Estimating stateless populations from identified at-risk populations

Contents

1 Overview

This section provides a step-by-step summary of a method to estimate the number of people at-risk of statelessness in a country of interest and proposes a way of translating at-risk populations to estimates of stateless populations. The method relies on data about migrant groups that is readily available for a large number of countries, and is probably most relevant for high-income countries that are traditionally high-migration countries (for example, USA, Canada, Australia). To be able to produce reasonable estimates of stateless persons, the method relies heavily on expert opinion about the likelihood that those who are at-risk are actually stateless. The case study presented here is a 'proof-of-concept' which could potentially be extended in future.

1.1 Case study

We discuss the case study of estimating stateless populations in the United States. We use data from the American Community Survey to extract information about the size of immigrant populations that are potentially at risk of statelessness. We also illustrate a modeling strategy that incorporates the probability of an individual with certain characteristics is stateless. This approach is illustrated in adjusting the Nepal immigrant population by their year of arrival.

2 Data requirements

2.1 Ideal requirements

This method relies on national-level survey or census data that collects information on different migrant groups. Ideally, variables available would include:

- Country of birth
- Year of arrival
- Age, sex, marital status
- Ethnic group/ancestory
- Language spoken at home
- Ability to speak English (or official language of country of interest)
- Characteristics of parents and/or spouse and/or children, including nativity, citizenship, language, etc

2.2 Minimum requirements

Many of the variables stated above are potentially useful in helping to assess the likelihood that a person at-risk of statelessness is actually stateless, however, they are not essential in defining the

at-risk groups. For the latter, the most important variables are:

- Country of birth
- Year of arrival
- Age, sex, marital status
- Ethnic group/ancestory

An individual's ethnic group/ancestory is important as it is often the case that only certain sub-populations from a country are at risk of statelessness (e.g. the Shona or Pemba from Kenya). If this information isn't available then some profiles may not be able to be identified.

3 Method

3.1 Background and rationale of method

This methods starts from a point of defining a set of profiles of migrant populations that are at-risk of statelessness in a particular country of interest. The profiles are defined based on previous knowledge about origin and destination citizenship requirements, laws and changes in legal definitions over time, and historical state secession, for example. Once a complete set of profiles can be identified, data on migrants by origin and other characteristics can then be used to get estimates of the size of at-risk populations. Additionally, the population size of each at-risk group can be estimated and projected over time with uncertainty, based on statistical time series models.

Based on the above, the best we can do is get an estimate of the population at risk of statelessness. However, if we had additional information on the likelihood of statelessness based on various individual characteristics then we could incorporate this information into the above model. There are many potential characteristics that could be associated with the likelihood of being stateless; for example education, English language ability, marital status, etc. With the help of expert knowledge, these variables could be codified into a probability function that allows the likely number of stateless persons to be estimated from at-risk populations.

In general, there is likely to be a substantial difference in the number of people identified as 'at-risk' based on a pre-defined set of criteria, and those who are actually stateless. As such, the step of defining likelihood of statelessness based on expert knowledge and other variables is the most important.

3.2 Assumptions

The main assumptions behind this method are:

- Stateless persons are equally likely to be captured in the data source as the general population of migrants from the same country
- Information on citizenship and other characteristics is reported accurately
- Information used to define at-risk profiles is up-to-date and accurate
- The likelihood that migrants from particular origin countries are stateless can be well-captured by expert opinion and//or other variables that are available

4 Step-by-step example

4.1 Data

We used data from the 1-year American Community Surveys (ACS) over the period 2005–2019. The ACS is a nationally representative survey that covers roughly 1 percent of the US population. Each year, it gathers detailed data for all states and for cities, counties, metropolitan statistical areas, and Public Use Microdata Areas (PUMAs), a US Census–defined geographic area of 100,000 people or more. We only considered sample respondents who foreign-born and were not US Citizens. We extracted variables on age, sex, place of birth, nativity, ancestry, year of entry, household language, nativity of parents, and state of residence.

To identify these groups of populations that are potentially stateless or potentially at risk of statelessness in the United States, we largely followed the approach of Kerwin et al. (2020). In their paper, Kerwin et al. developed at-risk profiles based on reviewing UNHCR 'Mapping Statelessness' reports, and other consultations and interviews. For more information on the groups, please see the case studies document.

4.2 Empirical analysis

All calculations were down in the statistical software package R. Data were extracted from the US Census Bureau's API using the `R` statistical language and the `censusapi` package. Implementation of the statistical model was in `Stan`.

4.3 Scripts

The data extraction and analysis relies on four scripts, which are in the project repo in the `code` folder. A summary of the purpose of each script is as follows:

- `1_get_acs_data`: extract migration data from ACS using the `censusapi` package and save the resulting data
- `2_at_risk`: extract at-risk populations from data above and save the results
- `3_descriptives`: plot some descriptive charts of the at risk populations — not necessary for modeling and not discussed any further here, but useful to reproduce graphs presented in the case studies document
- `4_model`: code to fit time series projection model on at-risk populations, and also to fit probability-adjusted model to Nepalese population

4.4 Step 1: Get ACS data

The first step is to extract data from the ACS. This relates to code in the `1_get_acs_data` script. Here we rely on the `censusapi` package to pull data. We need a few other auxiliary packages, listed below:

```
library(tidyverse)
library(censusapi)
library(tidycensus)
library(janitor)
library(srvyr)
library(ggrepel)
library(rstan)
library(tidybayes)
```

The code below loops of the years of interest, and for each year, extracts the variables of interest, which include age, nativity, place of birth, language, and year of entry into the United States. The data for each year are then saved into an .RDS file to be used in the next step.

```
rep_weights <- paste0("PWGTP", 1:80)

years <- as.character(2005:2019)



# Gets the  public microdata for non-US citizens only

for(i in 1:length(years)){
  print(years[i])
  lang_var <- ifelse(as.numeric(years[i])<2016, "HHL", "HHLANP")
  if(years[i]>2007){
    df <- getCensus(
      name = "acs/acs1/pums/",
      vintage = years[i],
      vars = c("PWGTP", #weight
               rep_weights,
               "AGEP", # age
               #"ENG", # ability to speak english
               "NATIVITY", # nativity
               "NOP", # nativity of parents
               "SEX", #sex
```

```
                "ANC1P", # ancestory
                lang_var, ## HH language
                "YOEP", ## year of entry
                "POBP"), # place of birth
      region = "state:*",
      CIT="2,3,5")
  }
  else{
    df <- getCensus(
      name = "acs/acs1/pums/",
      vintage = years[i],
      vars = c("PWGTP", #weight
                rep_weights,
                "AGEP", # age
                #"ENG", # ability to speak english
                "NATIVITY", # nativity
                #"NOP", # nativity of parents
                "SEX", #sex
                "ANC1P", # ancestory
                lang_var, ## HH language
                "YOEP", ## year of entry
                "POBP"), # place of birth
      region = "state:*",
      CIT="2,3,5")
  }



  write_rds(df, paste0("data/acs/acs1_", years[i], ".RDS"))



}
```

4.5 Step 2: Extract at risk populations

Now we have the broad ACS data, the next step is to extract the populations potentially at risk of statelessness, as covered in the `2_at_risk` script. This is done by translating the profiles listed in

the Kerwin paper into code that extracts migrants based on variables pulled in the first step. The code below is long, but in essence it loops through the years of interests, and for each at-risk profile pulls out persons based on the definition in the profiles. Each block below starts with a comment which indiactes the specific profile that the code refers to. Note that there are some profiles listed in the comments with no code; these are the cases where the ACS did not contain enough information to identify people in these groups. At the end of each loop, the at-risk populations by year are saved into an .RDS file to be used in the next step.

```
years <- 2005:2019

for(i in 1:length(years)){

  year <- years[i]
  print(year)
  # read in and clean up

  d <- read_rds(paste0("data/acs/acs1_", year,".RDS"))
  if(("nop" %in% colnames(d))==FALSE){
    d$nop <- NA
  }

  d <- d %>%
    clean_names() %>%
    mutate(nativity = ifelse(nativity ==1, "native", "foreign born")) %>%
    mutate(citizen = ifelse(cit == 2, "born in territories",
                            ifelse(cit==3, "born abroad US parents", "non-citizen"))) %>%
  mutate(nativity_parents = case_when(
    nop==0 ~ "NA",
    nop ==1 ~ "Living with two parents: Both parents NATIVE",
    nop == 5 ~ "Living with father only: Father NATIVE",
    nop == 8 ~ "Living with mother only: Mother FOREIGN BORN",
    nop == 7 ~ "Living with mother only: Mother NATIVE",
    nop == 3 ~ "Living with two parents: Mother only FOREIGN BORN",
    nop == 4 ~ "Living with two parents: BOTH parents FOREIGN BORN",
    nop == 6 ~ "Living with father only: Father FOREIGN BORN",
    nop == 2 ~ "Living with two parents: Father only FOREIGN BORN",
    TRUE ~ "NA"
  ))
  #mutate(speaks_only_english = ifelse(eng == 0, 1, 0)) %>%
  #mutate(english_only_hh = ifelse(hhl==1, 1, 0))
```

```
  d <- d %>%
    mutate(yob = year - as.numeric(agep))

  d <- d %>% filter(nativity == "foreign born")

  # Extract populations ------------------------------------------------

  # empty tibble to save
  ar_all <- tibble()

  ############ Former soviet countries arrived before 1992
  # no tajkistan, turkmenistan, estonia

  ar_all <- bind_rows(ar_all, d %>%
                        filter(pobp %in% c(156, 165, 160, 162, 164, 158, 159, 161, 218, 246, 15
                        filter(yoep<1992) %>%
                        mutate(group = "Former Soviet Union") %>%
                        mutate(region = "Europe and Eurasia"))


  # Estonia (No data)

  # Latvia: ancestory russia, belarus, polish, born before 1991
  ar_all <- bind_rows(ar_all, d %>%
                        filter(pobp == 156) %>%
                        filter(anc1p==148|anc1p==102|anc1p==142)%>%
                        mutate(group = "Ethnic Russians, Belarussians, and Poles
from Latvia")%>%
                        mutate(region = "Europe and Eurasia"))

  # Lithuania:

  ar_all <- bind_rows(ar_all, d %>%
                        filter(pobp==157) %>%
                        filter(anc1p==148|anc1p==102|anc1p==142)%>%
                        mutate(group = "Members of Ethnic Minority Groups from
Lithuania")%>%
                        mutate(region = "Europe and Eurasia"))
```

```
# Ethnic Armenians from Azerbaijan and Georgia

ar_all <- bind_rows(ar_all,d %>%
                      filter((pobp==159& anc1p==431)|(pobp==161 & anc1p==431))%>%
                      mutate(group = "Ethnic Armenians from Azerbaijan and Georgia") %>%
                      mutate(region = "Europe and Eurasia"))

# ethnic Azerbaijanis from Georgia
# no azerbaijan ethnicity

# Meskhetian Turks born in georgia, russia, USSR, Uzbekistan

# ar_all <- bind_rows(ar_all,d %>%
#   filter(pobp %in% c(161, 163, 165, 246)) %>%
#   filter(hhlanp==1675) )

# OR

ar_all <- bind_rows(ar_all, d %>%
                      filter(pobp %in% c(161, 163, 165, 246)) %>%
                      filter(anc1p==434)%>%
                      mutate(group = "Meskhetian Turks born in Georgia, Russia, USSR, Uzbeki
                      mutate(region = "Europe and Eurasia"))

### Yugoslavia

# Roma

ar_all <- bind_rows(ar_all, d %>% filter(anc1p==124) %>%
                      filter(pobp %in% c(100, 151, 154, 168, 152, 147, 150)) %>%
                      mutate(group = "Roma")%>%
                      mutate(region = "Europe and Eurasia"))

# Balkan egyptians

ar_all <- bind_rows(ar_all,d %>% filter(anc1p==402) %>%
                      filter(pobp %in% c(100, 154, 168, 152, 147))%>%
                      mutate(group = "Balkan Egyptians")%>%
```

```
                      mutate(region = "Europe and Eurasia"))

# Yugoslavian passports

ar_all <- bind_rows(ar_all,d %>%
                      filter(anc1p== 176) %>%
                      filter(yoep<1992) %>%
                      mutate(group = "Yugoslavian passports")%>%
                      mutate(region = "Europe and Eurasia"))

# Born in North Macedonia, Other Ex-Yugoslav Descent

ar_all <- bind_rows(ar_all,d %>%
                      filter(pobp==152) %>%
                      filter(anc1p %in% c(109, 154, 177, 152, 131)) %>%
                      mutate(group = "Born in North Macedonia, Other Ex-Yugoslav Descent")%>%
                      mutate(region = "Europe and Eurasia"))

# Born in Croatia, Serbian Descent

ar_all <- bind_rows(ar_all,d %>%
                      filter(pobp==151) %>%
                      filter(anc1p==152) %>%
                      mutate(group = "Born in Croatia, Serbian Descent")%>%
                      mutate(region = "Europe and Eurasia"))

# roma in italy and germany

ar_all <- bind_rows(ar_all,d %>% filter(anc1p==124) %>%
                      filter(pobp %in% c(110, 120)) %>%
                      mutate(group = "Roma Born in Italy and Germany")%>%
                      mutate(region = "Europe and Eurasia"))


## Middle east and north africa

# Syrian Refugee Children Born Abroad

ar_all <- bind_rows(ar_all, d %>%
```

```
                        filter(anc1p==429) %>%
                        filter(pobp %in% c(110, 216, 224, 136, 243, 134, 139, 119, 140)) %>%
                        mutate(group = "Syrian Refugee Children Born Abroad") %>%
                        mutate(region = "Middle East and North Africa"))

# Feyli Kurds from Iraq

ar_all <- bind_rows(ar_all,d %>% filter(pobp==213) %>%
                        filter(anc1p==442) %>%
                        filter(yoep>1980) %>%
                        mutate(group = "Feyli Kurds from Iraq")%>%
                        mutate(region = "Middle East and North Africa"))

# Syrian Kurds

ar_all <- bind_rows(ar_all,d %>% filter(pobp==239) %>%
                        filter(anc1p==442) %>%
                        mutate(group = "Syrian Kurds")%>%
                        mutate(region = "Middle East and North Africa"))

# Lebanese Kurds and Bedouin (can't identify)

ar_all <- bind_rows(ar_all, d %>% filter(pobp==224) %>%
                        filter(anc1p==442)%>%
                        mutate(group = "Lebanese Kurds")%>%
                        mutate(region = "Middle East and North Africa") )

# Palestinians

ar_all <- bind_rows(ar_all,d %>%
                        filter(anc1p == 465) %>%
                        filter(pobp %in% c(216, 224, 239, 214, 222, 235, 245, 414, 248, 254)) %
                        mutate(group = "Palestinians")%>%
                        mutate(region = "Middle East and North Africa"))

# Bidoon
# Iraqi and Saudi descent in Kuwait

ar_all <- bind_rows(ar_all,d %>% filter(pobp==222) %>%
```

```
                        filter(anc1p==427|anc1p==417) %>%
                        mutate(group ="Iraqi and Saudi descent in Kuwait" )%>%
                        mutate(region = "Middle East and North Africa"))

## Asia and South Pacific

# Nepal born after 1990

ar_all <- bind_rows(ar_all, d %>% filter(pobp==229) %>%
                        filter(yob>1990) %>%
                        mutate(group = "Nepal born after 1990")%>%
                        mutate(region = "Asia and South Pacific"))

# Nepalis born in Bhutan

ar_all <- bind_rows(ar_all, d %>%
                        filter(anc1p==609) %>%
                        filter(pobp==203) %>%
                        mutate(group = "Nepalis born in Bhutan")%>%
                        mutate(region = "Asia and South Pacific"))

# Nothing on Rohingya

# Other Minorities from Myanmar

ar_all <- bind_rows(ar_all, d %>% filter(pobp==205) %>%
                        filter(anc1p %in% c(918, 706, 609)) %>%
                        mutate(group = "Other Minorities from Myanmar")%>%
                        mutate(region = "Asia and South Pacific"))

# Hmong from Laos

ar_all <- bind_rows(ar_all,d %>%
                        filter(anc1p==768) %>%
                        filter(pobp==223) %>%
                        mutate(group = "Hmong from Laos")%>%
                        mutate(region = "Asia and South Pacific"))

# Hmong from Thailand
```

```
ar_all <- bind_rows(ar_all,d %>%
                      filter(anc1p==768) %>%
                      filter(pobp==242) %>%
                      mutate(group = "Hmong from Thailand")%>%
                      mutate(region = "Asia and South Pacific"))

# Thai-Born Children of Burmese Refugees

ar_all <- bind_rows(ar_all,d %>%
                      filter(anc1p==700)%>%
                      filter(pobp==242)%>%
                      filter(yoep>1982) %>%
                      mutate(group = "Thai-Born Children of Burmese Refugees")%>%
                      mutate(region = "Asia and South Pacific"))

# Tibetans

ar_all <- bind_rows(ar_all,d %>%
                      filter(anc1p==714) %>%
                      filter(pobp %in% c(207, 210, 229)) %>%
                      mutate(group = "Tibetans")%>%
                      mutate(region = "Asia and South Pacific"))

# Stateless Groups from India

# Indian-born persons with Sri Lankan Tamil Origin

ar_all <- bind_rows(ar_all,d %>%
                      filter(pobp==210) %>%
                      filter(anc1p == 690) %>%
                      mutate(group = "Indian-born persons with Sri Lankan Tamil Origin")%>%
                      mutate(region = "Asia and South Pacific"))

# Indian-Born Persons of Bhutanese Origin, including Lhotshampas

ar_all <- bind_rows(ar_all, d %>%
                      filter(pobp==210) %>%
                      filter(anc1p == 607) %>%
```

```
                        mutate(group = "Indian-Born Persons of Bhutanese Origin")%>%
                        mutate(region = "Asia and South Pacific"))

# Bengalis from Pakistan

ar_all <- bind_rows(ar_all,d %>%
                        filter(pobp==231) %>%
                        filter(anc1p == 618) %>%
                        mutate(group = "Bengalis from Pakistan")%>%
                        mutate(region = "Asia and South Pacific"))

# Malaysian ethnic minorities
ar_all <- bind_rows(ar_all,d %>%
                        filter(pobp==226) %>%
                        filter(anc1p %in% c(918, 730, 720)) %>%
                        mutate(group = "Malaysian ethnic minorities")%>%
                        mutate(region = "Asia and South Pacific"))

# Ethnic Vietnamese and Khmer Krom from Cambodia

ar_all <- bind_rows(ar_all,d %>%
                        filter(pobp==206) %>%
                        filter(anc1p == 785) %>%
                        mutate(group = "Ethnic Vietnamese and Khmer Krom from Cambodia")%>%
                        mutate(region = "Asia and South Pacific"))

# SSA

# KENYA

# Somalian ancestry, including Galjeel

ar_all <- bind_rows(ar_all,d %>%
                        filter(pobp==427) %>%
                        filter(anc1p==568) %>%
                        mutate(group = "Kenya, Somalian ancestry, including Galjeel")%>%
                        mutate(region = "Sub-Saharan Africa"))

# Shona speakers
```

```r
if(year>=2016){
  ar_all <- bind_rows(ar_all,d %>%
                        filter(pobp==427) %>%
                        filter(hhlanp==5525) %>%
                        mutate(group = "Shona")%>%
                        mutate(region = "Sub-Saharan Africa"))
}


# Ethiopians with Eritrean Ancestry

ar_all <- bind_rows(ar_all,d %>%
                      filter(pobp==416) %>%
                      filter(anc1p==523)%>%
                      filter(yoep>=1998) %>%
                      mutate(group = "Ethiopians with Eritrean Ancestry")%>%
                      mutate(region = "Sub-Saharan Africa"))

# Eritreans with Ethiopian Ancestry

ar_all <- bind_rows(ar_all,d %>%
                      filter(pobp==417) %>%
                      filter(anc1p==522)%>%
                      filter(yoep>=1998) %>%
                      mutate(group = "Eritreans with Ethiopian Ancestry")%>%
                      mutate(region = "Sub-Saharan Africa"))

# Banyarwanda and Banyamulenge from the Democratic Republic of Congo

# Americas

# Dominicans (Dominican Republic) of Haitian Ancestry

ar_all <- bind_rows(ar_all,d %>%
                      filter(pobp==329) %>%
                      filter(anc1p==336) %>%
                      mutate(group = "Dominicans (Dominican Republic) of Haitian Ancestry")%
                      mutate(region = "Americas"))
```

```
  # Bahamians of Haitian Ancestry

  ar_all <- bind_rows(ar_all,d %>%
                        filter(pobp==323) %>%
                        filter(anc1p==336) %>%
                        mutate(group = "Bahamians of Haitian Ancestry")%>%
                        mutate(region = "Americas"))

  write_rds(ar_all, paste0("data/acs/acs1_atrisk_", year, ".RDS"))



}
```

4.6 Step 3: Model time series of at-risk populations

This step relates to code in the `4_model` script.

First, we read in the data from the first step, and join all the years together in the one dataframe. In addition, we calculate totals by at-risk profile.

```
years <- 2005:2019


df <- tibble()

for(i in 1:length(years)){
  d <- read_rds(paste0("data/acs/acs1_atrisk_", years[i], ".RDS"))
  d$year <- years[i]
  df <- bind_rows(df, d)
}

colnames(df)[str_detect(colnames(df), "pwgt")] <- str_to_upper(colnames(df)[str_detect(colnames
df <- df %>% mutate(PWGTP = as.numeric(PWGTP))


df$age_group <- cut(as.numeric(df$agep),
                    breaks= c(seq(0, 85, by = 5), Inf),
                    labels = seq(0, 85, by = 5),
                    right = FALSE)
```

```r
df$age_group <- as.numeric(as.character(df$age_group))

df <- df %>%
  filter(group!="Nepal born after 1990"|(group=="Nepal born after 1990"&(nativity_parents=="NA

df_survey <- to_survey(df)

group_counts <- df_survey %>%
  survey_count(group, year) %>%
  filter(group!="Roma"&group!="Shona"&group!="Balkan Egyptians")
```

The next step is to get these data into the right format to input into a Stan model:

```r
y <- group_counts %>%
  select(-n_se) %>%
  pivot_wider(names_from = "group", values_from = "n") %>%
  select(-year) %>%
  replace(is.na(.), 0.1) %>%
  as.matrix()

se_y <- group_counts %>%
  select(-n) %>%
  pivot_wider(names_from = "group", values_from = "n_se") %>%
  select(-year) %>%
  replace(is.na(.), 1) %>%
  as.matrix()

groups <- colnames(y)

stan_data <- list(y = log(y),
                  se_y = ifelse(y>0, se_y/y,1),
                  P = 5,
                 N = nrow(y), S = ncol(y))
```

Now we can run the time series model using the `1_rw2.stan` Stan file, which is contained in the `code/models` folder in the repo. Once the model is run, the next lines of code pull out the estimates and projections for each group.

```
mod <- stan(data = stan_data, file = "code/acs/models/1_rw2.stan")

yhat <- mod %>%
  gather_draws(lambda[i,s]) %>%
  mutate(fit =exp(.value)) %>%
  median_qi() %>%
  mutate(year = years[i], group = groups[s])

proj <- mod %>%
  gather_draws(y_p[i,s]) %>%
  mutate(proj = exp(.value)) %>%
  median_qi(.width = 0.5) %>%
  mutate(year = 2019+i, group = groups[s])
```

The `4_model` script also contains code to plot a set of example groups as in the case studies document, and also run the time series model on the total number of at-risk person (i.e., not by profile group). This code is contained in lines 77-169 of the script.

4.7 Step 4: Estimate stateless populations using probability-adjusted method: Nepalese born after 1990

The code from line 173 onward illustrates an example of how to estimate the size of stateless populations from at-risk populations using a probability-adjusted technique. This illustration is for Nepalese born after 1990 only.

The idea is that the probability of those at-risk of statelessness actually being stateless could be determined by additional variables in the ACS data and/or expert opinion. In the code below, we are assuming that the probability of statelessness increases as the number of years spent in the US increases. This is calculated below.

```
df %>%
  filter(group=="Nepal born after 1990") %>%
  group_by(yoep) %>%
  tally() %>%
  ggplot(aes(yoep, n)) + geom_bar(stat = "identity")

df_nepal <- df %>%
  filter(group=="Nepal born after 1990") %>%
  mutate(year_diff = year - as.numeric(yoep)) %>%
  mutate(year_diff = ifelse(year_diff<0, 0, year_diff))
```

```
df_nepal <- df_nepal %>%
  rowwise() %>%
  mutate(prob = min(0.5+0.01*year_diff, 0.95))
```

The next step is to calculate the standard errors around data observations, get the data into Stan format, and run the model:

```
df_nepal_survey <- to_survey(df_nepal)

df_nepal_summary <- df_nepal_survey %>%
  select(year, year_diff, prob) %>%
  survey_count(year, year_diff, prob) %>%
  mutate(prob_stateless = 1- prob,
         n_stateless = n*prob_stateless,
         se_y = n_se) %>%
  mutate(var_binomial = n_stateless*prob*(1-prob)) %>%
  mutate(mu_comb = (var_binomial*n_stateless+se_y^2*n_stateless)/(var_binomial+se_y^2),
         se_comb = sqrt(1/(1/var_binomial+1/se_y^2))) %>%
  group_by(year) %>%
  summarize(n = sum(n_stateless), se_y = sqrt(sum(se_y^2)))

stan_data <- list(y = log(df_nepal_summary$n),
                  se_y = ifelse(df_nepal_summary$n>0, df_nepal_summary$se_y/df_nepal_summary$n
                  N = length(years),
                  P = 5)

mod <- stan(data = stan_data, file = "code/acs/models/2_rw2_prob.stan")
```

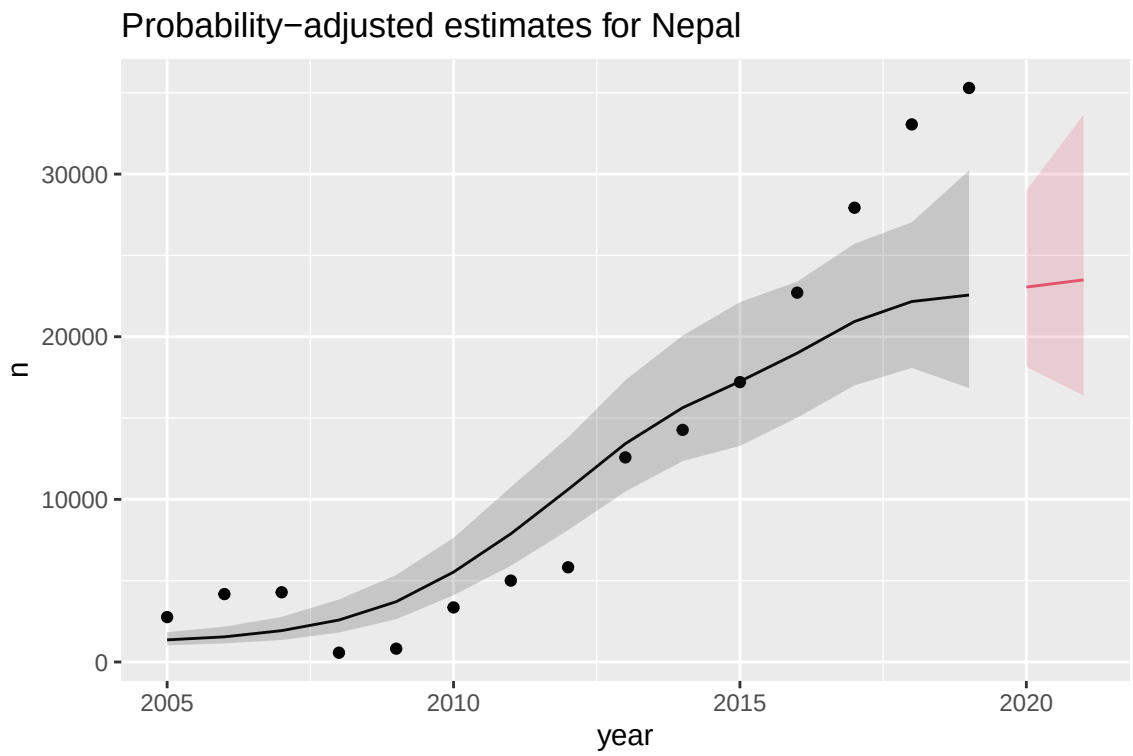
We can then extract estimates and projections and save the output:

```
yhat <- mod %>%
  gather_draws(eps[i]) %>%
  mutate(fit =exp(.value)) %>%
  median_qi() %>%
  mutate(year = years[i])

proj <- mod %>%
  gather_draws(y_p[i]) %>%
  mutate(proj = exp(.value)) %>%
```

```
  median_qi(.width = 0.5) %>%
  mutate(year = 2019+i)

yhat %>%
  ungroup() %>%
  select(year, fit, fit.lower, fit.upper) %>%
  mutate(type = "estimate") %>%
  bind_rows(proj %>% ungroup() %>%
              select(year, proj, proj.lower, proj.upper) %>%
              rename(fit = proj, fit.lower = proj.lower, fit.upper = proj.upper) %>%
              mutate(type = "projection")) %>%
  write_csv("output/acs_nepal.csv")
```

The model script also contains code to plot the original ACS data and probability adjusted estimates:

```
group_counts %>%
  filter(group=="Nepal born after 1990") %>%
  ggplot(aes(year, n)) + geom_point() +
  #facet_wrap(~group, scales = "free_y") +
  geom_line(data = yhat, aes(year, fit))+
  geom_line(data = proj%>%
              filter(year<2022), aes(year, proj), color = 2) +
  geom_ribbon(data = yhat, aes(year, y = fit, ymin = exp(.value.lower), ymax = exp(.value.upper
  geom_ribbon(data = proj%>%
                filter(year<2022),
              aes(year, y = proj, ymin = exp(.value.lower), ymax = exp(.value.upper)), alpha =
  ggtitle("Probability-adjusted estimates for Nepal")
```


Probability−adjusted estimates for Nepal
30000
20000
10000
0
n
2005
2010
2015
2020
year