ggplot in R and Python

ggplot

- ggplot2 is a plotting system for R, based on the Grammar of Graphics

```
# Install ggplot2 package
> install.package("ggplot2")

# load the ggplot2 package
> library("ggplot2")
```

ggplot2

- ggplot2 is designed to work in a layered fashion
- ggplot() is used to construct the initial plot object, and is almost always followed by + to add component to the plot.
- There are three common ways to invoke ggplot():

```
ggplot(df, aes(x, y, other aesthetics))

ggplot(df)

ggplot()
```

ggplot2

- ggplot(df, aes(x, y, other aesthetics)): The first method is recommended if all layers use the same data and the same set of aesthetics
  - This method can also be used to add a layer using data from another data frame.
- ggplot(df): The second method specifies the default data frame to use for the plot, but no aesthetics are defined up front.
  - This is useful when one data frame is used predominantly as layers are added, but the aesthetics may vary from one layer to another.
- ggplot(): the third method initializes a skeleton ggplot object which is fleshed out as layers are added.
  - This method is useful when multiple data frames are used to produce different layers, as is often the case in complex graphics.

Data file

- Cancer datat from CORGIS Dataset Project
- Information about the rates of cancer deaths in each state is reported.
- The data shows the total rate as well as rates based on sex, age, and race.
- Rates are also shown for three specific kinds of cancer: breast cancer, colorectal cancer, and lung cancer.

```
> cancer.data <- read.csv("Cancer-rates-DATA.csv", header = T)

# show the structure of the object cancer.data
> str(cancer.data)

# check whether dataset contains any missing value
> table(is.na(cancer.data))
```

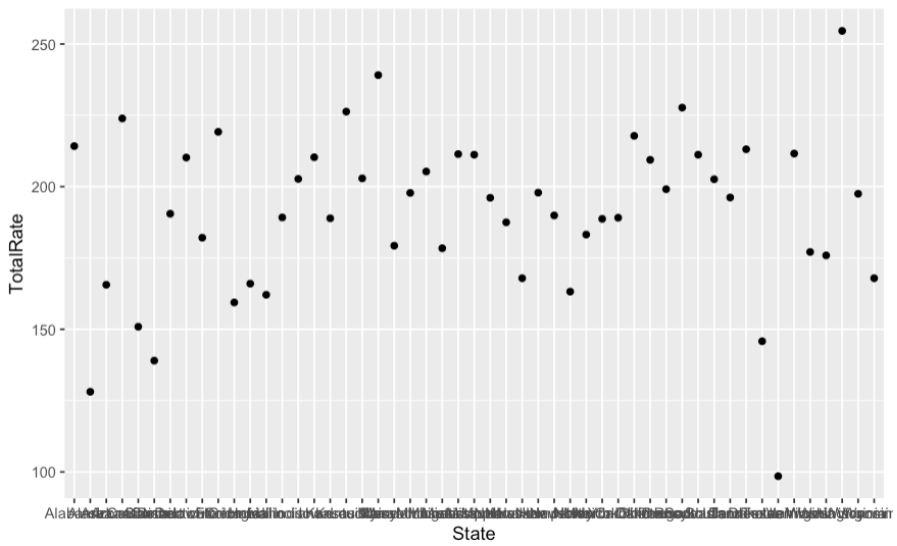
Scatter Plot

Question 1: What is the trend of the overall cancer death rate?

- Scatter plots are used to display the relationship between two continuous variables.
- In a scatter plot, each observation in a data set is represented by a point.
- Use **geom_point()**, and map one variable to x and one to y.

Scatter Plot

```
ggplot (cancer.data, aes(x = State,y = TotalRate)) + geom_point()
```



Improve appearance the plot

- Change the appearce of axis text
- Change the axis label
- Add a title to the figure
- Plot the data in the descending order

Changing the appearance of text: theme() and element_text()

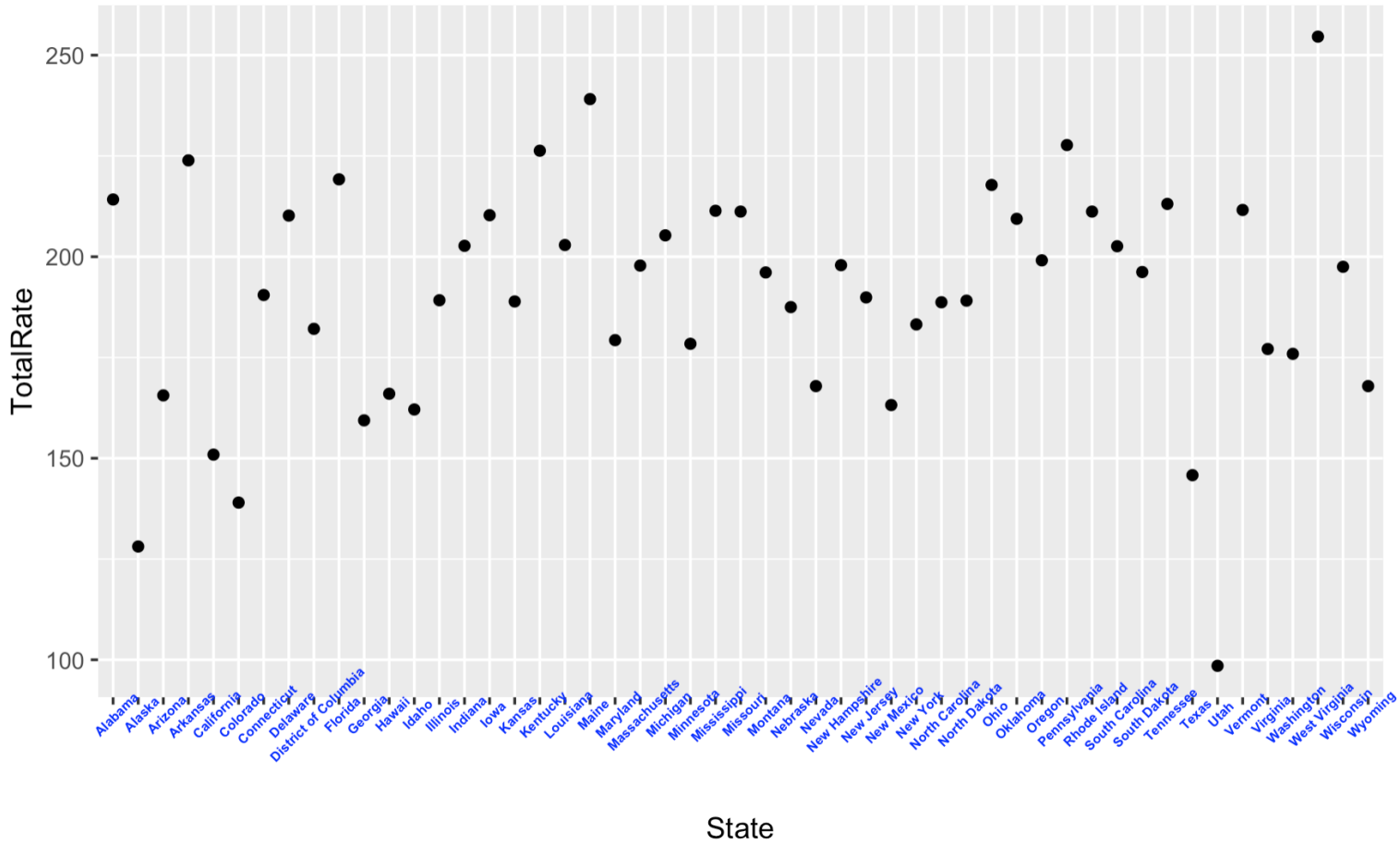
- To set the appearance of theme items such as the title, axis labels, and axis tick marks, use **theme()** and set the item with **element_text()**.
- Use **theme()** to modify individual components of a theme, allowing you to control the appearance of all **non-data** components of the plot
- **element_text()**: In conjunction with the **theme** system, the *element_* functions specify the display of how non-data components of the plot are drawn.
  - element_blank: draws nothing, and assigns no space.
  - element_rect: borders and backgrounds.
  - element_line: lines.
  - element_text: text.

Theme items that control text appearance in theme()

Element name	Description
`axis.title`	Appearance of axis labels on both axes
`axis.title.x`	Appearance of x-axis label
`axis.title.y`	Appearance of y-axis label
`axis.ticks`	Appearance of tick labels on both axes
`axis.ticks.x`	Appearance of x tick labels
`axis.ticks.y`	Appearance of y tick labels
`legend.title`	Appearance of legend title
`legend.text`	Appearance of legend items
`plot.title`	Appearance of overall plot title
`strip.text`	Appearance of facet labels in both directions
`strip.text.x`	Appearance of horizontal facet labels
`strip.text.y`	Appearance of vertical facet labels

Change the appearance of axis text

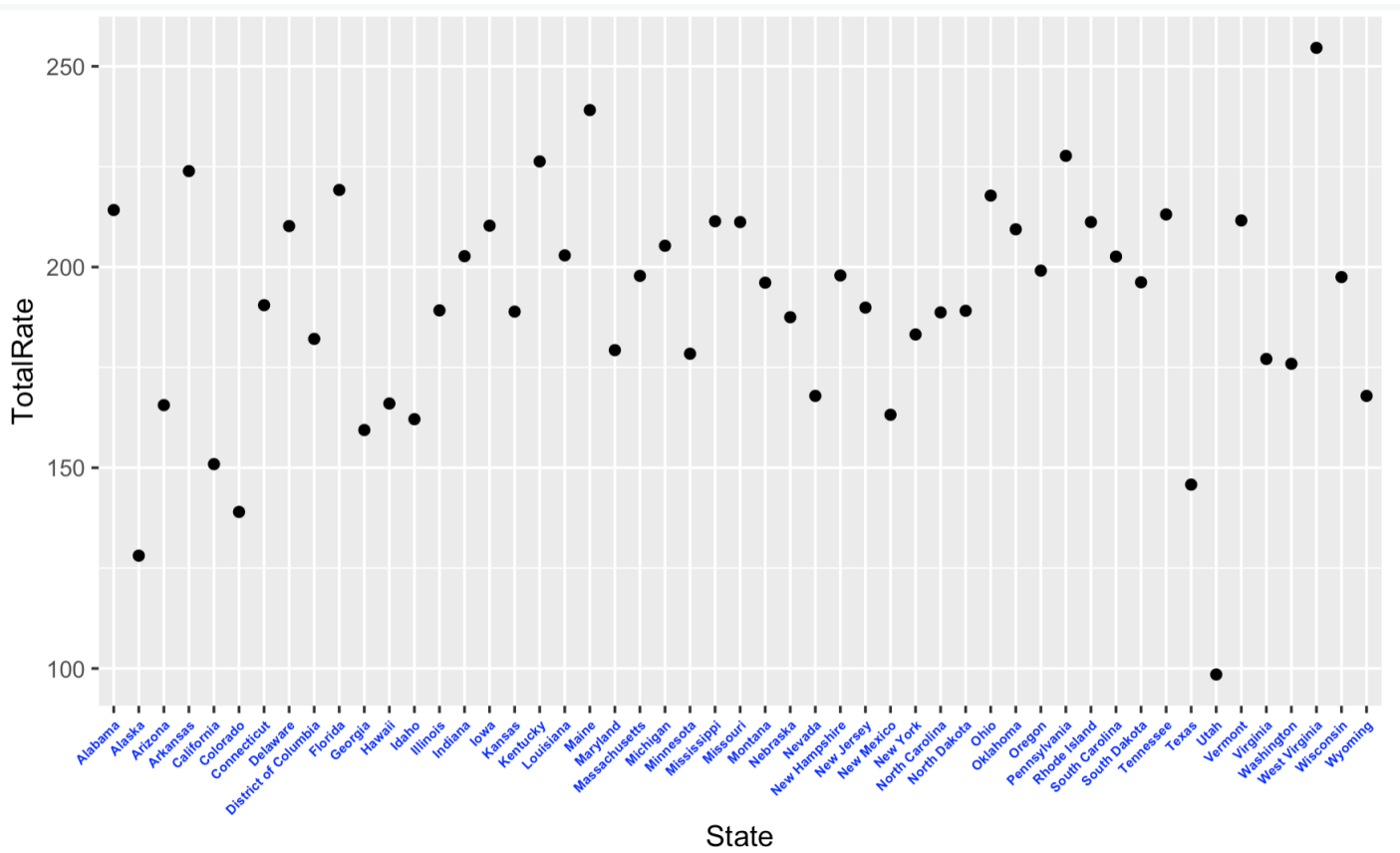
```
ggplot (cancer.data, aes(x = State,y = TotalRate)) + geom_point() +
theme(axis.text.x =element_text(face="bold", color="blue", size=5, angle=45))
```



Adjust the position of axis text

To adjust the position of the axis text, you can specify the argument **hjust** and **vjust**, whcih values should be comprised between 0 and 1.

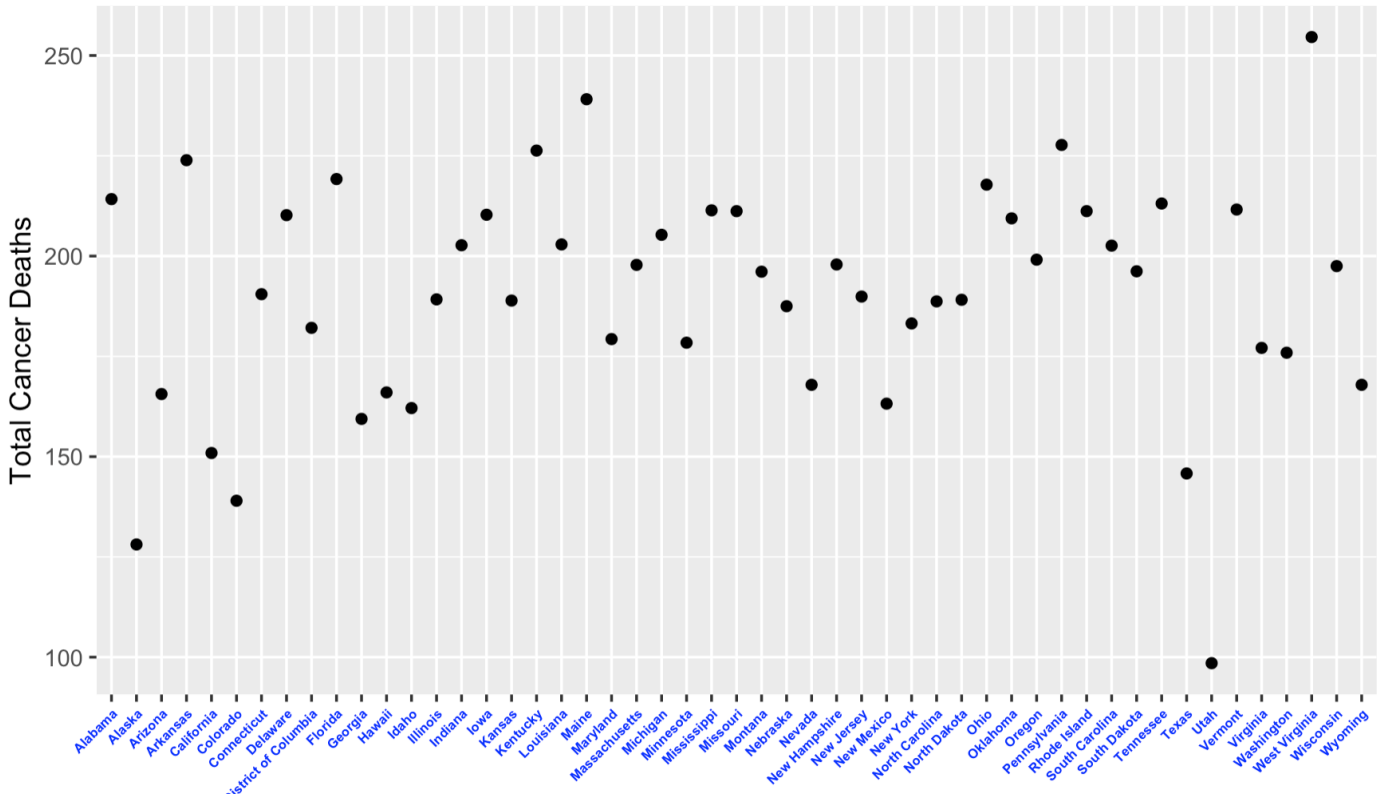
```
ggplot (cancer.data, aes(x = State,y = TotalRate)) + geom_point() +
theme(axis.text.x =element_text(face="bold", color="blue", size=5, angle=45, hjust=1))
```



Change the axis label

Use function **labs()** or use **xlab()** for x axis and **ylab()** for y axis

```
ggplot (cancer.data, aes(x = State,y = TotalRate)) + geom_point() +
  labs (y = "Total Cancer Deaths", x = "") +
theme(axis.text.x =element_text(face="bold", color="blue", size=5, angle=45, hjust=1))
```

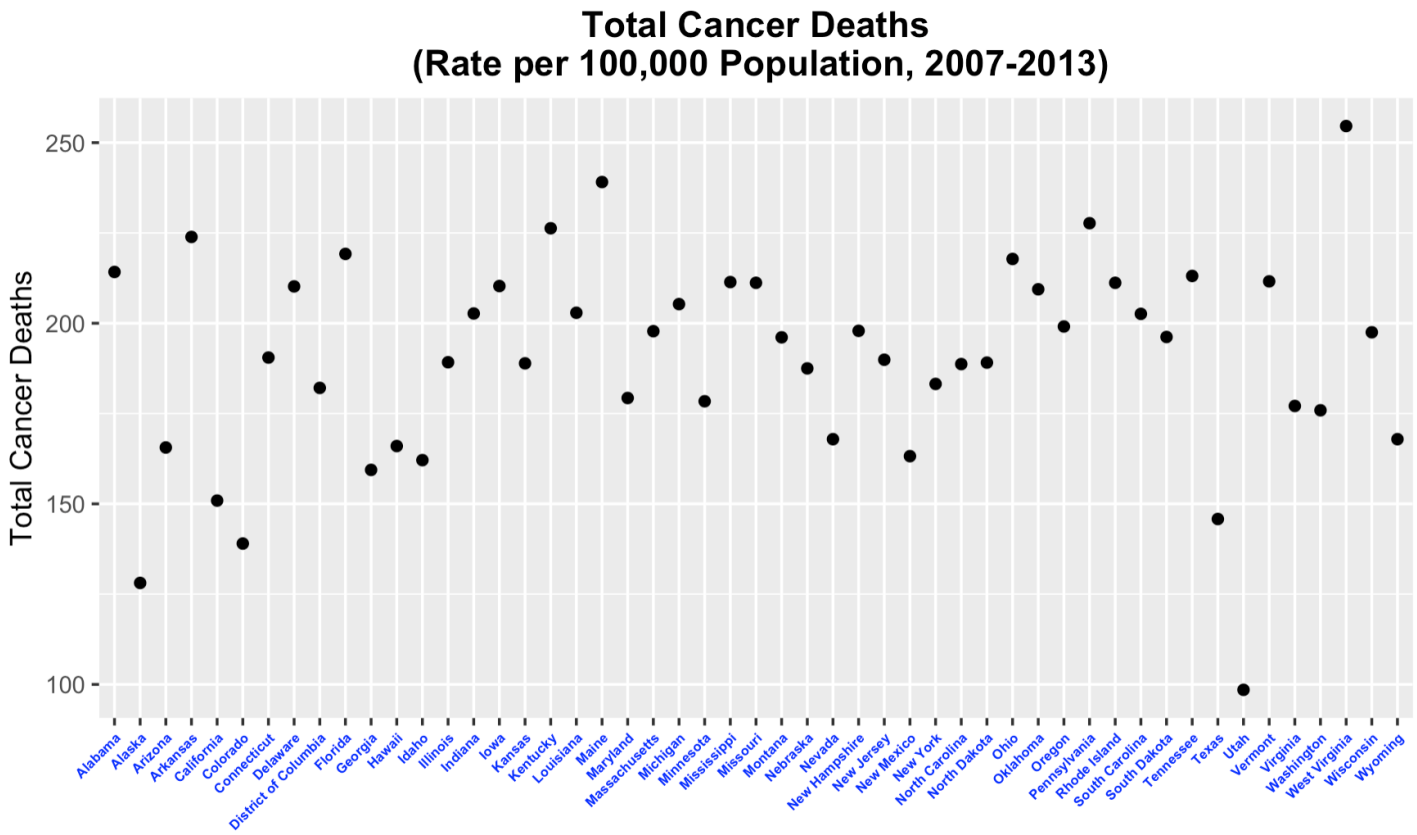


Add titles

- Add titles and subtitles by using either the function **ggtitle()** or **labs()**.
- The position, font size and color of a title can be changed
- The default ggplot title alignment is not centered. It is on the left.
- It's possible to put the title in the middle of the chart by specifying the argument hjust $= 0.5$ in the function
  - The options hjust $= 1$: place titles on the right of the plot
  - hjust $= 0$: place titles on the left of the plot

Add a title

```
ggplot (cancer.data, aes(x = State, y = TotalRate)) + geom_point() +
  labs (y = "Total Cancer Deaths", x="", title="Total Cancer Deaths\n (Rate per 100,000
        Population, 2007−2013)") +
theme(axis.text.x =element_text(face="bold", color="blue", size=5, angle=45, hjust=1)) +
theme(plot.title = element_text(face = "bold", hjust=0.5))
```



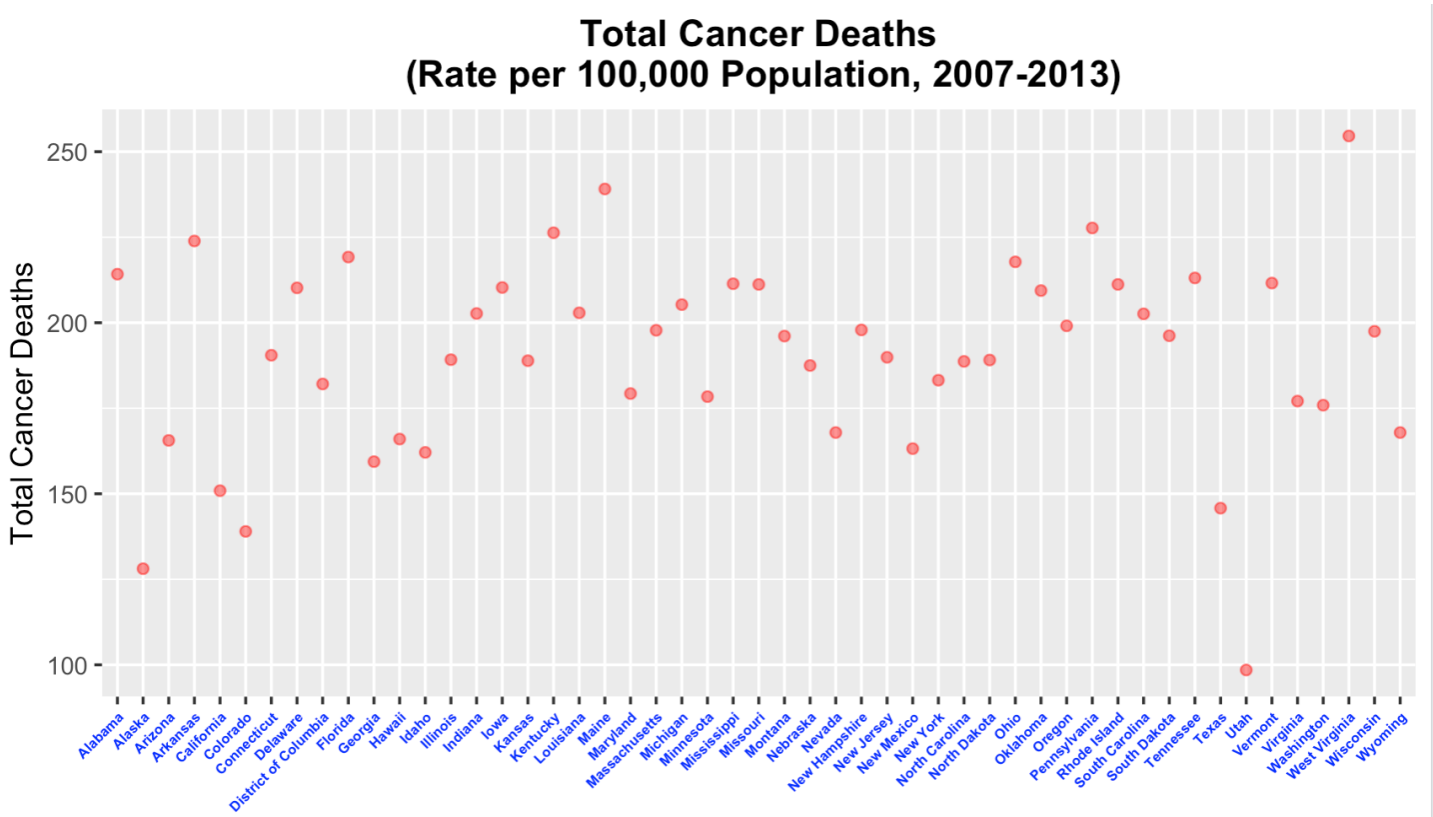
Aesthetics: geom_point

Aesthetics

- x
- y
- size
- shape
- color
- alpha

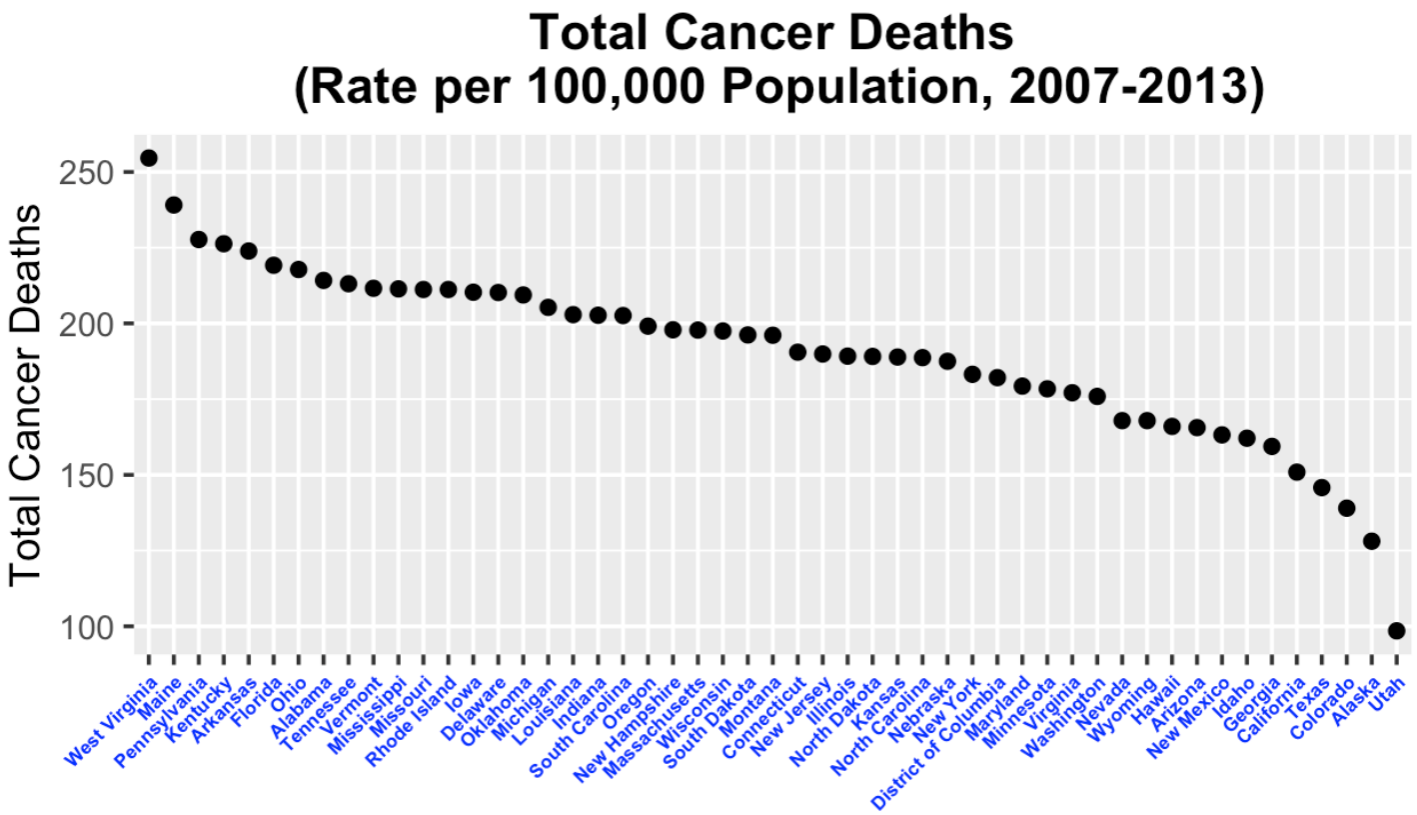
Aesthetics: geom_point

```
ggplot (cancer.data, aes(x = State, y = TotalRate)) + geom_point(color=''red'', alpha=0.5) +
  labs (y = "Total Cancer Deaths", x="", title="Total Cancer Deaths\n (Rate per 100,000
       Population, 2007−2013)") +
theme(axis.text.x =element_text(face="bold", color="blue", size=5, angle=45, hjust=1)) +
theme(plot.title = element_text(face = "bold", hjust=0.5))
```



Plot according to descending order of death rate

```
ggplot (cancer.data, aes(x = reorder(State, -TotalRate), y = TotalRate)) + geom_point() +
  labs (y = "Total Cancer Deaths", title="Total Cancer Deaths\n (Rate per 100,000
        Population, 2007-2013)", x="State") +
theme(axis.text.x =element_text(face="bold", color="blue", size=5, angle=45, hjust=1)) +
theme(plot.title = element_text(face = "bold", hjust=0.5))
```

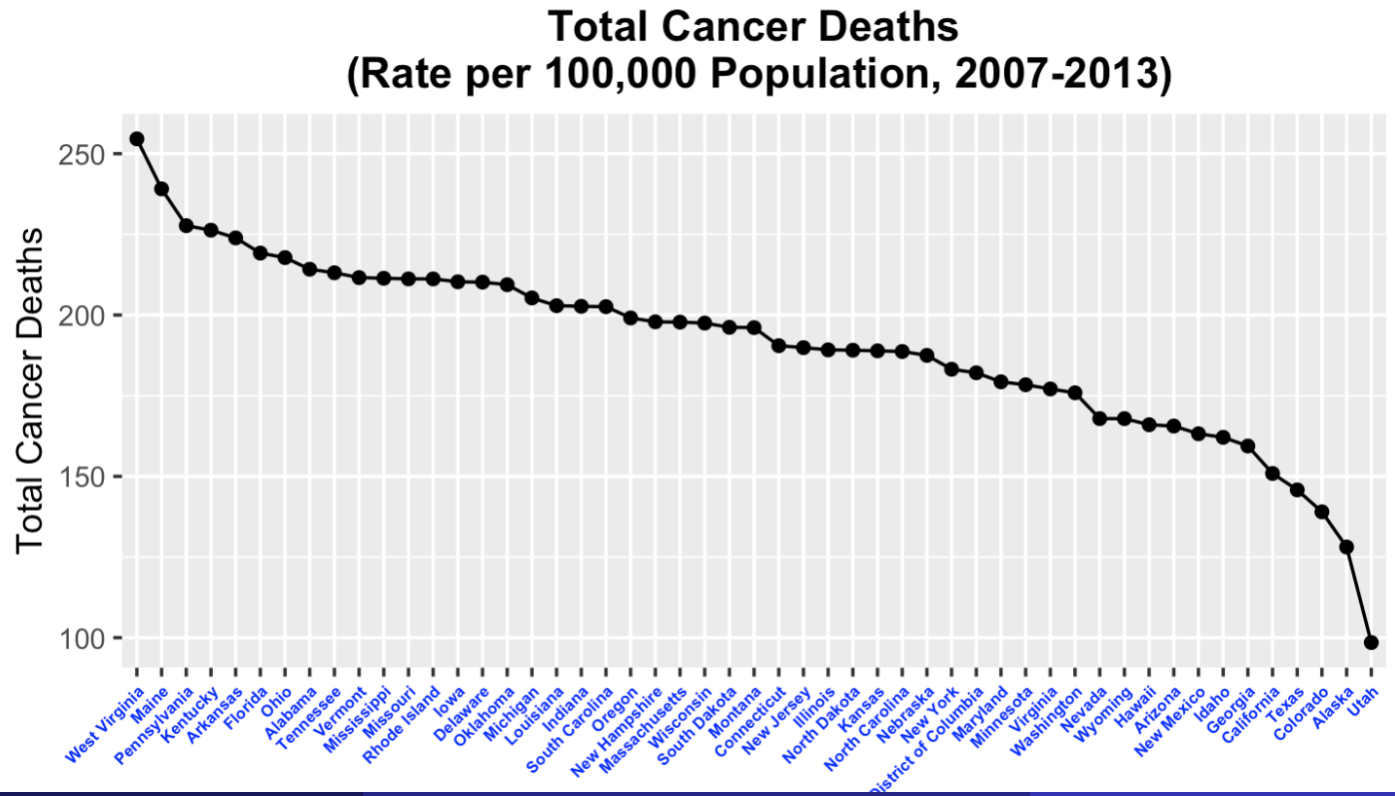


Line graph

- Line graphs are typically used for visualizing how one continuous variable, on the yaxis, changes in relation to another continuous variable, on the x-axis.
- Line graphs can also be used with a discrete variable on the x-axis.
- When the x variable is a factor, you must also use aes(group=1) to ensure that ggplot() knows that the data points belong together and should be connected with a line

Line graph

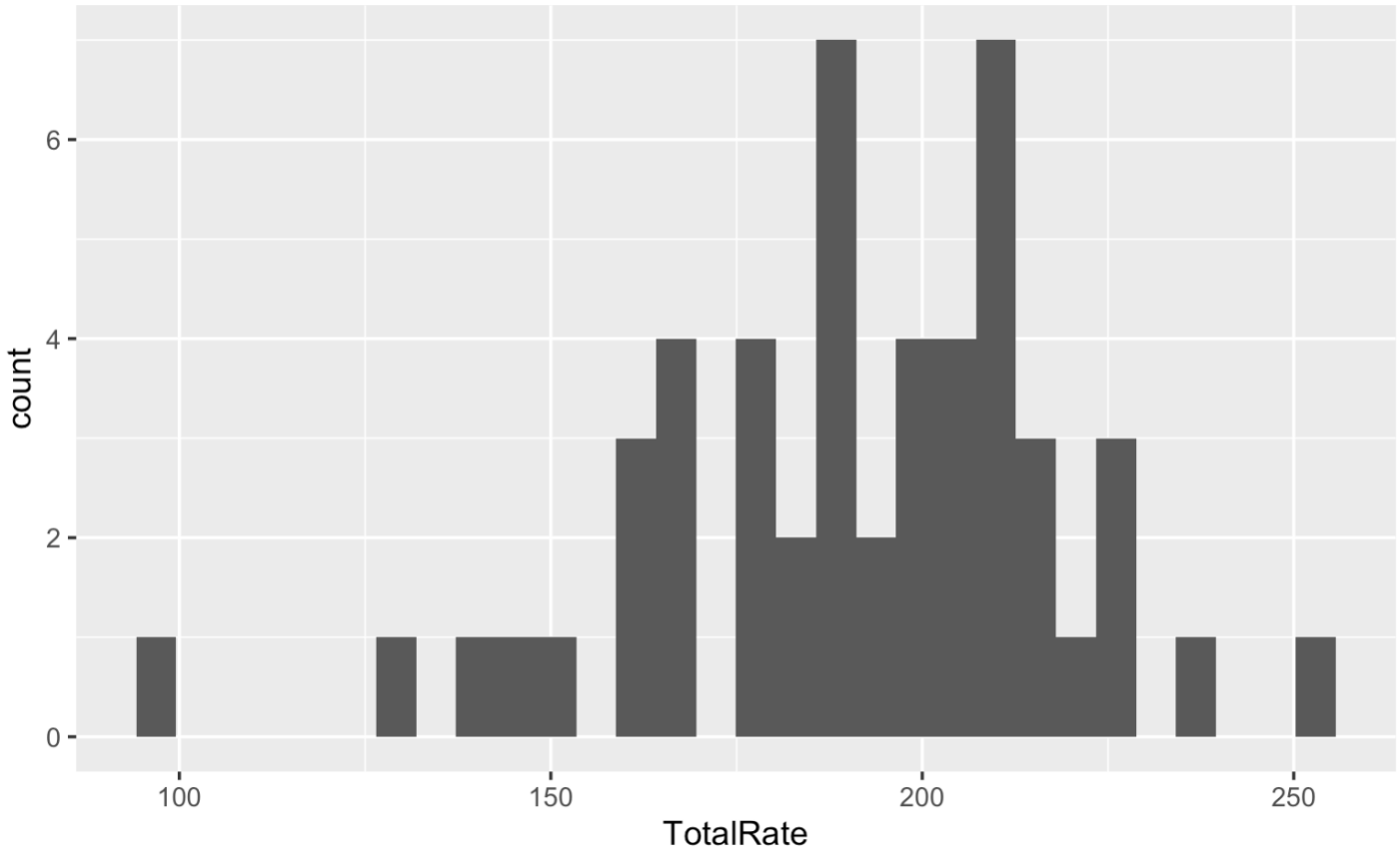
```
ggplot (cancer.data, aes(x = reorder(State, -TotalRate),y = TotalRate, group=1)) + geom_
      point() + geom_line() +
  labs (y = "Total Cancer Deaths", title="Total Cancer Deaths\n (Rate per 100,000
       Population, 2007-2013)", x="") +
theme(axis.text.x =element_text(face="bold", color="blue", size=5, angle=45, hjust=1)) +
theme(plot.title = element_text(face = "bold", hjust=0.5))
```



Histogram

- Use **geom_histogram()** and map a continuous variable to x

```
ggplot (cancer.data, aes (x=TotalRate)) + geom_histogram()
```

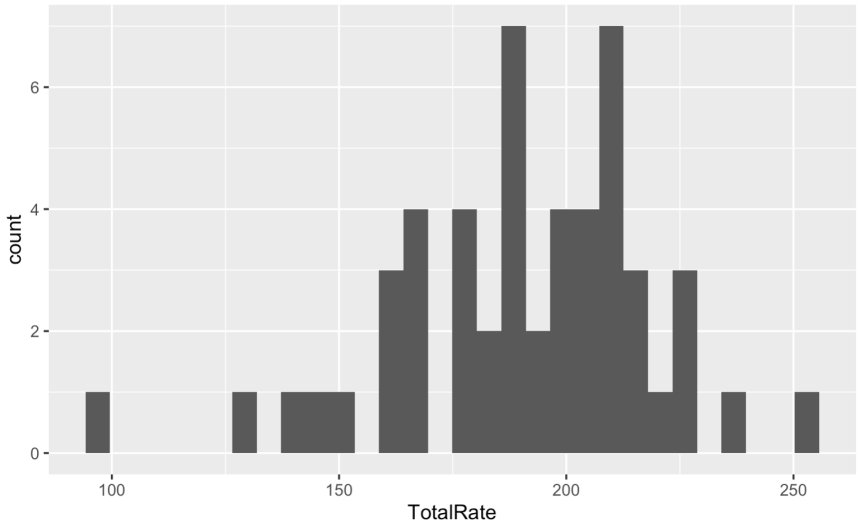


Histogram

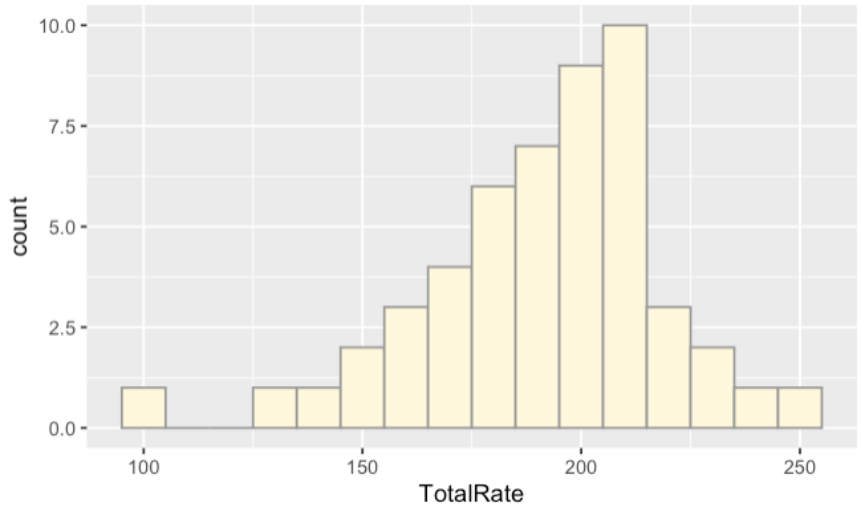
- You can change the size of the bins by using binwidth, or you can divide the range of the data into a specific number of bins.
- The default colors is a dark fill without an outline, which can make it difficult to see which bar corresponds to which value.
- We can change color of fill and outline of bins in the histogram

Histogram

```
ggplot (cancer.data, aes (x=TotalRate)) +
    geom_histogram()
```



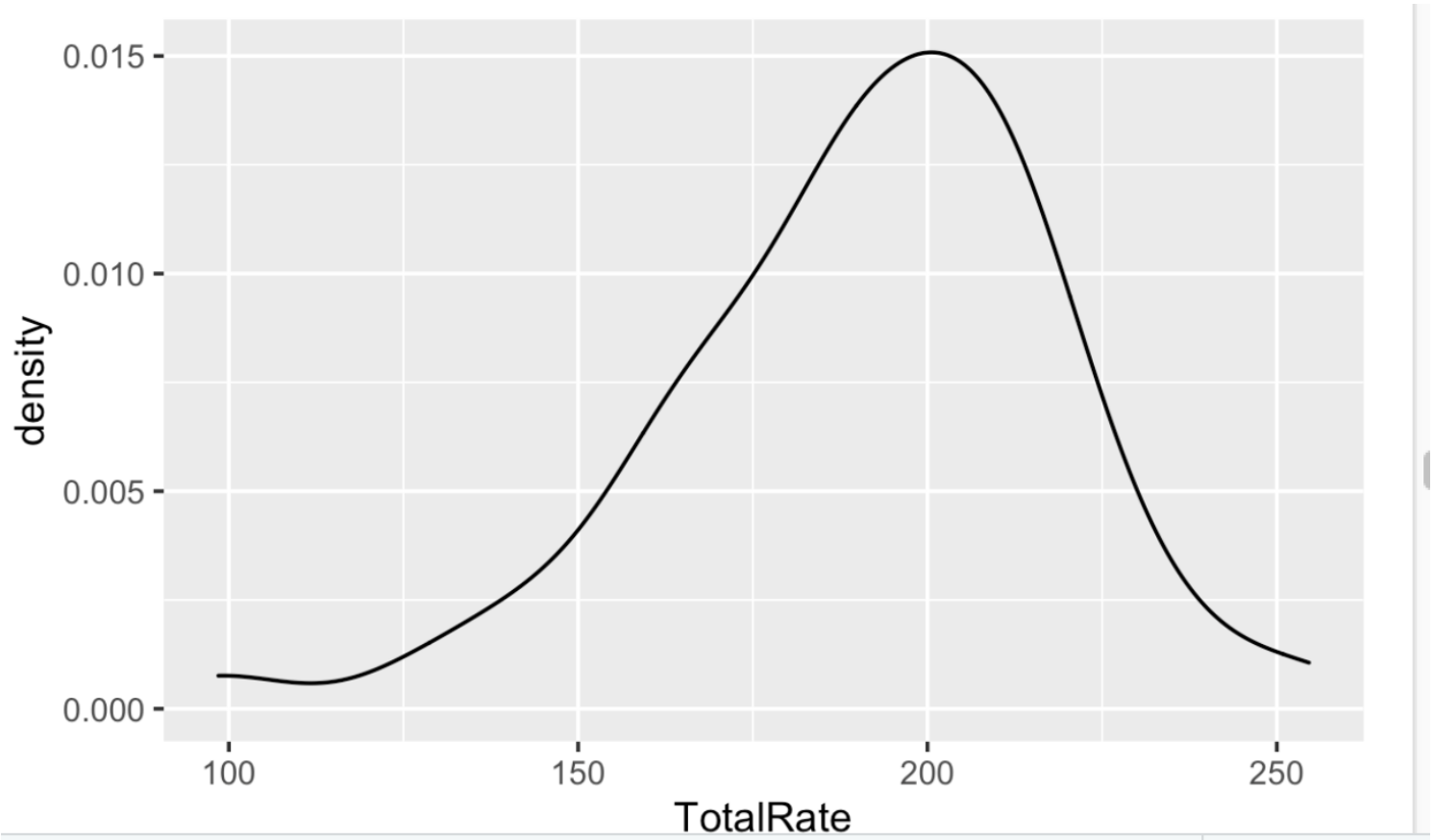

```
ggplot (cancer.data, aes (x=TotalRate)) +
    geom_histogram(binwidth=10, fill="
    cornsilk", colour="grey60")
```



Density for continuous variable

- Use **geom_density()** or **geom_line(stat="density")**, map a continuous variable to x axis)

```
ggplot (cancer.data, aes (x=TotalRate)) + geom_density()
```

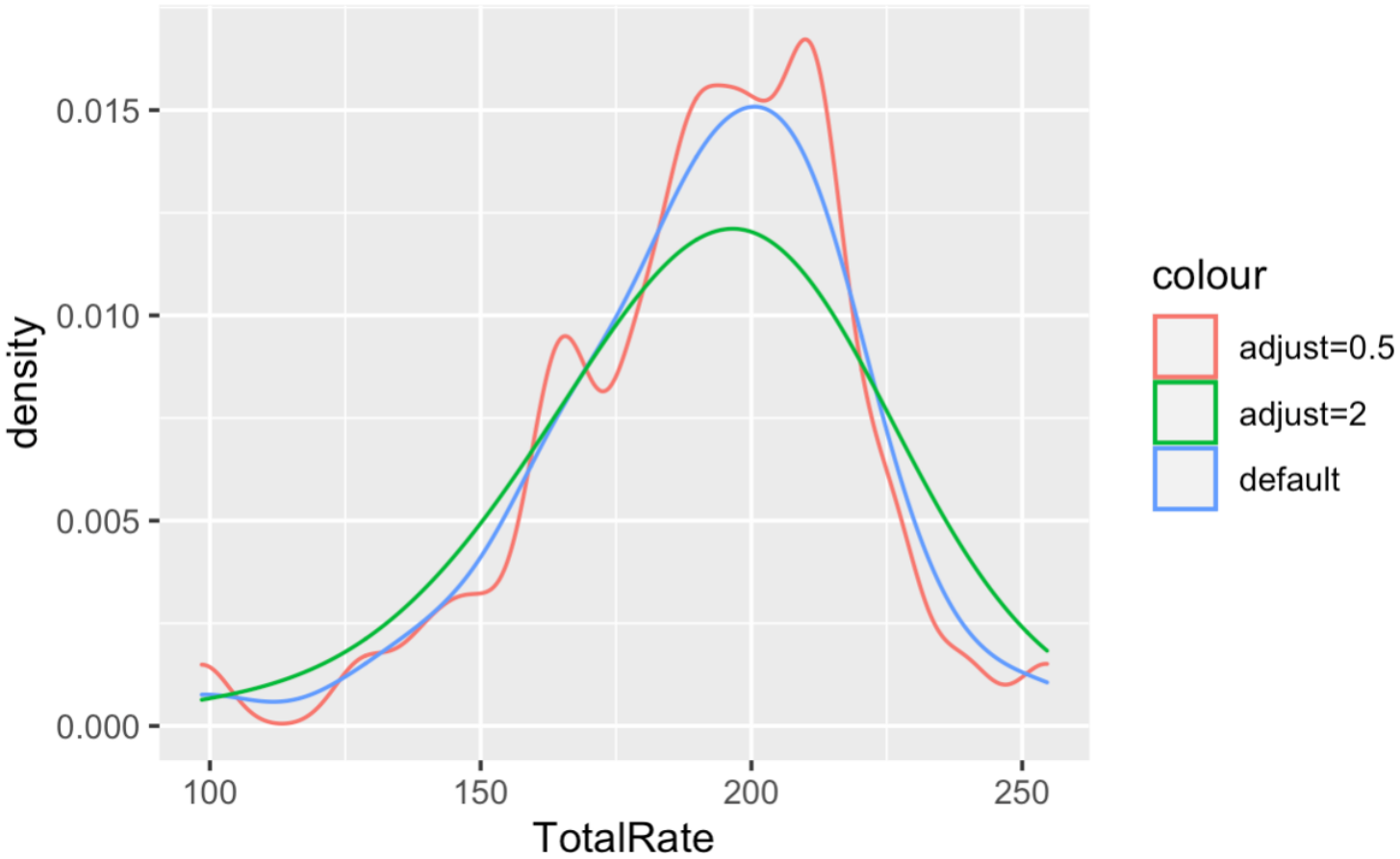


Density

- The kernel density curve is an estimate of the population distribution, based on the sample data.
- The amount of smoothing depends on the kernel bandwidth
- The larger the bandwidth, the more smoothing there is.
- The bandwidth can be set with the **adjust** parameter, which has a default value of 1.

Density

```
ggplot (cancer.data, aes (x=TotalRate)) + geom_density(aes(color='adjust=0.5'), adjust=0.5)
+ geom_density(aes(color='default')) + geom_density(aes(color='adjust=2'), adjust=2)
```

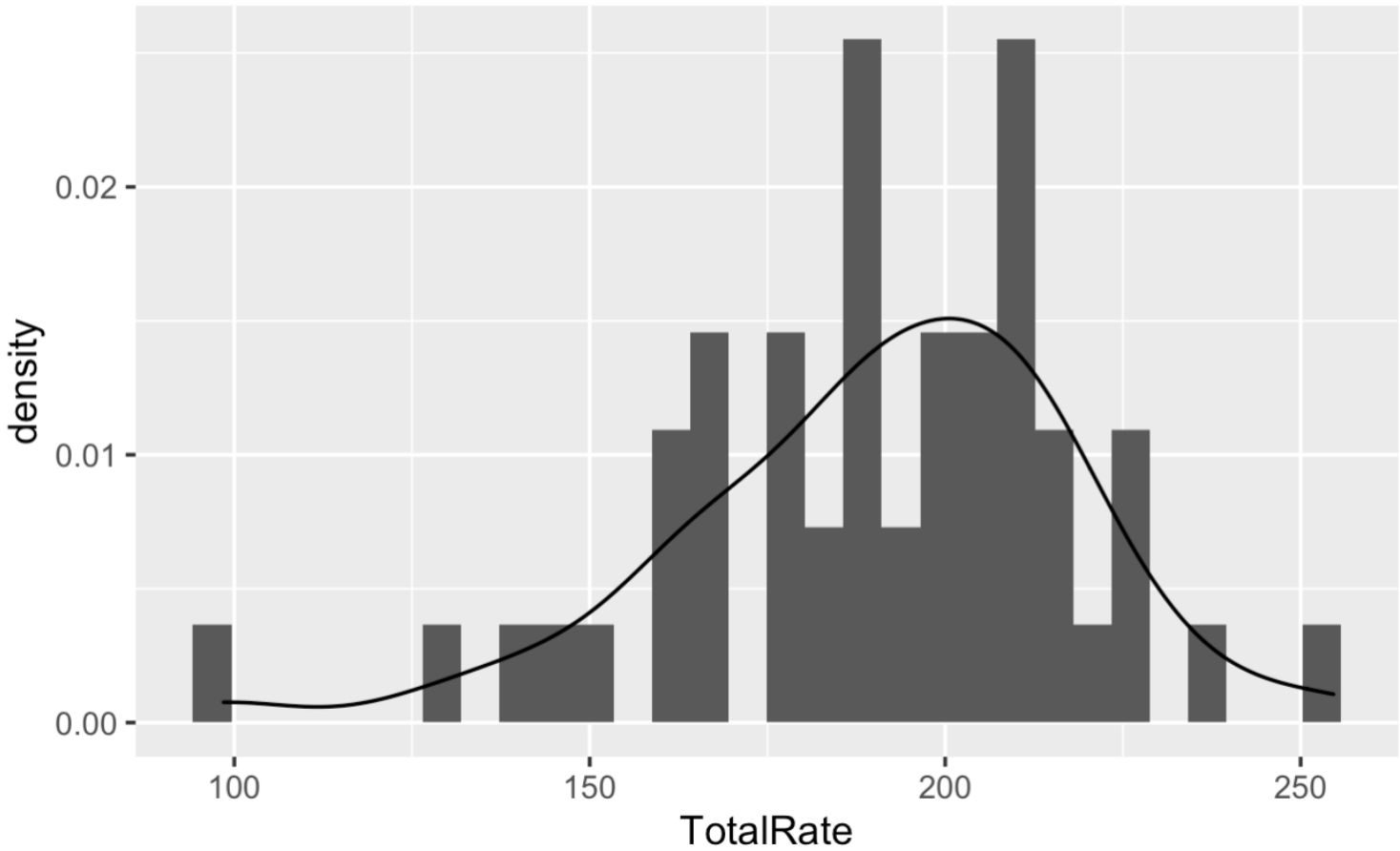


Histogram and Density

- To compare the theoretical and observed distributions, you can overlay the density curve with the histogram.
- Use geom density() and map a continuous variable to x axis
- Since the y values for the density curve are small (the area under the curve always sums to 1), it would be barely visible if you overlaid it on a histogram without any transformation.
- To solve this problem, you can scale down the histogram to match the density curve with the mapping **y=..density..**
- We will add geom_histogram() first, and then layer geom_density() on top

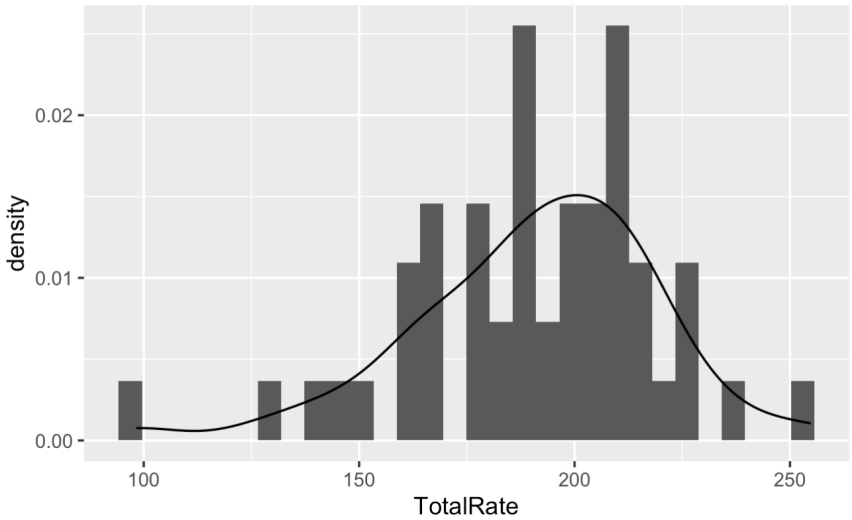
Histogram and Density

```
ggplot (cancer.data, aes (x=TotalRate, y=..density..)) + geom_histogram() + geom_density()
```

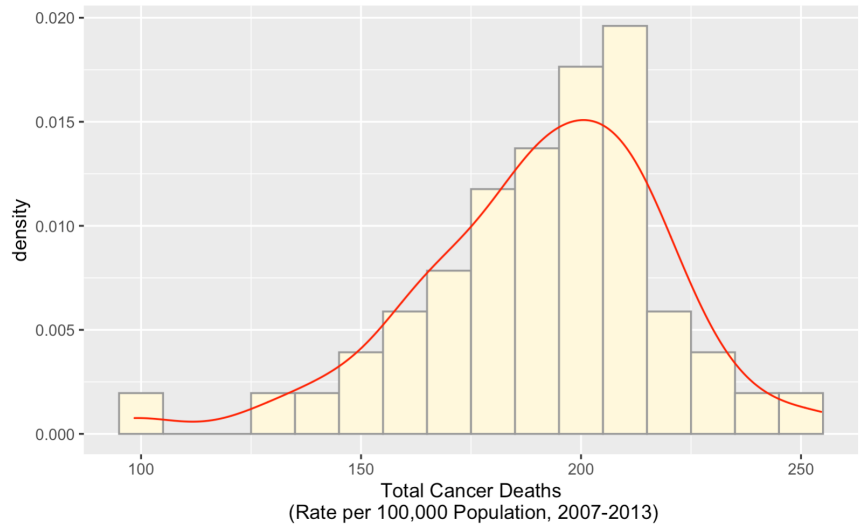


Change appearance of the histogram and density curve

```
ggplot (cancer.data, aes (x=TotalRate, y=..
      density..)) + geom_histogram() + geom_
      density()
```




```
ggplot (cancer.data, aes (x=TotalRate, y=..
      density..)) + geom_histogram(binwidth
      =10, fill="cornsilk", colour="grey60") +
       geom_density(color='red') + xlab("Total
       Cancer Deaths\n (Rate per 100,000
      Population, 2007−2013)")
```

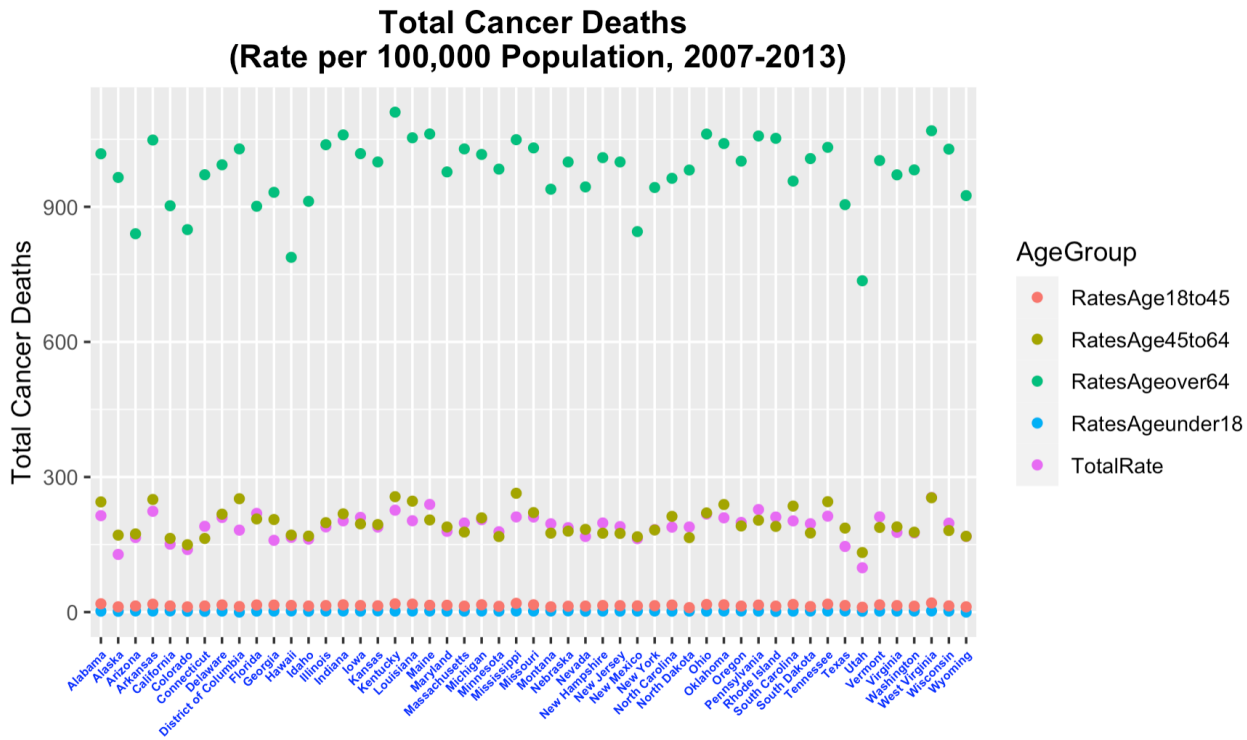


Question 2

Question 2: Compare death rates of diffferent age groups.

Compare different types of data

```
> all = ggplot(cancer.data, aes(x=State))+ geom_point(aes( y=TotalRate, color='all'))
> age1 = geom_point (aes( y=RatesAgeunder18, color='Age under 18'))
> age2 = geom_point (aes(y=RatesAge18to45, color='Age 18 to 45'))
> age3 = geom_point (aes(y=RatesAge45to64, color='Age 45 to 64'))
> age4 = geom_point (aes(y=RatesAgeover64, color='Age over 64'))

> all + age1 + age2 + age3 + age4 + labs (y = "Total Cancer Deaths", x="", title="Total Cancer Deaths\n (Rate per
      100,000 Population, 2007-2013)") +
theme(axis.text.x =element_text(face="bold", color="blue", size=5, angle=45, hjust=1)) +
theme(plot.title = element_text(face = "bold", hjust=0.5))
```



Reshape data into long format data for ggplot

- ggplo has built the entire "tidyverse" ecosystem on the concept of tidy data, which is essentially data in long format.
- In a wide format, columns represent state, TotalRate,RatesAgeunder18, RatesAge18to45, RatesAge45to64, RatesAgeover64 etc.
- In a long format, merge the information into a single column.
  - You can simply specify which variable you want to use as the grouping variable
  - Reshape the data frame into a long format using **pivot_longer()** in the package *tidyr*

Reshape data

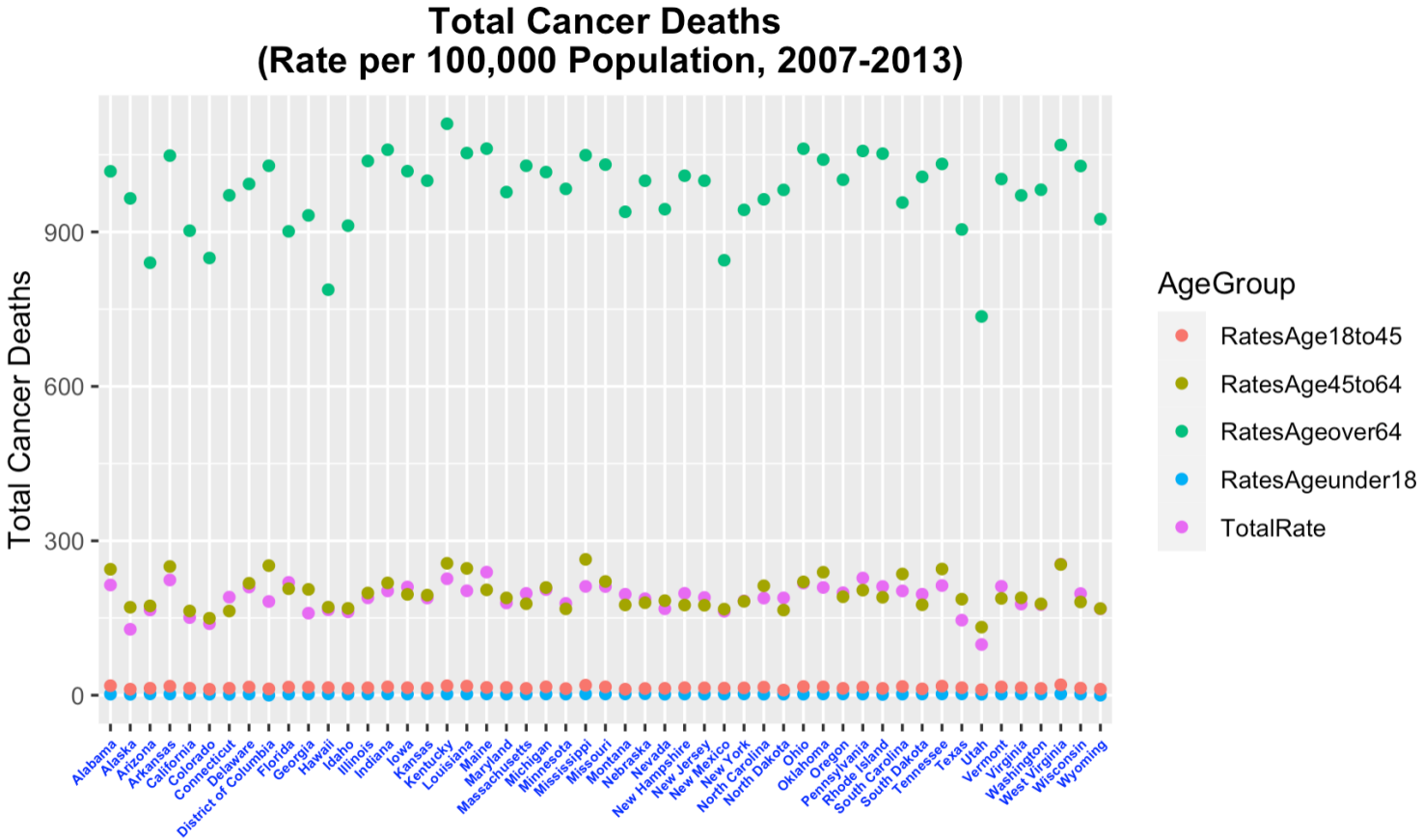
```
# install package tidyr if it has not been installed
if (!require("tidyr"))
  install.packages("tidyr")

# Load the package to your workplace
library(tidyr)

ageGroup = c('TotalRate', 'RatesAgeunder18', 'RatesAge18to45','
    RatesAge45to64','RatesAgeover64')
cancer.age= subset(cancer.data, select=c("State", ageGroup))
cancerr.age.longformat <- pivot_longer(cancer.age, cols=ageGroup, names_to
    ='AgeGroup', values_to = 'CancerRate')
```

Reshape data to long format

```
> str(cancer.age)
'data.frame':   51 obs. of  6 variables:
 $ State          : Factor w/ 51 levels "Alabama","Alaska",..: 1 2 3 4 5 6 7
 8 9 10 ...
 $ TotalRate      : num  214 128 166 224 151 ...
 $ RatesAgeunder18: num  2 1.7 2.5 2.3 2.6 1.9 1.6 2.2 0 2.1 ...
 $ RatesAge18to45 : num  18.5 11.8 13.6 17.6 13.7 11.7 13.6 16 12.4 15.9 ...
 $ RatesAge45to64 : num  245 171 174 250 164 ...
 $ RatesAgeover64 : num  1018 965 840 1048 902 ...
>
> str(cancerr.age.longformat)
tibble [255 × 3] (S3: tbl_df/tbl/data.frame)
 $ State     : Factor w/ 51 levels "Alabama","Alaska",..: 1 1 1 1 1 2 2 2 2 2
 ...
 $ AgeGroup  : chr [1:255] "TotalRate" "RatesAgeunder18" "RatesAge18to45" "Ra
tesAge45to64" ...
 $ CancerRate: num [1:255] 214.2 2 18.5 244.7 1017.8 ...
>
```

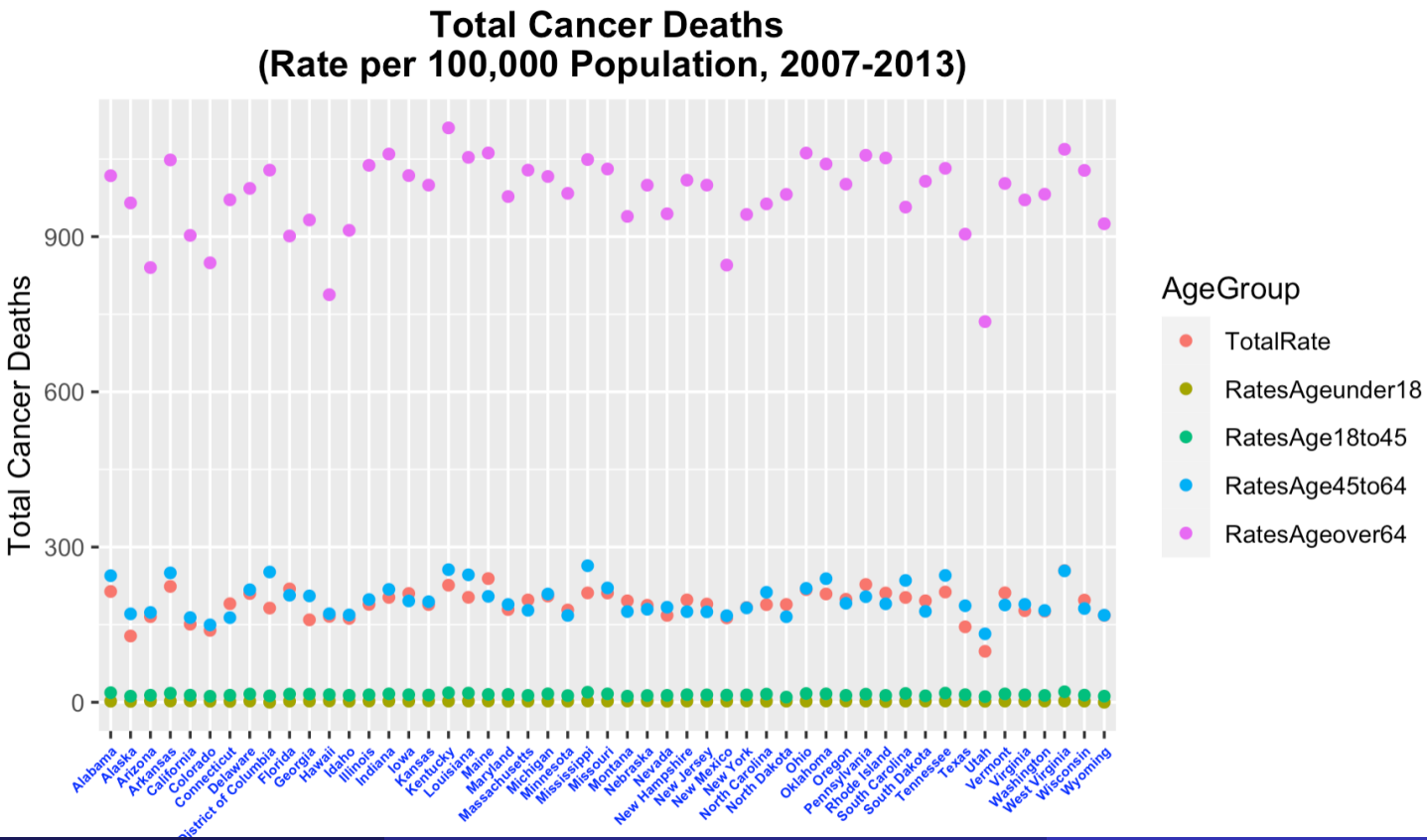
Scatter plot for comparison

```
ggplot (cancerr.age.longformat, aes(x=State, y=CancerRate, col=AgeGroup)) + geom_point()
    + labs (y = "Total Cancer Deaths",x="", title="Total Cancer Deaths\n (Rate per
    100,000 Population, 2007−2013)") +
theme(axis.text.x =element_text(face="bold", color="blue", size=5, angle=45, hjust=1)) +
   theme(plot.title = element_text(face = "bold", hjust=0.5))
```



Improved the scatter plot for comparison

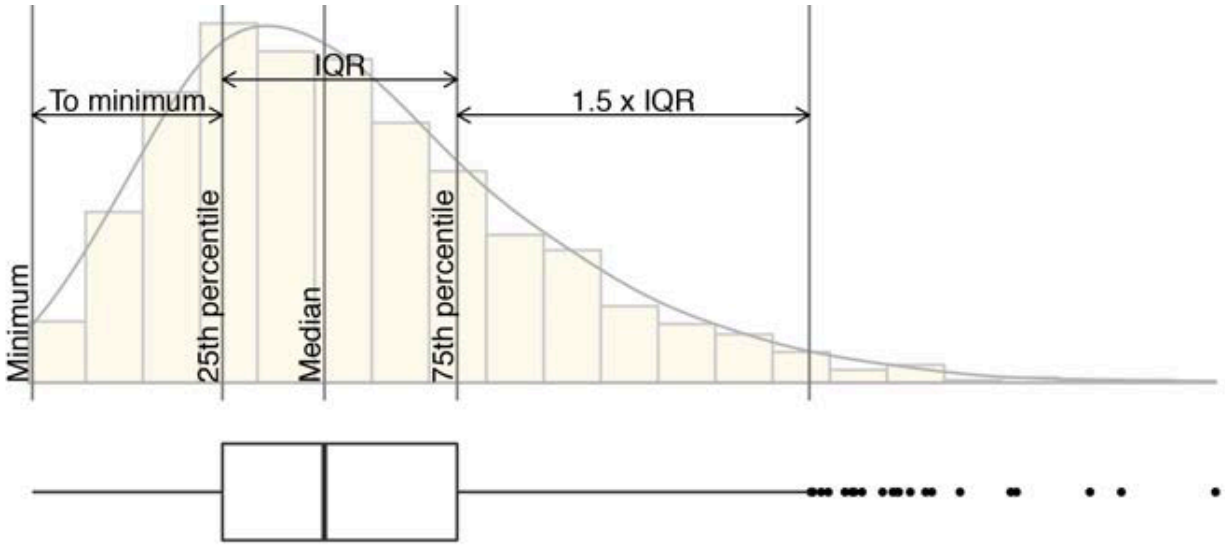
```
> ageGroup = c('TotalRate', 'RatesAgeunder18', 'RatesAge18to45','RatesAge45to64','RatesAgeover64')
#Change default levels of a factor, which is in alphabetical order.
> cancerr.age.longformat$AgeGroup = factor(cancerr.age.longformat$AgeGroup, levels = ageGroup)
> ggplot (cancerr.age.longformat, aes(x=State, y=CancerRate, col=AgeGroup)) + geom_point() + labs (y = "Total Cancer
      Deaths",x="", title="Total Cancer Deaths\n (Rate per 100,000 Population, 2007−2013)") +
theme(axis.text.x =element_text(face="bold", color="blue", size=5, angle=45, hjust=1)) +
    theme(plot.title = element_text(face = "bold", hjust=0.5))
```



Box plot

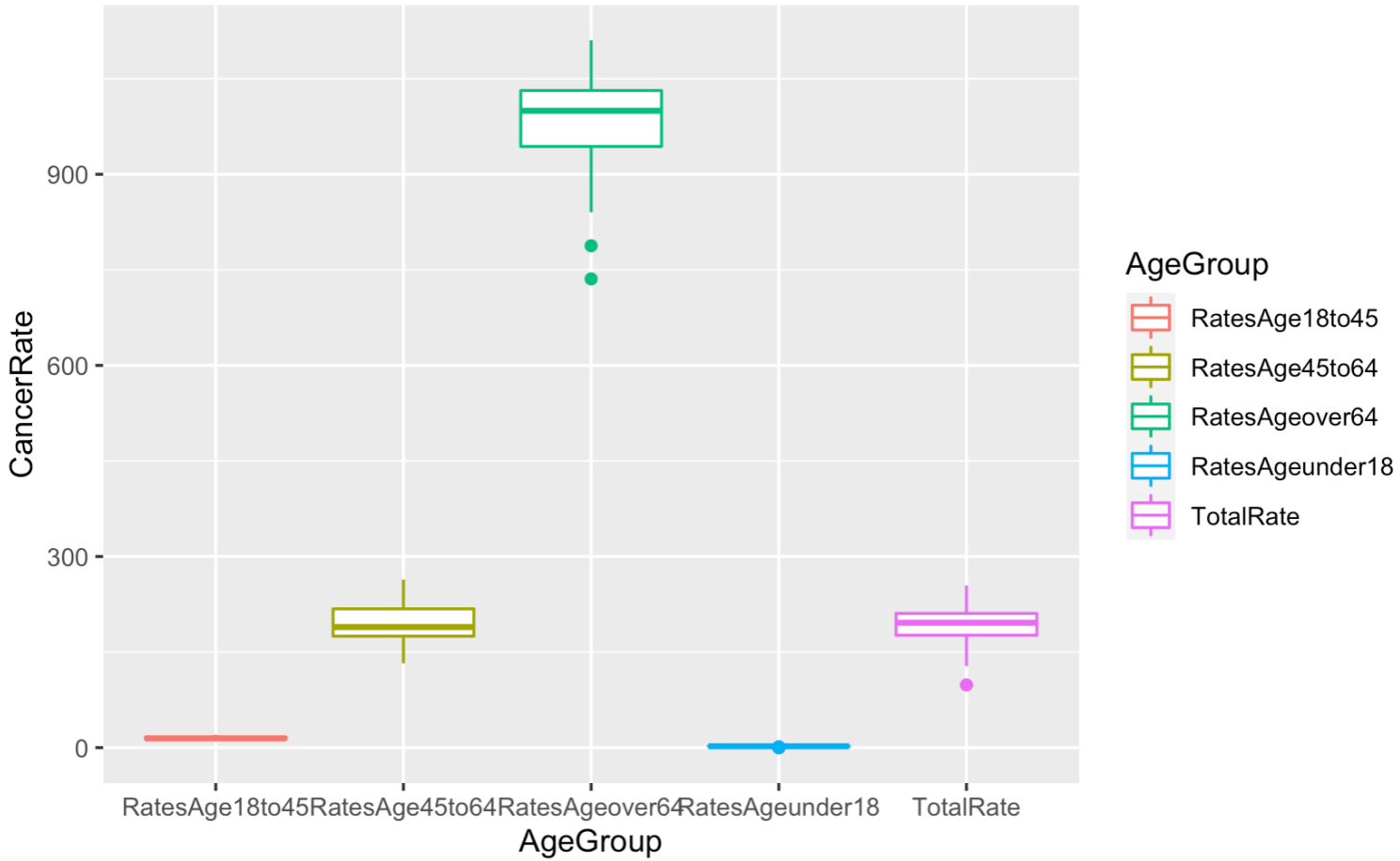
- A box plot consists of a box and "whiskers"
- The box goes from the 25th percentile to the 75th percentile of the data, also known as the interquartile range (IQR).
- There is a line indicating the median, or 50th percentile of the data.
- The whiskers start from the edge of the box and extend to the furthest data point that is within 1.5 times the IQR.
- If there are any data points that are past the ends of the whiskers, they are considered outliers and displayed with dots

.



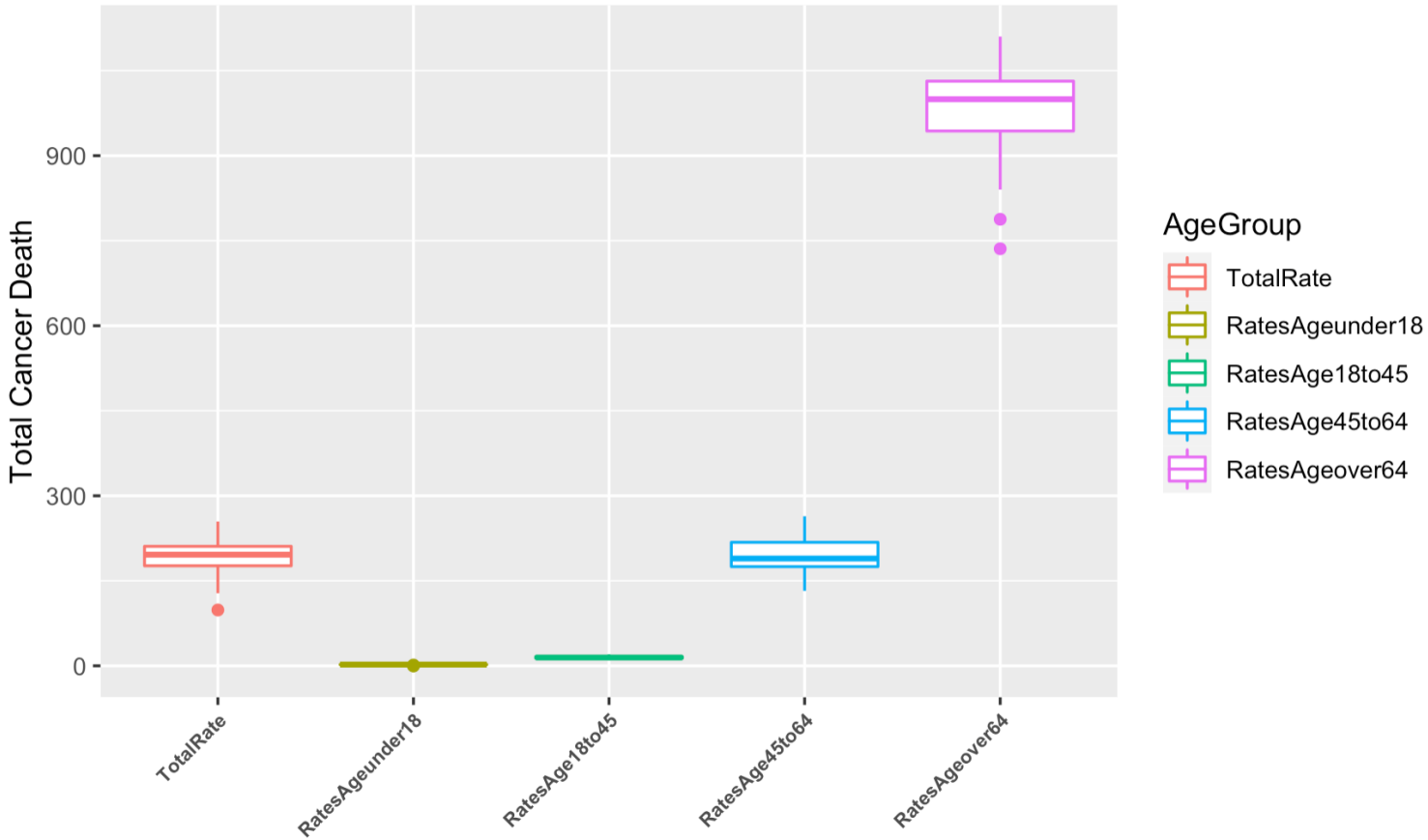
Box plot for comparison

```
ggplot (cancerr.age.longformat, aes(x=AgeGroup, y=CancerRate, col=AgeGroup)) + geom_
    boxplot()
```



Improved box plot for comparison

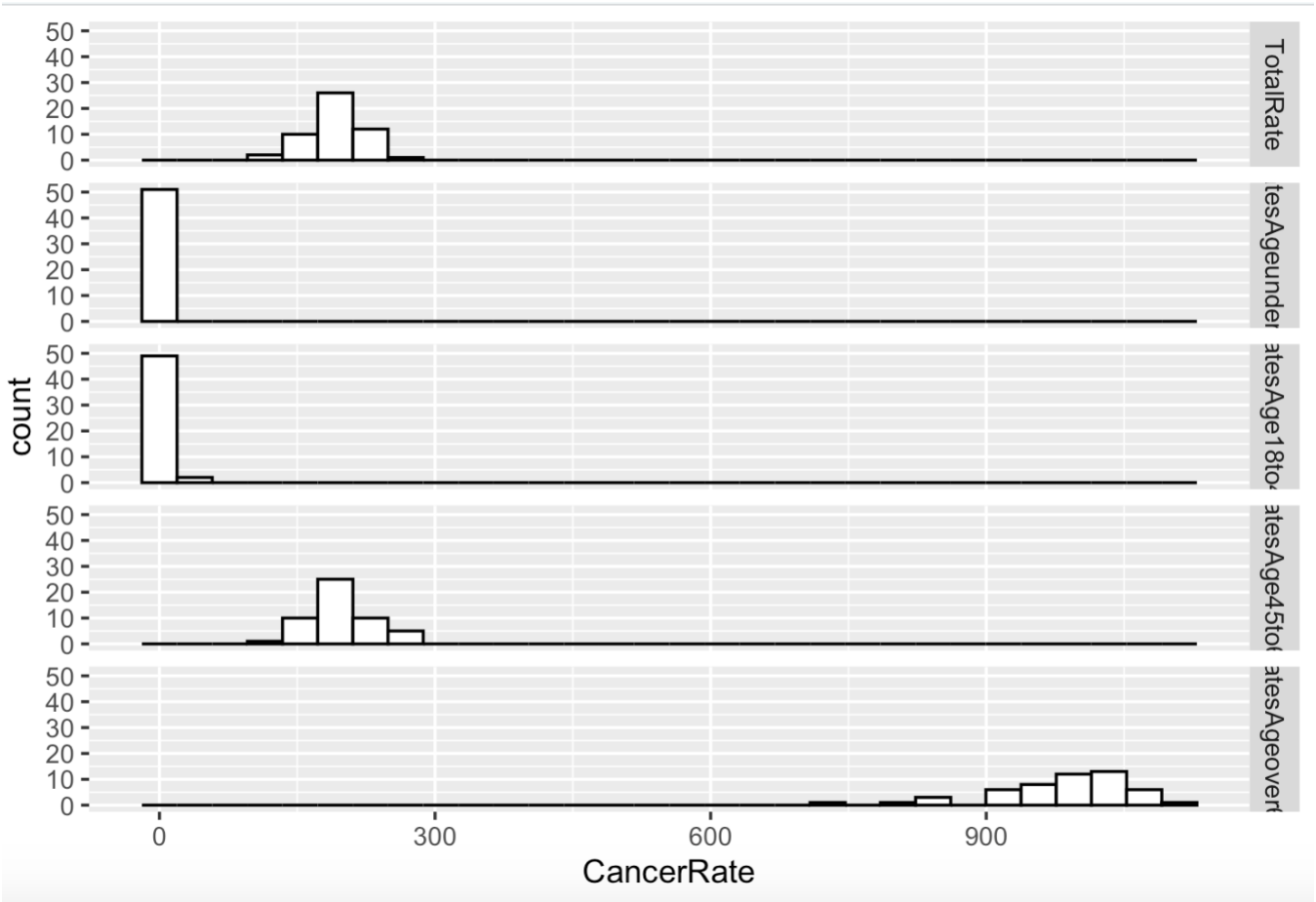
```
> ageGroup = c('TotalRate', 'RatesAgeunder18', 'RatesAge18to45','RatesAge45to64','RatesAgeover64')
#Change default levels of a factor, which is in alphabetical order.
> cancerr.age.longformat$AgeGroup = factor(cancerr.age.longformat$AgeGroup, levels = ageGroup)

> ggplot (cancerr.age.longformat, aes(x=AgeGroup, y=CancerRate, col=AgeGroup)) + geom_boxplot() + labs(x="", y="
      Total Cancer Death") + theme(axis.text.x =element_text(face="bold", size=7, angle=45, hjust=1))
```



facet plot: facet_grid()

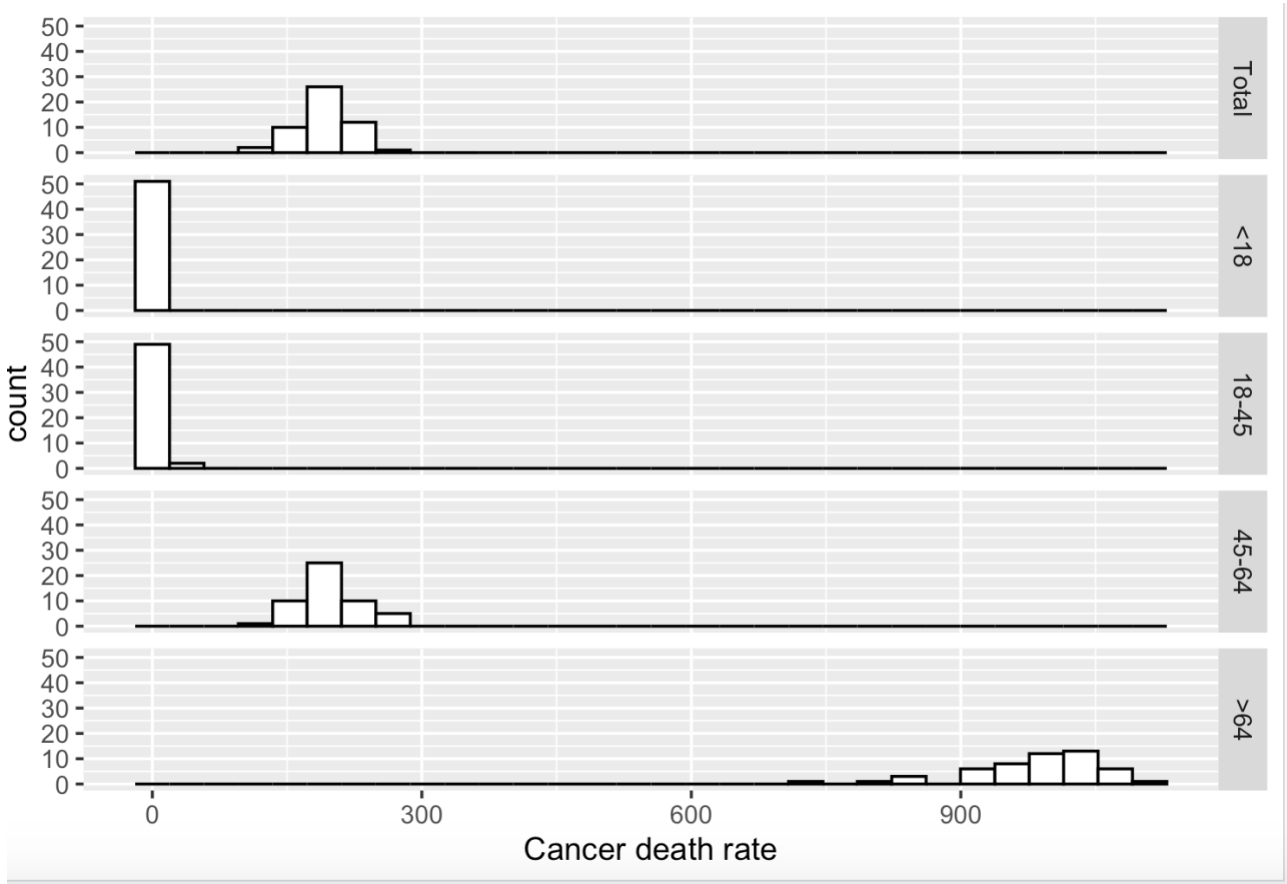
The facet approach partitions a plot into a matrix of panels

```
ggplot (cancerr.age.longformat, aes(x=CancerRate)) + geom_histogram(fill="white", colour="
    black") + facet_grid(rows=vars(AgeGroup))
```



Improved facet plot

```
> ageGroup = c('TotalRate', 'RatesAgeunder18', 'RatesAge18to45','RatesAge45to64','RatesAgeover64')
> age.lab = c("Total", "<18", "18-45","45-64", ">64")
> names(age.lab) = ageGroup

ggplot (cancerr.age.longformat, aes(x=CancerRate)) + geom_histogram(fill="white", colour="black") +
facet_grid(rows=vars(AgeGroup), labeller = labeller(AgeGroup=age.lab)) + labs(x="Cancer death rate")
```



Question 3

Question 3: Compare the two states that have the largest cancer death rate with the two states have the lowest cancer death rate.

Extact the required dataset

```
# Obtain index of the sorted column TotalRate
> index.order = order(cancer.data$TotalRate, decreasing = TRUE)

>ageGroup = c('TotalRate', 'RatesAgeunder18', 'RatesAge18to45','RatesAge45to64','
      RatesAgeover64')
> max.death = cancer.data[index.order[1:2], c('State', ageGroup)]
> min.death = cancer.data[index.order[(length(index.order)-1):length(index.order)], c('State'
      , ageGroup)]

# merge into a dataframe
> max.min.death = rbind(max.death,min.death )

# Covert to long format data
> max.min.death.long=pivot_longer(max.min.death, cols=ageGroup, names_to ='AgeGroup',
      values_to = 'CancerRate')
```

Long format data

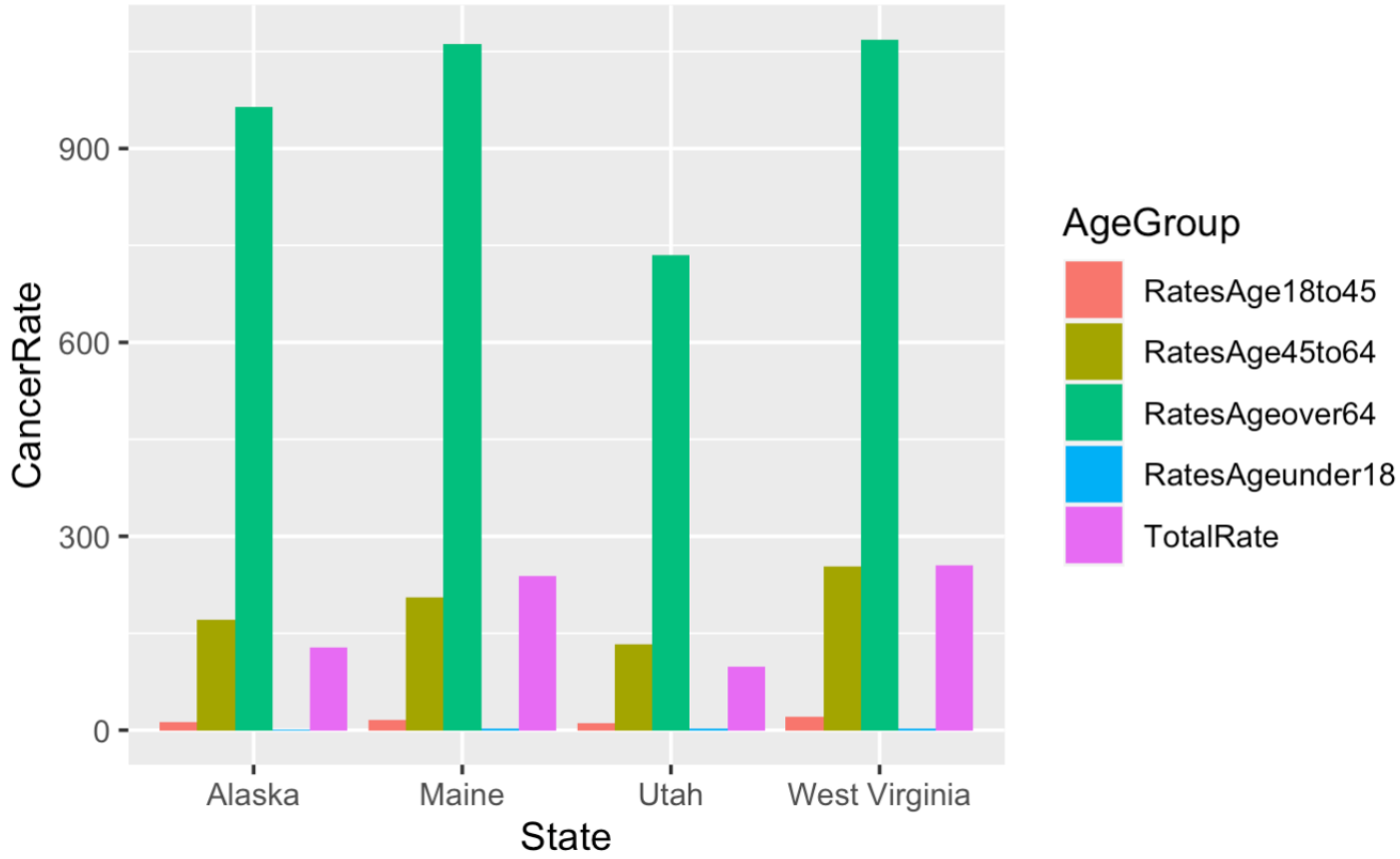
```
> str(max.min.death)
'data.frame':   4 obs. of  6 variables:
 $ State          : Factor w/ 51 levels "Alabama","Alaska",..: 49 20 2 45
 $ TotalRate      : num  254.6 239.1 128.1 98.5
 $ RatesAgeunder18: num  2.5 2.4 1.7 1.9
 $ RatesAge18to45 : num  20.3 15.2 11.8 10.9
 $ RatesAge45to64 : num  254 205 171 132
 $ RatesAgeover64 : num  1069 1062 965 736
>
> str(max.min.death.long)
tibble [20 × 3] (S3: tbl_df/tbl/data.frame)
 $ State     : Factor w/ 51 levels "Alabama","Alaska",..: 49 49 49 49 49 20 20 20 20 20 ...
 $ AgeGroup  : chr [1:20] "TotalRate" "RatesAgeunder18" "RatesAge18to45" "RatesAge45to64" ...
 $ CancerRate: num [1:20] 254.6 2.5 20.3 254 1068.9 ...
```

Bar plot: geom_bar

- By default, **geom_bar()** uses **stat="count"** which makes the height of the bar proportion to the number of cases in each group.
- If you want the heights of the bars to represent values in the data, use **stat="identity"** and map a variable to the y aesthetic.
- The default bar is a stacked plot. For side-by-side bar chart, use **position="dodge"**

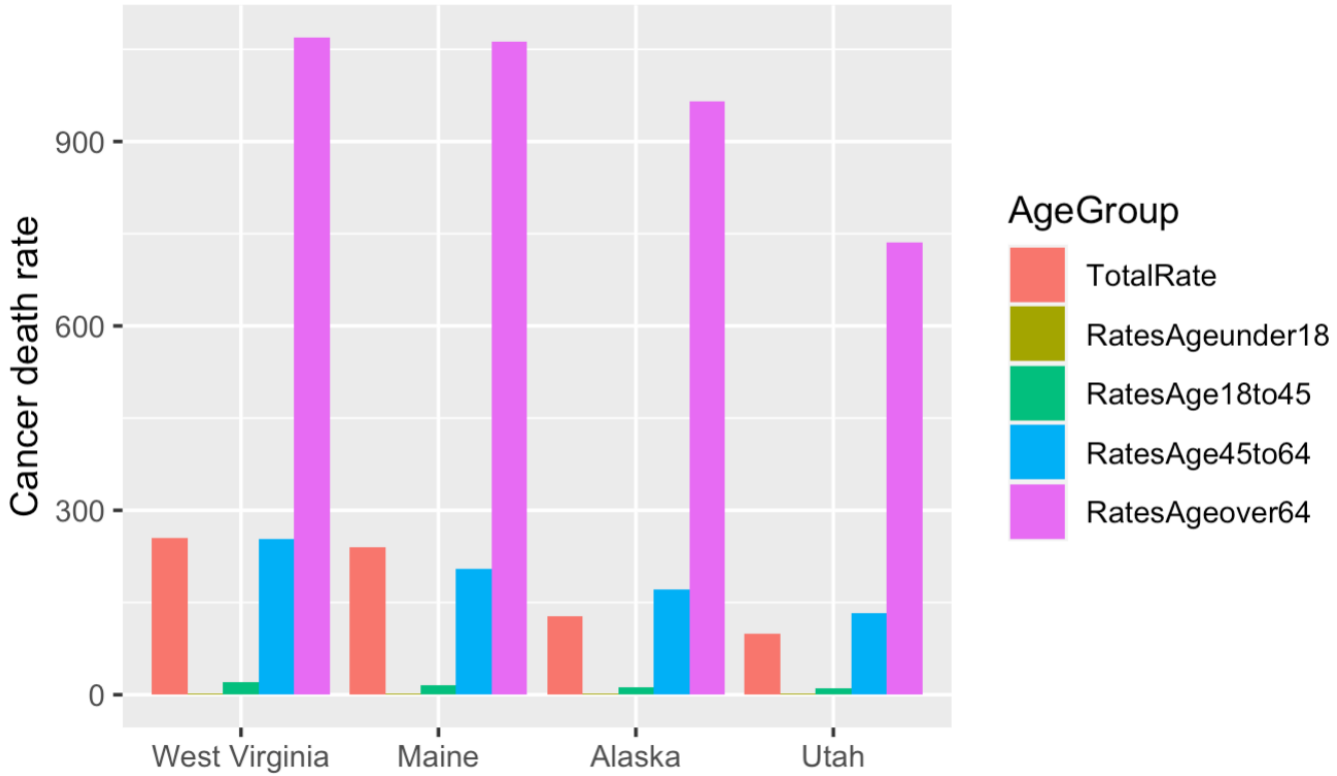
Bar plot

```
ggplot(max.min.death.long, aes(x=State, y=CancerRate, fill=AgeGroup)) + geom_bar(stat="
    identity", position="dodge")
```



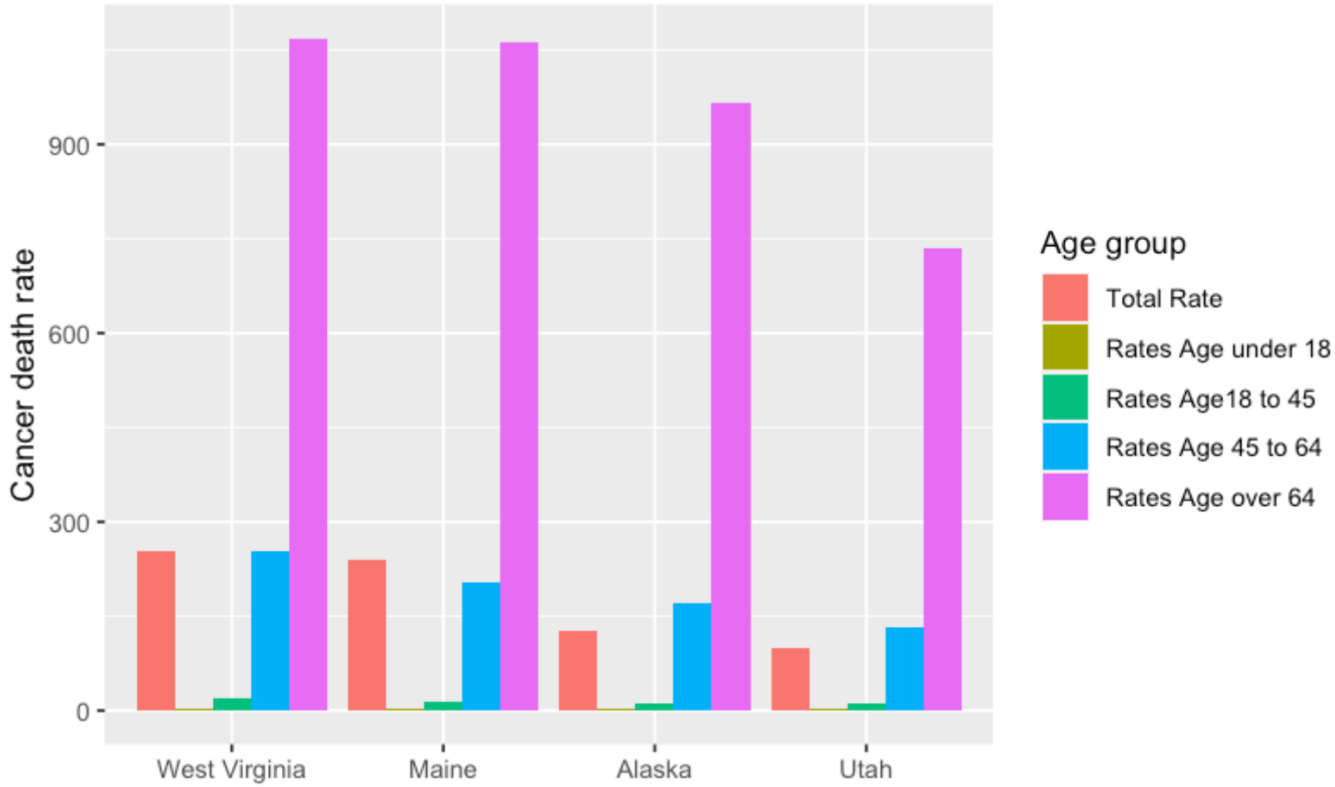
Improved bar plot

```
> max.min.death.long$AgeGroup = factor(max.min.death.long$AgeGroup, levels = ageGroup)
> max.min.death.long$State=factor(max.min.death.long$State, levels=c("West Virginia","
      Maine", "Alaska", "Utah"))

> ggplot(max.min.death.long, aes(x=State, y=CancerRate, fill=AgeGroup)) +
geom_bar(stat="identity", position="dodge") + labs(x="", y="Cancer death rate")
```



Modify legend

```
> legend.label = c('Total Rate', 'Rates Age under 18', 'Rates Age18 to 45','Rates Age 45 to
    64','Rates Age over 64')

> ggplot(max.min.death.long, aes(x=State, y=CancerRate, fill=AgeGroup)) + geom_bar(stat
    ="identity", position="dodge") + labs(x="", y="Cancer death rate") + scale_fill_
    discrete(name ="Age group", labels=legend.label)
```



Legend

- When a variable is mapped to fill, the default scale used is scale_fill_discrete() , which maps the factor levels to colors that are equally spaced around the color wheel.
- There are other scales for fill, such as scale_fill_manual()
- If you use scales for other aesthetics, such as colour (for lines and points) or shape (for points), you must use the appropriate scale.

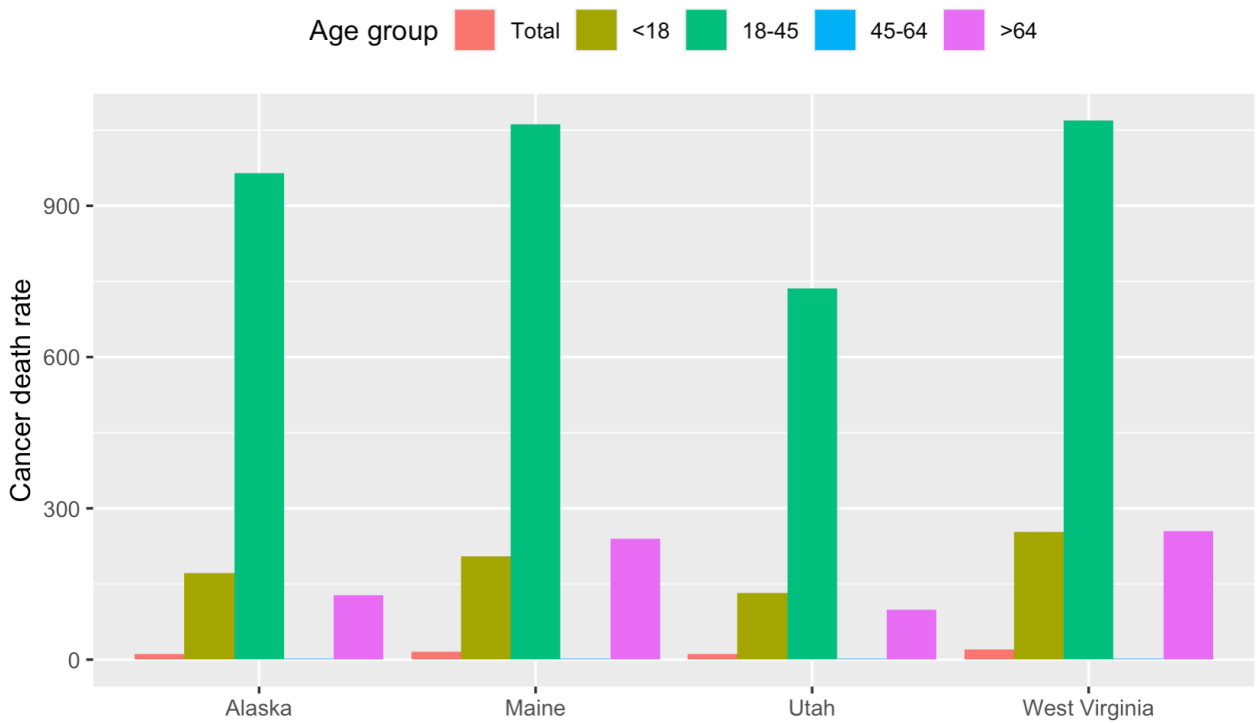
Commonly used scales

- scale_fill_discrete()
- scale_fill_hue()
- scale_fill_manual()
- scale_fill_grey()
- scale_fill_brewer()
- scale_colour_discrete()
- scale_colour_hue()
- scale_colour_manual()
- scale_colour_grey()
- scale_colour_brewer()
- scale_shape_manual()
- scale_linetype()

The position of a legend

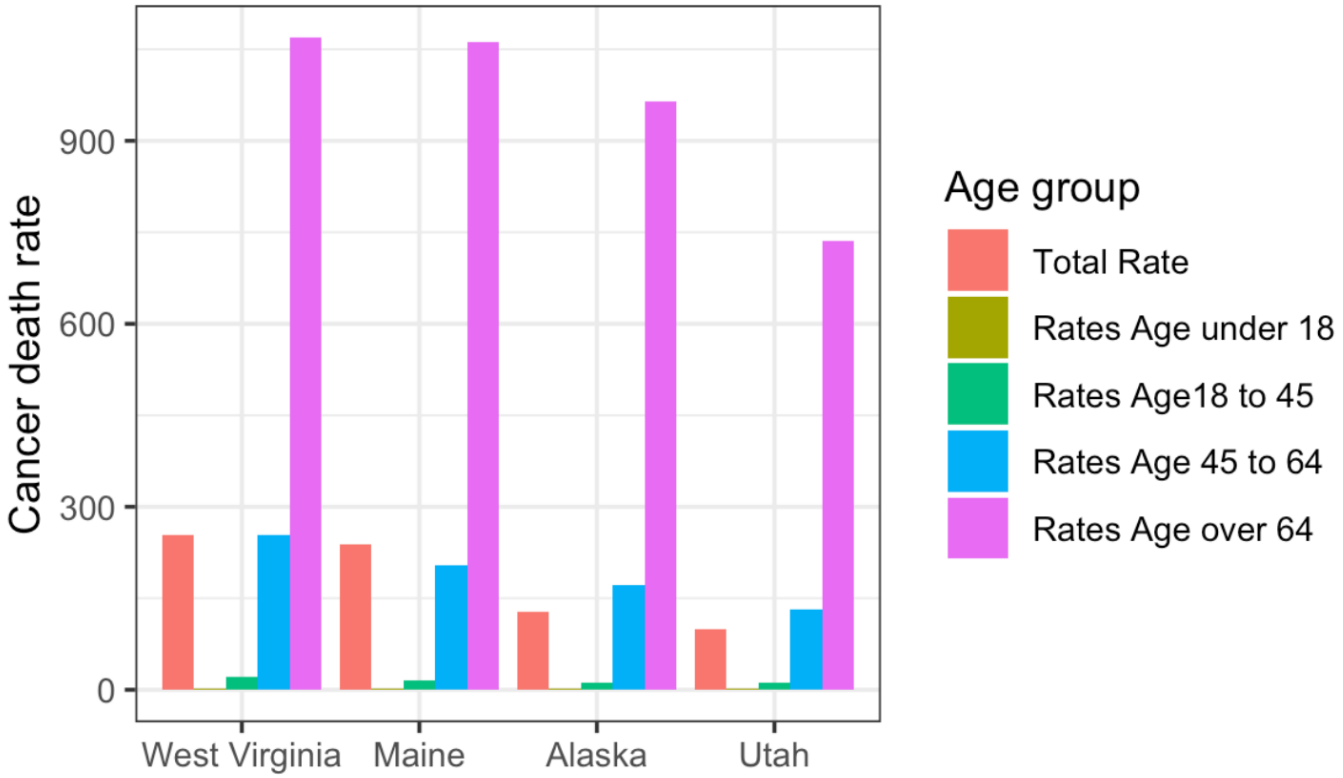
- The legend can also be placed inside the graphing area by specifying a coordinate position, using **legend.position** or use key word such as "top"

```
> legend.label = c("Total", "<18", "18−45","45−64", ">64")
>ggplot(max.min.death.long, aes(x=State, y=CancerRate, fill=AgeGroup)) + geom_bar(stat=
      "identity", position="dodge") + labs(x="", y="Cancer death rate") +
scale_fill_discrete(name ="Age group", labels=legend.label) +
theme(legend.position="top")
```



Change plot background

```
ggplot(max.min.death.long, aes(x=State, y=CancerRate, fill=AgeGroup)) + geom_bar(stat="
      identity", position="dodge") + labs(x="", y="Cancer death rate") +
  scale_fill_discrete(name ="Age group", labels=legend.label) + theme_bw()
```



ggplot in python

- **plotnine** is a Python package allowing you to use ggplot2 like code that is implementing the grammar of graphics
- Installation: Before getting started, you have to install plotnine. There are two main options for doing so: pip and conda.

```
#Using pip
pip install plotnine

# Using conda
onda install -c conda-forge plotnine
```

plotnine for ggplot

Work on google colab

log in to your google account, then go to this link
https://colab.research.google.com

Slides and code download