Data Sonification of Endangered Languages

Alex Lu

5/8/2022

Introduction

This semester, I had the honor of taking a linguistics class related to endangered languages, a stats class where we programmed in R, and a music class that connects music to data. Thus, with this final project I thought it would be cool to use something related to all three classes. From my linguistics class we learned that of the 7,000 endangered languages in existence today, around 3,000 of them are endangered. The data related to these 3,000 languages are used as the dataset for this project. From my stats class, I'll be using R in order to process my data and create my data visualizations. From this class, I plan on creating a data sonification that turns data into sounds through TwoTones.

1) Importing necessary packages to create data visualization.

For my project, I decided to use R in order to create my data visualizations. In order to do this I relied on the following packages in the code block below. The "tidyverse" and "maps" packages are necessary to create the scatterplots while "gganimate" is used to add animations to the generated scatterplot.

```
library(tidyverse)
library(gganimate)
require(maps)
library("av")
```

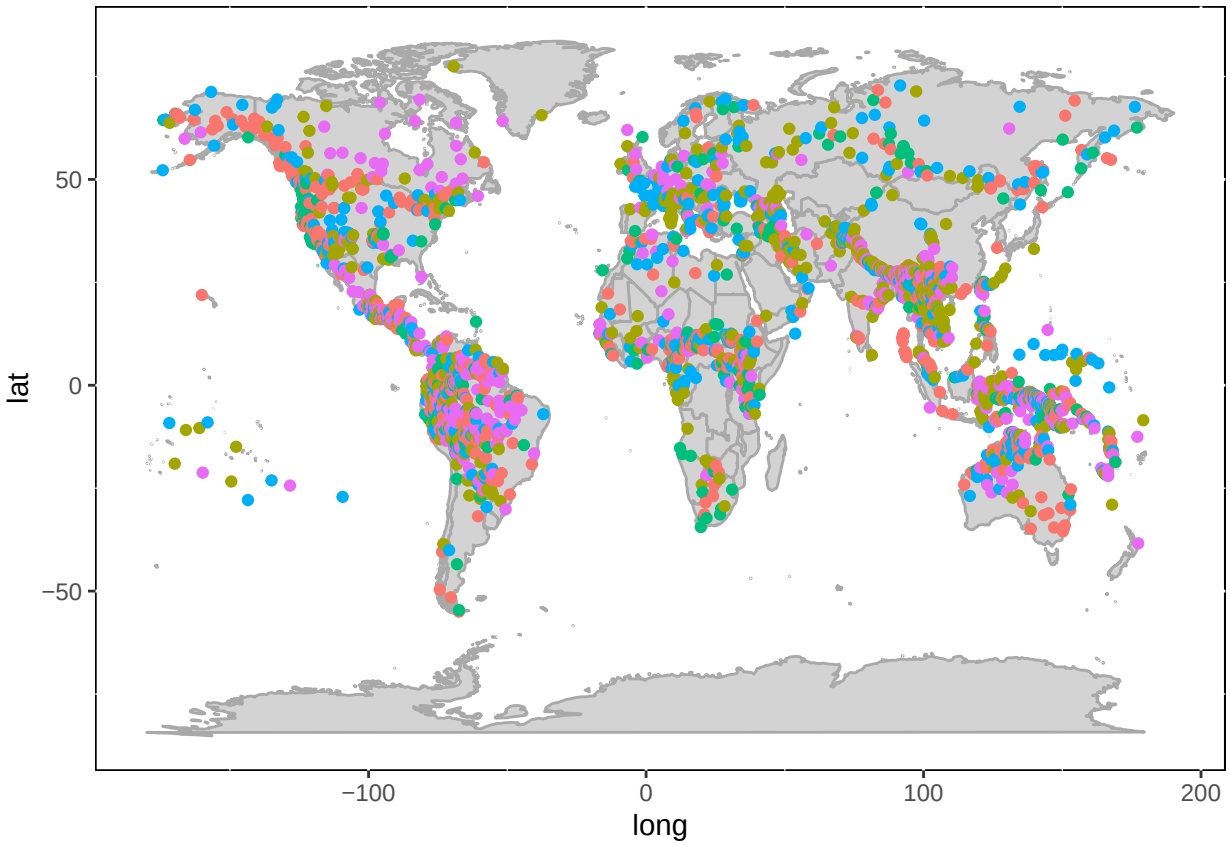
2) Import Endangered Languages data

The data used for this project was published by "The Guardian" and contains a list of endangered languages and different columns including the different names, ID number, country codes, latitude, longitude, etc. For the purposes of the project we only need the latitude,longitude, degree of endangerment, and name in English (although this isn't that important either). In the code below, all the extraneous columns are removed and the data is arranged in order of longitude.

```
data<- read.csv(file='data.csv')
sorted_data<-data %>%
  arrange(Longitude) %>%
  select(Name.in.English,Degree.of.endangerment,Latitude,Longitude)
```

If we were to plot all the data points now and color code each point by the degree of endangerment we'd get the following plot.

```
theme_set(
  theme(panel.background= element_rect(fill="white"),legend.position="none")
)
world_map <- map_data("world")
ggplot(world_map, aes(x = long, y = lat,group=group)) +
  geom_polygon(fill="lightgray", colour = "darkgray")+
  geom_point(data=sorted_data,aes(x=Longitude,y=Latitude,group=Degree.of.endangerment
                                  ,color=Degree.of.endangerment))
```



For this project, I decided it'd be better if the data were separated depending on the degree of endangerment. This makes the plots less clustered. Furthermore, given that TwoTones has a max bpm of 300, the final audio file would be super long which means the animation would take a couple of minutes which would take an extremely long time to render.

3) Creating CSVs for each Degree of Endangerment

In order to turn the data into sound, I need to create CSV files which will then be used in TwoTones to produce an audiofile with the sonified version of the data. For the purposes of this document, the "write.csv" commands have been commented out.

```
#Data for all "Critically Endangered" languages
critical_data<- sorted_data %>%
  filter(Degree.of.endangerment=="Critically endangered")
critical_data<-na.omit(critical_data)
#write.csv(critical_data,"critical_data.csv")
```

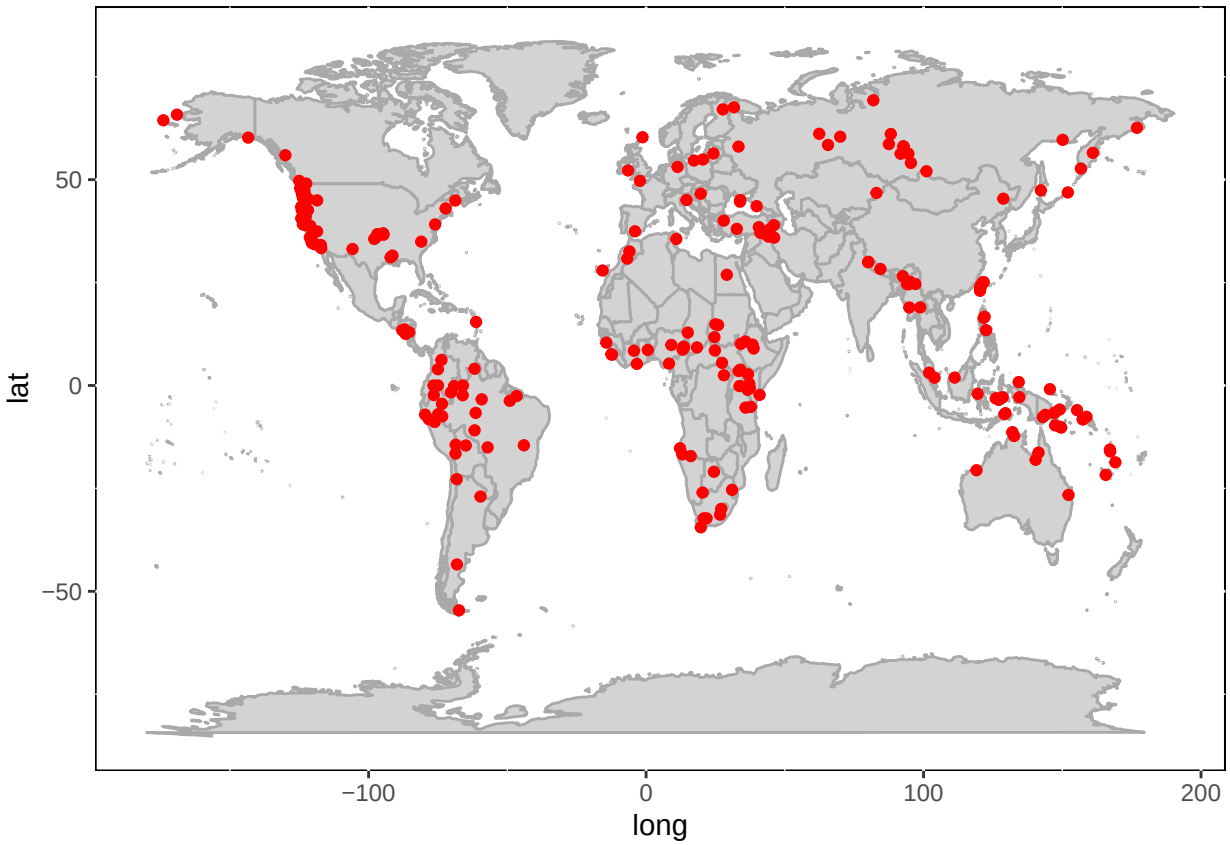
```r
#Data for all "Extinct" languages
extinct_data<- sorted_data %>%
  filter(Degree.of.endangerment=="Extinct")
extinct_data<-na.omit(extinct_data)
#write.csv(extinct_data,"extinct_data.csv")

#Data for all "Vulnerable" languages
vulnerable_data<- sorted_data %>%
  filter(Degree.of.endangerment=="Vulnerable")
vulnerable_data<-na.omit(vulnerable_data)
#write.csv(vulnerable_data,"vulnerable_data.csv")

#Data for all "Severely endangered" languages
severe_data<- sorted_data %>%
  filter(Degree.of.endangerment=="Severely endangered")
severe_data<-na.omit(severe_data)
#write.csv(severe_data,"severe_data.csv")
```

Using TwoTones is just a simple matter of dragging and dropping each csv file into the program and choosing which column to use as the datasource (Latitude). From then I tweaked the timing of the data to play at 300bpm. This an extremely fast tempo but it's necessary for me in order to ensure that the animations have a shorter duration which drastically decreases the amount of time it would take for my computer to render each animation.

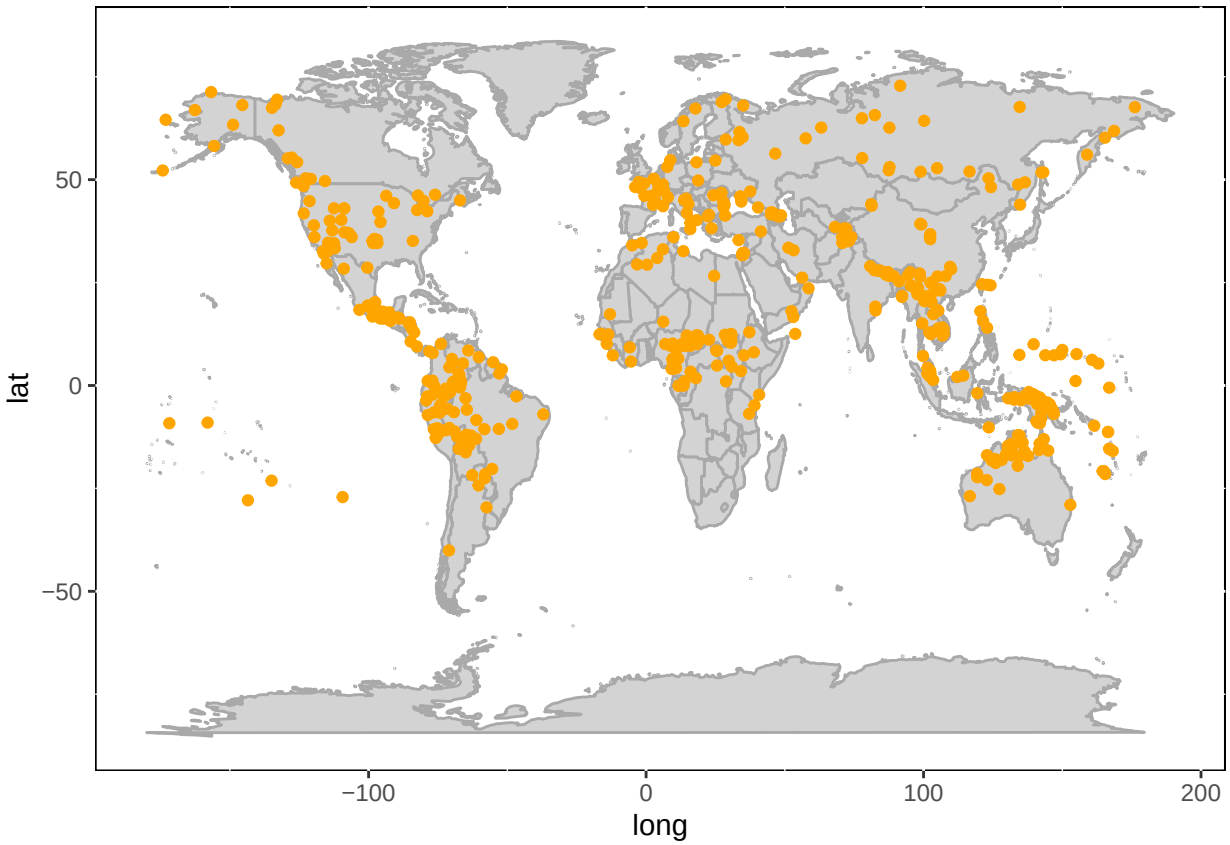
4) Creating animations

Now that all the data has been separated and processed, we can use "gganimate" to create animations for each scatterplot. Because animations can't be played in a pdf file, the following is what the scatter plots look like

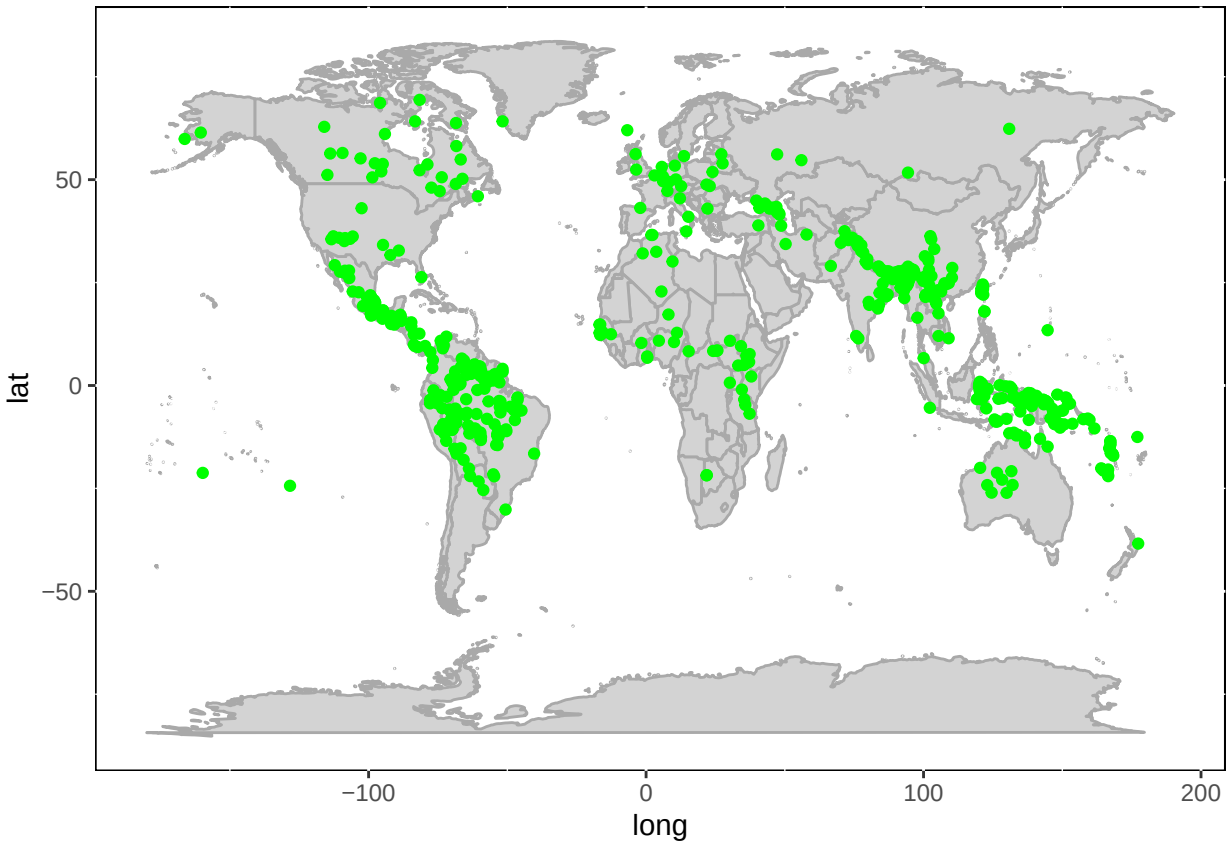
1) Extinct Plot



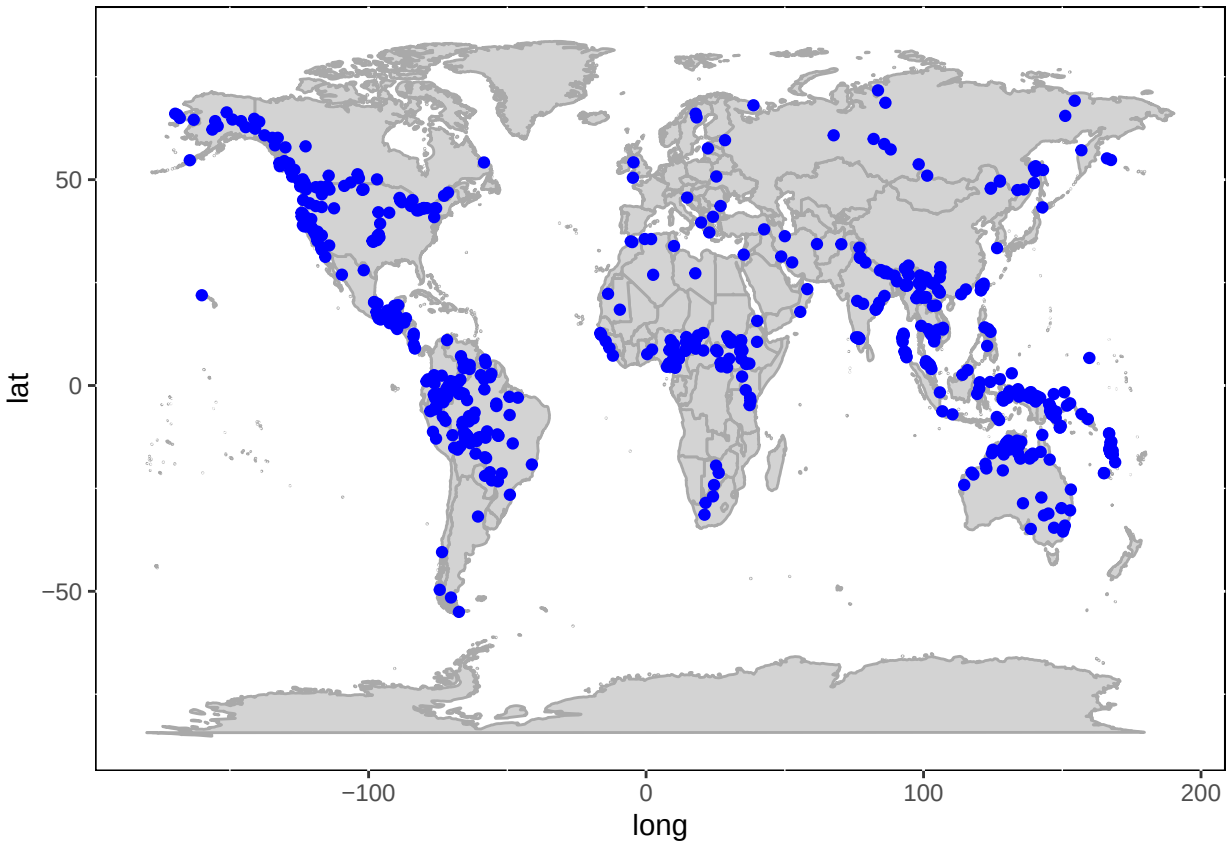
2) Severely Endangered Plot



3) Vulnerable Plot



4) Critically Endangered Plot



Animation Code This code is used to create the actual animations for the plots shown above. This was by far the most time consuming part of the project due to the amount of time it takes for each animation to render. The animations gradually add each data point into the scatter plot. The goal here is to sync each point with their respective note in the audio file.

Note: Yes, the code is repetitive and I definitely could have used a for loop to shorten down the code a lot.

```
#Animation for "Critically endangered" languages
critical_plot<-ggplot(world_map, aes(x = long, y = lat,group=group)) +
  geom_polygon(fill="lightgray", colour = "darkgray")+
  geom_point(data=critical_data,aes(x=Longitude,y=Latitude,group=Degree.of.endangerment)
             ,color="blue") + transition_states(Longitude,state_length=0.1
             ,transition_length=0) + shadow_mark(color="blue",past=T,future=F,alpha=0.7)

d<-animate(critical_plot,height=1050,width=1680,res=200,renderer=av_renderer()
             ,duration=121,fps=20)

#Animation for "Extinct" languages
extinct_plot<-ggplot(world_map, aes(x = long, y = lat,group=group)) +
  geom_polygon(fill="lightgray", colour = "darkgray")+
  geom_point(data=extinct_data,aes(x=Longitude,y=Latitude,group=Degree.of.endangerment)
             ,color="red") + transition_states(Longitude,state_length=0.1
             ,transition_length=0) + shadow_mark(color="red",past=T,future=F,alpha=0.7)

b<-animate(extinct_plot,height=1050,width=1680,res=200,renderer=av_renderer()
             ,duration=50,fps=20)
```

```
#Animation for "Severely Endangered" languages
severe_plot<-ggplot(world_map, aes(x = long, y = lat,group=group)) +
  geom_polygon(fill="lightgray", colour = "darkgray")+
  geom_point(data=severe_data,aes(x=Longitude,y=Latitude,group=Degree.of.endangerment)
             ,color="orange") + transition_states(Longitude,state_length=0.1
             ,transition_length=0) + shadow_mark(color="orange",past=T,future=F,alpha=0.7)

c<-animate(severe_plot,height=1050,width=1680,res=200,renderer=av_renderer()
             ,duration=110,fps=20)

#Animation for "Vulnerable" languages
vulnerable_plot<-ggplot(world_map, aes(x = long, y = lat,group=group)) +
  geom_polygon(fill="lightgray", colour = "darkgray")+
  geom_point(data=vulnerable_data,aes(x=Longitude,y=Latitude,group=Degree.of.endangerment)
             ,color="green") + transition_states(Longitude,state_length=0.1
             ,transition_length=0) + shadow_mark(color="green",past=T,future=F,alpha=0.7)

e<-animate(vulnerable_plot,height=1050,width=1680,res=200,renderer=av_renderer()
             ,duration=125,fps=20)

#Saving all animations to files

anim_save("extinctplot.mp4",b)
anim_save("severeplot.mp4",c)
anim_save("criticalTest.mp4",d)
anim_save("vulnerableplot.mp4",e)
```

5) Combining Audio and Animations

Now that we have our animations and our audio files, we can combine them using some video editing software like iMovie to create our final data sonification product. There are going to be some inconsistencies between the visual and audio components which is unfortunate. Part of these issues are most likely due to the timing of the animation being slightly off. Out of the four sonifications that were created, I think the sonification related to “Extinct” languages came out the best and was the most in sync.