Code for QSS tidyverse Chapter 5: Discovery

Kosuke Imai and Nora Webb Williams

First Printing

Discovery

Textual Data

The Disputed Authorship of The Federalist Papers

```
## load the packages
library("tm")
library("SnowballC")
library("stringr")
library("tidytext")
library("tidyverse")
```

```
## the location of files from qss package
DIR_SOURCE <- system.file("extdata/federalist", package = "qss")
## loading the corpus
corpus_raw <- VCorpus(DirSource(directory = DIR_SOURCE, pattern = "fp"))
## the corpus object
corpus_raw
```

```
## <<VCorpus>>
## Metadata:  corpus specific: 0, document level (indexed): 0
## Content:  documents: 85
```

```
## A tidy version of the corpus
corpus_tidy <- tidy(corpus_raw) %>%
  select(id, text) %>%
  mutate(new_id = as.integer(str_sub(id, start = 3, end = 4)))

glimpse(corpus_tidy)
```

```
## Rows: 85
## Columns: 3
## $ id     <chr> "fp01.txt", "fp02.txt", "fp03.txt", "fp04.t~
## $ text   <chr> "AFTER an unequivocal experience of the ine~
## $ new_id <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, ~
```

```r
tokens_raw <- corpus_tidy %>%
  ## tokenizes into words
  unnest_tokens(word, text, to_lower = TRUE) %>%
  ## stem the words
  mutate(stem = wordStem(word)) %>%
  ## remove any numbers in the strings
  mutate(word = str_replace_all(word, "\\d+", "")) %>%
  ## drop any empty strings
  filter(word != "")

## look at the result
glimpse(tokens_raw)
```

```
## Rows: 187,412
## Columns: 4
## $ id     <chr> "fp01.txt", "fp01.txt", "fp01.txt", "fp01.t~
## $ new_id <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ word   <chr> "after", "an", "unequivocal", "experience",~
## $ stem   <chr> "after", "an", "unequivoc", "experi", "of",~
```

```r
## load standard stop words
data("stop_words", package = "tidytext")
glimpse(stop_words)
```

```
## Rows: 1,149
## Columns: 2
## $ word    <chr> "a", "a's", "able", "about", "above", "acc~
## $ lexicon <chr> "SMART", "SMART", "SMART", "SMART", "SMART~
```

```r
## remove stopwords
tokens <- tokens_raw %>%
  anti_join(stop_words, by = "word")
```

```r
## the output is truncated here to save space
content(corpus_raw[[10]]) # Essay No. 10
```

```
##   [1] "AMONG the numerous advantages promised by a well-constructed Union, none "
##   [2] "        deserves to be more accurately developed than its tendency to break and "
##   [3] "        control the violence of faction. The friend of popular governments never "
...
```

Document-Term Matrix

```r
tokens_counts <- count(tokens, new_id, stem)
head(tokens_counts)
```

```
## # A tibble: 6 x 3
##   new_id stem           n
##    <int> <chr>      <int>
```

```
## 1      1 absurd         1
## 2      1 accid          1
## 3      1 acknowledg     1
## 4      1 act            1
## 5      1 actuat         1
## 6      1 add            1
```

```
dtm <- cast_dtm(tokens_counts,
                document = new_id,
                term = stem,
                value = n)
dtm
```

```
## <<DocumentTermMatrix (documents: 85, terms: 4674)>>
## Non-/sparse entries: 37214/360076
## Sparsity           : 91%
## Maximal term length: 17
## Weighting          : term frequency (tf)
```

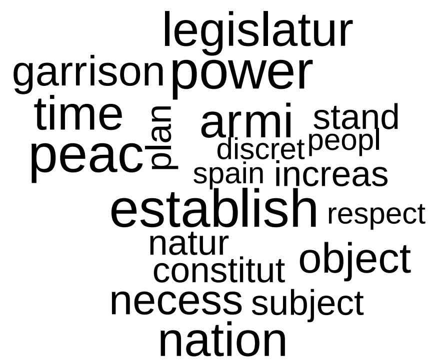
```
inspect(dtm[1:5, 1:8])
```

```
## <<DocumentTermMatrix (documents: 5, terms: 8)>>
## Non-/sparse entries: 16/24
## Sparsity           : 60%
## Maximal term length: 10
## Weighting          : term frequency (tf)
## Sample             :
##     Terms
## Docs absurd accid acknowledg act actuat add addit address
##    1      1     1          1   1      1   1     1       1
##    2      0     0          0   0      0   0     0       0
##    3      0     0          2   1      1   1     1       0
##    4      0     0          0   1      0   0     0       1
##    5      0     0          0   1      0   0     0       0
```

```
dtm.mat <- as.matrix(dtm)
```

Topic Discovery

```
library(wordcloud)
par(mar = c(0, 0, 0, 0))
## wordcloud for document 12
doc_12 <- filter(tokens_counts, new_id == 12)

wordcloud(words = doc_12$stem, freq = doc_12$n,
          max.words = 20)

## wordcloud for document 24
doc_24 <- filter(tokens_counts, new_id == 24)

wordcloud(words = doc_24$stem, freq = doc_24$n,
          max.words = 20)
```




```r
stemCompletion(c("revenu", "commerc", "peac", "armi"), corpus_raw)
```

```
##     revenu     commerc       peac        armi
##  "revenue" "commerce"    "peace"    "armies"
```

```r
tokens_counts <- bind_tf_idf(tokens_counts,
                             term = stem,
                             document = new_id,
                             n = n)
head(tokens_counts)
```

```
## # A tibble: 6 x 6
##   new_id stem            n      tf   idf   tf_idf
##    <int> <chr>       <int>   <dbl> <dbl>    <dbl>
## 1      1 absurd          1 0.00186 1.73  0.00323
## 2      1 accid           1 0.00186 3.75  0.00698
## 3      1 acknowledg      1 0.00186 1.55  0.00289
## 4      1 act             1 0.00186 0.400 0.000744
## 5      1 actuat          1 0.00186 2.14  0.00399
## 6      1 add             1 0.00186 1.35  0.00252
```

```r
dtm_tfidf <- weightTfIdf(dtm)
dtm_tfidf
```

```
## <<DocumentTermMatrix (documents: 85, terms: 4674)>>
## Non-/sparse entries: 37044/360246
## Sparsity           : 91%
## Maximal term length: 17
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
```

```r
## Top words for document 12
tokens_counts %>%
  filter(new_id == 12) %>%
  slice_max(tf_idf, n = 10)
```

```
## # A tibble: 10 x 6
##    new_id stem           n      tf   idf tf_idf
##     <int> <chr>      <int>   <dbl> <dbl>  <dbl>
##  1     12 revenu        11 0.0138   1.22 0.0169
##  2     12 contraband     3 0.00376  4.44 0.0167
##  3     12 patrol         3 0.00376  4.44 0.0167
##  4     12 excis          5 0.00627  2.65 0.0166
##  5     12 coast          3 0.00376  3.75 0.0141
##  6     12 tax            8 0.0100   1.31 0.0131
##  7     12 trade          6 0.00752  1.73 0.0130
##  8     12 cent           2 0.00251  4.44 0.0111
##  9     12 gallon         2 0.00251  4.44 0.0111
## 10     12 commerc        8 0.0100   1.11 0.0111
```

```
## Top words for document 24
tokens_counts %>%
  filter(new_id == 24) %>%
  slice_max(tf_idf, n = 10)
```

```
## # A tibble: 10 x 6
##    new_id stem           n      tf   idf tf_idf
##     <int> <chr>      <int>   <dbl> <dbl>  <dbl>
##  1     24 garrison       6 0.00926  2.83 0.0262
##  2     24 dock           3 0.00463  4.44 0.0206
##  3     24 yard           3 0.00463  4.44 0.0206
##  4     24 settlement     3 0.00463  3.75 0.0174
##  5     24 spain          4 0.00617  2.36 0.0146
##  6     24 armi           7 0.0108   1.26 0.0137
##  7     24 frontier       3 0.00463  2.83 0.0131
##  8     24 arsen          2 0.00309  3.75 0.0116
##  9     24 western        3 0.00463  2.50 0.0116
## 10     24 nearer         2 0.00309  3.34 0.0103
```

```
k <- 5 # number of clusters
## subset The Federalist papers written by Hamilton
hamilton <- c(1, 6:9, 11:13, 15:17, 21:36, 59:61, 65:85)
hamilton_docs <- filter(tokens_counts,
                        new_id %in% hamilton)

## convert into a document term matrix
## then calculate tf_idf
hamilton_dtm <- cast_dtm(hamilton_docs,
                document = new_id,
                term = stem,
                value = n) %>%
  weightTfIdf()

## check the output
hamilton_dtm
```

```
## <<DocumentTermMatrix (documents: 51, terms: 3918)>>
## Non-/sparse entries: 22478/177340
## Sparsity           : 89%
```

```
## Maximal term length: 17
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
```

```r
## run k-means, with a set seed for replication
set.seed(1234)
km.out <- kmeans(hamilton_dtm, centers = k)
km.out$iter # check the convergence; number of iterations may vary
```

```
## [1] 3
```

```r
## How many documents per cluster?
table(km.out$cluster)
```

```
##
##  1  2  3  4  5
##  8  6  1  1 35
```

```r
## label each centroid with the corresponding term
colnames(km.out$centers) <- colnames(hamilton_dtm)

for (i in 1:k) { # loop for each cluster
    print(str_c("CLUSTER ", i))
    print("Top 10 words: ")
    ## create a tibble of the cluster words
    ## print 10 most important terms
    cluster_centers <- enframe(km.out$centers[i, ]) %>%
     slice_max(value, n = 10)
    print(cluster_centers)
    print("Federalist Papers classified:") # extract essays classified
    ## create a tibble of cluster assignments
    cluster_docs <- enframe(km.out$cluster, "document", "cluster") %>%
      filter(cluster == i)
    print(as.vector(cluster_docs$document))
    cat("\n")
}
```

```
## [1] "CLUSTER 1"
## [1] "Top 10 words: "
## # A tibble: 10 x 2
##    name       value
##    <chr>      <dbl>
##  1 senat    0.0222
##  2 presid   0.0194
##  3 governor 0.0118
##  4 pardon   0.0114
##  5 treati   0.0113
##  6 offic    0.0104
##  7 appoint  0.0103
##  8 impeach  0.00974
##  9 nomin    0.00950
## 10 vote     0.00727
## [1] "Federalist Papers classified:"
```

```
## [1] "66" "68" "69" "74" "75" "76" "77" "79"
##
## [1] "CLUSTER 2"
## [1] "Top 10 words: "
## # A tibble: 10 x 2
##    name        value
##    <chr>       <dbl>
##  1 armi      0.0229
##  2 militia   0.0225
##  3 militari  0.0141
##  4 disciplin 0.00963
##  5 garrison  0.00805
##  6 peac      0.00793
##  7 troop     0.00758
##  8 liberti   0.00619
##  9 corp      0.00566
## 10 neighbor  0.00553
## [1] "Federalist Papers classified:"
## [1] "8"  "24" "25" "26" "28" "29"
##
## [1] "CLUSTER 3"
## [1] "Top 10 words: "
## # A tibble: 10 x 2
##    name          value
##    <chr>         <dbl>
##  1 northern     0.0607
##  2 southern     0.0455
##  3 confederaci  0.0448
##  4 list         0.0326
##  5 frontier     0.0265
##  6 comprehens   0.0238
##  7 civil        0.0219
##  8 jersei       0.0218
##  9 pennsylvania 0.0216
## 10 navig        0.0200
## [1] "Federalist Papers classified:"
## [1] "13"
##
## [1] "CLUSTER 4"
## [1] "Top 10 words: "
## # A tibble: 10 x 2
##    name     value
##    <chr>    <dbl>
##  1 vacanc  0.0947
##  2 recess  0.0552
##  3 session 0.0483
##  4 senat   0.0482
##  5 claus   0.0465
##  6 fill    0.0453
##  7 appoint 0.0312
##  8 expir   0.0237
##  9 presid  0.0216
## 10 unfound 0.0192
## [1] "Federalist Papers classified:"
```

```
## [1] "67"
##
## [1] "CLUSTER 5"
## [1] "Top 10 words: "
## # A tibble: 10 x 2
##    name        value
##    <chr>       <dbl>
##  1 court     0.00783
##  2 juri      0.00490
##  3 tax       0.00457
##  4 jurisdict 0.00395
##  5 taxat     0.00373
##  6 elect     0.00338
##  7 trial     0.00335
##  8 land      0.00334
##  9 revenu    0.00327
## 10 claus     0.00316
## [1] "Federalist Papers classified:"
##  [1] "1"  "6"  "7"  "9"  "11" "12" "15" "16" "17" "21" "22"
## [12] "23" "27" "30" "31" "32" "33" "34" "35" "36" "59" "60"
## [23] "61" "65" "70" "71" "72" "73" "78" "80" "81" "82" "83"
## [34] "84" "85"
```

Authorship Prediction

```
## essays written by Madison, Jay, or joint:
## "hamilton" defined earlier
madison <- c(10, 14, 37:48, 58)
jay <- c(2:5, 64)
joint <- c(18:20)

## the specific words of interest
STYLE_WORDS <- c("although", "always", "commonly", "consequently",
                 "considerable", "enough", "there", "upon",
                 "while", "whilst")

## add a variable for the author
tokens_raw <- tokens_raw %>%
  mutate(author = case_when(new_id %in% hamilton ~ "Hamilton",
                            new_id %in% madison ~ "Madison",
                            new_id %in% jay ~ "Jay",
                            new_id %in% joint ~ "Joint",
                            TRUE ~ "Disputed"))

## Average word use per thousand words by author
tfm <- tokens_raw %>%
  group_by(author, word) %>%
  ## total term use per author
  summarize(n = n()) %>%
  ungroup() %>%
  group_by(author) %>%
  ## average term use by author per 1000 words
```

```r
  mutate(tf_thou = n / sum(n) * 1000) %>%
  ## just the words of interest
  filter(word %in% STYLE_WORDS) %>%
  ## drop n for pivoting
  select(-n) %>%
  ## reshape
  pivot_wider(names_from = word,
              values_from = tf_thou) %>%
  mutate_at(vars(always:consequently), replace_na, 0)

tfm
```

```
## # A tibble: 5 x 11
## # Groups:   author [5]
##   author  although always commonly consequently considerable
##   <chr>      <dbl>  <dbl>    <dbl>        <dbl>        <dbl>
## 1 Disput~  0.137    0.228   0.0456       0.365        0.228
## 2 Hamilt~  0.00904  0.551   0.190        0.0361       0.416
## 3 Jay      0.597    0.955   0.119        0.477        0.119
## 4 Joint   NA        0.355   0.178        0            0.355
## 5 Madison  0.196    0.171   0            0.318        0.122
## # ... with 5 more variables: there <dbl>, upon <dbl>,
## #   whilst <dbl>, enough <dbl>, while <dbl>
```

```r
## Create new data set for regression
## Average word use per thousand words by author per document
reg_data <- tokens_raw %>%
  group_by(author, new_id, word) %>%
  ## total term use per author-document
  summarize(n = n()) %>%
  ## average term use by author per 1000 words per document
  mutate(tf_thou = n / sum(n) * 1000) %>%
  ## just the words of interest
  filter(word %in% STYLE_WORDS) %>%
  ## create the outcome variable
  mutate(author_outcome = case_when(author == "Hamilton" ~ 1,
                                    author == "Madison" ~ -1,
                                    TRUE ~ NA_real_)) %>%
  ## drop n to reshape
  select(-n) %>%
  pivot_wider(names_from = word,
              values_from = tf_thou) %>%
  mutate_at(vars(always:`while`), replace_na, 0) %>%
  ungroup()
```

```r
hm.fit <- lm(author_outcome ~ upon + there + consequently + whilst,
             data = reg_data)
hm.fit
```

```
##
## Call:
## lm(formula = author_outcome ~ upon + there + consequently + whilst,
```

```
##     data = reg_data)
##
## Coefficients:
##  (Intercept)          upon         there  consequently
##      -0.1955        0.2128        0.1180       -0.5964
##       whilst
##      -0.9090
```

```
hm.fitted <- fitted(hm.fit) # fitted values
sd(hm.fitted)
```

```
## [1] 0.7029763
```

Cross-Validation

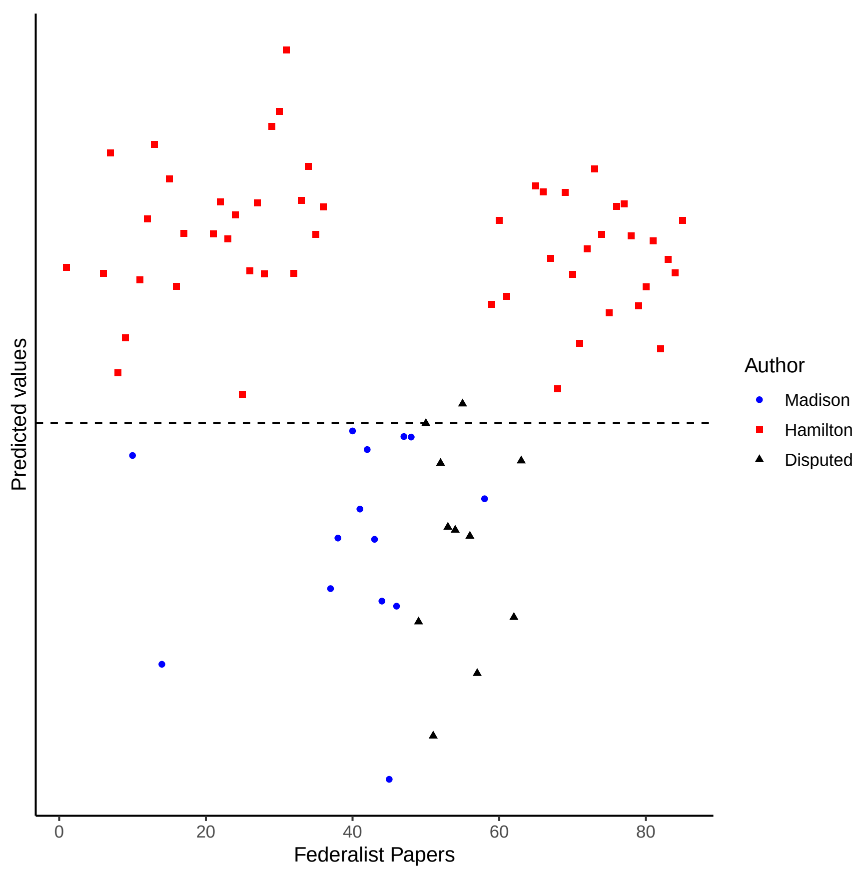
```
library(modelr)
## add author predictions
author_data <- reg_data %>%
  add_predictions(hm.fit) %>%
  mutate(pred_author = if_else(pred >= 0, "Hamilton", "Madison"))

## correct predictions rate
author_data %>%
  filter(!is.na(author_outcome)) %>%
  group_by(author) %>%
  summarize(`Proportion Correct` = mean(author == pred_author))
```

```
## # A tibble: 2 x 2
##   author   `Proportion Correct`
##   <chr>                   <dbl>
## 1 Hamilton                    1
## 2 Madison                     1
```

```
ham_mad <- filter(reg_data, !is.na(author_outcome))
n <- nrow(ham_mad)
hm.classify <- as.vector(rep(NA, n), mode = "list") # a container list
for (i in 1:n) {
  ## fit the model to the data after removing the ith observation
    sub.fit <- lm(author_outcome ~ upon + there +
                    consequently + whilst,
                  data = ham_mad[-i, ]) # exclude ith row
    ## predict the authorship for the ith observation
    hm.classify[[i]] <- slice(ham_mad, i) %>% add_predictions(sub.fit)
}
## create output table, calculate prediction rate
bind_rows(hm.classify) %>%
  mutate(pred_author = if_else(pred >= 0, "Hamilton", "Madison")) %>%
  group_by(author) %>%
  summarize(`Proportion Correct` = mean(author == pred_author))
```

```
## # A tibble: 2 x 2
##   author   `Proportion Correct`
##   <chr>                   <dbl>
## 1 Hamilton                1
## 2 Madison                 0.786
```

```
plot_data <- filter(author_data, !(author %in% c("Jay", "Joint")))

ggplot(plot_data,
       aes(x = new_id, y = pred,
           color = author, shape = author)) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  geom_point() +
  scale_y_continuous(breaks = seq(10, 80, by = 10),
                     minor_breaks = seq(5, 80, by = 5)) +
  scale_color_manual(values = c("Madison" = "blue",
                                "Hamilton" = "red",
                                "Disputed" = "black")) +
  scale_shape_manual(values = c("Madison" = 16, "Hamilton" = 15,
                                "Disputed" = 17)) +
  labs(color = "Author", shape = "Author",
       x = "Federalist Papers", y = "Predicted values")
```



Network Data

Marriage Network in Renaissance Florence

```
## load from QSS
data("florentine", package = "qss")
## print out the adjacency (sub)matrix for the first 5 families
florentine[1:5, 1:5]
```

```
##      FAMILY ACCIAIUOL ALBIZZI BARBADORI BISCHERI
## 1 ACCIAIUOL         0       0         0        0
## 2   ALBIZZI         0       0         0        0
## 3 BARBADORI         0       0         0        0
## 4  BISCHERI         0       0         0        0
## 5 CASTELLAN         0       0         1        0
```

```
## Count ties for each family
florentine %>%
  group_by(FAMILY) %>%
  rowwise() %>%
  summarize(connections = sum(c_across(ACCIAIUOL:TORNABUON)))
```

```
## # A tibble: 16 x 2
## # Groups:   FAMILY [16]
##    FAMILY    connections
##    <chr>           <int>
##  1 ACCIAIUOL           1
##  2 ALBIZZI             3
##  3 BARBADORI           2
##  4 BISCHERI            3
##  5 CASTELLAN           3
##  6 GINORI              1
##  7 GUADAGNI            4
##  8 LAMBERTES           1
##  9 MEDICI              6
## 10 PAZZI               1
## 11 PERUZZI             3
## 12 PUCCI               0
## 13 RIDOLFI             3
## 14 SALVIATI            2
## 15 STROZZI             4
## 16 TORNABUON           3
```

Undirected Graph and Centrality Measures

```
library("igraph")  # load the package

## Convert column to rownames, treat as matrix
florence <- florentine %>%
  column_to_rownames(var = "FAMILY") %>%
```

```r
  as.matrix()

## Convert adjacency matrix to graph object
florence <- graph.adjacency(florence, mode = "undirected", diag = FALSE)
```

```r
plot(florence) # plot the graph (default visualization)
```

```r
degree(florence)
```

```
## ACCIAIUOL   ALBIZZI BARBADORI  BISCHERI CASTELLAN    GINORI
##         1         3         2         3         3         1
##  GUADAGNI LAMBERTES    MEDICI     PAZZI   PERUZZI     PUCCI
##         4         1         6         1         3         0
##   RIDOLFI  SALVIATI   STROZZI TORNABUON
##         3         2         4         3
```

```r
closeness(florence)
```

```
##   ACCIAIUOL     ALBIZZI   BARBADORI    BISCHERI   CASTELLAN
## 0.018518519 0.022222222 0.020833333 0.019607843 0.019230769
##      GINORI    GUADAGNI   LAMBERTES      MEDICI       PAZZI
## 0.017241379 0.021739130 0.016949153 0.024390244 0.015384615
##     PERUZZI       PUCCI     RIDOLFI    SALVIATI     STROZZI
## 0.018518519 0.004166667 0.022727273 0.019230769 0.020833333
##   TORNABUON
## 0.022222222
```

```r
1 / (closeness(florence) * 15)
```

```
## ACCIAIUOL   ALBIZZI BARBADORI  BISCHERI CASTELLAN    GINORI
##  3.600000  3.000000  3.200000  3.400000  3.466667  3.866667
##  GUADAGNI LAMBERTES    MEDICI     PAZZI   PERUZZI     PUCCI
##  3.066667  3.933333  2.733333  4.333333  3.600000 16.000000
##   RIDOLFI  SALVIATI   STROZZI TORNABUON
##  2.933333  3.466667  3.200000  3.000000
```

```r
betweenness(florence)
```

```
## ACCIAIUOL   ALBIZZI BARBADORI  BISCHERI CASTELLAN    GINORI
##  0.000000 19.333333  8.500000  9.500000  5.000000  0.000000
##  GUADAGNI LAMBERTES    MEDICI     PAZZI   PERUZZI     PUCCI
## 23.166667  0.000000 47.500000  0.000000  2.000000  0.000000
##   RIDOLFI  SALVIATI   STROZZI TORNABUON
## 10.333333 13.000000  9.333333  8.333333
```

```r
plot(florence, vertex.size = closeness(florence) * 1000,
     main = "Closeness")
plot(florence, vertex.size = betweenness(florence),
```

Twitter Following Network

```r
## load the data
data("twitter.following", package = "qss")
data("twitter.senator", package = "qss")
## rename to be shorter
follow <- twitter.following
senator <- twitter.senator
## examine
head(follow)
```

```
##      following        followed
## 1 SenAlexander        RoyBlunt
## 2 SenAlexander     SenatorBurr
## 3 SenAlexander     JohnBoozman
## 4 SenAlexander SenJohnBarrasso
## 5 SenAlexander     SenBennetCO
## 6 SenAlexander     SenDanCoats
```

```r
head(senator)
```

```
##       screen_name            name party state
## 1    SenAlexander Lamar Alexander     R    TN
## 2        RoyBlunt       Roy Blunt     R    MO
## 3    SenatorBoxer   Barbara Boxer     D    CA
## 4 SenSherrodBrown   Sherrod Brown     D    OH
## 5     SenatorBurr    Richard Burr     R    NC
## 6  SenatorBaldwin   Tammy Baldwin     D    WI
```

```r
twitter_adj <- graph_from_edgelist(as.matrix(follow),
                                   directed = TRUE)
```

Directed Graph and Centrality

```r
senator <- mutate(senator,
         indegree = degree(twitter_adj, mode = "in"),
         outdegree = degree(twitter_adj, mode = "out"))
```

```r
## with slice and arrange
arrange(senator, desc(indegree)) %>%
  slice(1:3) %>%
  select(name, party, state, indegree, outdegree)
```

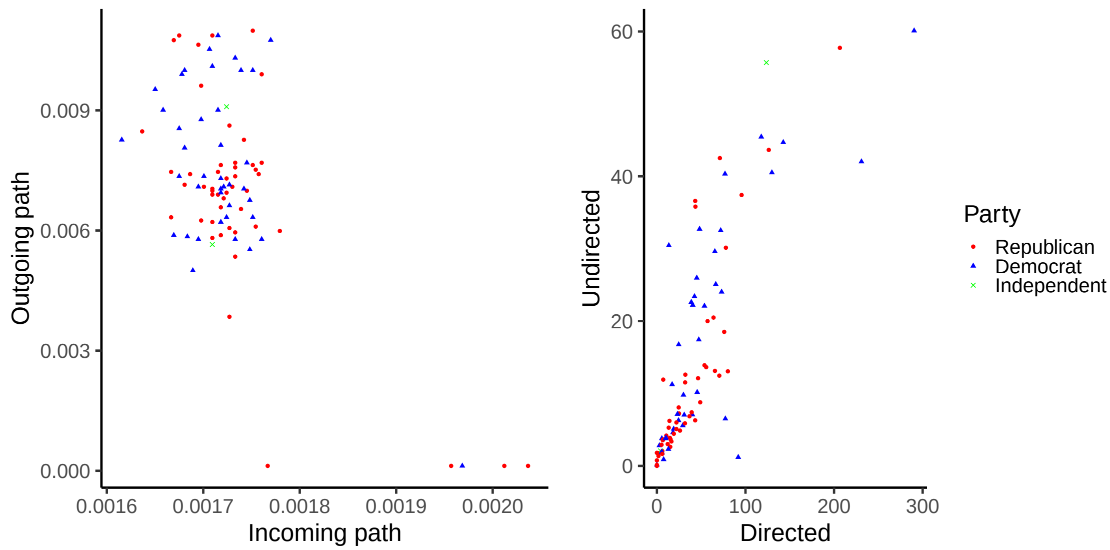
```
##                 name party state indegree outdegree
## 1         Tom Cotton     R    AR       64        15
## 2 Richard J. Durbin     D    IL       60        87
## 3     John Barrasso     R    WY       58        79
```

```r
## with slice_max
slice_max(senator, order_by = indegree, n = 3) %>%
  arrange(desc(indegree)) %>%
  select(name, party, state, indegree, outdegree)
```

```
##                name party state indegree outdegree
## 1        Tom Cotton     R    AR       64        15
## 2 Richard J. Durbin     D    IL       60        87
## 3     John Barrasso     R    WY       58        79
## 4      Joe Donnelly     D    IN       58         9
## 5    Orrin G. Hatch     R    UT       58        50
```

```r
# Define scales to reuse for the plots
scale_color_parties <- scale_color_manual("Party",
                                           values = c(R = "red",
                                                         D = "blue",
                                                         I = "green"),
                                          labels = c(R = "Republican",
                                                     D= "Democrat",
                                                     I = "Independent"))
scale_shape_parties <- scale_shape_manual("Party",
                                          values = c(R = 16,
                                                     D = 17,
                                                     I = 4),
                                          labels = c(R = "Republican",
                                                     D= "Democrat",
                                                     I = "Independent"))
## Calculate closeness measures and graph
senator %>%
  mutate(closeness_in = closeness(twitter_adj,
                                          mode = "in"),
         closeness_out = closeness(twitter_adj,
                                           mode = "out")) %>%
  ggplot(aes(x = closeness_in, y = closeness_out,
             color = party, shape = party)) +
  geom_point() +
  scale_color_parties +
  scale_shape_parties +
  labs(main = "Closeness", x = "Incoming path",
       y = "Outgoing path") +
  theme_classic(base_size = 22) +
  theme(legend.position = "none")

## Calculate betweenness measures and graph
senator %>%
  mutate(betweenness_dir = betweenness(twitter_adj,
                                            directed = TRUE),
         betweenness_undir = betweenness(twitter_adj,
                                              directed = FALSE)) %>%
  ggplot(aes(x = betweenness_dir,
             y = betweenness_undir, color = party,
             shape = party)) +
  geom_point() +
```

```
  scale_color_parties +
  scale_shape_parties +
  labs(main = "Betweenness", x = "Directed", y = "Undirected") +
  theme_classic(base_size = 22)
```




```
## Calculate the pagerank
senator <- mutate(senator,
                  page_rank = page_rank(twitter_adj)[["vector"]])

## Create igraph object
net <- graph_from_data_frame(d = follow,
                             vertices = senator,
                             directed=T)
## View the new object
net
```

```
## IGRAPH 40efe1f DN-- 91 3859 --
## + attr: name (v/c), party (v/c), state (v/c),
## | indegree (v/n), outdegree (v/n), page_rank (v/n)
## + edges from 40efe1f (vertex names):
##  [1] Lamar Alexander->Roy Blunt
##  [2] Lamar Alexander->Richard Burr
##  [3] Lamar Alexander->John Boozman
##  [4] Lamar Alexander->John Barrasso
##  [5] Lamar Alexander->Michael F. Bennet
##  [6] Lamar Alexander->Daniel Coats
##  [7] Lamar Alexander->Susan M. Collins
## + ... omitted several edges
```

```
## look at some network edges, nodes, and node (vertex) attributes
head(E(net)) ## E() for edges
```

```
## + 6/3859 edges from 40efe1f (vertex names):
```

```
## [1] Lamar Alexander->Roy Blunt
## [2] Lamar Alexander->Richard Burr
## [3] Lamar Alexander->John Boozman
## [4] Lamar Alexander->John Barrasso
## [5] Lamar Alexander->Michael F. Bennet
## [6] Lamar Alexander->Daniel Coats
```

```
head(V(net)) ## V() for vertex
```

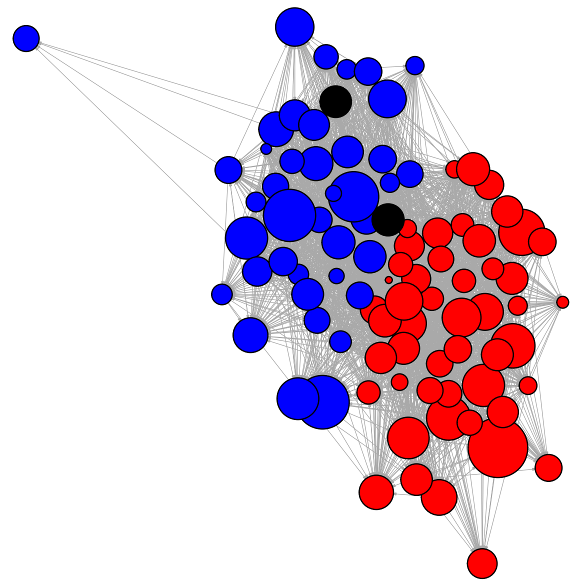
```
## + 6/91 vertices, named, from 40efe1f:
## [1] Lamar Alexander Roy Blunt       Barbara Boxer
## [4] Sherrod Brown   Richard Burr    Tammy Baldwin
```

```
head(V(net)$party)
```

```
## [1] "R" "R" "D" "D" "R" "D"
```

```
## Code will not run as-is
## Adding hypothetical weights to edges
E(net)$weight <- hypothetical_weights_vector
```

```
## Vector of colors of the nodes based on party
col <- senator %>%
  mutate(col = case_when(party == "R" ~ "red",
                         party == "D" ~ "blue",
                         TRUE ~ "black")) %>%
  select(col) %>% pull()

## plot the new object
## with node size based on page_rank
plot(net, vertex.size = V(net)$page_rank*1000,
     vertex.label = NA, vertex.color = col,
     edge.arrow.size = 0.1,
     edge.width = 0.5)
```



```r
PageRank <- function(n, A, d, pr) { # function takes 4 inputs
    deg <- degree(A, mode = "out") # outdegree calculation
    for (j in 1:n) {
        pr[j] <- (1 - d) / n +  d * sum(A[ ,j] * pr / deg)
    }
    return(pr)
}
```

```r
while (condition) {

    LOOP CONTENTS HERE

}
```

```r
nodes <- 4
## adjacency matrix with arbitrary values
adj <- matrix(c(0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0),
              ncol = nodes, nrow = nodes, byrow = TRUE)
adj
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    0    1    0    1
## [2,]    1    0    1    0
## [3,]    0    1    0    0
## [4,]    0    1    0    0
```

```r
adj <- graph.adjacency(adj)  # turn it into an igraph object
```

```r
d <- 0.85  # typical choice of constant
pr <- rep(1 / nodes, nodes) # starting values
## maximum absolute difference; use a value greater than threshold
diff <- 100
## while loop with 0.001 being the threshold
while (diff > 0.001) {
    pr.pre <- pr # save the previous iteration
    pr <- PageRank(n = nodes, A = adj, d = d, pr = pr)
    diff <- max(abs(pr - pr.pre))
}
pr
```

```
## [1] 0.2213090 0.4316623 0.2209565 0.1315563
```

Spatial Data

The 1854 Cholera Outbreak in London

Spatial Data in R

```r
data("us.cities", package = "maps")
glimpse(us.cities)
```

```
## Rows: 1,005
## Columns: 6
## $ name        <chr> "Abilene TX", "Akron OH", "Alameda CA"~
## $ country.etc <chr> "TX", "OH", "CA", "GA", "NY", "OR", "N~
## $ pop         <int> 113888, 206634, 70069, 75510, 93576, 4~
## $ lat         <dbl> 32.45, 41.08, 37.77, 31.58, 42.67, 44.~
## $ long        <dbl> -99.74, -81.52, -122.26, -84.18, -73.8~
## $ capital     <int> 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0,~
```

```r
## Filter just the continental US capitals
capitals <- filter(us.cities,
                   capital == 2,
                   !country.etc %in% c("HI", "AK"))

## Convert the USA map data from maps package to a dataframe
usa_map <- map_data("usa")

## Plot the map and capitals
ggplot() +
  geom_map(map = usa_map) +
  borders(database = "usa") +
  geom_point(aes(x = long, y = lat, size = pop),
             data = capitals) +
  # scale size area ensures: 0 = no area
  scale_size_area() +
```

```r
  coord_quickmap() +
  theme_void(base_size = 12) +
  labs(x = "", y = "",
       size = "Population")
```


```r
## Save the top 7 CA cities by population
cal_cities <- filter(us.cities, country.etc == "CA") %>%
  slice_max(pop, n = 7)

## Map the cities
ggplot() +
  borders(database = "state", regions = "California") +
  geom_point(aes(x = long, y = lat), data = cal_cities) +
  geom_text(aes(x = long, y = lat, label = name),
            position = position_jitter(width = 0.5, height = 0.5),
            data = cal_cities) +
  coord_quickmap() +
  theme_void() +
  labs(x = "", y = "")
```


```r
head(usa_map)
```

```
##        long      lat group order region subregion
## 1 -101.4078 29.74224     1     1   main      <NA>
## 2 -101.3906 29.74224     1     2   main      <NA>
## 3 -101.3620 29.65056     1     3   main      <NA>
## 4 -101.3505 29.63911     1     4   main      <NA>
## 5 -101.3219 29.63338     1     5   main      <NA>
## 6 -101.3047 29.64484     1     6   main      <NA>
```

```r
dim(usa_map)
```

```
## [1] 7243    6
```

```r
allcolors <- colors()
head(allcolors)   # some colors
```

```
## [1] "white"         "aliceblue"     "antiquewhite"
## [4] "antiquewhite1" "antiquewhite2" "antiquewhite3"
```

```r
length(allcolors) # number of color names
```

```
## [1] 657
```

```r
red <- rgb(red = 1, green = 0, blue = 0) # red
green <- rgb(red = 0, green = 1, blue = 0) # green
blue <- rgb(red = 0, green = 0, blue = 1) # blue
c(red, green, blue) # results
```

```
## [1] "#FF0000" "#00FF00" "#0000FF"
```

```r
black <- rgb(red = 0, green = 0, blue = 0) # black
white <- rgb(red = 1, green = 1, blue = 1) # white
c(black, white) # results
```

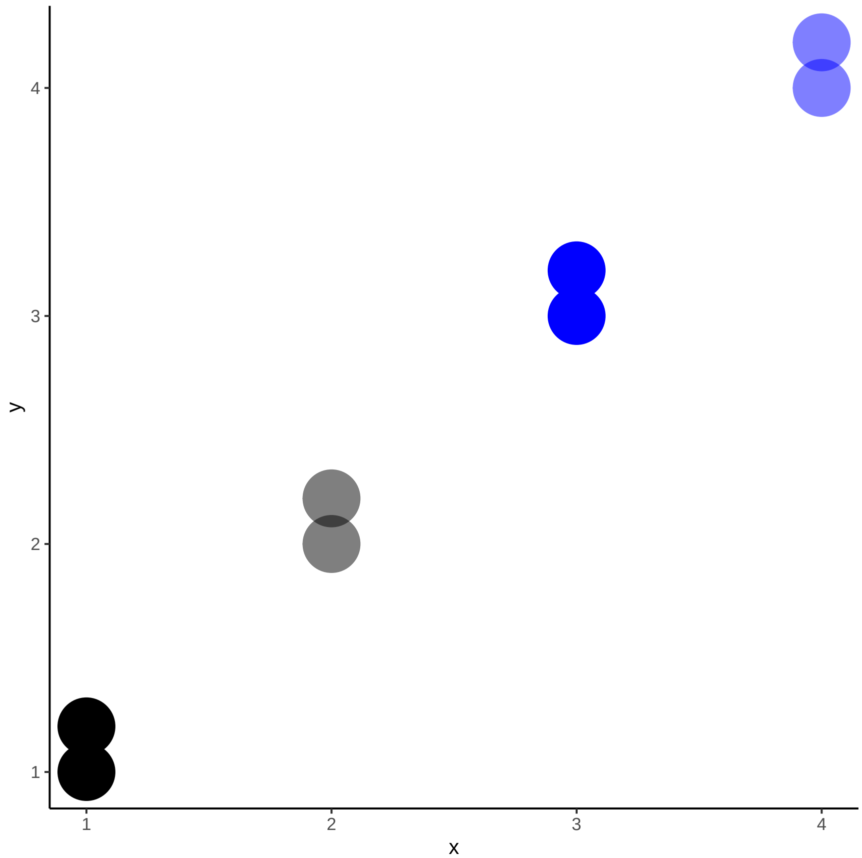
```
## [1] "#000000" "#FFFFFF"
```

```r
rgb(red = c(0.5, 1), green = c(0, 1), blue = c(0.5, 0))
```

```
## [1] "#800080" "#FFFF00"
```

```r
## semi-transparent blue
blue.trans <- rgb(red = 0, green = 0, blue = 1, alpha = 0.5)
## semi-transparent black
black.trans <- rgb(red = 0, green = 0, blue = 0, alpha = 0.5)
```

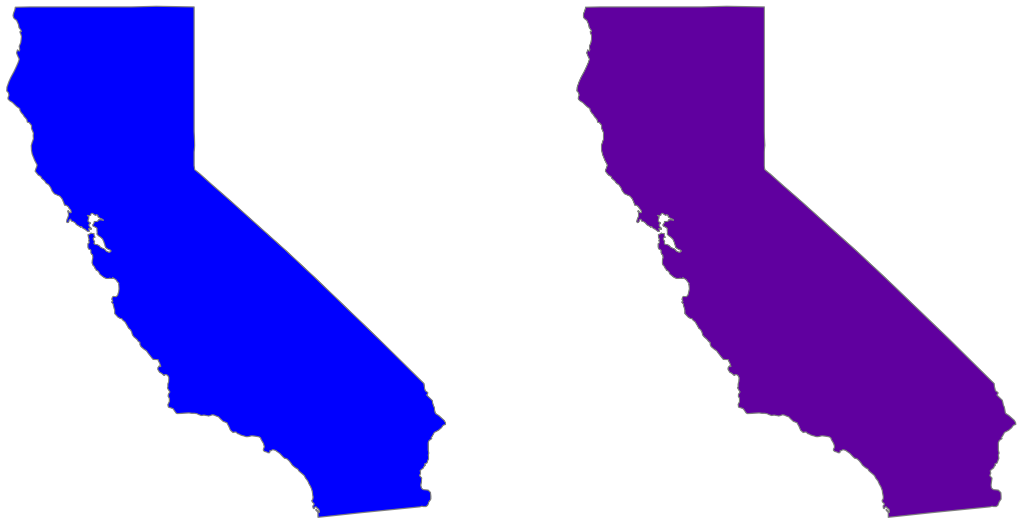
```r
## Sample data with color and alpha column
sample_data <- tibble(x = rep(1:4, each = 2),
              y = x + rep(c(0, 0.2), times = 2),
              color = rep(c("#000000", "#0000FF"), each = 4),
              alpha = c(1, 1, 0.5, 0.5, 1, 1, 0.5, 0.5))

## plot it
ggplot(aes(x = x, y = y, color = color, alpha = alpha),
       data = sample_data) +
  geom_point(size = 15) +
  scale_color_identity() +
  scale_alpha_identity()
```



United States Presidential Elections

```
## Load the data
data("pres08", package = "qss")
## Calculate vote-share
pres08 <- pres08 %>%
  mutate(Dem = Obama / (Obama + McCain),
         Rep = McCain / (Obama + McCain))

## Set the purple shade
cal_color <- filter(pres08, state == "CA") %>%
  mutate(purple_shade = rgb(red = Rep,
                            green = 0,
                            blue = Dem)) %>%
  select(purple_shade) %>% pull()

## Plot California as blue
ggplot() +
  borders(database = "state", regions = "California", fill = "blue") +
  coord_quickmap() +
  theme_void()

## Plot California as purple shade
ggplot() +
```

```
  borders(database = "state", regions = "California", fill = cal_color) +
  coord_quickmap() +
  theme_void()
```

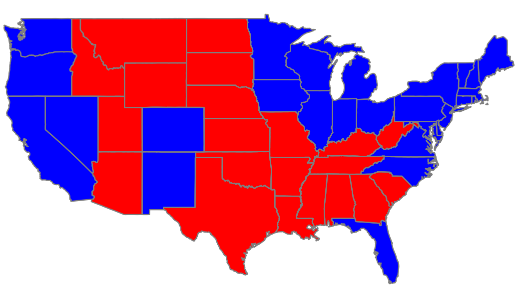


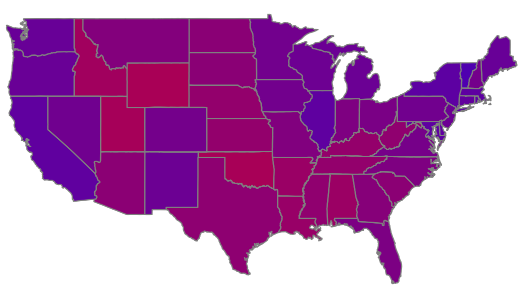
```
## prepare the presidential data for merge
## by changing case of the state variable
## and removing unneeded states (plus DC)
pres08 <- mutate(pres08, state = str_to_lower(state.name)) %>%
  filter(!(state %in% c("hawaii",
                         "d.c.",
                         "alaska")))

## take the states map data, remove DC
states <- map_data("state") %>%
  filter(!(region %in% c("hawaii",
                          "district of columbia",
                          "alaska"))) %>%
  ## merge with the presidential data
  full_join(pres08, by = c("region" = "state")) %>%
  ## create a party winner variable
  ## and a shade of winning variable
  mutate(party = if_else(Dem > Rep, "Dem", "Rep"),
         purple_shade = rgb(red = Rep,
                            green = 0,
                            blue = Dem))
## Check the data
glimpse(states)
```

```
## Rows: 15,527
## Columns: 14
## $ long        <dbl> -87.46201, -87.48493, -87.52503, -87.~
## $ lat         <dbl> 30.38968, 30.37249, 30.37249, 30.3323~
## $ group       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ order       <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12~
```

```
## $ region       <chr> "alabama", "alabama", "alabama", "ala~
## $ subregion    <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, N~
## $ state.name   <chr> "Alabama", "Alabama", "Alabama", "Ala~
## $ Obama        <int> 39, 39, 39, 39, 39, 39, 39, 39, 39, 3~
## $ McCain       <int> 60, 60, 60, 60, 60, 60, 60, 60, 60, 6~
## $ EV           <int> 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9~
## $ Dem          <dbl> 0.3939394, 0.3939394, 0.3939394, 0.39~
## $ Rep          <dbl> 0.6060606, 0.6060606, 0.6060606, 0.60~
## $ party        <chr> "Rep", "Rep", "Rep", "Rep", "Rep", "R~
## $ purple_shade <chr> "#9B0064", "#9B0064", "#9B0064", "#9B~
```

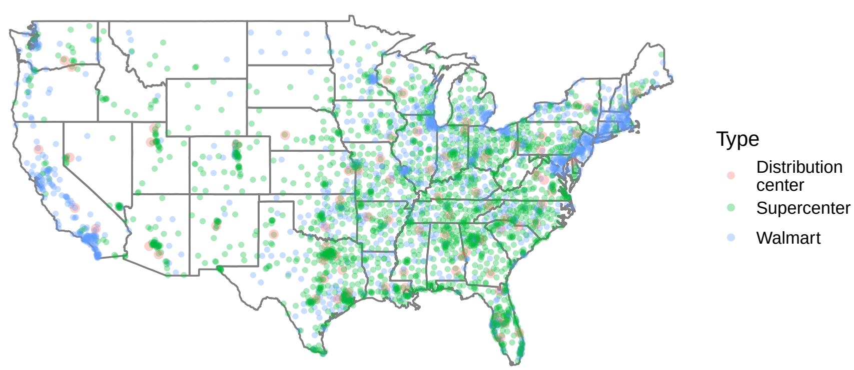
```
### Plot with red/blue
ggplot(states) +
  geom_polygon(aes(group = group, x = long, y = lat,
                   fill = party)) +
  borders(database = "state") +
  coord_quickmap() +
  scale_fill_manual(values = c("Rep" = "red", "Dem" = "blue"),
                    guide = "none") +
  theme_void() +
  labs(x = "", y = "")

## Plot with shading
ggplot(states) +
  geom_polygon(aes(group = group, x = long, y = lat,
                   fill = purple_shade)) +
  borders(database = "state") +
  scale_fill_identity() +
  coord_quickmap() +
  theme_void() +
  labs(x = "", y = "")
```





Expansion of Walmart

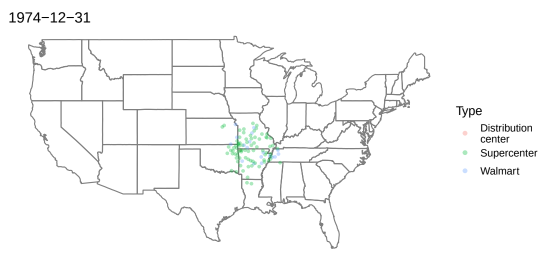
```r
## Load the data
data("walmart", package = "qss")

## add a "size" column for larger points for Distribution Centers
## Then recode the "type" levels
walmart <- walmart %>%
  mutate(size = if_else(type == "DistributionCenter", 2, 1),
         type = recode(type, "DistributionCenter" = "Distribution \ncenter",
                       "SuperCenter" = "Supercenter",
                       "Wal-MartStore" = "Walmart"))

## Map it
ggplot() +
  borders(database = "state") +
  geom_point(aes(x = long, y = lat, color = type, size = size),
             data = walmart,
             alpha = 1 / 3) +
  coord_quickmap() +
  scale_size_identity() +
  theme_void(base_size = 12) + # remove all extra formatting
  labs(color = "Type") # change the label for the legend
```

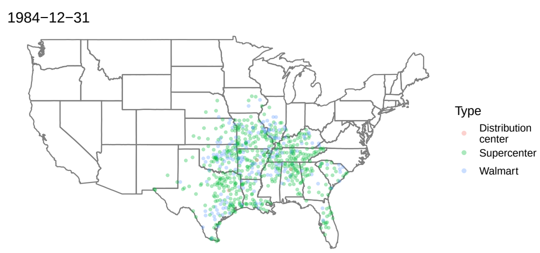


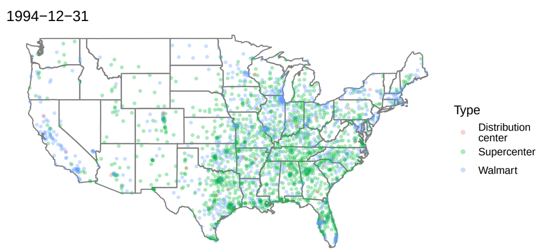
Animation in R

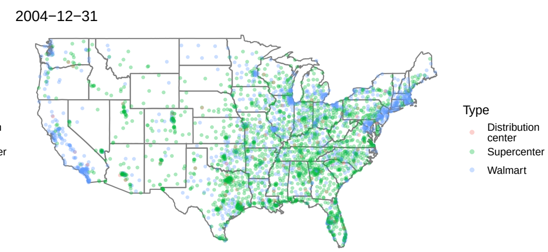
```
walmart.map <- function(data, date) {
    temp <- filter(data, opendate <= date) %>%
    mutate(size = if_else(type == "DistributionCenter", 2, 1))
    ggplot() +
    borders(database = "state") +
    geom_point(aes(x = long, y = lat, color = type, size = size),
              data = temp,
              alpha = 1 / 3) +
    coord_quickmap() +
    scale_size_identity() +
    theme_void(base_size = 12) +
    labs(color = "Type") +
    ggtitle(date)
}
```

```
walmart.map(walmart, as.Date("1974-12-31"))
walmart.map(walmart, as.Date("1984-12-31"))
walmart.map(walmart, as.Date("1994-12-31"))
walmart.map(walmart, as.Date("2004-12-31"))
```










```
library(gganimate)
library(lubridate)

## Round down to year from opendate
walmart <- walmart %>%
  mutate(year = floor_date(opendate, unit = "year"))

## Create the animation
walmart_animated <-
  ggplot() +
    borders(database = "state") +
    geom_point(aes(x = long, y = lat,
                   color = type),
               data = walmart) +
    coord_quickmap() +
    theme_void() +
    transition_states(states = year,
                      transition_length = 0,
                      state_length = 1) +
  shadow_mark()

## show the animation
walmart_animated

## save the animation
anim_save("DISCOVERY/walmart.gif")
```