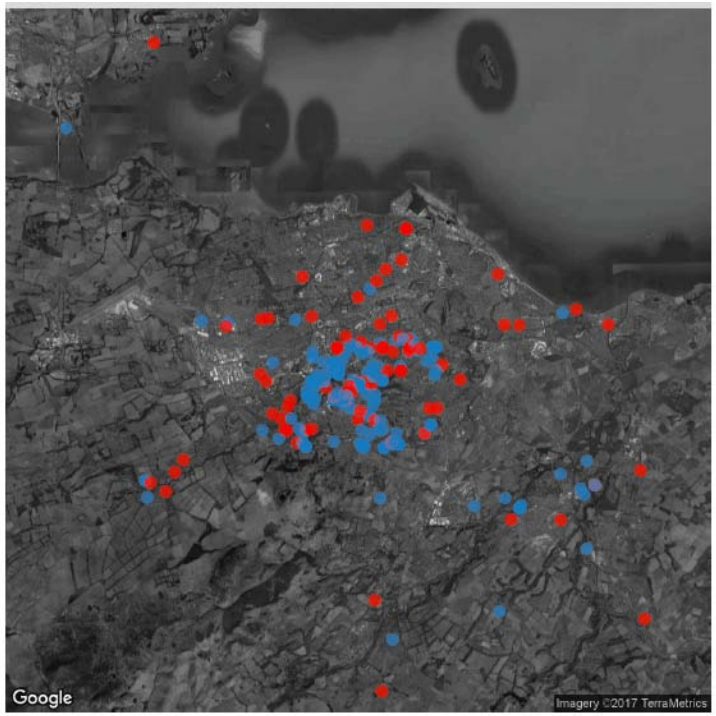
Plotting Postcodes onMaps



EdinbR User Group, 20 Sep 2017

Nevil Hopley

Newbie Notes

- R user since January 2017
- Previously programmed in BASIC, Assembly Code, Pascal, Lua
- Teaching since 1993
- Taught S6 AH Statistics since 1999
- Interested in Data Visualisation
- EdinbR Talk: Breakdown Plots (June 2017)

Nevil Hopley

Online Help

http://www.milanor.net/blog/maps-in-r-introduction-drawing-the-map-of-europe/
http://www.milanor.net/blog/maps-in-r-plotting-data-points-on-a-map/
http://www.milanor.net/blog/maps-in-r-choropleth-maps/

https://stackoverflow.com/questions/29036809/plotting-uk-postcodes-on-a-map-in-r

https://stevendkay.wordpress.com/2010/04/21/plotting-postcode-density-heatmaps-in-r/

UK Postcode Data

https://www.freemaptools.com/download-uk-postcode-lat-lng.htm

R Packages Used

```
library(ggplot2)
library(ggmap)
library(mapproj)
require(RColorBrewer)
```

Postcode Data Frame: postcodes_df

```
postcodes_df <- read.csv("ukpostcodes.csv",
                         header=TRUE,
                         sep=",",
                         na.strings = c("NA",""))
```

Contains 1,741,532 postcodes with their latitudes and longitudes

```
> head(postcodes_df)
  id postcode latitude longitude
1  1 AB10 1XG 57.14417 -2.114848
2  2 AB10 6RN 57.13788 -2.121487
3  3 AB10 7JB 57.12427 -2.127190
4  4 AB11 5QN 57.14270 -2.093295
5  5 AB11 6UL 57.13755 -2.112233
6  6 AB11 8RQ 57.13598 -2.072115
```

Tidy up Postcode Data Frame

```r
# strip spaces from Postcodes
postcodes_df$postcode <- mapply(function(b) gsub(" ","",b), b = postcodes_df$postcode)

# remove 'id' column as not needed
postcodes_df <- within(postcodes_df, rm(id))
```

mapply(…) versus unlist(lapply(...))

Pupil Postcodes Data Frame: df

```
df[order(df$Pupil_Postcode),]$Pupil_Postcode
 [1] "33626"     "55124"     "AB39 2NU"  "AB39 2NU"  "AB51 5HQ"  "AB51 5HQ"
[10] "EG10 5XD"  "EG10 5XD"  "EH1 1PG"   "EH1 2EL"   "EH1 2EL"   "EH1 2PW"
[19] "EH1 3PP"   "EH1 3PX"   "EH1 3PY"   "EH1 3PY"   "EH1 3RN"   "EH1 3RP"
[28] "EH10  4SG" "EH10 1DA"  "EH10 4AH"  "EH10 4AL"  "EH10 4AN"  "EH10 4AN"
[37] "EH10 4BL"  "EH10 4BL"  "EH10 4BL"  "EH10 4BL"  "EH10 4BN"  "EH10 4BN"
[46] "EH10 4BQ"  "EH10 4BQ"  "EH10 4BQ"  "EH10 4BQ"  "EH10 4BQ"  "EH10 4BR"
[55] "EH10 4BS"  "EH10 4BS"  "EH10 4BS"  "EH10 4BS"  "EH10 4BS"  "EH10 4BS"
[64] "EH10 4BW"  "EH10 4BW"  "EH10 4BW"  "EH10 4BW"  "EH10 4BW"  "EH10 4BW"
[73] "EH10 4DL"  "EH10 4DL"  "EH10 4DL"  "EH10 4DP"  "EH10 4DP"  "EH10 4DQ"
```

Spot the dirty Postcodes!

```
# strip spaces from Pupil_Postcodes
df$Pupil_Postcode <- mapply(function(b) gsub(" ","",b), b = df$Pupil_Postcode)
```

Merge Data Frames

postcodes_df$postcode contains UK postcode (and access to Latitude & Longitude)

df$Pupil_Postcode contains postcode of pupil (and access to other data about the pupil)

```
# join latitude and longitude to df via postcode id
# (all.x = TRUE) stops new rows being introduced
df <- merge(df, postcodes_df, by.x = 'Pupil_Postcode', by.y = 'postcode', all.x = TRUE)
```

get_map for Background Image

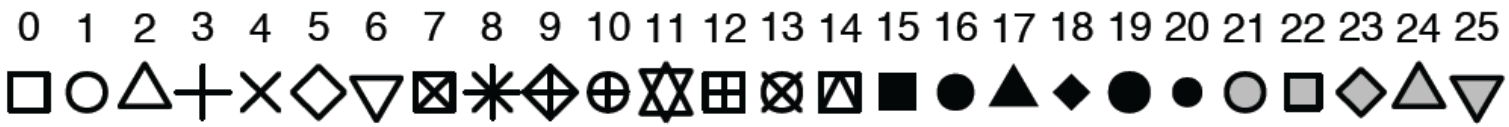
```
map <- get_map(location = 'EH10 5EG',
               zoom = 11,
               maptype = "satellite",
               color = "bw")
```

`location`	an address, longitude/latitude pair (in that order), or left/bottom/right/top bounding box
`zoom`	map zoom, an integer from 3 (continent) to 21 (building), default value 10 (city). openstreetmaps limits a zoom of 18, and the limit on stamen maps depends on the maptype. "auto" automatically determines the zoom for bounding box specifications, and is defaulted to 10 with center/zoom specifications. maps of the whole world currently not supported.
`scale`	scale argument of `get_googlemap` or `get_openstreetmap`
`maptype`	character string providing map theme. options available are "terrain", "terrain-background", "satellite", "roadmap", and "hybrid" (google maps), "terrain", "watercolor", and "toner" (stamen maps), or a positive integer for cloudmade maps (see ?get_cloudmademap)
`source`	Google Maps ("google"), OpenStreetMap ("osm"), Stamen Maps ("stamen"), or CloudMade maps ("cloudmade")

ggmap to Plot Data onto map

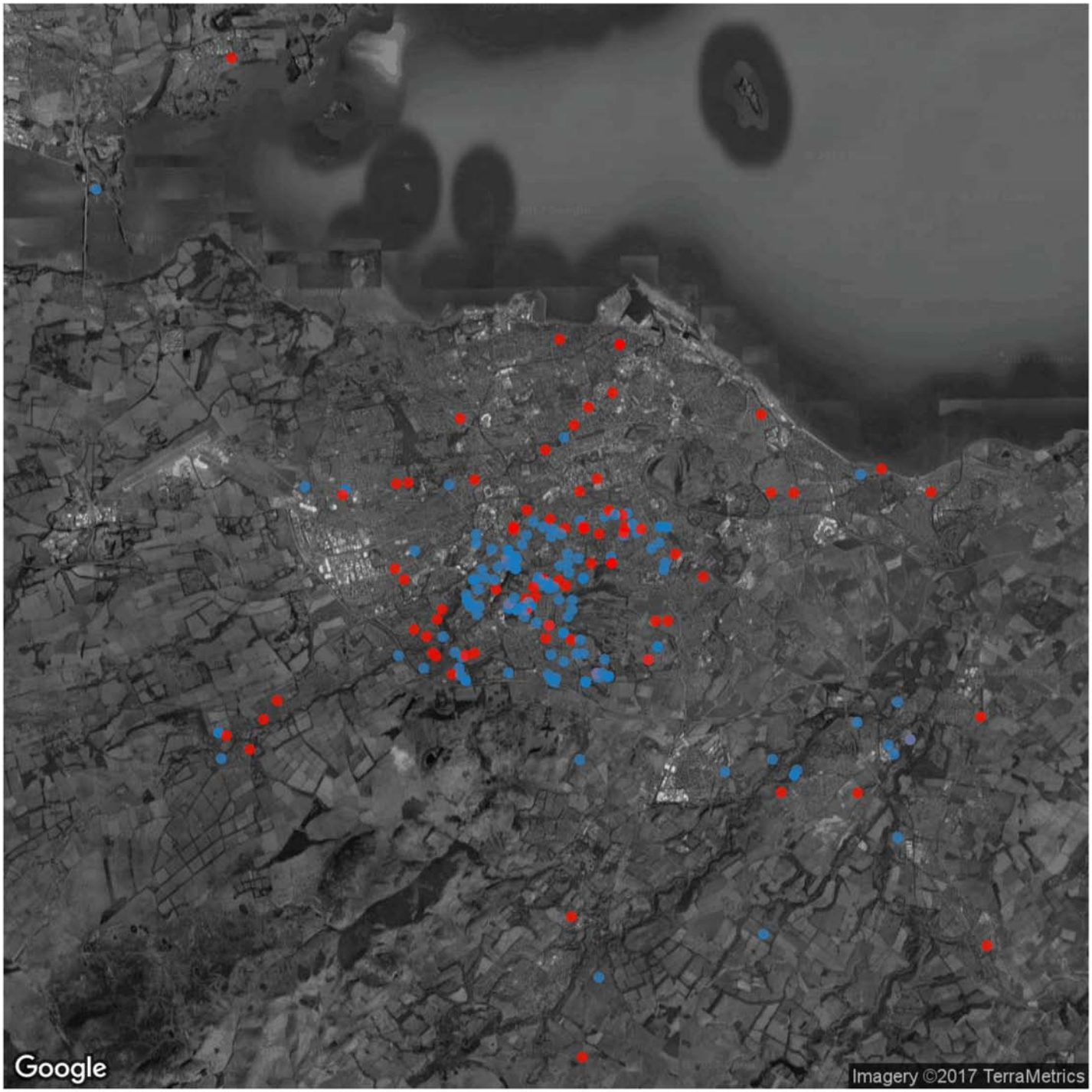
```
ggmap(map) +
  geom_point(data = subset(df, subset = Year_join_S1 > 2000 & Year_leave_S6 <2023),
             mapping = aes(x = longitude, y = latitude, colour = Analysis_Internal_External),
             size = 0.8,
             shape = 20,
             alpha = 0.8,
             position = "jitter") +
  scale_colour_brewer(palette = "Set1", direction = 1)
```

Shapes:

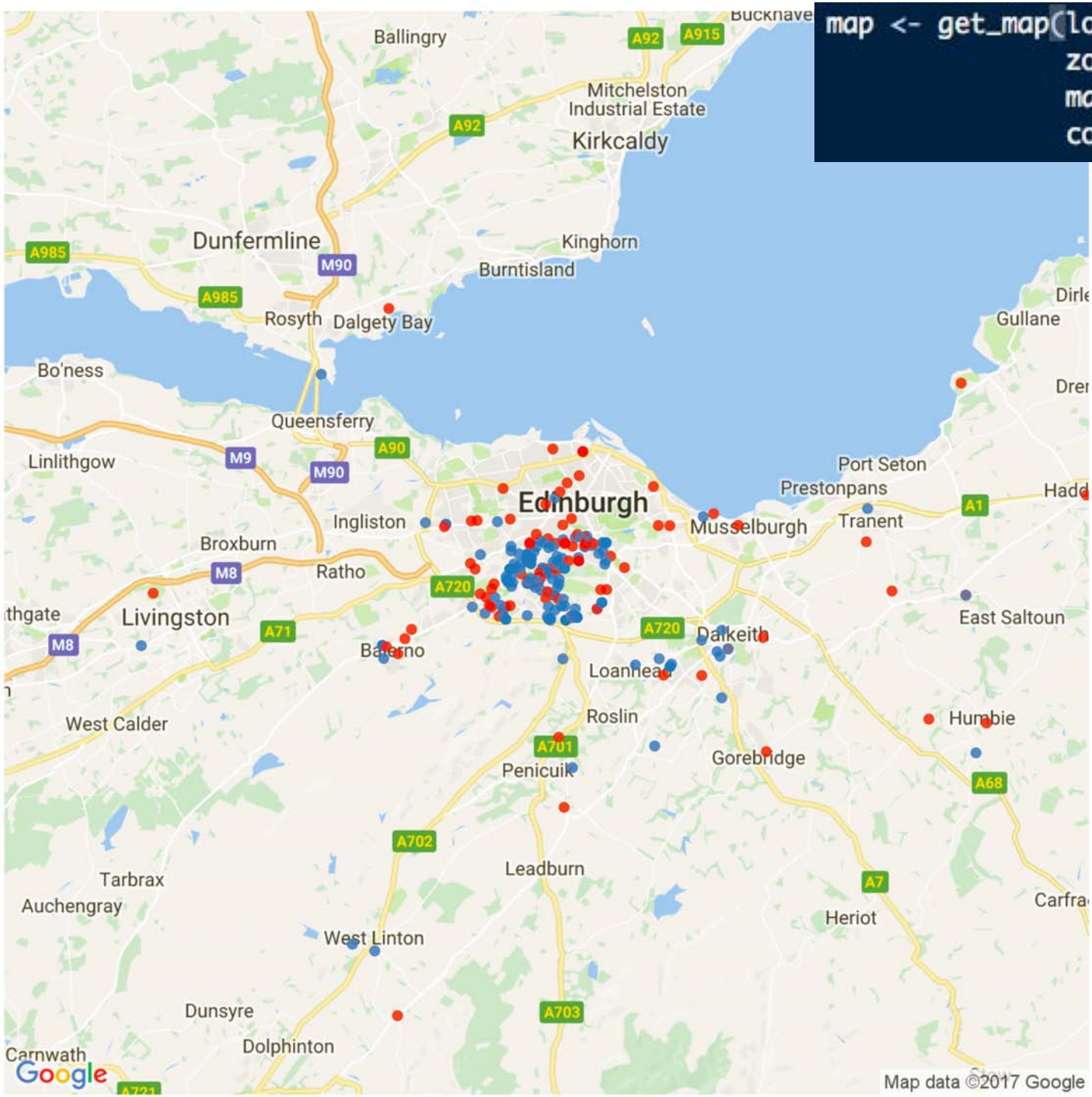


Colour Brewer Palettes:

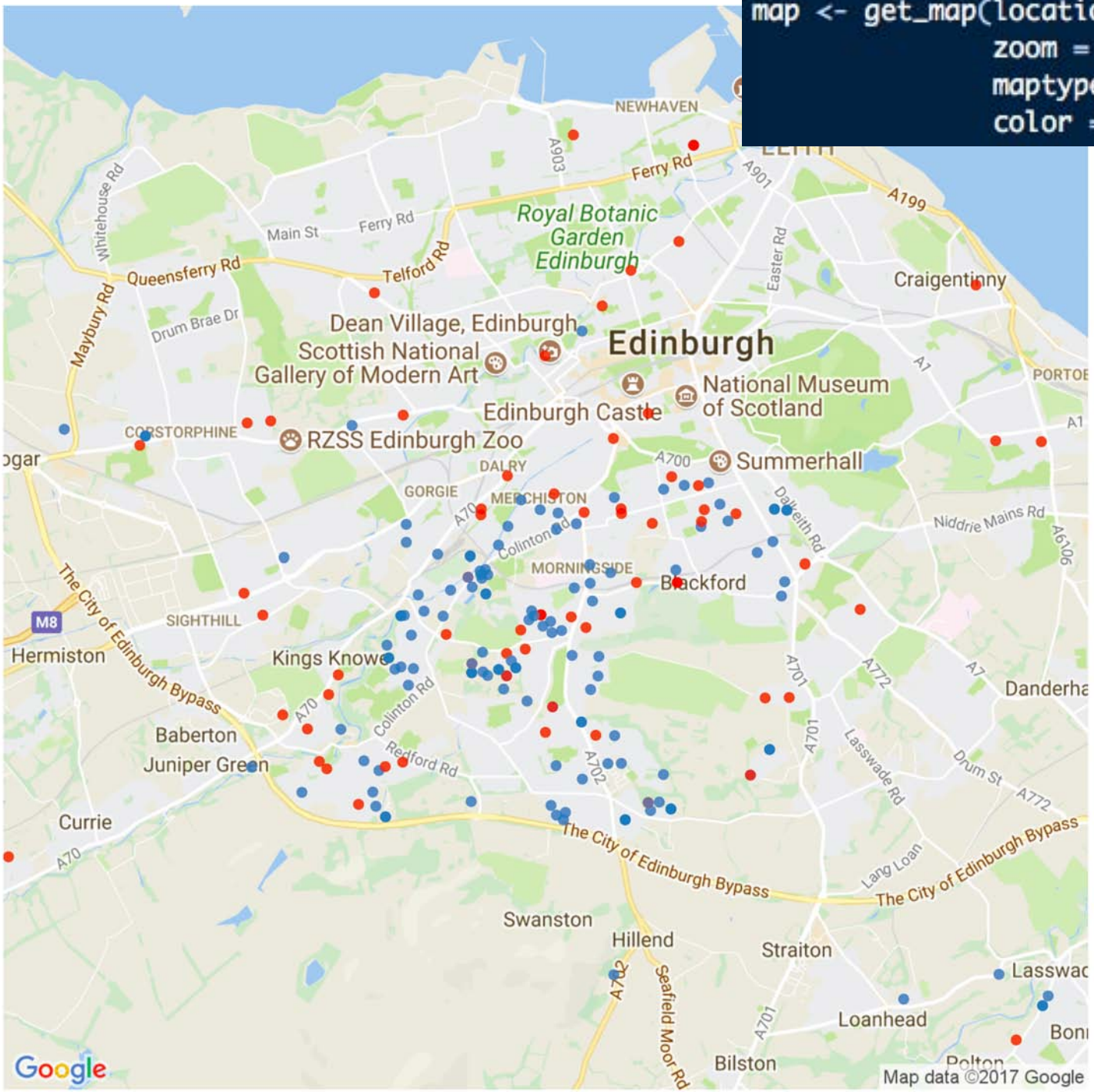
Tidy up axes and add a couple of facets….


Google
Imagery ©2017 TerraMetrics

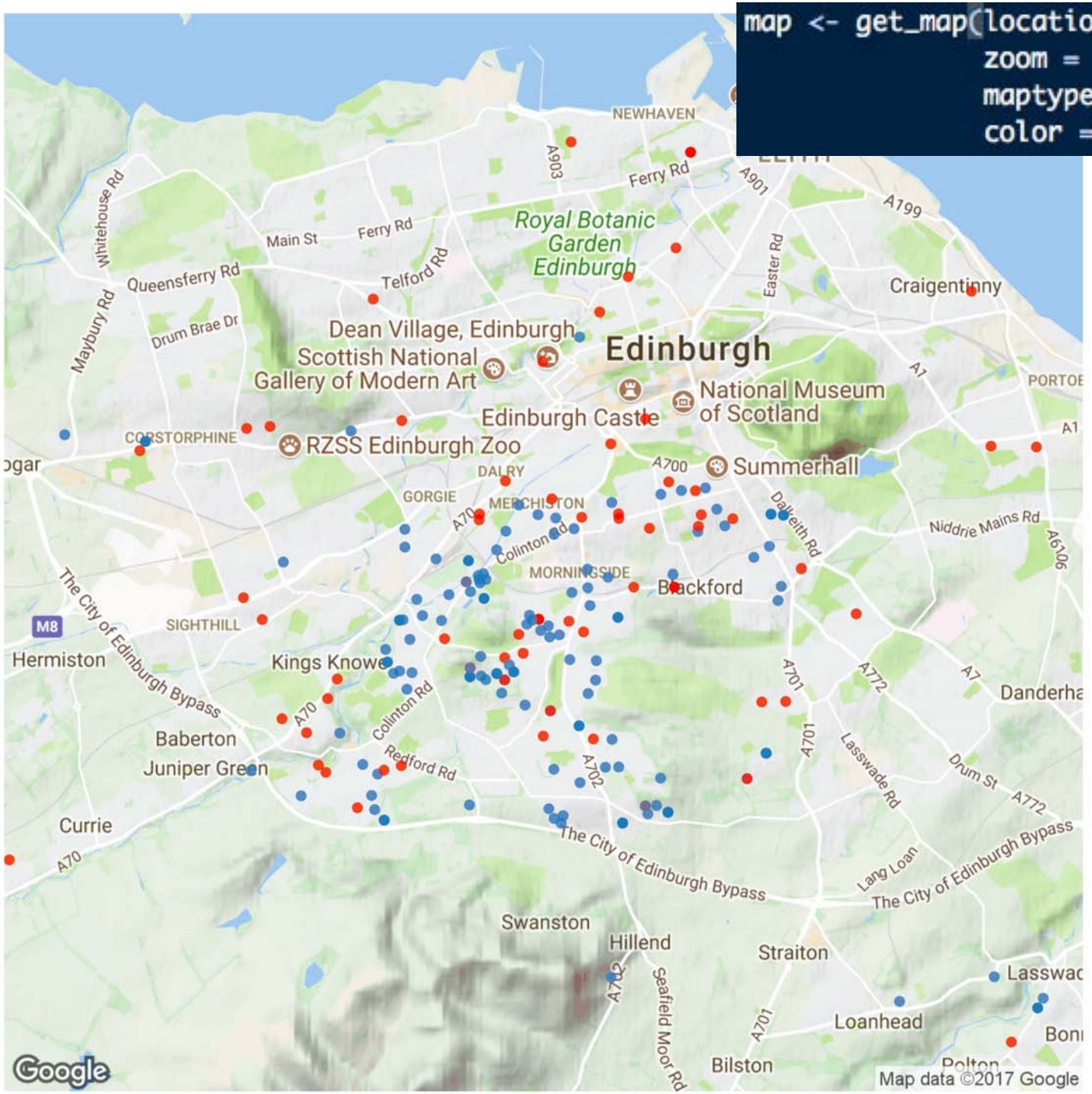
```
map <- get_map(location = 'EH10 5EG',
               zoom = 10,
               maptype = "roadmap",
               color = "color")
```



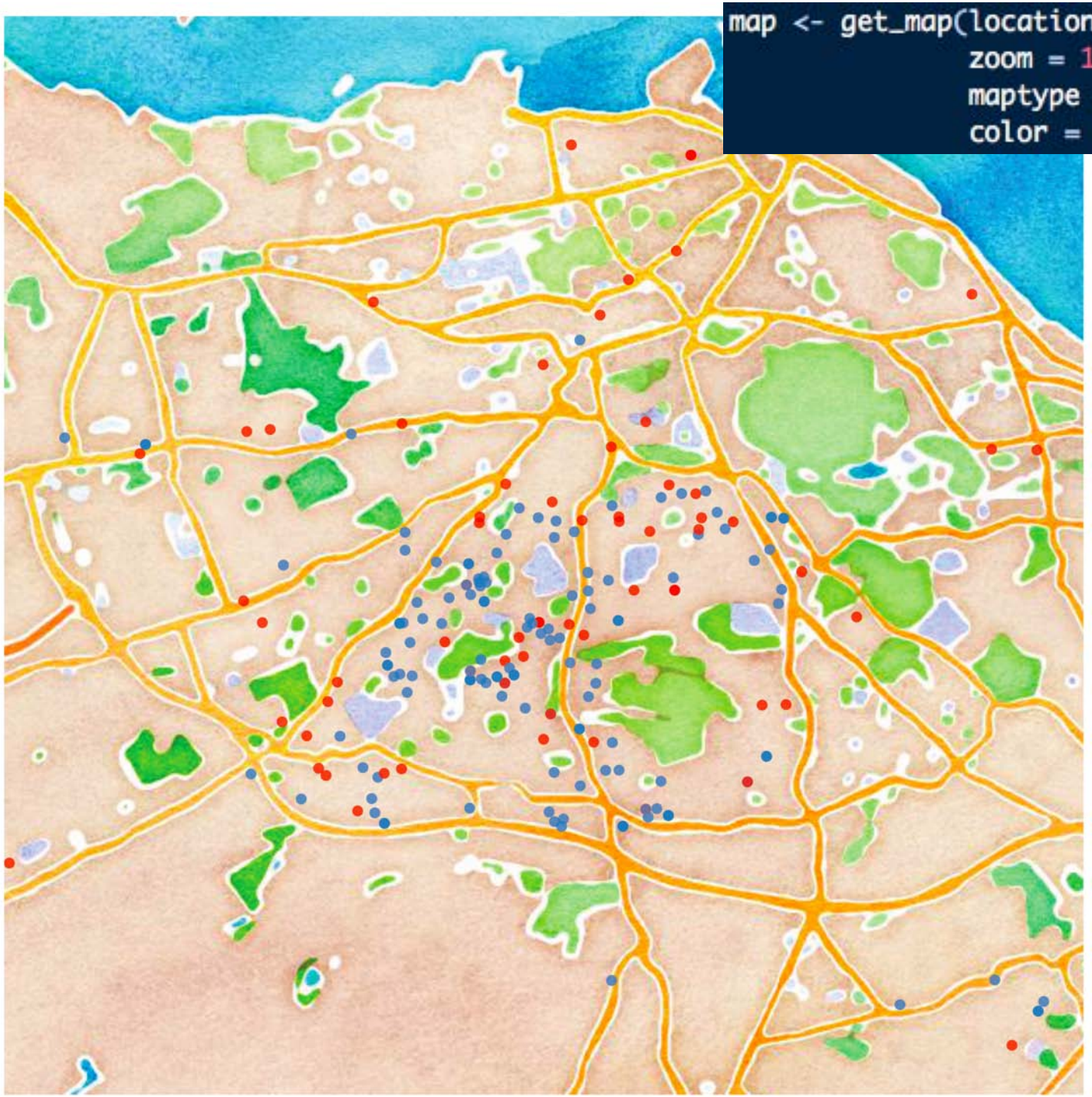
```
map <- get_map(location = 'EH10 5EG',
               zoom = 12,
               maptype = "roadmap",
               color = "color")
```



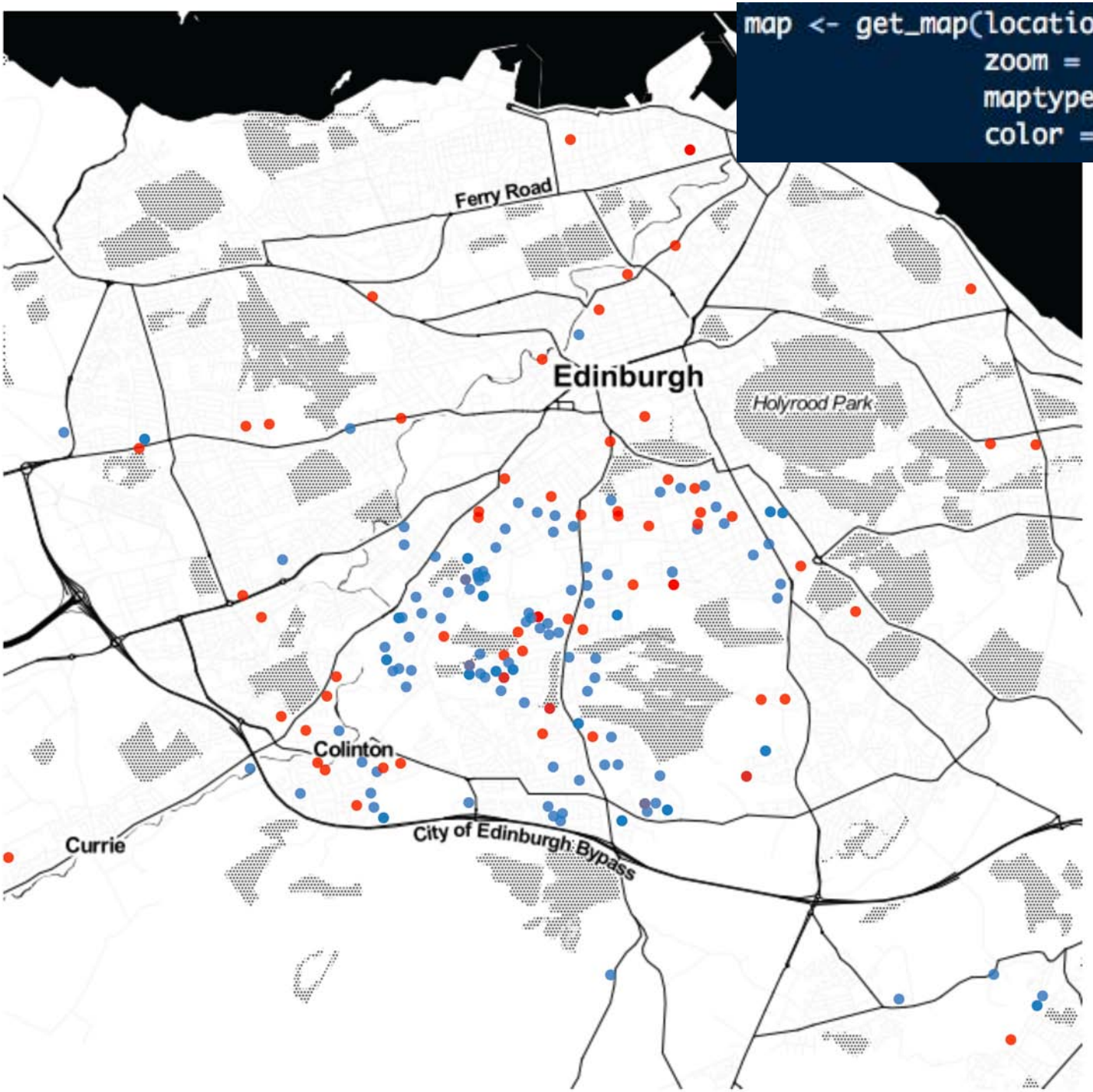
```
map <- get_map(location = 'EH10 5EG',
               zoom = 12,
               maptype = "terrain",
               color = "color")
```



```
map <- get_map(location = 'EH10 5EG',
               zoom = 12,
               maptype = "hybrid",
               color = "color")
```



```
map <- get_map(location = 'EH10 5EG',
               zoom = 12,
               maptype = "watercolor",
               color = "color")
```




```
map <- get_map(location = 'EH10 5EG',
               zoom = 12,
               maptype = "toner",
               color = "color")
```

Post-Mortem

Comments?

Questions?

Suggestions?

Criticisms?

Advice?

….all welcome!

Nevil Hopley

nevil@hopley.me