Final Project STP 530

Adam Kurth

2023-11-08

Context:

Crystallography, an interdisciplinary field primarily centered on the study of solid-state chemistry and physics, delves into various aspects of matter. A significant branch of this field, protein crystallography, and its use of X-rays, focus on unraveling the three-dimensional structure and interactions of proteins. This interaction broadly speaking leads to strides in research pertaining to drug and vaccine development. To illustrate its breadth and importance, consider the following examples:

- Investigating the interaction mechanisms between the Sars-Cov2 virus and human cells, a topic that remains somewhat enigmatic.
- Exploring how various proteins within the human body respond to pharmaceuticals, allowing scientists to see how drugs fit into protein structures, a crucial aspect of increasing drug efficiency.
- Deepening our understanding of photosynthesis by revealing the structure of photosynthetic proteins and complexes, and how these function in the energy transfer.

Potential Predictors:

- `Mean.Intensity`: refers to the *average* intensity of the diffracted X-rays. This data is in units of the counted number of photons from the diffracted light. This intensity of the diffracted X-rays is proportional to the square of the structure factor amplitude. This predictor can give insight into the overall electron density and crucial for crystal structure.
- `Mean.Phase`: Despite the nuances about the "Phase Problem" in crystallography (below if curious), each diffracted wave has a phase angle. The phase problem, central to crystallography, as phases cannot be measured. However, they're still crucial for determining the electron density map. The mean phase represents the *average* measure of these phase angles across the data set.
- `a, b, c, alpha, beta, gamma`: These are unit cell parameters which are simply the lengths (a,b,c) and the angles of the unit cell measured in degrees (alpha, beta, gamma). This is fundamental for determining the system and size of the crystal lattice.
- `Unitcell.Volume`: The volume is calculated directly from the unit cell parameters. This is an indicator of the density of the crystal structure.
- `Packing.Density`: Ratio of space within the crystal taken by the atoms and the total space. $\text{Packing.Density} = \frac{\text{Mass}}{\text{Volume}} = \frac{\text{Calc.Structure.Weight (kDa)*1000}}{\text{Unitcell.Volume}} = \frac{\text{g/mol}}{\text{Å}^3}$
- `Space Group`: The crystal's group symmetry, this is based on symmetric properties which are crucial for understanding the crystal structure. Note why these are separated is that different group symmetries imply different atomic arrangements, therefore different properties of the crystal.

- `Calc.Structure.Weight`: Computed value to represent the total mass of atoms in a molecule. Particularly in the context of protein crystals, this weight is an important characteristic because it reflects sum of the atomic weights of all the atoms present in the molecule, or repeated unit within the crystal lattice.

“The phase problem is the problem of information concerning the phase that can occur when making physical measurement [and] has to be solved for the determination of a structure from diffraction data.” (“Phase Problem” n.d.).

What is the Structure Factor?

$F(hkl) = \sum_j f_j e^{(2\pi i(hx_j+ky_j+lz_j))}$

We have calculated these values for all *hkl* values using CCP4, a commonly used crystallography program. This calculation is key for interpreting the structure characteristics of the protein. By calculating the structure factor, we gain insight into information about the diffraction pattern.

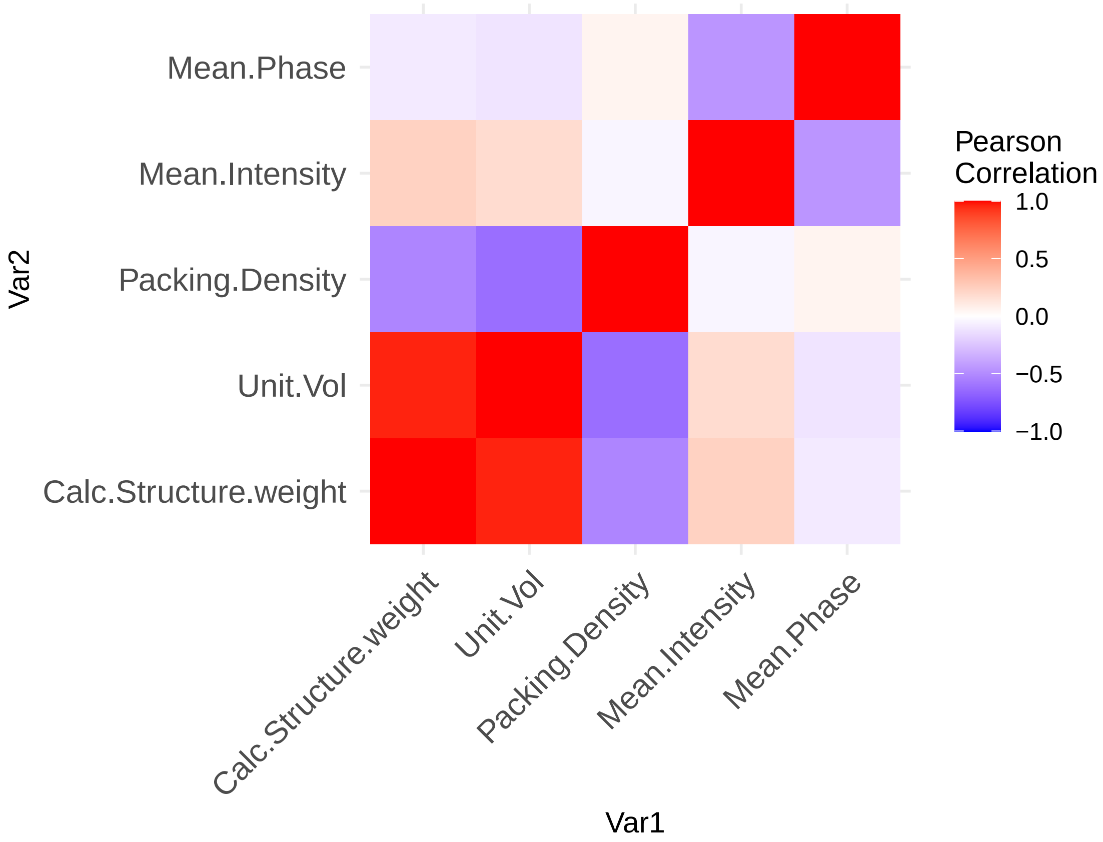
Research Question:

Let us see whether there is a linear dependence of X-ray diffraction intensities on the number of unit cells exposed to the X-rays. This further refines to the more specific, $ *within the same space group, does the packing density have any affect on the diffracted intensity?*$

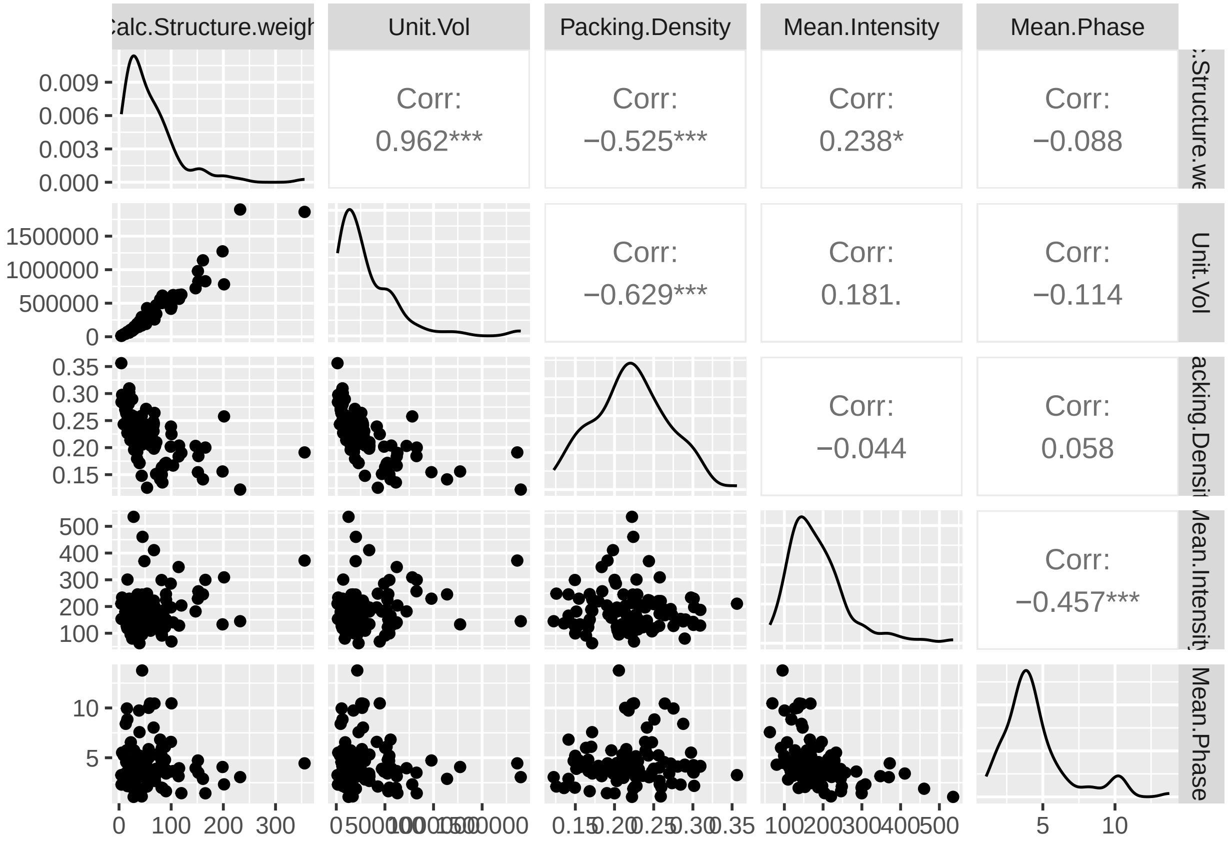
Data Exploration:

Two different ways to visualize the correlation among each numerical predcitor interested in testsing.

```r
# To focus on only on numerical values interested in testing.
columns_to_remove = c("Spacegroup.ID", "PDB.ID", "a", "b", "c", "Alpha", "Beta", "Gamma", "Max.Intensit
numeric_df1_P1211 = df1_P1211[, !(names(df1_P1211) %in% columns_to_remove)]
# Visualize correlation matrix
cor_matrix_melt <- melt(cor_matrix <- cor(numeric_df1_P1211))
ggplot(data = cor_matrix_melt, aes(Var1, Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1,1), space = "Lab",
                       name="Pearson\nCorrelation") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, size = 12, hjust = 1),
        axis.text.y = element_text(size = 12)) +
  coord_fixed()
```



```
# Using gg
ggpairs(numeric_df1_P1211, cardinality_threshold = 100)
```



From the correlation matrices, its easy to see that as unit cell volume predictors increase, so does the structure weight. This also applies to the packing density since they are proportions of each other. This makes sense, and is not interesting. The question rather, is whether the mean intensity predictor correlated with anything, specifically the packing density. As you can see that the largest Pearson's correlation in the matrix is that the predictor `Mean.Intensity` has is with `Structure.Weight` with 0.238. What we are interested in has a Pearson's correlation value of -0.044.

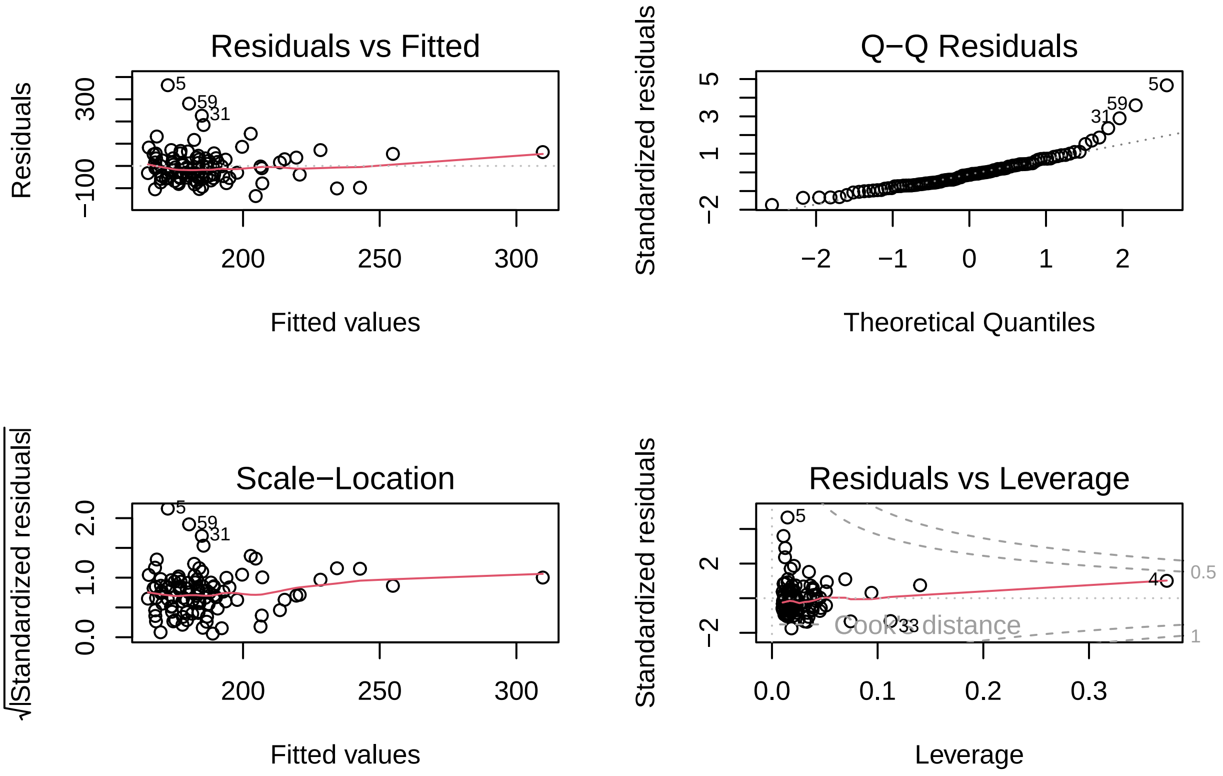
Also notice that the mean intensity and the mean phase have a seeming negative Pearson's correlation. The observed diffraction pattern is a result of the interference of X-rays scattered by electrons in the crystal. Of the diffracted waves, the interference pattern is governed by the intensity and the phase. Hence, there is interplay between when one varies significantly, then this can lead to a destructive interference pattern.

Another plausible explanation is that certain crystal structures may lead to destructive interference pattern due to the phase angle distributions, resulting in lower intensity.

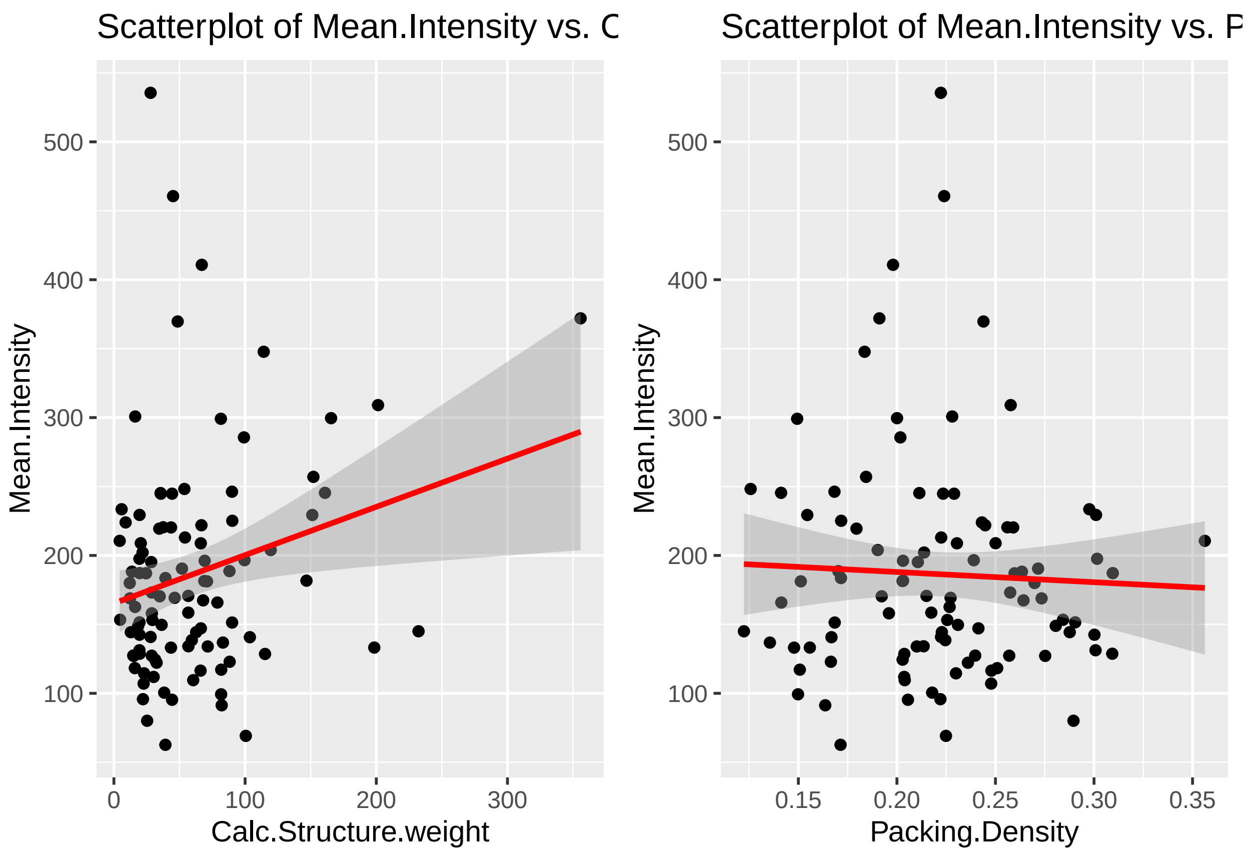
Model:

$$E\{\text{Mean.Intensity}\}_{P1211} = \beta_0 + \beta_1(\text{Calc.Structure.weight}) + \beta_2(\text{Packing.Density})$$

```r
########## MODEL for P1211
# ----------------------------------------------------------------------
m.P1211 = lm(Mean.Intensity ~ Calc.Structure.weight + Packing.Density, data = df1_P1211)
par(mfrow=c(2,2))
plot(m.P1211)
```

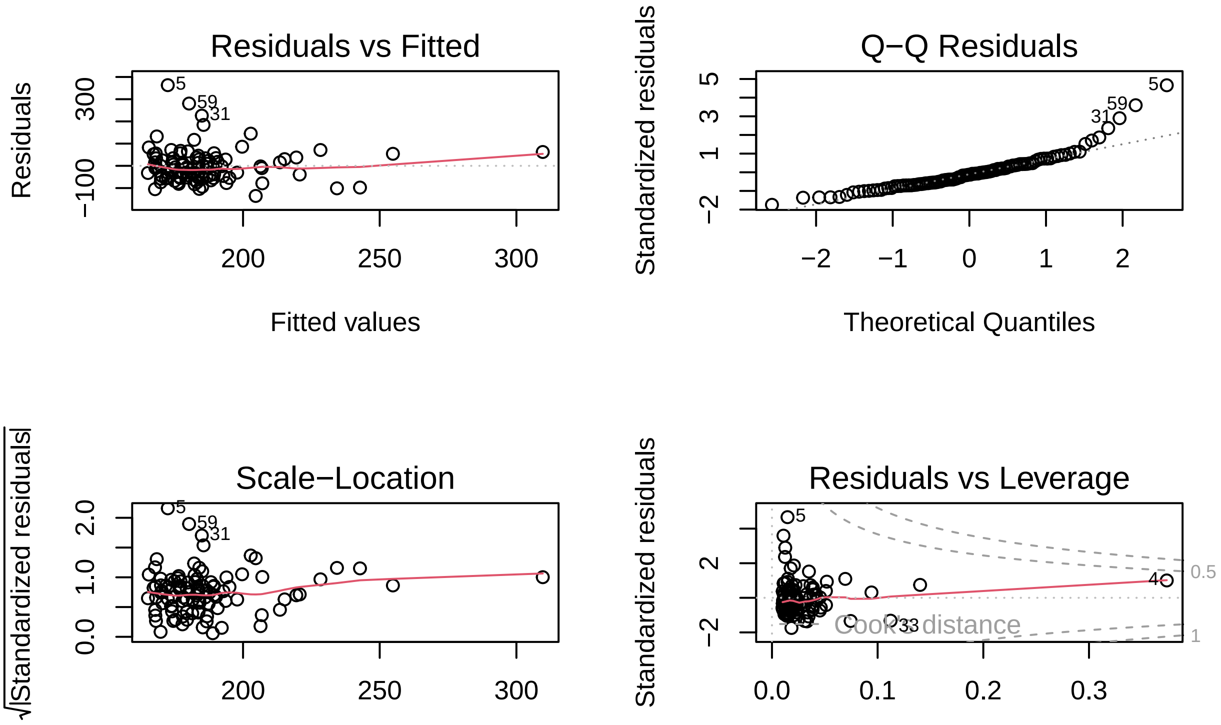


```
par(mfrow=c(1,1))
########## Diagnostics
# ------------------------------------------------------------------------
# Scatterplot of Calc.Structure.weight vs. Mean.Intensity with regression line
a = ggplot(data = df1_P1211, aes(x = Calc.Structure.weight, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Calc.Structure.weight", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Calc.Structure.weight")

# Scatterplot of Packing.Density vs. Mean.Intensity with regression line
b = ggplot(data = df1_P1211, aes(x = Packing.Density, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Packing.Density", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Packing.Density")
ggarrange(a,b)
```



```
# This does not show linear relationship off the bat.
```

```
par(mfrow=c(2,2))
plot(m.P1211)
```



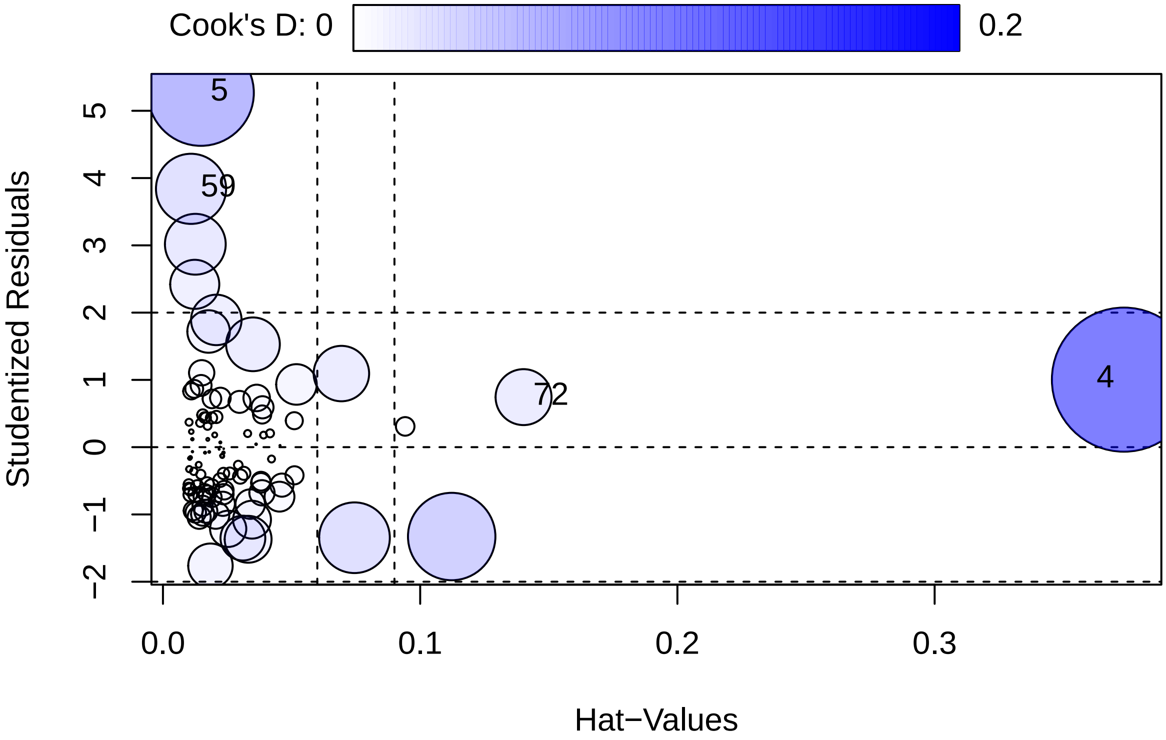
```
par(mfrow=c(1,1))
```

```
assumption.5.check.independence(m.P1211)
```

```
##
## Assumption 5: Checking for independence of errors...

##  lag Autocorrelation D-W Statistic p-value
##    1      0.05683202      1.879978   0.516
##  Alternative hypothesis: rho != 0
```

```
# Visualize outliers, leverage points, and influential points
```

```
car::influencePlot(m.P1211,id=list(labels=row.names(df1_P1211)))
```



```
##      StudRes        Hat      CookD
## 4  1.0039412 0.37359400 0.20035672
## 5  5.2666921 0.01473435 0.10839285
## 59 3.8401212 0.01091054 0.04749203
## 72 0.7437921 0.14018328 0.03020488
```

After checking the assumptions of multiple regression, this model passes except for potentially having a few outliers. As indicated on the scale-location plot, and qqplot.

Assumption 5: Cook's Distance:

Cook's Distance is a measure to estimate the influence of all the data points in multiple regression. This helps point out outliers with a strong influence on the data. Since the assumption 5 check, used the `cooks.distance()` function, and gave an autocorrelation of 0.05725871, this means that there's a low correlation between residuals. Thus, the residuals are independent.

D-W Test Statistic: Autocorrelation is the measure usually used in time series data which detects whether there is any correlation between the series of values.

This test statistic tests that the null hypothesis has no autocorrelation among the residuals. With a range of 0 to 4 for this particular test statistic shows a value very close to 2. When 4 means that there is positive autocorrelation, and 0 indicates that there's negative autocorrelation. Thus, there's little to no autocorrelation among the residuals in this model, since they're independent.

P-Value: Under the null hypothesis, the p-value states that there is no autocorrelation. With a p-value typically above 0.05, we fail to reject the null hypothesis. This does not give enough evidence to reject the null hypothesis. With a p-value of 0.496, this model indicates that there is strong statistical evidence of the contrary.

Influence Plot: As the influence plot shows, there is some high leverage points in this data.

```
assumption.6.check.multicollinearity(m.P1211)
```

```
##
## Assumption 6: Checking for multicollinearity...
## Calc.Structure.weight       Packing.Density
##              1.380975              1.380975
```

Assumption 6: Multicolinearity

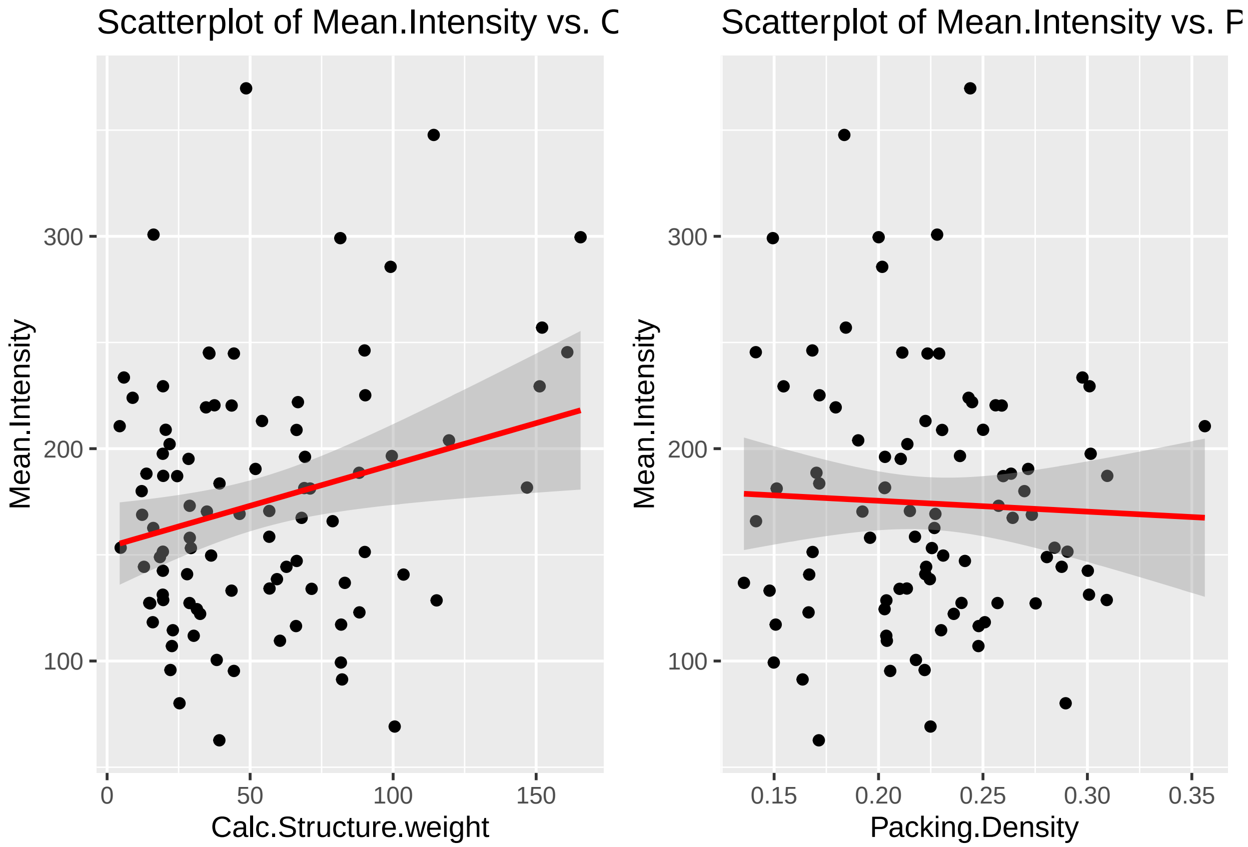
With a VIF score of around 1, this suggests that there is very low multicolinearity in the model. We can proceed with this model that there are low correlation between the multiple independent variables in the model. Since these show VIF values of roughly around 1, these are perfectly colinear.

Proceeding to see whether removing high influence points, improves the model fit.

```
########## Treating outliers.
# ------------------------------------------------------------------------
cooks.d = cooks.distance(m.P1211)
influence = cooks.d[(cooks.d > (3*mean(cooks.d, na.rm = T)))]
influence # see that most of the rows stated above are in this list
```

```
##          4          5         31         33         59         63         72
## 0.20035672 0.10839285 0.03566844 0.07377912 0.04749203 0.02967693 0.03020488
##         77
## 0.04820676
```

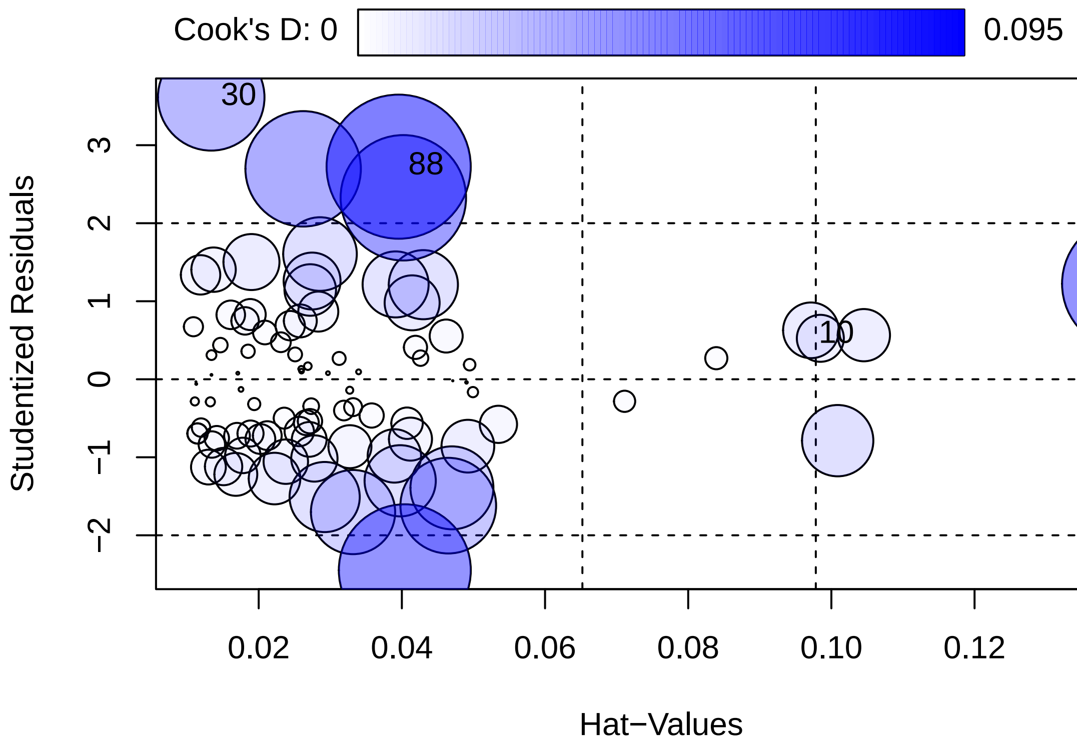
```
row.influence = names(influence)
outliers = df1_P1211[row.influence,]
df1_P1211.reduced = df1_P1211 %>% anti_join(outliers)
########## Any improvement to the model?
# ------------------------------------------------------------------------
m.P1211.R = lm(Mean.Intensity ~ Calc.Structure.weight + Packing.Density, data = df1_P1211.reduced)
# ------------------------------------------------------------------------
# Scatterplot of Calc.Structure.weight vs. Mean.Intensity with regression line
a = ggplot(data = df1_P1211.reduced, aes(x = Calc.Structure.weight, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Calc.Structure.weight", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Calc.Structure.weight")

# Scatterplot of Packing.Density vs. Mean.Intensity with regression line
b = ggplot(data = df1_P1211.reduced, aes(x = Packing.Density, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Packing.Density", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Packing.Density")
ggarrange(a,b)
```



```
# ----------------------------------------------------------------------------
vif(m.P1211.R) #higher but still insignificant
```

```
## Calc.Structure.weight       Packing.Density
##              1.698754              1.698754
```

```
car::influencePlot(m.P1211.R,id=list(labels=row.names(df1_P1211.reduced))) # did not help much
```



```
##       StudRes        Hat      CookD
## 10 0.5649939 0.10453303 0.01251714
## 30 3.6129660 0.01336559 0.05191308
## 78 1.2237681 0.14157348 0.08187180
## 88 2.7245624 0.03955091 0.09503646
```

```
# ------------------------------------------------------------------------
summary(m.P1211.R)
```

```
##
## Call:
## lm(formula = Mean.Intensity ~ Calc.Structure.weight + Packing.Density,
##     data = df1_P1211.reduced)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -133.591  -41.074   -2.865   31.839  192.824
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)              84.5062    45.5923   1.854  0.06712 .
## Calc.Structure.weight     0.5936     0.2046   2.901  0.00469 **
## Packing.Density         260.3335   167.3818   1.555  0.12342
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 57.25 on 89 degrees of freedom
## Multiple R-squared:  0.08787,    Adjusted R-squared:  0.06737
## F-statistic: 4.287 on 2 and 89 DF,  p-value: 0.01669
```

```r
r.a.2 = summary(m.P1211)$adj.r.squared
r.a.2.reduced = summary(m.P1211.R)$adj.r.squared
r.a.2.reduced - r.a.2
```

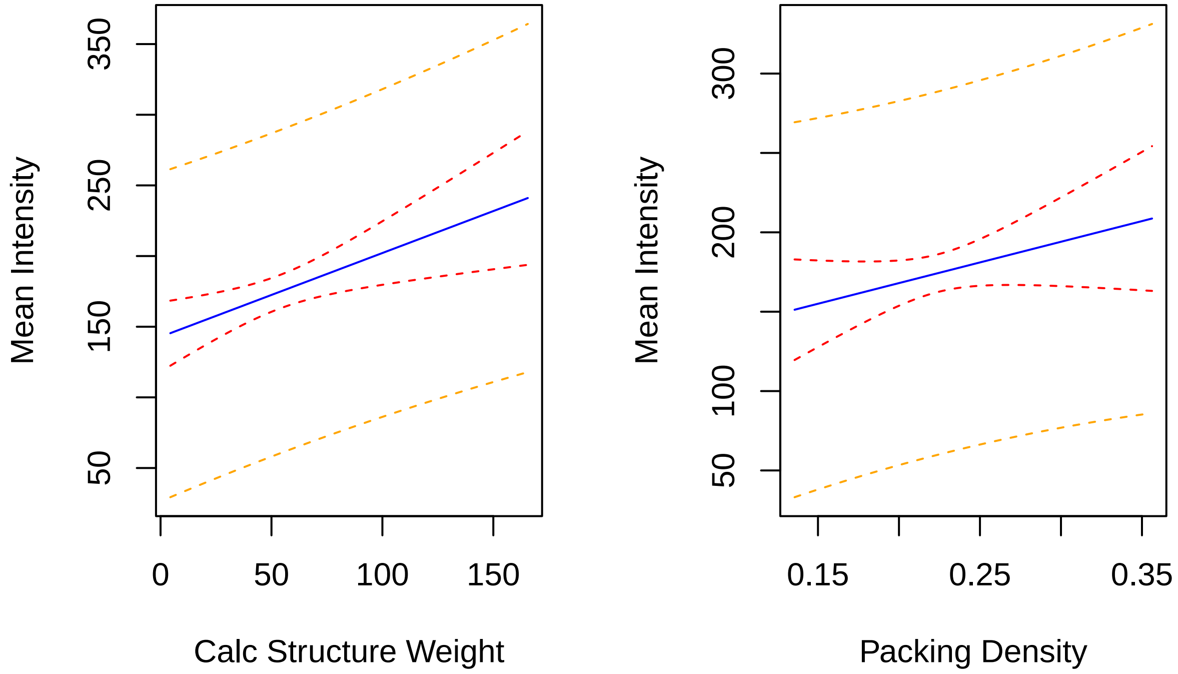
```
## [1] 0.02097751
```

```r
# -------------------------------------------------------------------------
```

Confidence/Prediction Interval Plots

```r
par(mfrow=c(1,2))
# -------------------------------------------------------------------------
# For Calc.Structure.weight
calc_structure_weight_mean <- mean(df1_P1211.reduced$Calc.Structure.weight, na.rm = TRUE)
calc_structure_weight_range <- seq(from = min(df1_P1211.reduced$Calc.Structure.weight),
                                   to = max(df1_P1211.reduced$Calc.Structure.weight), length.out = 100)
packing_density_mean <- mean(df1_P1211.reduced$Packing.Density, na.rm = TRUE)

x_new <- data.frame(Calc.Structure.weight = calc_structure_weight_range,
                    Packing.Density = rep(packing_density_mean, 100))

conf.int.1 <- predict(m.P1211.R, newdata = x_new, interval = "confidence")
pred.int.1 <- predict(m.P1211.R, newdata = x_new, interval = "prediction")

# Calculate custom y-axis limits to ensure both intervals are visible
y_limits_1 <- c(min(min(conf.int.1[, "lwr"]), min(pred.int.1[, "lwr"])),
                max(max(conf.int.1[, "upr"]), max(pred.int.1[, "upr"])))

# Plot the first confidence/prediction interval
plot(x_new$Calc.Structure.weight, conf.int.1[, "fit"], type = "l", col = "blue",
     ylim = y_limits_1,
     ylab = "Mean Intensity", xlab = "Calc Structure Weight")
lines(x_new$Calc.Structure.weight, conf.int.1[, "lwr"], col = "red", lty = 2)
lines(x_new$Calc.Structure.weight, conf.int.1[, "upr"], col = "red", lty = 2)

lines(x_new$Calc.Structure.weight, pred.int.1[, "lwr"], col = "orange", lty = 2)
lines(x_new$Calc.Structure.weight, pred.int.1[, "upr"], col = "orange", lty = 2)
# -------------------------------------------------------------------------
# For Packing.Density
packing_density_range <- seq(from = min(df1_P1211.reduced$Packing.Density),
                             to = max(df1_P1211.reduced$Packing.Density), length.out = 100)

x_new <- data.frame(Packing.Density = packing_density_range,
                    Calc.Structure.weight = rep(calc_structure_weight_mean, 100))

conf.int.2 <- predict(m.P1211.R, newdata = x_new, interval = "confidence")
```

```r
pred.int.2 <- predict(m.P1211.R, newdata = x_new, interval = "prediction")

# Calculate custom y-axis limits for the second plot
y_limits_2 <- c(min(min(conf.int.2[, "lwr"]), min(pred.int.2[, "lwr"])),
                max(max(conf.int.2[, "upr"]), max(pred.int.2[, "upr"])))

# Plot the second confidence/prediction interval
plot(x_new$Packing.Density, conf.int.2[, "fit"], type = "l", col = "blue",
     ylim = y_limits_2,
     ylab = "Mean Intensity", xlab = "Packing Density")
lines(x_new$Packing.Density, conf.int.2[, "lwr"], col = "red", lty = 2)
lines(x_new$Packing.Density, conf.int.2[, "upr"], col = "red", lty = 2)

lines(x_new$Packing.Density, pred.int.2[, "lwr"], col = "orange", lty = 2)
lines(x_new$Packing.Density, pred.int.2[, "upr"], col = "orange", lty = 2)
```



```r
# ----------------------------------------------------------------------
par(mfrow=c(1,1))
confint(m.P1211.R)
```

```
##                             2.5 %     97.5 %
## (Intercept)            -6.0847278 175.097147
## Calc.Structure.weight   0.1870051   1.000113
## Packing.Density       -72.2505876 592.917634
```

```r
summary(m.P1211.R)
```

```
##
## Call:
## lm(formula = Mean.Intensity ~ Calc.Structure.weight + Packing.Density,
##     data = df1_P1211.reduced)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -133.591  -41.074   -2.865   31.839  192.824
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)             84.5062    45.5923   1.854  0.06712 .
## Calc.Structure.weight    0.5936     0.2046   2.901  0.00469 **
## Packing.Density        260.3335   167.3818   1.555  0.12342
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 57.25 on 89 degrees of freedom
## Multiple R-squared:  0.08787,    Adjusted R-squared:  0.06737
## F-statistic: 4.287 on 2 and 89 DF,  p-value: 0.01669
```

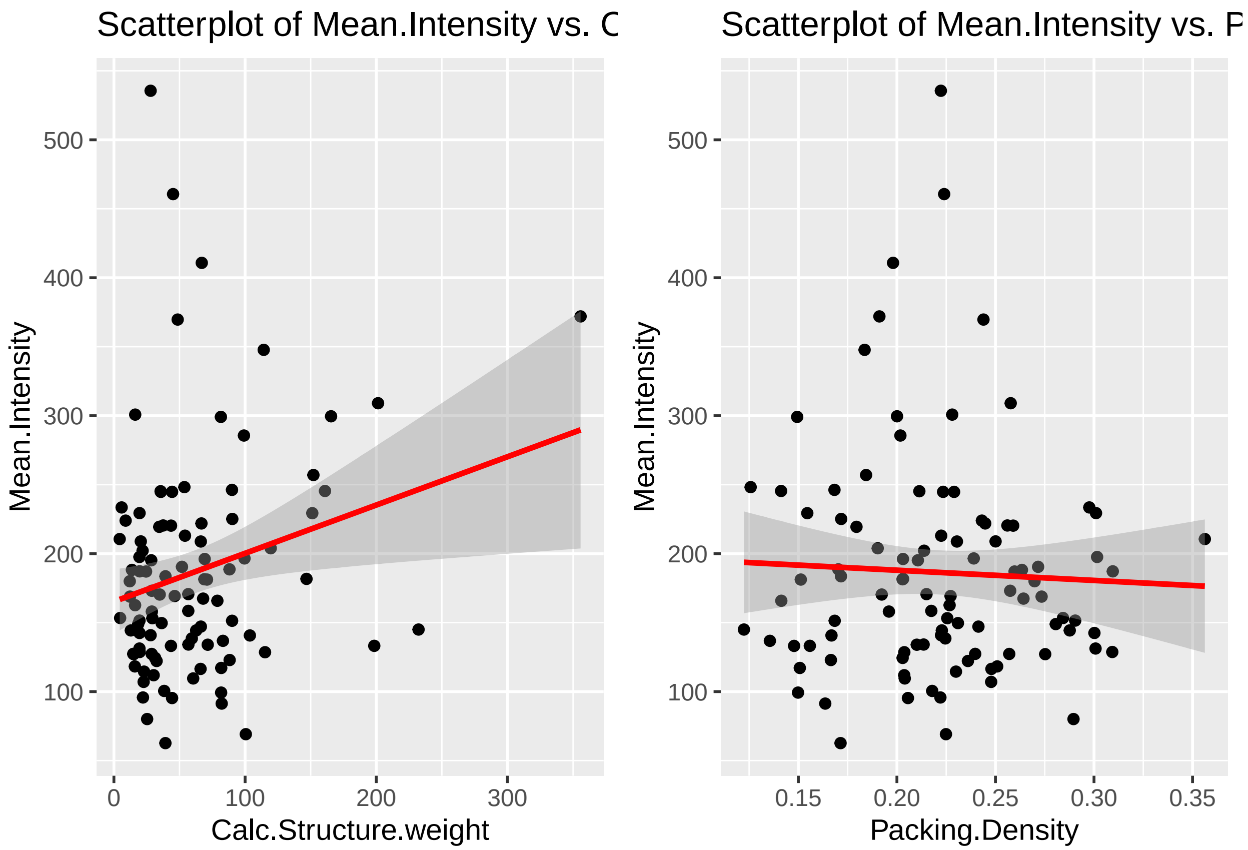
By reducing the data set for this space group, there’s an observed negative slope. When the packing density increases by 1-unit, there’s a corresponding increase in the average number of counted photons by 260.3335, holding all other predictors constant.

The confidence interval plots, and summary show that the linear relation between mean intensity and the packing density varies widely. Note that this interval’s lower bound is negative, and the upper bound is positive. Thus, the slope could be 0.

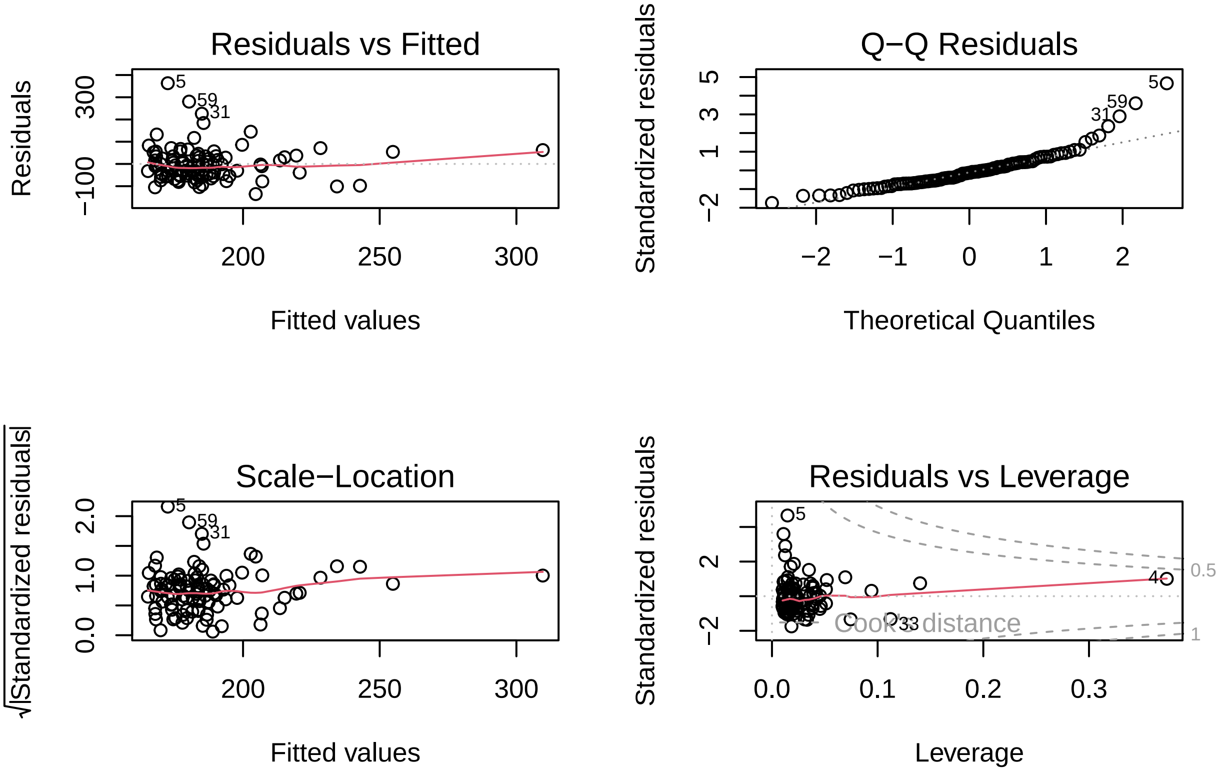
Before testing whether the packing ratio can be dropped from the model, other space groups should be investigated.

For P121 Space Group

```r
########## MODEL for P121
# -------------------------------------------------------------------------
m.P121 = lm(Mean.Intensity ~ Calc.Structure.weight + Packing.Density, data = df1_P121)
########## Diagnostics
# -------------------------------------------------------------------------
# Scatterplot of Calc.Structure.weight vs. Mean.Intensity with regression line
a = ggplot(data = df1_P121, aes(x = Calc.Structure.weight, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Calc.Structure.weight", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Calc.Structure.weight")

# Scatterplot of Packing.Density vs. Mean.Intensity with regression line
b = ggplot(data = df1_P121, aes(x = Packing.Density, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Packing.Density", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Packing.Density")
ggarrange(a,b)
```



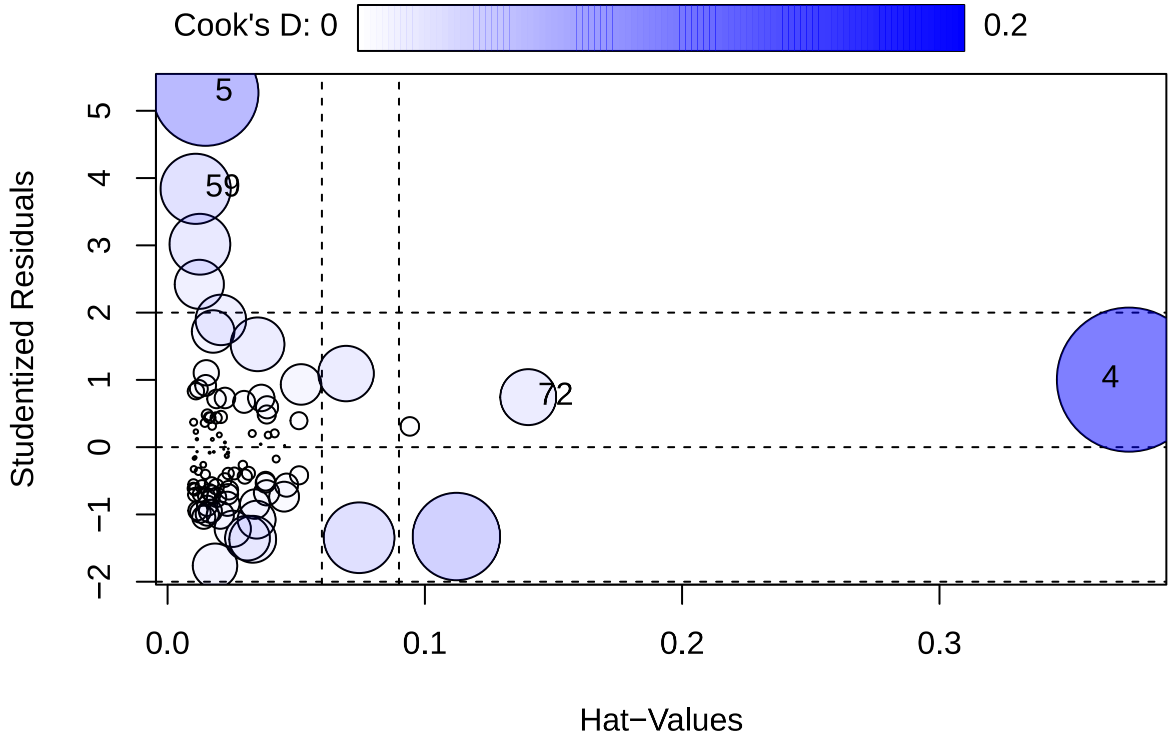
```
# This does not show linear relationship off the bat.
par(mfrow=c(2,2))
plot(m.P1211)
```



```
par(mfrow=c(1,1))

assumption.5.check.independence(m.P121)
```

```
##
## Assumption 5: Checking for independence of errors...


##  lag Autocorrelation D-W Statistic p-value
##    1      0.05683202      1.879978   0.526
##  Alternative hypothesis: rho != 0
```

```
# Visualize outliers, leverage points, and influential points
car::influencePlot(m.P121,id=list(labels=row.names(df1_P121)))
```



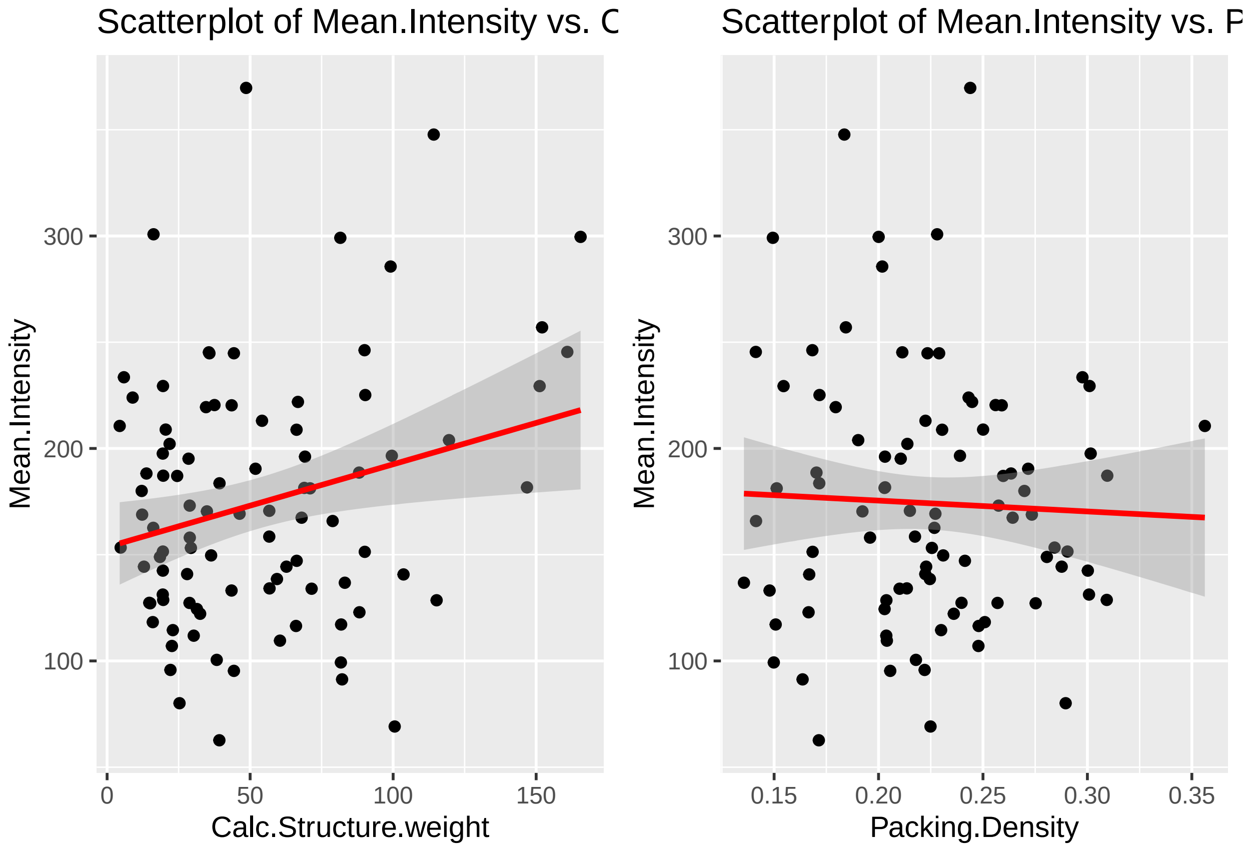
```
##      StudRes        Hat      CookD
## 4  1.0039412 0.37359400 0.20035672
## 5  5.2666921 0.01473435 0.10839285
## 59 3.8401212 0.01091054 0.04749203
## 72 0.7437921 0.14018328 0.03020488
```

```
# Outliers detected rows: 4, 53, 59, 5
########## Treating outliers.
# ---------------------------------------------------------------------------
cooks.d = cooks.distance(m.P1211)
influence = cooks.d[(cooks.d > (3*mean(cooks.d, na.rm = T)))]
influence # see that most of the rows stated above are in this list
```

```
##          4          5         31         33         59         63         72
## 0.20035672 0.10839285 0.03566844 0.07377912 0.04749203 0.02967693 0.03020488
##         77
## 0.04820676
```

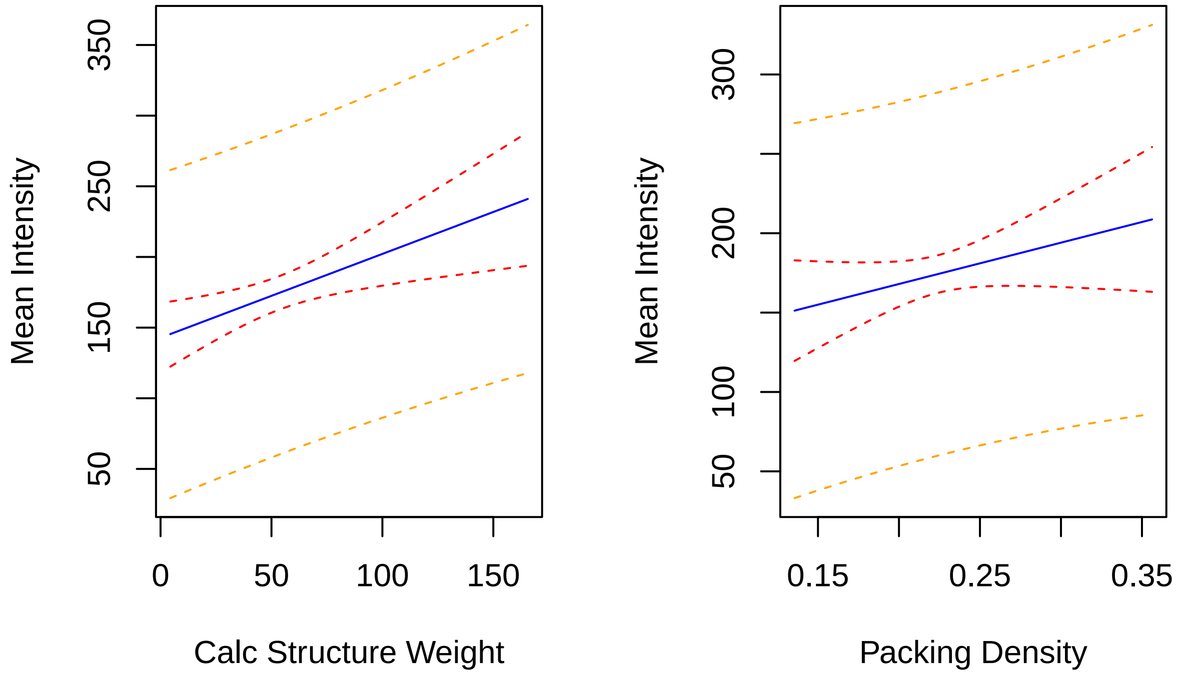
```
row.influence = names(influence)
outliers = df1_P121[row.influence,]
df1_P121.reduced = df1_P121 %>% anti_join(outliers)
########## Any improvement to the model?
# ---------------------------------------------------------------------------
m.P121.R = lm(Mean.Intensity ~ Calc.Structure.weight + Packing.Density, data = df1_P121.reduced)
# ---------------------------------------------------------------------------
```

```r
# Scatterplot of Calc.Structure.weight vs. Mean.Intensity with regression line
a = ggplot(data = df1_P121.reduced, aes(x = Calc.Structure.weight, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Calc.Structure.weight", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Calc.Structure.weight")

# Scatterplot of Packing.Density vs. Mean.Intensity with regression line
b =ggplot(data = df1_P121.reduced, aes(x = Packing.Density, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Packing.Density", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Packing.Density")
ggarrange(a,b)
```



```r
# ---------------------------------------------------------------------------
# Set up the plotting environment for two plots side by side
par(mfrow=c(1,2))
# ---------------------------------------------------------------------------
# For Calc.Structure.weight
calc_structure_weight_mean <- mean(df1_P121.reduced$Calc.Structure.weight, na.rm = TRUE)
calc_structure_weight_range <- seq(from = min(df1_P121.reduced$Calc.Structure.weight),
                                   to = max(df1_P121.reduced$Calc.Structure.weight), length.out = 100)
vol_unit_ratio_mean <- mean(df1_P121.reduced$Packing.Density, na.rm = TRUE)
```

```r
x_new <- data.frame(Calc.Structure.weight = calc_structure_weight_range,
                    Packing.Density = rep(vol_unit_ratio_mean, 100))

conf.int.1 <- predict(m.P121.R, newdata = x_new, interval = "confidence")
pred.int.1 <- predict(m.P121.R, newdata = x_new, interval = "prediction")

# Calculate custom y-axis limits to ensure both intervals are visible
y_limits_1 <- c(min(min(conf.int.1[, "lwr"]), min(pred.int.1[, "lwr"])),
                max(max(conf.int.1[, "upr"]), max(pred.int.1[, "upr"])))

# Plot the first confidence/prediction interval
plot(x_new$Calc.Structure.weight, conf.int.1[, "fit"], type = "l", col = "blue",
     ylim = y_limits_1,
     ylab = "Mean Intensity", xlab = "Calc Structure Weight")
lines(x_new$Calc.Structure.weight, conf.int.1[, "lwr"], col = "red", lty = 2)
lines(x_new$Calc.Structure.weight, conf.int.1[, "upr"], col = "red", lty = 2)

lines(x_new$Calc.Structure.weight, pred.int.1[, "lwr"], col = "orange", lty = 2)
lines(x_new$Calc.Structure.weight, pred.int.1[, "upr"], col = "orange", lty = 2)
# -------------------------------------------------------------------------
# For Packing.Density
vol_unit_ratio_range <- seq(from = min(df1_P121.reduced$Packing.Density),
                            to = max(df1_P121.reduced$Packing.Density), length.out = 100)

x_new <- data.frame(Packing.Density = vol_unit_ratio_range,
                    Calc.Structure.weight = rep(calc_structure_weight_mean, 100))

conf.int.2 <- predict(m.P121.R, newdata = x_new, interval = "confidence")
pred.int.2 <- predict(m.P121.R, newdata = x_new, interval = "prediction")

# Calculate custom y-axis limits for the second plot
y_limits_2 <- c(min(min(conf.int.2[, "lwr"]), min(pred.int.2[, "lwr"])),
                max(max(conf.int.2[, "upr"]), max(pred.int.2[, "upr"])))

# Plot the second confidence/prediction interval
plot(x_new$Packing.Density, conf.int.2[, "fit"], type = "l", col = "blue",
     ylim = y_limits_2,
     ylab = "Mean Intensity", xlab = "Packing Density")
lines(x_new$Packing.Density, conf.int.2[, "lwr"], col = "red", lty = 2)
lines(x_new$Packing.Density, conf.int.2[, "upr"], col = "red", lty = 2)

lines(x_new$Packing.Density, pred.int.2[, "lwr"], col = "orange", lty = 2)
lines(x_new$Packing.Density, pred.int.2[, "upr"], col = "orange", lty = 2)
```



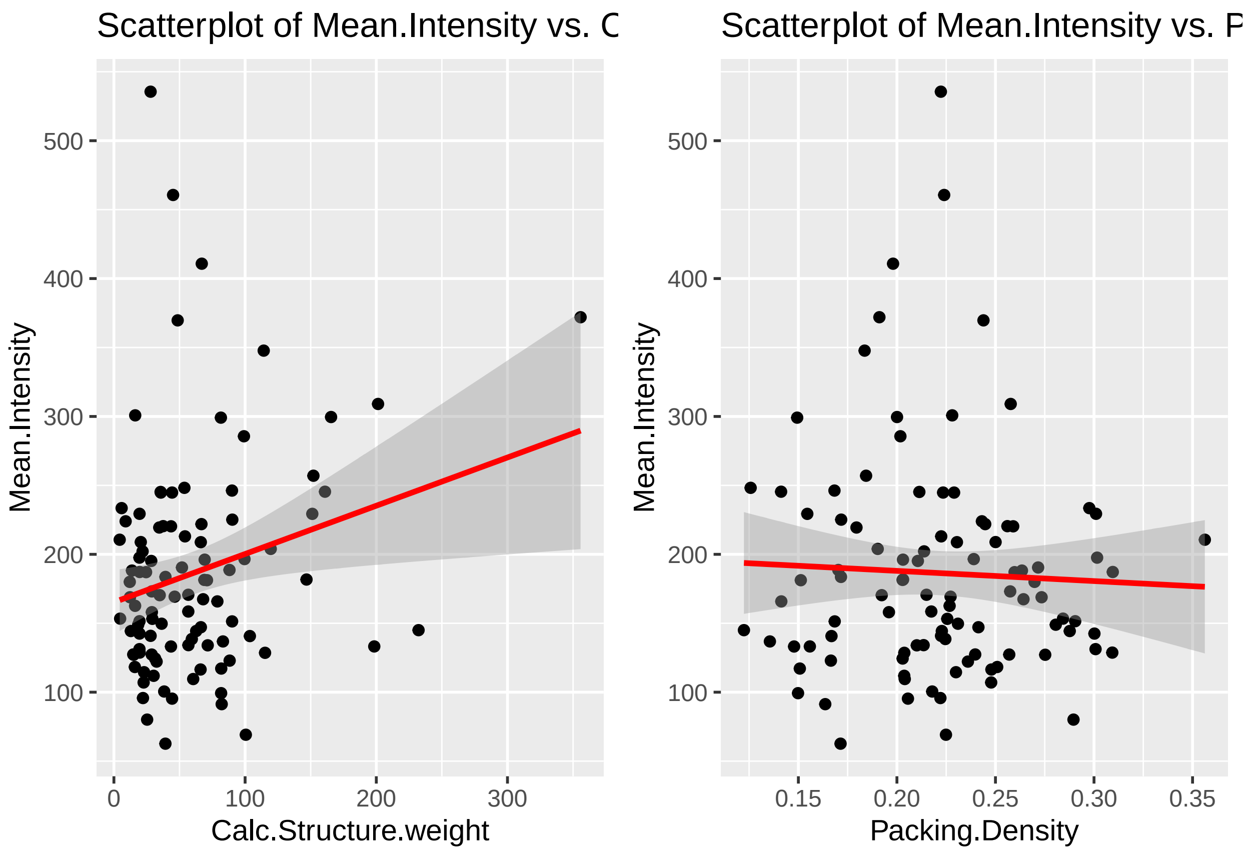
```
# ---------------------------------------------------------------------------
par(mfrow=c(1,1))
confint(m.P121.R)
```

```
##                              2.5 %     97.5 %
## (Intercept)             -6.0847278 175.097147
## Calc.Structure.weight    0.1870051   1.000113
## Packing.Density        -72.2505876 592.917634
```

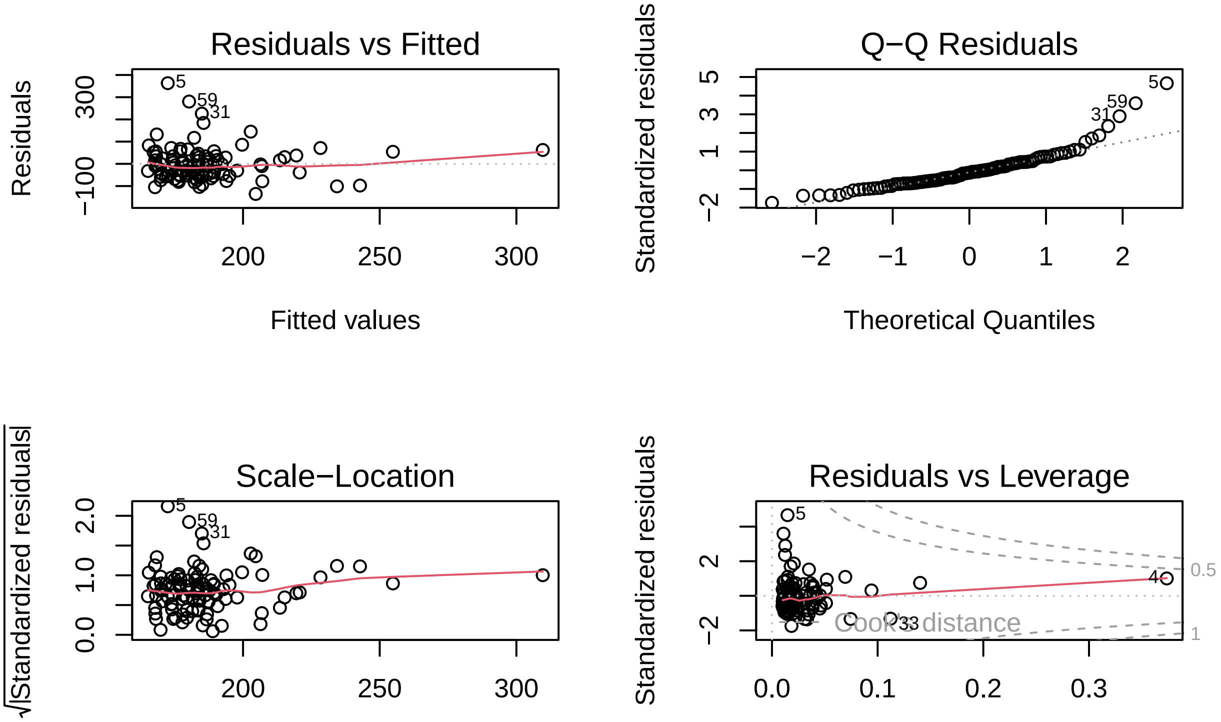
For C121 Space Group

```
########## MODEL for C121
# ---------------------------------------------------------------------------
m.C121 = lm(Mean.Intensity ~ Calc.Structure.weight + Packing.Density, data = df1_C121)
########## Diagnostics
# ---------------------------------------------------------------------------
# Scatterplot of Calc.Structure.weight vs. Mean.Intensity with regression line
a = ggplot(data = df1_C121, aes(x = Calc.Structure.weight, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Calc.Structure.weight", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Calc.Structure.weight")
```

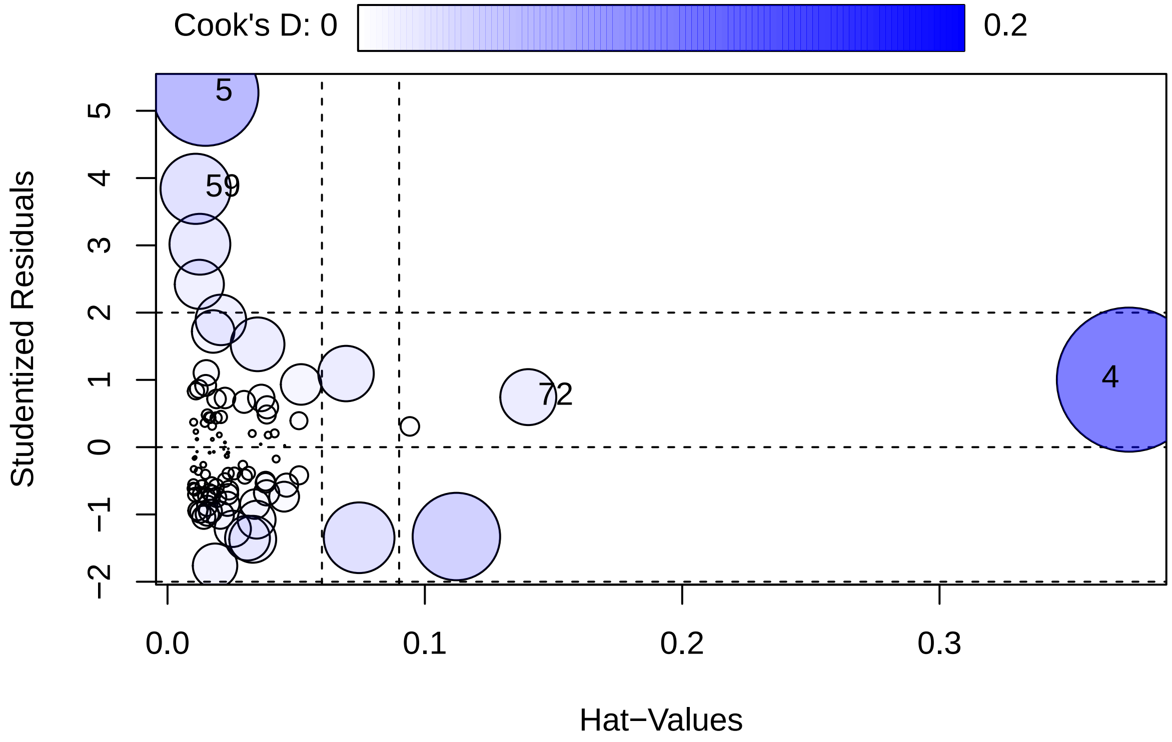
```
# Scatterplot of Packing.Density vs. Mean.Intensity with regression line
b = ggplot(data = df1_C121, aes(x = Packing.Density, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Packing.Density", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Packing.Density")
ggarrange(a,b)
```



```
# This does not show linear relationship off the bat.
# individually
par(mfrow=c(2,2))
plot(m.C121)
```



```r
par(mfrow=c(1,1))
# Visualize outliers, leverage points, and influential points
car::influencePlot(m.C121,id=list(labels=row.names(df1_C121)))
```



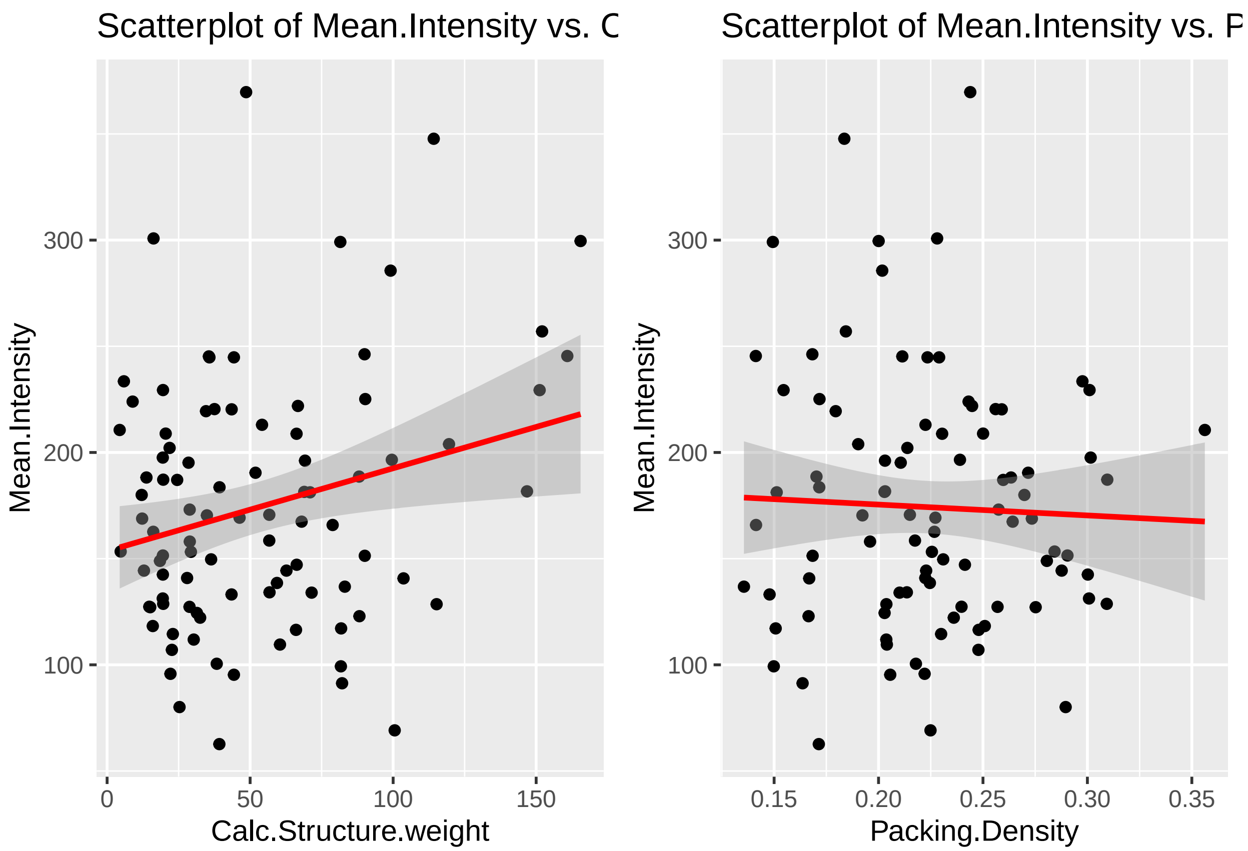
```
##      StudRes        Hat      CookD
## 4  1.0039412 0.37359400 0.20035672
## 5  5.2666921 0.01473435 0.10839285
## 59 3.8401212 0.01091054 0.04749203
## 72 0.7437921 0.14018328 0.03020488
```

```
# Outliers detected rows: 4, 53, 59, 5
########## Treating outliers.
# ----------------------------------------------------------------------
cooks.d = cooks.distance(m.C121)
influence = cooks.d[(cooks.d > (3*mean(cooks.d, na.rm = T)))]
influence # see that most of the rows stated above are in this list
```

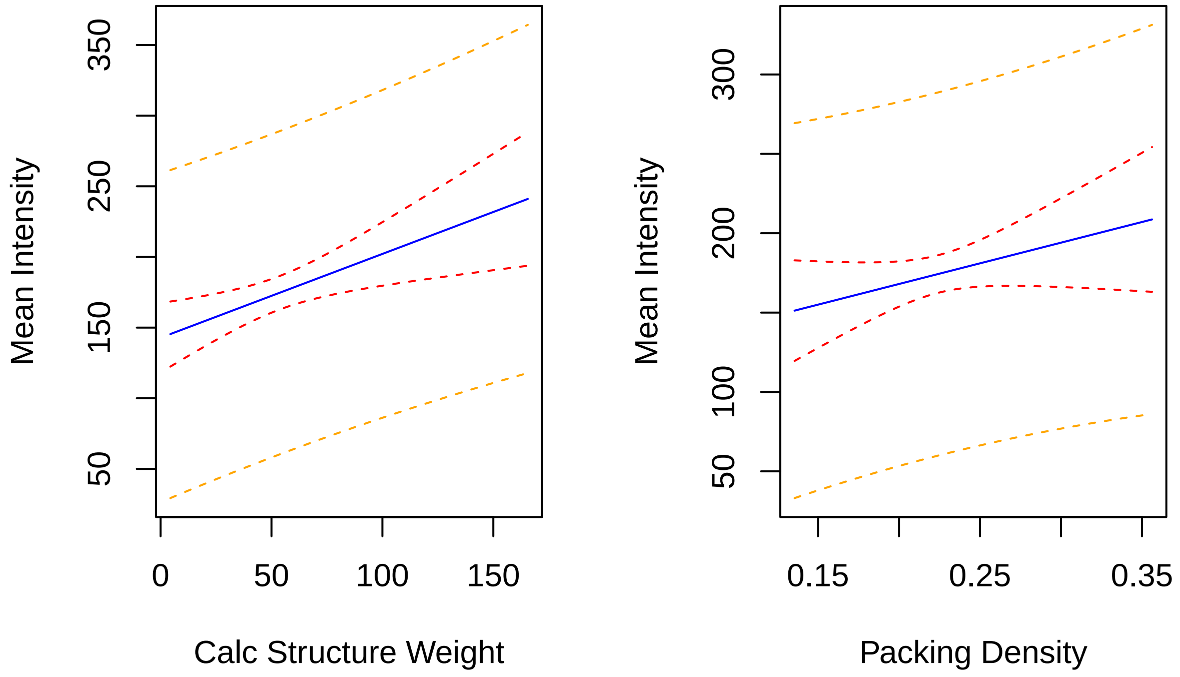
```
##          4          5         31         33         59         63         72
## 0.20035672 0.10839285 0.03566844 0.07377912 0.04749203 0.02967693 0.03020488
##         77
## 0.04820676
```

```
row.influence = names(influence)
outliers = df1_C121[row.influence,]
df1_C121.reduced = df1_C121 %>% anti_join(outliers)
########## Any improvement to the model?
# ----------------------------------------------------------------------
m.C121.R = lm(Mean.Intensity ~ Calc.Structure.weight + Packing.Density, data = df1_C121.reduced)
# ----------------------------------------------------------------------
```

```r
# Scatterplot of Calc.Structure.weight vs. Mean.Intensity with regression line
a = ggplot(data = df1_C121.reduced, aes(x = Calc.Structure.weight, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Calc.Structure.weight", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Calc.Structure.weight")

# Scatterplot of Packing.Density vs. Mean.Intensity with regression line
b = ggplot(data = df1_C121.reduced, aes(x = Packing.Density, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Packing.Density", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Packing.Density")
ggarrange(a,b)
```



```r
# ----------------------------------------------------------------------
par(mfrow=c(1,2))
# ----------------------------------------------------------------------
# For Calc.Structure.weight
calc_structure_weight_mean <- mean(df1_C121.reduced$Calc.Structure.weight, na.rm = TRUE)
calc_structure_weight_range <- seq(from = min(df1_C121.reduced$Calc.Structure.weight),
                                   to = max(df1_C121.reduced$Calc.Structure.weight), length.out = 100)
vol_unit_ratio_mean <- mean(df1_C121.reduced$Packing.Density, na.rm = TRUE)

x_new <- data.frame(Calc.Structure.weight = calc_structure_weight_range,
                    Packing.Density = rep(vol_unit_ratio_mean, 100))

conf.int.1 <- predict(m.C121.R, newdata = x_new, interval = "confidence")
pred.int.1 <- predict(m.C121.R, newdata = x_new, interval = "prediction")

# Calculate custom y-axis limits for the first plot
y_limits_1 <- c(min(min(conf.int.1[, "lwr"]), min(pred.int.1[, "lwr"])),
                max(max(conf.int.1[, "upr"]), max(pred.int.1[, "upr"])))

# Plot the first confidence interval
plot(x_new$Calc.Structure.weight, conf.int.1[, "fit"], type = "l", col = "blue",
     ylim = y_limits_1,
     ylab = "Mean Intensity", xlab = "Calc Structure Weight")
lines(x_new$Calc.Structure.weight, conf.int.1[, "lwr"], col = "red", lty = 2)
lines(x_new$Calc.Structure.weight, conf.int.1[, "upr"], col = "red", lty = 2)

# Add lines for the lower and upper bounds of the prediction interval in orange
lines(x_new$Calc.Structure.weight, pred.int.1[, "lwr"], col = "orange", lty = 2)
lines(x_new$Calc.Structure.weight, pred.int.1[, "upr"], col = "orange", lty = 2)
# -------------------------------------------------------------------------
# For Packing.Density
vol_unit_ratio_range <- seq(from = min(df1_C121.reduced$Packing.Density),
                            to = max(df1_C121.reduced$Packing.Density), length.out = 100)

x_new <- data.frame(Packing.Density = vol_unit_ratio_range,
                    Calc.Structure.weight = rep(calc_structure_weight_mean, 100))

conf.int.2 <- predict(m.C121.R, newdata = x_new, interval = "confidence")
pred.int.2 <- predict(m.C121.R, newdata = x_new, interval = "prediction")

# Calculate custom y-axis limits for the second plot
y_limits_2 <- c(min(min(conf.int.2[, "lwr"]), min(pred.int.2[, "lwr"])),
                max(max(conf.int.2[, "upr"]), max(pred.int.2[, "upr"])))

# Plot the second confidence interval
plot(x_new$Packing.Density, conf.int.2[, "fit"], type = "l", col = "blue",
     ylim = y_limits_2,
     ylab = "Mean Intensity", xlab = "Packing Density")
lines(x_new$Packing.Density, conf.int.2[, "lwr"], col = "red", lty = 2)
lines(x_new$Packing.Density, conf.int.2[, "upr"], col = "red", lty = 2)

# Add lines for the lower and upper bounds of the prediction interval in orange
lines(x_new$Packing.Density, pred.int.2[, "lwr"], col = "orange", lty = 2)
lines(x_new$Packing.Density, pred.int.2[, "upr"], col = "orange", lty = 2)
```



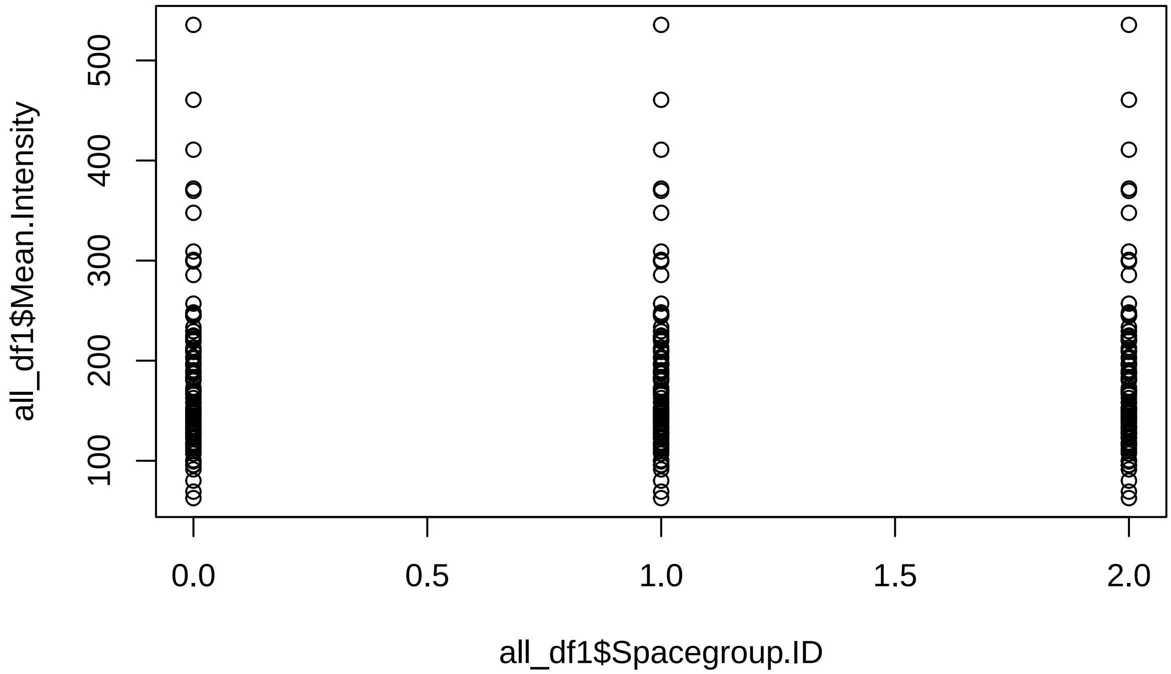
```
# ------------------------------------------------------------------------
par(mfrow=c(1,1))
confint(m.C121.R)
```

```
##                              2.5 %     97.5 %
## (Intercept)             -6.0847278 175.097147
## Calc.Structure.weight    0.1870051   1.000113
## Packing.Density        -72.2505876 592.917634
```

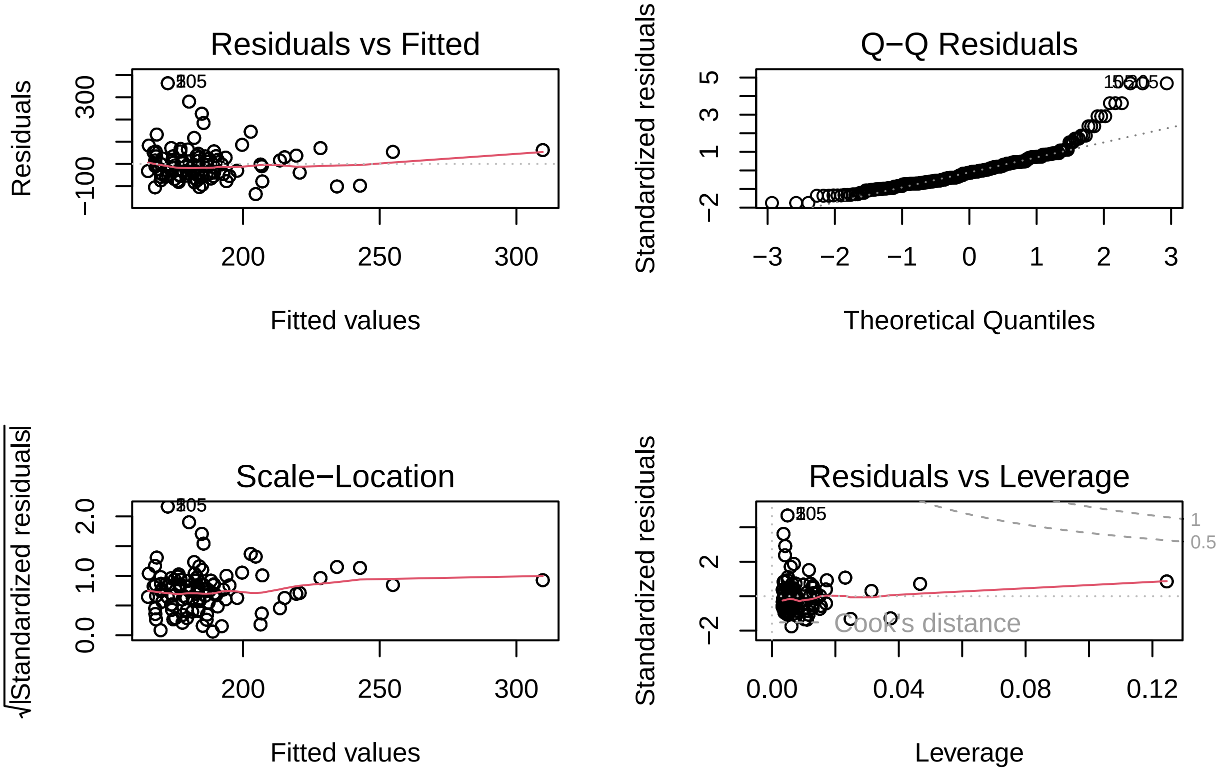
All of the steps for space group P1211 seem to hold for P121, and C121 space groups. As seen from the plot below, there's not a lot of variation in the points of the three space groups. With P1211 having a `Spacegroup.ID` of 0, P121 that of 1, and C121 has 2.

For ease of calculation, these space groups will be combined into one data frame, however this it's important to note that the group symmetry must be preserved when analyzing the nuances of each space group.

```
plot(all_df1$Spacegroup.ID, all_df1$Mean.Intensity, data=all_df1) # does not look like significant diff
```



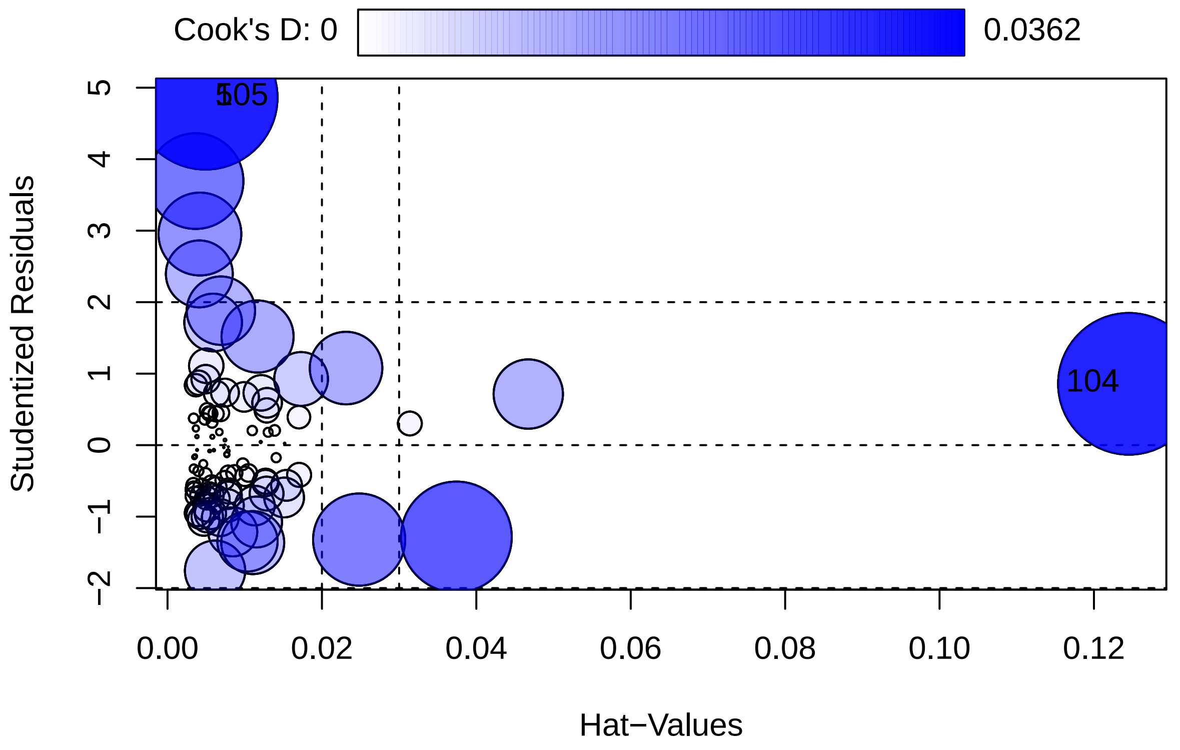
```
############## FULL MODEL
# Since the mean intensity does not seem to change for the given spacegroups...
model.all = lm(Mean.Intensity ~ Calc.Structure.weight + Packing.Density, data = all_df1)
par(mfrow=c(2,2))
plot(model.all)
```



```r
par(mfrow=c(1,1))
summary(model.all)
```

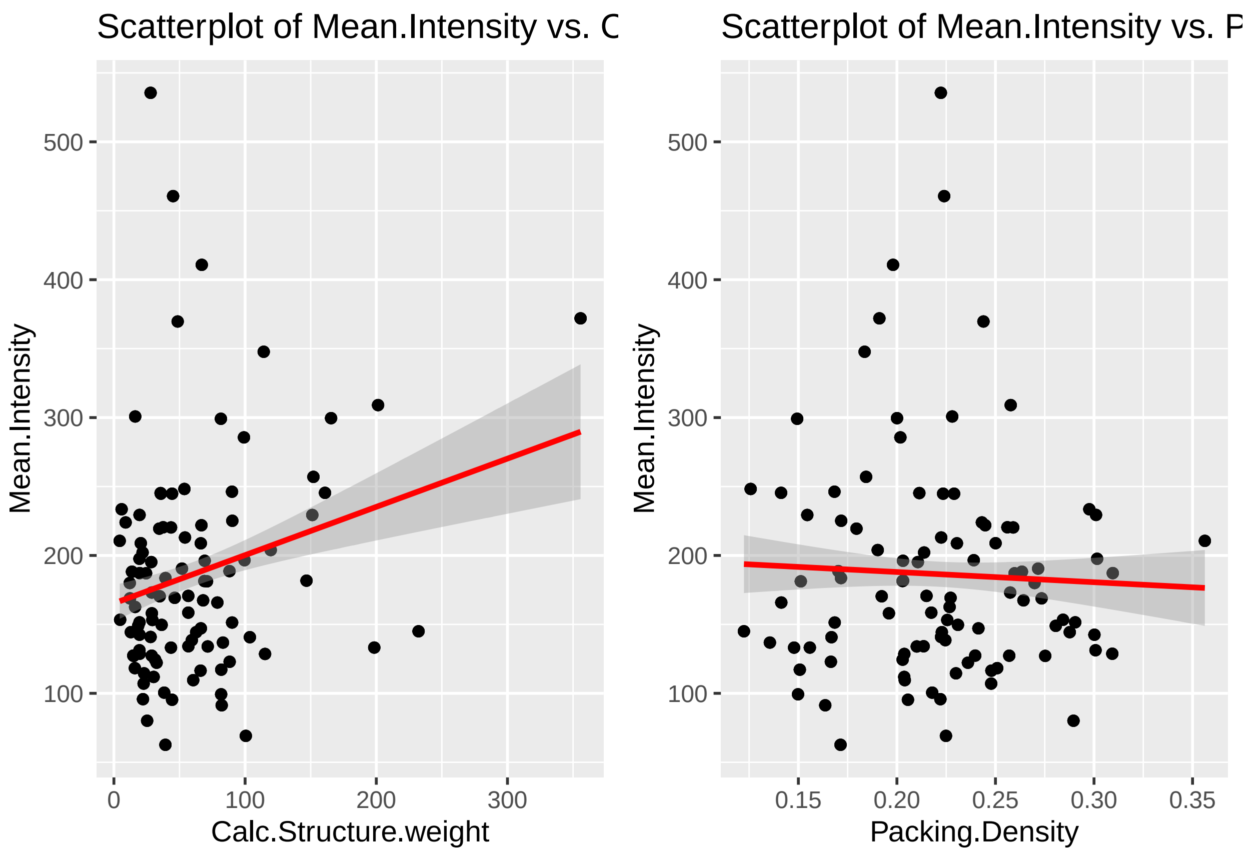
```
##
## Call:
## lm(formula = Mean.Intensity ~ Calc.Structure.weight + Packing.Density,
##     data = all_df1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -135.50  -50.28  -10.09   33.98  363.06
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)           118.36937   28.40845   4.167 4.06e-05 ***
## Calc.Structure.weight   0.43646    0.09697   4.501 9.72e-06 ***
## Packing.Density       188.47593  110.97152   1.698   0.0905 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 77.64 on 297 degrees of freedom
## Multiple R-squared:  0.06566,    Adjusted R-squared:  0.05937
## F-statistic: 10.44 on 2 and 297 DF,  p-value: 4.169e-05
```

```
car::influencePlot(model.all)
```



```
##         StudRes         Hat      CookD
## 4   0.8575044 0.124531334 0.03489609
## 5   4.8630451 0.004911449 0.03615148
## 104 0.8575044 0.124531334 0.03489609
## 105 4.8630451 0.004911449 0.03615148
```

```
########## Diagnostics
# -------------------------------------------------------------------------
# Scatterplot of Calc.Structure.weight vs. Mean.Intensity with regression line
a = ggplot(data = all_df1, aes(x = Calc.Structure.weight, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Calc.Structure.weight", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Calc.Structure.weight")

# Scatterplot of Packing.Density vs. Mean.Intensity with regression line
b = ggplot(data = all_df1, aes(x = Packing.Density, y = Mean.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Packing.Density", y = "Mean.Intensity",
       title = "Scatterplot of Mean.Intensity vs. Packing.Density")
ggarrange(a,b)
```



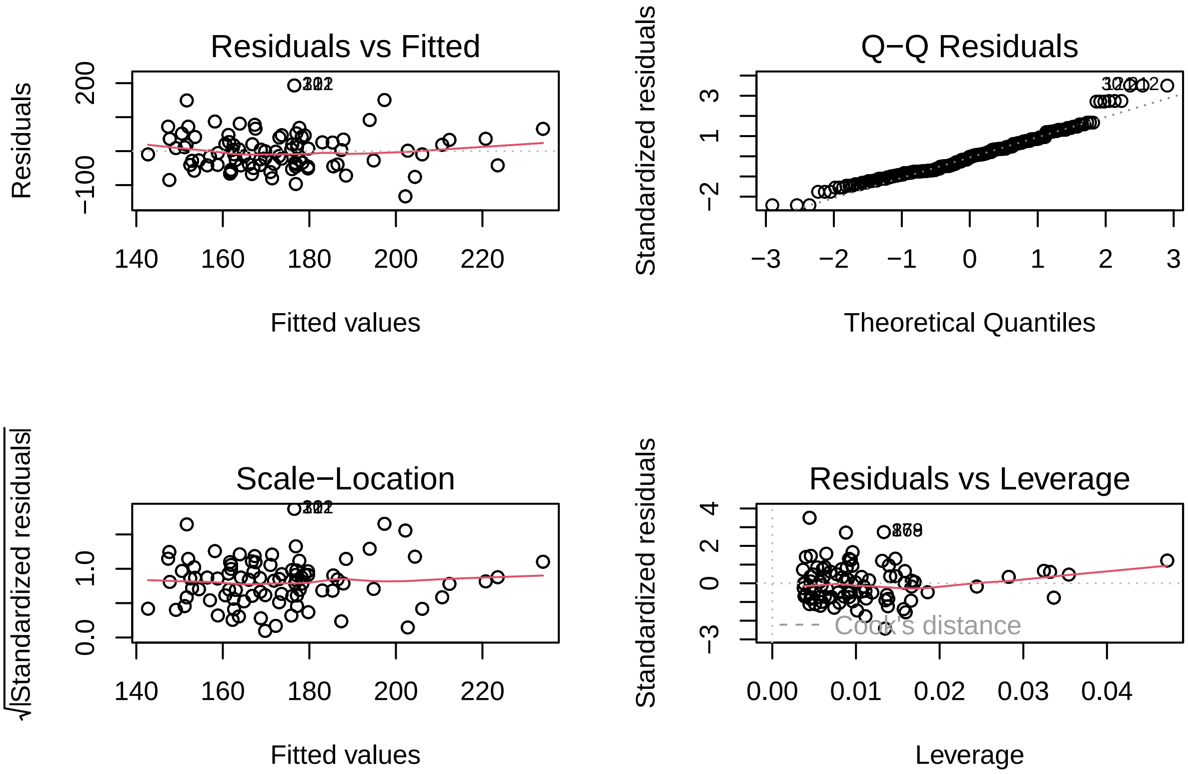
```
# ---------------------------------------------------------------------------
######### Remove outliers for model.all
cooks.d = cooks.distance(model.all)
influence = cooks.d[(cooks.d > (3*mean(cooks.d, na.rm = T)))]
influence # see that most of the rows stated above are in this list
```

```
##           4           5          31          33          47          59
## 0.034896087 0.036151480 0.011931058 0.021349197 0.009040848 0.015922042
##          63          72          77         104         105         131
## 0.009162959 0.008359826 0.014772993 0.034896087 0.036151480 0.011931058
##         133         147         159         163         172         177
## 0.021349197 0.009040848 0.015922042 0.009162959 0.008359826 0.014772993
##         204         205         231         233         247         259
## 0.034896087 0.036151480 0.011931058 0.021349197 0.009040848 0.015922042
##         263         272         277
## 0.009162959 0.008359826 0.014772993
```

```
row.influence = names(influence)
outliers = all_df1[row.influence,]
all_df1.reduced = all_df1 %>% anti_join(outliers)
```

```
## Joining with ‘by = join_by(Spacegroup.ID, PDB.ID, Calc.Structure.weight, a, b,
## c, Alpha, Beta, Gamma, Unit.Vol, Packing.Density, Mean.Intensity,
## Max.Intensity, Min.Intensity, MaxMin.Intensity.Dif, Mean.Phase, Max.Phase,
## Min.Phase, MaxMin.Phase.Difference)‘
```

```
########## Any improvement to the model?
model.all.reduced = lm(Mean.Intensity ~ Calc.Structure.weight + Packing.Density, data = all_df1.reduced)
par(mfrow=c(2,2))
plot(model.all.reduced)
```

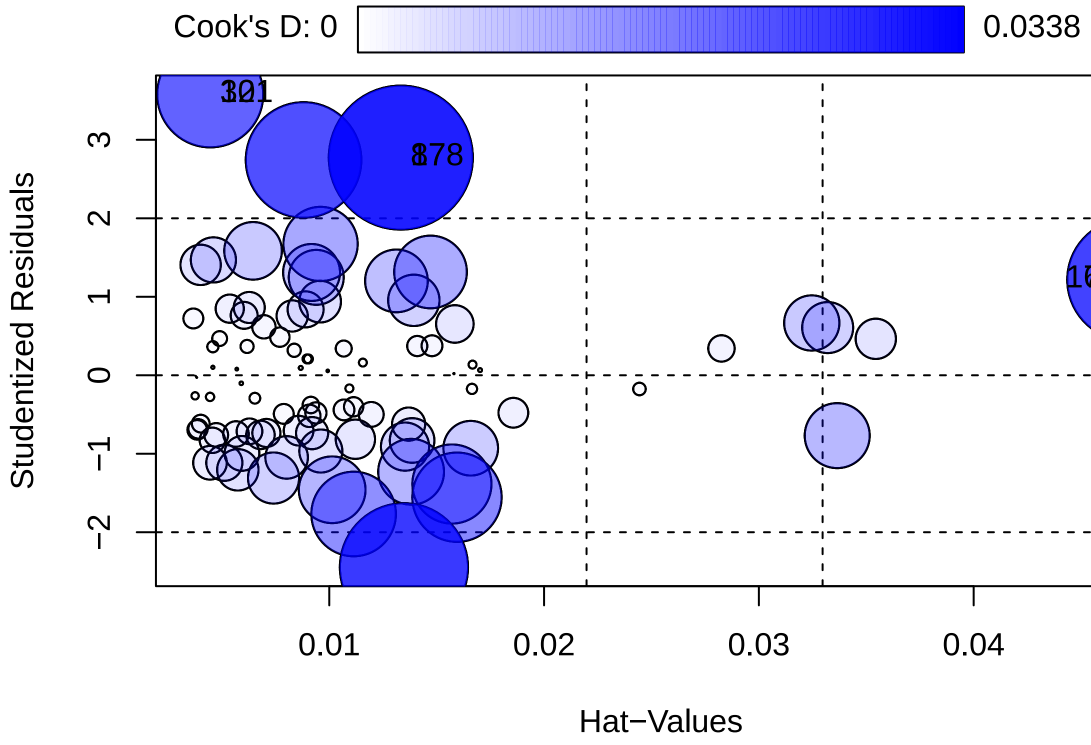


```
par(mfrow=c(1,1))
summary(model.all.reduced)
```

```
##
## Call:
## lm(formula = Mean.Intensity ~ Calc.Structure.weight + Packing.Density,
##     data = all_df1.reduced)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -133.056  -40.595   -1.571   34.035  193.161
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            68.8925    25.6944   2.681 0.007787 **
## Calc.Structure.weight   0.6114     0.1141   5.358  1.8e-07 ***
## Packing.Density       319.4114    94.3962   3.384 0.000821 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 55.26 on 270 degrees of freedom
## Multiple R-squared:  0.09612,    Adjusted R-squared:  0.08942
## F-statistic: 14.36 on 2 and 270 DF,  p-value: 1.188e-06
```

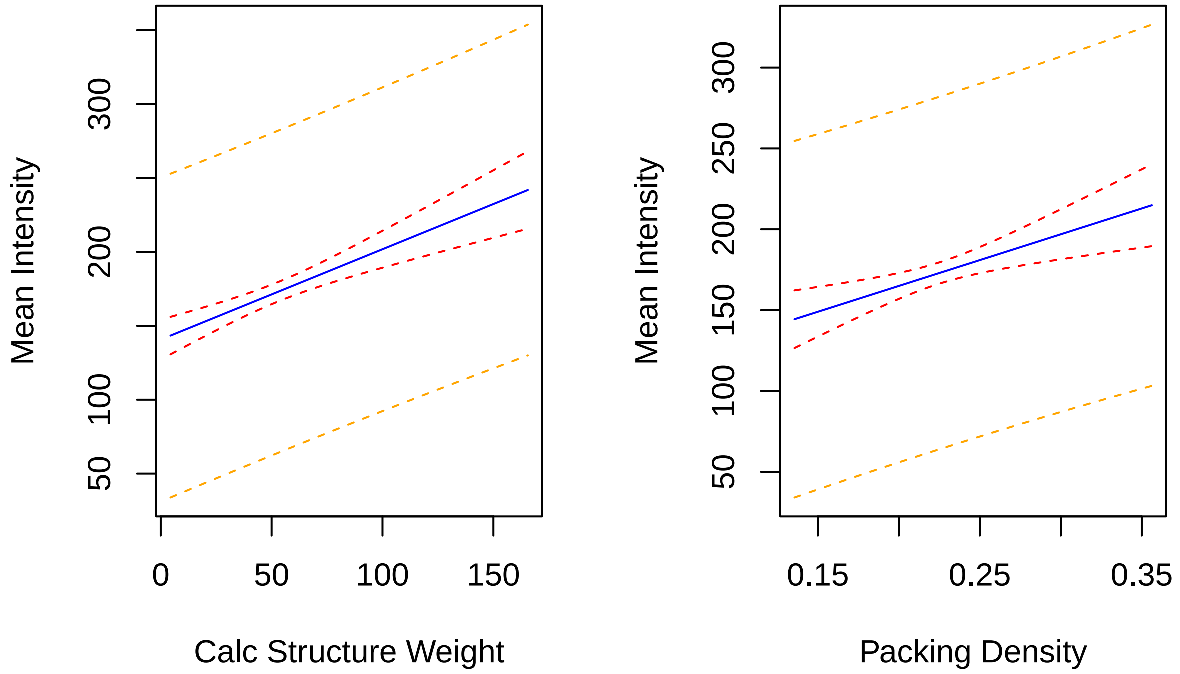
```
car::influencePlot(model.all.reduced)
```



```
##       StudRes         Hat      CookD
## 30   3.579035 0.004457441 0.01831659
## 77   1.216825 0.047205199 0.02440917
## 87   2.773119 0.013330724 0.03379620
## 121  3.579035 0.004457441 0.01831659
## 168  1.216825 0.047205199 0.02440917
## 178  2.773119 0.013330724 0.03379620
```

```
# ------------------------------------------------------------------------
par(mfrow=c(1,2))
# ------------------------------------------------------------------------
# For Calc.Structure.weight
calc_structure_weight_mean <- mean(all_df1.reduced$Calc.Structure.weight, na.rm = TRUE)
calc_structure_weight_range <- seq(from = min(all_df1.reduced$Calc.Structure.weight),
                                   to = max(all_df1.reduced$Calc.Structure.weight), length.out = 100)
vol_unit_ratio_mean <- mean(all_df1.reduced$Packing.Density, na.rm = TRUE)

x_new <- data.frame(Calc.Structure.weight = calc_structure_weight_range,
                    Packing.Density = rep(vol_unit_ratio_mean, 100))
```

```
conf.int.1 <- predict(model.all.reduced, newdata = x_new, interval = "confidence")
pred.int.1 <- predict(model.all.reduced, newdata = x_new, interval = "prediction")

# Calculate custom y-axis limits for the first plot
y_limits_1 <- c(min(min(conf.int.1[, "lwr"]), min(pred.int.1[, "lwr"])),
                max(max(conf.int.1[, "upr"]), max(pred.int.1[, "upr"])))

# Plot the first confidence interval
plot(x_new$Calc.Structure.weight, conf.int.1[, "fit"], type = "l", col = "blue",
     ylim = y_limits_1,
     ylab = "Mean Intensity", xlab = "Calc Structure Weight")
lines(x_new$Calc.Structure.weight, conf.int.1[, "lwr"], col = "red", lty = 2)
lines(x_new$Calc.Structure.weight, conf.int.1[, "upr"], col = "red", lty = 2)

# Add lines for the lower and upper bounds of the prediction interval in orange
lines(x_new$Calc.Structure.weight, pred.int.1[, "lwr"], col = "orange", lty = 2)
lines(x_new$Calc.Structure.weight, pred.int.1[, "upr"], col = "orange", lty = 2)
# -------------------------------------------------------------------------
# For Packing.Density
vol_unit_ratio_range <- seq(from = min(all_df1.reduced$Packing.Density),
                            to = max(all_df1.reduced$Packing.Density), length.out = 100)

x_new <- data.frame(Packing.Density = vol_unit_ratio_range,
                    Calc.Structure.weight = rep(calc_structure_weight_mean, 100))

conf.int.2 <- predict(model.all.reduced, newdata = x_new, interval = "confidence")
pred.int.2 <- predict(model.all.reduced, newdata = x_new, interval = "prediction")

# Calculate custom y-axis limits for the second plot
y_limits_2 <- c(min(min(conf.int.2[, "lwr"]), min(pred.int.2[, "lwr"])),
                max(max(conf.int.2[, "upr"]), max(pred.int.2[, "upr"])))

# Plot the second confidence interval
plot(x_new$Packing.Density, conf.int.2[, "fit"], type = "l", col = "blue",
     ylim = y_limits_2,
     ylab = "Mean Intensity", xlab = "Packing Density")
lines(x_new$Packing.Density, conf.int.2[, "lwr"], col = "red", lty = 2)
lines(x_new$Packing.Density, conf.int.2[, "upr"], col = "red", lty = 2)

# Add lines for the lower and upper bounds of the prediction interval in orange
lines(x_new$Packing.Density, pred.int.2[, "lwr"], col = "orange", lty = 2)
lines(x_new$Packing.Density, pred.int.2[, "upr"], col = "orange", lty = 2)
```



```r
# ----------------------------------------------------------------------
# Reset the plotting environment to one plot
par(mfrow=c(1,1))
# ----------------------------------------------------------------------
confint(model.all)
```

```
##                             2.5 %      97.5 %
## (Intercept)            62.4620152 174.2767191
## Calc.Structure.weight   0.2456281   0.6272997
## Packing.Density       -29.9141894 406.8660491
```

Hypothesis Test:

Testing whether the the reduced linear model provides a better fit. Going to use $\alpha = 0.05$ as the significance value.

1. Assumptions: $\varepsilon \overset{iid}{\sim} N(0,1)$

- Alternatives: – Define Models: Y as `Mean.Intensity`. – X_1 as `Calc.Structure.weight`. – X_2 as `Packing.Density`.

$$E\{Y\}_{\text{reduced.model}} = \beta_0 + \beta_1 X_1$$
$$E\{Y\}_{\text{full.model}} = \beta_0 + \beta_1 X_1 + \beta_2 X_2$$

2. Hypotheses:

$$H_0 : \beta_2 = 0$$
$$H_1 : \beta_2 \neq 0$$

Decision Rule:

$$p_{\text{val}} \leq \alpha = 0.05 \quad \Rightarrow \quad \text{reject null hypothesis}$$
$$p_{\text{val}} > \alpha = 0.05 \quad \Rightarrow \quad \text{do not reject null hypothesis}$$

3. Test Statistic:

```
model.all.F = model.all.reduced
model.all.R = lm(Mean.Intensity ~ Calc.Structure.weight, data=all_df1.reduced)
# model.all.reduced = all_df1.reduced dataframe without outliers
# model.all.R  = dropped PackingDensity all_df1.reduced dataframe without outliers
anova.table = anova(model.all.F, model.all.R)
anova.table
```

```
## Analysis of Variance Table
##
## Model 1: Mean.Intensity ~ Calc.Structure.weight + Packing.Density
## Model 2: Mean.Intensity ~ Calc.Structure.weight
##   Res.Df    RSS Df Sum of Sq     F    Pr(>F)
## 1    270 824524
## 2    271 859489 -1    -34965 11.45 0.0008211 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
F.crit = anova.table[2,5]
```

4. P-Value:

```
alpha = 0.05
pval = anova.table[2,6]
pval < alpha
```

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

5. Conclusion: Since the p_{value} is less than the specified significance value ($\alpha = 0.05$) by the specified decision rule, we have sufficient evidence to reject the null hypothesis. There is a significant effect on the calculated structure weight (in kiloDaltons) and the mean intensity (number of photons). Therefore, the calculated structure weight contributes meaningfully to the variations of the mean intensity.

Warranting further investigation, this result suggests that the *true* structure weight also has a significant effect on the observed, *experimental* intensity values.

Another Approach: Potential Predictors:

Let us introduce more context for the next approach:

- `h,k,l`: Also called Miller indices, are the orientation of the crystal lattice planes in three-dimensional reciprocal space. Depending on the notation, this can refer to a plane, point, or vector. For the sake of simplicity, this refers to a point in reciprocal space.

- `Common.Reflection.Max.Intensity` : Besides the motivation behind this predictor, this predictor gives a common maximum reflection across all of the structure factor files. This predictor was created to give a level of stability and consistency, this also provides information to a significant structure feature which is present in all of the crystals being analyzed.

Research Question:

Let us observe whether there is a linear dependence among the maximum intensity that is common to all structure factor files, and the packing density of the unit cell in question. This will give insights on how protein crystals are related to their outputted intensity values depending on their packing densities. Said another way, if we know the packing density of a specific unit cell under experimentation, we can infer about the maximum diffraction intensity output such that this maximum is common to all the crystals.

Motivation

Instead of analyzing the `Mean.Intensity` as before, we will pick the maximum common reflection intensity value and point, in all of the structure factor files (`.hkl`), and append these to a data frame. Hence, for all of the `h,k,l` coordinates, the same point in reciprocal space is chosen to analyze the hypothesized relation to the packing density. These coordinates and their max common intensity value were found from the Python script, for each space group is:

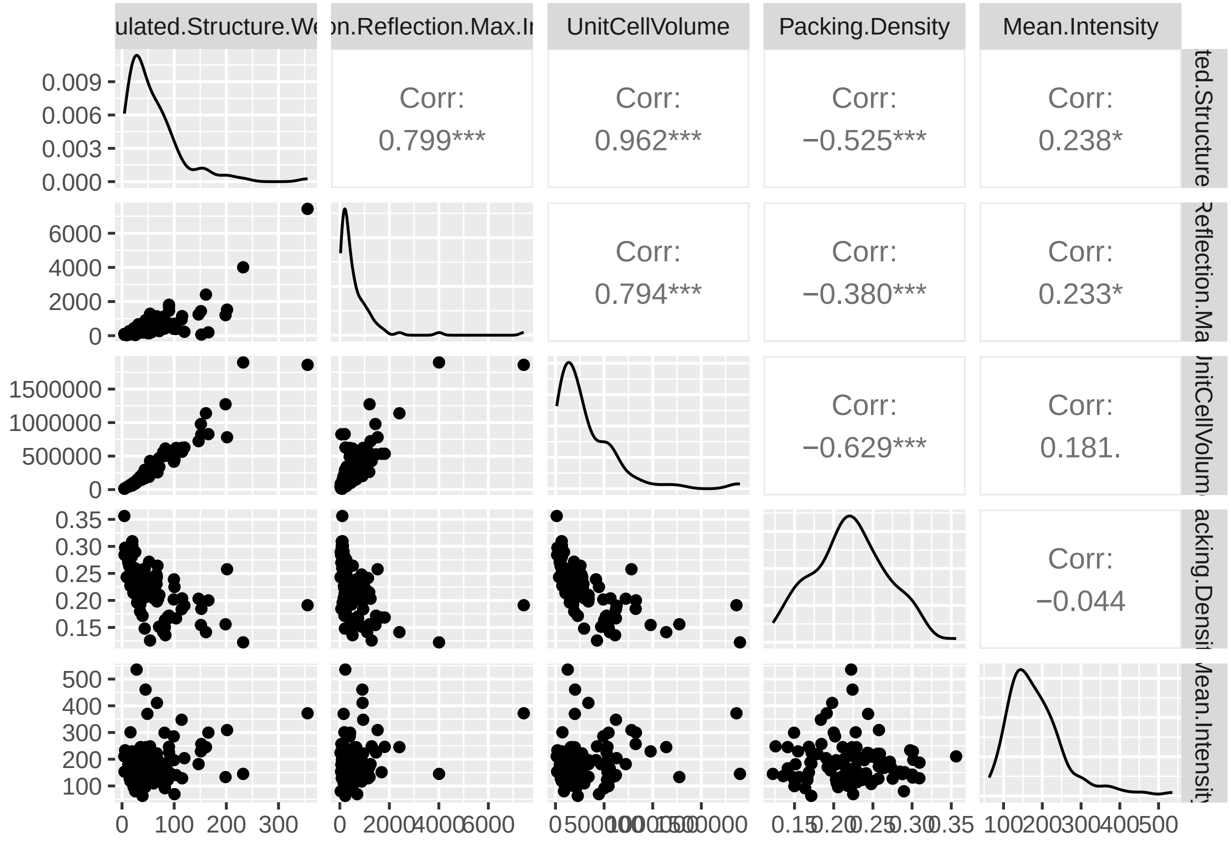
- P1211 : Maximum common intensity: 7429.6, HKL index: (6, 1, 2)
- P121 : Maximum common intensity: 11019.7, HKL index: (2, -7, 0)
- C121 : Maximum common intensity: 9892.4, HKL index: (2, 0, 11)

```
P1211 = read.table("../P1211_output/P1211_new_approach_crystal_df.txt", header = TRUE, fill = TRUE)
P121 = read.table("../P121_output/P121_new_approach_crystal_df.txt", header = TRUE, fill = TRUE)
C121 = read.table("../C121_output/C121_new_approach_crystal_df.txt", header = TRUE, fill = TRUE)
new_colnames = c('',
                'PDB_ID',
               'Spacegroup',
               'Calculated.Structure.Weight',
               'a', 'b', 'c',
               'alpha', 'beta', 'gamma',
               'h', 'k', 'l',
               'Common.Reflection.Max.Intensity',
               'UnitCellVolume',
               'Packing.Density',
               'Mean.Intensity',
               'Max.Intensity',
               'Min.Intensity',
               'Max.Min.Intensity.Difference',
               'Mean.Phase',
               'Max.Phase',
               'Min.Phase',
               'Max.Min.Phase.Difference')
colnames(P1211) = new_colnames
colnames(P121) = new_colnames
colnames(C121) = new_colnames
P1211 = P1211[,-1]
P121 = P121[, -1]
```

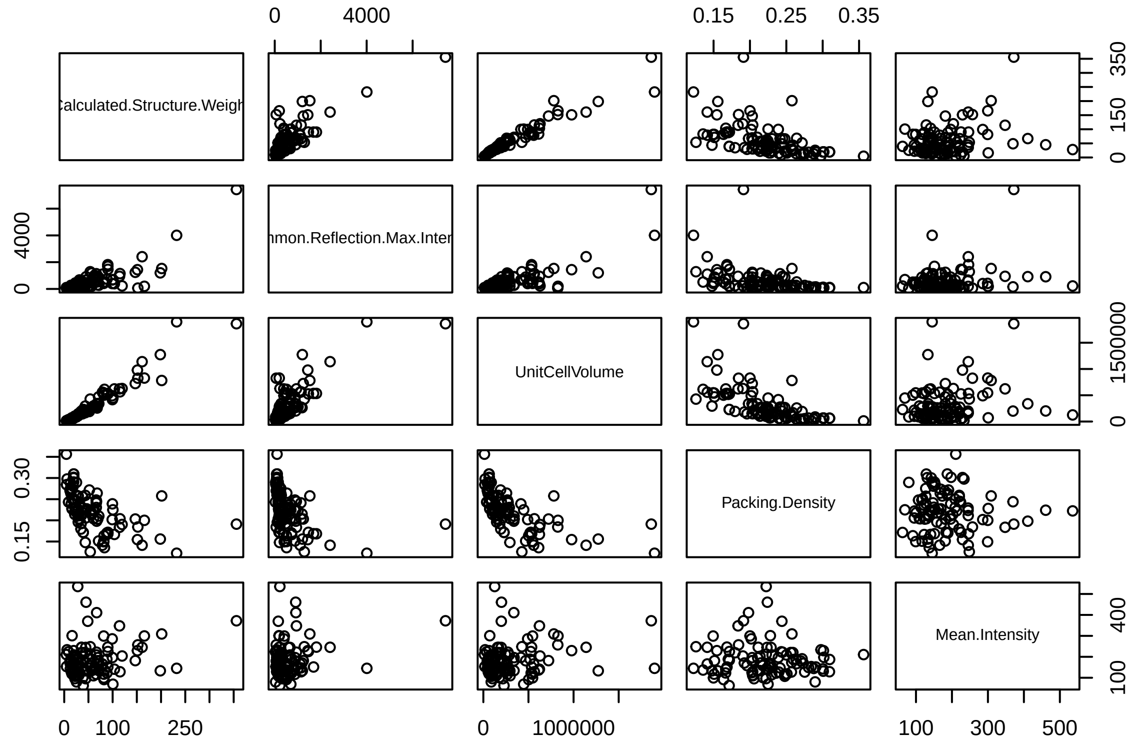
```
C121 = C121[, -1]
P1211 = P1211[, sapply(P1211, function(col) !all(is.na(col)))]
P121 = P121[, sapply(P121, function(col) !all(is.na(col)))]
C121 = C121[, sapply(C121, function(col) !all(is.na(col)))]
all_df = rbind(P1211,P121,C121)
```

Data Exploration

```
keep = c('Calculated.Structure.Weight',
 'Common.Reflection.Max.Intensity',
 'UnitCellVolume',
 'Packing.Density',
 'Mean.Intensity')
numeric = P1211[, (names(P1211) %in% keep)]
ggpairs(numeric,cardinality_threshold = 100)
```



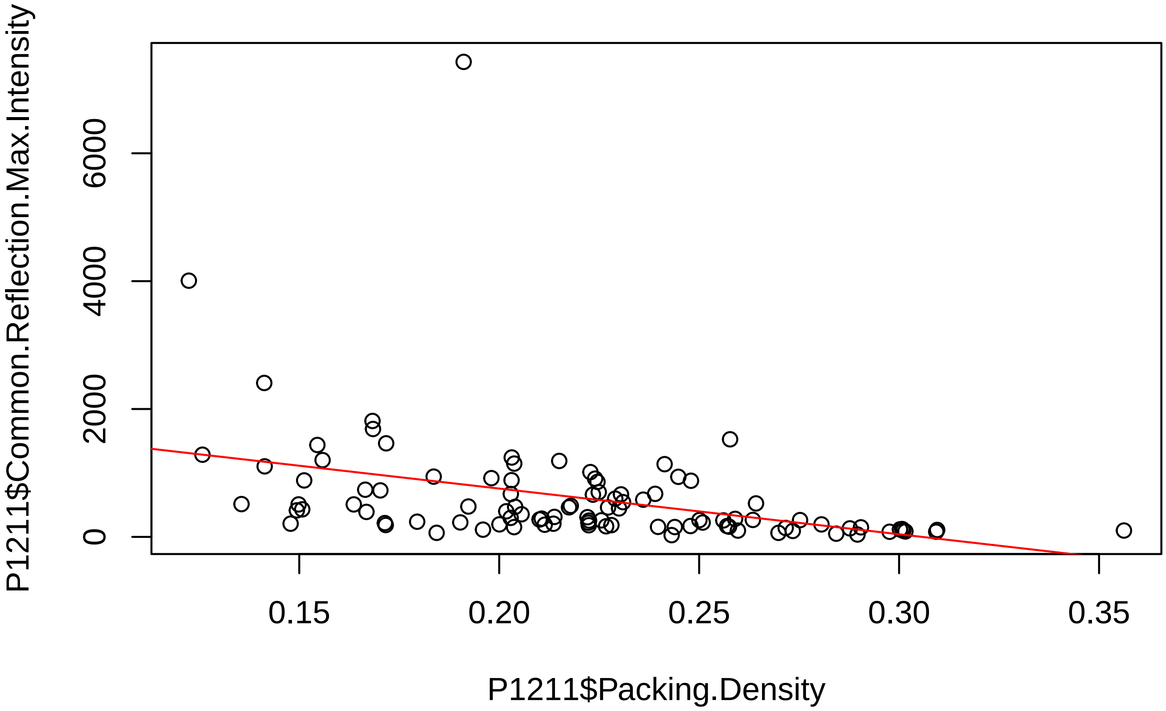
```r
pairs(numeric)
```



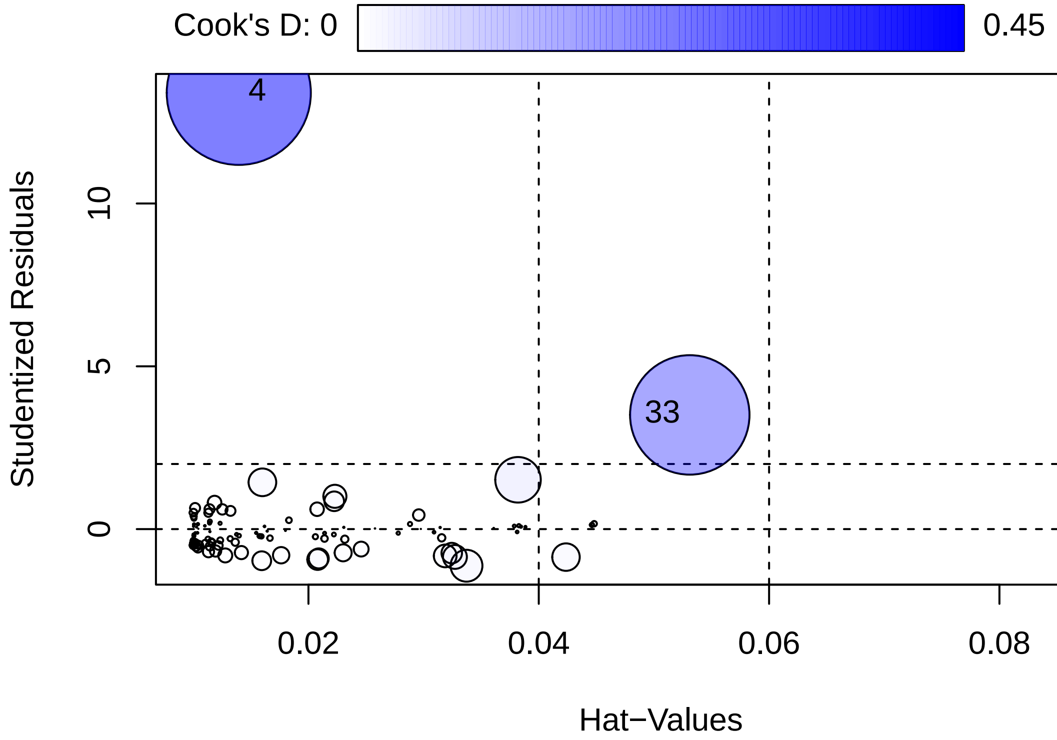
It seems that from the pairs plot, that the max reflection predictor (`Common.Reflection.Max.Intensity`) from first glance have a negative linear relationship. This is interesting since we have taken the maximum reflection which is common to all calculated structure factor

Recall, that the packing density is related by the following equation $\text{Packing.Density} = \frac{\text{Calc.Structure.Weight}*1000}{\text{Unit.Cell.Volume}} = \frac{\text{g/mol}}{\text{Å}^3}$. We will not consider this as a significant predictor.

```r
# exploring
plot(P1211$Packing.Density, P1211$Common.Reflection.Max.Intensity)
abline(lm(Common.Reflection.Max.Intensity ~ Packing.Density, data=P1211), col="red")
```



```
car::influencePlot(lm(Common.Reflection.Max.Intensity ~ Packing.Density, data=P1211))
```



```
##       StudRes        Hat      CookD
## 4  13.3979837 0.01395221 0.45010666
## 12  0.5768999 0.09120446 0.01681464
## 33  3.5072480 0.05311222 0.30931551
```

```
m = lm(Common.Reflection.Max.Intensity ~ Packing.Density, data=P1211)
m.all = lm(Common.Reflection.Max.Intensity ~ Packing.Density, data=all_df)
# summary(m)
# summary(m.all)
cat('Adjusted R^2 of only P1211 data', summary(m)$adj.r.squared)
```

```
## Adjusted R^2 of only P1211 data 0.1355912
```

```
cat('Adjusted R^2 of ALL data', summary(m.all)$adj.r.squared)
```

```
## Adjusted R^2 of ALL data 0.00547536
```

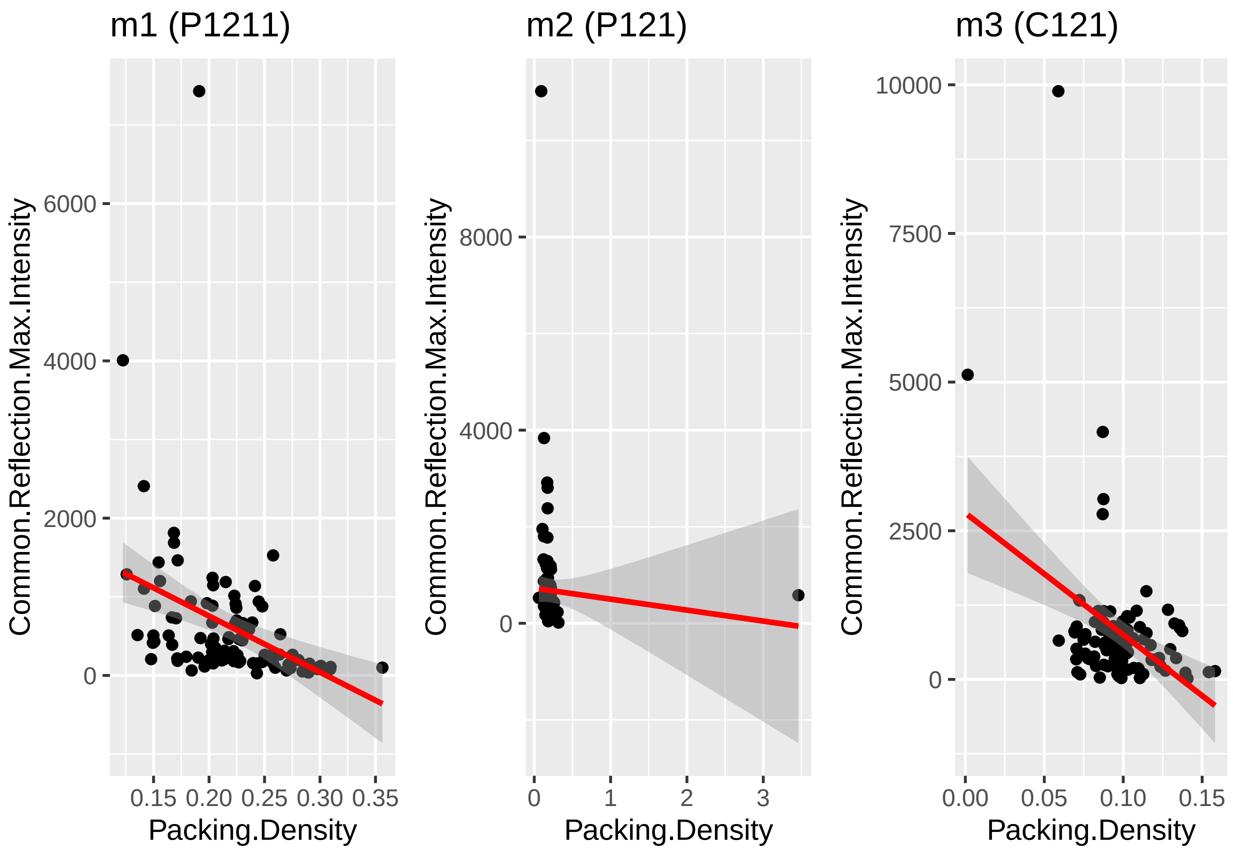
If we jump to conclusions, we see that the $P1211 : R_a^2 = 0.1355912$ and $All : R_a^2 = 0.00547536$. Thus the variation explained by the model decreases by $0.00547536 - 0.1355912 = -0.1301158$. We loose valiable information when ignoring the differentiation of the space group symmetry.

Model:

For all space groups, the model will take the following form, denoted in the code as `m1,m2,m3`.

$$E\{\text{Common.Reflection.Max.Intensity}\} = \beta_0 + \beta_1(\text{Packing.Density})$$

```
m1 = lm(Common.Reflection.Max.Intensity ~ Packing.Density, data=P1211)
m2 = lm(Common.Reflection.Max.Intensity ~ Packing.Density, data=P121)
m3 = lm(Common.Reflection.Max.Intensity ~ Packing.Density, data=C121)
################
a = ggplot(data = P1211, aes(x = Packing.Density, y = Common.Reflection.Max.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Packing.Density", y = "Common.Reflection.Max.Intensity",
       title = "m1 (P1211)")
b = ggplot(data = P121, aes(x = Packing.Density, y = Common.Reflection.Max.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Packing.Density", y = "Common.Reflection.Max.Intensity",
       title = "m2 (P121)")
c = ggplot(data = C121, aes(x = Packing.Density, y = Common.Reflection.Max.Intensity)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, col = "red") +
  labs(x = "Packing.Density", y = "Common.Reflection.Max.Intensity",
       title = "m3 (C121)")
ggarrange(a,b,c,ncol = 3,nrow = 1)
```



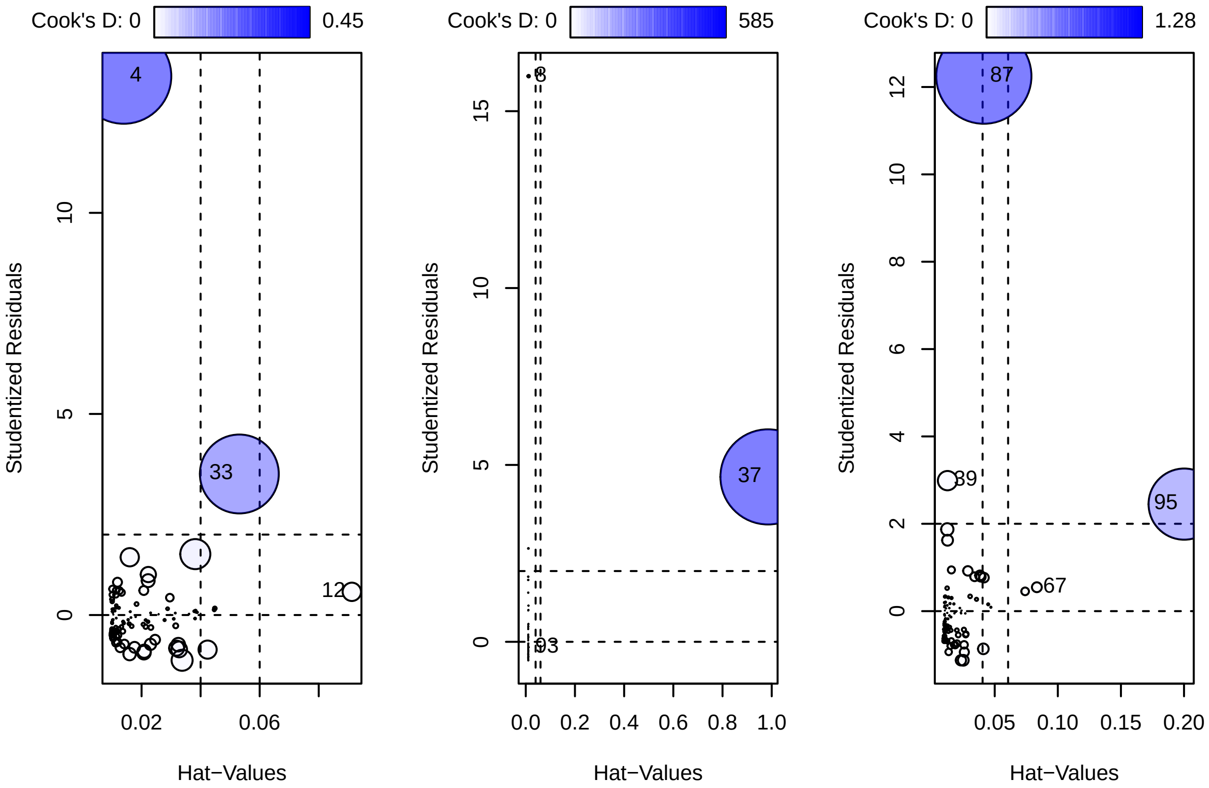
```
###############
par(mfrow=c(1,3))
influencePlot(m1)
```

```
##       StudRes        Hat      CookD
## 4  13.3979837 0.01395221 0.45010666
## 12  0.5768999 0.09120446 0.01681464
## 33  3.5072480 0.05311222 0.30931551
```

```
influencePlot(m2)
```

```
##       StudRes        Hat        CookD
## 8  15.9888855 0.01166930 4.194092e-01
## 37  4.6618385 0.98491209 5.854750e+02
## 93 -0.1518723 0.01250798 1.475475e-04
```

```
influencePlot(m3)
```



```
##       StudRes        Hat      CookD
## 39  2.9880141 0.01258430 0.05259492
## 67  0.5474373 0.08338263 0.01373008
## 87 12.2479289 0.04136008 1.27595994
## 95  2.4505314 0.20012265 0.71435362
```

```
par(mfrow=c(1,1))
###############
m = lm(Common.Reflection.Max.Intensity ~ Packing.Density + Spacegroup, data = all_df)
summary(m)
```

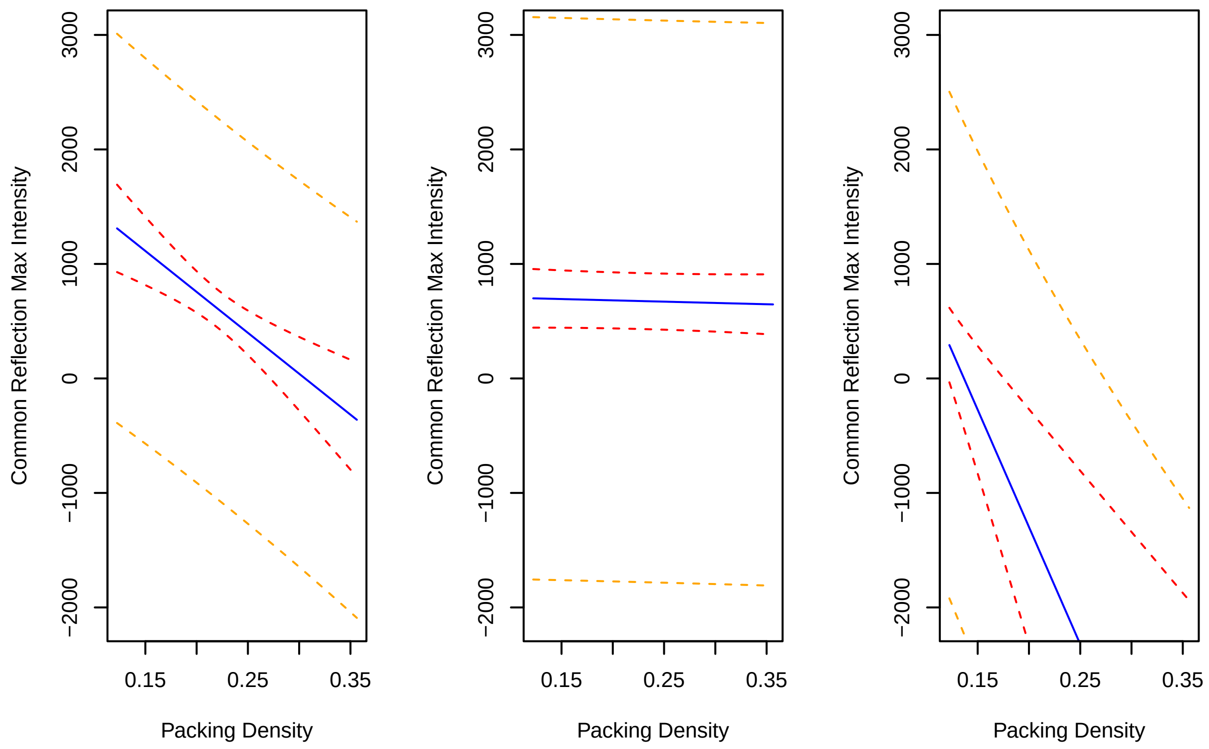
```
##
## Call:
## lm(formula = Common.Reflection.Max.Intensity ~ Packing.Density +
##     Spacegroup, data = all_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -768.8  -446.5  -296.9    88.2 10288.6
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       709.54     129.28   5.488 8.72e-08 ***
## Packing.Density  -455.55     330.37  -1.379    0.169
## Spacegroup         63.21      81.28   0.778    0.437
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 1110 on 296 degrees of freedom
## Multiple R-squared:  0.01083,    Adjusted R-squared:  0.00415
## F-statistic: 1.621 on 2 and 296 DF,  p-value: 0.1995
```

From the influence plots, it’s not worth removing the influence plots since there are at most 2 per space group. Also explored the variation of adding space group as a categorical variable, however this only worsens the model’s fit. Thus, we will explore the space group data separately. There seems to be a discrepancy among each space group.

Confidence and Prediction Interval Plots:

```
######### Confidence/prediction intervals
par(mfrow=c(1,3))
# Calculate mean and range for Packing.Density
packing_density_mean <- mean(P1211$Packing.Density, na.rm = TRUE)
packing_density_range <- seq(from = min(P1211$Packing.Density, na.rm = TRUE),
                             to = max(P1211$Packing.Density, na.rm = TRUE), length.out = 100)

# Create new data for predictions
x_new <- data.frame(Packing.Density = packing_density_range)

# Model m1
conf.int.m1 <- predict(m1, newdata = x_new, interval = "confidence")
pred.int.m1 <- predict(m1, newdata = x_new, interval = "prediction")

# Custom y-axis limits for m1 plot
y_limits_m1 <- range(c(conf.int.m1[, "lwr"], conf.int.m1[, "upr"],
                       pred.int.m1[, "lwr"], pred.int.m1[, "upr"]))
```

```r
# Plot m1
plot(x_new$Packing.Density, conf.int.m1[, "fit"], type = "l", col = "blue",
     ylim = y_limits_m1,
     ylab = "Common Reflection Max Intensity", xlab = "Packing Density")
lines(x_new$Packing.Density, conf.int.m1[, "lwr"], col = "red", lty = 2)
lines(x_new$Packing.Density, conf.int.m1[, "upr"], col = "red", lty = 2)
lines(x_new$Packing.Density, pred.int.m1[, "lwr"], col = "orange", lty = 2)
lines(x_new$Packing.Density, pred.int.m1[, "upr"], col = "orange", lty = 2)
######
# Model m2
conf.int.m2 <- predict(m2, newdata = x_new, interval = "confidence")
pred.int.m2 <- predict(m2, newdata = x_new, interval = "prediction")

# Custom y-axis limits for m1 plot
y_limits_m2 <- range(c(conf.int.m1[, "lwr"], conf.int.m2[, "upr"],
                       pred.int.m1[, "lwr"], pred.int.m2[, "upr"]))

# Plot m1
plot(x_new$Packing.Density, conf.int.m2[, "fit"], type = "l", col = "blue",
     ylim = y_limits_m1,
     ylab = "Common Reflection Max Intensity", xlab = "Packing Density")
lines(x_new$Packing.Density, conf.int.m2[, "lwr"], col = "red", lty = 2)
lines(x_new$Packing.Density, conf.int.m2[, "upr"], col = "red", lty = 2)
lines(x_new$Packing.Density, pred.int.m2[, "lwr"], col = "orange", lty = 2)
lines(x_new$Packing.Density, pred.int.m2[, "upr"], col = "orange", lty = 2)
######
# Model m3
conf.int.m3 <- predict(m3, newdata = x_new, interval = "confidence")
pred.int.m3 <- predict(m3, newdata = x_new, interval = "prediction")

# Custom y-axis limits for m1 plot
y_limits_m3 <- range(c(conf.int.m3[, "lwr"], conf.int.m3[, "upr"],
                       pred.int.m3[, "lwr"], pred.int.m3[, "upr"]))

# Plot m1
plot(x_new$Packing.Density, conf.int.m3[, "fit"], type = "l", col = "blue",
     ylim = y_limits_m1,
     ylab = "Common Reflection Max Intensity", xlab = "Packing Density")
lines(x_new$Packing.Density, conf.int.m3[, "lwr"], col = "red", lty = 2)
lines(x_new$Packing.Density, conf.int.m3[, "upr"], col = "red", lty = 2)
lines(x_new$Packing.Density, pred.int.m3[, "lwr"], col = "orange", lty = 2)
lines(x_new$Packing.Density, pred.int.m3[, "upr"], col = "orange", lty = 2)
```



```r
par(mfrow=c(1,1))
```

```r
confint(m1, level = 0.95)
```

```
##                        2.5 %    97.5 %
## (Intercept)         1396.746  2973.772
## Packing.Density -10637.132 -3659.020
```

```r
confint(m2, level = 0.95)
```

```
##                       2.5 %    97.5 %
## (Intercept)       430.9904 1023.7226
## Packing.Density -972.5347  518.3331
```

```r
confint(m3, level = 0.95)
```

```
##                        2.5 %     97.5 %
## (Intercept)         1804.217   3790.132
## Packing.Density -30342.100 -10605.731
```

Interpretation:

- m1: – Of the first space group data (P1211), for every 1-unit increase in the packing density for the 100 sample proteins, the corresponding common reflection max intensity value decreases by a value of

-7148.1, when holding all other predictors constant. Where the model explained $R^2_{P1211} = 0.1443$ of the total variation in the data. – With 95% confidence, we observe that the true expected packing density for P1211 space group lies within (-10637.132, -3659.020). Notice that for this data, 0 is not included within the interval which gives us strong evidence to suggest that the true relation to the common max intensity has an inverse relationship with the packing density.

- `m2`: –The second spacegroup (P121), for every 1-unit increase in the packing density for the 100 sample proteins, the corresponding common reflection max intensity value decreases by a value of -227.1, when holding all other predictors constant. Where the model explained $R^2_{C121} = 0.003716$ of the total variation in the data. – With 95% confidence, we observe that the true expected packing density for P121 lies within (-972.5347, 518.3331). Notice that for this data, 0 is included within the interval which gives us strong evidence to suggest that the true relation to the common max intensity for this space group does not have the same inverse relationship with the packing density.

`-m3`: –Finally, third spacegroup (C121), for every 1-unit increase in the packing density for the 100 sample proteins, the corresponding common reflection max intensity value decreases by a value of -20473.9, when holding all other predictors constant. Where the model explained $R^2_{C121} = 0.1488$ of the total variation in the data. – With 95% confidence, we observe that the true expected packing density for C121 space group lies within (-30342.100, -10605.731). Notice that for this data, 0 is not included within the interval which gives us strong evidence to suggest that the true relation to the common max intensity has an inverse relationship with the packing density.

"Phase Problem." n.d. Accessed November 10, 2023. https://en.wikipedia.org/wiki/Phase_problem.