Data Visualization for Environmental Epidemiology with ggplot2: Mastering Presentation Grade Figures

Alexandra E. Larsen, Alison K. Krajewski, Lauren H. Wyatt

7/7/2020

Contents

1 Dependencies

These are the libraries and packages we will be using today. If any of them are not loaded already in your RStudio, use [install.packages("library-name")].

```
library(ggplot2)
library(tidyr)
library(tidyverse)
library(datasets)
library(RColorBrewer)
```

```
library(colorRamps)
library(sf)
library(maps)
library(dplyr)
library(data.table)
library(stringr)
library(viridis)
library(psych)
library(mlbench)
library(extrafont)
```

2 Datasets

```
# Built-in data sets
data(iris)
data(faithful)
data(cars)
data(mpg)
data(Ozone)

# Created by Alison and Lauren
setwd("C:\\Users\\lwyatt\\Desktop\\Wyatt\\Workshops\\DataViz_ggplot2\\Data\\Data_for_github\\") # updat
air          <- read.csv("Criteria_Air_Pollutants.csv")
ht_top5      <-  read.csv("Hypertension_Top5.csv")
ht_top5_wide <- read.csv("Hypertension_Top5_Wide.csv")
air_long     <- read.csv("CriteriaAirPollutants_Long.csv")
```

3 Formating Data for GGPlot2

There are many libraries available in R for data manipulation, but we will be focusing on the package [tidyr]. Tidyr uses a pipe operator, just like ggplot2. We've loaded tidyr in the Dependencies section.

3.1 The Pipe Operator

Let's start with some exercises to get familiar with the pipe operator, [%>%]. For this example, we'll generate some N(0,1) data and take the mean.

```
set.seed(1234)
sample_data <- rnorm(n = 1000)
mean(sample_data)
```

```
## [1] -0.0265972
```

We can use a pipe operator like so:

```
sample_data %>%
  mean()
```

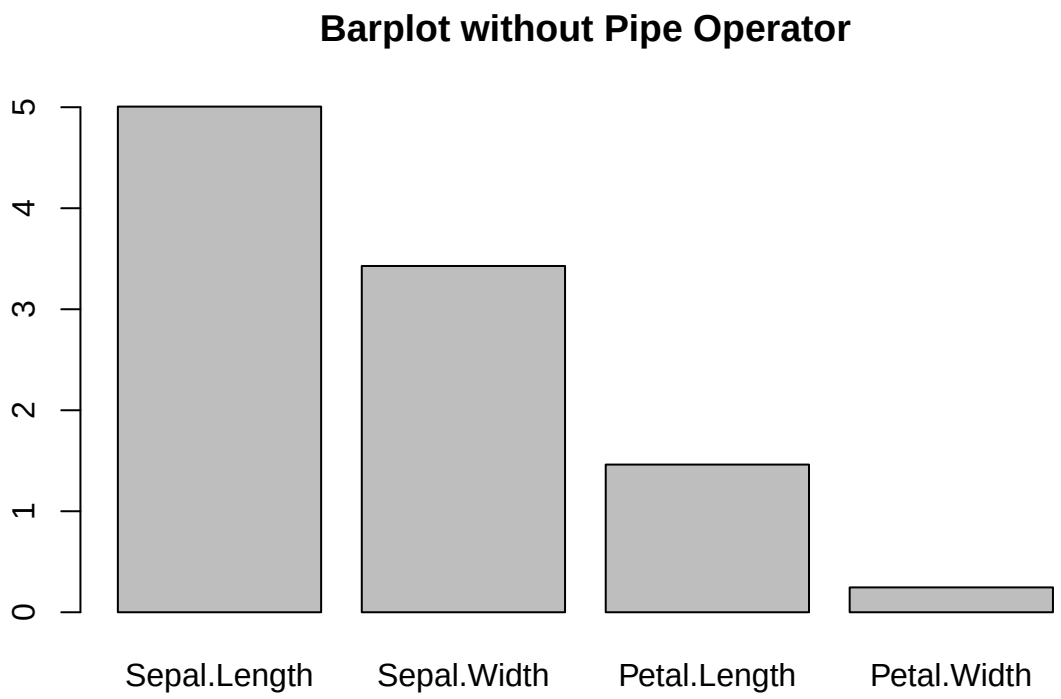
```
## [1] -0.0265972
```

Here is another example where we create a barplot of some of the iris data.

```r
head(iris)
```

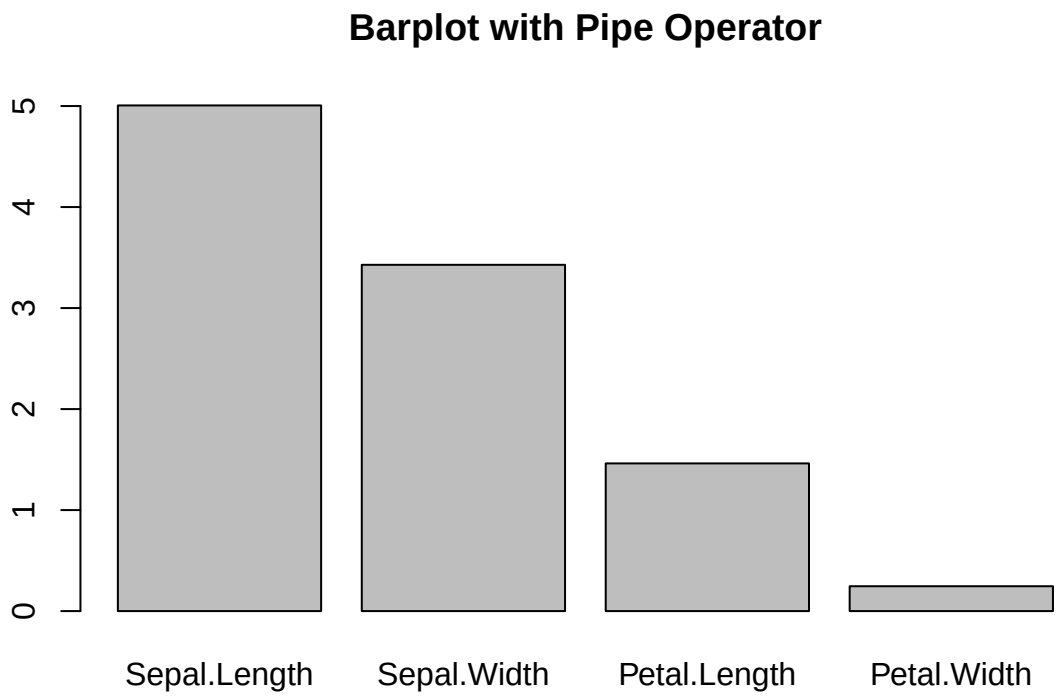
```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```r
barplot(colMeans(iris[iris$Species == "setosa", 1:4]), main = "Barplot without Pipe Operator")
```



Using the pipe operator instead:

```r
iris %>%
  filter(Species == "setosa") %>%
  select(Sepal.Length:Petal.Width) %>%
  colMeans() %>%
  barplot(main = "Barplot with Pipe Operator")
```



In the example using the pipe operator above, notice that we didn't have to create any variables and we can avoid long lines of code. Futher, tidyr has intuitive functions that we can take advantage of as we'll see in the next section.

3.2 Long vs. Wide

One of the most common data transformations that you need with using ggplot2() is converting long to wide and visa versa.

Tidyr refers to changing from long to wide as "pivoting" and uses pivot_longer() and pivot_wider(). The pivot functions have replaced gather() and spread() as the tidyr function for lengthening or widening data. Other packages are available to do this - you may be familiar with Reshape2, which uses melt() and dcast().

A good resource on gather(), spread(), or the reshape2 options is here.

We'll start with going from *wide to long data* in the iris data.

```
iris_long <- iris %>%
  pivot_longer(
    cols = Sepal.Length:Petal.Width,
    names_to = "Part",
    values_to = "Measurement")
head(iris_long)
```

```
## # A tibble: 6 x 3
##   Species Part         Measurement
##   <fct>   <chr>              <dbl>
## 1 setosa  Sepal.Length         5.1
```

```
## 2 setosa  Sepal.Width          3.5
## 3 setosa  Petal.Length         1.4
## 4 setosa  Petal.Width          0.2
## 5 setosa  Sepal.Length         4.9
## 6 setosa  Sepal.Width          3
```

```
# going back to wide... need another data set?
```

4 Basics of ggplot2

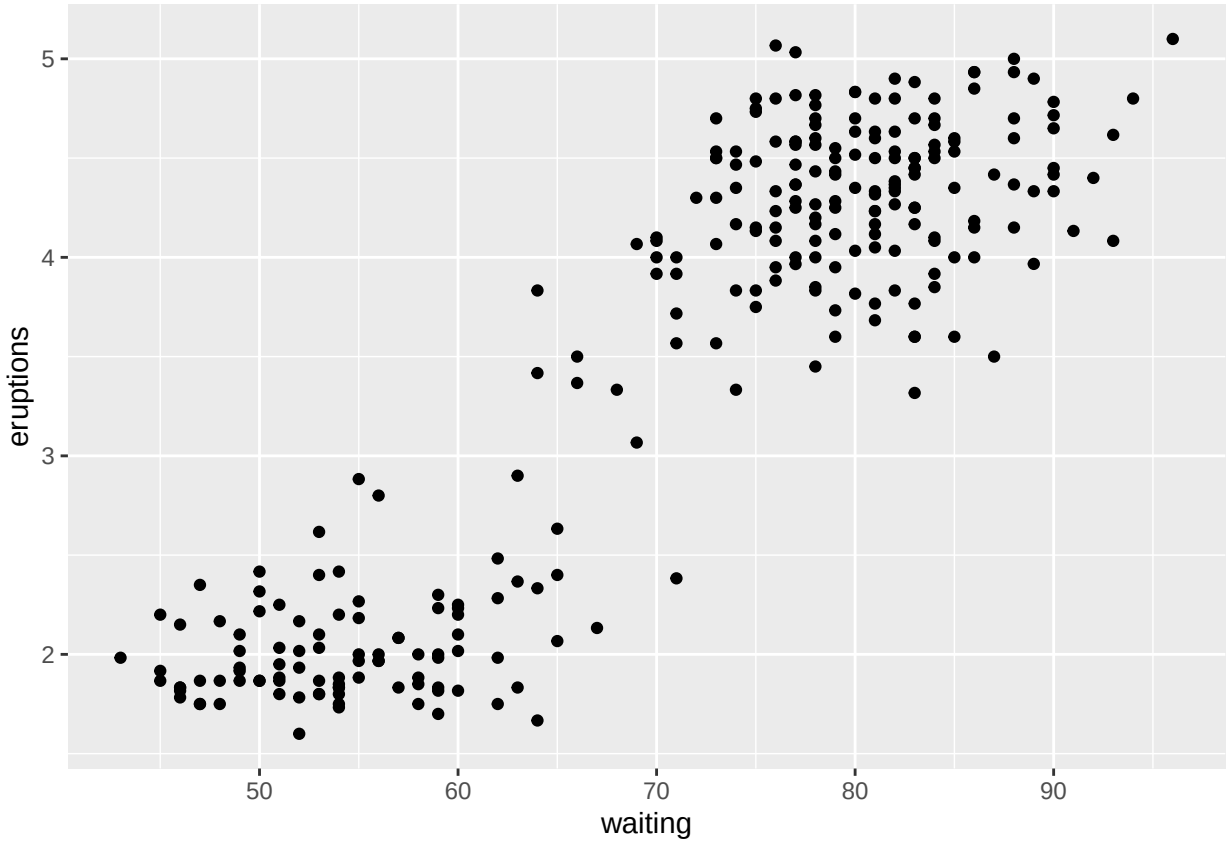
In this section, we'll spend time with data and aesthetic mapping, and try out a few types of graphs.

Let's start with a basic scatterplot of eruptions from Old Faithful.

```
head(faithful)
```

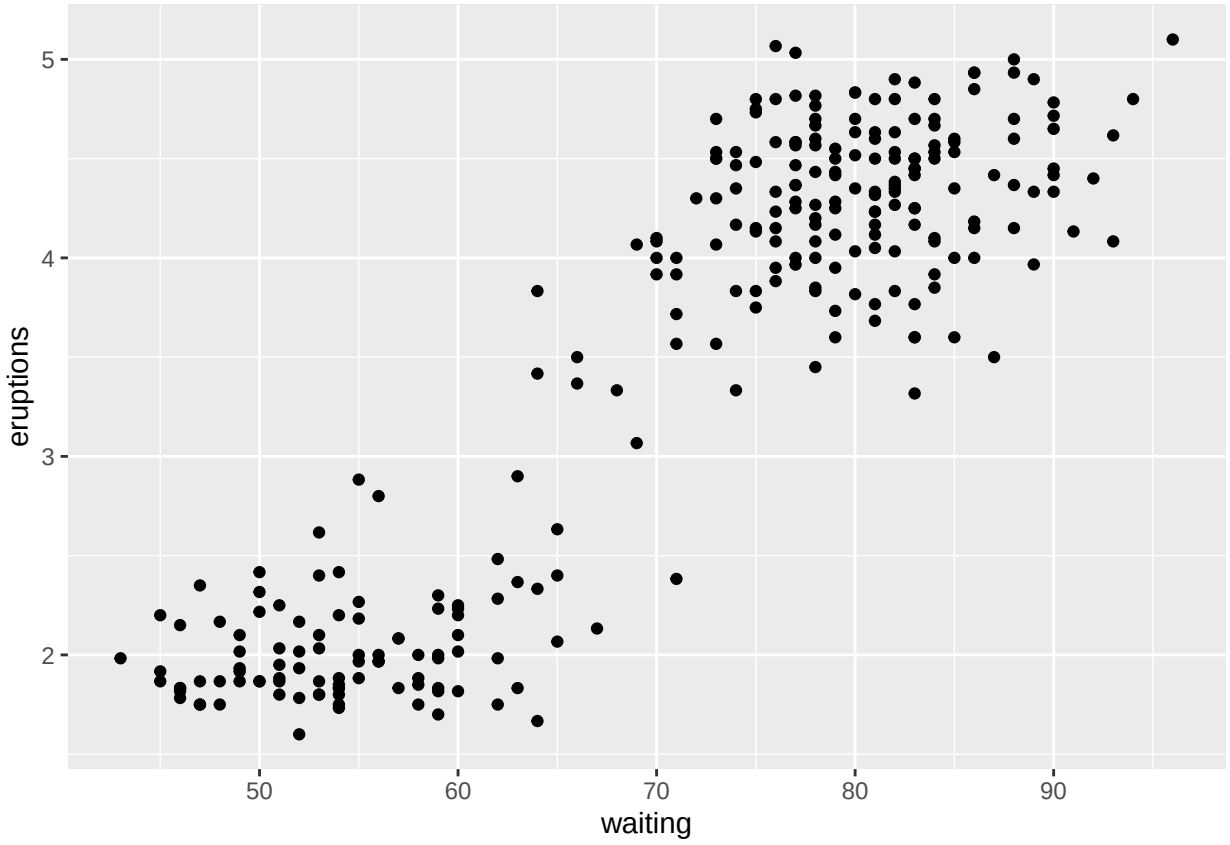
```
##   eruptions waiting
## 1     3.600      79
## 2     1.800      54
## 3     3.333      74
## 4     2.283      62
## 5     4.533      85
## 6     2.883      55
```

```
ggplot(data = faithful,
       mapping = aes(x = waiting, y = eruptions)) +
  geom_point()
```



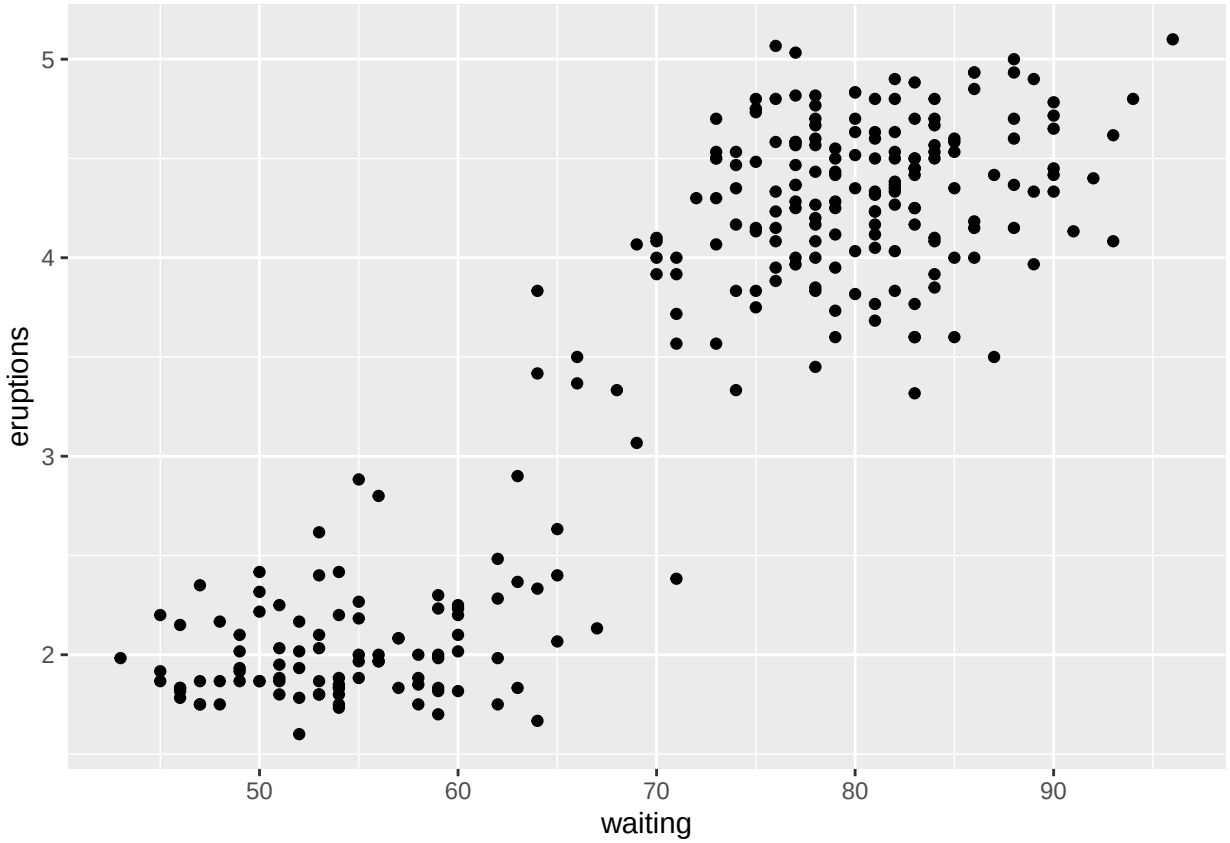
We can specify data and aesthetics in the layer too

```
ggplot() +
  geom_point(data = faithful,
             mapping = aes(x = waiting, y = eruptions))
```



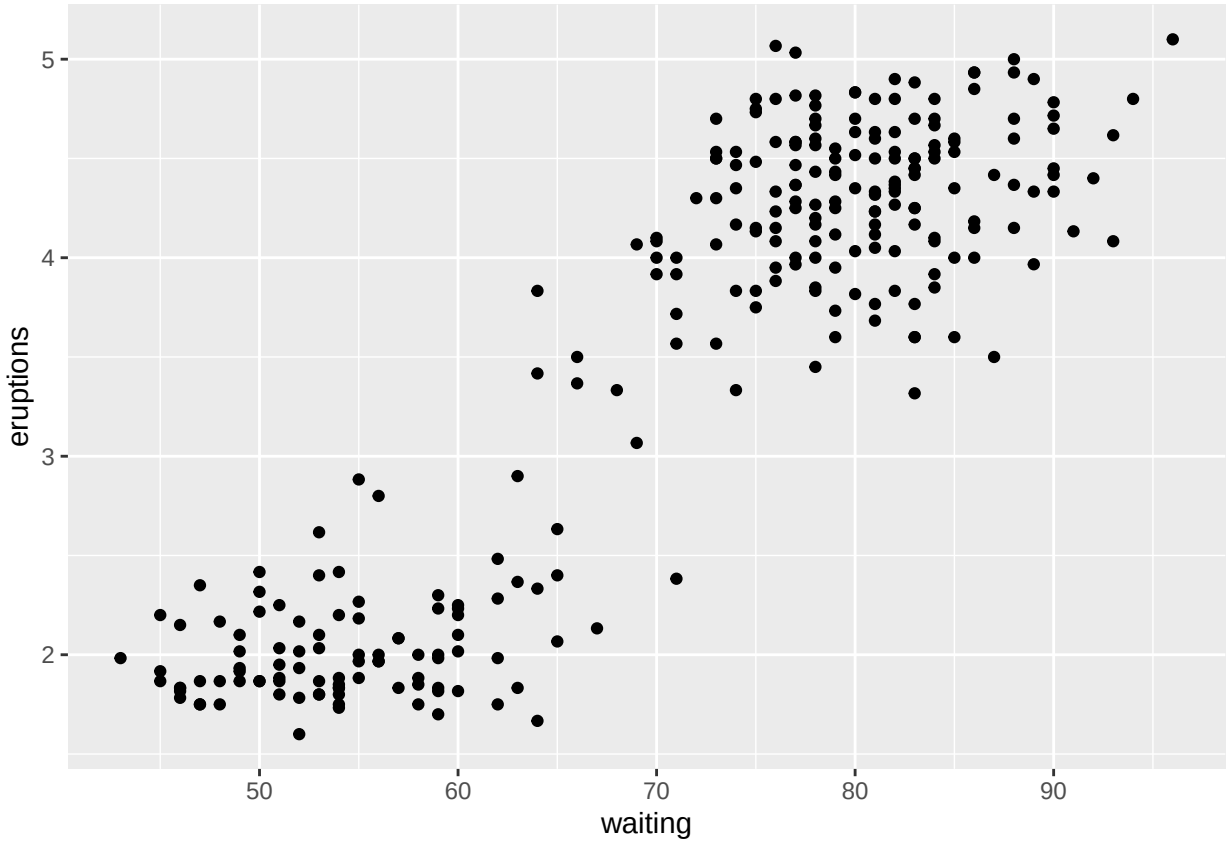
Or a mixture of both

```
ggplot(data = faithful) +
  geom_point(mapping = aes(x = waiting, y = eruptions))
```



And the data doesn't have to come first.

```
ggplot(mapping = aes(x = waiting, y = eruptions)) +
  geom_point(data = faithful)
```



But, you do need both. If you don't specify data within the ggplot, you need $:

```
ggplot(mapping = aes(x = waiting, y = eruptions)) +
  geom_point()
```

And if you don't specify mapping, nothing will happen. Remember, mapping is what connects the data to your plot.

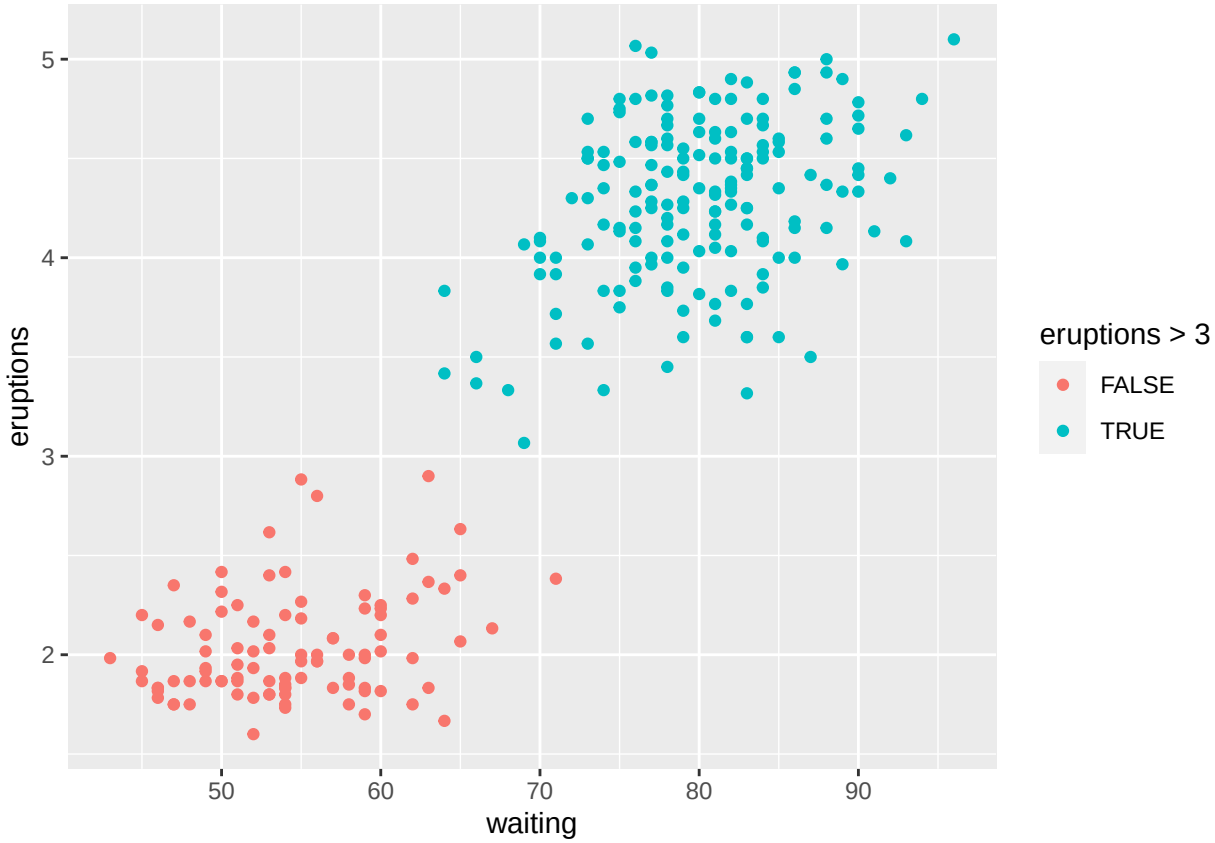
```
ggplot(data = faithful) +
  geom_point()
```

Without the mapping argument and aes, can't "talk" to the variables in the data.

```
ggplot() +
  geom_point(data = faithful, x = waiting, y = eruptions)
```

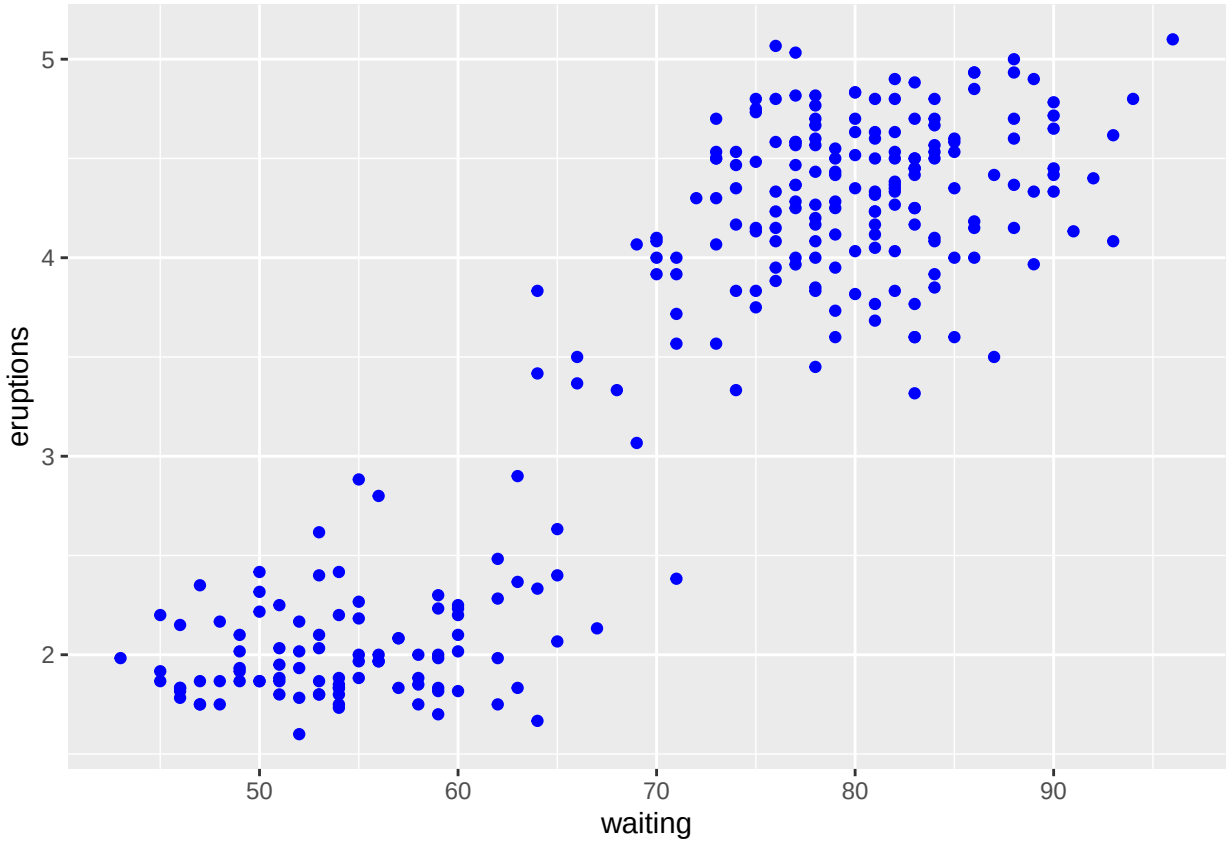
If you want to change an aesthetic, say color, based on the data, include in mapping

```
ggplot() +
  geom_point(data = faithful,
             mapping = aes(x = waiting, y = eruptions, color = eruptions > 3))
```



Otherwise, place it outside of mapping; it will apply to all the data.

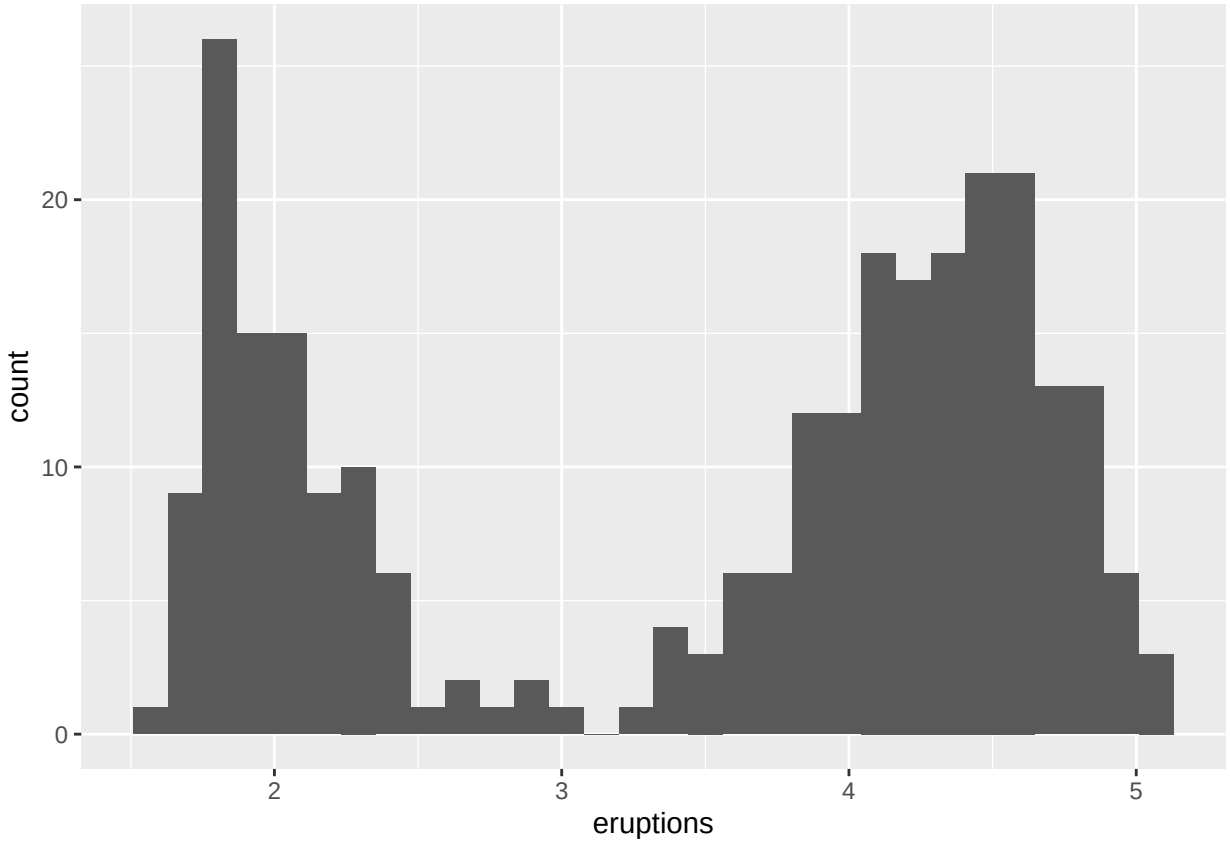
```
ggplot() +
  geom_point(data = faithful,
             mapping = aes(x = waiting, y = eruptions),
             color = "blue")
```



Let's look at a histogram of the eruptions.

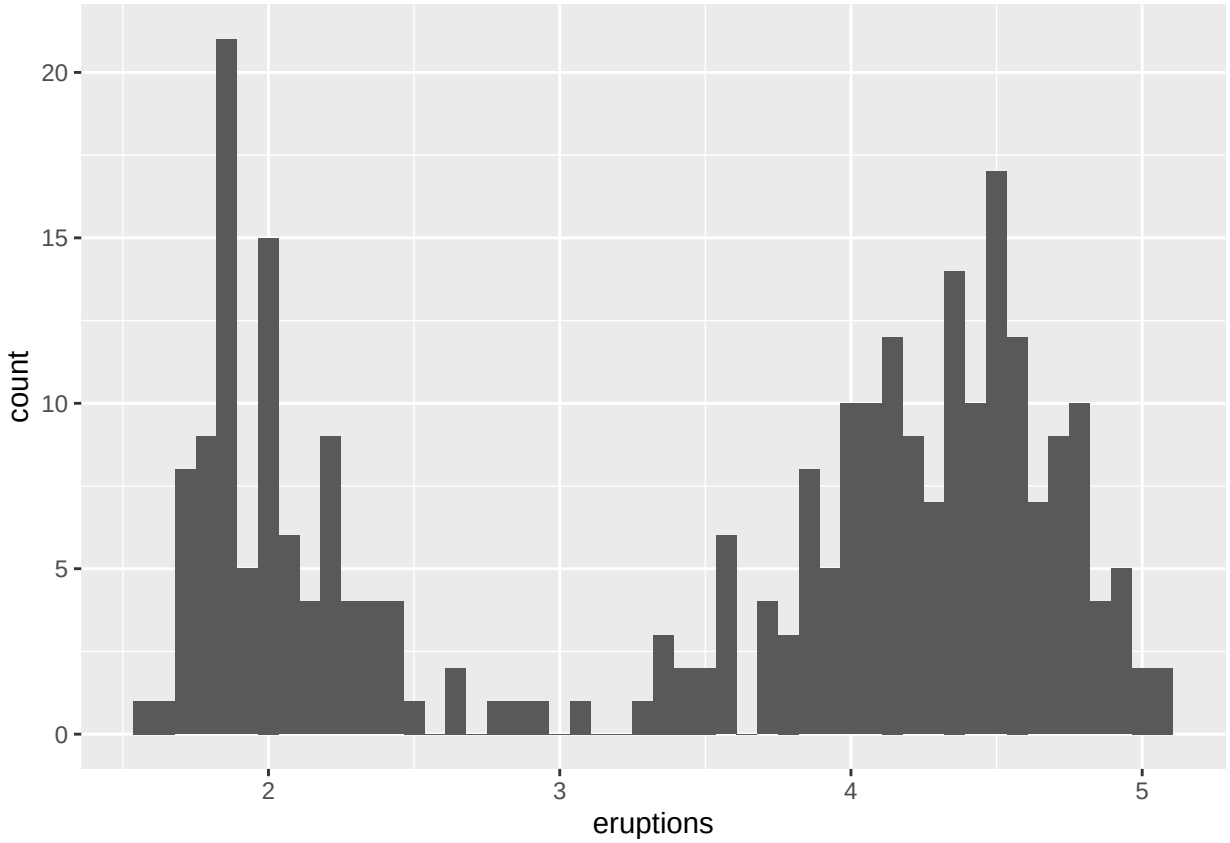
```
ggplot() +
  geom_histogram(data = faithful,
                 mapping = aes(x = eruptions))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Change with width of each bin from default (30)

```
ggplot() +
  geom_histogram(data = faithful,
                 mapping = aes(x = eruptions),
                 bins = 50)
```



Let's make a box plot with the setosa data from iris.

```
setosa <- iris %>%
  filter(Species == "setosa") %>%
  select(-Species)
head(setosa)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1          5.1         3.5          1.4         0.2
## 2          4.9         3.0          1.4         0.2
## 3          4.7         3.2          1.3         0.2
## 4          4.6         3.1          1.5         0.2
## 5          5.0         3.6          1.4         0.2
## 6          5.4         3.9          1.7         0.4
```

For the boxplot, we'll want the parts of the flower on the x-axis and the measurements on the y-axis, so we need "parts" and "measurements" to be variables or columns in the data set. To get there, we can pivot the data to long format like we did earlier. Let's re-use that code for our setosa subset:

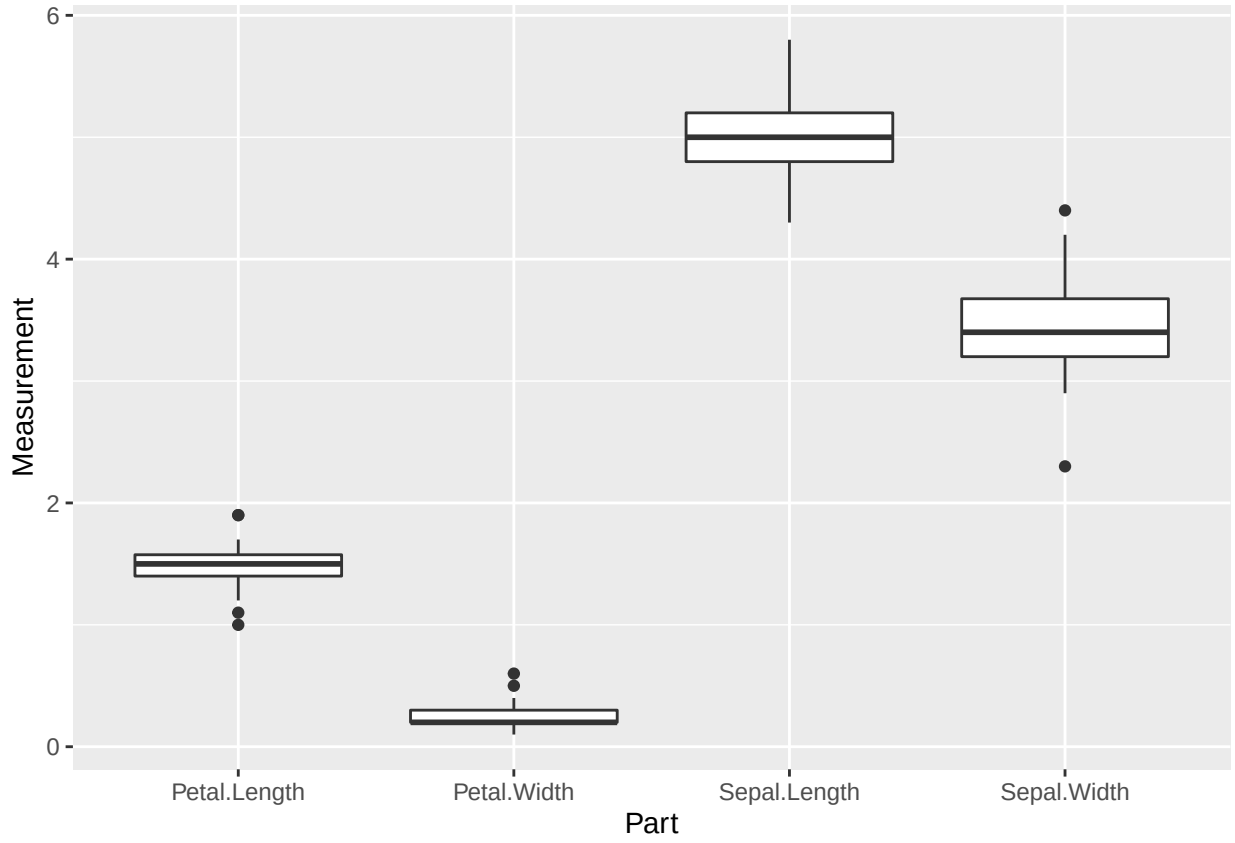
```
setosa_long <- setosa %>%
  pivot_longer(
    cols = Sepal.Length:Petal.Width,
    names_to = "Part",
    values_to = "Measurement")
head(setosa_long)
```

```
## # A tibble: 6 x 2
##   Part         Measurement
```

```
##   <chr>              <dbl>
## 1 Sepal.Length         5.1
## 2 Sepal.Width          3.5
## 3 Petal.Length         1.4
## 4 Petal.Width          0.2
## 5 Sepal.Length         4.9
## 6 Sepal.Width          3
```

Now, we can make our boxplot.

```
ggplot() +
  geom_boxplot(data = setosa_long, aes(x = Part, y = Measurement))
```



5 Scales

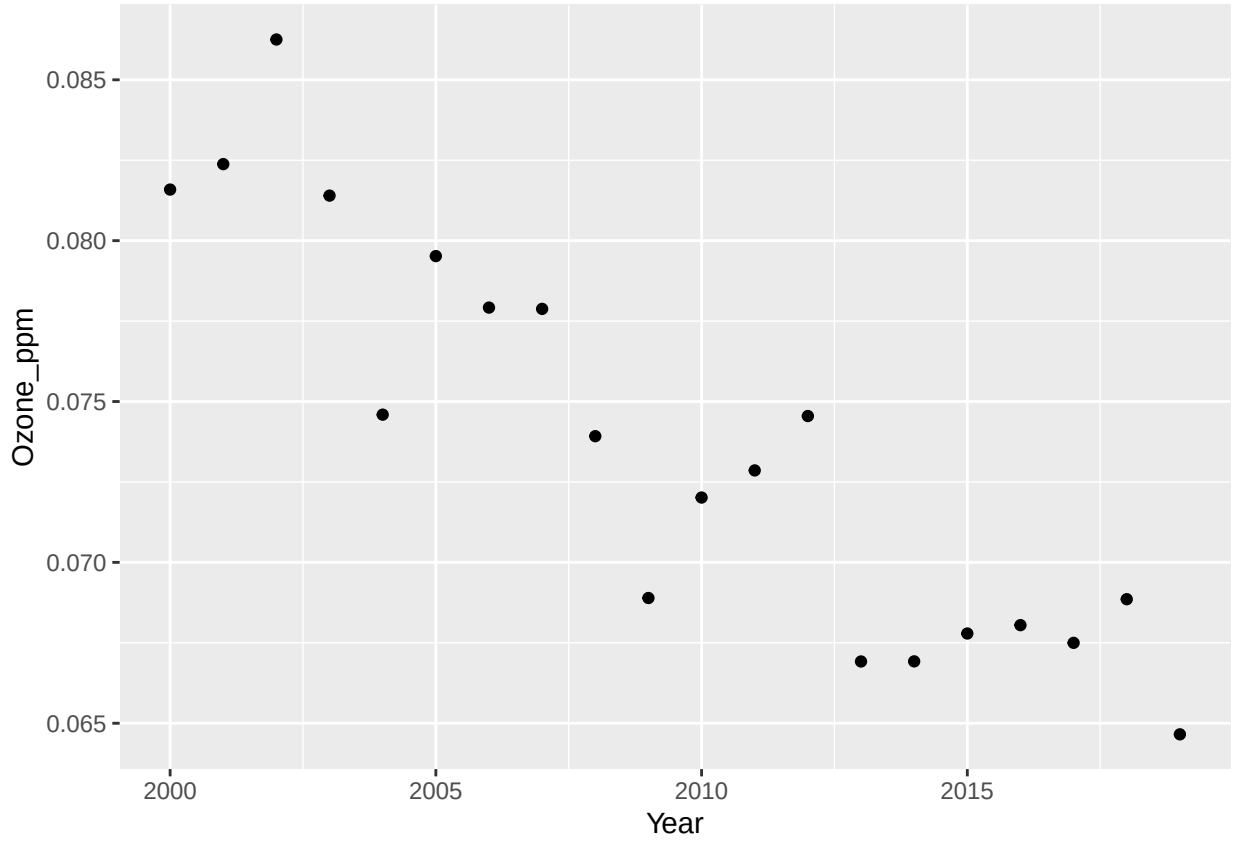
Everything inside the aes() will have scales

5.1 Modify Scales

Scales are made up of three pieces separated by an underscore (_)

Without specifying scales, the defaults are used.

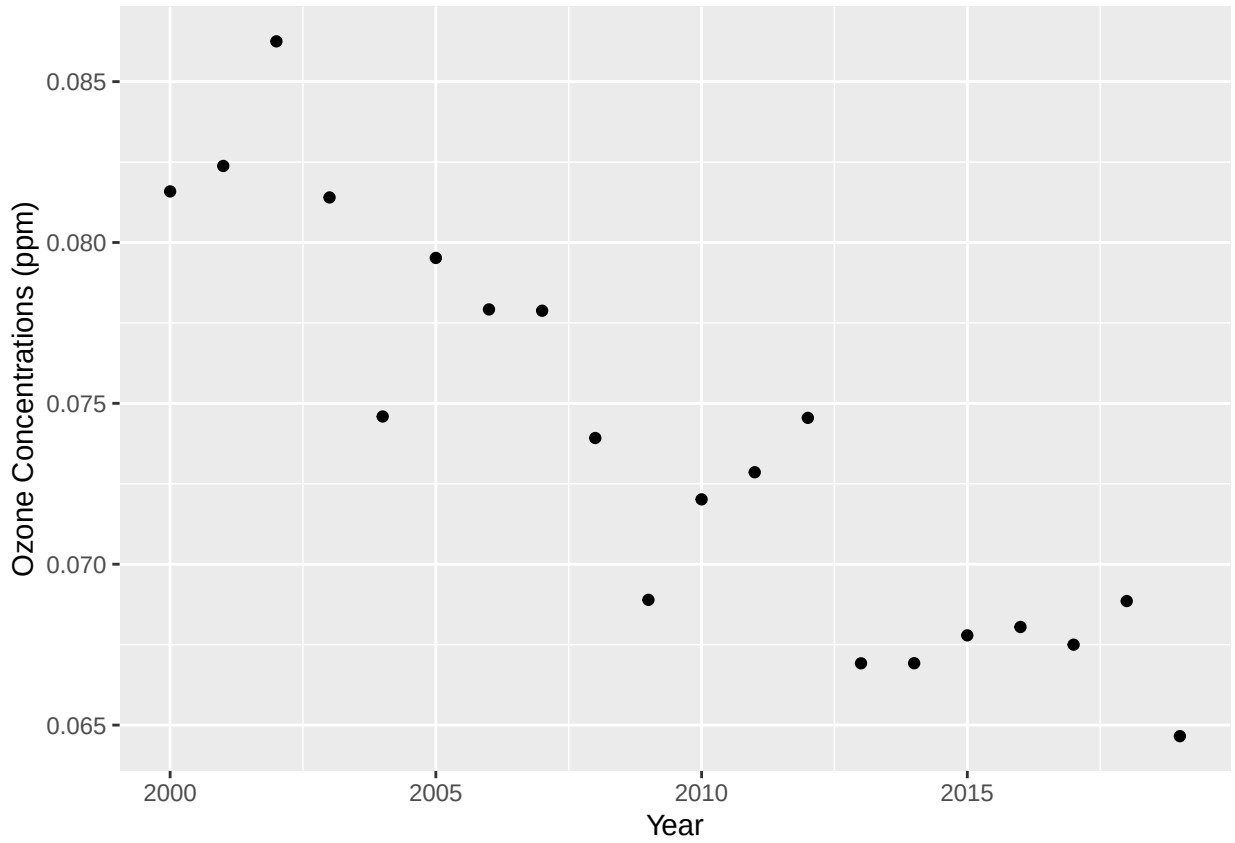
```
#Example of Basic Sccatterplot Without Specifying Scales
ozone_plot <- ggplot(air, aes(Year, Ozone_ppm)) + geom_point()
ozone_plot
```



5.2 Breaks and Labels

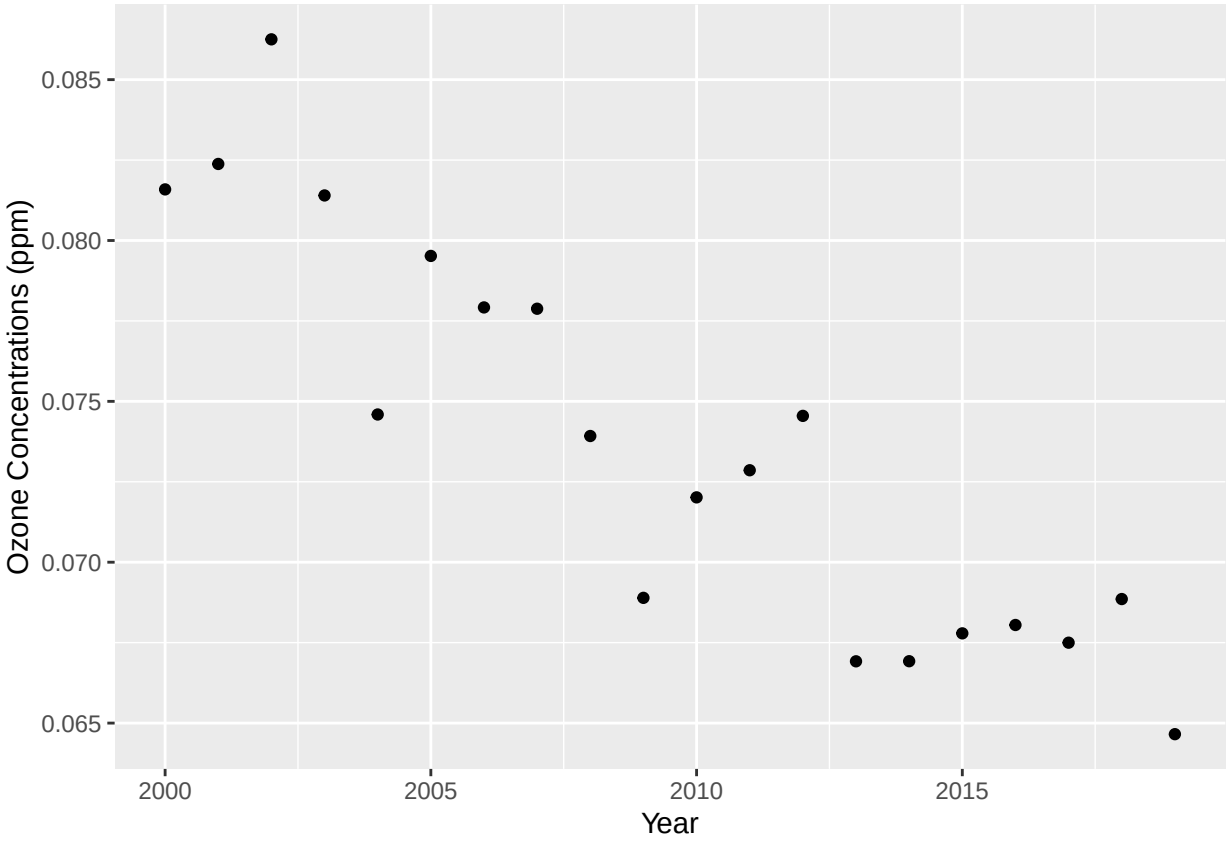
This example changes the lab name from the variable using scales.

```
#Adding in Labels
ozone_plot <- ggplot(air, aes(x=Year, y=Ozone_ppm)) + geom_point()
ozone_plot + scale_x_continuous(name="Year") + scale_y_continuous(name="Ozone Concentrations (ppm)")
```



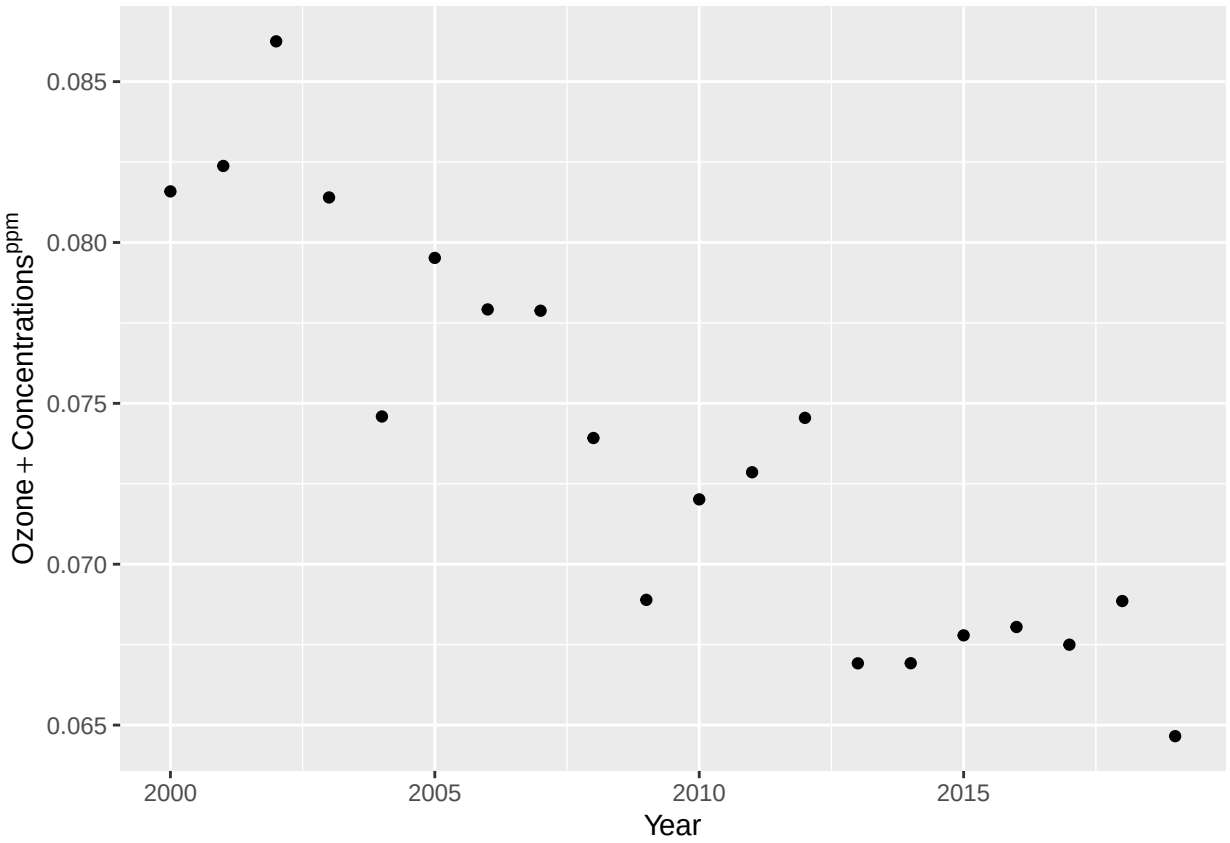
This examples uses the xlab and ylab options.

```
#Another way to add labels
ozone_plot <- ggplot(air, aes(x=Year, y=Ozone_ppm)) + geom_point() +
  xlab("Year") + ylab("Ozone Concentrations (ppm)")
ozone_plot
```

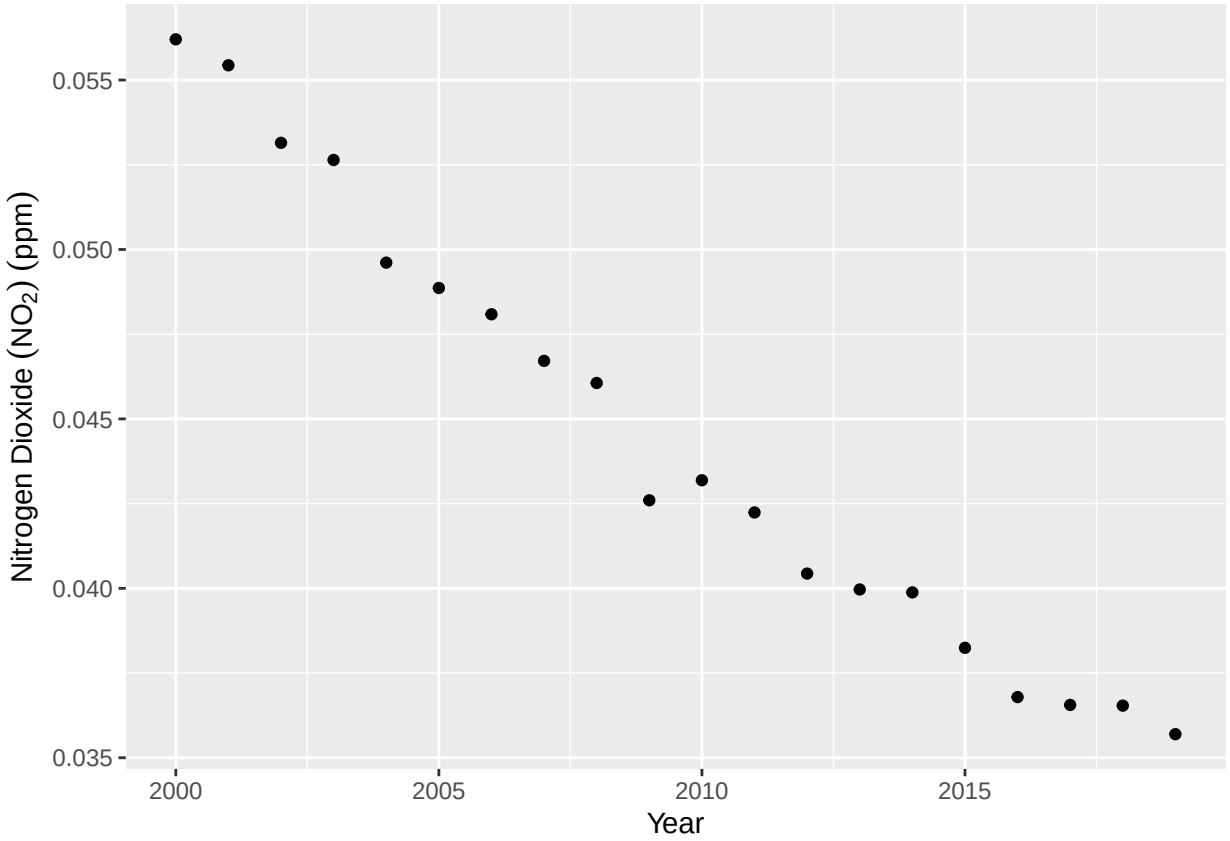


Labels can also be created for mathematical expressions.

```
#Example of X-Axis Title with A Mathematicel Expression to the Title
ozone_plot <- ggplot(air, aes(x=Year, y=Ozone_ppm)) + geom_point()
ozone_plot + scale_x_continuous(name="Year") +
  scale_y_continuous(quote(Ozone + Concentrations ^ ppm))
```

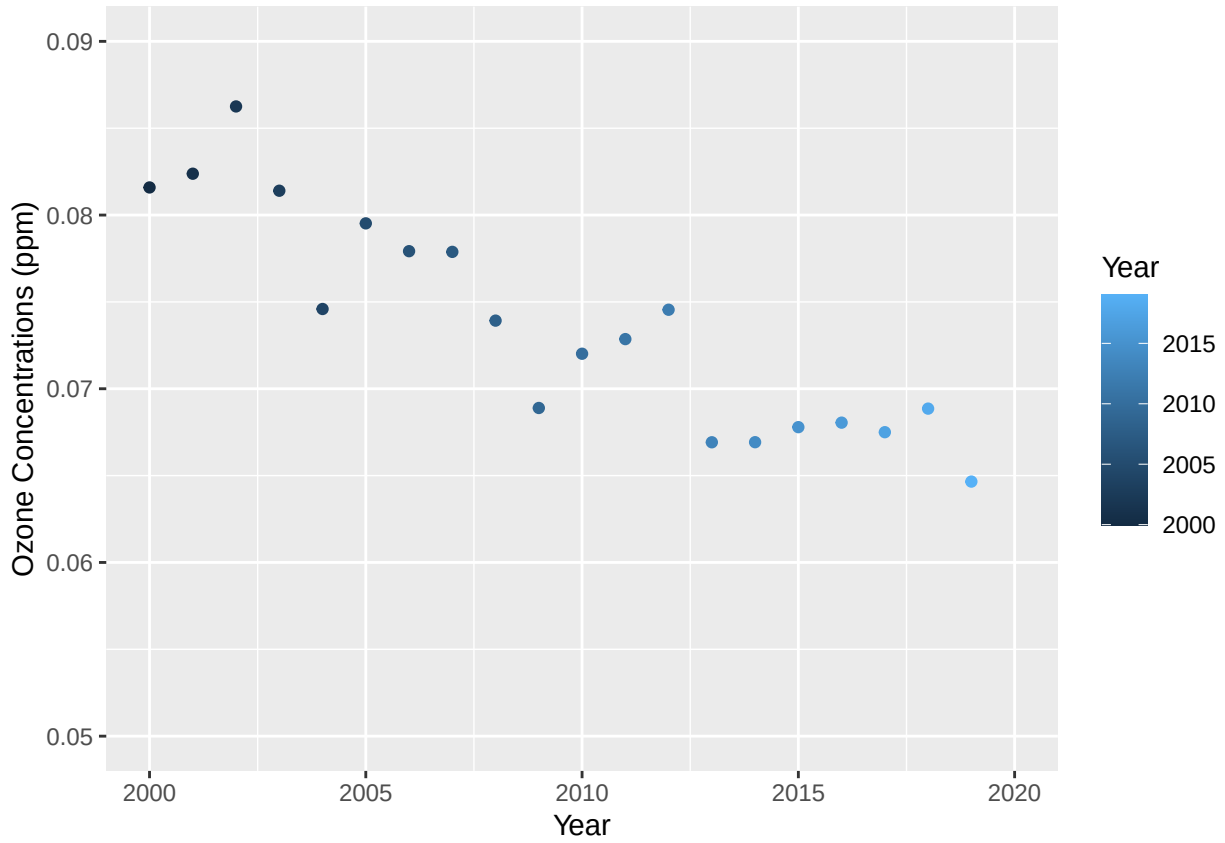



```
#Another example with Superscripts and Subscripts
NO2_plot_script <- ggplot(air, aes(x=Year, y=Nitrogen_Dioxide_ppm)) + geom_point() +
  xlab("Year") + ylab(expression(Nitrogen~Dioxide~(NO[2])~(ppm)))
NO2_plot_script
```



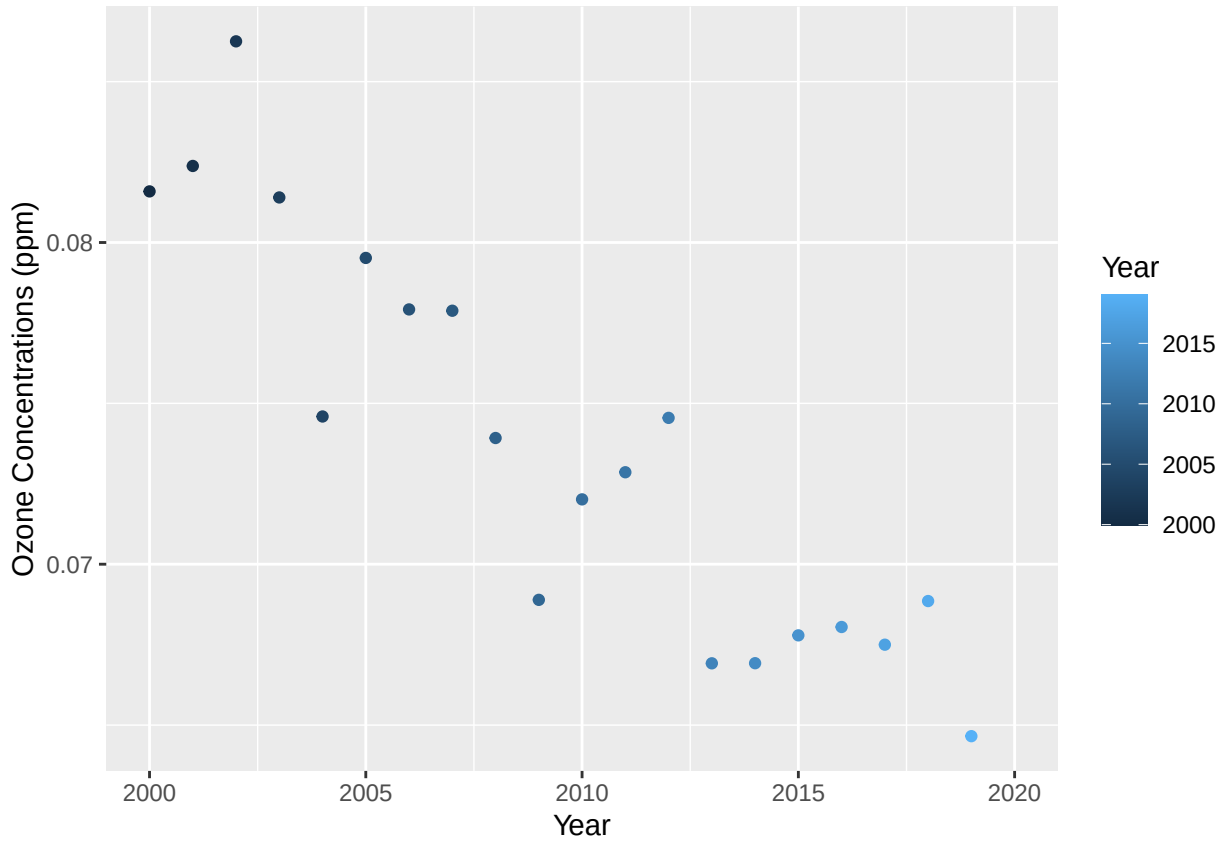
In addition to modifying the scales, you can also modify limits.

```
#Example with Modifying Limits
ozone_plot <- ggplot(air, aes(x=Year, y=Ozone_ppm)) + geom_point(aes(colour=Year)) +
  scale_x_continuous(name="Year", limits=c(2000,2020)) +
  scale_y_continuous(name="Ozone Concentrations (ppm)", limits=c(0.050, 0.090))
ozone_plot
```



Break points at specified points can also be created.

```
#Example of Breaks
#Ozone Example
ozone_plot <- ggplot(air, aes(x=Year, y=Ozone_ppm)) + geom_point(aes(colour=Year)) +
  scale_x_continuous(name="Year", limits=c(2000,2020)) +
  scale_y_continuous(name="Ozone Concentrations (ppm)", breaks=c(0.06, 0.07, 0.08, 0.09))
ozone_plot
```

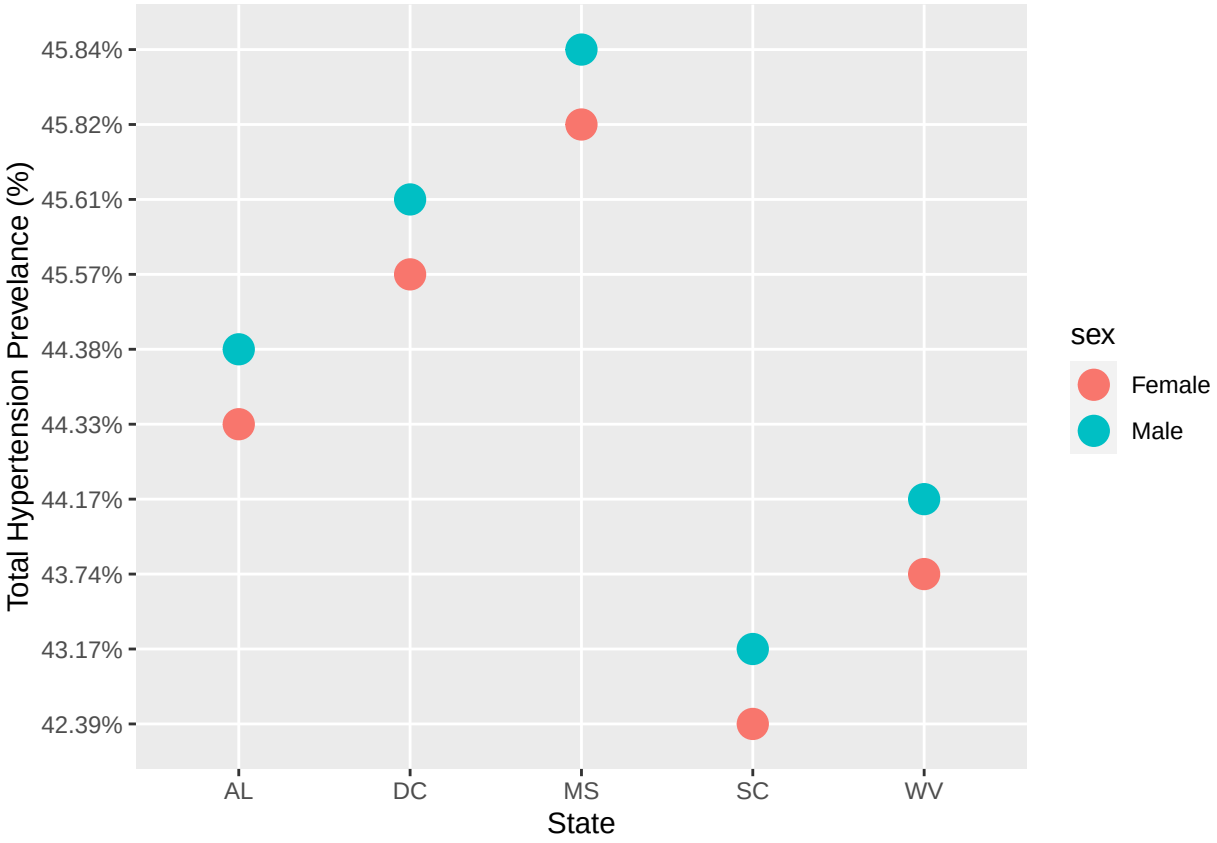


Hypertension example with the defaults

```
#Example of Labels
#Hypertension Example - Default
ht_top5_plot_labels <- ggplot(ht_top5, aes(x=State, y=Total, group=Sex, colour=sex)) +
  geom_point(aes(colour=Sex), size=5) +
  scale_x_discrete(name="State") +
  scale_y_discrete(name="Total Hypertension Prevelance (%)")
ht_top5_plot_labels
```

Hypertensions with breaks and labels

```
#Hypertension Example with Break and Labels
ht_top5_plot_labels <- ggplot(ht_top5, aes(x=State, y=Total, group=Sex, colour=sex)) +
  geom_point(aes(colour=Sex), size=5) +
  scale_x_discrete(name="State", labels=c("Alabama" = "AL", "District Of Columbia" = "DC", "Mississippi
                           "South Carolina" = "SC", "West Virginia" = "WV")) +
  scale_y_discrete(name="Total Hypertension Prevelance (%)")
ht_top5_plot_labels
```

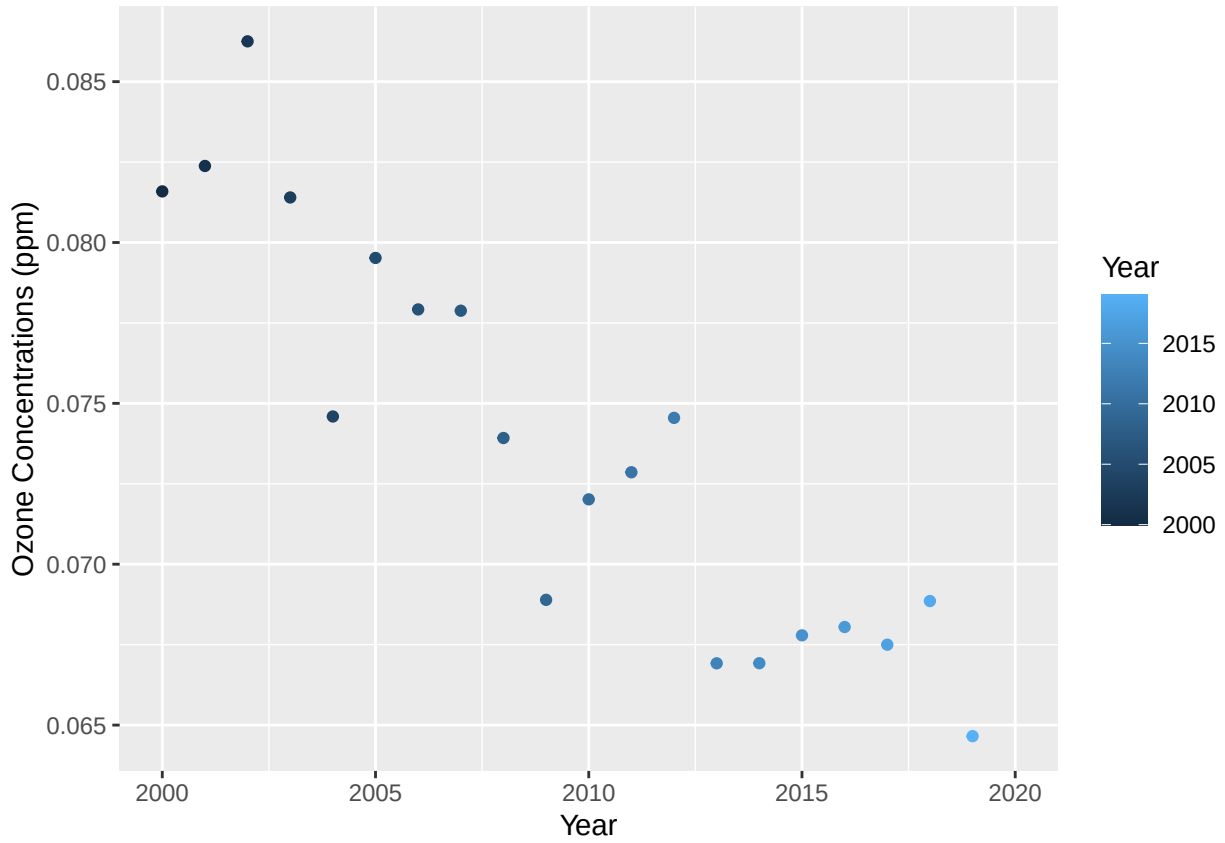


5.3 Legends

5.3.1 Changing the Position of the Legend

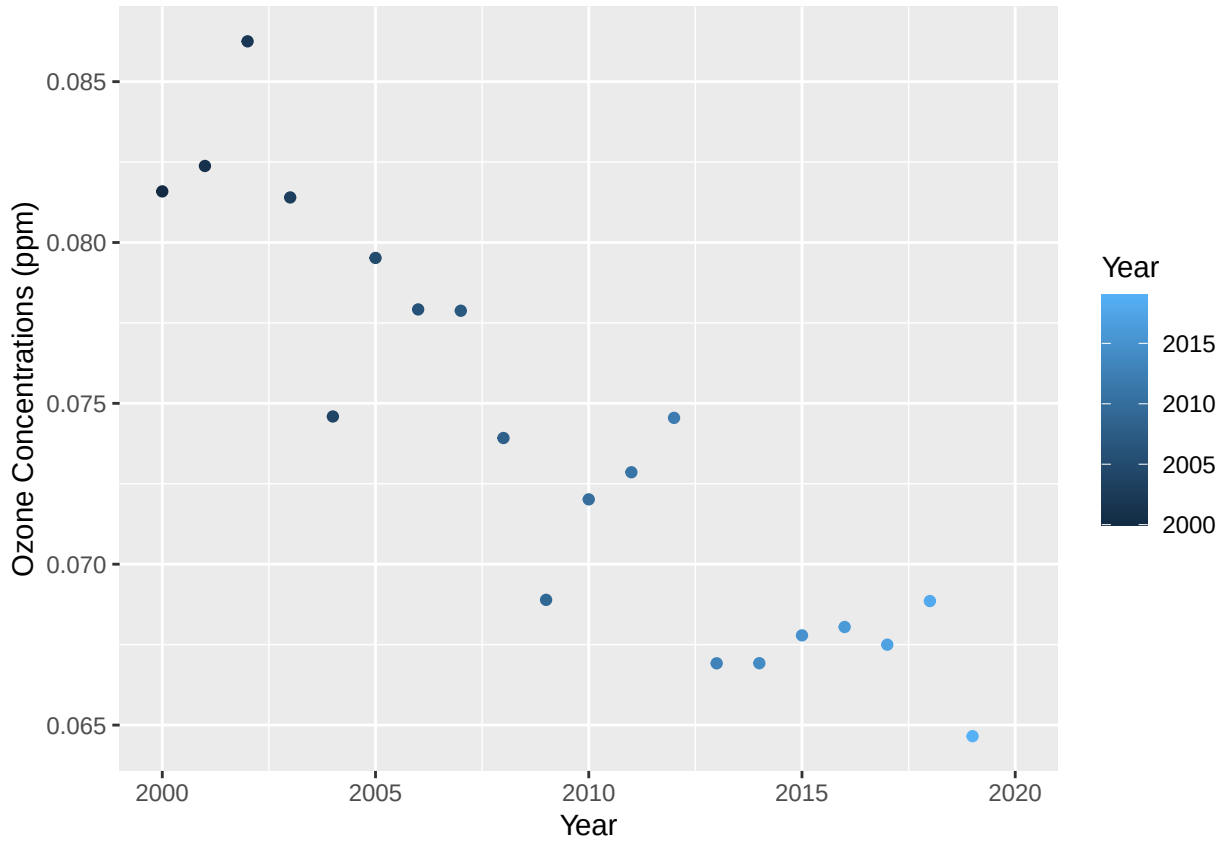
The default position is to the right of the figure.

```
ozone_plot <- ggplot(air, aes(x=Year, y=Ozone_ppm)) + geom_point(aes(colour=Year)) +
  scale_x_continuous(name="Year", limits=c(2000,2020)) +
  scale_y_continuous(name="Ozone Concentrations (ppm)")
ozone_plot
```



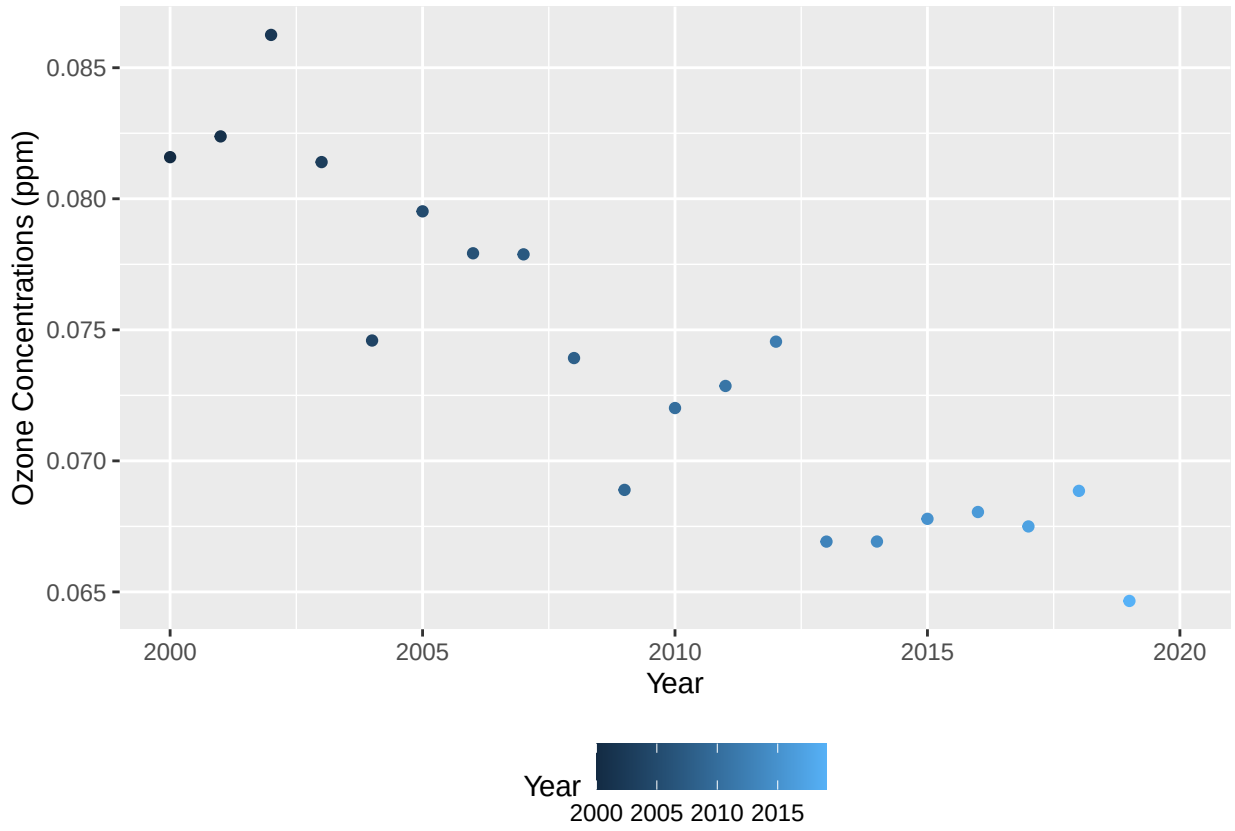
The designation of the legend to the right of the figure can also be specified.

```
ozone_plot <- ggplot(air, aes(x=Year, y=Ozone_ppm)) + geom_point(aes(colour=Year)) +
  scale_x_continuous(name="Year", limits=c(2000,2020)) +
  scale_y_continuous(name="Ozone Concentrations (ppm)") +
  theme(legend.position='right')
ozone_plot
```



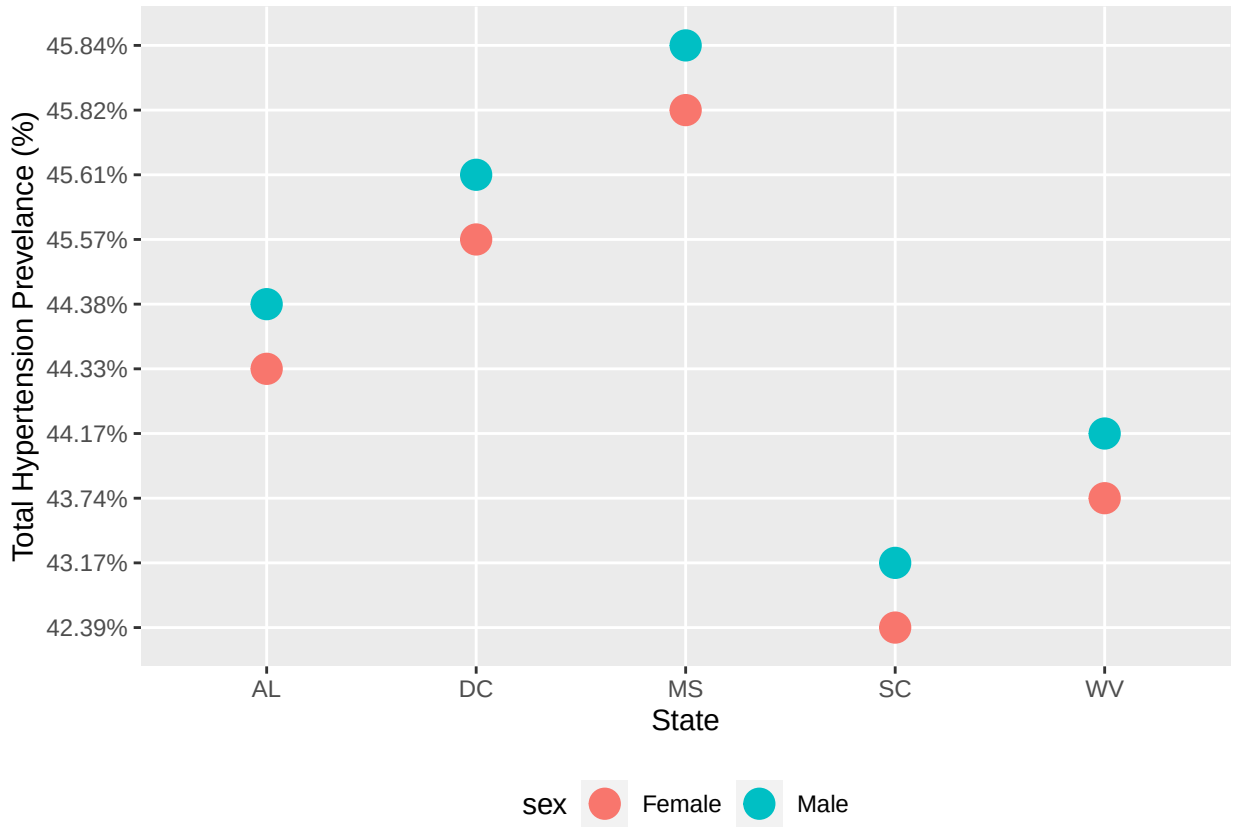
Positioning the legend on the bottom.

```
#Ozone Example
ozone_plot <- ggplot(air, aes(x=Year, y=Ozone_ppm)) + geom_point(aes(colour=Year)) +
  scale_x_continuous(name="Year", limits=c(2000,2020)) +
  scale_y_continuous(name="Ozone Concentrations (ppm)") +
  theme(legend.position="bottom")
ozone_plot
```



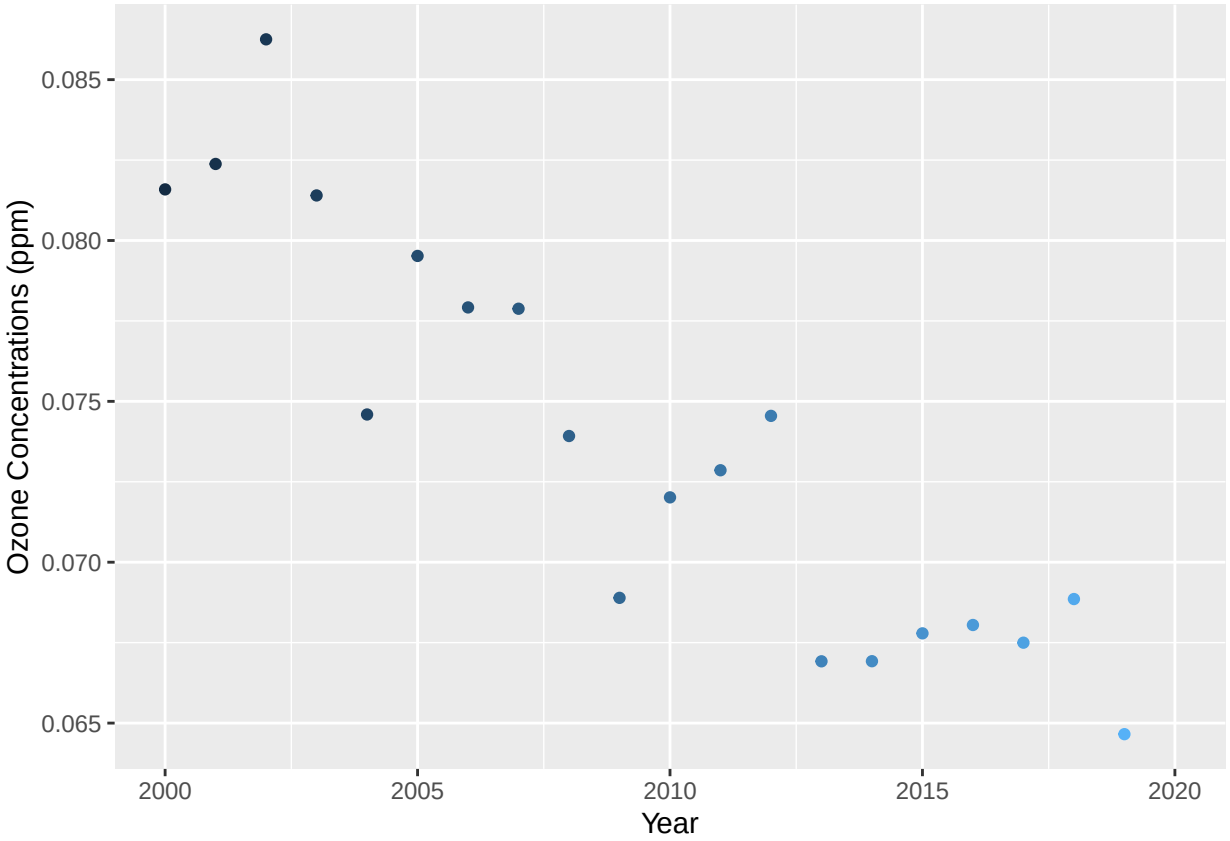
The legend can also be modified so that all the values of the legend will be in single row.

```
#Hypertension Example
ht_top5_plot_legend <- ggplot(ht_top5, aes(x=State, y=Total, group=Sex, colour=sex)) +
  geom_point(aes(colour=Sex), size=5) +
  scale_x_discrete(name="State", labels=c("Alabama" = "AL", "District Of Columbia" = "DC", "Mississippi
                           "South Carolina" = "SC", "West Virginia" = "WV")) +
  scale_y_discrete(name="Total Hypertension Prevelance (%)") +
  theme(legend.position="bottom") + guides(colour=guide_legend(nrow=1))
ht_top5_plot_legend
```



The legend can also be removed.

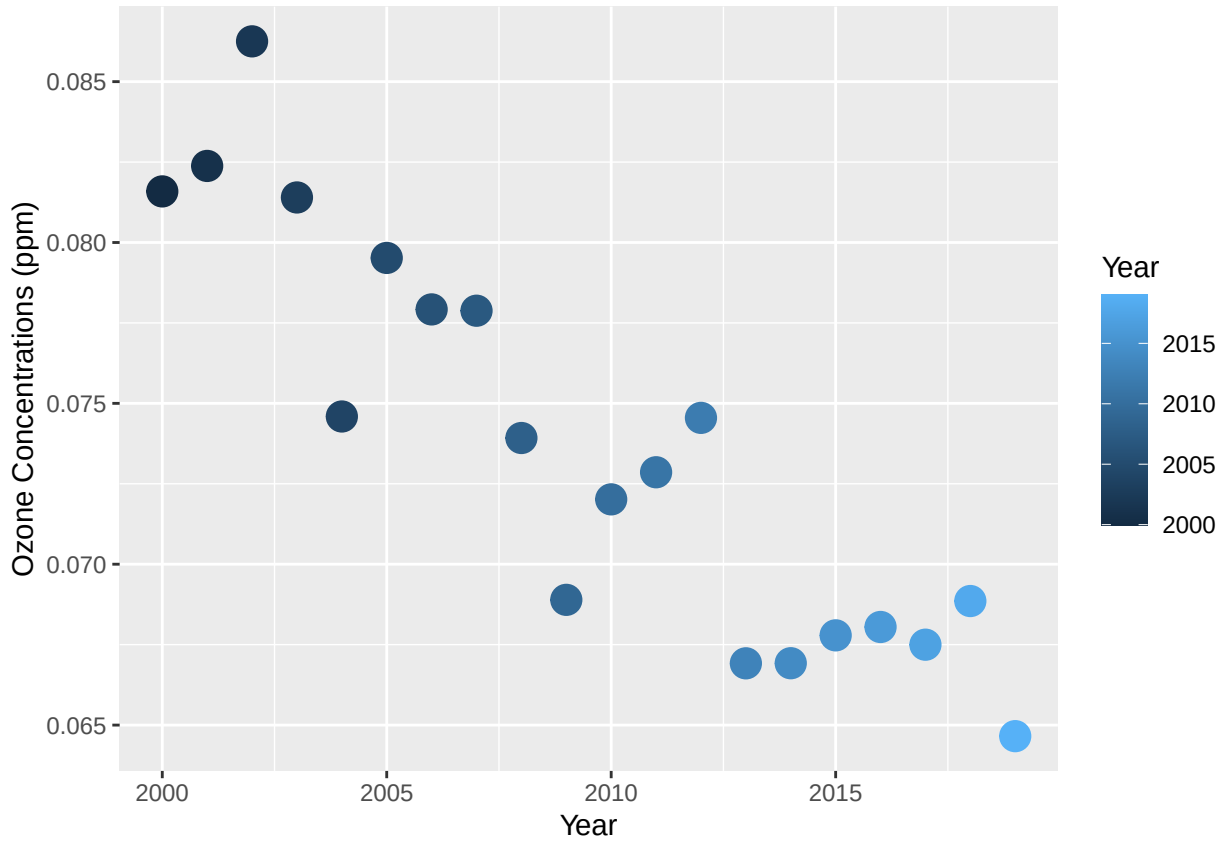
```
ozone_plot <- ggplot(air, aes(x=Year, y=Ozone_ppm)) + geom_point(aes(colour=Year)) +
  scale_x_continuous(name="Year", limits=c(2000,2020)) +
  scale_y_continuous(name="Ozone Concentrations (ppm)") +
  theme(legend.position="none")
ozone_plot
```



5.4 Colors

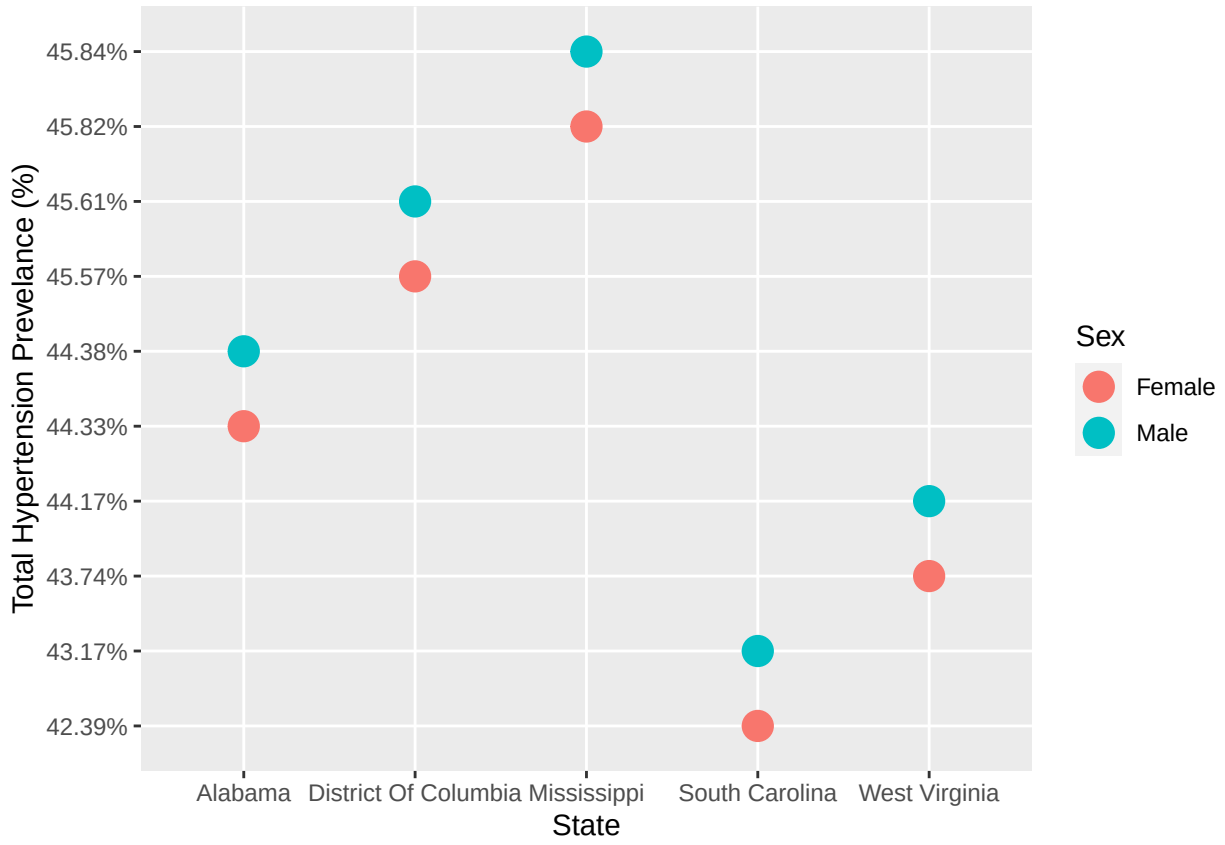
General considerations: * Avoid using red-green gradients * Red is alarming and can imply something bad * Avoid using rainbow gradients * Keep color schemes color blind friendly (Reference: http://www.cookbook-r.com/Graphs/Colors_(ggplot2)/#a-colorblind-friendly-palette)

```
#Adding in a scale for color
#Ozone Example
ozone_plot <- ggplot(air, aes(x=Year, y=Ozone_ppm)) + geom_point(aes(colour=Year), size=5) +
  scale_x_continuous(name="Year") +
  scale_y_continuous(name="Ozone Concentrations (ppm)")
ozone_plot
```



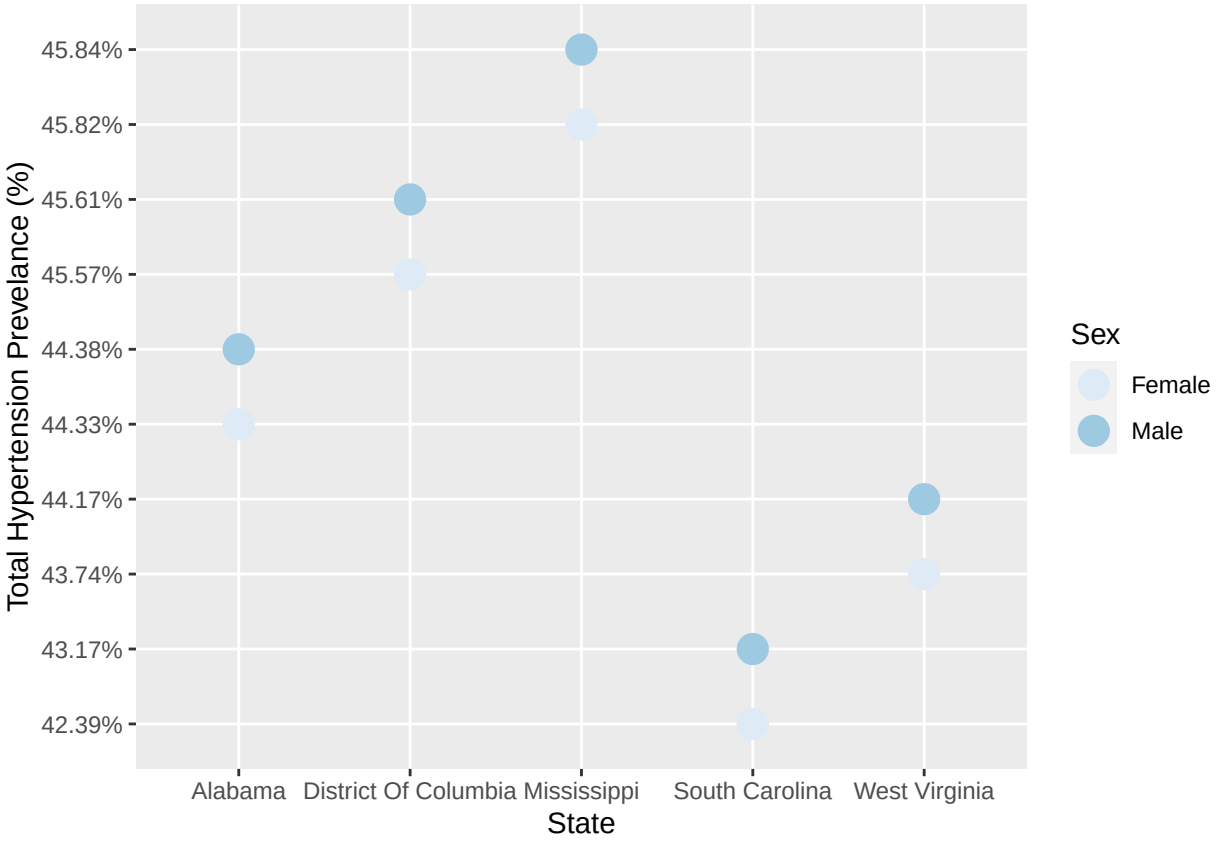
Hypertension examples with default colors

```
#Hypertension Example
ht_top5_plot <- ggplot(ht_top5, aes(x=State, y=Total)) + geom_point(aes(colour=Sex), size=5) +
  scale_x_discrete(name="State") +
  scale_y_discrete(name="Total Hypertension Prevelance (%)")
ht_top5_plot
```



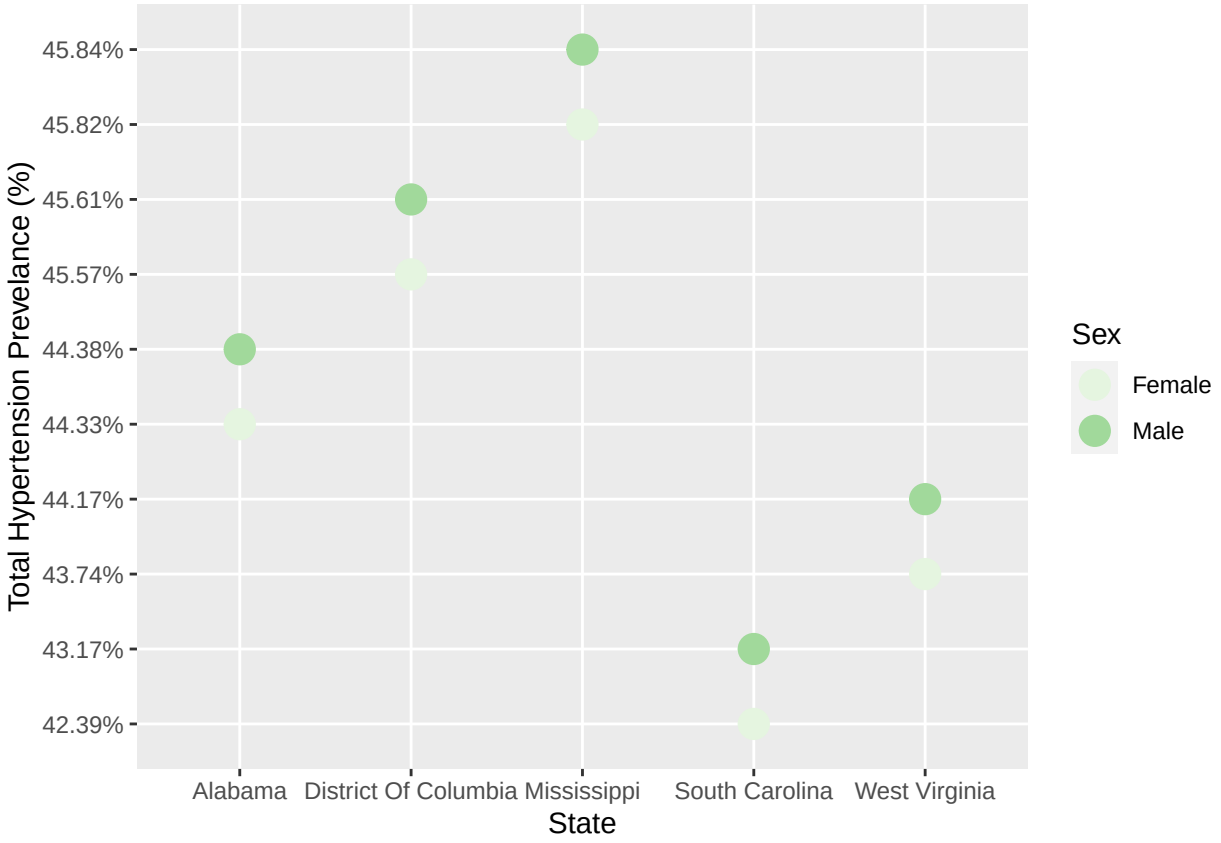
Using the hypertension example to add in a specificed color option

```
#Example using scale_color_brewer
ht_top5_plot <- ggplot(ht_top5, aes(x=State, y=Total)) + geom_point(aes(colour=Sex), size=5) +
  scale_x_discrete(name="State") +
  scale_y_discrete(name="Total Hypertension Prevelance (%)") +
  scale_color_brewer()
ht_top5_plot
```

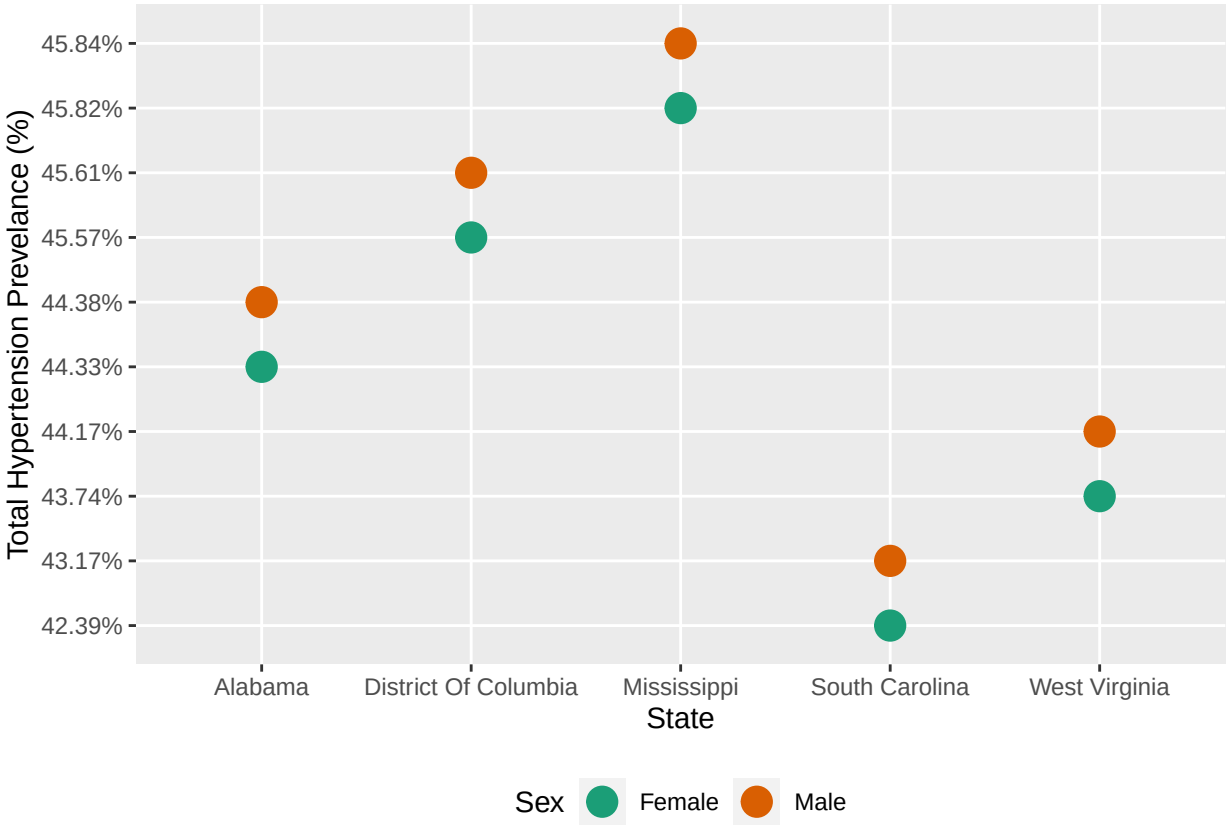


Changing the color manually

```
#Example using scale_color_brewer
ht_top5_plot_color <- ggplot(ht_top5, aes(x=State, y=Total)) + geom_point(aes(colour=Sex), size=5) +
  scale_x_discrete(name="State") +
  scale_y_discrete(name="Total Hypertension Prevelance (%)") +
  scale_color_brewer(palette = "Greens")
ht_top5_plot_color
```




```
#Example using scale_color_brewer
ht_top5_plot_color <- ggplot(ht_top5, aes(x=State, y=Total)) + geom_point(aes(colour=Sex), size=5) +
  scale_x_discrete(name="State") +
  scale_y_discrete(name="Total Hypertension Prevelance (%)") +
  scale_color_brewer(palette = "Dark2") +
  theme(legend.position="bottom") + guides(colour=guide_legend(nrow=1))
ht_top5_plot_color
```


Total Hypertension Prevelance (%)
45.84%
45.82%
45.61%
45.57%
44.38%
44.33%
44.17%
43.74%
43.17%
42.39%
Alabama
District Of Columbia
Mississippi
South Carolina
West Virginia
State
Sex
Female
Male

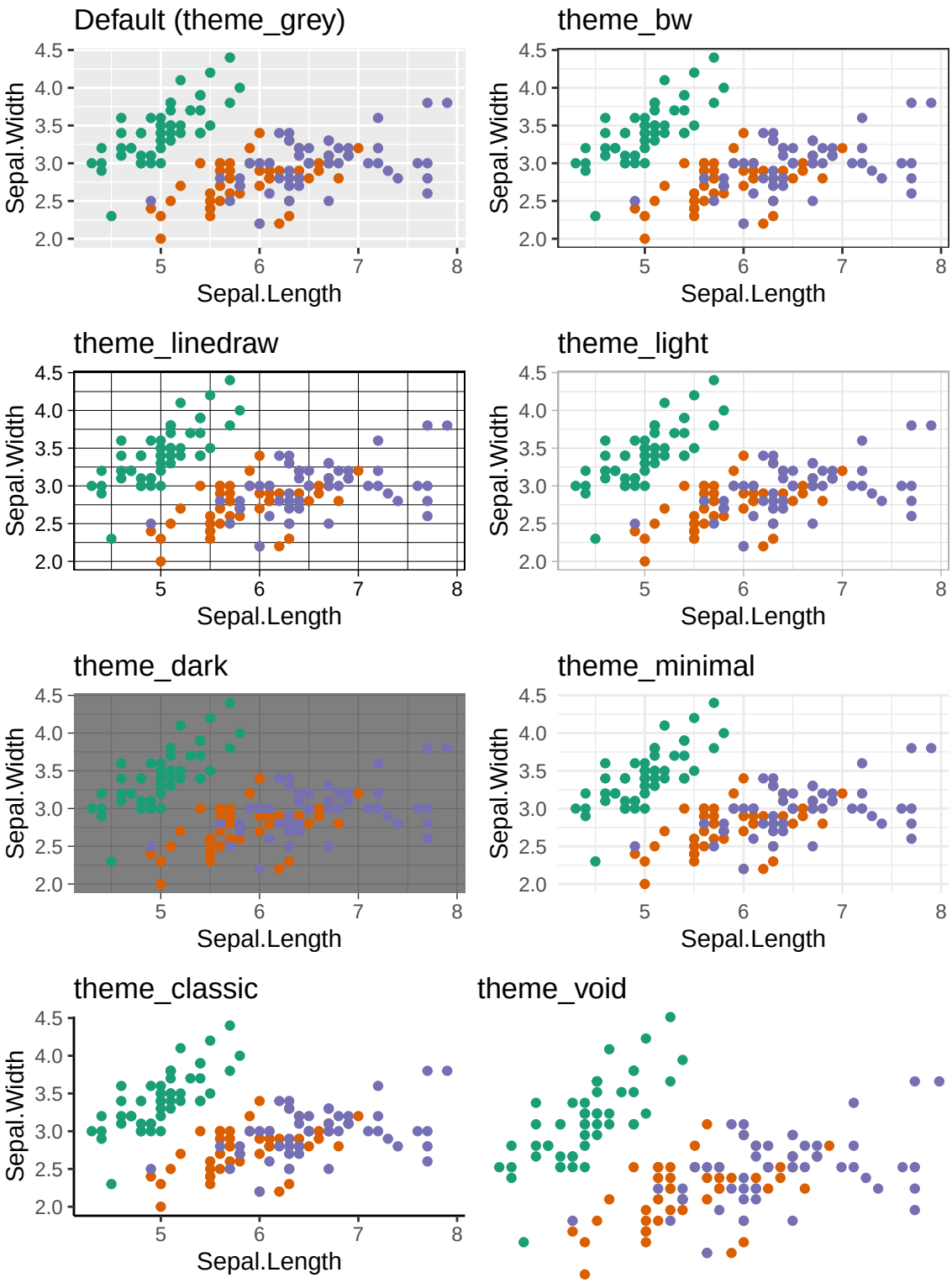
6 Themes

6.1 What is a theme?

Themes allow for the customization of the non-data parts of figures (ie. titles, labels, fonts, background, gridlines, legends).

6.2 Pre-existing themes

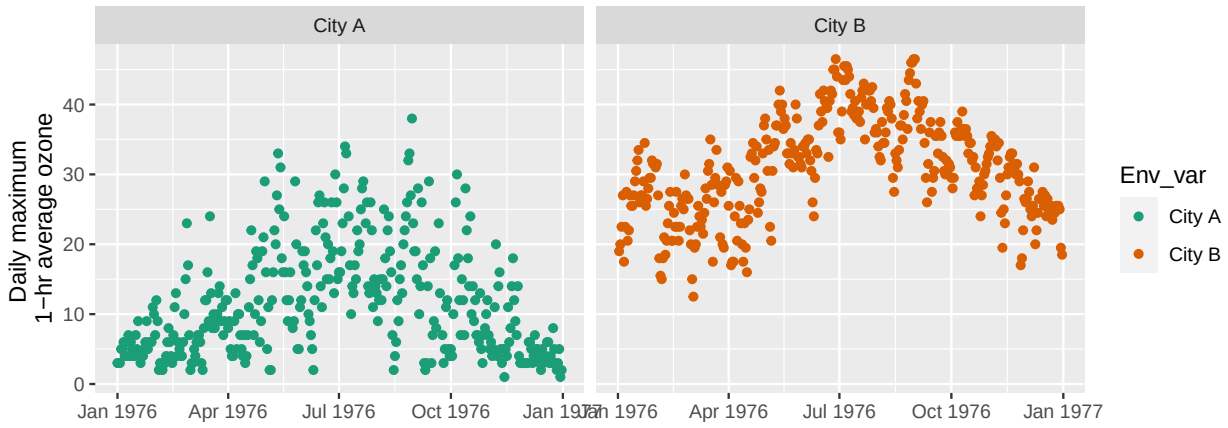
```
base_plot <- ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
  geom_point(show.legend = FALSE) +
  scale_color_brewer(palette = "Dark2")

default_plot_title <- base_plot +
  labs(title = "Default (theme_grey)")

bw_plot_title <- base_plot +
  labs(title = "theme_bw") +
  theme_bw()

linedraw_plot_title <- base_plot +
  labs(title = "theme_linedraw") +
  theme_linedraw()

light_plot_title <- base_plot +
  labs(title = "theme_light") +
  theme_light()

dark_plot_title <- base_plot +
  labs(title = "theme_dark") +
  theme_dark()

minimal_plot_title <- base_plot +
  labs(title = "theme_minimal") +
  theme_minimal()

classic_plot_title <- base_plot +
  labs(title = "theme_classic") +
  theme_classic()

void_plot_title <- base_plot +
  labs(title = "theme_void") +
  theme_void()

multiplot(default_plot_title, bw_plot_title, linedraw_plot_title,
          light_plot_title, dark_plot_title, minimal_plot_title,
          classic_plot_title, void_plot_title, layout = matrix(c(1,2,3,4,5,6,7,8),
                                                               nrow = 4, byrow = TRUE))
```


Default (theme_grey)
theme_bw
theme_linedraw
theme_light
theme_dark
theme_minimal
theme_classic
theme_void
Sepal.Width
Sepal.Length
4.5
4.0
3.5
3.0
2.5
2.0
5
6
7
8

6.3 Scatterplot example

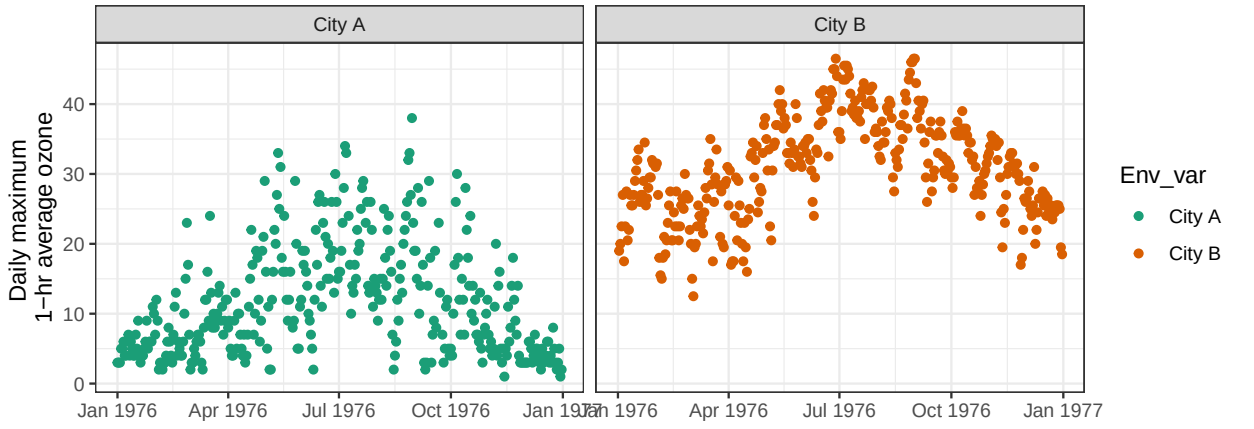
Original default plot

```
### Load altered Los Angeles Ozone Pollution Data, 1976 (package: mlbench)
# ozone data for a second city was created by subsitituting temp data

Ozone_sub_long <- readRDS(paste0(external_data_dir, "Ozone_ex\\Ozone_ex.rds"))

base_plot <- ggplot(Ozone_sub_long, aes(x = Date, y = measurement, color = Env_var)) +
  geom_point() +
  scale_color_brewer(palette = "Dark2") +
  facet_grid(. ~ Env_var) +
  labs(x = "",
       y = "Daily maximum \n1-hr average ozone")
base_plot
```



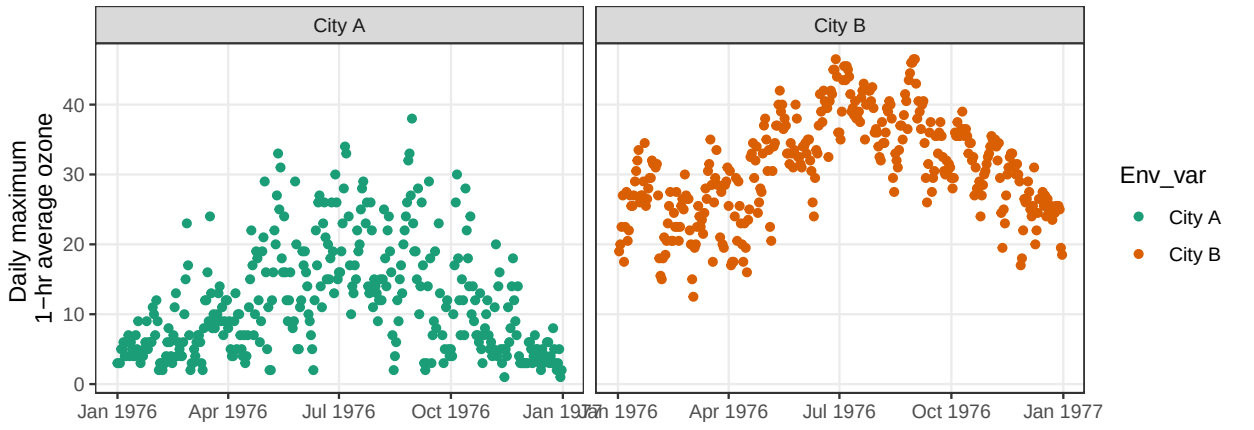
Originial plot with only theme_bw

```
bw_plot <- base_plot +
  theme_bw()
bw_plot
```



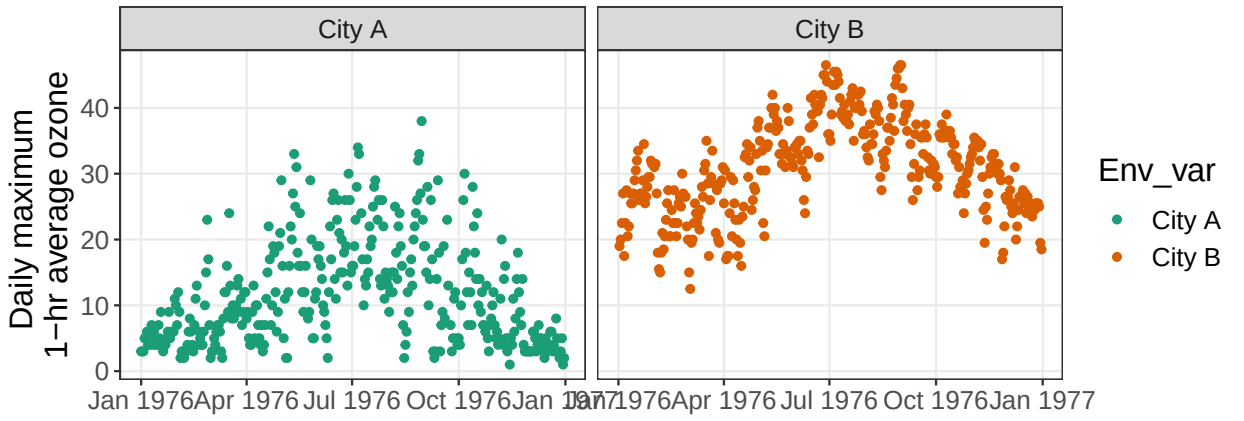
Adjustment to reduce gridlines

```
bw_plot_adj1 <- base_plot +
  theme_bw() +
  theme(
    panel.grid.minor = element_blank()
  )
bw_plot_adj1
```



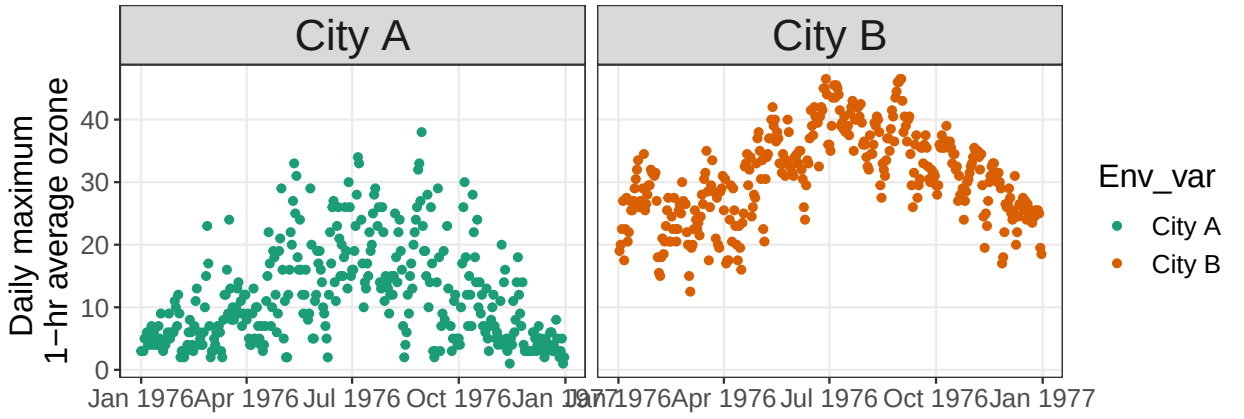
Adjustment to increase global text size

```
text_size <- 15

bw_plot_adj2 <- base_plot +
  theme_bw() +
  theme(
    panel.grid.minor = element_blank(),
    text = element_text(size = text_size)
)
bw_plot_adj2
```



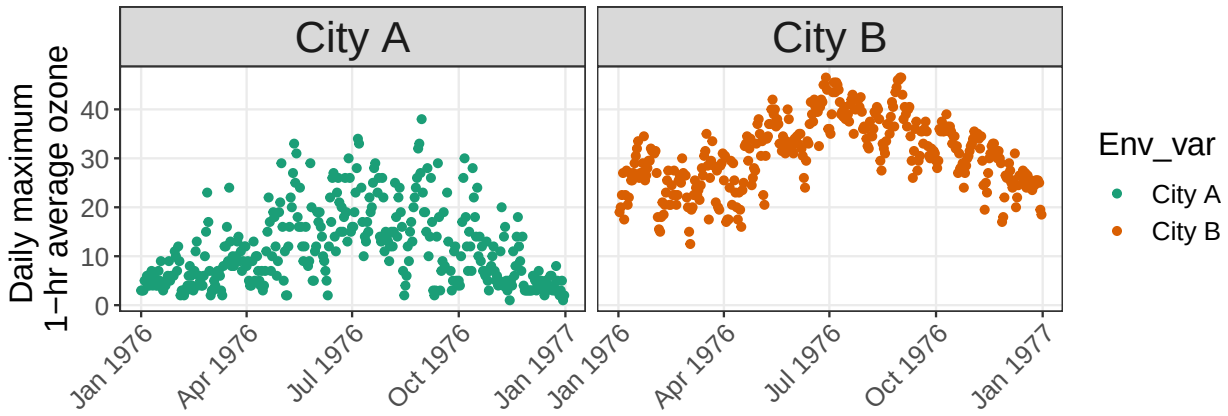
Adjustment to increase facet label text size

```
bw_plot_adj3 <- base_plot +
  theme_bw() +
  theme(
    panel.grid.minor = element_blank(),
    text = element_text(size = text_size),
    strip.text.x = element_text(size = 20)
)
bw_plot_adj3
```



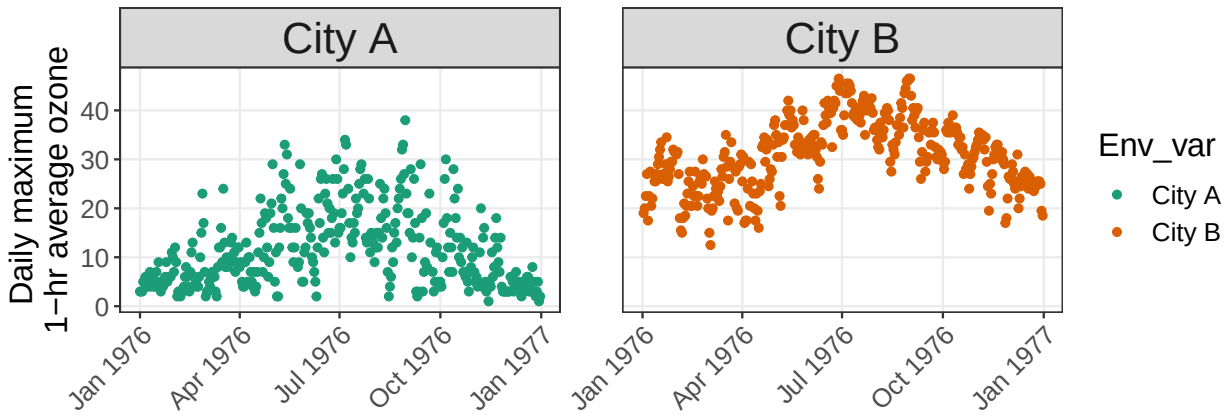
Adjustment to rotate x-axis labels

```
bw_plot_adj4 <- base_plot +
  theme_bw() +
  theme(
    panel.grid.minor = element_blank(),
    text = element_text(size = text_size),
    strip.text.x = element_text(size = 20),
    axis.text.x = element_text(angle = 45, hjust = 1)
)
bw_plot_adj4
```



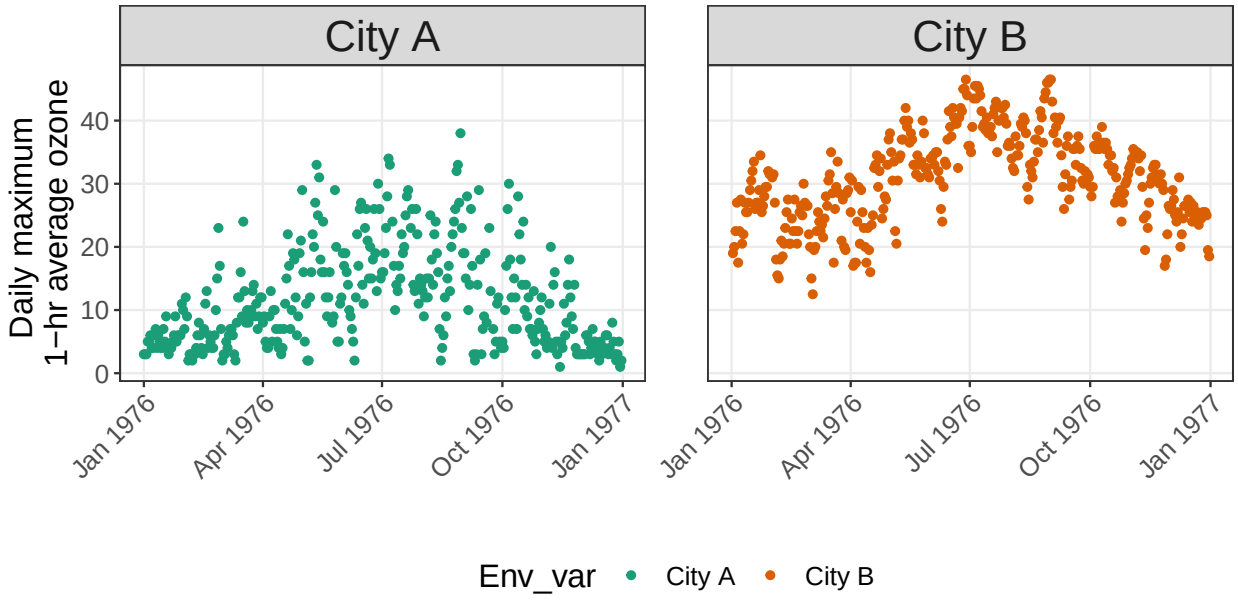
Adjustment to change spacing between facet panels

```
bw_plot_adj5 <- base_plot +
  theme_bw() +
  theme(
    panel.grid.minor = element_blank(),
    text = element_text(size = text_size),
    strip.text.x = element_text(size = 20),
    axis.text.x = element_text(angle = 45, hjust = 1),
    panel.spacing = unit(2, "lines")
)
bw_plot_adj5
```



Adjustment to change legend (position, text size)

```
bw_plot_adj6 <- base_plot +
  theme_bw() +
  theme(
    panel.grid.minor = element_blank(),
    text = element_text(size = text_size),
    strip.text.x = element_text(size = 20),
    axis.text.x = element_text(angle = 45, hjust = 1),
    panel.spacing = unit(2, "lines"),
    legend.position = "bottom"
)
bw_plot_adj6
```



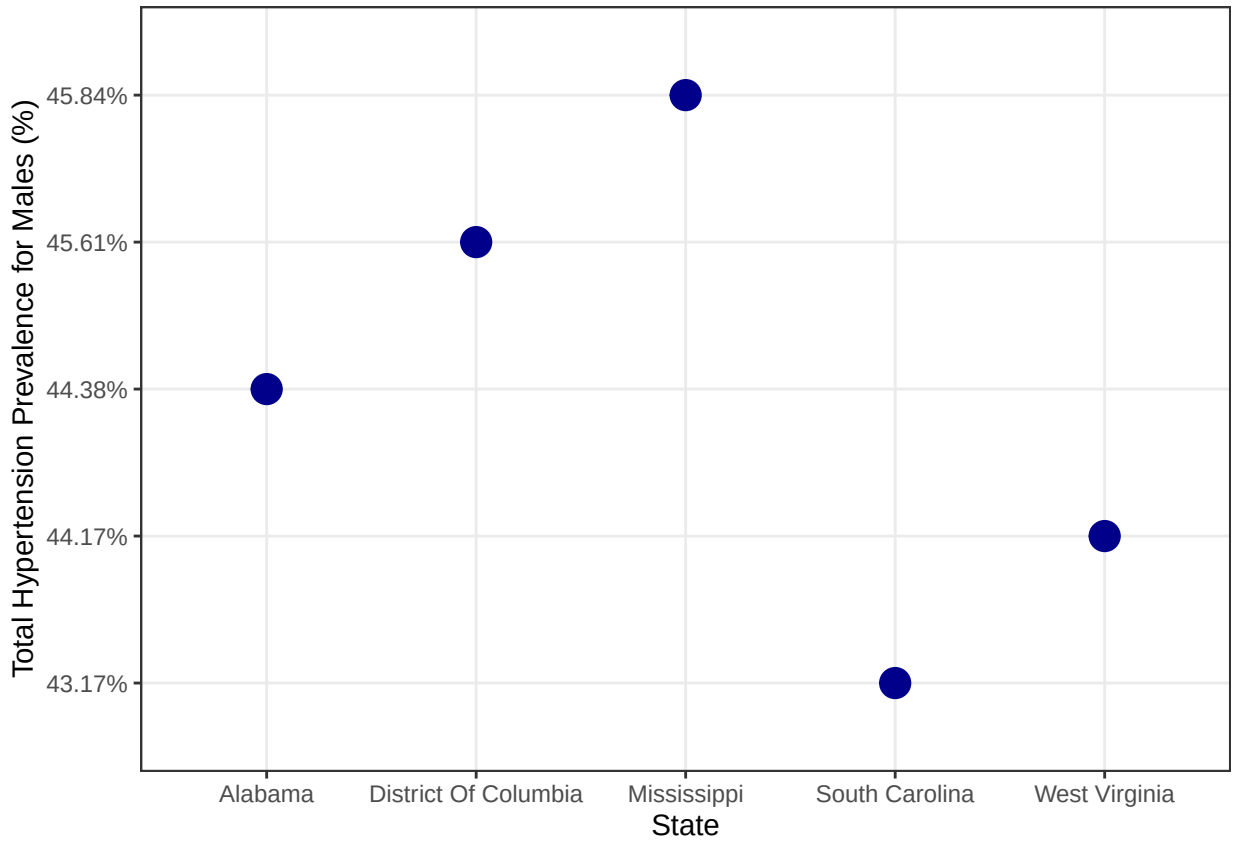
7 Facets

Facets generate small groupings with a different subset of the data.

##Facet Wrap Using the hypertension exmaple, two individual plots by sex are created.

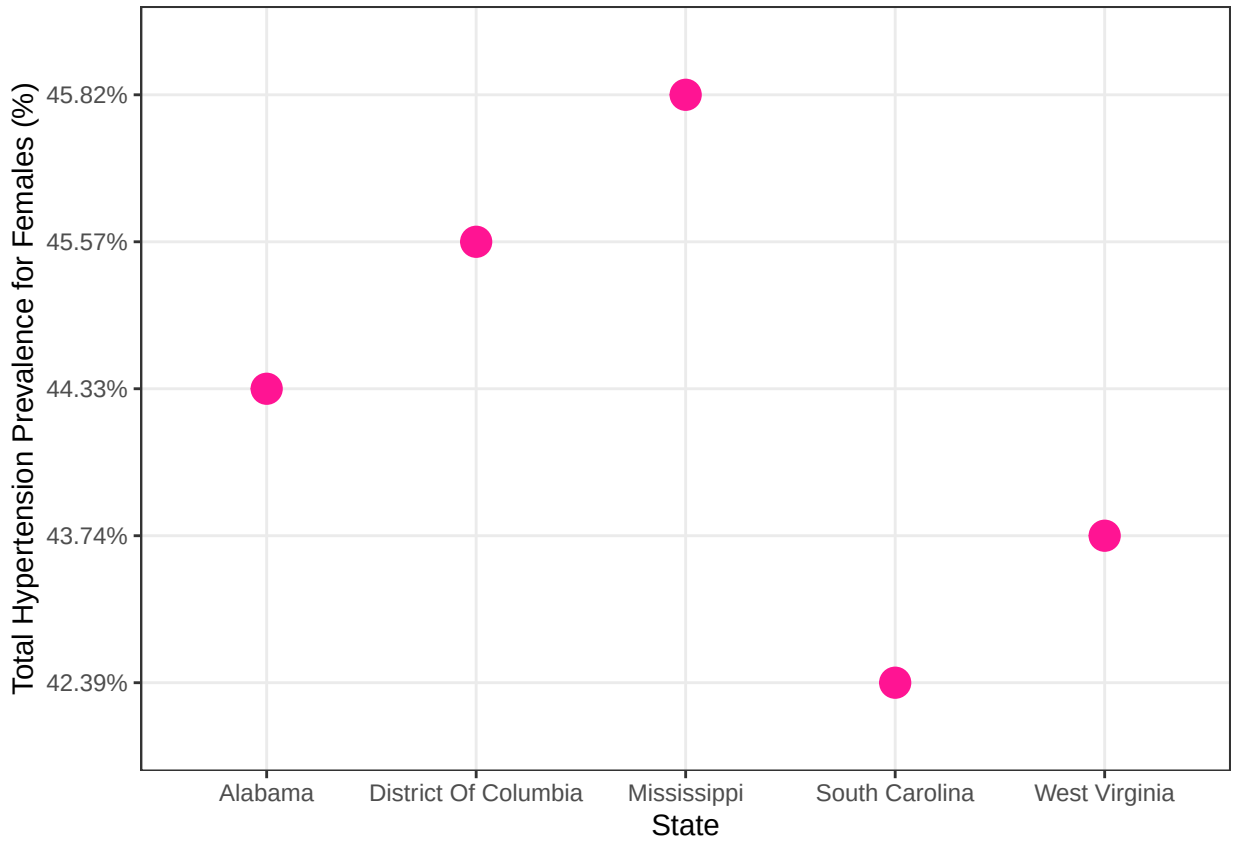
Males:

```
#Hypertension Example of Fracet Wrapping
#Hypertension Prevalence by Sex
#Males
ht_male <- ggplot(ht_top5_wide, aes(x=State, y=Total_Male)) + geom_point(size=5, color="darkblue") +
  scale_x_discrete(name="State") +
  scale_y_discrete(name="Total Hypertension Prevalence for Males (%)") +
  theme_bw()
ht_male
```



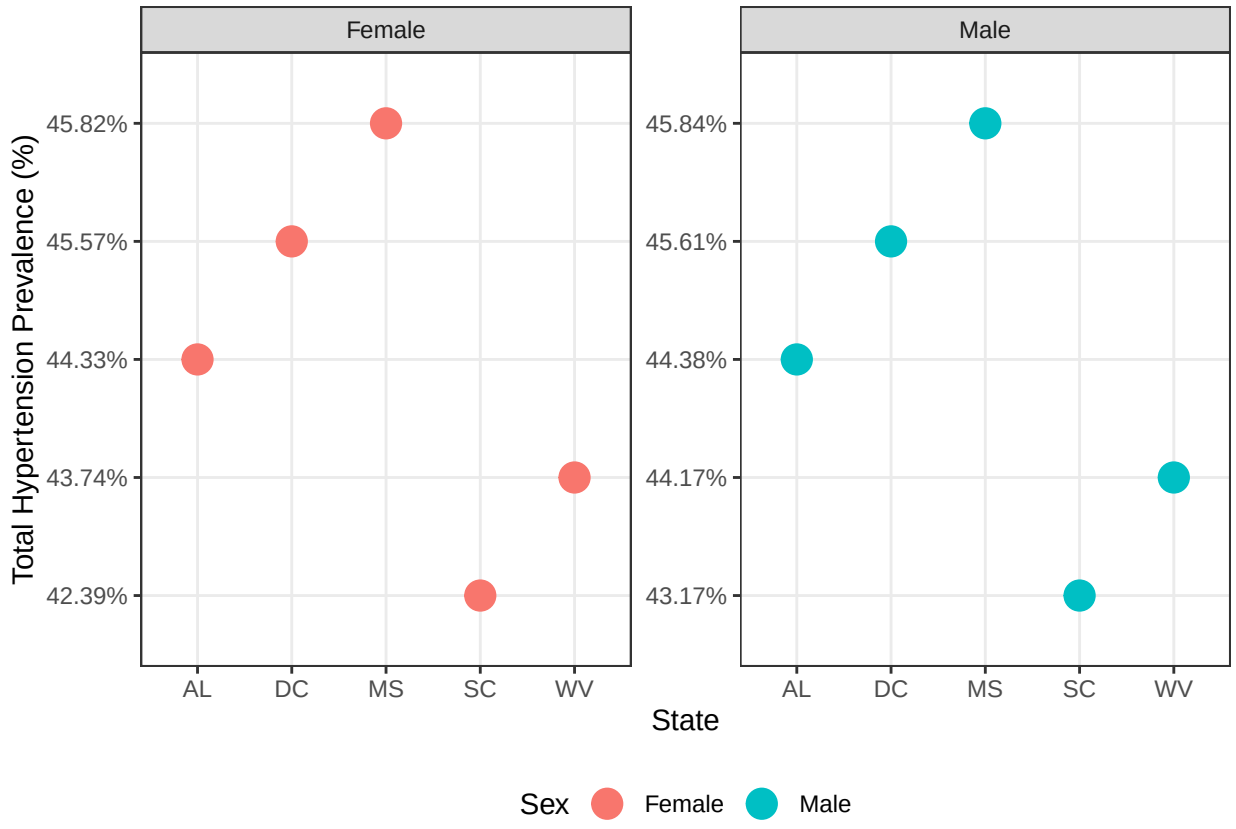
Females:

```
#Females
ht_female <- ggplot(ht_top5_wide, aes(x=State, y=Total_Female)) + geom_point(size=5, color="deeppink") +
  scale_x_discrete(name="State") +
  scale_y_discrete(name="Total Hypertension Prevalence for Females (%)") +
  theme_bw()
ht_female
```

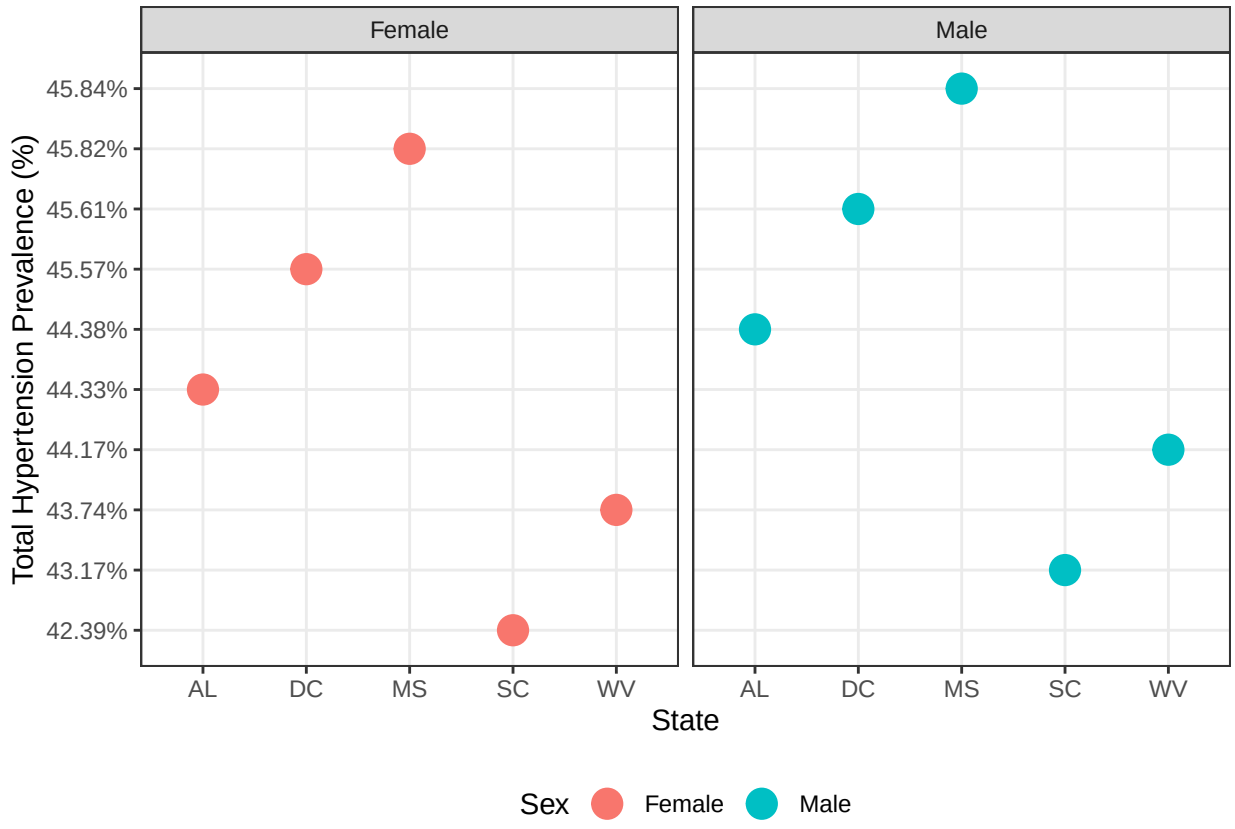


Using facet wrap to combine the two together.

```
#Using Facet Wrap
ht_top5_plot <- ggplot(ht_top5, aes(x=State, y=Total, group=Sex, colour=Sex)) + geom_point(size=5) +
  scale_x_discrete(name="State", labels=c("Alabama" = "AL", "District Of Columbia" = "DC", "Mississippi
                           "South Carolina" = "SC", "West Virginia" = "WV")) +
  scale_y_discrete(name="Total Hypertension Prevalence (%)") +
  theme_bw() +
  theme(legend.position='bottom') + guides(colour = guide_legend(nrow = 1)) +
  facet_wrap(~Sex, scales = "free")
ht_top5_plot
```



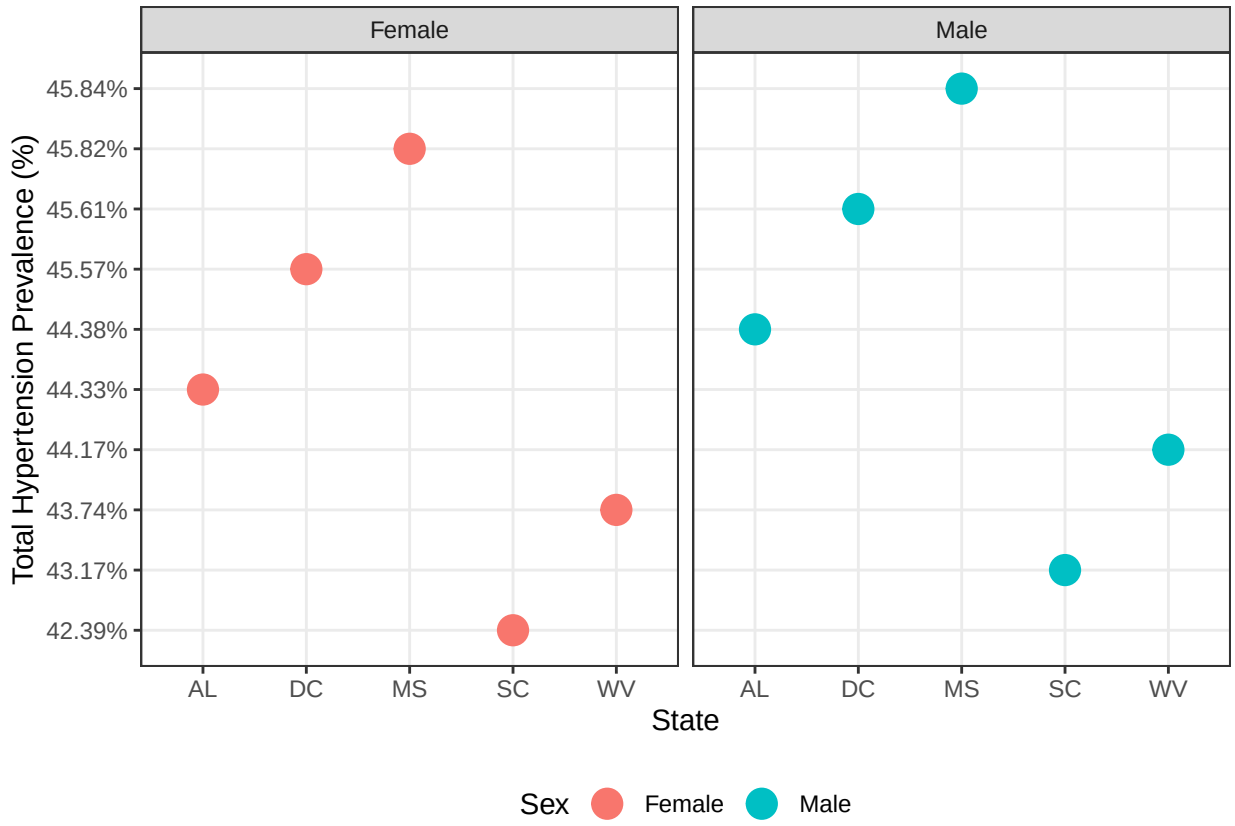

```
#This produces the same graph as above, but slightly different coding for facet_wrap
ht_top5_plot <- ggplot(ht_top5, aes(x=State, y=Total, group=Sex, colour=Sex)) + geom_point(size=5) +
  scale_x_discrete(name="State", labels=c("Alabama" = "AL", "District Of Columbia" = "DC", "Mississippi
                           "South Carolina" = "SC", "West Virginia" = "WV")) +
  scale_y_discrete(name="Total Hypertension Prevalence (%)") +
  theme_bw() +
  theme(legend.position='bottom') + guides(colour = guide_legend(nrow = 1)) +
  facet_wrap(~Sex, nrow = 1, dir="h")
ht_top5_plot
```



7.1 Facet Grid

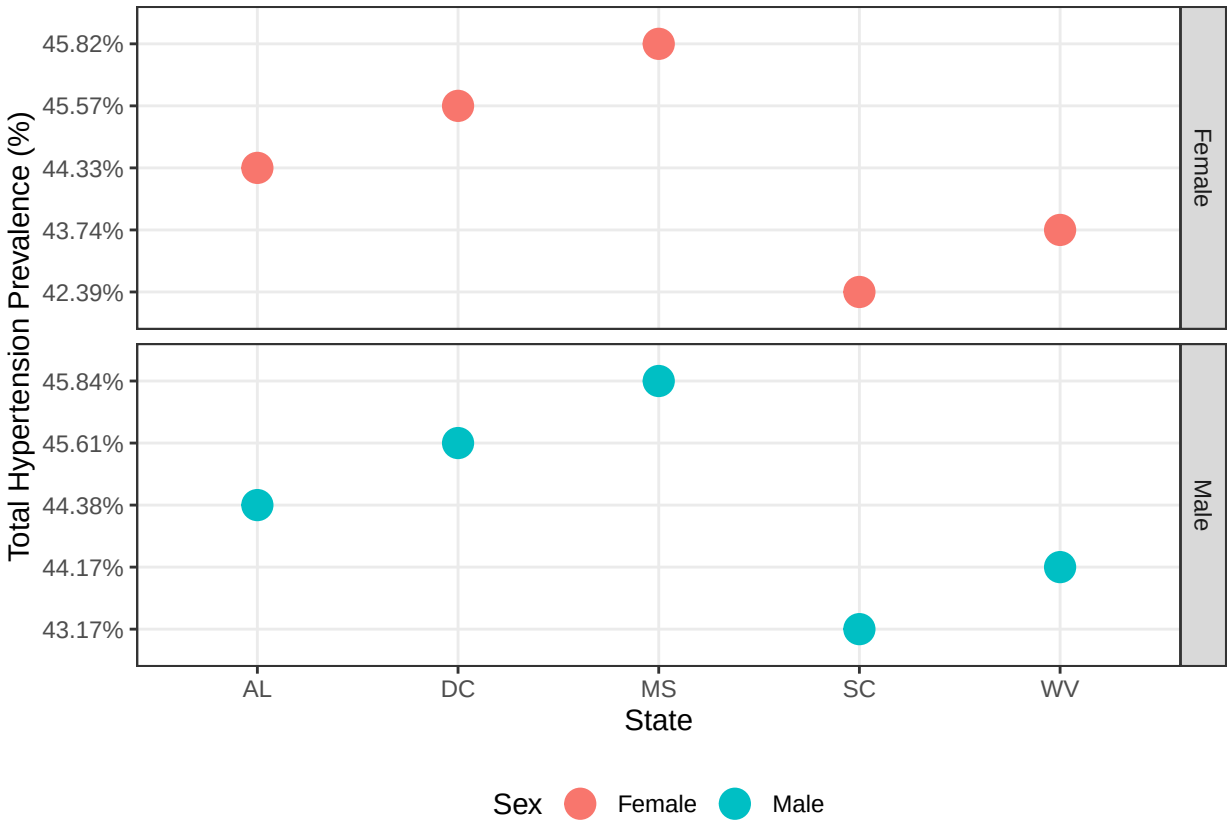
Creating a vertical grid

```
#Using Facet Grid
#Vertical
ht_top5_plot_vertical <- ggplot(ht_top5, aes(x=State, y=Total, group=Sex, colour=Sex)) + geom_point(size
  scale_x_discrete(name="State", labels=c("Alabama" = "AL", "District Of Columbia" = "DC", "Mississippi
                           "South Carolina" = "SC", "West Virginia" = "WV")) +
  scale_y_discrete(name="Total Hypertension Prevalence (%)") +
  theme_bw() +
  theme(legend.position='bottom') + guides(colour = guide_legend(nrow = 1)) +
  facet_grid(~Sex, scales = "free")
ht_top5_plot_vertical
```



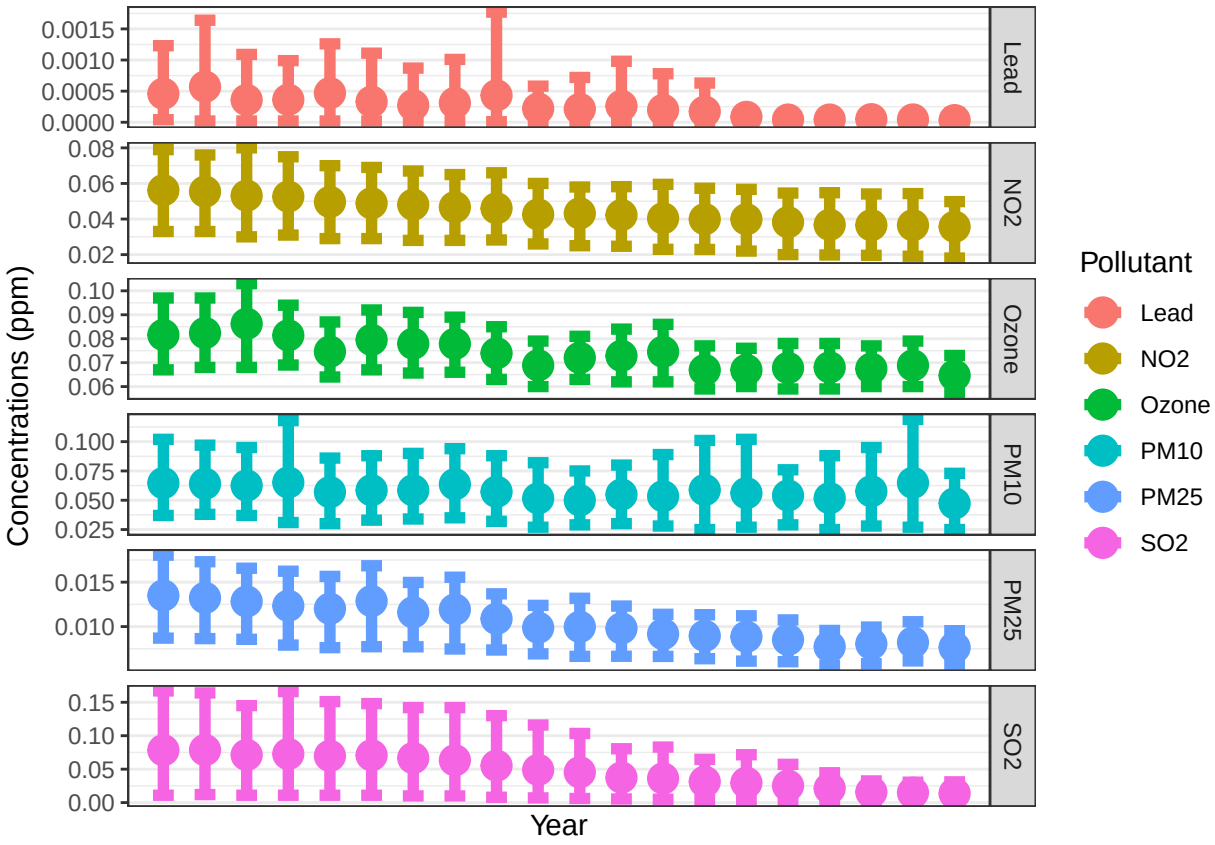
Creating a horizontal grid

```
#Horizontal
ht_top5_plot_horizontal <- ggplot(ht_top5, aes(x=State, y=Total, group=Sex, colour=Sex)) + geom_point(si
  scale_x_discrete(name="State", labels=c("Alabama" = "AL", "District Of Columbia" = "DC", "Mississippi
                           "South Carolina" = "SC", "West Virginia" = "WV")) +
  scale_y_discrete(name="Total Hypertension Prevalence (%)") +
  theme_bw() +
  theme(legend.position='bottom') + guides(colour = guide_legend(nrow = 1)) +
  facet_grid(Sex~., scales = "free")
ht_top5_plot_horizontal
```



Creating a facet with multiple air pollutants

```
#Creating multiple plots for the criteria air pollutants
#Only faceting the plots with the same scale (units)
cap_plot <- ggplot(air_long, aes(x=Year, y=Concentrations, group=Pollutant, color=Pollutant)) +
  geom_point(size=5) +
  geom_errorbar(aes(ymin=Lower_10th_Percentile, ymax=Upper_90th_Percentile), width=0.5, size=2) +
  scale_x_discrete(name="Year") +
  scale_y_continuous(name="Concentrations (ppm)") +
  theme_bw() +
  facet_grid(Pollutant~., scales = "free")
cap_plot
```

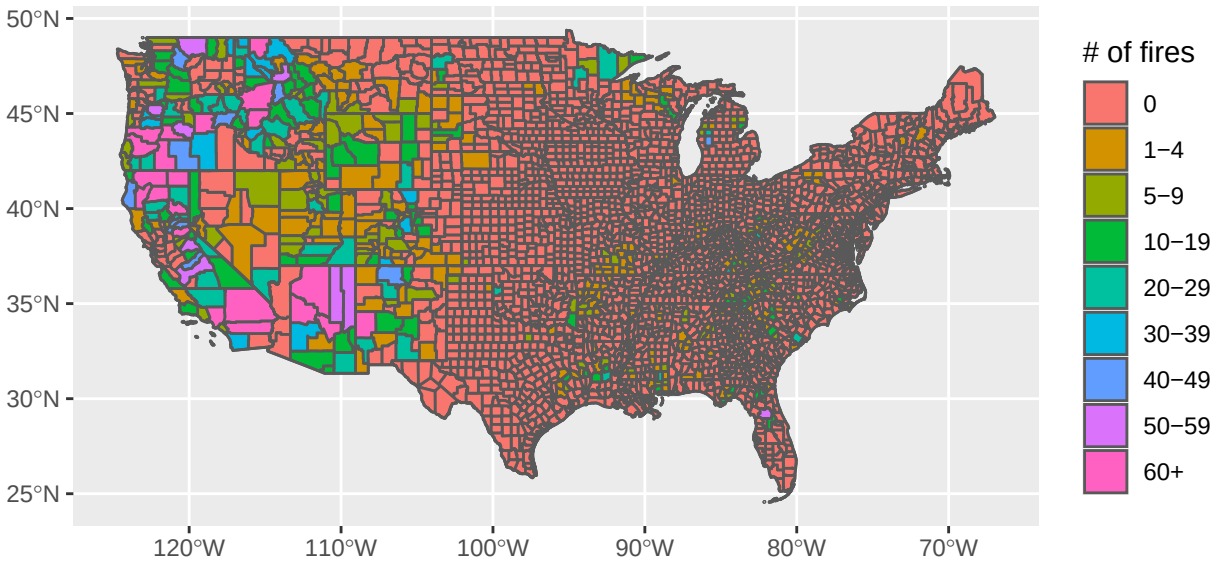

Concentrations (ppm)
0.0015
0.0010
0.0005
0.0000
Lead
0.08
0.06
0.04
0.02
NO2
0.10
0.09
0.08
0.07
0.06
Ozone
0.100
0.075
0.050
0.025
PM10
0.015
0.010
PM25
0.15
0.10
0.05
0.00
SO2
Year
Pollutant
Lead
NO2
Ozone
PM10
PM25
SO2

8 Mapping

8.1 Map example 1 - Fires

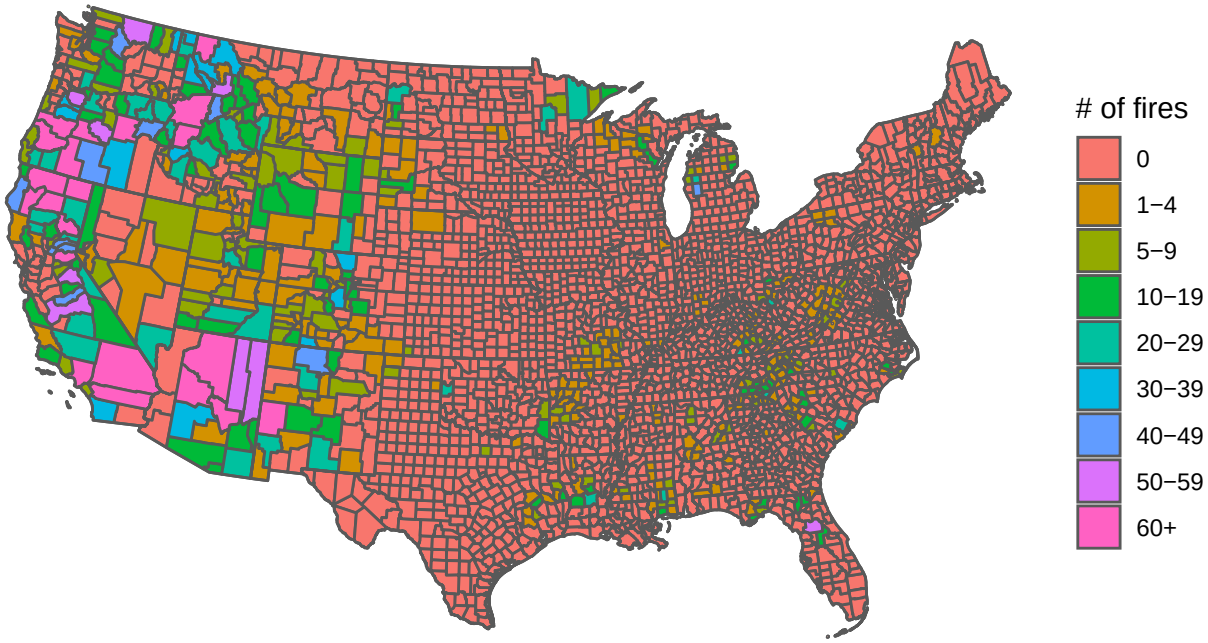
Default map for 2008

```
map_1 <- ggplot(subset(UScounties, FIRE_YEAR == 2008), aes(fill = fire_sum_total_cat)) +
  geom_sf() +
  geom_sf(data = USstates, fill = NA) +
  labs(fill = "# of fires")
map_1
```



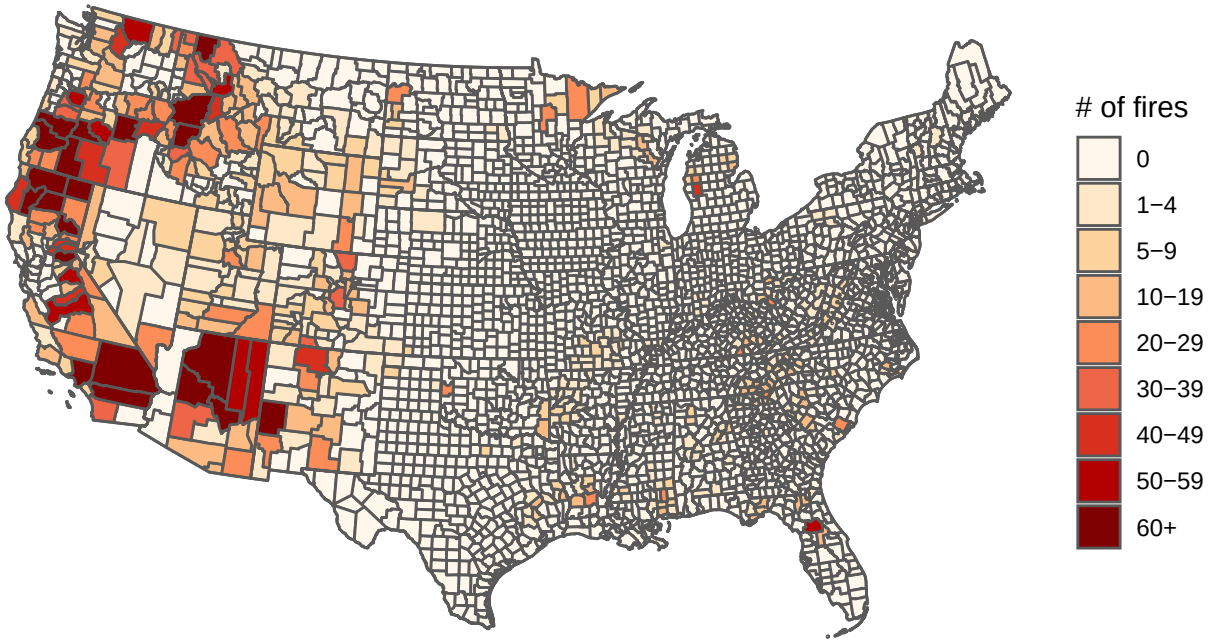
Adjust focus to map, minimizing background, adding conic projection

```
map_2 <- ggplot(subset(UScounties, FIRE_YEAR == 2008), aes(fill = fire_sum_total_cat)) +
  geom_sf() +
  geom_sf(data = USstates, fill = NA) +
  coord_sf(crs = st_crs(5070)) +
  theme_void() +
  labs(fill = "# of fires")
map_2
```



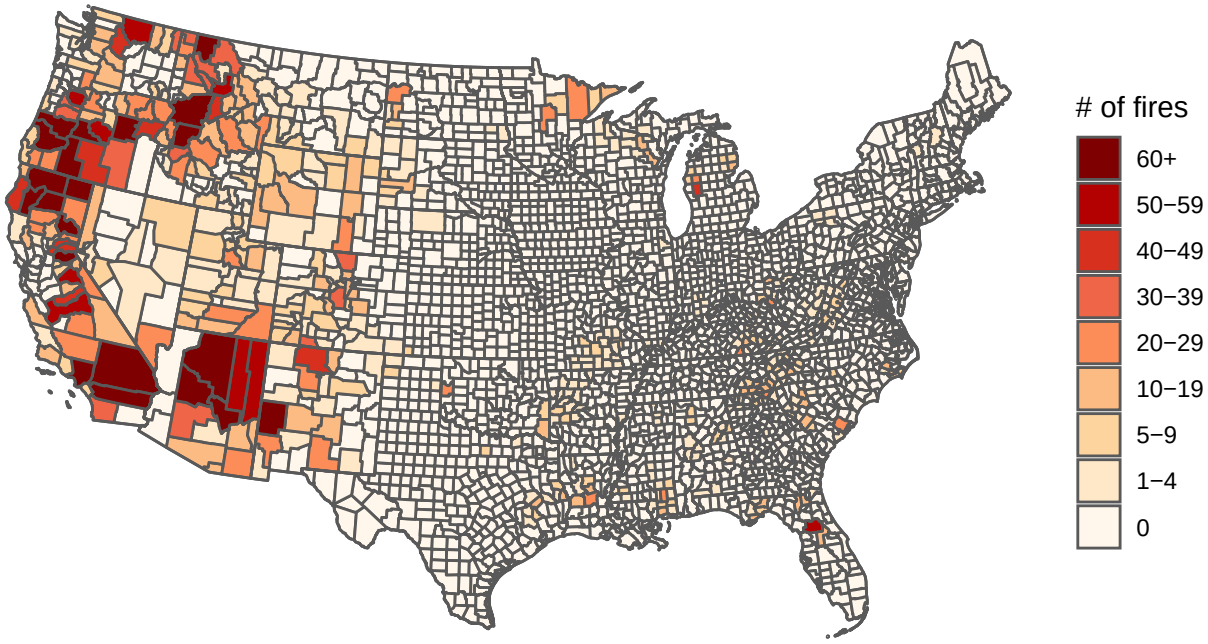
Specify color palette

```
map_3 <- ggplot(subset(UScounties, FIRE_YEAR == 2008), aes(fill = fire_sum_total_cat)) +
  geom_sf() +
  geom_sf(data = USstates, fill = NA) +
  scale_fill_brewer(palette = "OrRd") +
  coord_sf(crs = st_crs(5070)) +
  theme_void() +
  labs(fill = "# of fires")
map_3
```



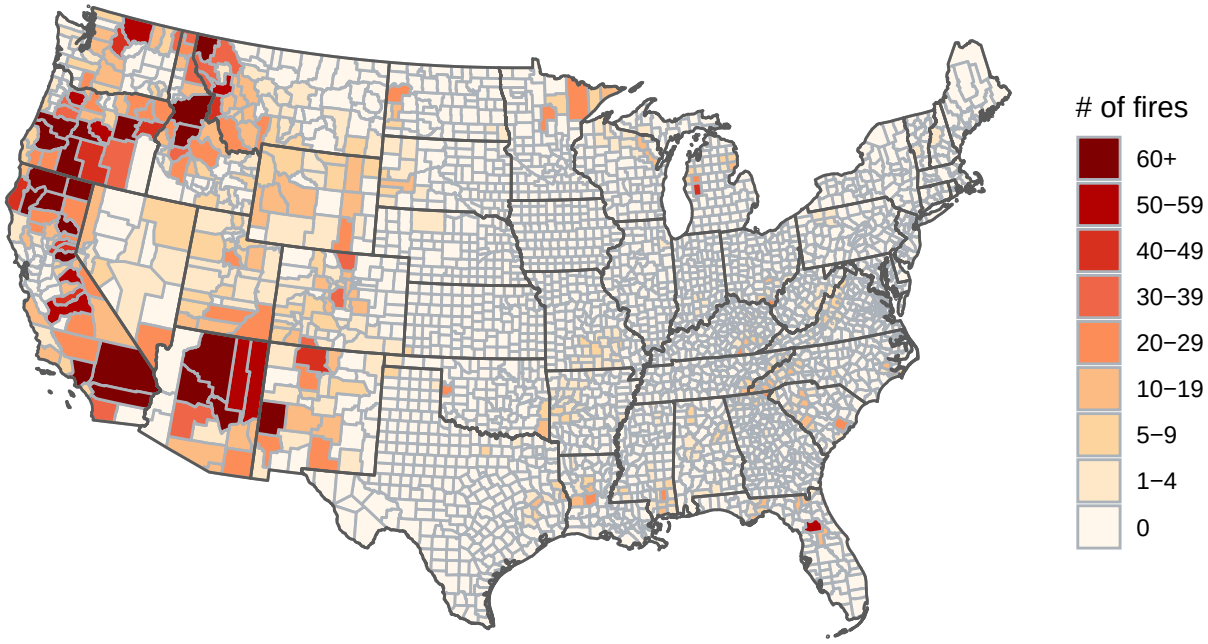
Adjust legend so higher values are on top

```
map_4 <- ggplot(subset(UScounties, FIRE_YEAR == 2008), aes(fill = fire_sum_total_cat)) +
  geom_sf() +
  geom_sf(data = USstates, fill = NA) +
  scale_fill_brewer(palette = "OrRd", guide = guide_legend(reverse = TRUE)) +
  coord_sf(crs = st_crs(5070)) +
  theme_void() +
  labs(fill = "# of fires")
map_4
```



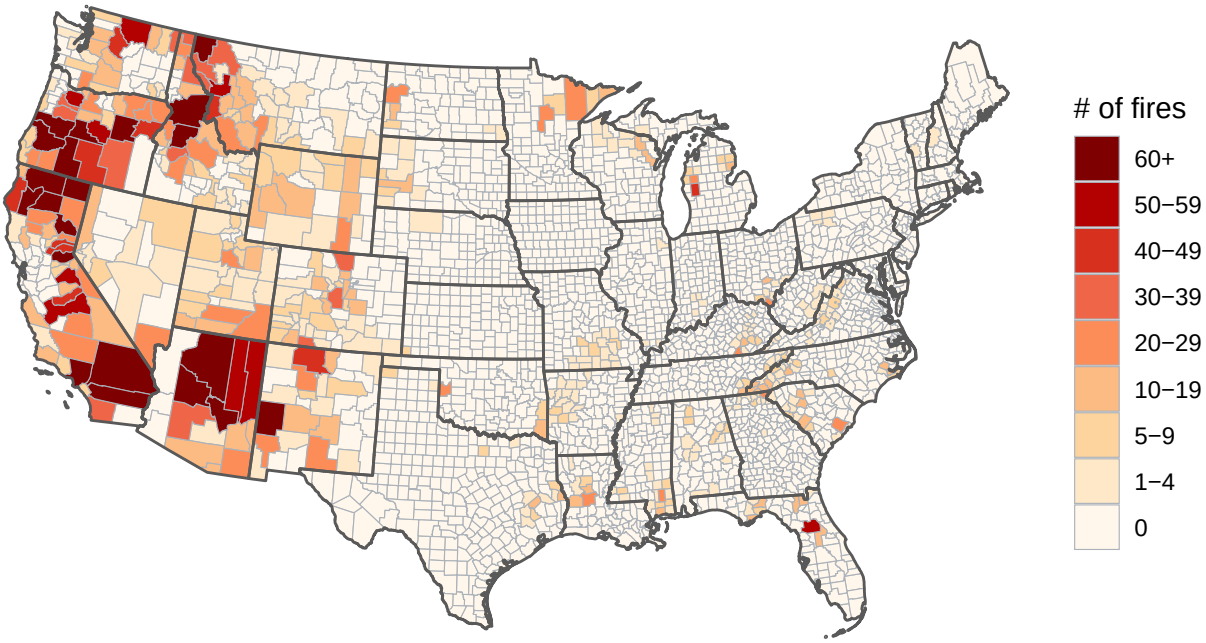
Change county line color so state shapes are apparent

```
map_5 <- ggplot(subset(UScounties, FIRE_YEAR == 2008), aes(fill = fire_sum_total_cat)) +
  geom_sf(color = "#abb2b9") +
  geom_sf(data = USstates, fill = NA) +
  scale_fill_brewer(palette = "OrRd", guide = guide_legend(reverse = TRUE)) +
  coord_sf(crs = st_crs(5070)) +
  theme_void() +
  labs(fill = "# of fires")
map_5
```



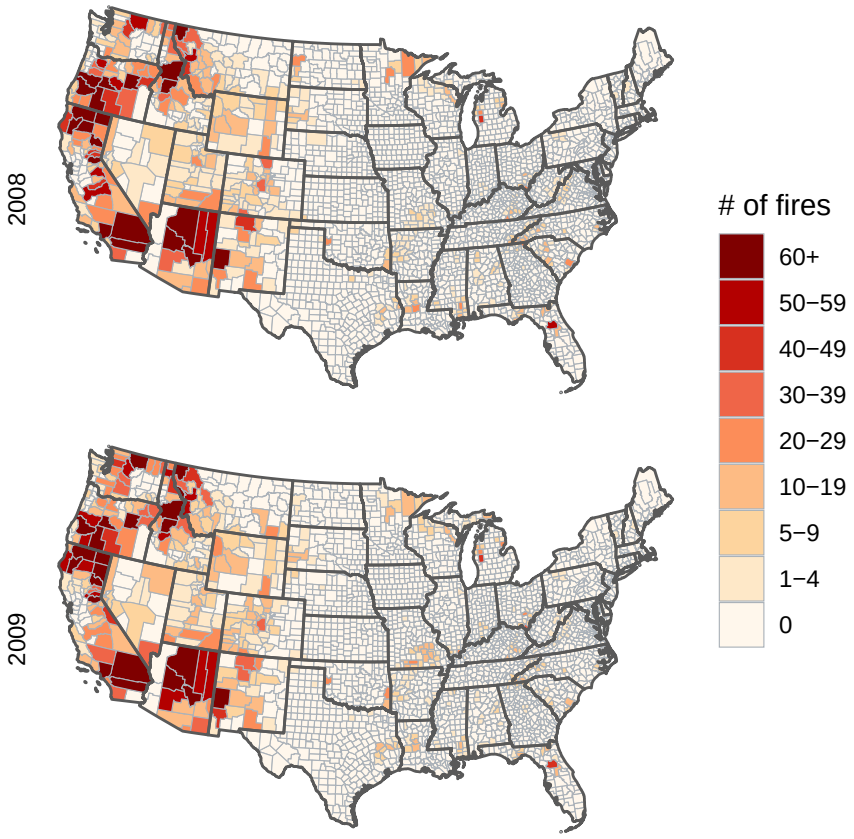
Reduce size of county lines

```
cty_line <- 0.2

map_6 <- ggplot(subset(UScounties, FIRE_YEAR == 2008), aes(fill = fire_sum_total_cat)) +
  geom_sf(color = "#abb2b9", size = cty_line) +
  geom_sf(data = USstates, fill = NA) +
  scale_fill_brewer(palette = "OrRd", guide = guide_legend(reverse = TRUE)) +
  coord_sf(crs = st_crs(5070)) +
  theme_void() +
  labs(fill = "# of fires")
map_6
```



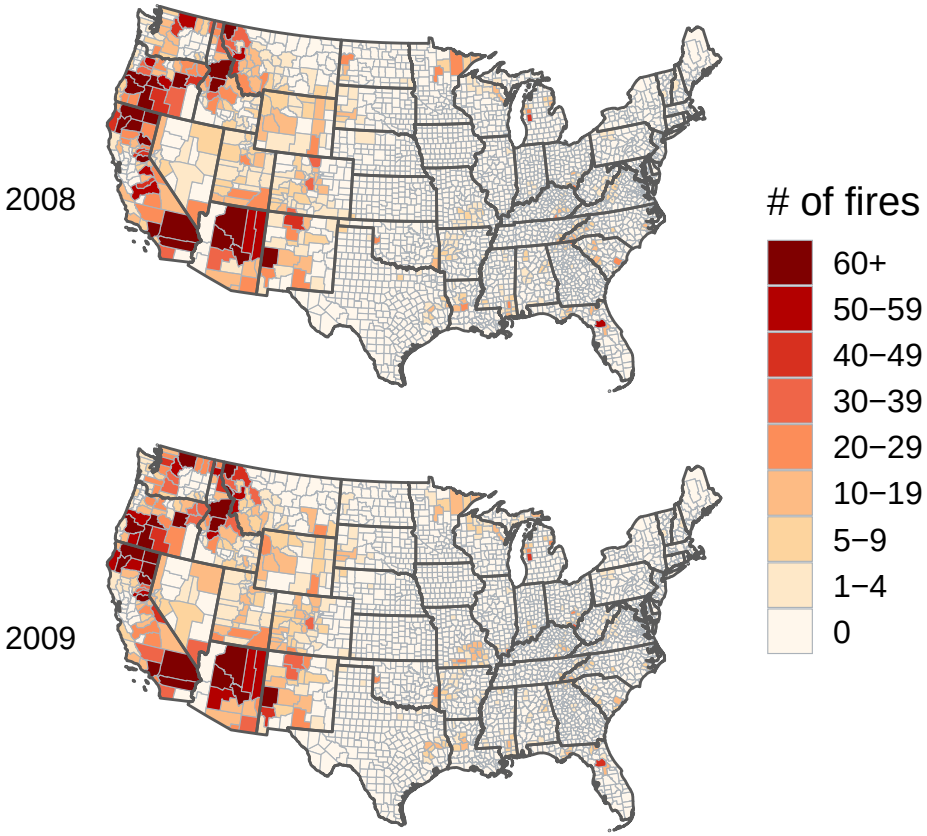
Facet by year, legend applies to all maps

```
map_7 <- ggplot(UScounties, aes(fill = fire_sum_total_cat)) +
  geom_sf(color = "#abb2b9", size = cty_line) +
  geom_sf(data = USstates, fill = NA) +
  scale_fill_brewer(palette = "OrRd", guide = guide_legend(reverse = TRUE)) +
  coord_sf(crs = st_crs(5070)) +
  facet_grid(FIRE_YEAR ~ ., switch = "y") +
  theme_void() +
  labs(fill = "# of fires")
map_7
```



Adjustments to facet label orientation and text size

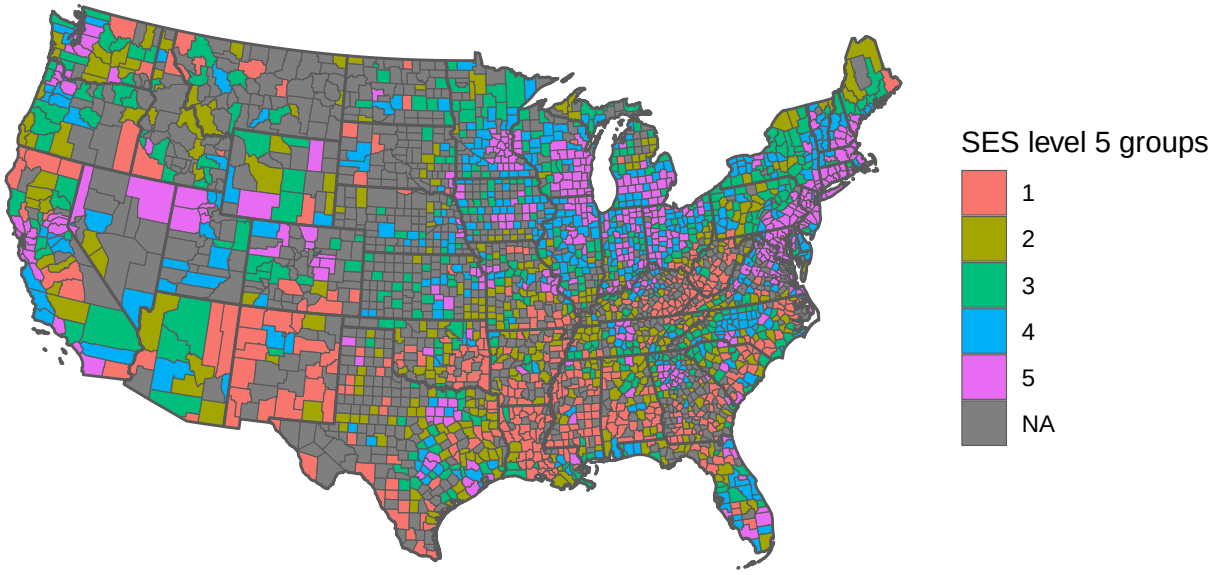
```
text_size <- 15

map_8 <- ggplot(UScounties, aes(fill = fire_sum_total_cat)) +
  geom_sf(color = "#abb2b9", size = cty_line) +
  geom_sf(data = USstates, fill = NA) +
  scale_fill_brewer(palette = "OrRd", guide = guide_legend(reverse = TRUE)) +
  coord_sf(crs = st_crs(5070)) +
  facet_grid(FIRE_YEAR ~ ., switch = "y") +
  theme_void() +
  theme(
    text = element_text(size = text_size),
    strip.text.y.left = element_text(angle = 0)
        ) +
  labs(fill = "# of fires")
map_8
```



8.2 Map example 2 - Socioeconomic status (SES)

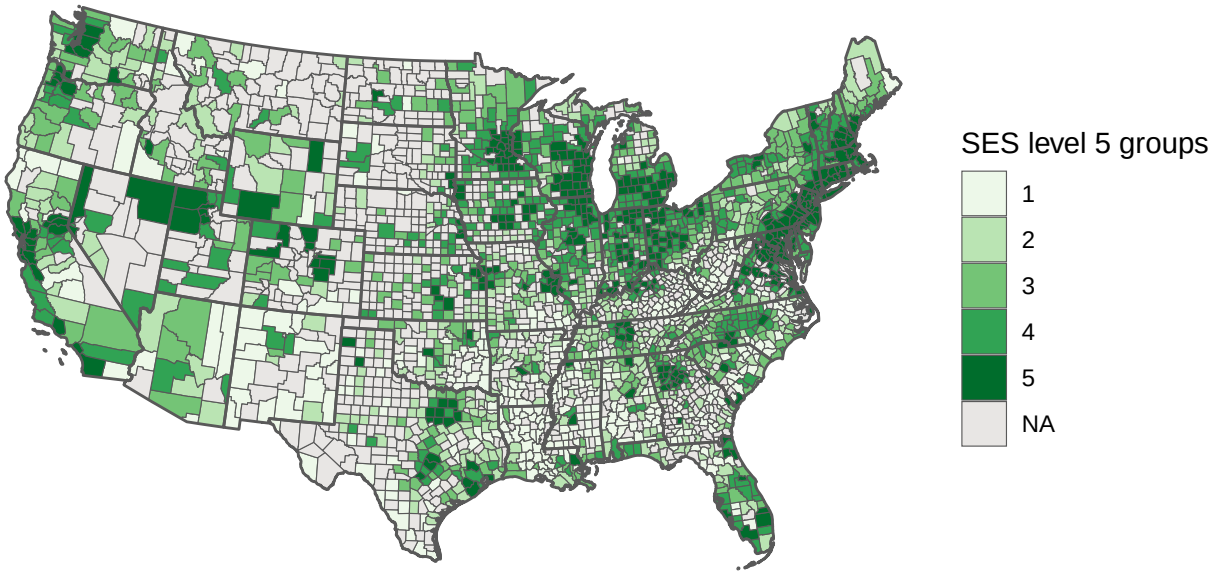
Default map for 5 SES groups

```
cty_line <- 0.2

map_1 <- ggplot(UScounties, aes(fill = Pov_factor_5)) +
  geom_sf(size = cty_line) +
  geom_sf(data = USstates, fill = NA) +
  coord_sf(crs = st_crs(5070)) +
  theme_void() +
  labs(fill = "SES level 5 groups")
map_1
```



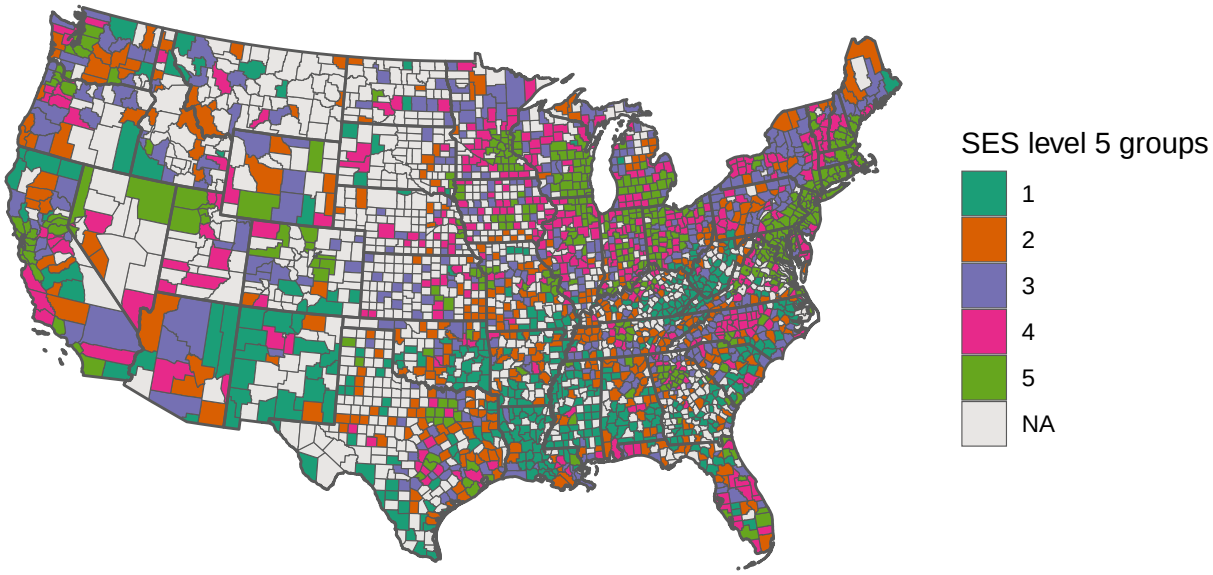
Color palette - one continuous color palate

```
map_2 <- ggplot(UScounties, aes(fill = Pov_factor_5)) +
  geom_sf(size = cty_line) +
  geom_sf(data = USstates, size = 0.5, fill = NA) +
  scale_fill_brewer(palette = "Greens", na.value = "#e8e6e4") +
  coord_sf(crs = st_crs(5070)) +
  theme_void() +
  labs(fill = "SES level 5 groups")
map_2
```



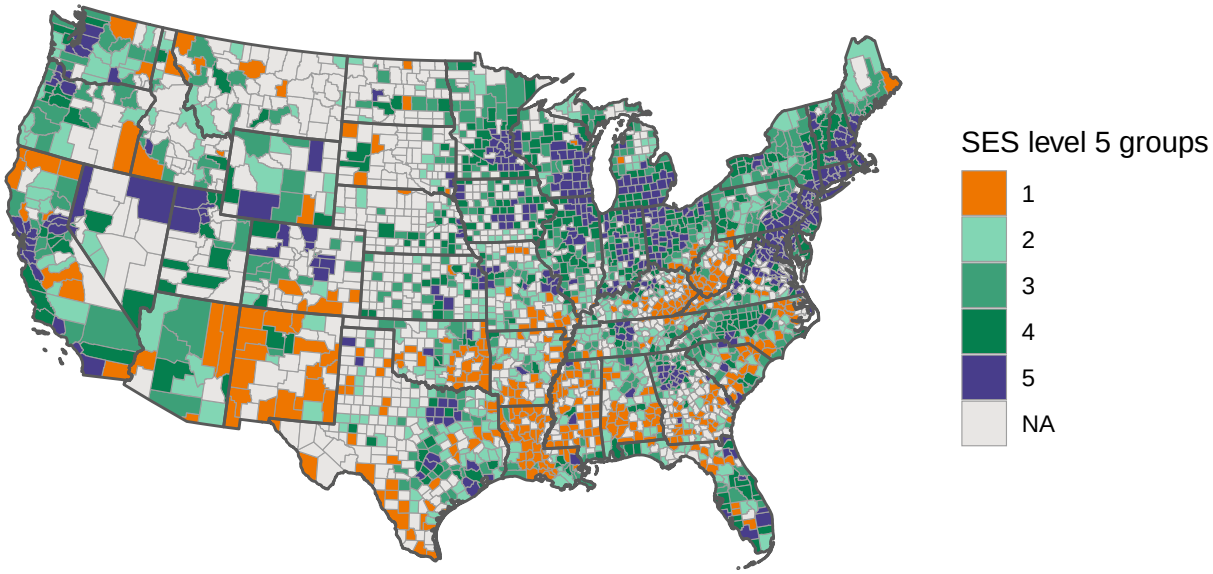
Color palette - discrete categories

```
map_3 <- ggplot(UScounties, aes(fill = Pov_factor_5)) +
  geom_sf(size = cty_line) +
  geom_sf(data = USstates, size = 0.5, fill = NA) +
  scale_fill_brewer(palette = "Dark2", na.value = "#e8e6e4") +
  coord_sf(crs = st_crs(5070)) +
  theme_void() +
  labs(fill = "SES level 5 groups")
map_3
```



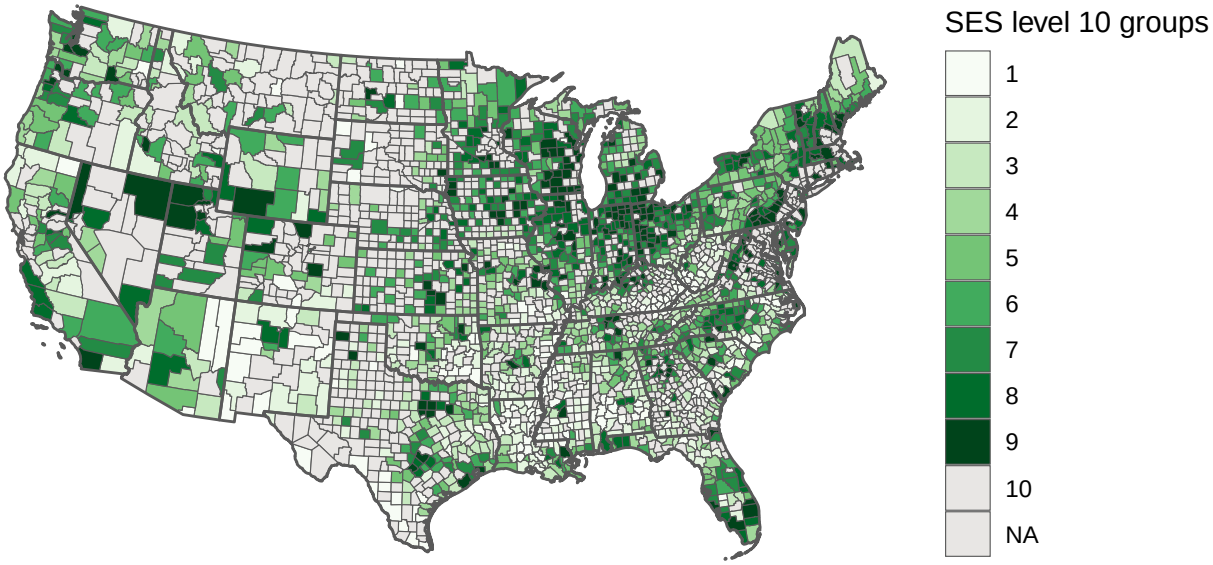
Color palette - discrete categories

```
colors_5 <- c("#EE7600", "#82d6b4", "#3da078", "#057f4e", "#483D8B")

map_4 <- ggplot(UScounties, aes(fill = Pov_factor_5)) +
  geom_sf(color = "grey62", size = cty_line) +
  geom_sf(data = USstates, size = 0.5, fill = NA) +
  scale_fill_manual(values = colors_5, na.value = "#e8e6e4") +
  # scale_fill_manual(values = colors_5, na.value = "grey90") +
  coord_sf(crs = st_crs(5070)) +
  theme_void() +
  labs(fill = "SES level 5 groups")
map_4
```



Default map for 10 SES groups

```
map_10_1 <- ggplot(UScounties, aes(fill = Pov_factor_10)) +
  geom_sf(size = cty_line) +
  geom_sf(data = USstates, fill = NA) +
  scale_fill_brewer(palette = "Greens", na.value = "#e8e6e4") +
  coord_sf(crs = st_crs(5070)) +
  theme_void() +
  labs(fill = "SES level 10 groups")
map_10_1
```



Create levels for map w/ 10 SES groups

```
colourCount <- 10
colors_use <- colorRampPalette(brewer.pal(8, "Greens"))(colourCount)

map_10_2 <- ggplot(UScounties, aes(fill = Pov_factor_10)) +
  geom_sf(size = cty_line) +
  geom_sf(data = USstates, fill = NA) +
  scale_fill_manual(values = colors_use, na.value = "#e8e6e4") +
  coord_sf(crs = st_crs(5070)) +
  theme_void() +
  labs(fill = "SES level 10 groups")
map_10_2
```