

ggplot2 Notes

Dayne Okuhara

August 31, 2017

First lets begin by examining the main data frame "midwest" and the columns which will be used to design the scatter plot.

1 Scatter Plots

```
> library(ggplot2)
> library(dplyr)
> options(scipen=999) # turn off scientific notation like 1e+06
> data("midwest", package = "ggplot2") # load the data
> #midwest <- read.csv("http://goo.gl/G1K41K") # alt source
>
>
> names(midwest)
```

[1]	"PID"	"county"	"state"
[4]	"area"	"poptotal"	"popdensity"
[7]	"popwhite"	"popblack"	"popamerindian"
[10]	"popasian"	"popother"	"percwhite"
[13]	"percblack"	"percamerindian"	"percasian"
[16]	"percother"	"popadults"	"perchsd"
[19]	"percollege"	"percprof"	"poppovertyknown"
[22]	"percpovertyknown"	"percbelowpoverty"	"percchildbelowpovert"
[25]	"percadultpoverty"	"percelderlypoverty"	"inmetro"
[28]	"category"		

```
> head(midwest[,c("area", "poptotal", "state")])

# A tibble: 6 x 3
  area poptotal state
<dbl>   <int> <chr>
1 0.052   66090  IL
2 0.014   10626  IL
3 0.022   14991  IL
4 0.017   30806  IL
5 0.018    5836  IL
6 0.050   35688  IL

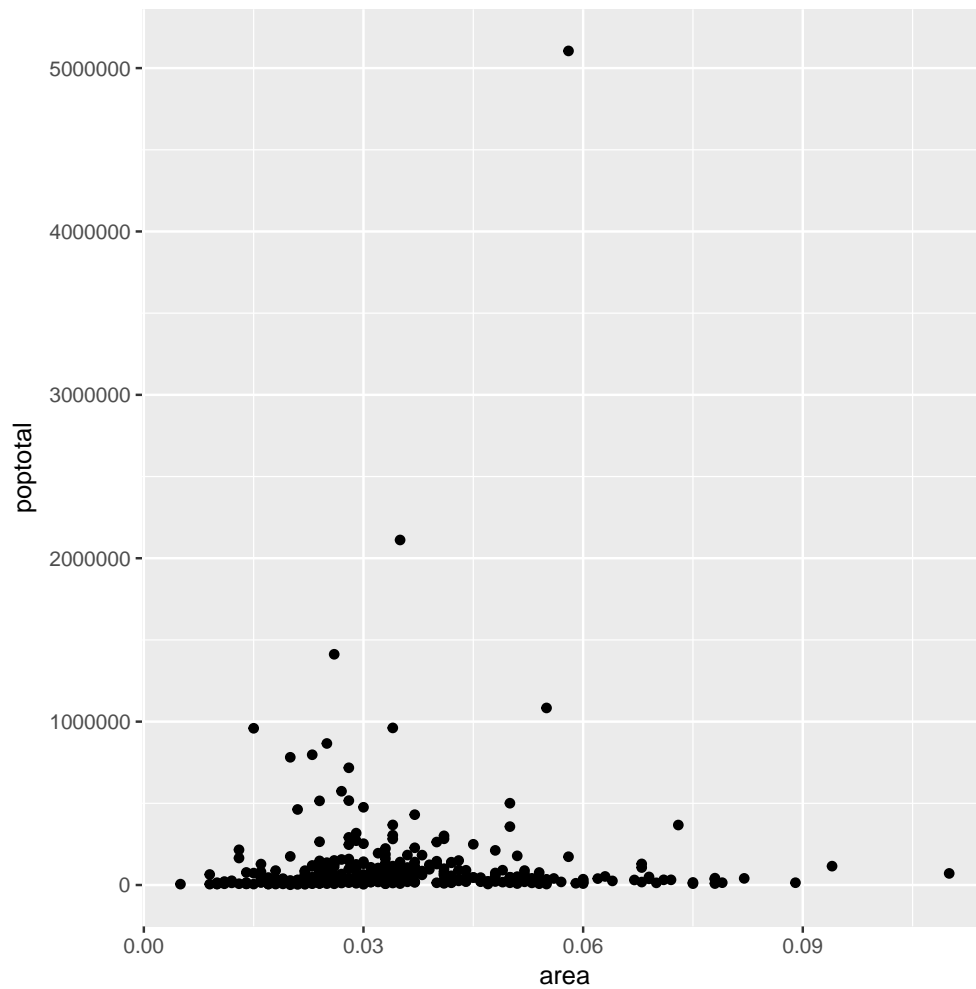
> sapply(midwest, function(y) sum(length(which(is.na(y)))))
```

PID	county	state
0	0	0
area	poptotal	popdensity
0	0	0
popwhite	popblack	popamerindian
0	0	0
popasian	popother	percwhite
0	0	0
percblack	percamerindian	percasian
0	0	0
percother	popadults	perchsd
0	0	0
percollege	percprof	poppovertyknown
0	0	0
percpovertyknown	percbelowpoverty	percchildbelowpovert
0	0	0
percadultpoverty	percelderlypoverty	inmetro
0	0	0
category		
0		

>

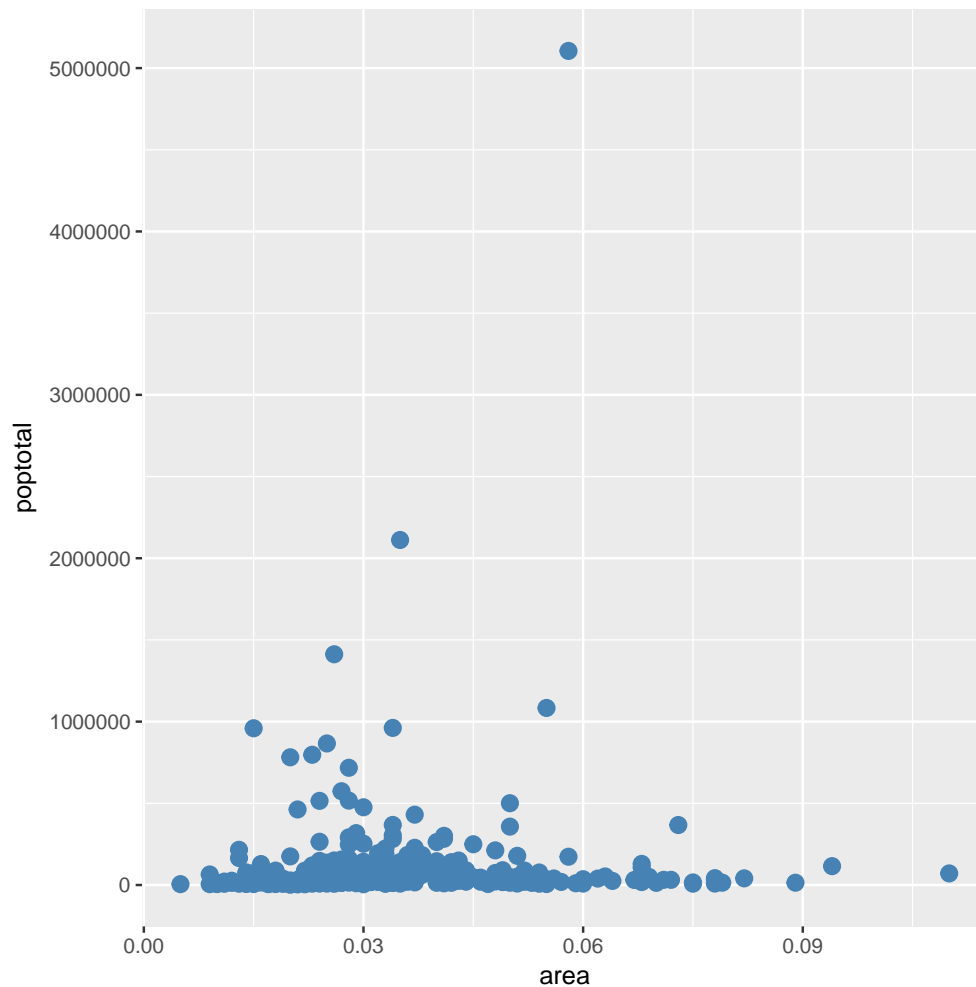
Good, there are no na's. Now let's look at the default scatterplot where each point represents a county

```
> ggplot(midwest, aes(x = area, y = poptotal)) + geom_point()
>
```



It is easiest if we start by setting the symbol size and points. This way we can save the plot as an object and gradually add more layers.

```
> g1 <- ggplot(midwest, aes(x = area, y = poptotal)) +  
+   geom_point(col = "steelblue", size=3)  
> plot(g1)
```

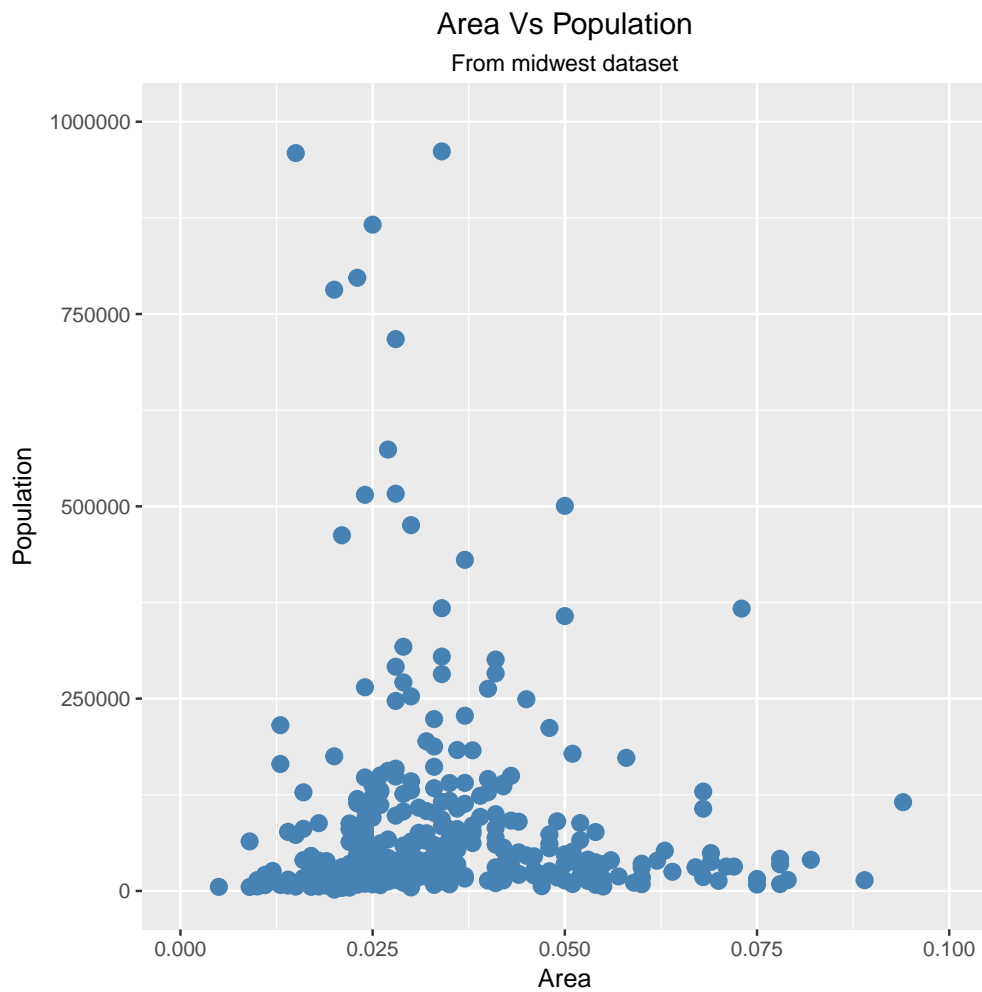


Next let's zoom in on the graph by manually setting the x and y limits to $\text{area} < 0.1$ and $\text{poptotal} < 1000000$. Also add a main title, subtitle, and change the axis labels. The main title and subtitle are left-aligned by default. We will center align the titles.

```
> # number of excluded points from plot
>
> sum(midwest$area > 0.1 | midwest$poptotal > 1000000)
```

```
[1] 5
```

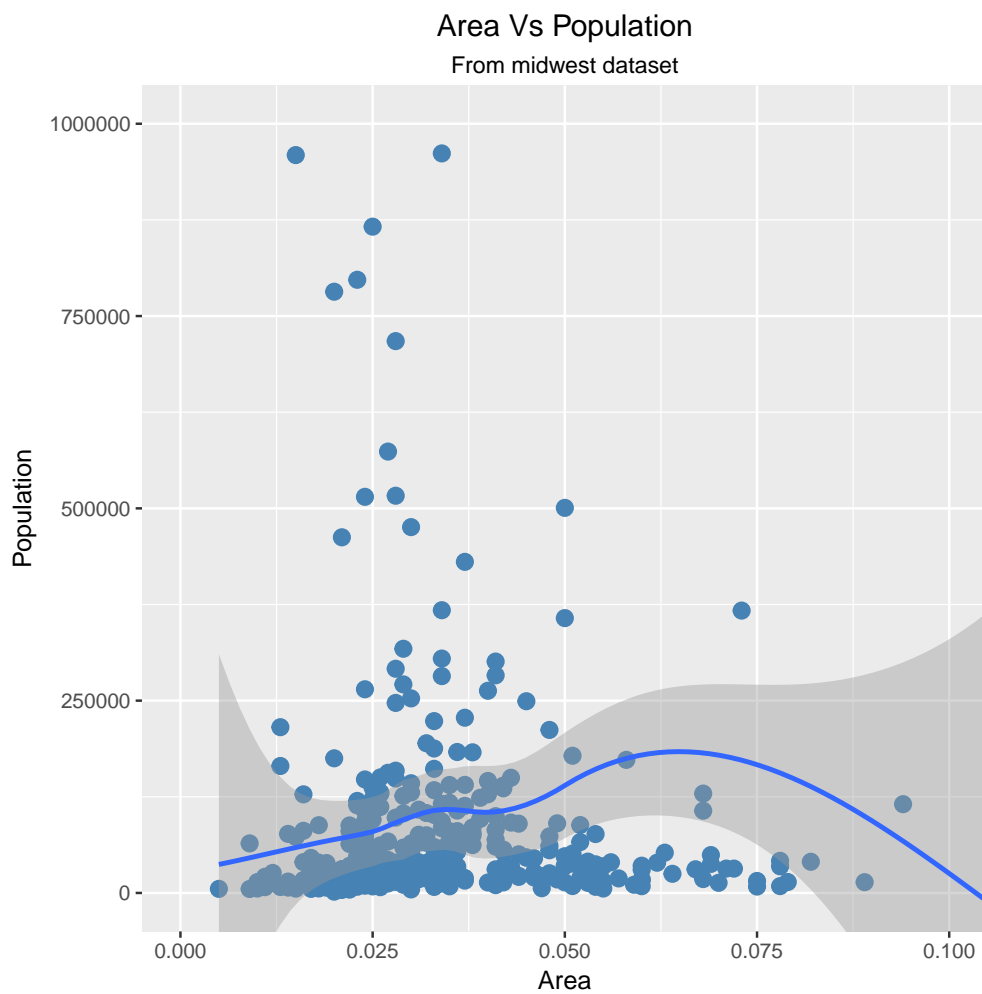
```
> g1.1 <- g1 + coord_cartesian(xlim = c(0,0.1), ylim = c(0, 1000000)) +
+   ggtitle("Area Vs Population", subtitle="From midwest dataset") +
+   theme(plot.title = element_text(hjust = 0.5), plot.subtitle = element_text(hjust = 0.5))
+   xlab("Area") +
+   ylab("Population")
> plot(g1.1)
```



1.1 Adding A Regression Line

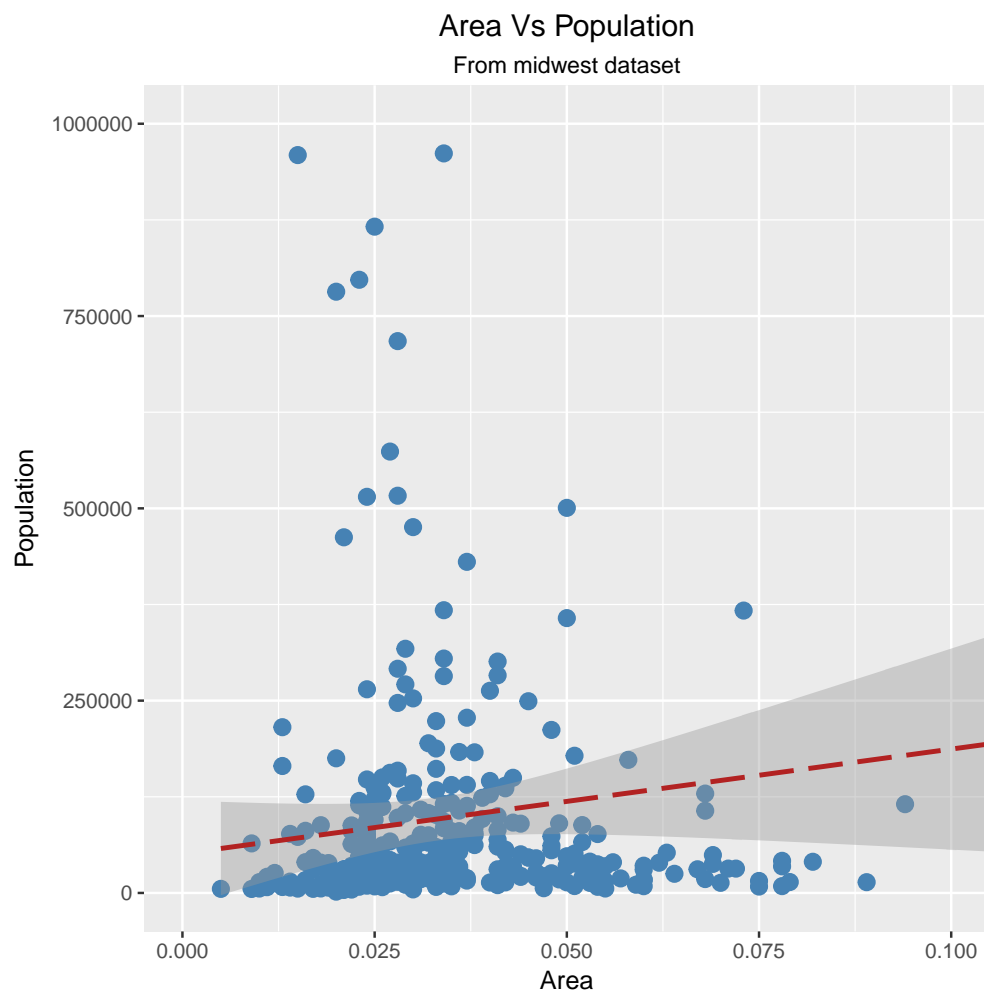
Add a regression line with `geom_smooth()`. The default method is a loess smoothed fit curve with 95% confidence region.

```
> g1.1 + geom_smooth()  
>  
> # set se=FALSE to turnoff confidence bands  
> #g1.1 + geom_smooth(se = FALSE)  
>
```



lm (linear regression) is another method. By default the line is solid and blue. Change it to longdash and "firebrick" color.

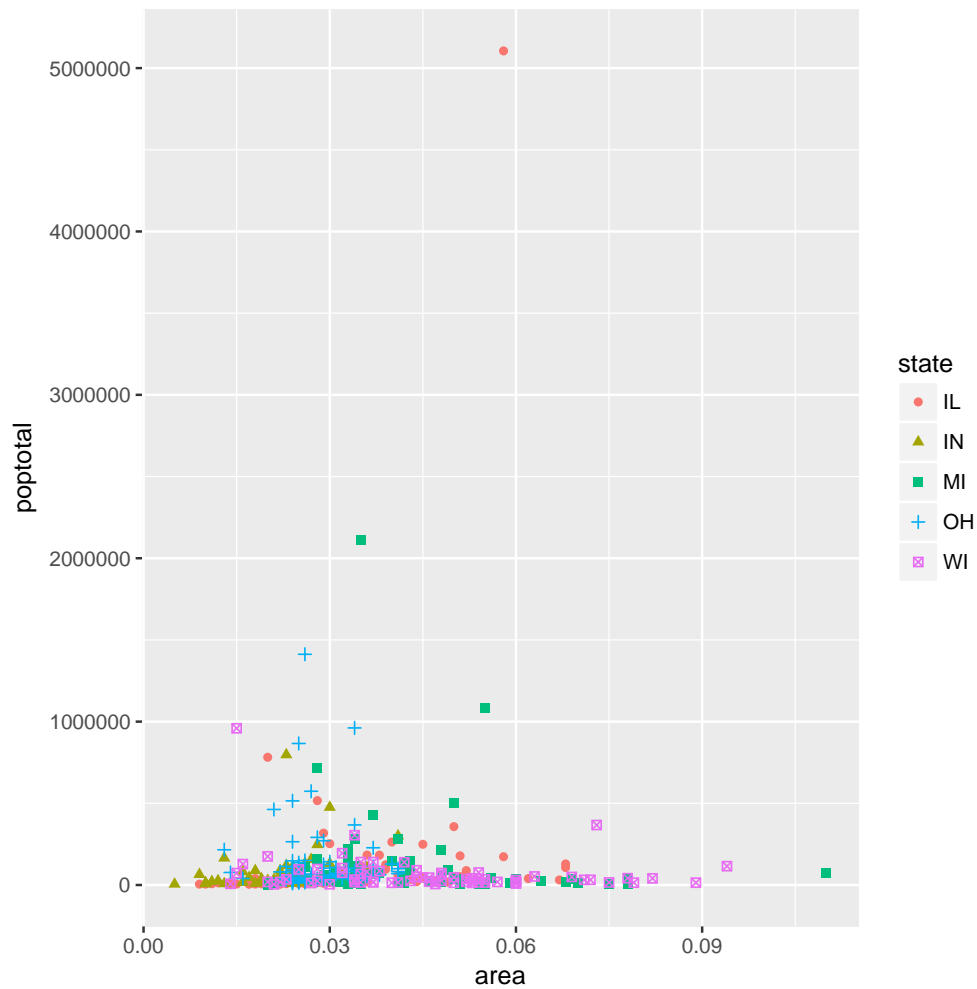
```
> g1.2 <- g1.1 + geom_smooth(method = "lm",  
+                             col = "firebrick",  
+                             linetype = "longdash")  
> plot(g1.2)
```



1.2 Scatterplot With Multiple Groups

First, examine plot with default symbols and colors for data grouped by state.

```
> ggplot(midwest, aes(x = area, y = poptotal, shape = state, col=state)) + geom_point()
```



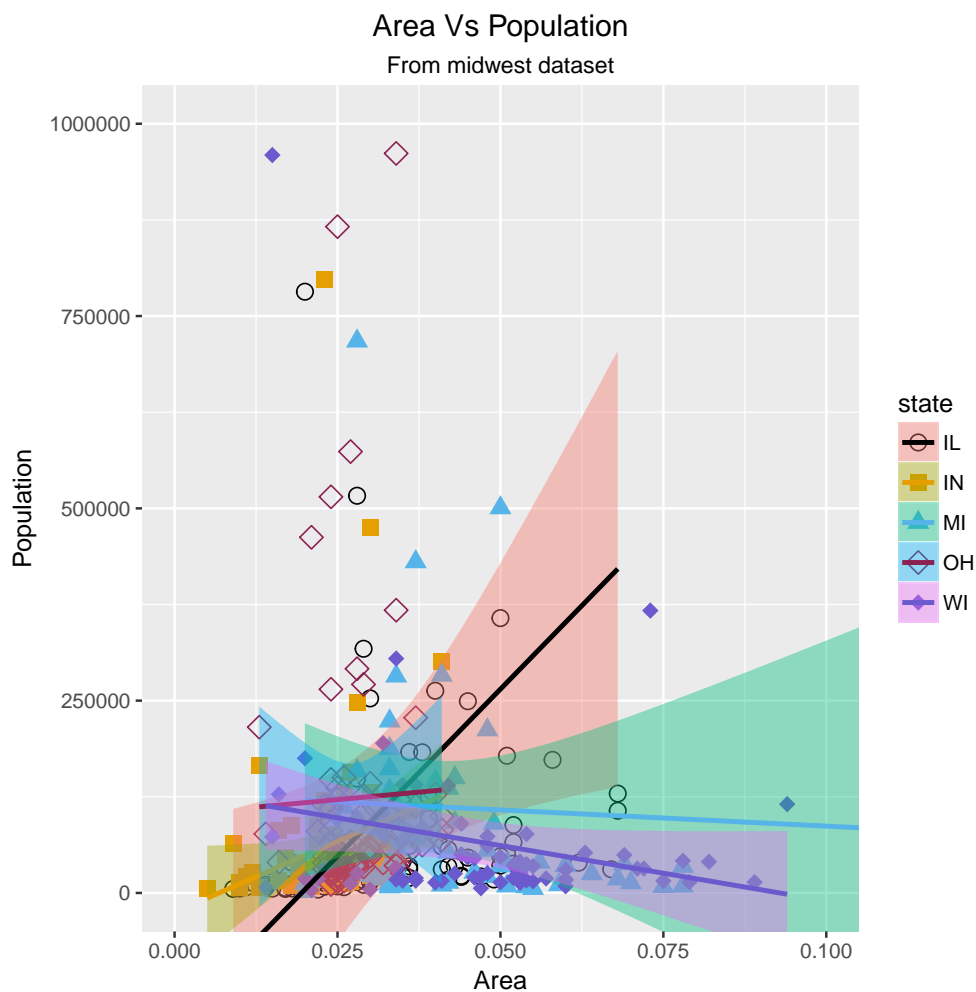
Set all other parameters as used above for g1.1. Change the colors, symbols and symbol size.

```
> g2 <- ggplot(midwest, aes(x = area, y = poptotal, shape = state, col = state)) + geom_point()
> g2.1 <- g2 + coord_cartesian(xlim = c(0,0.1), ylim = c(0, 1000000)) +
+   ggtitle("Area Vs Population", subtitle="From midwest dataset") +
+   theme(plot.title = element_text(hjust = 0.5), plot.subtitle = element_text(hjust = 0.5)) +
+   xlab("Area") +
+   ylab("Population")
> g2.2 <- g2.1 +
+   scale_shape_manual(values = c(1, 15, 17, 5, 18)) +
+   scale_color_manual(values = c('black', '#E69F00', '#56B4E9', "violetred4", "slateblue"))
> plot(g2.2)
```



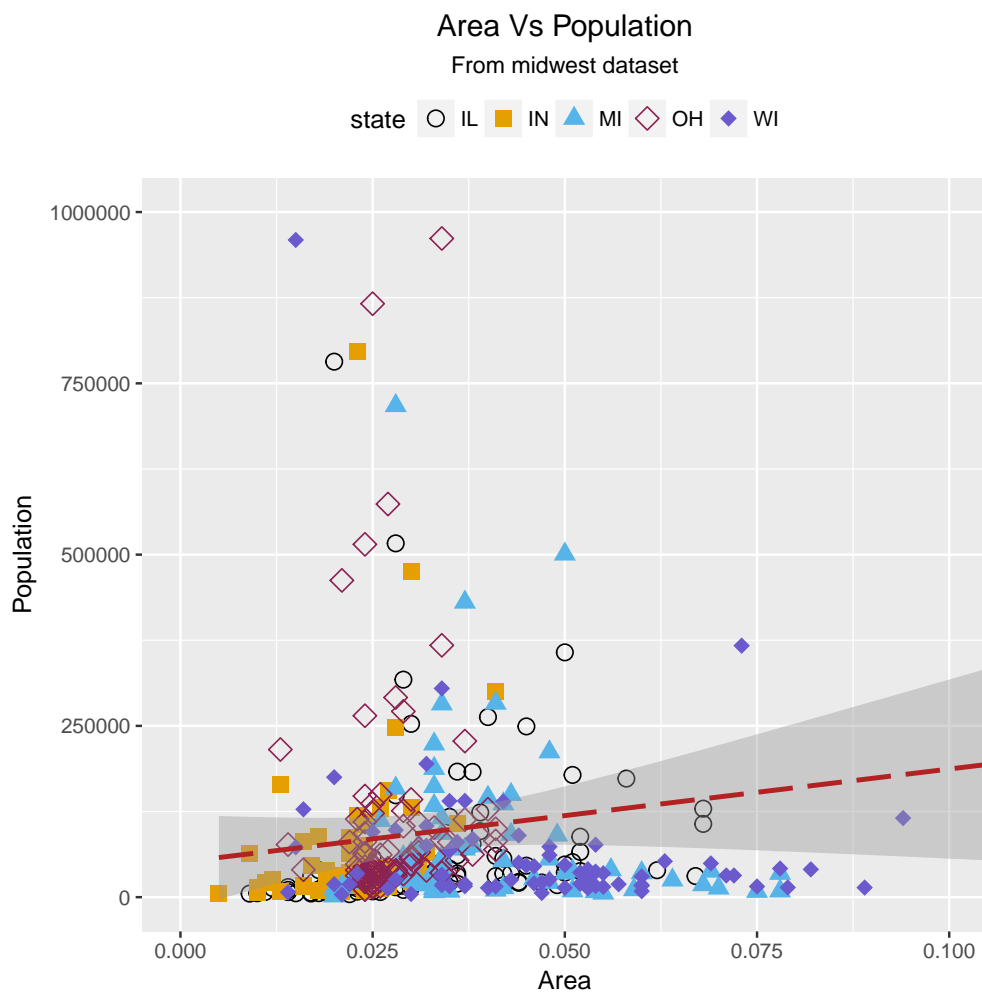

Add regression line...

```
> g2.2 + geom_smooth(method = "lm", aes(fill=state))
```



Oh snap 5 regression lines. ggplot is calculating the regression lines based on grouping variable state. This is because g2.2 was saved with data already grouped by state. We need to preform the `geom_smooth()` function with a ggplot object that has been grouped. Also, let's move the legend to the top.

```
> g2 <- ggplot(midwest, aes(x = area, y = poptotal)) +
+   coord_cartesian(xlim = c(0,0.1), ylim = c(0, 1000000)) +
+   ggtitle("Area Vs Population", subtitle="From midwest dataset") +
+   theme(plot.title = element_text(hjust = 0.5), plot.subtitle = element_text(hjust = 0.5))
+   xlab("Area") +
+   ylab("Population")
> g2.3 <- g2 + geom_point(aes(shape = state, col = state ), size = 3) +
+   scale_shape_manual(values = c(1, 15, 17, 5, 18)) +
+   scale_color_manual(values = c('black', '#E69F00', '#56B4E9', "violetred4", "slateblue"))
+   geom_smooth(method = "lm",
+               col = "firebrick",
+               linetype = "longdash") +
+   theme(legend.position="top")
> plot(g2.3)
>
```



2 Box Plots

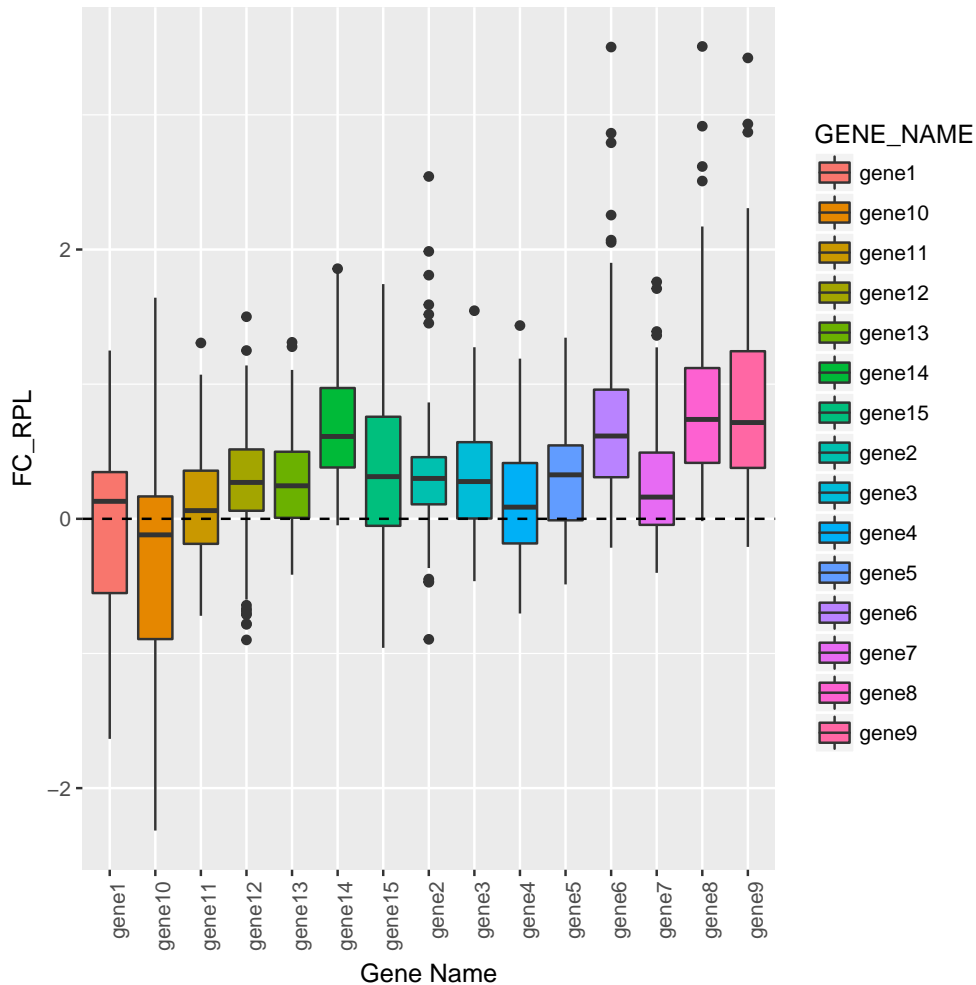
In this section we will create boxplots for two different data frames then add points from a third data frame. Box plots will be created for data frames df2 and df3. The overlay points will come from df1. Both df2 and df3 are subsets of df1.

2.1 Multiple Grouping Factors

```
> df1 <- read.csv("CSample_Data1.csv", header = T)
> df2 <- read.csv("CSample_tab1.csv", header = T)
> df3 <- read.csv("CSample_tab2.csv", header = T)
```

Let's examine the default box plot with no interacting factors and coloured by GENE_NAME

```
> g1 <- ggplot(df2, aes(x = GENE_NAME, y = FC_RPL, fill = GENE_NAME)) +
+   geom_boxplot() +
+   geom_hline(aes(yintercept=0), colour= "black", linetype = "dashed") +
+   theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
+   labs(x = "Gene Name")
> plot(g1)
```



The data in `df2` has already been sorted by Class and `GENE_NAME`. However the default plot has rearranged the genes in a different order. The code below correctly reorders the factors levels for all three data frames.

```
> attributes(df2$GENE_NAME)

$levels
[1] "gene1" "gene10" "gene11" "gene12" "gene13" "gene14" "gene15" "gene2"
[9] "gene3" "gene4" "gene5" "gene6" "gene7" "gene8" "gene9"

$class
[1] "factor"

> df1$GENE_NAME <- factor(df1$GENE_NAME, levels = c("gene1", "gene2", "gene3",
+ "gene4", "gene5", "gene6", "gene7", "gene8", "gene9", "gene10",
+ "gene11", "gene12", "gene13", "gene14", "gene15" ))
> df2$GENE_NAME <- factor(df2$GENE_NAME, levels = c("gene1", "gene2", "gene3",
+ "gene4", "gene5", "gene6", "gene7", "gene8", "gene9", "gene10",
+ "gene11", "gene12", "gene13", "gene14", "gene15" ))
> df3$GENE_NAME <- factor(df3$GENE_NAME, levels = c("gene1", "gene2", "gene3",
+ "gene4", "gene5", "gene6", "gene7", "gene8", "gene9", "gene10",
```

```

+           "gene11", "gene12", "gene13", "gene14", "gene15" ))
> attributes(df1$GENE_NAME)

$levels
[1] "gene1" "gene2" "gene3" "gene4" "gene5" "gene6" "gene7" "gene8"
[9] "gene9" "gene10" "gene11" "gene12" "gene13" "gene14" "gene15"

$class
[1] "factor"

> attributes(df2$GENE_NAME)

$levels
[1] "gene1" "gene2" "gene3" "gene4" "gene5" "gene6" "gene7" "gene8"
[9] "gene9" "gene10" "gene11" "gene12" "gene13" "gene14" "gene15"

$class
[1] "factor"

> attributes(df3$GENE_NAME)

$levels
[1] "gene1" "gene2" "gene3" "gene4" "gene5" "gene6" "gene7" "gene8"
[9] "gene9" "gene10" "gene11" "gene12" "gene13" "gene14" "gene15"

$class
[1] "factor"

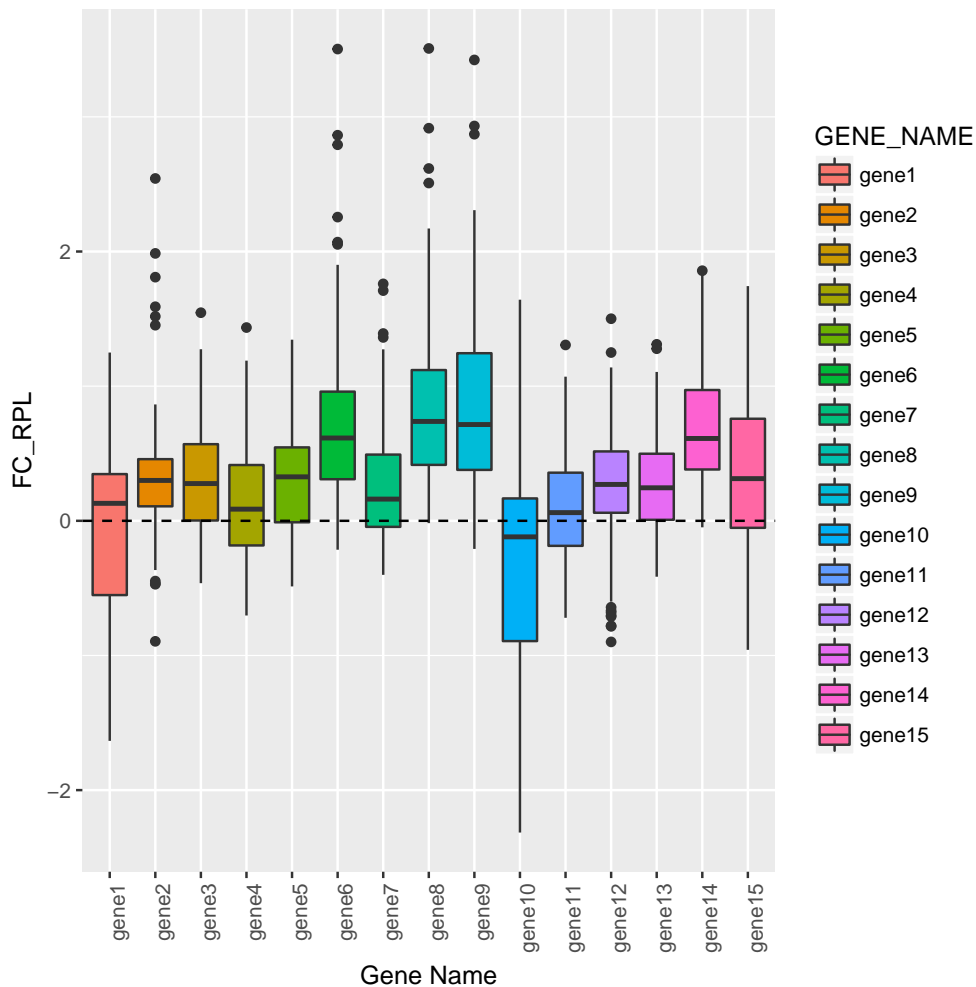
```

Now the gene order should be correct.

```

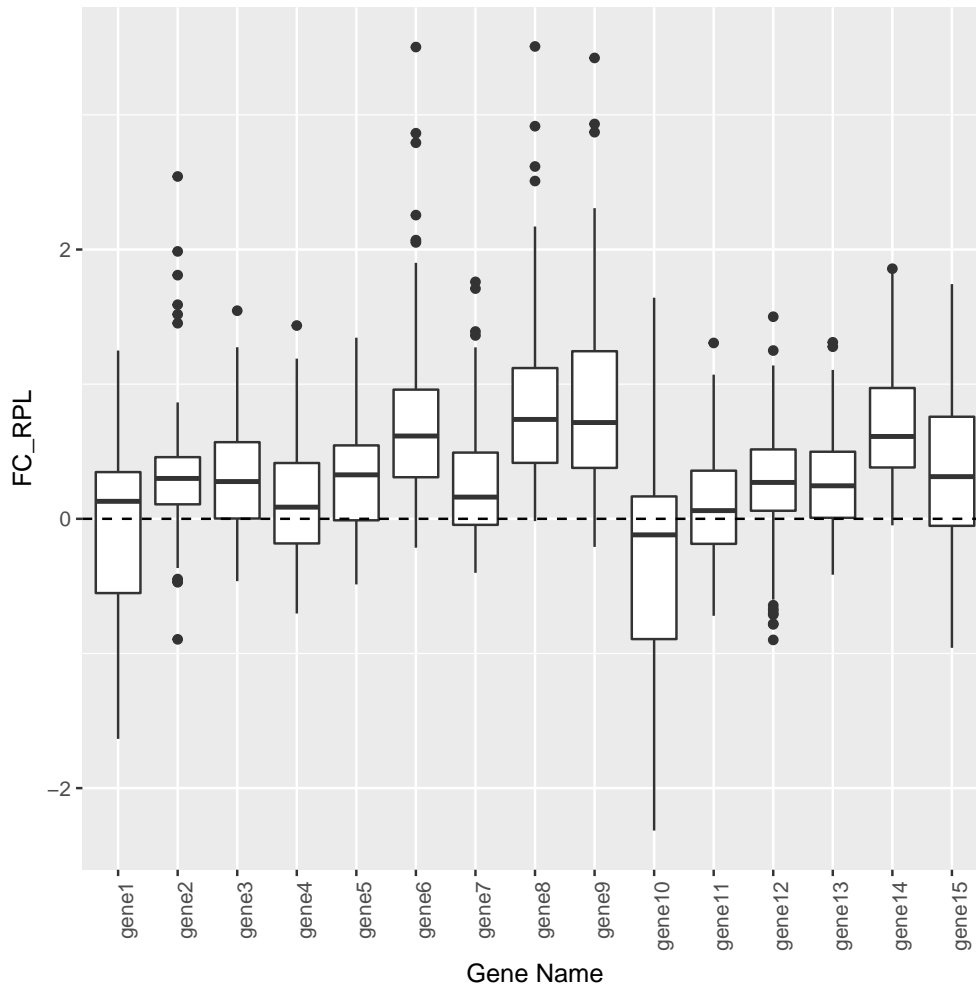
> g1 <- ggplot(df2, aes(x = GENE_NAME, y = FC_RPL, fill = GENE_NAME)) +
+   geom_boxplot() +
+   geom_hline(aes(yintercept=0), colour= "black", linetype = "dashed") +
+   theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
+   labs(x = "Gene Name")
> plot(g1)

```



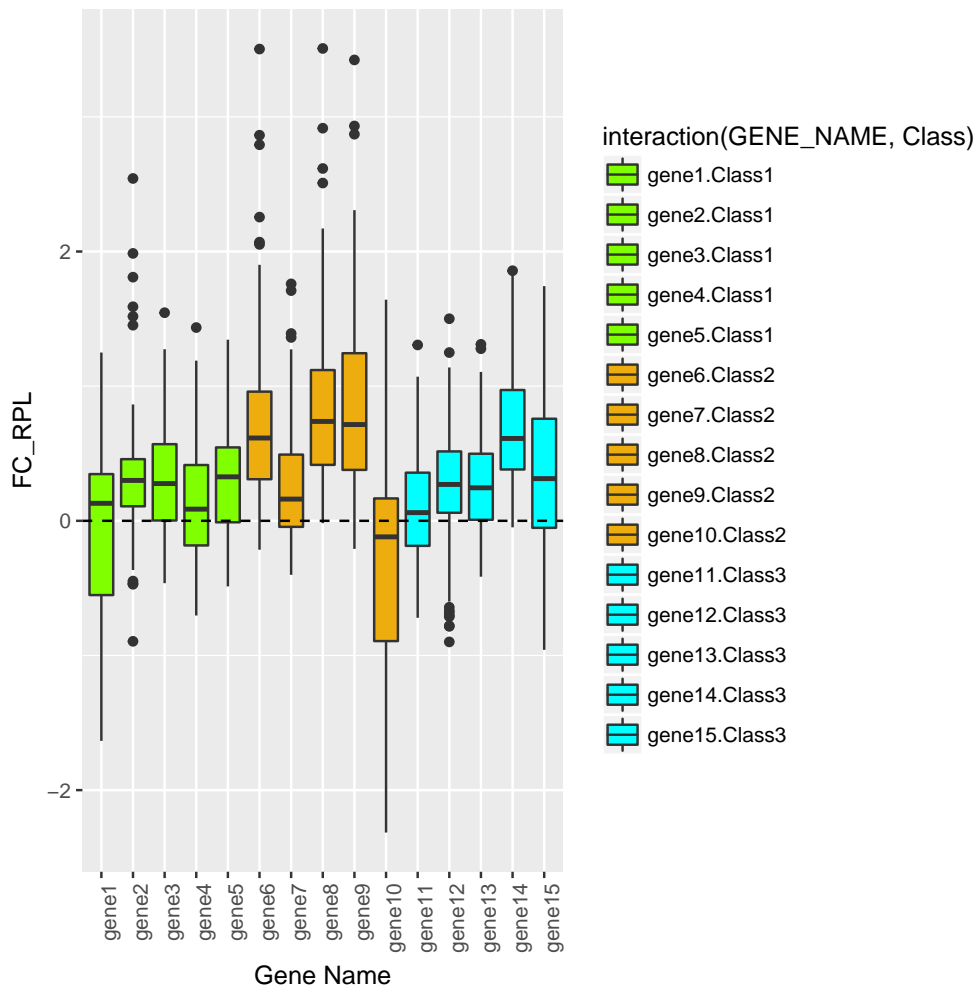
First lets manually select the colors for the box plots with the `scale_fill_manual(values = colors)` command. Notice the boxes are not filled.

```
> colors <- c(rep('chartreuse', 5), rep('darkgoldenrod2', 5), rep('cyan', 5))
> g1 <- ggplot(df2, aes(x = GENE_NAME, y = FC_RPL)) +
+   geom_boxplot() +
+   scale_fill_manual(values = colors) +
+   geom_hline(aes(yintercept=0), colour= "black", linetype = "dashed") +
+   theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
+   labs(x = "Gene Name")
> plot(g1)
```



The `interaction()` function is used to group the data based on two or more factors. In this case, we want to first group the boxplots based by `GENE_NAME` and `Class`. The use the `scale_fill_manual(values=colors)` command

```
> g1 <- ggplot(df2, aes(x = GENE_NAME, y = FC_RPL, fill = interaction(GENE_NAME, Class))) +
+   geom_boxplot() +
+   scale_fill_manual(values=colors) +
+   geom_hline(aes(yintercept=0), colour= "black", linetype = "dashed") +
+   theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
+   labs(x = "Gene Name")
> plot(g1)
```



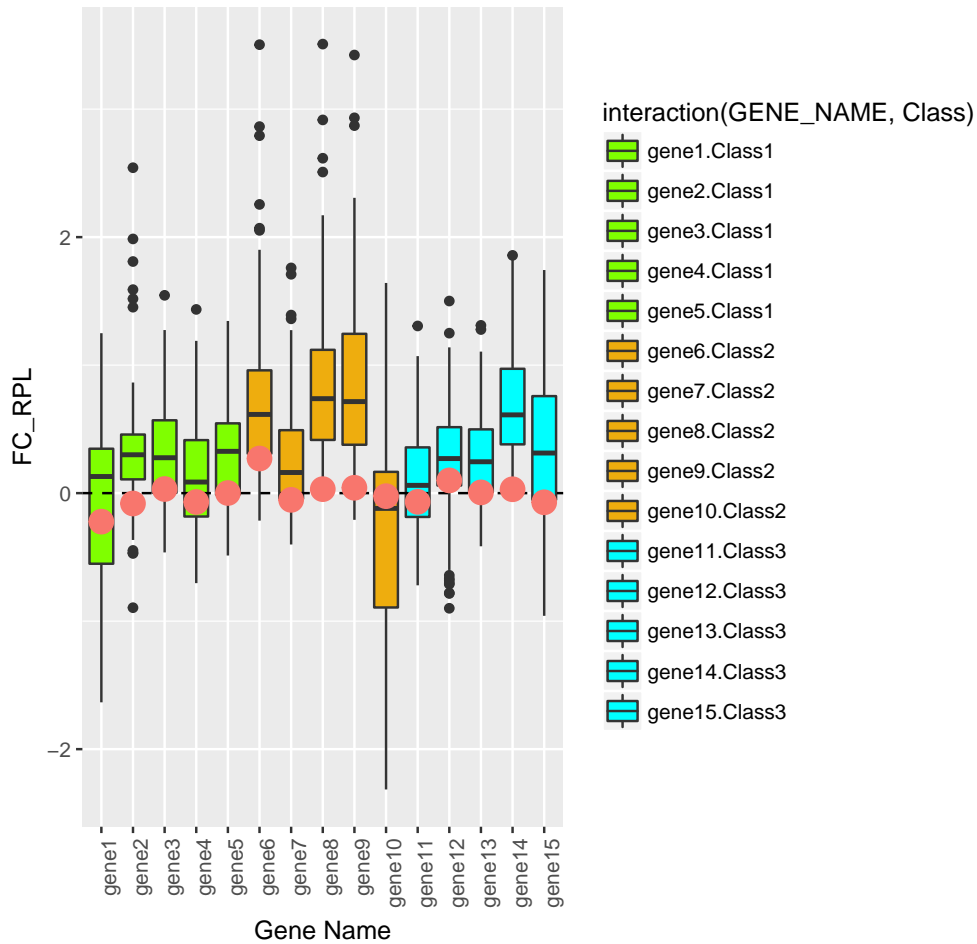
Next lets select a set of data from df1 and over lay the points on df2.

```
> sample <- "cmpd-25168828.0"
> tmp.df <- df1[df1$sample == sample,]
> title <- paste(sample, "vs df2")
> g2 <- g1 + geom_point(data = tmp.df, aes(x = GENE_NAME, y = FC_RPL, size = 3, col = 'red'))
+ ggtitle(title)
> plot(g2)
```

Notice how the legend includes the `geom_point()` data and it makes it look awkward. Since the `geom_point()` data is an added layer to the `g1` object, the `show.legend = FALSE` option will only hide the `geom_point()` legend.

```
> g2 <- g1 + geom_point(data = tmp.df, aes(x = GENE_NAME, y = FC_RPL, size = 3, col = 'red'))
+ ggtitle(title)
> plot(g2)
```


compd-25168828.0 vs df2



Finally, lets put it all together by creating an identical boxplots for df3

```
> title <- paste(sample,"vs df3")
> g3 <- ggplot(df3, aes(x = GENE_NAME, y = FC_RPL, fill = interaction(GENE_NAME, Class))) +
+   geom_boxplot() +
+   scale_fill_manual(values=colors) +
+   geom_hline(aes(yintercept=0), colour= "black", linetype = "dashed") +
+   theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
+   labs(x = "Gene Name")
> g4 <- g3 + geom_point(data = tmp.df, aes(x = GENE_NAME, y = FC_RPL, size = 3, col = 'red'))
+   ggtitle(title)
> plot(g4)
```

compd-25168828.0 vs df3

