Graphics With Ggplot2: Day 2

October,20, 2017

Outline

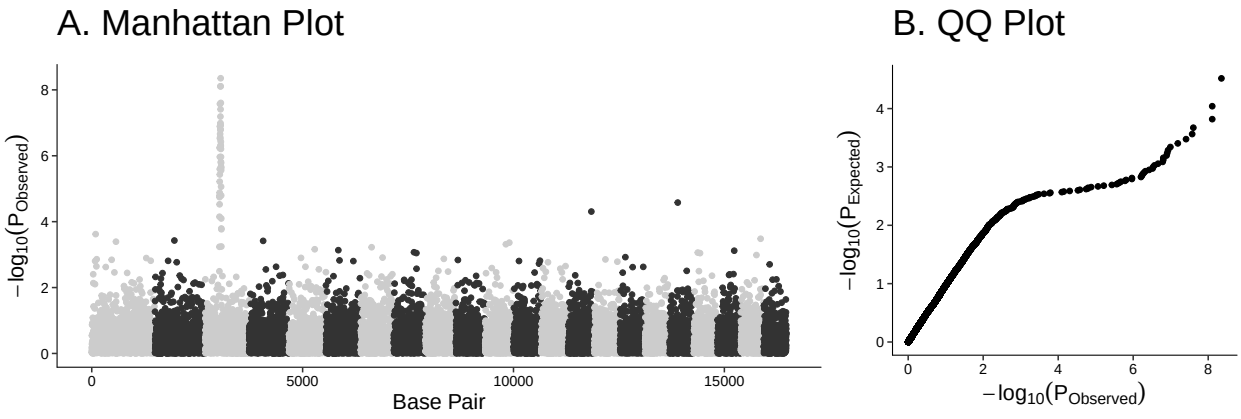
- Review exercise
- ggplot2 theming system
- gridExtra for arranging multiple graphs into one

Prerequesites

- ggplot2
- dplyr
- qqman
- gridExtra

Goal

Today we are going to (somewhat) replicate this graph.



Data Preparation

Let's get the data set from an R package.

```
install.packages("qqman")
library("qqman")
```

Here's the data set.

```
gwasResults <- tbl_df(gwasResults)
gwasResults
```

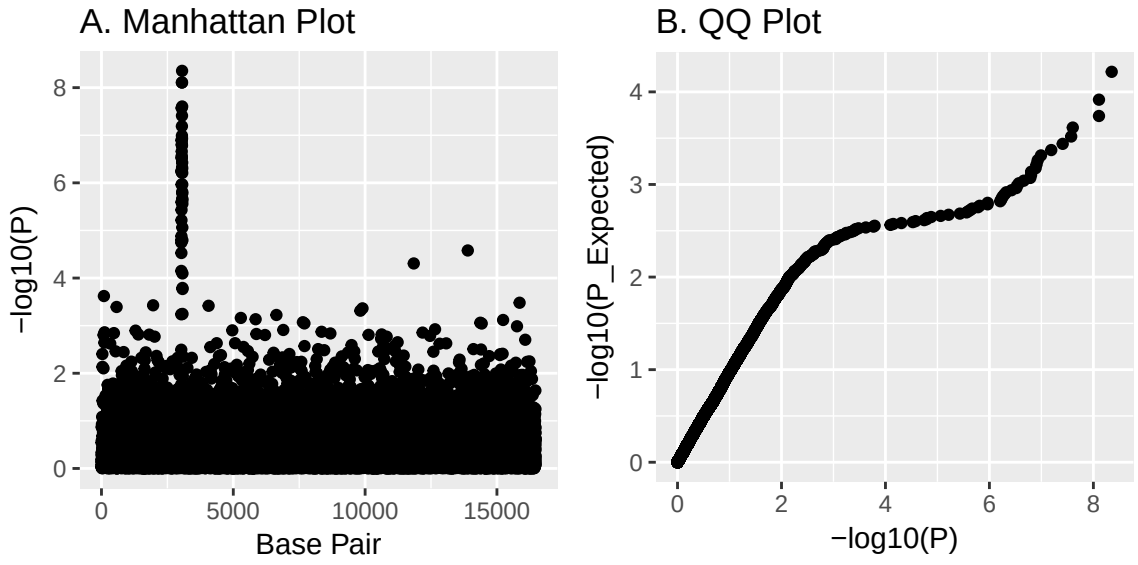
```
## # A tibble: 16,470 x 6
##      SNP   CHR    BP         P   POS P_Expected
##    <chr> <int> <int>     <dbl> <int>      <dbl>
##  1   rs1     1     1 0.9148060     1  0.9149970
##  2   rs2     1     2 0.9370754     2  0.9383121
##  3   rs3     1     3 0.2861395     3  0.2929569
##  4   rs4     1     4 0.8304476     4  0.8321190
##  5   rs5     1     5 0.6417455     5  0.6421372
##  6   rs6     1     6 0.5190959     6  0.5199757
##  7   rs7     1     7 0.7365883     7  0.7377656
##  8   rs8     1     8 0.1346666     8  0.1431694
##  9   rs9     1     9 0.6569923     9  0.6578628
## 10  rs10     1    10 0.7050648    10  0.7055252
## # ... with 16,460 more rows
```

Exercise

Can you re-create the following plots?

Figure A: The x-axis plots the position sorted by `CHR` and then `BP`. The y-axis plots the negative log-10 P-value.

Figure B: The expected P-value is calculated via the formula: `(rank(P)-0.5)/n`. The P-values in both the x- and the y-axis are in negative log-10 scale. read more about the expected P-value formula here: http://onlinestatbook.com/2/advanced_graphs/q-q_plots.html



Solution

```
gwasResults <- gwasResults %>% arrange(CHR, BP) %>% mutate(POS = 1:n())
myplot1 <- ggplot( data = gwasResults, aes(x = POS, y = -log10(P))) +
  geom_point( aes( color = factor(CHR %% 2)))+
  scale_colour_grey() +
```

```
  ggtitle("A. Manhattan Plot") +
  xlab("Base Pair")
myplot1

gwasResults <- gwasResults %>% mutate(P_Expected = (rank(P))/n())
myplot2 <- ggplot(data = gwasResults, aes(x = -log10(P), y = -log10(P_Expected))) +
              geom_point() +
              ggtitle("B. QQ Plot")
myplot2
```

Non-data elements

To clean the figure up for a publication we may need to change some of the non-data elements. For example,

- Fonts style
- Sizes
- Colour

ggplot2 themes

ggplot2 has provided us some default theme *functions*, which define the settings of a collection of theme elements for the purpose of creating a specific style of graphics production:

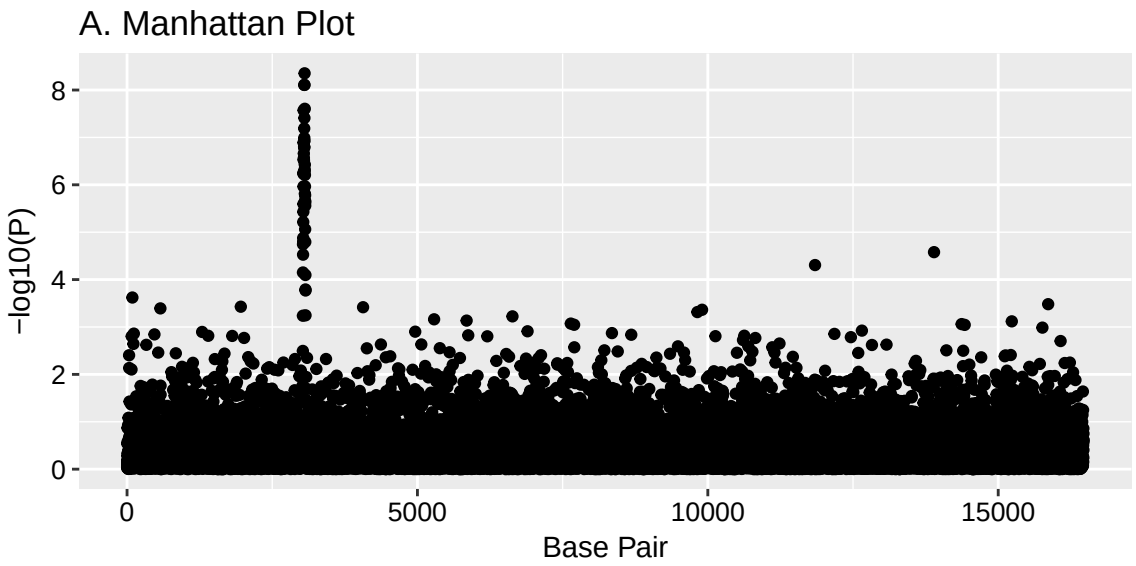
```
theme_grey(base_size = 11, base_family = "")
theme_gray(base_size = 11, base_family = "")
theme_bw(base_size = 12, base_family = "")
theme_linedraw(base_size = 12, base_family = "")
theme_light(base_size = 12, base_family = "")
theme_minimal(base_size = 12, base_family = "")
theme_classic(base_size = 12, base_family = "")
theme_dark(base_size = 12, base_family = "")
theme_void(base_size = 12, base_family = "")
```

See also here: http://docs.ggplot2.org/current/ggtheme.html

Customized themes

Here's an example of a customized theme:

```
myplot1 + theme(axis.text.x = element_text(colour = "black", size = 10),
                axis.text.y = element_text(colour = "black", size = 10))
```



See `?theme` for the full list of theme elements.

Theme element functions

Most theme elements have several properties that can be modified through a corresponding element function:

For controlling the drawing of labels and headings:

```
element_text(family = NULL, face = NULL, colour = NULL, size = NULL,
  hjust = NULL, vjust = NULL, angle = NULL, lineheight = NULL,
  color = NULL, margin = NULL, debug = NULL)
```

For drawing lines and segments such as axes and grid lines:

```
element_line(colour = NULL, size = NULL, linetype = NULL,
  lineend = NULL, color = NULL)
```

For drawing rectangles:

```
element_rect(fill = NULL, colour = NULL, size = NULL,
  linetype = NULL, color = NULL)
```

For drawing nothing:

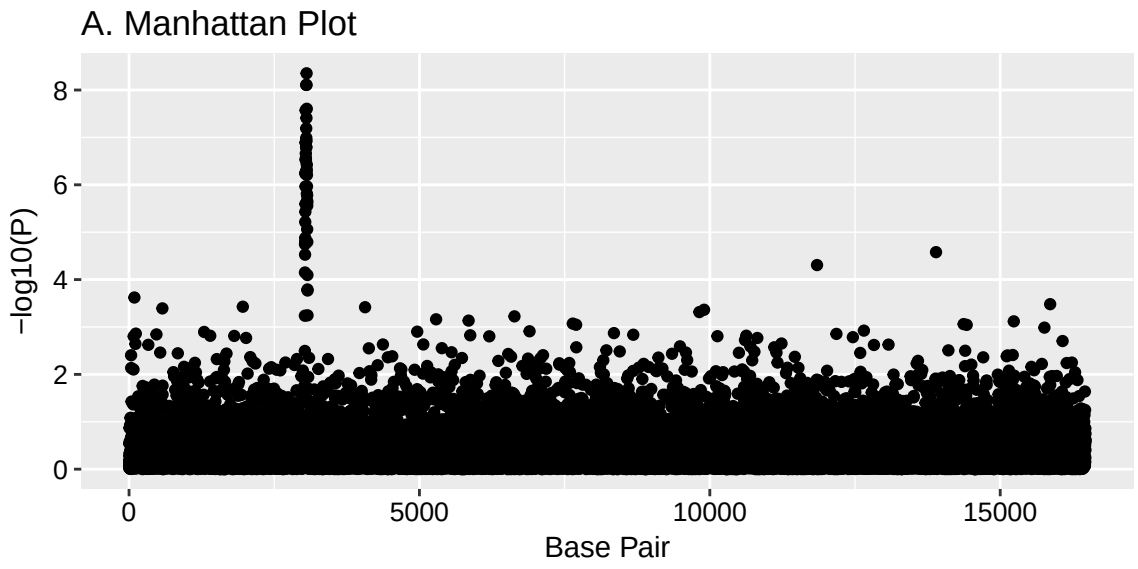
```
element_blank()
```

Exercise:

Can you add more to the theme function to:

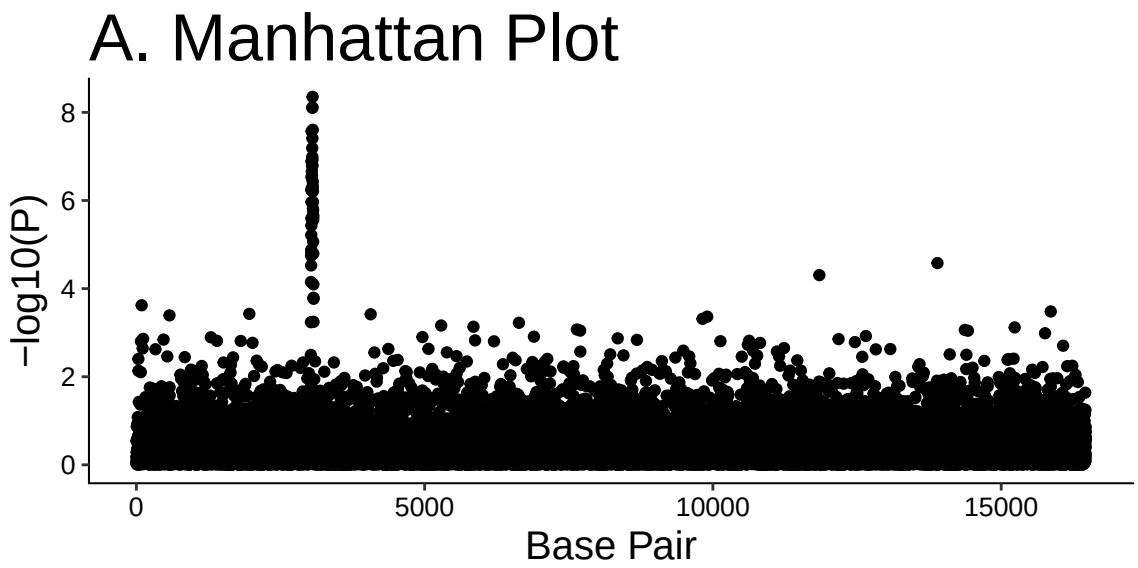
- Make the panel background white
- Make the plot title bigger and left justified
- Make the x-axis and y-axis line black
- Make the x-axis and y-axis title bigger

```
myplot1 + theme(axis.text.x = element_text(colour = "black", size = 10),
                axis.text.y = element_text(colour = "black", size = 10))
```



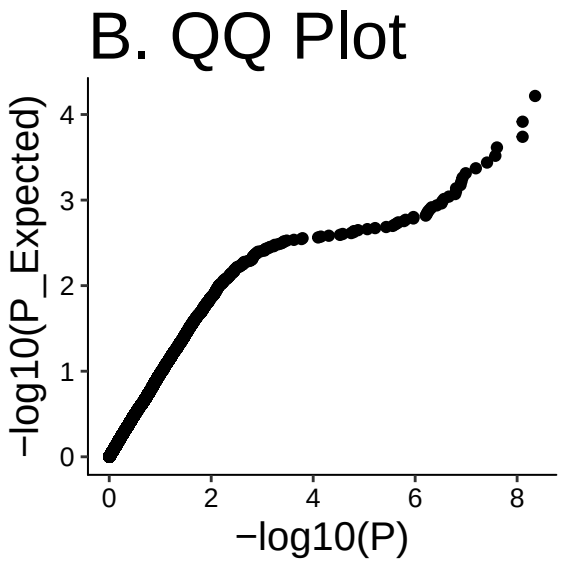
Solution:

```
mytheme <- theme(panel.background = element_rect(fill = "white"),
                 plot.title = element_text( colour = "black", size = 25, hjust = 0),
                 axis.line.x = element_line(colour = "black"),
                 axis.line.y = element_line(colour = "black"),
                 axis.text.x = element_text( colour = "black", size = 10),
                 axis.text.y = element_text( colour = "black", size = 10),
                 axis.title.x = element_text( colour = "black", size = 15),
                 axis.title.y = element_text( colour = "black", size = 15) )
myplot1 + mytheme
```



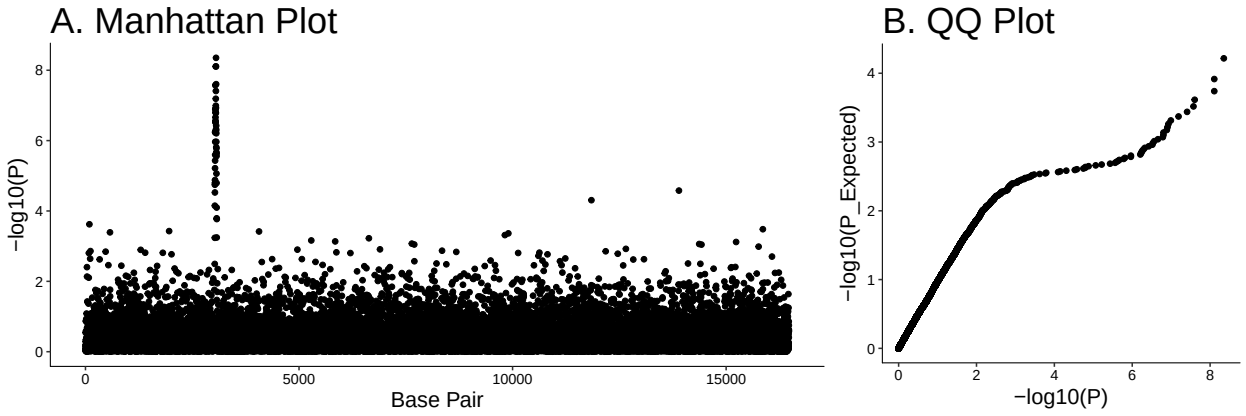
Customized themes can be saved and reused

```
myplot2 + mytheme
```



Arranging multiple graphical objects on a page

Sometimes we want to put two different graphical objects into one.



gridExtra

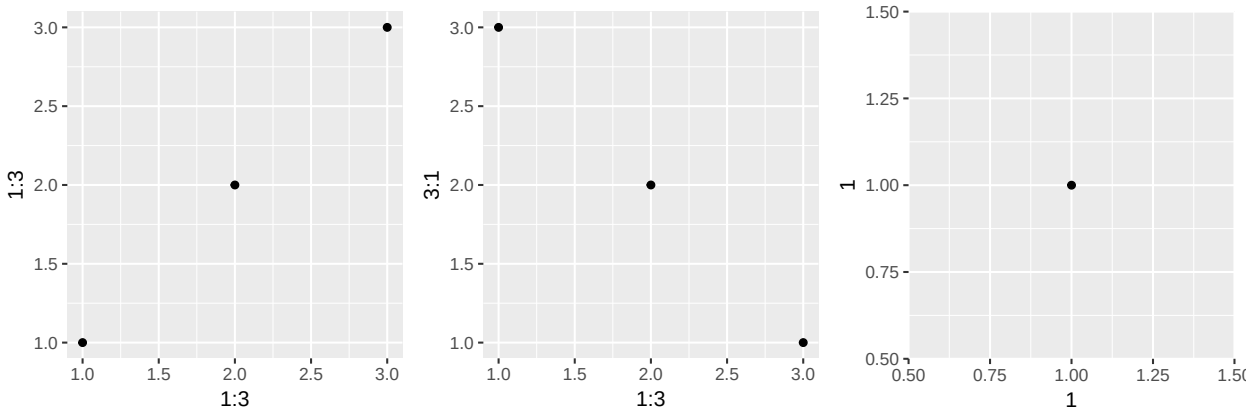
`gridExtra` package provides high-level functions to arrange multiple graphical objects (grobs) into one. It works well with grobs from `ggplot2`.

Let's install the package.

```
install.packages("gridExtra")
library("gridExtra")
```

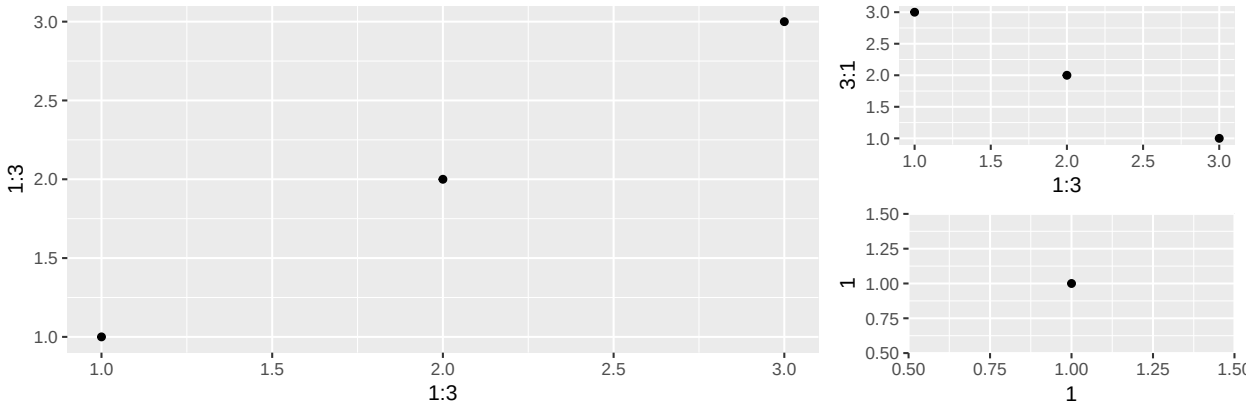
Basic usage

```
plt1 <- qplot(x=1:3, y=1:3)
plt2 <- qplot(x=1:3, y=3:1)
plt3 <- qplot(x=1, y=1)

grid.arrange(plt1, plt2, plt3, nrow=1)
```

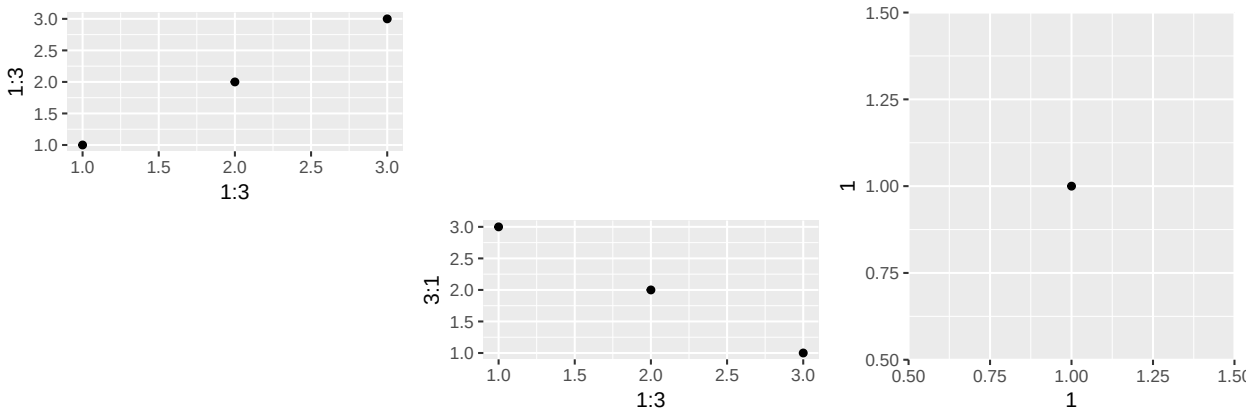


More complex layout

```
plt_list <- list(plt1, plt2, plt3)
plt_layout <- rbind(c(1, 1, 2),
                    c(1, 1, 3))

grid.arrange(grobs = plt_list, layout_matrix = plt_layout)
```

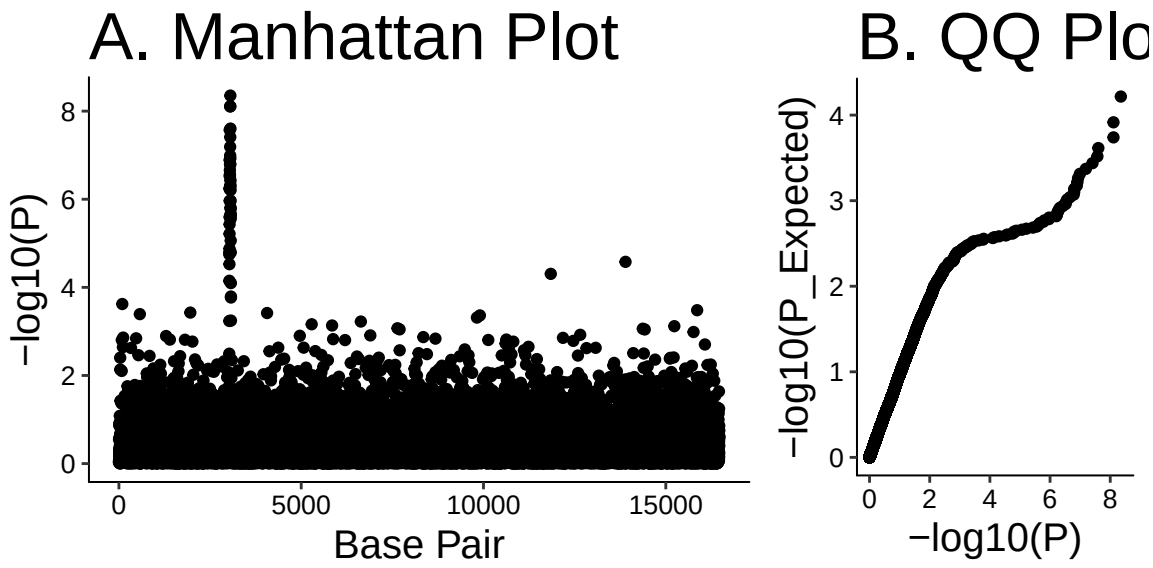



```
plt_layout <- rbind(c(1,  NA, 3),
                    c(NA, 2,  3))

grid.arrange(grobs = plt_list, layout_matrix = plt_layout)
```



Final touch

Can you now re-create this?



Solution:

```
myplot_list <- list( myplot1 + mytheme, myplot2 + mytheme)

grid.arrange(grobs = myplot_list, layout_matrix = matrix(c(1,1,2), nrow=1))
```

Where can we go from here?

R graphics cookbook (first edition)

http://amzn.to/2fYQIoC

ggplot2 book (second edition)

http://amzn.to/2eTbZvE

ggplot2 mailing list

http://groups.google.com/group/ggplot2

stackoverflow

http://stackoverflow.com/tags/ggplot2