Performance of Metal Matrix Composites

Hannah Sims, Department of Materials Science and Engineering CWRU

5/2/2019

Contents

1 Abstract

This data analysis looked to analyze a set of metal matrix composites (MMC). These metal matrix composites were processed in various ways, and tested in tensile testing, bending fatigue testing, and push pull fatigue testing. Not every variation of MMC was tested in each testing type, leaving three seperate data sets with different levels of variation in terms of process parameters. These data sets were combined into one and analyzed to answer the question "Are there models (both multiple regression and multivariate) that can be created to describe and analyze how the mechanical properties (tensile, bending fatigue, and push pull fatigue) are affected by variations of processing parameters of MMC?". Data manipulation, exploratory data analysis, and statistical learning in the form of multiple and multivariate regression were used to try and answer this question. It was found that the complex nature and small size of the data set complicated analysis, and no true statistical findings could be made. However, the main takeaway from this analysis was found to be the recommendations that could be drawn from this work. Ideas on how to obtain more data are presented in the form of other sources and more simple and inexpensive tests that could be run.

2 Introduction

Metal matrix composites (MMC) are composed of two different materials. One material is the metal matrix and the other is a metal or ceramic that is dispersed within the matrix as particles. These materials can offer very different properties than the base materials used to create them. Careful selection of many variables that can be changed in these materials can greatly enhance the material's performance in a specific application.

The specific metal matrix composite looked at in this study is composed of an aluminium matrix with silicon carbide particles dispersed within it. A novel processing method is used on these MMC that is believed to improve the mechanical properties of the material significantly.

Several variations of the MMC were looked at in this project with variation lying in the heat treatment imposed on the material, the grade of aluminium used for the matrix, the volume percent of silicon carbide introduced, and the size of the silicon carbide particles dispersed. Tensile tests, bending fatigue tests, and push pull fatigue tests were run on these materials. Unfortunately, not all variations of the material were tested in each of the different types of tests. For example, in push pull fatigue tests only one variation of MMC was tested, which is very unfortunate as these tests brought interesting results in the appearance of cones on the fracture surfaces of these specimens that had not been seen before.

The aim of this project is to answer the question "Are there models (both multiple regression and multivariate) that can be created to describe and analyze how the mechanical properties (tensile, bending fatigue, and push pull fatigue) are affected by variations of processing parameters of MMC?"

This question is mostly explorative in nature, while some predictive methods may be used there is not enough data available to truly make any definitive conclusions from the analysis performed. Mostly, the analysis will just provide an idea of what is actually going on throughout all variations of the MMC in regards to the mechanical properties looked at.

The lack of data also affects the metric of success. As stated before there is not enough data to make any definitive findings, so the metric of success will be how much insight the analysis provides. Additionally, a large metric of success will be the recommendations that can be made for future work out of this analysis. After this analyis it should be apparent as to what information is missing and what is most needed. Mechanical testing is expensive so one cannot simply say run more tests of everything, instead through analysis a more in depth set of recommendations should be able to be made for any future testing that balances both the need for more data and the cost behind obtaining this data.

3 Data Science Methods

To start data was combined and cleaned using tidyverse and many base r functions. Then exploratory data analysis was performed using maninly ggplot. For modeling, linear regressions were run to model the tensile properties (mainly the ultimate tensile strength UTS), push-pull fatigue properties (focusing on the cycles to failure), and bending fatigue properties (cycles to failure). Also, a multivariate model was run to model all of these different properties together and see if a map that describes the properties and variables can be made.

4 Exploratory Data Analysis

4.1 Data Source and Target Structure

4.1.1 Source

The data used in this project was obtained as three seperate excel spreadsheets, each corresponding to a specific type of mechanical testing, i.e. tensile testing, bending fatigue, and push pull fatigue. These excel sheets were created and used by multiple people so there was a lot of unneccessary and repeated information as well as many missing data points within each data set. Fortunately all who worked from these spreadsheets were well versed in the project and did follow a standard practice of naming the variations of MMC.

4.1.2 Target Structure

The three seperate data sets needed to be put into one large data set so manipulation in modeling is easier to perform.

4.2 Data Cleaning and Creation of Data Book

4.2.1 Data Cleaning

4.2.1.1 Importing the Data

```
library (plyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.1.0     v readr   1.3.1
## v tibble  2.0.1     v purrr   0.3.0
## v tidyr   0.8.3     v stringr 1.4.0
## v ggplot2 3.1.0     v forcats 0.4.0
```

```
## Warning: package 'tidyr' was built under R version 3.5.3
```

```
## -- Conflicts --------------------------------- tidyverse_conflicts() --
## x dplyr::arrange()   masks plyr::arrange()
## x purrr::compact()   masks plyr::compact()
## x dplyr::count()     masks plyr::count()
## x dplyr::failwith()  masks plyr::failwith()
## x dplyr::filter()    masks stats::filter()
## x dplyr::id()        masks plyr::id()
## x dplyr::lag()       masks stats::lag()
## x dplyr::mutate()    masks plyr::mutate()
## x dplyr::rename()    masks plyr::rename()
## x dplyr::summarise() masks plyr::summarise()
## x dplyr::summarize() masks plyr::summarize()
```

```
tension.flats <- read.csv("H:/Git/19s-dsci353-353m-453-hes53/1-assignments/SemProj-453/Sample Directory
bending.fatigue.flats <- read_csv("H:/Git/19s-dsci353-353m-453-hes53/1-assignments/SemProj-453/Sample Di
```

```
## Warning: Missing column names filled in: 'X17' [17], 'X18' [18]
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Heat Treatment` = col_character(),
##   Position = col_character(),
##   `Segment length (In)` = col_character(),
##   `Corrected Stress` = col_number(),
##   `Stress vs Yield Ratio` = col_character(),
##   Cycles = col_number(),
##   X17 = col_logical(),
##   X18 = col_logical()
## )
```

```
## See spec(...) for full column specifications.
```

```
pushPull.fatigue.rounds <- read.csv("H:/Git/19s-dsci353-353m-453-hes53/1-assignments/SemProj-453/Sample
```

```
colnames(bending.fatigue.flats)[colnames(bending.fatigue.flats)=="Heat Treatment"] <- "Heat.Treatment"
colnames(bending.fatigue.flats)[colnames(bending.fatigue.flats)=="Al Grade"] <- "Al.Grade"
colnames(bending.fatigue.flats)[colnames(bending.fatigue.flats)=="SiC Volume"] <- "SiC.Volume"
colnames(bending.fatigue.flats)[colnames(bending.fatigue.flats)=="Stress (Ksi)"] <- "stress"
colnames(bending.fatigue.flats)[colnames(bending.fatigue.flats)=="Load (Lbs)"] <- "load"
colnames(bending.fatigue.flats)[colnames(bending.fatigue.flats)=="Cycles"] <- "cycles"
#The column names in bending fatigue were changed as they were read in. These were changed for
#manipulation later on. As seen in the next code chunk only certain columns were used so these
#were the only ones changed
```

4.2.1.2 Cleaning each data set individually to get into standard form

```
tension.flats1 <- tension.flats %>%
dplyr::select(Heat.Treatment, Al.Grade, SiC.Volume, SiCp.Size..um., UTS..MPa.)
bending.fatigue.flats1 <- bending.fatigue.flats %>%
dplyr::select(Heat.Treatment, Al.Grade,  SiC.Volume, stress, load, cycles)
pushPull.fatigue.rounds1 <- pushPull.fatigue.rounds %>%
select(-Local.Stress, -Min.Stress, -Fracture.Deviation.D....,
       -Initiation, -Defect, -ID)
#These data sets had many repetitive columns
```

```
#and many columns that are not useful for the
#analysis. Data sets with 1 after the name represents
#clean data sets that only consist of the columns needed for analysis.
```

```
tension.flats2 <- drop_na(tension.flats1)
bending.fatigue.flats2 <- drop_na(bending.fatigue.flats1)
pushPull.fatigue.rounds2 <- drop_na(pushPull.fatigue.rounds1)
#These data sets had many missing data
#values represented as NA. This code gets rid of
#these missing values.
```

```
pushPull.fatigue.rounds2$Heat.Treatment <- "T4"
pushPull.fatigue.rounds2$Al.Grade <- "2124"
pushPull.fatigue.rounds2$SiC.Volume <- "25"
pushPull.fatigue.rounds3 <- pushPull.fatigue.rounds2[c(4,5,6,1,2,3)]
#This adds in details on the heat treatment, aluminium
#grade, and silicon carbide volume percent
#that was not previously in the data.
#The last line of code puts the columns
#in the order of the other data sets.
```

4.2.1.3 Combining data sets into one master data set

```
mechanical.data <- rbind.fill(tension.flats2, bending.fatigue.flats2, pushPull.fatigue.rounds3)
#This binds all three data sets into one,
#filling in missing data with NA.
```

```
Heat.Treatment <- factor(c("As Extruded", "T4", "T6", "T6", "T4", "As Extruded"))
levels(Heat.Treatment)
```

```
## [1] "As Extruded" "T4"          "T6"
```

```
nlevels(Heat.Treatment)
```

```
## [1] 3
```

```
Al.Grade <- factor(c("2124", "6061", "6061", "2124"))
levels(Al.Grade)
```

```
## [1] "2124" "6061"
```

```
nlevels(Al.Grade)
```

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

```
Cone <- factor(c("yes", "no", "no", "yes"))
levels(Cone)
```

```
## [1] "no"  "yes"
```

```
nlevels(Cone)
```

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

```
#This code defines the categorical data (heat treatment,
#aluminium grade, and silicon carbide percent)
#as factors which will be useful when
#building models, making graphs, and comparing between different variations
#mechanical.data$Cycles <- as.numeric(as.character(mechanical.data$Cycles))
```

4.2.2 Data Book

The final data set has 94 observations and 11 variables but does have a significant amount of missing data points, as again many tests were not run on each variation of MMC, these missing points are represented by NA.

The variables and units on variables are shown below:

- Heat. Treatment: Factor variable denoting the heat treatment that the sample underwent. "As Extruded" corrosponds to samples with no heat treatment, "T4" represents the sample being solution heat treated and naturally aged, "T6" samples were also solution heat treated and then artificially aged.
- Al. Grade: Factor variable noting the aluminium grade used for the matrix, either 6061 or 2124.
- SiC.Volume: Shows the percent of silicon carbide particles put into the MMC, either 20%, 25%, 40%
- SiCp..Size.um: Notes the size in micrometers of silicon carbide particles put into MMCs. Size is either 0.7um or 3.0um the MMC.
- UTS..MPa.: These values are the ultimate tensile strength found in tensile testing. Values are in units of MPa.
- stress: Stress that bending fatigue was run in. Stress was measured in ksi
- Load: Load at which bending fatigue tests were run at. Load is measured in pounds.
- cycles: the number of cycles until failure occurred in bending fatigue
- Load..N.: Load that push pull testing was run in. Load values are measured in newtons.
- Nf..cycles.: Cycles to failure in push pull fatigue testing.
- Cone: Factor variable denoting if a cone feature was found on push pull fatigue tests fracture surfaces

4.3 Exploratory Data Analysis

```
library(tidyverse)
```

```
summary(mechanical.data)
```

```
##  Heat.Treatment       Al.Grade          SiC.Volume        SiCp.Size..um.
##  Length:94          Length:94          Length:94          Min.   :0.700
##  Class :character   Class :character   Class :character   1st Qu.:0.700
##  Mode  :character   Mode  :character   Mode  :character   Median :3.000
##                                                           Mean   :2.233
##                                                           3rd Qu.:3.000
##                                                           Max.   :3.000
##                                                           NA's   :61
##    UTS..MPa.         stress           load           cycles
##  Min.   :387.0   Min.   :36.90   Min.   :331.2   Min.   :    6648
##  1st Qu.:452.0   1st Qu.:49.00   1st Qu.:471.9   1st Qu.:  105412
##  Median :490.0   Median :53.00   Median :691.9   Median :  278348
##  Mean   :497.1   Mean   :55.83   Mean   :642.8   Mean   : 3692789
##  3rd Qu.:520.0   3rd Qu.:61.25   3rd Qu.:758.9   3rd Qu.:10000000
##  Max.   :651.0   Max.   :81.00   Max.   :893.3   Max.   :10000000
##  NA's   :61      NA's   :58      NA's   :58      NA's   :58
##     Load..N.      Nf..cycles.        Cone
##  Min.   :14362   Min.   :  14241       : 0
##  1st Qu.:15405   1st Qu.:  46092   no  :15
##  Median :15541   Median : 199027   yes :10
```

```
##  Mean   :15693   Mean   :2193562   NA's:69
##  3rd Qu.:15833   3rd Qu.:3967356
##  Max.   :17839   Max.   :9558731
##  NA's   :69      NA's   :69
```

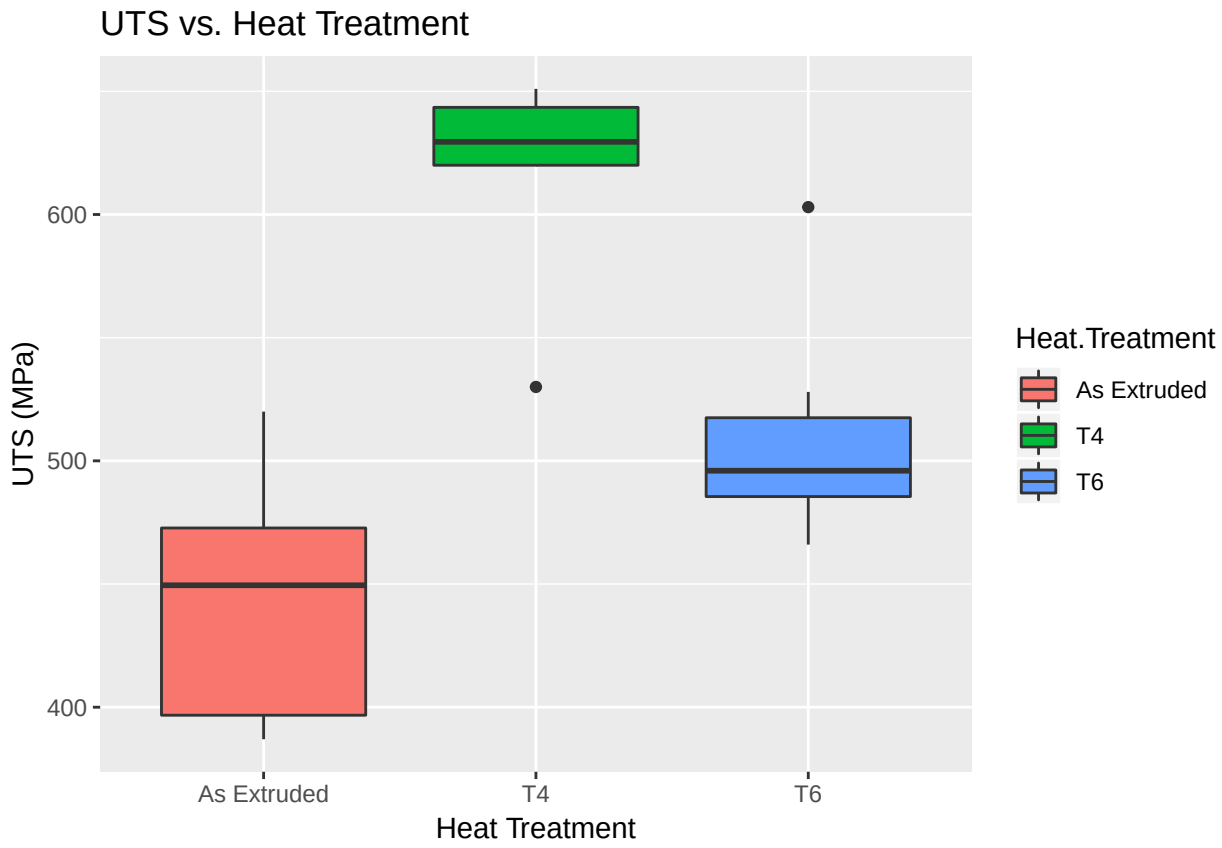
```
#This provides a summary of the mechanical data set.
```

4.3.1 Tensile Testing (UTS)

Each variation variable (independent variable) was plotted againt tensile UTS in boxplots to gain a better understanding of how that variable affects the UTS value.

```
ggplot(data = mechanical.data, aes(Heat.Treatment,UTS..MPa., fill = Heat.Treatment)) +
  geom_boxplot() +
  labs(title = "UTS vs. Heat Treatment", x = "Heat Treatment", y = "UTS (MPa)" )
```

```
## Warning: Removed 61 rows containing non-finite values (stat_boxplot).
```

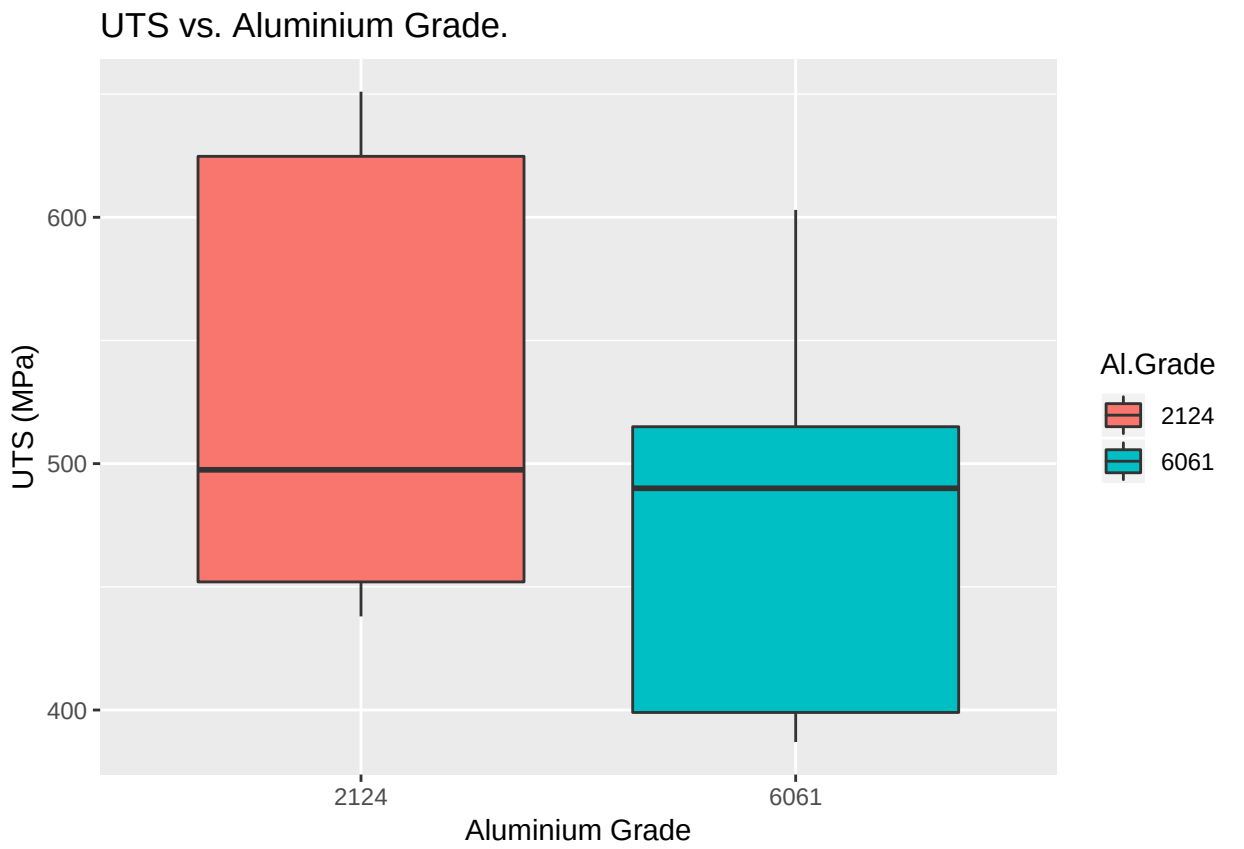



```
#This shows how the UTS in tensile testing varies through different heat treatments.
```

This plot shows that as extruded material has a lower UTS value than heat treated material. Also, the T4 heat treatment results in slightly higher UTS values than T6 as shown in this figure.

```
ggplot(data = mechanical.data, aes(Al.Grade,UTS..MPa., fill = Al.Grade)) +
  geom_boxplot() +
  labs(title = "UTS vs. Aluminium Grade.", x = "Aluminium Grade", y = "UTS (MPa)")
```

```
## Warning: Removed 61 rows containing non-finite values (stat_boxplot).
```

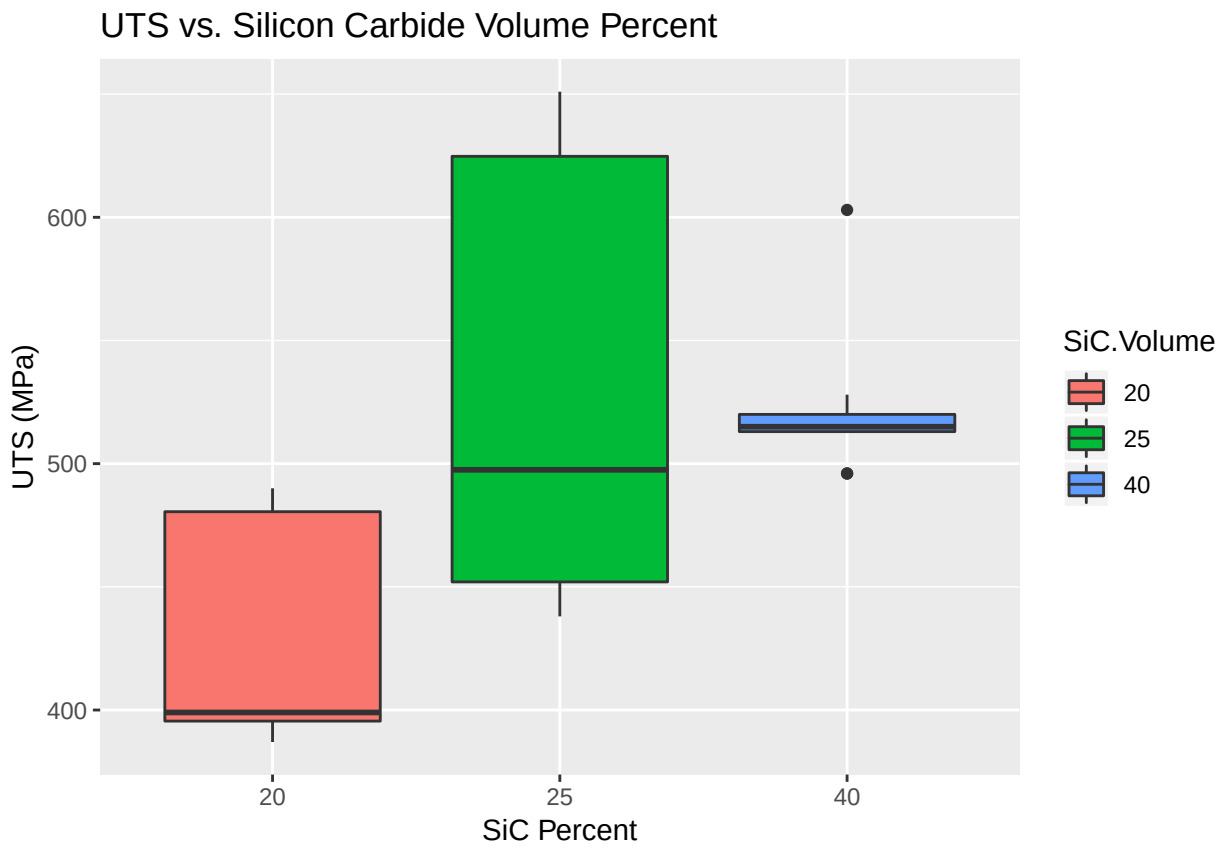



```
#This shows how the aluminium grade affects the UTS in tensile testing.
```

This plot shows that the average UTS value of samples with a 2124 aluminum matrix and a 6061 aluminum matrix are almost identical. 2124 has a larger spread and higher max and min values than 6061 does, but again on average these grades are pretty simmilar.

```
ggplot(data = mechanical.data, aes(SiC.Volume,UTS..MPa., fill = SiC.Volume)) +
  geom_boxplot() +
  labs(title = "UTS vs. Silicon Carbide Volume Percent", x = "SiC Percent", y = "UTS (MPa)")
```

```
## Warning: Removed 61 rows containing non-finite values (stat_boxplot).
```

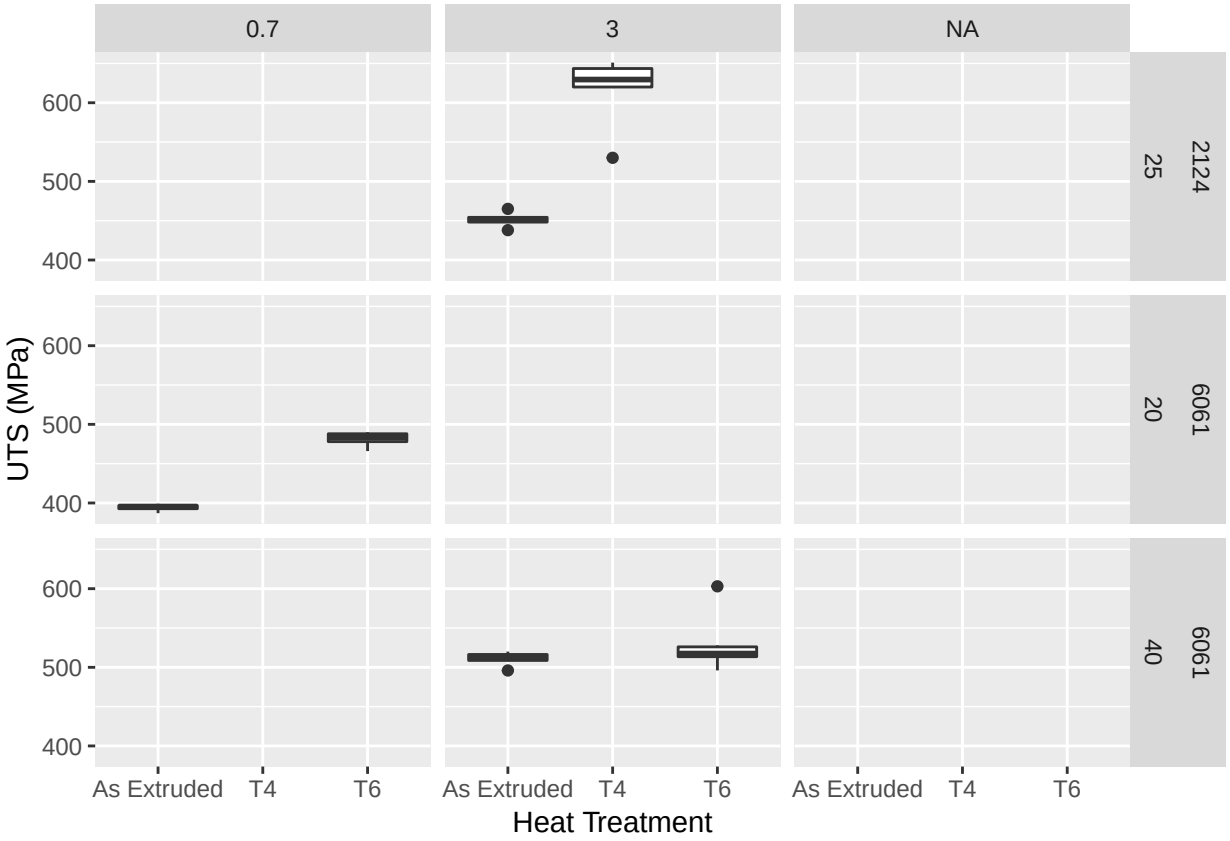



```
#This shows how the silicon carbide percent affects the UTS in tensile testing.
```

The averages of this plot show that 40% silicon carbide has a higher average UTS than 25% or 20%. However, 25% has a higher maximum value than 40% does. 25% has much more spread within its data points than 20% or 40%. 20% has the lowest values in both average values and maximum/minimum values.

```
ggplot(data = mechanical.data, aes(Heat.Treatment,UTS..MPa.)) +
  geom_boxplot() +
  facet_grid(Al.Grade ~ SiC.Volume ~ SiCp.Size..um.) +
  labs(x = "Heat Treatment", y = "UTS (MPa)")
```

```
## Warning: Removed 61 rows containing non-finite values (stat_boxplot).
```




```
#This shows how all four categorical variables affects UTS.
```

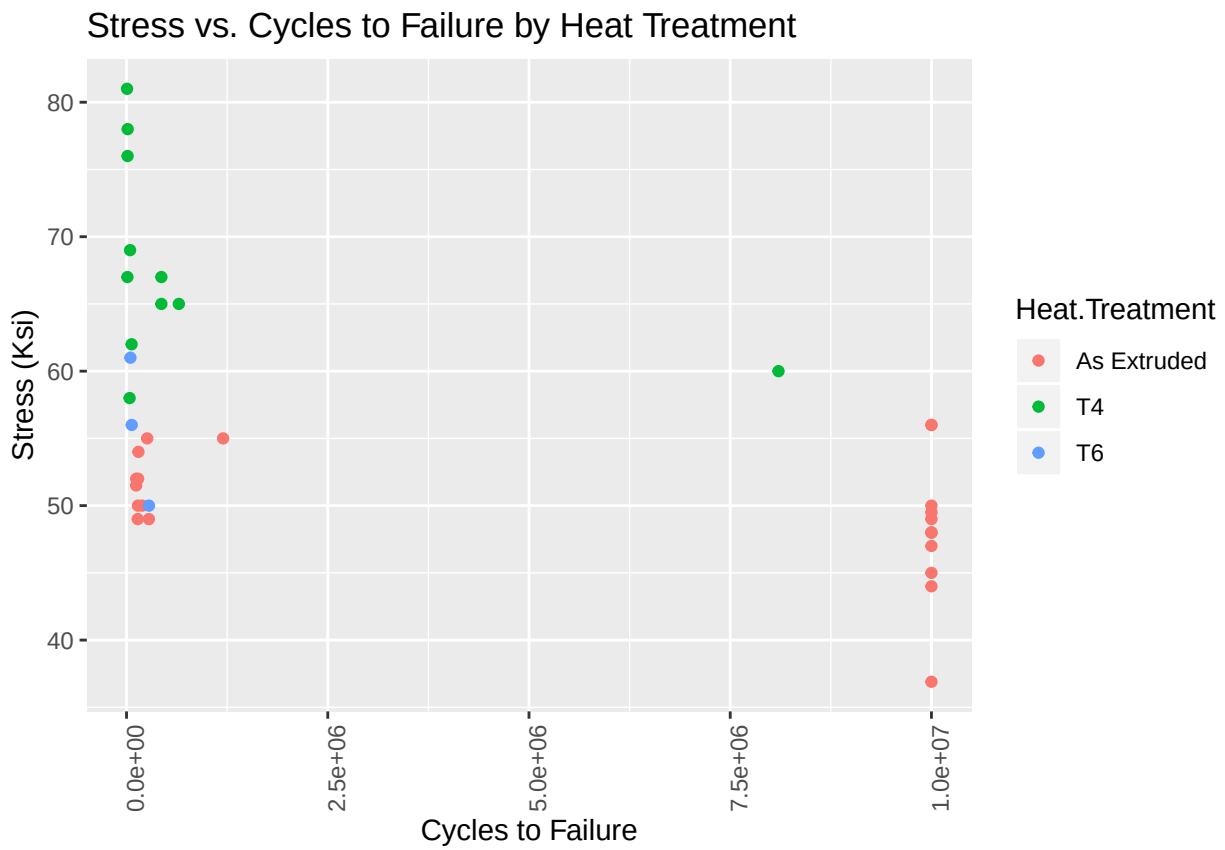
This plot shows how all four of the variables affect UTS together. Finding trends within this plot is a bit harder than some others as it is more complex, but one can see that the highest UTS values are coming from 2124 grade matrix with 25% SiC and 3um particles. The lowest UTS vallues are in 6061 matrix, 20% SiC and 0.7um particles.

4.3.2 Bending Fatigue (Cycles to failure)

Bending fatigue data was plotted in dot plots. Cycles to failure was plotted against the stress at which the test was run at, and each point was colored according to the independent variable that is being looked at.

```
ggplot(data = mechanical.data, aes(cycles, stress, color = Heat.Treatment)) +
  geom_point()+
  labs(title = "Stress vs. Cycles to Failure by Heat Treatment", x = "Cycles to Failure", y = "Stress (
  theme(axis.text.x = element_text(angle = 90))
```

```
## Warning: Removed 58 rows containing missing values (geom_point).
```

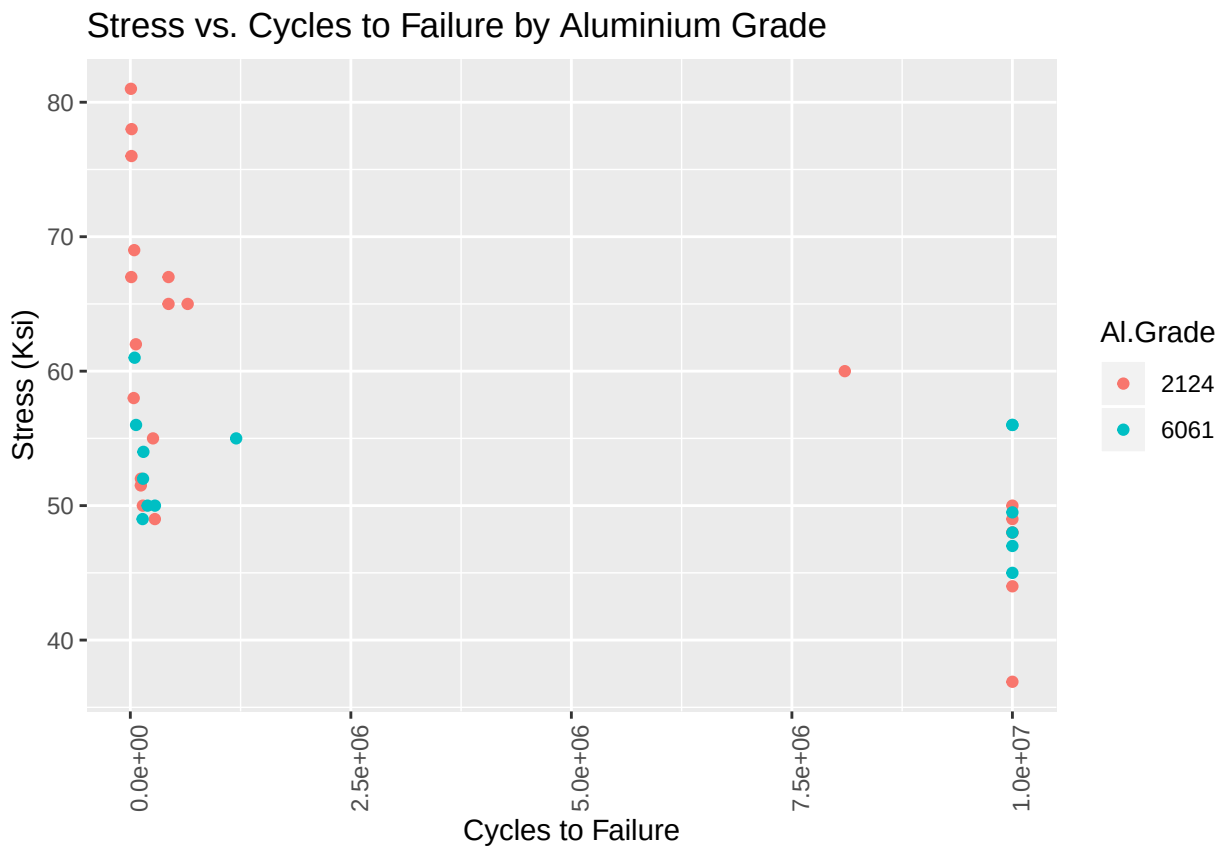



```
#This shows how the heat treatment affects the cycles to
#failure in bending fatigue. It seems that T4 and T6 heat
#treatments had higher cycles to failure than As Extruded
#did.
```

This plot shows that the T6 heat treatment had lower cycles to failure than as extruded or T4 did.

```
ggplot(data = mechanical.data, aes(cycles, stress, color = Al.Grade)) +
  geom_point()+
  labs(title = "Stress vs. Cycles to Failure by Aluminium Grade", x = "Cycles to Failure", y = "Stress
  theme(axis.text.x = element_text(angle = 90))
```

```
## Warning: Removed 58 rows containing missing values (geom_point).
```

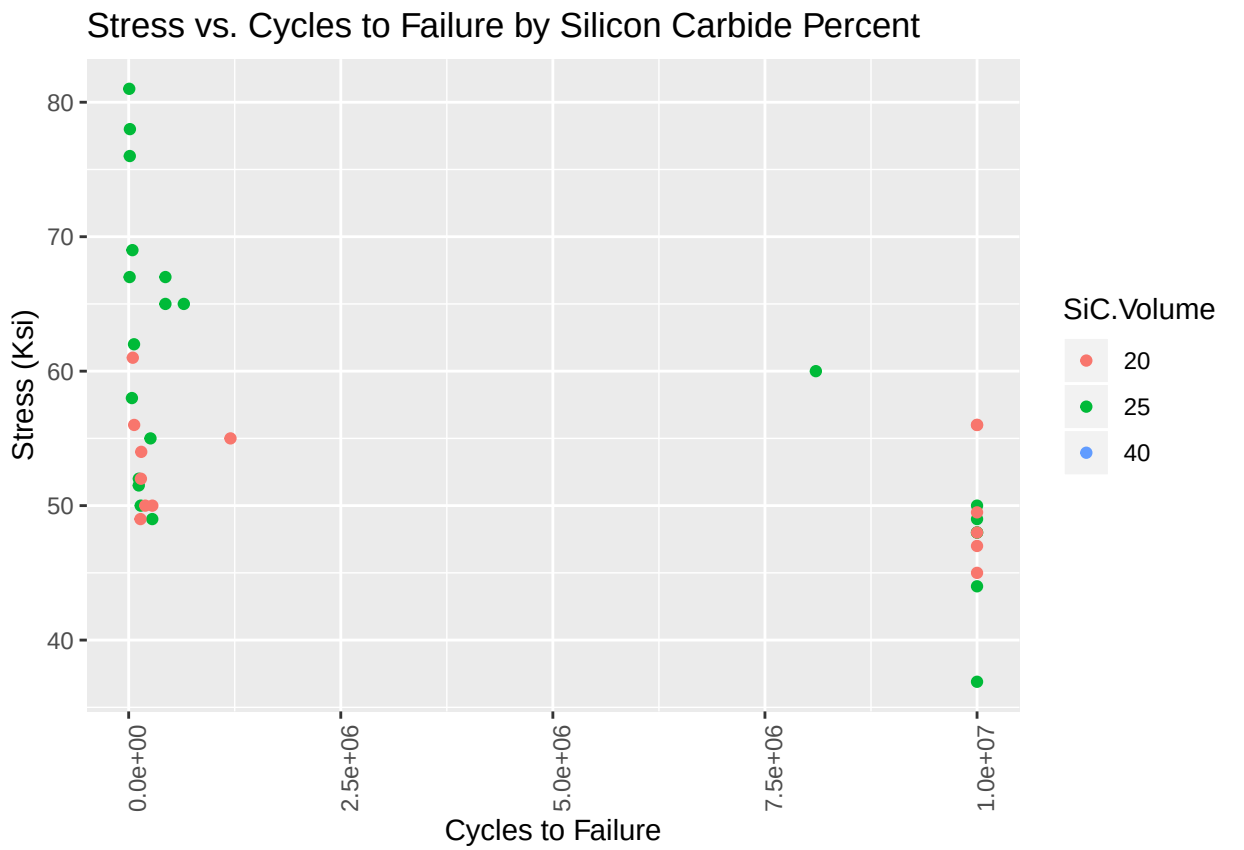



```
#This shows how the aluminium grade affects the cycles to failure in bending fatigue. There is not a cl
#in this graph.
```

This plot shows that neither aluminum grade matrix significantly outpreformed the other.

```
ggplot(data = mechanical.data, aes(cycles, stress, color = SiC.Volume)) +
  geom_point()+
  labs(title = "Stress vs. Cycles to Failure by Silicon Carbide Percent", x = "Cycles to Failure", y =
  theme(axis.text.x = element_text(angle = 90))
```

```
## Warning: Removed 58 rows containing missing values (geom_point).
```



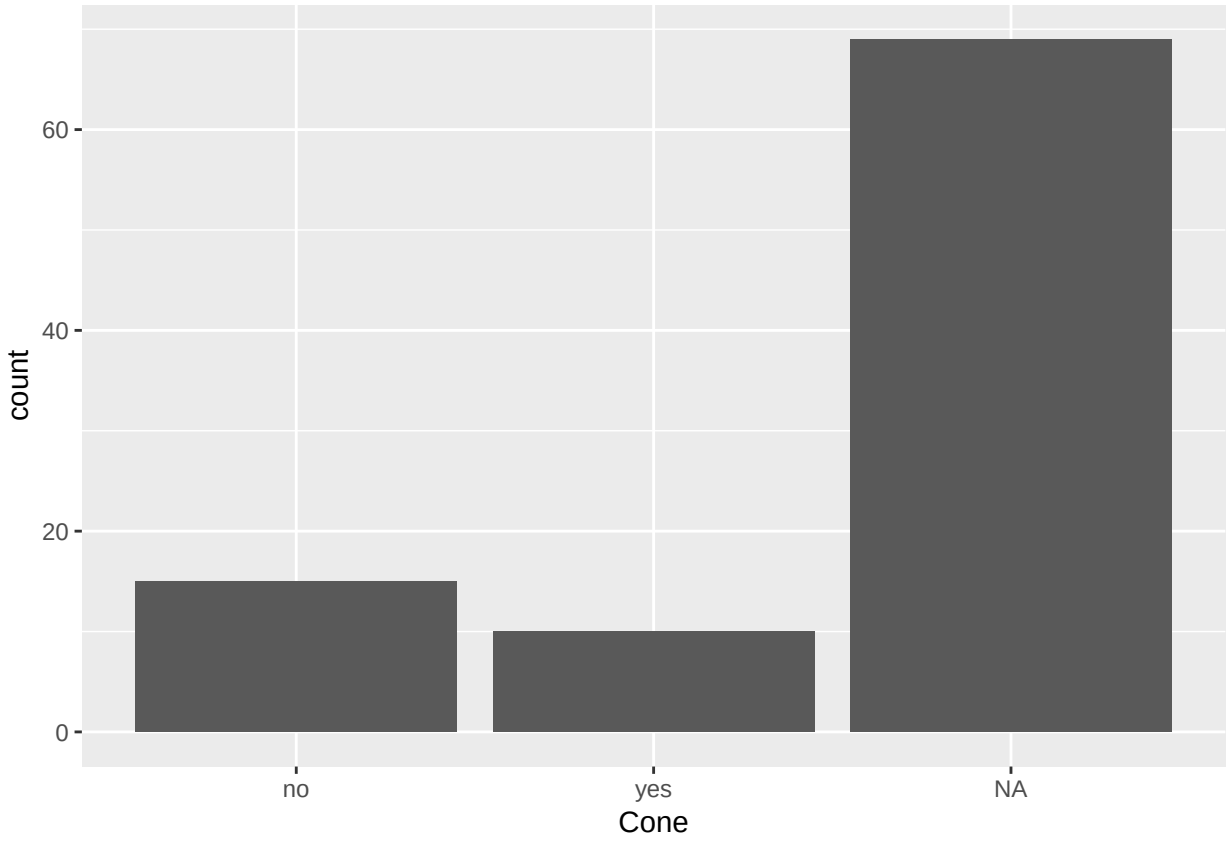

```
#This shows how the silicon carbide percent affects
#the cycles to failure in bending fatigue. It seems that
#40% silicon carbide has a much smaller fatigue life.
```

This plot shows that 20% SiC and 25% SiC were somewhat simmilar in terms of the affect they had on bending cycles to failure.

4.3.3 Push Pull Fatigue (Cycles to Failure)

Due to only one variation of MMC being tested in push pull fatigue testing, no plots could really be made to find trends in terms of the affect process parameters had on push pull cycles to failure. As noted in the introduction though unexplained cone features were seen on some of the fracture surfaces of specimens tested in this testing set. Though these are not directly being studied in this report gaining some more knowledge on them through data, and noting any differences that could be found while working through the data analysis in this report was of interest, even if this is something that can only be done in the future. Due to this interest plots corresponding to the cone surfaces were made.

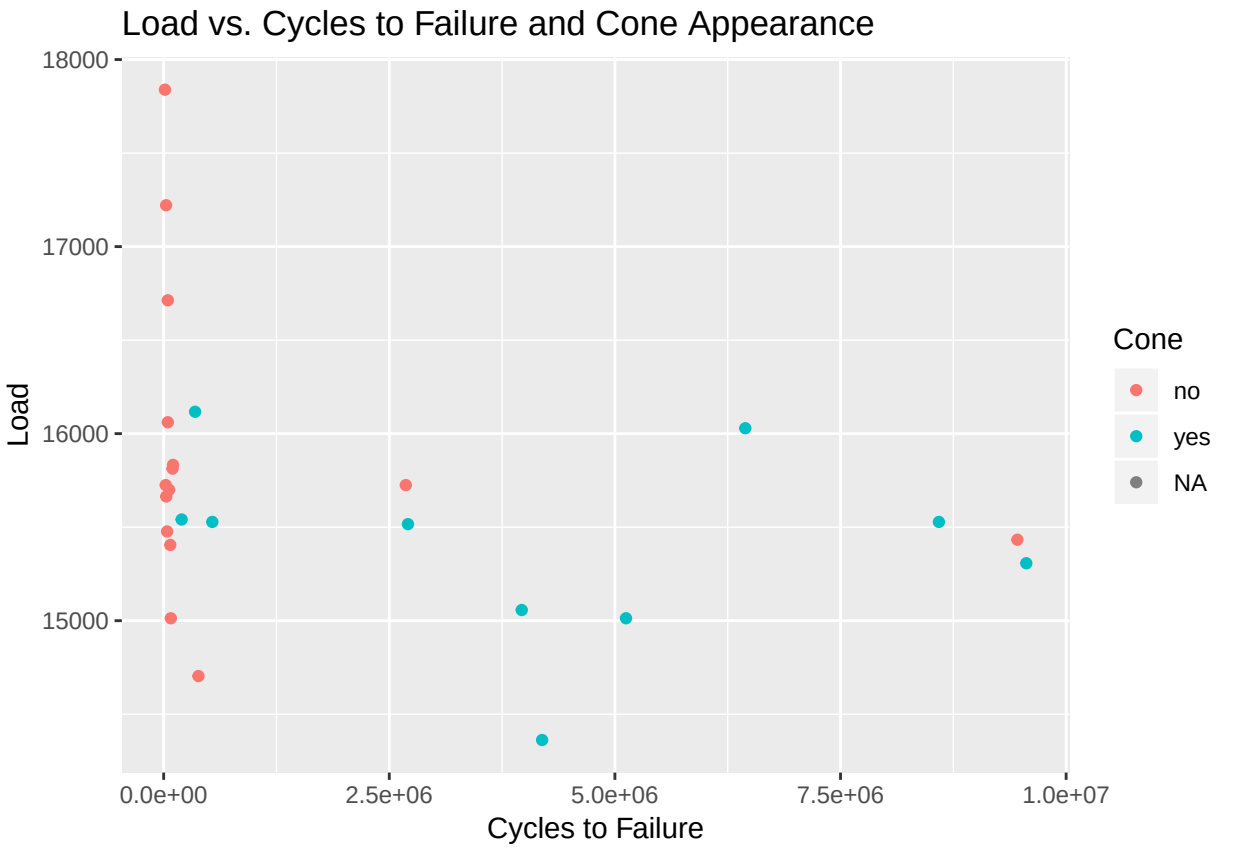
```
ggplot(data = mechanical.data, mapping = aes(x = Cone)) +
  geom_bar(na.rm = FALSE)
```



This bar chart shows how many specimens showed the cones and how many did not. As seen in the chart the majority of specimens tested did not show cones on the fracture surface. NA here represents variations of MMC that were not tested in push pull fatigue.

```
ggplot(data = mechanical.data, aes(Nf..cycles., Load..N., color = Cone)) +
    geom_point()+
  labs(title = "Load vs. Cycles to Failure and Cone Appearance", x = "Cycles to Failure", y = "Load")
```

```
## Warning: Removed 69 rows containing missing values (geom_point).
```




```
#This plots load vs cycles to failure in push pull fatigue
#and shows if a cone was present on the fracture surface or
#not.
```

This plot shows load vs cycles to failure and colors each data point by the appearance of a cone or not. It seems that specimens that had more cycles to failure were more likely to show a cone feature than others, but there are instances in which this does not hold true.

4.3.4 Correlation Plots

Pairwise correlation plots were made for both the tensile tests and bending fatigue tests. These are to help understand any relationships between the variables and understand the data better before modeling.

```
#library (psych)
#pairs.panels(mechanical.data[1:5])
#This shows correlations for variations in processing of MMCs and UTS.
```

```
#cycles <- mechanical.data [c(1:3, 8)]
#pairs.panels(cycles)
#This is a pairwise correlation plot of the varying process variables
#and cycles to failure in bending fatigue
```

5 Statistical Learning Modeling and Prediction

As discussed previously, the lack of data available in this project does complicate modeling. The first aspect of modeling that these complications are seen in stems from the fact that not every variation of MMC was tested in each type of testing. As discussed in the approach section of this report, to help with this issue

several different types of modeling were used.

The first sets of models presented are simple linear regression models for each variable in tensile testing and bending fatigue testing. Trends within each type of testing can be extracted and compared from these models. Simple linear regressions could not be run in push pull fatigue testing since only one variation of MMC was tested.

The next sets of models run were multivariate linear models on tensile testing and bending fatigue testing using all of the variation variables in one model. Again, push pull testing could not be modeled in this way since only one variation was tested. Additionally, forward step wise modeling was used on these sets of models to determine which variables are actually needed for the most accurate models.

From the multivariate modeling, the best model (determined by comparing adjusted R^2 values) was used to make a multivariate model with tensile testing UTS, bending fatigue testing cycles to failure, and push pull fatigue testing cycles to failure as responses and the variables (heat treatment, silicon carbide size, silicon carbide volume percent, aluminium grade) as predictors.

The second aspect of modeling that is complicated from lack of data is the splitting into test and train data sets. Any splitting method is going to take away from the already small data sets which could be an issue. To make sure that the correct splitting method is used several different methods were tried in each of the sets of models discussed above. These methods included a 75%-25% split, leave one out cross validation (LOOCV), and bootstrapping. The 75%-25% split randomly extracted 25% of the data out for testing, creating a training set from the remaining 75%. LOOCV uses one data point as the test set, and bootstrapping is a resampling method.

5.1 Simple Linear Regression Models

5.1.1 Tensile Testing

5.1.1.1 75%-25% Test/ Train Split

```
smp_size <- floor(0.75 * nrow(mechanical.data))
set.seed(123)
train_ind <- sample(seq_len(nrow(mechanical.data)), size = smp_size )
train <- mechanical.data[train_ind, ]
test <- mechanical.data[-train_ind, ]
#Split into test and train data sets. Split is 75% of data
# in training data set and 25% in testing data set
```

```
set.seed(1)
tens_ht_fit_split <- lm (UTS..MPa. ~ Heat.Treatment, data = train)
#Fit linear model of UTS based on heat treatment on train data set
summary (tens_ht_fit_split)
```

```
##
## Call:
## lm(formula = UTS..MPa. ~ Heat.Treatment, data = train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -87.50 -14.38   2.50  25.16  58.38
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)        437.62      12.51  34.972  < 2e-16 ***
## Heat.TreatmentT4   179.88      19.11   9.410 1.39e-08 ***
## Heat.TreatmentT6    56.00      17.70   3.164  0.00511 **
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 35.39 on 19 degrees of freedom
##   (48 observations deleted due to missingness)
## Multiple R-squared:  0.8263, Adjusted R-squared:  0.808
## F-statistic: 45.18 on 2 and 19 DF,  p-value: 6.011e-08
```

```
#Detailed information on model built
library(broom)
tidy(tens_ht_fit_split)
```

```
## # A tibble: 3 x 5
##   term             estimate std.error statistic  p.value
##   <chr>               <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)         438.       12.5     35.0  1.03e-18
## 2 Heat.TreatmentT4    180.       19.1      9.41 1.39e- 8
## 3 Heat.TreatmentT6     56.0      17.7      3.16 5.11e- 3
```

```
confint(tens_ht_fit_split)
```

```
##                       2.5 %    97.5 %
## (Intercept)      411.43399 463.81601
## Heat.TreatmentT4 139.86757 219.88243
## Heat.TreatmentT6  18.96031  93.03969
```

```
#Obtain confidence intervals from model built
p.ht.t <- predict(tens_ht_fit_split, data = test)
#predict values for the test data set using the model built
test.UTS <- na.omit(test$UTS..MPa.)
#Omit na values from the test values
test.MSE.ht.t <- sqrt(mean((p.ht.t- test.UTS) ^ 2))
#Compute the test MSE
test.MSE.ht.t
```

```
## [1] 111.9331
```

This models tensile UTS by heat treatment. As seen in the coefficients both heat treatments have a positive relationship with UTS suggesting that heat treatments do help to increase tensile UTS. p- values for both are also low meaning there is some significance associated with them. The adjusted R^2 value is about 80% which is somewhat high.

```
set.seed(1)
tens_algrade_fit_split <- lm (UTS..MPa. ~ Al.Grade, data = train)
#Fit linear model of UTS based on aluminum grade on train data set
summary (tens_algrade_fit_split)
```

```
##
## Call:
## lm(formula = UTS..MPa. ~ Al.Grade, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -100.90  -72.83   16.67   57.17   98.10
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    552.90      22.19  24.921   <2e-16 ***
```

```
## Al.Grade6061   -84.07      30.04  -2.798   0.0111 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 70.16 on 20 degrees of freedom
##   (48 observations deleted due to missingness)
## Multiple R-squared:  0.2814, Adjusted R-squared:  0.2455
## F-statistic: 7.831 on 1 and 20 DF,  p-value: 0.0111
```

```
#Detailed information on model built
tidy(tens_algrade_fit_split)
```

```
## # A tibble: 2 x 5
##   term         estimate std.error statistic  p.value
##   <chr>           <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)     553.       22.2     24.9  1.56e-16
## 2 Al.Grade6061    -84.1      30.0     -2.80 1.11e- 2
```

```
confint(tens_algrade_fit_split)
```

```
##                  2.5 %    97.5 %
## (Intercept)   506.6197 599.18032
## Al.Grade6061 -146.7305 -21.40281
```

```
#Obtain confidence intervals from model built
p.al.t <- predict(tens_algrade_fit_split, data = test)
#predict values for the test data set using the model built
tens.UTS <- na.omit(test$UTS..MPa.)
#Omit na values from the test values
test.MSE.al.t <- sqrt(mean((p.al.t- tens.UTS) ^ 2))
#Compute the test MSE
test.MSE.al.t
```

```
## [1] 89.39752
```

This models the tensile UTS with aluminium grade. The coefficient of this model shows that the 6061 grade has a negative relationship with UTS, meaning it actually lowers the UTS values. The ajusted R^2 value associated with this model is only about 24% which is very low, meaning the model is not fitting the training data very well.

```
set.seed(1)
tens_sicvol_fit_split <- lm (UTS..MPa. ~ SiC.Volume, data = train)
#Fit linear model of UTS based on silicon volume percent on train data set
summary (tens_sicvol_fit_split)
```

```
##
## Call:
## lm(formula = UTS..MPa. ~ SiC.Volume, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -100.90  -52.25   13.88   41.25   98.10
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    448.25      24.10  18.600 1.19e-13 ***
## SiC.Volume25   104.65      32.33   3.237  0.00434 **
## SiC.Volume40    61.75      41.74   1.479  0.15544
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 68.16 on 19 degrees of freedom
##   (48 observations deleted due to missingness)
## Multiple R-squared:  0.3556, Adjusted R-squared:  0.2878
## F-statistic: 5.242 on 2 and 19 DF,  p-value: 0.01538
```

```
#Detailed information on model built
tidy(tens_sicvol_fit_split)
```

```
## # A tibble: 3 x 5
##   term         estimate std.error statistic  p.value
##   <chr>           <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)     448.       24.1     18.6  1.19e-13
## 2 SiC.Volume25    105.       32.3      3.24 4.34e- 3
## 3 SiC.Volume40     61.7      41.7      1.48 1.55e- 1
```

```
confint(tens_sicvol_fit_split)
```

```
##                  2.5 %   97.5 %
## (Intercept)  397.80890 498.6911
## SiC.Volume25  36.97616 172.3238
## SiC.Volume40 -25.61655 149.1166
```

```
#Obtain confidence intervals from model built
p.sicvol.t <- predict(tens_sicvol_fit_split, data = test)
#predict values for the test data set using the model built
tens.UTS <- na.omit(test$UTS..MPa.)
#Omit na values from the test values
test.MSE.sicvol.t <- sqrt(mean((p.sicvol.t- tens.UTS) ^ 2))
#Compute the test MSE
test.MSE.sicvol.t
```

```
## [1] 90.88179
```

This models the tensile UTS values with silicon carbide volume percent. Coefficients show a positive relationship between both 25% and 40%. The adjusted R^2 value is about 28% which points to the model not fitting the training data very well.

```
set.seed(1)
tens_sicsize_fit_split <- lm (UTS..MPa. ~ SiCp.Size..um., data = train)
#Fit linear model of UTS based on silicon carbide size on train data set
summary (tens_sicsize_fit_split)
```

```
##
## Call:
## lm(formula = UTS..MPa. ~ SiCp.Size..um., data = train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -88.64 -52.25 -11.64  41.25 110.36
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      420.13      32.02  13.119 2.77e-11 ***
## SiCp.Size..um.    40.17      13.18   3.048  0.00635 **
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 68.39 on 20 degrees of freedom
##   (48 observations deleted due to missingness)
## Multiple R-squared:  0.3172, Adjusted R-squared:  0.2831
## F-statistic: 9.292 on 1 and 20 DF,  p-value: 0.006346
```

```
#Detailed information on model built
tidy(tens_sicsize_fit_split)
```

```
## # A tibble: 2 x 5
##   term           estimate std.error statistic  p.value
##   <chr>             <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)       420.       32.0     13.1  2.77e-11
## 2 SiCp.Size..um.     40.2      13.2      3.05 6.35e- 3
```

```
confint(tens_sicsize_fit_split)
```

```
##                    2.5 %    97.5 %
## (Intercept)    353.32858 486.93229
## SiCp.Size..um.  12.68162  67.65999
```

```
#Obtain confidence intervals from model built
p.sicsize.t <- predict(tens_sicsize_fit_split, data = test)
#predict values for the test data set using the model built
tens.UTS <- na.omit(test$UTS..MPa.)
#Omit na values from the test values
test.MSE.sicsize.t <- sqrt(mean((p.sicsize.t- tens.UTS) ^ 2))
#Compute the test MSE
test.MSE.sicsize.t
```

```
## [1] 87.2218
```

This tries to model the relationship between silicon carbide size and tensile UTS. The ajusted R^2 value is about 0.01% which is very low.

Most of these models did have very low adjusted R^2 values, which could be due to a severe lack of data after splitting.

5.1.1.2 Leave One Out Cross Validation

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.5.3
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
set.seed(1)
tens_ht_fit_loocv <- train (UTS..MPa. ~ Heat.Treatment, method = "lm", data = tension.flats2, trControl
summary(tens_ht_fit_loocv)
```

```
##
## Call:
```

```
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -87.50 -29.27   2.50  20.73  95.73
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)        444.94      10.96  40.582  < 2e-16 ***
## Heat.TreatmentT4   172.56      20.99   8.220 3.56e-09 ***
## Heat.TreatmentT6    62.34      17.18   3.629  0.00105 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 43.86 on 30 degrees of freedom
## Multiple R-squared:  0.6953, Adjusted R-squared:  0.675
## F-statistic: 34.23 on 2 and 30 DF,  p-value: 1.814e-08
```

```
#This fits a linear regression model using leave one out
#cross validation as the test/train splitting method.
```

This models tensile UTS by heat treatment but using LOOCV. This follows the same model created with the 75-25 split in which both heat treatments have a positive relationship to UTS. The adjusted R^2 value here is 68% which is lower than the 75-25 split model.

```
set.seed(1)
tens_algrade_fit_loocv <- train (UTS..MPa. ~ Al.Grade, method = "lm", data = tension.flats2, trControl =
summary(tens_algrade_fit_loocv)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -96.50 -79.50  12.29  44.29 127.29
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 566.214721  33.332996  16.987   <2e-16 ***
## Al.Grade     -0.014932   0.006664  -2.241   0.0324 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 72.5 on 31 degrees of freedom
## Multiple R-squared:  0.1394, Adjusted R-squared:  0.1116
## F-statistic:  5.02 on 1 and 31 DF,  p-value: 0.03236
```

```
#This fits a linear regression model using leave one out
#cross validation as the test/train splitting method.
```

This models the tensile UTS with aluminium matrix grade using LOOCV. The adjusted R^2 for this model is only about 2% higher than the 75-25 split model created, and at 11% is still very low.

```
set.seed(1)
tens_sicvol_fit_loocv <- train (UTS..MPa. ~ SiC.Volume, method = "lm", data = tension.flats2, trControl
summary(tens_sicvol_fit_loocv)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -84.87 -39.89 -20.89  16.13 163.12
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  407.860     44.880   9.088 2.99e-10 ***
## SiC.Volume     3.201      1.543   2.074   0.0465 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 73.24 on 31 degrees of freedom
## Multiple R-squared:  0.1218, Adjusted R-squared:  0.09348
## F-statistic:   4.3 on 1 and 31 DF,  p-value: 0.04651
```

```
#This fits a linear regression model using leave one out
#cross validation as the test/train splitting method.
```

This models tensile UTS with silicon carbide volume perecent. Again the R^2 value is very low.

```
set.seed(1)
tens_sicsize_fit_loocv <- train (UTS..MPa. ~ SiCp.Size..um., method = "lm", data = tension.flats2, trCo
summary(tens_sicsize_fit_loocv)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -90.77 -38.73 -13.77  49.27 122.23
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      404.80      25.12  16.116  < 2e-16 ***
## SiCp.Size..um.    41.32      10.12   4.084 0.000289 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 63.02 on 31 degrees of freedom
## Multiple R-squared:  0.3499, Adjusted R-squared:  0.3289
## F-statistic: 16.68 on 1 and 31 DF,  p-value: 0.0002887
```

```
#This fits a linear regression model using leave one out
#cross validation as the test/train splitting method.
```

This models the tensile UTS with silicon carbide size. The adjusted R^2 value is about 33% which is low.

5.1.2 Bending Fatigue Testing

5.1.2.1 75%-25% Split

```r
set.seed(1)
bend_ht_fit_split <- lm (cycles ~ Heat.Treatment, data = train)
#Fit linear model of Cycles to failure based on heat treatment on train data set
summary(bend_ht_fit_split)
```

```
##
## Call:
## lm(formula = cycles ~ Heat.Treatment, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5951733  -928489  -504313  3929267  7162687
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        6070733    1060462   5.725 6.73e-06 ***
## Heat.TreatmentT4 -5133420    1676737  -3.062  0.00536 **
## Heat.TreatmentT6 -6014205    3091751  -1.945  0.06355 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4107000 on 24 degrees of freedom
##   (43 observations deleted due to missingness)
## Multiple R-squared:  0.3161, Adjusted R-squared:  0.2592
## F-statistic: 5.548 on 2 and 24 DF,  p-value: 0.01046
```

```r
#Detailed information on model built
tidy(bend_ht_fit_split)
```

```
## # A tibble: 3 x 5
##   term              estimate std.error statistic    p.value
##   <chr>                <dbl>     <dbl>     <dbl>      <dbl>
## 1 (Intercept)       6070733.  1060462.      5.72 0.00000673
## 2 Heat.TreatmentT4 -5133420.  1676737.     -3.06 0.00536
## 3 Heat.TreatmentT6 -6014205.  3091751.     -1.95 0.0636
```

```r
confint(bend_ht_fit_split)
```

```
##                      2.5 %     97.5 %
## (Intercept)        3882048  8259418.9
## Heat.TreatmentT4  -8594036 -1672804.3
## Heat.TreatmentT6 -12395266   366854.8
```

```r
#Obtain confidence intervals from model built
p.ht.b <- predict(bend_ht_fit_split, data = test)
#predict values for the test data set using the model built
bend.cycles <- na.omit(test$Cycles)
#Omit na values from the test values
test.MSE.ht.b <- sqrt(mean((p.ht.b- bend.cycles) ^ 2))
#Compute the test MSE
test.MSE.ht.b
```

```
## [1] NaN
```

This models bending fatigue with heat treatment. Both of the heat treatments have a negative relationship with bending fatigue cycles to failure. The adjusted R^2 value for this model is only about 26% which is pretty low.

```r
set.seed(1)
bend_algrade_fit_split <- lm (cycles ~ Al.Grade, data = train)
#Fit linear model of Cycles to failure based on aluminium grade on train data set
summary(bend_algrade_fit_split)
```

```
##
## Call:
## lm(formula = cycles ~ Al.Grade, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4294455 -3493614 -3250357  5657135  6492643
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3507357    1084798   3.233  0.00343 **
## Al.Grade6061   835509    2130501   0.392  0.69826
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4851000 on 25 degrees of freedom
##   (43 observations deleted due to missingness)
## Multiple R-squared:  0.006114,   Adjusted R-squared:  -0.03364
## F-statistic: 0.1538 on 1 and 25 DF,  p-value: 0.6983
```

```r
#Detailed information on model built
tidy(bend_algrade_fit_split)
```

```
## # A tibble: 2 x 5
##   term         estimate std.error statistic p.value
##   <chr>           <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)  3507357.  1084798.      3.23  0.00343
## 2 Al.Grade6061  835508.  2130501.      0.392 0.698
```

```r
confint(bend_algrade_fit_split)
```

```
##                 2.5 %  97.5 %
## (Intercept)   1273174 5741540
## Al.Grade6061 -3552340 5223357
```

```r
#Obtain confidence intervals from model built
```

This model is looking at the relationship between cycles to failure and aluminium grade. 6061 has a negative relationship with bending cycles to failure. The adjusted R^2 value is actually negative meaning this model is not closely showing the data in any way.

```r
set.seed(1)
bend_sicvol_fit_split <- lm (cycles ~ SiC.Volume, data = train)
#Fit linear model of Cycles to failure based on silicon carbide volume on train data set
summary(bend_sicvol_fit_split)
```

```
##
## Call:
## lm(formula = cycles ~ SiC.Volume, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -4294455 -3493614 -3250357  5657135  6492643
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4342865    1833643   2.368   0.0259 *
## SiC.Volume25  -835509    2130501  -0.392   0.6983
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4851000 on 25 degrees of freedom
##   (43 observations deleted due to missingness)
## Multiple R-squared:  0.006114,   Adjusted R-squared:  -0.03364
## F-statistic: 0.1538 on 1 and 25 DF,  p-value: 0.6983
```

```
#Detailed information on model built
tidy(bend_sicvol_fit_split)
```

```
## # A tibble: 2 x 5
##   term          estimate std.error statistic p.value
##   <chr>            <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)   4342865.  1833643.     2.37   0.0259
## 2 SiC.Volume25  -835508.  2130501.    -0.392  0.698
```

```
confint(bend_sicvol_fit_split)
```

```
##                   2.5 %  97.5 %
## (Intercept)    566406.7 8119324
## SiC.Volume25 -5223356.7 3552340
```

```
#Obtain confidence intervals from model built
```

This model is looking at bending cycles to failure and silicon carbide volume percent. 25% SiC has a negative relationship with cycles to failure, but the adjusted R^2 value is negative so this model is not capturing the data very well.

5.1.2.2 Leave One Out Cross Validation

```
set.seed(1)
bend_ht_fit_loocv <- train (cycles ~ Heat.Treatment, method = "lm", data = bending.fatigue.flats2, trCo
summary(bend_ht_fit_loocv)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5460205 -4609705  -458467  4420795  7208533
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       5579205     889666   6.271 4.35e-07 ***
## Heat.TreatmentT4 -4687739    1540947  -3.042  0.00458 **
## Heat.TreatmentT6 -5448621    2568245  -2.122  0.04147 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 4173000 on 33 degrees of freedom
## Multiple R-squared:  0.2607, Adjusted R-squared:  0.2159
## F-statistic:  5.82 on 2 and 33 DF,  p-value: 0.006841
```

```
#This fits a linear regression model using leave one out
#cross validation as the test/train splitting method.
```

This model is again looking at cycles to failure and heat treatment but is using LOOCV instead of the random test/train split. This model also shows that both T4 and T6 heat treatments have negative relationships with bending cycles to failure. The adjusted R^2 value is a bit lower than the test/train split was.

```
set.seed(1)
bend_algrade_fit_loocv <- train (cycles ~ Al.Grade, method = "lm", data = bending.fatigue.flats2, trCon
summary(bend_algrade_fit_loocv)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4395752 -3204730 -3015142  5555838  6785358
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2551320.2  1699803.8   1.501    0.143
## Al.Grade        312.3      411.7   0.758    0.453
##
## Residual standard error: 4741000 on 34 degrees of freedom
## Multiple R-squared:  0.01664,    Adjusted R-squared:  -0.01228
## F-statistic: 0.5753 on 1 and 34 DF,  p-value: 0.4534
```

```
#This fits a linear regression model using leave one out
#cross validation as the test/train splitting method.
```

This model uses LOOCV to look at aluminium grade and bending cycles to failure. The adjusted R^2 value is negative so this model is not modeling the data very well.

```
set.seed(1)
bend_sicvol_fit_loocv <- train (cycles ~ SiC.Volume, method = "lm", data = bending.fatigue.flats2, trCo
summary(bend_sicvol_fit_loocv)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4395752 -3204730 -3015142  5555838  6785358
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9362240    7516395   1.246    0.221
## SiC.Volume   -245904     324206  -0.758    0.453
##
## Residual standard error: 4741000 on 34 degrees of freedom
## Multiple R-squared:  0.01664,    Adjusted R-squared:  -0.01228
```

```
## F-statistic: 0.5753 on 1 and 34 DF,  p-value: 0.4534
```

```
#This fits a linear regression model using leave one out
#cross validation as the test/train splitting method.
```

Again, the ajusted R^2 value of this model is negative so any findings from it really are not true as it is not modeling the data very correctly.

5.2 Multivariate Modeling

5.2.1 Tensile Testing

```
set.seed (10)
tensile.linmodel.split <- lm (UTS..MPa. ~ Heat.Treatment + Al.Grade + SiC.Volume + SiCp.Size..um., data
summary (tensile.linmodel.split)
```

```
##
## Call:
## lm(formula = UTS..MPa. ~ Heat.Treatment + Al.Grade + SiC.Volume +
##     SiCp.Size..um., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -87.500  -6.929   1.786  13.107  36.786
##
## Coefficients: (2 not defined because of singularities)
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)        456.00      14.17  32.189  < 2e-16 ***
## Heat.TreatmentT4   161.50      18.29   8.831 9.27e-08 ***
## Heat.TreatmentT6    67.71      17.49   3.872  0.00122 **
## Al.Grade6061       -50.07      20.51  -2.442  0.02584 *
## SiC.Volume25           NA         NA      NA       NA
## SiC.Volume40        53.29      17.49   3.047  0.00728 **
## SiCp.Size..um.         NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 28.33 on 17 degrees of freedom
##   (48 observations deleted due to missingness)
## Multiple R-squared:  0.9004, Adjusted R-squared:  0.8769
## F-statistic: 38.41 on 4 and 17 DF,  p-value: 2.648e-08
```

```
tidy(tensile.linmodel.split)
```

```
## # A tibble: 5 x 5
##   term             estimate std.error statistic  p.value
##   <chr>               <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)         456.       14.2     32.2  1.13e-16
## 2 Heat.TreatmentT4    161.       18.3      8.83 9.27e- 8
## 3 Heat.TreatmentT6     67.7      17.5      3.87 1.22e- 3
## 4 Al.Grade6061        -50.1      20.5     -2.44 2.58e- 2
## 5 SiC.Volume40         53.3      17.5      3.05 7.28e- 3
```

```
confint(tensile.linmodel.split)
```

```
##                     2.5 %     97.5 %
## (Intercept)      426.11142 485.888576
```

```
## Heat.TreatmentT4 122.91401 200.085986
## Heat.TreatmentT6  30.81903 104.609545
## Al.Grade6061     -93.33495  -6.807902
## SiC.Volume25            NA         NA
## SiC.Volume40      16.39046  90.180974
## SiCp.Size..um.          NA         NA
```

```
#Linear model of tensile properties
pred.tensile.linmodel.split <- predict(tensile.linmodel.split, test)
```

```
## Warning in predict.lm(tensile.linmodel.split, test): prediction from a
## rank-deficient fit may be misleading
```

```
summary (pred.tensile.linmodel.split)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   405.9   405.9   457.6   480.9   526.9   617.5
```

```
#Testing linear model of tensile properties on test data.
tens.UTS <- na.omit(test$UTS..MPa.)
#Omit na values from the test values
test.MSE.tensile.linmodel.split <-
sqrt(mean((pred.tensile.linmodel.split- tens.UTS) ^ 2))
```

```
## Warning in pred.tensile.linmodel.split - tens.UTS: longer object length is
## not a multiple of shorter object length
```

```
#Compute the test MSE
test.MSE.tensile.linmodel.split
```

```
## [1] 81.44605
```

```
tss.MSE.tensile.linmodel.split <-
mean((pred.tensile.linmodel.split-mean(pred.tensile.linmodel.split))^2)
rss.MSE.tensile.linmodel.split <-
1- test.MSE.tensile.linmodel.split/ tss.MSE.tensile.linmodel.split
rss.MSE.tensile.linmodel.split
```

```
## [1] 0.987271
```

This models the tensile UTS using all of the variables available. The results of this model do mirror the individual linear models, where aluminium grade 6061 is the only variable that has a negative relationship to UTS. The adjusted R^2 value associated with this model is about 87% which is good and means that the model does fit well to the training data. It is important to point out though, R is giving a warning that predictions from a rank-deficient fit could be misleading.

```
library(caret)
set.seed(10)
tensile.linmodel.loocv<- train (UTS..MPa. ~ Heat.Treatment + Al.Grade + SiC.Volume + SiCp.Size..um., me
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```
summary(tensile.linmodel.loocv)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -87.500 -13.908   2.489  21.500  59.259
##
## Coefficients: (1 not defined because of singularities)
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      363.369217  24.473386  14.848 8.42e-15 ***
## Heat.TreatmentT4 166.000000  17.001846   9.764 1.63e-10 ***
## Heat.TreatmentT6  54.602837  13.005094   4.199 0.000246 ***
## Al.Grade          -0.005724   0.003894  -1.470 0.152759
## SiC.Volume         4.011525   0.650255   6.169 1.16e-06 ***
## SiCp.Size..um.           NA         NA      NA       NA
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 29.45 on 28 degrees of freedom
## Multiple R-squared:  0.8718, Adjusted R-squared:  0.8534
## F-statistic: 47.59 on 4 and 28 DF,  p-value: 4.294e-12
```

```
#This fits a linear regression model using leave one out
#cross validation as the test/train splitting method.
tens.UTS.loocv <- na.omit(tension.flats2$UTS..MPa.)
#Omit na values from the test values
pred.tensile.linmodel.loocv <- predict(tensile.linmodel.loocv, tension.flats2)
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```
test.MSE.tensile.linmodel.loocv <-
sqrt(mean((pred.tensile.linmodel.loocv- tens.UTS.loocv) ^ 2))
#Compute the test MSE
test.MSE.tensile.linmodel.loocv
```

```
## [1] 27.12556
```

```
tss.MSE.tensile.linmodel.loocv <-
mean((pred.tensile.linmodel.loocv-mean(pred.tensile.linmodel.loocv))^2)
rss.MSE.tensile.linmodel.loocv <-
1- test.MSE.tensile.linmodel.loocv/ tss.MSE.tensile.linmodel.loocv
rss.MSE.tensile.linmodel.loocv
```

```
## [1] 0.9945771
```

This is the same model as above, but with leave one out cross validation used as the splitting method. Many of the same conclusions can be drawn, but the adjusted R^2 value is a bit lower.

5.2.2 Bending Fatigue Testing

```
set.seed(10)
bending.linmodel.split <- lm (cycles ~ Heat.Treatment + Al.Grade + SiC.Volume, data = train)
bending.linmodel.split
```

```
##
## Call:
## lm(formula = cycles ~ Heat.Treatment + Al.Grade + SiC.Volume,
##     data = train)
##
## Coefficients:
##      (Intercept)  Heat.TreatmentT4  Heat.TreatmentT6      Al.Grade6061
##          6077400          -5140087          -6000872            -20000
##     SiC.Volume25
##               NA
```

```
summary(bending.linmodel.split)
```

```
##
## Call:
## lm(formula = cycles ~ Heat.Treatment + Al.Grade + SiC.Volume,
##     data = train)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -5958400  -928489  -504313  3922600  7162687
##
## Coefficients: (1 not defined because of singularities)
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       6077400    1326727   4.581 0.000132 ***
## Heat.TreatmentT4 -5140087    1876276  -2.740 0.011680 *
## Heat.TreatmentT6 -6000872    3510190  -1.710 0.100808
## Al.Grade6061       -20000    2297959  -0.009 0.993131
## SiC.Volume25           NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4195000 on 23 degrees of freedom
##   (43 observations deleted due to missingness)
## Multiple R-squared:  0.3162, Adjusted R-squared:  0.227
## F-statistic: 3.544 on 3 and 23 DF,  p-value: 0.03036
```

```
tidy(bending.linmodel.split)
```

```
## # A tibble: 4 x 5
##   term              estimate std.error statistic  p.value
##   <chr>                <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)       6077400   1326727.    4.58    0.000132
## 2 Heat.TreatmentT4 -5140087.  1876276.  -2.74    0.0117
## 3 Heat.TreatmentT6 -6000872.  3510190.  -1.71    0.101
## 4 Al.Grade6061       -20000.  2297959.  -0.00870 0.993
```

```
#Linear model of tensile properties
```

This is a multivariate model that looks at the bending fatigue data. This model looks to model cycles to failure in bending by the variations within the MMCs. This model finds a negative relationship with all variables. However, the adjusted R^2 value is only 22.7%, meaning the model does not fit the data very well.

```
library(caret)
set.seed(10)
bending.linmodel.loocv<- train (cycles ~ Heat.Treatment + Al.Grade + SiC.Volume, method = "lm", data = l
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading

## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```r
summary(bending.linmodel.loocv)
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5481592 -4630399  -458467  4379408  7208533
##
## Coefficients: (1 not defined because of singularities)
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       5.493e+06  2.084e+06   2.636   0.0128 *
## Heat.TreatmentT4 -4.646e+06  1.807e+06  -2.572   0.0150 *
## Heat.TreatmentT6 -5.490e+06  2.760e+06  -1.989   0.0553 .
## Al.Grade          2.102e+01  4.589e+02   0.046   0.9637
## SiC.Volume               NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 4237000 on 32 degrees of freedom
## Multiple R-squared:  0.2608, Adjusted R-squared:  0.1915
## F-statistic: 3.763 on 3 and 32 DF,  p-value: 0.02021
```

```
#This fits a linear regression model using leave one out
#cross validation as the test/train splitting method.
bend.cycles.loocv <- na.omit(bending.fatigue.flats2$cycles)
#Omit na values from the test values
pred.bending.linmodel.loocv <- predict(bending.linmodel.loocv, bending.fatigue.flats2)
```

```
## Warning in predict.lm(modelFit, newdata): prediction from a rank-deficient
## fit may be misleading
```

```
test.MSE.bending.linmodel.loocv <-
sqrt(mean((pred.bending.linmodel.loocv- bend.cycles.loocv) ^ 2))
#Compute the test MSE
test.MSE.bending.linmodel.loocv
```

```
## [1] 3995120
```

```
tss.MSE.bending.linmodel.loocv <-
mean((pred.bending.linmodel.loocv-mean(pred.bending.linmodel.loocv))^2)
rss.MSE.bending.linmodel.loocv <-
1- test.MSE.bending.linmodel.loocv/ tss.MSE.bending.linmodel.loocv
rss.MSE.bending.linmodel.loocv
```

```
## [1] 0.9999993
```

This model shows that all variables have a negative relationship with bending cycles to failure other than aluminum grade 6061. The adjusted R^2 value is less than 20% which is somewhat low.

5.2.2.1 Bootstrapping

```
set.seed(1)
tensile.linmodel.boot <- function(data, index) + return(coef(lm(UTS..MPa. ~ Heat.Treatment + Al.Grade +
tensile.linmodel.boot(train)
```

```
##      (Intercept) Heat.TreatmentT4 Heat.TreatmentT6     Al.Grade6061
##        456.00000        161.50000         67.71429        -50.07143
##     SiC.Volume25     SiC.Volume40
##               NA         53.28571
```

Bootstrapping was also tried as a resampling method. It is showing the same trends as above models where aluminium grade 6061 does have a negative affect on UTS.

```
set.seed(1)
bending.linmodel.boot <- function(data, index) + return(coef(lm(cycles ~ Heat.Treatment + Al.Grade + Si
bending.linmodel.boot(train)
```

```
##      (Intercept) Heat.TreatmentT4 Heat.TreatmentT6     Al.Grade6061
##          6077400         -5140087         -6000872           -20000
##     SiC.Volume25
##               NA
```

Bootstrapping also shows a negative relationship with all variables, which is different than other models run.

5.2.2.2 Variable Selection

5.2.2.2.1 Best Subset Regression-Tensile

```
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 3.5.3
```

```
tensile.regfit.full <- regsubsets(UTS..MPa. ~ Heat.Treatment + Al.Grade + SiC.Volume + SiCp.Size..um., d
```

```
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 2 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
summary(tensile.regfit.full)
```

```
## Subset selection object
## Call: regsubsets.formula(UTS..MPa. ~ Heat.Treatment + Al.Grade + SiC.Volume +
##     SiCp.Size..um., data = mechanical.data)
## 6 Variables  (and intercept)
##                  Forced in Forced out
## Heat.TreatmentT4     FALSE      FALSE
## Heat.TreatmentT6     FALSE      FALSE
## Al.Grade6061         FALSE      FALSE
## SiC.Volume40         FALSE      FALSE
## SiC.Volume25         FALSE      FALSE
## SiCp.Size..um.       FALSE      FALSE
## 1 subsets of each size up to 4
## Selection Algorithm: exhaustive
##          Heat.TreatmentT4 Heat.TreatmentT6 Al.Grade6061 SiC.Volume25
## 1  ( 1 ) "*"              " "              " "           " "
## 2  ( 1 ) "*"              " "              " "           " "
## 3  ( 1 ) "*"              "*"              " "           " "
## 4  ( 1 ) "*"              "*"              " "           "*"
##          SiC.Volume40 SiCp.Size..um.
## 1  ( 1 ) " "          " "
## 2  ( 1 ) "*"          " "
## 3  ( 1 ) " "          "*"
## 4  ( 1 ) "*"          " "
```

5.2.2.2.2 Forward Stepwise-Tensile

```
tensile.regfit.fwd <- regsubsets(UTS..MPa. ~ Heat.Treatment + Al.Grade + SiC.Volume + SiCp.Size..um., da
```

```
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 2 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to
## replace is not a multiple of replacement length
```

```
summary (tensile.regfit.fwd)
```

```
## Subset selection object
## Call: regsubsets.formula(UTS..MPa. ~ Heat.Treatment + Al.Grade + SiC.Volume +
##     SiCp.Size..um., data = mechanical.data, nvmax = 19, method = "forward")
## 6 Variables  (and intercept)
##                  Forced in Forced out
## Heat.TreatmentT4     FALSE      FALSE
```

```
## Heat.TreatmentT6     FALSE      FALSE
## Al.Grade6061         FALSE      FALSE
## SiC.Volume40         FALSE      FALSE
## SiC.Volume25         FALSE      FALSE
## SiCp.Size..um.       FALSE      FALSE
## 1 subsets of each size up to 4
## Selection Algorithm: forward
##          Heat.TreatmentT4 Heat.TreatmentT6 Al.Grade6061 SiC.Volume25
## 1  ( 1 ) "*"              " "              " "           " "
## 2  ( 1 ) "*"              " "              " "           " "
## 3  ( 1 ) "*"              "*"              " "           " "
## 4  ( 1 ) "*"              "*"              "*"           " "
##          SiC.Volume40 SiCp.Size..um.
## 1  ( 1 ) " "          " "
## 2  ( 1 ) "*"          " "
## 3  ( 1 ) "*"          " "
## 4  ( 1 ) "*"          " "
```

5.2.2.2.3 Best Subset Regression-Bending

```
library(leaps)
bending.regfit.full <- regsubsets(UTS..MPa. ~ Heat.Treatment + Al.Grade + SiC.Volume, data = mechanical
```

```
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 1 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
summary(bending.regfit.full)
```

```
## Subset selection object
## Call: regsubsets.formula(UTS..MPa. ~ Heat.Treatment + Al.Grade + SiC.Volume,
##     data = mechanical.data)
## 5 Variables  (and intercept)
##                  Forced in Forced out
## Heat.TreatmentT4     FALSE      FALSE
## Heat.TreatmentT6     FALSE      FALSE
## Al.Grade6061         FALSE      FALSE
## SiC.Volume40         FALSE      FALSE
## SiC.Volume25         FALSE      FALSE
## 1 subsets of each size up to 4
## Selection Algorithm: exhaustive
##          Heat.TreatmentT4 Heat.TreatmentT6 Al.Grade6061 SiC.Volume25
## 1  ( 1 ) "*"              " "              " "           " "
## 2  ( 1 ) "*"              " "              " "           " "
## 3  ( 1 ) "*"              "*"              " "           " "
## 4  ( 1 ) "*"              "*"              "*"           " "
##          SiC.Volume40
## 1  ( 1 ) " "
## 2  ( 1 ) "*"
## 3  ( 1 ) "*"
## 4  ( 1 ) "*"
```

5.2.2.2.4 Forward Stepwise-Tensile

```
tensile.regfit.fwd <- regsubsets(UTS..MPa. ~ Heat.Treatment + Al.Grade + SiC.Volume + SiCp.Size..um., da
```

```
## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax,
## force.in = force.in, : 2 linear dependencies found
```

```
## Reordering variables and trying again:
```

```
## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to
## replace is not a multiple of replacement length
```

```
summary (tensile.regfit.fwd)
```

```
## Subset selection object
## Call: regsubsets.formula(UTS..MPa. ~ Heat.Treatment + Al.Grade + SiC.Volume +
##     SiCp.Size..um., data = mechanical.data, nvmax = 19, method = "forward")
## 6 Variables  (and intercept)
##                   Forced in Forced out
## Heat.TreatmentT4      FALSE      FALSE
## Heat.TreatmentT6      FALSE      FALSE
## Al.Grade6061          FALSE      FALSE
## SiC.Volume40          FALSE      FALSE
## SiC.Volume25          FALSE      FALSE
## SiCp.Size..um.        FALSE      FALSE
## 1 subsets of each size up to 4
## Selection Algorithm: forward
##          Heat.TreatmentT4 Heat.TreatmentT6 Al.Grade6061 SiC.Volume25
## 1  ( 1 ) "*"              " "              " "           " "
## 2  ( 1 ) "*"              " "              " "           " "
## 3  ( 1 ) "*"              "*"              " "           " "
## 4  ( 1 ) "*"              "*"              "*"           " "
##          SiC.Volume40 SiCp.Size..um.
## 1  ( 1 ) " "          " "
## 2  ( 1 ) "*"          " "
## 3  ( 1 ) "*"          " "
## 4  ( 1 ) "*"          " "
```

5.3 Multivariate Modeling of all testing types together

The original goal of this project was to create a multivariate model of all testing types together. However, in creating this model issues arose with there only being one variation of push pull fatigue testing. Due to these issues a model could only be created using tensile testing and bending fatigue testing together.

```
dep <-list("UTS..MPa.~", "cycles~")
indep1 <- list("Heat.Treatment", "Al.Grade")
indep2 <- list(train$SiC.Volume)
all <- Map(function(x,y,z) lm(as.formula(paste(x,paste(list(y,z),collapse="+"))),data=train),dep,indep1
all
```

```
## [[1]]
##
## Call:
## lm(formula = as.formula(paste(x, paste(list(y, z), collapse = "+"))),
##     data = train)
##
## Coefficients:
##
```

```
##
##
##
##
##
## c("40", "25", "25", "25", "25", "25", "25", "25", "25", "25", "25", "25", "25", "20", "25", "25", "25", "2
##
## c("40", "25", "25", "25", "25", "25", "25", "25", "25", "25", "25", "25", "25", "20", "25", "25", "25", "2
##
##
##
## [[2]]
##
## Call:
## lm(formula = as.formula(paste(x, paste(list(y, z), collapse = "+"))),
##     data = train)
##
## Coefficients:
##
##
##
##
## c("40", "25", "25", "25", "25", "25", "25", "25", "25", "25", "25", "25", "25", "20", "25", "25", "25", "2
##
```

This model shows that each of the variables in this model have a positive relationship with both cycles to failure and UTS.

6 Discussion

From the simple linear models built a few trends were found. In tensile testing it was found that all variables had a positive relationship with tensile UTS other than aluminum matrix grade 6061. The adjusted R^2 values associated with these models are low, so the models are not fitting the data very well. In bending fatigue testing all variables had positive relationship with bending cycles to failure other than 6061 and 40% SiC. Again, the adjusted R^2 values are very low.Trends follow throughout both splitting methods, and the adjusted R^2 values simmilar. These findings do not follow expected trends from EDA. EDA found that 40% SiC had the highest UTS values, and aluminum grade 6061 did not have any apparent negative affects on UTS.

Trends could also be found from the multivariate models built. Tensile models found trends that agree with simple linear models (Al Grade 6061 is the only variable with negative relationship to UTS). Adjusted R^2 values are much higher than in simple linear models, and the 75%-25% split has 2% higher adjusted R^2 than LOOCV.Bending models showed trends that do not agree with simple linear models. Trends from multivariate models show all variables have a negative relationship with cycles to failure. Adjusted R^2 values are still very low in multivariate bending modeling, and again the 75%-25% split has a 4% higher adjusted R^2 than LOOCV.

Bootstrapping the tensile data resulted in the same trends being seen in both simple linear models and multivariate models. Bootstrapping the bending fatigue data showed negative relationships with all variables, agreeing with the multivariate models built.

Variable selection was preformed on both sets of data and it was found that using all variables in the model was best.

Finally, a model was built integrating both bending fatigue and tensile testing into one model. This model showed that all variables had a positive relationship with UTS and cycles to failure.

7 Conclusion

In conclusion the accuracy of all models built is relatively low, and trends do not follow throughout all steps of the data analysis process. Tensile data seemed to show two different things in EDA and modeling. Also, bending fatigue data did not draw the same trends in simple linear modeling and multivariate modeling. This makes it hard to draw any solid conclusions, but could simply be an artifact of not having enough data to make reliable models from.

This was an expected issue from the start of this project and even if the accuracy was high there is not enough data to be statistically significant. This is why one of the main original goals was to make recommendations for future testing/ways to improve testing method if redone. This project did allow for some insights to be drawn in this regard.

First, the most obvious recommendation would be to obtain more data. Though this could be achieved through more testing, another way in which this could be achieved would be to use prior data collected by others. An example of this is obtaining the data from Ashby plots. To build an Ashby plot a lot of data has to be collected and used. Though it would be hard to extract the data directly from the plot, if access to the data on the backend of these plots was accessible in some way this could be of great use, especially for this project. Though the MMC studied here were processed in a novel way there are Ashby plots for the exact variations of MMC looked at in terms of heat treatment, aluminum grade, and silicon carbide percent and volume. Having both sets of data could help to draw better understanding of exactly how the process parameters are affecting mechanical properties, and could make direct comparisons between the methods of processing easier as well.

One other idea to obtain more data would be to run a simple test on all materials in addition to the original test run. An example of this is hardness testing. Hardness tests could be run on all samples after they had been tested in either tensile, bending fatigue, and push pull fatigue. This would give the same quantitative information on all variations of MMC tested and is relatively experimentally easy and non-expensive to do. Hardness values have been related to mechanical properties before, and could be of use when thought of in this way as well.

8 Acknowledgements

The author would like to acknowldege Ji Xia, Conrad Park, and Erica Bindas as the data used in this analysis was collected by these individuals.

Additionally, acknowledgement should be given to Professor Roger French and Menghong Wang for direction and help to achieve the data analysis presented in this report.

9 References, Citations

10 License: CCBY-BY-SA 4.0