Data Cleaning

Part 2: Cleaning 2024 Precinct-Level Data

Loading the Raw Data

I started by loading the raw precinct-level Statement of Vote data from the 2024 general election.

```
library(tidyverse)

# Load raw precinct data
precincts_raw <- read_csv("data/g24_sov_by_g24_svprec.csv", show_col_types =
FALSE)

# Inspect the structure
glimpse(precincts_raw)
```

```
Rows: 51,123
Columns: 76
$ COUNTY     <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1,…
$ FIPS       <chr> "06001", "06001", "06001", "06001", "06001", "06001",
"0600…
$ SVPREC     <chr> "200100", "200100A", "200200", "200200A", "201400",
"201400…
$ ADDIST     <dbl> 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14, 14,
14,…
$ SVPREC_KEY <chr> "06001200100", "06001200100A", "06001200200",
"06001200200A…
$ ELECTION   <chr> "g24", "g24", "g24", "g24", "g24", "g24", "g24", "g24",
"g2…
$ GEO_TYPE   <chr> "svprec", "svprec", "svprec", "svprec", "svprec",
"svprec",…
$ CDDIST     <dbl> 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12,
12,…
$ SDDIST     <dbl> 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
7,…
$ BEDIST     <dbl> 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2,…
$ TOTREG     <dbl> 3535, 0, 2442, 0, 3773, 0, 541, 0, 1105, 0, 948, 0, 2721,
0…
$ DEMREG     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ REPREG     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

```
0,…
$ AIPREG     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ GRNREG     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ LIBREG     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ NLPREG     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ REFREG     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ DCLREG     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ MSCREG     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ TOTVOTE    <dbl> 256, 2804, 262, 1816, 283, 2782, 89, 343, 394, 297, 837,
29…
$ DEMVOTE    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ REPVOTE    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ AIPVOTE    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ GRNVOTE    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ LIBVOTE    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ NLPVOTE    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ REFVOTE    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ DCLVOTE    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ MSCVOTE    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ PRCVOTE    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ ABSVOTE    <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0,…
$ ASSAIP01   <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0",
"0",…
$ ASSDEM01   <chr> "94", "444", "117", "348", "107", "588", "45", "105",
"181"…
$ ASSDEM02   <chr> "110", "2023", "91", "1243", "128", "1841", "24", "172",
"1…
$ ASSREP01   <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0",
"0",…
$ ASSREP02   <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0",
```

```
"0",…
$ CNGDEM01   <chr> "102", "1668", "108", "1063", "139", "1688", "35", "192",
"…
$ CNGDEM02   <chr> "102", "771", "99", "513", "98", "739", "38", "93", "143",
…
$ CNGIND01   <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0",
"0",…
$ CNGREP01   <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0",
"0",…
$ CNGREP02   <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0",
"0",…
$ PRSAIP01   <chr> "3", "10", "1", "6", "3", "13", "2", "2", "3", "4", "5",
"2…
$ PRSDEM01   <chr> "181", "2562", "207", "1647", "231", "2522", "73", "297",
"…
$ PRSGRN01   <chr> "9", "48", "13", "41", "9", "52", "0", "13", "6", "13",
"23…
$ PRSLIB01   <chr> "1", "10", "2", "3", "4", "13", "0", "0", "3", "2", "3",
"0…
$ PRSPAF01   <chr> "5", "17", "7", "12", "8", "32", "2", "4", "3", "7", "16",
…
$ PRSREP01   <chr> "51", "108", "26", "83", "17", "111", "11", "23", "55",
"24…
$ PR_2_N     <chr> "58", "493", "45", "342", "39", "399", "17", "45", "52",
"3…
$ PR_2_Y     <chr> "169", "2156", "196", "1385", "226", "2231", "66", "278",
"…
$ PR_32_N    <chr> "78", "636", "55", "439", "74", "536", "14", "60", "81",
"6…
$ PR_32_Y    <chr> "148", "1966", "187", "1261", "190", "2070", "68", "255",
"…
$ PR_33_N    <chr> "136", "1774", "105", "1092", "124", "1509", "30", "127",
"…
$ PR_33_Y    <chr> "86", "784", "126", "584", "133", "1053", "49", "177",
"231…
$ PR_34_N    <chr> "123", "1485", "121", "1027", "144", "1515", "46", "186",
"…
$ PR_34_Y    <chr> "98", "980", "105", "601", "96", "941", "33", "107",
"174",…
$ PR_35_N    <chr> "54", "581", "45", "419", "58", "563", "20", "57", "61",
"5…
$ PR_35_Y    <chr> "171", "2003", "188", "1261", "196", "1988", "58", "248",
"…
$ PR_36_N    <chr> "106", "1356", "142", "888", "146", "1487", "49", "197",
"2…
$ PR_36_Y    <chr> "118", "1223", "99", "786", "119", "1084", "31", "112",
"14…
$ PR_3_N     <chr> "51", "133", "25", "116", "33", "152", "10", "26", "38",
```

```
"2…
$ PR_3_Y     <chr> "183", "2553", "220", "1646", "240", "2508", "74", "295",
"…
$ PR_4_N     <chr> "52", "381", "37", "271", "37", "330", "14", "41", "40",
"2…
$ PR_4_Y     <chr> "181", "2294", "209", "1472", "231", "2316", "68", "279",
"…
$ PR_5_N     <chr> "94", "961", "66", "605", "63", "742", "19", "76", "72",
"6…
$ PR_5_Y     <chr> "132", "1660", "168", "1096", "197", "1862", "61", "240",
"…
$ PR_6_N     <chr> "75", "607", "59", "407", "57", "532", "17", "53", "85",
"5…
$ PR_6_Y     <chr> "143", "1958", "180", "1274", "196", "2029", "62", "257",
"…
$ SENDEM01   <chr> "107", "1719", "101", "1102", "136", "1578", "37", "153",
"…
$ SENDEM02   <chr> "103", "809", "114", "516", "105", "908", "34", "133",
"174…
$ SENREP01   <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0",
"0",…
$ SENREP02   <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0",
"0",…
$ USPDEM01   <chr> "172", "2461", "199", "1572", "217", "2444", "67", "285",
"…
$ USPREP01   <chr> "53", "155", "34", "111", "32", "153", "11", "23", "53",
"3…
$ USSDEM01   <chr> "173", "2487", "207", "1593", "222", "2478", "67", "288",
"…
$ USSREP01   <chr> "55", "155", "29", "109", "33", "151", "12", "23", "51",
"2…
```

Data Cleaning Steps

Step 1: Handle Asterisks in Vote Columns

The Statewide Database uses asterisks (*) to indicate certain data conditions. These need to be removed and converted to numeric values.

```
# Remove asterisks from all vote-related columns
precincts_clean <- precincts_raw |>
  mutate(across(matches("CNG|ASS|PRS|SEN|USP|USS|PR_"),
                ~str_replace_all(.x, "\\*", "")))
```

Step 2: Convert Vote Columns to Numeric

After removing asterisks, convert all vote columns to numeric type.

```
precincts_clean <- precincts_clean |>
  mutate(across(matches("CNG|ASS|PRS|SEN|USP|USS|PR_"),
                readr::parse_number))
```

Step 3: Ensure Correct Data Types

Make categorical variables factors where appropriate.

```
precincts_clean <- precincts_clean |>
  mutate(COUNTY = factor(COUNTY))
```

Step 4: Check for Missing Values

```
# Check for NA values in key columns
sum(is.na(precincts_clean$CNGDEM01))
```

```
[1] 4684
```

```
sum(is.na(precincts_clean$CNGREP01))
```

```
[1] 4684
```

Most NA values represent races where candidates didn't run, which is expected.

Step 5: Verify Data Integrity

```
# Check that precinct IDs are unique
n_distinct(precincts_clean$SVPREC) == nrow(precincts_clean)
```

```
[1] FALSE
```

```
# Summary of key variables
summary(precincts_clean |> select(TOTVOTE, TOTREG, CNGDEM01, CNGREP01))
```

```
    TOTVOTE            TOTREG           CNGDEM01          CNGREP01
 Min.   :      0   Min.   :      0   Min.   :      0   Min.   :      0.0
 1st Qu.:      1   1st Qu.:      0   1st Qu.:      0   1st Qu.:      0.0
 Median :     91   Median :      0   Median :     48   Median :     45.0
 Mean   :   2202   Mean   :   3081   Mean   :   1327   Mean   :    860.2
 3rd Qu.:    446   3rd Qu.:    500   3rd Qu.:    260   3rd Qu.:    169.5
 Max.   :3793980   Max.   :5745214   Max.   :2273160   Max.   :1050936.0
                                     NA's   :4684      NA's   :4684
```

Save Cleaned Data

```
write_csv(precincts_clean, "data/g24_precinct_clean.csv")
```

The cleaned data was saved to `data/g24_precinct_clean.csv` for use in subsequent analyses.

Part 5: Re-running 2024 Election with AB 604 Map

To estimate what the 2024 election results would have been under the proposed AB 604 congressional map, I used area-weighted interpolation.

Approach

The method allocates votes from SR precincts to new districts in proportion to the area of overlap. This assumes voters are distributed evenly within each precinct.

Load SR Precinct Data

```
library(sf)

# Load SR precinct voting data
SR_precincts_24 <- read_csv("data/state_g24_sov_data_by_g24_srprec.csv",
show_col_types = FALSE)

# Clean the data (remove asterisks and convert to numeric)
SR_precincts_24 <- SR_precincts_24 |>
  mutate(across(matches("CNG|ASS|PRS|SEN|USP|USS|PR_"),
                ~str_replace_all(.x, "\\*", ""))) |>
  mutate(across(matches("CNG|ASS|PRS|SEN|USP|USS|PR_"),
                readr::parse_number))
```

Load Geographic Data

```
# Load SR precinct shapefiles
srprec_shapes <- st_read("data/srprec_state_g24_v01_shp.shp", quiet = TRUE)

# Fix geometries and transform to equal-area projection
srprec_shapes <- srprec_shapes |>
  st_transform(3310) |>
  st_set_precision(1) |>
  st_make_valid() |>
  st_collection_extract("POLYGON")

# Join voting data with geometries
srprec_shapes_full <- srprec_shapes |>
  left_join(SR_precincts_24, by = "SRPREC")
```

Calculate Precinct-Level Vote Totals

```r
# Sum all Democratic and Republican congressional votes in each precinct
srprec_shapes_full <- srprec_shapes_full |>
  mutate(
    dem_precinct = rowSums(across(matches("CNGDEM\\d{2}$")), na.rm = TRUE),
    rep_precinct = rowSums(across(matches("CNGREP\\d{2}$")), na.rm = TRUE)
  )
```

Load AB 604 Proposed Map

```r
Proposed_Map_25 <- st_read("data/AB604.shp", quiet = TRUE) |>
  st_transform(3310)
```

Perform Area-Weighted Interpolation

```r
# Calculate intersections between precincts and new districts
intersections <- st_intersection(
  srprec_shapes_full |> select(SRPREC, dem_precinct, rep_precinct, geometry),
  Proposed_Map_25 |> select(DISTRICT, geometry)
)

# Calculate area weights
intersections <- intersections |>
  mutate(
    intersect_area = as.numeric(st_area(geometry)),
    precincts_area = as.numeric(
      st_area(srprec_shapes_full$geometry)[match(SRPREC,
srprec_shapes_full$SRPREC)]
    ),
    a_weighted = intersect_area / precincts_area
  )

# Allocate votes proportionally
intersections <- intersections |>
  mutate(
    dem_props = dem_precinct * a_weighted,
    rep_props = rep_precinct * a_weighted
  )
```

Aggregate Votes by New District

```r
proposed_2025_results <- intersections |>
  st_drop_geometry() |>
  group_by(DISTRICT) |>
  summarise(
    dem_votes_new = sum(dem_props, na.rm = TRUE),
    rep_votes_new = sum(rep_props, na.rm = TRUE),
    .groups = "drop"
```

```
  ) |>
  mutate(total_two_party = dem_votes_new + rep_votes_new)

# Display first few districts
head(proposed_2025_results)
```

```
# A tibble: 6 × 4
  DISTRICT dem_votes_new rep_votes_new total_two_party
  <chr>            <dbl>         <dbl>           <dbl>
1 01        15015425115.  34033073249.    49048498365.
2 02        18869825279.  13649368307.    32519193586.
3 03          176828404.     54351393.      231179797.
4 04           11048563.      5835339.       16883902.
5 05       382479921666. 535935883347.   918415805012.
6 06           27536150.     40618056.       68154206.
```

Save Results

```
write_csv(proposed_2025_results, "data/g24_ab604_district_results.csv")
```

The estimated results under the AB 604 map were saved to `data/g24_ab604_district_results.csv`.

Summary

Using area-weighted interpolation, I successfully re-allocated the 2024 election votes to the proposed AB 604 congressional districts. This allows us to compare gerrymandering metrics between the current map and the proposed map while holding voter behavior constant.