Creating graphics with ggplot

```
library("dplyr")
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
library("tidyr")
library("readr")
library("ggplot2")
```

Layer upon layer: Building our first plot

Every ggplot starts with a base layer. This is an empty layer from which we build up our graphic.

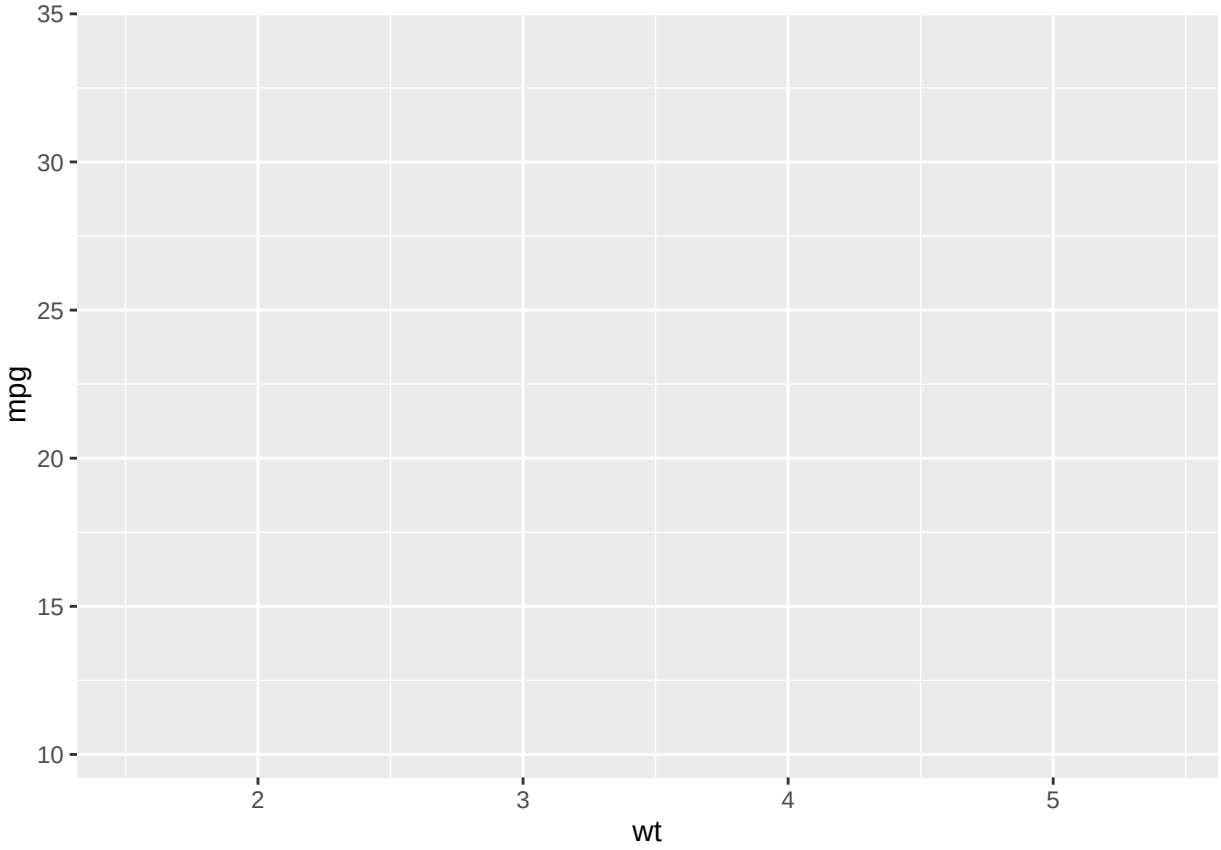
```
ggplot()
```


Now we need some data. Notice that our plot is still empty as we have yet to define how we want to plot the data.

```
ggplot(data = mtcars)
```

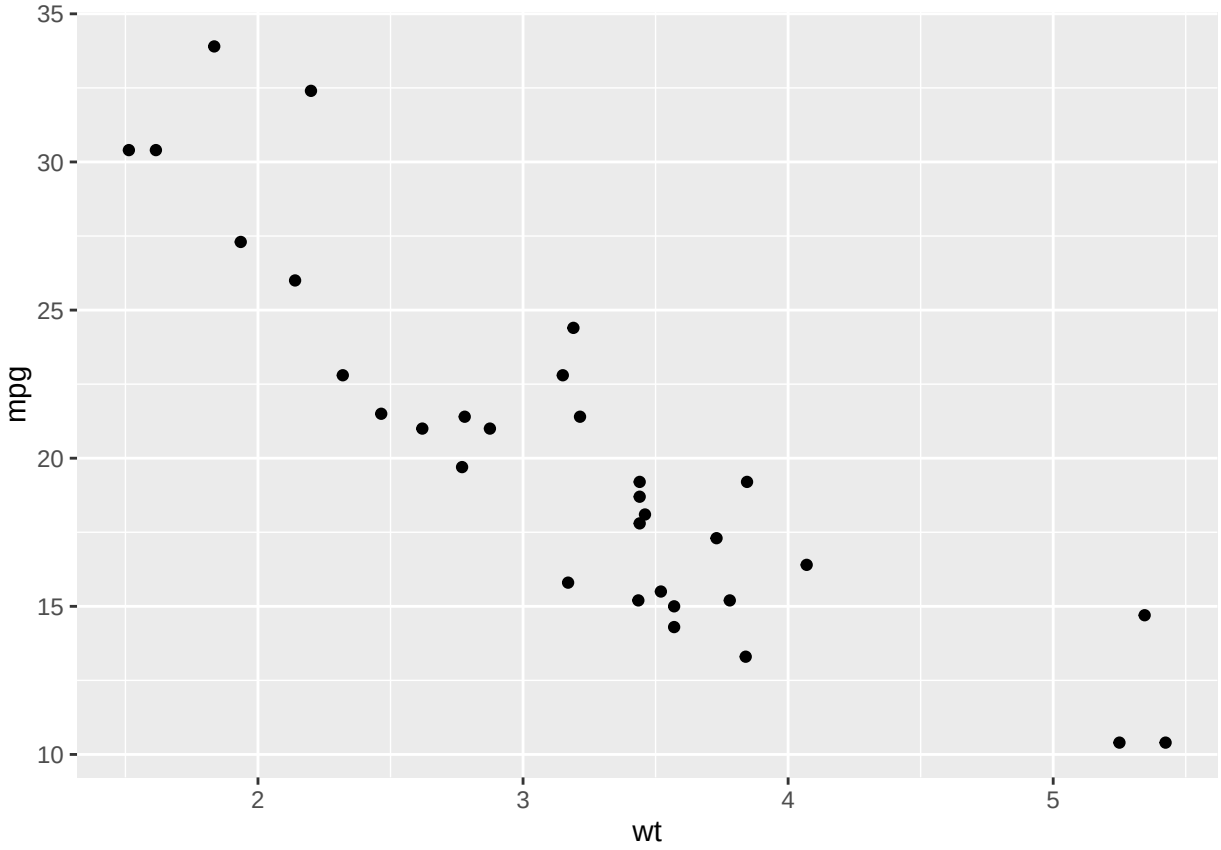

To construct a plot we must define our asthetics. This is how we connect the data to our scales, such as the x and y scales, the colour scale, the shape scale, etc. We do this using the `aes()` function.

```
ggplot(data = mtcars, aes(x = wt, y = mpg))
```



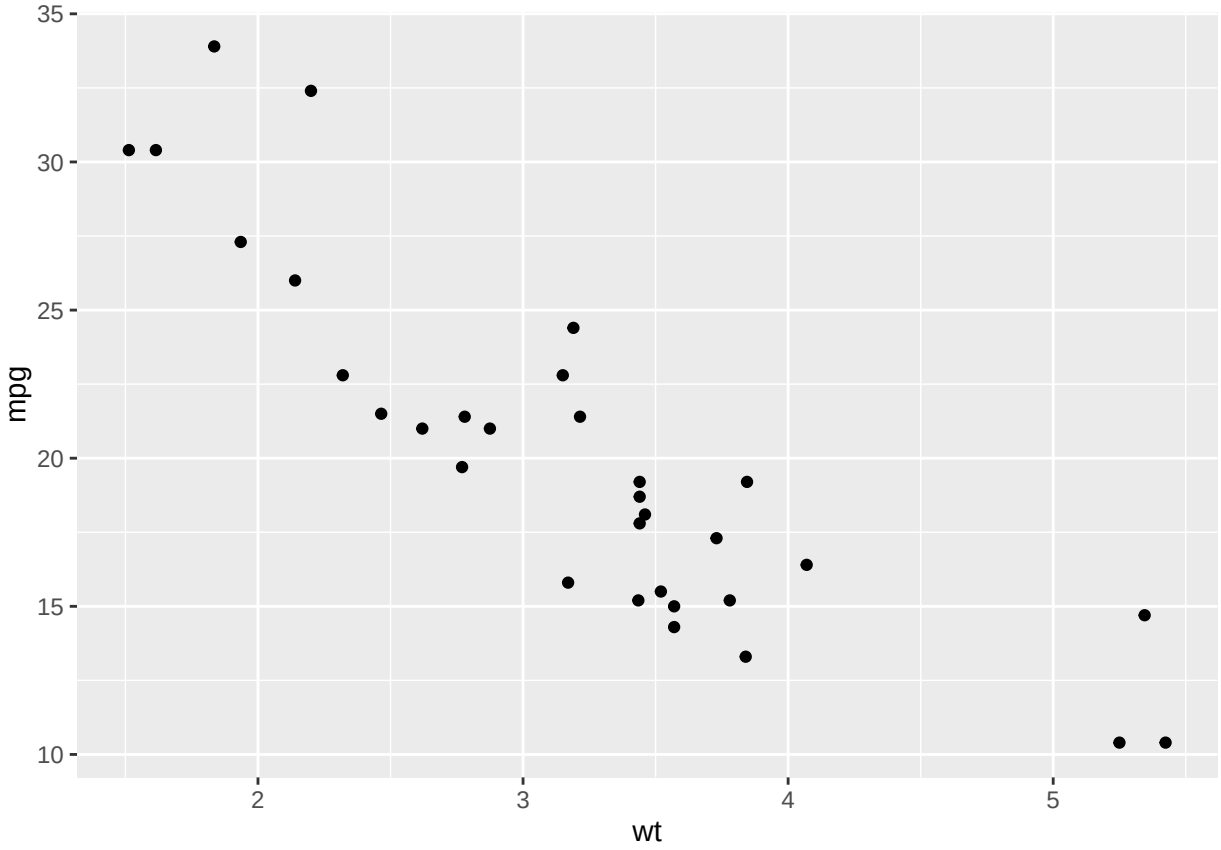
Notice now we have some scales on our plot but still no data is shown becuase ggplot still doesn't know the type of plot we'd like to use. To define what we want to plot we use `geom_XXX` functions. In this case we want points.

```
ggplot(data = mtcars, aes(x = wt, y = mpg)) + geom_point()
```



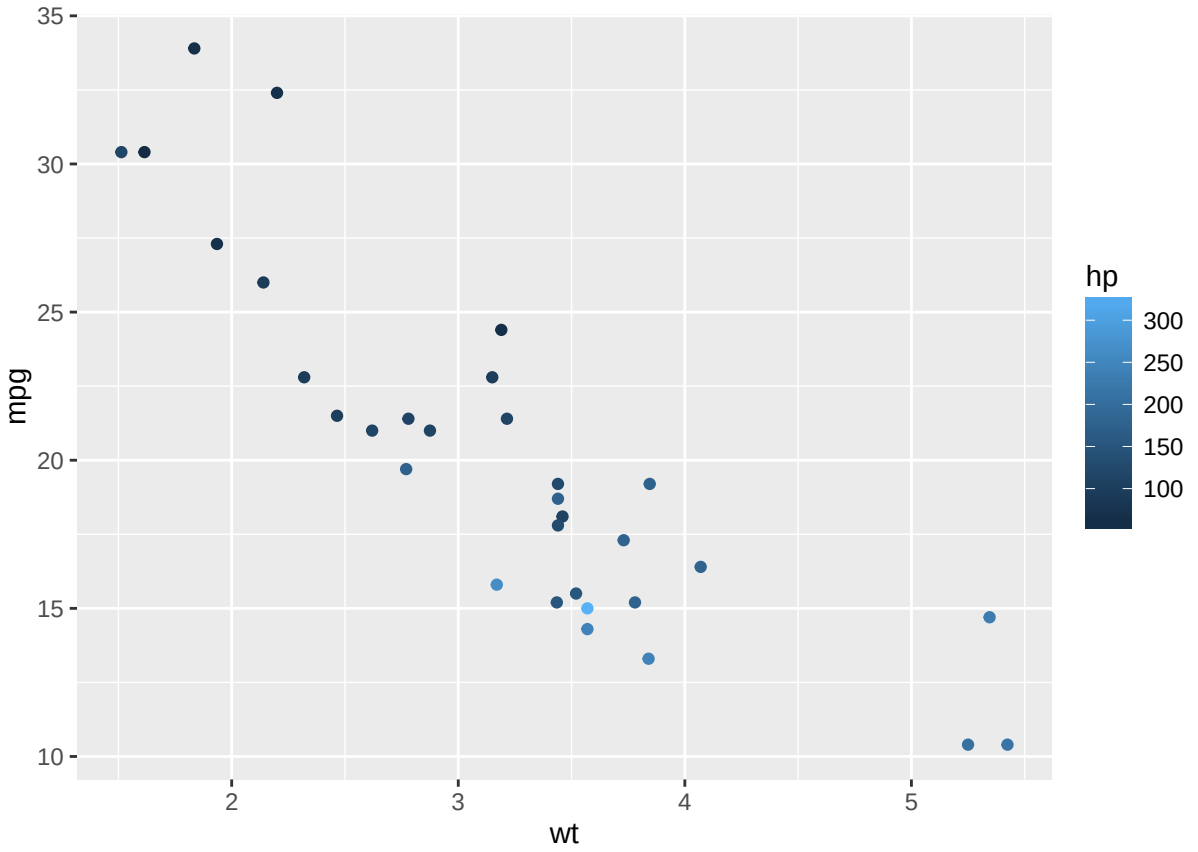
Here we use + to "add" the plot layers together. This notation allows us to create plot objects which can be modified.

```
p = ggplot(data = mtcars, aes(x = wt, y = mpg))
p + geom_point()
```

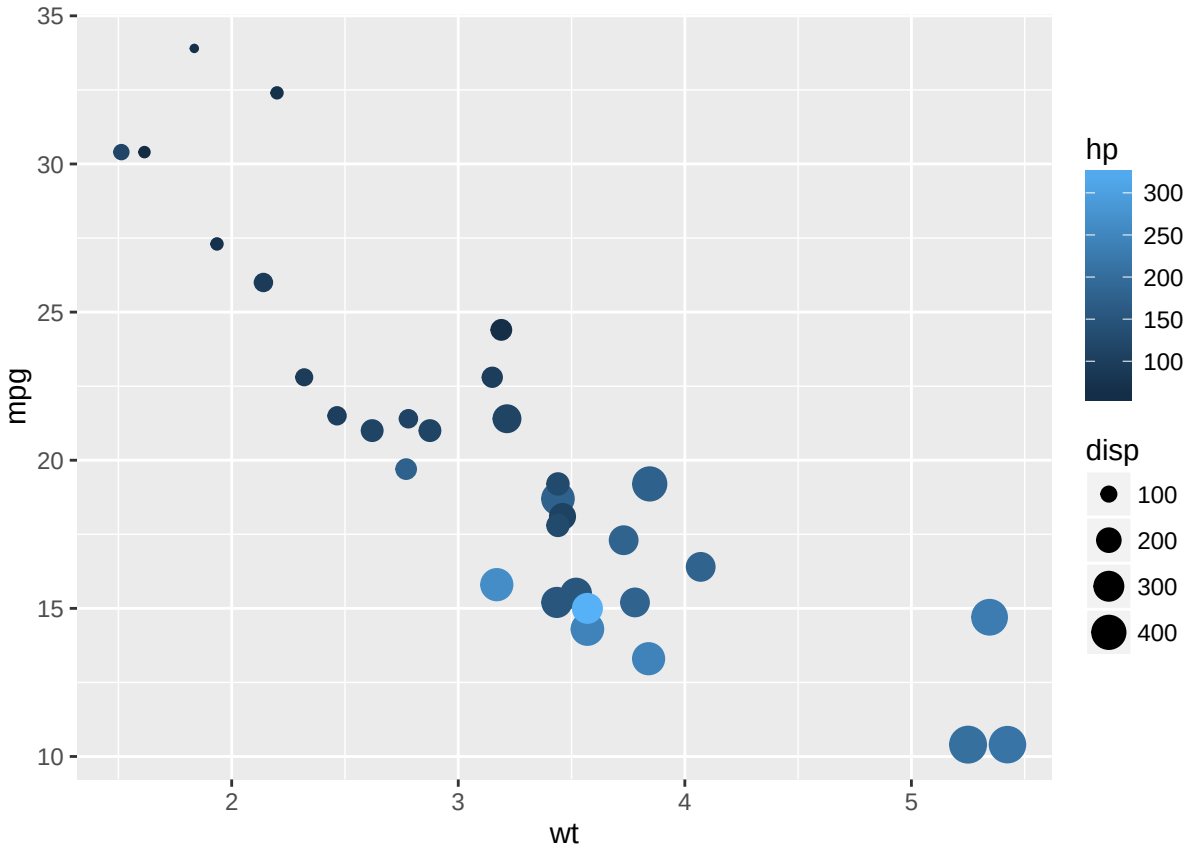


We can map other variables in our data to other scales, such as colour and size

```
ggplot(data = mtcars, aes(x = wt, y = mpg)) +
    geom_point(aes(colour = hp))
```

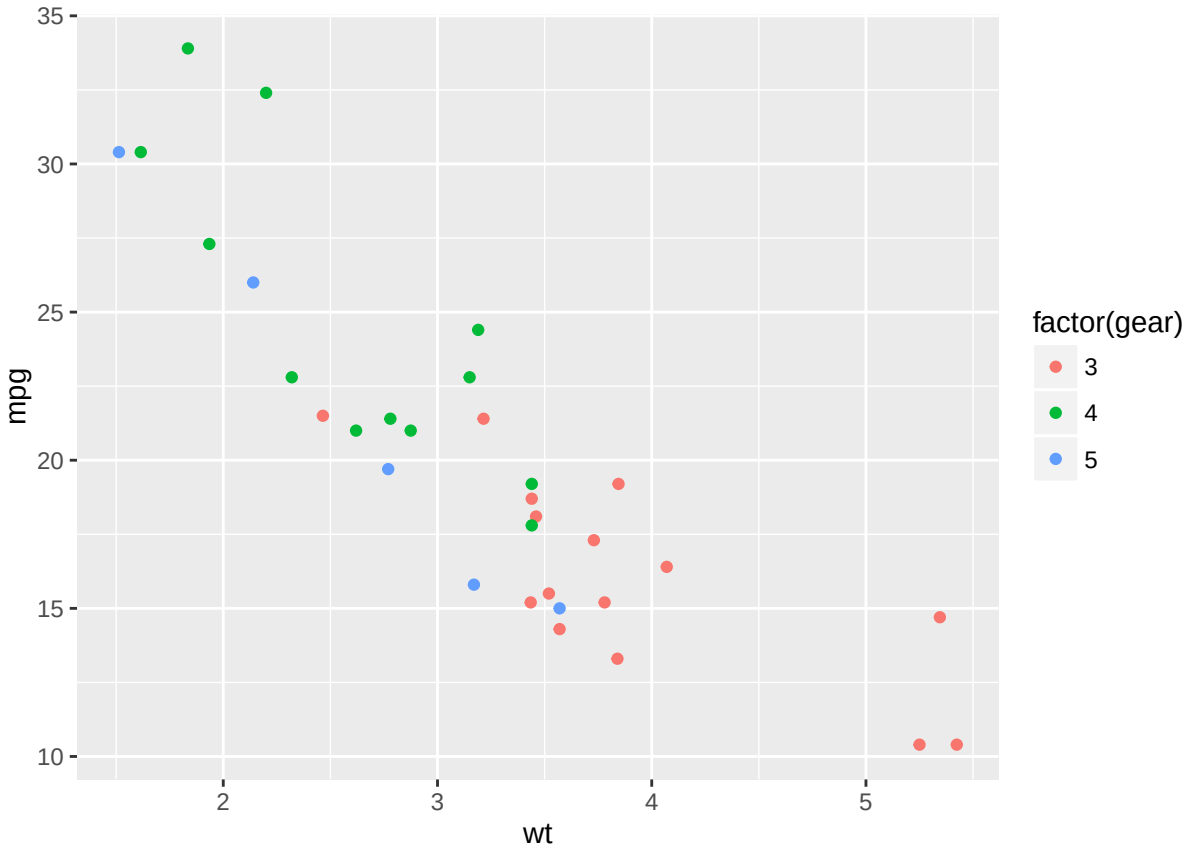



```
ggplot(data = mtcars, aes(x = wt, y = mpg)) +
    geom_point(aes(colour = hp, size = disp))
```



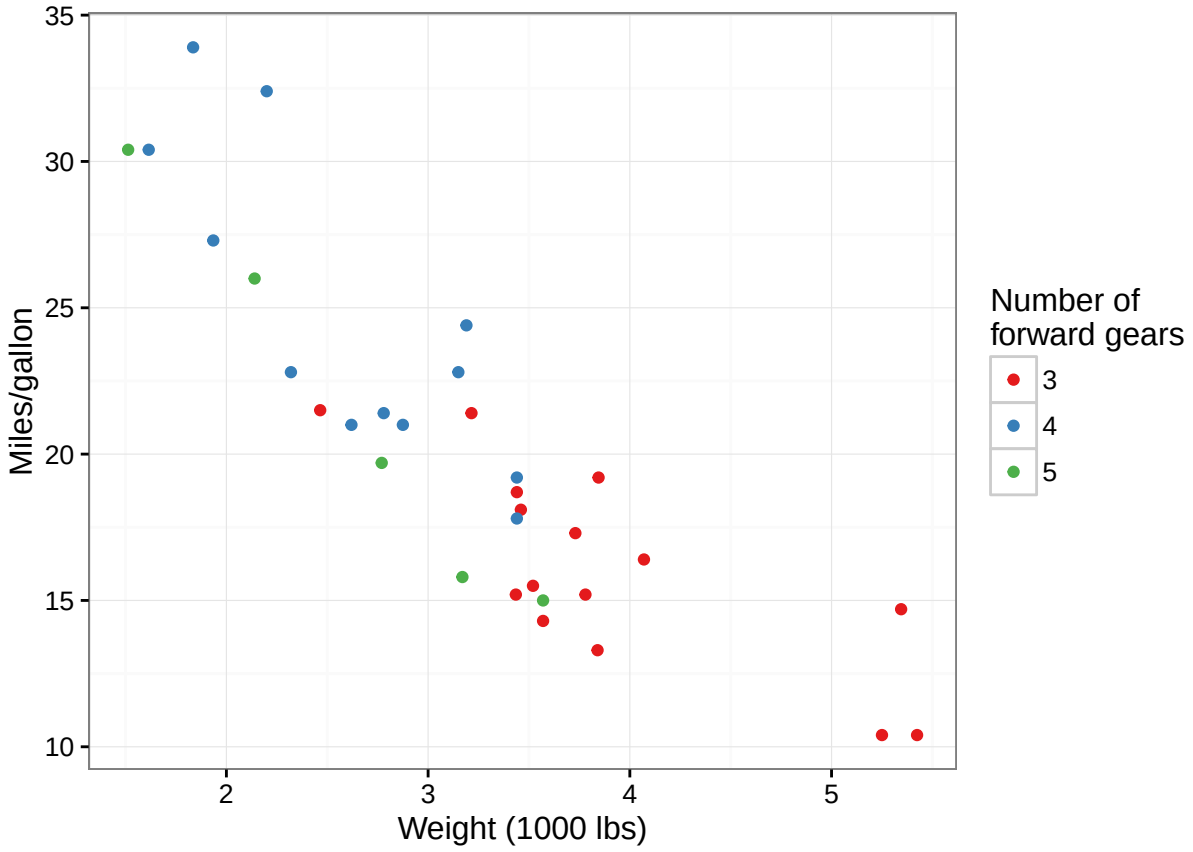
Mapping factors gives us discrete scales instead of continuous

```
ggplot(data = mtcars, aes(x = wt, y = mpg)) +
    geom_point(aes(colour = factor(gear)))
```



Ok, now we have a figure that we like, let's put on some finishing touches

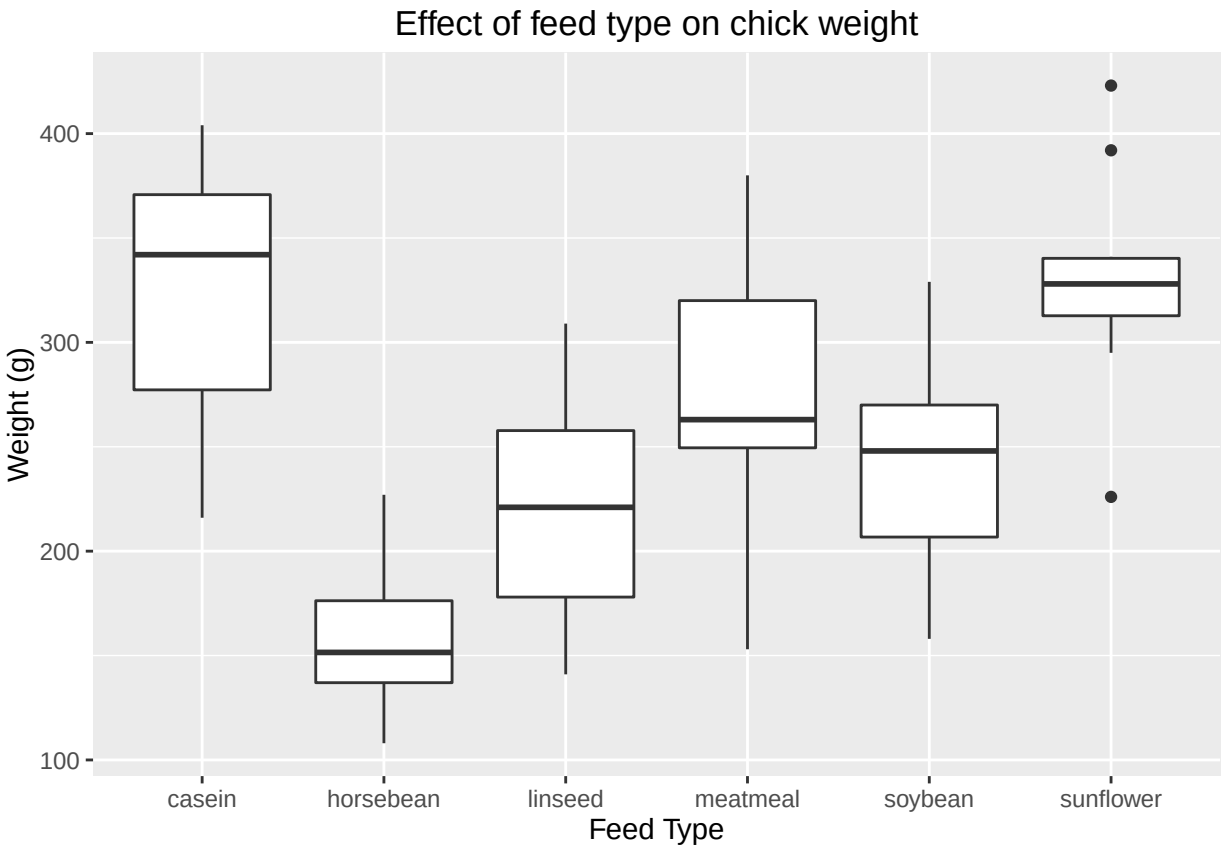
```
ggplot(mtcars, aes(x = wt, y = mpg)) +
    geom_point(aes(colour = factor(gear))) +
    theme_bw() +
    scale_colour_brewer("Number of\nforward gears", palette = "Set1") +
    labs(x = "Weight (1000 lbs)", y = "Miles/gallon")
```



A few more plot examples

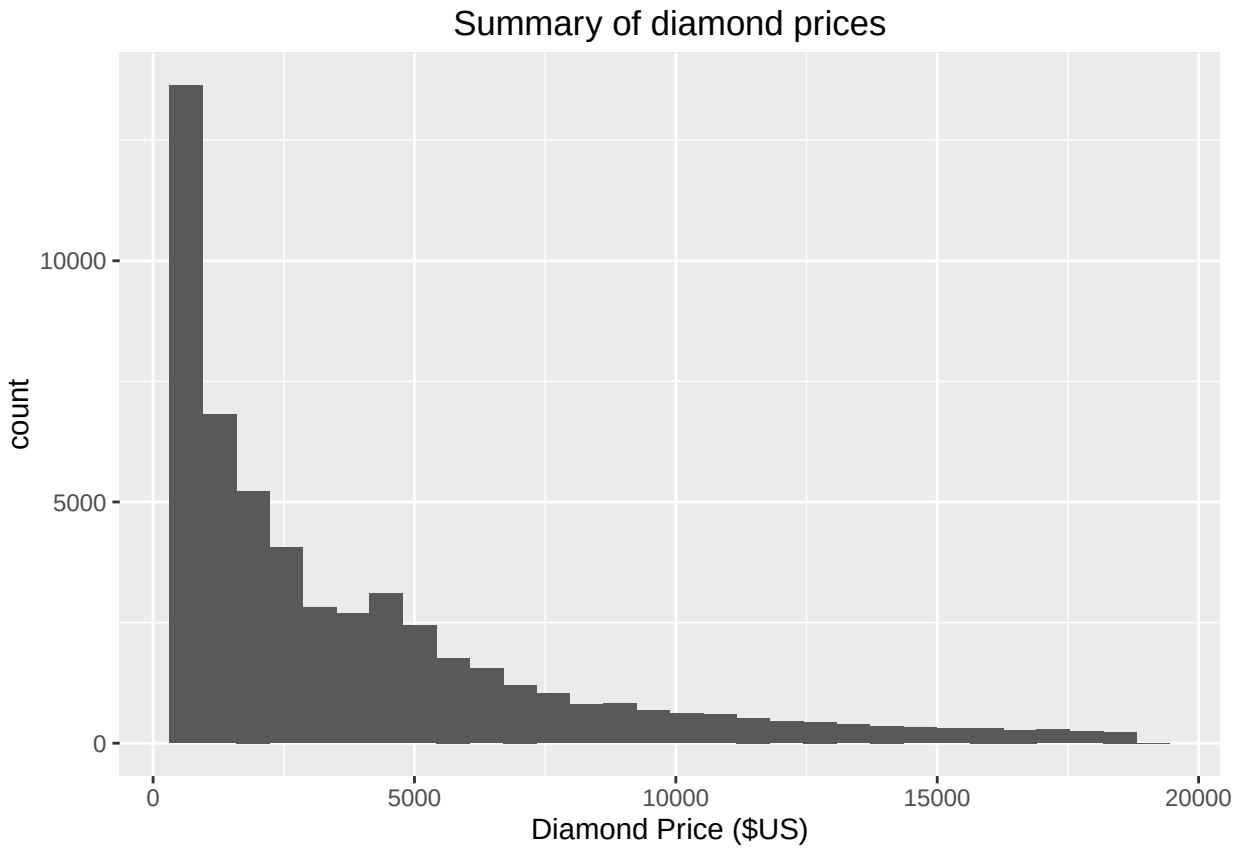
Boxplot

```
ggplot(chickwts, aes(x = feed, y = weight)) +
    geom_boxplot() +
    labs(x = "Feed Type", y = "Weight (g)", title = "Effect of feed type on chick weight")
```



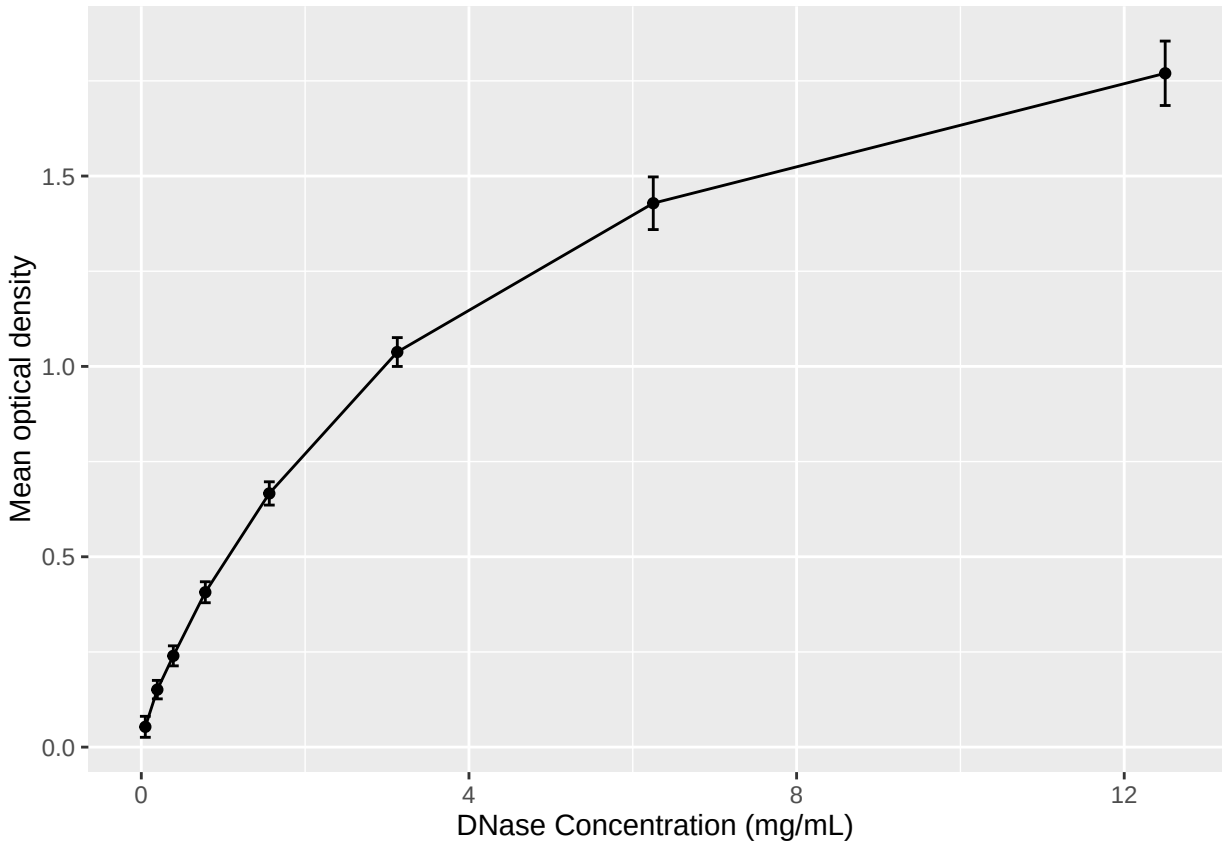
Histogram

```
ggplot(diamonds, aes(x = price)) +
    geom_histogram() +
    labs(x = "Diamond Price ($US)", title = "Summary of diamond prices")
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Line plot

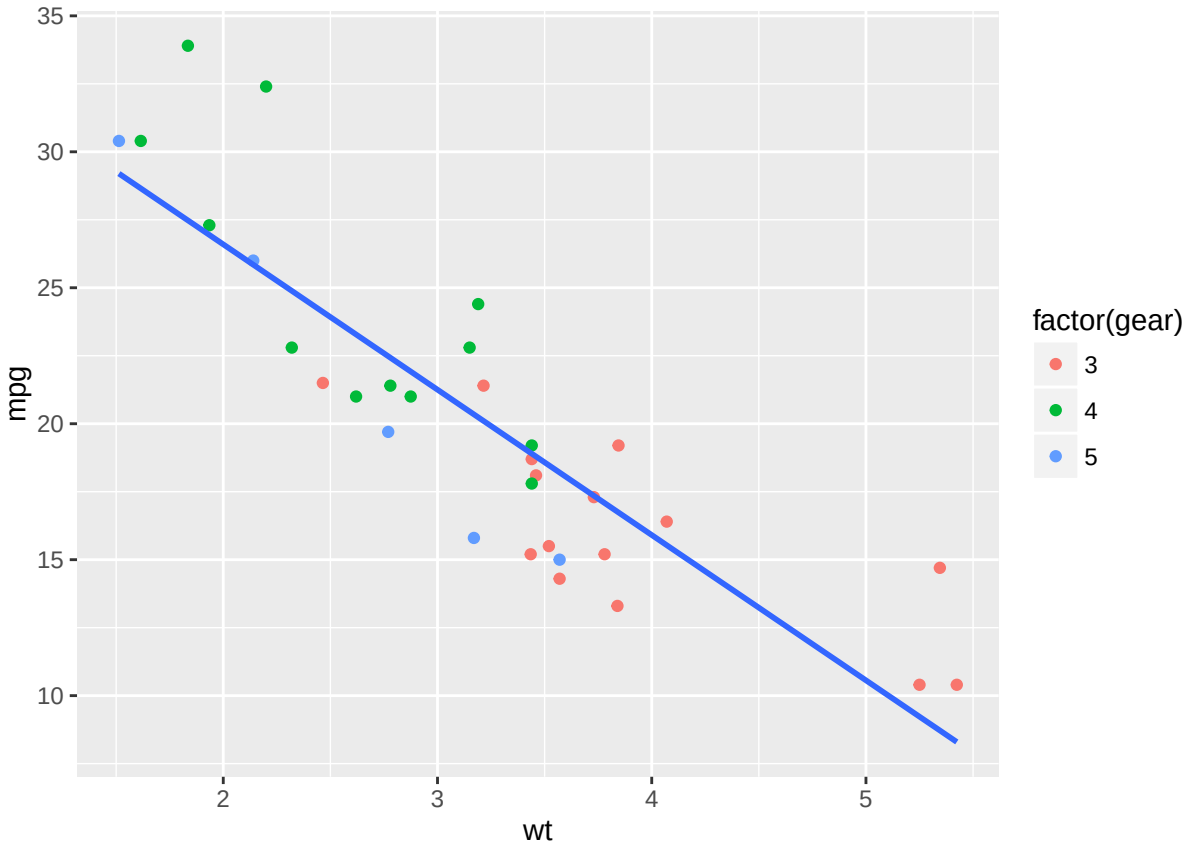
```
DNase %>% group_by(conc) %>% summarise(mean = mean(density), sd = sd(density)) %>%
    ggplot(aes(x = conc, y = mean)) +
    geom_errorbar(aes(ymax = mean + sd, ymin = mean - sd)) +
    geom_point() + geom_line() +
    labs(x = "DNase Concentration (mg/mL)", y = "Mean optical density")
```



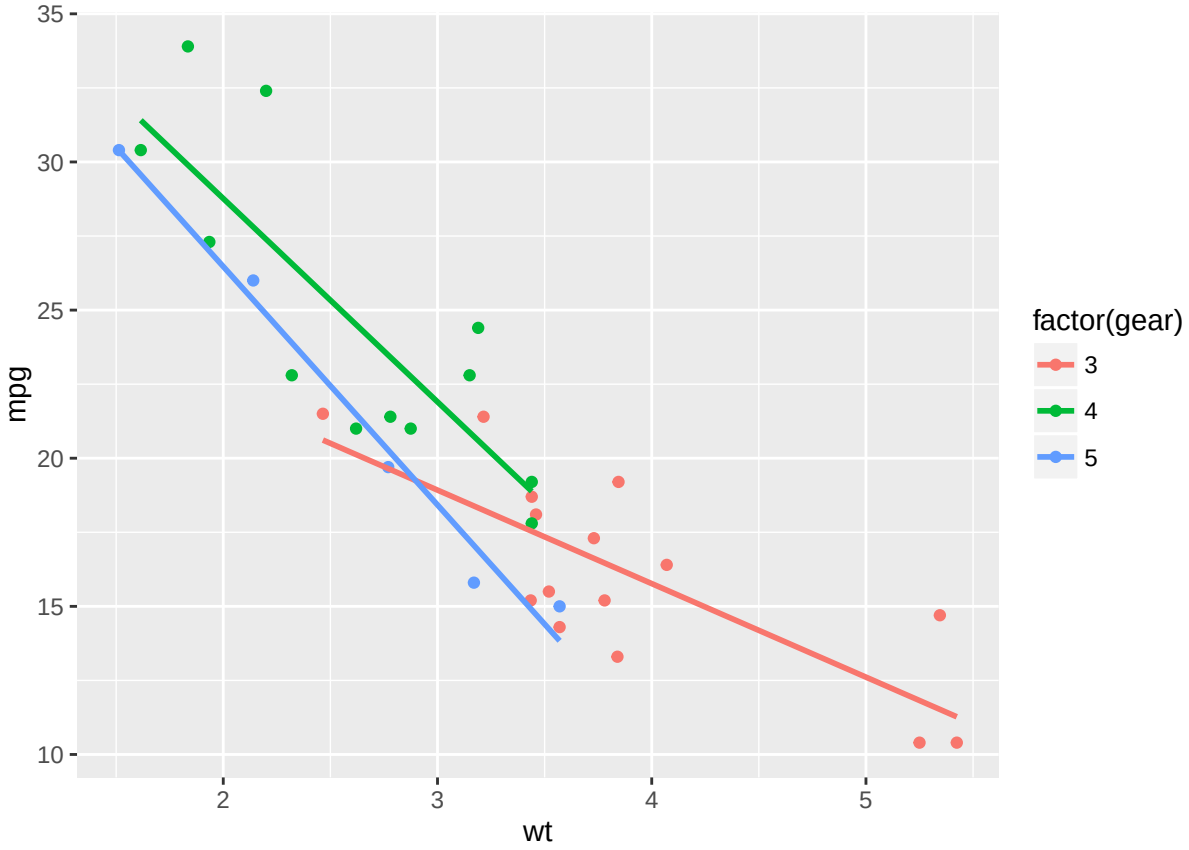
Details

Aesthetics can be defined for each geom individually or for all geoms in the `ggplot()` call.

```
ggplot(data = mtcars, aes(x = wt, y = mpg)) +
    geom_point(aes(colour = factor(gear))) +
    geom_smooth(method = "lm", se = FALSE)
```

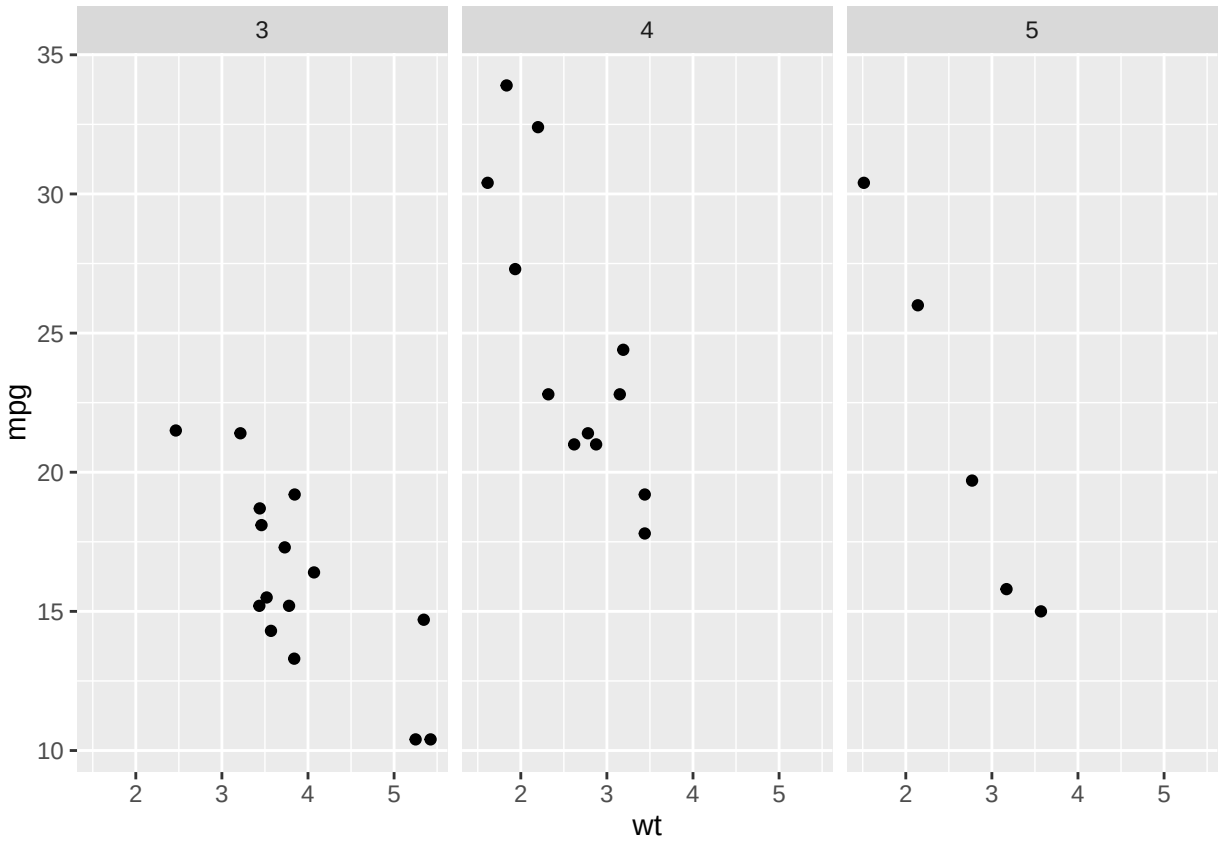



```
ggplot(data = mtcars, aes(x = wt, y = mpg, colour = factor(gear))) +
    geom_point() + geom_smooth(method = "lm", se = FALSE)
```

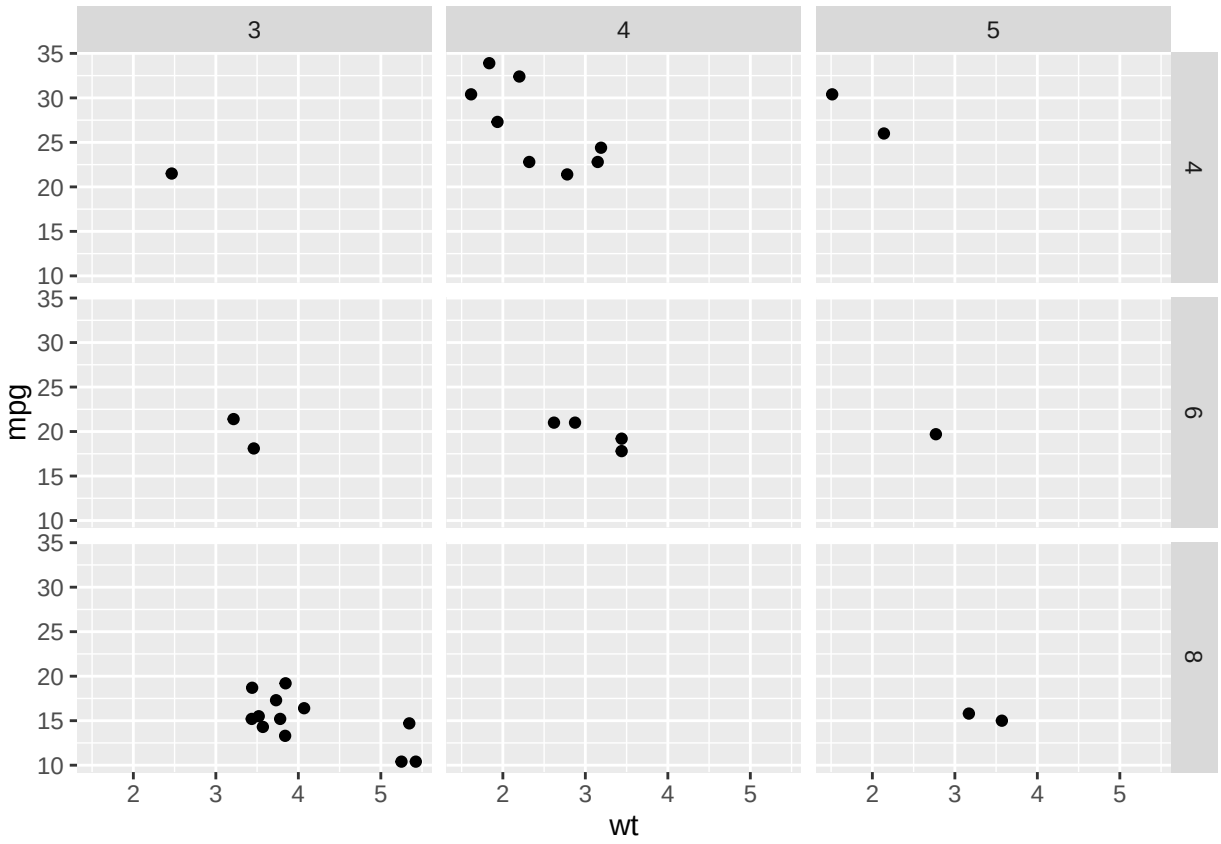


We use facets to split our data into small multiples

```
ggplot(data = mtcars, aes(x = wt, y = mpg)) +
    geom_point() +
    facet_wrap(~gear)
```




```
ggplot(data = mtcars, aes(x = wt, y = mpg)) +
    geom_point() +
    facet_grid(cyl ~ gear)
```



Exploring our data graphically

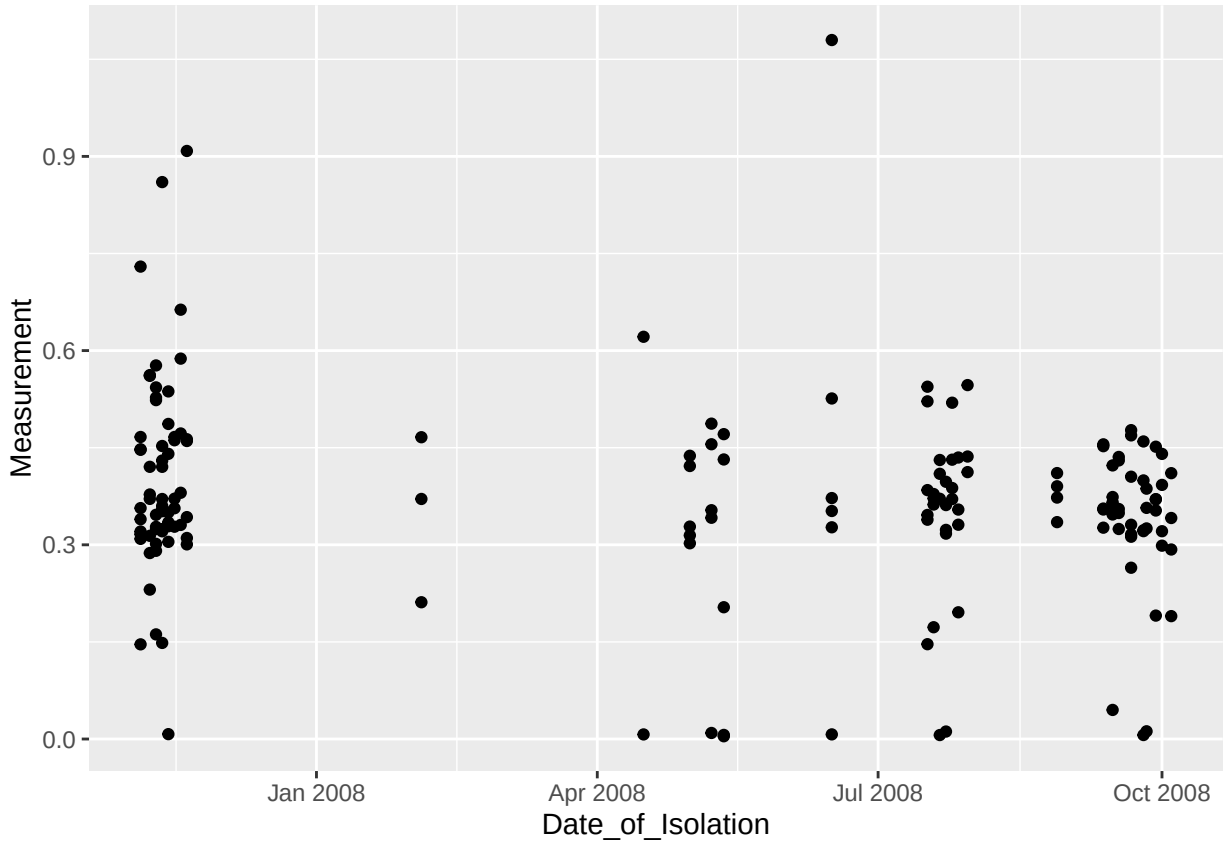
We'll use the pa phenotype dataset here and tidy it up a bit

```
pa = read_csv("../datasets/pa_phenotype_data.csv")
## Warning: Missing column names filled in: 'X1' [1]
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   X1 = col_integer(),
##   Strain_ID = col_integer(),
##   Event_num = col_character(),
##   Clinical_Status = col_character(),
##   Date_of_Isolation = col_date(format = ""),
##   CFU = col_integer(),
##   Morphotype = col_integer(),
##   Mucoid = col_character(),
##   SXT = col_integer(),
##   CAZ = col_integer(),
##   CIP = col_integer(),
##   CT = col_integer(),
##   TOB = col_integer(),
##   Swarm = col_integer()
## )
## See spec(...) for full column specifications.
pa_tidy = pa %>% select(-X1) %>%
```

```
    gather(Phenotype, Measurement, -Strain_ID, -Event_num, -Clinical_Status,
                -Date_of_Isolation, -Morphotype, -Mucoid)
```

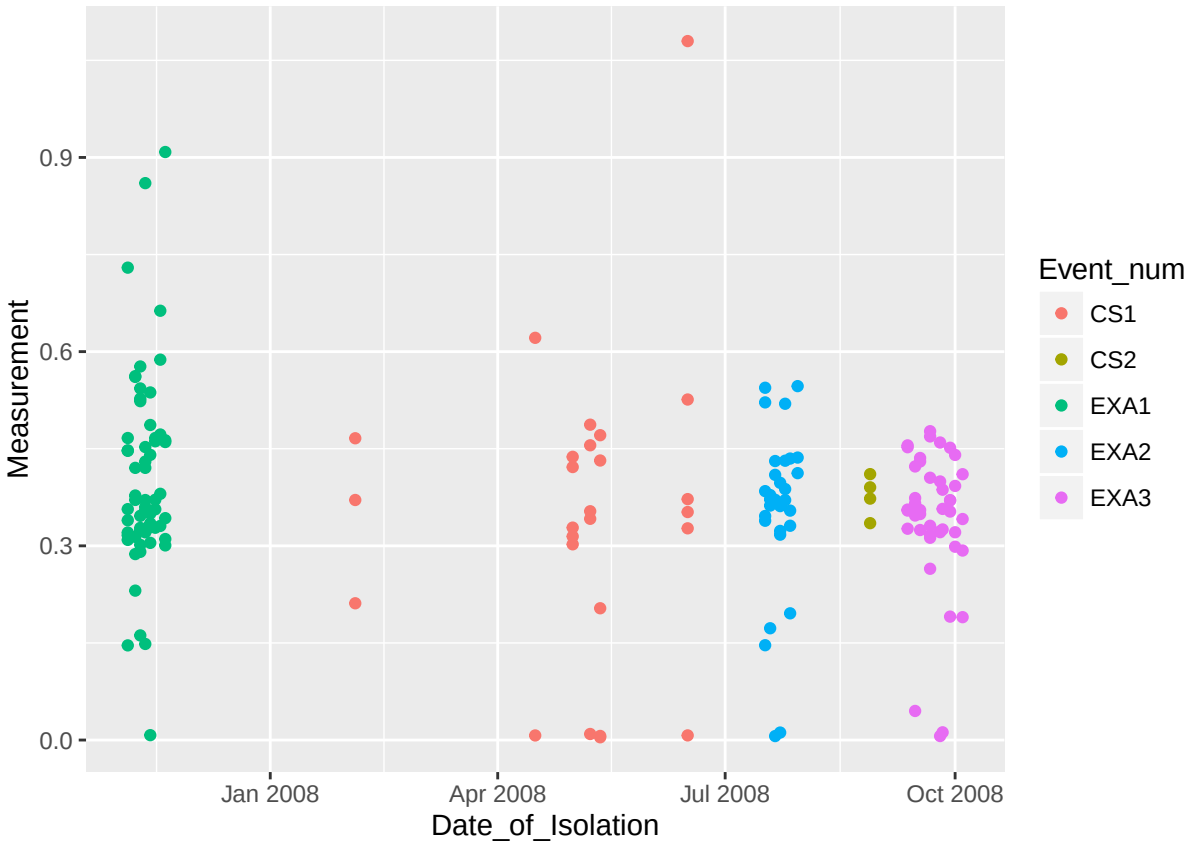
For the growth rate in LB lets look at the change over time and see if there are differences between clinical states

```
pa_tidy %>% filter(Phenotype == "LB") %>%
    ggplot(aes(x = Date_of_Isolation, y = Measurement)) +
    geom_point()
```



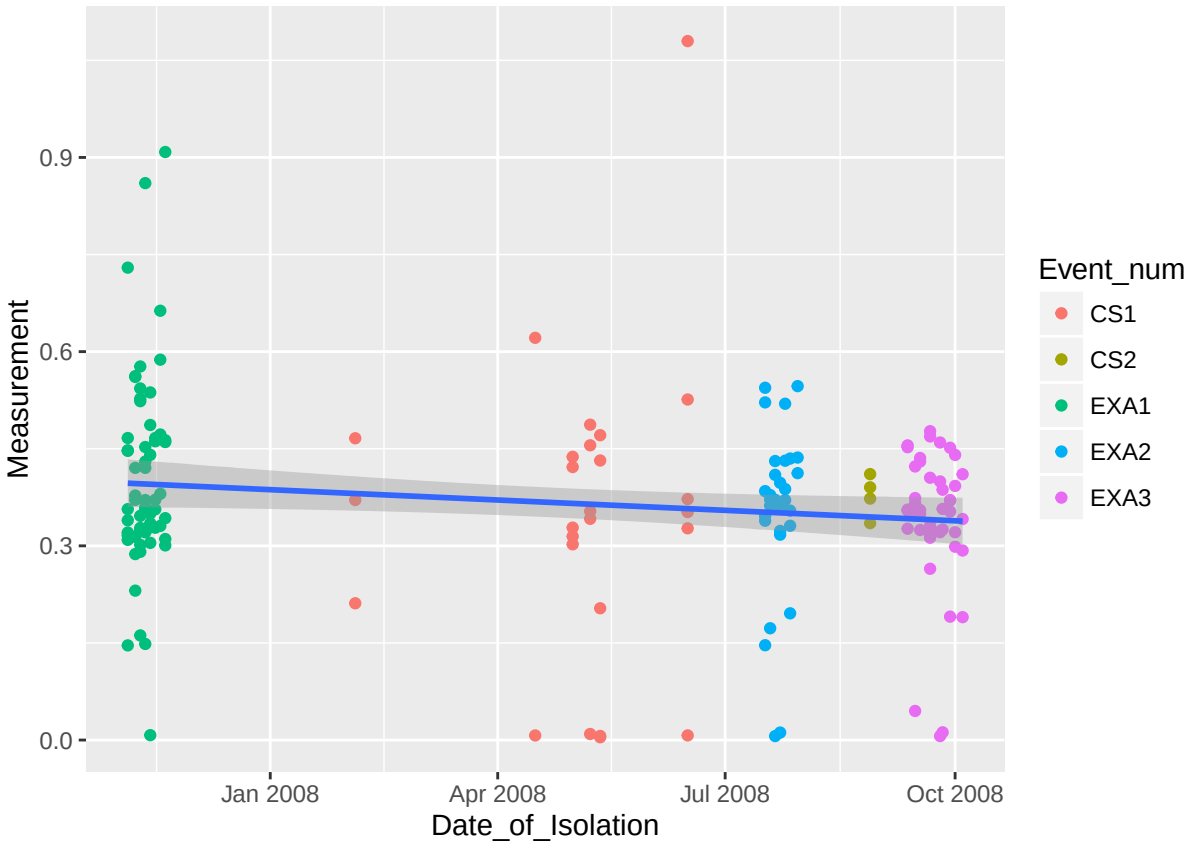
Here we can see that the isolates are collected in bursts over short periods of time. Presumably these correspond to the excacerbation events.

```
lb_plot = pa_tidy %>% filter(Phenotype == "LB") %>%
    ggplot(aes(x = Date_of_Isolation, y = Measurement)) +
    geom_point(aes(colour = Event_num))
lb_plot
```



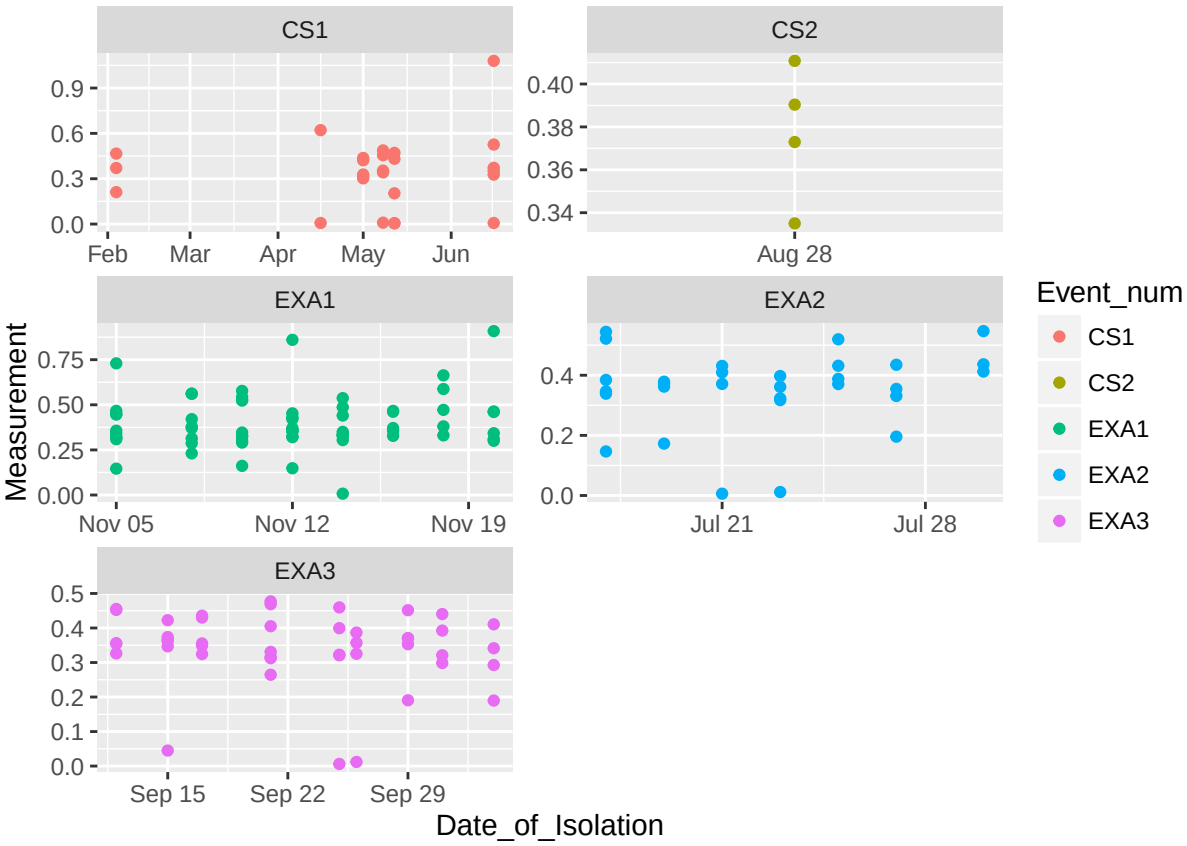
We can add a trend line to see if there is a decrease.

```
lb_plot + geom_smooth(method = "lm")
```



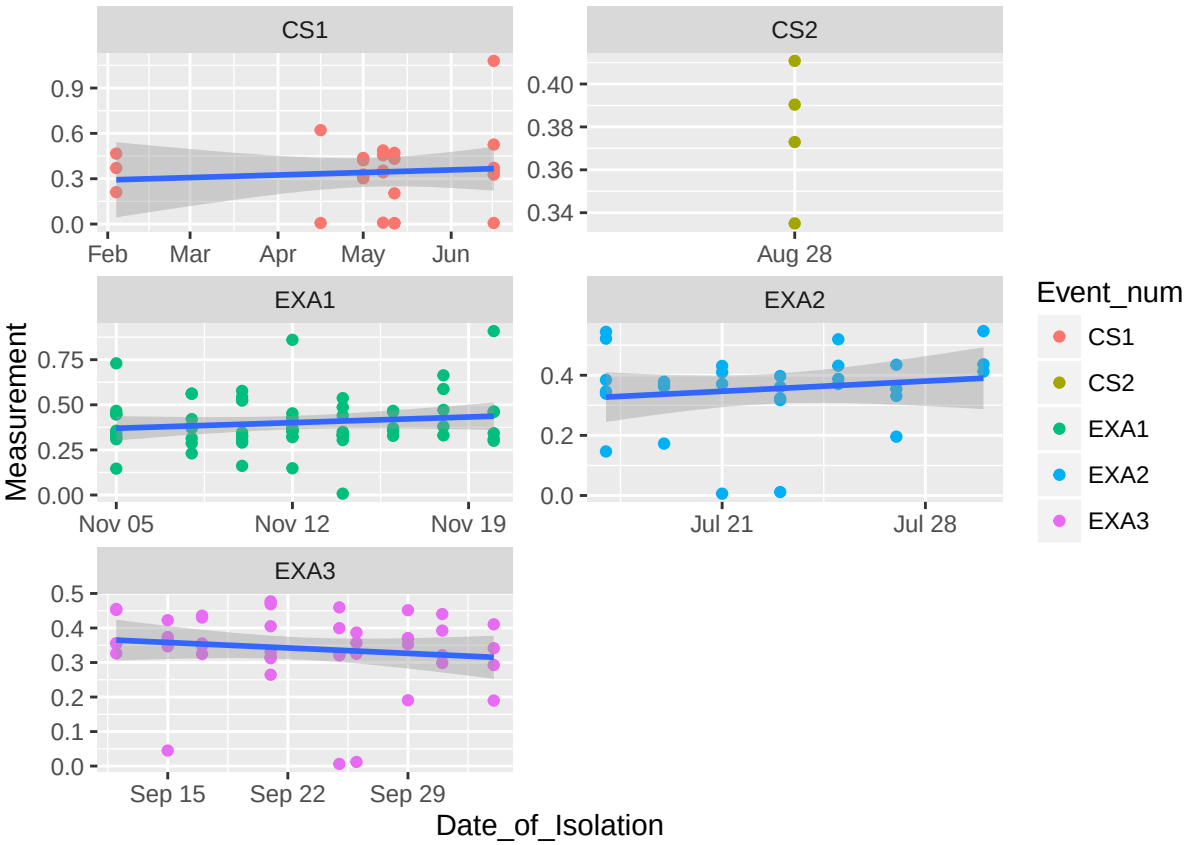
To get a closer look at each clinical event we can use facets

```
lb_plot + facet_wrap(~Event_num, scales = "free", ncol = 2)
```



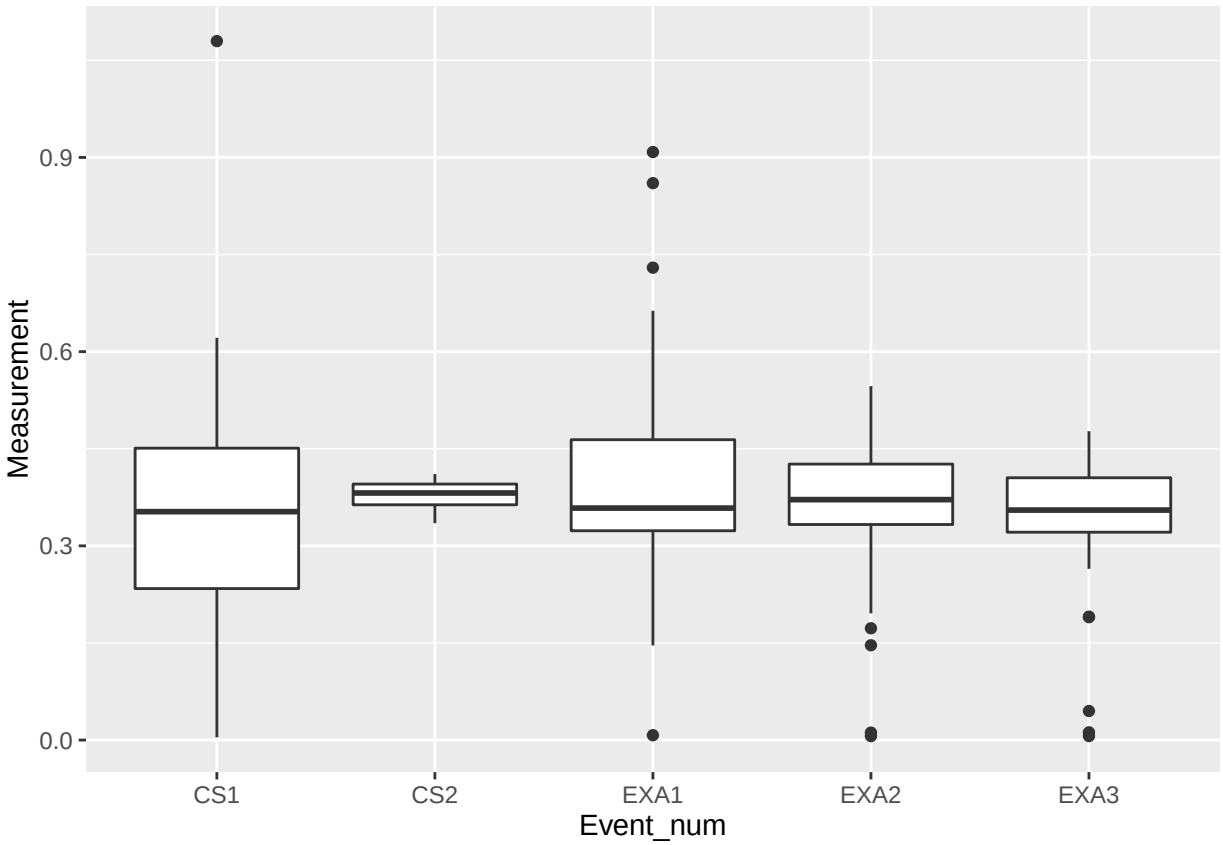
Now add the trend line

```
lb_plot + facet_wrap(~Event_num, scales = "free", ncol = 2) + geom_smooth(method = "lm")
```



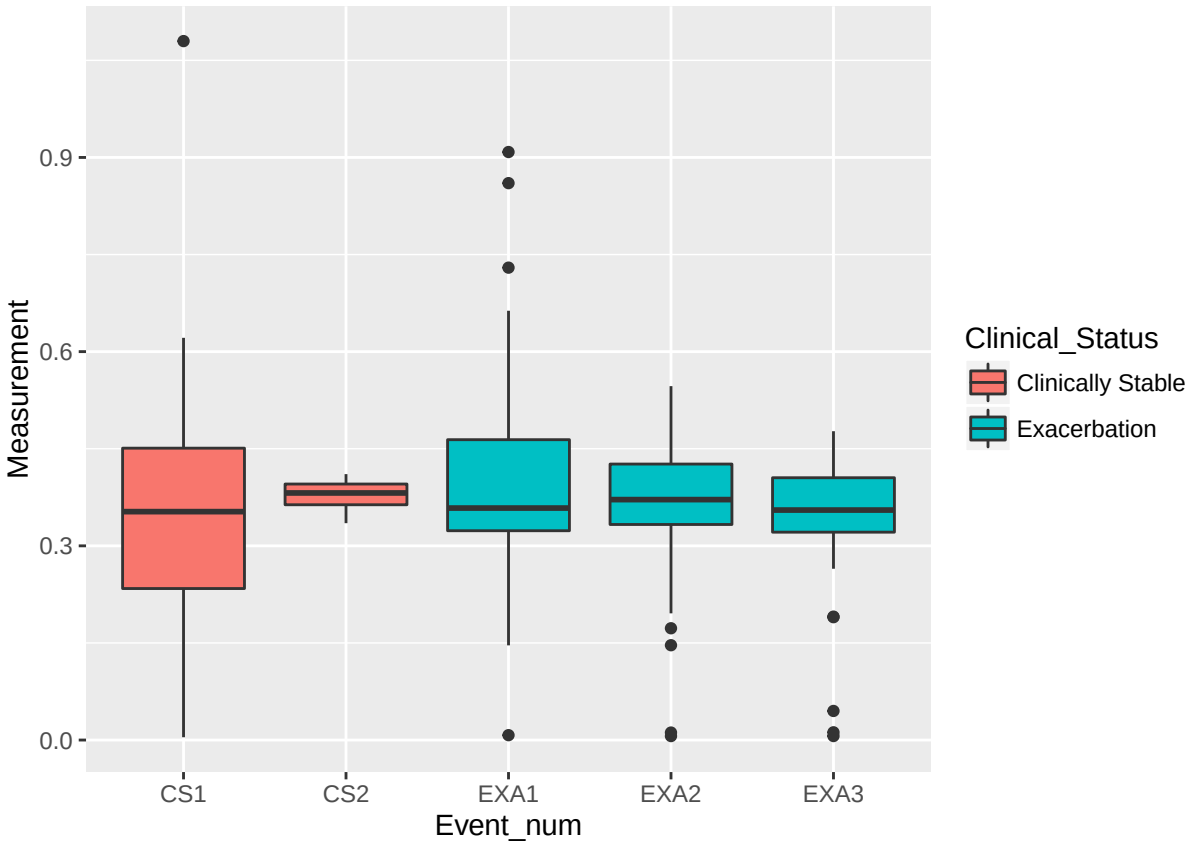
An alternative view would be to look at the clincal events as boxplots

```
pa_tidy %>% filter(Phenotype == "LB") %>%
    ggplot(aes(x = Event_num, y = Measurement)) +
    geom_boxplot()
```



We can add a colour variable to help view the plot.

```
pa_tidy %>% filter(Phenotype == "LB") %>%
    ggplot(aes(x = Event_num, y = Measurement)) +
    geom_boxplot(aes(fill = Clinical_Status))
```


Measurement
0.9
0.6
0.3
0.0
CS1
CS2
EXA1
EXA2
EXA3
Event_num
Clinical_Status
Clinically Stable
Exacerbation