TidyTuesday - UN Votes

Moriah Taylor

3/23/2021

This Week's Data

The data this week comes from Harvard's Dataverse by way of Mine Çetinkaya-Rundel, David Robinson, and Nicholas Goguen-Compagnoni. *Interestingly enough, Mine was my instructor for my Stats101 course in college - small world!*

Original Data Citation: Erik Voeten "Data and Analyses of Voting in the UN General Assembly" Routledge Handbook of International Organization, edited by Bob Reinalda (published May 27, 2013). Available at SSRN: http://ssrn.com/abstract=2111149.

```
library(tidytuesdayR)
library(tidyverse)
library(lubridate)
library(ggdark)
library(unvotes)
library(extrafont)
library(sysfonts)
library(showtext)
library(dplyr)
library(readxl)
library(countrycode)
library(tidyr)
library(forcats)
library(Cairo)
library(rmarkdown)
```

a

```
# Get the Data
# Code sourced from TidyTuesday GitHub
# (https://github.com/rfordatascience/tidytuesday/tree/master/data/2021/2021-03-23)

unvotes <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2
roll_calls <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data
issues <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/20
```

Data Cleaning

The code data cleaning can be found in the data-raw folder of the package.

```
vlevels <- c("yes", "abstain", "no")

load("UNVotes2021.RData")
```

```
un_votes <- completeVotes %>%
  filter(vote <= 3) %>%
  mutate(
    country = countrycode(ccode, "cown", "country.name"),
    country_code = countrycode(ccode, "cown", "iso2c"),
    # Match based on old version of data from unvotes package
    country_code = case_when(
      country == "Czechoslovakia" ~ "CS",
      country == "Yugoslavia" ~ "YU",
      country == "German Democratic Republic" ~ "DD",
      country == "Yemen People's Republic" ~ "YD",
      TRUE ~ country_code
    ),
    country = if_else(!is.na(Countryname) & Countryname == "German Federal Republic", "Federal Republic
  ) %>%
  select(rcid, country, country_code, vote) %>%
  mutate(vote = as.character(vote)) %>%
  mutate(vote = fct_recode(vote, yes = "1", abstain = "2", no = "3"))

descriptions <- completeVotes %>%
  select(session, rcid, abstain, yes, no, importantvote, date, unres, amend, para, short, descr, me, nu
  distinct(rcid, .keep_all = TRUE)

un_roll_calls <- descriptions %>%
  select(rcid, session, importantvote:descr) %>%
  mutate(rcid = as.integer(rcid),
         date = as.Date(date)) %>%
  arrange(rcid)

un_roll_call_issues <- descriptions %>%
  select(rcid, me:ec) %>%
  gather(short_name, value, me:ec) %>%
  mutate(rcid = as.integer(rcid),
         value = as.numeric(value)) %>%
  filter(value == 1) %>%
  select(-value) %>%
  mutate(issue = fct_recode(short_name,
                            "Palestinian conflict" = "me",
                            "Nuclear weapons and nuclear material" = "nu",
                            "Colonialism" = "co",
                            "Human rights" = "hr",
                            "Economic development" = "ec",
                            "Arms control and disarmament" = "di"))
```

Data Exploration

First, I looked at what issues are addressed at UN meetings.

```
#find out what kinds of issues are voted on
unique(un_roll_call_issues$issue)
```

```
## [1] Palestinian conflict                Nuclear weapons and nuclear material
## [3] Arms control and disarmament        Human rights
## [5] Colonialism                         Economic development
```

```
## 6 Levels: Colonialism Arms control and disarmament ... Nuclear weapons and nuclear material
```

The topic of nuclear weapons and material caught my attention, so I decided to create a visual showing the number of times this issue was addressed in UN meetings during the Cold War period. To make it a little bit more informative/interesting, I also decided to add some "timeline events" relating to the Cold War. I chose a few based off of this list by JB Berglund (https://www.preceden.com/timelines/46853-the-top-10-events-of-the-cold-war).

Data Wrangling

My first step for data wrangling was to add separate columns for month, day, and year. I did this proactively, as I suspected the date format would cause me some difficulties.

```
#add columns for year, month, and day
un_roll_calls$year = format(as.Date(un_roll_calls$date, format="%d/%m/%Y"),"%Y")
un_roll_calls$month = format(as.Date(un_roll_calls$date, format="%d/%m/%Y"),"%m")
un_roll_calls$day = format(as.Date(un_roll_calls$date, format="%d/%m/%Y"),"%d")
```

Next, I had to find a way to identify which UN sessions addressed nuclear weapons. This task wasn't as straightforward because the topics addressed were contained in the `un_roll_call_issues` dataset whereas the dates of the sessions were in the `un_roll_calls` dataset. I started by getting the roll call id numbers from the issues dataset. Then, I used these id numbers to select rows from the roll calls which corresponded to meetings in which nuclear weapons were addressed as an issue. Finally, I subset the data to be between the years 1947 and 1991 so that the data represents events which occurred during the Cold War era.

```
#get rcids for nuclear issues from un_roll_call_issues
nuclear_subset <- subset(un_roll_call_issues, short_name=="nu")
nuclear_rcids <- unique(nuclear_subset$rcid)

#subset un_roll_calls to only nuclear issues
nuclear_roll_calls = subset(un_roll_calls, rcid %in% nuclear_rcids)

#subset nuclear_roll_calls to cold war period
cold_war = subset(nuclear_roll_calls, year>=1947 & year<=1991)
```

Visualization - Preparation

Before I made my visualization, there were a few tasks I had to complete first, namely:
* **(1) Group my data by date in a way which would facilitate plotting -** To accomplish this, I converted the month to a numerical value and added it to year
* **(2) Create a dataframe containing the information about the events I wanted to include**
* **(3) Add some custom fonts**

```
#tidy data to prepare for plot
cold_war_nuclear_issues <- cold_war %>%
  group_by(date) %>%
  mutate(count_issues = length(unique(rcid))) %>%
  mutate(month_decimal = as.numeric(month) * 0.08) %>%
  mutate(x_axis = as.numeric(year) + month_decimal) %>%
  select(date, year, month, day, count_issues, x_axis)
data_ = cold_war_nuclear_issues


#create dataframe for cold war events
event = c("Korean War", "U-2 Incident", "Cuban Missile Crisis", "Strategic Arms Limitation Talks", "Sov
x_end = c(1950.48, 1960.4, 1962.8, 1969.88, 1979.96, 1989.88)
```

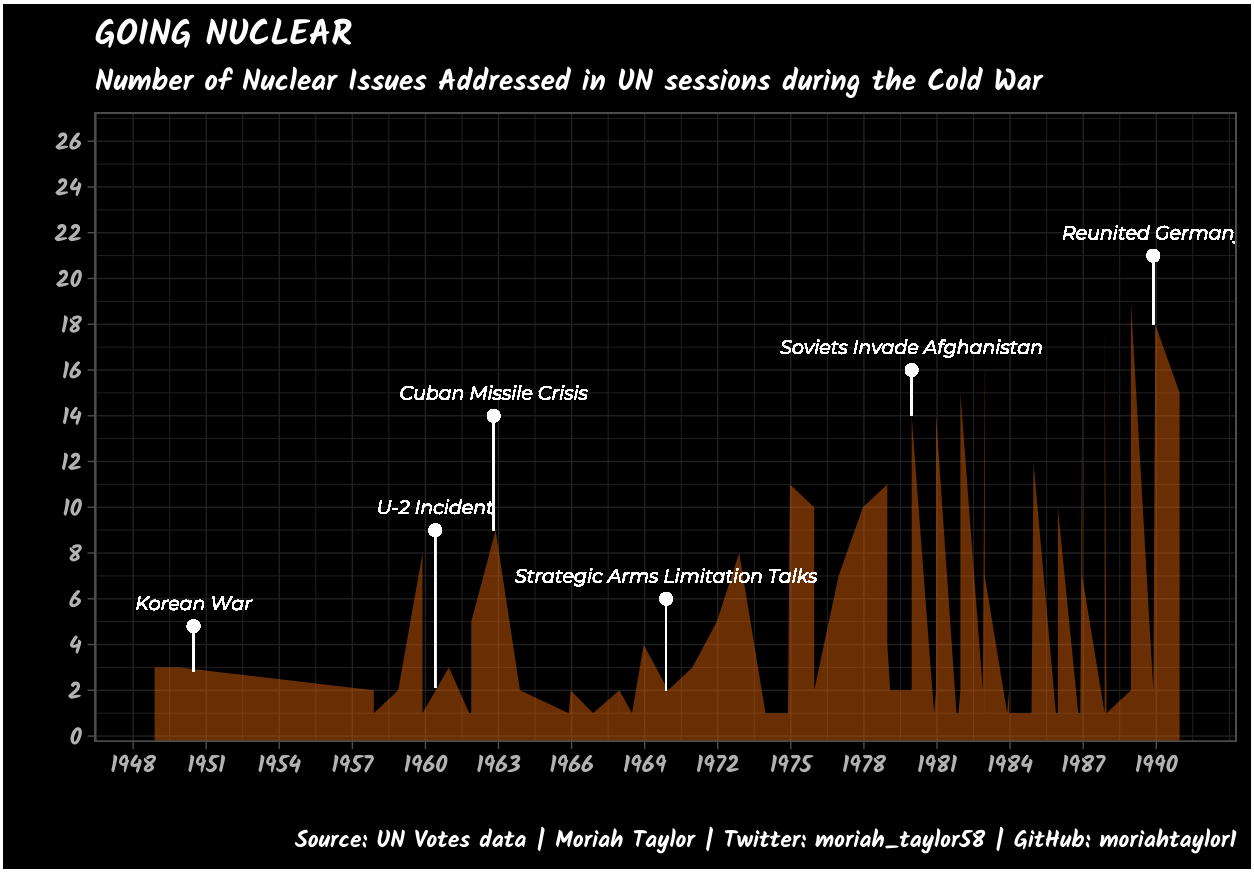
```
y_end = c(2.8, 2.1, 9, 2, 14, 18)   #adjusted these values manually using trial and error for each plot
y_ = c(4.8, 9, 14, 6, 16, 21)       #adjusted these values manually using trial and error for each plot
events = data.frame(event, x_end, y_, y_end)


#import custom text
font_add(family = "kalam", "Kalam-Bold.ttf")
font_add(family = "montserrat-ital", "Montserrat-Italic.ttf")
showtext.auto()
loadfonts()
```

Visualization - Results

```
#create visualization
plot <- ggplot(data=data_, aes(x=x_axis, y=count_issues, ymax=count_issues)) +  #line graph
  geom_ribbon(ymin=-1, fill='#D25c07', alpha=0.5) +    #fill under line graph
  #x-axis
  scale_x_continuous(
    breaks = seq(1948, 1990, 3),
    labels = paste0(seq(1948, 1990, 3)),
    limits = c(1948.56, 1991.16)
  ) +
  #y-axis
  scale_y_continuous(
    breaks = seq(0,26,2),
    labels = paste0(seq(0,26,2)),
    limits = c(1,26),
  ) +
  #titles
  labs(
    title = "GOING NUCLEAR",
    subtitle = "Number of Nuclear Issues Addressed in UN sessions during the Cold War",
    caption = "Source: UN Votes data | Moriah Taylor | Twitter: moriah_taylor58 | GitHub: moriahtaylor1
  ) + xlab("") + ylab("") +
  #styling
  theme(
    axis.text.x = element_text(size = 14),
    axis.text.y = element_text(size = 14),
    axis.ticks = element_blank(),
    panel.background = element_rect(fill = "#dadbde"),
    plot.background = element_rect(fill = "#dadbde"),
    panel.grid.major = element_line(color = "#848690"),
    axis.line = element_line(color = "#848690"),
    plot.title = element_text(face = "bold", hjust = 0.5, size = 25),
    plot.subtitle = element_text(hjust = 0.5, size = 18),
    plot.caption = element_text(size = 12)) +
      dark_theme_light(base_family="kalam") +
  #add lines for each event
  geom_segment(aes(x=events$x_end[1], xend=events$x_end[1], y=events$y_[1], yend=events$y_end[1]), size=
  geom_segment(aes(x=events$x_end[2], xend=events$x_end[2], y=events$y_[2], yend=events$y_end[2]), size=
  geom_segment(aes(x=events$x_end[3], xend=events$x_end[3], y=events$y_[3], yend=events$y_end[3]), size=
  geom_segment(aes(x=events$x_end[4], xend=events$x_end[4], y=events$y_[4], yend=events$y_end[4]), size=
  geom_segment(aes(x=events$x_end[5], xend=events$x_end[5], y=events$y_[5], yend=events$y_end[5]), size=
```

```
  geom_segment(aes(x=events$x_end[6], xend=events$x_end[6], y=events$y_[6], yend=events$y_end[6]), size=
  #add dots for each event
  geom_point(aes(x=events$x_end[1], y=events$y_[1]), shape=16, size=2) +
  geom_point(aes(x=events$x_end[2], y=events$y_[2]), shape=16, size=2) +
  geom_point(aes(x=events$x_end[3], y=events$y_[3]), shape=16, size=2) +
  geom_point(aes(x=events$x_end[4], y=events$y_[4]), shape=16, size=2) +
  geom_point(aes(x=events$x_end[5], y=events$y_[5]), shape=16, size=2) +
  geom_point(aes(x=events$x_end[6], y=events$y_[6]), shape=16, size=2) +
  #add event info
  geom_text(aes(x= events$x_end[1], y=events$y_[1]+1,
                label=events$event[1]), family="montserrat-ital", size=2.5) +
  geom_text(aes(x= events$x_end[2], y=events$y_[2]+1,
                label=events$event[2]), family="montserrat-ital", size=2.5) +
  geom_text(aes(x= events$x_end[3], y=events$y_[3]+1,
                label=events$event[3]), family="montserrat-ital", size=2.5) +
  geom_text(aes(x= events$x_end[4], y=events$y_[4]+1,
                label=events$event[4]), family="montserrat-ital", size=2.5) +
  geom_text(aes(x= events$x_end[5], y=events$y_[5]+1,
                label=events$event[5]), family="montserrat-ital", size=2.5) +
  geom_text(aes(x= events$x_end[6], y=events$y_[6]+1,
                label=events$event[6]), family="montserrat-ital", size=2.5)
plot
```




```
#save plot to pdf
#ggsave("nuclear_issues_plot.pdf")
```

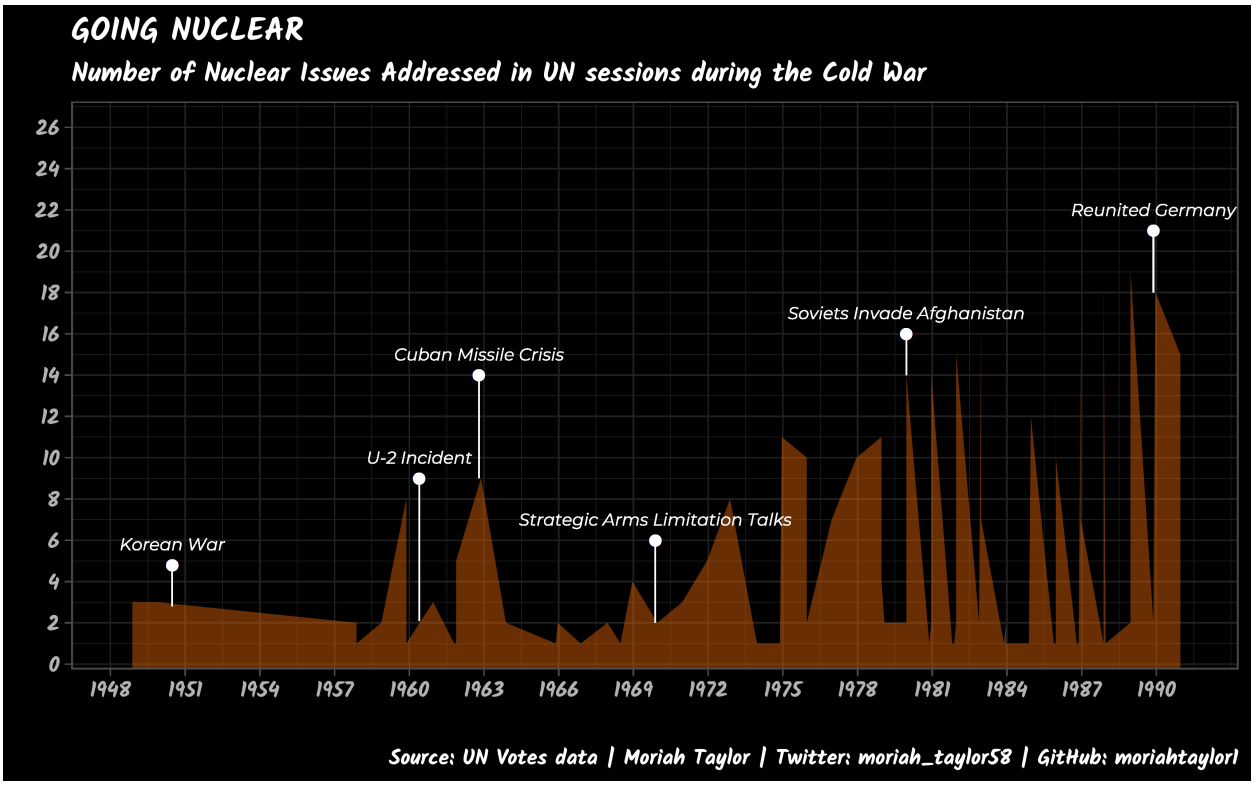


Figure 1: Going Nuclear: A TidyTuesday Creation

```
#show R where GhostScript is
    #Sys.setenv(R_GSCMD = "C:/Program Files/gs/gs9.53.3/bin/gswin64c.exe")    #commented out to compile
#embed fonts into the pdf generated earlier
    #embed_fonts("nuclear_issues_plot.pdf")    #commented out to compile output
```

Acknowledgments

Giving Credit Where Credit is Due

There are many different resources I used to get to the end result, the first of which is code from @etiennebacher's submission last week (code: https://t.co/OyTCaqSYHI?amp=1). I knew I wanted to do an area chart with some annotations and since I didn't even know where to start, I used this as a reference for beginning to build this visual. Another significant source I referenced was an article called "Visualizing Real World Data Timelines in R" authored by Sophia Shalhout and David Michael Miller. This can be found on The Miller Lab's website: https://t.co/OyTCaqSYHI?amp=1. In general, I also referenced the code of other members of the R4DS community that have been posted on GitHub, as well as the guides given on the R Graph Gallery website (r-graph-gallery.com).

TidyTuesday

Join the R4DS Online Learning Community in the weekly #TidyTuesday event! Every week we post a raw dataset, a chart or article related to that dataset, and ask you to explore the data. While the dataset will be "tamed", it will not always be tidy! As such you might need to apply various R for Data Science techniques to wrangle the data into a true tidy format. The goal of TidyTuesday is to apply your R skills, get feedback,

explore other's work, and connect with the greater #RStats community! As such we encourage everyone of all skills to participate!