I caught a re-tweet of this tweet by [@harry\_stevens](http://twitter.com/harry_stevens):

THREAD: I wrote a post on [@observablehq](http://twitter.com/observablehq) about a map I made today. It shows a typical day in the life of a graphics journalist: You never know what problems you'll have to solve on deadline! <https://t.co/yRhW1wbLxN> [#d3js](https://twitter.com/hashtag/d3js?src=hash&ref_src=twsrc%5Etfw) [#dataviz](https://twitter.com/hashtag/dataviz?src=hash&ref_src=twsrc%5Etfw) 1/7 [pic.twitter.com/7N6mmK0nz3](https://t.co/7N6mmK0nz3)

— Harry Stevens ([@Harry\_Stevens](http://twitter.com/Harry_Stevens)) [May 16, 2019](https://twitter.com/Harry_Stevens/status/1129160171161300995?ref_src=twsrc%5Etfw)

Harry’s thread and Observable post are great on their own and both show power and utility of Observable javascript notebooks.

However, the re-tweet (which I’m not posting because it’s daft) took a swipe at both Python & R. Now, I’m all for a good swipe at Python (mostly to ensure we never forget all those broken spacebars and tab keys that language has caused) but I’ll gladly defend it and R together when it comes to Getting Things Done, even on deadline.

Let’s walk through what one of us might have done had we been in the same scenario as Harry.

**Mapping On A Deadline**

So, we have to create a map of historical tornado frequency trends on deadline.

We emailed researchers and [received three txt files](https://rud.is/dl/tornado.zip). One is a set of latitudes, another longitudes, and the final one is the trend value. It’s gridded data.

Download that ZIP and pretend you got three files in email vs a nice ZIP and make a new RStudio project called “tornado” and put those three files in a local-to-the-project-root data/ directory. Let’s read them in and look at them:

library(hrbrthemes) # not 100% necessary but i like my ggplot2 theme(s) :-)

library(tidyverse) # data wrangling & ggplot2

tibble(

lat = scan(here::here("data/lats.txt")),

lon = scan(here::here("data/lons.txt")),

trend = scan(here::here("data/trends.txt"))

) -> tornado

You very likely never directly use the `base::scan()` function, but it's handy here since we just have files of doubles with each value separated by whitespace. Now, let's see what we have:

```r

tornado

## # A tibble: 30,000 x 3

## lat lon trend

##

## 1 0.897 -180. 0

## 2 0.897 -179. 0

## 3 0.897 -178. 0

## 4 0.897 -176. 0

## 5 0.897 -175. 0

## 6 0.897 -174. 0

## 7 0.897 -173. 0

## 8 0.897 -172. 0

## 9 0.897 -170. 0

## 10 0.897 -169. 0

## # … with 29,990 more rows

summary(tornado)

## lat lon trend

## Min. : 0.8973 Min. :-179.99808 Min. :-0.4733610

## 1st Qu.:22.0063 1st Qu.: -90.00066 1st Qu.: 0.0000000

## Median :43.1154 Median : -0.00323 Median : 0.0000000

## Mean :43.1154 Mean : -0.00323 Mean : 0.0002756

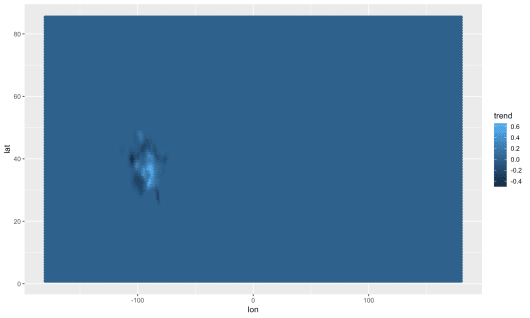
## 3rd Qu.:64.2245 3rd Qu.: 89.99419 3rd Qu.: 0.0000000

## Max. :85.3335 Max. : 179.99161 Max. : 0.6314569

#+ grid-overview

ggplot(tornado, aes(lon, lat)) +

geom\_point(aes(color = trend))

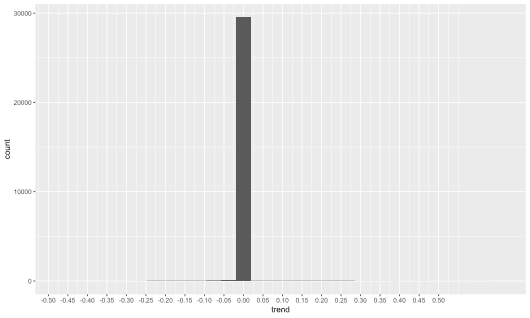
[](https://i1.wp.com/rud.is/b/wp-content/uploads/2019/05/grid-overview-1.png?ssl=1)

#+ trend-overview

ggplot(tornado, aes(trend)) +

geom\_histogram() +

scale\_x\_continuous(breaks = seq(-0.5, 0.5, 0.05))

[](https://i0.wp.com/rud.is/b/wp-content/uploads/2019/05/trend-overview-1.png?ssl=1)

Since we’re looking for trends (either direction) in just the [United States](https://www.nature.com/articles/s41612-018-0048-2.epdf?author_access_token=PQZthaEqlkut62uLi4HlpNRgN0jAjWel9jnR3ZoTv0Ofpugx93Jq3uh7IKWsjvSCCm9cT6oavbBDxy4CNfmgPbnVGCtRW0GfAXKcI3DSQ1vbeVbyw-jzqriwQAlEDMNsLcaDsYkvTU-SaxpOcafW-Q%3D%3D) the latitude and longitude ranges will need to be shrunk down a bit (it does indeed look like globally gridded data) and we’ll be able to shrink the data set a bit more since we only want to look at large or small tends.

We don’t really need modern R/ggplot2 mapping idioms for this project (i.e. the new {sf} ecosystem), so we’ll keep it “simple” (scare quotes since that’s a loaded term) and just use the built in maps and geom\_map(). First, let’s get the U.S. states and extract their bounding boxes/limits:

maps::map("state", ".", exact = FALSE, plot = FALSE, fill = TRUE) %>%

fortify(map\_obj) %>%

as\_tibble() -> state\_map

xlim <- range(state\_map$long)

ylim <- range(state\_map$lat)

NOTE: I tend not to use the handy ggplot::map\_data() function since it ends up clobbering purrr::map() which I use heavily (though not in this post). I also try to use {sf} these days so this tends to not be an issue anymore anyway.

Now, let’s focus in on the target area in the original paper and the Axios article:

filter(

tornado,

between(lon, -107, xlim[2]), between(lat, ylim[1], ylim[2]), # -107 gets us ~left-edge of TX

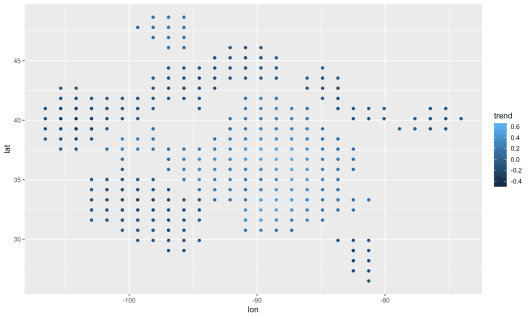
((trend < -0.07) | (trend > 0.07)) # approximates notebook selection range

) -> tornado

#+ grid-overview-2

ggplot(tornado, aes(lon, lat)) +

geom\_point(aes(color = trend))

[](https://i1.wp.com/rud.is/b/wp-content/uploads/2019/05/grid-overview-2-1.png?ssl=1)

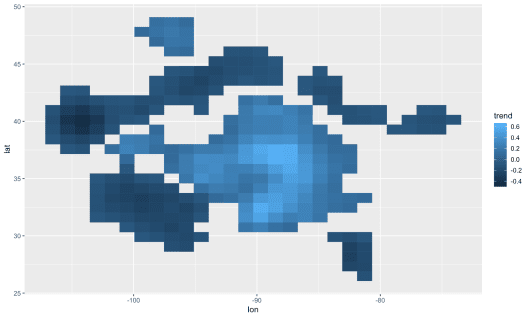
Now we’re getting close to our final solution.

As stated in the Observable notebook and implied by the word “grid” these dots are centroids of grid rectangles. This means we really want boxes, not points. The article got all fancy but it’s not really necessary since we can use ggplot2::geom\_tile() to get us said boxes:

#+ grid-overview-3

ggplot(tornado, aes(lon, lat)) +

geom\_tile(aes(fill = trend, color = trend))

[](https://i2.wp.com/rud.is/b/wp-content/uploads/2019/05/grid-overview-3-1.png?ssl=1)

Now, we just need to add in map layers, and tweak some aesthetics to make it look like a map. We’ll start naively:

#+ map-1

ggplot() +

geom\_tile(

data = tornado,

aes(lon, lat, fill = trend, color = trend)

) +

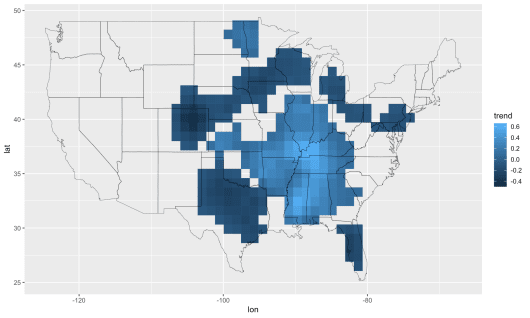
geom\_map(

data = state\_map, map = state\_map,

aes(long, lat, map\_id = region),

color = "black", size = 0.125, fill = NA

)

[](https://i2.wp.com/rud.is/b/wp-content/uploads/2019/05/map-1-1.png?ssl=1)

Our gridded data is definitely covering the right/same areas so we just need to make this more suitable for an article. We’ll use Harry’s palette and layer in U.S. state borders, an overall country border, and approximate the title and legend aesthetics:

#+ map-final

c(

"#023858", "#045a8d", "#0570b0", "#3690c0", "#74a9cf",

"#a6bddb", "#d0d1e6", "#ece7f2", "#fff7fb", "#ffffff",

"#ffffcc", "#ffeda0", "#fed976", "#feb24c", "#fd8d3c",

"#fc4e2a", "#e31a1c", "#bd0026", "#800026"

) -> grad\_cols # colors from article

ggplot() +

# tile layer

geom\_tile(

data = tornado,

aes(lon, lat, fill = trend, color = trend)

) +

# state borders

geom\_map(

data = state\_map, map = state\_map,

aes(long, lat, map\_id = region),

color = ft\_cols$slate, size = 0.125, fill = NA

) +

# usa border

borders("usa", colour = "black", size = 0.5) +

# color scales

scale\_colour\_gradientn(

colours = grad\_cols,

labels = c("Fewer", rep("", 4), "More"),

name = "Change in tornado frequency, 1979-2017"

) +

scale\_fill\_gradientn(

colours = grad\_cols,

labels = c("Fewer", rep("", 4), "More"),

name = "Change in tornado frequency, 1979-2017"

) +

# make it Albers-ish and ensure we can fit the borders in

coord\_map(

projection = "polyconic",

xlim = scales::expand\_range(range(tornado$lon), add = 2),

ylim = scales::expand\_range(range(tornado$lat), add = 2)

) +

# tweak legend aesthetics

guides(

colour = guide\_colourbar(

title.position = "top", title.hjust = 0.5

),

fill = guide\_colourbar(

title.position = "top", title.hjust = 0.5

)

) +

labs(

x = NULL, y = NULL

) +

theme\_ipsum\_rc(grid="") +

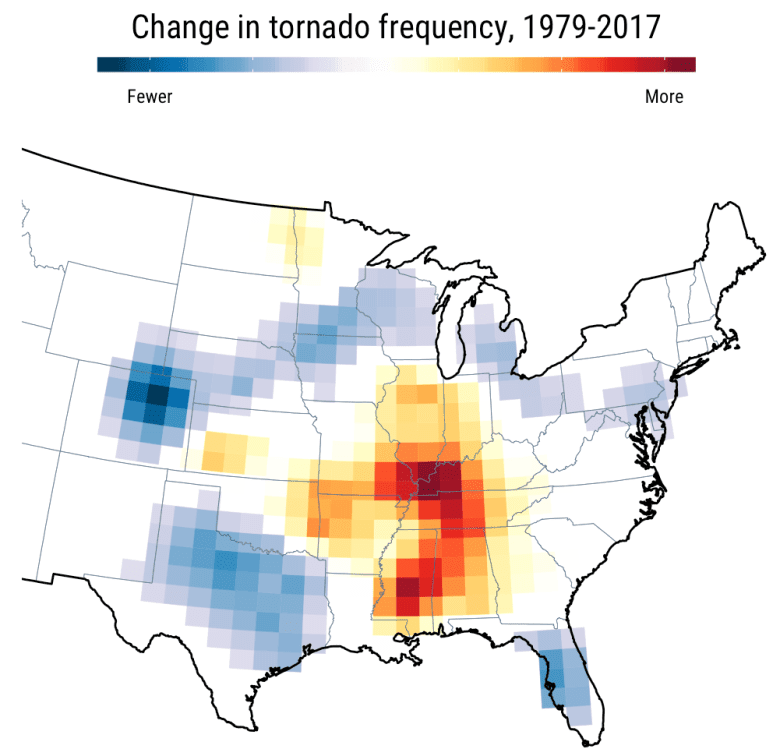
theme(axis.text = element\_blank()) +

theme(legend.position = "top") +

theme(legend.title = element\_text(size = 16, hjust = 0.5)) +

theme(legend.key.width = unit(4, "lines")) +

theme(legend.key.height = unit(0.5, "lines"))

[](https://i2.wp.com/rud.is/b/wp-content/uploads/2019/05/map-final-header.png?ssl=1)

**FIN**

I went through some extra steps for folks new to R but the overall approach was at the very least equally as expedient as the Observable one and — despite the claims by the quite daft retweet — this is no less “shareable” or “reusable” than the Observable notebook. You can clone the repo (<https://git.sr.ht/~hrbrmstr/tornado>) and reuse this work immediately.

If you take a stab at an alternate approach — especially if you do use {sf} — definitely blog about it and drop a link here or on Twitter.