Data Visualization - 8. Polishing and Presenting Plots

Kieran Healy
Code Horizons

December 10, 2023

Polishing your plots and Presenting them

Load our packages

```
library(here)       # manage file paths
library(tidyverse)  # your friend and mine
library(socviz)     # data and some useful functions
library(ggrepel)    # Text and labels
library(colorspace) # luminance-balanced palettes
library(scales)      # scale adjustments and enhancements
library(ggforce)    # useful enhancements to ggplot
```

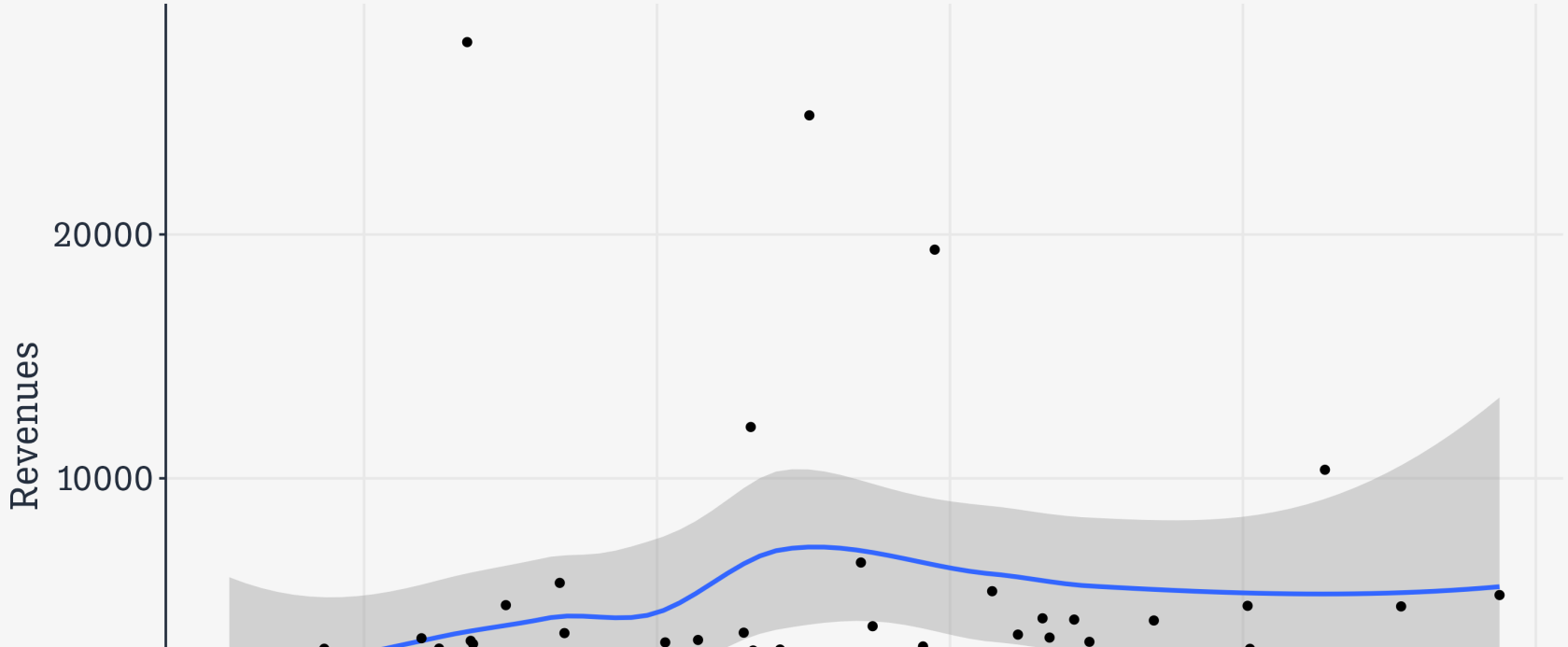
Piece by piece,
Layer by layer

Build your plots a piece at a time

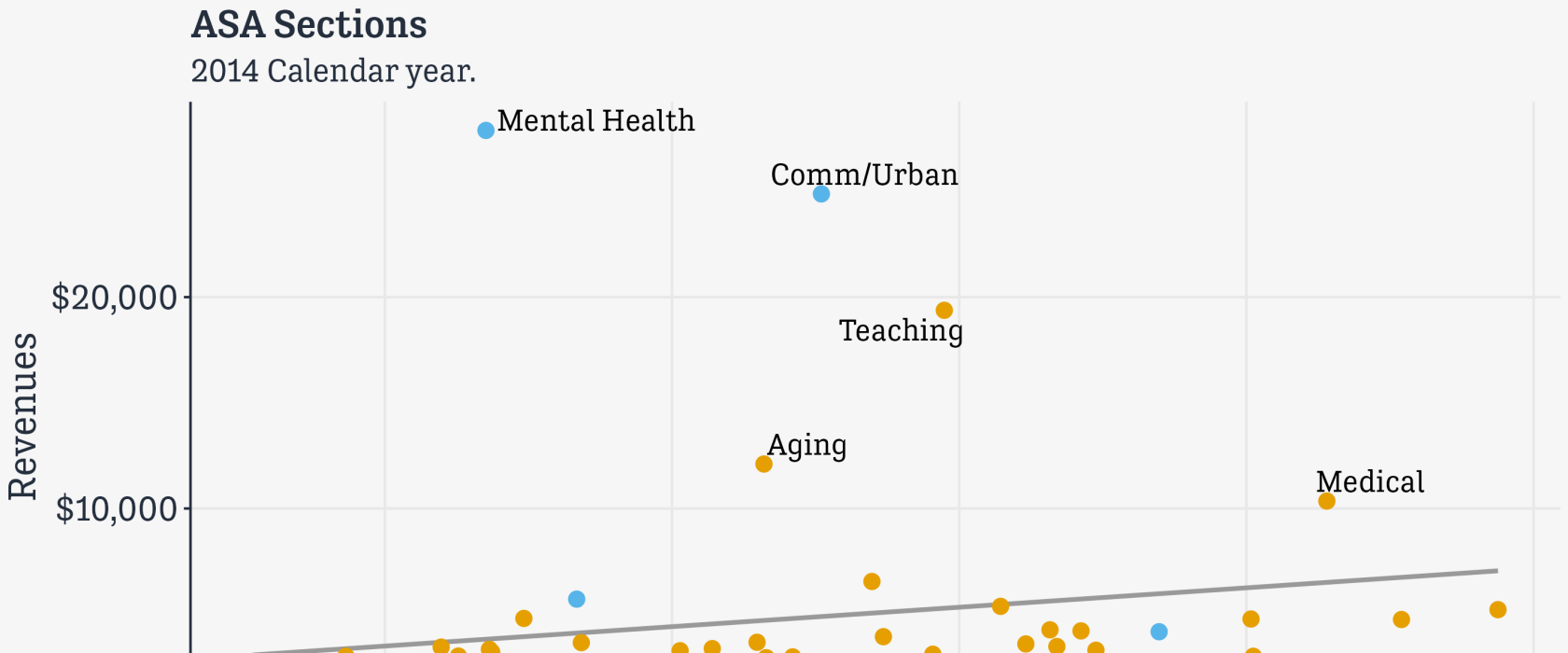
```
asasec ← as_tibble(asasec)
asasec
```

```
# A tibble: 572 × 9
   Section        Sname Beginning Revenues Expenses Ending Journal  Year Members
   <fct>          <fct>     <int>    <int>    <int>  <int> <fct>   <int>   <int>
 1 Aging and the… Aging     12752    12104    12007  12849 No       2005     598
 2 Alcohol, Drug… Alco…     11933     1144      400  12677 No       2005     301
 3 Altruism and … Altr…      1139     1862     1875   1126 No       2005      NA
 4 Animals and S… Anim…       473      820     1116    177 No       2005     209
 5 Asia/Asian Am… Asia       9056     2116     1710   9462 No       2005     365
 6 Body and Embo… Body       3408     1618     1920   3106 No       2005      NA
 7 Children and … Chil…      3692     3653     3713   3632 No       2005     418
 8 Coll Behavior… CBSM       8127     3470     2704   8893 No       2005     708
 9 Communication… CITA…     17093     4800     4804  17089 No       2005     301
10 Community and… Comm…     26598    24883    23379  28102 Yes      2005     721
# ℹ 562 more rows
```

Build your plots a piece at a time



Build your plots a piece at a time



Build your plots a piece at a time

```
1  asasec
```

```
# A tibble: 572 × 9
   Section        Sname Beginning Revenues Expenses Ending Journal  Year Members
   <fct>          <fct>     <int>    <int>    <int> <int> <fct>   <int>   <int>
 1 Aging and the… Aging     12752    12104    12007 12849 No       2005     598
 2 Alcohol, Drug… Alco…     11933     1144      400 12677 No       2005     301
 3 Altruism and … Altr…      1139     1862     1875  1126 No       2005      NA
 4 Animals and S… Anim…       473      820     1116   177 No       2005     209
 5 Asia/Asian Am… Asia        9056     2116     1710  9462 No       2005     365
 6 Body and Embo… Body        3408     1618     1920  3106 No       2005      NA
 7 Children and … Chil…       3692     3653     3713  3632 No       2005     418
 8 Coll Behavior… CBSM        8127     3470     2704  8893 No       2005     708
 9 Communication… CITA…      17093     4800     4804 17089 No       2005     301
10 Community and… Comm…      26598    24883    23379 28102 Yes      2005     721
```

Build your plots a piece at a time

```
1 asasec ▷
2   filter(Year == 2014)
```

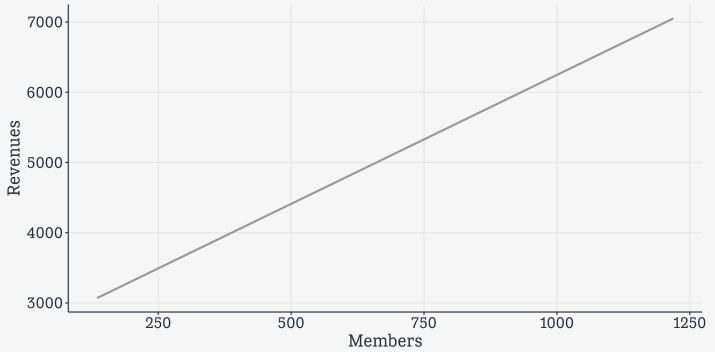
```
# A tibble: 52 × 9
   Section        Sname Beginning Revenues Expenses
Ending Journal  Year Members
   <fct>          <fct>     <int>    <int>    <int>
<int> <fct>   <int>   <int>
 1 Aging and the… Aging     12752    12104    12007
12849 No       2014     580
 2 Alcohol, Drug… Alco…     11933     1144      400
12677 No       2014     173
 3 Altruism and … Altr…      1139     1862     1875
1126 No       2014     318
 4 Animals and S… Anim…       473      820     1116
177 No       2014     154
 5 Asia/Asian Am… Asia        9056     2116     1710
9462 No       2014     336
 6 Body and Embo… Body        3408     1618     1920
3106 No       2014     312
 7 Children and … Chil…      3692     3653     3713
3632 No       2014     421
 8 Coll Behavior… CBSM        8127     3470     2704
8893 No       2014     835
 9 Communication… CITA…     17093     4800     4804
17089 No       2014     371
10 Community and… Comm…     26598    24883    23379
28102 Yes      2014     630
```

Build your plots a piece at a time

```
asasec ▷
  filter(Year == 2014) ▷
  ggplot(mapping = aes(x = Members,
                       y = Revenues,
                       label = Sname))
```

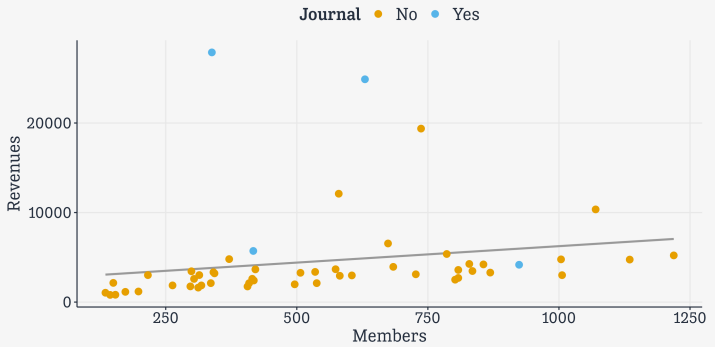
Build your plots a piece at a time

```
asasec ▷
  filter(Year == 2014) ▷
  ggplot(mapping = aes(x = Members,
                       y = Revenues,
                       label = Sname)) +
  geom_smooth(method = "lm",
              se = FALSE,
              color = "gray60")
```



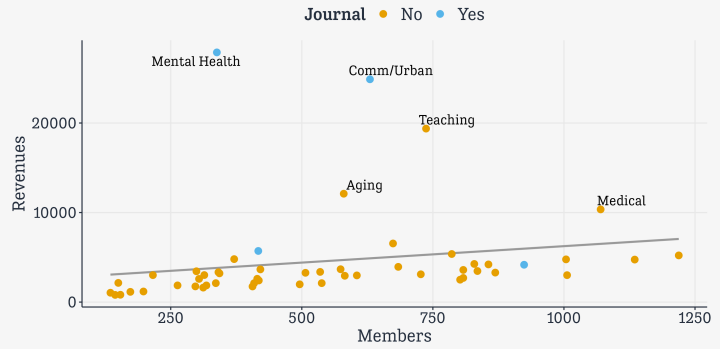
Build your plots a piece at a time

```
asasec ▷
  filter(Year == 2014) ▷
  ggplot(mapping = aes(x = Members,
                       y = Revenues,
                       label = Sname)) +
  geom_smooth(method = "lm",
              se = FALSE,
              color = "gray60") +
  geom_point(mapping = aes(color = Journal),
             size = rel(3))
```



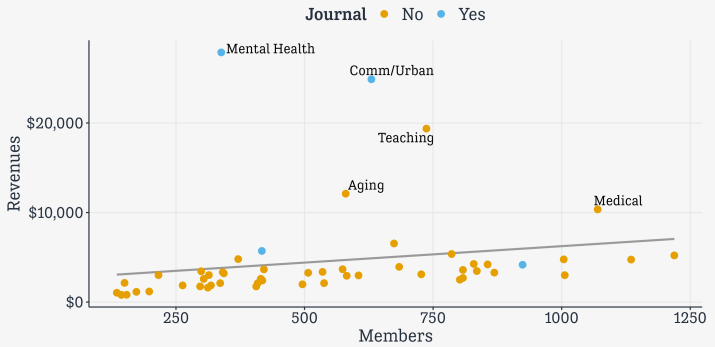
Build your plots a piece at a time

```
asasec ▷
  filter(Year == 2014) ▷
  ggplot(mapping = aes(x = Members,
                       y = Revenues,
                       label = Sname)) +
  geom_smooth(method = "lm",
              se = FALSE,
              color = "gray60") +
  geom_point(mapping = aes(color = Journal),
             size = rel(3)) +
  geom_text_repel(data=subset(asasec,
                    Year == 2014 &
                    Revenues > 7000),
                    size = rel(5),
                    mapping =
                    aes(family = "Tenso Slide"))
```



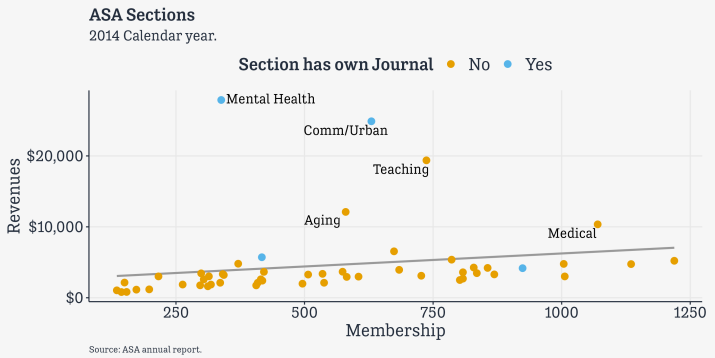
Build your plots a piece at a time

```
asasec ▷
  filter(Year == 2014) ▷
  ggplot(mapping = aes(x = Members,
                       y = Revenues,
                       label = Sname)) +
  geom_smooth(method = "lm",
              se = FALSE,
              color = "gray60") +
  geom_point(mapping = aes(color = Journal),
             size = rel(3)) +
  geom_text_repel(data=subset(asasec,
                    Year == 2014 &
                    Revenues > 7000),
                    size = rel(5),
                    mapping =
                    aes(family = "Tenso Slide")) +
  scale_y_continuous(labels =
                       scales::label_dollar())
```



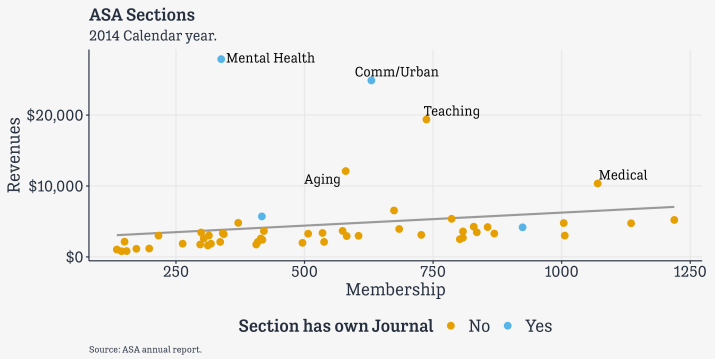
Build your plots a piece at a time

```
asasec ▷
  filter(Year == 2014) ▷
  ggplot(mapping = aes(x = Members,
                       y = Revenues,
                       label = Sname)) +
  geom_smooth(method = "lm",
              se = FALSE,
              color = "gray60") +
  geom_point(mapping = aes(color = Journal),
             size = rel(3)) +
  geom_text_repel(data=subset(asasec,
                  Year == 2014 &
                  Revenues > 7000),
                  size = rel(5),
                  mapping =
                  aes(family = "Tenso Slide")) +
  scale_y_continuous(labels =
                       scales::label_dollar()) +
  labs(x="Membership", y="Revenues",
       color = "Section has own Journal",
       title = "ASA Sections",
       subtitle = "2014 Calendar year.",
       caption = "Source: ASA annual report.")
```



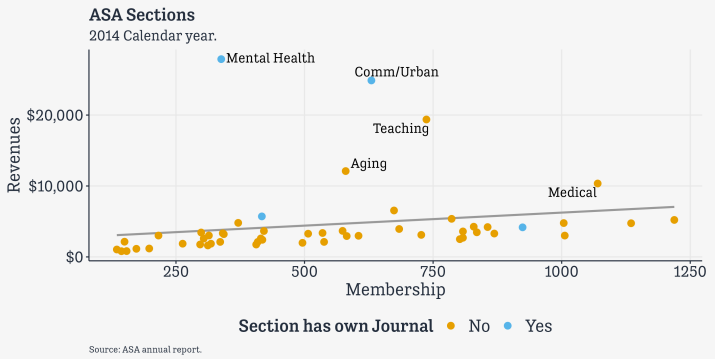
Build your plots a piece at a time

```
asasec ▷
  filter(Year == 2014) ▷
  ggplot(mapping = aes(x = Members,
                       y = Revenues,
                       label = Sname)) +
  geom_smooth(method = "lm",
              se = FALSE,
              color = "gray60") +
  geom_point(mapping = aes(color = Journal),
             size = rel(3)) +
  geom_text_repel(data=subset(asasec,
                    Year == 2014 &
                    Revenues > 7000),
                    size = rel(5),
                    mapping =
                    aes(family = "Tenso Slide")) +
  scale_y_continuous(labels =
                       scales::label_dollar()) +
  labs(x="Membership", y="Revenues",
        color = "Section has own Journal",
        title = "ASA Sections",
        subtitle = "2014 Calendar year.",
        caption = "Source: ASA annual report.") +
```



Build your plots a piece at a time

```
asasec ▷
  filter(Year == 2014) ▷
  ggplot(mapping = aes(x = Members,
                       y = Revenues,
                       label = Sname)) +
  geom_smooth(method = "lm",
              se = FALSE,
              color = "gray60") +
  geom_point(mapping = aes(color = Journal),
             size = rel(3)) +
  geom_text_repel(data=subset(asasec,
                    Year == 2014 &
                    Revenues > 7000),
                    size = rel(5),
                    mapping =
                    aes(family = "Tenso Slide")) +
  scale_y_continuous(labels =
                       scales::label_dollar()) +
  labs(x="Membership", y="Revenues",
        color = "Section has own Journal",
        title = "ASA Sections",
        subtitle = "2014 Calendar year.",
        caption = "Source: ASA annual report.") +
```



More about Scales

Working with `color` and `fill` scales

scale_<MAPPING>_<KIND>()

.large[Scale functions control the display of the variables they map. So to change the colors for `color` or `fill` mappings, you adjust the corresponding `scale_` function, not the `theme()` function.]

.large[`ggplot` has several color palettes built in. A variety of packages provide others.]

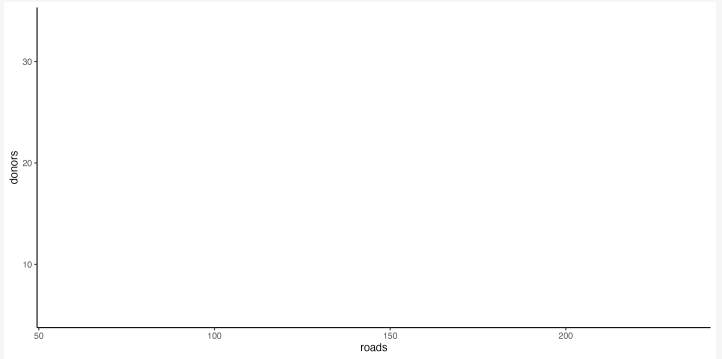
You can specify scales manually

```
1  organdata
```

```
# A tibble: 238 × 21
   country   year       donors   pop pop_dens   gdp
gdp_lag health health_lag
   <chr>     <date>      <dbl> <int>    <dbl> <int>
<int>  <dbl>      <dbl>
 1 Australia NA          NA    17065    0.220 16774
16591   1300       1224
 2 Australia 1991-01-01  12.1  17284    0.223 17171
16774   1379       1300
 3 Australia 1992-01-01  12.4  17495    0.226 17914
17171   1455       1379
 4 Australia 1993-01-01  12.5  17667    0.228 18883
17914   1540       1455
 5 Australia 1994-01-01  10.2  17855    0.231 19849
18883   1626       1540
 6 Australia 1995-01-01  10.2  18072    0.233 21079
19849   1737       1626
 7 Australia 1996-01-01  10.6  18311    0.237 21923
21079   1846       1737
 8 Australia 1997-01-01  10.3  18518    0.239 22961
21923   1948       1846
 9 Australia 1998-01-01  10.5  18711    0.242 24148
22961   2077       1948
10 Australia 1999-01-01   8.67 18926    0.244 25445
24148   2231       2077
```

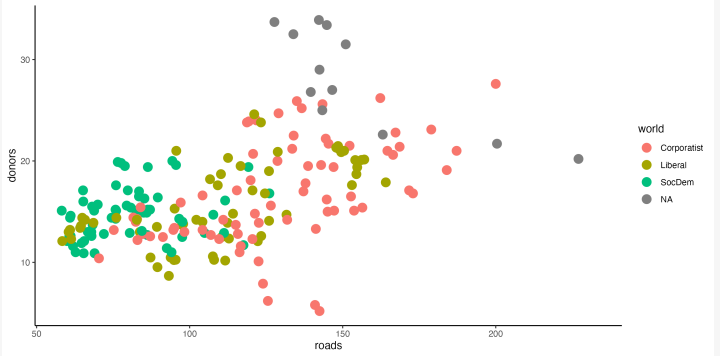
You can specify scales manually

```
organdata ▷
  ggplot(mapping = aes(x = roads,
                       y = donors,
                       color = world))
```



You can specify scales manually

```
organdata ▷
  ggplot(mapping = aes(x = roads,
                       y = donors,
                       color = world)) +
  geom_point(size = 4)
```



You can specify scales manually

```
organdata ▷
  ggplot(mapping = aes(x = roads,
                       y = donors,
                       color = world)) +
  geom_point(size = 4) →
  p
```

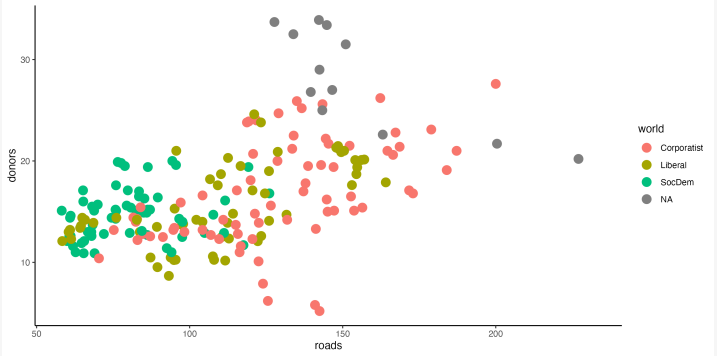
You can specify scales manually

```
organdata ▷
  ggplot(mapping = aes(x = roads,
                       y = donors,
                       color = world)) +
  geom_point(size = 4) →
  p
```

```
p ← organdata ▷
  ggplot(mapping = aes(x = roads,
                       y = donors,


                       color = world)) +
  geom_point(size = 4)
```

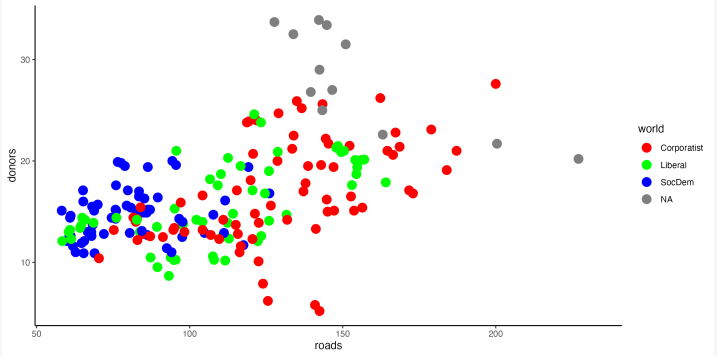
You can always specify scales manually

```
1  p
```



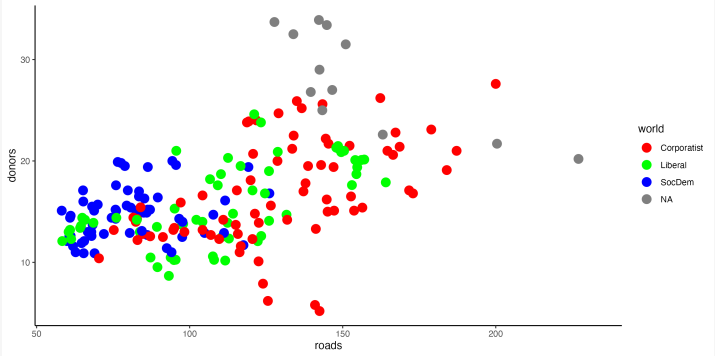
You can always specify scales manually

```
1  p +
2    scale_color_manual(
3    values = c("red", "green", "blue"))
```



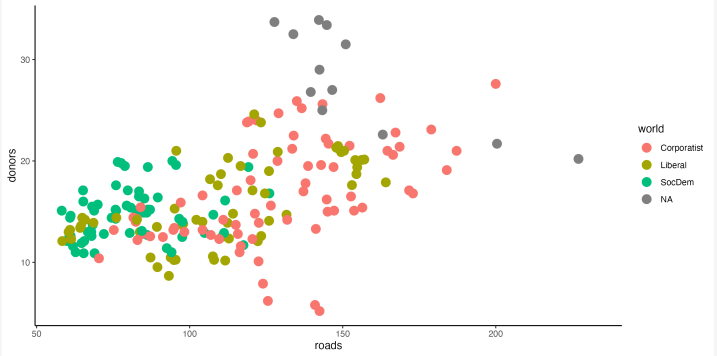
You can always specify scales manually

```
1  p +
2    scale_color_manual(
3    values = c("red", "green", "blue"))
```



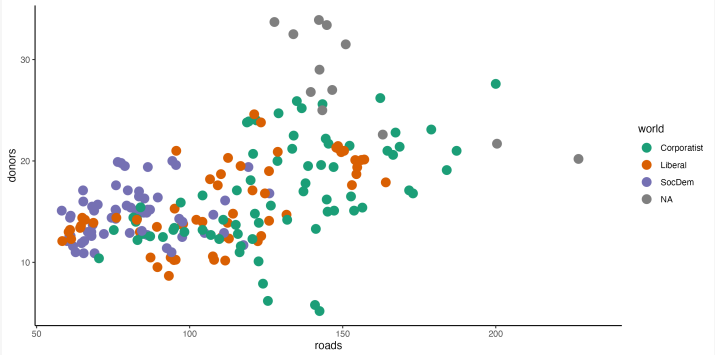
You can always specify scales manually

```
1  p
```



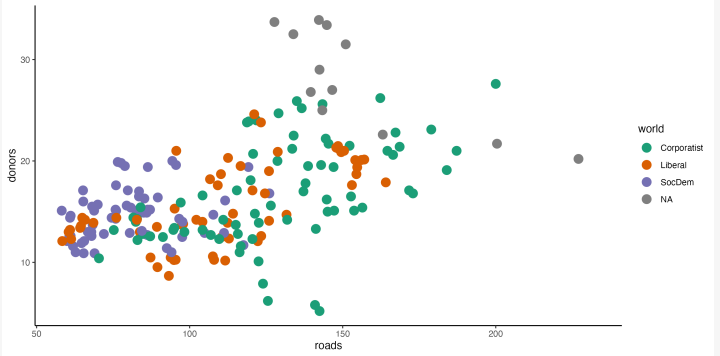
You can always specify scales manually

```
1  p +
2    scale_color_manual(
3    values = c("#1B9E77","#D95F02", "#7570B3"))
```



You can always specify scales manually

```
p +
  scale_color_manual(
  values = c("#1B9E77","#D95F02", "#7570B3"))
```



You can specify scales manually

```
colkey ← c("Corporatist" = "pink",
           "Liberal" = "goldenrod",
           "SocDem" = "firebrick")
```

You can specify scales manually

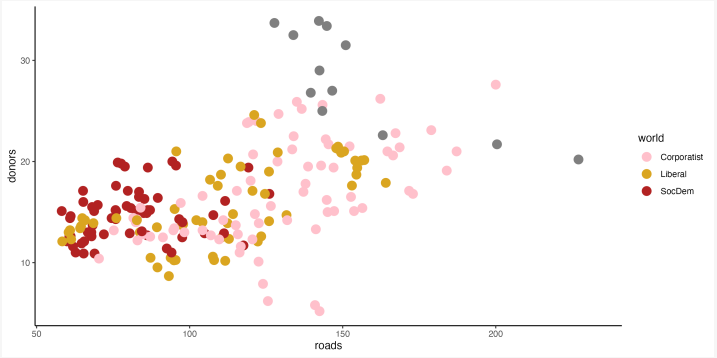
```
colkey ← c("Corporatist" = "pink",
           "Liberal" = "goldenrod",
           "SocDem" = "firebrick")
colkey
```

```
Corporatist     Liberal      SocDem
     "pink" "goldenrod" "firebrick"
```

You can specify scales manually

```
colkey ← c("Corporatist" = "pink",
           "Liberal" = "goldenrod",
           "SocDem" = "firebrick")
colkey

p + scale_color_manual(
  values = colkey)
```

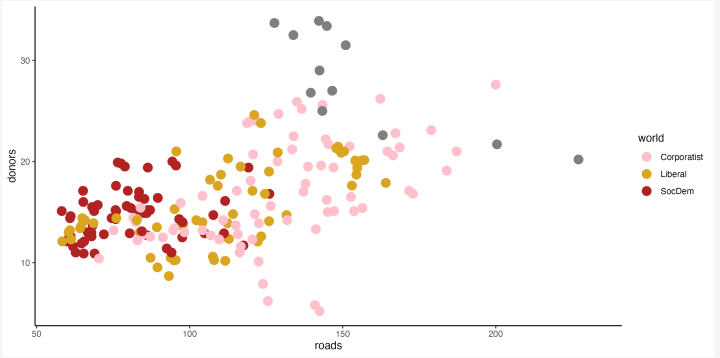
```
Corporatist     Liberal      SocDem
     "pink" "goldenrod" "firebrick"
```



You can specify scales manually

```
colkey ← c("Corporatist" = "pink",
           "Liberal" = "goldenrod",
           "SocDem" = "firebrick")
colkey

p + scale_color_manual(
  values = colkey)
```

```
Corporatist     Liberal      SocDem
     "pink" "goldenrod" "firebrick"
```



Use balanced palettes

E.g., the `RColorBrewer` Palettes

These are available through the
`scale_color-brewer()` and
`scale_fill_brewer()` funct
as independently.

See the palettes with
`RColorBrewer::display.b`

Qualitative palettes

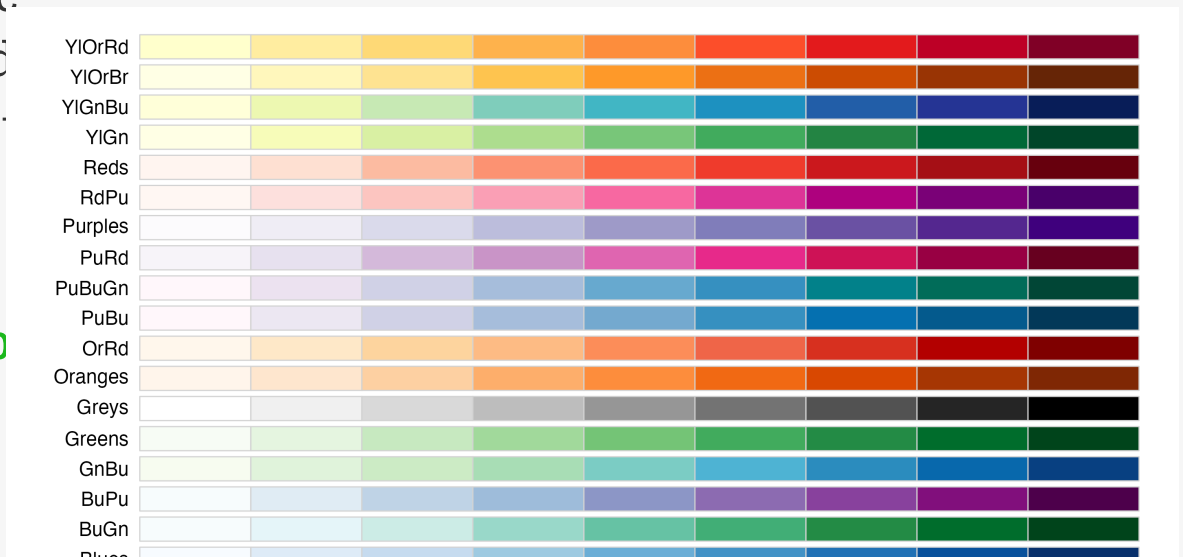
Use balanced palettes

E.g., the `RColorBrewer` Palettes

These are available through the
`scale_color-brewer()` and
`scale_fill_brewer()` func
as independently.

See the palettes with
`RColorBrewer::display.b`

Sequential palettes



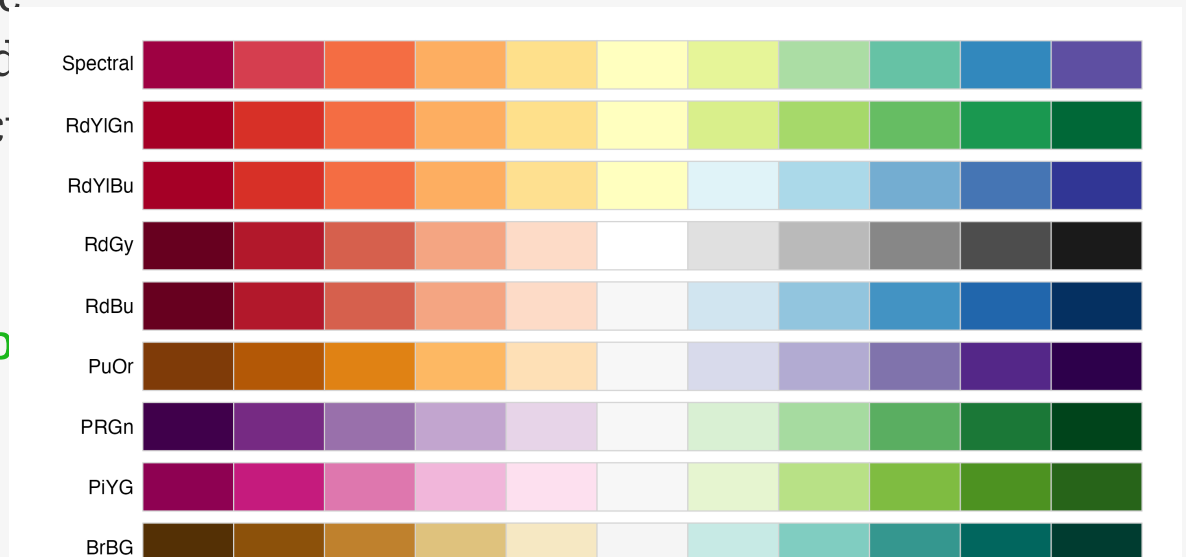
Use balanced palettes

E.g., the `RColorBrewer` Palettes

These are available through the
`scale_color-brewer()` and
`scale_fill_brewer()` func
as independently.

See the palettes with
`RColorBrewer::display.b`

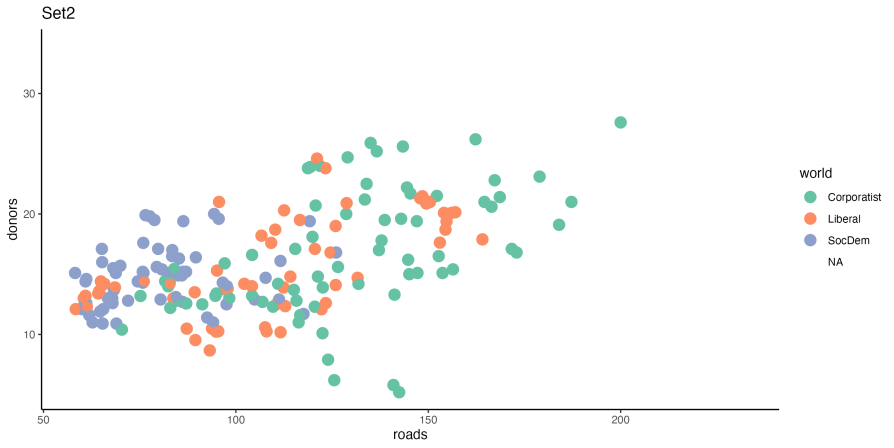
Diverging palettes

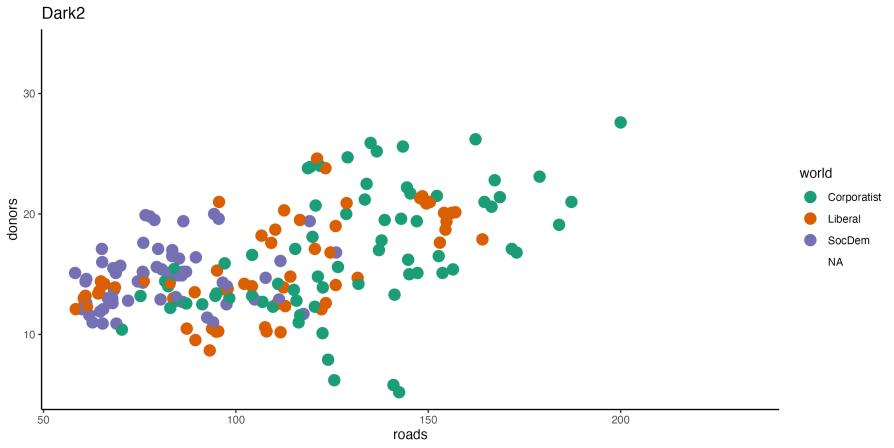


Qualitative Brewer Palettes

```
p + geom_point(size = 2) +
  scale_color_brewer(palette = "Set2") +
  labs(title = "Set2")

p + geom_point(size = 2) +
  scale_color_brewer(palette = "Pastel2") +
  labs(title = "Pastel2")

p + geom_point(size = 2) +

  scale_color_brewer(palette = "Dark2") +
  labs(title = "Dark2")

p + geom_point(size = 2) +
  scale_color_brewer(palette = "Accent") +
  labs(title = "Accent")
```

Some color palettes



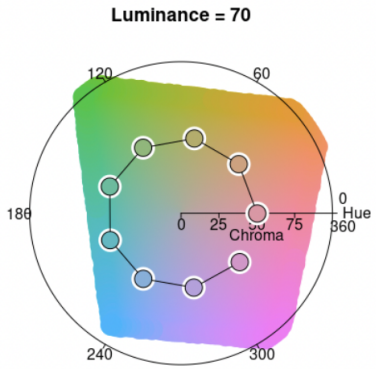


The colorspace package has more

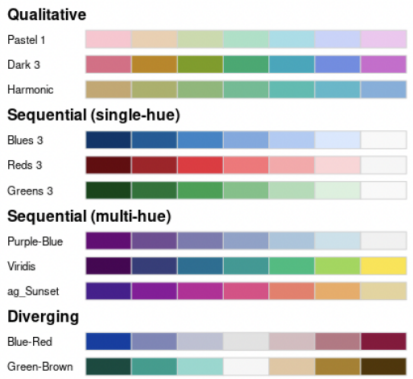
colorspace 2.0-4 Get started Articles Reference Changelog Contact

A Toolbox for Manipulating and Assessing Colors and Palettes

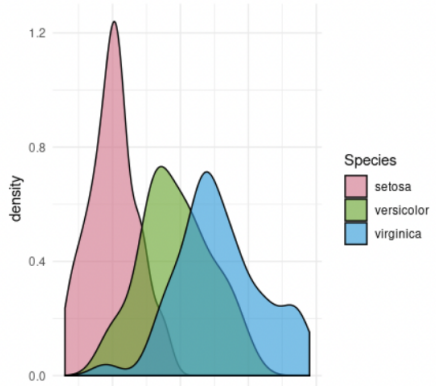
Color spaces



HCL-based palettes



ggplot2 scales



Links

Download from CRAN at
https://cloud.r-project.org/package=colorspace

Report a bug at
https://colorspace.R-Forge.R-project.org/contact.html

Online color apps at
https://hclwizard.org/

License

BSD_3_clause + file LICENSE

Citation

Citing colorspace


Qualitative
Pastel 1
Dark 2
Dark 3
Set 2
Set 3
Warm
Cold
Harmonic
Dynamic
Sequential (single-hue)
Grays
Light Grays
Reds 2
Reds 3
Greens 2
Greens 3
Oslo
Sequential (multi-hue)
Purple-Blue
Red-Purple
Red-Blue
Purple-Orang
Purple-Yello
Blue-Yellow
Green-Yello
Terrain 2
Viridis
Plasma
Inferno
Rocket
Mako
Dark Mint
Mint
BluGrn
Teal
TealGrn
Emrld
BluYl
BurgYl
RedOr
OrYel
Purp
PurpOr
Sunset
Magenta
SunsetDark
ag_Sunset
BrwnYl
YlOrRd
YlOrBr
OrRd
RdPu
PuRd
Purples
PuBuGn
PuBu
Greens
BuGn
GnBu
BuPu
Blues
Lajolla
Turku
Hawaii
Blue-Red 3
Red-Green
Purple-Gree
Purple-Brow
Green-Brow
Blue-Yellow
Blue-Yellow
Green-Oran
Cyan-Mage
Tropic
Broc
Cork
Vik

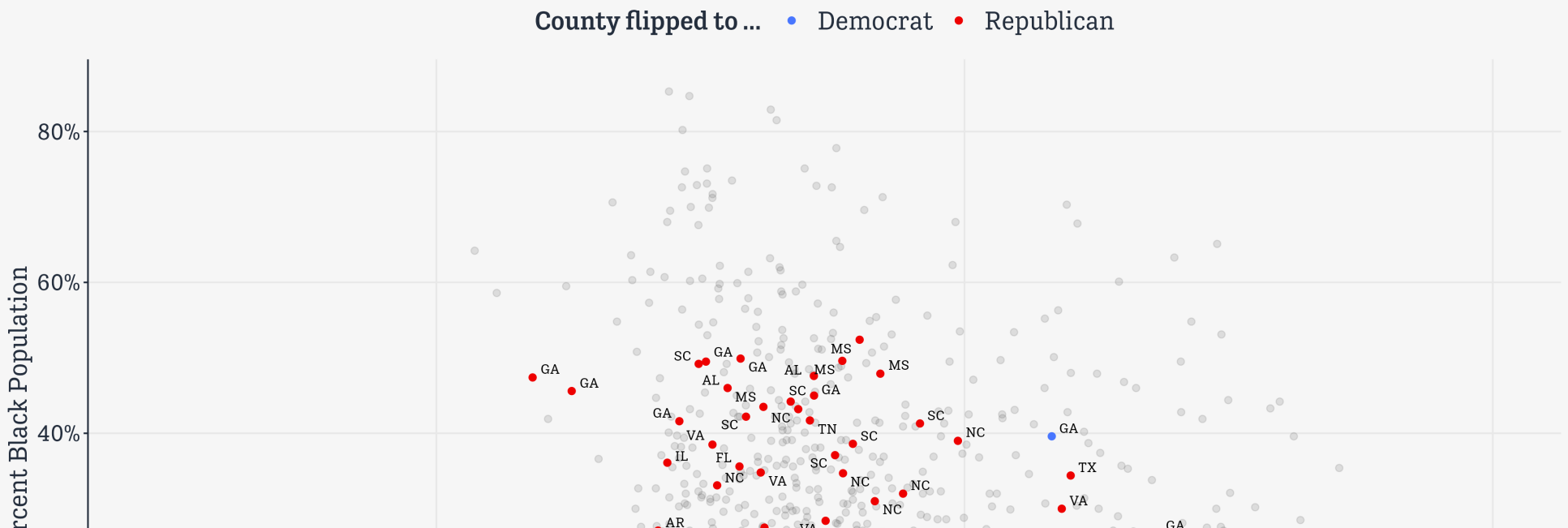
The `colorspace` function convention

`scale_<MAPPING>_<KIND><COLORSCALE>()`

```
scale_color_binned_diverging()
scale_color_binned_qualitative()
scale_color_binned_sequential()
scale_color_continuous_diverging()
scale_color_continuous_qualitative()
scale_color_continuous_sequential()
scale_fill_binned_diverging()
scale_fill_binned_divergingx()
scale_fill_binned_qualitative()
scale_fill_binned_sequential()
scale_fill_continuous_divergi
scale_fill_continuous_qualita
```

Layer color and text to your advantage

Flipped counties, 2016

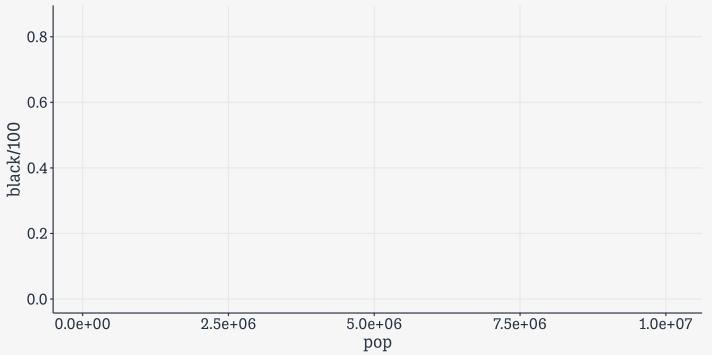


We know how to build this

```
# Brighter Blue and Red
party_colors ← c("royalblue1", "red2")
```

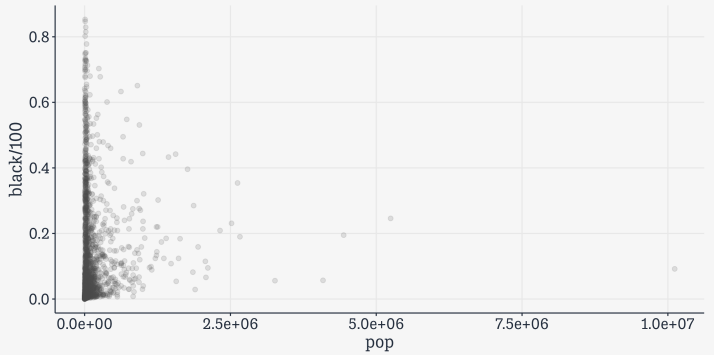
We know how to build this

```
# Brighter Blue and Red
party_colors ← c("royalblue1", "red2")

ggplot(data = subset(county_data,
                     flipped == "No"),
       mapping = aes(x = pop,
                     y = black/100))
```



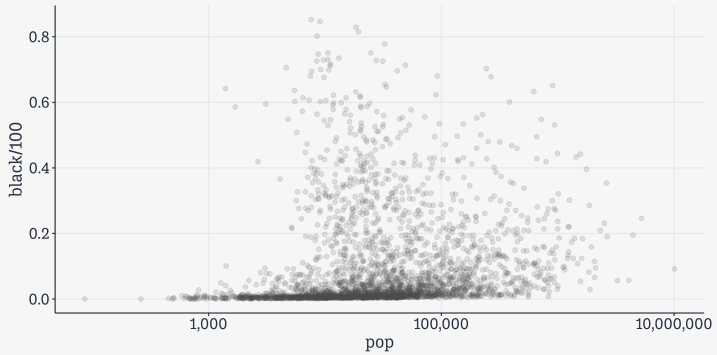
We know how to build this

```r
# Brighter Blue and Red
party_colors <- c("royalblue1", "red2")

ggplot(data = subset(county_data,
                     flipped == "No"),
       mapping = aes(x = pop,
                     y = black/100)) +
  geom_point(alpha = 0.15, color = "gray30",
             size = rel(2))
```



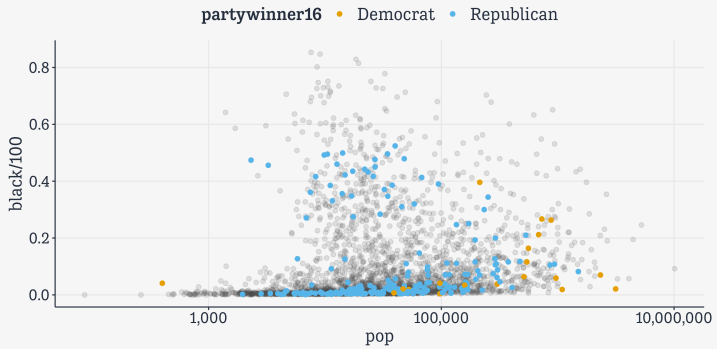
We know how to build this

```r
# Brighter Blue and Red
party_colors <- c("royalblue1", "red2")

ggplot(data = subset(county_data,
                     flipped == "No"),
       mapping = aes(x = pop,
                     y = black/100)) +
  geom_point(alpha = 0.15, color = "gray30",
             size = rel(2)) +
  scale_x_log10(labels = label_comma())
```



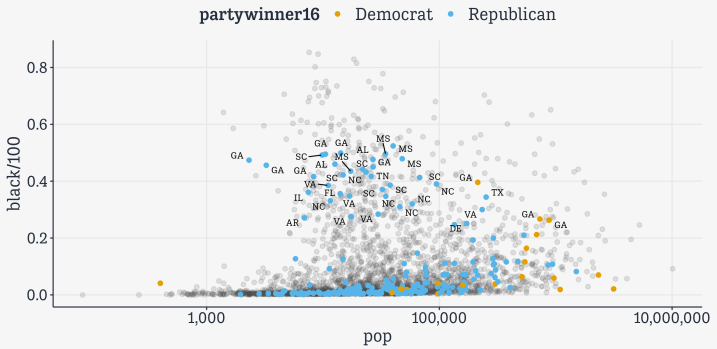
We know how to build this

```r
# Brighter Blue and Red
party_colors <- c("royalblue1", "red2")

ggplot(data = subset(county_data,
                     flipped == "No"),
       mapping = aes(x = pop,
                     y = black/100)) +
  geom_point(alpha = 0.15, color = "gray30",
             size = rel(2)) +
  scale_x_log10(labels = label_comma()) +
  geom_point(data = subset(county_data,
                      flipped == "Yes"),
             mapping = aes(x = pop, y = black/100,
                           color = partywinner16),
             size = rel(2))
```



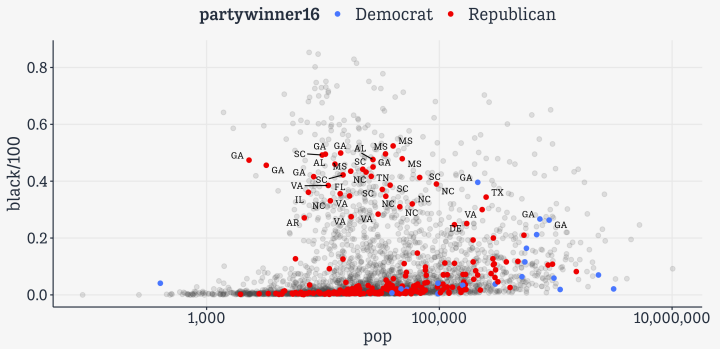
We know how to build this

```r
# Brighter Blue and Red
party_colors <- c("royalblue1", "red2")

ggplot(data = subset(county_data,
                     flipped == "No"),
       mapping = aes(x = pop,
                     y = black/100)) +
  geom_point(alpha = 0.15, color = "gray30",
             size = rel(2)) +
  scale_x_log10(labels = label_comma()) +
  geom_point(data = subset(county_data,
                      flipped == "Yes"),
             mapping = aes(x = pop, y = black/100,
                           color = partywinner16),
             size = rel(2)) +
  geom_text_repel(data = subset(county_data,
              flipped == "Yes" & black  > 25),
                  mapping = aes(x = pop,
                    y = black/100, label = state,
                    family = "Tenso Slide",
                    face = "bold"), size = rel(3.5))
```



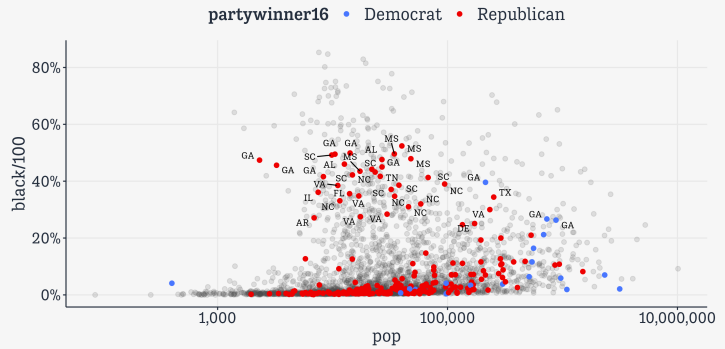
We know how to build this

```
# Brighter Blue and Red
party_colors ← c("royalblue1", "red2")

ggplot(data = subset(county_data,
                     flipped == "No"),
       mapping = aes(x = pop,
                     y = black/100)) +
  geom_point(alpha = 0.15, color = "gray30",
             size = rel(2)) +
  scale_x_log10(labels = label_comma()) +
  geom_point(data = subset(county_data,
                      flipped == "Yes"),
             mapping = aes(x = pop, y = black/100,
                           color = partywinner16),
             size = rel(2)) +
  geom_text_repel(data = subset(county_data,
              flipped == "Yes" & black  > 25),
                  mapping = aes(x = pop,
                    y = black/100, label = state,
                    family = "Tenso Slide",
                    face = "bold"), size = rel(3.5)) +
  scale_color_manual(values = party_colors)
```



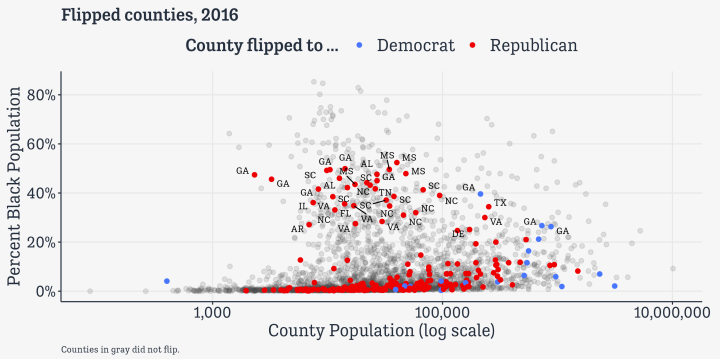
We know how to build this

```
# Brighter Blue and Red
party_colors ← c("royalblue1", "red2")

ggplot(data = subset(county_data,
                     flipped == "No"),
       mapping = aes(x = pop,
                     y = black/100)) +
  geom_point(alpha = 0.15, color = "gray30",
             size = rel(2)) +
  scale_x_log10(labels = label_comma()) +
  geom_point(data = subset(county_data,
                      flipped == "Yes"),
             mapping = aes(x = pop, y = black/100,
                           color = partywinner16),
             size = rel(2)) +
  geom_text_repel(data = subset(county_data,
              flipped == "Yes" & black  > 25),
                  mapping = aes(x = pop,
                    y = black/100, label = state,
                    family = "Tenso Slide",
                    face = "bold"), size = rel(3.5)) +
  scale_color_manual(values = party_colors) +
  scale_y_continuous(labels = label_percent())
```



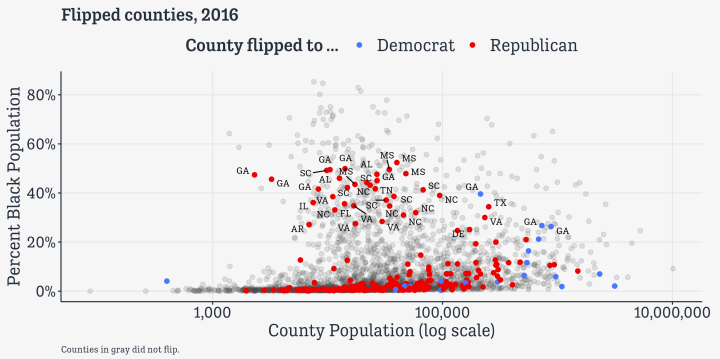
We know how to build this

```
# Brighter Blue and Red
party_colors <- c("royalblue1", "red2")

ggplot(data = subset(county_data,
                     flipped == "No"),
       mapping = aes(x = pop,
                     y = black/100)) +
  geom_point(alpha = 0.15, color = "gray30",
             size = rel(2)) +
  scale_x_log10(labels = label_comma()) +
  geom_point(data = subset(county_data,
                      flipped == "Yes"),
             mapping = aes(x = pop, y = black/100,
                           color = partywinner16),
             size = rel(2)) +
  geom_text_repel(data = subset(county_data,
              flipped == "Yes" & black  > 25),
                  mapping = aes(x = pop,
                    y = black/100, label = state,
                    family = "Tenso Slide",
                    face = "bold"), size = rel(3.5)) +
  scale_color_manual(values = party_colors) +
  scale_y_continuous(labels = label_percent()) +
```

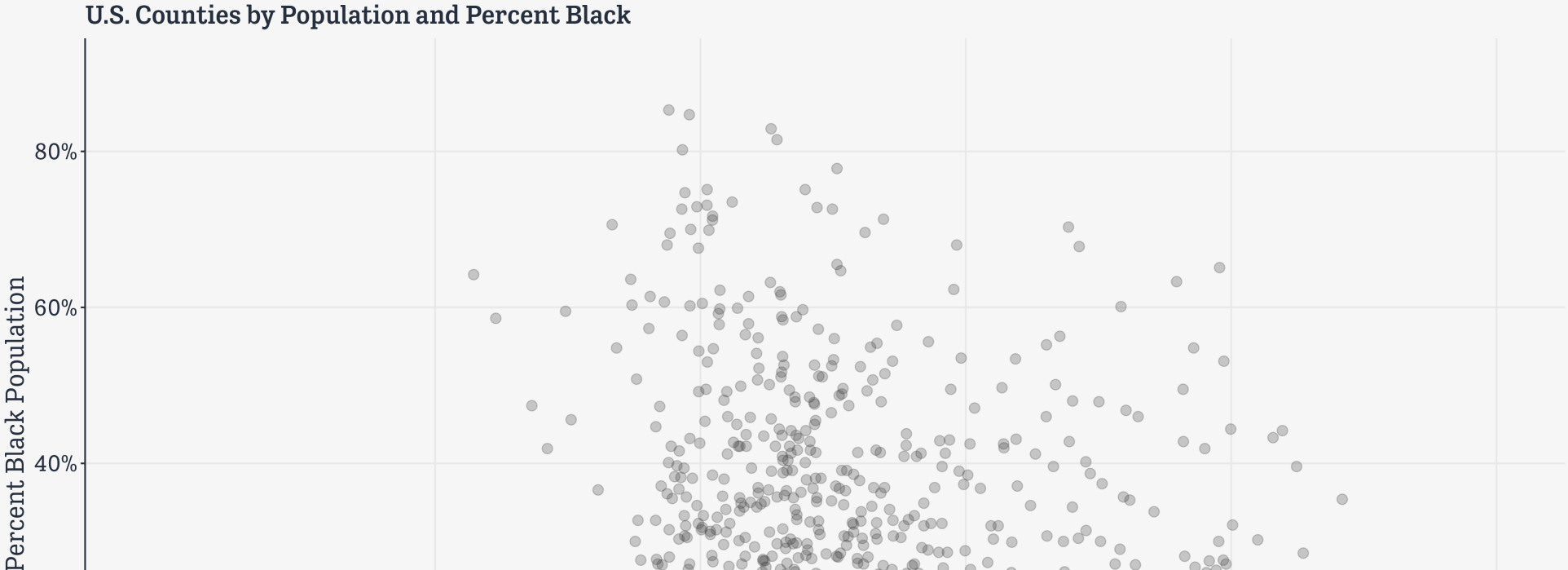


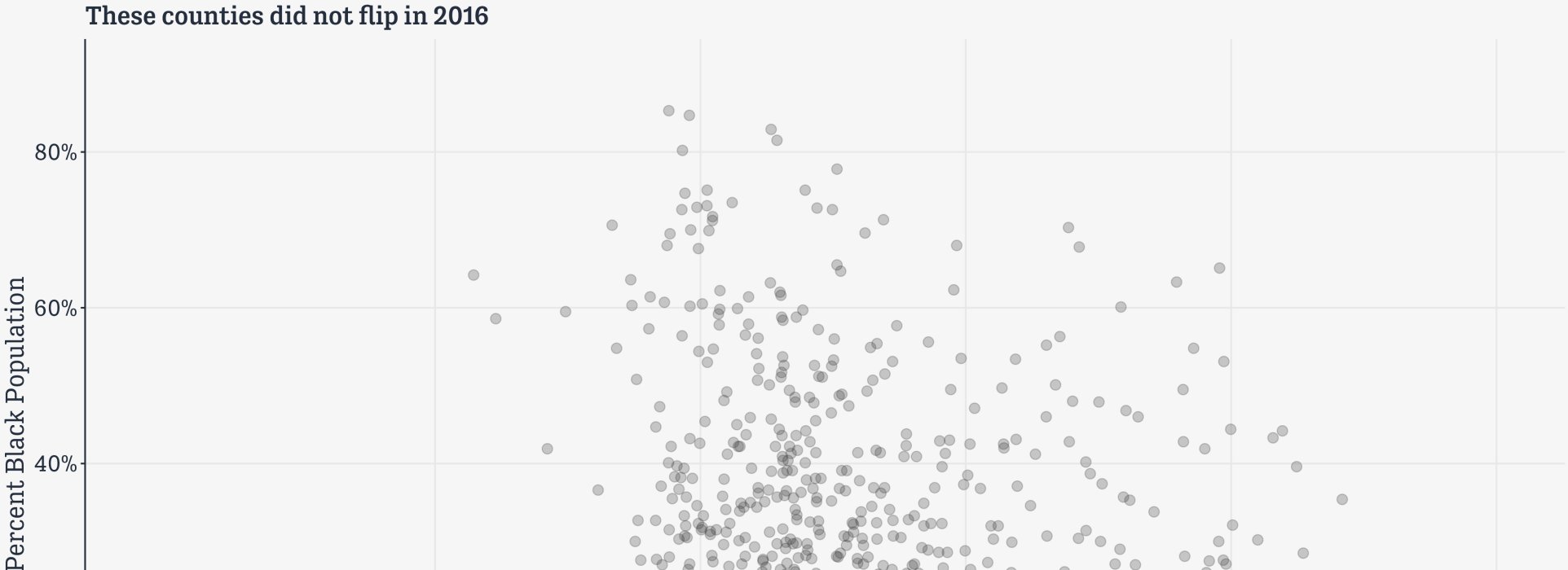
We know how to build this

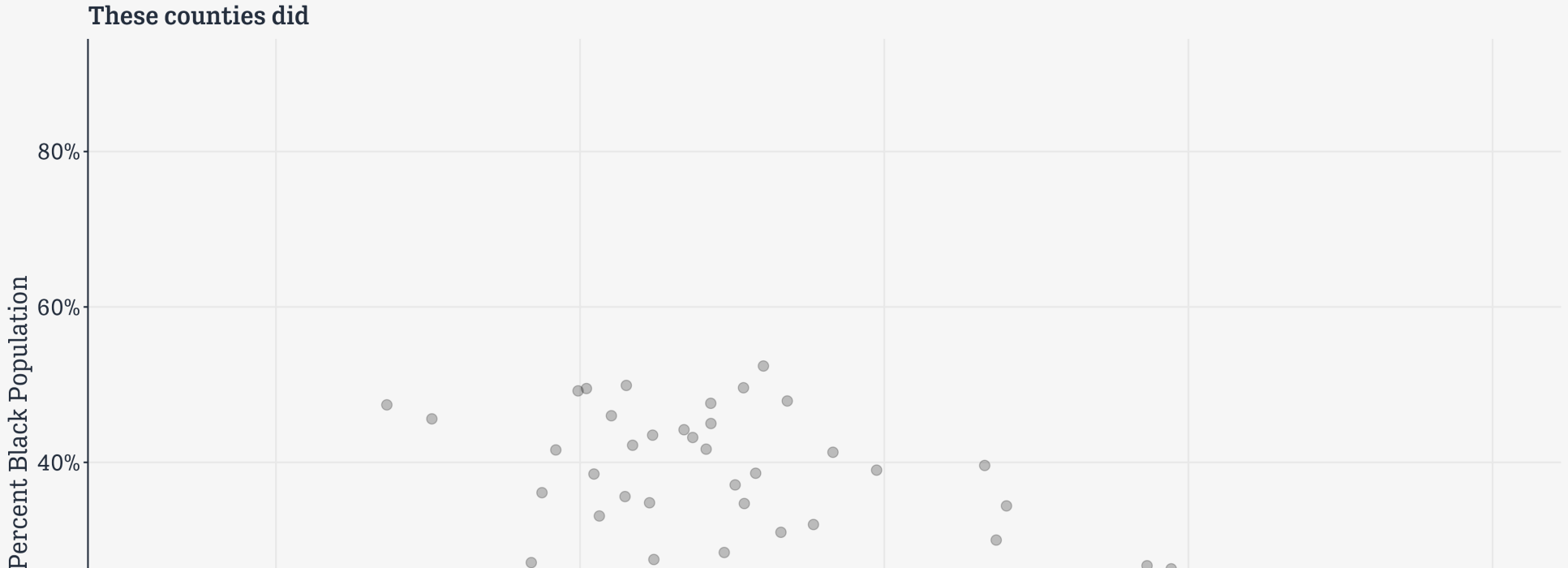
```
# Brighter Blue and Red
party_colors <- c("royalblue1", "red2")

ggplot(data = subset(county_data,
                     flipped == "No"),
       mapping = aes(x = pop,
                     y = black/100)) +
  geom_point(alpha = 0.15, color = "gray30",
             size = rel(2)) +
  scale_x_log10(labels = label_comma()) +
  geom_point(data = subset(county_data,
                      flipped == "Yes"),
             mapping = aes(x = pop, y = black/100,
                           color = partywinner16),
             size = rel(2)) +
  geom_text_repel(data = subset(county_data,
              flipped == "Yes" & black  > 25),
                  mapping = aes(x = pop,
                    y = black/100, label = state,
                    family = "Tenso Slide",
                    face = "bold"), size = rel(3.5)) +
  scale_color_manual(values = party_colors) +
  scale_y_continuous(labels = label_percent()) +
```

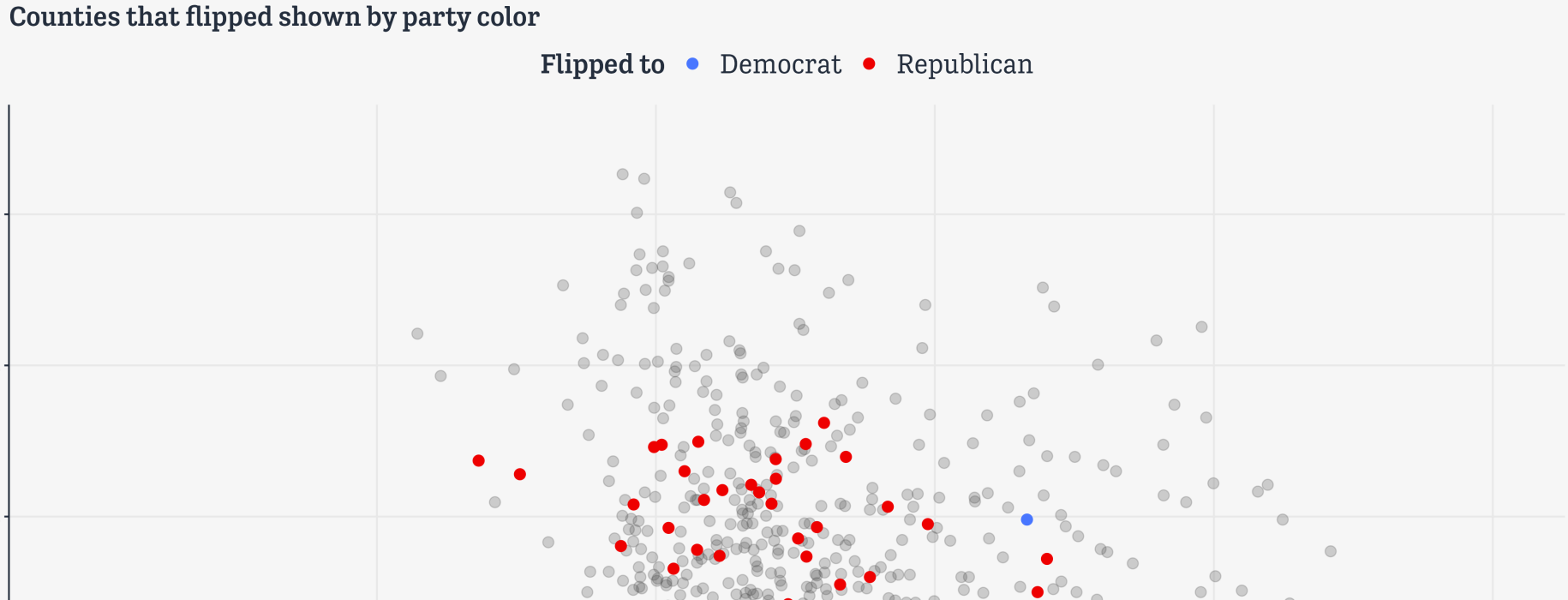


Leverage ggplot's layered approach

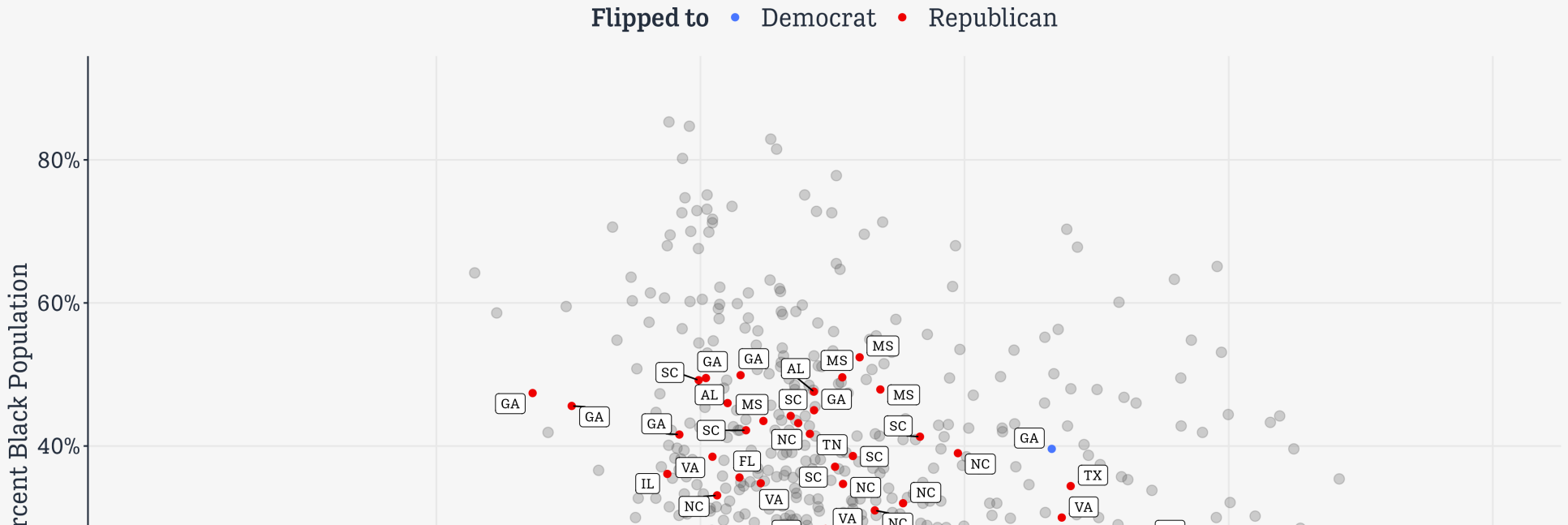

U.S. Counties by Population and Percent Black
Percent Black Population
80%
60%
40%

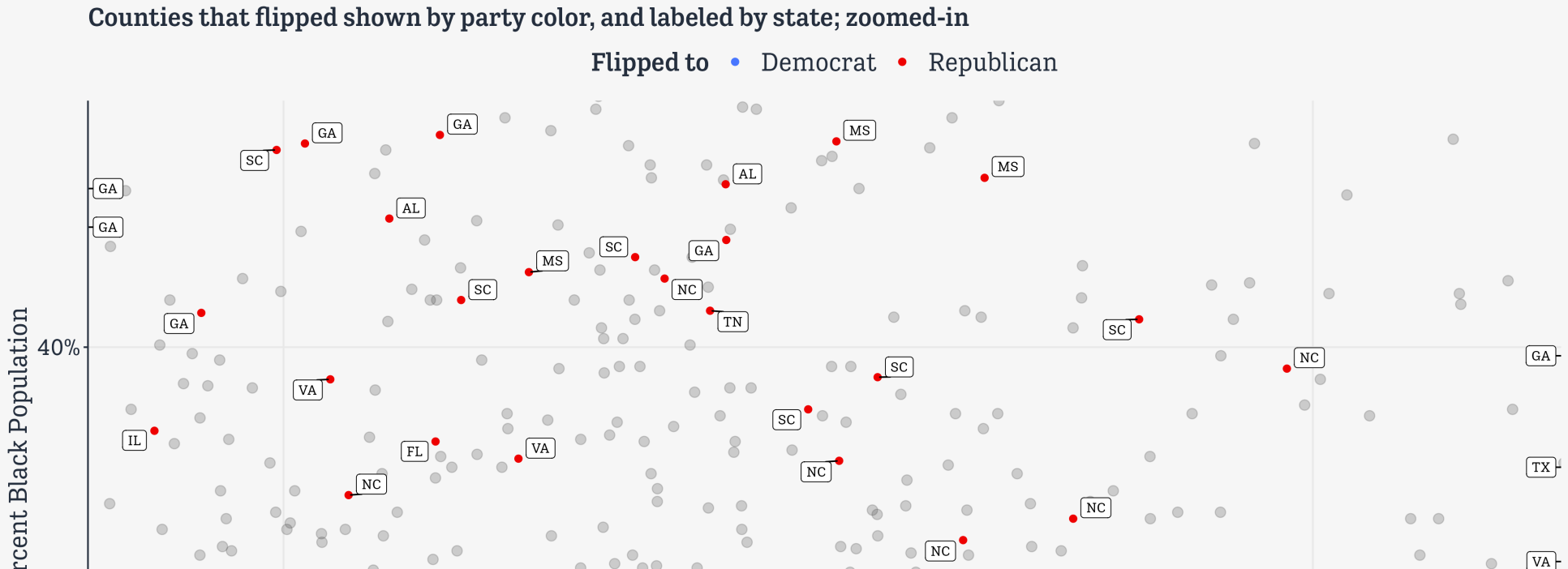

These counties did not flip in 2016
Percent Black Population
80%
60%
40%

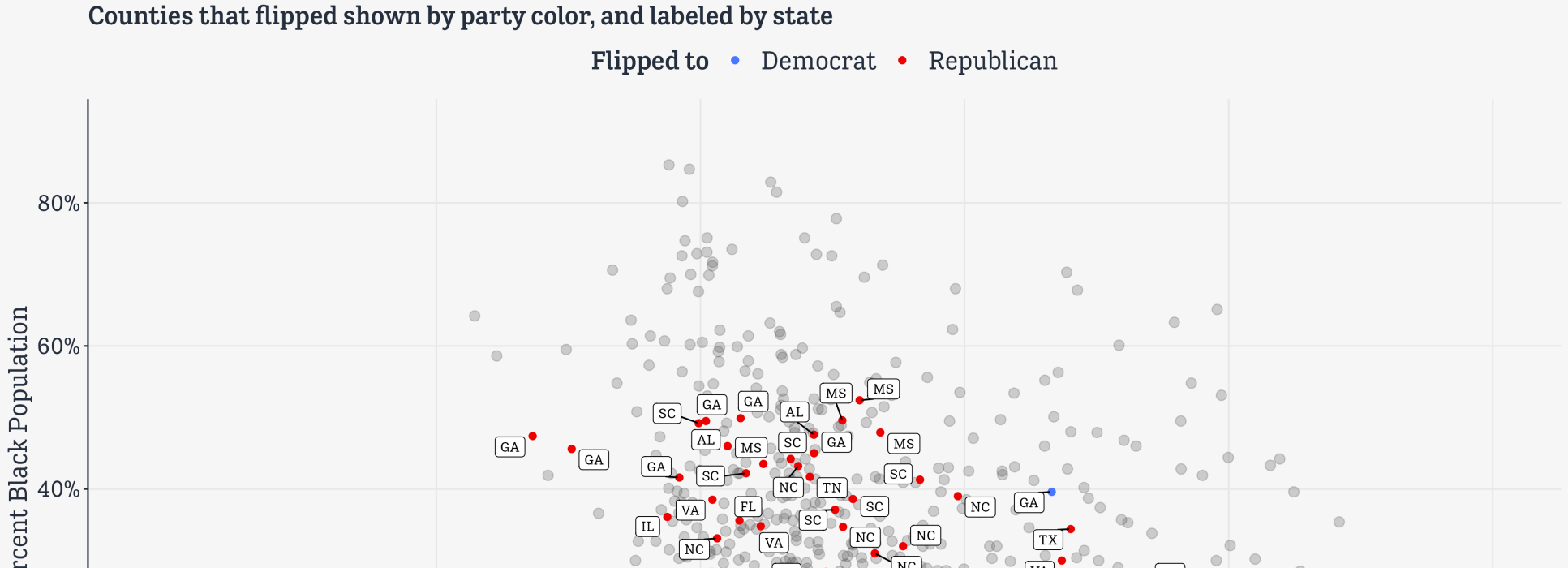

These counties did
80%
60%
40%
Percent Black Population


Counties that flipped shown by party color
Flipped to
Democrat
Republican
80%
60%
40%
rcent Black Population

Counties that flipped shown by party color, and labeled by state




Counties that flipped shown by party color, and labeled by state; zoomed-in
Flipped to
Democrat
Republican
40%
GA
SC
GA
GA
AL
GA
GA
MS
AL
MS
SC
GA
MS
SC
NC
TN
GA
SC
NC
GA
VA
SC
IL
SC
FL
VA
NC
NC
TX
NC
NC
VA


Counties that flipped shown by party color, and labeled by state
Flipped to
Democrat
Republican
80%
60%
40%
Percent Black Population
GA
GA
GA
SC
GA
GA
AL
MS
MS
AL
MS
SC
GA
MS
SC
SC
NC
TN
VA
FL
SC
NC
GA
IL
SC
NC
NC
NC
VA
NC
TX

Layer,

Highlight,

Repeat

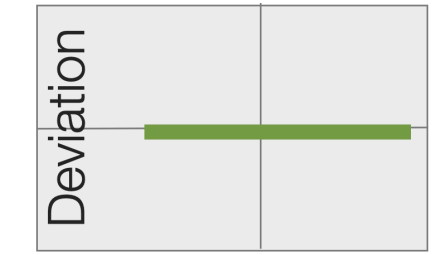
Vote Difference from Median Score

0 +

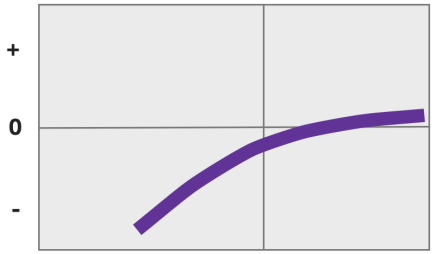
Deviation from Consensus

Build from ideas to data

Build from ideas to data

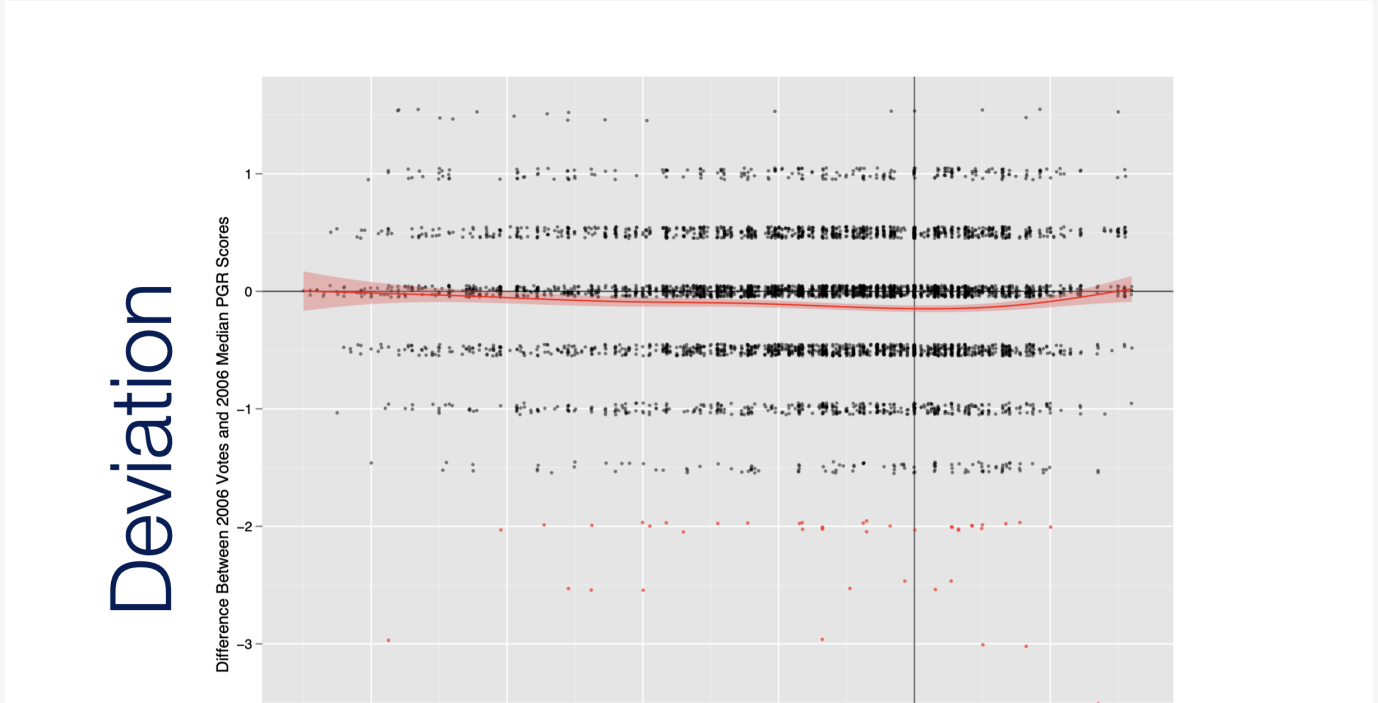


1. Pure Objectivity

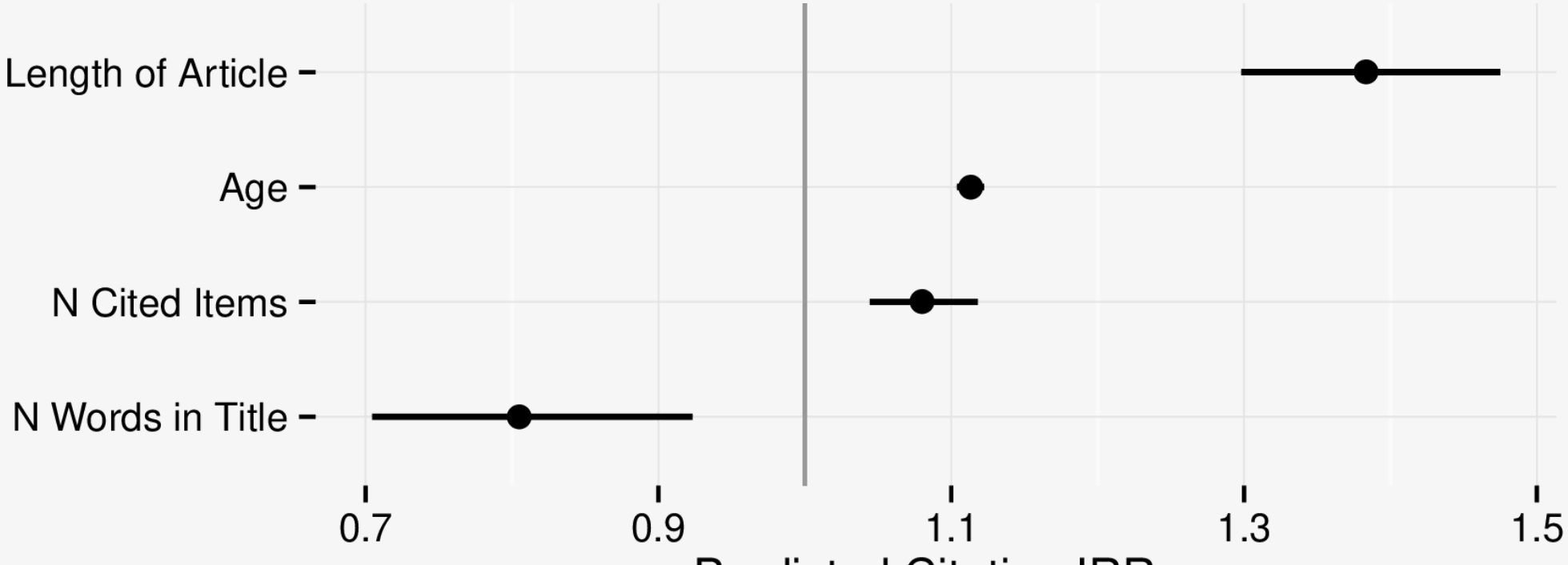


2. Distant Envy

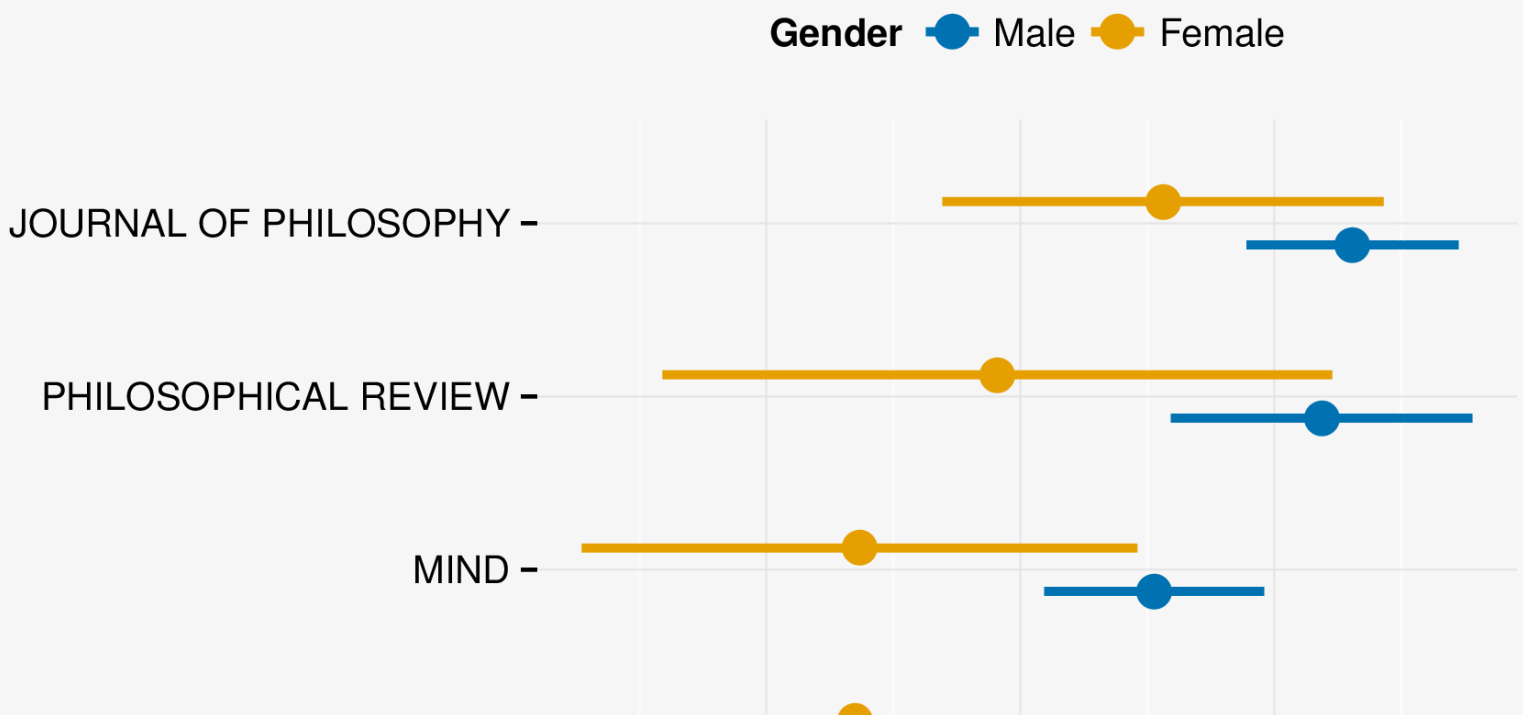
Build from ideas to data



Repeat to differentiate

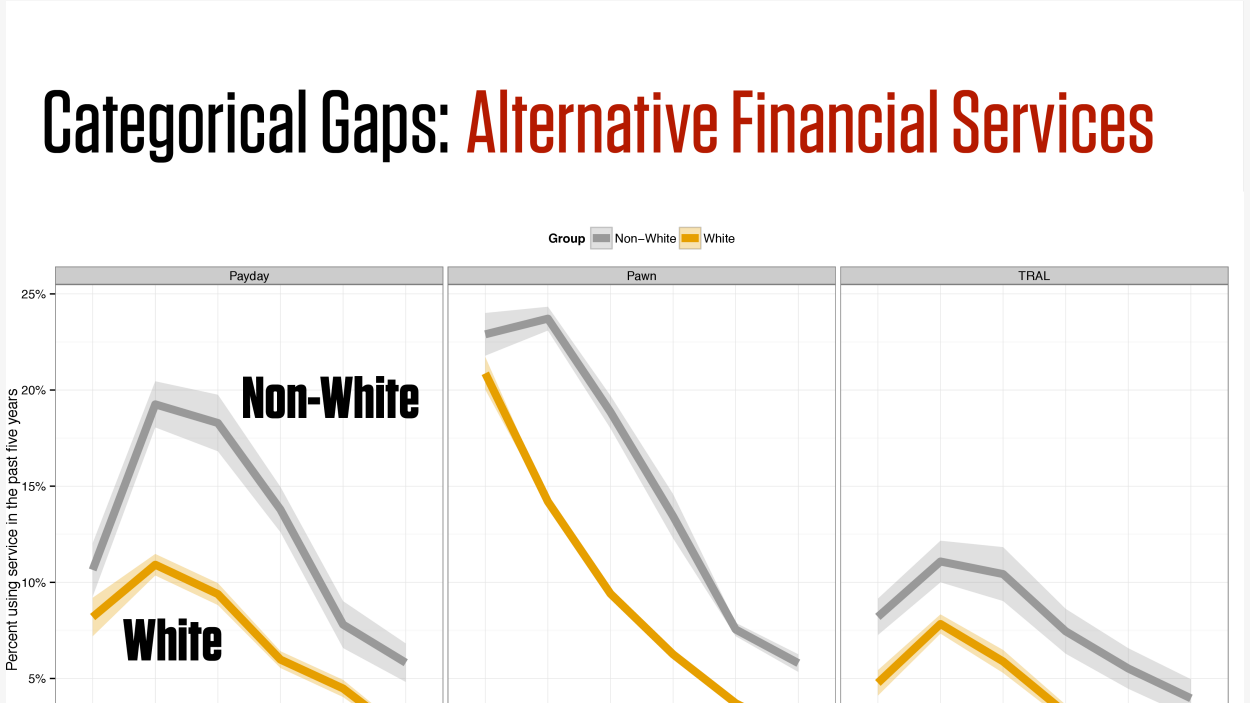


Repeat to differentiate

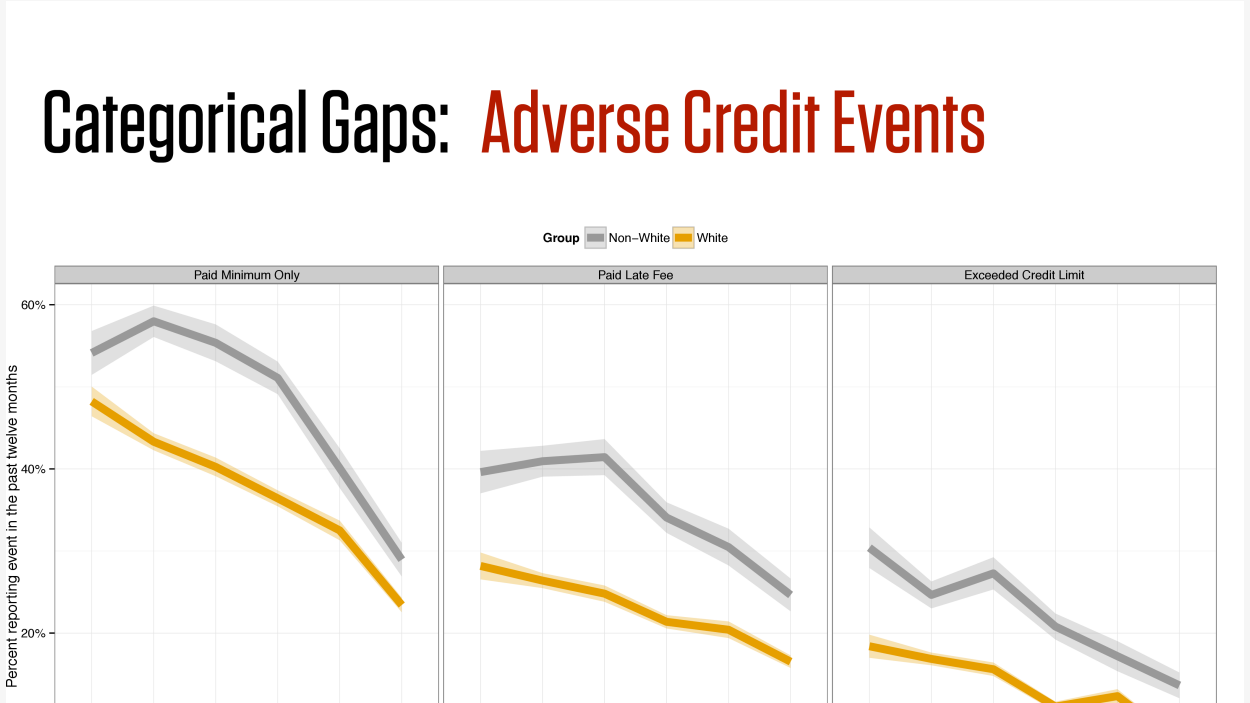


Layer and repeat with facets

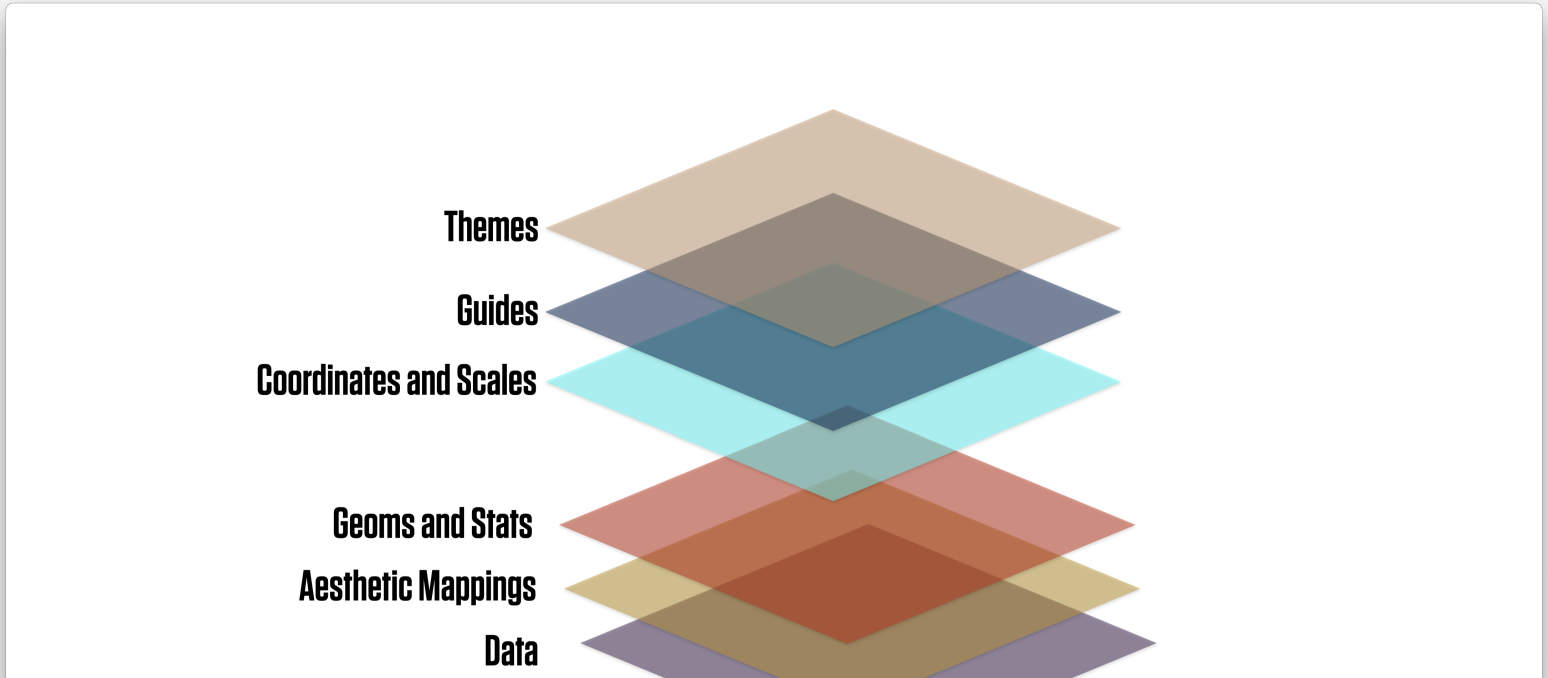
Layer and repeat across facets



Layer and repeat across facets



X-Ray Vision


Themes
Guides
Coordinates and Scales
Geoms and Stats
Aesthetic Mappings
Data

Theme.kjh-grey[s]

Themes ...

are controlled by the `theme()` function

can be bundled into functions of their own, like `theme_bw()` or `theme_minimal()`

can be set for the duration of a file or project with `theme_set()`

make changes that are applied *additively*

and most importantly ...

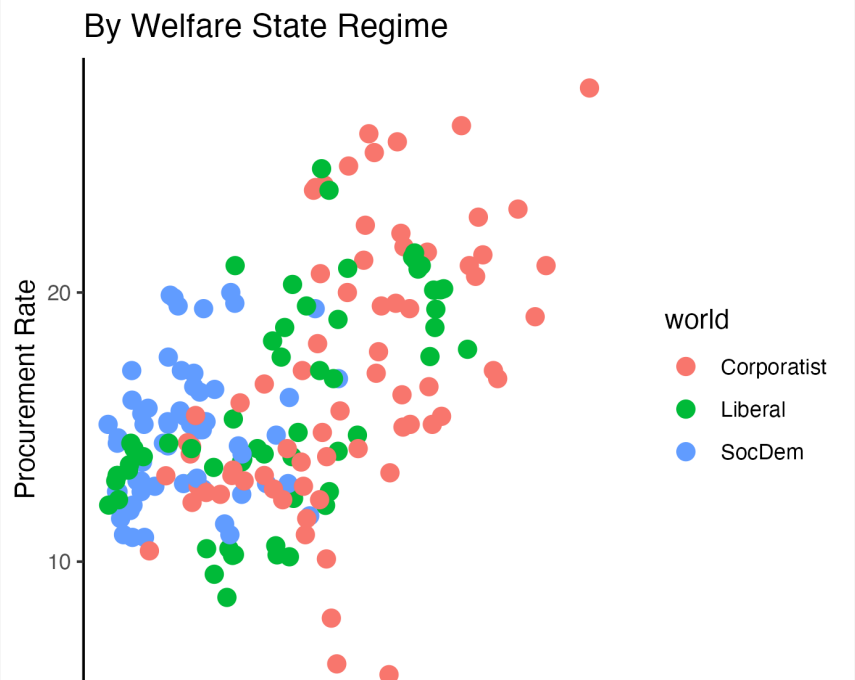
Thematic elements do not

represent data directly

Make a plot

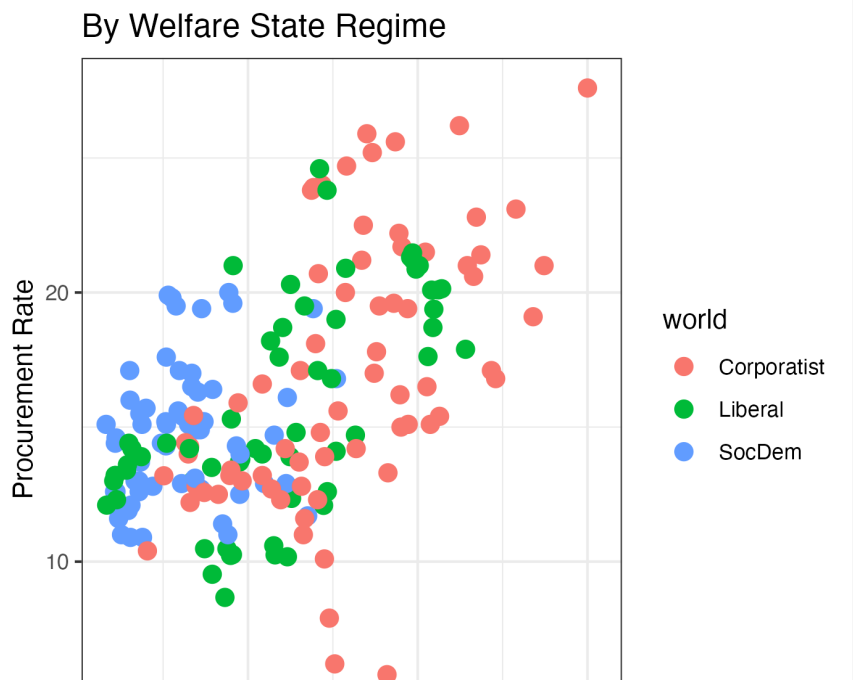
```
kjh_set_classic_theme(3)
```

```
p ← organdata ▷
  drop_na(world) ▷
  ggplot(mapping = aes(x = roads, y = donors,
                       color = world)) +
  geom_point(size = 3) +
  labs(x = "Road Deaths",

       y = "Procurement Rate",
       title = "By Welfare State Regime")

p
```



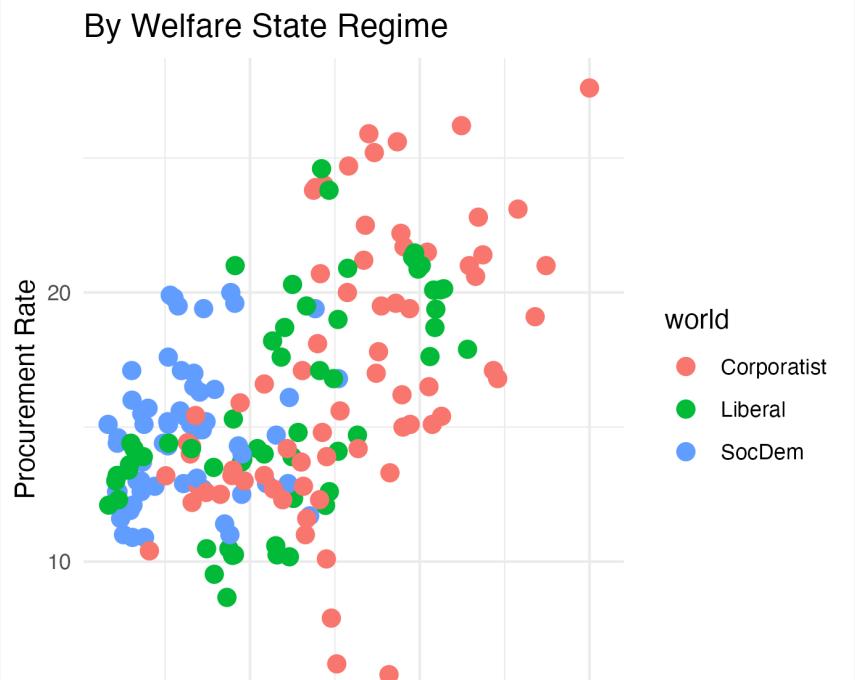
Add a theme ... theme_bw()

```
p + theme_bw()
```



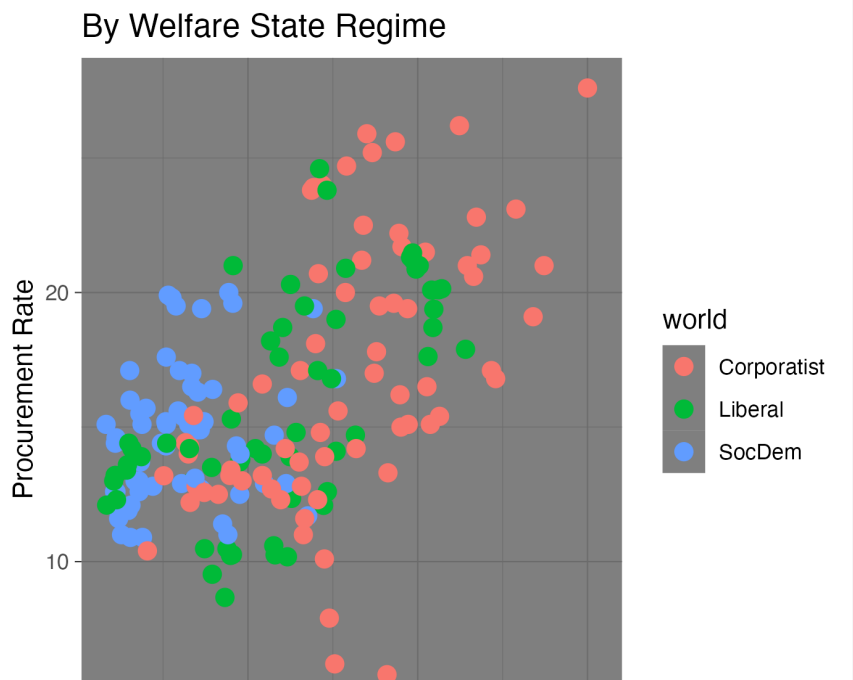
Add a theme ... theme_minimal()

```
p + theme_minimal()
```



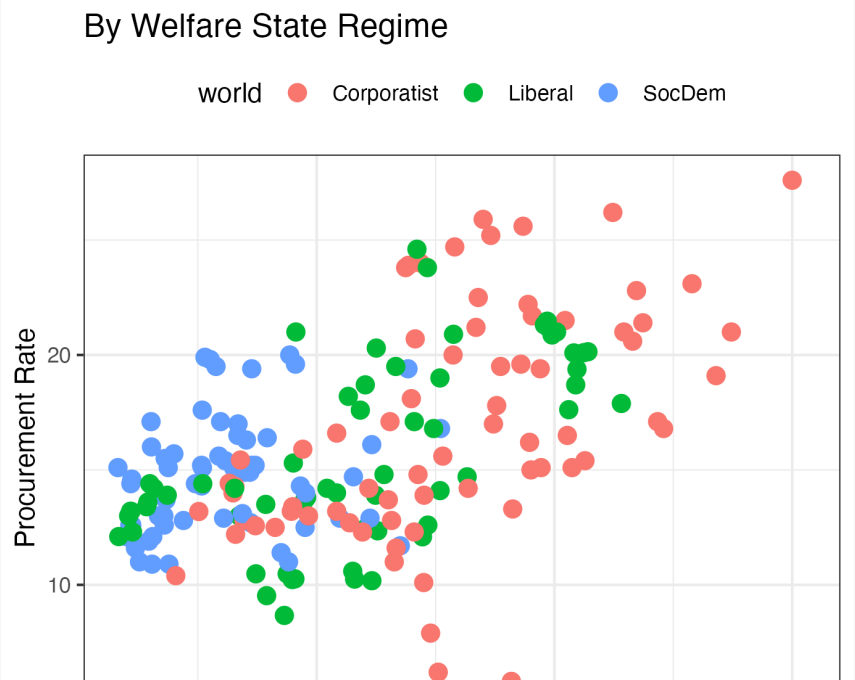
Add a theme ... theme_dark()

```
p + theme_dark()
```



Adjust with the `theme()` function

```
p + theme_bw() +
  theme(legend.position = "top")
```



There are *many* adjustable theme elements

line rect text title
aspect.ratio
axis.title axis.title.x
axis.title.x.top
axis.title.x.bottom axis.title.y
axis.title.y.left
axis.title.y.right axis.text
axis.text.x axis.text.x.top
axis.text.x.bottom axis.text.y
axis.text.y.left

legend.background
legend.margin
legend.spacing
legend.spacing.x
legend.spacing.y legend.key
legend.key.size
legend.key.height
legend.key.width legend.text
legend.text.align legend.title
legend.title.align

But they are structured

line rect text title

aspect.ratio

axis.title axis.title.x

axis.title.x.top

axis.title.x.bottom axis.title.y

axis.title.y.left

axis.title.y.right axis.text

axis.text.x axis.text.x.top

axis.text.x.bottom axis.text.y

axis.text.y.left

axis.text.y.right axis.ticks

axis.ticks.x axis.ticks.x.top

legend.background

legend.margin

legend.spacing

legend.spacing.x

legend.spacing.y legend.key

legend.key.size

legend.key.height

legend.key.width legend.text

legend.text.align legend.title

legend.title.align

legend.position

legend.direction

And *inherit*

line rect text title

aspect.ratio

axis.title axis.title.x
axis.title.x.top
axis.title.x.bottom axis.title.y
axis.title.y.left
axis.title.y.right] **axis.text**
axis.text.x axis.text.x.top
axis.text.x.bottom axis.text.y
axis.text.y.left
axis.text.y.right] **axis.ticks**
axis.ticks.x axis.ticks.x.top

legend.background
legend.margin
legend.spacing
legend.spacing.x
legend.spacing.y] **legend.key**
legend.key.size
legend.key.height
legend.key.width] **legend.text**
legend.text.align legend.title
legend.title.align
legend.position
legend.direction

Two kinds of adjustment

It's a single setting.

E.g, `legend.position` can be `"none"`, `"left"`, `"right"`, `"bottom"`, or `"top"`

Hence, e.g., `theme(legend.position = "top")`, which we have seen several times. Similarly for e.g. `legend.direction` (can be "horizontal" or "vertical").

It's a component of the plot that might be styled in several ways. E.g., The text on the axes, or the lines in the plot panel.

If the latter ...

If adjusting a thematic element ask...

Where on the plot is it?

Is it part of an *axis*, part of the *panel*, the *strip* (facet title) box, or the *legend*? This will help you find the name of the thing you want to adjust.

E.g. “I want to adjust the text for the markings on the x-axis”

You want axis.ticks.x

E.g. “I want to adjust the styling of the main y-axis grid lines inside the plot”

You want panel.grid.major.y

If adjusting a thematic element, ask...

What *kind* of element is it?

Is it *text*, or a *line*, or a *rectangle*?

This will tell you what function to use to make the adjustment to the named element.

If it's text, adjust the element with `element_text()`

If it's a line, adjust it with `element_line()`

If it's a rectangle, with `element_rect()`

If you want to *fully turn off* an element, use `element_blank()`

For example ...

“I want to adjust the styling of the plot title”

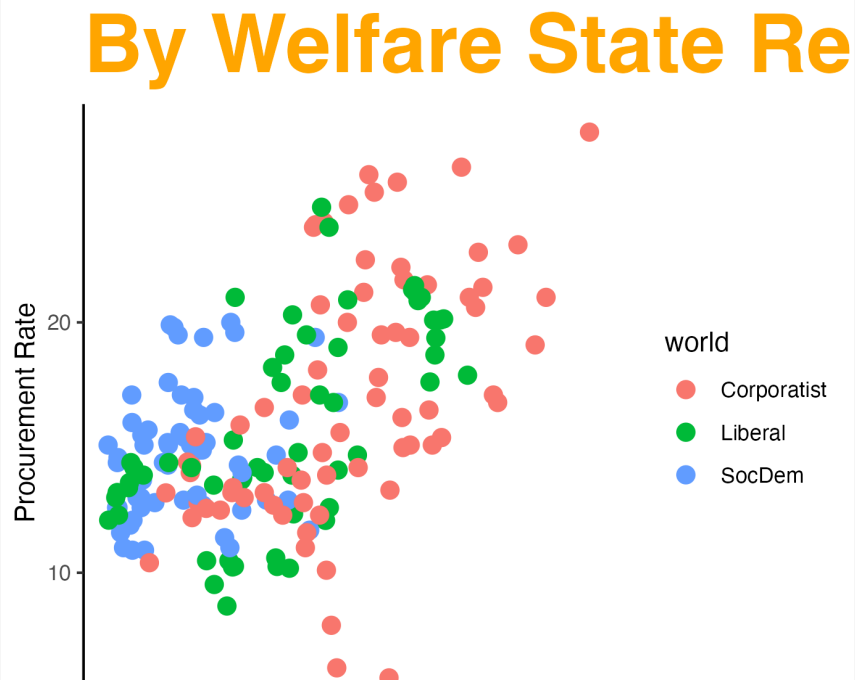
The relevant element is `plot.title`.

It’s *text*.

Inside the theme function, adjust it with `element_text()`.

For example ...

```
p + theme(plot.title =
            element_text(size = rel(3),
                         face = "bold",
                         color = "orange"))
```



For example ...

“I want to adjust y axis grid lines on the plot”

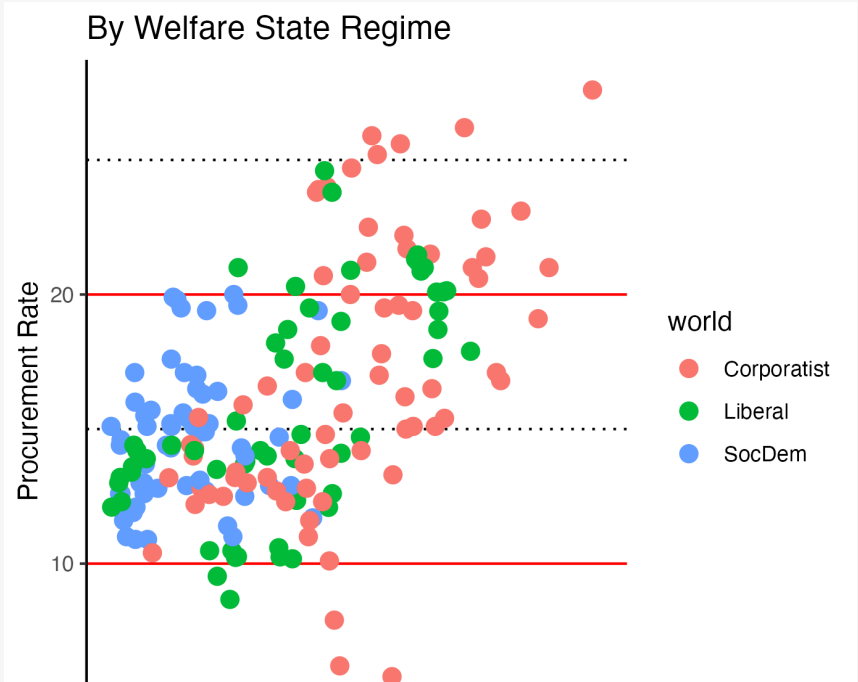
The relevant elements are `panel.grid.major.y` and `panel.grid.minor.y`.

These are *lines*.

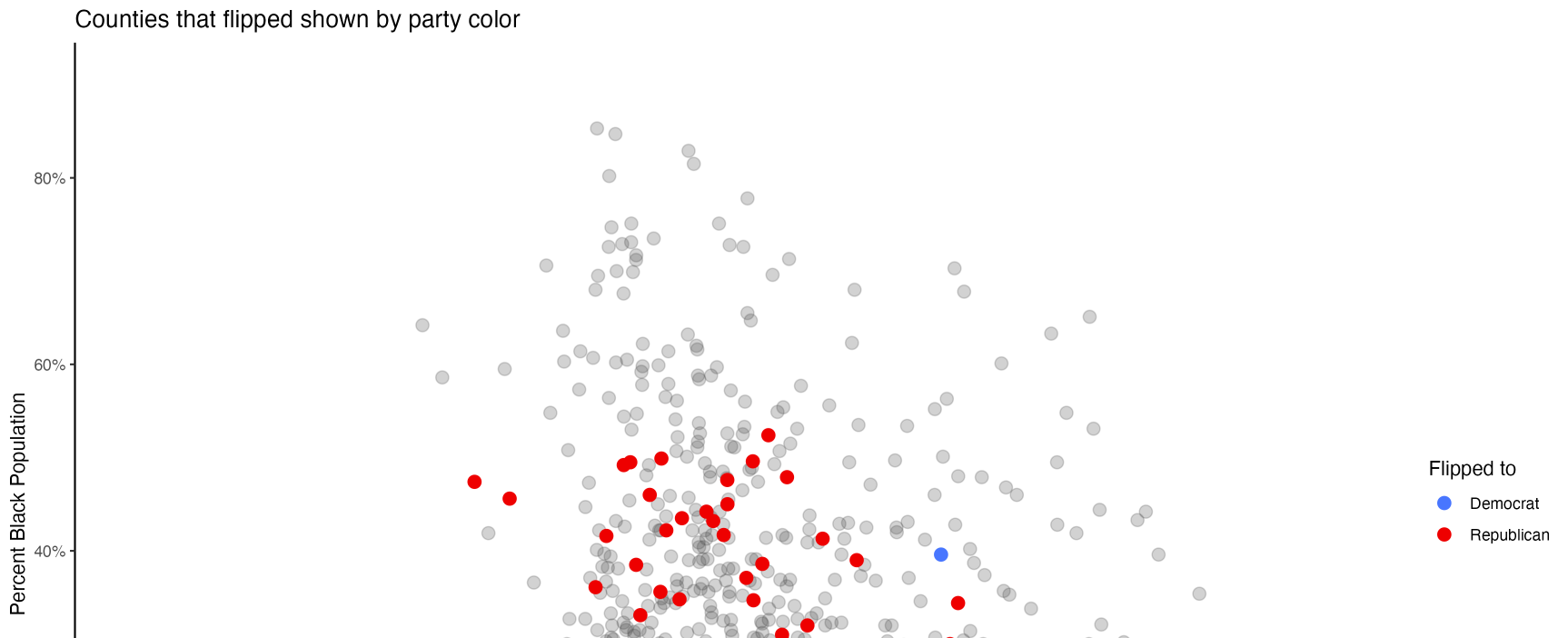
Inside the theme function, adjust it with `element_line()`.

For example ...

```
p + theme(panel.grid.major.y =
            element_line(color = "red"),
          panel.grid.minor.y =
            element_line(color = "black",
                         linetype = "dotted"))
```



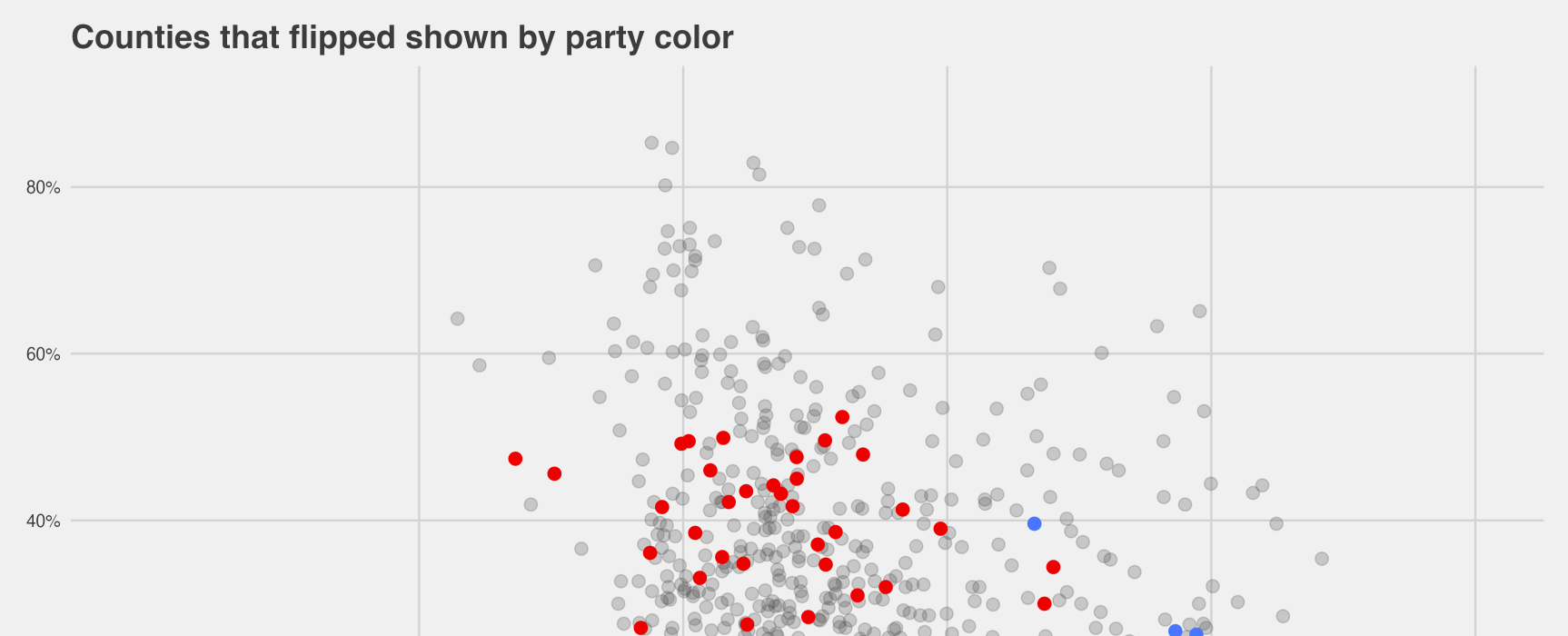
The ggthemes package


Counties that flipped shown by party color
Percent Black Population
80%
60%
40%
Flipped to
Democrat
Republican

Theming a plot

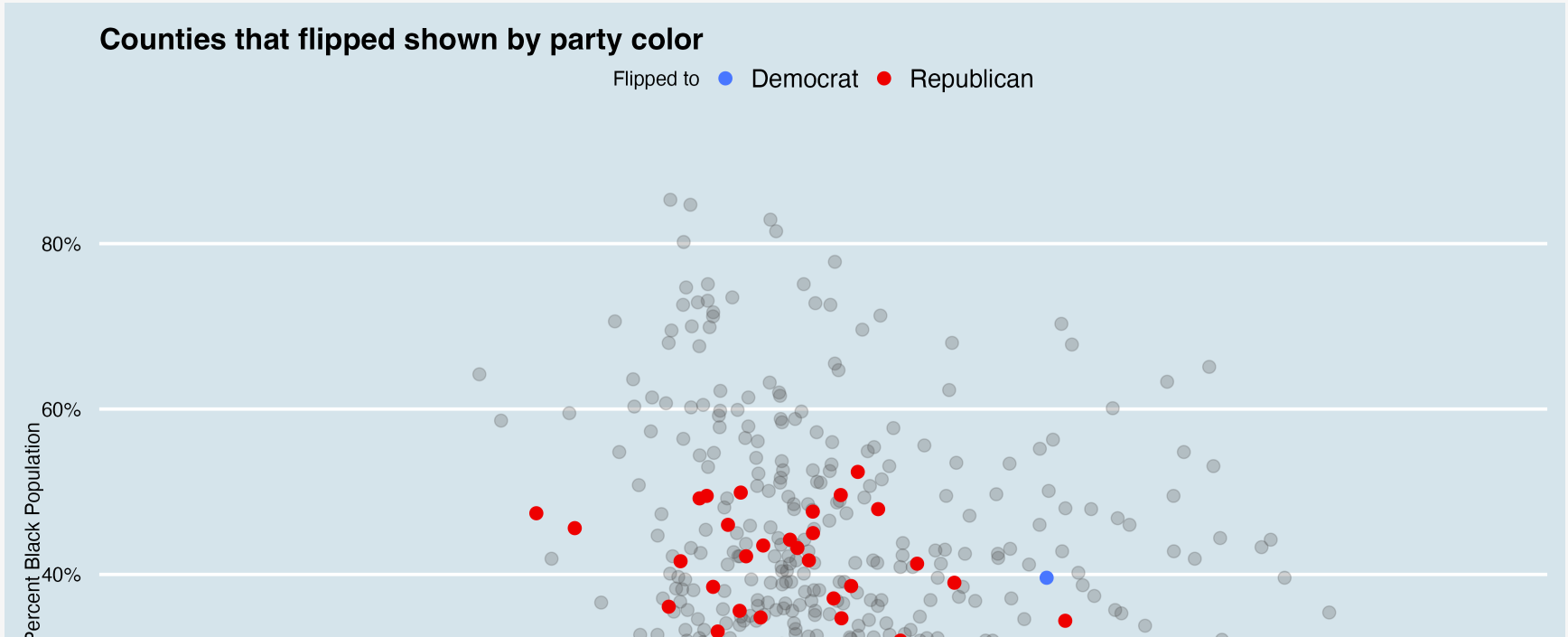
```
library(ggthemes)
theme_set(theme_fivethirtyeight())
```

See how the full function call goes inside `theme_set()`, including the parentheses, because we are actually running that function to set all the elements.


Counties that flipped shown by party color
80%
60%
40%

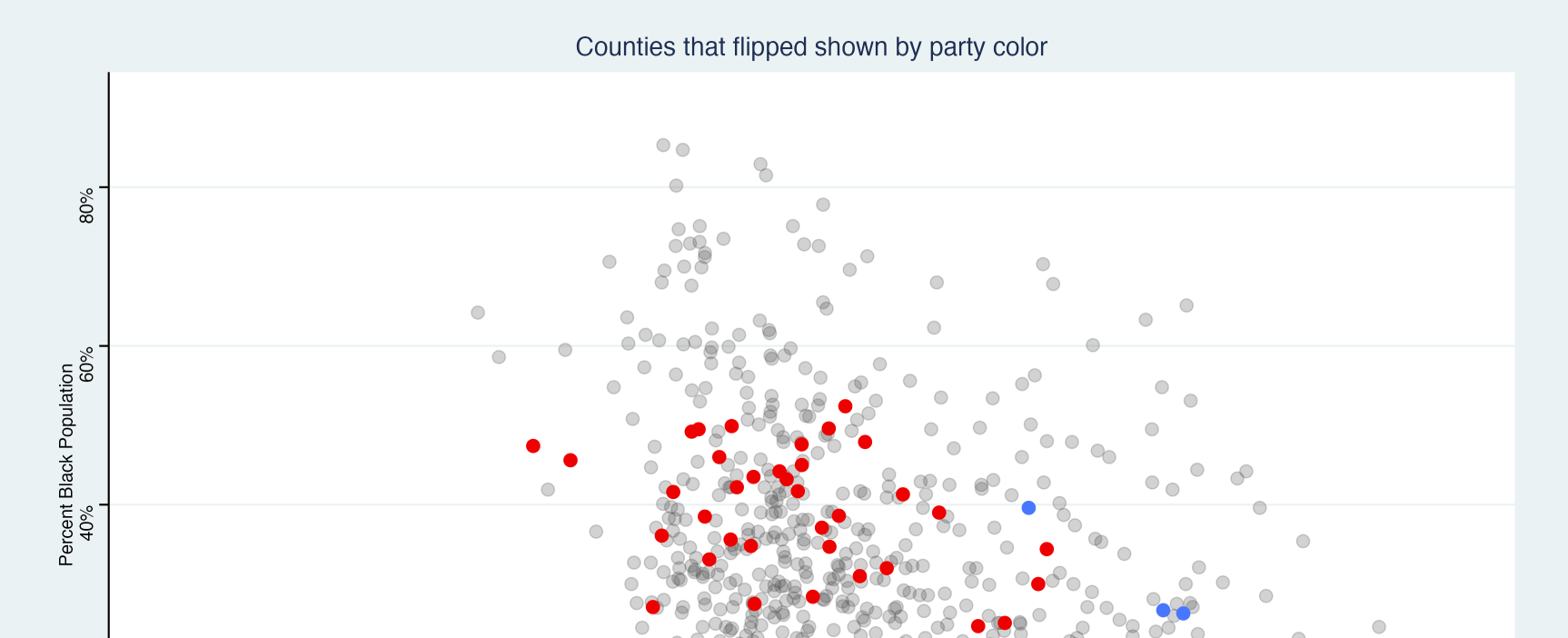
Theming a plot

```
theme_set(theme_economist())
```


Counties that flipped shown by party color
Flipped to
Democrat
Republican
80%
60%
40%
Percent Black Population

This seems morally wrong

```
theme_set(theme_stata())
```


Counties that flipped shown by party color
Percent Black Population
80%
60%
40%

Pick a theme
and stick with it