Previously, when Rcrastinate was still on blogspot.com, I had a first look at [ten years of my playback history on Last.FM](http://rcrastinate.blogspot.com/2019/01/10-years-of-playback-history-on-lastfm.html). But there is still a lot one can do with this dataset. I wanted to try [{gganimate}](https://github.com/thomasp85/gganimate) for a long time and this nice longitudinal dataset gives me the opportunity.

**Loading and preparing the data**

First, I am loading the dataset. I already did some preparations like extracting the top five tags for each track and some other stuff I used in my [previous entry](https://rcrastinate.rbind.io/post/animating-10-years-of-listening-history-on-last-fm/).

library(knitr) # for 'kable'

scrobs <- readRDS("LastFM-history-proc2.Rds")

# I am getting rid of some columns we won't need.

scrobs <- scrobs[, c("artist", "track", "album",

"timestamp")]

nrow(scrobs)

## [1] 65356

kable(head(scrobs))

| **artist** | **track** | **album** | **timestamp** |
| --- | --- | --- | --- |
| Cat Power | Ramblin’ (Wo)man | Cat Power – Jukebox (Deluxe Edition) | 2009-01-01 13:30:00 |
| Cat Power | Metal Heart (2008 Version) | Cat Power – Jukebox (Deluxe Edition) | 2009-01-01 13:33:00 |
| Cat Power | Silver Stallion | Cat Power – Jukebox (Deluxe Edition) | 2009-01-01 13:37:00 |
| Cat Power | Aretha, Sing One for Me | Cat Power – Jukebox (Deluxe Edition) | 2009-01-01 13:40:00 |
| Cat Power | Lost Someone | Cat Power – Jukebox (Deluxe Edition) | 2009-01-01 13:43:00 |
| Cat Power | Lord, Help the Poor & Needy | Cat Power – Jukebox (Deluxe Edition) | 2009-01-01 13:46:00 |

So, you see: There is one row in the dataset for each song I played (65356 in total). Artist, track, and album name as well as the the timestamp of the scrobble are associated with each playback. We will now extract the date from the timestamp and create a new dataset art.day that contains the number of playbacks (or “scrobbles” in Last.FM’s terminology) for each artist on each day in the ten-years period.

scrobs$date <- substr(scrobs$timestamp, 1, 10)

art.day <- as.data.frame(table(scrobs$artist, scrobs$date))

names(art.day) <- c("Artist", "Date", "Scrobbles")

kable(tail(art.day[art.day$Artist == "Beck",]),

row.names = F)

| **Artist** | **Date** | **Scrobbles** |
| --- | --- | --- |
| Beck | 2018-12-25 | 1 |
| Beck | 2018-12-27 | 0 |
| Beck | 2018-12-28 | 0 |
| Beck | 2018-12-29 | 0 |
| Beck | 2018-12-30 | 9 |
| Beck | 2018-12-31 | 0 |

What you see above are the last 6 days of my scrobble history for *Beck*. Now, for some [{dplyr}](https://cran.r-project.org/web/packages/dplyr/vignettes/dplyr.html) magic. I am starting to really like what you can do with all the piping and stuff.

**Calculating cumulative sums**

What I want to do now is:

* Group art.day by artist.
* Create a new column called cum.plays (for “cumulated playbacks”) which holds the number of scrobbles for this artist up to the respective point in time.
* To achieve this, I am sorting the different groups of the dataset by date before computing the cumulated sum – just to be sure that the cumulated sum is correct.
* In the end, I am only ungrouping the dataset again and overwrite the variable holding the old one (because the operations before are “[non-destructive](https://www.cs.cmu.edu/~112/notes/notes-1d-lists.html#destructiveFunctions)”).

library(dplyr)

art.day %>% group\_by(Artist) %>%

arrange(Date) %>%

mutate(cum.plays = cumsum(Scrobbles)) %>%

ungroup() -> art.day

kable(tail(art.day[art.day$Artist == "Radiohead",]),

row.names = F)

| **Artist** | **Date** | **Scrobbles** | **cum.plays** |
| --- | --- | --- | --- |
| Radiohead | 2018-12-25 | 0 | 1524 |
| Radiohead | 2018-12-27 | 4 | 1528 |
| Radiohead | 2018-12-28 | 2 | 1530 |
| Radiohead | 2018-12-29 | 0 | 1530 |
| Radiohead | 2018-12-30 | 4 | 1534 |
| Radiohead | 2018-12-31 | 0 | 1534 |

With the last lines for *Radiohead* we can see that this worked: On Dec 25th, 2018, 1524 scrobbles were registered for Radiohead. On Dec 26th, I didn’t listen to any music, so this day is missing from the dataset altogether. This is no problem because I am converting the Date column to date format later. On Dec 27th, four more scrobbles were registered for Radiohead, so cum.plays is increased by four to 1528 and so on.

**Selecting top ten artists**

Now, I am doing three things:

* Create a table with the overall artist scrobbles to get the top ten artists.
* Restricting art.day only to scrobbles by these top ten artists and save the result in top.art.day.
* Convert the Date column in the new dataset to date format.

art.tab <- sort(table(scrobs$artist), decreasing = T)

top.art.day <- art.day[art.day$Artist %in% names(art.tab[1:10]),]

top.art.day$Date <- as.Date(top.art.day$Date)

nrow(top.art.day)

## [1] 28260

**Scrobbles distribution**

Since we’re already here, we can also see how much percent of *all* scrobbles are registered for the *top ten artists only*. We can also look at the overall distribution of scrobbles over artists using the table that I created before.

library(ggplot2)

sum(top.art.day$Scrobbles) / sum(art.day$Scrobbles) \* 100

## [1] 20.11751

ggplot() + aes(x = 1:length(art.tab), y = unname(art.tab), group = 1) +

geom\_line() +

labs(x = "Artist rank",

y = "Artist scrobbles") +

geom\_vline(aes(xintercept = 20), col = "red") +

theme\_minimal()



So, roughly 20% of all scrobbles come from the top ten artists alone. We are dealing with a massively scewed distribution here. See that red line in the plot? 20% of all scrobbles lie to the left of this line! It’s almost like word frequency distributions in a corpus of natural language data.

**Animating scrobble history**

Now for the main aim of this post. I am assuming {gganimate} is already installed. I then create a plot object p which also includes a call to transition\_reveal(Date). This is a function from {gganimate} which tells the plot to be revealed over time. The calls to geom\_segment() and geom\_text() create the artist labels on the right side of the plot that are connected to the moving dot. The artist names rise according to the current value of cum.plays. I got this code fragment from a [nice example by Thomas Lin Pedersen](https://github.com/thomasp85/gganimate/wiki/Temperature-time-series). You can read the discussion there for further explanations. Just a quick summary of important points:

* I had to switch off the legend with guides(col = F) to prevent overplotting of the moving labels and the color legend.
* I had to increase the plot margin on the right side to fit the longest artist name.
* Clipping is turned of with coord\_cartesian(clip = 'off') because the moving labels are not shown otherwise.

library(gganimate)

p <- ggplot(top.art.day, aes(x = Date, y = cum.plays, col = Artist, group = Artist)) +

geom\_line() +

geom\_point(size = 2) +

geom\_segment(aes(xend = as.Date("2019-01-10"),

yend = cum.plays),

linetype = 2,

colour = 'grey') +

geom\_text(aes(x = as.Date("2019-01-20"),

label = Artist),

hjust = 0, size = 4) +

coord\_cartesian(clip = 'off') +

transition\_reveal(Date) +

guides(col = F) +

labs(x = "Date", y = "Playcount",

col = "Artist",

title = "Artist playcount through time") +

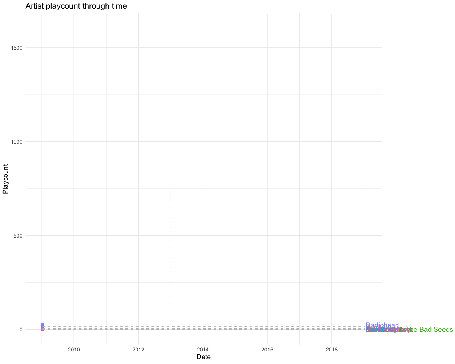
theme\_minimal() +

theme(plot.margin = margin(5.5, 120, 5.5, 5.5))

Now for animating the plot. Depending on the detailed settings, this could take a while. Especially the number of frames is heavily influencing rendering time.

animate(p, nframes = 200, end\_pause = 20,

height = 1000, width = 1250, res = 120)



It takes a little bit of tweaking of the parameters to get a readable result. Check out the different parameters for ?animate if you want to tweak your animations a little more. I have a tip for you: If you’re still tweaking your plot parameters like the dimensions or margins, you might want to set the nframes parameter to a lower number like 50 or even 20. This rapidly accelerates the rendering of your animation.

A few things are worth mentioning when I look at this plot:

* Do you see how I discovered Scott Matthew only in the end of 2013? He quickly made it into my top ten artists with a rapid sprint of scrobbles.
* Radiohead, Prince, and PJ Harvey were a lonely group at the top for quite some time. Then, Menomena entered the picture and joined this top group. Later, Beck also joins. At the end of 2018, with me discovering the album “Colors”, he takes second place after briefly being my top artist.
* Janelle Monáe hasn’t been in my top ten artists up until 2018. Then, she released “Dirty Computer”. I listened to this album a lot and she made a final sprint into the final top ten artists.
* That’s maybe the biggest flaw of this animation: Only the *final* top ten artists are included. The plot does not show the top ten artists *over time*. I would love to see the development of the top ten artists through time, maybe with greyed out lines whenever an artist leaves the top ten. For that, however, I would have to learn a lot more of {gganimate}.

Alright, that’s it. See you next time.