Blog Post 1: Animation and Ridges

Nishali

October 21, 2017

Motivation: I was really interested in learning how to utilize my newly acquired data visualization skills to create frame by frame animations. Having previously used Photoshop to create similar animations, I was interested in learning how I could tell similar stories using scatterplots. This motivated me to utilize `ggplot` to create aesthetic GIFs that incorporate scatterplots into every frame. As a statistics major I was also really interested in analyzing the distributions of various data sets faceted by one variable. This motivated me to to find ggridges, a ggplot extension that allows me to overlap density plots. In addition to utilizing these new packages I also wanted to practice complex file structure, which is why my post01 folder has multiple files within it.

> **Purpose:**
>
> - To gain a better understanding of the features of ggplot and the useful nature of isolating/faceting by certain variables to notice trends.
> - To practice complex file structure.
> - To learn how to data wrangle with several NAs.

---

ggplot Gifs and Density Plots Ridges

(This tuturial assumes a basic understanding of existing ggplot functionality and is for more advanced students)

Introduction

As someone who really enjoys concise visualized data and GIFs, I decided to take a deeper look at what ggplot can do in terms of creating more visually stimulating data visuals.This post will look at two aspects of ggplot extensions:

Content

1. Basics of scatterplots **using ggplot and utilizing magick to create graph animations**
    - *(NOTE: I initially tried to use the gganimate function but R 3.3.3 does not support the fftw and ffmpeg animation function of ImageMagick in chunks so I used an alternative method)*
2. **ggridges of density plots,** that display the densities of the x variable "faceted" by the y variable simultaneouly on one plot.

Importing Required Packages and Importing data

[You must first install the package in the r console, before you can call the package within your r script.The packages to install appear in the form below]

Wrangling `install.packages("readr")` `install.packages("dplyr")`

Graphing `install.packages("ggplot2")`

Animation `install.packages("magick")` `install.packages("knitr")`

Ridges `install.packages("ggridges")`

(NOTE:Look at the r script file to look at the data wrangling performed and some basic ggplot graphs created.)

Link to Data Wrangling r Script, Summary of data, ggplot graphs

https://youtu.be/Y6OSR0og2EY

Downloading the Data and Packages

```
library(readr)    # importing data
library(ggplot2)  # graphics
library(knitr)    # animation
library(magick)   # animation
```

```
## Linking to ImageMagick 6.9.6.6
## Enabled features: cairo, fontconfig, freetype, pango, rsvg, webp
## Disabled features: fftw, ghostscript, lcms, x11
```

```
library(reshape2) # animation
library(ggridges) # ridges
dat <- read.csv('~/stat133/stat133-hws-fall17/post01/data/college-act-sat-bystate.csv')
```

---

ggplot Animations

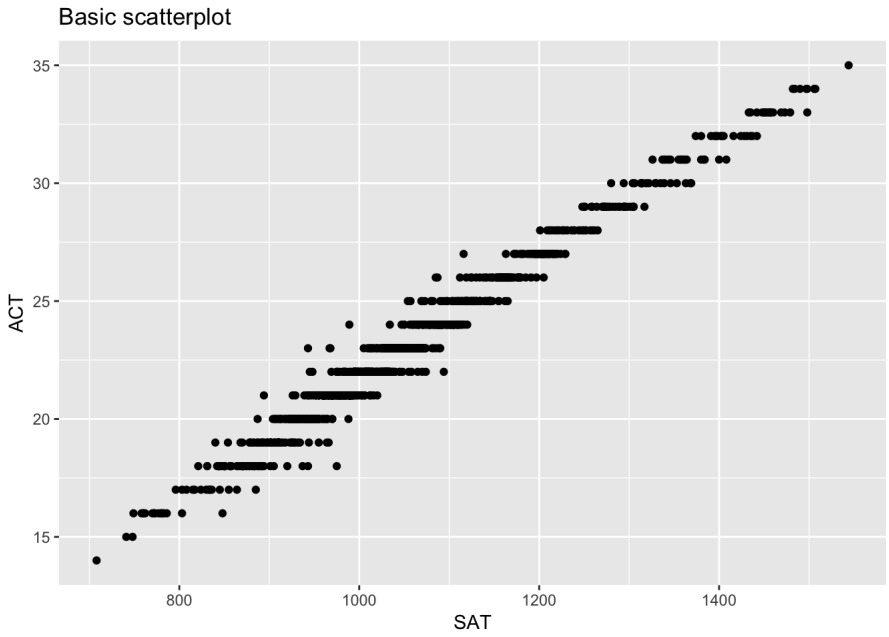
(All gifs below were created using a combination of the packages `magick` and `ggplot`, as well as some functions I created myself to merge all the layers of the gif) **Please click on animation folder on github to view the gifs created below.**

Creating Basic Scatter Plots Using ggplot

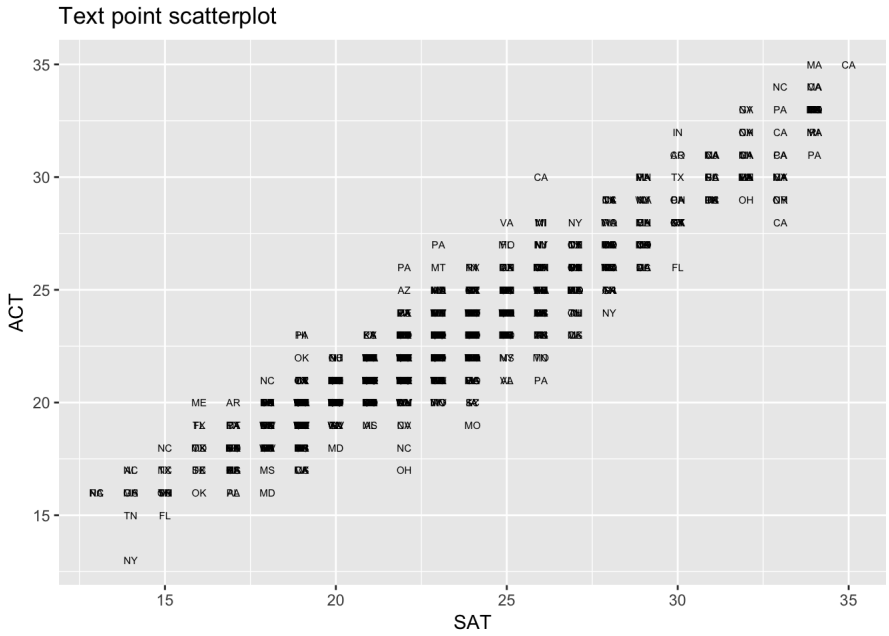
The function `ggplot` is used to create simple but aesthetic graphs, the function `geom_point` helps identify the plot as a scatterplot.The code below is a quick recap on the previous knowledge you have gained from your practice with ggplot.

[REMEMBER: this tutorial is for people with a basic understanding of ggplot and rstudio functionality]

```
#Basic scatterplot
ggplot(dat, aes(x=SAT,y=dat$ACTCM50)) +
  geom_point() + labs(x="SAT", y="ACT", title="Basic scatterplot")
```

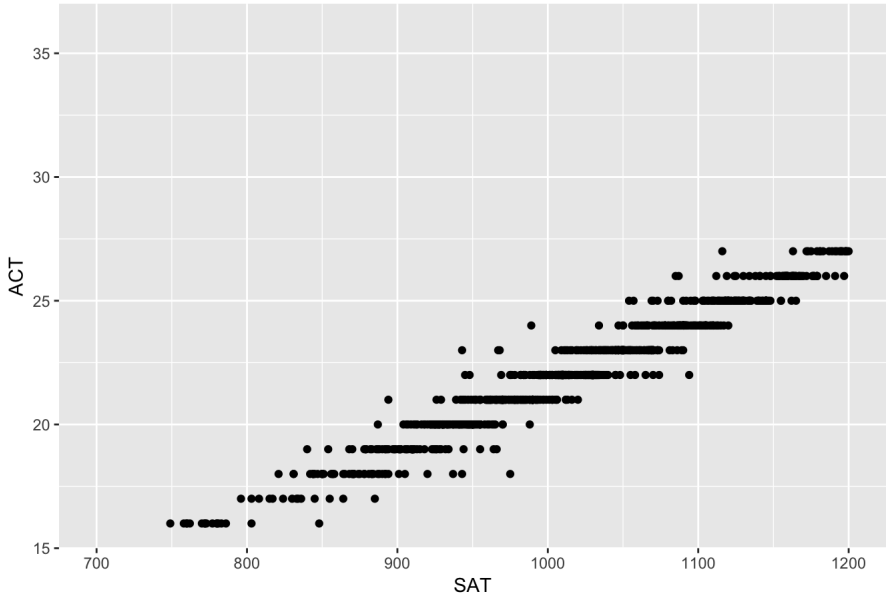



```
#Changing the points to represent variable names
 ggplot(dat, aes(x=ACTEN50,y=dat$ACTMT50)) +
  geom_text(aes(label=dat$State), size=2)+
   labs(x="SAT", y="ACT", title="Text point scatterplot")
```



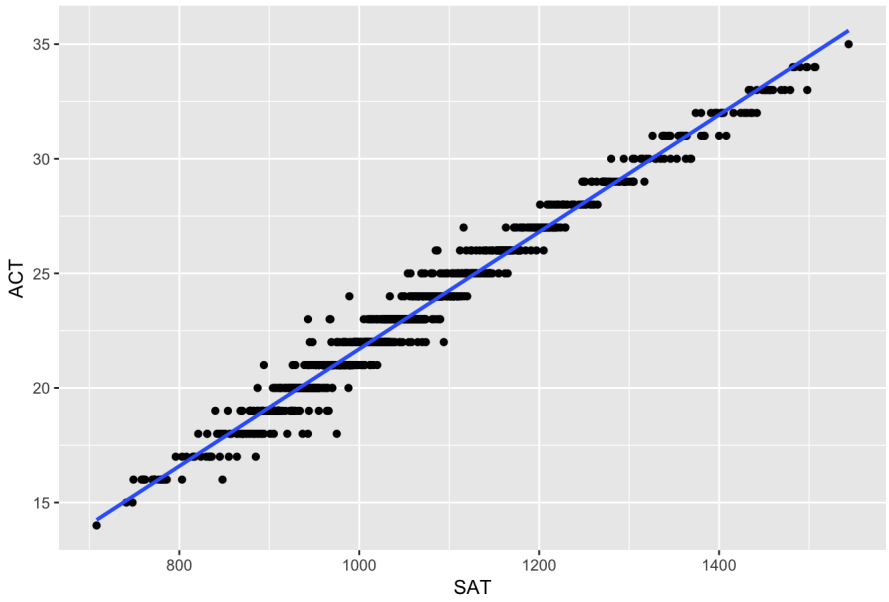

```
#Changing the range of values contained on the axis
ggplot(dat, aes(x=SAT,y=dat$ACTCM50)) +
  geom_point()+ xlim(700,1200) + ylim(16, 36)+
  labs(x="SAT", y="ACT", title="Axis change scatterplot")
```

Axis change scatterplot

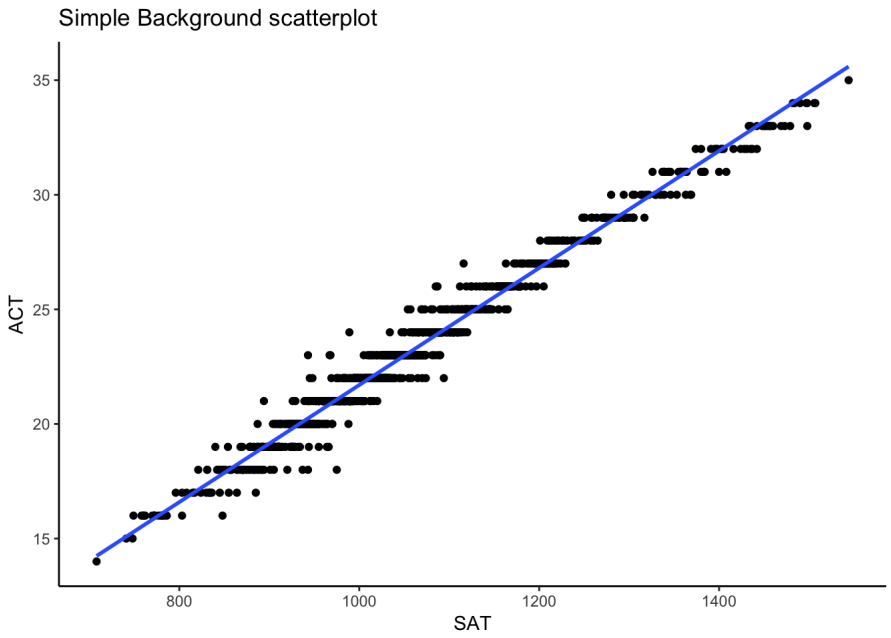



```
#Adding a regression line
 ggplot(dat, aes(x=SAT,y=dat$ACTCM50)) +
  geom_point()+ geom_smooth (method=lm, se=FALSE)+
  labs(x="SAT", y="ACT", title="Regression line scatterplot")
```

Regression line scatterplot




```
#Changing background theme to simpler ggplot
 ggplot(dat, aes(x=SAT,y=dat$ACTCM50)) +
  geom_point()+ geom_smooth (method=lm, se=FALSE)+
   theme_classic()+
  labs(x="SAT", y="ACT", title="Simple Background scatterplot")
```



Creating basic Gifs using one variable as the frame changer

- The functions below were used in the preceding chunk:
  - `image_graph()` to create the individual frames of the gif as a Magick image instead of a jpeg.
  - `split()` to seperate the factor that comprises the change of each frame from the data
  - `lapply()` to return result of function and corresponding element x
  - `function(x){}` to create the function that prints each frame.
  - `dev.off()` to stop runnning the function
  - `image_background` to determine the background image and color.
  - `image_animate()` to manipulate the frames and prepare for animation.
  - `image_write` to combine the images into the output of a gif to be saved in an external folder.

1st Example: Scatterplot Gif based on State

While most people tend to frame their animations based on time, because there is no time variable in my chosen data set, I decided to create each frame (or facet) based on the State. This can be used to determine which states had the Universities with the best SAT and ACT Scores. This graph looks at SAT and ACT scores for specfific states.

Plotting collected ACT to SAT scores by State

```
img <- image_graph(600, 400, res = 96)
datalist <- split(dat, dat$State)
out <- lapply(datalist, function(data){
  p <- ggplot(data, aes(x=SAT,y=ACTCM50))+
    geom_point()+ geom_smooth (method=lm,se=FALSE, color="blue") +xlim(700,1200) + ylim(16, 36) + ggtitle(data$Sta
te) + theme_classic()
  print(p)
})
dev.off()
```

```
## quartz_off_screen
##                 2
```

```
img <- image_background(image_trim(img), 'white')
animation <- image_animate(img, fps = 2)
image_write(animation, "~/stat133/stat133-hws-fall17/post01/extensions/animation/SAT_vs_ACT_byState.gif")
```

2nd Example: Scatterplot Gif based on Median ACT score

This time I decided to change the frame by the median ACT score and looked at the corresponding ACT Math and English subscores for that particular cumulative score. I also decided to alter the aesthetic of the data point by having the color depend on the state the university ACT score was collected from.This graph looks at ACT English and math sub scores as a function of the cumulative ACT Score.

Scatterplot Gif based on Median ACT

```
img2 <- image_graph(800, 400, res = 96)
datalist2 <- split(dat, dat$ACTCM50)
out2 <- lapply(datalist2, function(data2){
  p <- ggplot(data2, aes(x=ACTEN50,y=ACTMT50, color= data2$State))+
    geom_point() +
    ggtitle(data2$ACTCM50)+
    xlim(12,36) +
    ylim(12, 36)+
    theme_classic()
  print(p)
})
dev.off()
```

```
## quartz_off_screen
##                 2
```

```
img2 <- image_background(image_trim(img2), 'white')
animation2 <- image_animate(img2, fps = 2)
image_write(animation2, "~/stat133/stat133-hws-fall17/post01/extensions/animation/ACTenglish_vs_math_bymedianACT.gif")
```

3. Customization: Creating more complex Gifs

- This section looks at two types of complex gifs
  - Faceting
  - Image background

1. Creating a gif using faceting

Scatterplot Gif based on State

This graph requires that you split the data so that the variable within the facet is not changing but the data inside is. Because my data set contains missing values the number of graphs facted also change with each frame, the range of the facet is constantly changing.This particular graph looks at ACT Math and English subscores and how the points change when grouped by cumulative ACT.

```
#Specifying image criteria
img4 <- image_graph(800, 400, res = 96)
#function that create frames
datalist4 <- split(dat,dat$ACTCM50)
out4 <- lapply(datalist4, function(data4){
  p <- ggplot(data4, aes(x=ACTEN50,y=ACTMT50)) +
    geom_point()+ xlim(12,36) + ylim(12, 36)+
    facet_wrap(~ACTEN50, ncol=2, nrow=4) +
    labs(x="ACT English", y= "ACT Math", title="English to Math Scores")
  print(p)
})
dev.off()
```

```
## quartz_off_screen
##                 2
```

```
img4 <- image_background(image_trim(img4), 'white')
animation5 <- image_animate(img4, fps = 2)
#creating and saving finalised gif
image_write(animation5, "~/stat133/stat133-hws-fall17/post01/extensions/animation/ACTenglish_vs_math_facet.gif")
```

2. Gif over a graph background

- The functions below *(in addition to the ones mentioned above)* were used in the preceding chunk:
  - `theme()` specify criteria within the ggplot function to ensure that the graph is transparent.
    - Examples: `panel.background`, `panel.grid.minor`, `panel.grid.major`, etc.,
  - `image_apply()` similar to lapply, it returns a function and the element x.
  - `image_composite()` combines the specified elements into one frame.

The Process

Firstly you should create a foreground gif that is comprised of the animated portion with a transparent background. Then using the `image_read` function to break it down into frames, recombine these frames with the background image of choice. Because I had to create my own background image I created the ggplot using the `image_graph` function to create a magick object then used the `image_background` function to make it the background image. I also made this graph transparent to avoid distractions.This style gif highlights certain sets of data by having those layers of the plot animated and then a single background frame hidden behind it that shows all the data.

Step 1: ACT showing median ACT english and math subscores as a transparent gif

```
#Creating the foreground gif that has a Transparent Background
img3 <- image_graph(600, 400, res = 96,bg = "transparent")
datalist3 <- split(dat, dat$State)
out3 <- lapply(datalist3, function(data3){
  p <- ggplot(data3, aes(x=ACTEN50,y=ACTMT50)) +
    geom_point(aes(colour=data3$State)) +
    xlim(12,36) + ylim(12,36)+ theme_classic()+
    labs(x="ACT English", y= "ACT Math",
         title="English to Math Scores by State")
  p <- p + theme(
    panel.background = element_rect(fill = "transparent",colour = NA),
    panel.grid.minor = element_line(colour = NA),
    panel.grid.major = element_line(colour = NA),
    plot.background = element_rect(fill = "transparent",colour = NA),
    legend.background = element_rect(fill = "transparent",colour = NA),
    legend.position= c(1,0),
    legend.title= element_text(colour = NULL)
  )
  print(p)
})
dev.off()
```

```
## quartz_off_screen
##                 2
```

```
animation3 <- image_animate(img3, fps = 2)

#Saving the Image
image_write(animation3, "~/stat133/stat133-hws-fall17/post01/extensions/animation/foreground.gif")
```

In order to have the background image be shown it is very important that you specify `bg = "transparent"` in `image_graph`, and include all the ggplot specifications within the `theme()` function of the graph.

Step 2: ACT showing median ACT english and math subscores based on State overlayed over the entire data set

```
#Creating the graoh for the background image
fig <- image_graph(width = 600, height = 400, res = 96, bg="transparent")
ggplot2::ggplot(dat, aes(x=ACTEN50,y=ACTMT50))  +
    xlim(12,36) + ylim(12, 36) + geom_point(alpha = .7) + theme_classic()+
  labs(x="ACT English", y= "ACT Math", title="English to Math Scores")
dev.off()
```

```
## quartz_off_screen
##                 2
```

```
#Creating the frames of the Transparent gif
animation4 <- image_read("~/stat133/stat133-hws-fall17/post01/extensions/animation/foreground.gif")
image_info(animation4)
```

```
##    format width height colorspace filesize
## 1     GIF   600    400       sRGB   159122
## 2     GIF   600    400       sRGB   159122
## 3     GIF   600    400       sRGB   159122
## 4     GIF   600    400       sRGB   159122
## 5     GIF   600    400       sRGB   159122
## 6     GIF   600    400       sRGB   159122
## 7     GIF   600    400       sRGB   159122
## 8     GIF   600    400       sRGB   159122
## 9     GIF   600    400       sRGB   159122
## 10    GIF   600    400       sRGB   159122
## 11    GIF   600    400       sRGB   159122
## 12    GIF   600    400       sRGB   159122
## 13    GIF   600    400       sRGB   159122
## 14    GIF   600    400       sRGB   159122
## 15    GIF   600    400       sRGB   159122
## 16    GIF   600    400       sRGB   159122
## 17    GIF   600    400       sRGB   159122
## 18    GIF   600    400       sRGB   159122
## 19    GIF   600    400       sRGB   159122
## 20    GIF   600    400       sRGB   159122
## 21    GIF   600    400       sRGB   159122
## 22    GIF   600    400       sRGB   159122
## 23    GIF   600    400       sRGB   159122
## 24    GIF   600    400       sRGB   159122
## 25    GIF   600    400       sRGB   159122
## 26    GIF   600    400       sRGB   159122
## 27    GIF   600    400       sRGB   159122
## 28    GIF   600    400       sRGB   159122
## 29    GIF   600    400       sRGB   159122
## 30    GIF   600    400       sRGB   159122
## 31    GIF   600    400       sRGB   159122
## 32    GIF   600    400       sRGB   159122
## 33    GIF   600    400       sRGB   159122
## 34    GIF   600    400       sRGB   159122
## 35    GIF   600    400       sRGB   159122
## 36    GIF   600    400       sRGB   159122
## 37    GIF   600    400       sRGB   159122
## 38    GIF   600    400       sRGB   159122
## 39    GIF   600    400       sRGB   159122
## 40    GIF   600    400       sRGB   159122
## 41    GIF   600    400       sRGB   159122
## 42    GIF   600    400       sRGB   159122
## 43    GIF   600    400       sRGB   159122
## 44    GIF   600    400       sRGB   159122
## 45    GIF   600    400       sRGB   159122
## 46    GIF   600    400       sRGB   159122
## 47    GIF   600    400       sRGB   159122
## 48    GIF   600    400       sRGB   159122
## 49    GIF   600    400       sRGB   159122
## 50    GIF   600    400       sRGB   159122
## 51    GIF   600    400       sRGB   159122
```

```
# Background image as magick object
background <- image_background(fig, 'white')

# Combine and flatten frames
frames <- image_apply(animation4, function(frame) {
  image_composite(background, frame)
})

#Saving and animating the gif
animation4 <- image_animate(frames, fps = 2)
image_write(animation4, "~/stat133/stat133-hws-fall17/post01/extensions/animation/cumulative.gif")
```

This graph looks at how the ACT Math and English subscores grouped by State vary in comparison to the overall data set, hence the overlap of the gif over the original data set.

Video on how to acces GIFs on github repositiory

Because html will not allow for gif functionality I have attached a link to the gifs below, feel free to click it:

https://youtu.be/FZHqjuYsKqA

ggridges

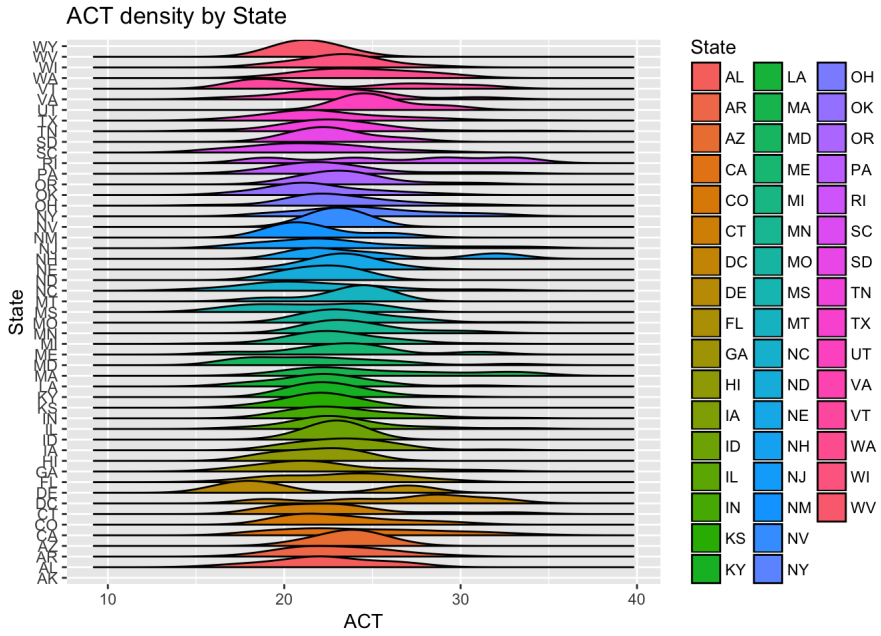
1. Basic Ridgeline Density plot

The `geom_density_ridges` can be used to plot the density with a bordered outline and filled center, while simultaenously “faceting” by the chosen y variable. While `geom_density_ridges` does not have a line underneath the end of the data density plot, `geom_density_ridges2` has this line that differentiates the start of the data from the remaining portion.

The graphs below look at the density distributions of scores based on State.

```
#geom_density_ridges2
f1 <- ggplot(dat, aes(x=ACTCM50,y=State))+
  geom_density_ridges2(aes(fill = State))+ labs(x="ACT", y="State", title= "ACT density by State")
f1
```

```
## Picking joint bandwidth of 1.6
```



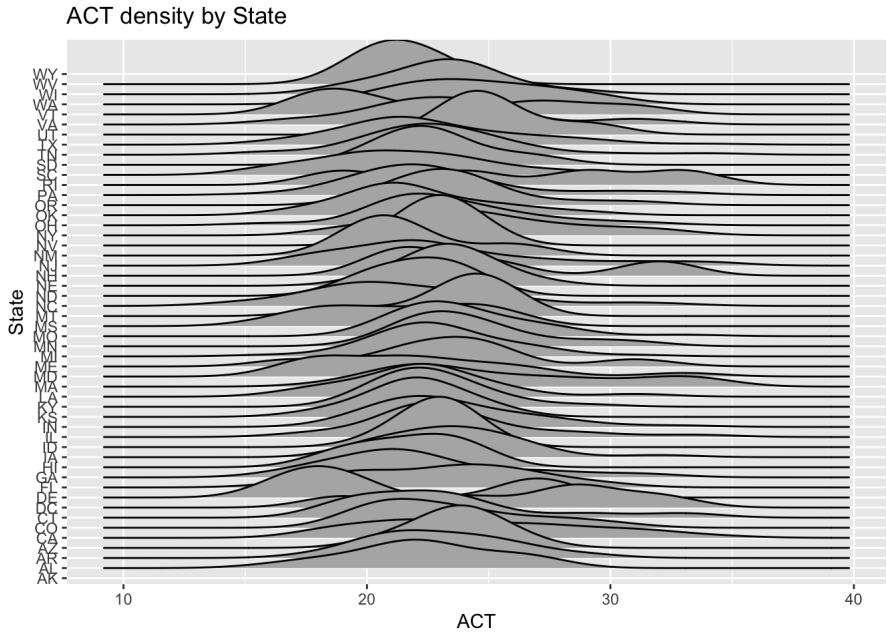

```
ggsave ("~/stat133/stat133-hws-fall17/post01/extensions/Ridges/geom_density_ridges2.png")
```

```
## Saving 7 x 5 in image
## Picking joint bandwidth of 1.6
```

The extent to which the different densities overlap can be controlled with the scale parameter. A setting of scale=1 means the tallest density curve just touches the baseline of the next higher one. Smaller scale values increases separation between the curves.

```
#Changing the amount of overlap amongst graphs
f2 <- ggplot(dat, aes(x=ACTCM50,y=State))+
  geom_density_ridges(scale =5)+
  labs(x="ACT", y="State", title= "ACT density by State")
f2
```

```
## Picking joint bandwidth of 1.6
```

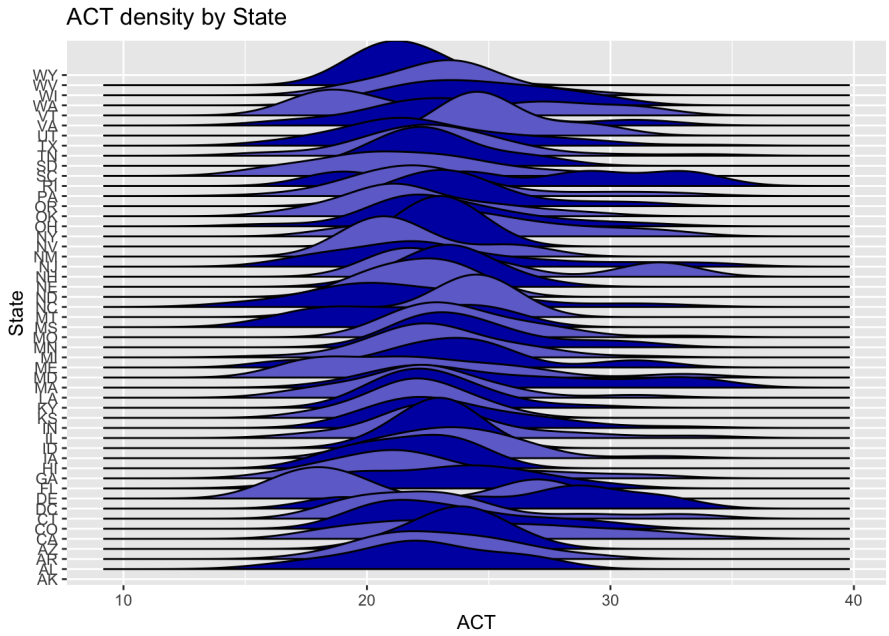



```
ggsave ("~/stat133/stat133-hws-fall17/post01/extensions/Ridges/geom_density_scale.png")
```

```
## Saving 7 x 5 in image
## Picking joint bandwidth of 1.6
```

```
#Adding color to the graph
f3 <- ggplot(dat, aes(x=ACTCM50,y=State)) +
  geom_density_ridges2(scale =5, aes(fill = State))+ scale_fill_cyclical(values = c("#0000B0", "#7070D0"))+
  labs(x="ACT", y="State", title= "ACT density by State")
f3
```

```
## Picking joint bandwidth of 1.6
```




```
ggsave ("~/stat133/stat133-hws-fall17/post01/extensions/Ridges/geom_density_altcolor.png")
```

```
## Saving 7 x 5 in image
## Picking joint bandwidth of 1.6
```

Conclusions

This is my first time using command-based software, analyzing data, and working with complex file projects which is why I have included all of these elements within this post. To summarize what I have discussed here ggplot is really useful on its own, but with the addition of extensions the basics can create cool new visuals. magick can be used to create gifs that rely on faceting on one variable as the source of each gif frame. You can use this concept to create more complex gifs such as double faceting by the facet_wrap function and the split function, or combining a still graph of the data with animated data. On the other hand, ggridges can be used to create cool graphs that compare distributions of data by a faceted variable.

References:

This file contains both information learnt in class and information I researched to better display the data I present to you. Below is a list of references I used:

1. **Data from:**
    - Data
    - US department of education
2. **Basics Recap:**
    - Scatterplot basics
    - ggplot Densities
    - Distributions
    - Transparent ggplot
    - Transparent background
    - transparent Legend
3. **Extensions from:**
    - animated ggplot
    - ggridges basics
4. **List of references:**
    - ggplot cheat sheet
    - dyplr cheat sheet