Updating Data to be used in SRA

Julia Levine

07 October, 2022

Contents

Load in old data

The following code reads in the old data. It then takes the most recent data for each country and replicates it for the years that we are updating – this let's us easily keep the time-invariant values, like region. I then set aside the old data, and update the data that will be appended.

```
# read in data from the previous years , called dat



  dat <- fread(base_data)

# carry forward all data from the most recent year

  carry_from <- max(dat$year)

  # this only applies if updating multiple years simultaneously
  # it carries forward the most recent data for each of the years to be updated
  dat_new <- rbindlist(rep(list(dat[year == carry_from]), times = length(update_years)))

  # relabel the years

  dat_new$year <- rep(update_years, each = nrow(dat_new))

# rename old data and leave it alone until you've updated all
# variables except for the lag/leads

  dat_old <- dat
  rm(dat)
```

Update

Define functions

carry_forward

The following is a function to carry forward recent values of a variable for countries where it's NA

```
  carry_forward <- function(variable,
                            carry_to = update_years,
                            data = dat,
                            carry_from = (update_years - 1)){
    na_var <- data[year == carry_to & is.na(get(variable))]$sftgcode

    cat(paste0("Missing values for ", length(na_var), " countries."))

    carry_var <- subset(data[year == carry_from & sftgcode %in% na_var],
                        select = c("sftgcode", variable))
    cat(paste0("\nFilling in values for ",
               nrow(carry_var[!is.na(get(variable))]), " countries."))

    for(i in carry_var[!is.na(get(variable))]$sftgcode){
      data[year == carry_to & sftgcode == i,
          (variable) := carry_var[sftgcode == i, get(variable)]]
      data[year == carry_to & sftgcode == i, (paste(variable, "carry_from", sep = "_")) := carry_from]
    }
  }
```

merge_dat

The following is a function that removes the variables that were carried forward from the base data, and then merges in the updated variables from the appropriate year.

```
merge_dat <- function(new_data, base_data = dat_new,
                      merge_new_by = c("sftgcode", "year"),
                      merge_base_by = c("sftgcode", "year")){
  new_data_vars <- colnames(new_data)[!colnames(new_data) %in% merge_new_by]
  dat1 <- subset(base_data,
                 select = colnames(base_data)[!(colnames(base_data) %in%
                                                  new_data_vars)])
  dat1 <- merge(dat1, new_data,
                by.x = merge_base_by,
                by.y = merge_new_by, all.x = TRUE)
  dat1
}
```

Country age

This chunk updates country age by adding the difference between the updated years and the year the data was carried from to the country age variable.

```
# update country age

  diff <- update_years - carry_from

  for(i in 1:length(update_years)){
    dat_new$countryage[dat_new$year == update_years[i]] <-
    dat_new$countryage[dat_new$year == update_years[i]] + diff[i]
  }

# update logged country age variable

  dat_new$countryage.ln <- log(dat_new$countryage + 1)
```

VDEM

The following code reads in the most recent V-DEM data, selecting the relevant variables and renaming them.

```
# read in data

  vdem <- fread("input/Country_Year_V-Dem_Fullothers_CSV_v12/Country_Year_V-Dem_Full+others_CSV_v12/V-D

# subset to just use these variables


  vdem_vars <- c("v2cldmovem_ord",
                 "v2cldmovew_ord",
                 "v2clkill_ord",
                 "v2clsocgrp_ord",
                 "v2clrgunev_ord",
                 "v2csreprss_ord",
                 "v2pepwrsoc_ord",
                 "v2elrstrct",
                 "v2psparban_ord",
                 "v2psoppaut_ord",
                 "v2jureform_ord",
                 "v2clrelig_ord",
                 "v2xcl_disc",
                 "v2pepwrses_ord",
                 "v2pepwrses",
                 "e_migdpgro",
                 "e_mipopula",
                 "e_cow_exports",
                 "e_cow_imports",
                 "e_migdppc",
                 "v2csgender_ord",
                 "v2pepwrgen_ord")


  keep <- c("COWcode", "country_name", "year", vdem_vars)

# function that renames and formats variables,
# takes dataset and vector of variables to keep as inputs

  source("helper scripts/format_vdem.R")

# function that maps COW codes onto PITF country codes,
# contained in format_vdem

  source("helper scripts/cowtopitf2018.R")

# format and save vdem

  vdem <- format_vdem(dat = vdem, keep = keep)

  save(vdem, file = "input/temp/vdem.Rdata")
```

```
# read in formatted VDEM variables for the update years

  load("input/temp/vdem.Rdata")

# limit to the update years

  vdem <- vdem[year %in% update_years]

# count missing values
# check which variables are missing values

  missing <- apply(vdem[country_name %in% dat_new$country_name], 2,
                   function(x) sum(is.na(x)))
  kable(missing, col.names = "NA count")
```

	NA count
COWcode	0

	NA count
country_name	0
year	0
v2cldmovem_ord	0
v2cldmovew_ord	0
v2clkill_ord	0
v2clsocgrp_ord	0
v2clrgunev_ord	0
v2csreprss_ord	0
v2pepwrsoc_ord	0
candidaterestriction	1
partyban	0
barrierstoparties	5
judicialreform	0
religiousfreedom	0
freediscussion	0
v2pepwrses_ord	0
ses_power_dist	0
e_mipopula	159
e_cow_exports	159
e_cow_imports	159
v2csgender_ord	0
v2pepwrgen_ord	0
e_migdpgro	159
e_migdppc	159
sftgcode	0
gdppcgrowth	159
popsize	159
popsize.ln	159
tradeshare	159
pol_killing_approved	0
freemove_men4	0
freemove_women4	0
social_inequality	0
even_civilrights	0
repress_civilsoc	0
social_power_dist	0
minorityrule	0

```
# check which variables are missing for all countries, indicating that they
# have not been updated yet, and are potentially discontinued

  all.missing <- names(missing)[missing == nrow(vdem)]
  if(length(all.missing) > 0){
      cat(paste0("The following variables have not been updated as of ",
            Sys.Date(),":\n", paste(all.missing, collapse = "\n")))
  }


# this ensures that we dont replace old country names
# VDEM sometimes changes them

  vdem$country_name <- NULL
```

```
# update V-DEM data

  dat_new <- merge_dat(new_data = vdem)
```

Polity

According to the Center for Systemic Peace website, 2019 is the last year with updated polity variables. We've decided to drop the polity variables, but I'm just updating the durable variable here the same way I updated country age, by adding one year.

```
# update durable the same way we update country age
# Note: pol.durable and durable are the same thing

  diff <- update_years - carry_from

  for(i in 1:length(update_years)){
    dat_new$durable[dat_new$year == update_years[i]] <-
    dat_new$durable[dat_new$year == update_years[i]] + diff[i]

  }

# update the logged variables

  dat_new$durable.ln <- log(dat_new$durable + 1)
```

WDI

This chunk pulls select variables from the WDI API.

```
# In June 2020 WDI decommissioned V1 of the API
# so you may need to re-install the WDI package.
# install.packages("devtools")
  # devtools::install_github("vincentarelbundock/WDI")
  library(WDI)

# list indicators we want to pull

  wdilist <- c("NE.TRD.GNFS.ZS",     # Trade (% of GDP)
               "NY.GDP.PCAP.KD",     # GDP per capita (constant 2000 US$)
               "NY.GDP.MKTP.KD.ZG",  # GDP growth (annual %)
               "SP.POP.TOTL",        # Population, total
               "SP.DYN.IMRT.IN"      # Infant mortality rate

    )

  # confirm that all of the old indicator names are in use

    (check.names <- rbindlist(lapply(wdilist, function(x) {
      # search the indicator list for each of the variables in wdilist
       res <- WDIsearch(x, field = "indicator")
       # take the first match (drops those with extra letters on the end)
       if(length(res) > 2){
         res <- res[1, ]
       }
```

```r
       data.table("indicator" = res[1], "name" = res[2])})))

# Extract latest version of desired variables from WDI
# also pull 5 years from before so that we can carry forward
# more recent values for variables like infant mortality that are slow to update

  wdi <- WDI(country="all", indicator=wdilist, extra=FALSE,
             start=(min(update_years) - 5))

# Add PITF country codes for merging
  source("helper scripts/f.pitfcodeit.R")
  wdi <- pitfcodeit(wdi, "country")
  wdi$country <- as.character(wdi$country)

# Subset to drop cases with missing PITF codes, cut extra id vars

  wdi <- subset(wdi, !is.na(sftgcode), select=-c(1, 2))

# Reorder for easier review

  wdi <- wdi[order(wdi$sftgcode, wdi$year),]

# Rename variables-- add a "new" to indicate these are newly brought in from wdi
# to avoid reusing names already in the old EWP data

  setDT(wdi)
  wdi_cols <- c("wdi.trade.new",
             "wdi.gdppc.new",
             "wdi.gdppcgrow.new",
             "wdi.popsize.new",
             "wdi.imrate.new")
  setnames(wdi,
           c("NE.TRD.GNFS.ZS",
             "NY.GDP.PCAP.KD",
             "NY.GDP.MKTP.KD.ZG",
             "SP.POP.TOTL",
             "SP.DYN.IMRT.IN"),
           wdi_cols)


# save to the input folder
  if(first.pass == T){dir.create("input/temp/wdi")}
  fwrite(wdi, paste0("input/temp/wdi/pulled-", Sys.Date(), ".csv"))
```

This chunk reads in the latest pull from the WDI API, counts the missing values, and merges it into the main dataset.

```r
# list all files in the wdi directory and read in the latest pull

  wdi.files <- list.files("input/temp/wdi/")
  latest.pull <- which.max(lapply(wdi.files,
                                  function(x) {
                                    as.Date(gsub("pulled-|.csv", "", x))}))
  wdi <- fread(paste0("input/temp/wdi/", wdi.files[latest.pull]))
  new.wdi <- wdi[year %in% update_years]

# check which variables are missing values

  wdi_cols <- c("wdi.trade.new",
              "wdi.gdppc.new",
              "wdi.gdppcgrow.new",
              "wdi.popsize.new",
              "wdi.imrate.new")
  missing = melt(new.wdi[, lapply(.SD, function(x) sum(is.na(x))), .SDcols = wdi_cols])

  kable(missing, col.names = c("Variable", "NA count"))
```

Variable	NA count
wdi.trade.new	50
wdi.gdppc.new	12
wdi.gdppcgrow.new	12
wdi.popsize.new	1
wdi.imrate.new	169

```
# check which variables are missing for all countries, indicating that they
# have not been updated yet, but potentially will be in the future

  all.missing <- missing[value == length(unique(wdi$sftgcode))]$variable
  cat(paste("\n", all.missing, "not updated as of", Sys.Date()))
```

```
##
##  wdi.imrate.new not updated as of 2022-10-07
```

```
  # Carry forward the last non-missing value of infant mortality rate
  # do the same thing for missing values of tradeshare and gdp as they were
  # mostly updated in 2018 and the earliest values from CIA factbook are  2017

  carry_forward(variable = "wdi.imrate.new", carry_from = update_years - 1, data = wdi)
```

```
## Missing values for 169 countries.
## Filling in values for 168 countries.
```

```
  carry_forward(variable = "wdi.imrate.new", carry_from = update_years - 2, data = wdi)
```

```
## Missing values for 1 countries.
## Filling in values for 0 countries.
```

```
  carry_forward(variable = "wdi.imrate.new", carry_from = update_years - 3, data = wdi)
```

```
## Missing values for 1 countries.
## Filling in values for 0 countries.
```

```
  carry_forward(variable = "wdi.trade.new", carry_from = update_years - 1, data = wdi)
```

```
## Missing values for 50 countries.
## Filling in values for 30 countries.
```

```
  carry_forward(variable = "wdi.trade.new", carry_from = update_years - 2, data = wdi)
```

```
## Missing values for 20 countries.
## Filling in values for 5 countries.
```

```
  carry_forward(variable = "wdi.trade.new", carry_from = update_years - 3, data = wdi)
```

```
## Missing values for 15 countries.
## Filling in values for 2 countries.
```

```
  carry_forward(variable = "wdi.gdppc.new", carry_from = update_years - 1, data = wdi)
```

```
## Missing values for 12 countries.
## Filling in values for 7 countries.
```

```
  carry_forward(variable = "wdi.gdppc.new", carry_from = update_years - 2, data = wdi)
```

```
## Missing values for 5 countries.
## Filling in values for 1 countries.
```

```
  carry_forward(variable = "wdi.gdppc.new", carry_from = update_years - 3, data = wdi)
```

```
## Missing values for 4 countries.
## Filling in values for 1 countries.
```

```
  missing = melt(wdi[, lapply(.SD, function(x) sum(is.na(x))), .SDcols = wdi_cols])
  kable(missing, col.names = c("Variable", "NA count"))
```

Variable	NA count
wdi.trade.new	87
wdi.gdppc.new	21
wdi.gdppcgrow.new	30
wdi.popsize.new	6
wdi.imrate.new	6

```
# Merge it in to main data

  dat_new <- merge_dat(new_data = wdi[year %in% update_years])
  dat_new[, imr.sqrt := sqrt(wdi.imrate.new)]
  dat_new[, wdi.trade.ln.new := log(wdi.trade.new)]
```

UCDP battle-related deaths

The following code reads in the UCDP data and changes country names to match. Below, I sum the number of battledeaths for each country in the update years, and merge this onto dat_new.

```
# read in UCDP data

  ucdp <- fread("input/ucdp-brd-dyadic-221.csv")

# making sure we have the same variables as in 18.1 version.
# colnames for 18.1 are uppercase, use tolower when checking

  colnames(ucdp) <- gsub("_", "", colnames(ucdp))

# limit to the update years and conflict type >= 3

  ucdp <- ucdp[year %in% update_years & typeofconflict >= 3]
```

```
# change country names to match

  diff_loc <- setdiff(unique(ucdp$locationinc), unique(dat_new$country_name))

  # drops the locations that involve multiple countries

    diff_loc <- diff_loc[grepl(", ", diff_loc) == F]

  # change names

    ucdp$locationinc[ucdp$locationinc=="Russia (Soviet Union)"] = "Russia"
    ucdp$locationinc[ucdp$locationinc=="Myanmar (Burma)"] = "Burma/Myanmar"
    ucdp$locationinc[ucdp$locationinc=="Yemen (North Yemen)"] = "Yemen"
    ucdp$locationinc[ucdp$locationinc=="DR Congo (Zaire)"] = "Democratic Republic of Congo"

    if(length(setdiff(unique(ucdp$locationinc),
                      unique(dat_new$country_name)))>0){
      stop("Different country names")}

# group by year/country and sum battledeaths

  bd <- ucdp[, .("battledeaths" = sum(bdbest)), by = c("locationinc", "year")]

# save battledeath data

  fwrite(bd, "input/temp/battledeaths.csv")
```

I then read in the battledeaths for each country and merge it into the new data, replacing missing values with zeroes and updating the logged version of this variable.

```
# read in battledeaths data

  bd <- fread("input/temp/battledeaths.csv")

# merge battledeaths into main data

  dat_new <- merge_dat(new_dat = bd,
                       merge_new_by = c("locationinc", "year"),
                       merge_base_by = c("country_name", "year"))


# fill in zeroes and update logged variable

  dat_new[, battledeaths := ifelse(is.na(battledeaths), 0, battledeaths)]
  dat_new[, battledeaths.ln := log(battledeaths + 1)]
```

Coup Attempts

This code reads in the most updated version of the Powell and Thyne data which is posted on their website. It selects coups in the years to be updated

```
# pull coup data from powell and thyne websit

  coup_dat <- as.data.table(read.delim("http://www.uky.edu/~clthyn2/coup_data/powell_thyne_coups_final.
# keep coups in the update years

  new_coup <- coup_dat[year %in% update_years]

# where coup == 2 it was a successful coup
# where coup == 1 it was a failed coup

  new_coup[, cou.s.d := ifelse(coup == 2, 1, 0)]
  new_coup[, cou.f.d := ifelse(coup == 1, 1, 0)]

# save coup data

  fwrite(new_coup, paste0("input/temp/", "coups-pulled-", Sys.Date(), ".csv"))
```

This updates the coup variables in dat_new.

```
# read in the coup data

  temp.files <- list.files("input/temp")
  coup.files <- grep("coups", temp.files, value = T)
  latest.pull <- which.max(lapply(coup.files,
                                function(x) {
                                  as.Date(gsub("coups-pulled-|.csv", "", x))}))
  new_coup <- fread(paste0("input/temp/", coup.files[latest.pull]))


# select variables and print

  new_coup <- new_coup[, .(cou.s.d = as.integer(sum(cou.s.d)>0),
              cou.f.d = as.integer(sum(cou.f.d)>0)), by = .(country, year)]

  kable(new_coup)
```

country	year	cou.s.d	cou.f.d
Mali	2021	1	0
Niger	2021	0	1
Guinea	2021	1	0
Chad	2021	1	0
Sudan	2021	1	1
Myanmar	2021	1	0

```
######################################################################
# CHECK NAMES BEFORE MERGE
######################################################################
check_names = rbindlist(lapply(seq_len(nrow(new_coup)), function(i){
  dat_new[grepl(new_coup[i]$country, country_name),
          .(coup_country = new_coup[i]$country,
            countries_in_sra = country_name)]
}))
check_names[, exact_match := coup_country == countries_in_sra]
check_names[, has_exact_match := sum(exact_match)>0, by = .(coup_country)]
check_names = check_names[!(has_exact_match == T & exact_match == F)]
check_names[has_exact_match == F]
```

```
##    coup_country countries_in_sra exact_match has_exact_match
## 1:      Myanmar    Burma/Myanmar       FALSE           FALSE
```

```r
new_coup[country == "Myanmar", country := "Burma/Myanmar"]
######################################################################

# merge coup data into dat_new

  dat_new <- merge_dat(new_data = new_coup,
                       merge_new_by = c("country", "year"),
                       merge_base_by = c("country_name", "year"))

# fill in missing values with zeroes

  coup_cols <- c("cou.s.d", "cou.f.d")
  dat_new[, (coup_cols) := lapply(.SD, function(x)
    ifelse(is.na(x), 0, x)), .SD = coup_cols]

# update cou.any variable (1 if there was either a failed or successful coup)

  dat_new[, cou.any := ifelse(cou.s.d>0 | cou.f.d>0, 1, 0)]
```

Mass Killing Variables

- state-led onset, Burma/Myanmar (began in 2021)
- state-led onset, Ethiopia (began in 2020)
- non-state-led mk ended, Afghanistan

```r
# initialize state led mk vars

  dat_new$mkl.end <- 0
  dat_new$mkl.start <- 0

# initialize non-state led mk vars

  dat_new$nonstatemk.end <- 0
  dat_new$nonstatemk.start <- 0

  dat_new[country_name %in%
            c("Burma/Myanmar"),
          mkl.start := 1]
  dat_new[country_name %in%
            c("Afghanistan"),
          nonstatemk.end := 1]
  dat_new[country_name %in%
            c("Afghanistan"),
          nonstatemk.ongoing := 0]
```

Append new data

This appends dat_new to dat_old, creating the full data-set

```r
# check to make sure you can append the new data

  if(length(setdiff(colnames(dat_old), colnames(dat_new)))>0){
    stop("different colnames")}


# append new data

  setcolorder(dat_new, colnames(dat_old))
  dat <- rbind(dat_old, dat_new, fill = T)
```

Mass Killing Onsets

I now make changes to the mass killing variables in years prior to the update years.

There are no changes to be made.

```
# change old data

  # dat[country_name == "Ethiopia", .(year, mkl.start, mkl.ongoing)]
  dat[country_name == "Ethiopia" & year == 2020, mkl.start := 1]

# combined onset variables

  dat[, anymk.start := ifelse((!is.na(mkl.start) & mkl.start == 1)|
                               nonstatemk.start == 1, 1, 0)]

  dat[, anymk.ongoing := ifelse((!is.na(mkl.ongoing) & mkl.ongoing == 1)|
                               nonstatemk.ongoing == 1, 1, 0)]

  dat[, anymk.ever := ifelse((!is.na(mkl.ever) & mkl.ever == 1)|
                               nonstatemk.ever == 1, 1, 0)]
```

```
# save mass killing variables

  mkl_dat <- subset(dat, select = c("country_name", "sftgcode", "year",
                                   "mkl.start", "mkl.end", "mkl.ongoing",
                                   "mkl.ever", "nonstatemk.start",
                                   "nonstatemk.end", "nonstatemk.ongoing",
                                   "nonstatemk.ever"))
  setkey(mkl_dat, country_name, year)

  fwrite(mkl_dat, file = "output/mkl_data.csv", row.names = F)
```

Leads and Lags

In this section, I update the variables that involve leads and lags

MK lead variables

```
# read in list of the first time Sftgcodes are used in PITF data

  map <- fread("../../../EWP (1)/2019SRA/Make data/sftg_name_map.csv")
  setnames(map, "V1", "min_ewp_year")

  # aggregate over different country names and drop duplicates
```

```
    map <- map[country != ""]
    map[, min_ewp_year := min(min_ewp_year), by = "sftgcode"]
    map[, ':=' (country = NULL, country_name = NULL)]
    map <- map[!duplicated(map)]


# merge in first year a sftgcode was used in the PITF data

  dat[, min_ewp_year := NULL]
  dat <- merge(dat, map, by = "sftgcode", all.x = T)

# order by year, country name, and sftgcode

  setkey(dat, year, country_name, sftgcode)

# create anymk lead variable, shifting by sftgcode

  dat[, anymk.start.1 := shift(anymk.start, 1, type = "lead"), by = "sftgcode"]

  # For new states we don't attribute mass killings to their origin country.
  # if the year is the year before the PITF starts first used an sftgcode,
  # then reset to 0

    dat[, anymk.start.1 := ifelse(year == min_ewp_year - 1, 0, anymk.start.1)]


#  create the rest of the lead variables based off of the one-year lead

  dat[, anymk.start.2 := shift(anymk.start.1, type = "lead"), by = "sftgcode"]
  dat[, anymk.start.3 := shift(anymk.start.1, n= 2, type = "lead"), by = "sftgcode"]

  dat[, anymk.start.2window := as.double((anymk.start.1 + anymk.start.2) > 0) ]
  dat[, anymk.start.3window := as.double((anymk.start.2window + anymk.start.3) > 0)]
```

Coup in the last 5 years

Create dataframe of sftgcodes that are created as a result of a coup. We then turn the coup.try.5yr indicator to 1 for the first four years of that sftgcode's existence.

```
# this is a table of sftgcodes created by a coup

  new_sftg <- data.table("sftgcode" = c("GAB", "ETI", "DJI"),
                         "coup.create.yr" = c(1964, 1989, 2000))

# merge this into main

  dat[, coup.create.yr := NULL]
  dat <- merge(dat, new_sftg, by = "sftgcode", all = T)

# create a variable for the last year when an sftgcode had a coup

  dat[, last_coup_yr := ifelse(cou.any == 1, year, NA)]
  dat[, last_coup_yr := na.locf(last_coup_yr, fromLast = F, na.rm = F),
      by = "sftgcode"]
  dat[, last_coup_yr := ifelse(is.na(last_coup_yr), -Inf, last_coup_yr)]

# for sftgcodes that were initiated due to a coup,
# replace -Inf with the year of the coup

  dat[, last_coup_yr := ifelse(!is.na(coup.create.yr) &
                                coup.create.yr > last_coup_yr,
                              coup.create.yr, last_coup_yr)]

# create variable for whether there was a coup in the last 5 years

  dat[, coup.try.5yr := as.double((year - last_coup_yr) <= 4)]
```

Data Preparation

Remake some variables

This section fills in some V-DEM variables by creating an adjusted version of the relevant WDI variable. This significantly cuts down on missingness.

```
make.combined <- function(vdem.var, wdi.proxy){
  # Make an adjusted version of the wdi one to fit the tradeshare (VDEM) one:
    # drop update year as we've carried forward some values to this year

    lm.adjust <- lm(as.formula(paste0(vdem.var, "~", wdi.proxy)),
                   data = dat[year < update_years])

  # fill in update years with fitted values from regression on all years

    dat[year %in% update_years, lm.adjust := coef(lm.adjust)[1] +
          coef(lm.adjust)[2]*get(wdi.proxy)]

  # Where vdem var is missing, replace with the adjusted wdi proxy
    combined.var <- paste0(vdem.var, ".combined")
    dat[year %in% update_years,
        (combined.var) :=
          ifelse(!is.na(get(vdem.var)), get(vdem.var), lm.adjust)]

  # summarize differnce in missingness
    na.old <- sum(is.na(subset(dat[year %in% update_years], select = vdem.var)))
    na.new <- sum(is.na(subset(dat[year %in% update_years], select = combined.var)))

    cat(paste0("In the update years we were missing ", na.old,
           " observations for ", vdem.var))
    cat(paste0("\nUsing the WD indicator, ",  wdi.proxy,
               ", as a proxy we are missing ",  na.new, " observations"))

    dat[, lm.adjust := NULL]
}
```

```
# create combined variable for tradeshare

  make.combined(vdem.var = "tradeshare.ln", wdi.proxy = "wdi.trade.ln.new")
```

```
## In the update years we were missing 163 observations for tradeshare.ln
## Using the WD indicator, wdi.trade.ln.new, as a proxy we are missing 15 observations
```

```
  cat("\n")
```

```
# create combined variable for popsize

  make.combined(vdem.var = "popsize", wdi.proxy = "wdi.popsize.new")
```

```
## In the update years we were missing 163 observations for popsize
## Using the WD indicator, wdi.popsize.new, as a proxy we are missing 5 observations
```

```
  cat("\n")
```

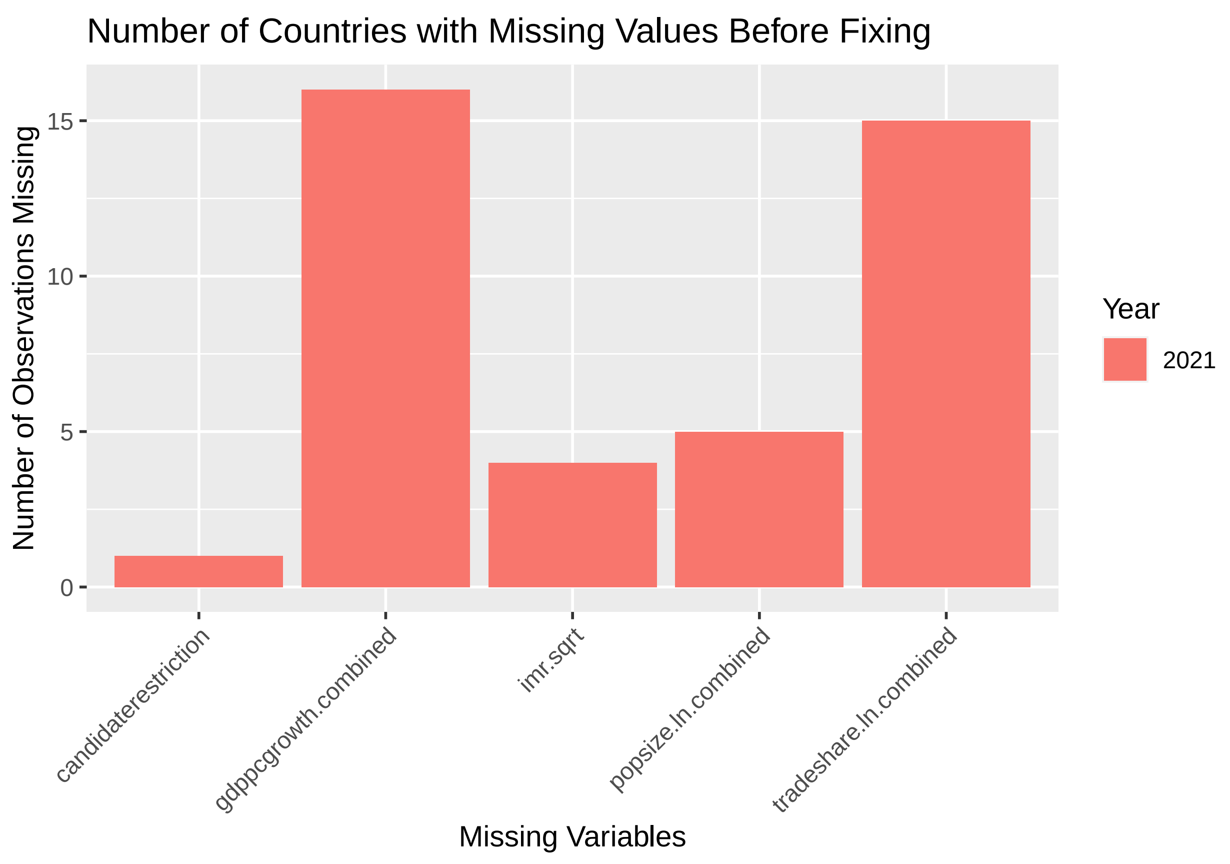
```
  # update logged version

    dat[year %in% update_years, popsize.ln.combined := log(popsize.combined)]

# create combined variable for GDP per capita growth

  make.combined(vdem.var = "gdppcgrowth", wdi.proxy = "wdi.gdppcgrow.new")
```

```
## In the update years we were missing 163 observations for gdppcgrowth
## Using the WD indicator, wdi.gdppcgrow.new, as a proxy we are missing 16 observations
```

Missingness

This section looks at remaining missingness and attempts to fill in values.

```
# select variables we would like to check missingness for

  predictornames <- c("anymk.ongoing","anymk.ever",
                      "reg.afr", "reg.eap", "reg.eur", "reg.mna", "reg.sca",
                      "countryage.ln", "popsize.ln.combined", "imr.sqrt",
                      "gdppcgrowth.combined", "ios.iccpr1","includesnonstate",
                      "durable.ln","minorityrule", "elf.ethnic",
                      "battledeaths.ln", "candidaterestriction", "partyban",
                      "judicialreform", "religiousfreedom",
                      "pol_killing_approved", "freemove_men4",
                      "freemove_women4", "freediscussion",
                      "social_inequality","even_civilrights","repress_civilsoc",
                      "social_power_dist", "ses_power_dist",
                      "tradeshare.ln.combined",
                      "coup.try.5yr", "polity2.fl.2","polity2.fl.3")

# look for missingness in the update years for select variables

  dat.check <- subset(dat, year %in% update_years,
                      select = c("country_name","year", predictornames))

# for each of the update years, count the number of NAs for each variable
# select variables with positive NA counts

  comp <- lapply(update_years,
                 function(y) apply(dat.check[year == y],
                                   2, function(x) sum(is.na(x))))
  na.count <- unlist(comp)
  years <- rep(update_years, each = ncol(dat.check))
  check <- data.table(na.count, years, "var"=names(na.count))
  check <- check[na.count > 0]

# visualize missingness

  (missing1 <- ggplot(check) +
      geom_bar(aes(y = na.count, x = var, fill = factor(years)),
               stat = "identity")+
      theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
      labs(x = "Missing Variables", y = "Number of Observations Missing",
           fill = "Year",
           title = "Number of Countries with Missing Values Before Fixing"))
```



Fix missingness

This chunk defines functions to look at patterns in NAs and carry forward values.

```
# function to look at patterns in NAs over time for each country with missing values

  look_na <- function(variable, data = dat){
    missing <- data[is.na(get(variable)) &
                     year %in% update_years, c("year", "country_name")]
    countries <- unique(missing$country_name)
    if(length(countries) == 0){
      look <- "No missing values"
    }else{
      look <-lapply(1:length(countries), function(x){
        out <- subset(data, country_name == countries[x],
                      select = c("year", variable))
        colnames(out) <- c("year", countries[x])
        out})

      # merge all data.tables in the list by year
        look <- Reduce(function(...) merge(..., all = T, by = "year"), look)
        colnames(look) <- c("year", as.character(countries))
        look <- look[order(look$year, decreasing = TRUE), ]
      }
    look
  }


# create blank spreadsheets to fill out

  fill.tab <- function(var){
    out <- subset(dat[is.na(get(var)) & year %in% update_years],
                  select = c("country_name", var))
    out[, cia.factbook.est := NA]
    out[, est.year := NA]
    fwrite(out, file = paste0("input/temp/missing/", var, ".csv"))
  }


# look at last present value for countries  missing data in update years

  last.present <- function(var){
    look.na.long <- melt(look.na.vars[[var]], id.vars = "year",
                         variable.name = "country_name")
    setorder(look.na.long, country_name, year)
    look.na.long[, value_last := shift(value), by = country_name]
    look.na.long[, same_as_last := value == value_last]
    look.na.long[!is.na(value) & is.na(value_last), same_as_last := F]
    look.na.long[!is.na(same_as_last) & same_as_last == F,
                 .("carried_value" = value[which.max(year)],
                   "carry_from" = max(year)), by = .(country_name)]

  }
```

```
# for each of the variables that we have countries with missing values
# look at the values in the last few years for that country

  look.na.vars <- lapply(c(check$var, "wdi.gdppc.new"), look_na)
  names(look.na.vars) <- c(check$var, "wdi.gdppc.new")

  if(first.pass == T){
    dir.create("input/temp/missing")
    lapply(check$var, fill.tab)
    fill.tab("wdi.gdppc.new")
  }
```

Candidate Restriction

```
# candidate restriction

  head(look.na.vars$candidaterestriction)
```

```
##    year Jamaica
## 1: 2021      NA
## 2: 2020       1
## 3: 2019       1
## 4: 2018       1
## 5: 2017       1
## 6: 2016       1
```

```
# just carry forward this value

  carry_forward("candidaterestriction")
```

```
## Missing values for 1 countries.
## Filling in values for 1 countries.
```

For the variables addressed below, I read in values from the CIA Factbook, and carry those values forward if they're more recent than the last updated version of the variable.

In the tables presented for each variable, I show the value that is filled in for that country in column 2. This is either the "carried_value", carried from the year listed under "carry_from", or it is the "cia.factbook.est".

Infant Mortality

We have been using CIA Factbook values for these countries since 2019:

- Macedonia
- North Korea

```
  head(look.na.vars$imr.sqrt)
```

```
##    year Macedonia North Korea Taiwan   Turkey
## 1: 2021        NA          NA     NA       NA
## 2: 2020  2.736786    4.734976     NA 2.932576
## 3: 2019  2.720294    4.472136     NA 3.016621
## 4: 2018  2.792848    4.690416     NA 3.016621
## 5: 2017  2.792848    4.690416     NA 3.162278
## 6: 2016  2.792848    4.690416     NA 3.286335
```

```
# read in CIA Factbook estimates

  cia.factbook <- fread("input/temp/missing/imr.sqrt.csv")
  cia.factbook[, cia.factbook.est := sqrt(cia.factbook.est)]

  # subtract one from the carry_from value becuase there is a lag
  # in updating the infant mortality rate

  comp <- merge(cia.factbook, last.present("imr.sqrt"),
               by = "country_name")
  comp[, imr.sqrt := ifelse(carry_from >= est.year,
                                       carried_value, cia.factbook.est)]
  kable(comp)
```

country_name	imr.sqrt	cia.factbook.est	est.year	carried_value	carry_from
Macedonia	2.705550	2.705550	2022	2.736786	2020
North Korea	4.712749	4.712749	2022	4.734976	2020
Turkey	4.398864	4.398864	2022	2.932576	2020

```
# replace missing

  for(i in 1:nrow(comp)){
    country.fill <- comp[i, ]
    dat[country_name ==  country.fill$country_name & year == update_years,
        imr.sqrt := country.fill$imr.sqrt]
  }
```

Population size

We have been using CIA factbook values for these countries since 2019:

- Eritrea
- Macedonia
- North Korea

```
  head(look.na.vars$popsize.ln.combined)
```

```
##    year  Eritrea Macedonia North Korea Taiwan   Turkey
## 1: 2021       NA        NA          NA     NA       NA
## 2: 2020 15.63154  14.57082    17.06710     NA 18.24325
## 3: 2019 15.62071  14.56974    17.05980     NA 18.23244
## 4: 2018 15.60237  14.64728    17.04908     NA 18.21909
## 5: 2017 15.59385  14.64728    17.04908     NA 18.19969
## 6: 2016 15.59385  14.64644    17.04908     NA 18.18436
```

```
# read in CIA Factbook estimates

  cia.factbook <- fread("input/temp/missing/popsize.ln.combined.csv")

  comp <- merge(cia.factbook, last.present("popsize.ln.combined"),
                by = "country_name")

  comp[, popsize.ln.combined := ifelse(carry_from >= est.year,
                                       carried_value, log(cia.factbook.est))]
  comp[, popsize.combined := ifelse(carry_from >= est.year,
                                    carried_value, cia.factbook.est)]
  kable(comp)
```

country_name	popsize.ln.combined	cia.factbook.est	est.year	carried_value	carry_from	popsize.combined
Eritrea	15.64155	6209262	2022	15.63154	2020	6209262
Macedonia	14.57207	2130936	2022	14.57082	2020	2130936
North Korea	17.07188	25955138	2022	17.06710	2020	25955138
Turkey	18.23493	83047706	2022	18.24325	2020	83047706

```
# replace missing
```

```
  for(i in 1:nrow(comp)){
    country.fill <- comp[i, ]
    dat[country_name ==  country.fill$country_name & year == update_years,
        popsize.ln.combined := country.fill$popsize.ln.combined]
  }

  for(i in 1:nrow(comp)){
    country.fill <- comp[i, ]
    dat[country_name ==  country.fill$country_name & year == update_years,
        popsize.combined := country.fill$popsize.combined]
  }
```

GDP Per capita growth

In 2021, this variable is missing for the following countries:

```
## Cuba, Eritrea, Japan, Kuwait, Macedonia, Oman, North Korea, South Sudan, Syria, Taiwan, Turkmenistan
```

The following countries are newly missing, and have more recent estimates from the WDI than the CIA factbook, so we carry old WDI values forward.

```
missing_22 = colnames(look.na.vars$gdppcgrowth.combined)[-1]
cat(paste(paste(setdiff(missing_22, missing_21), collapse = ", "), "\n"))
```

```
## Afghanistan, Bhutan, Iran, Turkey
```

```
# read in CIA Factbook estimates
# "Real GDP growth rate" IN FACTBOOK

  cia.factbook <- fread("input/temp/missing/gdppcgrowth.combined.csv")
  comp <- merge(cia.factbook, last.present("gdppcgrowth.combined"),
                by = "country_name")
  comp[, cia.factbook.est := cia.factbook.est/100]


  comp[, gdppcgrowth.combined := ifelse(carry_from >= est.year,
                                        carried_value, cia.factbook.est)]
  comp[, source := factor(carry_from >= est.year & as.character(round(carried_value, 4)) != as.characte

  kable(comp[order(source)], digits = 3)
```

country_name	gdppcgrowth.combined	cia.factbook.est	est.year	carried_value	carry_from	source
Afghanistan	-0.023	0.027	2017	-0.023	2020	carry_WDI
Bhutan	-0.062	0.074	2017	-0.062	2020	carry_WDI
Iran	0.007	0.037	2017	0.007	2020	carry_WDI
Kuwait	-0.003	-0.033	2017	-0.003	2019	carry_WDI
North Korea	-0.017	-0.011	2015	-0.017	2015	carry_WDI
Oman	-0.003	-0.009	2017	-0.003	2019	carry_WDI
Syria	-0.138	-0.365	2014	-0.138	2016	carry_WDI
Taiwan	0.029	0.027	2019	0.029	2019	carry_WDI
Turkey	0.008	0.010	2019	0.008	2020	carry_WDI
United Arab Emirates	0.007	0.008	2017	0.007	2019	carry_WDI
Venezuela	-0.140	-0.197	2018	-0.140	2019	carry_WDI
Cuba	0.016	0.016	2017	0.016	2019	CIA_Factbook

country_name	gdppcgrowth.combined	cia.factbook.est	est.year	carried_value	carry_from	source
Eritrea	0.050	0.050	2017	0.050	2019	CIA_Factbook
Macedonia	0.000	0.000	2017	0.000	2019	CIA_Factbook
South Sudan	-0.052	-0.052	2017	-0.052	2019	CIA_Factbook
Turkmenistan	0.065	0.065	2017	0.065	2019	CIA_Factbook

```
# replace missing

  for(i in 1:nrow(comp)){
    country.fill <- comp[i, ]
    dat[country_name ==  country.fill$country_name & year == update_years,
        gdppcgrowth.combined := country.fill$gdppcgrowth.combined]
  }
```

tradeshare.ln.combined

```
  cia.factbook <- fread("input/temp/missing/wdi.gdppc.new.csv")
  comp <- merge(cia.factbook, last.present("wdi.gdppc.new"),
                by = "country_name", all.x = T)

  comp[, wdi.gdppc.new := ifelse((est.year < (carry_from-1) & !is.na(carry_from))|
                                 is.na(cia.factbook.est), carried_value,
                               cia.factbook.est)]
  comp[, source := factor(carry_from-1 > est.year & as.character(round(carried_value, 4)) != as.characte
  kable(comp[order(source)], digits = 3)
```

First fill in GDP per capita

country_name	wdi.gdppc.new	cia.factbook.est	est.year	carried_value	carry_from	source
Eritrea	1600	1600	2017	1600.00	2019	CIA_Factbook
Macedonia	15800	15800	2020	16479.00	2020	CIA_Factbook
North Korea	1700	1700	2015	1700.00	2019	CIA_Factbook
South Sudan	1600	1600	2017	1600.00	2019	CIA_Factbook
Taiwan	24502	24502	2018	24502.00	2020	CIA_Factbook
Turkey	28400	28400	2020	15225.61	2020	CIA_Factbook
Venezuela	7704	7704	2018	7704.00	2020	CIA_Factbook

```
  # replace missing

  for(i in 1:nrow(comp)){
    country.fill <- comp[i, ]
    dat[country_name ==  country.fill$country_name & year == update_years,
        wdi.gdppc.new := country.fill$wdi.gdppc.new]
  }
```

Calculate Tradeshare In the 2020 update, we took CIA Factbook estimates if they were made no earlier than 2016 for the following countries:

```
## Eritrea, Fiji, Macedonia, North Korea, Papua New Guinea, Solomon Islands, Somalia, South Sudan, Syri
```

Many of these countries still have missing values during the 2021 update, so we continue to use CIA Factbook values for them. The following countries are newly missing tradeshare data in 2021:

```
## Afghanistan, Guyana, Laos, Liberia, Malawi, Taiwan, Turkey, Democratic Republic of Vietnam
```

```
# read in CIA Factbook estimates

  cia.factbook <- fread("input/temp/missing/tradeshare.ln.combined.csv")
  comp <- merge(cia.factbook, last.present("tradeshare.ln.combined"),
                by = "country_name")
  other.vars <- dat[year == update_years,
                    c("wdi.gdppc.new", "popsize.combined", "country_name")]
  comp <- merge(comp, other.vars, by= "country_name", all.x = T)


  # The VDEM imports and exports are in millions of USDs, then x10^6.
  # tradeshare = (imports + exports (in trillions))/(gdppc*popsize)
  # The CIA factbook data is in billions, so needs to be multiplied by 1e+9
  comp[, cia.factbook.est := log(1e+9*cia.factbook.est/(wdi.gdppc.new*popsize.combined))]



    comp[, tradeshare.ln.combined :=
           ifelse(est.year <= carry_from, carried_value, cia.factbook.est)]
    comp[, source := factor(carry_from >= est.year & as.character(round(carried_value, 4)) != as.charac
    comp[country_name %in% intersect(missing_21, missing_20), source := "CIA_Factbook"]
    kable(subset(comp[order(source)], select = c("country_name", "tradeshare.ln.combined",
                                "cia.factbook.est", "est.year",
                                "carried_value", "carry_from", "source")))
```

country_name	tradeshare.ln.combined	cia.factbook.est	est.year	carried_value	carry_from	source
Democratic Republic of Vietnam	-0.8888857	0.5012140	2019	-0.8888857	2020	carry_WDI
Guyana	-1.9638614	-0.6745026	2019	-1.9638614	2019	carry_WDI
Laos	-2.0210334	-0.2958945	2019	-2.0210334	2019	carry_WDI
Liberia	-1.4747539	1.4619059	2019	-1.4747539	2020	carry_WDI
Malawi	-2.1534896	-0.5771616	2019	-2.1534896	2019	carry_WDI
Turkey	-2.2524742	-1.6897689	2020	-2.2524742	2020	carry_WDI
Afghanistan	-0.9102506	-0.9102506	2020	-2.5713766	2019	CIA_Factbook
Eritrea	-1.7148466	-1.7356873	2017	-1.7148466	2019	CIA_Factbook
Macedonia	-0.7477396	-0.7477396	2020	-1.0369120	2019	CIA_Factbook
North Korea	-2.8419659	-2.8540467	2018	-2.8419659	2019	CIA_Factbook
Papua New Guinea	-0.7669202	-0.4670185	2018	-0.7669202	2019	CIA_Factbook
South Sudan	-1.2915939	-1.1087424	2019	-1.2915939	2019	CIA_Factbook
Trinidad and Tobago	-0.4289219	-0.3312453	2019	-0.4289219	2019	CIA_Factbook
Venezuela	-2.1122894	-0.7741132	2018	-2.1122894	2019	CIA_Factbook

```
# replace missing

  for(i in 1:nrow(comp)){
    country.fill <- comp[i, ]
    dat[country_name ==  country.fill$country_name & year == update_years,
        tradeshare.ln.combined := country.fill$tradeshare.ln.combined]
  }
  # carry_forward("tradeshare.ln.combined")
```

Check that all is filled in

```
# repeat to make sure there are no more missing

  look.na.vars <- lapply(check$var, function(x)
    look_na(x, dat = dat[country_name!="Taiwan"]))
  names(look.na.vars) <- check$var
  look.na.vars
```

```
## $popsize.ln.combined
## [1] "No missing values"
##
## $imr.sqrt
## [1] "No missing values"
##
## $gdppcgrowth.combined
## [1] "No missing values"
##
## $candidaterestriction
## [1] "No missing values"
##
## $tradeshare.ln.combined
## [1] "No missing values"
```

```
fwrite(dat, file = paste0("output/prepared", update_years, "predictors-", Sys.Date(), ".csv"))
```

Checking the append

```
# function to check all.equal for each var

  compare_var <- function(var){
    vars <- paste(var, c("x", "y"), sep = ".")

    check <- subset(compare, select = c("year", "sftgcode", vars[1], vars[2]) )
    colnames(check) <- c("year", "sftgcode", "var.x", "var.y")

    all.equal(check$var.x, check$var.y)
  }


# look into differences where all.equal == F for each var

  check_diffs <- function(var){
    vars <- paste(var, c("x", "y"), sep = ".")

    check <- subset(compare, select = c("year", "sftgcode", vars[1], vars[2]) )

    check[get(vars[1]) != get(vars[2])|
            is.na(get(vars[1])) & !is.na(get(vars[2]))|
            is.na(get(vars[2])) & !is.na(get(vars[1]))]
  }
```

```
# read in newly updated data and just focus on the last year in old

  dat.files <- list.files("output/")[grepl("prepared", list.files("output/"))]
  latest.pull <- which.max(lapply(dat.files,
                                  function(x) {
                                    as.Date(gsub(paste0(paste0("prepared", update_years, "predictors-")
  dat <- fread(paste0("output/", dat.files[latest.pull]))


  new_dat <- dat[year <= (update_years-1)]

# read in original base data

  dat <- fread(base_data)
  old_dat <- dat
  old_dat[, c("pol.durable.ln", "pol.durable") := NULL]


# check row count

  if(nrow(old_dat) != nrow(new_dat)){stop("Different observation count")}

# merge new and old data to compare

  compare <- merge(new_dat, old_dat, by = c("sftgcode", "year"))

# make sure they match on all variables in the old data

  check.vars <- colnames(old_dat)[!colnames(old_dat) %in% c("sftgcode", "year")]

  comp <- data.table("all.equal" = sapply(check.vars, compare_var),
                     "var" = check.vars)

# look into the differences where all.equal == F
# and where the issue isn't an NA value mismatch -- come back to this later

  differences <- comp[all.equal!="TRUE"][!grep("is.NA", all.equal)]
  diff.look <- lapply(differences$var, check_diffs)
  names(diff.look) <- differences$var

  cat(paste0("There are ", length(diff.look),
             " variables where the two datasets disagree on non-missing values."))
```

```
## There are 2 variables where the two datasets disagree on non-missing values.
```

```
  if(length(diff.look) > 0){
      for(i in 1:length(diff.look)){
    print(diff.look[[i]])
  }
  }
```

```
##    year sftgcode mkl.start.x mkl.start.y
## 1: 2020      ETI           1           0
##    year sftgcode anymk.start.x anymk.start.y
## 1: 2020      ETI              1             0
```

```
# now let's look at differences in missing values

  na.diffs <- comp[all.equal!="TRUE"][grep("is.NA", all.equal)]
  na.diff.look <- lapply(na.diffs$var, check_diffs)
  names(na.diff.look) <- na.diffs$var

  # for each variable, count the instances of differences in NAs by year
  count.by.yr <- rbindlist(lapply(seq_along(na.diff.look),
                                  function(i) na.diff.look[[i]][, .N, by = "year"][, var := na.diffs$va

  # We can see that the only differences are the filling of NAs
  # for the appropriate lead variables
  count.by.yr
```

```
##     year   N                var
##  1: 2020 163       anymk.start.1
##  2: 2019   1       anymk.start.1
##  3: 2019 163 anymk.start.2window
##  4: 2018   1 anymk.start.2window
##  5: 2018 163 anymk.start.3window
##  6: 2017   1 anymk.start.3window
##  7: 2019 163       anymk.start.2
##  8: 2018   1       anymk.start.2
##  9: 2018 163       anymk.start.3
## 10: 2017   1       anymk.start.3
```