Supplementary Material 1: Data Analysis in R

Code

This R notebook, which is the first part of the supplementary information to be accompanied with the IEEE journal paper submitted as part of the IoT PT Master's. This notebook and the relevant code is available at this GitHub link (https://github.com/BecsLutton/masters-project).

This Notebook mostly contains the various ways the data was explored. It was known that Tool Wear was a focus but that it was not being collected in any real terms, just the Tool Life Remaining in minutes, which is fixed at the beginning of production and is ignorant of external factors other than active tool time. As such the data needed to be pulled apart and investigated in different ways. There are many machine tools used in the manufacture of the specific part of focus and there are certain tools that, logically would make good candiates for examination. This is where the analysis started: determining the best machine tool for examination.

Setup

Hide

```
#-------------------------
# LIBRARIES
#-------------------------

library(tidyverse)
library(dplyr)
library(DT)
library(knitr)
library(stringr)
library(ggplot2)
library(plotly)
library(reshape2)
library(gridExtra)
library(cowplot)
library(ggpubr)
library(chemometrics)
library(cluster)
library(factoextra)
library(FeatureImpCluster)
library(flexclust)
library(qpcR)
#library(egg)
```

Functions

This function calculates the mode of values. In R, according to the help documentation, 'mode' returns or 'sets the 'mode' (a kind of 'type'), or the storage mode of an R object'. In this project, the requirement of 'Mode' is in the statistical sense i.e. to return the most common value.

Hide

```
Mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}
```

This function replaces NaNs. Unlike 'is.na', R does not yet have a method for replacing NaNs in data frames.

Hide

```
is.nan.data.frame <- function(x) {do.call(cbind, lapply(x, is.nan))}
```

Loading the csv dataset

Hide

```
raw24 <- read.csv("~/onedrive/Scenic/data/indexG220/220224.csv", comment.char="#", stringsAsFactors=FALSE)
```

Data Preparation

Hide

```
#----DATASET----
raw<-raw24
#---------------

raw_cat<-raw[c(4:nrow(raw)),c(6, 7, 8)] #removing csv header info and dropping unnecessary columns
colnames(raw_cat)<-c("time", "value", "metric") #renaming columns
row.names(raw_cat)<-1:nrow(raw_cat) #changing indexes from 4:end back to 1:end-3 due to loss of header & info row
s
kable(head(raw_cat)) #Checking outcome is as desired
```

time	value	metric
2022-02-24T09:07:07.207652739Z	0	fields_Channels_0_Desired Feed Rate
2022-02-24T09:07:07.899710642Z	0	fields_Channels_0_Desired Feed Rate
2022-02-24T09:07:08.648171734Z	0	fields_Channels_0_Desired Feed Rate
2022-02-24T09:07:09.851296558Z	0	fields_Channels_0_Desired Feed Rate
2022-02-24T09:07:10.604647442Z	0	fields_Channels_0_Desired Feed Rate
2022-02-24T09:07:11.296754372Z	0	fields_Channels_0_Desired Feed Rate

Hide

```
length(unique(raw_cat$metric)) #checking no. of metics, should be 141
```

```
[1] 141
```

Hide

```
df_cat <- with(raw_cat, subset(raw_cat, grepl(glob2rx("fields_*"), raw_cat$metric))) #dropping metrics that don't
begin with "fields_"
length(unique(df_cat$metric)) #checking removal of anomalous metrics
```

```
[1] 140
```

Hide

```
#Data frame is in wrong format. The parameters of interest are not column headers as needed but instead, all pres
ent in one column. The df needs pivoting to transpose these metrics from single column list to column headers.
df<-df_cat %>% pivot_wider(names_from = metric, values_from = value)
#head(df)

date_time<-as.matrix(str_split_fixed(df$time, "T", 2)) #splitting time column into date and time
timecat<-substr(date_time[,2], 1, 8) #tidying the time by rounding up to seconds

df1<-na.omit(as.data.frame(cbind(date_time[,1], timecat, df[-1])))#dropping the columns with _NA_ in rows and rep
lacing time with new date and time columns
names(df1)[1]<-"Date"
names(df1)[2]<-"Time"

head(df1)
```

	Date <chr>	Time <chr>	fields_Channels_0_Desired Feed Rate <chr>
5869	2022-02-24	10:17:14	360000
5870	2022-02-24	10:17:15	360000
5871	2022-02-24	10:17:16	360000
5872	2022-02-24	10:17:17	360000
5873	2022-02-24	10:17:19	216000
5874	2022-02-24	10:17:20	216000

6 rows | 1-4 of 142 columns

Due to this being still in the experimental stage, the processes for data extraction and collection aren't seamless. In some instances, it was notices that the part count hadn't been reset and instead of starting from 0, it started at 74, inevitably affecting later data analysis.

This *if statement* adjusts the data so that if it is evident the data collection was incorrect, it would fix it. The reason it starts at 15 is because in some other instances during exploration and debugging of the data extraction and collection processes, some initial samples were missed. These datasets were not used in the data analysis as they would be incomplete, unlike the datasets affected by the below function where the only issue is an incorrectly labelled part count.

Hide

```
if (as.numeric(as.character(df1$`fields_General_0_Part Count`[1])) > 15){
  x<-as.numeric(as.character(df1$`fields_General_0_Part Count`[1]))
  df1$`fields_General_0_Part Count`<-(as.numeric(as.character(df1$`fields_General_0_Part Count`))-x)
}
```

Getting Tool Names and creating datasets for tools of interest

Hide

```
ch1ToolNames <-as.factor(unique(df1$fields_Tool_0_tool_name))
kable(ch1ToolNames)
```

x
11.5MM_3D_CHAMDRILL_TC
11.5MM_8D_CHAMDRILL_TC
10MM_ECO_MSP
11.8MM_GUNDRILL_TC
F_TURN_S_TC
4MM_ENDMILL_TC
F_TURN_MONO_CSP
10.8MM_5D_CHAMDRILL_TC
10MM_ECO_CSP
INT_U_CUT_TC
PROBE
FULL_RAD_118_TC
ROUGH_SPLINES_TC
SPOT_DRILL
FINISH_SPLINES_TC
PART_OFF_TC

The CNC machine utilised in this project is an Index G220 which is a mill-turn machine and as such has 4 spindles: 2 are chucks (S4 and S5) used to hold the raw/machined bar and the other 2 (S1 and S2) are the machining spindles. For the sake of a starting point, spindle 1 was chosen to focus analysis as this has the greatest variety of activity, requires more maintenance and is the most important of the machining spindles.

Considering the choice of spindle, the machining tools in ch1ToolNames not used by S1 were dropped. Similarly, after reviewing the CNC machine code, those tools that had the most activity and least life remaining at the end of production, were kept. For some of the selected tools, they are so frequently used that they have a replica tool available called a 'sister tool' which is present for when the tool in question reaches end of life during production.

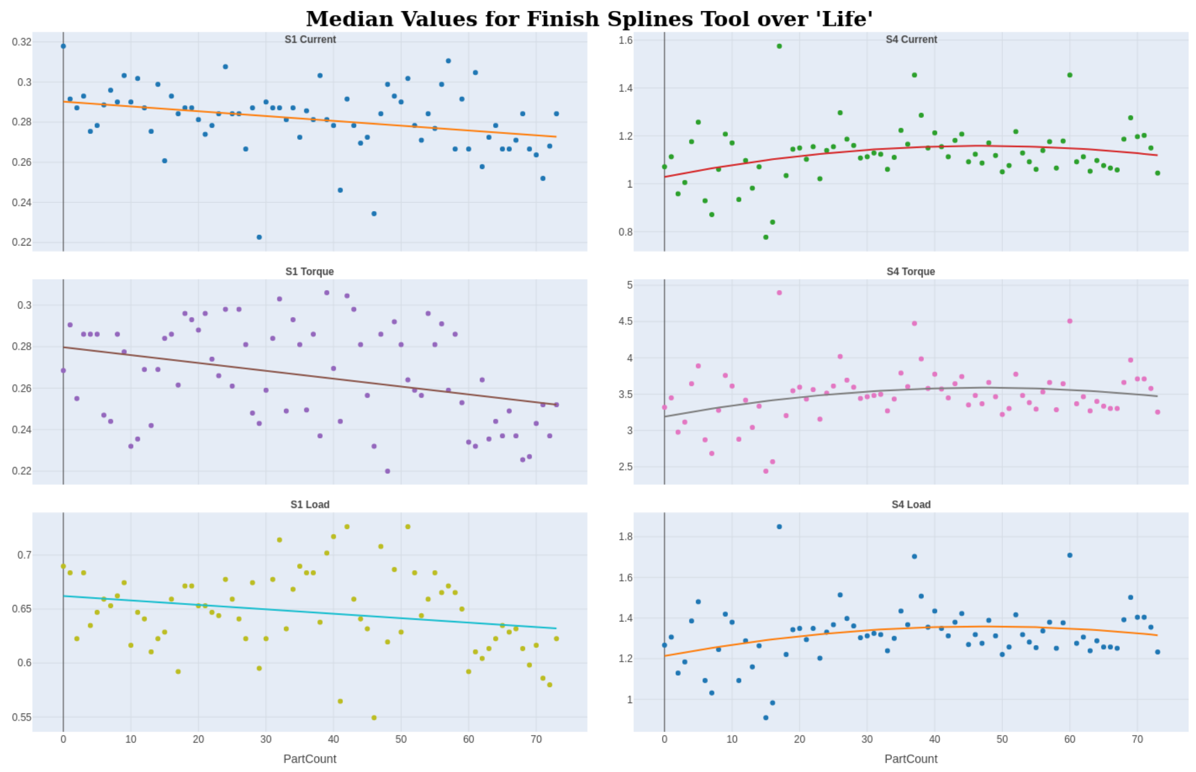
These were the selected tools of interest for further analysis:

Hide

```
# Tools of interest:
fullRad <- filter(df1, fields_Tool_0_tool_name == "FULL_RAD_118_TC")
finishSplines <- filter(df1, fields_Tool_0_tool_name == "FINISH_SPLINES_TC")
chamdrill8D <- filter(df1, fields_Tool_0_tool_name == "11.5MM_8D_CHAMDRILL_TC")
chamdrill3D <- filter(df1, fields_Tool_0_tool_name == "11.5MM_3D_CHAMDRILL_TC")
gundrill <- filter(df1, fields_Tool_0_tool_name == "11.8MM_GUNDRILL_TC")
partoff <- filter(df1, fields_Tool_0_tool_name == "PART_OFF_TC")
```

NOTE

The code presented below is the same code that was used to analyse the other tools list above. The outcomes of these are not included in the notebook or the GitHub repo for the sake of brevity and tidyness. The reason these analyses were dropped and the focus shifted to the Gundrill is because the results revealed that there was little that could be done to explore effect of tool wear. The image below is a plot of the Median Current, Torque and Load values for both spindles S1 and S4 for the tool named Finish Splines. It was expected that as tool wear was incurred, these parameters would increase, trending upwards; the opposite was true.



It is believed that the reason for the downwards trend in these values is due to the fixed nature of the tool coordinates. In this production, all machine tools but the drills commence machining when they reach certain coordinates, at which point they then move in only one direction, across the material surface. In some instances, the tool may reset this axis, move closer by its cut depth and repeat the action. However, for every part these coordinates remain unchanged and as a result, as tool wear occurs, the total contact of the tool and the part reduces over time. This is the cause for the deterioration of precision as the dimensions of later parts near their acceptable tolerance thresholds. As such, these tools are not appropriate for study and the analyses performed on them, abandoned.

S4 was included in all analyses since it is the spindle responsible for holding the material as the tool machines its surface and it was thought that it could expose useful information.

Data Exploration of Gundrill

This section attempts to unravel the part of production that uses this tool. First the data is filtered for the active state (program status == 3) and then for an arbitrary part count so that the returned data is for one part, not all parts; a means of reducing data volume. After conversing with the operator and reviewing the G-code, it is understood that the Gundrill approaches the part, locating the pilot holes, rotating anti-clockwise and at lower speed and feed rates, to reduce potential damage of tool and part. Once the tool has reached its desired coordinates it begins to rotate clockwise at the required cutting RPM and Feed Rate (FR); this is the time period that most wear is likely to occur and, as such, is the focus for analysis.

Once the dataset for a single part, has been filtered for its active state, it is then queried for the values of Desired RPM. Likewise, the data is filtered for tool number and program message. These latter two parameters are a requirement resulting from lag and overlap of processes and updates of code. The values for each of the parameters available do not synchronise at exactly the same time and in some cases incur a significant lag before updating. An example is the program message and tool number. When a tool is being changed the HMI of the machine will present the program message indicating the process that is about to commence and this message will remain on the display until it ends and a new tool is being changed. Sometimes, the message does not change for a significant time period after the tool has been changed and in some cases, it does so before the tool has reached the tool changer.

One aspect of CNC data analysis that has transpired to be of utmost importance from this project is the critical need for domain knowledge. These blurred lines and approximations occur in all machines but manifest in different ways dependent on the machine itself, the NC/PLC controller, the operator, the part being machined and the program writer. All of these aspects were unacknowledged before this project and until this particular stage of the data analysis section. Getting to this point of data analysis was both interesting and frustrating as data behaved differently to what was expected, requiring investigation and in-depth understanding of the machine, the operator behaviour and the machining process in question. The time and effort invested in realising these quirks, can not be overstated.

Filtering the Gundrill data

Hide

```
#------------------------------------------------------------------
#checking what RPM bands the tools operate in and no of occurrences
#------------------------------------------------------------------
gundrill2<-filter(gundrill,
                  `fields_General_0_Part Count` == 2,   #filtering for one part (number 2)
                  `fields_Channels_0_Program Status`==3) #filtering for active program (status = 2)

gundrill2_rpm <- gundrill2[, grepl("RPM", names(gundrill2))] #sort for columns with RPM in column title for partc
ount = 2

unique(gundrill2_rpm$`fields_S1_0_Desired RPM`) #unique RPMS for spindle S1
```

```
[1] "0"   "300"
```

Hide

```
unique(gundrill2_rpm$`fields_S4_0_Desired RPM`) #unique RPMS for spindle S4
```

```
[1] "-52.719811300133834" "-51.90899682381817"  "-600"
```

Hide

```
table(gundrill2_rpm$`fields_S1_0_Desired RPM`) # get count for the unique RPMS for spindle S1
```

```

  0 300
  7 166
```

Hide

```
table(gundrill2_rpm$`fields_S4_0_Desired RPM`) # get count for the unique RPMS for spindle S4
```

```

 -51.90899682381817 -52.719811300133834                -600
                  2                   2                 169
```

Hide

```
table(gundrill$fields_Tool_0_tool_no) # count of tool numbers to check where tool change occurred
```

```

 124  149
6029 6474
```

Hide

```
gundrill$`fields_General_0_Part Count`[6475] # checking part count after tool change
```

```
[1] "38"
```

Hide

```
table(gundrill2$fields_Channels_0_Msg) # count of processes to ensure dataset has been appropriately filtered
```

```

Drilling
     173
```

Hide

```
table(gundrill2$`fields_S1_0_Desired Feed Rate`) # checking what feed rates are available
```

```

     0 108000 432000
     6    166      1
```

Hide

```
table(gundrill2$`fields_General_0_Axes Status`) # checking how many are of value 1 i.e. not active
```

```
  0   1
168   5
```

Hide

```
table(gundrill2$`fields_Channels_0_Line Number`) # retrieving the counts of line numbers to know when the gundril
l is rotating clockwise, i.e. completing main purpose
```

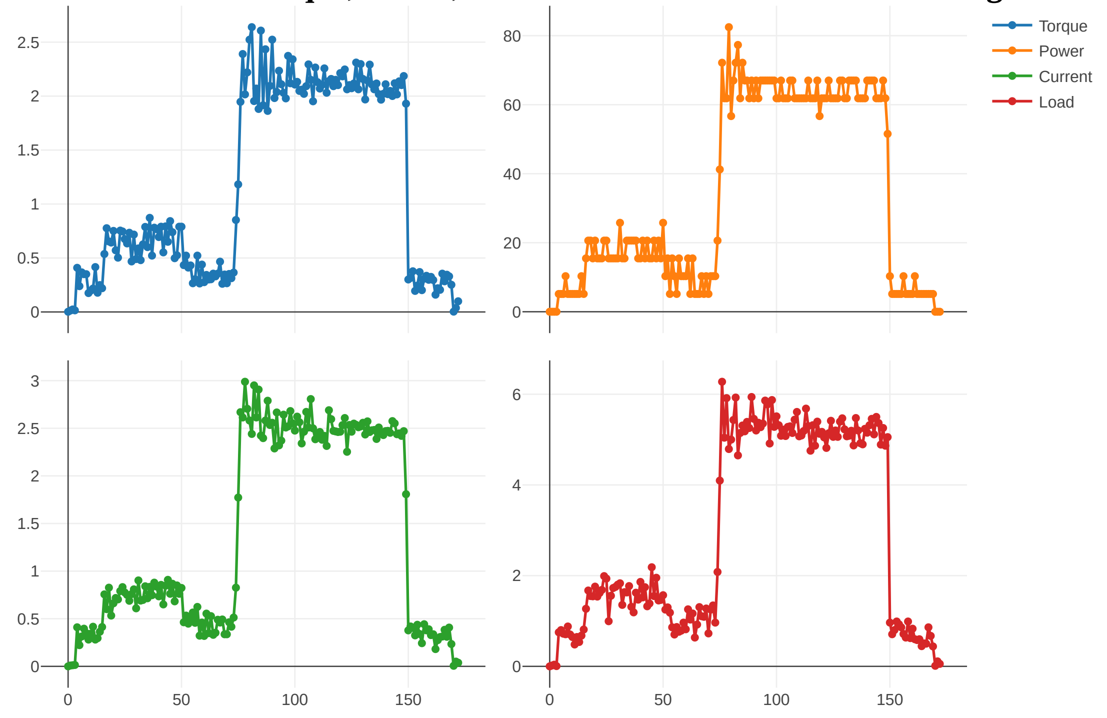
```
185 192 220 221 224  23 230 242 244 246 247 248 249 250 251 253 257  79
  1   1   1   1   1   1   1   1   1   6   1  38  37   1  62  17   1   1
```

Quick check of values for Torque, Power, Current and Load.

Hide

```
grundrill2_T_plot<-plot_ly(gundrill2, y = ~abs(as.numeric(fields_S1_0_Torque)), type = 'scatter',  mode = 'lines+
markers', name="Torque")
grundrill2_P_plot<-plot_ly(gundrill2, y = ~abs(as.numeric(fields_S1_0_Power)), type = 'scatter',  mode = 'lines+m
arkers', name = "Power")
grundrill2_C_plot<-plot_ly(gundrill2, y = ~abs(as.numeric(fields_S1_0_Current)), type = 'scatter',  mode = 'lines
+markers', name = "Current")
grundrill2_L_plot<-plot_ly(gundrill2, y = ~abs(as.numeric(fields_S1_0_Load)), type = 'scatter',  mode = 'lines+ma
rkers', name = "Load")

subplot(grundrill2_T_plot,
        grundrill2_P_plot,
        grundrill2_C_plot,
        grundrill2_L_plot,
               nrows = 2,
               shareX = TRUE) %>%
  layout(title = list(text="<b>Absolute values for Torque, Power, Current and Load of Gundrill during Part=2</b
>",
                      font = t,
                      y = 1, x = 0.5,
                      xanchor = 'center',
                      yanchor =  'top'),
         plot_bgcolor='white')
```

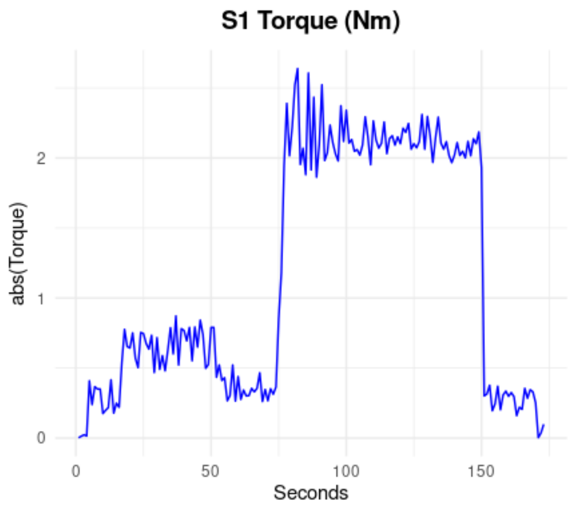
Absolute values for Torque, Power, Current and Load of Gundrill during Part=2

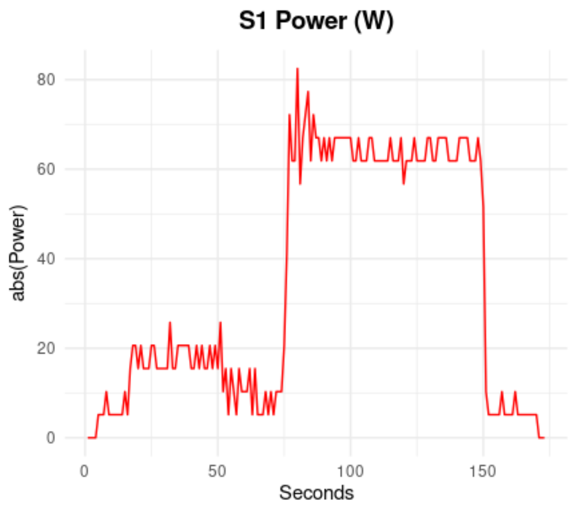


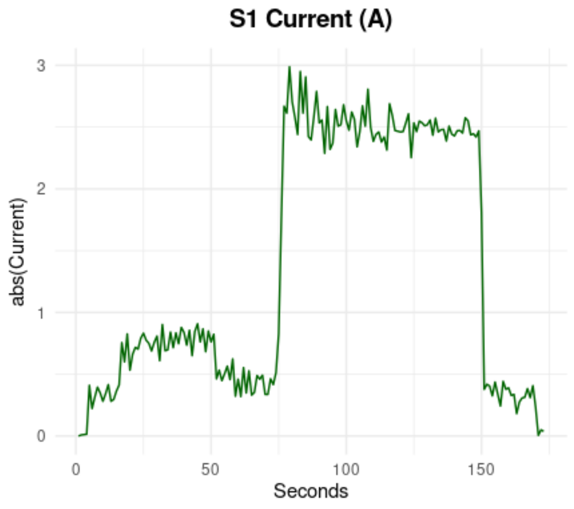
Prettier plot for later repurposing. Note that the Power plot confirms that this is a Constant Power driven machine and as such, Current would likely be most useful moving forward.

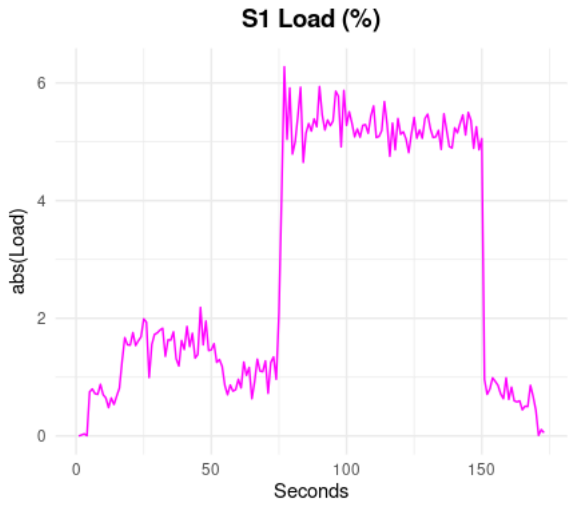
Hide

```r
g2t_plot<-ggplot(gundrill2,
          aes(
            x=seq_along(gundrill2$fields_S1_0_Torque),
            y=abs(as.numeric(gundrill2$fields_S1_0_Torque))
            )
          ) +
  geom_line(color="blue")+
  labs(x = expression("Seconds"),
       y=expression("abs(Torque)"),
       title = expression(bold("S1 Torque (Nm)"))
       )+
  theme_minimal()+
  theme(plot.title=element_text(hjust=0.5, size=14), legend.position = "none")


g2p_plot<-ggplot(gundrill2,
          aes(
            x=seq_along(gundrill2$fields_S1_0_Power),
            y=abs(as.numeric(gundrill2$fields_S1_0_Power))
            )
          ) +
  geom_line(color="red")+
  labs(x = expression("Seconds"),
       y=expression("abs(Power)"),
       title = expression(bold("S1 Power (W)"))
       )+
  theme_minimal()+
  theme(plot.title=element_text(hjust=0.5, size=14), legend.position = "none")

g2c_plot<-ggplot(gundrill2,
          aes(
            x=seq_along(gundrill2$fields_S1_0_Current),
            y=abs(as.numeric(gundrill2$fields_S1_0_Current))
            )
          ) +
  geom_line(color="dark green")+
  labs(x = expression("Seconds"),
       y=expression("abs(Current)"),
       title = expression(bold("S1 Current (A)"))
       )+
  theme_minimal()+
  theme(plot.title=element_text(hjust=0.5, size=14), legend.position = "none")

g2l_plot<-ggplot(gundrill2,
          aes(
            x=seq_along(gundrill2$fields_S1_0_Load),
            y=abs(as.numeric(gundrill2$fields_S1_0_Load))
            )
          ) +
  geom_line(color="magenta")+
  labs(x = expression("Seconds"),
       y=expression("abs(Load)"),
       title = expression(bold("S1 Load (%)"))
       )+
  theme_minimal()+
  theme(plot.title=element_text(hjust=0.5, size=14), legend.position = "none")


ggarrange(g2t_plot, g2p_plot, g2c_plot, g2l_plot)
```









Hide

```
#Using the information gained from the exercises above to filter the data and create a data frame.

Gundrill<-data.frame(group_by(gundrill, as.factor(gundrill$`fields_General_0_Part Count`)))

gundrill300<-filter(
    Gundrill,
    fields_S4_0_Desired.RPM== -600,
    fields_Channels_0_Msg == "Drilling",
    fields_S1_0_Desired.Feed.Rate =="108000",
    fields_Channels_0_Program.Status==3,
    fields_General_0_Axes.Status==0,
    fields_Channels_0_Line.Number == 251,
    as.numeric(fields_General_0_Part.Count)<38)

#head(gundrill300)

length(table(gundrill300$`fields_General_0_Part.Count`)) # checking that all parts have experienced this RPM (38)
```

```
[1] 38
```

Extraction of parameters of interest

For the same reason that the S4 spindle was captured alongside the main milling spindle S1, the Drive variables were also deemed potentially revealing and so it was of interest to capture them too.

From research it was noted that a useful parameter in tool parameter optimisation is something called Chip Load (CL) and is the size of the chip removed during machining. This isn’t something that is captured by this particular machine but can be easily calculated by dividing RPM by FR. It was decided that it might prove useful to calculate this parameter from the Desired RPM and FR as well as the actual and determine if there any differences as time proceeds and tool wear increases.

Hide

```r
#-----------------------------
#  DRIVE VARIABLES
#-----------------------------
g_300<-setNames(data.frame(gundrill300$Date), "Date")
g_300$Time<-gundrill300$Time
g_300$PartCount<-as.numeric(gundrill300$fields_General_0_Part.Count)
g_300$DrivePower <- as.numeric(gundrill300$fields_Channels_0_DrivePower)
g_300$DriveCurrent <- as.numeric(gundrill300$fields_Channels_0_DriveCurrent)
g_300$DriveTorque <- as.numeric(gundrill300$fields_Channels_0_DriveTorque)
g_300$DriveLoad <- as.numeric(gundrill300$fields_Channels_0_DriveLoad)
g_300$DriveTemp<-as.numeric(gundrill300$fields_Channels_0_Drive.Temperature)
g_300$Name<-as.factor(gundrill300$fields_Tool_0_tool_name)
g_300$ToolNo<-as.factor(gundrill300$fields_Tool_0_tool_no)
g_300$LineNo <-as.numeric(gundrill300$fields_Channels_0_Line.Number)
g_300$Type<-as.factor("Mill")
g_300$Life<-as.numeric(gundrill300$fields_Tool_0_tool_life_min)
g_300$X<-as.numeric(gundrill300$fields_Channels_0_X_Coord)
g_300$Y<-as.numeric(gundrill300$fields_Channels_0_Y_Coord)
g_300$Z<-as.numeric(gundrill300$fields_Channels_0_Z_Coord)

#-----------------------------
#  S1 VARIABLES
#-----------------------------
g_300_s1<-setNames(data.frame(as.factor(matrix("S1", nrow = nrow(g_300)))), "Spindle")
g_300_s1$DRPM<-as.numeric(gundrill300$fields_S1_0_Desired.RPM)
g_300_s1$RPM<-as.numeric(gundrill300$fields_S1_0_RPM)
g_300_s1$Load<-as.numeric(gundrill300$fields_S1_0_Load)
g_300_s1$Power<-as.numeric(gundrill300$fields_S1_0_Power)
g_300_s1$Torque<-as.numeric(gundrill300$fields_S1_0_Torque)
g_300_s1$Current<-as.numeric(gundrill300$fields_S1_0_Current)
g_300_s1$FR<-as.numeric(gundrill300$fields_S1_0_Feed.Rate)
g_300_s1$DFR<-as.numeric(gundrill300$fields_S1_0_Desired.Feed.Rate)
g_300_s1$DCL<-as.numeric(gundrill300$fields_S1_0_Desired.RPM)/as.numeric(gundrill300$fields_S1_0_Desired.Feed.Rat
e) #Desired CL
g_300_s1$CL<-as.numeric(gundrill300$fields_S1_0_RPM)/as.numeric(gundrill300$fields_S1_0_Feed.Rate) #Actual CL

g_300_s1$DCL[is.nan(g_300_s1$DCL)] <-0  # Dropping NaNs incurred from the CL calculation
g_300_s1$CL[is.nan(g_300_s1$CL)] <-0 # Dropping NaNs incurred from the CL calculation

#-----------------------------
#  S4 VARIABLES
#-----------------------------
g_300_s4<-setNames(data.frame(as.factor(matrix("S4", nrow = nrow(g_300)))), "Spindle")
g_300_s4$DRPM<-as.numeric(gundrill300$fields_S4_0_Desired.RPM)
g_300_s4$RPM<-as.numeric(gundrill300$fields_S4_0_RPM)
g_300_s4$Load<-as.numeric(gundrill300$fields_S4_0_Load)
g_300_s4$Power<-as.numeric(gundrill300$fields_S4_0_Power)
g_300_s4$Torque<-as.numeric(gundrill300$fields_S4_0_Torque)
g_300_s4$Current<-as.numeric(gundrill300$fields_S4_0_Current)
g_300_s4$DFR<-as.numeric(gundrill300$fields_S4_0_Desired.Feed.Rate)
g_300_s4$FR<-as.numeric(gundrill300$fields_S4_0_Feed.Rate)
g_300_s4$DCL<-as.numeric(gundrill300$fields_S4_0_Desired.RPM)/as.numeric(gundrill300$fields_S4_0_Desired.Feed.Rat
e) #Desired CL
g_300_s4$CL<-as.numeric(gundrill300$fields_S4_0_RPM)/as.numeric(gundrill300$fields_S4_0_Feed.Rate) #Actual CL

g_300_s4$DCL[is.nan(g_300_s4$DCL)] <-0 # Dropping NaNs incurred from the CL calculation
g_300_s4$CL[is.nan(g_300_s4$CL)] <-0 # Dropping NaNs incurred from the CL calculation

g_300_all <- cbind(g_300, g_300_s1, g_300_s4)
#write.csv(g_300_all,"~/onedrive/Scenic/data/indexG220/ml/gundrill_3.csv", row.names = FALSE)

g_300_Drive<-g_300
g_300_S1<-cbind(g_300, g_300_s1)
g_300_S4<-cbind(g_300, g_300_s4)

# This dropping of columns was done iteratively as analysis was completed and redundant information was revealed.
# The data was grouped by Part Count to visualise overall trends over the life of the tool
drop<-c("Date", "Time", "Name", "ToolNo", "Type", "Spindle", "X", "Y")
s1_sum_mean<-g_300_S1[,!(names(g_300_S1) %in% drop)] %>%
  group_by(PartCount) %>%
  summarise(across(everything(), list(mean)))

kable(head(s1_sum_mean))
```

PartCount	DrivePower_1	DriveCurrent_1	DriveTorque_1	DriveLoad_1	DriveTemp_1	LineNo_1	Life_1	Z_1	DRPM_1	RPM_1	Load_1
0	336.9654	-1.761485	-5.272017	2.043661	28.21430	251	87.34180	-101.2267	300	300.0019	5.176188

PartCount	DrivePower_1	DriveCurrent_1	DriveTorque_1	DriveLoad_1	DriveTemp_1	LineNo_1	Life_1	Z_1	DRPM_1	RPM_1	Load_1
1	333.3968	-1.739585	-5.242590	1.943219	31.22598	251	85.00106	-101.1701	300	299.9928	5.244421
2	334.8904	-1.703647	-5.073129	2.045367	31.64973	251	82.65758	-101.1719	300	299.9988	5.245381
3	322.9480	-1.656795	-5.170905	1.916698	32.13634	251	80.31007	-101.2585	300	299.9375	5.263265
4	310.4115	-1.611363	-4.800500	1.941833	32.02482	251	77.97095	-101.1410	300	300.0207	5.309448
5	323.4846	-1.696997	-5.367629	2.047926	32.43523	251	75.63398	-101.0302	300	299.9863	5.283479

Hide

```
#write.csv(s1_sum_mean,"~/onedrive/Scenic/data/indexG220/ml/s1_sum_mean_1.csv", row.names = FALSE)
```

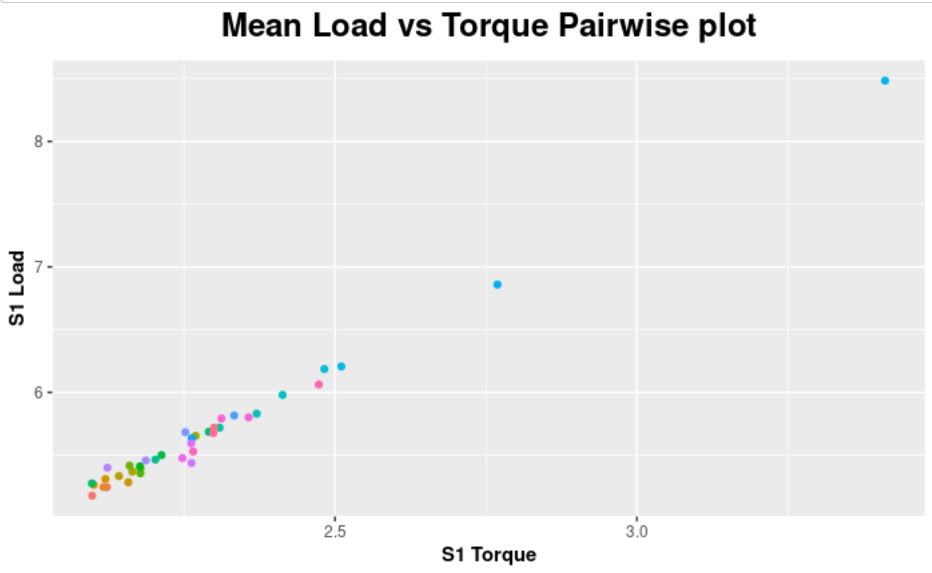
Analysis

Power, Load, Torque, Current and Temperature were thought to be the most useful parameters to explore. The data was filtered for these parameters and separate datasets created. The statistical measure used to summarise the data was Median. This was chosen over mean as it was understood that the data was likely to have a large number of outliers; this evaluation was from time spent observing the Grafana plots during the framework setup process.

Despite preference for Median at this time, the Mean was still included. Standard Deviation (SD) was also included, as were the trimmed SD and the Median Absolute Deviation (MAD), which was used to check data volatility and variability as tool wear increased; this metric was chosen over Mean Absolute Deviation and other comparators due to the decreased sensitivity to outliers. It is believed that as tool wear progresses the frequency of outliers will increase due to decreasing uniformity. Choosing this type of robust measure means that slight increases in anomalous behaviour will be ignored but a sufficient volume of outliers that affects the trend of the data, will be detected. Originally trimmed mean was included but was then removed in favour of ‘normal’ mean

Hide

```
#-----------------------------
# SUMMARISING DATA
#-----------------------------

#----------- DRIVE

g_DPower<-g_300%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean(DrivePower),
    Median  = median(DrivePower),
    Mode = Mode(DrivePower),
    Q1 = quantile(DrivePower, 0.25),
    Q3 = quantile(DrivePower, 0.75),
    IQR = Q3-Q1,
    SD = sd(DrivePower),
    SD_trim = sd_trim(DrivePower, trim=0.1),
    mad=mad(DrivePower, constant = 1.4826),
    size = length(DrivePower))

g_DCurrent<-g_300%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean(DriveCurrent),
    Median  = median(DriveCurrent),
    Mode = Mode(DriveCurrent),
    Q1 = quantile(DriveCurrent, 0.25),
    Q3 = quantile(DriveCurrent, 0.75),
    IQR = Q3-Q1,
    SD = sd(DriveCurrent),
    SD_trim = sd_trim(DriveCurrent, trim=0.1),
    mad=mad(DriveCurrent, constant = 1.4826),
    size = length(DriveCurrent))

g_DLoad<-g_300%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean(DriveLoad),
    Median  = median(DriveLoad),
    Mode = Mode(DriveLoad),
    Q1 = quantile(DriveLoad, 0.25),
    Q3 = quantile(DriveLoad, 0.75),
    IQR = Q3-Q1,
    SD = sd(DriveLoad),
    SD_trim = sd_trim(DriveLoad, trim=0.1),
    mad=mad(DriveLoad, constant = 1.4826),
    size = length(DriveLoad))

g_DTorque<-g_300%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean(DriveTorque),
    Median  = median(DriveTorque),
    Mode = Mode(DriveTorque),
    Q1 = quantile(DriveTorque, 0.25),
    Q3 = quantile(DriveTorque, 0.75),
    IQR = Q3-Q1,
    SD = sd(DriveTorque),
    SD_trim = sd_trim(DriveTorque, trim=0.1),
    mad=mad(DriveTorque, constant = 1.4826),
    size = length(DriveTorque))

g_DTemp<-g_300%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean(DriveTemp),
    Median  = median(DriveTemp),
    Mode = Mode(DriveTemp),
    Q1 = quantile(DriveTemp, 0.25),
    Q3 = quantile(DriveTemp, 0.75),
    IQR = Q3-Q1,
    SD = sd(DriveTemp),
    SD_trim = sd_trim(DriveTemp, trim=0.1),
    mad=mad(DriveTemp, constant = 1.4826),
    size = length(DriveTemp))

summarised_Drive_mean <- setNames(data.frame(cbind(g_DPower$PartCount , g_DPower$Mean, g_DCurrent$Mean, g_DLoad$M
ean, g_DTorque$Mean, g_DTemp$Mean)),
                                  c("PartCount", "DPow.mean", "DCurr.mean", "DLoad.mean", "DTorque.mean", "DTemp.
mean"))

summarised_Drive_median <- setNames(data.frame(cbind(g_DPower$PartCount , g_DPower$Median, g_DCurrent$Median, g_D
Load$Median, g_DTorque$Median, g_DTemp$Median)),
                                  c("PartCount", "DPow.median", "DCurr.median", "DLoad.median", "DTorque.median",
"DTemp.median"))

summarised_Drive_mad <- setNames(data.frame(cbind(g_DPower$PartCount , g_DPower$mad, g_DCurrent$mad, g_DLoad$mad,
g_DTorque$mad, g_DTemp$mad)),
                                  c("PartCount", "DPow.mad", "DCurr.mad", "DLoad.mad", "DTorque.mad", "DTemp.ma
d"))


#----------- S1

g_Spow<-g_300_S1%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean(abs(Power)),
    Mean_trim= mean(abs(Power), trim = 0.1),
    Median  = median(abs(Power)),
    Mode = Mode(abs(Power)),
    Q1 = quantile(Power, 0.25),
    Q3 = quantile(Power, 0.75),
    IQR = Q3-Q1,
    SD = sd(Power),
    SD_trim = sd_trim(Power, trim=0.1),
    mad=mad(Power, constant = 1.4826),
    size = length(Power))

g_Scurr<-g_300_S1%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean(abs(Current)),
    Mean_trim=mean(abs(Current), trim =0.1),
    Median  = median(abs(Current)),
    Mode = Mode(abs(Current)),
    Q1 = quantile(Current, 0.25),
    Q3 = quantile(Current, 0.75),
    IQR = Q3-Q1,
    SD = sd(Current),
    SD_trim = sd_trim(Current, trim=0.1),
    mad=mad(Current, constant = 1.4826),
    size = length(Current))

g_Sload<-g_300_S1%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean(Load),
    Mean_trim=mean(abs(Load), trim =0.1),
    Median  = median(Load),
    Mode = Mode(Load),
    Q1 = quantile(Load, 0.25),
    Q3 = quantile(Load, 0.75),
    IQR = Q3-Q1,
    SD = sd(Load),
    SD_trim = sd_trim(Load, trim=0.1),
    mad=mad(Load, constant = 1.4826),
    size = length(Load))

g_STorque<-g_300_S1%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean((Torque)),
    Mean_trim=mean(abs(Torque), trim =0.1),
    Median  = median((Torque)),
    Mode = Mode(Torque),
    Q1 = quantile(Torque, 0.25),
    Q3 = quantile(Torque, 0.75),
    IQR = Q3-Q1,
    SD = sd(Torque),
    SD_trim = sd_trim(Torque, trim=0.1),
    mad=mad(Torque, constant = 1.4826),
    size = length(Torque))

# Creating the data frames and respective csv files for future use

summarised_S1_mean <- setNames(data.frame(
  cbind(g_Spow$PartCount ,
        g_Spow$Mean,
        g_Scurr$Mean,
        g_Sload$Mean,
        g_STorque$Mean)),
  c("PartCount", "S1Pow.mean", "S1Curr.mean", "S1Load.mean", "S1Torque.mean"))

#write.csv(summarised_S1_mean,"~/onedrive/Scenic/data/indexG220/ml/summarised_S1_mean_3.csv", row.names = FALSE)

summarised_S1_Median <- setNames(data.frame(
  cbind(g_Spow$PartCount ,
        g_Spow$Median,
        g_Scurr$Median,
        g_Sload$Median,
        g_STorque$Median)),
  c("PartCount", "S1Pow.Median", "S1Curr.Median", "S1Load.Median", "S1Torque.Median"))

summarised_S1_mad <- setNames(data.frame(
  cbind(g_Spow$PartCount ,
        g_Spow$mad,
        g_Scurr$mad,
        g_Sload$mad,
        g_STorque$mad)),
  c("PartCount", "S1Pow.mad", "S1Curr.mad", "S1Load.mad", "S1Torque.mad"))

# Pairwise plots to confirm belief that there should be a linear relationship between Torque and Load.

ggplot(summarised_S1_mean,
       aes(x=abs(S1Torque.mean),
           y=abs(S1Load.mean),
           color = as.factor(PartCount))) +
  geom_point()+
  labs(x = expression(bold("S1 Torque")),
       y=expression(bold("S1 Load")),
       title = expression(bold("Mean Load vs Torque Pairwise plot"))) +
  theme(plot.title=element_text(hjust=0.5, size=18)) +
  theme(legend.position = "none")
```



Repeating the same as above for Spindle S4

Hide

```
#----------- S4

g_S4_pow<-g_300_S4%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean(abs(Power)),
    Mean_trim=mean(abs(Power), trim =0.1),
    Median  = median(abs(Power)),
    Mode = Mode(abs(Power)),
    Q1 = quantile(Power, 0.25),
    Q3 = quantile(Power, 0.75),
    IQR = Q3-Q1,
    SD = sd(Power),
    SD_trim = sd_trim(Power, trim=0.1),
    mad=mad(Power, constant = 1.4826),
    size = length(Power))

g_S4_curr<-g_300_S4%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean(abs(Current)),
    Mean_trim=mean(abs(Current), trim =0.1),
    Median  = median(abs(Current)),
    Mode = Mode(Current),
    Q1 = quantile(Current, 0.25),
    Q3 = quantile(Current, 0.75),
    IQR = Q3-Q1,
    SD = sd(Current),
    SD_trim = sd_trim(Current, trim=0.1),
    mad=mad(Current, constant = 1.4826),
    size = length(Current))

g_S4_load<-g_300_S4%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean(Load),
    Mean_trim=mean(abs(Load)),
    Median  = median(Load),
    Mode = Mode(Load),
    Q1 = quantile(Load, 0.25),
    Q3 = quantile(Load, 0.75),
    IQR = Q3-Q1,
    SD = sd(Load),
    SD_trim = sd_trim(Load, trim=0.1),
    mad=mad(Load, constant = 1.4826),
    size = length(Load))

g_S4_Torque<-g_300_S4%>%
  group_by(PartCount) %>%
  summarise(
    Mean = mean(abs(Torque)),
    Mean_trim=mean(abs(Torque), trim =0.1),
    Median  = median(abs(Torque)),
    Mode = Mode(Torque),
    Q1 = quantile(Torque, 0.25),
    Q3 = quantile(Torque, 0.75),
    IQR = Q3-Q1,
    SD = sd(Torque),
    SD_trim = sd_trim(Torque, trim=0.1),
    mad=mad(Torque, constant = 1.4826),
    size = length(Torque))

summarised_S4_mean <- setNames(data.frame(
  cbind(g_Spow$PartCount ,
        g_S4_pow$Mean,
        g_S4_curr$Mean,
        g_S4_load$Mean,
        g_S4_Torque$Mean)),
  c("PartCount", "S4Pow.mean", "S4Curr.mean", "S4Load.mean", "S4Torque.mean"))

#write.csv(summarised_S4_mean,"~/onedrive/Scenic/data/indexG220/ml/summarised_S4_mean_3.csv", row.names = FALSE)

summarised_S4_Median <- setNames(data.frame(
  cbind(g_Spow$PartCount ,
        g_S4_pow$Median,
        g_S4_curr$Median,
        g_S4_load$Median,
        g_S4_Torque$Median)),
  c("PartCount", "S4Pow.Median", "S4Curr.Median", "S4Load.Median", "S4Torque.Median"))

summarised_S4_mad <- setNames(data.frame(
  cbind(g_Spow$PartCount ,
        g_S4_pow$mad,
        g_S4_curr$mad,
        g_S4_load$mad,
        g_S4_Torque$mad)),
  c("PartCount", "S4Pow.mad", "S4Curr.mad", "S4Load.mad", "S4Torque.mad"))
```

Plots

From this point onwards, the majority of this section is a display of plots which were used to explore the data in different ways. The ultimate result was that it was realised that focusing on the Median values was misguided. The thought was that the anomalous data was a distraction, masking more valuable data when in reality, the anomalous data was precisely what was useful.

Configuring the plot layout

Hide

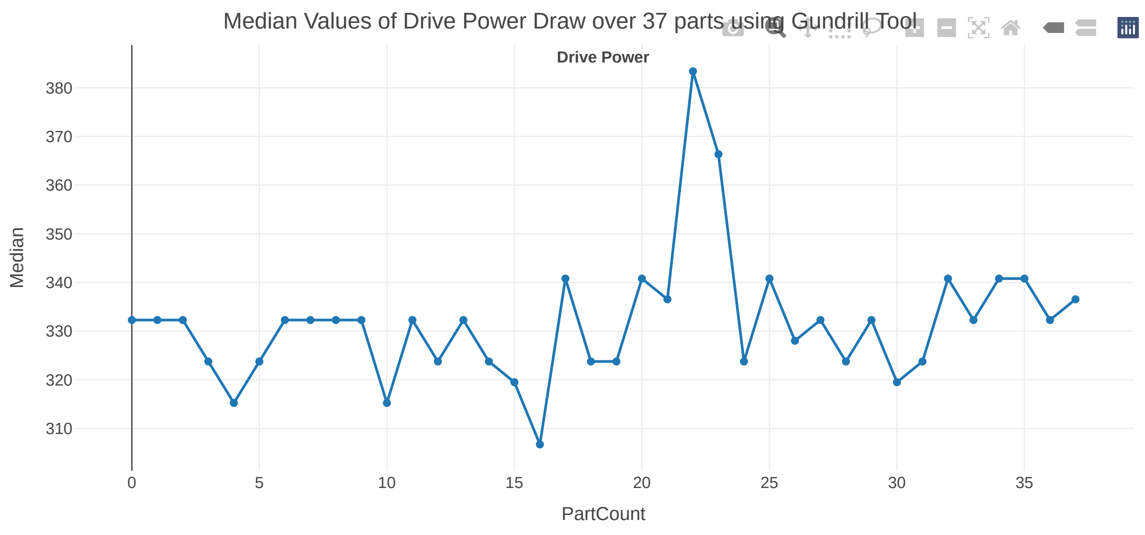
```
Dpow <- list(
  text = "<b>Drive Power</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "top",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

Dcu <- list(
  text = "<b>Drive Current</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "top",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

Dlo <- list(
  text = "<b>Drive Load</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "bottom",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

Dto <- list(
  text = "<b>Drive Torque</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "bottom",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

s1pow <- list(
  text = "<b>S1 Power</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "top",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

s1cu <- list(
  text = "<b>S1 Current</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "top",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

s1lo <- list(
  text = "<b>S1 Load</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "bottom",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

s1to <- list(
  text = "<b>S1 Torque</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "bottom",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

s4pow <- list(
  text = "<b>S4 Power</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "top",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

s4cu <- list(
  text = "<b>S4 Current</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "top",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

s4lo <- list(
  text = "<b>S4 Load</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "bottom",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

s4to <- list(
  text = "<b>S4 Torque</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "bottom",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)


pow_title <- list(
  text = "<b>Power</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "top",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)
```

```
curr_title <- list(
  text = "<b>Current</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "top",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

lo_title <- list(
  text = "<b>Load</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "bottom",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)

to_title <- list(
  text = "<b>Torque</b>",
  xref = "paper",
  yref = "paper",
  yanchor = "bottom",
  xanchor = "center",
  align = "center",
  x = 0.5,
  y = 1,
  showarrow = FALSE
)


t <- list(
  family = "Calibri",
  size = 25,
  color = 'Black')
```

Summarised Plots

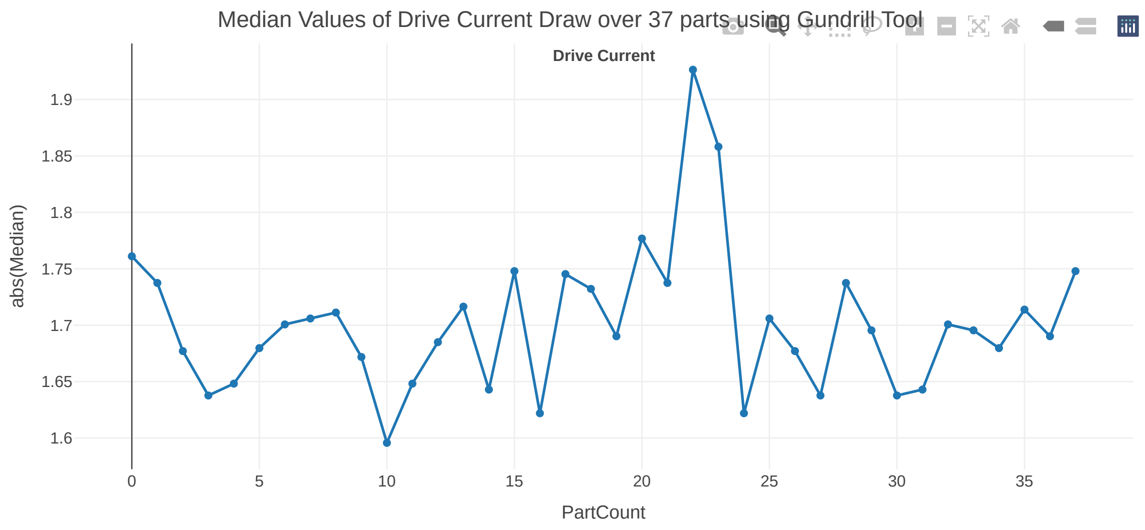
Hide

```
#--------------------------------
#        DRIVE
#--------------------------------

Dpow_plot<-plot_ly(g_DPower,
                   x = ~PartCount,
                   y = ~Median, type = 'scatter',
                   mode = 'lines+markers') %>%
  layout(annotations = Dpow,
         title = "Median Values of Drive Power Draw over 37 parts using Gundrill Tool")
Dpow_plot
```



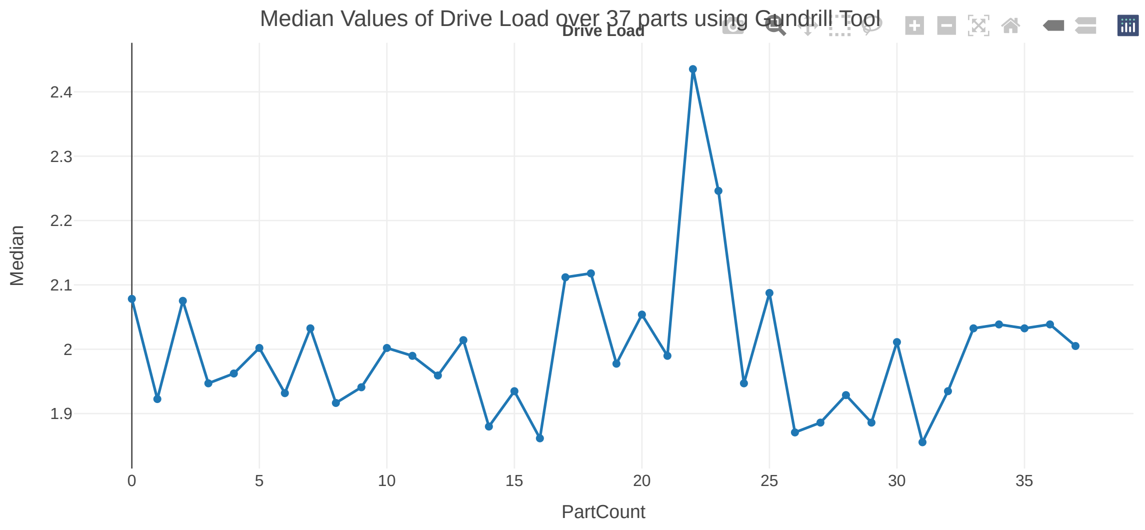
Hide

```
Dcurr_plot<-plot_ly(g_DCurrent,
                    x = ~PartCount,
                    y = ~abs(Median),
                    type = 'scatter',
                    mode = 'lines+markers') %>%
  layout( annotations = Dcu,
          title = "Median Values of Drive Current Draw over 37 parts using Gundrill Tool")
Dcurr_plot
```



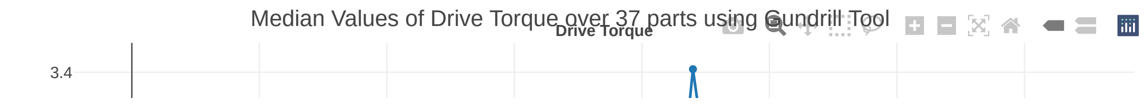
Hide

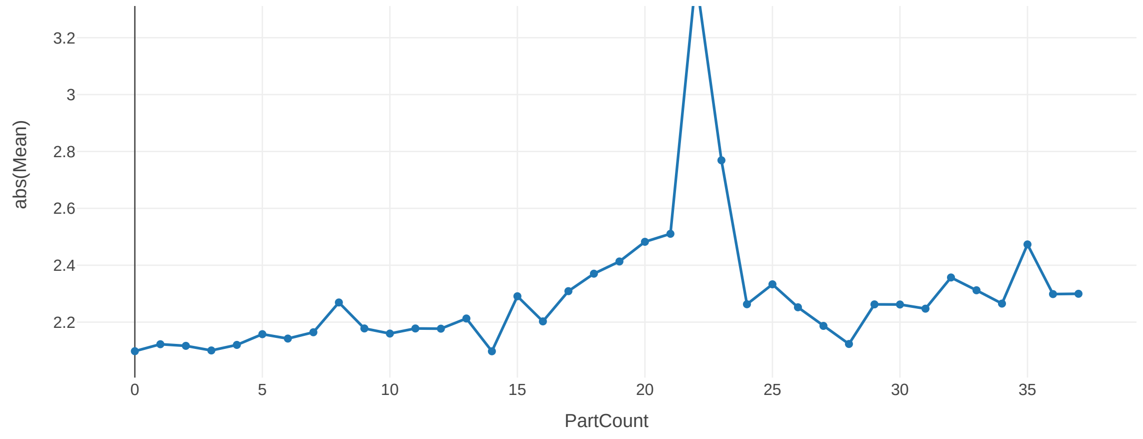
```
Dload_plot<-plot_ly(g_DLoad,
                    x = ~PartCount,
                    y = ~Median, type = 'scatter',
                    mode = 'lines+markers') %>%
  layout(annotations = Dlo,
         title = "Median Values of Drive Load over 37 parts using Gundrill Tool")
Dload_plot
```



Hide

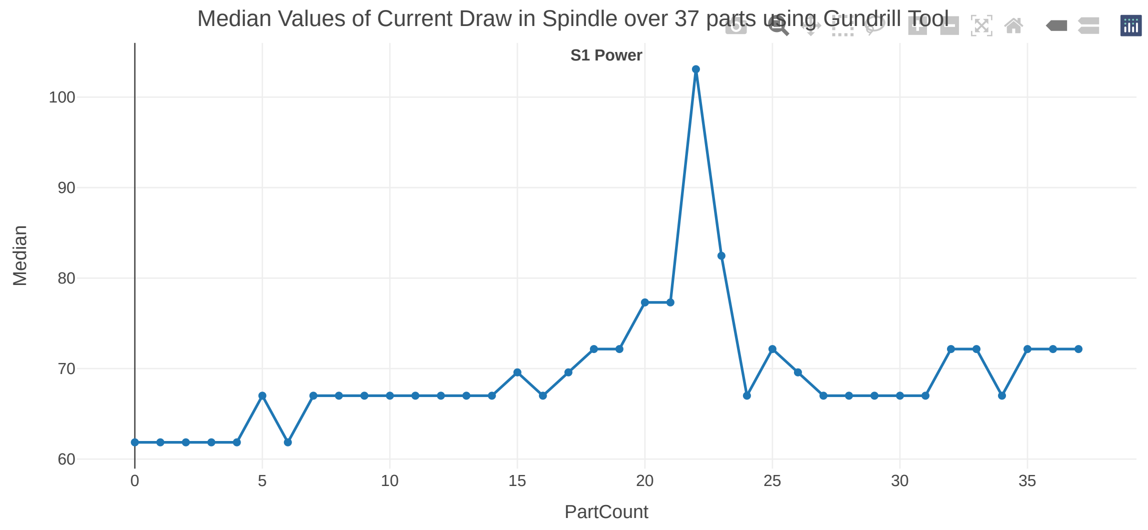
```
Dtorque_plot<-plot_ly(g_STorque,
                      x = ~PartCount,
                      y = ~abs(Mean), type = 'scatter',
                      mode = 'lines+markers',
                      name="Mean") %>%
  layout(annotations = Dto,
         title = "Median Values of Drive Torque over 37 parts using Gundrill Tool")
Dtorque_plot
```





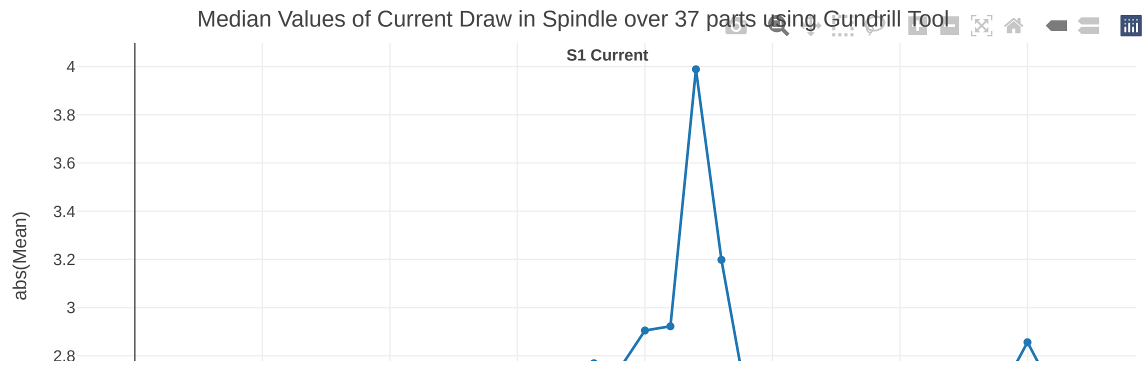
Hide

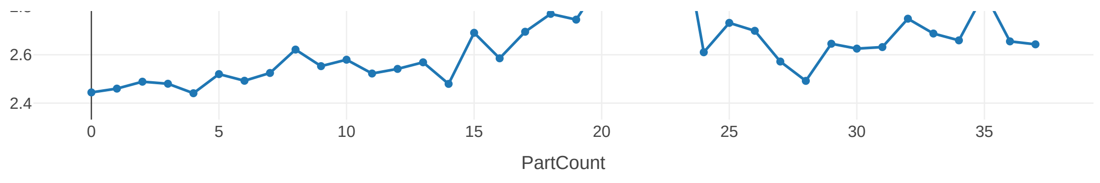
```
#-------------------------------
#      SPINDLE 1
#-------------------------------

s1pow_plot<-plot_ly(g_Spow,
                    x = ~PartCount,
                    y = ~Median, type = 'scatter',
                    mode = 'lines+markers') %>%
  layout(annotations = s1pow,
         title = "Median Values of Current Draw in Spindle over 37 parts using Gundrill Tool")
s1pow_plot
```



Hide

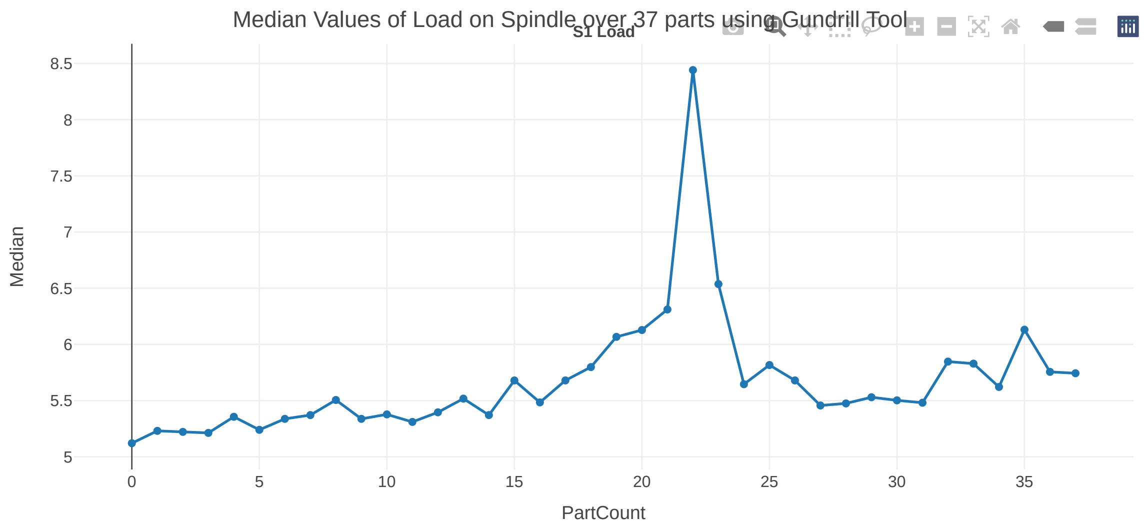
```
s1curr_plot<-plot_ly(g_Scurr,
                     x = ~PartCount,
                     y = ~abs(Mean),
                     type = 'scatter',
                     mode = 'lines+markers') %>%
  layout(annotations = s1cu,
         title = "Median Values of Current Draw in Spindle over 37 parts using Gundrill Tool")
s1curr_plot
```





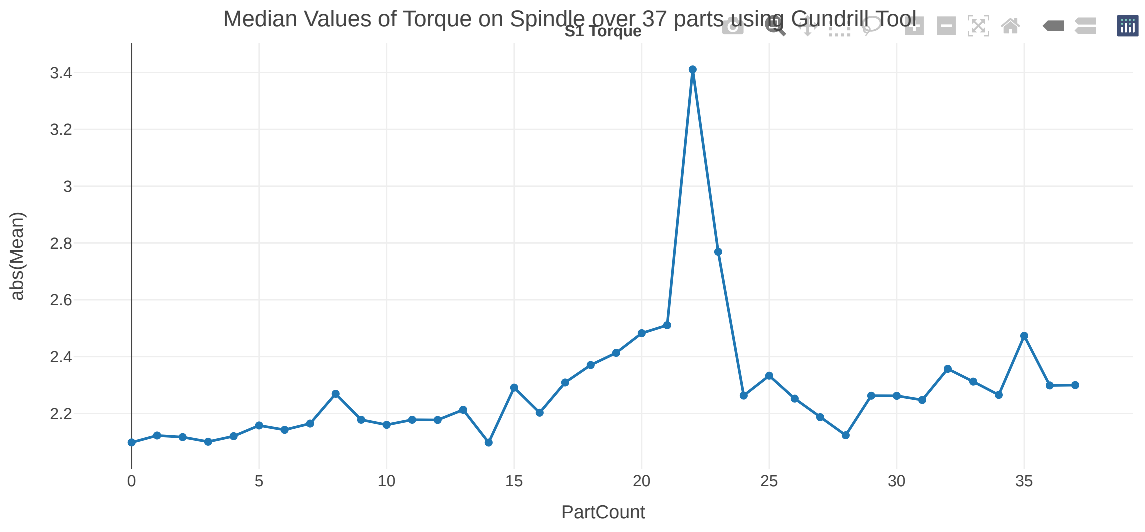
Hide

```
s1load_plot<-plot_ly(g_Sload,
                     x = ~PartCount,
                     y = ~Median,
                     type = 'scatter',
                     mode = 'lines+markers') %>%
  layout(annotations = s1lo,
         title = "Median Values of Load on Spindle over 37 parts using Gundrill Tool")
s1load_plot
```



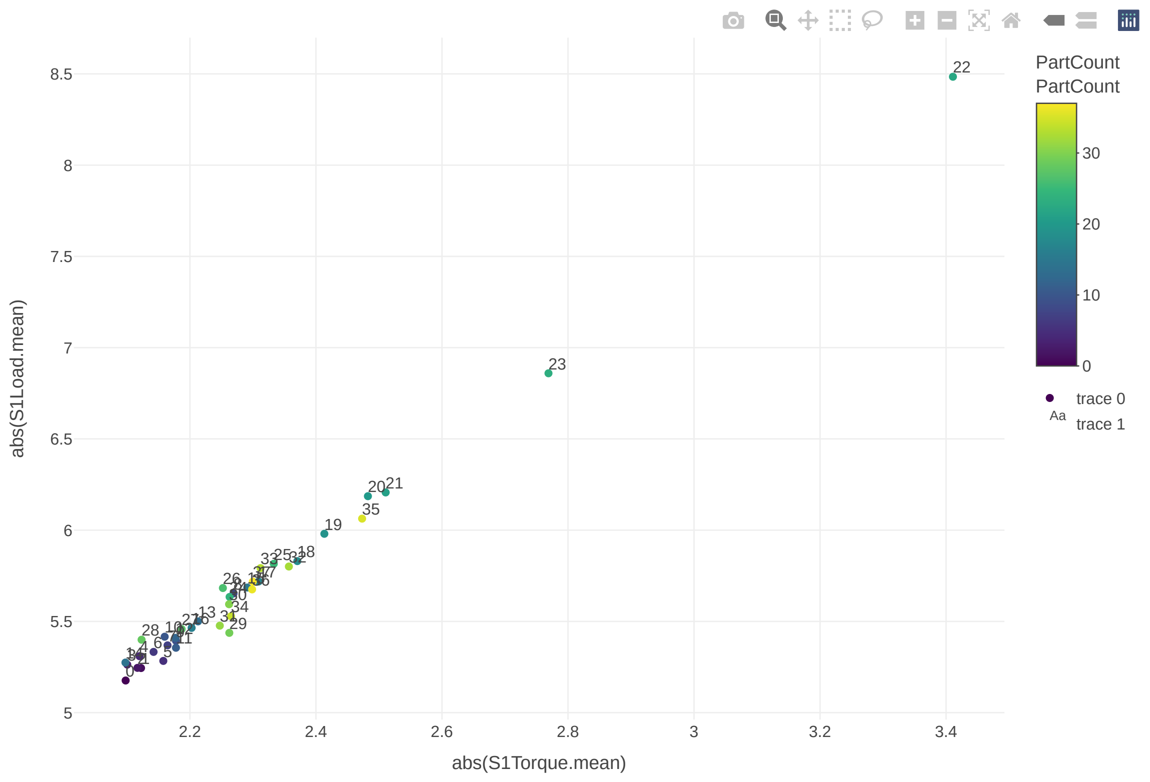
Hide

```
s1torque_plot<-plot_ly(g_STorque,
                       x = ~PartCount,
                       y = ~abs(Mean),
                       type = 'scatter',
                       mode = 'lines+markers',
                       name="Mean") %>%
  layout(annotations = s1to,
         title = "Median Values of Torque on Spindle over 37 parts using Gundrill Tool")
s1torque_plot
```



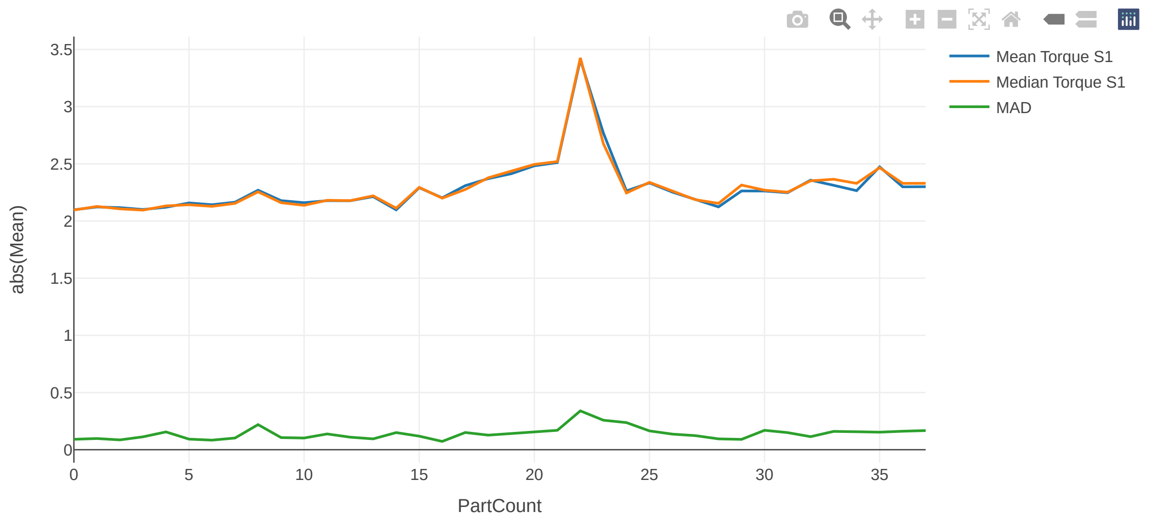
Hide

```
#----------------------------------- messing around ------------------------------------
plot_ly(summarised_S1_mean, x=~abs(S1Torque.mean), y=~abs(S1Load.mean), color=~PartCount, text=~PartCount) %>%
  add_markers() %>%
  add_text(textposition = "top right")
```



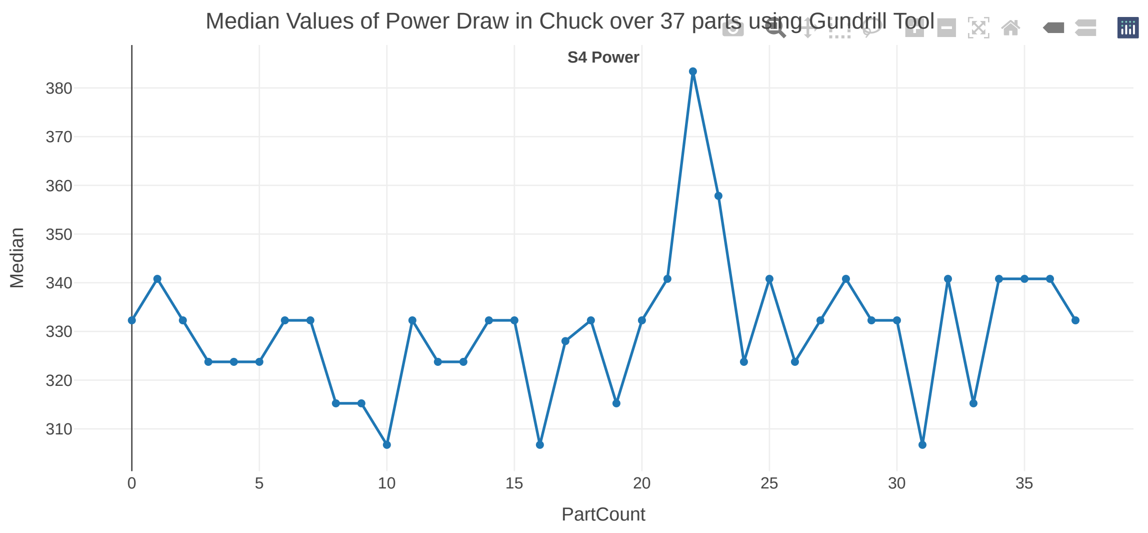
Hide

```
plot_ly(g_STorque, x = ~PartCount, y = ~abs(Mean),
        type = 'scatter',
        mode = 'lines',
        name = "Mean Torque S1") %>%
  add_trace(y = ~abs(Median),
            type = 'scatter',
            mode = 'lines',
            name = 'Median Torque S1')  %>%
  add_trace(y = ~abs(mad),
            type = 'scatter',
            mode = 'lines',
            name = 'MAD')
```



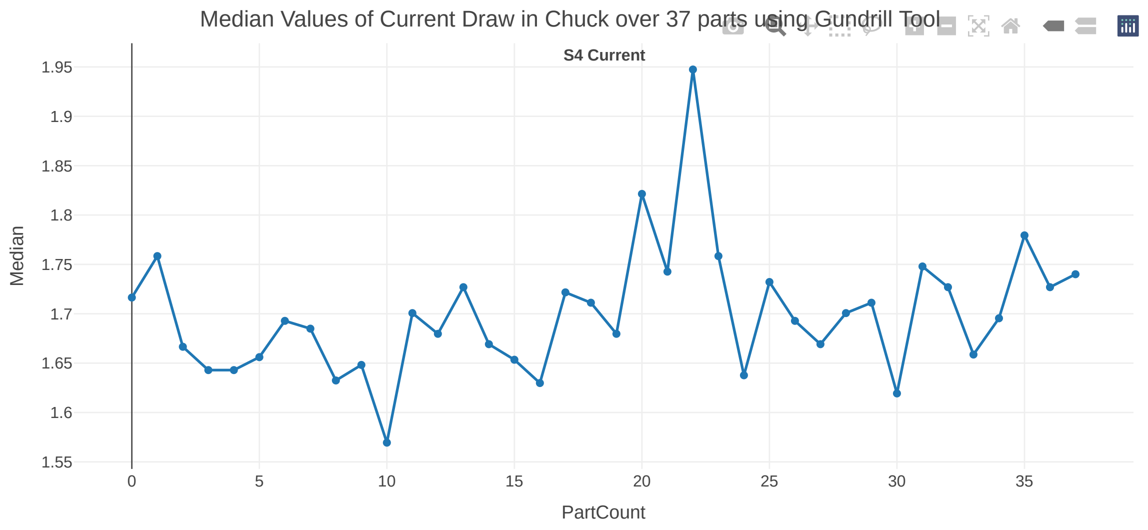
Hide

```
#--------------------------------
#      SPINDLE 4
#--------------------------------

s4pow_plot<-plot_ly(g_S4_pow,
                    x = ~PartCount,
                    y = ~Median,
                    type = 'scatter',
                    mode = 'lines+markers') %>%
  layout( annotations = s4pow,
          title = "Median Values of Power Draw in Chuck over 37 parts using Gundrill Tool")
s4pow_plot
```



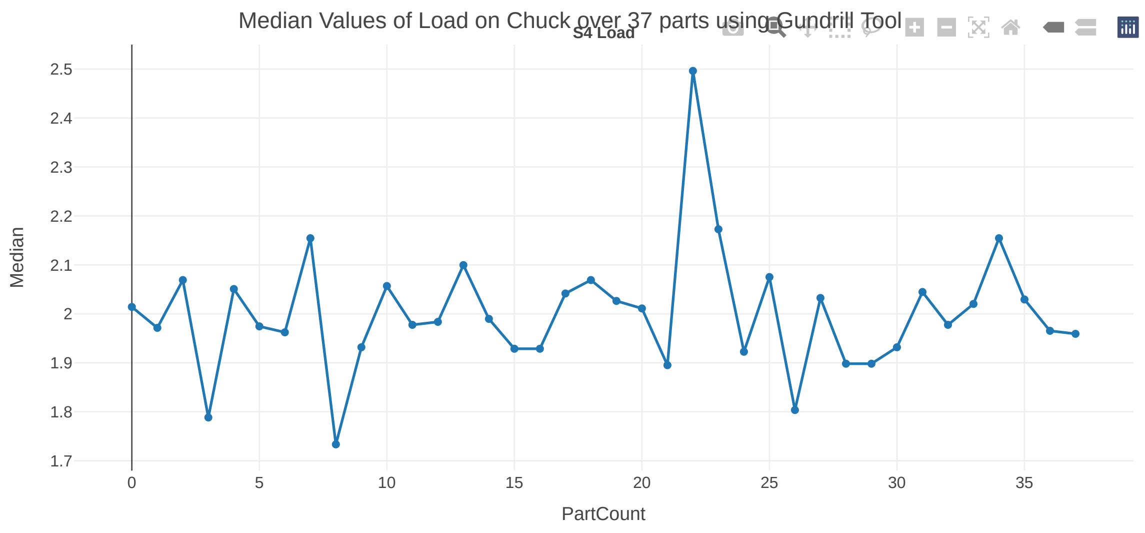
Hide

```
s4curr_plot<-plot_ly(g_S4_curr,
                     x = ~PartCount,
                     y = ~Median,
                     type = 'scatter',
                     mode = 'lines+markers') %>%
  layout(annotations = s4cu,
         title = "Median Values of Current Draw in Chuck over 37 parts using Gundrill Tool")
s4curr_plot
```



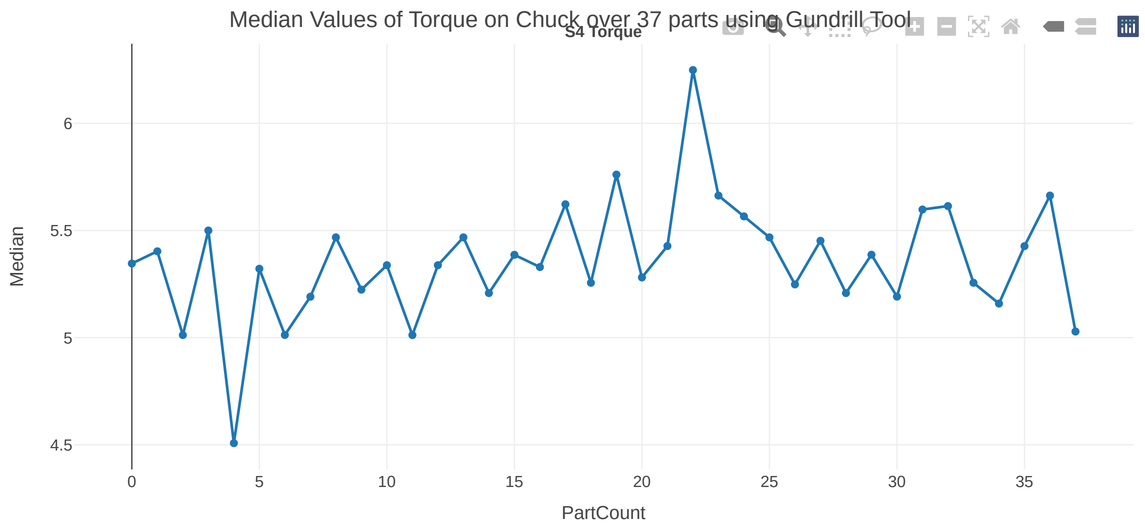
Hide

```
s4load_plot<-plot_ly(g_S4_load,
                     x = ~PartCount,
                     y = ~Median,
                     type = 'scatter',
                     mode = 'lines+markers') %>%
  layout(annotations = s4lo,
         title = "Median Values of Load on Chuck over 37 parts using Gundrill Tool")
s4load_plot
```



Hide

```
s4torque_plot<-plot_ly(g_S4_Torque,
                       x = ~PartCount,
                       y = ~Median,
                       type = 'scatter',
                       mode = 'lines+markers') %>%
  layout(annotations = s4to, title = "Median Values of Torque on Chuck over 37 parts using Gundrill Tool")
s4torque_plot
```



Hide

```
NA
```

Creating data frames combining the values of all the spindles together for each individual parameter and statistical measure.

Hide

```
#--------------------------------
#      MEAN
#--------------------------------

power_mean<-setNames(data.frame(cbind(g_DPower$PartCount, g_DPower$Mean, g_Spow$Mean, g_S4_pow$Mean)), c("PartCou
nt", "Drive", "S1", "S4"))
current_mean<-setNames(data.frame(cbind(g_DCurrent$PartCount, g_DCurrent$Mean, g_Scurr$Mean, g_S4_curr$Mean)), c
("PartCount", "Drive", "S1", "S4"))
load_mean<-setNames(data.frame(cbind(g_DLoad$PartCount, g_DLoad$Mean, g_Sload$Mean, g_S4_load$Mean)), c("PartCoun
t", "Drive", "S1", "S4"))
torque_mean<-setNames(data.frame(cbind(g_DTorque$PartCount, g_DTorque$Mean, g_STorque$Mean, g_S4_Torque$Mean)), c
("PartCount", "Drive", "S1", "S4"))
```

Hide

```
#--------------------------------
#      MEDIAN
#--------------------------------
power_median<-setNames(data.frame(cbind(g_DPower$PartCount, g_DPower$Median, g_Spow$Median, g_S4_pow$Median)), c
("PartCount", "Drive", "S1", "S4"))
current_median<-setNames(data.frame(cbind(g_DCurrent$PartCount, g_DCurrent$Median, g_Scurr$Median, g_S4_curr$Medi
an)), c("PartCount", "Drive", "S1", "S4"))
load_median<-setNames(data.frame(cbind(g_DLoad$PartCount, g_DLoad$Median, g_Sload$Median, g_S4_load$Median)), c
("PartCount", "Drive", "S1", "S4"))
torque_median<-setNames(data.frame(cbind(g_DTorque$PartCount, g_DTorque$Median, g_STorque$Median, g_S4_Torque$Med
ian)), c("PartCount", "Drive", "S1", "S4"))
```

Hide

```
#--------------------------------
#      MAD
#--------------------------------
power_mad<-setNames(data.frame(cbind(g_DPower$PartCount, g_DPower$mad, g_Spow$mad, g_S4_pow$mad)), c("PartCount",
"Drive", "S1", "S4"))
current_mad<-setNames(data.frame(cbind(g_DCurrent$PartCount, g_DCurrent$mad, g_Scurr$mad, g_S4_curr$mad)), c("Par
tCount", "Drive", "S1", "S4"))
load_mad<-setNames(data.frame(cbind(g_DLoad$PartCount, g_DLoad$mad, g_Sload$mad, g_S4_load$mad)), c("PartCount",
"Drive", "S1", "S4"))
torque_mad<-setNames(data.frame(cbind(g_DTorque$PartCount, g_DTorque$mad, g_STorque$mad, g_S4_Torque$mad)), c("Pa
rtCount", "Drive", "S1", "S4"))
```

Creating plots of the above

Hide

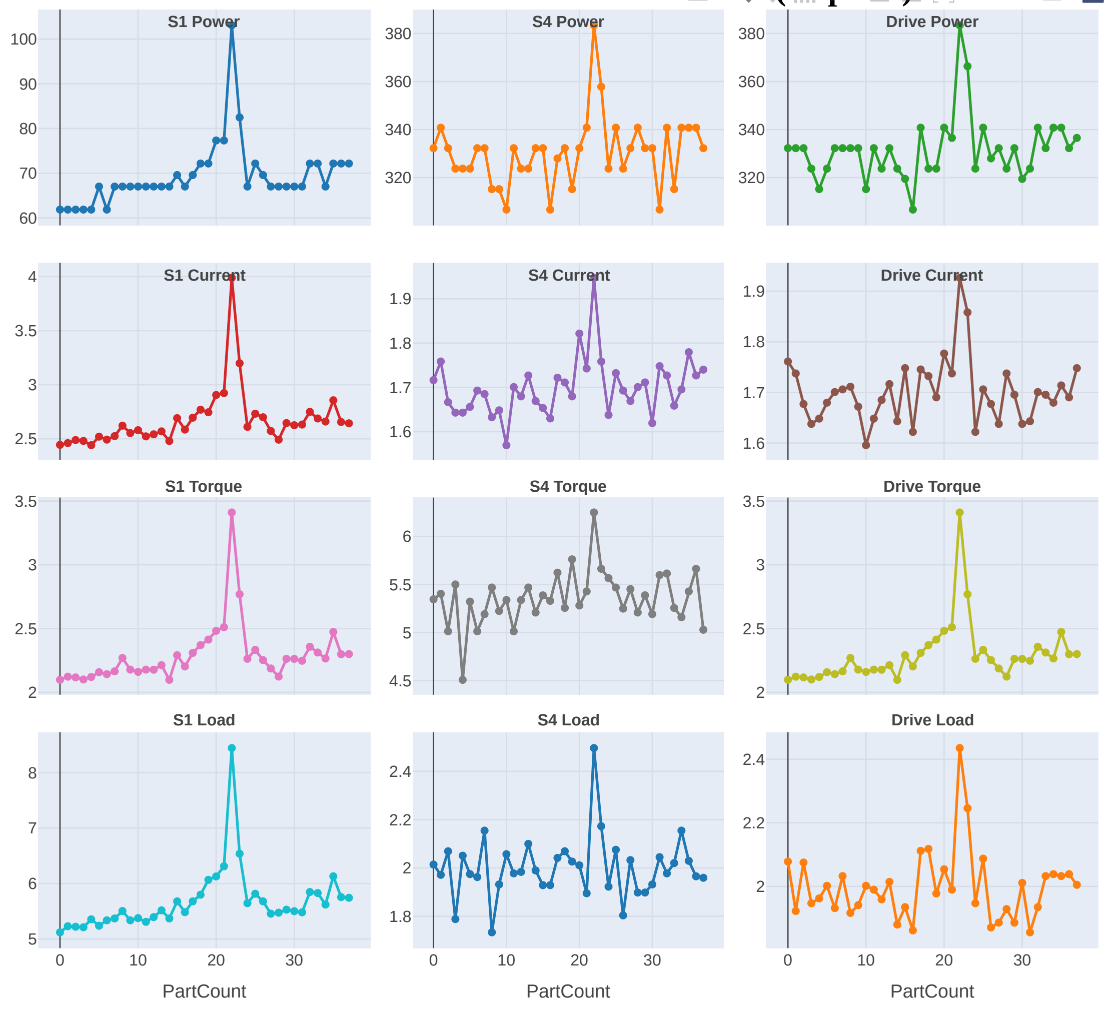
```r
mean_power_plot<-plot_ly(power_mean, x = ~PartCount, y = ~abs(Drive),
                        type = 'scatter',
                        mode = 'lines+markers',
                        name = "Drive Power") %>%
  add_trace(y = ~abs(S1),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S1 Power')  %>%
  add_trace(y = ~abs(S4),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S4 Power') %>%
  layout(annotations = pow_title, showlegend = TRUE,
         yaxis = list(title = "abs(Current)", range = c(50,500)),
         title = "Mean Values of Power Draw over 37 parts using Gundrill Tool")

mean_current_plot<-plot_ly(current_mean, x = ~PartCount, y = ~abs(Drive),
                          type = 'scatter',
                          mode = 'lines+markers',
                          name = "Drive Current") %>%
  add_trace(y = ~abs(S1),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S1 Current')  %>%
  add_trace(y = ~abs(S4),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S4 Current')%>%
  layout(annotations = curr_title, showlegend = TRUE,
         yaxis = list(title = "abs(Current)", range = c(0,5)),
         title = "Mean Values of Current Draw over 37 parts using Gundrill Tool")

mean_load_plot<-plot_ly(load_mean, x = ~PartCount, y = ~abs(Drive),
                       type = 'scatter',
                       mode = 'lines+markers',
                       name = "Drive Load") %>%
  add_trace(y = ~abs(S1),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S1 Load')  %>%
  add_trace(y = ~abs(S4),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S4 Load') %>%
  layout(annotations = lo_title, showlegend = TRUE,
         yaxis = list(title = "abs(Current)", range = c(0,10)),
         title = "Mean Values of Load over 37 parts using Gundrill Tool")

mean_torque_plot<-plot_ly(torque_mean, x = ~PartCount, y = ~abs(Drive),
                         type = 'scatter',
                         mode = 'lines+markers',
                         name = "Drive Torque") %>%
  add_trace(y = ~abs(S1),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S1 Torque')  %>%
  add_trace(y = ~abs(S4),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S4 Torque') %>%
  layout(annotations = to_title, showlegend = TRUE,
         yaxis = list(title = "abs(Current)", range = c(0,8)),
         title = "Mean Values of Torque over 37 parts using Gundrill Tool")


mad_power_plot<-plot_ly(power_mad, x = ~PartCount, y = ~abs(Drive),
        type = 'scatter',
        mode = 'lines+markers',
        name = "Drive Power") %>%
  add_trace(y = ~abs(S1),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S1 Power')  %>%
  add_trace(y = ~abs(S4),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S4 Power') %>%
  layout(annotations = pow_title, showlegend = TRUE,
         yaxis = list(title = "abs(Current)"),
         title = "MAD values of Power Draw over 37 parts using Gundrill Tool")

mad_current_plot<-plot_ly(current_mad, x = ~PartCount, y = ~abs(Drive),
        type = 'scatter',
        mode = 'lines+markers',
        name = "Drive Current") %>%
  add_trace(y = ~abs(S1),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S1 Current')  %>%
  add_trace(y = ~abs(S4),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S4 Current')%>%
  layout(annotations = curr_title, showlegend = TRUE,
         yaxis = list(title = "abs(Current)"),
         title = "MAD Values of Current Draw over 37 parts using Gundrill Tool")

mad_load_plot<-plot_ly(load_mad, x = ~PartCount, y = ~abs(Drive),
        type = 'scatter',
        mode = 'lines+markers',
        name = "Drive Load") %>%
  add_trace(y = ~abs(S1),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S1 Load')  %>%
  add_trace(y = ~abs(S4),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S4 Load') %>%
  layout(annotations = lo_title, showlegend = TRUE,
         yaxis = list(title = "abs(Current)"),
         title = "MAD Values of Load over 37 parts using Gundrill Tool")

mad_torque_plot<-plot_ly(torque_mad, x = ~PartCount, y = ~abs(Drive),
        type = 'scatter',
        mode = 'lines+markers',
        name = "Drive Torque") %>%
  add_trace(y = ~abs(S1),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S1 Torque')  %>%
  add_trace(y = ~abs(S4),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S4 Torque') %>%
  layout(annotations = to_title, showlegend = TRUE,
         yaxis = list(title = "abs(Current)"),
         title = "MAD Values of Torque over 37 parts using Gundrill Tool")
```

Grid plot of the median each Parameter for each spindle

Hide

```
#---------------
#    PLOTS
#---------------

fig1 <- subplot(s1pow_plot,
               s4pow_plot,
               Dpow_plot,
               s1curr_plot,
               s4curr_plot,
               Dcurr_plot,
               s1torque_plot,
               s4torque_plot,
               Dtorque_plot,
               s1load_plot,
               s4load_plot,
               Dload_plot,
               nrows = 4,
               shareX = TRUE) %>%
  layout(title = list(text="<b>Median Values for Gundrill Tool over 'Life'(37 parts)</b>",
                      font = t,
                      y = 1, x = 0.5,
                      xanchor = 'center',
                      yanchor =  'top'),
         showlegend =FALSE,
         plot_bgcolor='#e5ecf6')

fig1
```

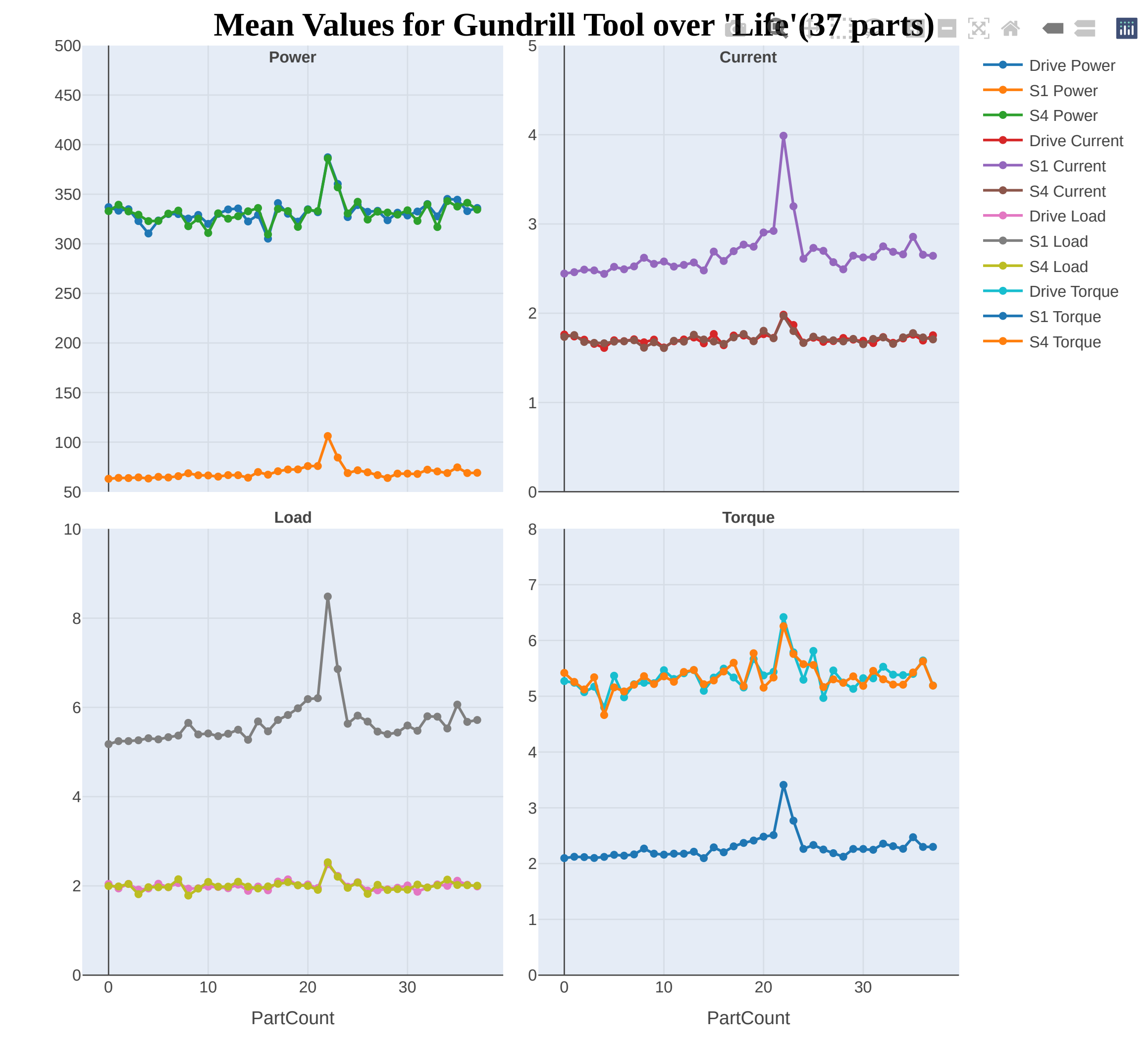
Median Values for Gundrill Tool over 'Life'(37 parts)



Grid of mean values of each Parameter with the spindles superimposed on one plot.

Hide

```
fig2 <- subplot(mean_power_plot,
                mean_current_plot,
                mean_load_plot,
                mean_torque_plot,
                nrows = 2,
                shareX = TRUE) %>%
  layout(title = list(text="<b>Mean Values for Gundrill Tool over 'Life'(37 parts)</b>",
                      font = t,
                      y = 1, x = 0.5,
                      xanchor = 'center',
                      yanchor =  'top'),
         showlegend =TRUE,
         plot_bgcolor='#e5ecf6')

fig2
```

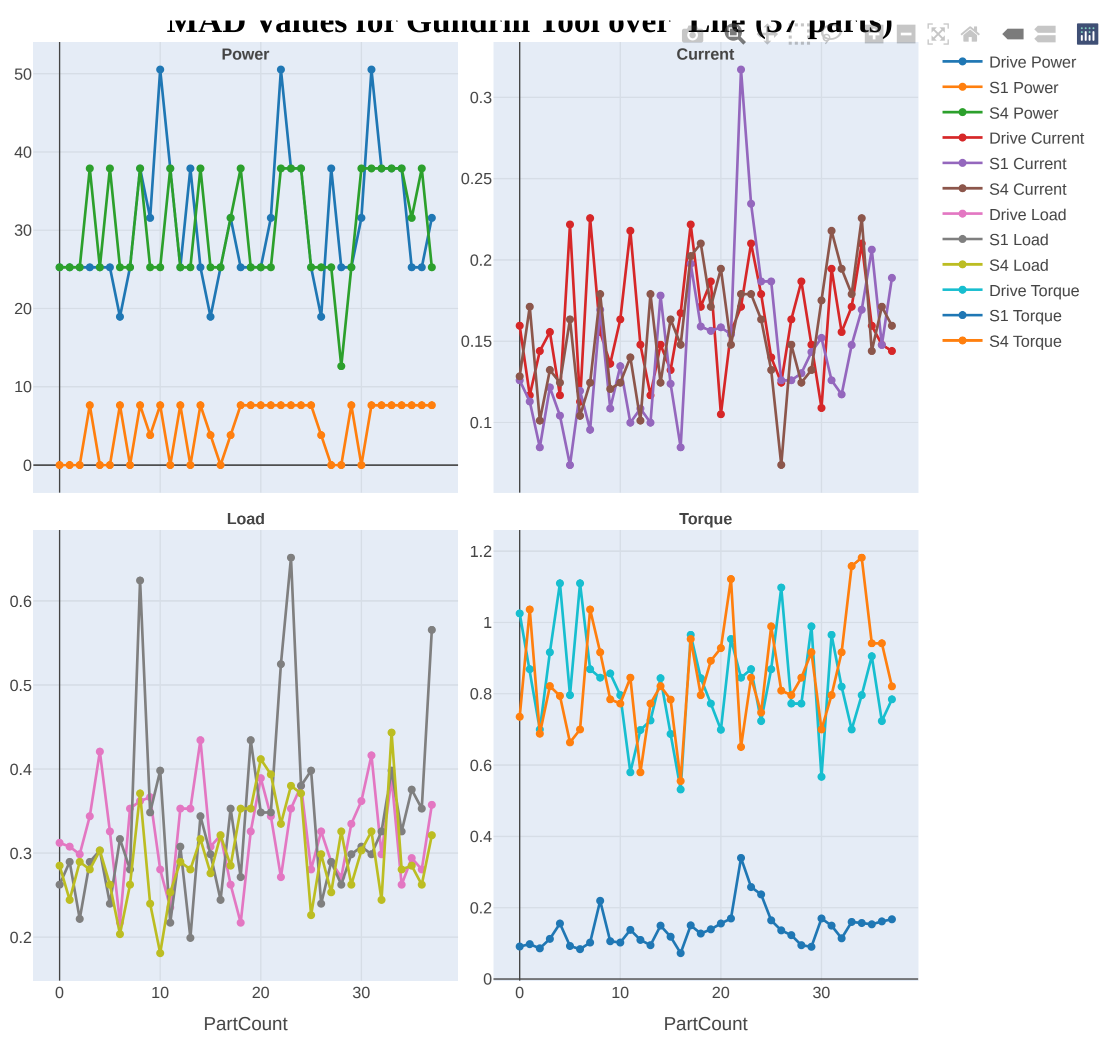


Same but for MAD

Hide

```
fig3 <- subplot(mad_power_plot,
                mad_current_plot,
                mad_load_plot,
                mad_torque_plot,
                nrows = 2,
                shareX = TRUE) %>%
  layout(title = list(text="<b>MAD Values for Gundrill Tool over 'Life'(37 parts)</b>",
                      font = t,
                      y = 1, x = 0.5,
                      xanchor = 'center',
                      yanchor =  'top'),
         showlegend =TRUE,
         plot_bgcolor='#e5ecf6')

fig3
```

MAD Values for Gundrill Tool over 'Life'(37 parts)



Instead of visualising the summarised values for each part count as above, it was decided to explore the whole, unsummarised, dataset grouping the values into small clusters of 5 parts. Since the Gundrill is used so extensively it has a sister tool and from the data analysis at the beginning, it was noted that the tool expires and is replaced after Part 37, hence the sets only go up to 40 here. The *for loop* extends up to 75 because this code was used for other tools that did not have a sister tool.

Hide

```
#-----------------------------------------

df_g_s1<-g_300_S1
df_g_s4<-g_300_S4

count1 <-c()
for (val in df_g_s1$PartCount){
  if(val<6){
    count1<-append(count1,5)
  }else if(val>=6 & val<11){
    count1<-append(count1,10)
  }else if(val>=11 & val<16){
    count1<-append(count1,15)
  }else if(val>=16 & val<21){
    count1<-append(count1,20)
  }else if(val>=21 & val<26){
    count1<-append(count1,25)
  }else if(val>=26 & val<31){
    count1<-append(count1,30)
  }else if(val>=31 & val<36){
    count1<-append(count1,35)
  }else if(val>=36 & val<41){
    count1<-append(count1,40)
  }else if(val>=41 & val<46){
    count1<-append(count1,45)
  }else if(val>=46 & val<51){
    count1<-append(count1,50)
  }else if(val>=51 & val<56){
    count1<-append(count1,55)
  }else if(val>=56 & val<61){
    count1<-append(count1,60)
  }else if(val>=61 & val<66){
    count1<-append(count1,65)
  }else if(val>=66 & val<71){
    count1<-append(count1,70)
  }else if(val>=71 & val<76){
    count1<-append(count1,75)
  }
}

#--------
# S1
#--------

length(df_g_s1)
```

```
[1] 27
```

Hide

```
nrow(df_g_s1)
```

```
[1] 2303
```

Hide

```
df_g_s1$PartCount <- count1
unique(df_g_s1$PartCount)
```

```
[1]  5 10 15 20 25 30 35 40
```

Hide

```
length(df_g_s1$PartCount)
```

```
[1] 2303
```

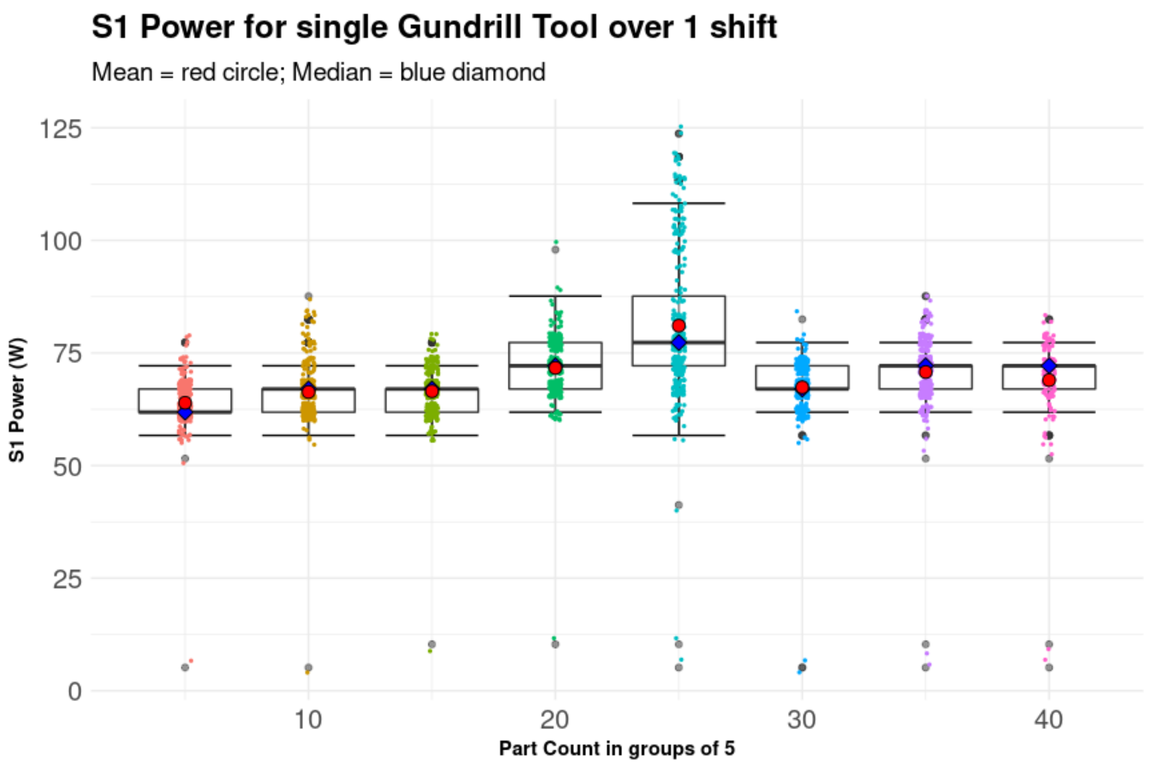
Hide

```
dftime<-which(df_g_s1$PartCount != dplyr::lag(df_g_s1$PartCount))
timez<-df_g_s1[dftime,]$Time
time<-append(df_g_s1$Time[1], timez)
time
```

```
[1] "10:21:36" "12:17:51" "13:45:41" "15:15:05" "16:44:31" "18:13:55" "19:43:22" "21:12:48"
```

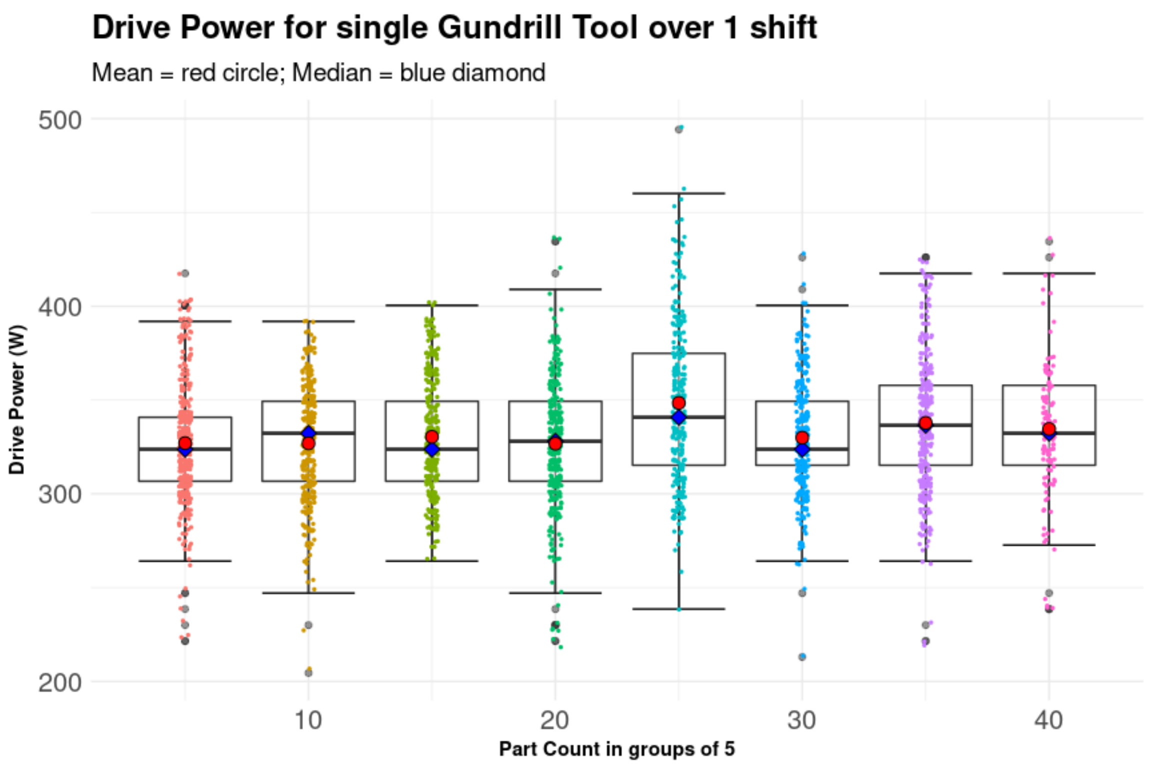
Hide

```
#Power
g_s1_power_plot <-
  ggplot(df_g_s1, aes(PartCount, Power, group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.5)+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
  #scale_y_continuous(limits = quantile(df_g_s1$Power, c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")+
  theme(legend.position = "none")+
 labs(x=expression(bold("Part Count in groups of 5")),
      y=expression(bold("S1 Power (W)")),
      title=expression(bold("S1 Power for single Gundrill Tool over 1 shift")),
      subtitle=expression("Mean = red circle; Median = blue diamond")) +
 theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size=
15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
 scale_color_discrete(name =expression(underline("Group of 5 Parts"))) +theme(legend.title.align = 1, legend.titl
e = element_text(size=15)) +
 guides(color = guide_legend(override.aes = list(size = 3)))

g_s1_power_plot
```



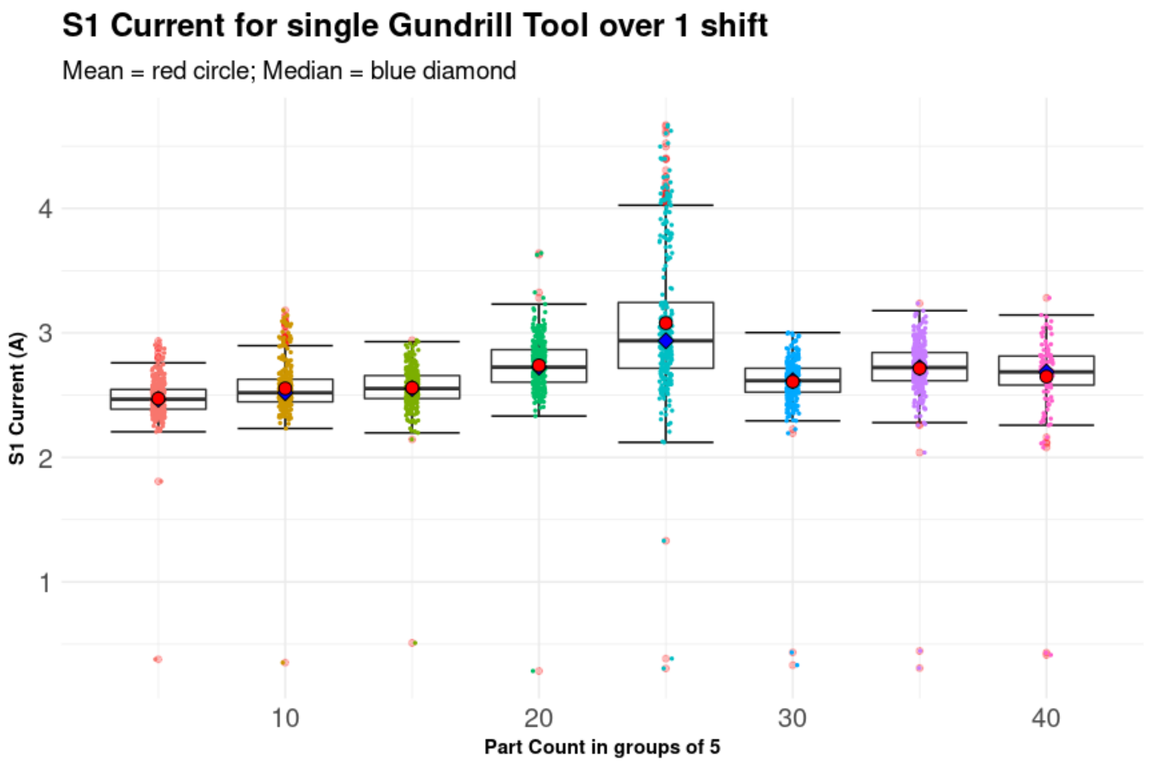
Hide

```
g_c1_power_plot <-
  ggplot(df_g_s1, aes(PartCount, DrivePower, group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.5)+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
  #scale_y_continuous(limits = quantile(df_g_s1$DrivePower, c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")+
  theme(legend.position = "none")+
  labs(x=expression(bold("Part Count in groups of 5")),
      y=expression(bold("Drive Power (W)")),
      title=expression(bold("Drive Power for single Gundrill Tool over 1 shift")),
      subtitle=expression("Mean = red circle; Median = blue diamond")) +
 theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size=
15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
 scale_color_discrete(name =expression(underline("Group of 5 Parts"))) +theme(legend.title.align = 1, legend.titl
e = element_text(size=15)) +
 guides(color = guide_legend(override.aes = list(size = 3)))

g_c1_power_plot
```



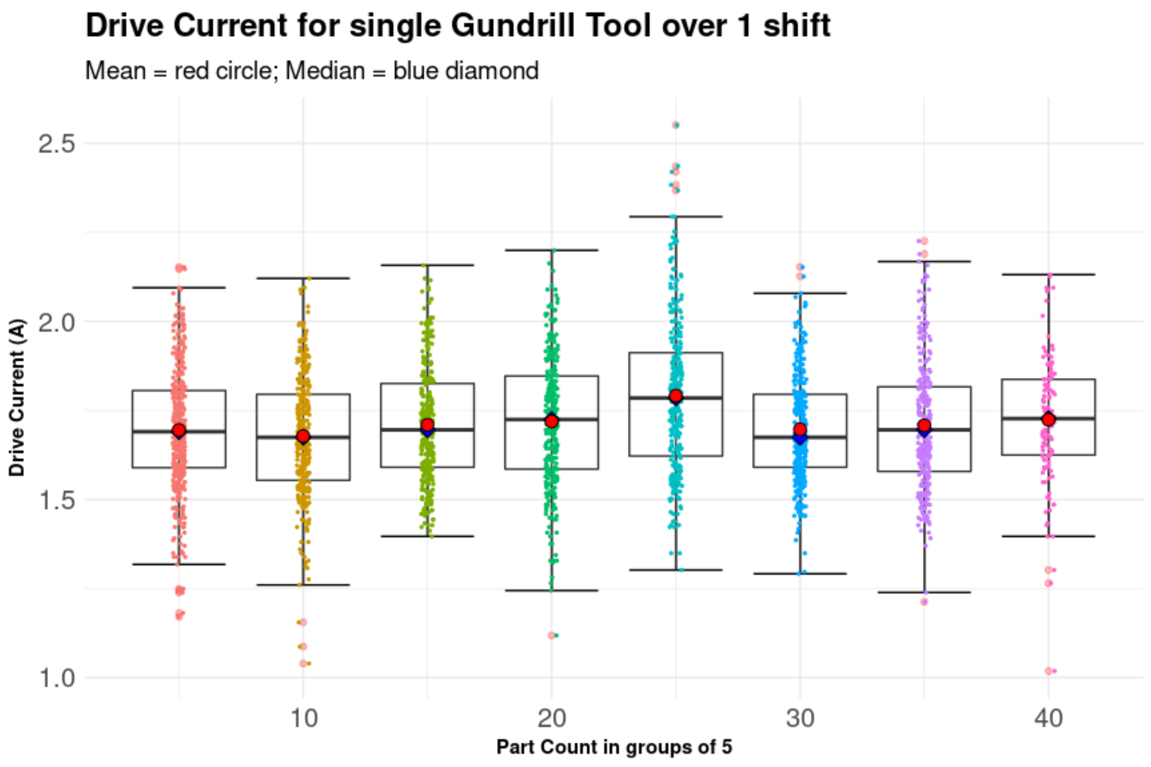
Hide

```
g_s1_current_plot <-
  ggplot(df_g_s1, aes(PartCount, abs(Current), group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
  #scale_y_continuous(limits = quantile(abs(df_g_s1$Current), c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")+
  theme(legend.position = "none")+
 labs(x=expression(bold("Part Count in groups of 5")),
      y=expression(bold("S1 Current (A)")),
      title=expression(bold("S1 Current for single Gundrill Tool over 1 shift")),
      subtitle=expression("Mean = red circle; Median = blue diamond")) +
 theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size=
15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
 scale_color_discrete(name =expression(underline("Group of 5 Parts"))) +theme(legend.title.align = 1, legend.titl
e = element_text(size=15)) +
 guides(color = guide_legend(override.aes = list(size = 3)))

g_s1_current_plot
```



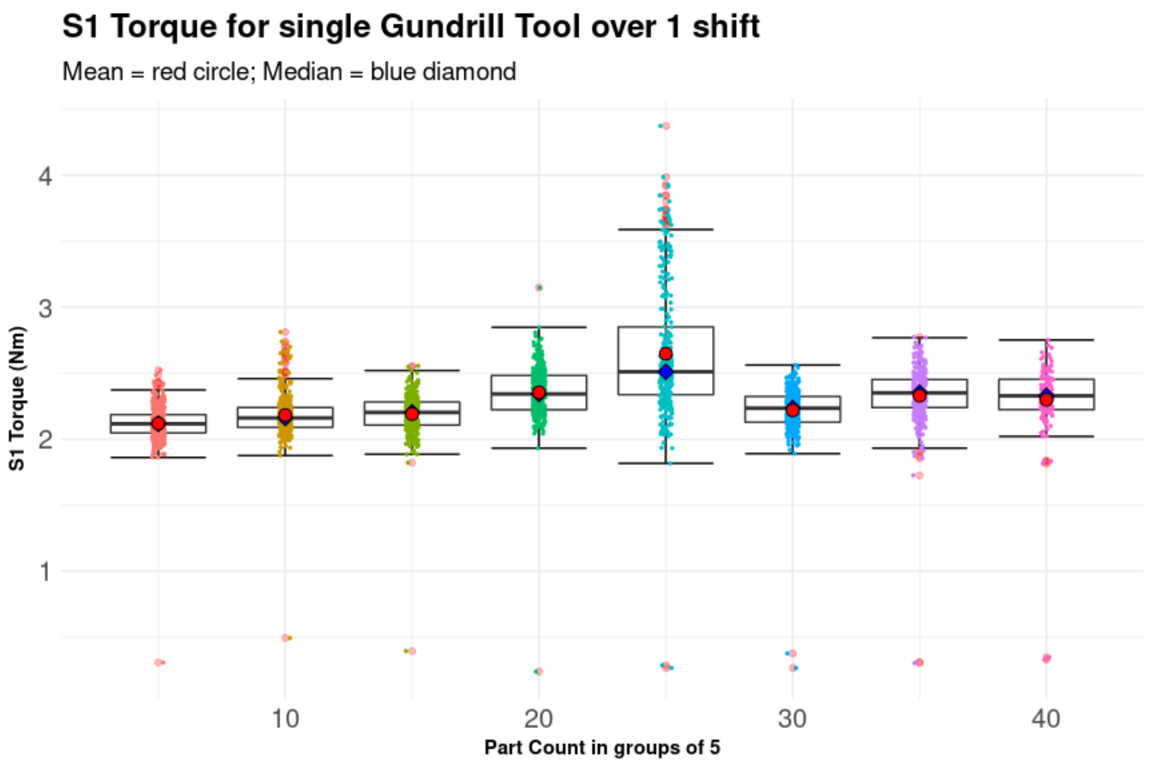
Hide

```
g_c1_current_plot <-
  ggplot(df_g_s1, aes(PartCount, abs(DriveCurrent), group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
 # scale_y_continuous(limits = quantile(abs(df_g_s1$DriveCurrent), c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")+
  theme(legend.position = "none")+
  labs(x=expression(bold("Part Count in groups of 5")),
      y=expression(bold("Drive Current (A)")),
      title=expression(bold("Drive Current for single Gundrill Tool over 1 shift")),
      subtitle=expression("Mean = red circle; Median = blue diamond")) +
 theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size=
15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
 scale_color_discrete(name =expression(underline("Group of 5 Parts"))) +theme(legend.title.align = 1, legend.titl
e = element_text(size=15)) +
 guides(color = guide_legend(override.aes = list(size = 3)))

g_c1_current_plot
```



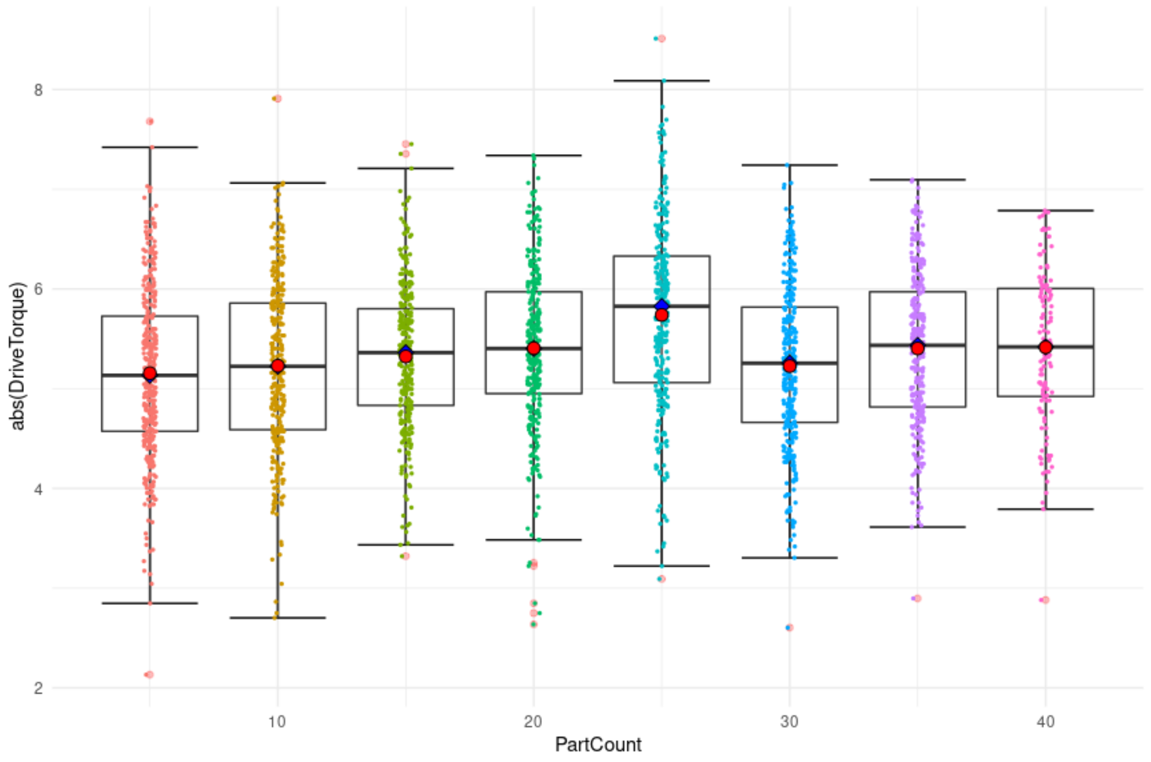
Hide

```
# Torque
g_s1_torque_plot <-
  ggplot(df_g_s1, aes(PartCount, abs(Torque), group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
  #scale_y_continuous(limits = quantile(abs(df_g_s1$Torque), c(0.01, 0.99)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")+
  theme(legend.position = "none")+
  labs(x=expression(bold("Part Count in groups of 5")),
      y=expression(bold("S1 Torque (Nm)")),
      title=expression(bold("S1 Torque for single Gundrill Tool over 1 shift")),
      subtitle=expression("Mean = red circle; Median = blue diamond")) +
 theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size=
15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
 scale_color_discrete(name =expression(underline("Group of 5 Parts"))) +theme(legend.title.align = 1, legend.titl
e = element_text(size=15)) +
 guides(color = guide_legend(override.aes = list(size = 3)))

g_s1_torque_plot
```



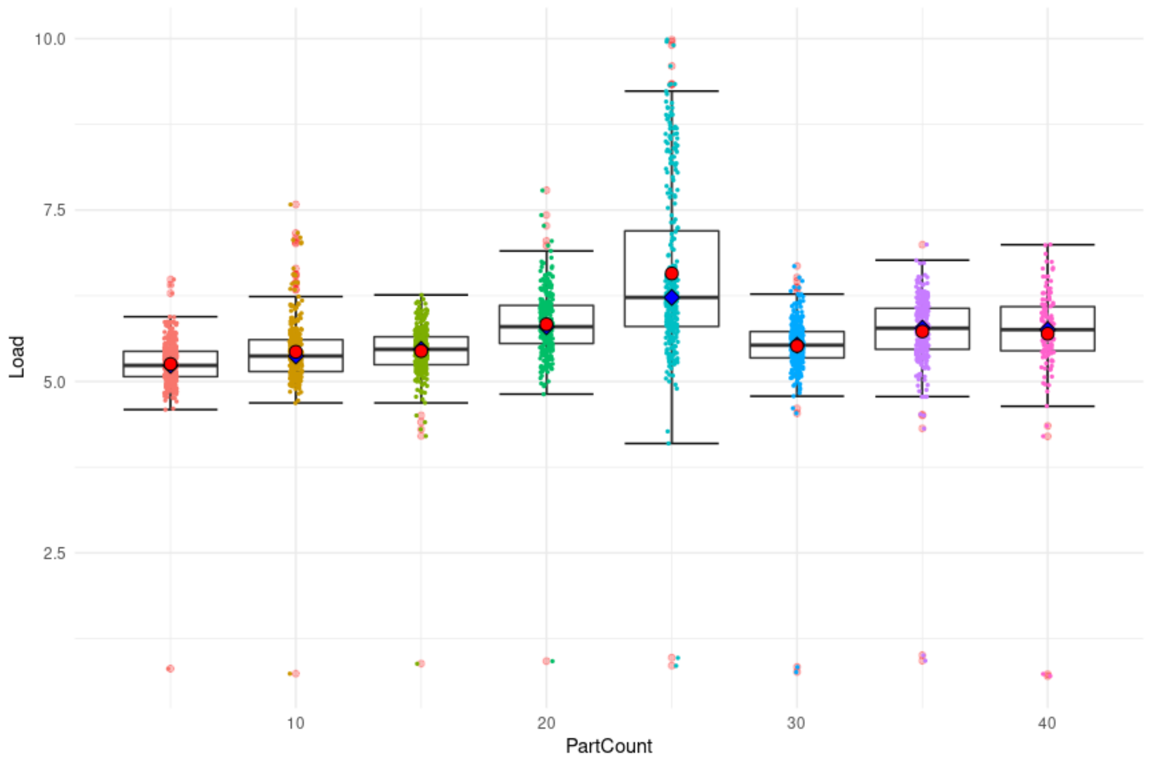
Hide

```
g_c1_torque_plot <-
  ggplot(df_g_s1, aes(PartCount, abs(DriveTorque), group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
  #scale_y_continuous(limits = quantile(abs(df_g_s1$DriveTorque), c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")#+  scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position
= "none")
# +labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 Torque (Nm)")),
#      title=expression(bold("S1 Torque for Finish Splines Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))
g_c1_torque_plot
```



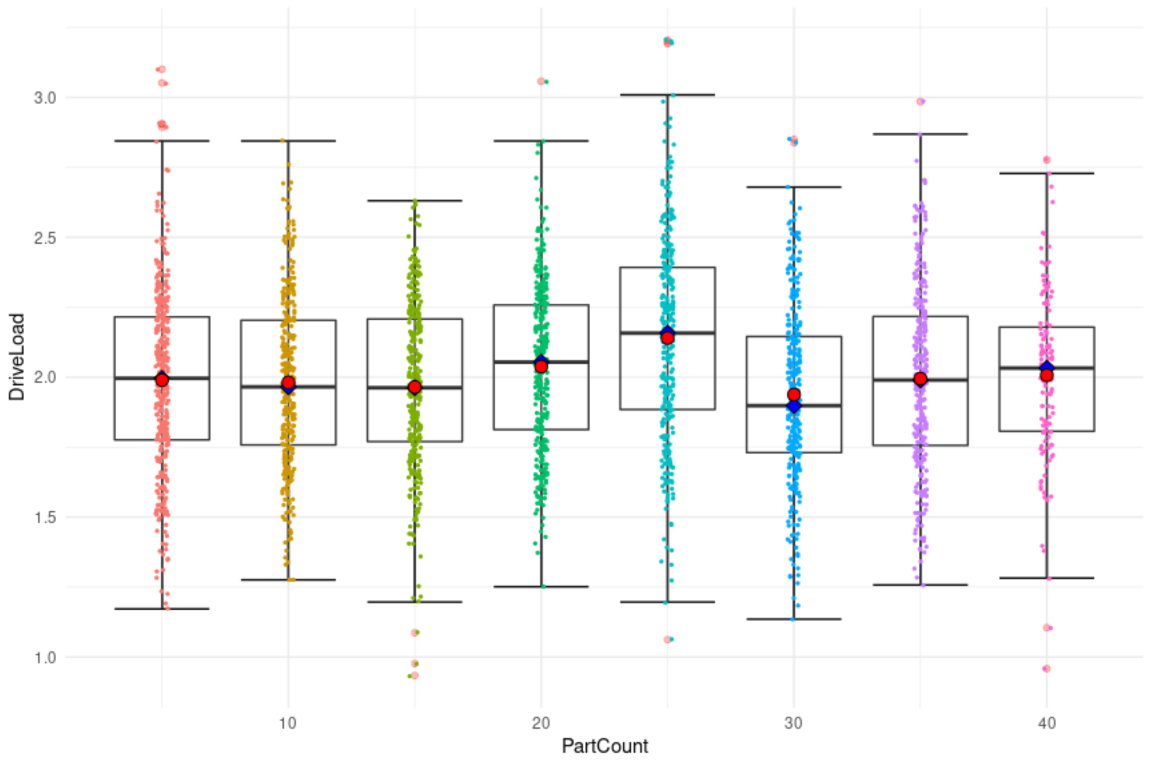
Hide

```
# Load
g_s1_load_plot <-
  ggplot(df_g_s1, aes(PartCount, Load, group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
 # scale_y_continuous(limits = quantile(df_g_s1$Load, c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")#+  scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position
= "none")
#+ labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 Load (%)")),
#      title=expression(bold("S1 Load for Finish Splines Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))
g_s1_load_plot
```



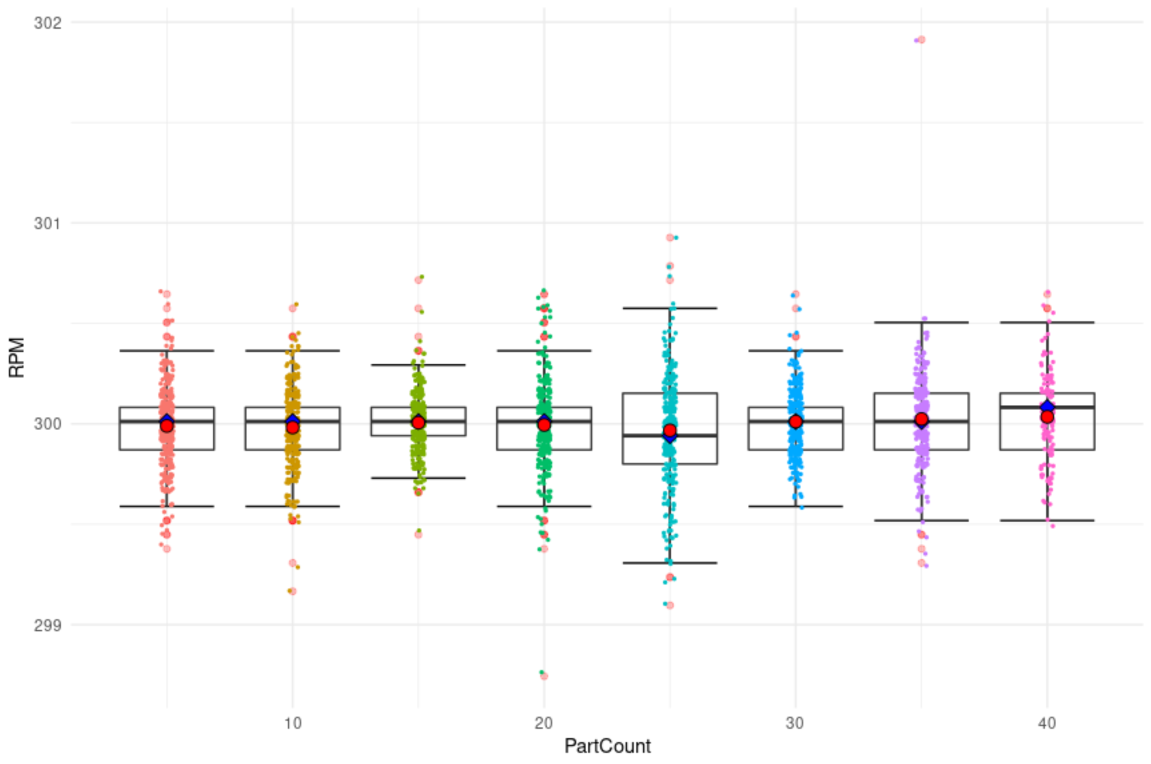
Hide

```
# Load
g_c1_load_plot <-
  ggplot(df_g_s1, aes(PartCount, DriveLoad, group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
 # scale_y_continuous(limits = quantile(df_g_s1$DriveLoad, c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")#+  scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position
= "none")
#+ labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 Load (%)")),
#      title=expression(bold("S1 Load for Finish Splines Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))
g_c1_load_plot
```



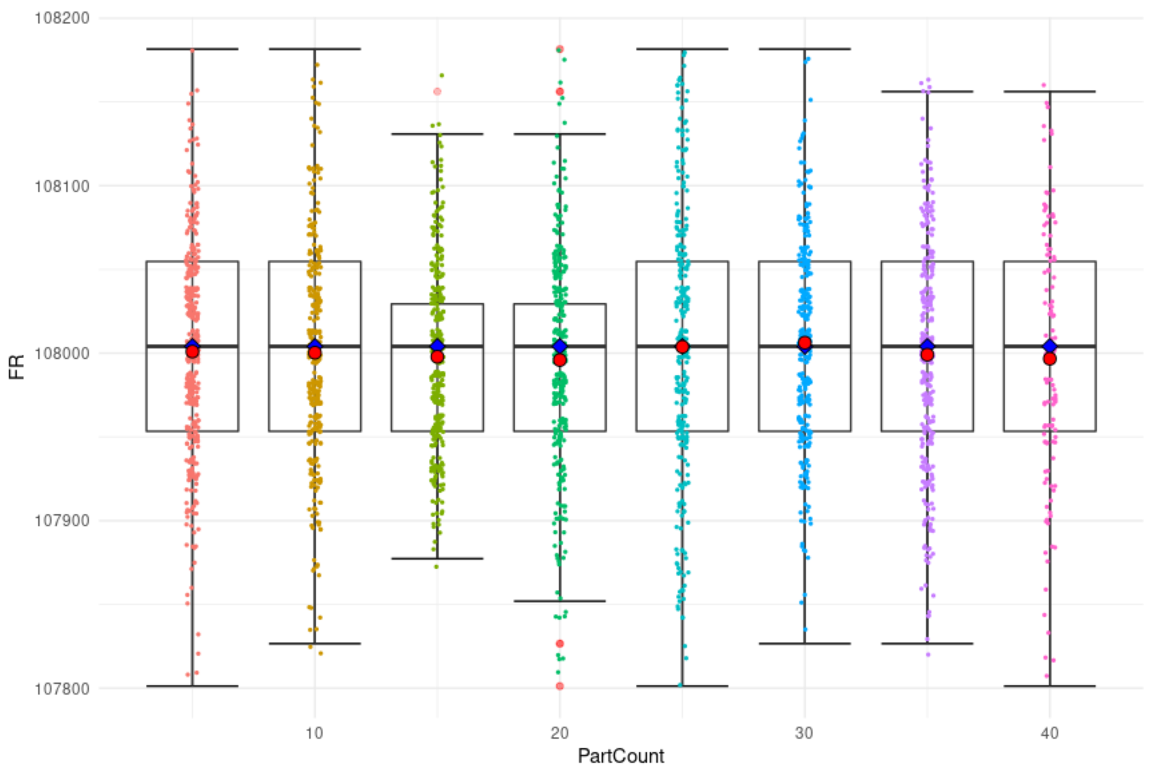
Hide

```
# RPM
g_s1_rpm_plot <-
  ggplot(df_g_s1, aes(PartCount, RPM, group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
 # scale_y_continuous(limits = quantile(df_g_s1$RPM, c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")#+  scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position
= "none")
#+labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 RPM")),
#      title=expression(bold("S1 RPM for Finish Splines Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))
g_s1_rpm_plot
```



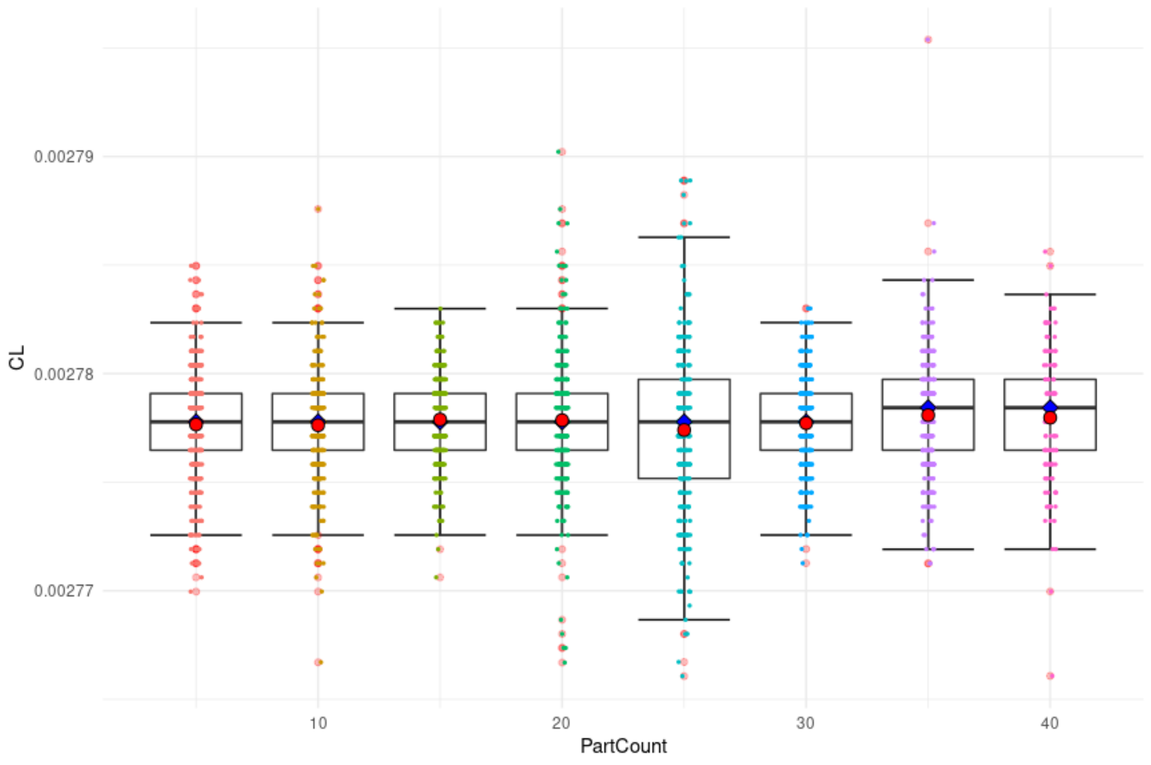
Hide

```
# Feed Rate
g_s1_fr_plot <-
  ggplot(df_g_s1, aes(PartCount, FR, group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
  scale_y_continuous(limits = quantile(df_g_s1$FR, c(0.01, 0.99)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")#+  scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position
= "none")
# +labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 Feed Rate (inch/min)")),
#      title=expression(bold("S1 Power for Finish Splines Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))
g_s1_fr_plot
```



Hide

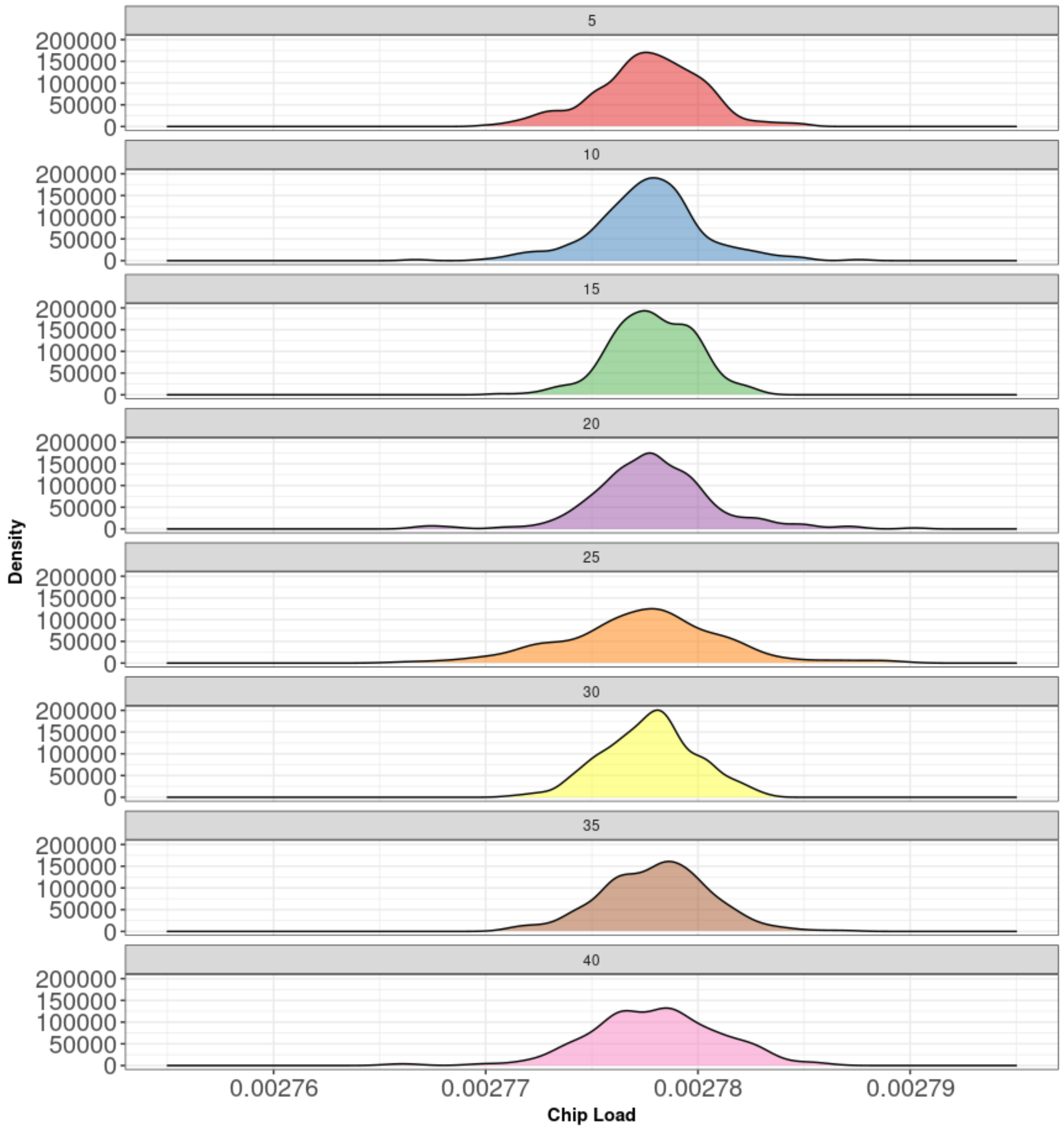
```
# Chip Load
g_s1_cl_plot <-
  ggplot(df_g_s1, aes(PartCount, CL, group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
 # scale_y_continuous(limits = quantile(df_g_s1$FR, c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")#+  scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position
= "none")
# +labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 Feed Rate (inch/min)")),
#      title=expression(bold("S1 Power for Finish Splines Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))
g_s1_cl_plot
```



Hide

```
g_s1_cl_spread <-
  df_g_s1 %>%
  ggplot(aes(x =CL , fill= factor(PartCount))) +
  geom_density(aes(y = ..density..), adjust = 1, alpha = 0.5) +
  theme_bw() +
  facet_wrap(~PartCount, ncol=1) +
  theme(legend.position = "none") +
  scale_fill_brewer(palette = "Set1") + scale_x_continuous(limits = c(0.002755, 0.002795)) +
  labs(x=expression(bold("Chip Load")),
       y=expression(bold("Density")),
       title=expression(bold("Theoretical Chip Load Density Plots for Finish Splines Tool over 1 shift on 11/02/2
2"))) +
  theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
  # scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.
title = element_text(size=15)) +
  guides(color = guide_legend(override.aes = list(size = 3)))
g_s1_cl_spread
```

Theoretical Chip Load Density Plots for Finish Splines Tool over 1



Hide

```

#--------
# S4
#--------

count2 <-c()
for (val in df_g_s4$PartCount){
  if(val<6){
    count2<-append(count2,5)
  }else if(val>=6 & val<11){
    count2<-append(count2,10)
  }else if(val>=11 & val<16){
    count2<-append(count2,15)
  }else if(val>=16 & val<21){
    count2<-append(count2,20)
  }else if(val>=21 & val<26){
    count2<-append(count2,25)
  }else if(val>=26 & val<31){
    count2<-append(count2,30)
  }else if(val>=31 & val<36){
    count2<-append(count2,35)
  }else if(val>=36 & val<41){
    count2<-append(count2,40)
  }else if(val>=41 & val<46){
    count2<-append(count2,45)
  }else if(val>=46 & val<51){
    count2<-append(count2,50)
  }else if(val>=51 & val<56){
    count2<-append(count2,55)
  }else if(val>=56 & val<61){
    count2<-append(count2,60)
  }else if(val>=61 & val<66){
    count2<-append(count2,65)
  }else if(val>=66 & val<71){
    count2<-append(count2,70)
  }else if(val>=71 & val<76){
    count2<-append(count2,75)
  }
}

length(df_g_s4)
```

```
[1] 27
```

Hide

```
unique(df_g_s4$PartCount)
```

```
 [1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
36 37
```

Hide

```
df_g_s4$PartCount <- count2
unique(df_g_s4$PartCount)
```

```
[1]  5 10 15 20 25 30 35 40
```

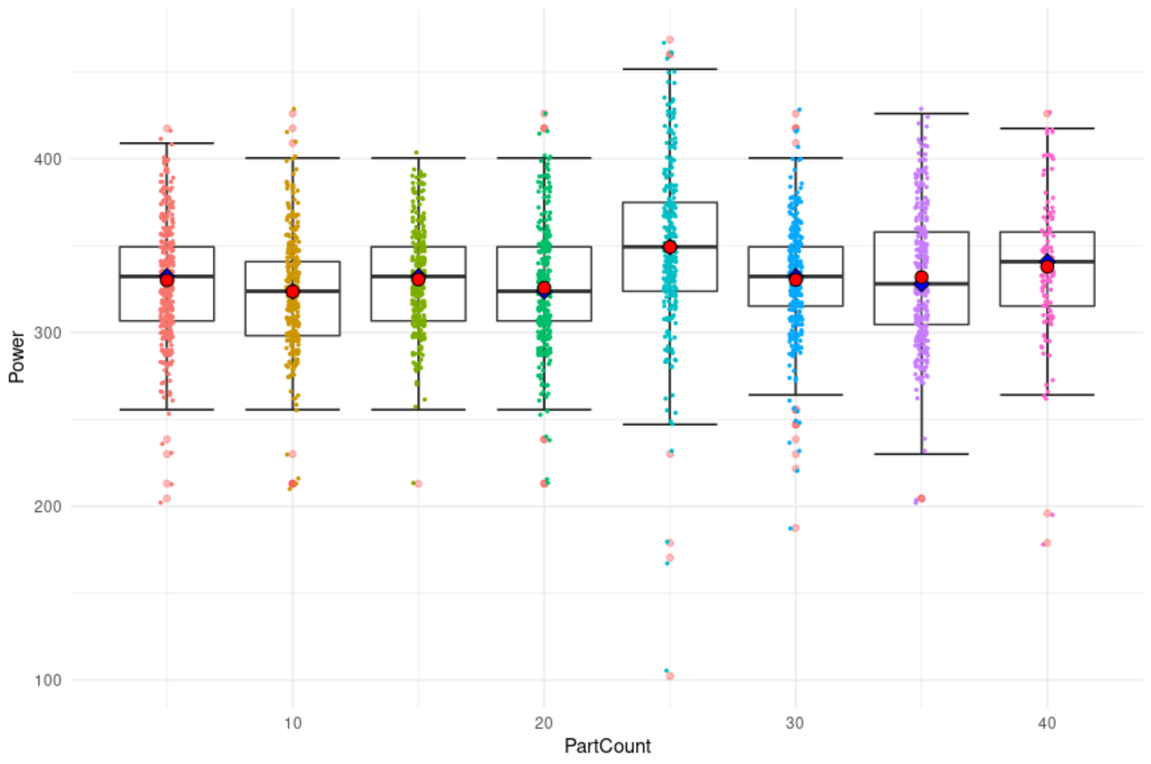
Hide

```
length(df_g_s4$PartCount)
```

```
[1] 2303
```

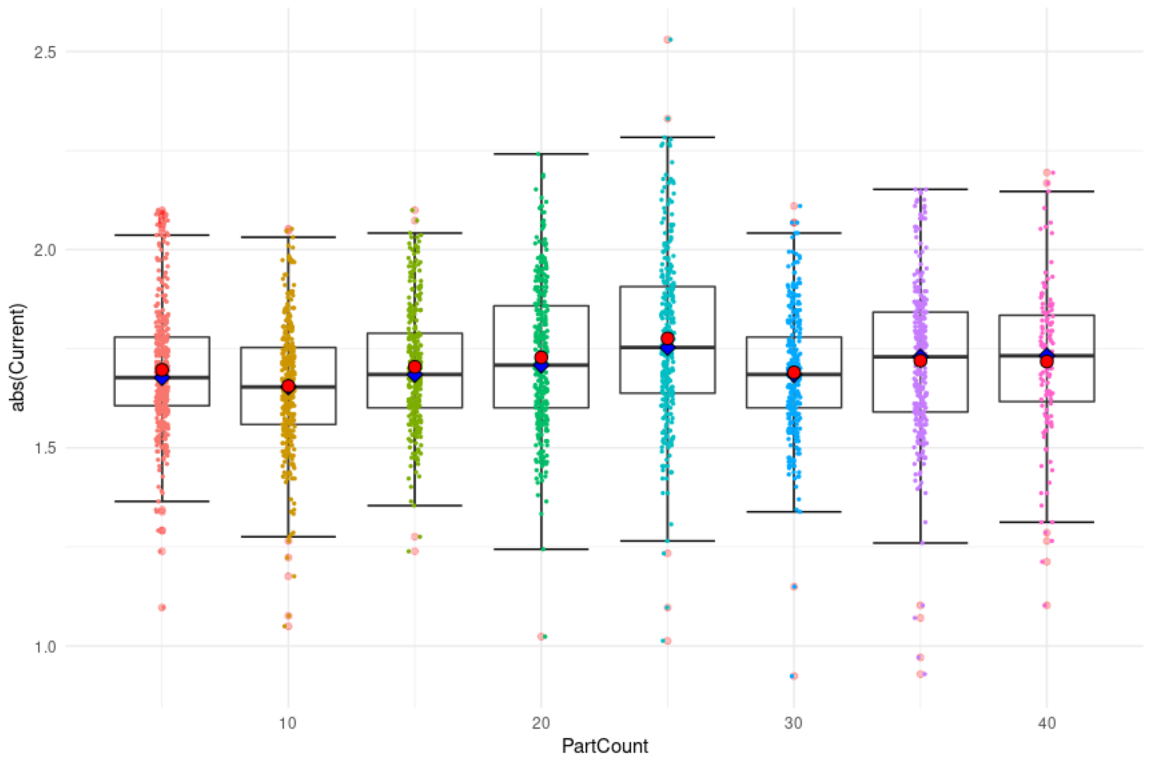
Hide

```
#Power
g_s4_power_plot <-
  ggplot(df_g_s4, aes(PartCount, Power, group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)),  position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
  #scale_y_continuous(limits = quantile(df_g_s4$Power, c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")
  #scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position = "none")
# +labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 Power (W)")),
#      title=expression(bold("S1 Power for Chamdrill 8D Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))
g_s4_power_plot
```



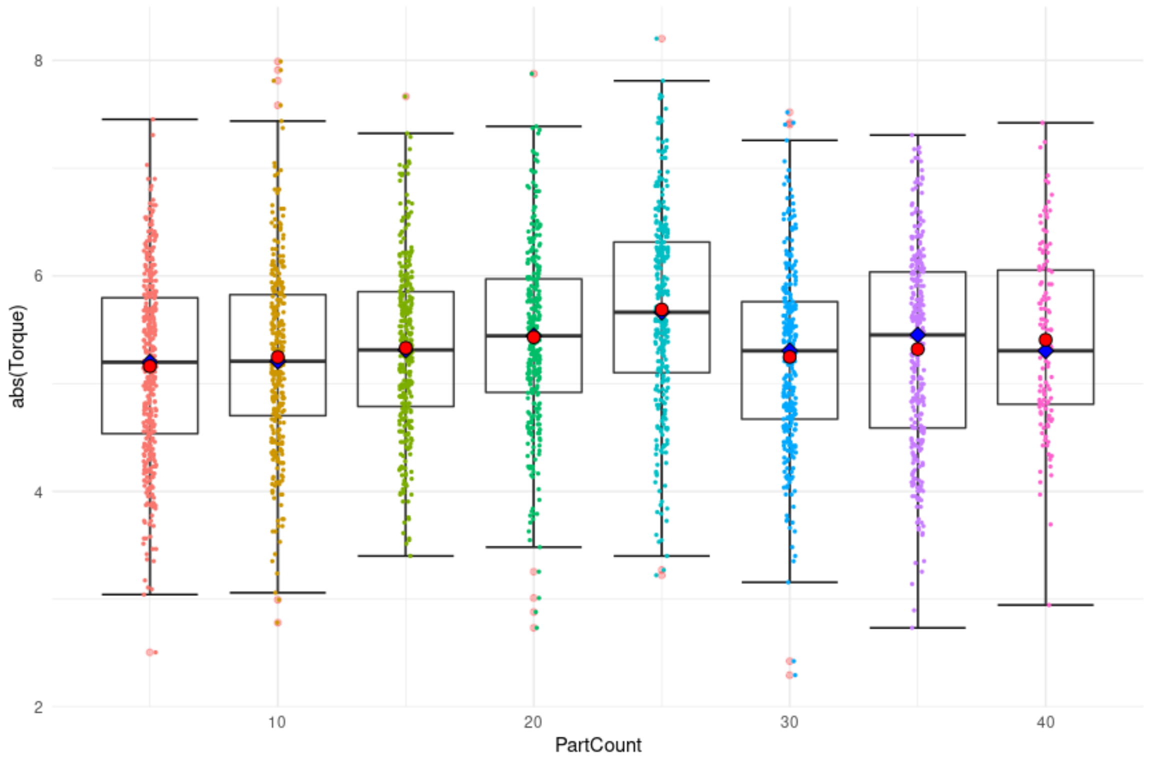
Hide

```
g_s4_current_plot <-
  ggplot(df_g_s4, aes(PartCount, abs(Current), group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
 # scale_y_continuous(limits = quantile(abs(df_g_s4$Current), c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")#  scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position
= "none")
# +labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 Current (A)")),
#      title=expression(bold("S1 Current for Chamdrill 8D Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))

g_s4_current_plot
```



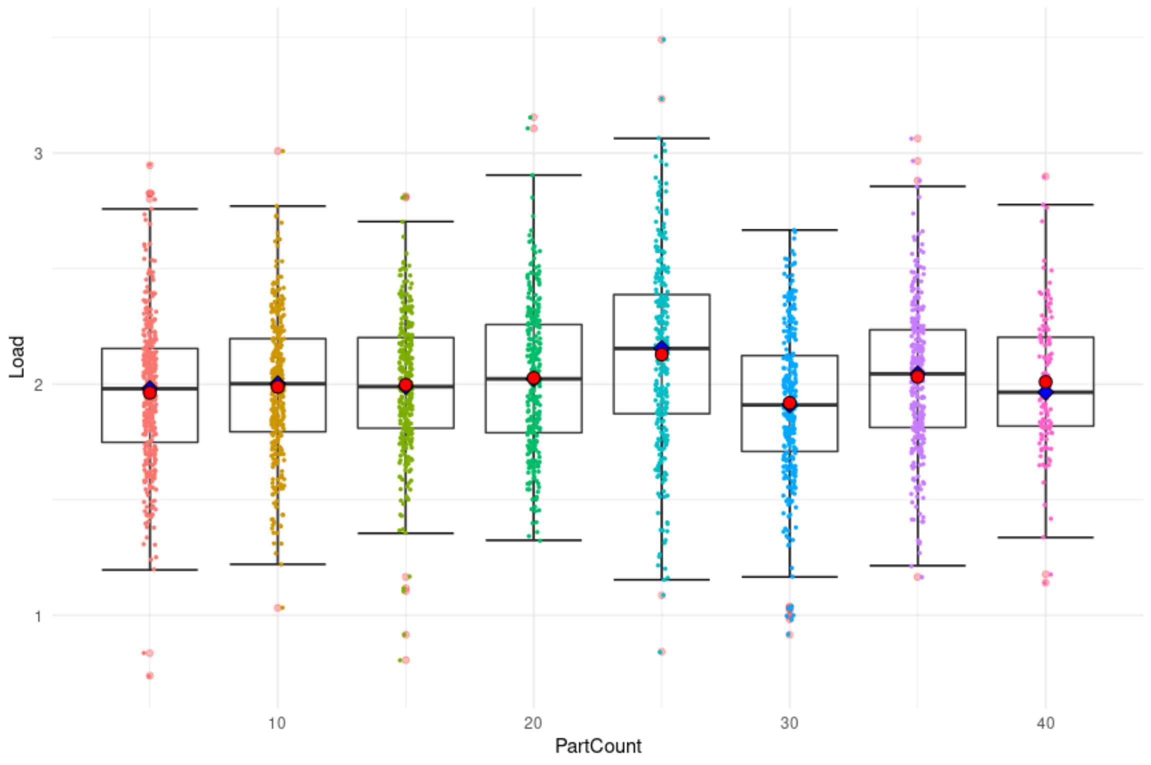
Hide

```
# Torque
g_s4_torque_plot <-
  ggplot(df_g_s4, aes(PartCount, abs(Torque), group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
  #scale_y_continuous(limits = quantile(abs(df_g_s4$Torque), c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")#  scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position
= "none")
# +labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 Torque (Nm)")),
#      title=expression(bold("S1 Torque for Chamdrill 8D Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))
g_s4_torque_plot
```



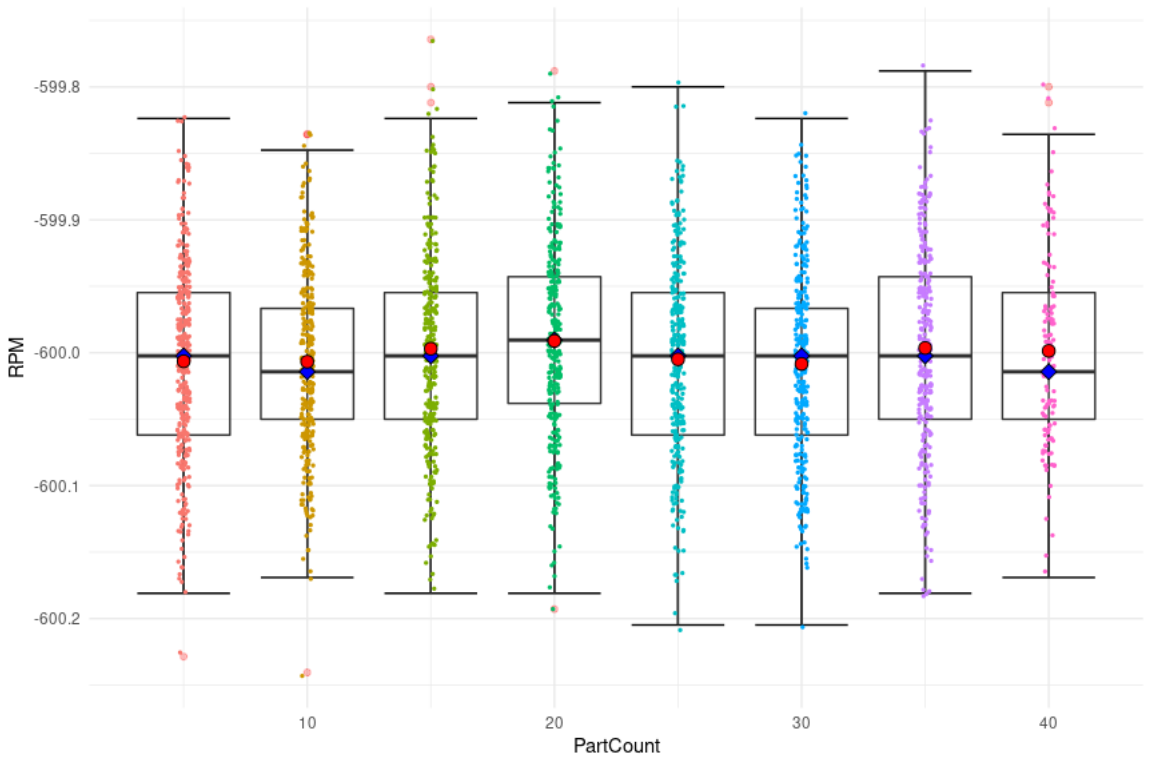
Hide

```
# Load
g_s4_load_plot <-
  ggplot(df_g_s4, aes(PartCount, Load, group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
 # scale_y_continuous(limits = quantile(df_g_s4$Load, c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")#  scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position
= "none")
#+ labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 Load (%)")),
#      title=expression(bold("S1 Load for Chamdrill 8D Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))
g_s4_load_plot
```



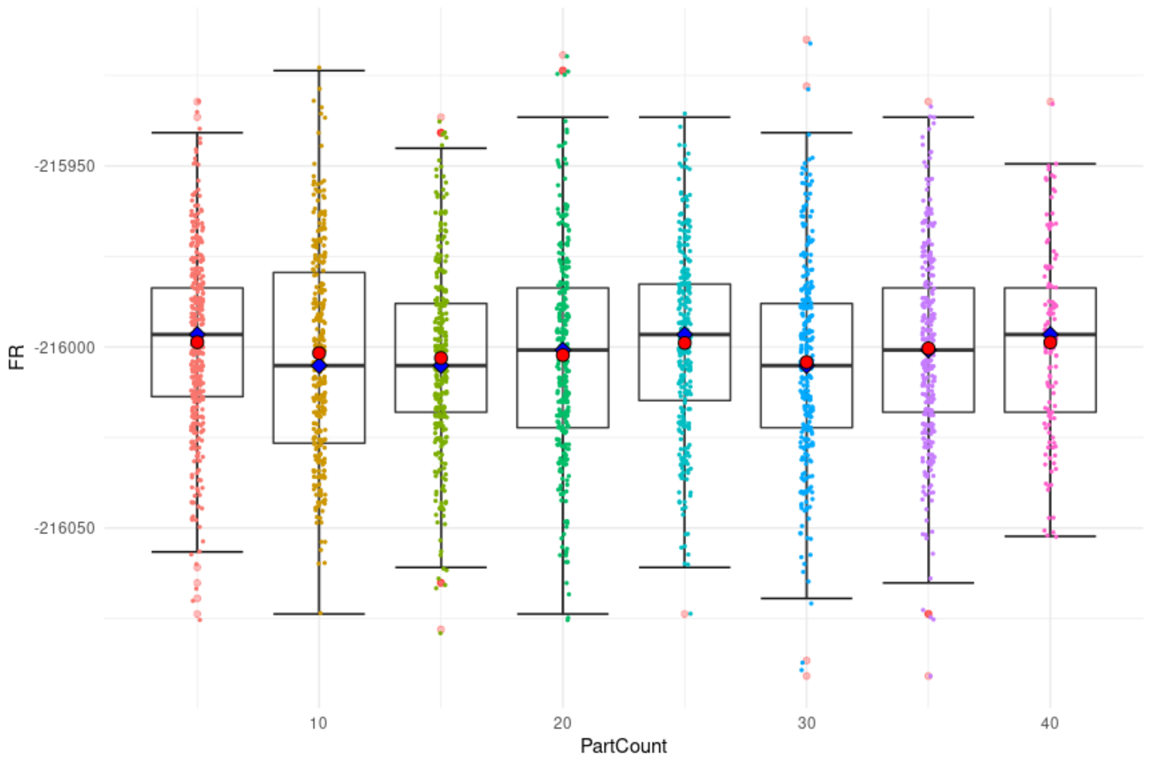
Hide

```
# RPM
g_s4_rpm_plot <-
  ggplot(df_g_s4, aes(PartCount, RPM, group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
  #scale_y_continuous(limits = quantile(df_g_s4$RPM, c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")#  scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position
= "none")
#+labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 RPM")),
#      title=expression(bold("S1 RPM for Chamdrill 8D Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))
g_s4_rpm_plot
```



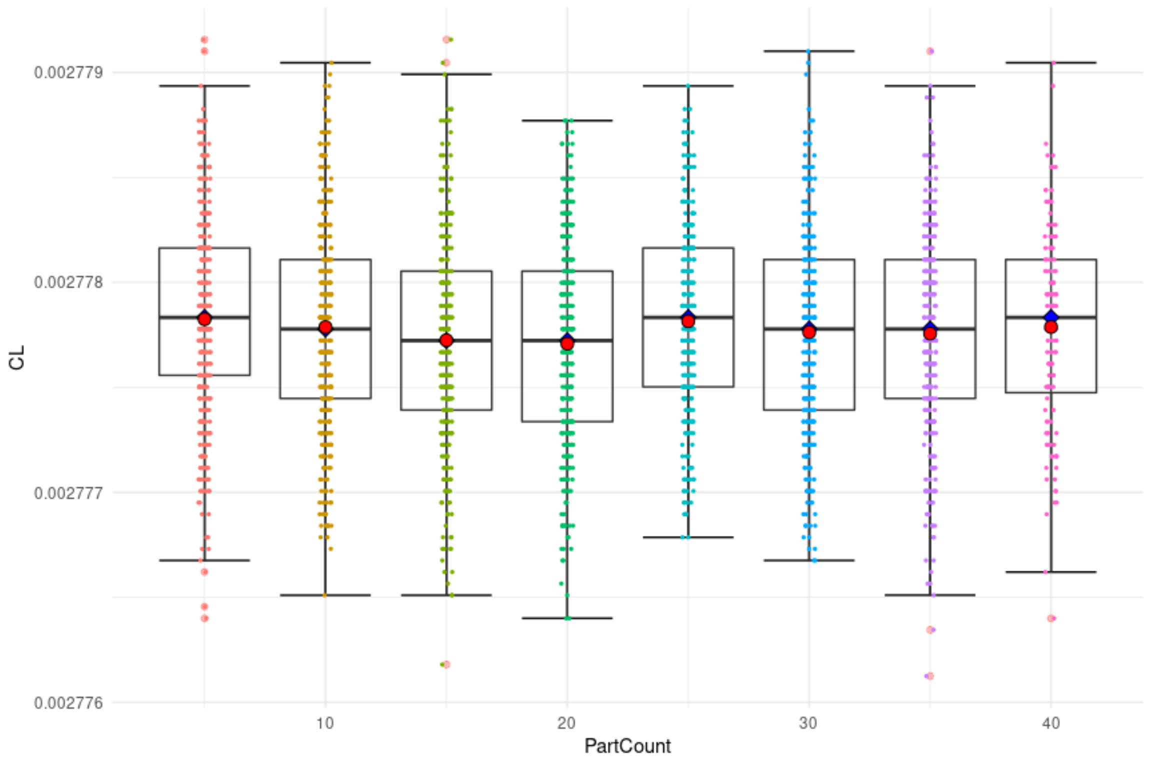
Hide

```
# Feed Rate
g_s4_fr_plot <-
  ggplot(df_g_s4, aes(PartCount, FR, group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
  #scale_y_continuous(limits = quantile(df_g_s4$FR, c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")#  scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position
= "none")
# +labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 Feed Rate (inch/min)")),
#      title=expression(bold("S1 Power for Chamdrill 8D Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))
g_s4_fr_plot
```



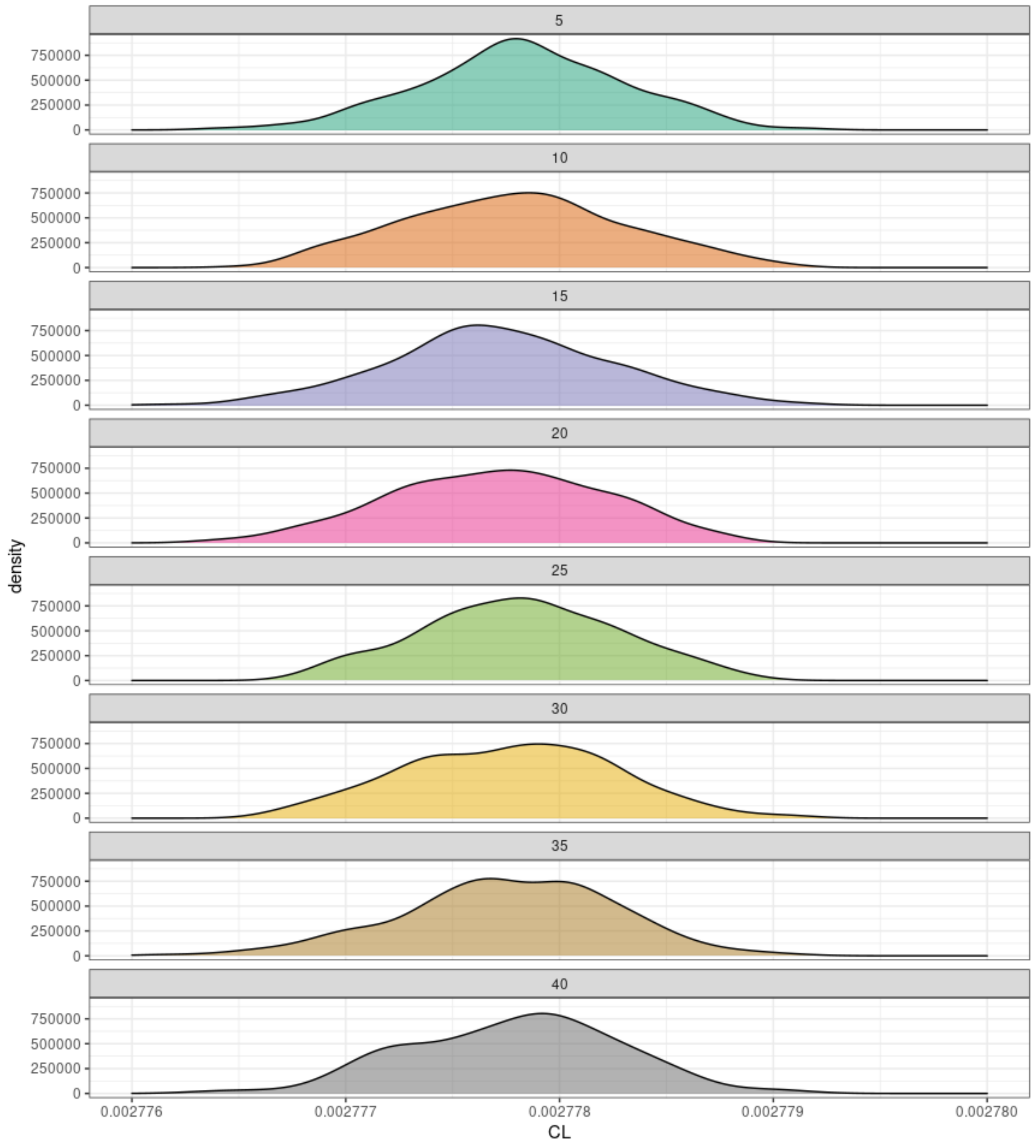
Hide

```
# Chip Load
g_s4_cl_plot <-
  ggplot(df_g_s4, aes(PartCount, CL, group = PartCount)) +
  stat_boxplot(geom ='errorbar')+
  geom_boxplot(alpha = 0.25, outlier.colour = "red")+
  geom_jitter(aes(color=as.factor(PartCount)), position=position_jitter(0.25), size=0.5)+
  stat_summary(fun = "median", geom = "point", shape = 23, size = 3, fill="blue") +
  stat_summary(fun = "mean", geom = "point", shape = 21, size = 3, fill="red",  colour="black" )+
  # scale_y_continuous(limits = quantile(df_g_s1$FR, c(0.1, 0.9)))+
  theme_minimal()+
  theme(legend.position = "none")+
  scale_fill_brewer(palette="Dark2")#  scale_x_discrete(name ="Time", limits=unique(time))+theme(legend.position
= "none")
# +labs(x=expression(bold("Part Count in groups of 10")),
#      y=expression(bold("S1 Feed Rate (inch/min)")),
#      title=expression(bold("S1 Power for Finish Splines Tool over 1 shift on 11/02/22")),
#      subtitle=expression("Mean = red circle; Median = blue diamond")) +
# theme(plot.title=element_text(size=18.), plot.subtitle = element_text(size=14), legend.text = element_text(size
=15), legend.key.size = unit(2,"line"), axis.text = element_text(size=15)) +
# scale_color_discrete(name =expression(underline("Group of 10 Parts"))) +theme(legend.title.align = 1, legend.ti
tle = element_text(size=15)) +
# guides(color = guide_legend(override.aes = list(size = 3)))
g_s4_cl_plot
```



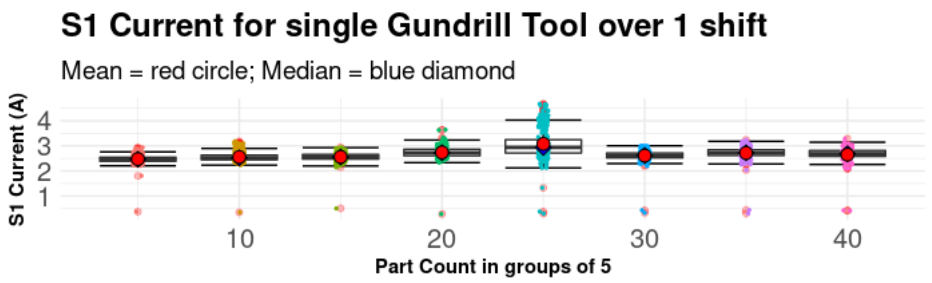
Hide

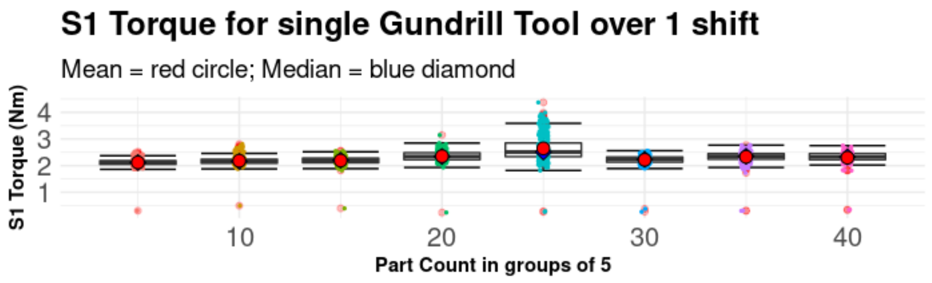
```
g_s4_cl_spread <-
  df_g_s4 %>%
  ggplot(aes(x =CL , fill= factor(PartCount))) +
  geom_density(aes(y = ..density..), adjust = 1, alpha = 0.5) +
  theme_bw() +
  facet_wrap(~PartCount, ncol=1) +
  theme(legend.position = "none") +
  scale_fill_brewer(palette = "Dark2") + scale_x_continuous(limits = c(0.002776, 0.00278))
g_s4_cl_spread
```



Hide

```
test<-ggarrange(g_s1_current_plot, g_s1_torque_plot, ncol=1)
test
```

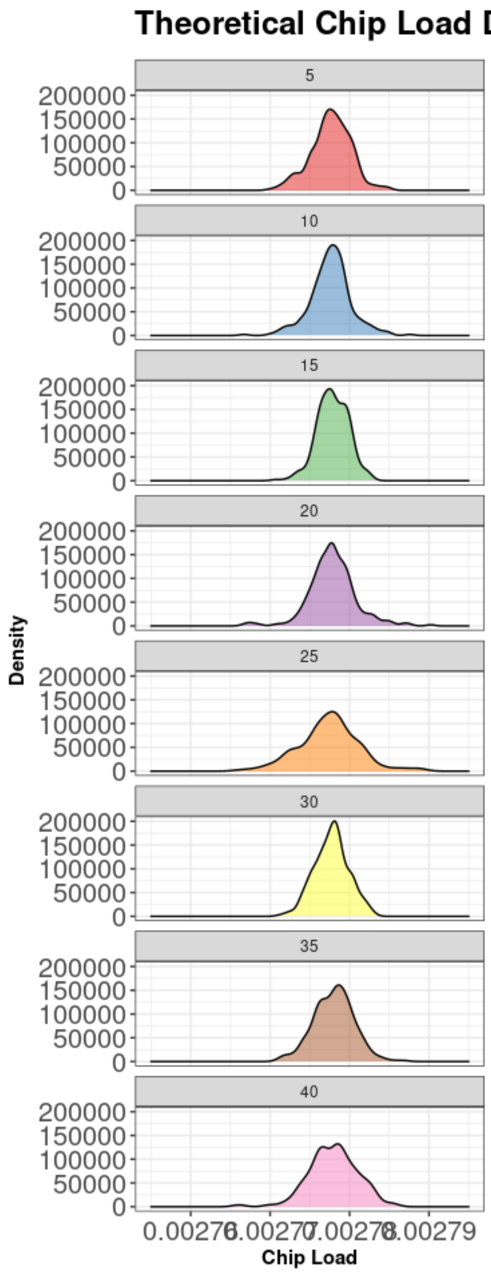


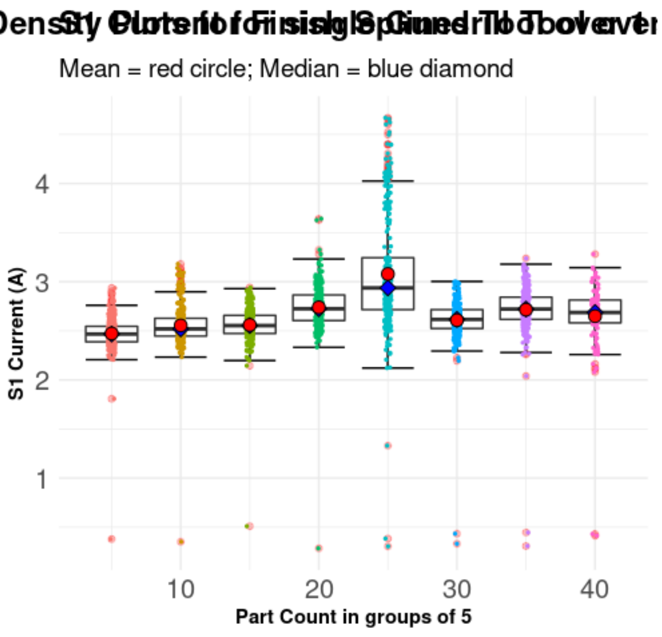


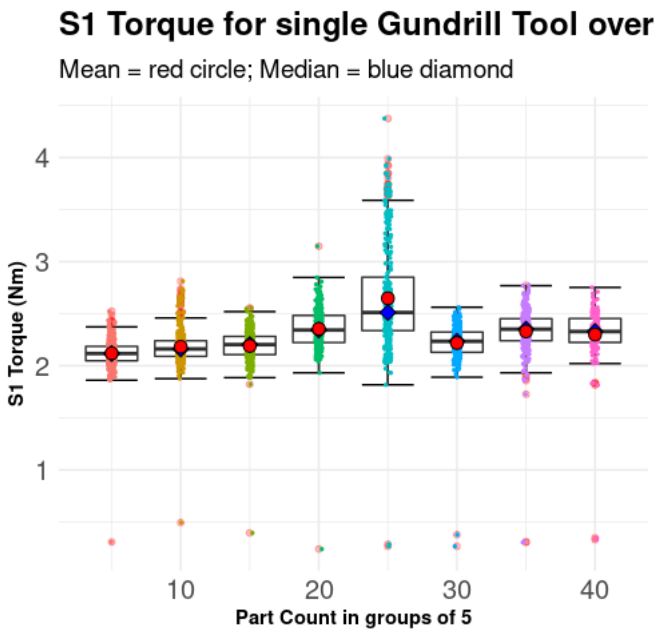
Hide

```
ggarrange(g_s1_cl_spread, test, widths = c(1.5,2))
```

```
Warning: Removed 1 rows containing non-finite values (stat_density).
```



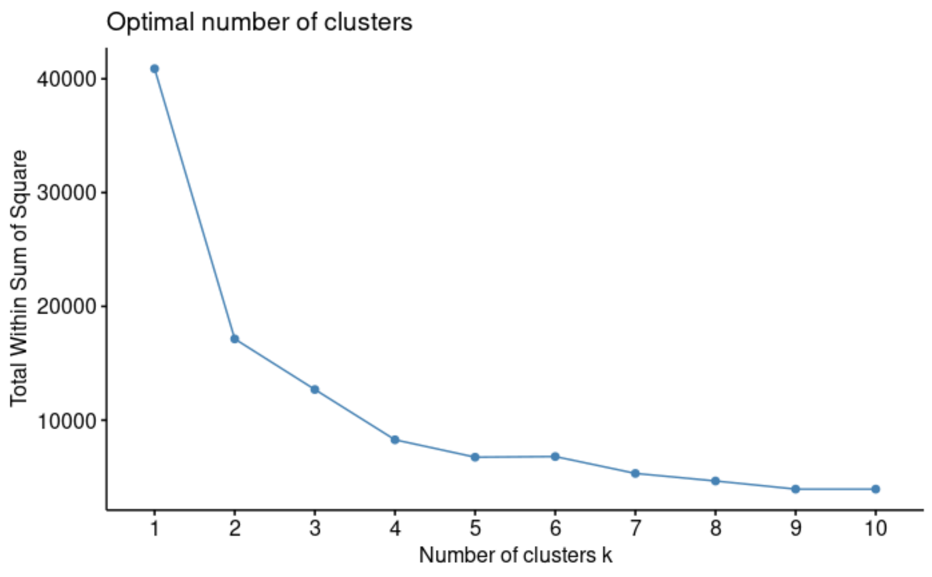




Clustering code

Hide

```
palette <- c("#8250C4", "#db0a16", "#004753",  "#107C10")


#------------- ALL ATTRIBUTES -------------------
my_data = s1_sum_mean
fviz_nbclust(my_data, kmeans, method = "wss")
```



Hide

```
set.seed(123)
km.model <- kmeans(my_data, 4, nstart = 25)
```

fviz_cluster(km.model, data = my_data, ellipse.type = “convex”, palette = palette, ggtheme = theme_minimal())

Hide

```
which(apply(s1_sum_mean, 2, var)==0)
```

```
LineNo_1   DRPM_1    DFR_1    DCL_1 
       7       10       17       18 
```

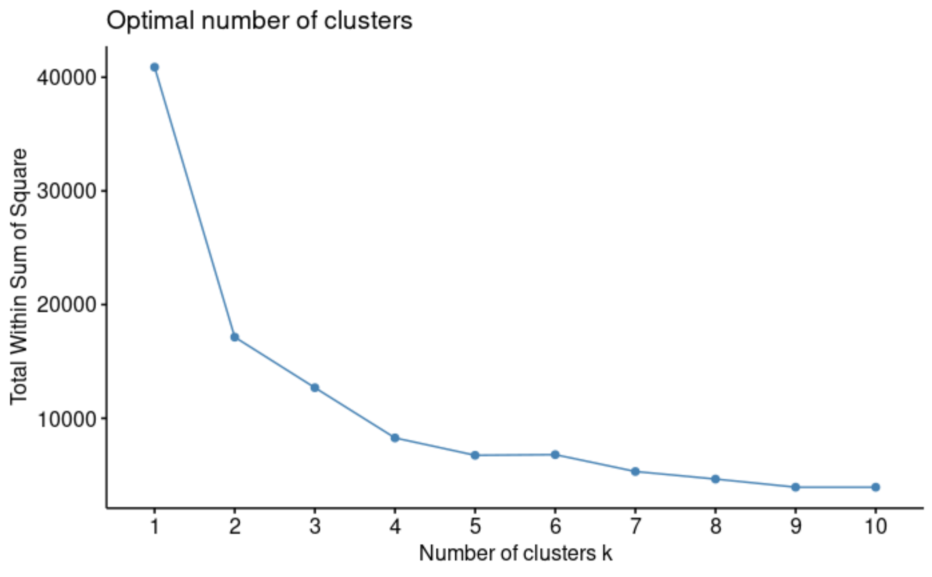
Hide

```
var0sumMeanS1<-s1_sum_mean[ , which(apply(s1_sum_mean, 2, var) != 0)]
rownames(var0sumMeanS1)<-0:(nrow(var0sumMeanS1)-1)
```

```
Warning: Setting row names on a tibble is deprecated.
```

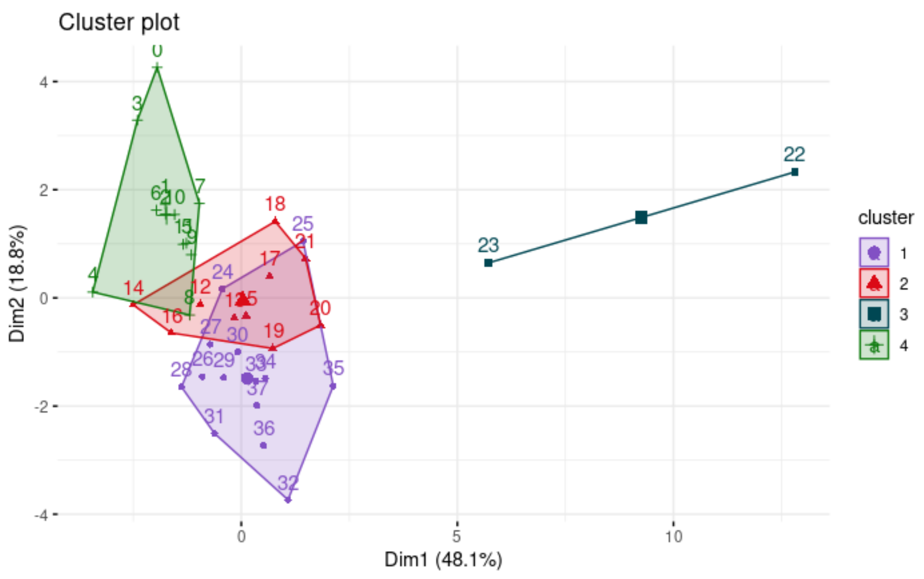
Hide

```
fviz_nbclust(var0sumMeanS1, kmeans, method = "wss")
```



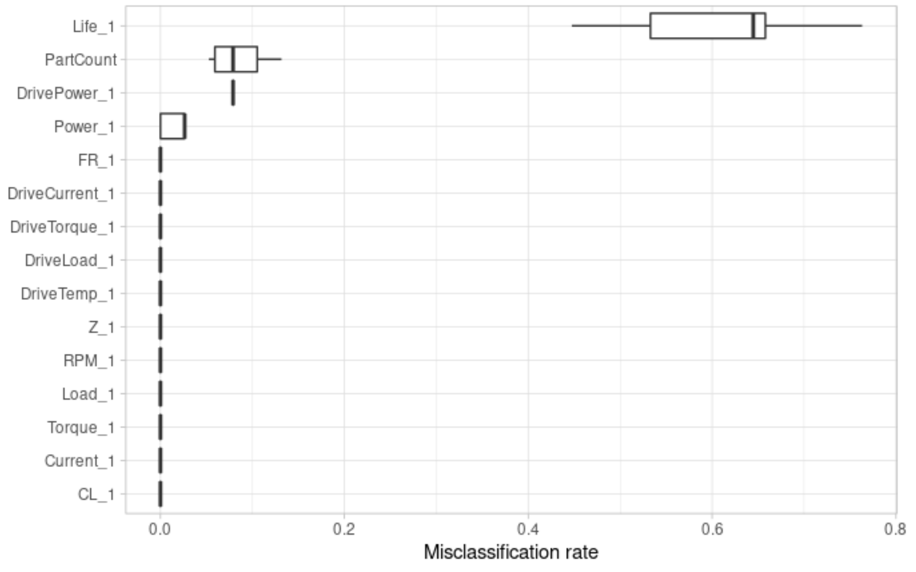
Hide

```
km.model <- kmeans(var0sumMeanS1, 4, nstart = 25)

fviz_cluster(km.model, data = var0sumMeanS1,
             ellipse.type = "convex",
             palette = palette,
             ggtheme = theme_minimal())
```



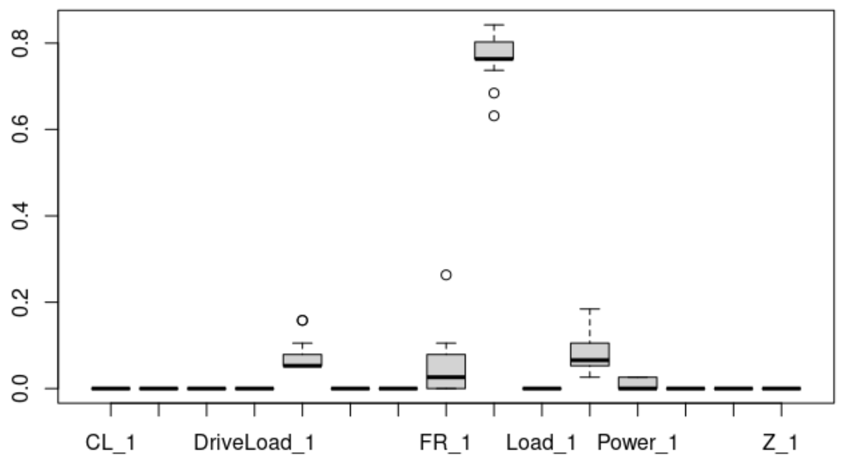
Hide

```
set.seed(001)
res <- kcca(var0sumMeanS1,k=4)
set.seed(001)
FeatureImp_res <- FeatureImpCluster(res,as.data.table(var0sumMeanS1))
plot(FeatureImp_res)
```



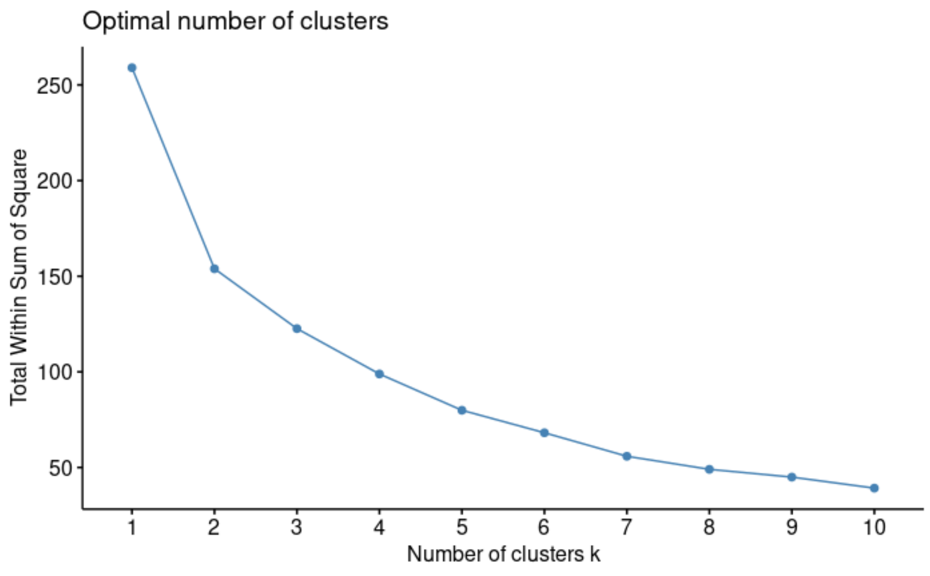
Hide

```
set.seed(12)
nr_seeds <- 20
seeds_vec <- sample(1:1000,nr_seeds)

savedImp <- data.frame(matrix(0,nr_seeds,dim(var0sumMeanS1)[2]))
count <- 1
for (s in seeds_vec) {
  set.seed(s)
  res <- kcca(var0sumMeanS1,k=4)
  set.seed(s)
  FeatureImp_res <- FeatureImpCluster(res,as.data.table(var0sumMeanS1),sub = 1,biter = 1)
  savedImp[count,] <- FeatureImp_res$featureImp[sort(names(FeatureImp_res$featureImp))]
  count <- count + 1
}
names(savedImp) <- sort(names(FeatureImp_res$featureImp))
boxplot(savedImp)
```



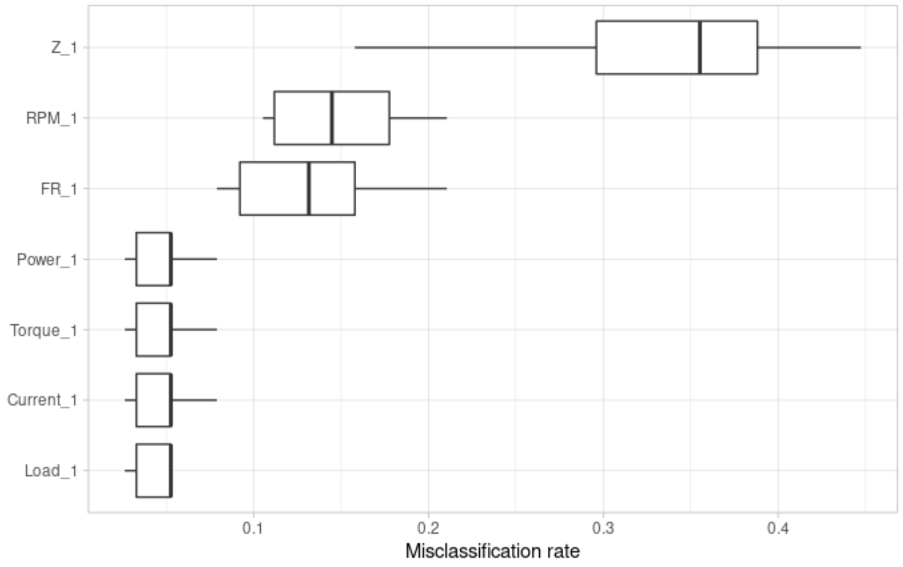
Hide

```
#-------------ATTEMPTING JUST S1 ATTRIBUTES -------------------------------
s1_sum <-s1_sum_mean[,9:length(s1_sum_mean)]
s1_sum_scale<-subset(s1_sum, select = -c(DRPM_1, DFR_1,DCL_1, CL_1)) %>% scale()
rownames(s1_sum_scale)<-0:(nrow(s1_sum_scale)-1)

fviz_nbclust(s1_sum_scale, kmeans, method = "wss")
```



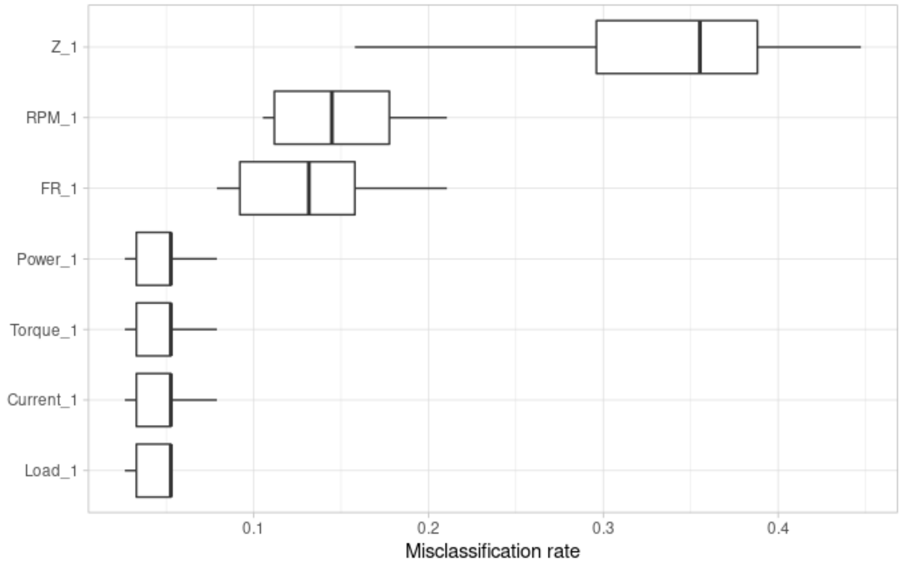
Hide

```
set.seed(002)
res2 <- kcca(s1_sum_scale,k=4)
set.seed(002)
FeatureImp_res2 <- FeatureImpCluster(res2,as.data.table(s1_sum_scale))
plot(FeatureImp_res2)
```



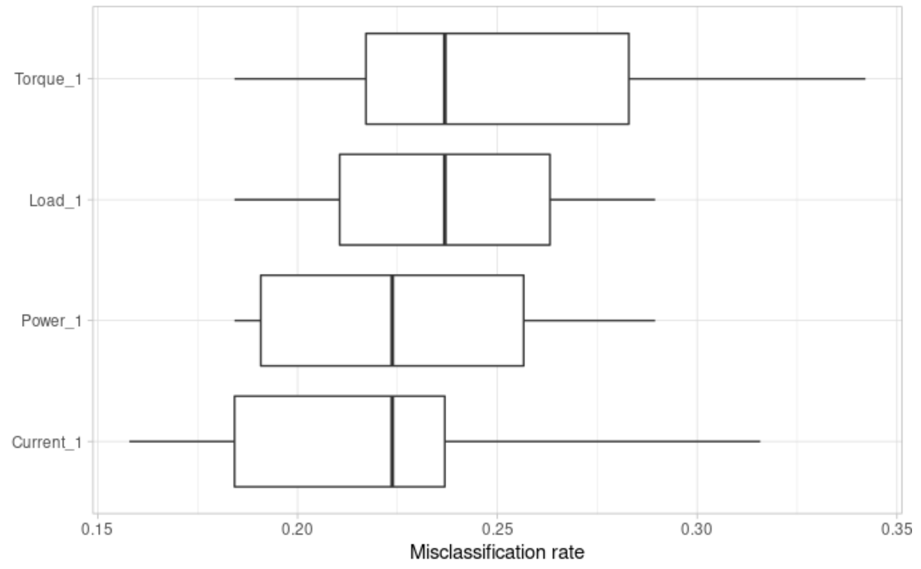
Hide

```
feature_importance_S1<-FeatureImp_res2
plot(feature_importance_S1)
```



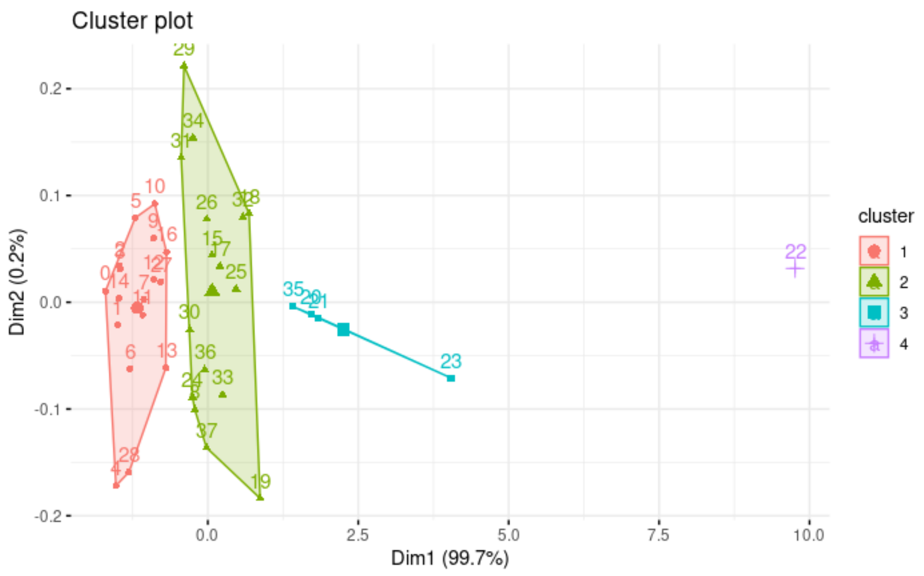
Hide

```
s1_clus_ <-s1_sum_mean[,9:length(s1_sum_mean)]
s1_clus<-subset(s1_clus_, select = -c(DRPM_1, DFR_1,DCL_1, CL_1, Z_1, RPM_1, FR_1)) %>% scale()
s1_clus<-data.frame(s1_clus)
rownames(s1_clus)<-0:(nrow(s1_clus)-1)

set.seed(00)
res <- kcca(s1_clus,k=4)
set.seed(00)
FeatureImp_res <- FeatureImpCluster(res,as.data.table(s1_clus))
plot(FeatureImp_res)
```



Hide

```
#feature_importance_S1_opt<-FeatureImp_res
#plot(feature_importance_S1_opt)

#set.seed(42)
#model4 <- kmeans(s1_clus, 4, nstart = 25)
clus_model<-readRDS("kmeansModel.rds")
fviz_cluster(clus_model, data = s1_clus,
             ellipse.type = "convex",
             ggtheme = theme_minimal())
```



Hide

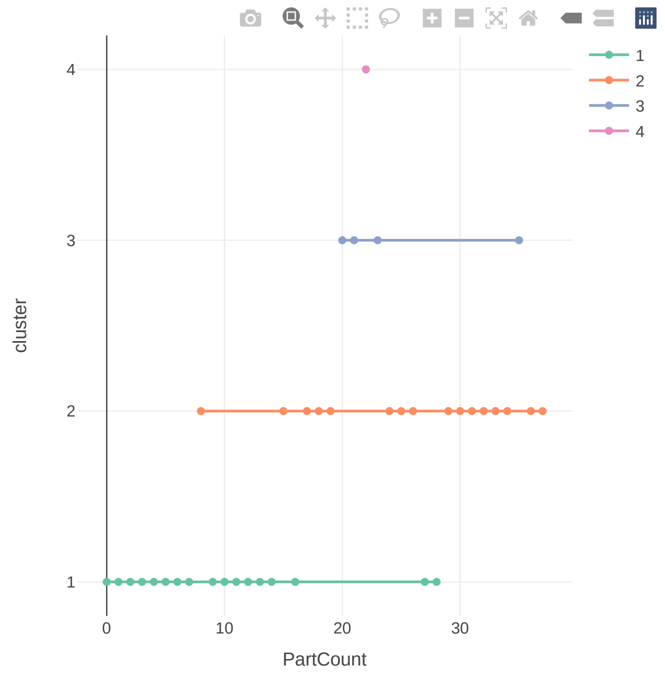
```
#saveRDS(model4, "kmeansModel.rds")


s1_clus %>%
  mutate(Cluster = clus_model$cluster) %>%
  group_by(Cluster) %>%
  summarise_all("mean")
```

Cluster <int>	Load_1 <dbl>	Power_1 <dbl>	Torque_1 <dbl>	Current_1 <dbl>
1	-0.57351616	-0.58822403	0.6018886	0.58224200
2	0.02107647	0.04601706	-0.0479738	-0.03127911
3	1.13328585	1.09142778	-1.1598273	-1.11966380
4	4.87940784	4.89782454	-4.8252160	-4.91899294

4 rows

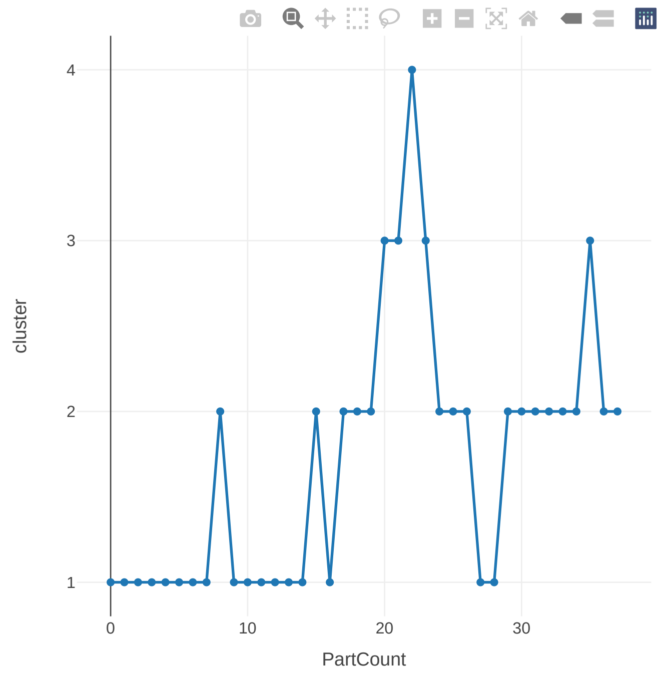
Hide

```
df<-setNames(data.frame(cbind(g_S4_curr$PartCount, as.numeric(clus_model$cluster))), c("PartCount", "cluster"))
df$cluster<-as.factor(df$cluster)
plot_ly(df, x=~PartCount, y=~cluster, color=~cluster, text=~PartCount, type='scatter', mode='lines+markers')
```



Hide

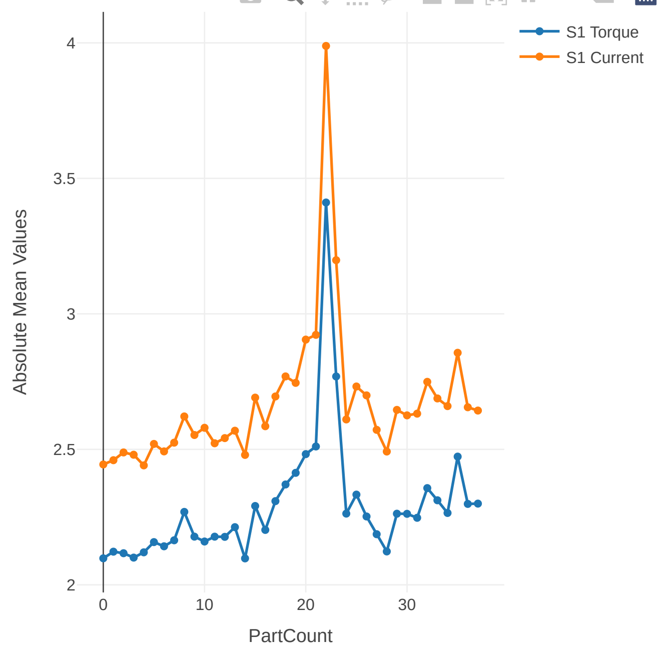
```
plot_ly(df, x=~PartCount, y=~cluster, type='scatter', mode='lines+markers')
```



Hide

```
# Comparing feature importance classification with the plots

plot_ly(summarised_S1_mean, x = ~PartCount, y = ~abs(S1Torque.mean),
        type = 'scatter',
        mode = 'lines+markers',
        name = "S1 Torque") %>%
  add_trace(y = ~abs(S1Curr.mean),
            type = 'scatter',
            mode = 'lines+markers',
            name = 'S1 Current')  %>%
  layout(showlegend = TRUE,
         yaxis = list(title = "Absolute Mean Values"),
         title = "Comparison of Mean Torque and Current values over Gundrill Life")
```

Comparison of Mean Torque and Current values over Gundrill Life



Hide

```
#-------------------trial on new data-------------------
data1<- read.csv("~/onedrive/Scenic/data/indexG220/ml/s1_sum_mean_1.csv", comment.char="#", stringsAsFactors=FALS
E)
data2<- read.csv("~/onedrive/Scenic/data/indexG220/ml/s1_sum_mean_2.csv", comment.char="#", stringsAsFactors=FALS
E)
data3<- read.csv("~/onedrive/Scenic/data/indexG220/ml/s1_sum_mean_3.csv", comment.char="#", stringsAsFactors=FALS
E)
data4<- read.csv("~/onedrive/Scenic/data/indexG220/ml/s1_sum_mean_4.csv", comment.char="#", stringsAsFactors=FALS
E)

head(data1)
```

	PartCount <int>	DrivePower_1 <dbl>	DriveCurrent_1 <dbl>	DriveTorque_1 <dbl>	DriveLoad_1 <dbl>	DriveTemp_1 <dbl>
1	0	336.9654	-1.761485	-5.272017	2.043660	28.21430
2	1	333.3968	-1.739585	-5.242590	1.943219	31.22598
3	2	334.8904	-1.703647	-5.073129	2.045367	31.64973
4	3	322.9480	-1.656795	-5.170905	1.916698	32.13634
5	4	310.4115	-1.611363	-4.800500	1.941833	32.02482
6	5	323.4846	-1.696997	-5.367629	2.047926	32.43523

6 rows | 1-7 of 19 columns

Hide

```
head(data2)
```

	PartCount <int>	DrivePower_1 <dbl>	DriveCurrent_1 <dbl>	DriveTorque_1 <dbl>	DriveLoad_1 <dbl>	DriveTemp_1 <dbl>
1	38	315.2395	-1.674508	-5.127333	1.871541	32.38667
2	39	306.4307	-1.578799	-5.259966	1.962022	32.46150
3	40	309.8964	-1.618745	-5.003254	1.940297	32.27332
4	41	303.1566	-1.552569	-5.104800	1.849476	32.36652
5	42	309.4175	-1.568303	-5.008200	1.878967	32.23359
6	43	317.1633	-1.628944	-4.921565	1.967301	32.21624

6 rows | 1-7 of 19 columns

Hide

```
head(data3)
```

	PartCount <int>	DrivePower_1 <dbl>	DriveCurrent_1 <dbl>	DriveTorque_1 <dbl>	DriveLoad_1 <dbl>	DriveTemp_1 <dbl>
1	0	326.4132	-1.674697	-5.379623	2.131027	30.62203
2	1	338.7044	-1.737943	-5.314230	2.018062	31.96370
3	2	334.4781	-1.713298	-5.157097	1.868266	32.19314
4	3	328.4928	-1.722575	-5.415111	2.006507	32.31685
5	4	310.6303	-1.654905	-5.099852	1.842061	32.05936
6	5	313.3461	-1.634696	-4.945143	1.893931	32.29702

6 rows | 1-7 of 19 columns

Hide

```
head(data4)
```

	PartCount <int>	DrivePower_1 <dbl>	DriveCurrent_1 <dbl>	DriveTorque_1 <dbl>	DriveLoad_1 <dbl>	DriveTemp_1 <dbl>
1	38	340.6525	-1.748558	-5.205259	2.010267	32.34756
2	39	318.6774	-1.620291	-4.909158	1.900121	32.26306
3	40	316.2335	-1.672514	-5.084567	1.951294	32.45467
4	41	324.5975	-1.640846	-5.085328	1.928411	32.35102
5	42	325.5752	-1.641872	-5.281246	2.044578	32.41111
6	43	310.4115	-1.650937	-5.302817	1.974487	32.30569

6 rows | 1-7 of 19 columns

Hide

```
PartCount1 <- 0:37
PartCount2 <- 0:35

# function to extract the clusters from applied model by calculating the Euclidean distance between the value and
the model centres for each cluster; then returning the cluster number it's closest to.
closest.cluster <- function(x) {
  cluster.dist <- apply(clus_model$centers, 1, function(y) sqrt(sum((x-y)^2)))
  return(which.min(cluster.dist)[1])
}


#preparing data
d1 <-data1[,9:length(data1)]
df1 <-subset(d1, select = -c(DRPM_1, DFR_1,DCL_1, CL_1, Z_1, RPM_1, FR_1)) %>% scale()
rownames(df1)<-0:(nrow(df1)-1)
#compare with s1_clus (dataset model is based on)
head(df1)
```

```
      Load_1    Power_1  Torque_1 Current_1
0 -0.8700551 -0.8656570 0.8227296 0.8441072
1 -0.7514680 -0.7440199 0.7175787 0.7853496
2 -0.7497992 -0.7708061 0.7419612 0.6787186
3 -0.7187172 -0.6764139 0.8116609 0.7103005
4 -0.6384515 -0.8311995 0.7278738 0.8566793
5 -0.6835860 -0.6040759 0.5655159 0.5614619
```

Hide

```
head(s1_clus)
```

	Load_1 <dbl>	Power_1 <dbl>	Torque_1 <dbl>	Current_1 <dbl>
0	-0.8700551	-0.8656570	0.8227296	0.8441072
1	-0.7514680	-0.7440199	0.7175787	0.7853496
2	-0.7497992	-0.7708061	0.7419612	0.6787186
3	-0.7187172	-0.6764139	0.8116609	0.7103005
4	-0.6384515	-0.8311995	0.7278738	0.8566793
5	-0.6835860	-0.6040759	0.5655159	0.5614619

6 rows

Hide

```
#apply model to new data and get clusters
new_clusters1 <- apply(df1, 1, closest.cluster)
new_clusters1
```

```
 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37
 1  1  1  1  1  1  1  1  2  1  1  1  1  1  1  2  1  2  2  2  3  3  4  3  2  2  2  1  1  2  2  2  2  2  2  3  2  2
```

Hide

```
# validation (comparison of model application between s1_clus and df1 -> should be 0 since datasets are identica
l)
setdiff(matrix(clus_model$cluster), matrix(new_clusters1))
```

```
integer(0)
```

Hide

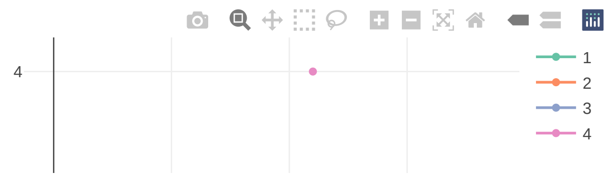
```
#prepping for plotting
clus_1<-setNames(data.frame(cbind(PartCount1, as.numeric(new_clusters1))), c("PartCount", "cluster"))
clus_1$cluster<-as.factor(clus_1$cluster)
head(clus_1)
```

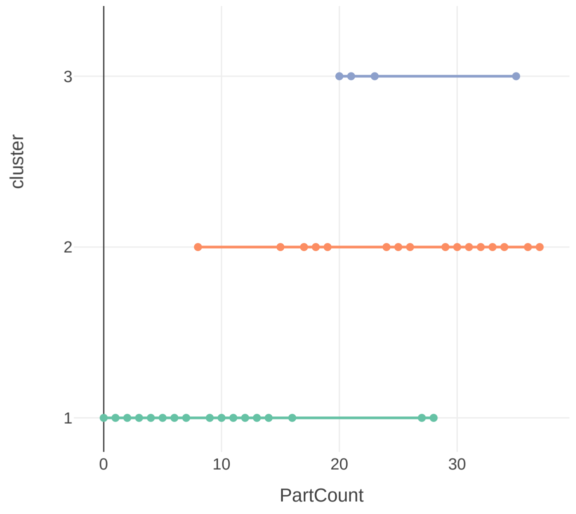
	PartCount <dbl>	cluster <fctr>
1	0	1
2	1	1
3	2	1
4	3	1
5	4	1
6	5	1

6 rows

Hide

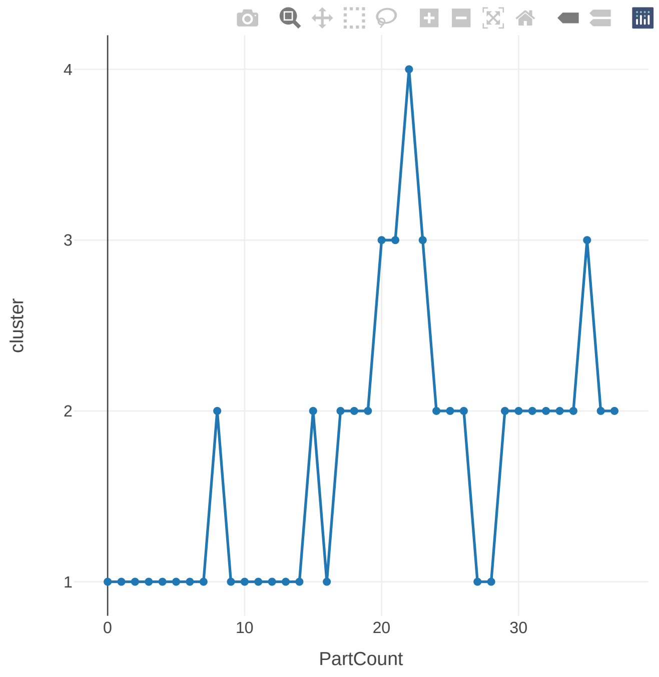
```
plot_ly(clus_1, x=~PartCount, y=~cluster, color=~cluster, type='scatter', mode='lines+markers')
```





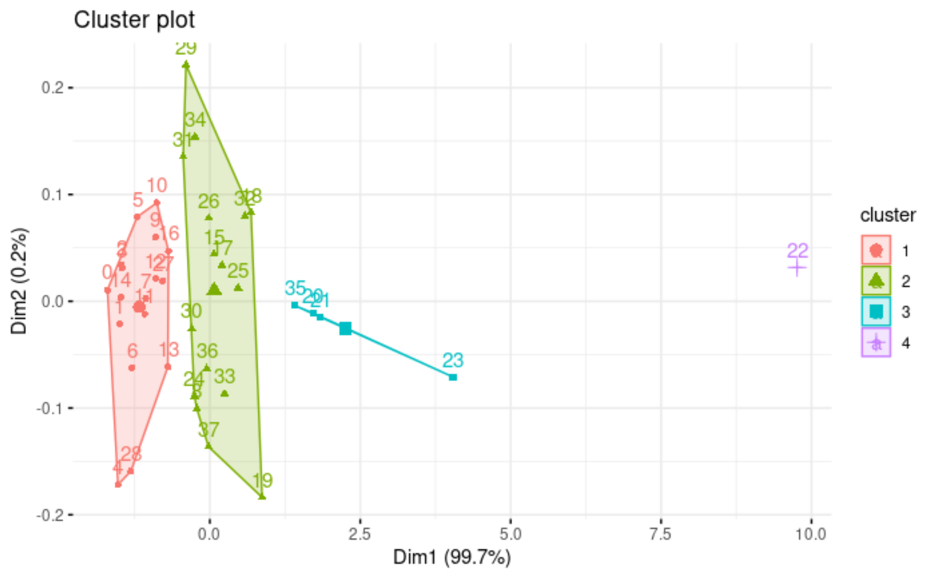
Hide

```
plot_ly(clus_1, x=~PartCount, y=~cluster, type='scatter', mode='lines+markers')
```



Hide

```
#visualising the clusters in a cluster plot
fviz_cluster(clus_model, data = df1,ellipse.type = "convex",ggtheme = theme_minimal())
```



Hide

```
d2 <-data2[,9:length(data2)]
df2 <-subset(d2, select = -c(DRPM_1, DFR_1,DCL_1, CL_1, Z_1, RPM_1, FR_1)) %>% scale()
rownames(df2)<-0:(nrow(df2)-1)
head(df2)
```

```
      Load_1    Power_1  Torque_1 Current_1
0 -0.6874419 -0.6757877 0.6569155 0.6819760
1 -0.8345258 -1.0173179 0.9137302 0.9741501
2 -0.7451944 -0.7465010 0.7034669 0.6820374
3 -0.8849739 -0.9194955 0.9678428 0.9760790
4 -0.8488824 -0.8000625 0.6612001 0.7438957
5 -0.8093221 -0.9191832 0.9984777 0.9648147
```

Hide

```
new_clusters2 <- apply(df2, 1, closest.cluster)
new_clusters2
```

```
 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
 1  1  1  1  1  1  1  1  1  1  2  1  2  1  1  2  2  2  2  2  2  2  2  3  3  2  2  1  2  2  2  2  1  2  4  4
```

Hide

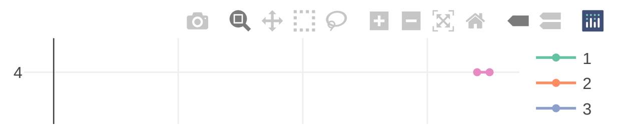
```
clus_2<-setNames(data.frame(cbind(PartCount2, as.numeric(new_clusters2))), c("PartCount", "cluster"))
clus_2$cluster<-as.factor(clus_2$cluster)
tail(clus_2)
```

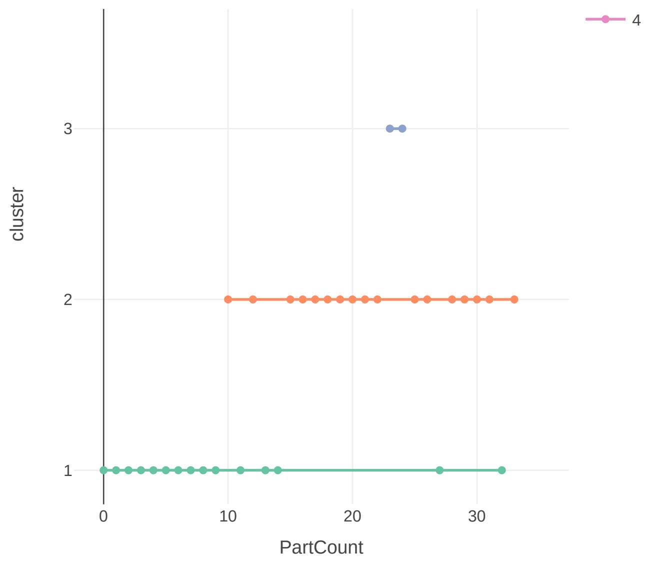
	PartCount <dbl>	cluster <fctr>
31	30	2
32	31	2
33	32	1
34	33	2
35	34	4
36	35	4

6 rows

Hide

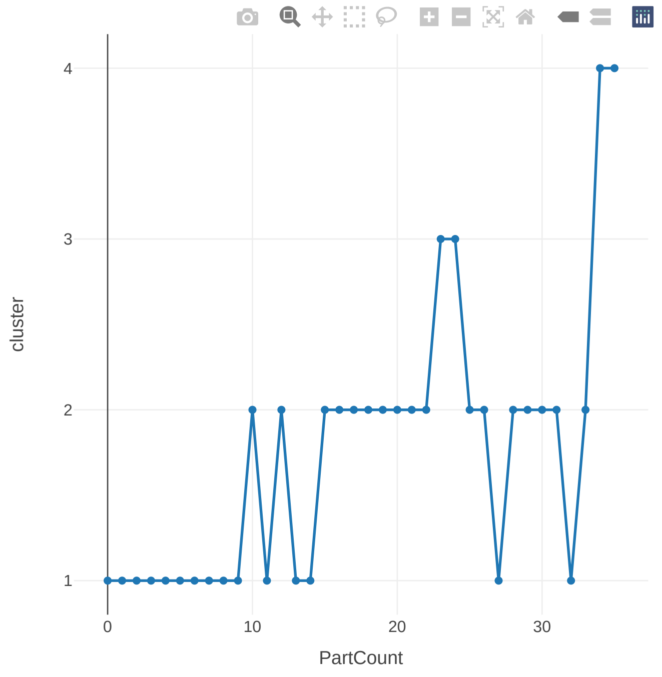
```
plot_ly(clus_2, x=~PartCount, y=~cluster, color=~cluster, type='scatter', mode='lines+markers')
```





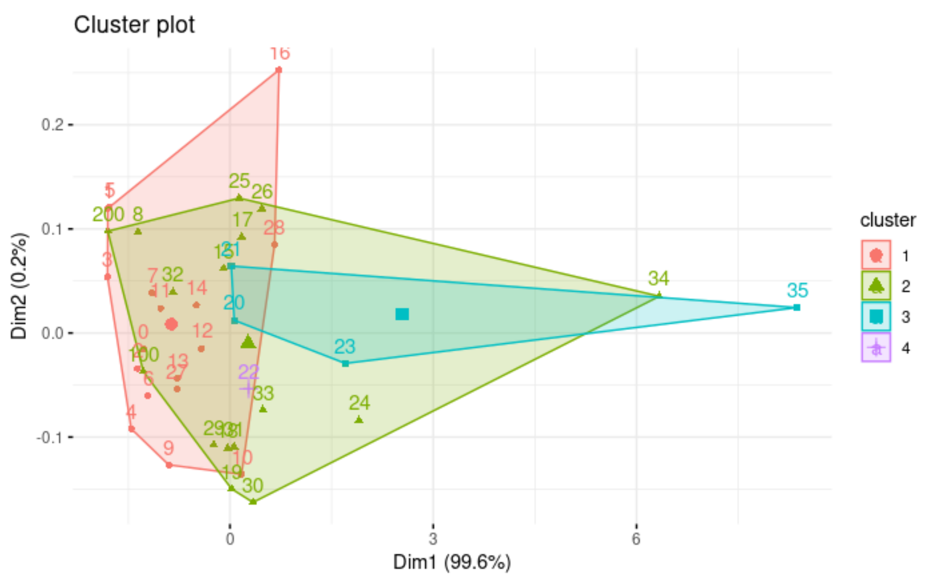
Hide

```
plot_ly(clus_2, x=~PartCount, y=~cluster, type='scatter', mode='lines+markers')
```



Hide

```
#dataset is too short for original model, reshaping by 'making up' 2 new rows.
append1<-df2[1:2,]
append<-append1-0.02
rownames(append)<-c('100', '200') #rownames will be easy to spot and know they're the dud ones
df2_2 <-rbind(df2, append)
fviz_cluster(clus_model, data = df2_2, ellipse.type = "convex", ggtheme = theme_minimal())
```



Hide

```
d3 <-data3[,9:length(data3)]
df3 <-subset(d3, select = -c(DRPM_1, DFR_1,DCL_1, CL_1, Z_1, RPM_1, FR_1)) %>% scale()
rownames(df3)<-0:(nrow(df3)-1)
head(d3)
```

	Z_1 <dbl>	DRPM_1 <int>	RPM_1 <dbl>	Load_1 <dbl>	Power_1 <dbl>	Torque_1 <dbl>	Current_1 <dbl>	FR_1 <dbl>	DFR_1 <int>
1	-101.3089	300	300.0136	5.586818	67.93373	-2.258770	-2.621544	108009.0	108000
2	-101.0019	300	300.0113	5.540191	67.68024	-2.252820	-2.612849	108004.1	108000
3	-100.9899	300	300.0124	5.495625	66.92115	-2.225306	-2.581811	108007.3	108000
4	-101.1638	300	299.9800	5.532788	67.57697	-2.244381	-2.627641	108001.2	108000
5	-101.1283	300	300.0124	4.923036	60.66718	-1.978967	-2.315126	108002.8	108000
6	-101.0793	300	300.0135	4.871574	58.90486	-1.957571	-2.289946	108008.1	108000

6 rows | 1-10 of 11 columns

Hide

```
new_clusters3 <- apply(df3, 1, closest.cluster)
new_clusters3
```

```
 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37
 3  3  3  3  1  1  1  1  1  1  1  2  1  1  2  1  2  1  1  1  1  2  2  2  2  3  2  3  1  1  1  1  1  1  3  3  3  1
```

Hide

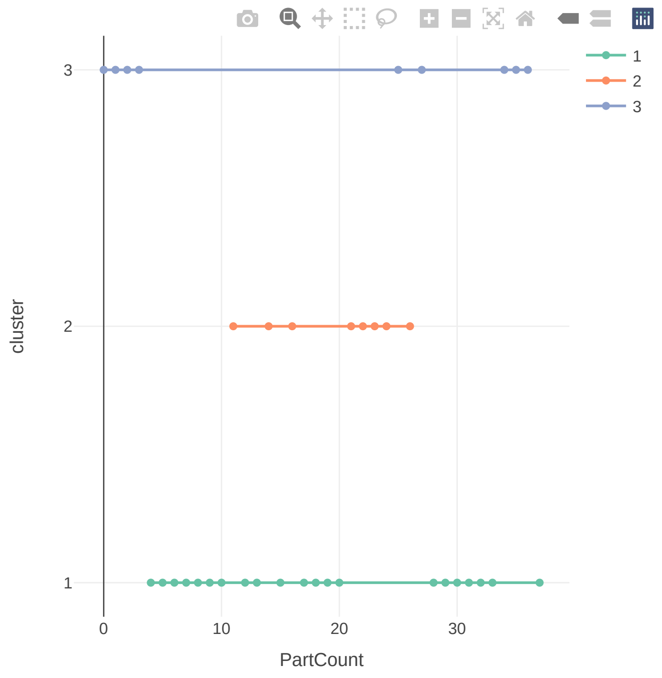
```
clus_3<-setNames(data.frame(cbind(PartCount1, as.numeric(new_clusters3))), c("PartCount", "cluster"))
clus_3$cluster<-as.factor(clus_3$cluster)
head(clus_3)
```

	PartCount <dbl>	cluster <fctr>
1	0	3
2	1	3
3	2	3
4	3	3
5	4	1
6	5	1

6 rows

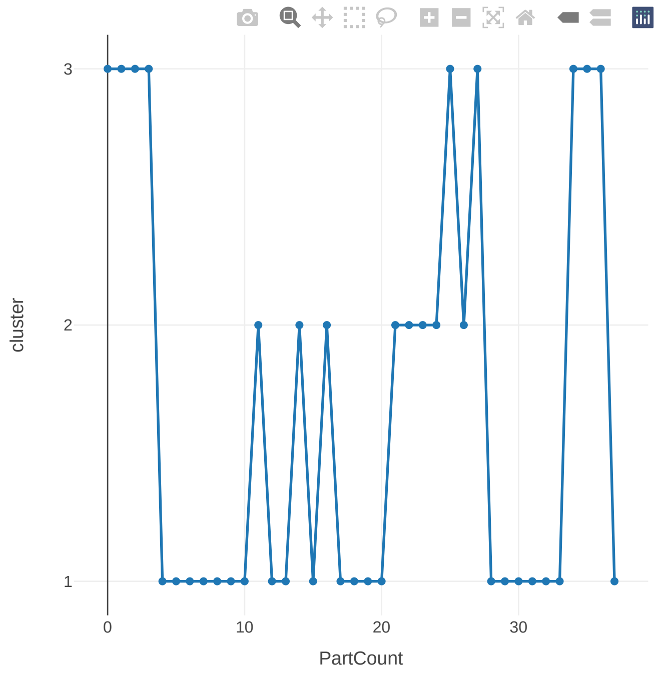
Hide

```
plot_ly(clus_3, x=~PartCount, y=~cluster, color=~cluster, type='scatter', mode='lines+markers')
```



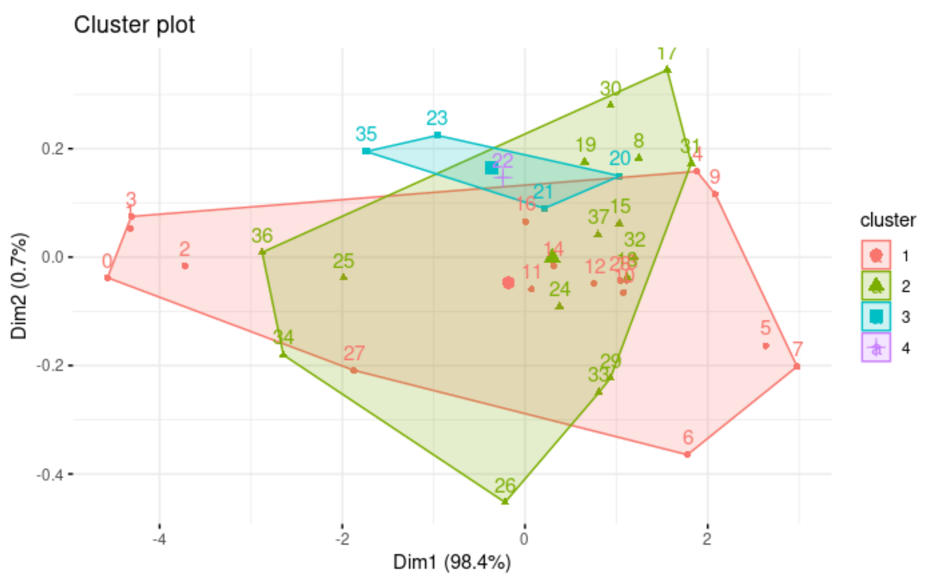
Hide

```
plot_ly(clus_3, x=~PartCount, y=~cluster, type='scatter', mode='lines+markers')
```



Hide

```
fviz_cluster(clus_model, data = df3, ellipse.type = "convex", ggtheme = theme_minimal())
```



Hide

```
d4 <-data4[,9:length(data4)]
df4 <-subset(d4, select = -c(DRPM_1, DFR_1,DCL_1, CL_1, Z_1, RPM_1, FR_1)) %>% scale()
rownames(df4)<-0:(nrow(df4)-1)
head(df4)
```

```
      Load_1    Power_1   Torque_1  Current_1
0  0.7085167  1.0322514 -0.9878015 -0.9837945
1 -1.5902350 -1.4859198  1.4942940  1.5830311
2 -1.2068720 -1.4159524  1.3943453  1.6707288
3 -1.2050782 -1.3055093  1.2441620  1.1053866
4 -1.1826964 -1.2168593  1.2191585  1.1085675
5 -0.8200395 -0.6273371  0.6903390  0.7665127
```

Hide

```
new_clusters4 <- apply(df4, 1, closest.cluster)
new_clusters4
```

```
 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
 3  1  1  1  1  1  2  1  3  1  1  1  1  3  1  1  1  2  1  2  2  1  2  3  2  2  3  1  3  3  1  3  3  3  3  2
```

Hide

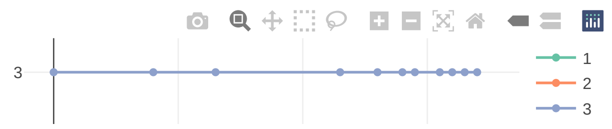
```
clus_4<-setNames(data.frame(cbind(PartCount2, as.numeric(new_clusters4))), c("PartCount", "cluster"))
clus_4$cluster<-as.factor(clus_4$cluster)
head(clus_4)
```

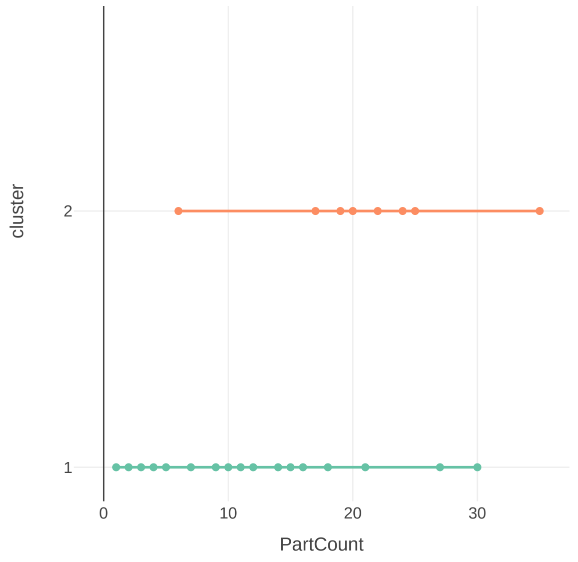
	PartCount <dbl>	cluster <fctr>
1	0	3
2	1	1
3	2	1
4	3	1
5	4	1
6	5	1

6 rows

Hide

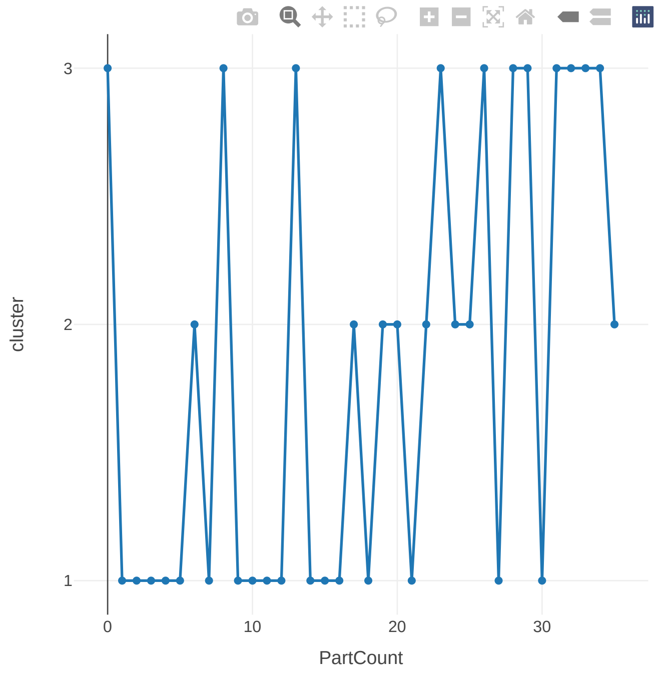
```
plot_ly(clus_4, x=~PartCount, y=~cluster, color=~cluster, type='scatter', mode='lines+markers')
```





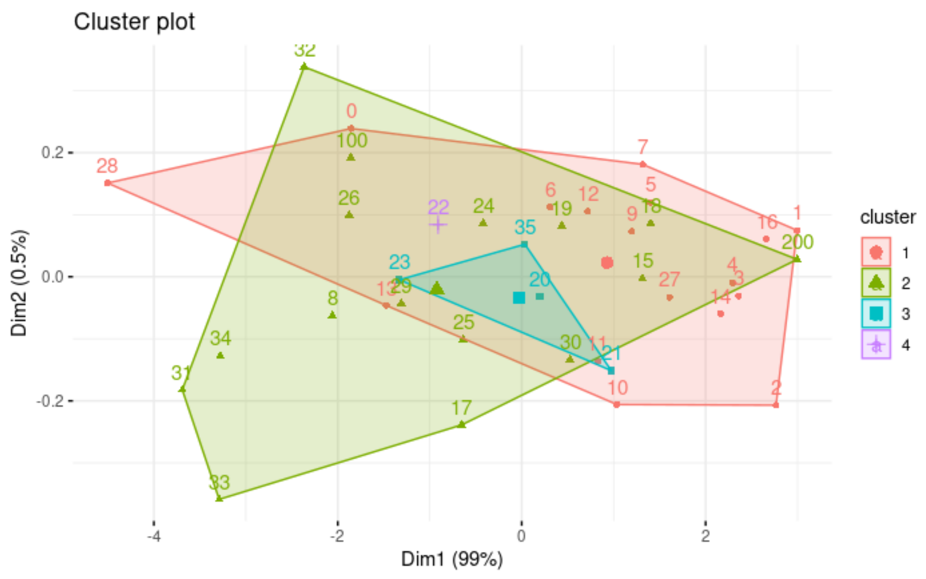
Hide

```
plot_ly(clus_4, x=~PartCount, y=~cluster, type='scatter', mode='lines+markers')
```



Hide

```
append1<-df4[1:2,]
append<-append1+0.05
rownames(append)<-c('100', '200')
df4_2 <-rbind(df4, append)
fviz_cluster(clus_model, data = df4_2, ellipse.type = "convex", ggtheme = theme_minimal())
```



Hide

```
data1_clus<-setNames(data.frame(cbind(data1, new_clusters1)), c(colnames(data1), "clus_val"))
head(data1_clus)
```

	PartCount <int>	DrivePower_1 <dbl>	DriveCurrent_1 <dbl>	DriveTorque_1 <dbl>	DriveLoad_1 <dbl>	DriveTemp_1 <dbl>
0	0	336.9654	-1.761485	-5.272017	2.043660	28.21430
1	1	333.3968	-1.739585	-5.242590	1.943219	31.22598
2	2	334.8904	-1.703647	-5.073129	2.045367	31.64973
3	3	322.9480	-1.656795	-5.170905	1.916698	32.13634
4	4	310.4115	-1.611363	-4.800500	1.941833	32.02482
5	5	323.4846	-1.696997	-5.367629	2.047926	32.43523

6 rows | 1-7 of 20 columns

Hide

```
#write.csv(data1_clus,"~/onedrive/Scenic/data/indexG220/ml/data1_clus.csv", row.names = FALSE)
data2_clus<-setNames(data.frame(cbind(data2, new_clusters2)), c(colnames(data2), "clus_val"))
head(data2_clus)
```

	PartCount <int>	DrivePower_1 <dbl>	DriveCurrent_1 <dbl>	DriveTorque_1 <dbl>	DriveLoad_1 <dbl>	DriveTemp_1 <dbl>
0	38	315.2395	-1.674508	-5.127333	1.871541	32.38667
1	39	306.4307	-1.578799	-5.259966	1.962022	32.46150
2	40	309.8964	-1.618745	-5.003254	1.940297	32.27332
3	41	303.1566	-1.552569	-5.104800	1.849476	32.36652
4	42	309.4175	-1.568303	-5.008200	1.878967	32.23359
5	43	317.1633	-1.628944	-4.921565	1.967301	32.21624

6 rows | 1-7 of 20 columns

Hide

```
#write.csv(data2_clus,"~/onedrive/Scenic/data/indexG220/ml/data2_clus.csv", row.names = FALSE)
data3_clus<-setNames(data.frame(cbind(data3, new_clusters3)), c(colnames(data3), "clus_val"))
head(data3_clus)
```

	PartCount <int>	DrivePower_1 <dbl>	DriveCurrent_1 <dbl>	DriveTorque_1 <dbl>	DriveLoad_1 <dbl>	DriveTemp_1 <dbl>
0	0	326.4132	-1.674697	-5.379623	2.131027	30.62203

	PartCount <int>	DrivePower_1 <dbl>	DriveCurrent_1 <dbl>	DriveTorque_1 <dbl>	DriveLoad_1 <dbl>	DriveTemp_1 <dbl>
1	1	338.7044	-1.737943	-5.314230	2.018062	31.96370
2	2	334.4781	-1.713298	-5.157097	1.868266	32.19314
3	3	328.4928	-1.722575	-5.415111	2.006507	32.31685
4	4	310.6303	-1.654905	-5.099852	1.842061	32.05936
5	5	313.3461	-1.634696	-4.945143	1.893931	32.29702

6 rows | 1-7 of 20 columns

Hide

```
#write.csv(data3_clus,"~/onedrive/Scenic/data/indexG220/ml/data3_clus.csv", row.names = FALSE)
data4_clus<-setNames(data.frame(cbind(data4, new_clusters4)), c(colnames(data4), "clus_val"))
head(data4_clus)
```

	PartCount <int>	DrivePower_1 <dbl>	DriveCurrent_1 <dbl>	DriveTorque_1 <dbl>	DriveLoad_1 <dbl>	DriveTemp_1 <dbl>
0	38	340.6525	-1.748558	-5.205259	2.010267	32.34756
1	39	318.6774	-1.620291	-4.909158	1.900121	32.26306
2	40	316.2335	-1.672514	-5.084567	1.951294	32.45467
3	41	324.5975	-1.640846	-5.085328	1.928411	32.35102
4	42	325.5752	-1.641872	-5.281246	2.044578	32.41111
5	43	310.4115	-1.650937	-5.302817	1.974487	32.30569

6 rows | 1-7 of 20 columns

Hide

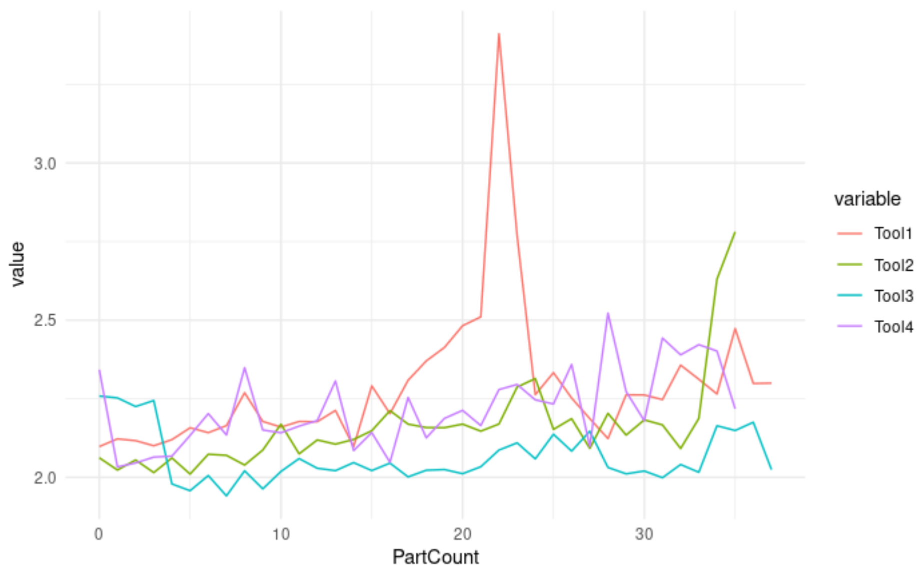
```
#write.csv(data4_clus,"~/onedrive/Scenic/data/indexG220/ml/data4_clus.csv", row.names = FALSE)
```

Torque and clustering

Hide

```
torque<-setNames(data.frame(
  qpcR:::cbind.na(data1$PartCount,
                  abs(d1$Torque_1),
                  abs(d2$Torque_1),
                  abs(d3$Torque_1),
                  abs(d4$Torque_1))),
  c("PartCount", "Tool1", "Tool2", "Tool3", "Tool4"))

torque[torque== "NA"] <-''

tplot<-reshape2::melt(torque[,c("PartCount", "Tool1", "Tool2", "Tool3", "Tool4")], id.vars=1)
ggplot(tplot, aes(PartCount, y=value)) + geom_line(aes(color = variable)) + theme_minimal()
```

```
Warning: Removed 4 row(s) containing missing values (geom_path).
```



Hide

```
dataset_clus1<-setNames(data.frame(cbind(PartCount1, abs(torque$Tool1), clus_1$cluster)), c("PartCount", "Torqu
e", "Cluster"))
gg1<-
  ggplot(dataset_clus1, aes(PartCount, Torque)) +
  geom_point(aes(color = as.factor(Cluster)), size=2.5) +
  geom_line(color="grey") +
  theme_bw() +
  scale_y_continuous(limits = c(1.75, 3.5))+
  theme(legend.position = "none")
 # scale_color_discrete(name =expression("Cluster"))

dataset_clus2<-setNames(data.frame(cbind(PartCount2, abs(torque$Tool2), clus_2$cluster)), c("PartCount", "Torqu
e", "Cluster"))
```

```
Warning in cbind(PartCount2, abs(torque$Tool2), clus_2$cluster) :
  number of rows of result is not a multiple of vector length (arg 1)
```

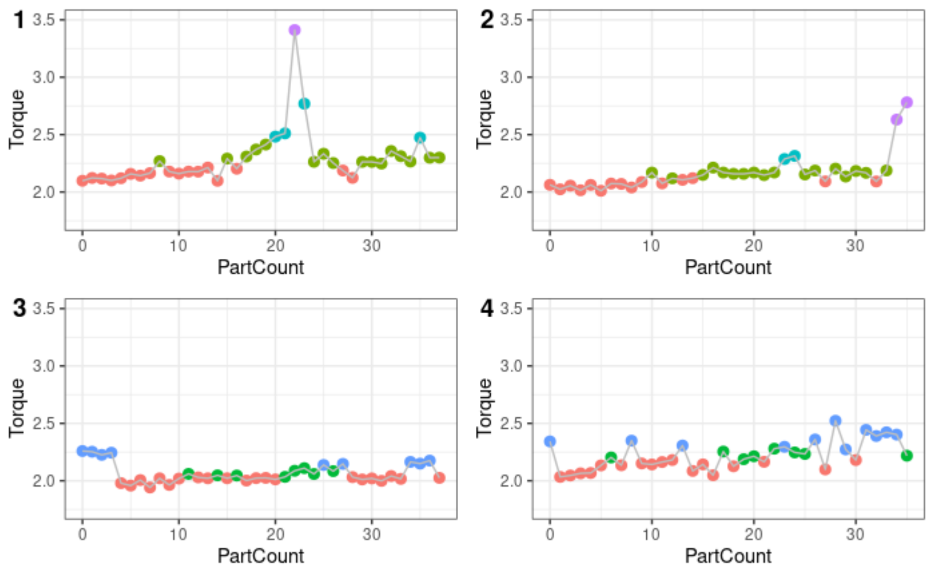
Hide

```
dataset_clus2<-dataset_clus2[1:36,]
gg2<-
  ggplot(dataset_clus2, aes(PartCount, Torque)) +
  geom_point(aes(color = as.factor(Cluster)), size=2.5) +
  geom_line(color="grey") +
  theme_bw()+
  scale_y_continuous(limits = c(1.75, 3.5))+
  #labs(title="Cluster Identification on Torque Values for Gundrill") +
  theme(plot.title = element_text(hjust = 0.5, size = 18, face="bold"))+
  scale_color_discrete(name =expression("Cluster"))

dataset_clus3<-setNames(data.frame(cbind(PartCount1, abs(torque$Tool3), clus_3$cluster)), c("PartCount", "Torqu
e", "Cluster"))
gg3<-
  ggplot(dataset_clus3, aes(PartCount, Torque)) +
  geom_point(aes(color = as.factor(Cluster)), size=2.5) +
  geom_line(color="grey") +
  theme_bw() +
  scale_y_continuous(limits = c(1.75, 3.5))+
  theme(legend.position = "none")
 # scale_color_discrete(name =expression("Cluster"))

dataset_clus4<-setNames(data.frame(cbind(PartCount2, abs(torque$Tool4), clus_4$cluster)), c("PartCount", "Torqu
e", "Cluster"))
```

```
Warning in cbind(PartCount2, abs(torque$Tool4), clus_4$cluster) :
  number of rows of result is not a multiple of vector length (arg 1)
```

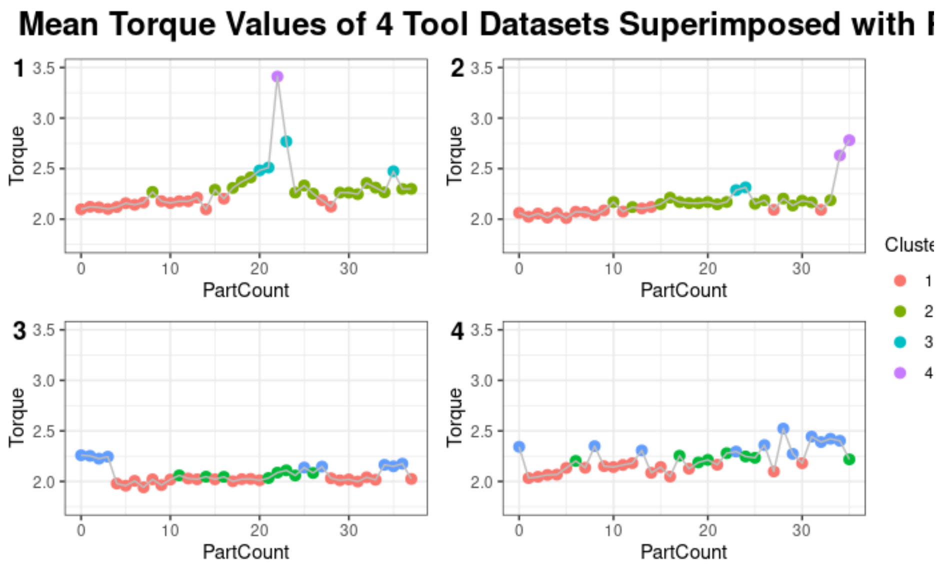
Hide

```
dataset_clus4<-dataset_clus4[1:36,]
gg4<-
  ggplot(dataset_clus4, aes(PartCount, Torque)) +
  geom_point(aes(color = as.factor(Cluster)), size=2.5) +
  geom_line(color="grey") +
  theme_bw() +
  scale_y_continuous(limits = c(1.75, 3.5))+
  theme(legend.position = "none")
  #scale_color_discrete(name =expression("Cluster"))

plot_row<-plot_grid(
  gg1,
  gg2 + theme(legend.position="none"),
  gg3,
  gg4,
  align = 'vh',
  hjust = -1,
  nrow = 2,
  labels = c('1', '2', '3', '4'))
plot_row
```



Hide

```

legend <- get_legend(gg2 + theme(legend.box.margin = margin(0, 0, 0, 12)))
grid<-plot_grid(plot_row, legend, rel_widths = c(3,.2))

title <- get_title(gg2)

title <- ggdraw() +
  draw_label("Mean Torque Values of 4 Tool Datasets Superimposed with Respective Clusters",
    fontface = 'bold',x = 0.02, hjust = 0, size = 18)

plot_grid(title, grid, rel_heights = c(0.1, 1), nrow=2)
```



Data I used to generate plots for a gif for better visualisation of the progression of Torque

Hide

```
plot_ly(g_300_S1, x = ~PartCount, y = ~abs(Torque), type = 'scatter',  mode = 'markers')
plot_ly(g_300_S1, x = ~PartCount, y = ~abs(RPM), type = 'scatter',  mode = 'markers') # might be interesting to c
ompare RPM and clusters

plot_ly(g_STorque, x = ~PartCount, y = ~abs(Mean), type = 'scatter',  mode = 'lines+markers', name="Mean") %>%
  layout(title = "<b>S1 Torque for Sister Tool</b>", yaxis = list(title = "abs(Torque)", range = c(0,5))) %>%
#        error_y = ~list(array = mad, color = '#000000')) %>%
  add_trace(y = ~abs(Median), type = 'scatter',  mode = 'lines+markers', name = "Median") %>%
  add_trace(y = ~mad, type = 'scatter',  mode = 'lines+markers', name = "mad", line = list(color = "red"), marker
= list(color = "red"))


test<-data.frame(cbind(g_300_S1$PartCount, g_300_S1$Torque))
colnames(test)<-c("PartCount", "Torque")
test$PartCount<-as.factor(test$PartCount)
head(test)
glimpse(test)

test2<-test %>% pivot_wider(names_from = PartCount, values_from = Torque)
head(test2)
colnames(test2)
ncol(test2)

test2<-
  test%>%
  group_by(PartCount) %>%
  mutate(row = row_number()) %>%
  tidyr::pivot_wider(names_from = PartCount, values_from = Torque) %>%
  select(-row)

glimpse(test2)
test2<-na.omit(test2)
```

The plots for the gif

Hide

```

Sys.sleep(2)
plot_ly(test2, y = ~abs(test2$`0`), type = 'scatter',  mode = 'lines+markers') %>% layout(title = "<b>Part 1</b
>", yaxis = list(title = "abs(Torque)", range = c(0,5)))
```

```
Error in abs(test2$`0`) : non-numeric argument to mathematical function
```