Section 6: Empirical Synthesis

Milan Svolik

4/20/2023

Load the survey data

```
load("data/df_surveys.RData")
```

Prepare the survey data for re-weighting

- I will only need vote_march and vote_june
- For re-weighting,
  - use the most detailed information available and therefore include the THIRD (PARTY) category (i.e. differentiate between those who abstain and those who vote for a third party or candidate)
  - remove NAs and the June undecided (these cannot be used for re-weighting)

```
df_surveys_synthesis <- df_surveys %>%
  select(rid, vote_march, vote_june, vote_march_3, vote_june_3) %>%
  mutate(vote_june = fct_recode(vote_june, NULL = "Undecided")) %>%
  filter(!is.na(vote_march) & !is.na(vote_june)) %>%
  mutate(vote_june = fct_drop(vote_june)) %>%
  mutate(vote_march_char = as.character(vote_march),
         vote_june_char = as.character(vote_june))
```

The survey data frame that will be re-weighted

```
library(survey)
```

```
## Loading required package: grid
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survey'
```

```
## The following object is masked from 'package:graphics':
##
##     dotchart
```

```
df_torake <- svydesign(ids=~1, data=df_surveys_synthesis)
```

```
## Warning in svydesign.default(ids = ~1, data = df_surveys_synthesis): No weights
## or probabilities supplied, assuming equal probability
```

Load the election results

```
load("data/df_2019istanbul_results.RData")
```

Summarize how aggregating lower-level election outcomes up in an evenly divided electoratethe swamps the heterogeneity in outcome shifts

The ballot box level

```
df_results %>%
  mutate(
    REG_march = AKP_march + CHP_march + ABS_march + THIRD_march,
    AKP_share_march = AKP_march/REG_march,
    CHP_share_march = CHP_march/REG_march,
    ABS_share_march = ABS_march/REG_march,
    THIRD_share_march = THIRD_march/REG_march,
    REG_june = AKP_june + CHP_june + ABS_june + THIRD_june,
    AKP_share_june = AKP_june/REG_june,
    CHP_share_june = CHP_june/REG_june,
    ABS_share_june = ABS_june/REG_june,
    THIRD_share_june = THIRD_june/REG_june,
    AKP_share_diff = AKP_share_june - AKP_share_march,
    CHP_share_diff = CHP_share_june - CHP_share_march,
    ABS_share_diff = ABS_share_june - ABS_share_march,
    THIRD_share_diff = THIRD_share_june - THIRD_share_march) %>%
  summarize(
    across(AKP_share_march:THIRD_share_diff, list(min=min, max=max))
  ) %>%
  pivot_longer(AKP_share_march_min:THIRD_share_diff_max, names_to = "statistic", values_to = "value") %
  mutate(across(value, round, 2)) %>%
  separate(col=statistic, into=c("outcome", "statistic"), sep = -3) %>%
  pivot_wider(names_from = statistic, values_from = value)
```

```
## # A tibble: 13 x 3
##    outcome              min    max
##    <chr>              <dbl>  <dbl>
##  1 AKP_share_march_    0.04   0.91
##  2 CHP_share_march_    0.01   0.83
##  3 ABS_share_march_    0.02   0.76
##  4 THIRD_share_march_  0      0.15
##  5 REG_june_          13    355
##  6 AKP_share_june_     0.02   0.89
##  7 CHP_share_june_     0.02   0.88
##  8 ABS_share_june_     0.02   0.77
##  9 THIRD_share_june_   0      0.05
## 10 AKP_share_diff_    -0.24   0.1
## 11 CHP_share_diff_    -0.06   0.2
## 12 ABS_share_diff_    -0.16   0.15
```

```
## 13 THIRD_share_diff_  -0.11   0.03
```

The neigborhood level

```
df_results %>%
  group_by(neighborhood_id) %>%
  mutate(
    REG_march = AKP_march + CHP_march + ABS_march + THIRD_march,
    REG_june = AKP_june + CHP_june + ABS_june + THIRD_june) %>%
  summarize(
    across(AKP_march:REG_june, sum)
  ) %>%
  mutate(
    REG_march = AKP_march + CHP_march + ABS_march + THIRD_march,
    AKP_share_march = AKP_march/REG_march,
    CHP_share_march = CHP_march/REG_march,
    ABS_share_march = ABS_march/REG_march,
    THIRD_share_march = THIRD_march/REG_march,
    REG_june = AKP_june + CHP_june + ABS_june + THIRD_june,
    AKP_share_june = AKP_june/REG_june,
    CHP_share_june = CHP_june/REG_june,
    ABS_share_june = ABS_june/REG_june,
    THIRD_share_june = THIRD_june/REG_june,
    AKP_share_diff = AKP_share_june - AKP_share_march,
    CHP_share_diff = CHP_share_june - CHP_share_march,
    ABS_share_diff = ABS_share_june - ABS_share_march,
    THIRD_share_diff = THIRD_share_june - THIRD_share_march) %>%
  summarize(
    across(AKP_share_march:THIRD_share_diff, list(min=min, max=max))
  ) %>%
  pivot_longer(AKP_share_march_min:THIRD_share_diff_max, names_to = "statistic", values_to = "value") %
  mutate(across(value, round, 2)) %>%
  separate(col=statistic, into=c("outcome", "statistic"), sep = -3) %>%
  pivot_wider(names_from = statistic, values_from = value)
```

```
## # A tibble: 12 x 3
##    outcome              min   max
##    <chr>              <dbl> <dbl>
##  1 AKP_share_march_    0.09  0.87
##  2 CHP_share_march_    0.03  0.71
##  3 ABS_share_march_    0.03  0.46
##  4 THIRD_share_march_  0     0.09
##  5 AKP_share_june_     0.07  0.86
##  6 CHP_share_june_     0.04  0.78
##  7 ABS_share_june_     0.03  0.36
##  8 THIRD_share_june_   0     0.05
##  9 AKP_share_diff_    -0.24  0.06
## 10 CHP_share_diff_    -0.05  0.16
## 11 ABS_share_diff_    -0.12  0.15
## 12 THIRD_share_diff_  -0.08  0.03
```

The district level

```
df_results %>%
  group_by(district_id) %>%
  mutate(
    REG_march = AKP_march + CHP_march + ABS_march + THIRD_march,
    REG_june = AKP_june + CHP_june + ABS_june + THIRD_june) %>%
  summarize(
    across(AKP_march:REG_june, sum)
  ) %>%
  mutate(
    REG_march = AKP_march + CHP_march + ABS_march + THIRD_march,
    AKP_share_march = AKP_march/REG_march,
    CHP_share_march = CHP_march/REG_march,
    ABS_share_march = ABS_march/REG_march,
    THIRD_share_march = THIRD_march/REG_march,
    REG_june = AKP_june + CHP_june + ABS_june + THIRD_june,
    AKP_share_june = AKP_june/REG_june,
    CHP_share_june = CHP_june/REG_june,
    ABS_share_june = ABS_june/REG_june,
    THIRD_share_june = THIRD_june/REG_june,
    AKP_share_diff = AKP_share_june - AKP_share_march,
    CHP_share_diff = CHP_share_june - CHP_share_march,
    ABS_share_diff = ABS_share_june - ABS_share_march,
    THIRD_share_diff = THIRD_share_june - THIRD_share_march) %>%
  summarize(
    across(AKP_share_march:THIRD_share_diff, list(min=min, max=max))
  ) %>%
  pivot_longer(AKP_share_march_min:THIRD_share_diff_max, names_to = "statistic", values_to = "value") %
  mutate(across(value, round, 2)) %>%
  separate(col=statistic, into=c("outcome", "statistic"), sep = -3) %>%
  pivot_wider(names_from = statistic, values_from = value)
```

```
## # A tibble: 12 x 3
##    outcome              min   max
##    <chr>              <dbl> <dbl>
##  1 AKP_share_march_    0.15  0.54
##  2 CHP_share_march_    0.24  0.64
##  3 ABS_share_march_    0.14  0.23
##  4 THIRD_share_march_  0.01  0.04
##  5 AKP_share_june_     0.13  0.54
##  6 CHP_share_june_     0.27  0.71
##  7 ABS_share_june_     0.14  0.21
##  8 THIRD_share_june_   0     0.01
##  9 AKP_share_diff_    -0.07  0
## 10 CHP_share_diff_     0.03  0.1
## 11 ABS_share_diff_    -0.05  0.01
## 12 THIRD_share_diff_  -0.03 -0.01
```

The city level

```
df_results %>%
  mutate(city_id=1) %>%
  group_by(city_id) %>%
```

```
  mutate(
    REG_march = AKP_march + CHP_march + ABS_march + THIRD_march,
    REG_june = AKP_june + CHP_june + ABS_june + THIRD_june) %>%
  summarize(
    across(AKP_march:REG_june, sum)
  ) %>%
  mutate(
    REG_march = AKP_march + CHP_march + ABS_march + THIRD_march,
    AKP_share_march = AKP_march/REG_march,
    CHP_share_march = CHP_march/REG_march,
    ABS_share_march = ABS_march/REG_march,
    THIRD_share_march = THIRD_march/REG_march,
    REG_june = AKP_june + CHP_june + ABS_june + THIRD_june,
    AKP_share_june = AKP_june/REG_june,
    CHP_share_june = CHP_june/REG_june,
    ABS_share_june = ABS_june/REG_june,
    THIRD_share_june = THIRD_june/REG_june,
    AKP_share_diff = AKP_share_june - AKP_share_march,
    CHP_share_diff = CHP_share_june - CHP_share_march,
    ABS_share_diff = ABS_share_june - ABS_share_march,
    THIRD_share_diff = THIRD_share_june - THIRD_share_march) %>%
  summarize(
    across(AKP_share_march:THIRD_share_diff, list(min=min, max=max))
  ) %>%
  pivot_longer(AKP_share_march_min:THIRD_share_diff_max, names_to = "statistic", values_to = "value") %>
  mutate(across(value, round, 2)) %>%
  separate(col=statistic, into=c("outcome", "statistic"), sep = -3) %>%
  pivot_wider(names_from = statistic, values_from = value)
```

```
## # A tibble: 12 x 3
##    outcome              min   max
##    <chr>              <dbl> <dbl>
##  1 AKP_share_march_    0.39  0.39
##  2 CHP_share_march_    0.39  0.39
##  3 ABS_share_march_    0.19  0.19
##  4 THIRD_share_march_  0.02  0.02
##  5 AKP_share_june_     0.37  0.37
##  6 CHP_share_june_     0.45  0.45
##  7 ABS_share_june_     0.17  0.17
##  8 THIRD_share_june_   0.01  0.01
##  9 AKP_share_diff_    -0.02 -0.02
## 10 CHP_share_diff_     0.05  0.05
## 11 ABS_share_diff_    -0.02 -0.02
## 12 THIRD_share_diff_  -0.01 -0.01
```

Post-stratify

- Post-stratification requires non-zero cell counts in the population. This is only an issue with THIRD (PARTY VOTES), which are zero in a number of boxes.
- Solve this by replacing 0 with .000001, but only for the purposes of post-stratification

```
df_results_synthesis <- df_results %>%
  select(box_id, ends_with(c("march", "june"))) %>%
  mutate(THIRD_march = ifelse(THIRD_march==0, .000001, THIRD_march),
```

```
        THIRD_june = ifelse(THIRD_june==0, .000001, THIRD_june))
```

Go ballot box by ballot box

- this takes about 20hrs; to run remove "eval = FALSE"
- note that post-stratifying takes into account third-party voters, but the joint PMF (and mechanisms) only employs the outcomes AKP, CHP, ABS
- the bootstrap sample is the size of the ballot box, to calibrate uncertainty involved in having a sample of that size
- instead of saving the PMF, save the actual NUMBER of voters in the cross-tab: this will make it easy to aggregate up

```
boot_rep <- 1000
n_boxes <- length(df_results_synthesis$box_id)

# SAVE THE MATRIX OF BOOTSTRAPPED Ps INTO THIS LIST, ONE ENTRY PER BALLOT BOX
box_list <- vector("list", n_boxes)

for (i in 1:n_boxes) {

  # Load ballot box level result
  box_id <- df_results_synthesis$box_id[i]
  AKP_march <- df_results_synthesis$AKP_march[i]
  CHP_march <- df_results_synthesis$CHP_march[i]
  ABS_march <- df_results_synthesis$ABS_march[i]
  THIRD_march <- df_results_synthesis$THIRD_march[i]
  REG_march <- AKP_march + CHP_march + ABS_march + THIRD_march

  AKP_june <- df_results_synthesis$AKP_june[i]
  CHP_june <- df_results_synthesis$CHP_june[i]
  ABS_june <- df_results_synthesis$ABS_june[i]
  THIRD_june <- df_results_synthesis$THIRD_june[i]

  # Construct the margins that will be used for post-stratifying
  march_freq <- data.frame(vote_march_char=c("AKP","CHP","ABS", "THIRD"), Freq=c(AKP_march, CHP_march, A
  june_freq <- data.frame(vote_june_char=c("AKP","CHP","ABS", "THIRD"), Freq=c(AKP_june, CHP_june, ABS_
  df_raked <- rake(df_torake, list(~vote_march_char, ~vote_june_char), list(march_freq, june_freq), con

  # Extract weights and use them as bootstrap probabilities
  df_toboot <- df_surveys_synthesis %>%
    mutate(rid = seq(1:nrow(df_surveys_synthesis))) %>%
    mutate(weight = weights(df_raked)) %>%
    select(rid, vote_march, vote_june, vote_march_3, vote_june_3, weight)

  # BOOTSTRAP
  p_boot_save <- c()
  for (j in 1:boot_rep) {
    # DRAW RIDS
    rid_boot <- sample(x=df_toboot$rid, size= REG_march, replace=TRUE, prob=df_toboot$weight)
    # EXTRACT THE DATA BASED ON THE DRAW
    df_boot <- df_toboot[rid_boot,]
    # COMPUTE THE CROSS-TAB
    p_boot <- table(df_boot$vote_march_3, df_boot$vote_june_3)
    #rowSums(p_boot)
```

```
    #colSums(p_boot)
    # TRANSFORM TO A VECTOR THAT IS A ROWWISE TRANSFORMATION OF THE MATRIX
    p_boot <- c(t(p_boot))
    # SAVE IT
    p_boot_save <- rbind(p_boot_save, p_boot)
  }

box_list[[i]] <- p_boot_save
}
save(box_list, file = "temp/box_list.RData")
```

AGGREGATE THE JOINT PMF/CROSS-TAB TO THE CITY LEVEL

- Takes about 12hrs; to run remove "eval = FALSE"
- Go boot-draw by boot-draw: each entry in the list is a ballot box, each row in the ballot box df is one bootstrap draw
- p_boot_city: the city-level P matrix for each boot draw

```
load("temp/box_list.RData")

# LOAD THE RESULTS DATA TO GET NUMBER OF BOXES
load("data/df_2019istanbul_results.RData")

boot_rep <- 1000
p_boot_city_list <- vector("list", boot_rep)
n_boxes <- length(df_results$box_id)

p_boot_city <- c()
for (j in 1:boot_rep) {
  #print(j)
  p_boot_extract <- c()
  for (i in 1:n_boxes) {
    # EXTRACT jTH BOOT DRAW FOR ALL BOXES
    p_boot_extract <- rbind(p_boot_extract, box_list[[i]][j,])
    # AGGREGATE jTH BOOT DRAW TO THE CITY LEVEL
    p_boot_city_j <-  apply(p_boot_extract, 2, sum)
  }
  p_boot_city_list[[j]] <- p_boot_city_j
}

# TURN THE LIST INTO A DATA FRAME AND SAVE
save(p_boot_city_list, file = "temp/p_boot_city.RData")
```

Compute the mechanisms at the CITY LEVEL

```
load("temp/p_boot_city.RData")

boot_rep <- 1000
#boot_rep <- 10
```

```
P_boot <- c()
switching_boot <- c()
backlash_boot <- c()
disenchantment_boot <- c()

for (k in 1:boot_rep) {
  P_boot <- rbind(P_boot, p_boot_city_list[[k]])
  P <- matrix(p_boot_city_list[[k]], nrow=3, byrow=TRUE)

  switching <- P[1,2] - P[2,1]
  switching_boot <- rbind(switching_boot, switching)

  backlash <- P[3,2] - P[2,3]
  backlash_boot <- rbind(backlash_boot, backlash)

  disenchantment <- P[1,3] - P[3,1]
  disenchantment_boot <- rbind(disenchantment_boot, disenchantment)
}

#### Summarize the transition matrix
P_boot_mean <- apply(P_boot, 2, mean, na.rm=TRUE)
P_boot_mean <- matrix(P_boot_mean, nrow=3, byrow=TRUE)
P_boot_lower <- apply(P_boot, 2, quantile, probs=c(.025), na.rm=TRUE)
P_boot_lower <- matrix(P_boot_lower, nrow=3, byrow=TRUE)
P_boot_upper <- apply(P_boot, 2, quantile, probs=c(.975), na.rm=TRUE)
P_boot_upper <- matrix(P_boot_upper, nrow=3, byrow=TRUE)

#### Summarize the mechanisms
switching_boot_mean <- apply(switching_boot, 2, mean, na.rm=TRUE)
switching_boot_lower <- apply(switching_boot, 2, quantile, probs=c(.025), na.rm=TRUE)
switching_boot_upper <- apply(switching_boot, 2, quantile, probs=c(.975), na.rm=TRUE)

backlash_boot_mean <- apply(backlash_boot, 2, mean, na.rm=TRUE)
backlash_boot_lower <- apply(backlash_boot, 2, quantile, probs=c(.025), na.rm=TRUE)
backlash_boot_upper <- apply(backlash_boot, 2, quantile, probs=c(.975), na.rm=TRUE)

disenchantment_boot_mean <- apply(disenchantment_boot, 2, mean, na.rm=TRUE)
disenchantment_boot_lower <- apply(disenchantment_boot, 2, quantile, probs=c(.025), na.rm=TRUE)
disenchantment_boot_upper <- apply(disenchantment_boot, 2, quantile, probs=c(.975), na.rm=TRUE)

# CONSOLIDATE
switching_boot_out <- c(switching_boot_mean, switching_boot_lower, switching_boot_upper)
backlash_boot_out <- c(backlash_boot_mean, backlash_boot_lower, backlash_boot_upper)
disenchantment_boot_out <- c(disenchantment_boot_mean, disenchantment_boot_lower, disenchantment_boot_u
```

SUMMARIZE THE CITY-WIDE MECHS AND SHIFTS

```
round(P_boot_mean)
```

```
##         [,1]    [,2]    [,3]
## [1,] 3631659  298230  219737
## [2,]   87973 4036215   37792
```

```
## [3,]  210371  399198 1633581
```

```
round(P_boot_lower)
```

```
##         [,1]    [,2]    [,3]
## [1,] 3628615  297068  218874
## [2,]   87445 4033297   37415
## [3,]  209503  397971 1631352
```

```
round(P_boot_upper)
```

```
##         [,1]    [,2]    [,3]
## [1,] 3634496  299277  220630
## [2,]   88537 4039282   38171
## [3,]  211277  400478 1635709
```

```
round(P_boot_lower - P_boot_upper)
```

```
##       [,1]  [,2]  [,3]
## [1,] -5881 -2208 -1756
## [2,] -1092 -5985  -756
## [3,] -1774 -2508 -4357
```

```
round(100*switching_boot_out/sum(P), 2)
```

```
## [1] 1.99 1.98 2.00
```

```
round(100*backlash_boot_out/sum(P), 2)
```

```
## [1] 3.42 3.41 3.44
```

```
round(100*disenchantment_boot_out/sum(P), 2)
```

```
## [1] 0.09 0.08 0.10
```

SUMMARIZE THE NUMBER OF VOTERS THAT PARTICIPATED IN EACH MECHANISM

```
P_boot_mean
```

```
##            [,1]      [,2]       [,3]
## [1,] 3631658.73  298230.1  219737.47
## [2,]   87973.18 4036215.1   37792.03
## [3,]  210371.48  399197.5 1633581.34
```

```
count_total <- sum(P_boot_mean)
count_total
```

```
## [1] 10554757
```

```
# NO CHANGE IN THEIR ACTION
count_nochange <- P_boot_mean[1,1] + P_boot_mean[2,2] + P_boot_mean[3,3]
count_nochange
```

```
## [1] 9301455
```

```
round(count_nochange/10^6, 1)
```

```
## [1] 9.3
```

```
count_nochange/count_total
```

```
## [1] 0.8812572
```

```
# CHANGE
count_total - count_nochange
```

```
## [1] 1253302
```

```
round((count_total - count_nochange)/10^6, 1)
```

```
## [1] 1.3
```

```
(count_total - count_nochange)/count_total
```

```
## [1] 0.1187428
```

```
# SWITCHING
P_boot_mean[1,2]
```

```
## [1] 298230.1
```

```
P_boot_mean[2,1]
```

```
## [1] 87973.18
```

```
P_boot_mean[1,2] - P_boot_mean[2,1]
```

```
## [1] 210257
```

```
# BACKLASH
P_boot_mean[3,2]
```

```
## [1] 399197.5
```

```
P_boot_mean[2,3]
```

```
## [1] 37792.03
```

```
P_boot_mean[3,2] - P_boot_mean[2,3]
```

```
## [1] 361405.5
```

```
# DISENCHANTMENT
P_boot_mean[1,3]
```

```
## [1] 219737.5
```

```
P_boot_mean[3,1]
```

```
## [1] 210371.5
```

```
P_boot_mean[1,3] - P_boot_mean[3,1]
```

```
## [1] 9365.99
```