Data Visualization - 7. Make Maps (2)

Kieran Healy

Code Horizons

October 2, 2024

Making Maps

Load our packages

```r
library(here)      # manage file paths
library(socviz)    # data and some useful functions
library(tidyverse) # your friend and mine
library(maps)      # Some basic maps
library(sf)        # Simple Features Geometries and geom_sf()
library(ggforce)   # Useful enhancements to ggplot
```

Maps using Simple Features

`geom_polygon()` is limiting

It's very useful to have the intuition that, when drawing maps, we're just working with tables of `x` and `y` coordinates, and shapes represent quantities in our data, in a way that's essentially the same as any other geom. This makes it worth getting comfortable with what `geom_polygon()` and `coord_map()` are doing. But the business of having very large map tables and manually specifying projections is inefficient.

In addition, sometimes our data *really is* properly spatial, at which point we need a more rigorous and consistent way of specifying those elements. There's a whole world of Geodesic standards and methods devoted to specifying these things for GIS applications. R is not a dedicated GIS, but we can take advantage of these tools.

`geom_polygon()` is limiting

It's very useful to have the intuition that, when drawing maps, we're just working with tables of `x` and `y` coordinates, and shapes represent quantities in our data, in a way that's essentially the same as any other geom. This makes it worth getting comfortable with what `geom_polygon()` and `coord_map()` are doing. But the business of having very large map tables and manually specifying projections is inefficient.

In addition, sometimes our data *really is* properly spatial, at which point we need a more rigorous and consistent way of specifying those elements. There's a whole world of Geodesic standards and methods devoted to specifying these things for GIS applications. R is not a dedicated GIS, but we can take advantage of these tools.

Enter simple features, the `sf` package, and `geom_sf()`

The Simple Features package

When we load `sf` it creates a way to use several standard GIS concepts and tools, such as the GEOS library for computational geometry, the PROJ software that transforms spatial coordinates from one reference system to another, as in map projections, and the Simple Features standard for specifying the elements of spatial attributes.

```r
library(sf)
```

```
Linking to GEOS 3.11.0, GDAL 3.5.3, PROJ 9.1.0; sf_use_s2() is TRUE
```

Let's see the main upshot for us.

The nycdogs package

```r
library(nycdogs)
nyc_license
```

```
# A tibble: 493,072 × 9
   animal_name animal_gender animal_birth_year breed_rc         borough zip_code
   <chr>       <chr>                     <dbl> <chr>            <chr>      <int>
 1 Paige       F                          2014 Pit Bull (or Mi… Manhat…    10035
 2 Yogi        M                          2010 Boxer            Bronx      10465
 3 Ali         M                          2014 Basenji          Manhat…    10013
 4 Queen       F                          2013 Akita Crossbreed Manhat…    10013
 5 Lola        F                          2009 Maltese          Manhat…    10028
 6 Ian         M                          2006 Unknown          Manhat…    10013
 7 Buddy       M                          2008 Unknown          Manhat…    10025
 8 Chewbacca   F                          2012 Labrador (or Cr… Manhat…    10013
 9 Heidi-Bo    F                          2007 Dachshund Smoot… Brookl…    11215
10 Massimo     M                          2009 Bull Dog, French Brookl…    11201
# ℹ 493,062 more rows
# ℹ 3 more variables: license_issued_date <date>, license_expired_date <date>,
#   extract_year <dbl>
```

The nycdogs package

The metadata tells you this is not a regular tibble.

```
nyc_zips
```

```
Simple feature collection with 262 features and 11 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -74.25576 ymin: 40.49584 xmax: -73.6996 ymax: 40.91517
Geodetic CRS:  WGS 84
# A tibble: 262 × 12
   objectid zip_code po_name      state borough st_fips cty_fips bld_gpostal_code
      <int>    <int> <chr>        <chr> <chr>   <chr>   <chr>               <int>
 1        1    11372 Jackson He… NY    Queens  36      081                     0
 2        2    11004 Glen Oaks    NY    Queens  36      081                     0
 3        3    11040 New Hyde P… NY    Queens  36      081                     0
 4        4    11426 Bellerose    NY    Queens  36      081                     0
 5        5    11365 Fresh Mead… NY    Queens  36      081                     0
 6        6    11373 Elmhurst     NY    Queens  36      081                     0
 7        7    11001 Floral Park NY    Queens  36      081                     0
 8        8    11375 Forest Hil… NY    Queens  36      081                     0
 9        9    11427 Queens Vil… NY    Queens  36      081                     0
10       10    11374 Rego Park    NY    Queens  36      081                     0
# ℹ 252 more rows
```

The nycdogs package

```r
nyc_zips |>
  select(objectid:borough)
```

```
Simple feature collection with 262 features and 5 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -74.25576 ymin: 40.49584 xmax: -73.6996 ymax: 40.91517
Geodetic CRS:  WGS 84
# A tibble: 262 × 6
   objectid zip_code po_name          state borough                       geometry
      <int>    <int> <chr>            <chr> <chr>                    <POLYGON [°]>
 1        1    11372 Jackson Heights NY    Queens   ((-73.86942 40.74916, -73.89…
 2        2    11004 Glen Oaks        NY    Queens   ((-73.71068 40.75004, -73.70…
 3        3    11040 New Hyde Park    NY    Queens   ((-73.70098 40.7389, -73.703…
 4        4    11426 Bellerose        NY    Queens   ((-73.7227 40.75373, -73.722…
 5        5    11365 Fresh Meadows    NY    Queens   ((-73.81089 40.72717, -73.81…
 6        6    11373 Elmhurst         NY    Queens   ((-73.88722 40.72753, -73.88…
 7        7    11001 Floral Park      NY    Queens   ((-73.70098 40.7389, -73.699…
 8        8    11375 Forest Hills     NY    Queens   ((-73.85625 40.73672, -73.85…
 9        9    11427 Queens Village   NY    Queens   ((-73.74169 40.73682, -73.73…
10       10    11374 Rego Park        NY    Queens   ((-73.86451 40.73407, -73.85…
# ℹ 252 more rows
```

The `polygon` column is a list of lat/lon points that, when joined, draw the outline of the zip code area. This is *much* more compact than a big

Let’s make a summary table

```
nyc_license
```

```
# A tibble: 493,072 × 9
   animal_name animal_gender animal_birth_year
breed_rc         borough zip_code
   <chr>       <chr>                     <dbl>
<chr>            <chr>      <int>
 1 Paige       F                          2014
Pit Bull (or Mi… Manhat…    10035
 2 Yogi        M                          2010
Boxer            Bronx      10465
 3 Ali         M                          2014
Basenji          Manhat…    10013
 4 Queen       F                          2013
Akita Crossbreed Manhat…    10013
 5 Lola        F                          2009
Maltese          Manhat…    10028
 6 Ian         M                          2006
Unknown          Manhat…    10013
 7 Buddy       M                          2008
Unknown          Manhat…    10025
```

Let’s make a summary table

```r
nyc_license |>
  filter(extract_year == 2018)
```

```
# A tibble: 117,371 × 9
   animal_name animal_gender animal_birth_year
breed_rc          borough zip_code
   <chr>       <chr>                     <dbl>
<chr>             <chr>      <int>
 1 Ali         M                          2014
Basenji           Manhat…    10013
 2 Ian         M                          2006
Unknown           Manhat…    10013
 3 Chewbacca   F                          2012
Labrador (or Cr… Manhat…    10013
 4 Lola        F                          2006
Miniature Pinsc… Manhat…    10022
 5 Lucy        F                          2014
Dachshund Smoot… Brookl…    11215
 6 June        F                          2010
Cavalier King C… Brookl…    11238
 7 Apple       M                          2013
Havanese          Manhat…    10025
```

Let’s make a summary table

```
nyc_license ▷
  filter(extract_year == 2018) ▷
  group_by(breed_rc, zip_code)
```

```
# A tibble: 117,371 × 9
# Groups:   breed_rc, zip_code [18,945]
   animal_name animal_gender animal_birth_year
breed_rc          borough zip_code
   <chr>       <chr>                     <dbl>
<chr>             <chr>      <int>
 1 Ali         M                          2014
Basenji           Manhat…    10013
 2 Ian         M                          2006
Unknown           Manhat…    10013
 3 Chewbacca   F                          2012
Labrador (or Cr… Manhat…    10013
 4 Lola        F                          2006
Miniature Pinsc… Manhat…    10022
 5 Lucy        F                          2014
Dachshund Smoot… Brookl…    11215
 6 June        F                          2010
Cavalier King C… Brookl…    11238
 7 Apple       M                          2013
```

Let's make a summary table

```r
nyc_license |>
  filter(extract_year == 2018) |>
  group_by(breed_rc, zip_code) |>
  tally()
```

```
# A tibble: 18,945 × 3
# Groups:   breed_rc [311]
   breed_rc      zip_code     n
   <chr>            <int> <int>
 1 Affenpinscher    10005     1
 2 Affenpinscher    10011     1
 3 Affenpinscher    10013     1
 4 Affenpinscher    10014     1
 5 Affenpinscher    10016     1
 6 Affenpinscher    10017     1
 7 Affenpinscher    10018     1
 8 Affenpinscher    10019     1
 9 Affenpinscher    10021     1
10 Affenpinscher    10023     1
# ℹ 18,935 more rows
```

Let's make a summary table

```r
nyc_license |>
  filter(extract_year == 2018) |>
  group_by(breed_rc, zip_code) |>
  tally() |>
  mutate(freq = n / sum(n))
```

```
# A tibble: 18,945 × 4
# Groups:   breed_rc [311]
   breed_rc      zip_code     n   freq
   <chr>            <int> <int>  <dbl>
 1 Affenpinscher    10005     1 0.0303
 2 Affenpinscher    10011     1 0.0303
 3 Affenpinscher    10013     1 0.0303
 4 Affenpinscher    10014     1 0.0303
 5 Affenpinscher    10016     1 0.0303
 6 Affenpinscher    10017     1 0.0303
 7 Affenpinscher    10018     1 0.0303
 8 Affenpinscher    10019     1 0.0303
 9 Affenpinscher    10021     1 0.0303
10 Affenpinscher    10023     1 0.0303
# ℹ 18,935 more rows
```

Let’s make a summary table

```r
nyc_license |>
  filter(extract_year == 2018) |>
  group_by(breed_rc, zip_code) |>
  tally() |>
  mutate(freq = n / sum(n)) |>
  filter(breed_rc == "French Bulldog")
```

```
# A tibble: 161 × 4
# Groups:   breed_rc [1]
   breed_rc       zip_code     n    freq
   <chr>             <int> <int>   <dbl>
 1 French Bulldog    10001    27 0.0167
 2 French Bulldog    10002    20 0.0123
 3 French Bulldog    10003    36 0.0222
 4 French Bulldog    10004     9 0.00555
 5 French Bulldog    10005    15 0.00925
 6 French Bulldog    10006     8 0.00494
 7 French Bulldog    10007    17 0.0105
 8 French Bulldog    10009    51 0.0315
 9 French Bulldog    10010    31 0.0191
10 French Bulldog    10011    88 0.0543
# ℹ 151 more rows
```

Let’s make a summary table

```r
nyc_license |>
  filter(extract_year == 2018) |>
  group_by(breed_rc, zip_code) |>
  tally() |>
  mutate(freq = n / sum(n)) |>
  filter(breed_rc == "French Bulldog") ->
  nyc_fb
```

Let's make a summary table

```r
nyc_license |>
  filter(extract_year == 2018) |>
  group_by(breed_rc, zip_code) |>
  tally() |>
  mutate(freq = n / sum(n)) |>
  filter(breed_rc == "French Bulldog") ->
  nyc_fb
```

Now we have two tables again

```r
nyc_zips |> select(objectid:st_fips)
```

```
Simple feature collection with 262 features and 6 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -74.25576 ymin: 40.49584 xmax: -73.6996 ymax: 40.91517
Geodetic CRS:  WGS 84
# A tibble: 262 × 7
   objectid zip_code po_name     state borough st_fips                  geometry
      <int>    <int> <chr>       <chr> <chr>   <chr>               <POLYGON [°]>
 1        1    11372 Jackson He… NY    Queens  36      ((-73.86942 40.74916, -7…
 2        2    11004 Glen Oaks   NY    Queens  36      ((-73.71068 40.75004, -7…
 3        3    11040 New Hyde P… NY    Queens  36      ((-73.70098 40.7389, -73…
 4        4    11426 Bellerose   NY    Queens  36      ((-73.7227 40.75373, -73…
 5        5    11365 Fresh Mead… NY    Queens  36      ((-73.81089 40.72717, -7…
 6        6    11373 Elmhurst    NY    Queens  36      ((-73.88722 40.72753, -7…
 7        7    11001 Floral Park NY    Queens  36      ((-73.70098 40.7389, -73…
 8        8    11375 Forest Hil… NY    Queens  36      ((-73.85625 40.73672, -7…
 9        9    11427 Queens Vil… NY    Queens  36      ((-73.74169 40.73682, -7…
10       10    11374 Rego Park   NY    Queens  36      ((-73.86451 40.73407, -7…
# ℹ 252 more rows
```

```r
nyc_fb |> select(breed_rc:n)
```

```
# A tibble: 161 × 3
# Groups:   breed_rc [1]
   breed_rc       zip_code     n
   <chr>             <int> <int>
 1 French Bulldog    10001    27
 2 French Bulldog    10002    20
 3 French Bulldog    10003    36
 4 French Bulldog    10004     9
 5 French Bulldog    10005    15
 6 French Bulldog    10006     8
 7 French Bulldog    10007    17
 8 French Bulldog    10009    51
 9 French Bulldog    10010    31
10 French Bulldog    10011    88
# ℹ 151 more rows
```

Join them:

```r
fb_map <- left_join(nyc_zips, nyc_fb, by = "zip_code")
```

Ready to map

```r
fb_map ▷ select(zip_code, po_name, borough, breed_rc:freq, geometry)
```

```
Simple feature collection with 262 features and 6 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -74.25576 ymin: 40.49584 xmax: -73.6996 ymax: 40.91517
Geodetic CRS:  WGS 84
# A tibble: 262 × 7
   zip_code po_name    borough breed_rc      n     freq                  geometry
      <int> <chr>      <chr>   <chr>     <int>    <dbl>             <POLYGON [°]>
 1    11372 Jackson H… Queens  French …     13  8.02e-3 ((-73.86942 40.74916, -7…
 2    11004 Glen Oaks  Queens  French …      1  6.17e-4 ((-73.71068 40.75004, -7…
 3    11040 New Hyde … Queens  <NA>         NA NA       ((-73.70098 40.7389, -73…
 4    11426 Bellerose  Queens  French …      1  6.17e-4 ((-73.7227 40.75373, -73…
 5    11365 Fresh Mea… Queens  French …      7  4.32e-3 ((-73.81089 40.72717, -7…
 6    11373 Elmhurst   Queens  French …     14  8.64e-3 ((-73.88722 40.72753, -7…
 7    11001 Floral Pa… Queens  <NA>         NA NA       ((-73.70098 40.7389, -73…
 8    11375 Forest Hi… Queens  French …      8  4.94e-3 ((-73.85625 40.73672, -7…
 9    11427 Queens Vi… Queens  French …      2  1.23e-3 ((-73.74169 40.73682, -7…
10    11374 Rego Park  Queens  French …      6  3.70e-3 ((-73.86451 40.73407, -7…
# ℹ 252 more rows
```

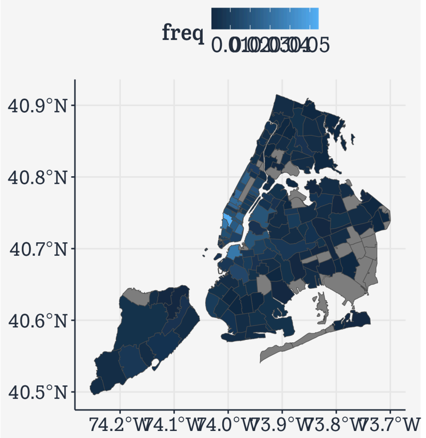
A NYC map theme

Just moving the legend, really.

```
theme_nymap <- function(base_size=9, base_family="") {
    require(grid)
    theme_bw(base_size=base_size, base_family=base_family) %+replace%
        theme(axis.line=element_blank(),
              axis.text=element_blank(),
              axis.ticks=element_blank(),
              axis.title=element_blank(),
              panel.background=element_blank(),
              panel.border=element_blank(),
              panel.grid=element_blank(),
              panel.spacing=unit(0, "lines"),
              plot.background=element_blank(),
              legend.justification = c(0,0),
              legend.position = c(0.05, 0.58),
              legend.direction = "horizontal"
        )
}
```

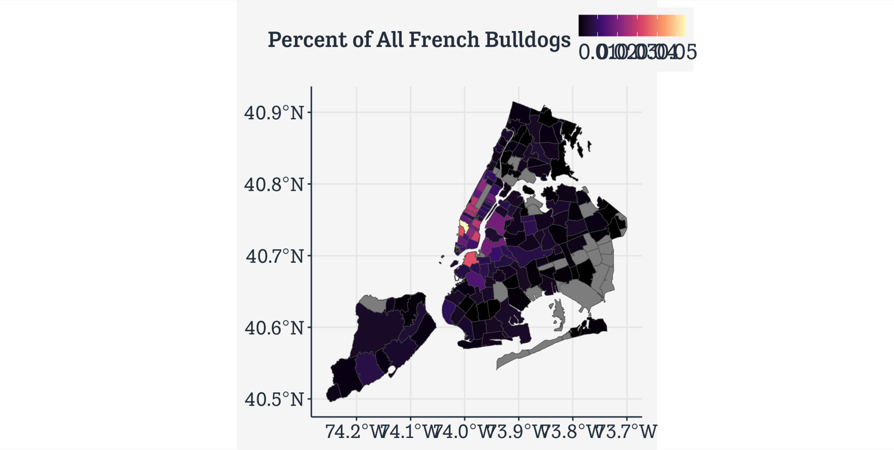
First cut at a map

```r
fb_map ▷
  ggplot(mapping = aes(fill = freq)) +
  geom_sf(color = "gray30", size = 0.1)
```



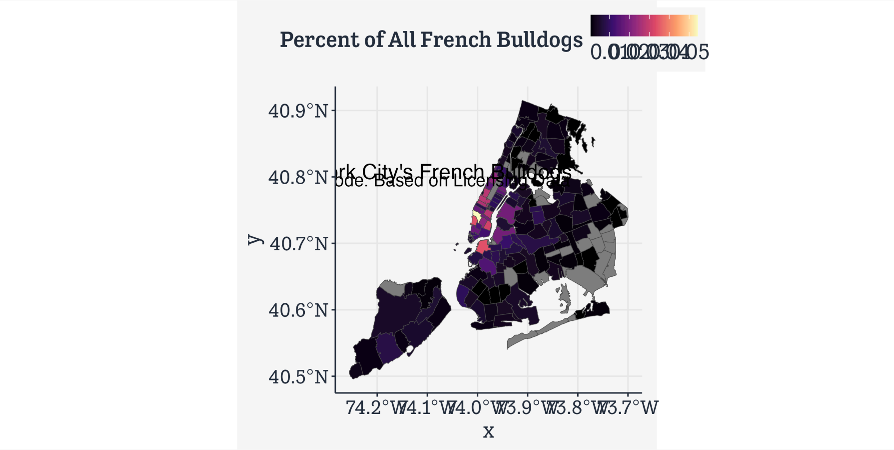
First cut at a map

```r
fb_map |>
  ggplot(mapping = aes(fill = freq)) +
  geom_sf(color = "gray30", size = 0.1) +
  scale_fill_viridis_c(option = "A") +
  labs(fill = "Percent of All French Bulldogs")
```



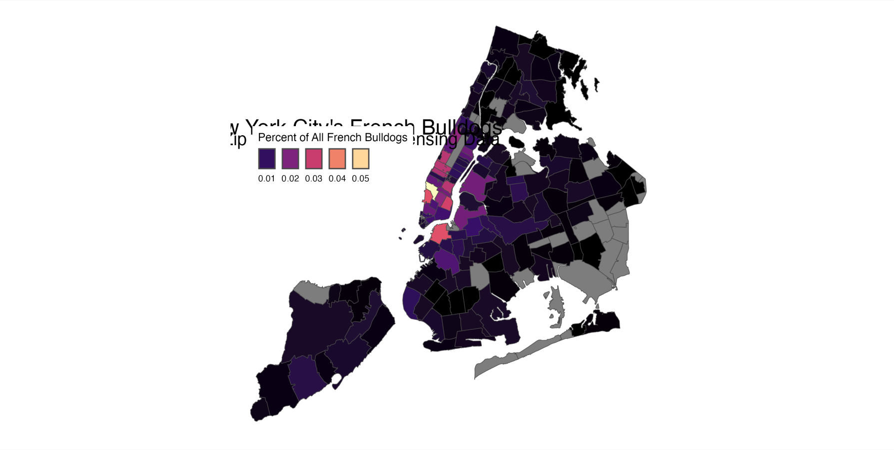
First cut at a map

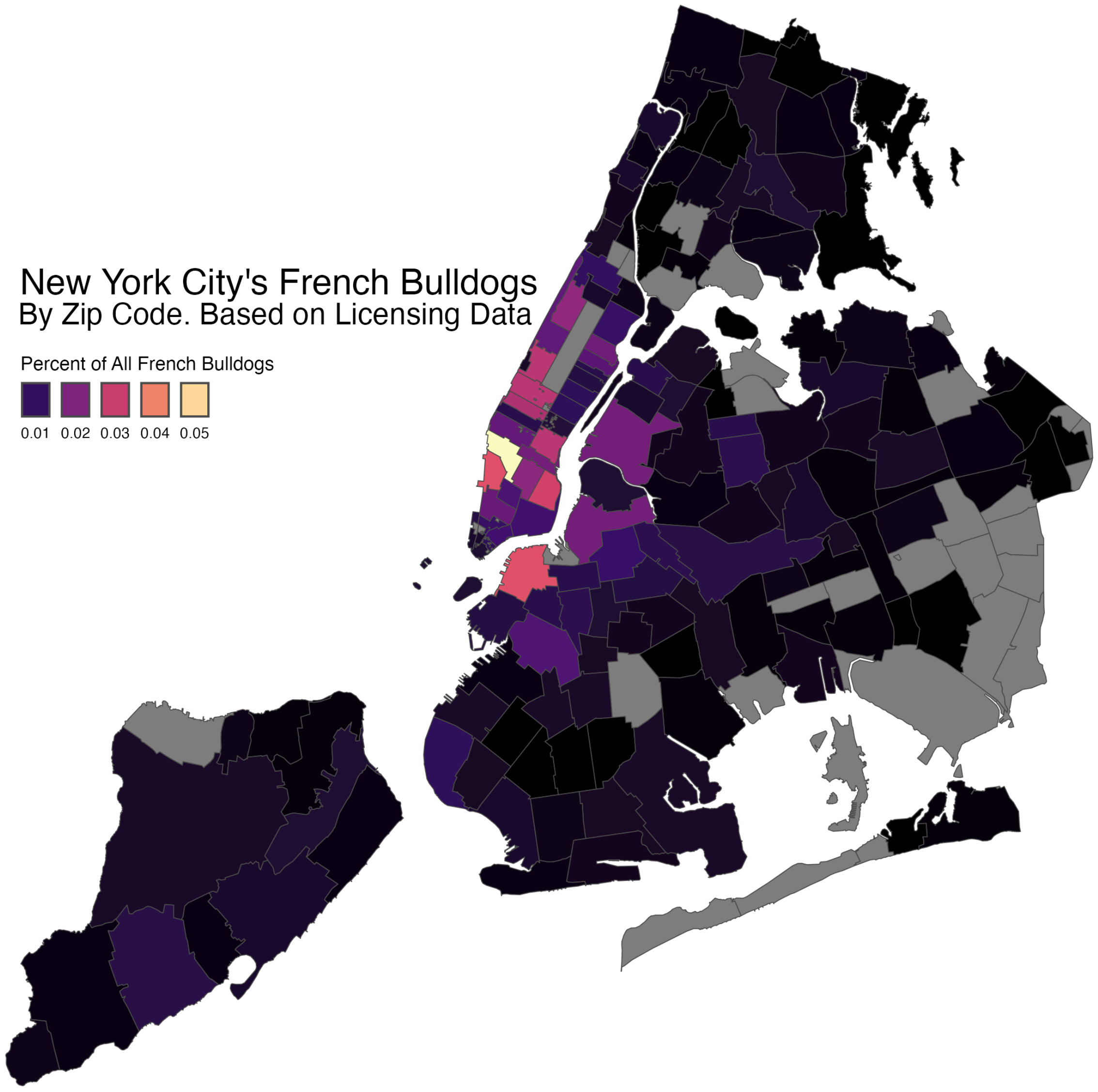
```r
fb_map ▷
  ggplot(mapping = aes(fill = freq)) +
  geom_sf(color = "gray30", size = 0.1) +
  scale_fill_viridis_c(option = "A") +
  labs(fill = "Percent of All French Bulldogs")
  annotate(geom = "text",
           x = -74.145 + 0.029,
           y = 40.82-0.012,
         label = "New York City's French Bull
         size = 6) +
    annotate(geom = "text",
           x = -74.1468 + 0.029,
           y = 40.8075-0.012,
         label = "By Zip Code. Based on Licen
         size = 5)
```



First cut at a map

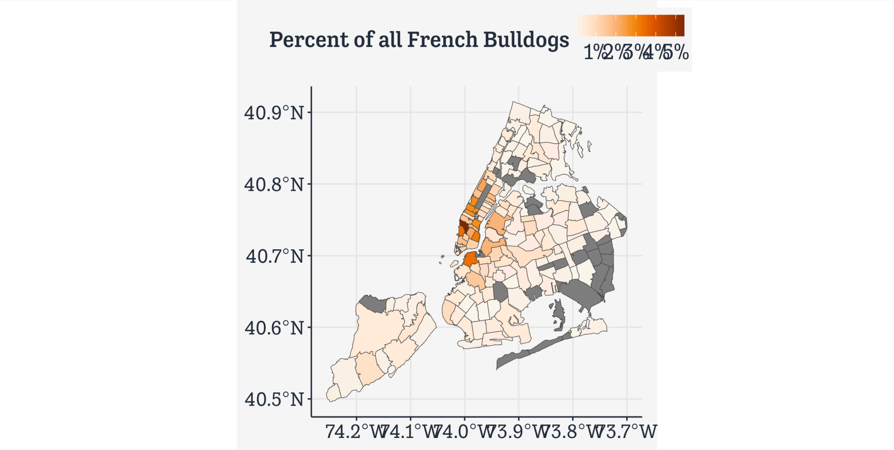
```
fb_map ▷
  ggplot(mapping = aes(fill = freq)) +
  geom_sf(color = "gray30", size = 0.1) +
  scale_fill_viridis_c(option = "A") +
  labs(fill = "Percent of All French Bulldogs")
  annotate(geom = "text",
             x = -74.145 + 0.029,
             y = 40.82-0.012,
           label = "New York City's French Bull
           size = 6) +
  annotate(geom = "text",
             x = -74.1468 + 0.029,
             y = 40.8075-0.012,
           label = "By Zip Code. Based on Licen
           size = 5) +
  kjhslides::kjh_theme_nymap() +
  guides(fill =
           guide_legend(title.position = "top",
                label.position = "bottom",
                keywidth = 1,
                nrow = 1))
```




New York City's French Bulldogs
By Zip Code. Based on Licensing Data
Percent of All French Bulldogs
0.01
0.02
0.03
0.04
0.05

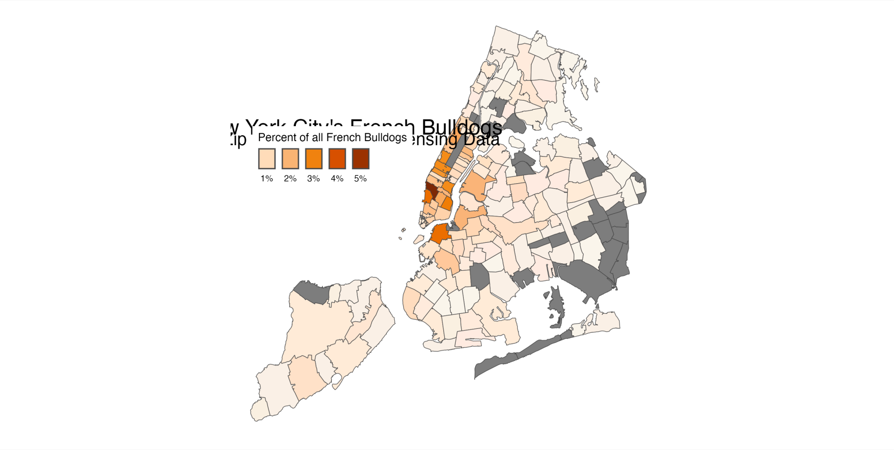
Use a different palette

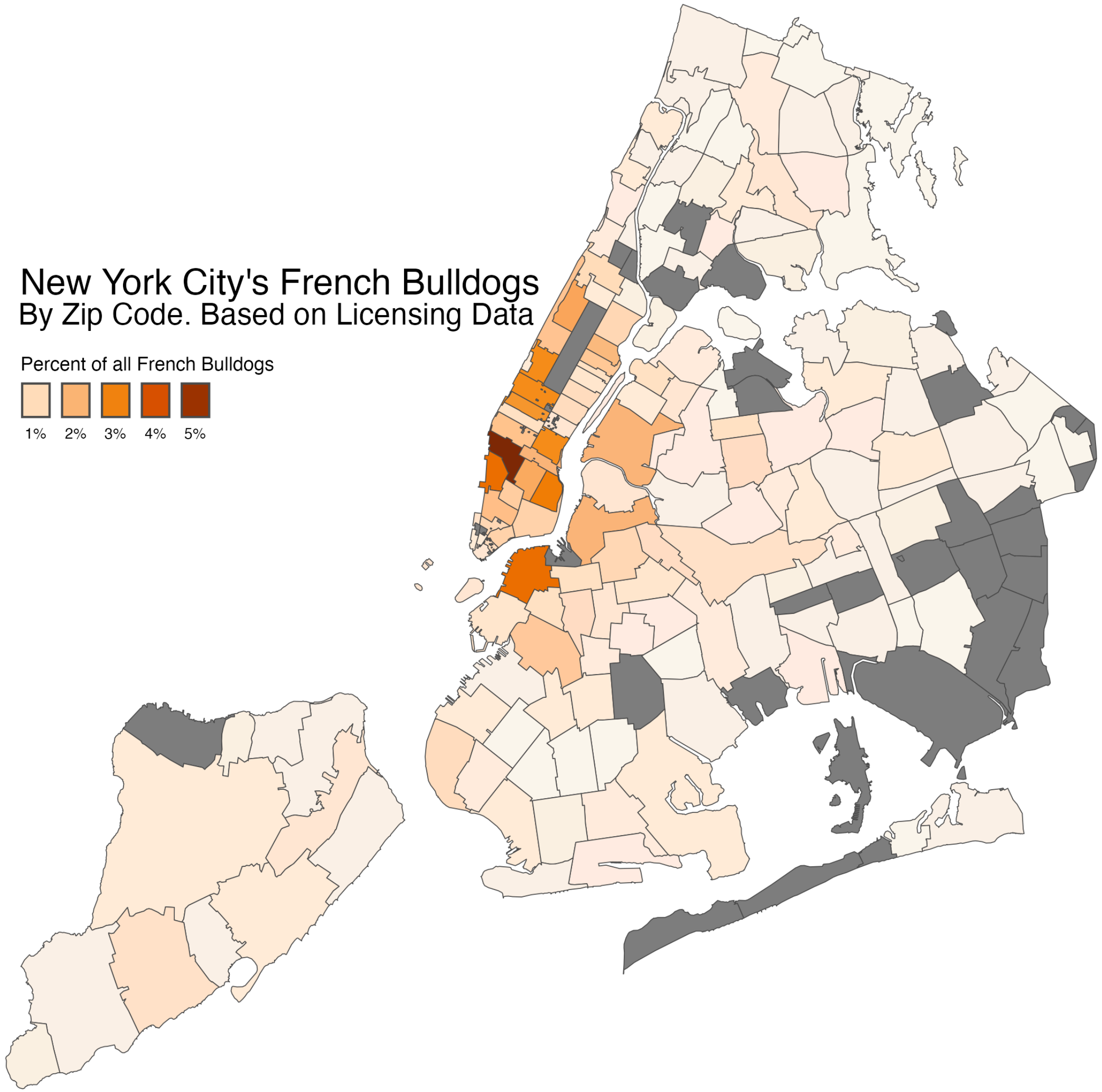
```r
library(colorspace)

fb_map |>
  ggplot(mapping = aes(fill = freq)) +
  geom_sf(color = "gray30", size = 0.1) +
  scale_fill_continuous_sequential(
    palette = "Oranges",
    labels = scales::label_percent()) +
  labs(fill = "Percent of all French Bulldogs")
```



Use a different palette

```r
fb_map |>
  ggplot(mapping = aes(fill = freq)) +
  geom_sf(color = "gray30", size = 0.1) +
  scale_fill_continuous_sequential(
    palette = "Oranges",
    labels = scales::label_percent()) +
  labs(fill = "Percent of all French Bulldogs")
  annotate(geom = "text",
           x = -74.145 + 0.029,
           y = 40.82-0.012,
           label = "New York City's French Bull
           size = 6) +
  annotate(geom = "text",
           x = -74.1468 + 0.029,
           y = 40.7955,
           label = "By Zip Code. Based on Licen
           size = 5) +
  kjhslides::kjh_theme_nymap() +
  guides(fill =
           guide_legend(title.position = "top",
                label.position = "bottom",
                keywidth = 1,
```





NYC Dogs Map mark 2

Keep the Zero-count Zips

```r
nyc_license  |>
  filter(extract_year == 2018) |>
  group_by(breed_rc, zip_code) |>
  tally() |>
  ungroup() |>
  complete(zip_code, breed_rc,
           fill = list(n = 0)) |>
  # Regroup to get the right denominator
  group_by(breed_rc) |>
  mutate(freq = n / sum(n)) |>
  filter(breed_rc == "French Bulldog") ->
  nyc_fb2

fb_map2 <- left_join(nyc_zips,
                     nyc_fb2,
                     by = "zip_code")
```

Keep the Zero-count Zips

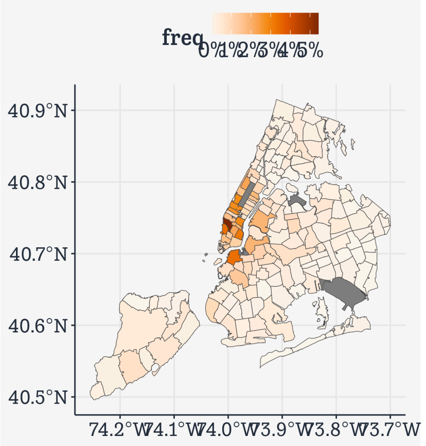
```r
fb_map2 |> select(zip_code, po_name, borough, breed_rc:freq, geometry)
```

```
Simple feature collection with 262 features and 6 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: -74.25576 ymin: 40.49584 xmax: -73.6996 ymax: 40.91517
Geodetic CRS:  WGS 84
# A tibble: 262 × 7
   zip_code po_name     borough breed_rc     n    freq                  geometry
      <int> <chr>       <chr>   <chr>    <int>   <dbl>             <POLYGON [°]>
 1    11372 Jackson He… Queens  French …    13 8.02e-3 ((-73.86942 40.74916, -7…
 2    11004 Glen Oaks   Queens  French …     1 6.17e-4 ((-73.71068 40.75004, -7…
 3    11040 New Hyde P… Queens  French …     0 0       ((-73.70098 40.7389, -73…
 4    11426 Bellerose   Queens  French …     1 6.17e-4 ((-73.7227 40.75373, -73…
 5    11365 Fresh Mead… Queens  French …     7 4.32e-3 ((-73.81089 40.72717, -7…
 6    11373 Elmhurst    Queens  French …    14 8.64e-3 ((-73.88722 40.72753, -7…
 7    11001 Floral Park Queens  French …     0 0       ((-73.70098 40.7389, -73…
 8    11375 Forest Hil… Queens  French …     8 4.94e-3 ((-73.85625 40.73672, -7…
 9    11427 Queens Vil… Queens  French …     2 1.23e-3 ((-73.74169 40.73682, -7…
10    11374 Rego Park   Queens  French …     6 3.70e-3 ((-73.86451 40.73407, -7…
# ℹ 252 more rows
```

This time, a number of previous NA rows are now zeroes instead.

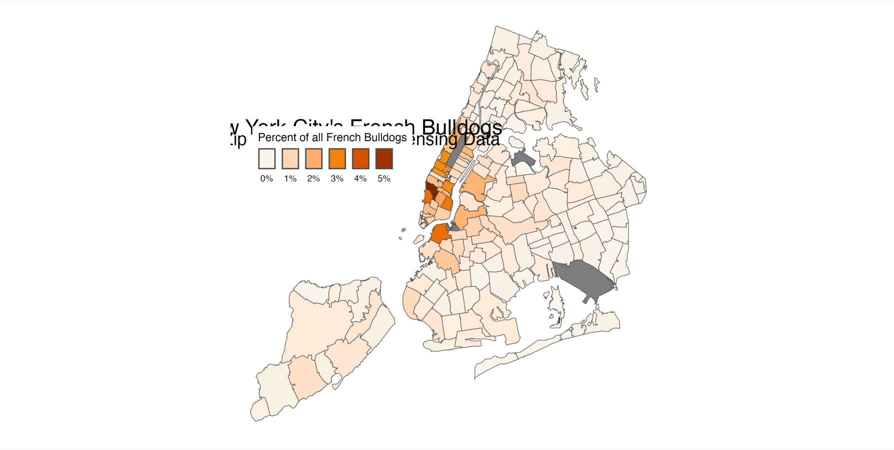
Keep the Zero-count Zips

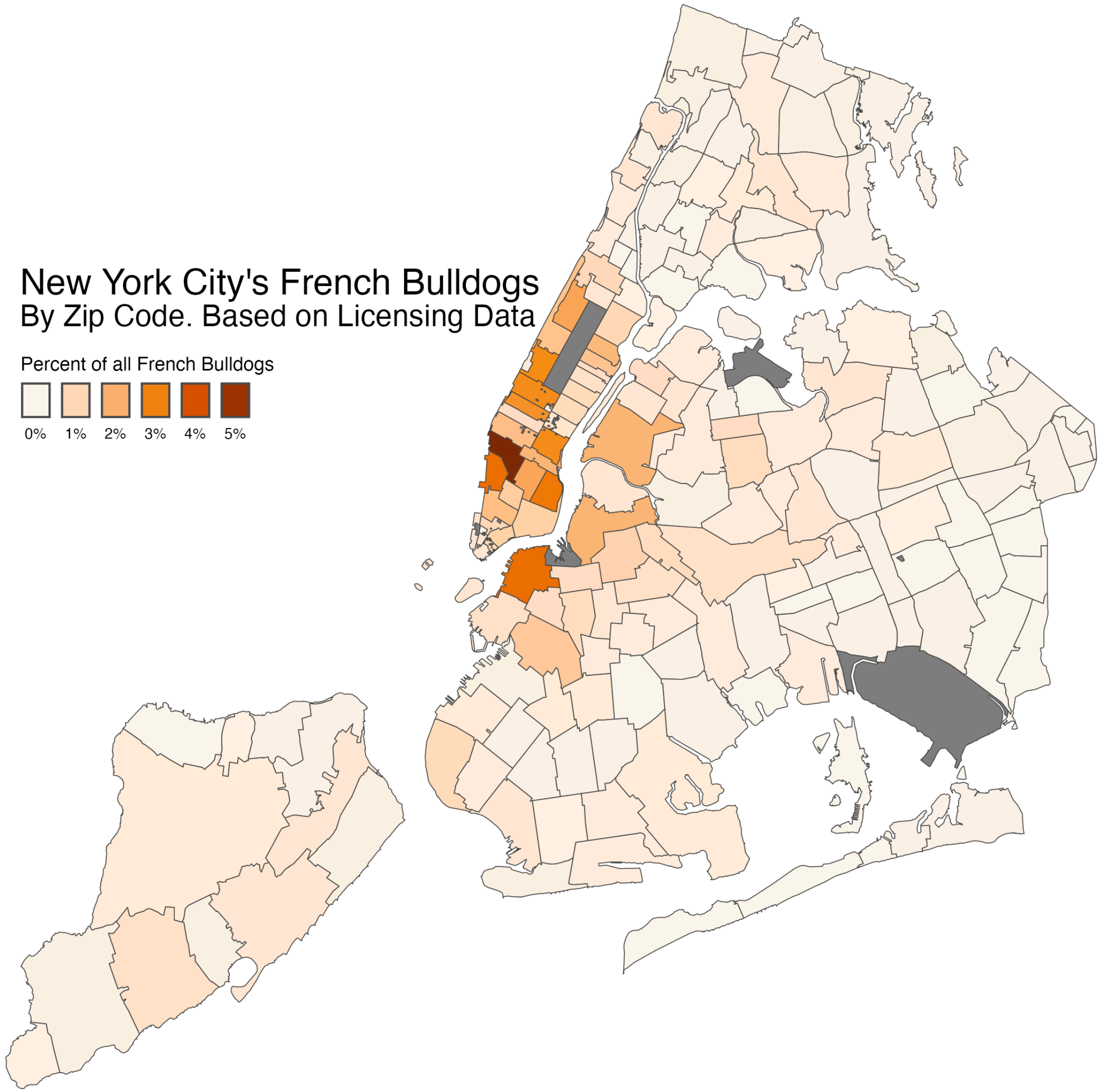
```r
fb_map2 ▷
  ggplot(mapping = aes(fill = freq)) +
  geom_sf(color = "gray30", size = 0.1) +
  scale_fill_continuous_sequential(
    palette = "Oranges",
    labels = scales::label_percent())
```



Keep the Zero-count Zips

```r
fb_map2 ▷
  ggplot(mapping = aes(fill = freq)) +
  geom_sf(color = "gray30", size = 0.1) +
  scale_fill_continuous_sequential(
    palette = "Oranges",
    labels = scales::label_percent()) +
  labs(fill = "Percent of all French Bulldogs")
  annotate(geom = "text",
           x = -74.145 + 0.029,
           y = 40.82-0.012,
           label = "New York City's French Bull
           size = 6) +
  annotate(geom = "text",
           x = -74.1468 + 0.029,
           y = 40.7955,
           label = "By Zip Code. Based on Licen
           size = 5) +
  kjhslides::kjh_theme_nymap() +
  guides(fill =
           guide_legend(title.position = "top",
                        label.position = "botto
                        keywidth = 1,
```



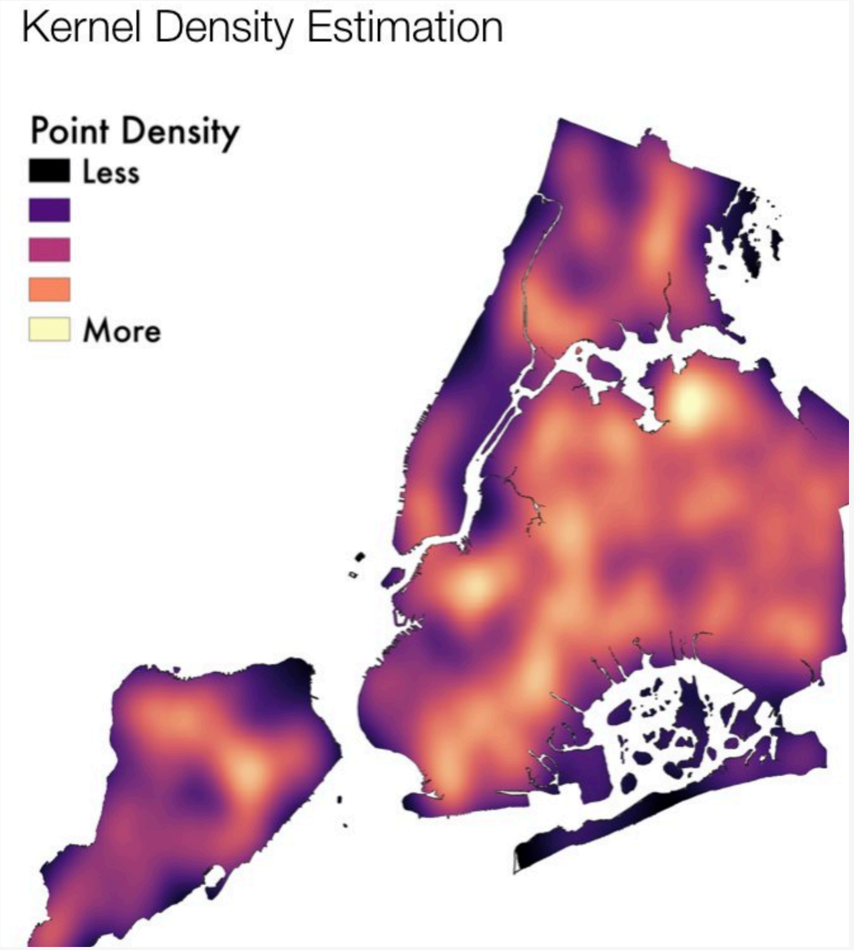


Zero areas properly zero, missing areas properly missing.

Care with Spatial Distribution

A random point-process

Care with Spatial Distribution



A heatmap derived from the random process

Care with Spatial Distribution

A formal test of significant hotspots

Example: Dorling Cartograms

Dorling Cartograms

```r
# install.packages("cartogram")
library(cartogram)
options(tigris_use_cache = TRUE)
```

Dorling Cartograms

```r
pop_names <- tribble(
    ~varname, ~clean,
    "B01003_001", "pop",
    "B01001B_001", "black",
    "B01001A_001", "white",
    "B01001H_001", "nh_white",
    "B01001I_001", "hispanic",
    "B01001D_001", "asian"
  )

pop_names
```

```
# A tibble: 6 × 2
  varname     clean
  <chr>       <chr>
1 B01003_001  pop
2 B01001B_001 black
3 B01001A_001 white
4 B01001H_001 nh_white
5 B01001I_001 hispanic
6 B01001D_001 asian
```

Dorling Cartograms

```r
library(tidycensus)
fips_pop <- get_acs(geography = "county",
                    variables = pop_names$varname,
                    cache_table = TRUE) |>
  left_join(pop_names, join_by(variable == varname)) |>
  mutate(variable = clean) |>
  select(-clean, -moe) |>
  pivot_wider(names_from = variable, values_from = estimate) |>
  rename(fips = GEOID, name = NAME) |>
  mutate(prop_pop = pop/sum(pop),
         prop_black = black/pop,
         prop_hisp = hispanic/pop,
         prop_white = white/pop,
         prop_nhwhite = nh_white/pop,
         prop_asian = asian/pop)

fips_map <- get_acs(geography = "county",
                    variables = "B01001_001",
                    geometry = TRUE,
                    shift_geo = FALSE,
                    cache_table = TRUE) |>
  select(GEOID, NAME, geometry) |>
```

Dorling Cartograms

```r
pop_cat_labels <- c("<5", as.character(seq(10, 95, 5)), "100")

counties_sf <- fips_map |>
  left_join(fips_pop, by = c("fips", "name")) |>
  mutate(black_disc = cut(prop_black*100,
                          breaks = seq(0, 100, 5),
                          labels = pop_cat_labels,
                          ordered_result = TRUE),
         hisp_disc = cut(prop_hisp*100,
                          breaks = seq(0, 100, 5),
                          labels = pop_cat_labels,
                          ordered_result = TRUE),
         nhwhite_disc = cut(prop_nhwhite*100,
                          breaks = seq(0, 100, 5),
                          labels = pop_cat_labels,
                          ordered_result = TRUE),
        asian_disc = cut(prop_asian*100,
                          breaks = seq(0, 100, 5),
                          labels = pop_cat_labels,
                          ordered_result = TRUE)) |>
  sf::st_transform(crs = 2163)
```

Dorling Cartograms

```r
counties_sf
```

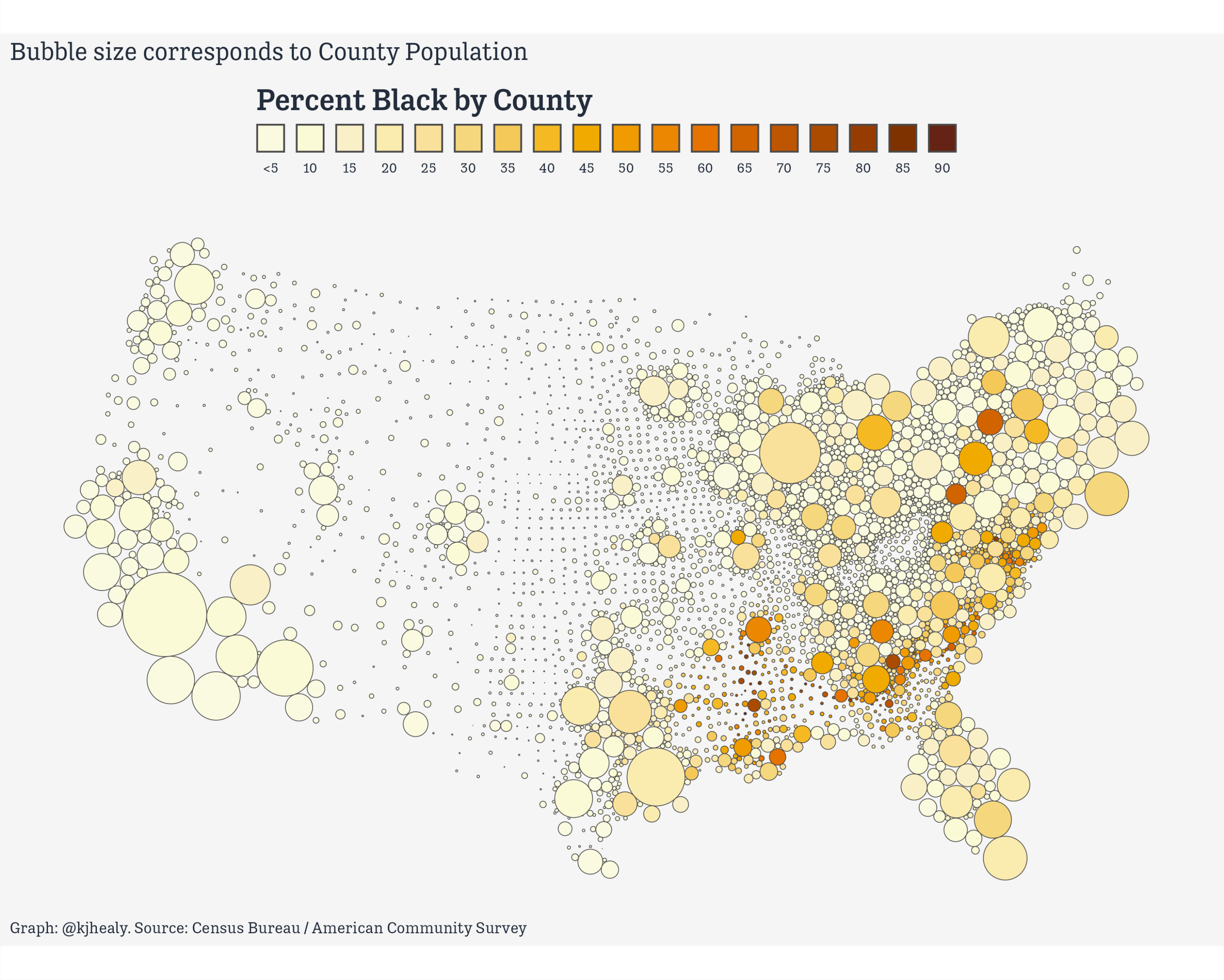
```
Simple feature collection with 3222 features and 18 fields
Geometry type: MULTIPOLYGON
Dimension:     XY
Bounding box:  xmin: -6433624 ymin: -2354597 xmax: 3667987 ymax: 3912355
Projected CRS: NAD27 / US National Atlas Equal Area
First 10 features:
   fips                  name white black asian nh_white hispanic    pop
1 01069 Houston County, Alabama 71260 29166   987    69420     3844 107040
2 01023 Choctaw County, Alabama  7180  5062    15     7162      143  12669
3 01005 Barbour County, Alabama 11309 11668   126    11084     1202  24877
4 01107 Pickens County, Alabama 10880  7506    10    10141      987  18925
5 01033 Colbert County, Alabama 44698  9185   214    44485     1837  57270
      prop_pop prop_black  prop_hisp prop_white prop_nhwhite   prop_asian
1 3.201244e-04  0.2724776 0.03591181  0.6657324    0.6485426 0.0092208520
2 3.788917e-05  0.3995580 0.01128739  0.5667377    0.5653169 0.0011839924
3 7.439962e-05  0.4690276 0.04831772  0.4545966    0.4455521 0.0050649194
4 5.659898e-05  0.3966182 0.05215324  0.5749009    0.5358520 0.0005284016
5 1.712773e-04  0.1603807 0.03207613  0.7804784    0.7767592 0.0037366859
                        geometry black_disc hisp_disc nhwhite_disc asian_disc
```

Dorling Cartograms

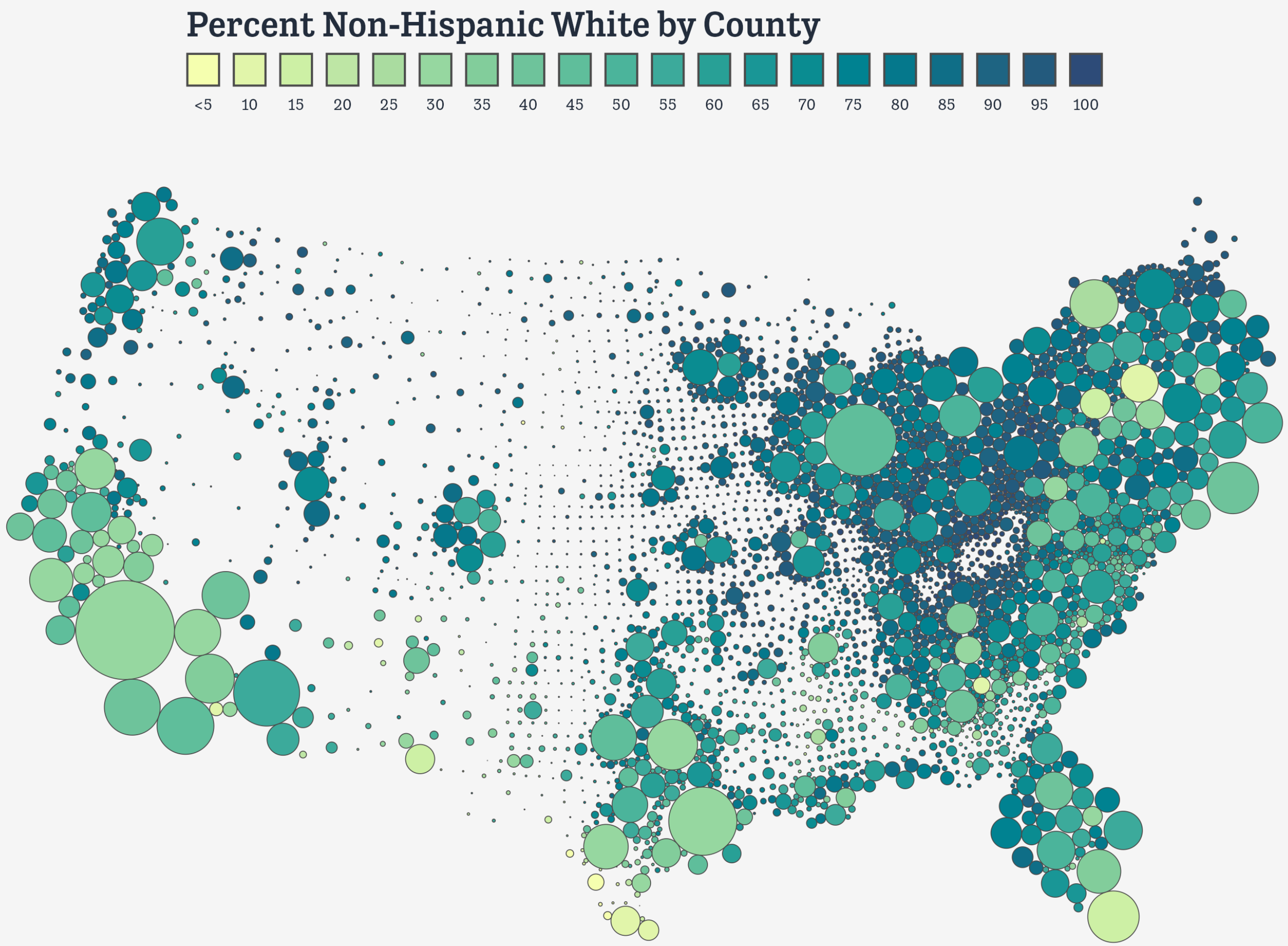
```
## Be patient
county_dorling <- cartogram_dorling(x = counties_sf,
    weight = "prop_pop",
    k = 0.2, itermax = 100)

out_black <- county_dorling |>
  filter(!str_detect(name, "Alaska|Hawaii|Puerto|Guam")) |>
  ggplot(aes(fill = black_disc)) +
  geom_sf(color = "grey30", size = 0.1) +
  coord_sf(crs = 2163, datum = NA) +
  scale_fill_discrete_sequential(palette = "YlOrBr",
                                 na.translate=FALSE) +
  guides(fill = guide_legend(title.position = "top",
                             label.position = "bottom",
                             nrow = 1)) +
  labs(
     subtitle = "Bubble size corresponds to County Population",
     caption = "Graph: @kjhealy. Source: Census Bureau / American Community Survey",
       fill = "Percent Black by County") +
  theme(legend.position = "top",
        legend.spacing.x = unit(0, "cm"),
        legend.title = element_text(size = rel(1.5), face = "bold"),
```

Dorling Cartograms

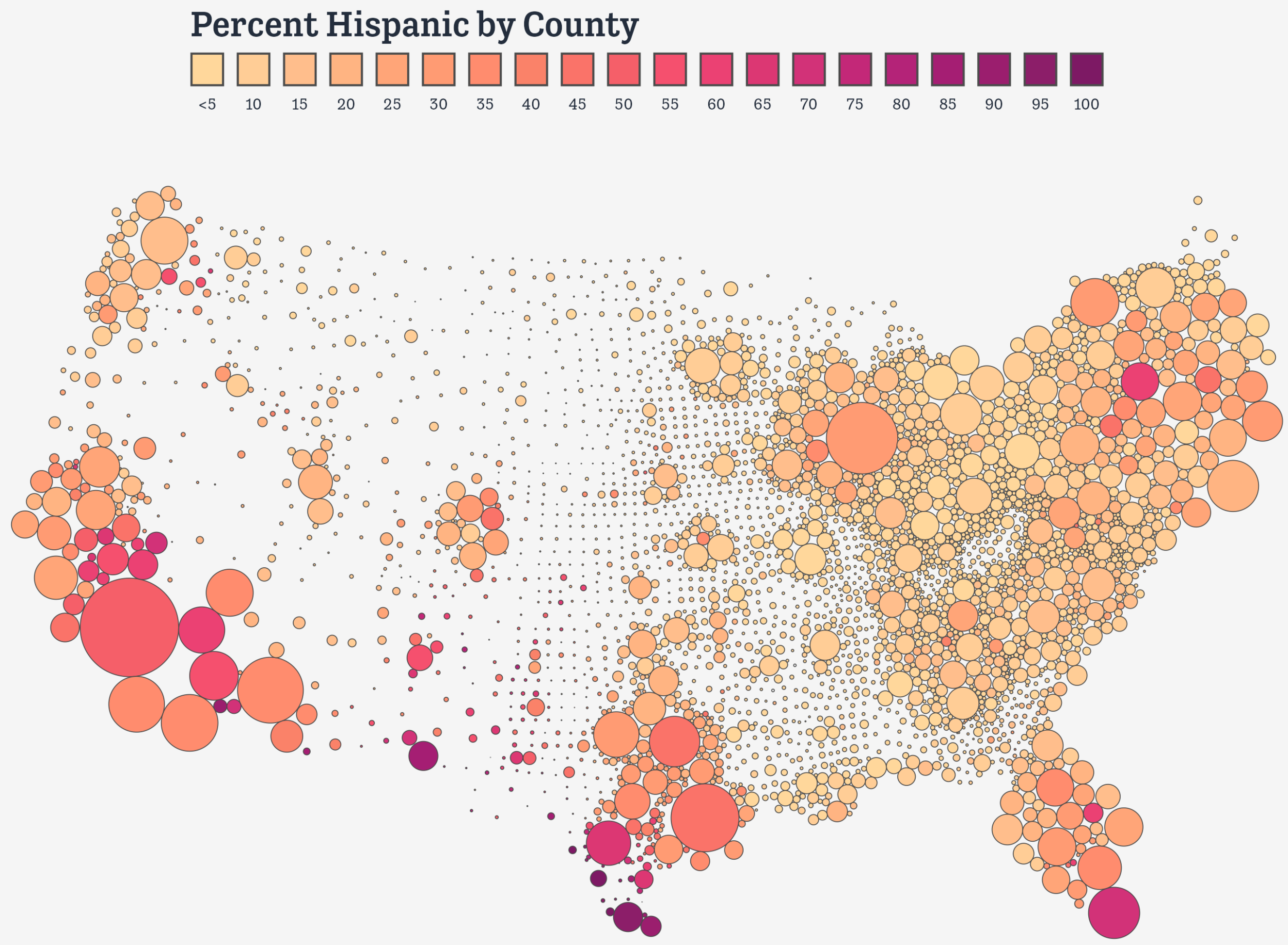
```
print(out_black)
```



```
print(out_white)
```



```
print(out_hispanic)
```



```
print(out_asian)
```