Adding to the story: Annotations, maps and interactions

Abhijit Dasgupta, PhD

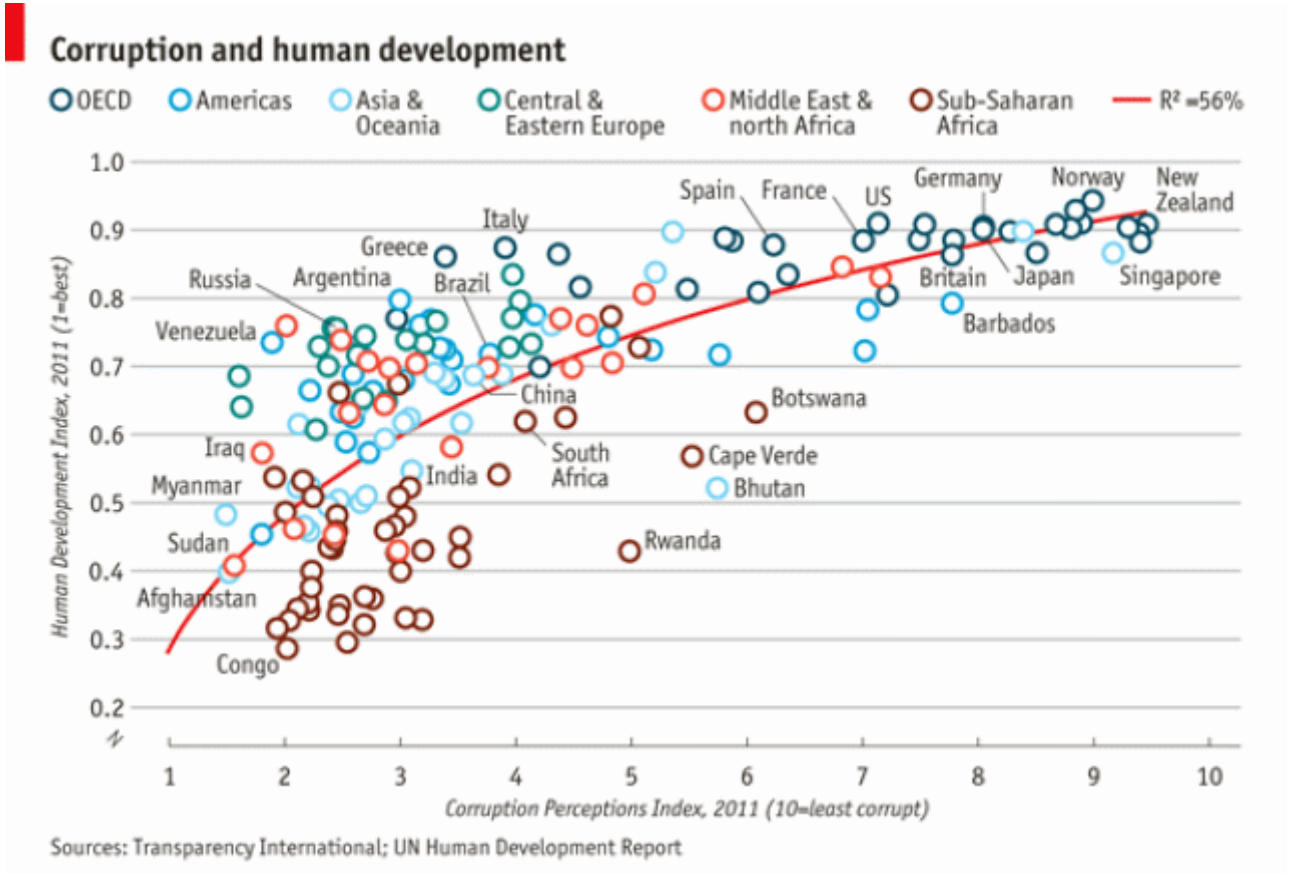
Spring 2019

Annotations

Stand-alone stories

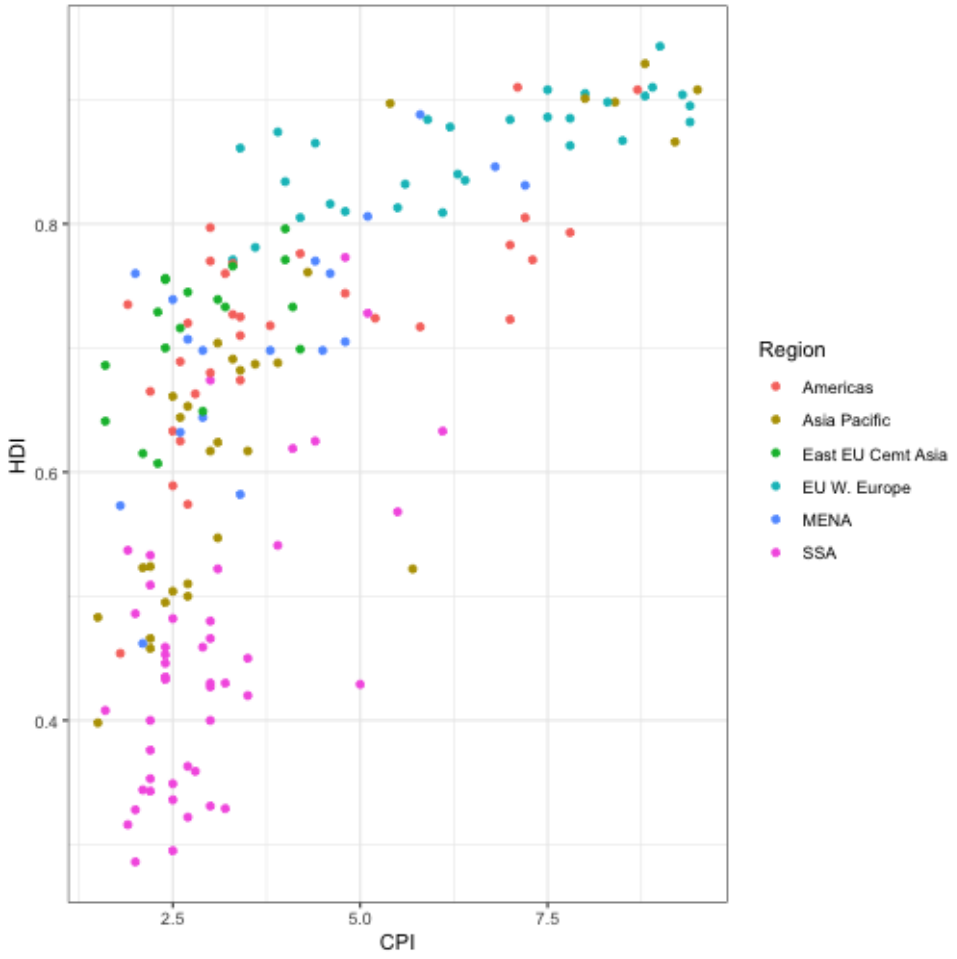
- You would like a data visualization to stand on its own
- Relevant information should be placed on the graph
- However, you need to balance the information content with real estate
    - Don't clutter the graph and make it not readable

An example


Corruption and human development
OECD
Americas
Asia & Oceania
Central & Eastern Europe
Middle East & north Africa
Sub-Saharan Africa
R² =56%
Human Development Index, 2011 (1=best)
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
Spain
France
US
Germany
Norway
New Zealand
Italy
Greece
Russia
Argentina
Brazil
Britain
Japan
Singapore
Venezuela
Barbados
China
Botswana
Iraq
South Africa
Cape Verde
India
Myanmar
Bhutan
Sudan
Rwanda
Afghanistan
Congo
1
2
3
4
5
6
7
8
9
10
Corruption Perceptions Index, 2011 (10=least corrupt)
Sources: Transparency International; UN Human Development Report

Reconstructing this annotated graph

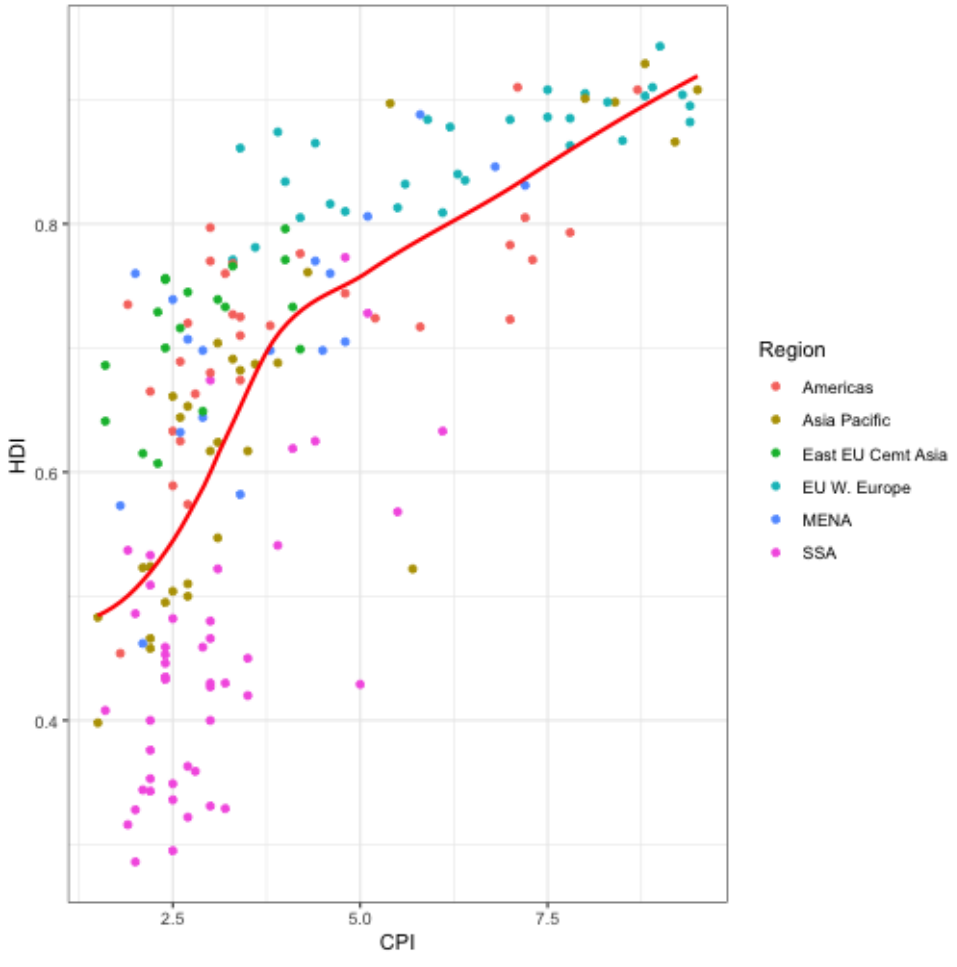
```
library(tidyverse)
econ_data <- rio::import('data/EconomistData.csv')
ggplot(econ_data,
       aes(x = CPI, y = HDI, color=Region))+
  geom_point()
```



Reconstructing this annotated graph

```
ggplot(econ_data,
       aes(x = CPI, y = HDI, color=Region))+
  geom_point() +
  geom_smooth(color='red', se=F)
```

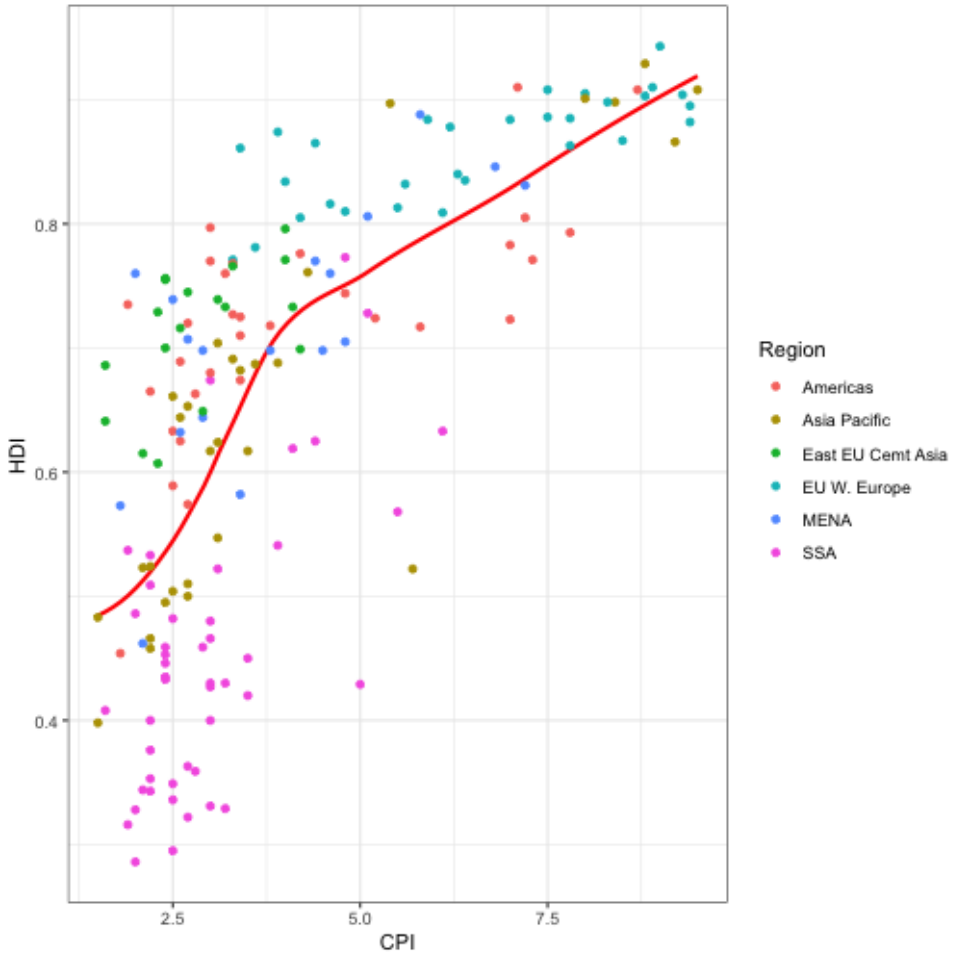
Add a trend line



Reconstructing this annotated graph

```
ggplot(econ_data,
       aes(x = CPI, y = HDI, color=Region))+
  geom_smooth(color='red', se=F) +
  geom_point()
```

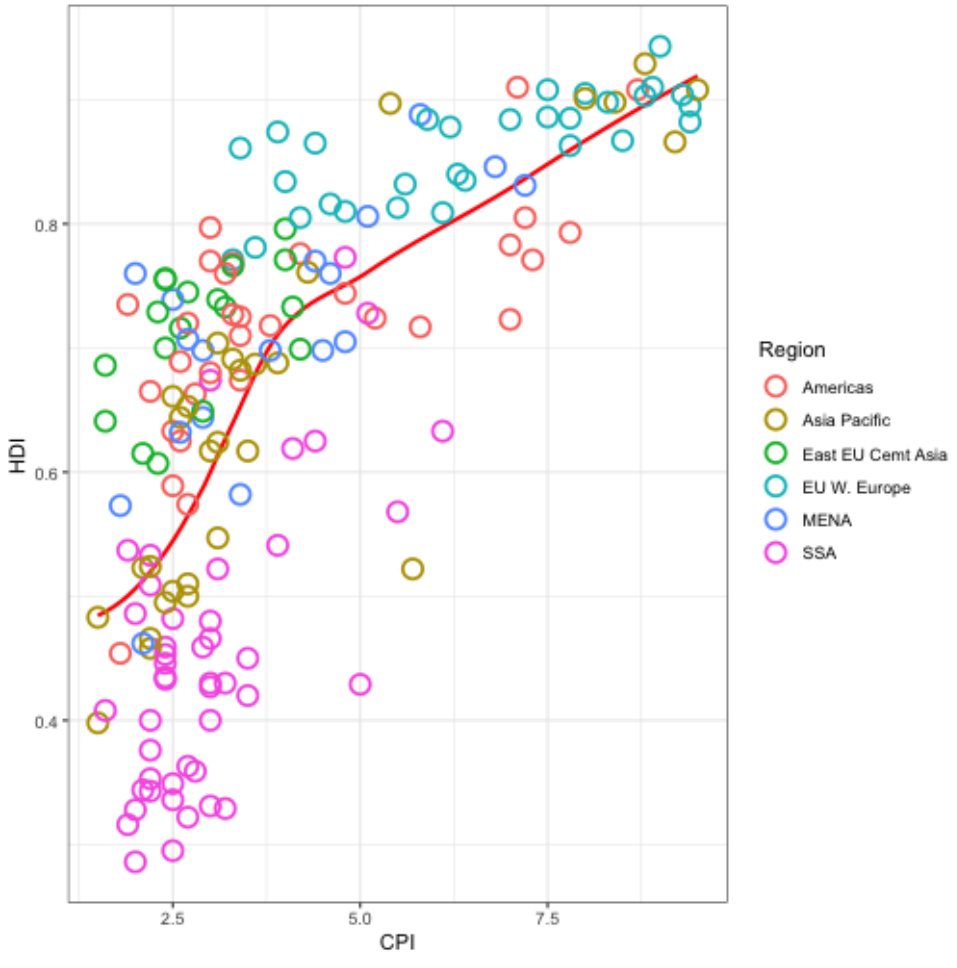
Reverse order so points are above line



Reconstructing this annotated graph

```
ggplot(econ_data,
       aes(x = CPI, y = HDI, color=Region))+
  geom_smooth(color='red', se=F) +
  geom_point(shape = 1, size = 4, stroke=1.25)
```

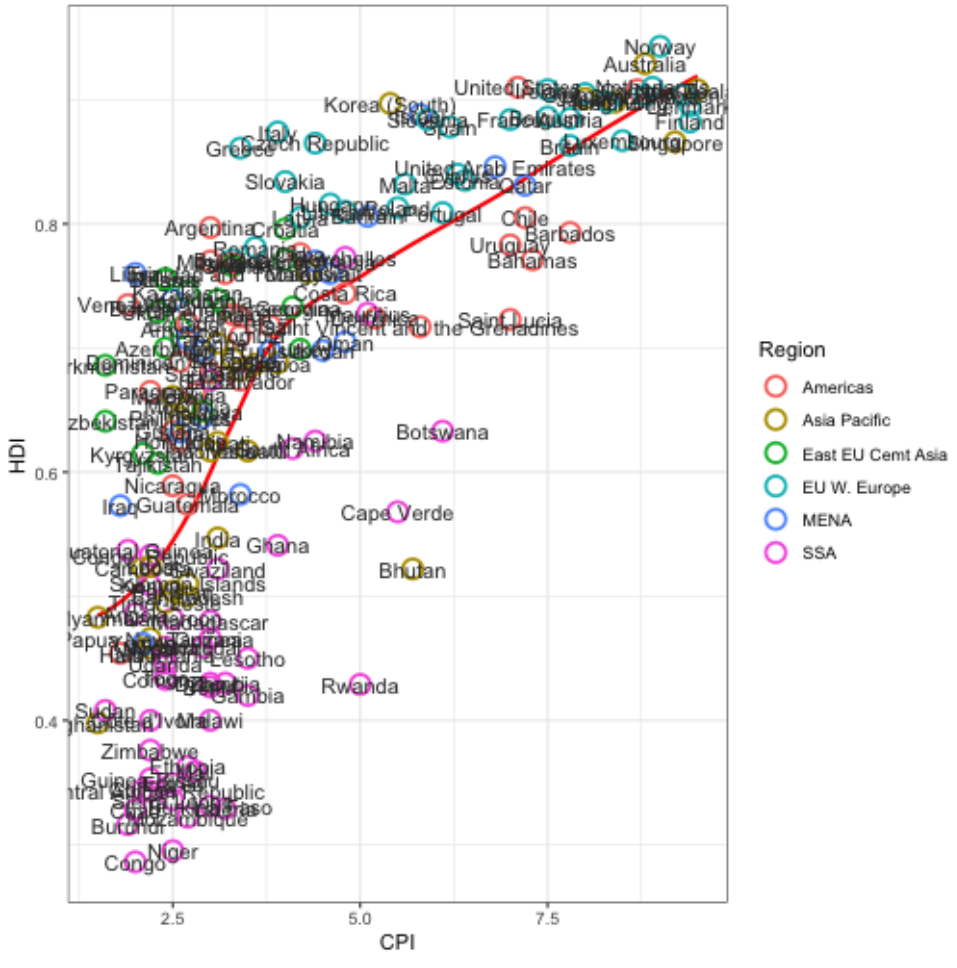
Different shape for points



Reconstructing this annotated graph

```
pointsToLabel <- c("Russia", "Venezuela", "Iraq", "My
                   "Afghanistan", "Congo", "Greece",
                   "India", "Italy", "China", "South
                   "Botswana", "Cape Verde", "Bhutan"
                   "United States", "Germany", "Brita
                   "New Zealand", "Singapore")
ggplot(econ_data,
       aes(x = CPI, y = HDI, color=Region))+
  geom_smooth(color='red', se=F) +
  geom_point(shape = 1, size = 4, stroke=1.25) +
  geom_text(aes(label=Country),
            color = 'gray20')
```

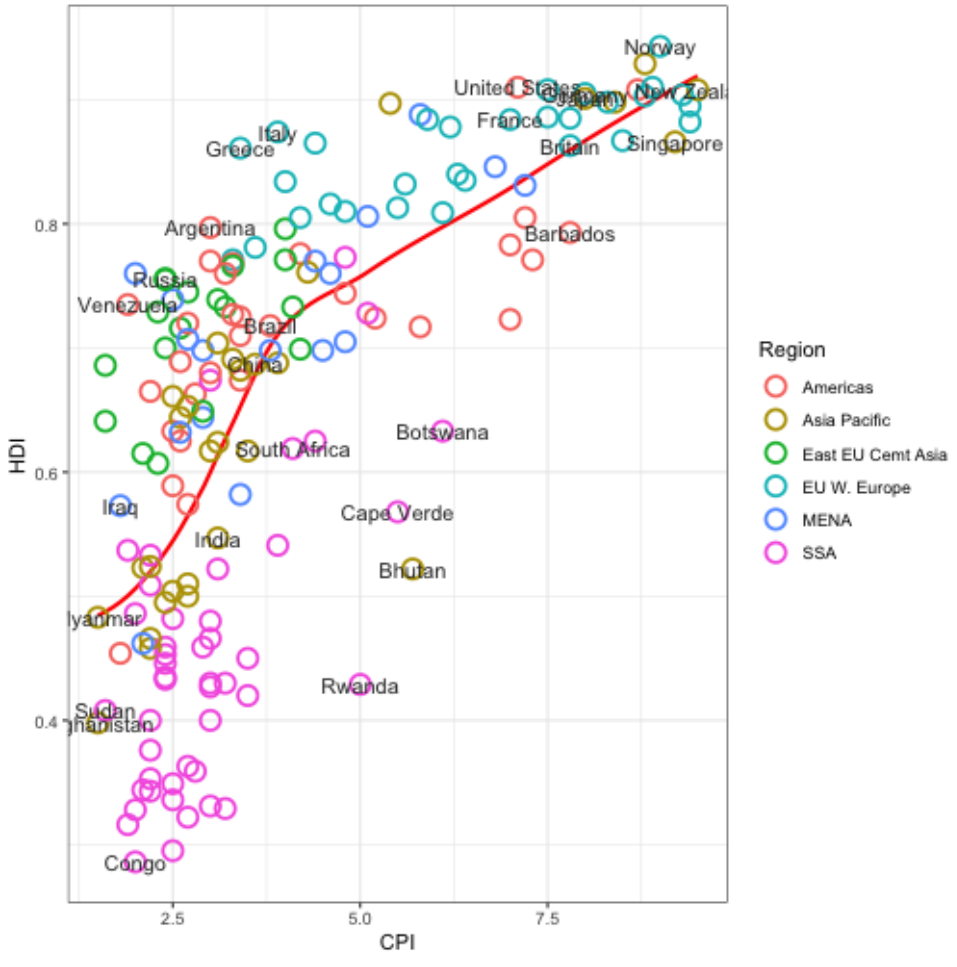
Label countries



Reconstructing this annotated graph

```
pointsToLabel <- c("Russia", "Venezuela", "Iraq", "My
                   "Afghanistan", "Congo", "Greece",
                   "India", "Italy", "China", "South
                   "Botswana", "Cape Verde", "Bhutan"
                   "United States", "Germany", "Brita
                   "New Zealand", "Singapore")
ggplot(econ_data,
       aes(x = CPI, y = HDI, color=Region))+
  geom_smooth(color='red', se=F) +
  geom_point(shape = 1, size = 4, stroke=1.25) +
  geom_text(aes(label=Country),
            color = 'gray20',
            data = econ_data %>%
              filter(Country %in% pointsToLabel))
```

Better, but labels are overlayed on points



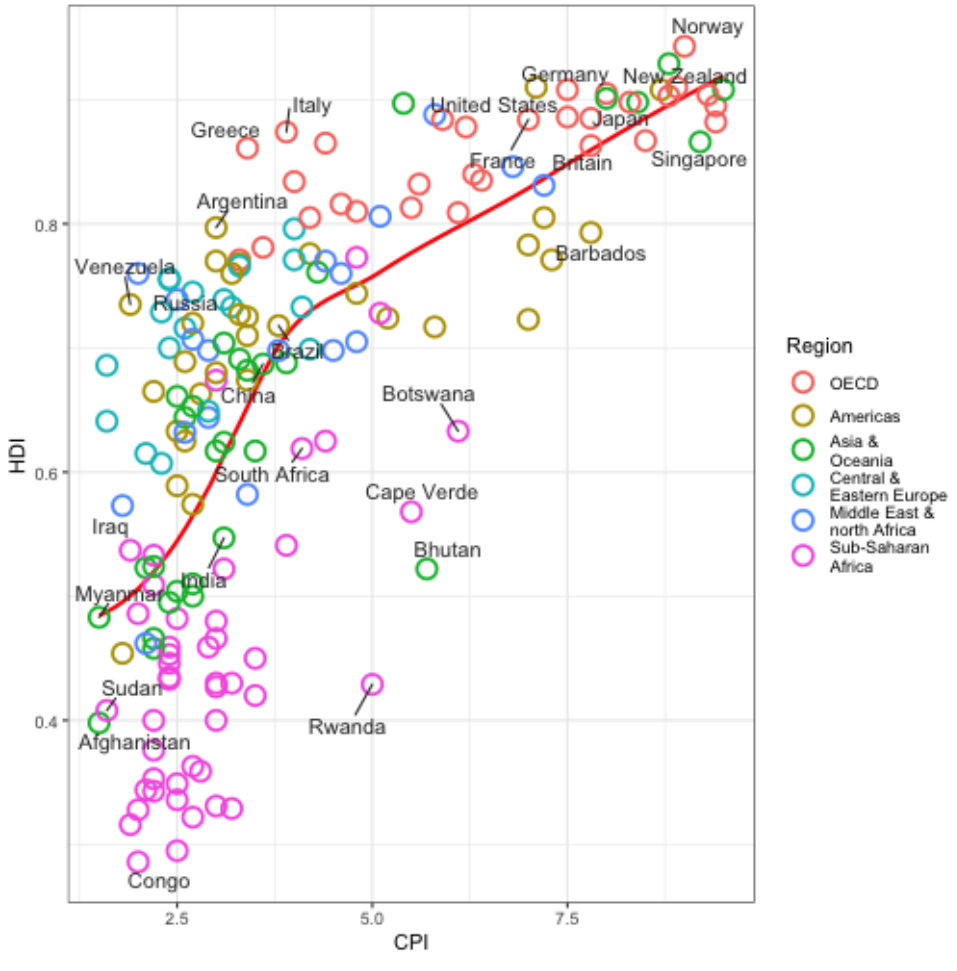
Reconstructing this annotated graph

```
library(ggrepel)
pointsToLabel <- c("Russia", "Venezuela", "Iraq", "My
                   "Afghanistan", "Congo", "Greece",
                   "India", "Italy", "China", "South
                   "Botswana", "Cape Verde", "Bhutan"
                   "United States", "Germany", "Brita
                   "New Zealand", "Singapore")
(plt <- ggplot(econ_data,
       aes(x = CPI, y = HDI, color=Region))+
  geom_smooth(color='red', se=F) +
  geom_point(shape = 1, size = 4, stroke=1.25) +
  geom_text_repel(aes(label=Country),
            color = 'gray20',
            force=20,
            data = econ_data %>%
              filter(Country %in% pointsToLabel)))
```

Reconstructing this annotated graph

Let's re-order the regions

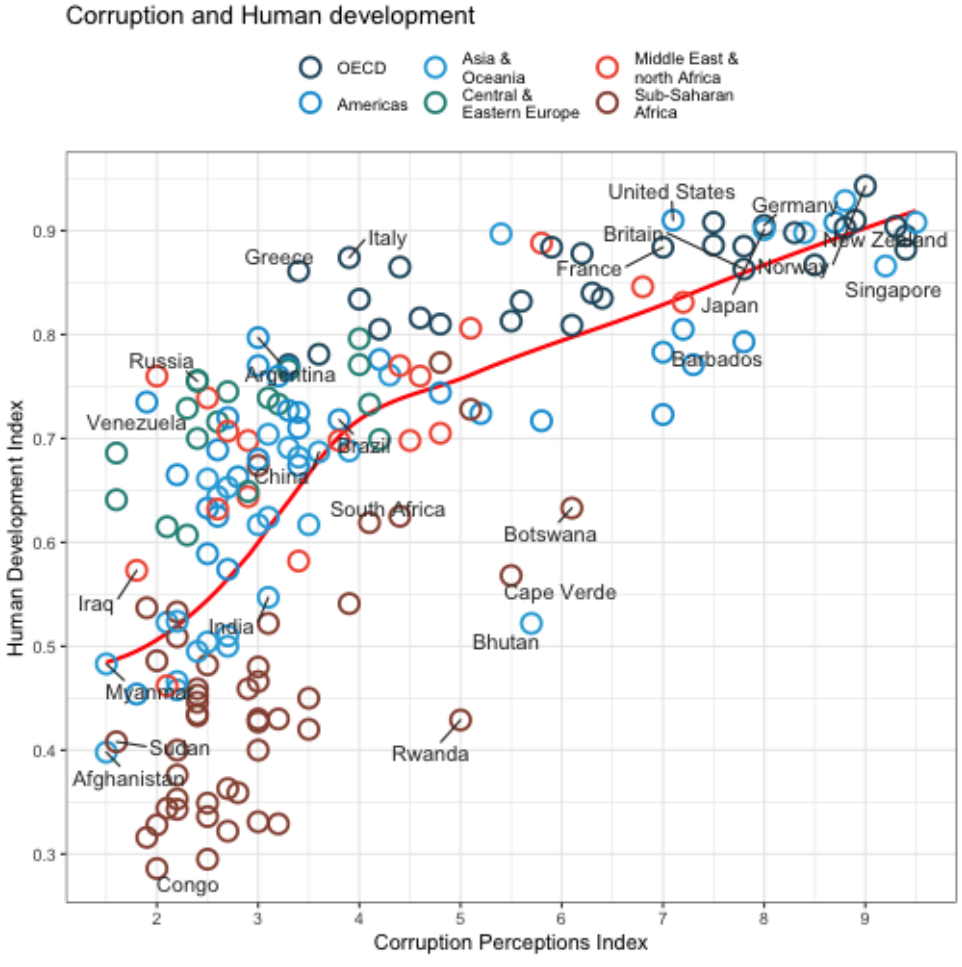
```
econ_data$Region <-
  factor(econ_data$Region,
         levels = c("EU W. Europe",
                    "Americas",
                    "Asia Pacific",
                    "East EU Cemt Asia",
                    "MENA",
                    "SSA"),
         labels = c("OECD",
                    "Americas",
                    "Asia &\nOceania",
                    "Central &\nEastern Europe",
                    "Middle East &\nnorth Africa",
                    "Sub-Saharan\nAfrica"))

plt$data = econ_data
plt
```



Reconstructing this annotated graph

Clean up the graphic

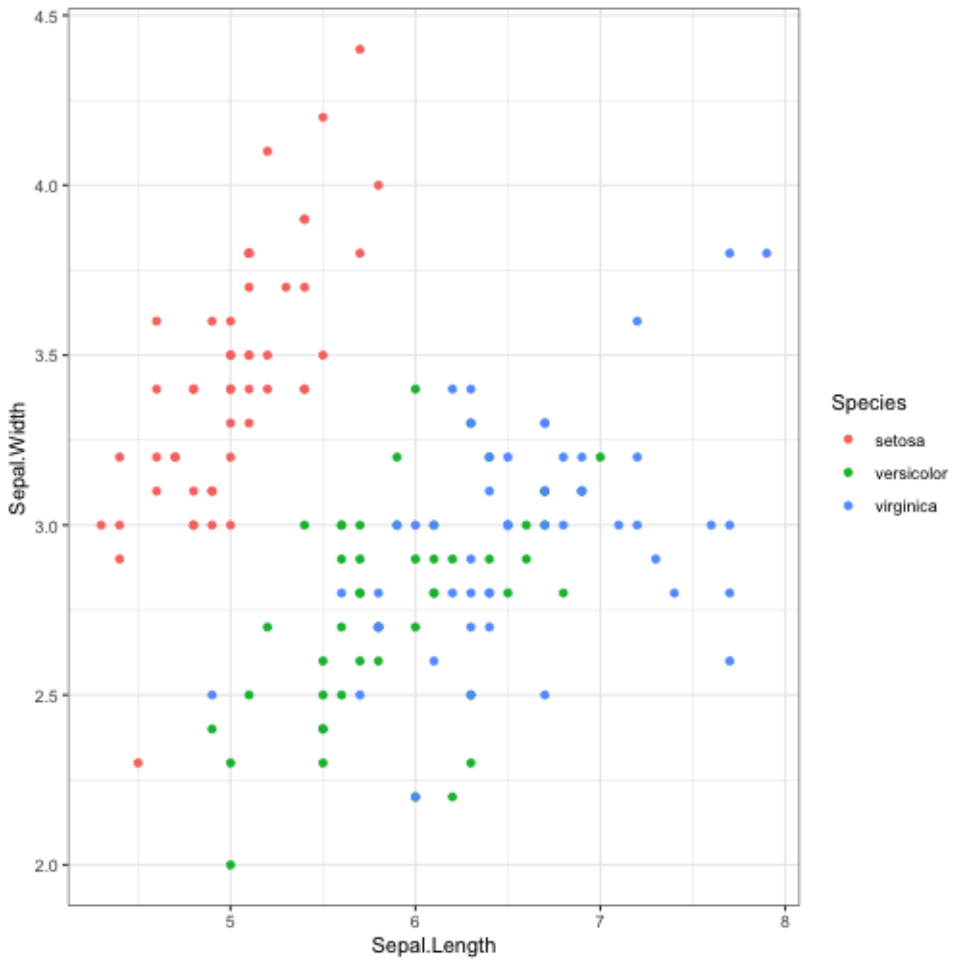
```
(plt_corrupt <-
   plt +
  scale_x_continuous(name = 'Corruption Perceptions I
                     breaks = 1:10) +
  scale_y_continuous(name="Human Development Index",
                     breaks = seq(0.2, 1, by = 0.1))+
  scale_color_manual(name = '',
                     values = c("#24576D",
                                "#099DD7",
                                "#28AADC",
                                "#248E84",
                                "#F2583F",
                                "#96503F")) +
  ggtitle("Corruption and Human development")+
  theme_bw()+
  theme(legend.position='top',
        legend.direction='horizontal')
)
```



Adding derived statistics to a plot

Adding group means

```
ggplot(iris,
       aes(x = Sepal.Length,
           y = Sepal.Width,
           color = Species))+
  geom_point()+
  theme_bw()
```

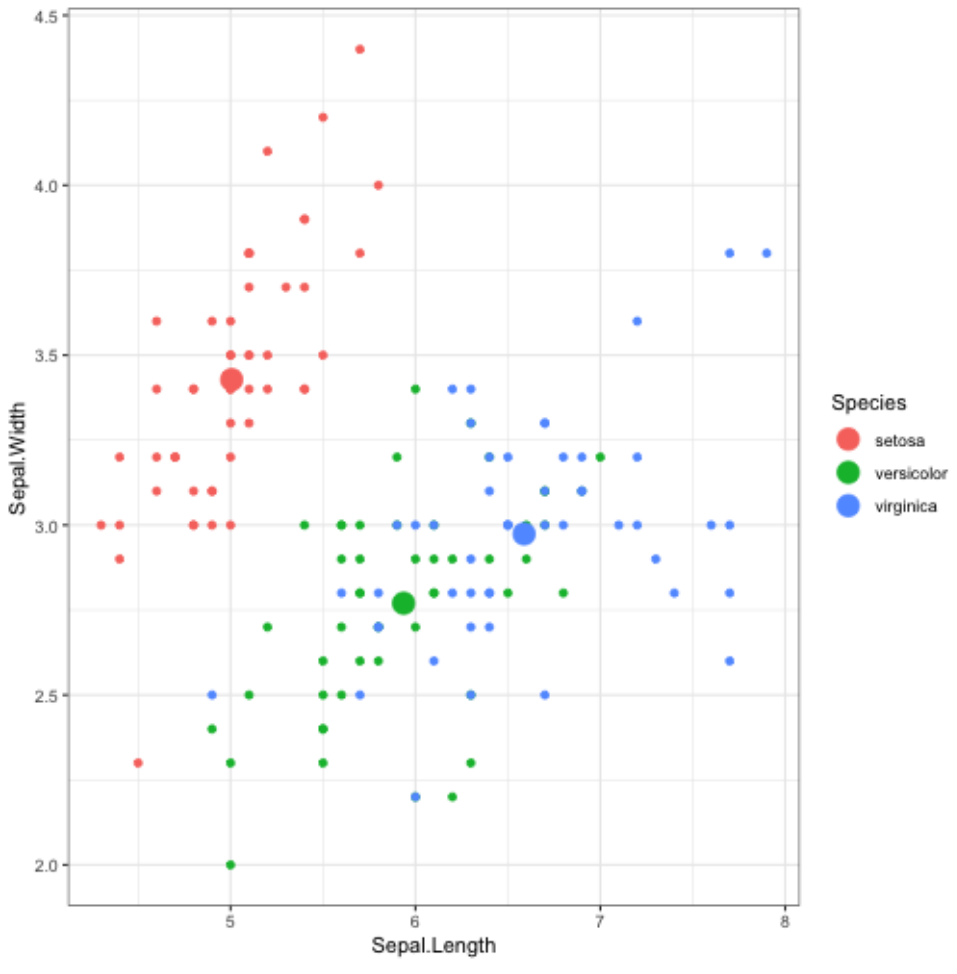


Adding group means

```
means <- iris %>% group_by(Species) %>%
  summarize_at(vars(starts_with('Sepal')),
               mean)
means
```

```
#>  # A tibble: 3 x 3
#>    Species    Sepal.Length Sepal.Width
#>    <fct>             <dbl>       <dbl>
#>  1 setosa             5.01        3.43
#>  2 versicolor         5.94        2.77
#>  3 virginica          6.59        2.97
```

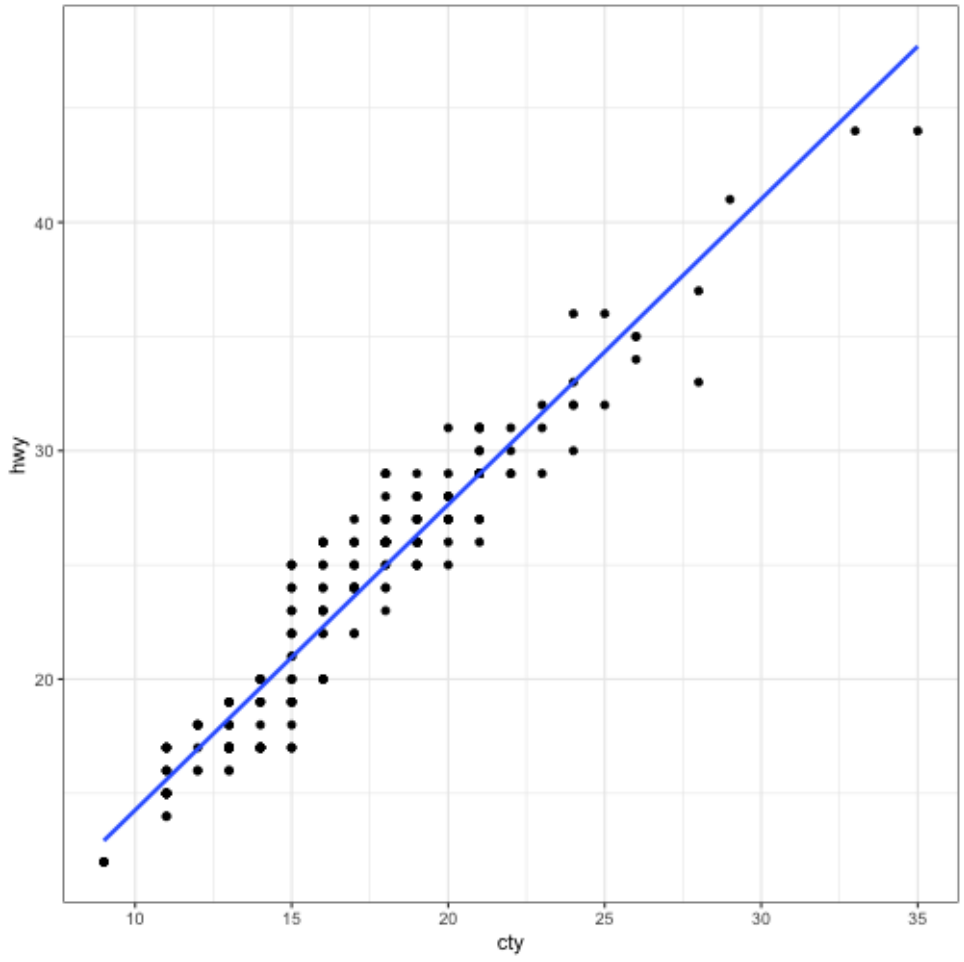
```
ggplot(iris,
       aes(x = Sepal.Length,
           y = Sepal.Width,
           color = Species))+
  geom_point() +
  geom_point(data = means,
             size=5) +
  theme_bw()
```



Adding regression metrics

Regress highway mileage on city mileage (data: mpg)

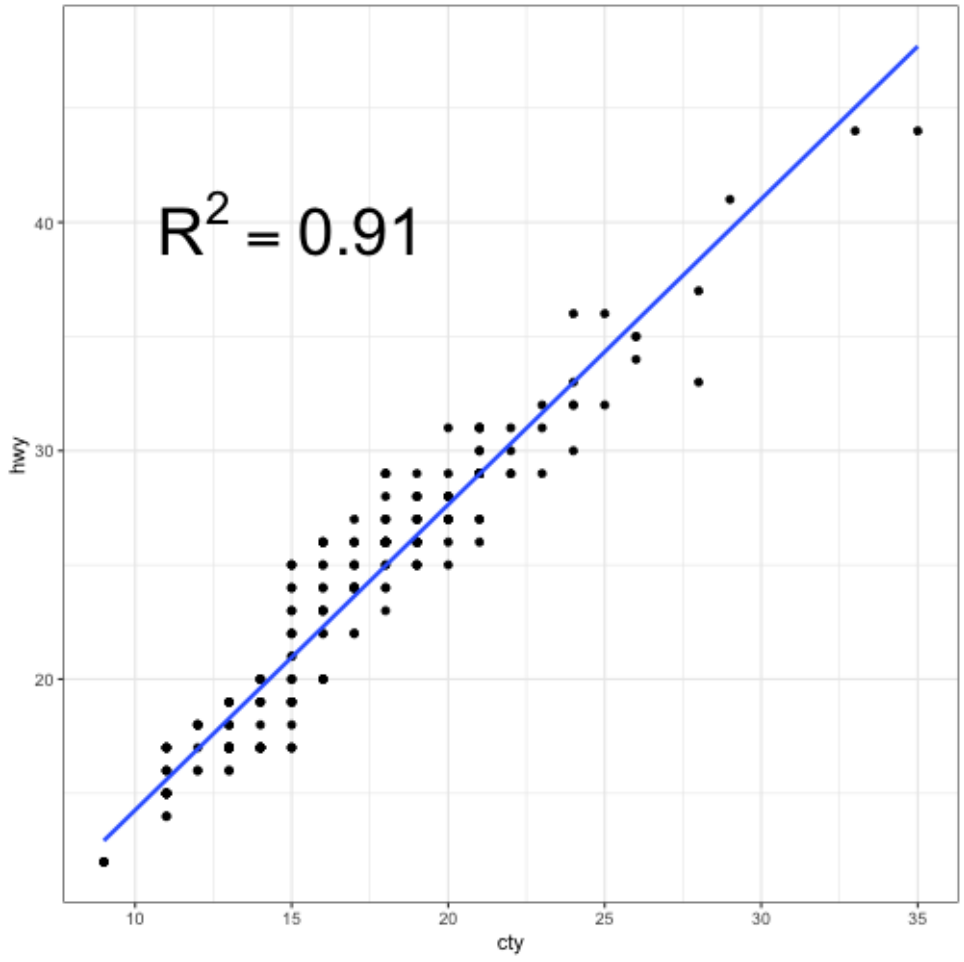
```
mod1 <- lm(hwy ~ cty, data = mpg)
r2 <- broom::glance(mod1) %>% pull(r.squared)

ggplot(mpg,
       aes(x = cty, y = hwy))+
  geom_point() +
  geom_smooth(method = 'lm', se=F) +
  theme_bw()
```



Adding regression metrics

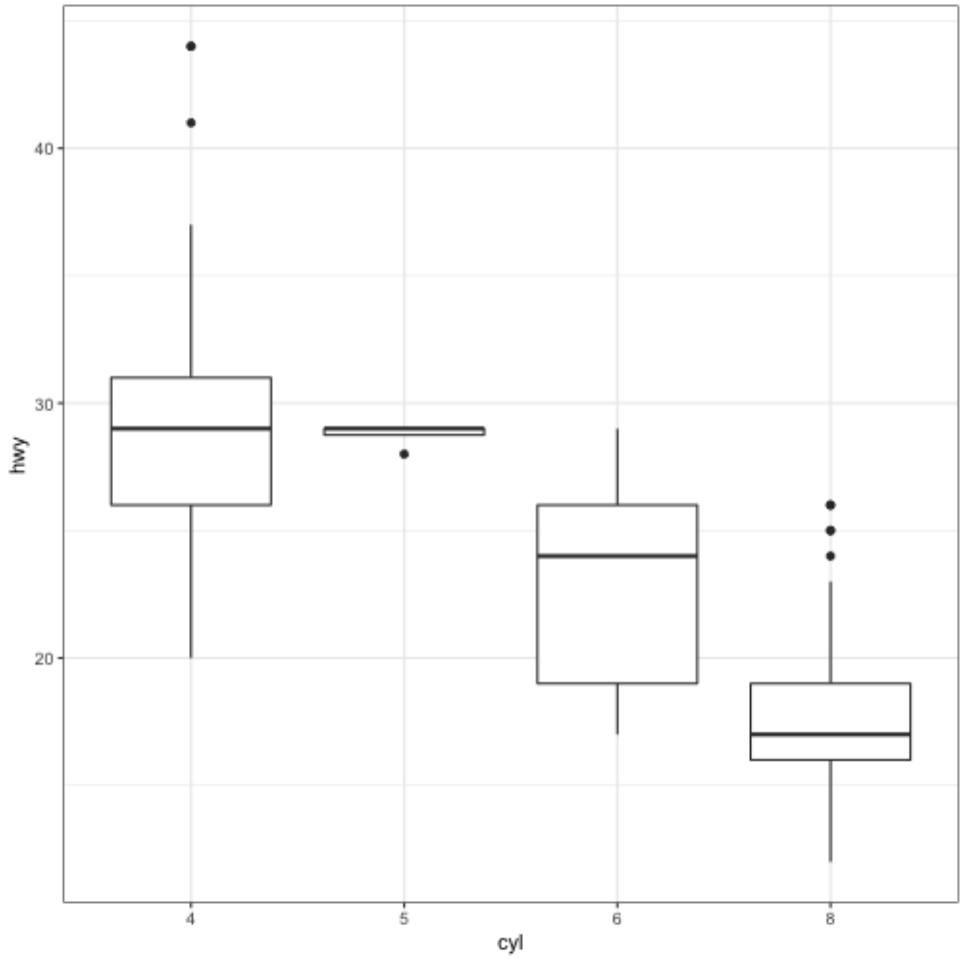
Regress highway mileage on city mileage (data: mpg)

```
mod1 <- lm(hwy ~ cty, data = mpg)
r2 <- broom::glance(mod1) %>% pull(r.squared) %>%
  round(., 2)

ggplot(mpg,
       aes(x = cty, y = hwy))+
  geom_point() +
  geom_smooth(method = 'lm', se=F)+
  annotate(geom='text',
           x = 15, y = 40,
           label=glue::glue("R^2 == {r}",r=r2),
           size=12,
           parse=T) +
  theme_bw()
```



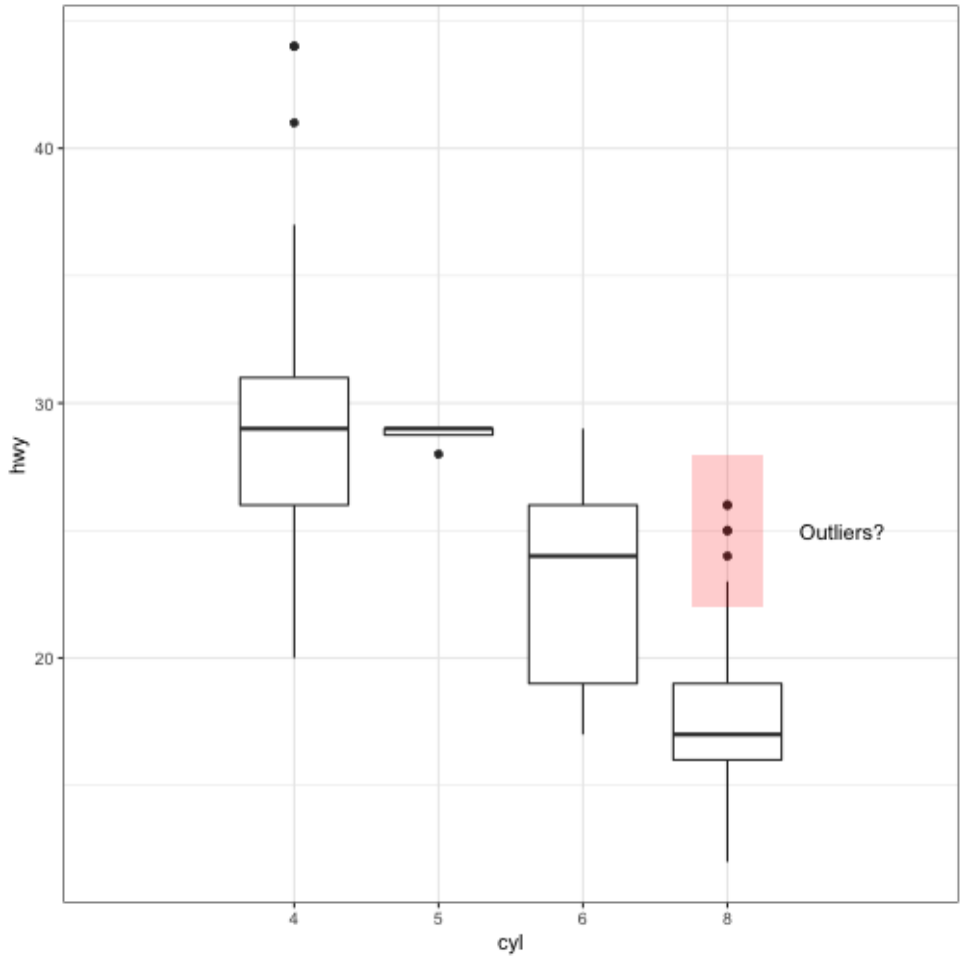
Highlighting regions

```
mpg %>%
  mutate(cyl = as.factor(cyl)) %>%
  ggplot(aes(x = cyl, y = hwy)) +
  geom_boxplot() +
  theme_bw()
```



Highlighting regions

```
mpg %>%
  mutate(cyl = as.factor(cyl)) %>%
  ggplot(aes(x = cyl, y = hwy)) +
  geom_boxplot() +
  theme_bw()+
  annotate(geom = 'rect',
           xmin=3.75,xmax=4.25,
           ymin = 22, ymax = 28,
           fill = 'red',
           alpha = 0.2) +
  annotate('text',
           x = 4.5, y = 25,
           label = 'Outliers?',
           hjust = 0)+
  coord_cartesian(xlim = c(0,5))+
  theme_bw()
```



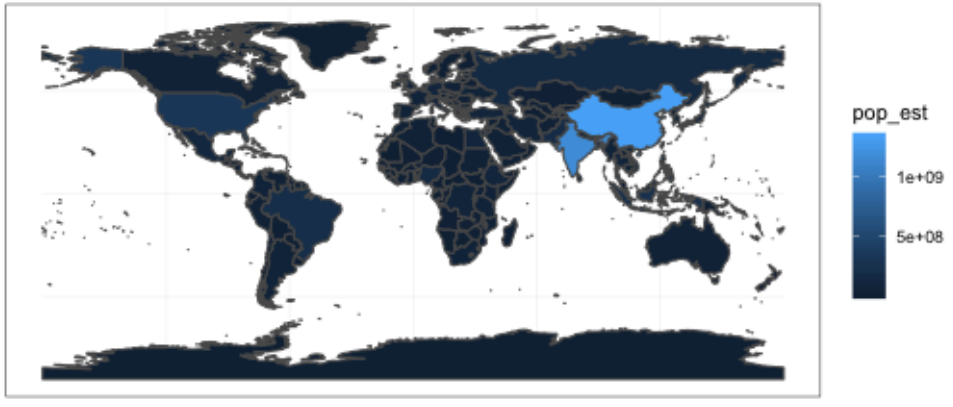
Maps

For maps, we need a couple of new packages.

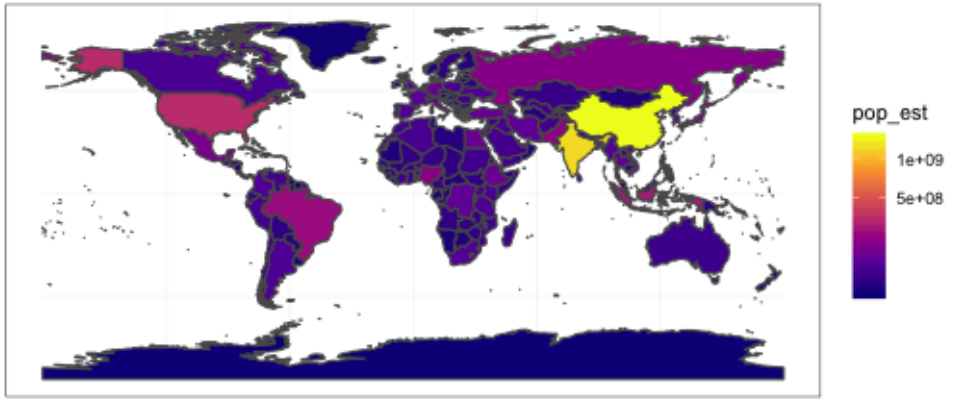
- `sf`: Simple features in R
- `rnaturalearth` & `rnaturalearthdata`: map data

```
library(sf)
library(rnaturalearth)
library(rnaturalearthdata)

world <- ne_countries(scale='medium', returnclass='sf
ggplot(data = world) +
  geom_sf()
```

```
library(sf)
library(rnaturalearth)
library(rnaturalearthdata)

world <- ne_countries(scale='medium', returnclass='sf
ggplot(data = world) +
  geom_sf(aes(fill = pop_est))
```

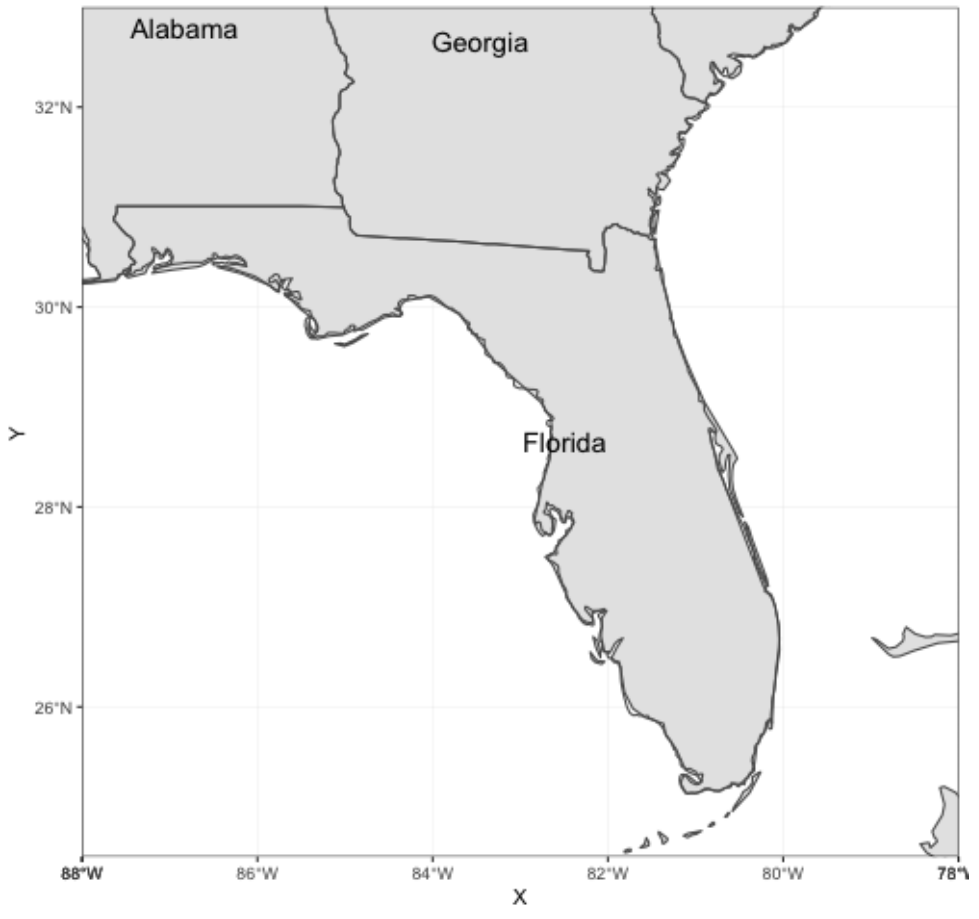


```
library(sf)
library(rnaturalearth)
library(rnaturalearthdata)

world <- ne_countries(scale='medium', returnclass='sf
ggplot(data = world) +
  geom_sf(aes(fill = pop_est))+
  scale_fill_viridis_c(option = 'plasma', trans='sqrt
```



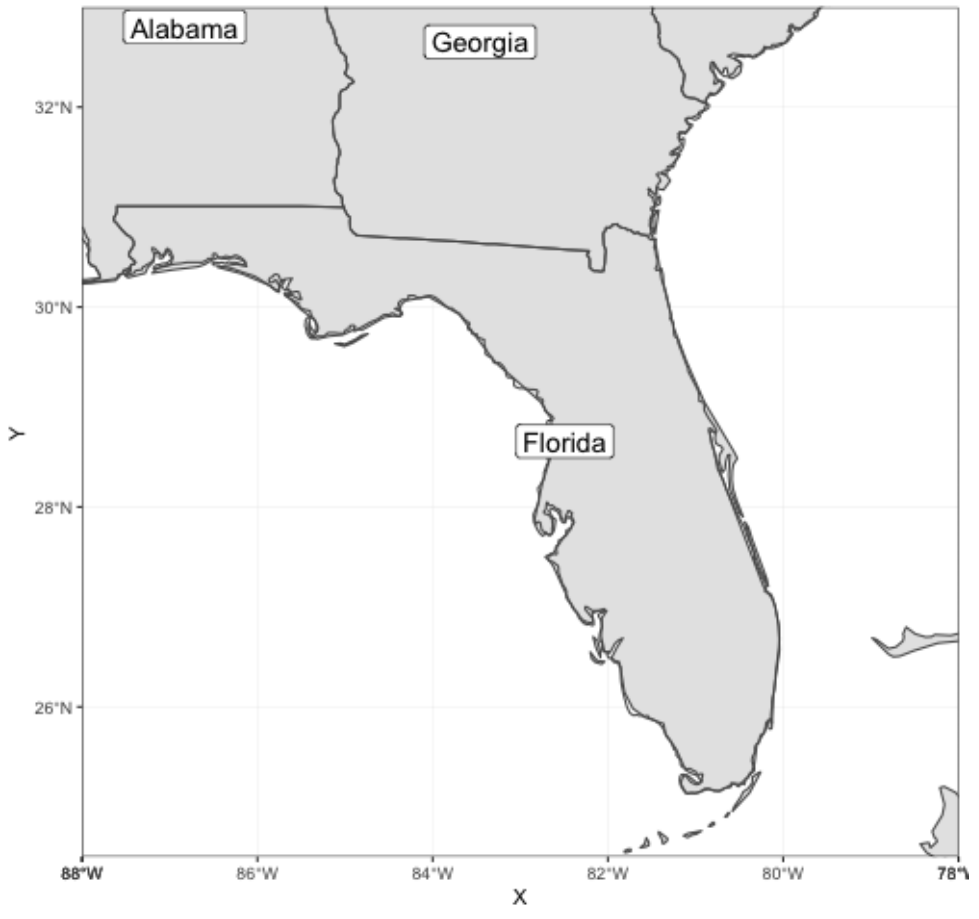
Looking at Florida

```
library(maps)
states <- st_as_sf(maps::map('state', plot = F, fill
  cbind(st_coordinates(st_centroid(.))) %>%
  mutate(ID = str_to_title(ID))

ggplot(data = world)+
  geom_sf() +
  geom_sf(data = states, fill = NA) +
  geom_text(data = states, aes(X, Y, label = ID),
            size = 5) +
  coord_sf(xlim = c(-88, -78), ylim = c(24.5, 33),
           expand = F)
```



Looking at Florida

```
library(maps)
states <- st_as_sf(map('state', plot = F, fill = T))
  cbind(st_coordinates(st_centroid(states))) %>%
  mutate(ID = str_to_title(ID))

ggplot(data = world)+
  geom_sf() +
  geom_sf(data = states, fill = NA) +
  geom_label(data = states, aes(X, Y, label = ID),
             size = 5) +
  coord_sf(xlim = c(-88, -78), ylim = c(24.5, 33),
           expand = F)
```



Looking at the Florida elections

```
source('data/florida.R')
head(florida_election)
florida_election <- florida_election %>%
  mutate_at(vars(ends_with('perc')),
            ~as.numeric(str_remove(., '%')))
```

```
#>  # A tibble: 6 x 14
#>    County   Bush Bush_perc   Gore Gore_perc Nader
#>    <chr>   <dbl> <chr>      <dbl> <chr>     <dbl>
#>  1 Alach…  34135 39.80%     47380 55.25%     3228
#>  2 Baker    5611 68.80%      2392 29.33%       53
#>  3 Bay     38682 65.70%     18873 32.06%      830
#>  4 Bradf…   5416 62.43%      3075 35.45%       84
#>  5 Breva… 115253 52.75%     97341 44.55%     4471
#>  6 Browa… 177939 30.93%    387760 67.41%     7105
#>  # … with 6 more variables: Buchanan_perc <chr>, C
#>  #   Other_perc <chr>, Margin <dbl>, Margin_perc <
```

Looking at the Florida election

Now we need the map information

```
library(maps)
counties <- st_as_sf(maps::map('county', plot = F, fi
head(counties)
counties <- counties %>% filter(str_detect(ID, 'flori
counties <- counties %>% separate(ID, c('State','Coun
  mutate_at(vars(State:County), str_to_title)
```

```
#>  Simple feature collection with 6 features and 1 f
#>  geometry type:  MULTIPOLYGON
#>  dimension:      XY
#>  bbox:           xmin: -88.01778 ymin: 30.24071 xm
#>  epsg (SRID):    4326
#>  proj4string:    +proj=longlat +datum=WGS84 +no_de
#>                            geometry             ID
#>  1 MULTIPOLYGON (((-86.50517 3... alabama,autauga
#>  2 MULTIPOLYGON (((-87.93757 3... alabama,baldwin
#>  3 MULTIPOLYGON (((-85.42801 3... alabama,barbour
#>  4 MULTIPOLYGON (((-87.02083 3...    alabama,bibb
#>  5 MULTIPOLYGON (((-86.9578 33...  alabama,blount
#>  6 MULTIPOLYGON (((-85.66866 3... alabama,bullock
```

Looking at the Florida election

The nice thing about the `sf` package is that it renders all the data into a data frame, so adding to it, or merging new data becomes easy.

We will now merge the election data with the map data

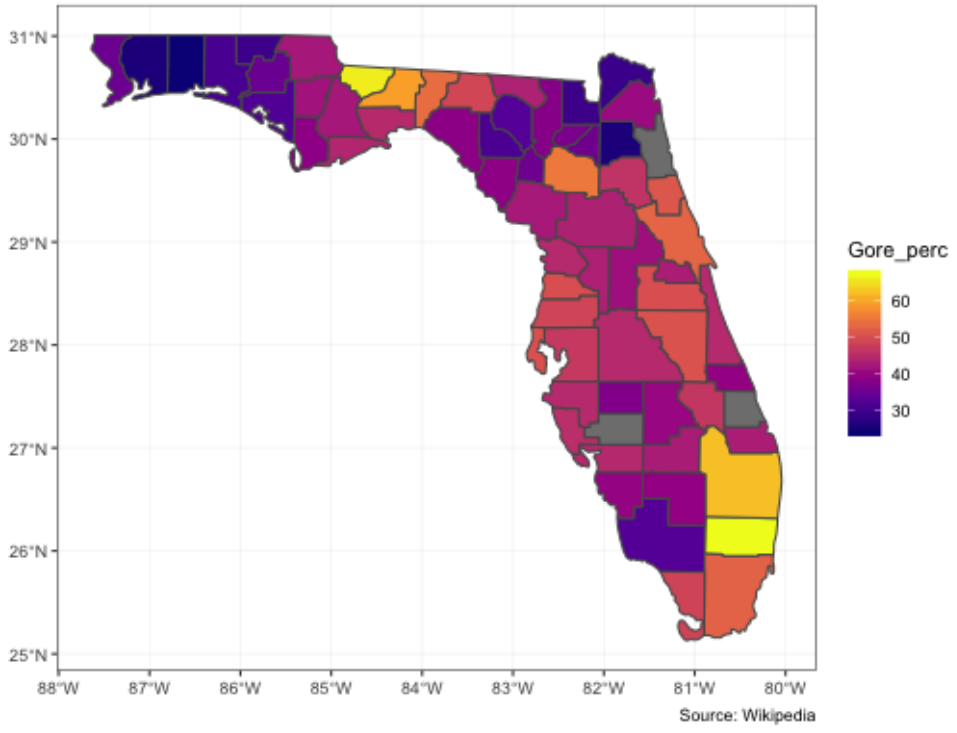
```
election_by_county <- counties %>% left_join(florida_
head(election_by_county)
```

```
#>  Simple feature collection with 6 features and 15
#>  geometry type:  MULTIPOLYGON
#>  dimension:      XY
#>  bbox:           xmin: -85.98951 ymin: 25.94926 xm
#>  epsg (SRID):    4326
#>  proj4string:    +proj=longlat +datum=WGS84 +no_de
#>      State   County   Bush Bush_perc   Gore Gore_p
#>  1 Florida  Alachua  34135     39.80  47380     55
#>  2 Florida    Baker   5611     68.80   2392     29
#>  3 Florida      Bay  38682     65.70  18873     32
#>  4 Florida Bradford   5416     62.43   3075     35
#>  5 Florida  Brevard 115253     52.75  97341     44
#>  6 Florida  Broward 177939     30.93 387760     67
#>    Buchanan Buchanan_perc Other Other_perc  Margin
#>  1      263          0.31   751       0.88  -13245
#>  2       73          0.90    26       0.32    3219
#>  3      248          0.42   243       0.41   19809
#>  4       65          0.75    35       0.40    2341
#>  5      571          0.26   852       0.39   17912
#>  6      795          0.14  1640       0.29 -209821
#>                          geometry
#>  1 MULTIPOLYGON (((-82.66062 2...
#>  2 MULTIPOLYGON (((-82.04182 3...
#>  3 MULTIPOLYGON (((-85.40509 3...
#>  4 MULTIPOLYGON (((-82.4257 29...
#>  5 MULTIPOLYGON (((-80.94747 2...
#>  6 MULTIPOLYGON (((-80.89018 2...
```

Looking at the Florida election

Now we're ready to plot

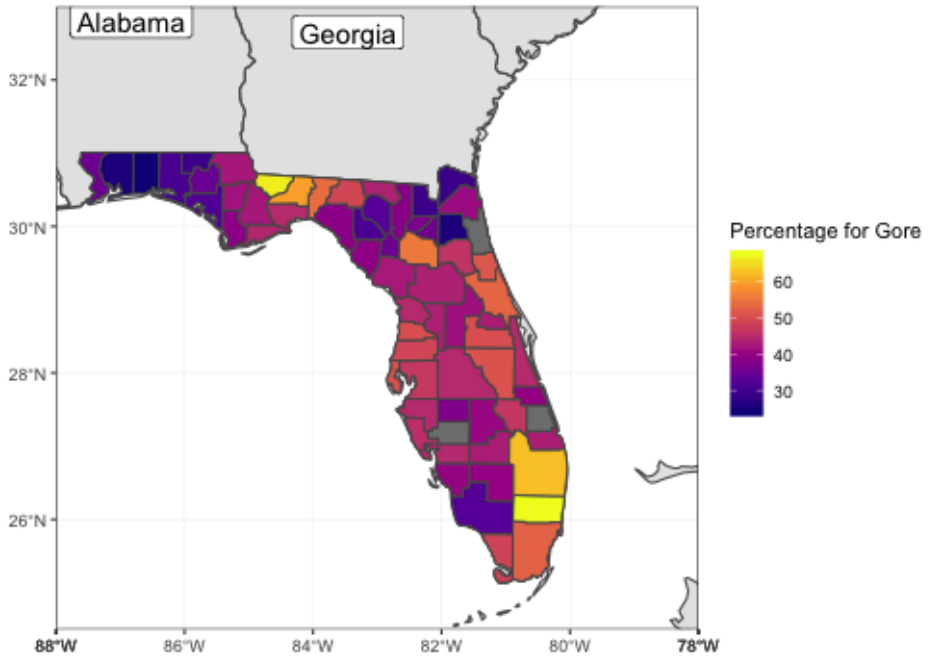
```
ggplot(election_by_county) +
  geom_sf(aes(fill = Gore_perc)) +
  scale_fill_viridis_c(option = 'plasma') +
  labs(caption = 'Source: Wikipedia')
```



Looking at the Florida election

We can clean this up a bit, and add surrounding states

```
(plt_map <- ggplot(data = world)+
  geom_sf() +
  geom_sf(data = states, fill = NA) +
  geom_sf(data = election_by_county, aes(fill=Gore_pe
  geom_label(data = states %>% filter(ID != 'Florida'
             size = 5) +
  coord_sf(xlim = c(-88, -78), ylim = c(24.5, 33),
           expand = F) +
  labs(x = '', y = '', fill = 'Percentage for Gore')
  scale_fill_viridis_c(option = 'plasma'))
```



Genomic data

Visualizing a proteomic network

We read a dataset that contains the network relationships between different proteins

```
library(ggnetwork)
datf <- rio::import('data/string_graph.txt')
head(datf)
```

```
#>      node1  node2 node1_string_id node2_string_id
#>  1   CXCR3   CCR7         1855969         1843829
#>  2   ITGA4    EED         1858446         1845338
#>  3    SMC3  CENPK         1854200         1843648
#>  4 HNRNPA1 LUC7L3         1852510         1843556
#>  5    SMC2    RB1         1847012         1845924
#>  6   RBBP4  CENPK         1855919         1843648
#>    node2_external_id neighborhood fusion cooccuren
#>  1   ENSP00000246657            0      0
#>  2   ENSP00000263360            0      0
#>  3   ENSP00000242872            0      0
#>  4   ENSP00000240304            0      0
#>  5   ENSP00000267163            0      0
#>  6   ENSP00000242872            0      0
#>    experimental knowledge textmining combined_scor
#>  1        0.000       0.9      0.878          0.91
#>  2        0.566       0.0      0.312          0.68
#>  3        0.000       0.9      0.081          0.90
#>  4        0.309       0.0      0.394          0.56
#>  5        0.000       0.9      0.097          0.91
#>  6        0.000       0.9      0.046          0.90
```

Visualizing a proteomic network

The `igraph` package allows the creation of network graphs.

However, here, we're only using it for data ingestion

```
grs <- igraph::graph_from_data_frame(datf[,c('node1',
                                             directed = F)
grs
```

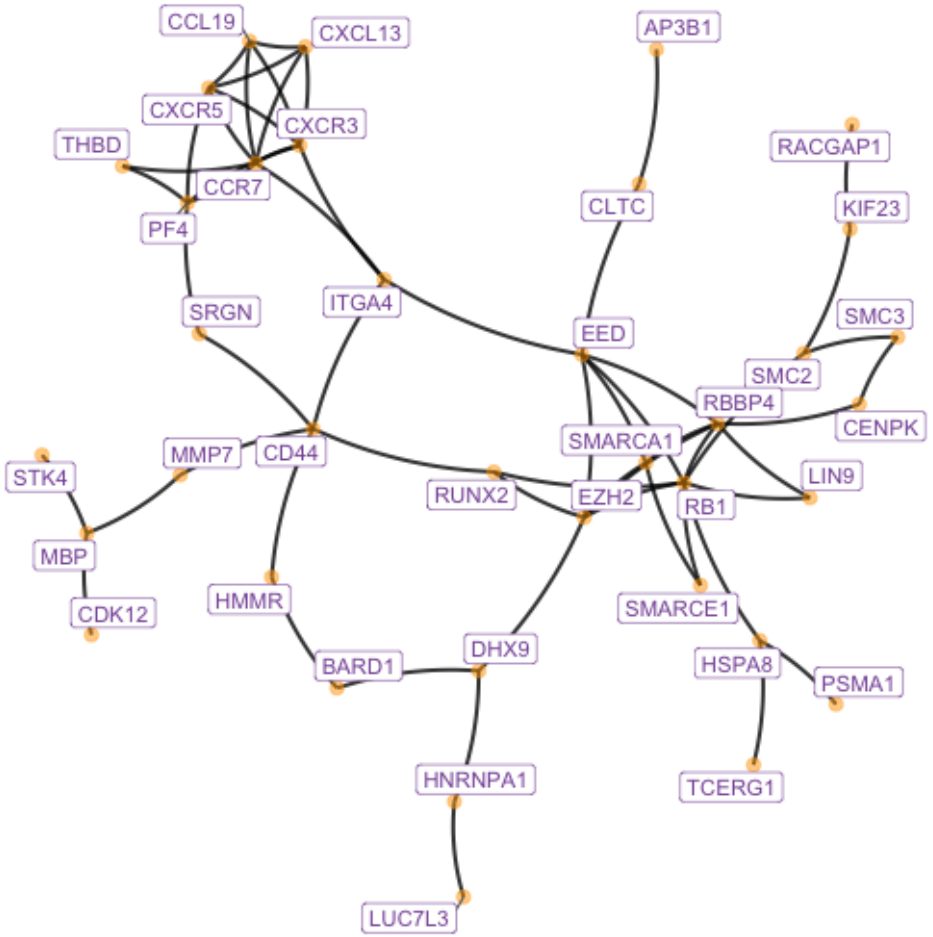
We see that this object holds the different connections.

```
#>  IGRAPH 82ff109 UN-- 37 58 --
#>  + attr: name (v/c)
#>  + edges from 82ff109 (vertex names):
#>   [1] CXCR3  --CCR7    ITGA4  --EED     SMC3   --C
#>   [5] SMC2   --RB1     RBBP4  --CENPK   CXCR5  --C
#>   [9] CXCR5  --PF4     PF4    --THBD    SMARCA1--E
#>  [13] MBP    --MMP7    CCL19  --CCR7    RBBP4  --E
#>  [17] RB1    --HSPA8   DHX9   --BARD1   CXCL13 --C
#>  [21] CD44   --HMMR    ITGA4  --CD44    RB1    --S
#>  [25] MBP    --STK4    RBBP4  --LIN9    RB1    --E
#>  [29] PSMA1  --HSPA8   RBBP4  --SMARCA1 CXCR3  --I
#>  + ... omitted several edges
```

Visualizing a proteomic network

We can then transform this data into `ggplot`-friendly data, to use `ggplot` for the plotting

```
ggdf <- ggnetwork(grs, layout = 'fruchtermanreingold'

ggplot(ggdf, aes(x = x, y = y, xend = xend, yend = ye
  geom_edges(color = "black",
             curvature = 0.1, size = 0.95, alpha = 0.
  geom_nodes(aes(x = x, y = y),
             size = 3,
             alpha = 0.5,
             color = "orange") +
  geom_nodelabel_repel(aes(label = vertex.names),
                       size=4, color="#8856a7") +
  theme_blank() + theme(legend.position = "none")
```



Composing different genomic data into tracks

The ggbio package

The ggbio package has several functions that allow graphical representations of different genomic entities.

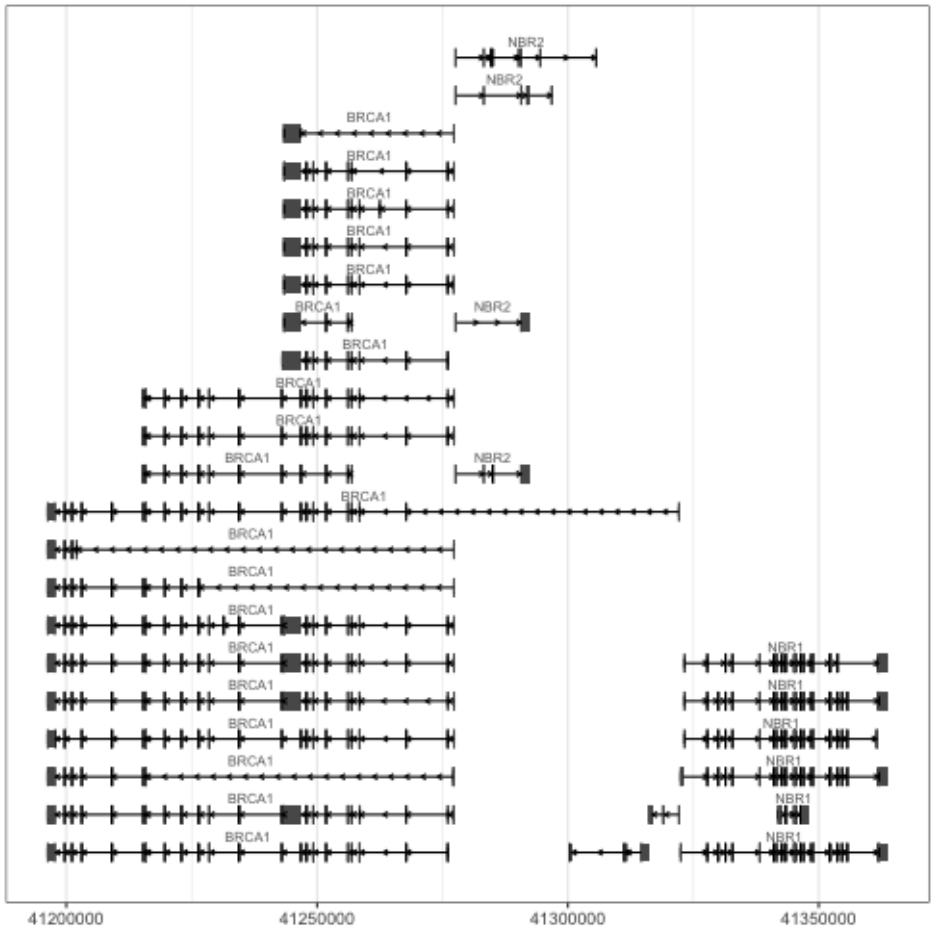
An ideogram

```
library(ggbio)
p.ideo <- Ideogram(genome = 'hg19')
p.ideo
```

The ggbio package

Visualizing the gene model

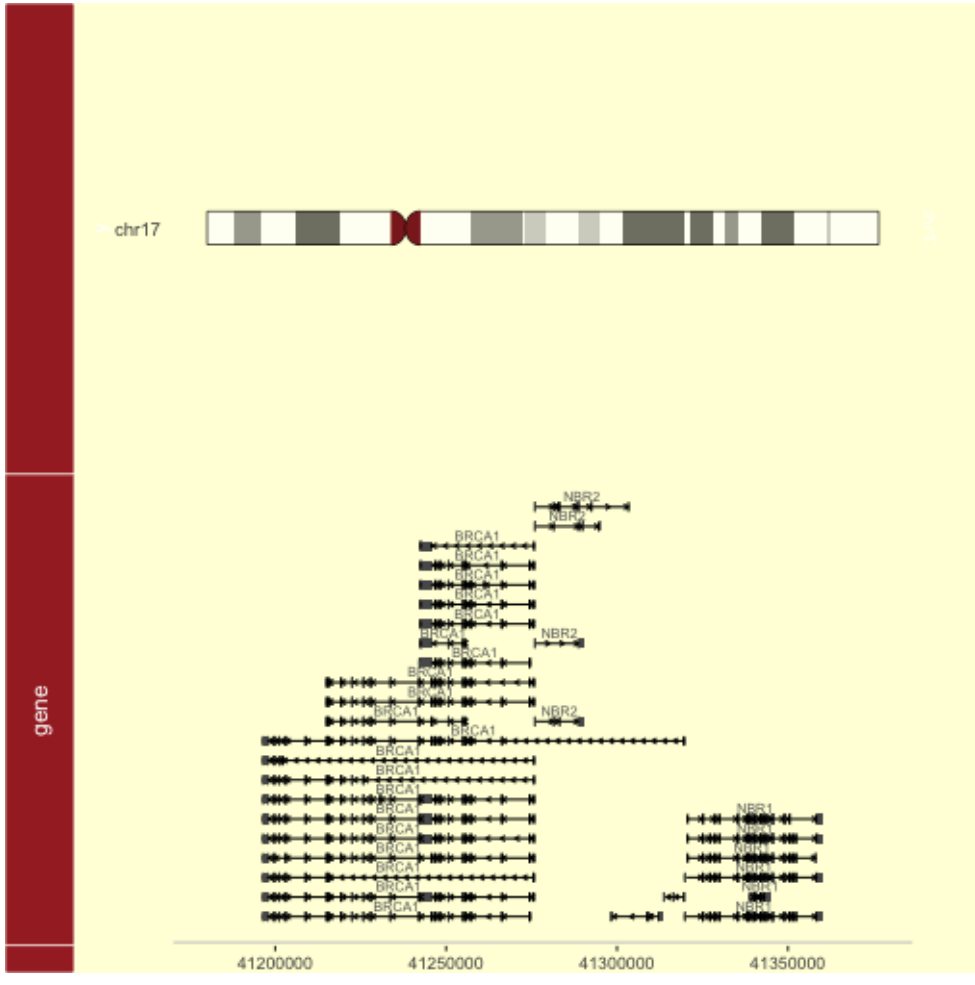
```
library(Homo.sapiens)
data(genesymbol, package='biovizBase')
wh <- genesymbol[c('BRCA1','NBR1')]
wh <- range(wh, ignore.strand=T)
p.txdb <- autoplot(Homo.sapiens, which = wh)
p.txdb
```



The ggbio package

Putting it into tracks

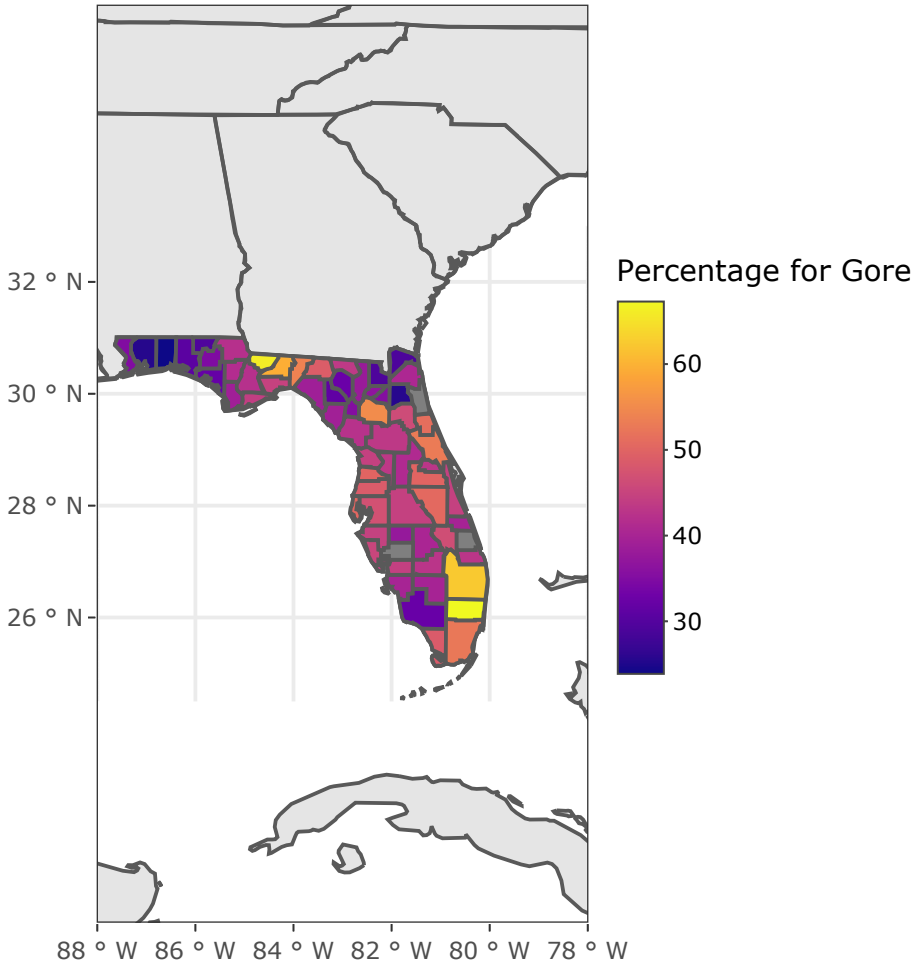
```
library(GenomicRanges)
gr17 <- GRanges("chr17", IRanges(41234415, 41234569))
tks <-  tracks(p.ideo, gene = p.txdb) + xlim(gr17)
tks + theme_tracks_sunset()
```



Interactive maps

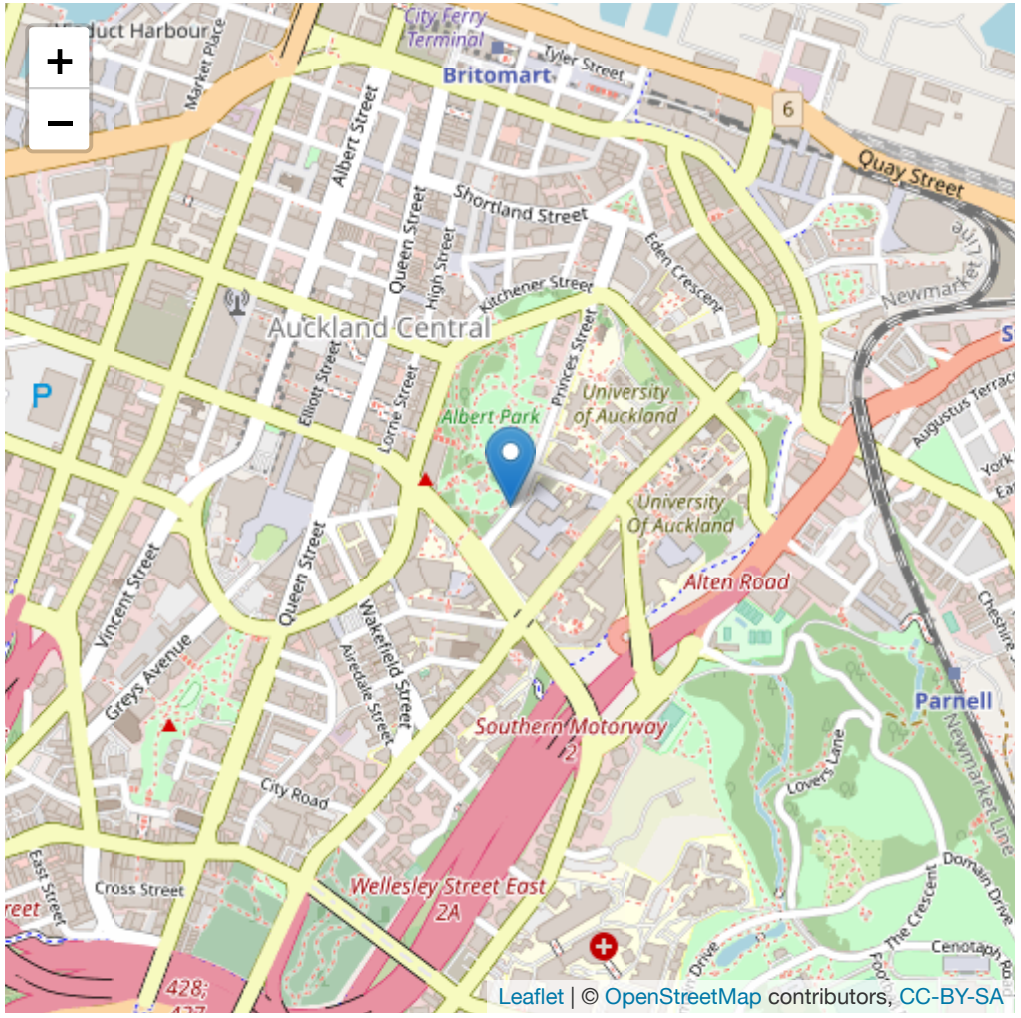
Using plotly

```
library(plotly)

ggplotly(plt_map)
```



Using real maps

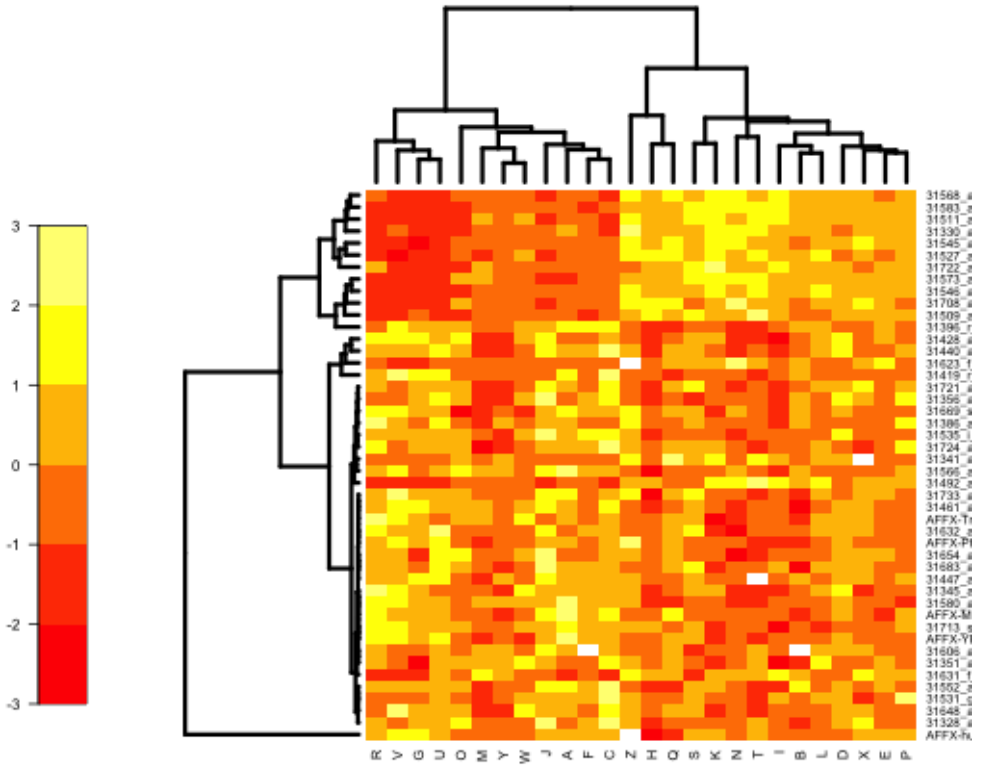
```
library(leaflet)

m <- leaflet() %>%
  addTiles() %>%  # Add default OpenStreetMap map til
  addMarkers(lng=174.768, lat=-36.852, popup="The bir
m
```



Heatmaps

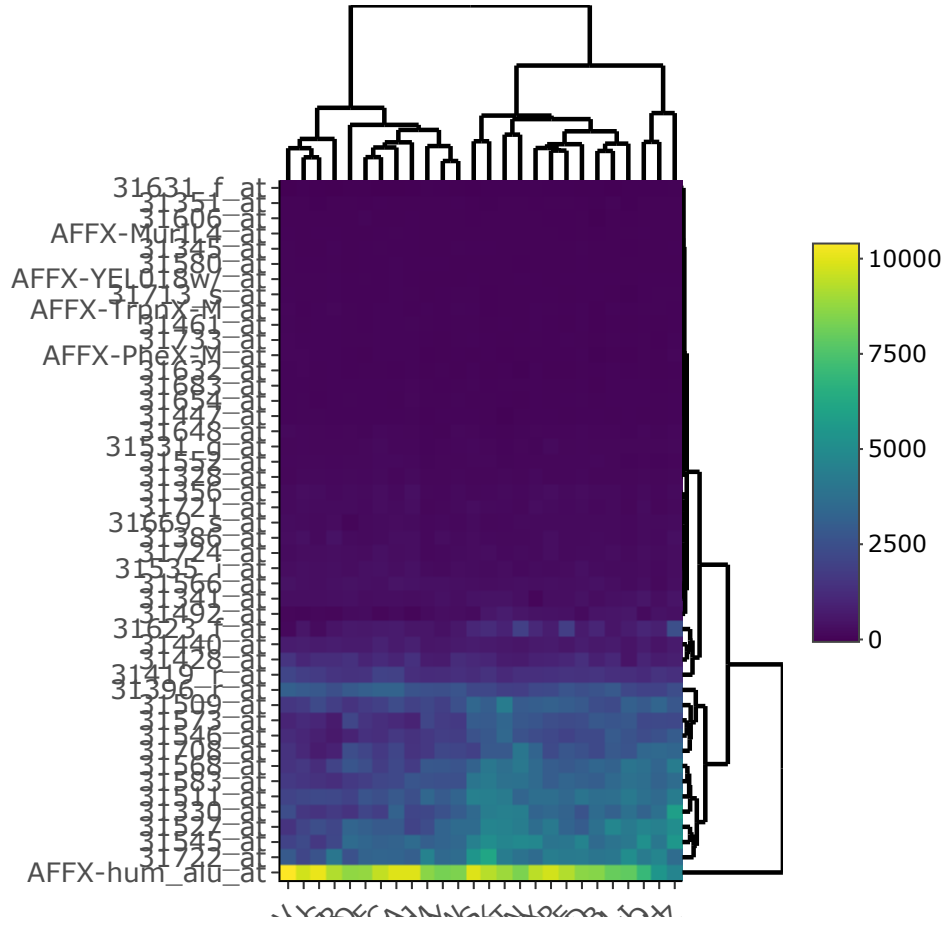
Recall our heatmap

```
library(Biobase)
#data(sample.ExpressionSet)
exdat <-  readRDS('data/exprset.rds')
library(limma)
design1 <- model.matrix(~type, data=pData(exdat))
lm1 <- lmFit(exprs(exdat), design1)
lm1 <- eBayes(lm1) # compute linear model for each pr
geneID <- rownames(topTable(lm1, coef = 2, number = 1
                            adjust.method = 'none',
                            p.value = 0.05))
exdat2 <- exdat[geneID,] # Keep features with p-value

library(Heatplus)
reg1 <- regHeatmap(exprs(exdat2), legend=2, col=heat.
                   breaks=-3:3)
plot(reg1)
```



Using heatmaply

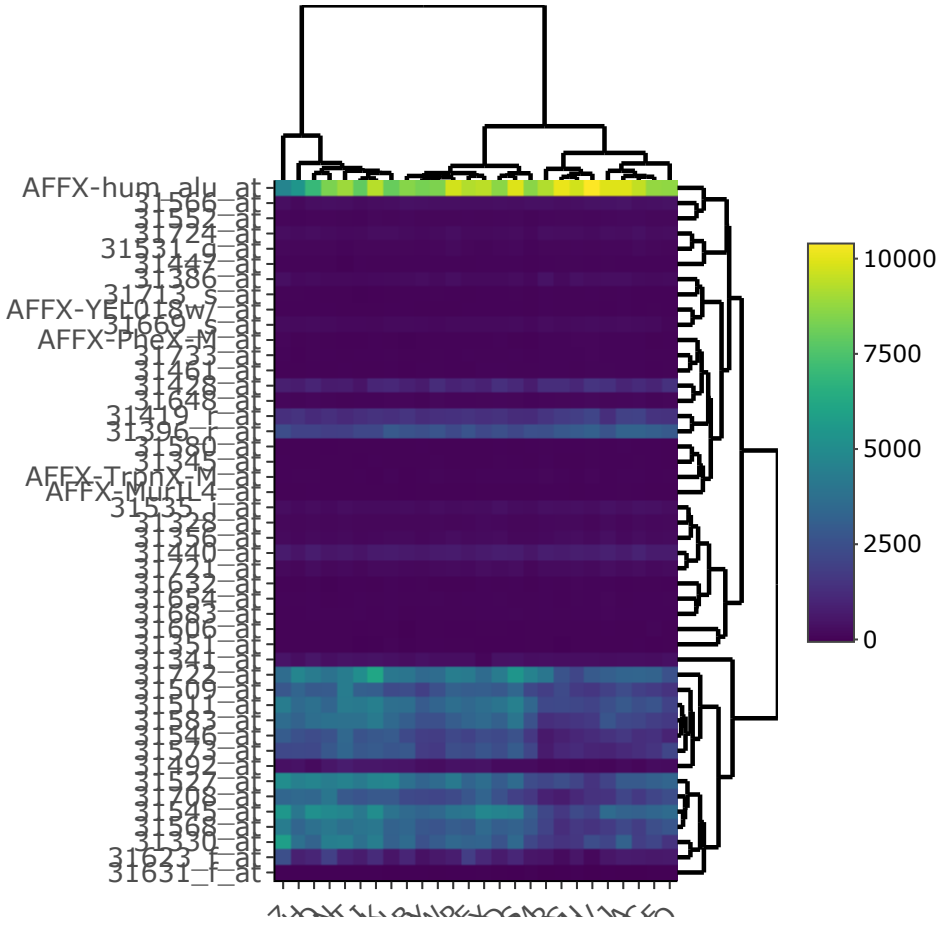
```
library(heatmaply)
heatmaply(exprs(exdat2))
```



Using heatmaply

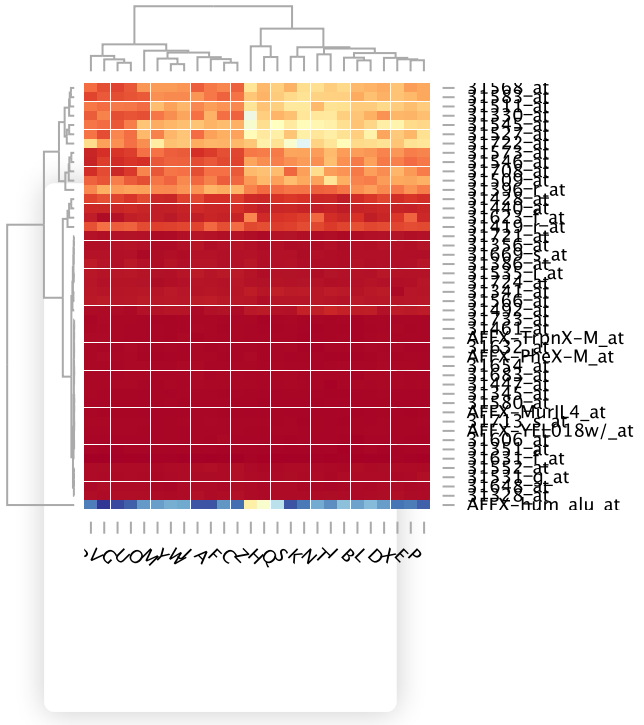
You can also easily use the correlation metric

```
heatmaply(exprs(exdat2), distfun = 'pearson')
```



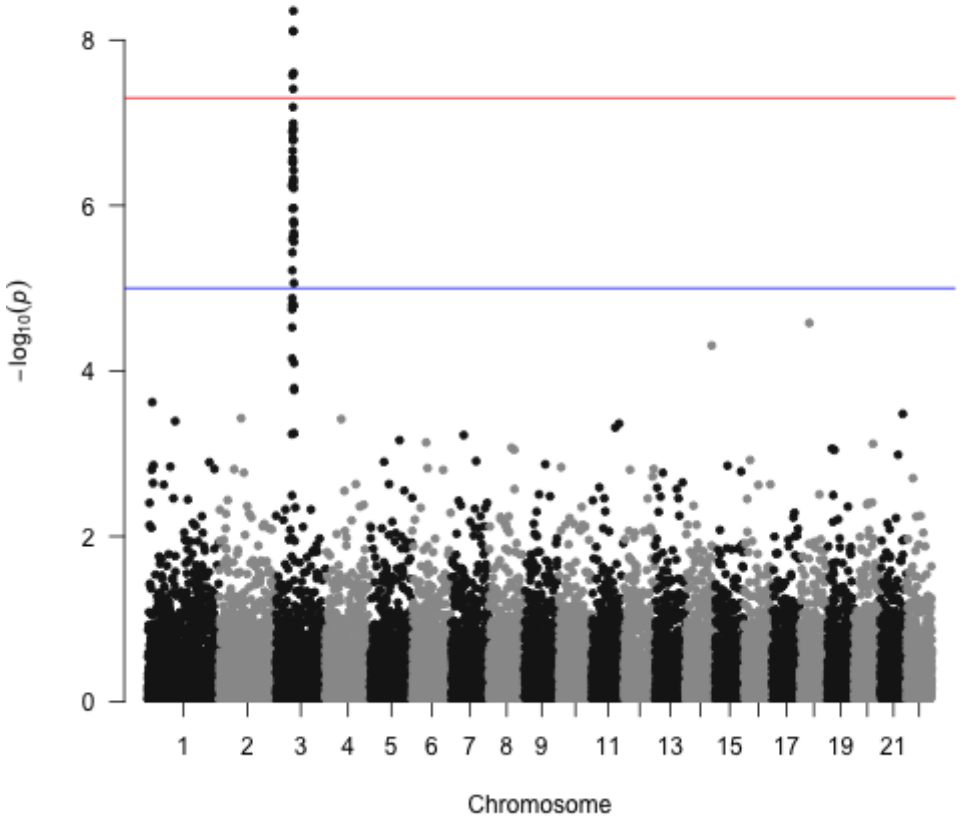
Using d3heatmap

```
library(d3heatmap)
d3heatmap(exprs(exdat2))
```

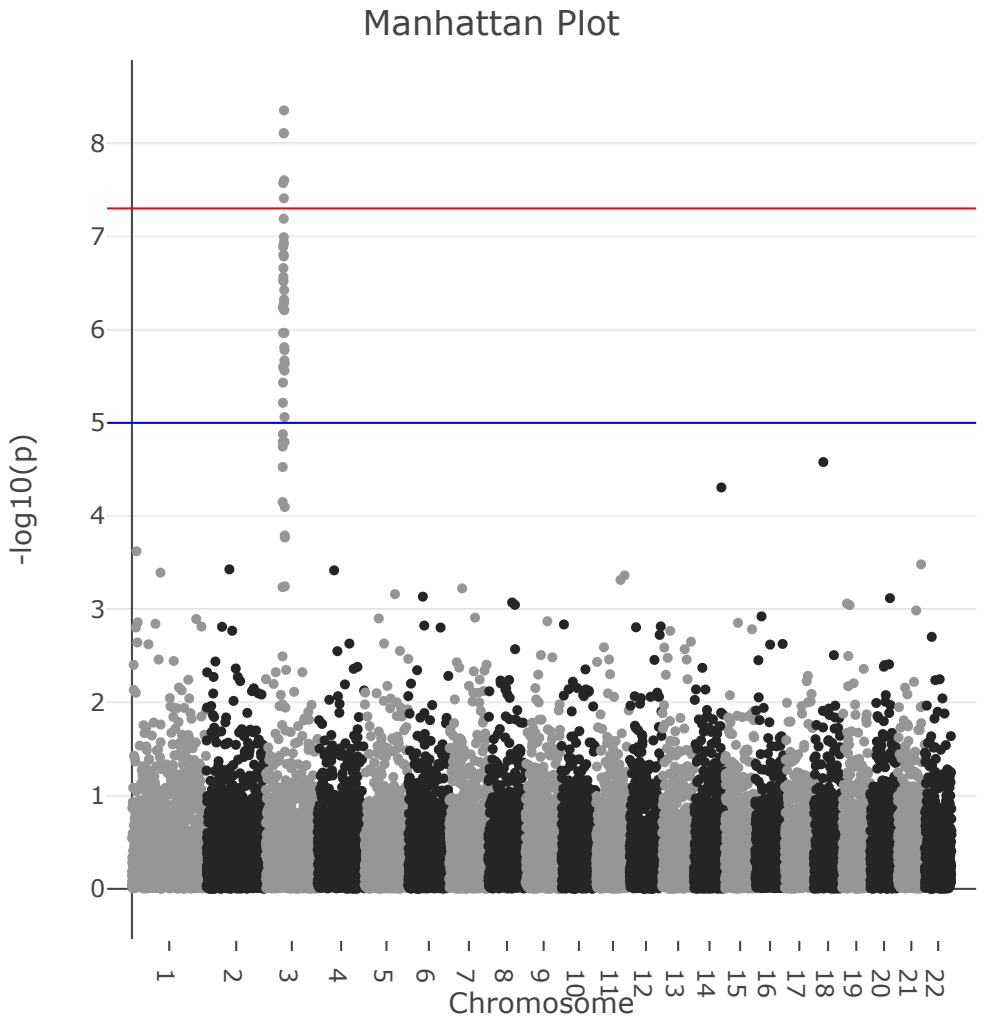


Manhattan plots

```
library(qqman)
manhattan(gwasResults)
```



```
library(manhattanly)
manhattanly(gwasResults)
```



Interactions using plotly

- Plot.ly is a web service that produces interactive graphics from data
- They made their backend open-source
- In R, you can interact with plot.ly using the package `plotly`.

Interactions using plotly

- Plotly makes it very easy to create interactive plots based on ggplot

```
plt_corrupt
```

Interactions using plotly

- Plotly makes it very easy to create interactive plots based on ggplot

```
library(plotly)

ggplotly(plt_corrupt)
```