**Bonus: Adding an image to patchwork**

As a bonus, I’ll show y’all how I added an image to the patchwork layout by placing it within a ggplot graph and fixing the coordinates to avoid weird scaling issues.

**Set Up**

**Install and Load the Packages**

Thank you to “Dusty” who posted the tip to install and load packages using “easypackages”.

#install.packages("easypackages")

library(easypackages)

packages("tidyverse", "rtweet", "tidytext", "rtweet", "wordcloud2", "patchwork", "cran.stats", "data.table",

"gameofthrones", "ggimage", "magick", "ggpubr", "jpeg", "png")

**Set up our colour palette**

Using the beautiful Game of Thrones color palette from [Alejandro Jiménez](https://twitter.com/aljrico) in his [“gameofthrones” package.](https://github.com/aljrico/gameofthrones) Thank you to [Divya Seernani](https://twitter.com/DSeernani) for sharing!

#Set the palette using the beautiful GOT Arya palette from Alejandro Jiménez

pal <- got(20, option = "Arya")

#cherry pick a few extended

c <-"#889999"

c2 <- "#AAB7AF"

**Add your twitter credentials**

Create your twitter authentication token by following the steps in [Michael Kearneys](https://mikewk.com/) [beautiful documentation](https://rtweet.info/articles/auth.html). Replace the “ADD YOUR CREDS” with your own credentials.

#create\_token(

# app = "ADD YOUR CREDS",

# consumer\_key = "ADD YOUR CREDS",

# consumer\_secret = "ADD YOUR CREDS")

**1st Plot – Create a plot of the tweet stats (favorites, retweets)**

**Lookup the tweet and view stats**

lt <-lookup\_tweets('1229176433123168256')

lt

**Create a chart with the tweet stats**

p1 <- lt %>%

rename(Faves = favorite\_count, RTs = retweet\_count) %>%

select(Faves, RTs) %>% #select only the desired columns

gather("stat", "value") %>% #reformat to make the table long which is easier for bar charts to consume

ggplot(aes(stat, value)) + #plot the bar chart

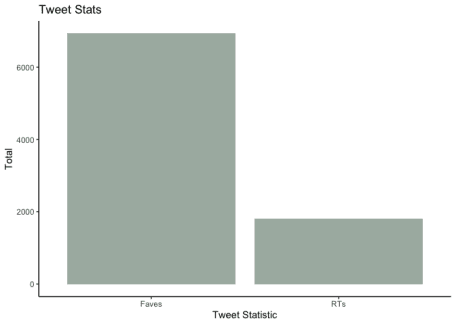
geom\_bar(stat="identity", fill=c2) +

theme\_classic() +

labs(title = "Tweet Stats",

x = "Tweet Statistic", y = "Total")

p1



**2nd Plot – Create a plot of the tweet stats (favorites, retweets)**

**Gather approx 1K of the retweet data**

The [get\_retweets()](https://rtweet.info/reference/get_retweets.html" \t "_blank) function only allows a max of 100 retweets to be pulled via the API at a time. This is a rate imposed by the twitter API. When pulling this data, I had quite a difficult time. Not only, did a lot of the suggested methods to getting cursors fail, the rate limiting wasn’t consistent. Sometimes I was able to get close to 1K tweets in 100 batches. Sometimes it blocked me for 15 min intervals (as expected). Since this is just an example to show patchwork, I decided to just grab 1K of the retweets which is roughly half of the full set of retweets. Further, I should let you know that I did attempt to put it in a function, but I couldn’t find an appropriate system wait time that would complete in a reasonable time and/or actually return the data. Please reach out if you have a better/proven method! In the meantime, here is my brute force method.

statusid <- '1229176433123168256' #set the first lowest retweet statusid to be the id of the original tweet

rtweets <- get\_retweets(statusid, n=100, parse = TRUE) # get 100 retweets

min\_id <- min(rtweets$status\_id)

rtweets2 <- get\_retweets(statusid, n=100, max\_id = min\_id, parse = TRUE) # get 100 retweets

min\_id <- min(rtweets2$status\_id)

**Graph the most common words used in the retweeters profile descriptions**

Most of the techniques used below to process the data and graph the data are taken from the [tidy text mining](https://www.tidytextmining.com/) book by [Julia Silge](https://twitter.com/juliasilge) and [David Robinson](https://twitter.com/drob)

data(stop\_words)

#Unnest the words - code via Tidy Text

rtweet\_table2 <- rtweet\_table %>%

unnest\_tokens(word, description) %>%

anti\_join(stop\_words) %>%

count(word, sort = TRUE) %>%

filter(!word %in% c('[t.co](http://t.co)', 'https'))

p2 <- rtweet\_table2 %>%

filter(n> 50) %>%

mutate(word = reorder(word, n)) %>%

ggplot(aes(word, n)) +

theme\_classic() +

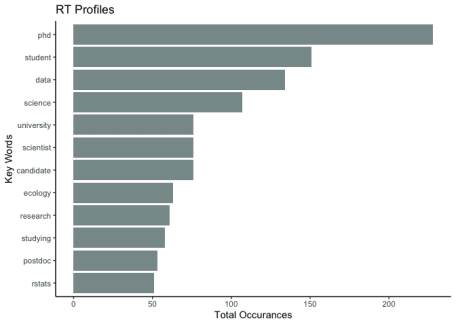
geom\_col(fill= c) +

labs(title = "RT Profiles",

x = "Key Words", y = "Total Occurances") +

coord\_flip()

p2



**3rd Plot – Plot the patchwork CRAN download stats**

**Gather the data**

To gather the patchwork download stats, I used the [“cran.stats” package](https://www.rdocumentation.org/packages/cran.stats/versions/0.1?tap_a=5644-dce66f&tap_s=10907-287229). The examples to process the download stats were very easy to follow and I used them as the basis for gathering the data.

dt = read\_logs(start = as.Date("2020-02-01"),

end = as.Date("2020-02-29"),

verbose = TRUE)

patchwork <- stats\_logs(dt, type="daily", packages=c("patchwork"),

dependency=TRUE, duration = 30L)

**Plot the CRAN download data**

I plotted the download data using the ggplot, the geom\_line() function and just a little extra fanciness to annotate the graph with the annotate() function. Great annotation examples [here](https://ggplot2.tidyverse.org/reference/annotate.html)

p3 <- ggplot(patchwork, aes(x=key, y=tot\_N, group=1)) +

geom\_line() + theme\_classic() + theme(axis.text.x = element\_text(angle = 60, hjust = 1)) +

ylim(0, 1500) +

labs(title = "Downloads of the R Patchwork Package",

x = "Date", y = "Total Downloads") +

annotate("rect", xmin = "2020-02-16", xmax = "2020-02-20", ymin = 400, ymax = 900,

alpha = .3, fill = c2) +

annotate(

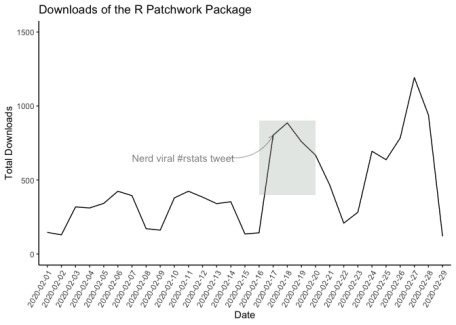
geom = "curve", alpha = 0.3, x = "2020-02-14", y = 650, xend = "2020-02-17", yend = 800,

curvature = .3, arrow = arrow(length = unit(2, "mm"))

) +

annotate(geom = "text", x = "2020-02-07", y = 650, label = "Nerd viral #rstats tweet", hjust = "left", alpha = 0.5)

p3



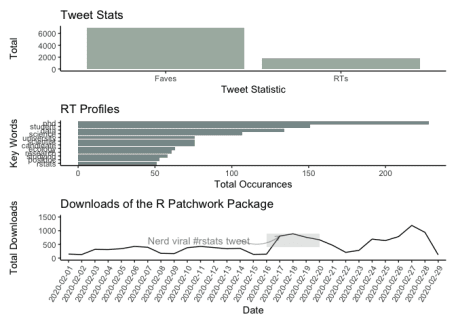
**Add the plots to the same graphic using patchwork**

As is the focus of this post, when this package was [shared on twitter](https://twitter.com/LittleMissData/status/1229176433123168256), people were very excited about it. Not only is it incredibly easy to use

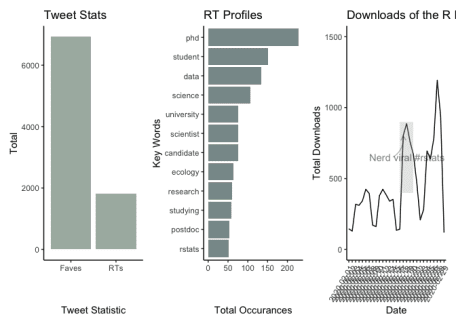
**Try a few layouts**

Using the plots p1, p2, p3 created above, try a few layouts following the package documentation

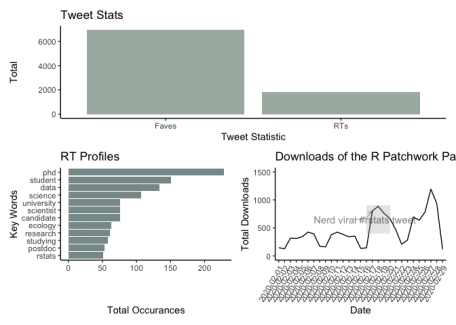
p1/p2/p3



p1 + p2 + p3



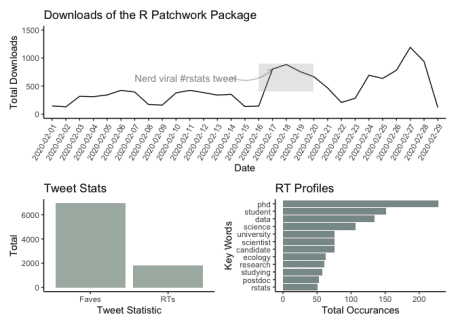
p1/ (p2 +p3)



#Final Layout

p <- p3 / (p1 + p2)

p



**Annotate the final layout**

We will select the final layout from the above code block and then add some overall titles, captioning and formatting.

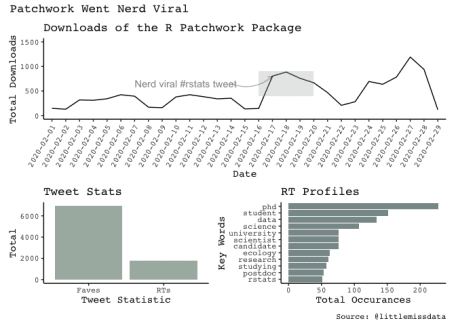
p + plot\_annotation(

title = 'Patchwork Went Nerd Viral',

caption = 'Source: @littlemissdata'

) &

theme(text = element\_text('mono'))



**Add an image to the patchwork graphic**

**Bring in the image**

Using an empty ggplot and the background\_image() function, you can bring an image into a graph object. Further, you can prevent image resizing with the coord\_fixed() function. This is important so the actual image doesn’t get resized with the patchwork placement.

twitter <- image\_read('<https://raw.githubusercontent.com/lgellis/MiscTutorial/master/Patchwork/twitter_post.png>')

twitter <- ggplot() +

background\_image(twitter) + coord\_fixed()

**Plot the image with patchwork**

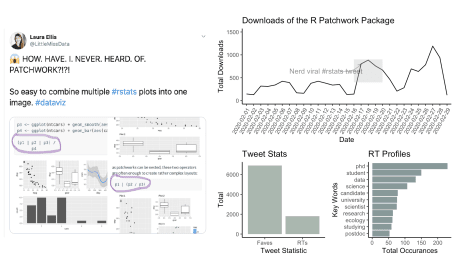
pF <- twitter + (p3/ (p1 + p2))

pF + plot\_annotation(

title = 'Patchwork Went Nerd Viral',

caption = 'Source: @littlemissdata'

)



**Thank You**

Please comment below if you enjoyed this blog, have questions, or would like to see something different in the future.