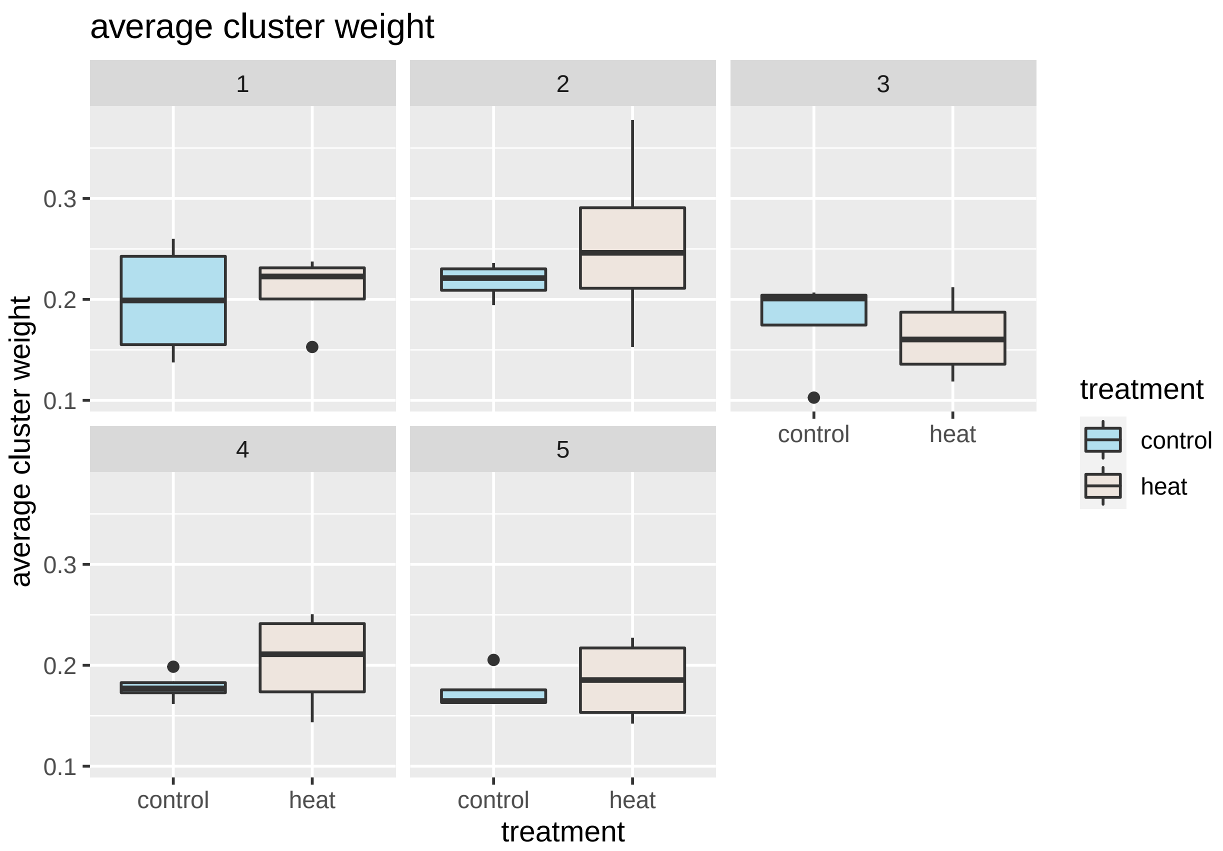
```
## check assumptions of number of clusters
chardonnay2020 %>%
  ggplot(aes(x=treatment, y = X..of.clusters, fill=treatment))+
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "number of clusters") +
  ylab("number of clusters")+
  facet_wrap(~block)
```



```
## check assumptions of yield
chardonnay2020 %>%
  ggplot(aes(x=treatment, y = yield..kg., fill=treatment)) +
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "yield") +
  ylab("yield")+
  facet_wrap(~block)
```



```
## check assumptions for average cluster weight
chardonnay2020 %>%
  ggplot(aes(x=treatment, y = av.cluster.weight..kg., fill=treatment)) +
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "average cluster weight") +
  ylab("average cluster weight")+
  facet_wrap(~block)
```



### merlot 2019 1. table of mean and variance under control and treatment for each response variable

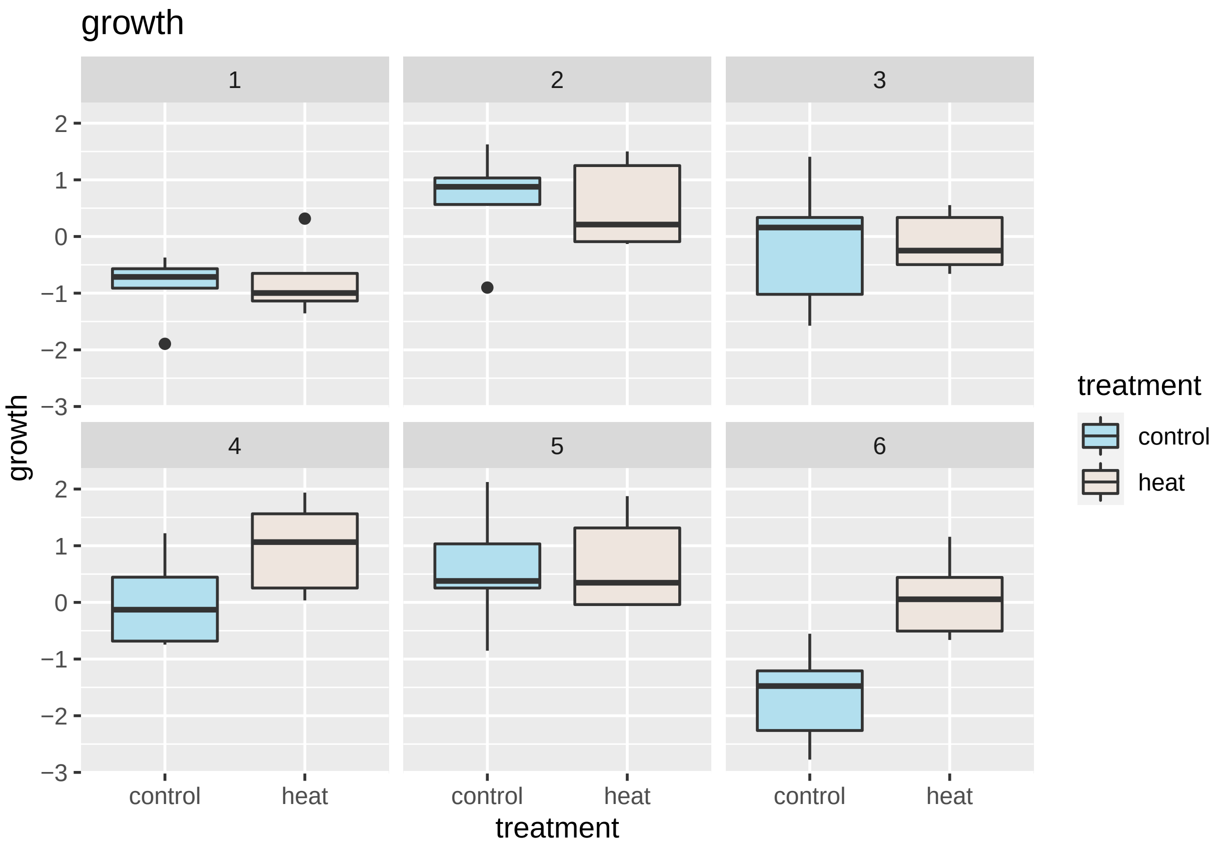
```
merlot2019.pca %>%
  group_by(block, treatment) %>%
  summarise(var.growth=var(merlot2019.growth.pc),
            mean.growth=mean(merlot2019.growth.pc),
            var.yield=var(merlot2019.yield.pc),
            mean.yield=mean(merlot2019.yield.pc),
            var.quality=var(merlot2019.quality.pc),
            mean.quality=mean(merlot2019.quality.pc),
            var.yield=var(merlot2019.pruning.pc),
            mean.yield=mean(merlot2019.pruning.pc))
```

```
## `summarise()` has grouped output by 'block'. You can override using the `.groups` argument.
```

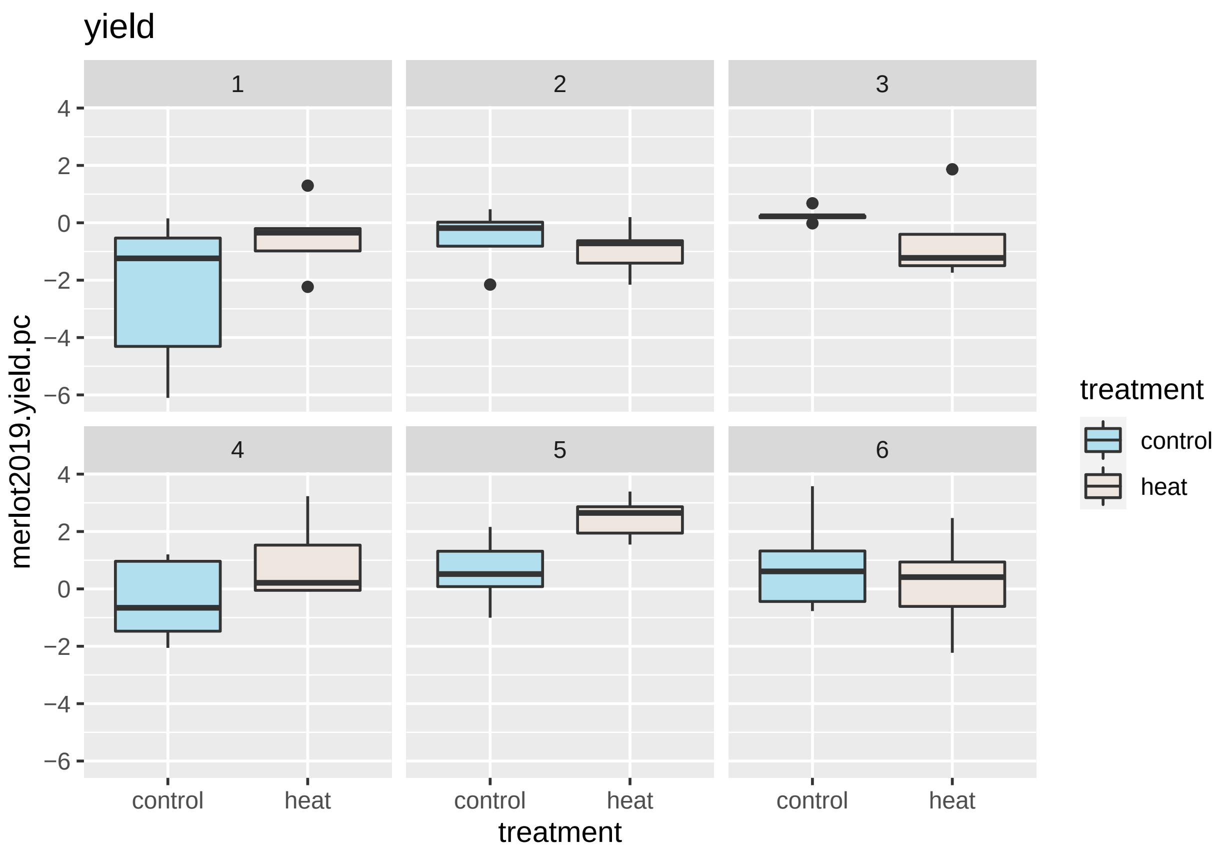
```
## # A tibble: 12 x 8
## # Groups:   block [6]
##    block treatment var.growth mean.growth var.yield mean.yield var.quality
##    <fct> <fct>          <dbl>       <dbl>     <dbl>      <dbl>       <dbl>
##  1 1     control        0.353     -0.892     1.69      -1.04         2.28
##  2 1     heat           0.431     -0.765     0.995     -0.524        0.212
##  3 2     control        0.891      0.640     2.02      -0.974        3.43
##  4 2     heat           0.597      0.548     0.468      0.556        0.662
##  5 3     control        1.39      -0.139     0.318      0.353        0.405
##  6 3     heat           0.277     -0.103     1.40       0.909        0.525
##  7 4     control        0.681      0.0216    0.889      0.655        0.776
##  8 4     heat           0.671      0.971     0.453     -0.0178       1.54
##  9 5     control        1.20       0.587     3.16       0.494        2.29
## 10 5     heat           0.751      0.688      1.20     -0.670        1.57
## 11 6     control        0.767     -1.65       1.31      1.19         2.77
## 12 6     heat           0.546      0.0970     1.08     -0.934        0.902
## # ... with 1 more variable: mean.quality <dbl>
```

2. Visualization (side by side boxplots for each response variable)

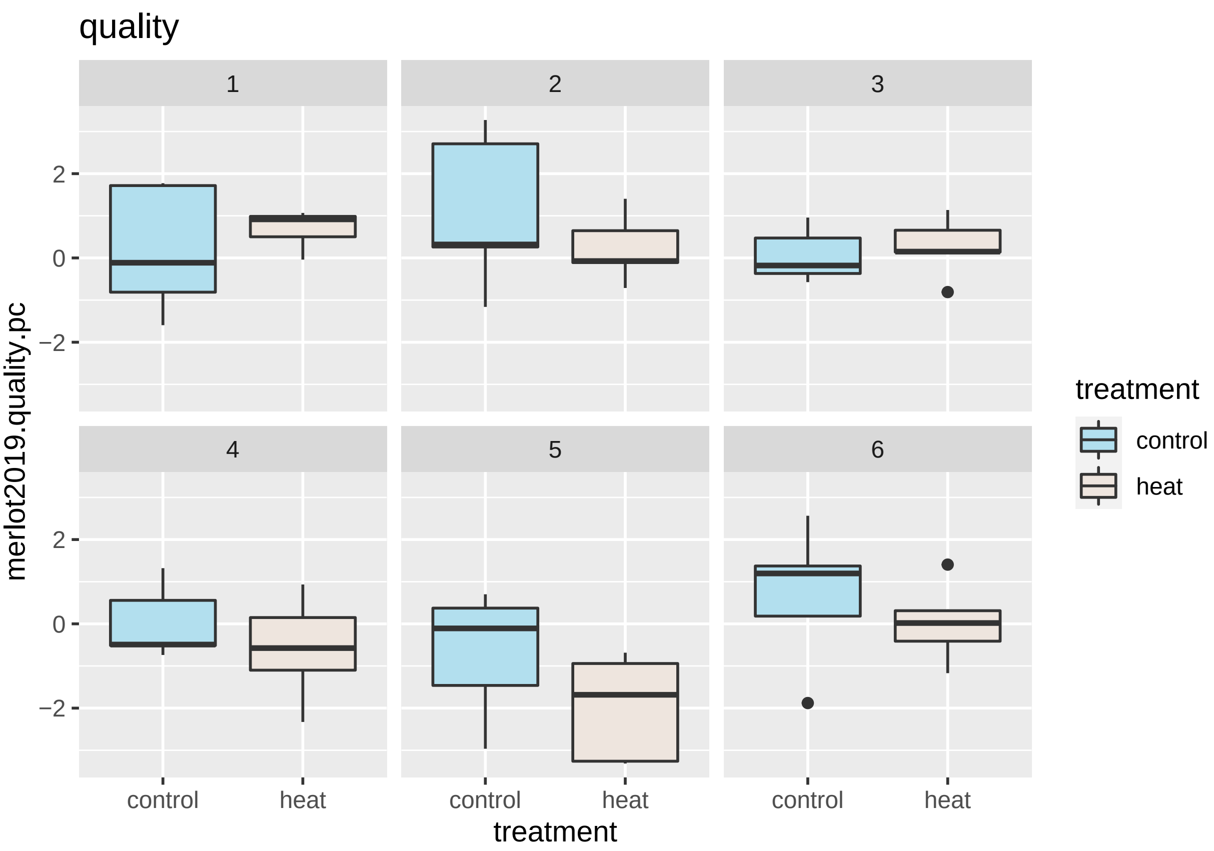
```
ggplot(merlot2019.pca, aes(x=treatment, y = merlot2019.growth.pc, fill=treatment)) +
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "growth") +
  ylab("growth")+
  facet_wrap(~block)
```



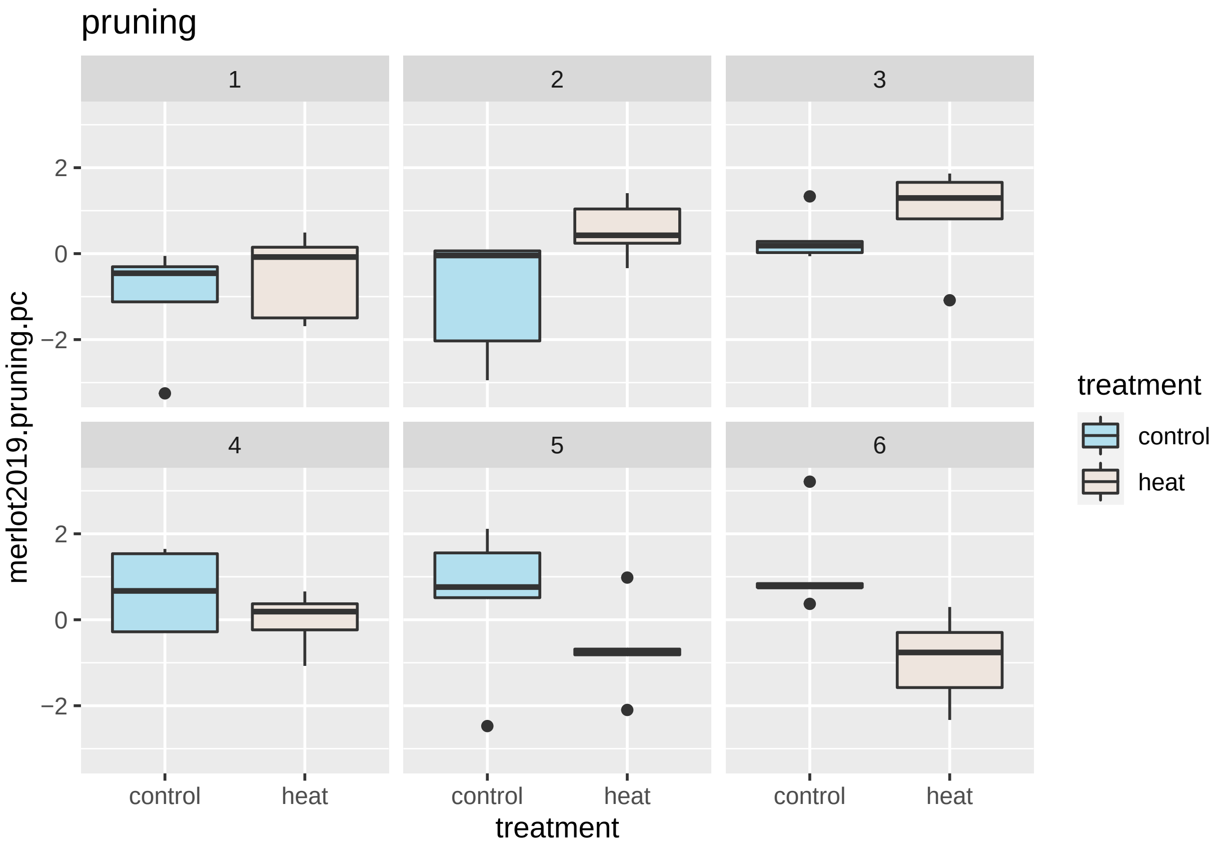
```
ggplot(merlot2019.pca, aes(x=treatment, y = merlot2019.yield.pc, fill=treatment)) +
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "yield") +
  facet_wrap(~block)
```



```
ggplot(merlot2019.pca, aes(x=treatment, y = merlot2019.quality.pc, fill=treatment)) +
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "quality") +
  facet_wrap(~block)
```



```r
ggplot(merlot2019.pca, aes(x=treatment, y = merlot2019.pruning.pc, fill=treatment)) +
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "pruning") +
  facet_wrap(~block)
```



merlot 2020

1. table of mean and variance under control and treatment for each response variable

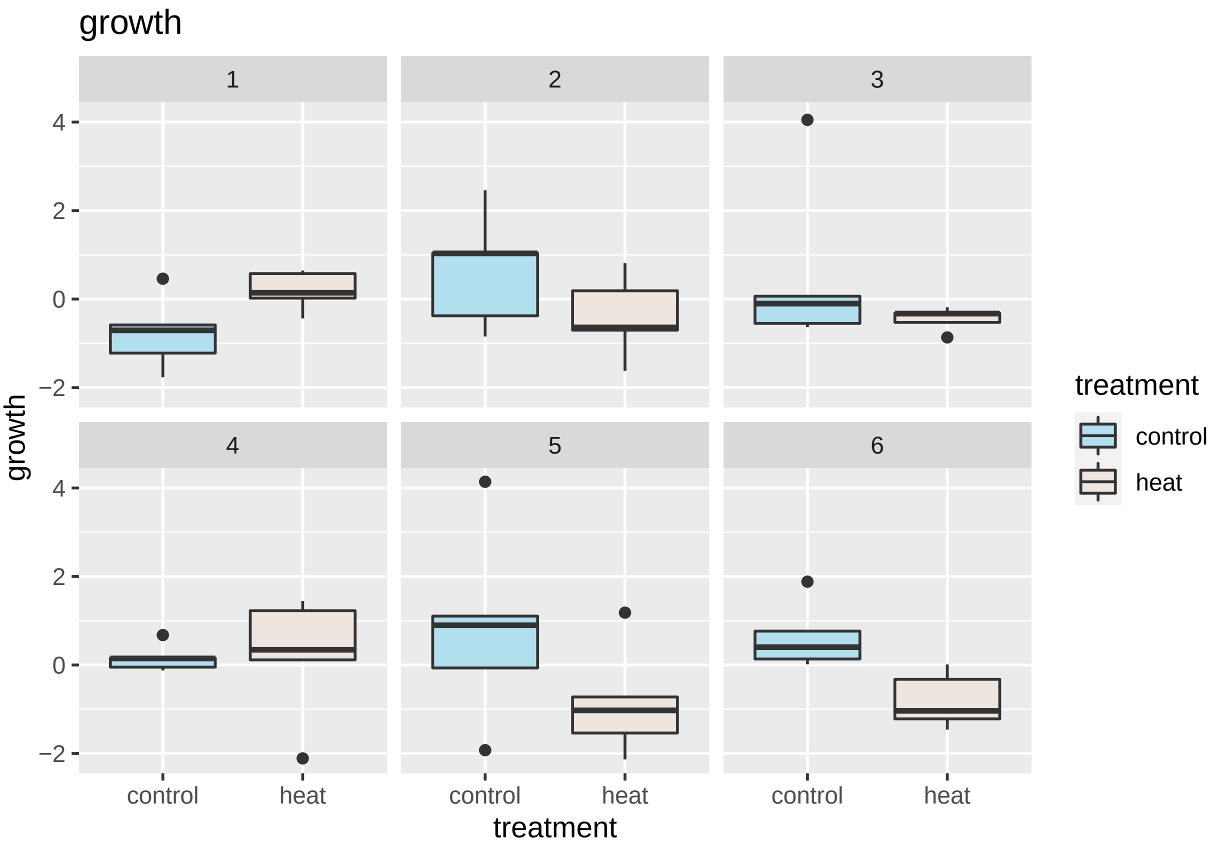
```
merlot2020.pca %>%
  group_by(block, treatment) %>%
  summarise(var.growth=var(merlot2020.growth.pc),
            mean.growth=mean(merlot2020.growth.pc),
            var.yield=var(merlot2020.yield.pc),
            mean.yield=mean(merlot2020.yield.pc),
            var.quality=var(merlot2020.quality.pc),
            mean.quality=mean(merlot2020.quality.pc))
```

```
## `summarise()` has grouped output by 'block'. You can override using the `.groups` argument.
```

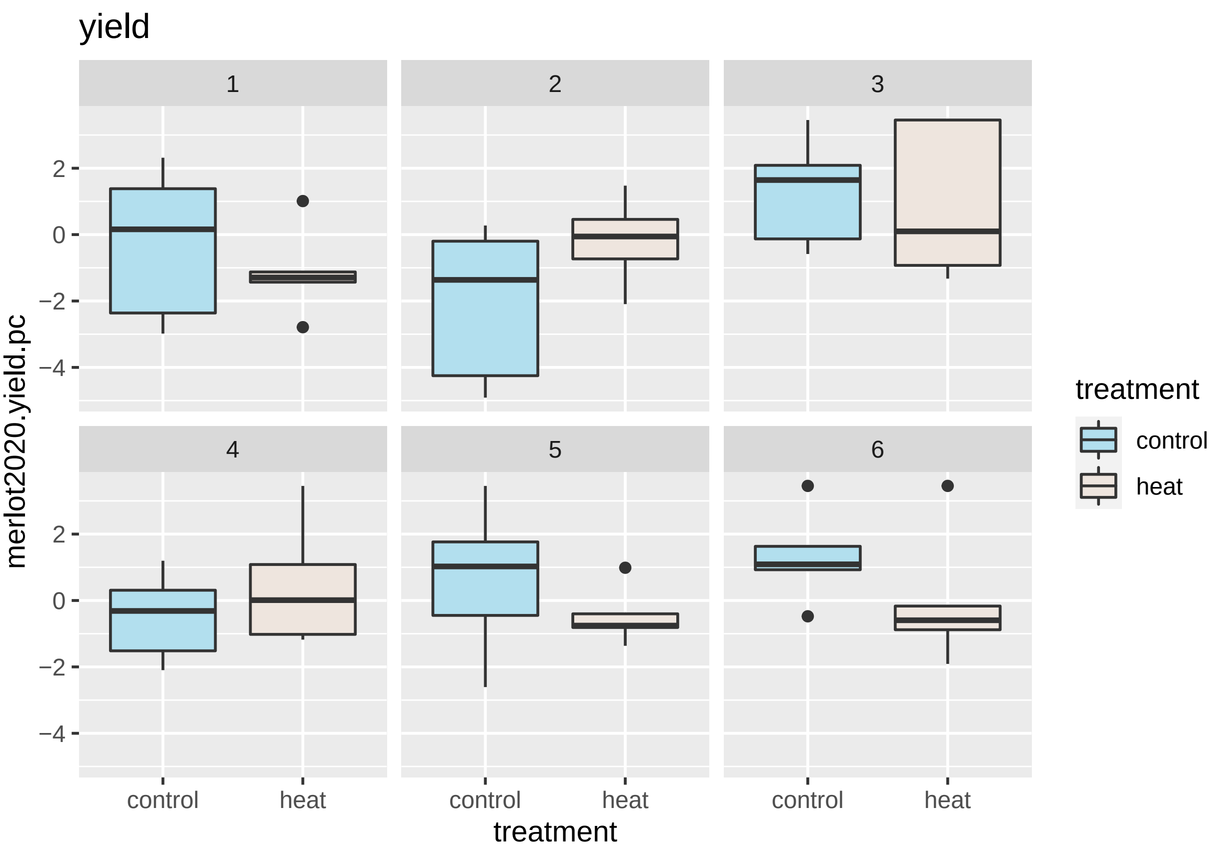
```
## # A tibble: 12 x 8
## # Groups:   block [6]
##    block treatment var.growth mean.growth var.yield mean.yield var.quality
##    <fct> <fct>          <dbl>       <dbl>     <dbl>      <dbl>       <dbl>
##  1 1     control       0.688      -0.764      5.33     -0.297        3.06
##  2 1     heat          0.195       0.190      1.86     -1.13         0.198
##  3 2     control       1.72        0.661      5.58     -2.09         2.40
##  4 2     heat          0.865      -0.393      1.78     -0.190        0.315
##  5 3     control       3.88        0.567      2.74      1.29         2.49
##  6 3     heat          0.0703     -0.445      5.49      0.950        5.11
##  7 4     control       0.0977      0.161      1.79     -0.484        0.777
##   8 4     heat           1.99          0.204     3.61      0.469        4.31
##   9 5     control        4.86          0.830     5.26      0.637        4.53
##  10 5     heat           1.57         -0.848     0.782    -0.470        2.29
##  11 6     control        0.565         0.641     2.02      1.32         1.63
##  12 6     heat           0.388        -0.805     4.18     -0.0205       3.38
## # ... with 1 more variable: mean.quality <dbl>
```

2. Visualization (side by side boxplots for each response variable)

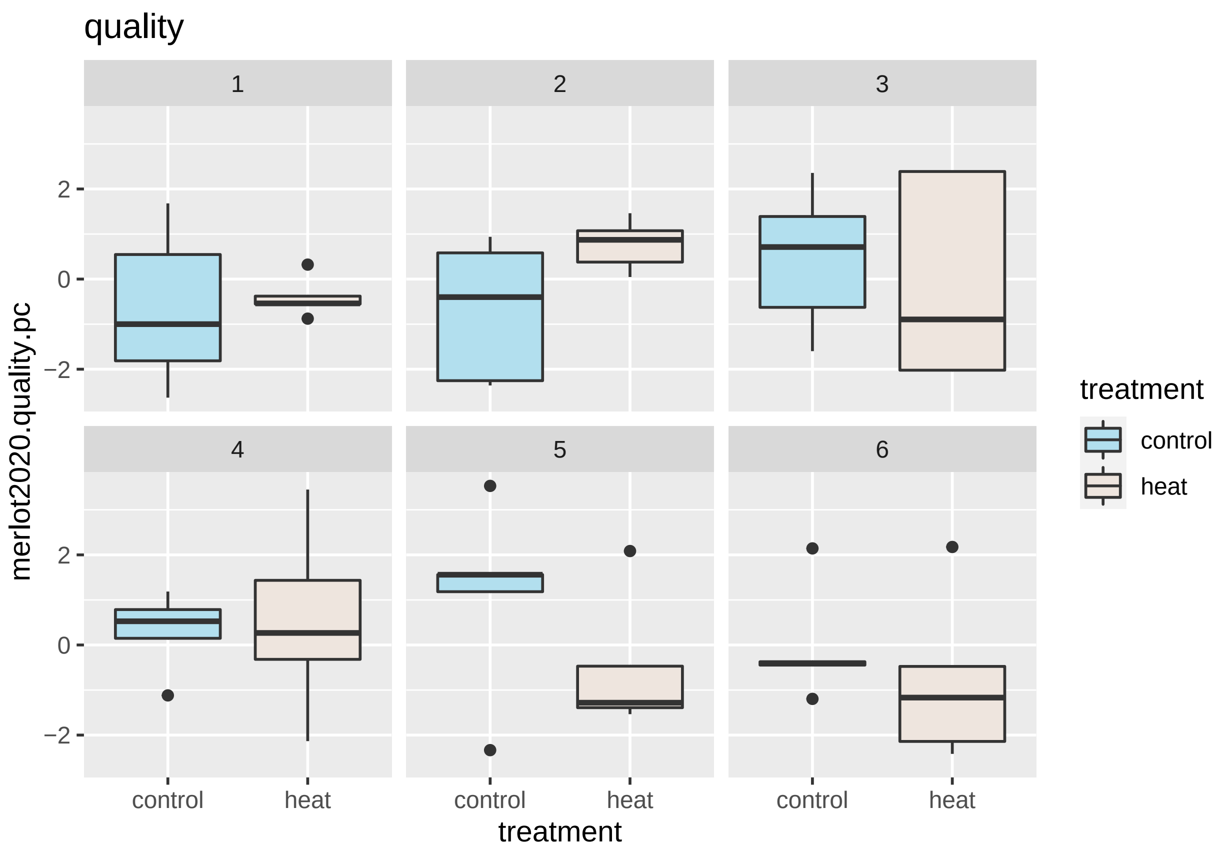
```
ggplot(merlot2020.pca, aes(x=treatment, y = merlot2020.growth.pc, fill=treatment)) +
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "growth") +
  ylab("growth")+
  facet_wrap(~block)
```



```
ggplot(merlot2020.pca, aes(x=treatment, y = merlot2020.yield.pc, fill=treatment)) +
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "yield") +
  facet_wrap(~block)
```



```
ggplot(merlot2020.pca, aes(x=treatment, y = merlot2020.quality.pc, fill=treatment)) +
  scale_fill_manual(values=c("lightblue2", "seashell2"))+
  geom_boxplot()+
  labs(title = "quality") +
  facet_wrap(~block)
```



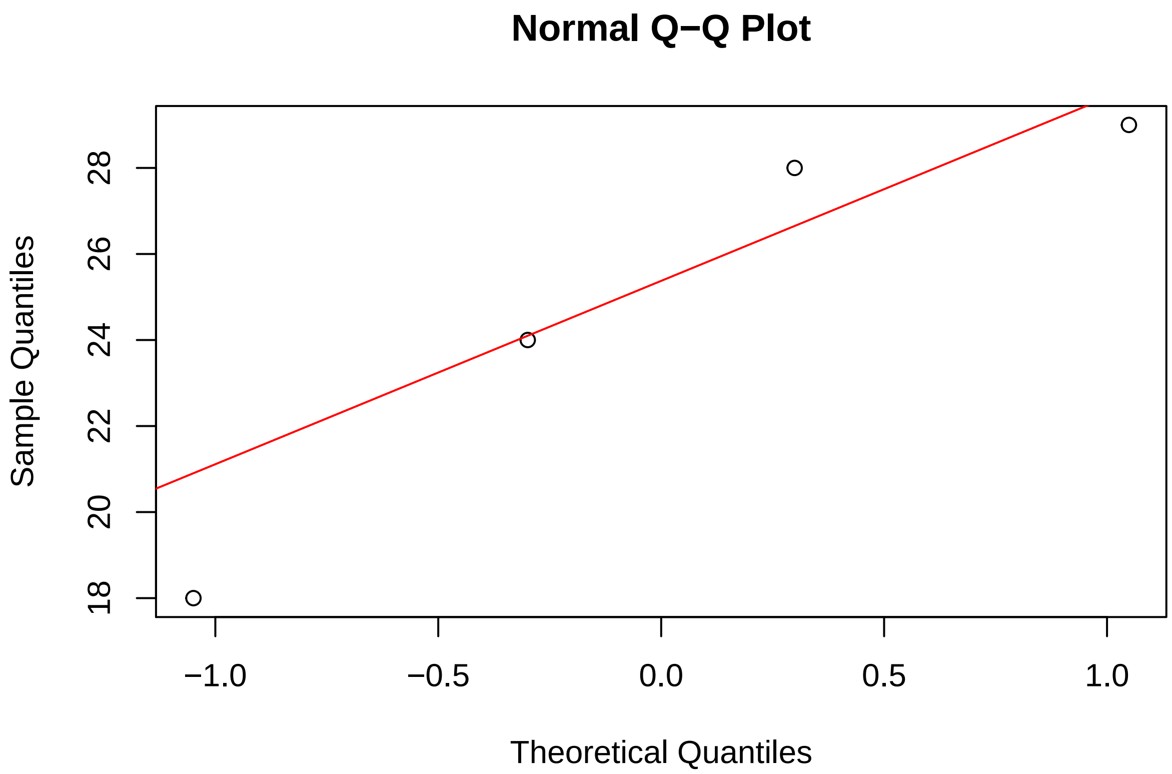
Based on the side-by-side box plots above, it seems that many outcome variables do not have equal variance, or the distribution is not quite normal. One limitation which can explain this is that our sample size is really small for each treatment in each block(5 or 6 datapoints). Hence, the plot cannot really tell much useful information about the data.

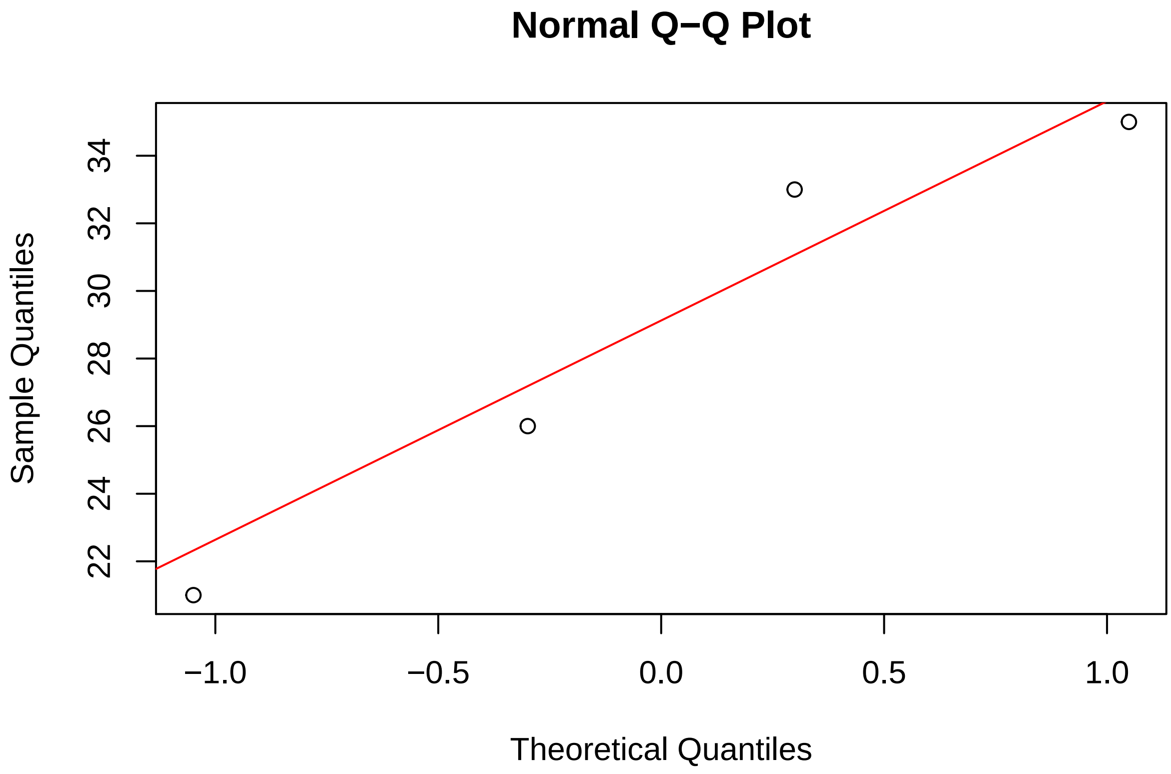
Below, wilcoxon rank tests can be performed to test whether the distribution in control and treatment groups are the same. (that would be a lot of tests… I only tried out the test for number of clusters in Char2019 for five blocks.) ### randomization test chardonnay 2019

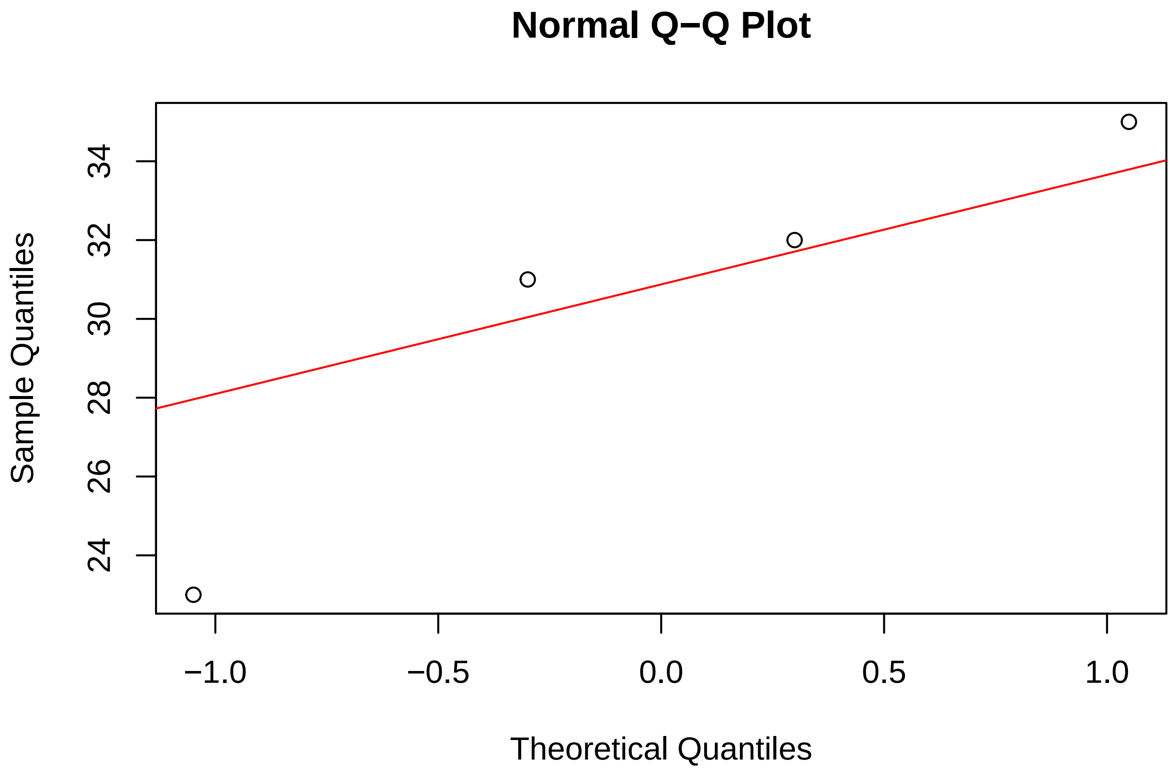
```
for (i in unique(chardonnay2019$block)){
  res=wilcox.test(chardonnay2019$X..of.clusters[chardonnay2019$block==i & chardonnay2019$treatment=="hea
                  chardonnay2019$X..of.clusters[chardonnay2019$block==i & chardonnay2019$treatment=="co
                  exact=FALSE)
  print(res)
}
```

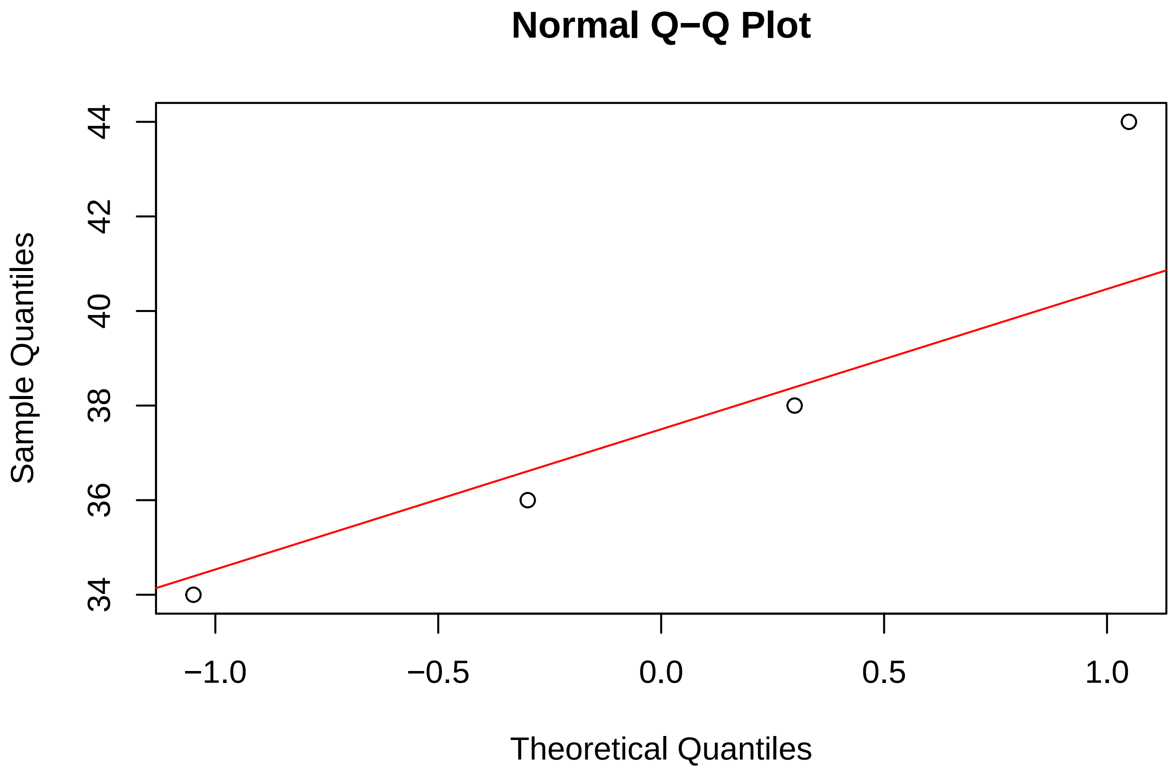
```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  chardonnay2019$X..of.clusters[chardonnay2019$block == i & chardonnay2019$treatment == "heat"]
## W = 5, p-value = 0.4705
## alternative hypothesis: true location shift is not equal to 0
##
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  chardonnay2019$X..of.clusters[chardonnay2019$block == i & chardonnay2019$treatment == "heat"]
## W = 1, p-value = 0.0606
## alternative hypothesis: true location shift is not equal to 0
##
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  chardonnay2019$X..of.clusters[chardonnay2019$block == i & chardonnay2019$treatment == "heat"]
## W = 13, p-value = 0.1913
## alternative hypothesis: true location shift is not equal to 0
##
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  chardonnay2019$X..of.clusters[chardonnay2019$block == i & chardonnay2019$treatment == "heat"]
## W = 2.5, p-value = 0.1465
## alternative hypothesis: true location shift is not equal to 0
##
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  chardonnay2019$X..of.clusters[chardonnay2019$block == i & chardonnay2019$treatment == "heat"]
## W = 4, p-value = 0.3005
## alternative hypothesis: true location shift is not equal to 0
```

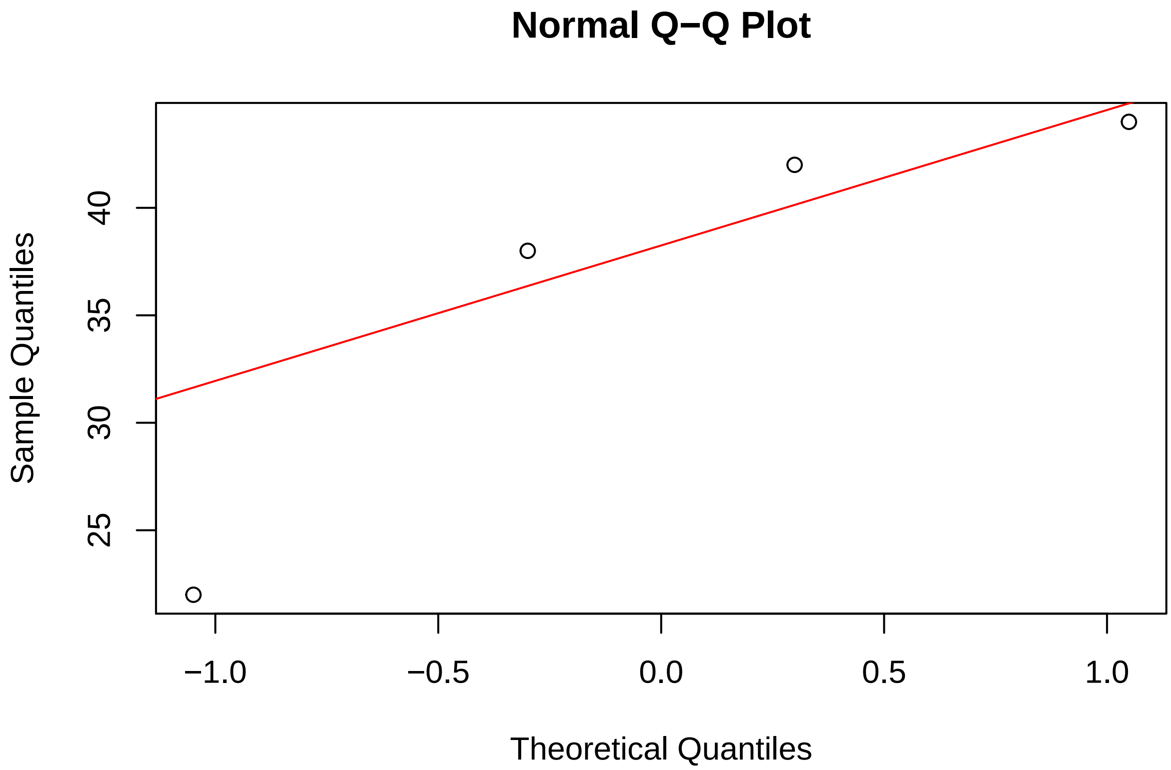
QQ plot for number of clusters in chardonnary 2019 (for each block/treatment combo)

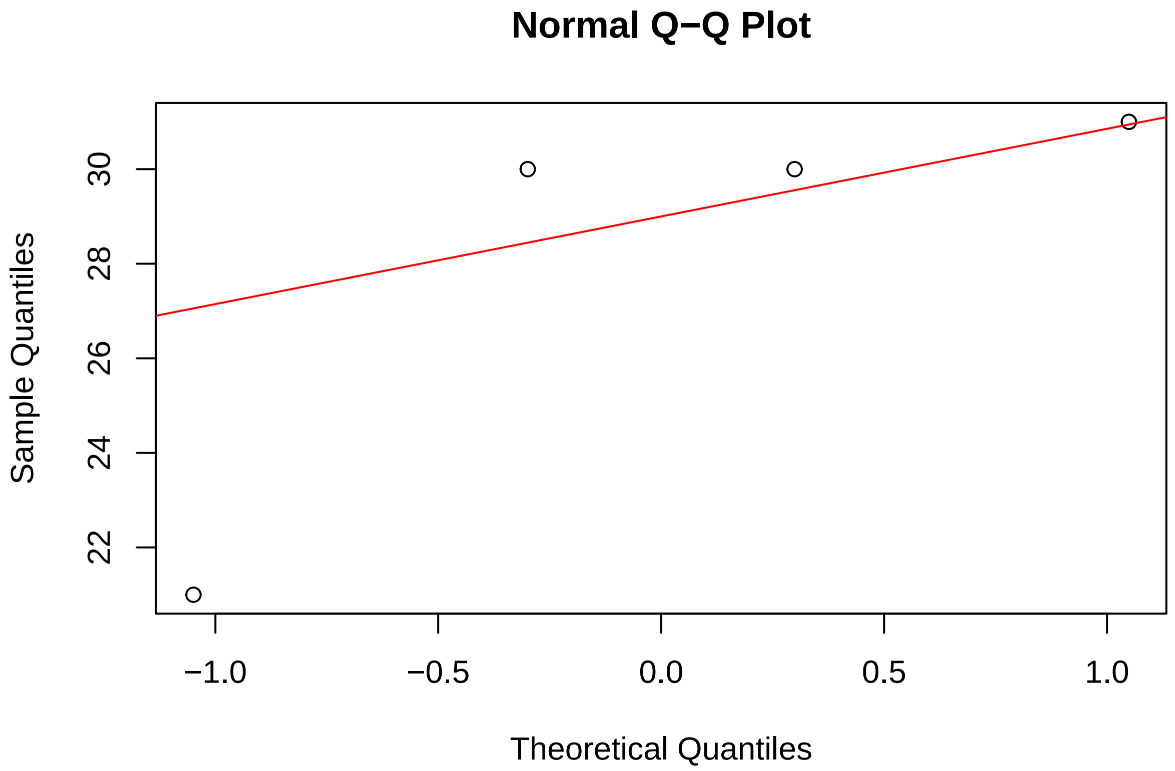


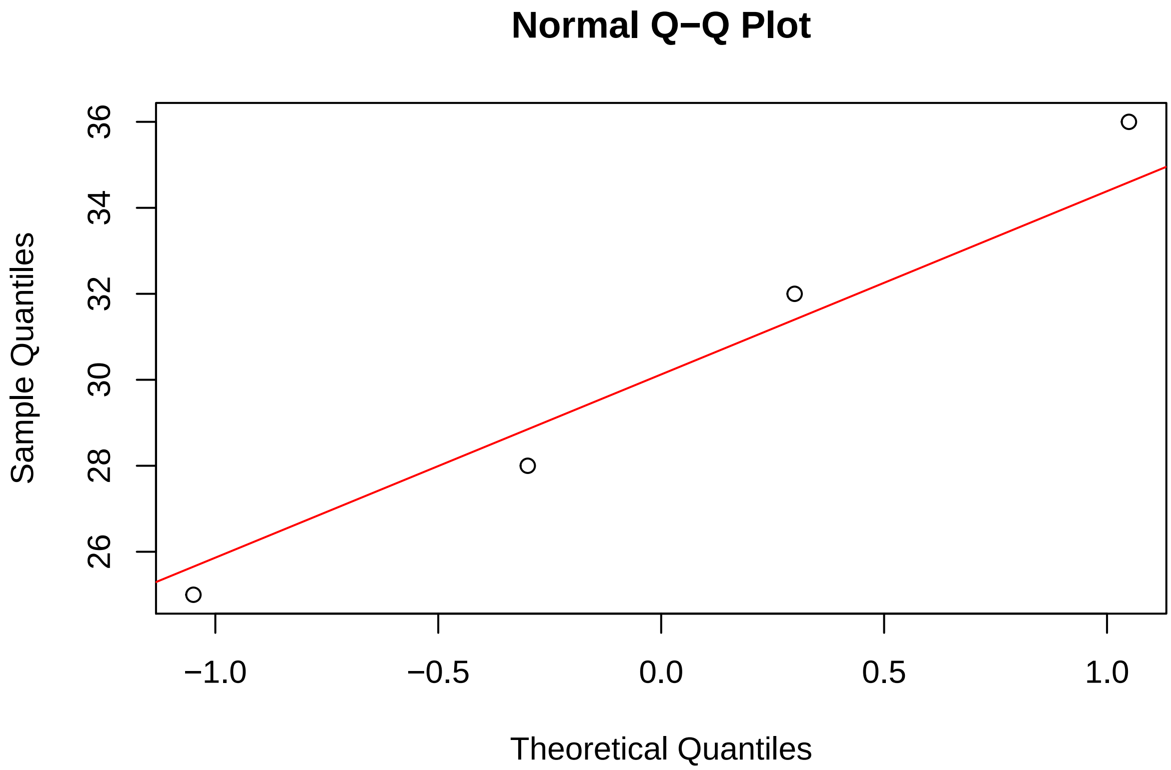

Normal Q−Q Plot
Sample Quantiles
22
24
26
28
30
32
34
Theoretical Quantiles
−1.0
−0.5
0.0
0.5
1.0

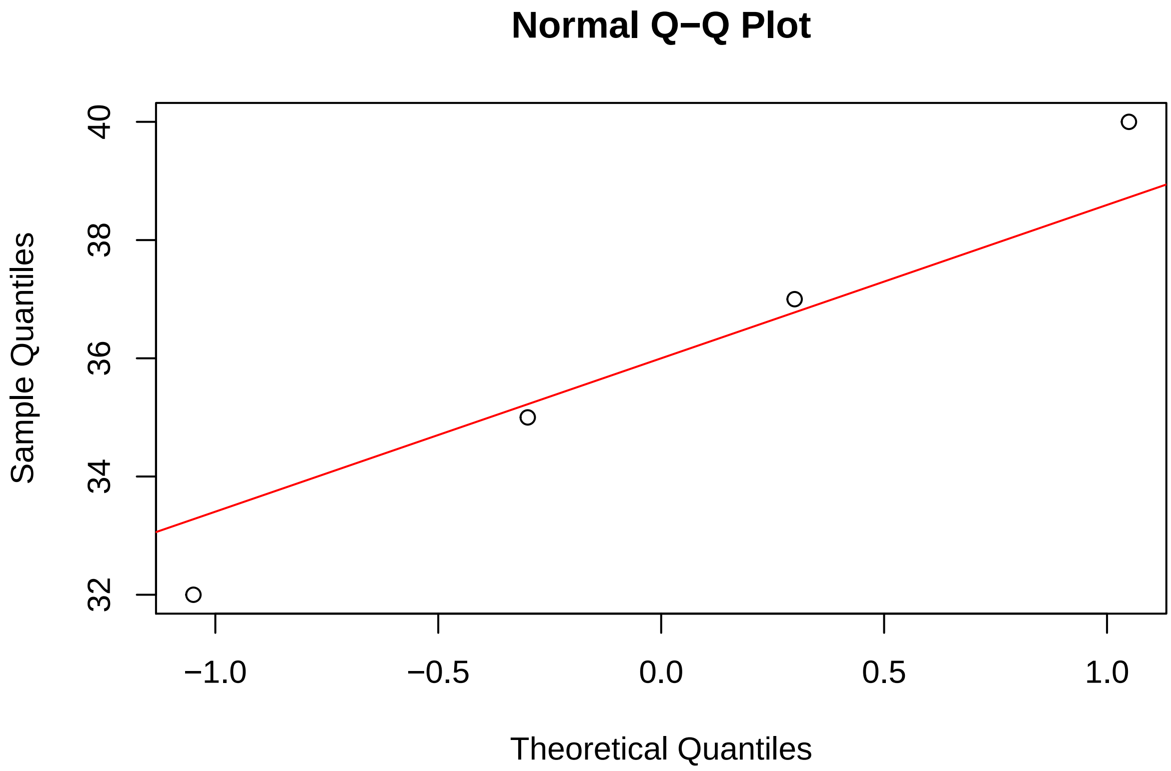

Normal Q−Q Plot
Sample Quantiles
24
26
28
30
32
34
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles

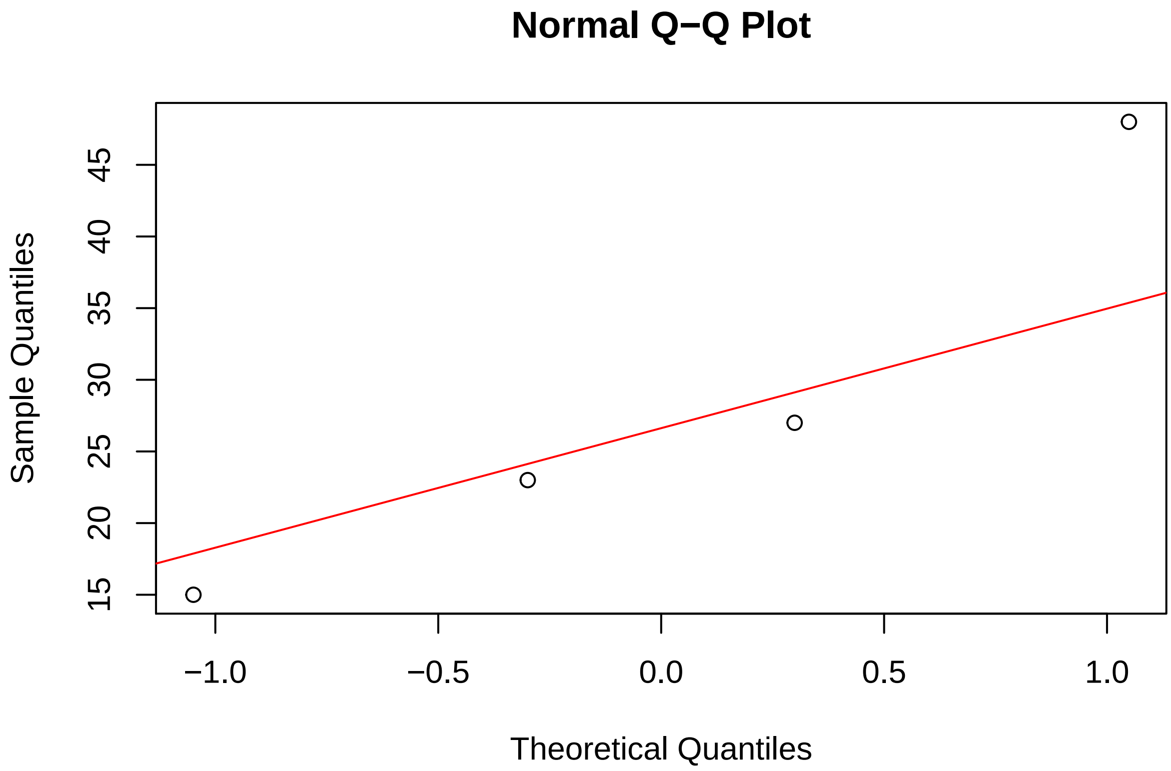

Normal Q−Q Plot
Sample Quantiles
34
36
38
40
42
44
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles

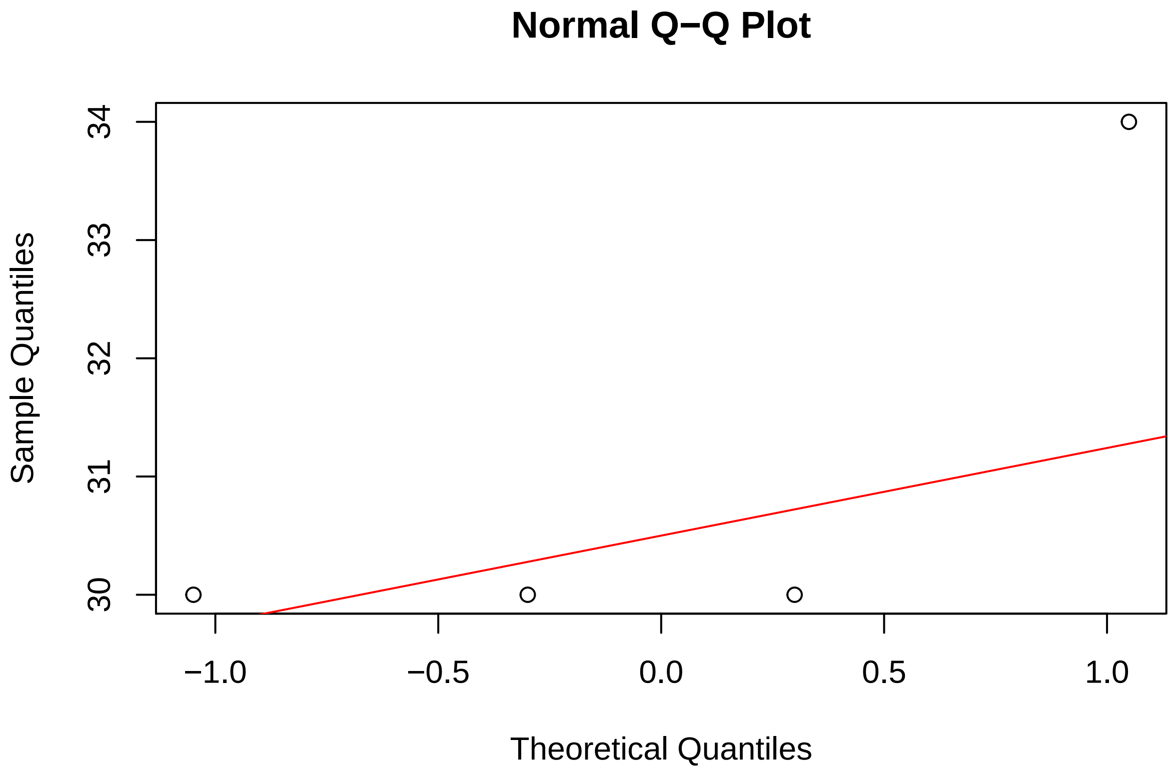

Normal Q−Q Plot
Sample Quantiles
25
30
35
40
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles


Normal Q−Q Plot
Sample Quantiles
22
24
26
28
30
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles


Normal Q−Q Plot
Sample Quantiles
26
28
30
32
34
36
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles


Normal Q−Q Plot
Sample Quantiles
32
34
36
38
40
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles

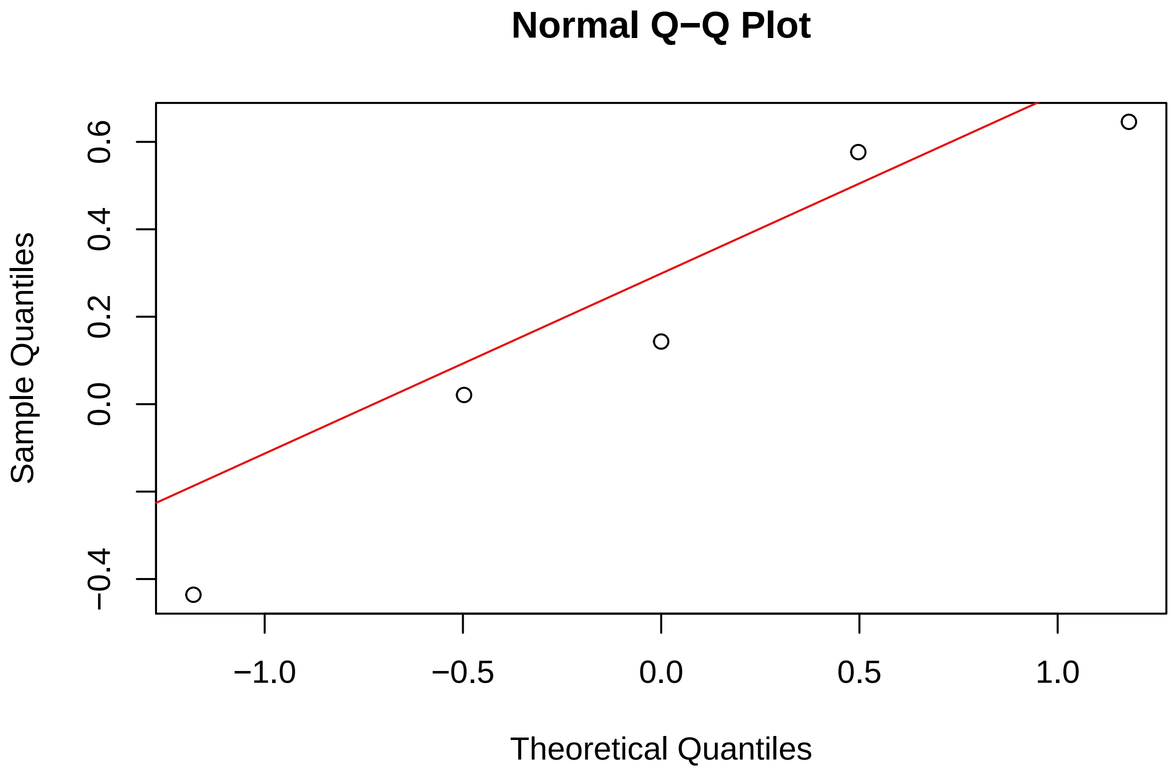

Normal Q−Q Plot
Sample Quantiles
15
20
25
30
35
40
45
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles

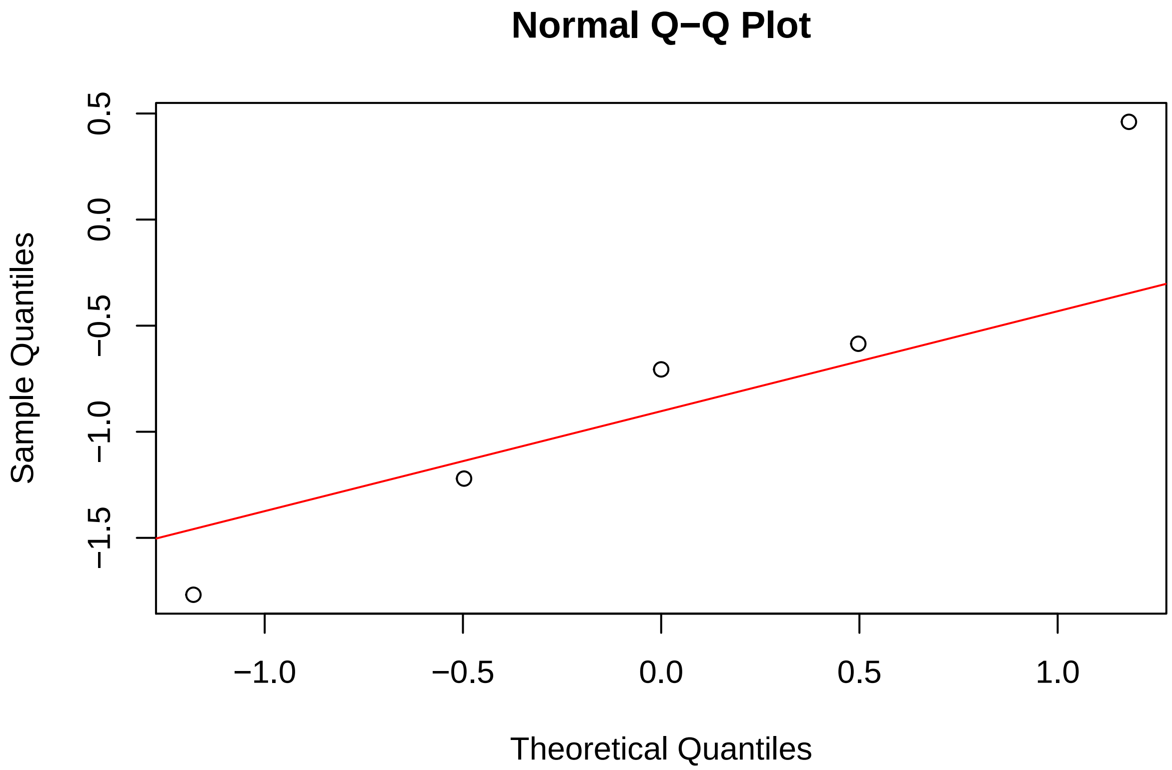


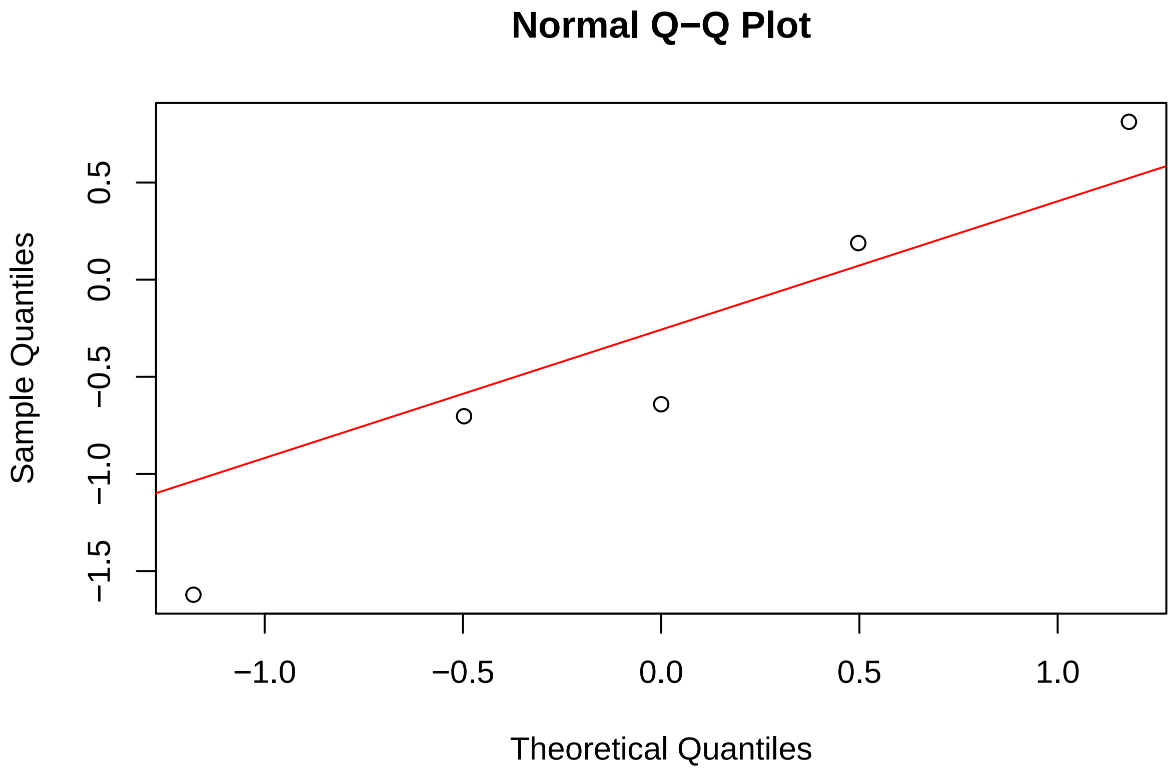
F test and randomization test for equal variance between treatments for each block in chardonnary 2019

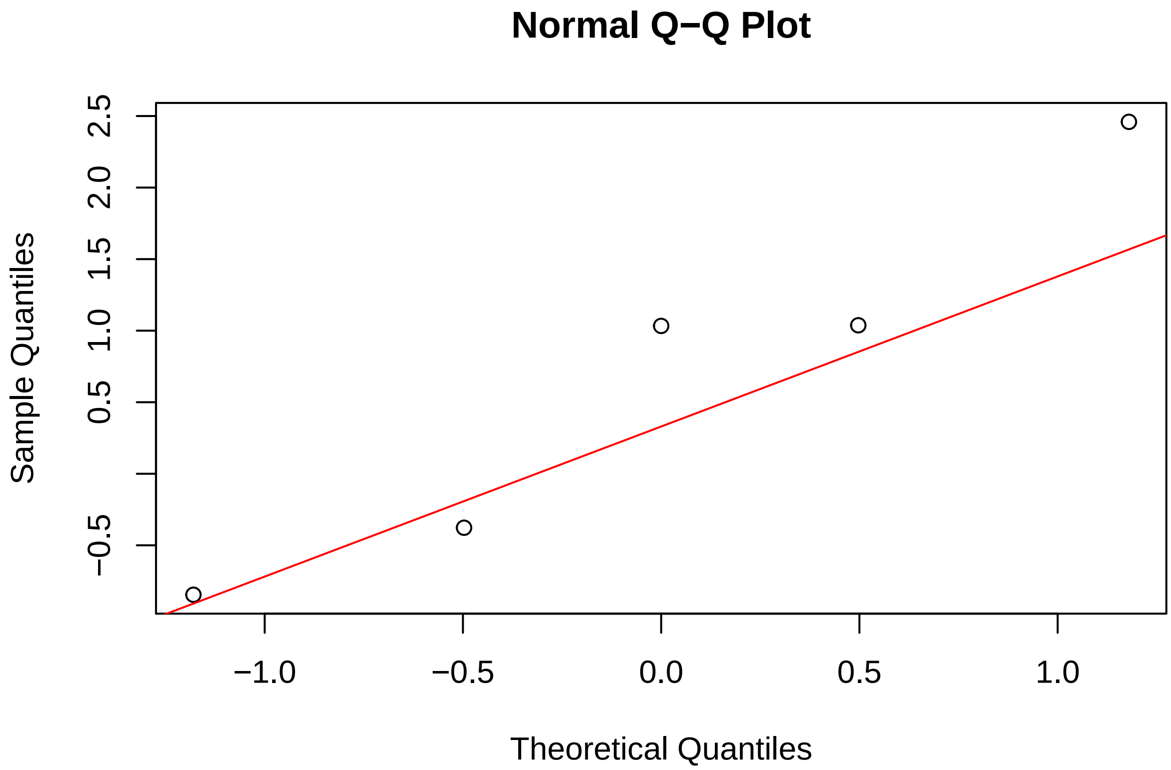
```
## [1] "F test"
## [1] 0.6842653
## [1] "rand test"
## [1] 0.6428571
## [1] "F test"
## [1] 0.7860631
## [1] "rand test"
## [1] 0.7571429
## [1] "F test"
## [1] 0.2464142
## [1] "rand test"
## [1] 0.1714286
## [1] "F test"
## [1] 0.5777467
## [1] "rand test"
## [1] 0.6
## [1] "F test"
## [1] 0.009387336
## [1] "rand test"
## [1] 0.04285714
```

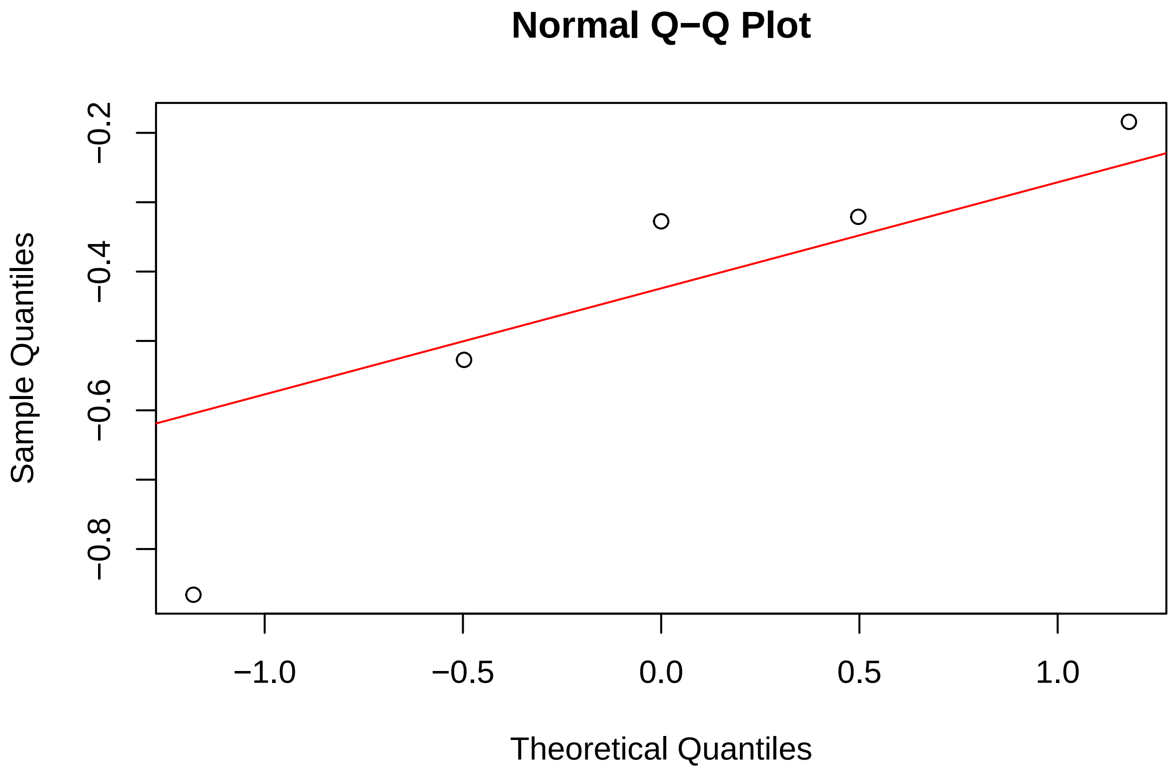
QQ plot for the growth PC in merlot 2020 (for each block/treatment combo)

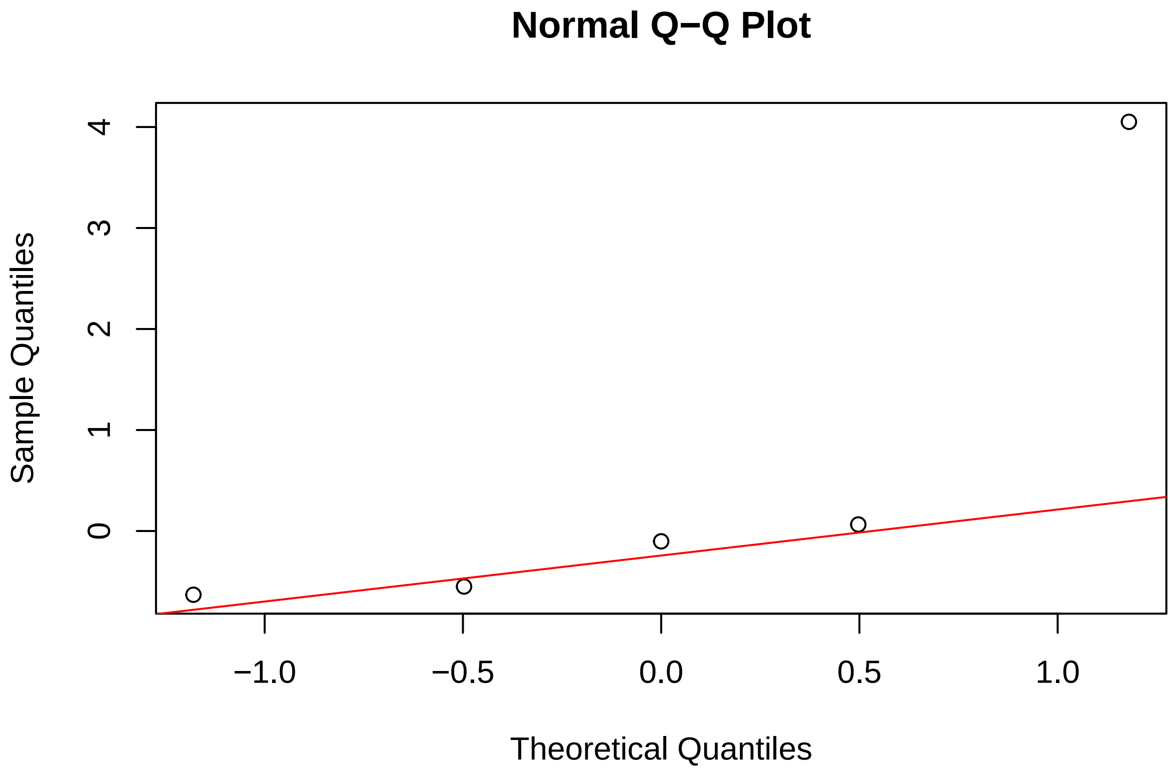

Normal Q−Q Plot
Sample Quantiles
−0.4
0.0
0.2
0.4
0.6
Theoretical Quantiles
−1.0
−0.5
0.0
0.5
1.0

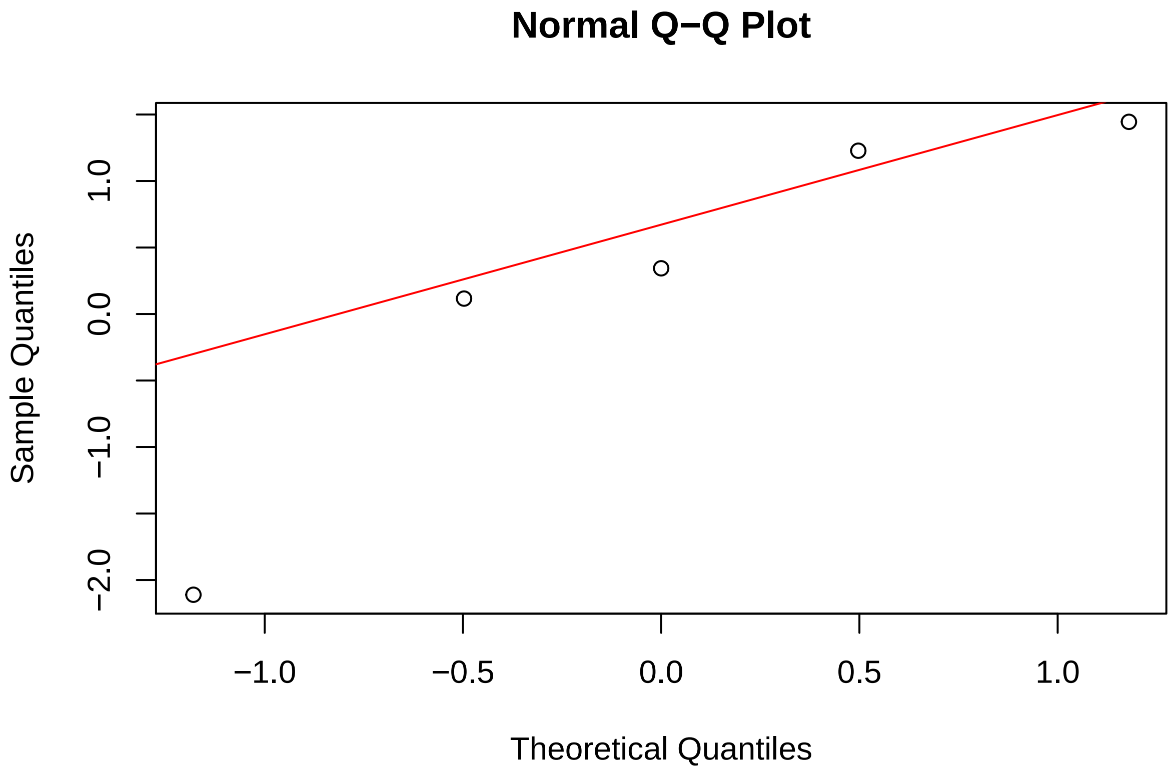

Normal Q−Q Plot
Sample Quantiles
0.5
0.0
−0.5
−1.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles

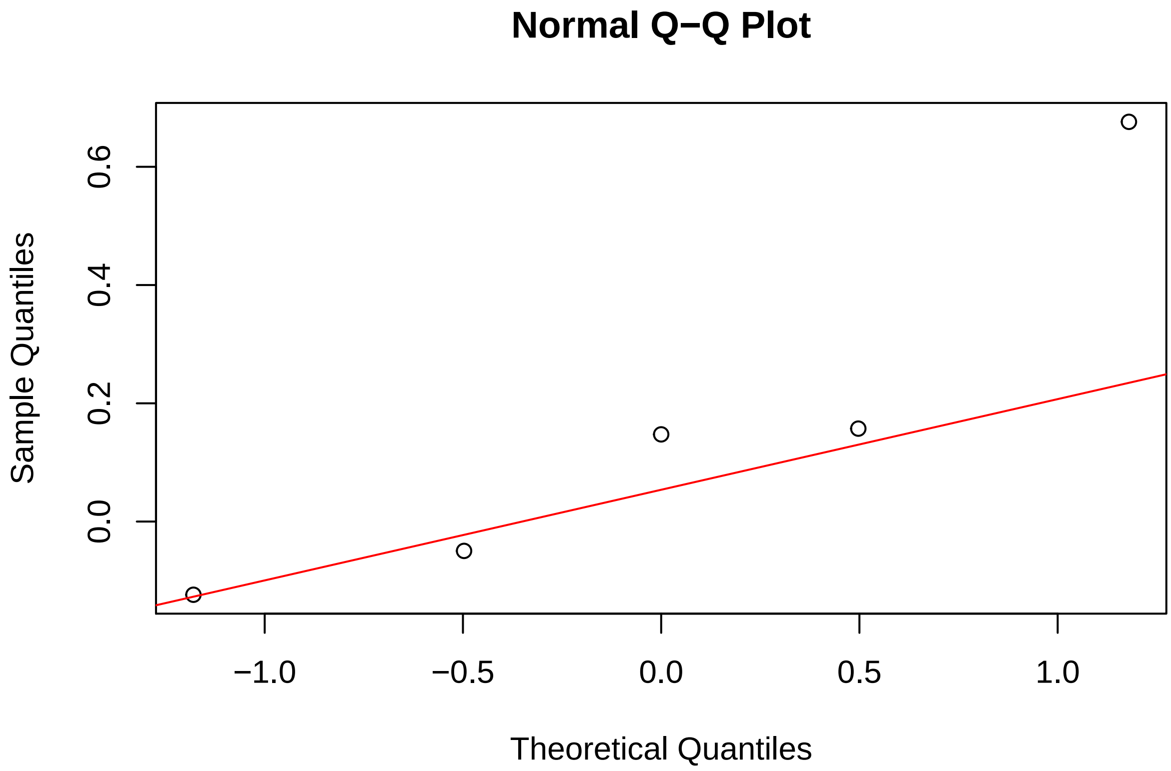

Normal Q−Q Plot
Sample Quantiles
0.5
0.0
−0.5
−1.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles

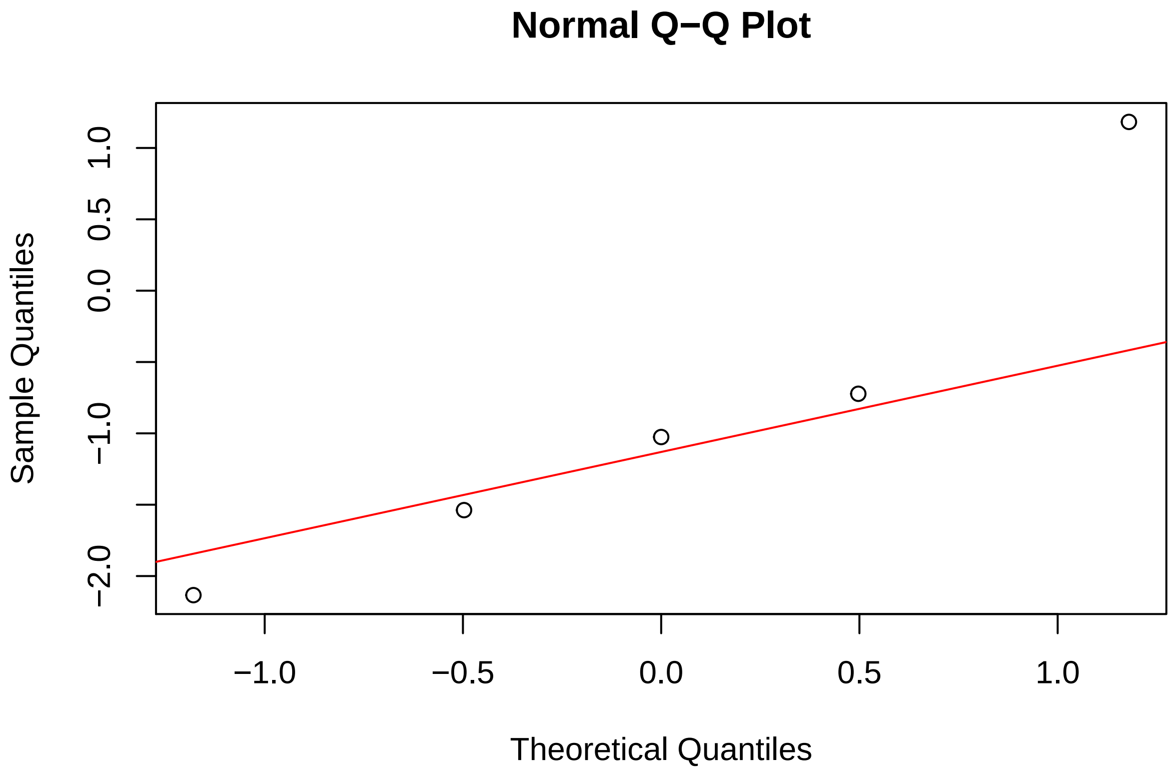

Normal Q−Q Plot
Sample Quantiles
Theoretical Quantiles
−0.5
0.5
1.0
1.5
2.0
2.5
−1.0
−0.5
0.0
0.5
1.0

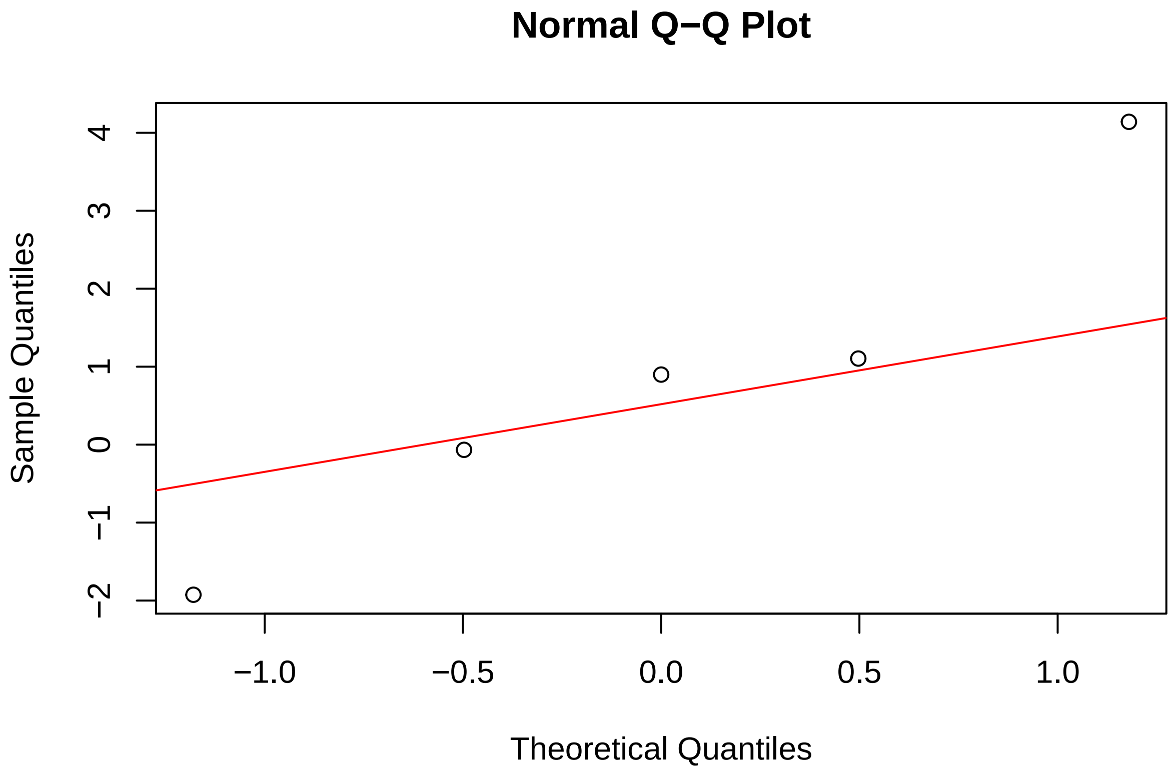

Normal Q−Q Plot
Sample Quantiles
−0.2
−0.4
−0.6
−0.8
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles

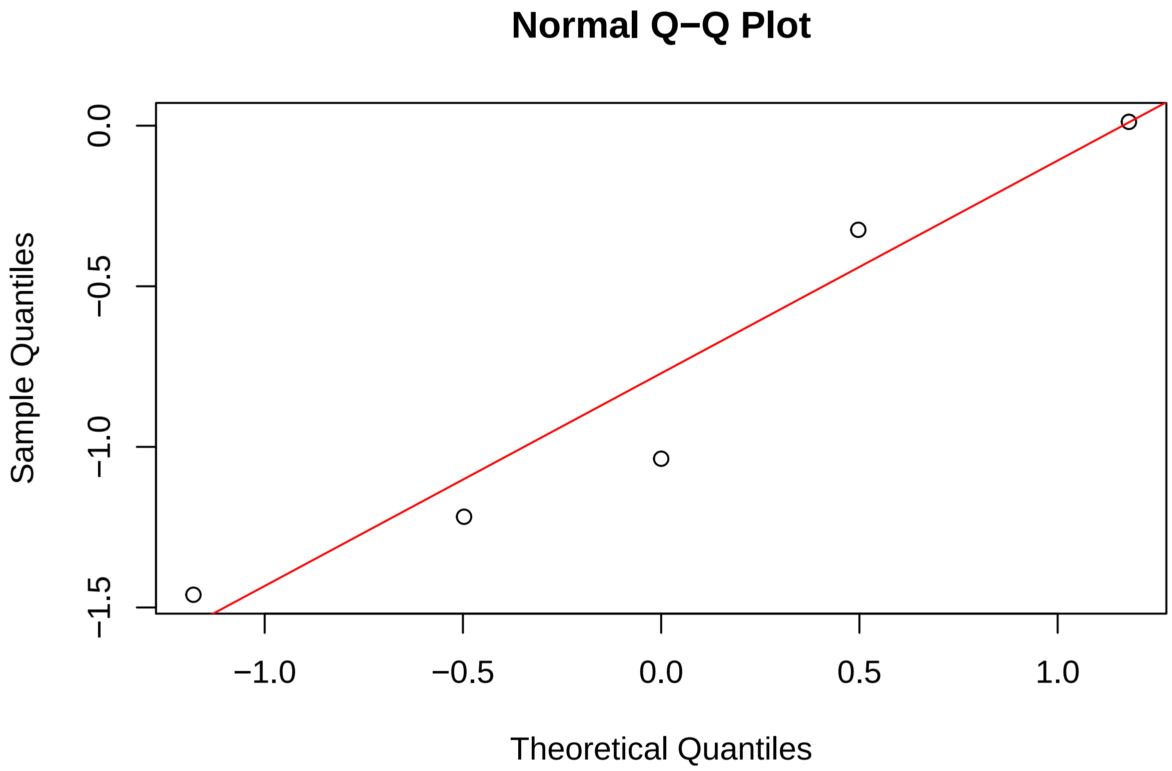

Normal Q−Q Plot
Sample Quantiles
0
1
2
3
4
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles

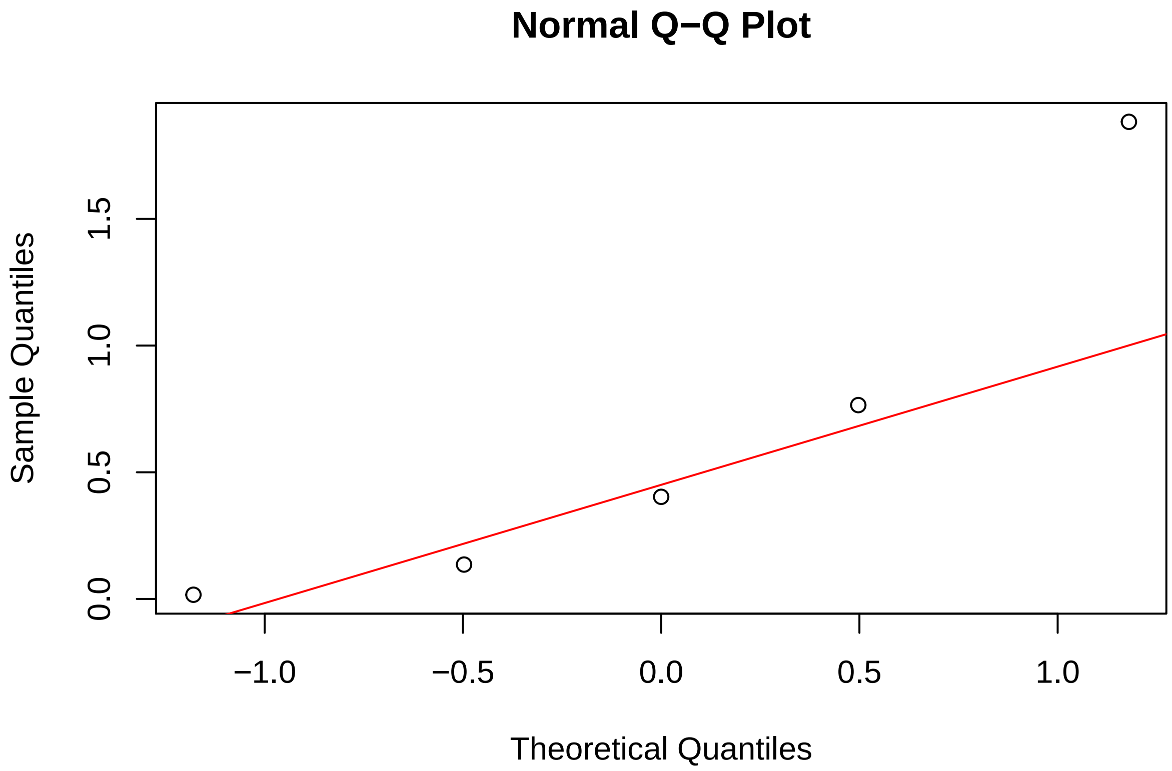

Normal Q−Q Plot
Sample Quantiles
−2.0
−1.0
0.0
1.0
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles


Normal Q−Q Plot
Sample Quantiles
0.0
0.2
0.4
0.6
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles


Normal Q−Q Plot
Sample Quantiles
−2.0
−1.0
0.0
0.5
1.0
Theoretical Quantiles
−1.0
−0.5
0.0
0.5
1.0


Normal Q−Q Plot
Sample Quantiles
−2
−1
0
1
2
3
4
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles


Normal Q−Q Plot
Sample Quantiles
0.0
−0.5
−1.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
Theoretical Quantiles



F test and randomization test for equal variance between treatments for each block in merlot 2020

```
## [1] "F test"
## [1] 0.2494025
## [1] "rand test"
## [1] 0.4722222
## [1] "F test"
## [1] 0.5232279
## [1] "rand test"
## [1] 0.5119048
## [1] "F test"
## [1] 0.001876501
## [1] "rand test"
## [1] 0.6706349
## [1] "F test"
## [1] 0.01270178
## [1] "rand test"
## [1] 0.1230159
## [1] "F test"
## [1] 0.3007981
## [1] "rand test"
## [1] 0.6944444
## [1] "F test"
## [1] 0.7247435
## [1] "rand test"
## [1] 0.9087302
```

We see that the normality assumptions are slightly violated, and the equal variance assumptions are also violated even though we only compared variances between treatments within each block. Therefore we will use both a parametric and a non parametric test in the following analysis. (Should repeat the above for all other variables.)

Aligned rank transform ANOVA (nonparametric test for two-way layout)

Chardonnay 2019 number of clusters

```
## Registered S3 methods overwritten by 'lme4':
##   method                          from
##   cooks.distance.influence.merMod car
##   influence.merMod                car
##   dfbeta.influence.merMod         car
##   dfbetas.influence.merMod        car
```

```
## Analysis of Variance of Aligned Rank Transformed Data
## 
## Table Type: Analysis of Variance Table (Type I) 
## Model: No Repeated Measures (lm)
## Response: art(X..of.clusters)
## 
##                   Df Df.res F value   Pr(>F)  
## 1 treatment        1     30  1.2226 0.277632  
## 2 block            4     30  2.4189 0.070334 .
## 3 treatment:block  4     30  2.4056 0.071534 .
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Chardonnay 2019 yield

```
## Analysis of Variance of Aligned Rank Transformed Data
## 
## Table Type: Analysis of Variance Table (Type I) 
## Model: No Repeated Measures (lm)
## Response: art(yield..kg.)
## 
##                   Df Df.res  F value  Pr(>F)
## 1 treatment        1     30 0.991545 0.32733
## 2 block            4     30 1.355386 0.27276
## 3 treatment:block  4     30 0.046131 0.99576
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Chardonnay 2019 average cluster weight

```
## Analysis of Variance of Aligned Rank Transformed Data
## 
## Table Type: Analysis of Variance Table (Type I) 
## Model: No Repeated Measures (lm)
## Response: art(av.cluster.weight..kg.)
##
##                   Df Df.res F value  Pr(>F)
## 1 treatment        1     30 0.00902 0.92497
## 2 block            4     30 0.64523 0.63454
## 3 treatment:block  4     30 1.36546 0.26932
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Chardonnay 2020 number of clusters

```
## Analysis of Variance of Aligned Rank Transformed Data
##
## Table Type: Analysis of Variance Table (Type I)
## Model: No Repeated Measures (lm)
## Response: art(X..of.clusters)
##
##                   Df Df.res F value   Pr(>F)
## 1 treatment        1     30  1.0012 0.325025
## 2 block            4     30  2.9615 0.035634 *
## 3 treatment:block  4     30  2.8050 0.043283 *
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Chardonnay 2020 yield

```
## Analysis of Variance of Aligned Rank Transformed Data
##
## Table Type: Analysis of Variance Table (Type I)
## Model: No Repeated Measures (lm)
## Response: art(yield..kg.)
##
##                   Df Df.res F value   Pr(>F)
## 1 treatment        1     30 0.11283 0.739286
## 2 block            4     30 2.99336 0.034257 *
## 3 treatment:block  4     30 1.50554 0.225576
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Chardonnay 2020 average cluster weight

```
## Analysis of Variance of Aligned Rank Transformed Data
##
## Table Type: Analysis of Variance Table (Type I)
## Model: No Repeated Measures (lm)
## Response: art(av.cluster.weight..kg.)
##
##                   Df Df.res F value   Pr(>F)
## 1 treatment        1     30 0.71637 0.404037
## 2 block            4     30 2.45415 0.067265 .
## 3 treatment:block  4     30 0.36349 0.832608
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Merlot 2019 growth PC

```
## Analysis of Variance of Aligned Rank Transformed Data
##
## Table Type: Analysis of Variance Table (Type I)
## Model: No Repeated Measures (lm)
## Response: art(merlot2019.growth.pc)
##
##                   Df Df.res F value     Pr(>F)
## 1 treatment        1     48  2.8169   0.099774   .
## 2 block            5     48  7.0881 4.9361e-05 ***
## 3 treatment:block  5     48  1.6448   0.166313
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Merlot 2019 yield PC

```
## Analysis of Variance of Aligned Rank Transformed Data
##
## Table Type: Analysis of Variance Table (Type I)
## Model: No Repeated Measures (lm)
## Response: art(merlot2019.yield.pc)
##
##                   Df Df.res F value    Pr(>F)
## 1 treatment        1     48  1.5771 0.2152602
## 2 block            5     48  4.8827 0.0010855 **
## 3 treatment:block  5     48  1.8329 0.1242053
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Merlot 2019 quality PC

```
## Analysis of Variance of Aligned Rank Transformed Data
##
## Table Type: Analysis of Variance Table (Type I)
## Model: No Repeated Measures (lm)
## Response: art(merlot2019.quality.pc)
##
##                   Df Df.res F value   Pr(>F)
## 1 treatment        1     48 1.59616 0.212550
## 2 block            5     48 2.90081 0.022858 *
## 3 treatment:block  5     48 0.86846 0.509231
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Merlot 2019 pruning PC

```
## Analysis of Variance of Aligned Rank Transformed Data
## 
## Table Type: Analysis of Variance Table (Type I) 
## Model: No Repeated Measures (lm)
## Response: art(merlot2019.pruning.pc)
## 
##                   Df Df.res F value   Pr(>F)   
## 1 treatment        1     48  1.0191 0.317784   
## 2 block            5     48  2.0081 0.094339  .
## 3 treatment:block  5     48  4.0950 0.003546 **
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Merlot 2020 growth PC

```
## Analysis of Variance of Aligned Rank Transformed Data
## 
## Table Type: Analysis of Variance Table (Type I) 
## Model: No Repeated Measures (lm)
## Response: art(merlot2020.growth.pc)
## 
##                   Df Df.res F value   Pr(>F)  
## 1 treatment        1     48 5.62031 0.021817 *
## 2 block            5     48 0.63308 0.675400  
## 3 treatment:block  5     48 2.98813 0.019904 *
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Merlot 2020 yield PC

```
## Analysis of Variance of Aligned Rank Transformed Data
## 
## Table Type: Analysis of Variance Table (Type I) 
## Model: No Repeated Measures (lm)
## Response: art(merlot2020.yield.pc)
## 
##                   Df Df.res F value  Pr(>F)
## 1 treatment        1     48 0.14007 0.70986
## 2 block            5     48 1.42597 0.23200
## 3 treatment:block  5     48 1.03306 0.40908
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Merlot 2020 quality PC

```
## Analysis of Variance of Aligned Rank Transformed Data
## 
## Table Type: Analysis of Variance Table (Type I) 
## Model: No Repeated Measures (lm)
## Response: art(merlot2020.quality.pc)
##
##                    Df Df.res F value  Pr(>F)
## 1 treatment         1     48 0.35140 0.55611
## 2 block             5     48 0.85977 0.51494
## 3 treatment:block   5     48 1.30212 0.27885
## ---
## Signif. codes:   0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

None of the treatment effects are significant. Although some of the interaction effects are. Maybe heating only helps under certain geographical conditions?

Another thing to check is the actual estimate of the treatment/blocking/interaction effects. We can also construct SEs of each of these effects.

Can also combine data from both years and do repeated measures ANOVA?