AEA Report

Stuti Goyal

```r
library(tidyverse)
```

```
-- Attaching packages --------------------------------------- tidyverse 1.3.2 --
v ggplot2 3.3.6      v purrr   0.3.5
v tibble  3.1.8      v dplyr   1.0.10
v tidyr   1.2.1      v stringr 1.4.1
v readr   2.1.3      v forcats 0.5.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
```

```r
library(janitor)
```

```
Attaching package: 'janitor'

The following objects are masked from 'package:stats':

    chisq.test, fisher.test
```

```r
library(dataverse)
library(ggplot2)
library(qdapRegex)
```

```
Attaching package: 'qdapRegex'
```

```
The following object is masked from 'package:dplyr':

    explain

The following object is masked from 'package:ggplot2':

    %+%
```

```r
library(here)
```

```
here() starts at /home/rstudio/AEA_registryanalysis
```

```r
library(digest)

# dynamically set directory
basedir <- here::here()

# configure some placeholders

dv.fileid <- "6690545"
dv.doi    <- "DVN/TGMJFD"

# directories

outputs  <- file.path(basedir,"Output")
data     <- file.path(basedir,"Data")


for ( dir in list(outputs,data)){
  if (file.exists(dir)){
  } else {
    dir.create(file.path(dir))
  }
}

# convenience functions outsourced...

source(file.path(basedir,"Scripts","00_functions.R"))
```

```
[1] "File for export to LaTeX found: /home/rstudio/AEA_registryanalysis/Output/latexnums.Rda"
```

```r
# For graphing

evenbreaks = c(2014, 2016, 2018, 2020, 2022)
oddbreaks = c(2013, 2015, 2017, 2019, 2021)


# file names

rct.file.local <- file.path(data,"trials.Rds")
rct.file.chk256 <- "f99e0af9804a253960738bfbb255aa85359042efbab213ab46ad047d3ae515ab"
```

```r
if ( file.exists(rct.file.local)) {
  message(paste0("Using local file ",rct.file.local))
  aea_orig <- readRDS(file=rct.file.local)

} else {
   aea_orig <- get_dataframe_by_id(file = dv.fileid,
                             server = "dataverse.harvard.edu",
                             .f = read.csv, original = T, )
   saveRDS(aea_orig,file=rct.file.local)
}
```

```
Using local file /home/rstudio/AEA_registryanalysis/Data/trials.Rds
```

```r
  rct.test.chksum <- digest(rct.file.local,algo="sha256")
  message(paste0("SHA256: ",rct.test.chksum))
```

```
SHA256: f99e0af9804a253960738bfbb255aa85359042efbab213ab46ad047d3ae515ab
```

```r
  if ( rct.test.chksum != rct.file.chk256) {
    warning("Checksum is not equal to config")
  }

## Yearwise Visualisations

## This can also be used to pull the dataset, but harvard dataverse's API works more
#frequently with fewer bugs with file ids
# aea_data <- get_dataframe_by_name(filename ="trials.tab", dataset = https://doi.org/10.7
#                             server = "dataverse.harvard.edu",.f = read.csv, original = T)
```

```r
metadata <- get_dataset(dataset = paste0("https://doi.org/10.7910/",dv.doi),
                        server = "dataverse.harvard.edu")

## Printing the metadata here

### The title for the data retrieved above:
print(metadata$metadataBlocks$citation$fields$value[[1]][1])
```

```
[1] "Registrations in the AEA RCT Registry (2013-05-15 through 2022-11-01)"
```

```r
print(metadata$datasetPersistentId)
```

```
[1] "doi:10.7910/DVN/TGMJFD"
```

```r
aea_data <- clean_names(aea_orig)
```

Data Wrangling

```r
aea_data_year <- aea_data %>%
  mutate(first_registered_on = as.Date(first_registered_on, format = "%Y-%m-%d")) %>%
#   mutate(first_registered_on = str_replace(first_registered_on, "00", "20" )) %>%
  mutate(first_registered_year = format(as.Date(first_registered_on), "%Y"))
```

```r
year_cnt <- aea_data_year %>%
  group_by(first_registered_year) %>%
  summarise(cnt = n()) %>%
  ungroup()
```

```r
# Creating data set with month and year
aea_year_month <- aea_data_year %>%
  mutate(
    first_month_year = format(as.Date(first_registered_on), "%Y-%m"),
    first_month = format(as.Date(first_registered_on), "%m")
  )
```

```r
# Creating variable to reflect month (used for cumulative count predictions)
aea_year_month <- aea_data_year %>%
  mutate(
    first_month_year = format(as.Date(first_registered_on), "%m-%Y"),
    first_month = format(as.Date(first_registered_on), "%m")
  )
```

```r
# Creating counts of registrations by year
year_cnt <- aea_year_month %>%
  group_by(first_registered_year) %>%
  summarise(cnt_yr = n()) %>%
  ungroup()

#Data set of counts by year (excluding 2022)
year_wo_2022 <- year_cnt %>%
  filter(first_registered_year != 2022)
```

Summary of code

In this section, the data has been loaded through the API, following which, the first registered year and month were extracted from the variable `first_registered_on`; these variables are called `first_registered_year` and `first_month`, respectively. After this, I created a dataset which summarizes the total number of registrations for each year. I also created a dataset with these counts, but without the observations for 2022 to facilitate the predictions below.

Predictions for total counts

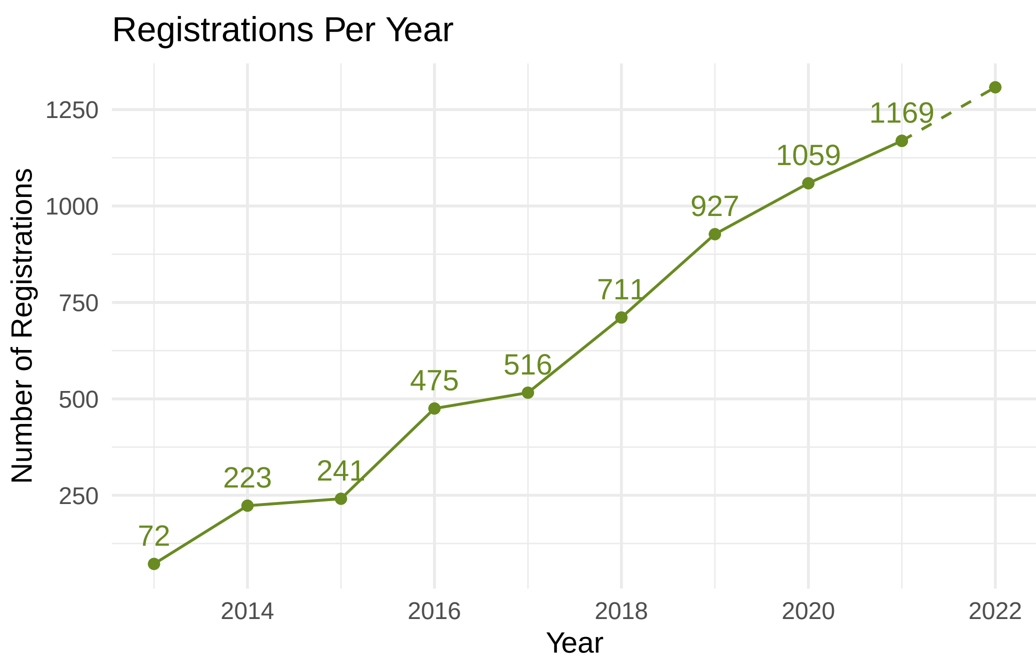
```r
cnt_lm <- lm(cnt_yr ~ as.numeric(first_registered_year), data = year_wo_2022)
summary(cnt_lm)
```

```

Call:
lm(formula = cnt_yr ~ as.numeric(first_registered_year), data = year_wo_2022)

Residuals:
   Min     1Q Median     3Q    Max 
-83.22 -29.96  17.51  39.71  48.98 

Coefficients:
```

```
                                    Estimate Std. Error t value Pr(>|t|)
(Intercept)                       -2.853e+05  1.422e+04  -20.07 1.91e-07 ***
as.numeric(first_registered_year)  1.417e+02  7.048e+00   20.11 1.88e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 54.59 on 7 degrees of freedom
Multiple R-squared:  0.983, Adjusted R-squared:  0.9806
F-statistic: 404.4 on 1 and 7 DF,  p-value: 1.882e-07
```

```
# Creating data frame with prediction for 2022
years <- data.frame(first_registered_year = c(2022))

pred_cnt_2022 <- data.frame(year = c(2021, 2022), cnt = c(year_cnt$cnt_yr[year_cnt$first_r
```

```
year_wo_2022 %>%
  mutate(first_registered_year = as.numeric(first_registered_year)) %>%
  ggplot(aes(x = first_registered_year, y = cnt_yr)) +
  geom_line(color = "olivedrab4") +
  geom_text(aes(label = cnt_yr), color = "olivedrab4", position = position_nudge(y = -2),
  geom_line(aes(x = year, y = cnt), data = pred_cnt_2022, linetype = "dashed", color = "ol
  geom_point(aes(x = year, y = cnt), data = pred_cnt_2022, color = "olivedrab4") +
  geom_point(color = "olivedrab4") +
  scale_x_continuous(
    name = "Year",
    limits = c(2013, 2022), breaks = evenbreaks)+
  scale_y_continuous(
    breaks = seq(0, 1500, 250)
  ) +
  # geom_text(aes(y = round(pred_cnt_2022$cnt[pred_cnt_2022$year == "2022"], 2), x = 2022,
  labs(title = "Registrations Per Year", y = "Number of Registrations") +
  theme(legend.position = c(0.87, 0.25))+
  theme_minimal()
```




```
ggsave(file.path(outputs,"reg_pre_year.pdf"),width=3.5,height=3.5,units="in")
ggsave(file.path(outputs,"reg_pre_year.png"),width=3.5,height=3.5,units="in")

num_regsyearly <- round(pred_cnt_2022 %>% filter(year==2022) %>%
                                        pull(cnt),0)

update_latexnums("regsyearly",num_regsyearly)
```

```
Updating existing field regsyearly
```

In the last year, 1308 registrations were added.

Summary of code

In this section, I created a linear regression model to predict the total number of registrations in 2022, based on the year. For this, I used the dataset with the registration counts for each year (excluding 2022). I did this since the data reflected records until October of 2022, which means that using the existing data would be an inaccurate reflection of the number of registrations in 2022. Finally, I graphed the existing data and the predictions, with the prediction indicated by a dashed line.

Predictions for cumulative counts

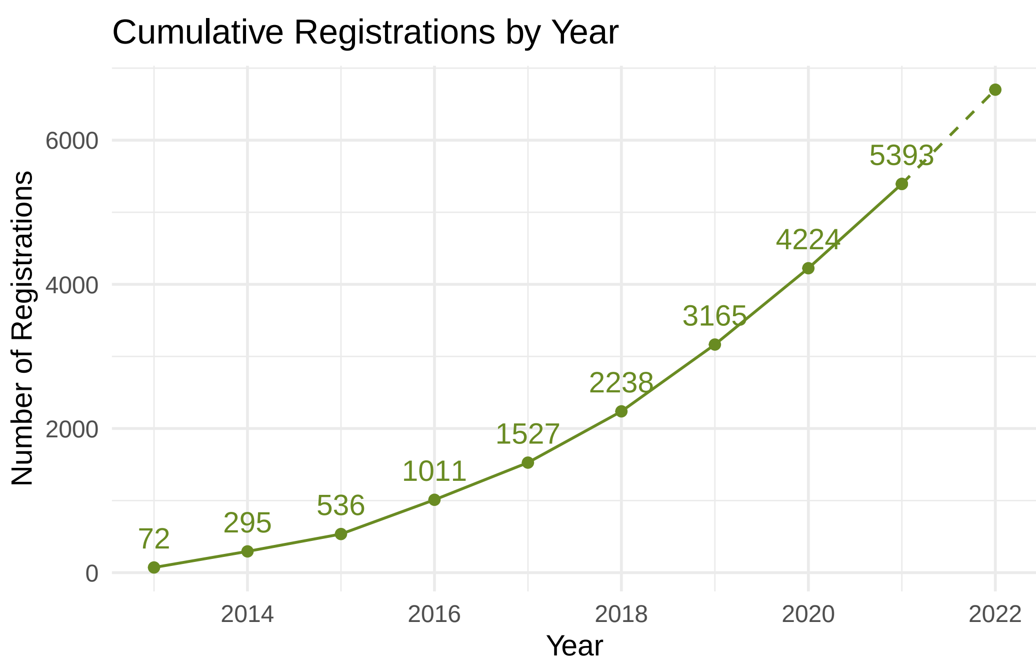
```r
cnt_mnth_yr_2022 <- aea_year_month %>%
  filter(first_registered_year == "2022") %>%
  group_by(first_month) %>%
  summarise(cnt = n()) %>%
  ungroup()
```

```r
yr_cnt_no_2022 <- filter(year_cnt, first_registered_year != "2022")

cum_cnt_no_2022 <- data.frame(
  cum_cnt = cumsum(yr_cnt_no_2022[, 2])
  )

cum_cnt_no_2022 <- cbind(yr_cnt_no_2022[, 1], cum_cnt_no_2022)
```

```r
years <- data.frame(first_registered_year = c(2022))

cnt_2022 <- data.frame(
  "year" = c(2021, 2022),
  "cnt" = c(
    cum_cnt_no_2022$cnt_yr[cum_cnt_no_2022$first_registered_year == "2021"],
    ( predict(cnt_lm, years)+ cum_cnt_no_2022$cnt_yr[cum_cnt_no_2022$first_registered_year
  )
)
```

```r
cum_cnt_no_2022 %>%
  mutate(first_registered_year = as.numeric(first_registered_year)) %>%
  ggplot(mapping = aes(x = first_registered_year, y = cnt_yr)) +
  geom_line(color = "olivedrab4") +
  geom_point(color = "olivedrab4") +
  geom_line(
    data = cnt_2022,
    mapping = aes(x = year, y = cnt),
    linetype = "dashed",
    color = "olivedrab4"
  ) +
  geom_point(
    data = cnt_2022,
    mapping = aes(x = year, y = cnt),
```

```
    color = "olivedrab4"
  ) +
  geom_text(
    aes(x = first_registered_year, y = cnt_yr, label = cnt_yr),
    vjust = -0.9,
    color = "olivedrab4"
  ) +
  scale_x_continuous(
    limits = c(2013, 2022), breaks = evenbreaks)+
  labs(
    title = "Cumulative Registrations by Year",
    x = "Year",
    y = "Number of Registrations"
  ) +
  theme(legend.position = c(0.87, 0.25))+
  theme_minimal()
```




```
num_regscumul <- round((cnt_2022 %>% filter(year==2022) %>%
                                      pull(cnt))/100,0)*100

update_latexnums("regscumul",num_regscumul)
```

```
Updating existing field regscumul
```

```
ggsave(file.path(outputs,"reg_cumulative.pdf"),width=3.5,height=3.5,units="in")
ggsave(file.path(outputs,"reg_cumulative.png"),width=3.5,height=3.5,units="in")
```

As of this year, there are a total of 6700 registrations.

Summary of code

Similarly, in this section, I created a linear regression model to predict the cumulative number of registrations by 2022, based on the year. For this, I used the dataset with the registration counts for each year (excluding 2022). I did this since the data reflected records until October of 2022, which means that using the existing data would be an inaccurate reflection of the number of registrations by 2022. Finally, I graphed the existing data and the predictions, with the prediction indicated by a dashed line.

Predictions for pre vs post reg

```
aea_data_year <- aea_data_year %>%
  mutate(intervention_start_date = as.Date(intervention_start_date, format = "%Y-%m-%d"))
# %>%
#   mutate(intervention_start_date = str_replace(intervention_start_date, "00", "20" ))
```

```
pre_reg_cnt <- aea_data_year %>%
  mutate(pre_post = ifelse(first_registered_on < intervention_start_date, "pre_reg", "post
  group_by(first_registered_year, pre_post) %>%
  summarise(reg_cnt = n()) %>%
  ungroup()
```

```
`summarise()` has grouped output by 'first_registered_year'. You can override
using the `.groups` argument.
```

```
pre_reg_no_2022 <- pre_reg_cnt %>%
  filter(first_registered_year != "2022")
```

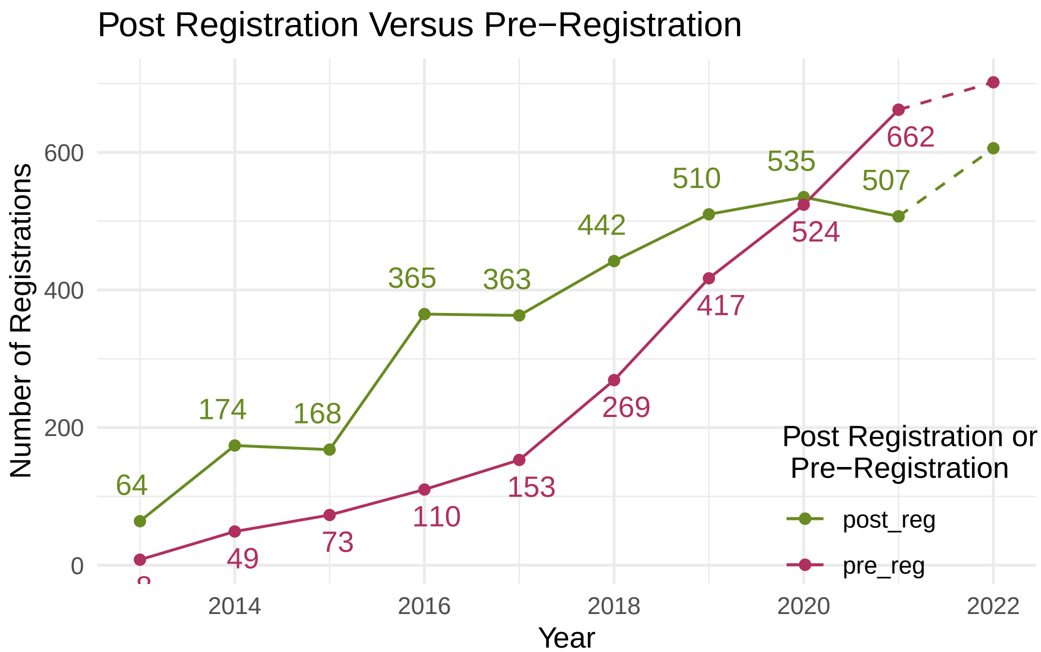
```
pre_reg_lm <- lm(reg_cnt ~ as.numeric(first_registered_year) + pre_post, data = pre_reg_no
summary(pre_reg_lm)
```

```
Call:
lm(formula = reg_cnt ~ as.numeric(first_registered_year) + pre_post,
    data = pre_reg_no_2022)

Residuals:
    Min      1Q  Median      3Q     Max
-124.02  -37.60   12.69   35.18  126.87

Coefficients:
                                    Estimate Std. Error t value Pr(>|t|)
(Intercept)                       -1.426e+05  1.248e+04 -11.429 8.39e-09 ***
as.numeric(first_registered_year)  7.087e+01  6.186e+00  11.457 8.12e-09 ***
pre_postpre_reg                   -9.589e+01  3.194e+01  -3.002  0.00894 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 67.76 on 15 degrees of freedom
Multiple R-squared:  0.9034,    Adjusted R-squared:  0.8905
F-statistic: 70.13 on 2 and 15 DF,  p-value: 2.441e-08
```

```r
years_reg <- data.frame(first_registered_year = c(2022), pre_post = c("post_reg", "pre_reg

reg_pred_2022 <- data.frame(
  year = c("2021", "2021", "2022", "2022"),
  pre_post = c("pre_reg", "post_reg"),
  cnt = c(
    pre_reg_cnt$reg_cnt[pre_reg_cnt$first_registered_year == "2021" & pre_reg_cnt$pre_post
    pre_reg_cnt$reg_cnt[pre_reg_cnt$first_registered_year == "2021" & pre_reg_cnt$pre_post
    predict(pre_reg_lm, years_reg))
  )
```

```r
pre_reg_no_2022$first_registered_year <- as.numeric(pre_reg_no_2022$first_registered_year)

reg_pred_2022$year <- as.numeric(reg_pred_2022$year)

ggplot(data = pre_reg_no_2022, mapping = aes(x = first_registered_year, y = reg_cnt, color
  geom_path(aes(group = pre_post)) +
  geom_point() +
  theme_minimal() +
  geom_path(aes(x = year, y = cnt, color = pre_post, group = pre_post), data = reg_pred_20
```

```
geom_point(aes(x = year, y = cnt, color = pre_post), data = reg_pred_2022) +
geom_text(
  data = filter(pre_reg_no_2022, pre_post == "post_reg"),
  aes(x = first_registered_year, y = reg_cnt, label = reg_cnt), vjust = -1.25, hjust = 0
  color = "olivedrab4"
) +
geom_text(
  data = filter(pre_reg_no_2022, pre_post == "pre_reg"),
  aes(x = first_registered_year, y = reg_cnt, label = reg_cnt), vjust = 1.75, hjust = 0.
  color = "maroon"
) +
scale_color_manual(
  values = c("olivedrab4", "maroon")
) +
labs(
  title = "Post Registration Versus Pre-Registration",
  x = "Year",
  y = "Number of Registrations",
  color = "Post Registration or \n Pre-Registration"
)+
scale_x_continuous( limits = c(2013, 2022),
                    breaks = evenbreaks)+
theme_minimal()+
theme(legend.position = c(0.87, 0.15))
```




```
ggsave(file.path(outputs,"post_pre_reg.pdf"),width=3.5,height=3.5,units="in")
ggsave(file.path(outputs,"post_pre_reg.png"),width=3.5,height=3.5,units="in")
```

Summary of code

In this section, I created a linear regression model to predict the number of registrations in 2022 that were pre registered and those that weren't. For this, I used the dataset with the pre - registration and post registration counts for each year (excluding 2022). I did this since the data reflected records until October of 2022, which means that using the existing data would be an inaccurate reflection of the number of registrations in 2022. Finally, I graphed the existing data and the predictions, with the prediction indicated by a dashed line.

Predictions for pap vs total counts

```
pap_cnt <- aea_data_year %>%
  filter(analysis_plan_documents != "None") %>%
  group_by(first_registered_year) %>%
  summarise(pap_cnt = n()) %>%
  cbind(year_cnt[, 2])
```

```r
pap_wo_2022 <- pap_cnt %>%
  filter(first_registered_year != 2022)
```

```r
pap_lm <- lm(pap_cnt ~ as.numeric(first_registered_year), data = pap_wo_2022)
summary(pap_lm)
```

```

Call:
lm(formula = pap_cnt ~ as.numeric(first_registered_year), data = pap_wo_2022)

Residuals:
    Min      1Q  Median      3Q     Max 
-39.933 -17.533   5.267  13.467  37.267 

Coefficients:
                                    Estimate Std. Error t value Pr(>|t|)    
(Intercept)                       -1.236e+05  6.544e+03  -18.89 2.90e-07 ***
as.numeric(first_registered_year)  6.140e+01  3.244e+00   18.93 2.86e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 25.13 on 7 degrees of freedom
Multiple R-squared:  0.9808,	Adjusted R-squared:  0.9781 
F-statistic: 358.2 on 1 and 7 DF,  p-value: 2.859e-07
```

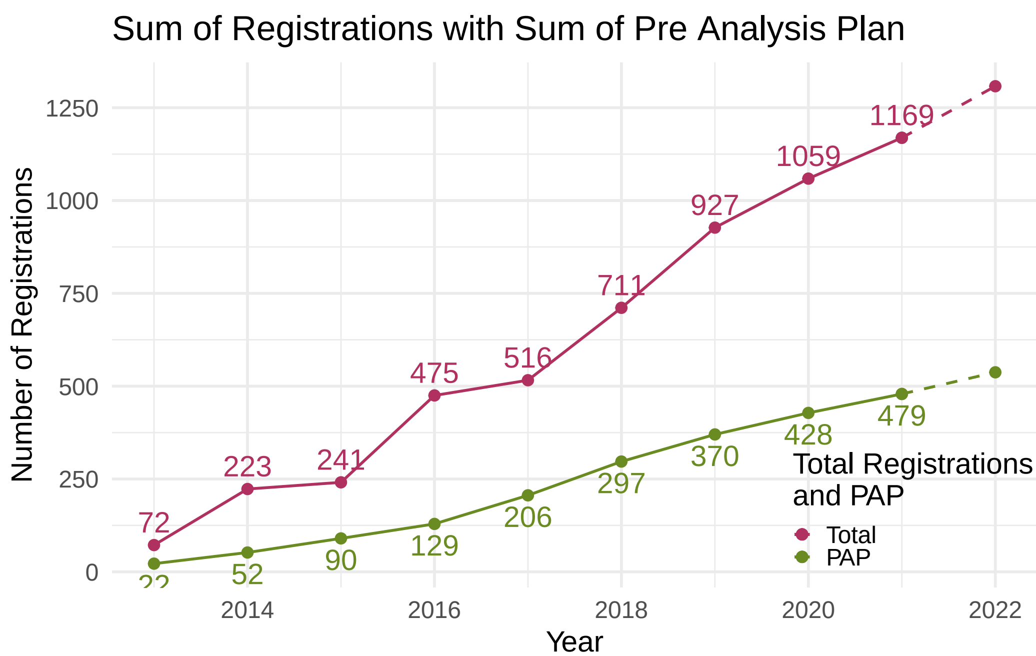
```r
pap_reg <- data.frame(first_registered_year = c(2022))

pap_pred_2022 <- data.frame(
  year = c(2021, 2022),
  pap_cnt = c(
    pap_cnt$pap_cnt[pap_cnt$first_registered_year == "2021"],
    predict(pap_lm, pap_reg)),
  cnt_yr = c(
    pap_cnt$cnt_yr[pap_cnt$first_registered_year == "2021"],
    predict(cnt_lm, pap_reg))
  )
```

```r
pap_wo_2022 %>%
  mutate(first_registered_year = as.numeric(first_registered_year)) %>%
  ggplot() +
```

```r
geom_line(aes(x = first_registered_year, y = pap_cnt, color = "PAP")) +
geom_point(aes(x = first_registered_year, y = pap_cnt, color = "PAP")) +
geom_line(
  data = pap_pred_2022,
  mapping = aes(x = year, y = pap_cnt, color = "PAP"),
  linetype = "dashed"
) +
geom_point(
  data = pap_pred_2022,
  mapping = aes(x = year, y = pap_cnt, color = "PAP")
) +
geom_line(aes(x = first_registered_year, y = cnt_yr, color = "Total")) +
geom_point(aes(x = first_registered_year, y = cnt_yr, color = "Total")) +
geom_line(
  data = pap_pred_2022,
  mapping = aes(x = year, y = cnt_yr, color = "Total"),
  linetype = "dashed"
) +
geom_point(
  data = pap_pred_2022,
  mapping = aes(x = year, y = cnt_yr, color = "Total")
) +
scale_color_manual(
  values = c("Total" = "maroon", "PAP" = "olivedrab4")
) +
theme_minimal() +
scale_x_continuous( limits = c(2013, 2022), breaks = evenbreaks)+
scale_y_continuous(
  breaks = seq(0, 1500, 250)
) +
scale_x_continuous(limits = c(2013, 2022), breaks = evenbreaks)+
labs(
  title = "Sum of Registrations with Sum of Pre Analysis Plan",
  x = "Year",
  y = "Number of Registrations",
  color = "Total Registrations \nand PAP"
) +
geom_text(
  aes(x = first_registered_year, y = pap_cnt, label = pap_cnt),
  vjust = 1.5,
  color = "olivedrab4"
```

```
  ) +
  geom_text(
    aes(x = first_registered_year, y = cnt_yr, label = cnt_yr),
    vjust = -0.6,
    color = "maroon"
  ) +
  theme(legend.position = c(0.87, 0.15), legend.key.size = unit(0.25, 'cm'))
```

```
Scale for 'x' is already present. Adding another scale for 'x', which will
replace the existing scale.
```




```
ggsave(file.path(outputs,"pap_reg.pdf"),width=3.5,height=3.5,units="in")
ggsave(file.path(outputs,"pap_reg.png"),width=3.5,height=3.5,units="in")
```

Summary of code

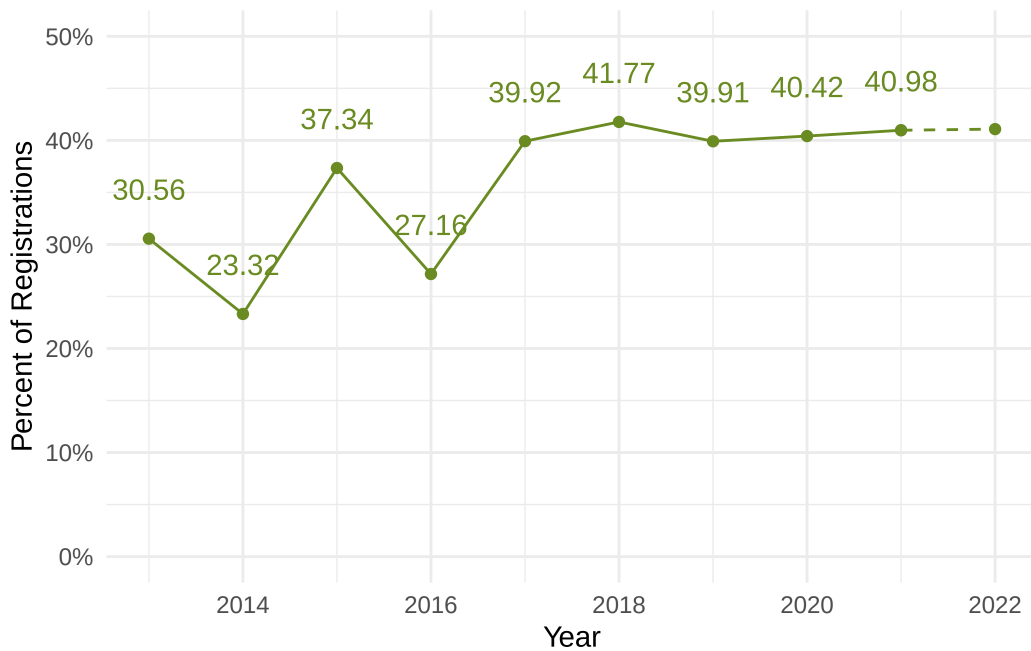
In this section, I created a linear regression model to predict the number of registrations in 2022 that had a Pre Analysis Plan attached. For this, I used the dataset with the PAP registration counts for each year (excluding 2022). I did this since the data reflected records until October of 2022, which means that using the existing data would be an inaccurate reflection of the number of registrations in 2022. Finally, I graphed the existing data and the predictions, with the prediction indicated by a dashed line.

Predictions for percentage PAP

```
pap_wo_2022 <- pap_wo_2022 %>%
  mutate(pap_percent = (pap_cnt / cnt_yr))

pap_pred_2022 <- pap_pred_2022 %>%
  mutate(pap_percent = (pap_cnt / cnt_yr))
```

```
pap_wo_2022 %>%
  mutate(first_registered_year = as.numeric(first_registered_year)) %>%
  ggplot() +
  geom_line(
    mapping = aes(x = first_registered_year, y = pap_percent),
    color = "olivedrab4"
  ) +
  geom_point(
    mapping = aes(x = first_registered_year, y = pap_percent),
    color = "olivedrab4"
  ) +
  geom_line(
    data = pap_pred_2022,
    mapping = aes(x = year, y = pap_percent),
    linetype = "dashed",
    color = "olivedrab4"
  ) +
  geom_point(
    data = pap_pred_2022,
    mapping = aes(x = year, y = pap_percent),
    color = "olivedrab4"
  ) +
  geom_text(
    aes(x = first_registered_year, y = pap_percent, label = round(pap_percent * 100, 2)),
    vjust = -1.85,
    color = "olivedrab4"
  ) +
  labs(
    name = "Percentage of PAP Registered",
```

```
    x = "Year",
    y = "Percent of Registrations"
  ) +
  scale_y_continuous(
    labels = scales::percent,
    limits = c(0, 0.5)
  ) +
  theme_minimal()+
  scale_x_continuous(limits = c(2013, 2022),
                     breaks = evenbreaks)
```




```
ggsave(file.path(outputs,"pap_per_reg.pdf"),width=3.5,height=3.5,units="in")
ggsave(file.path(outputs,"pap_per_reg.png"),width=3.5,height=3.5,units="in")
```

Summary of code

For this, I used my predictions for PAP registrations in 2022 along with the existing data for previous years and found the percent of the registrations that had a PAP attached to the registration.

Functions to help split and reshape data such that one observation is one project-PI pair:

```r
#### Functions:
sep_help <- function(col, sep){
  max <- 0
  for (i in 1:length(col)){
    if (is.na(col[i])){
      next
    }
    iter <- str_count(col[i], sep)
    if (iter > max){
      max = iter
    }
    else{
      next
    }
  }
  return(max)
}

sep_ls <- function(col,sep,name){
  num <- sep_help(col,sep)
  num <- num + 1
  nums <- 1:num
  listy <-  paste(name, nums,sep = "")
  return(listy)
}
better_sep <- function(df,col,sep,name){
  names <- sep_ls(col,sep,name)
  str <-deparse(substitute(col))
  co <- sub(".*\\$", "",str)
  new_df <- df %>%
    separate(co, names, sep =sep)
  return(new_df)
}

reshape_long <- function(df, var_name){
  nms <- colnames(df)
  end1 <- str_detect(nms,var_name)
  end <- sum(end1 == TRUE)
  vn <- paste(var_name,"1",sep = "")
  pos <-match(vn, nms)
  pos_min <- pos-1
```

```r
  nms <- nms[1:pos_min]
  nms <- c(nms, var_name)
  new_df <- data.frame(matrix(ncol= length(nms), nrow = 0))
  colnames(new_df) <- nms
  for (q in 1:nrow(df)){
    iter <- df[q,]
    bool <- sapply(df, is.na)[q,]
    x<-1
    e <- var_name
    for (i in x:end){
      y <- paste(e,i, sep = "")
      if (bool[y] == TRUE){
        break
      }
      else{
        x = x+1
      }
    }
    x = x-1
    df_dup <- iter[rep(seq_len(nrow(iter)), each = x), ]
    df_new <- data.frame(matrix(ncol= length(new_df), nrow = x))
    df_new[,1:pos_min] <- df_dup[,1:pos_min]
    colnames(df_new) <- nms
    j = pos
    for (k in 1:nrow(df_new)){
      res <- as.character(df_dup[k,j])
      df_new[k,pos] = res
      j = j+1
    }
    new_df <- rbind(new_df, df_new)
  }
  return(new_df)
}
```

Graphing the growth of registered users

The following graphs the growth in the number of users registered on the AEA website (that is, those that have created a profile and are therefore capable of submitting trials). The data itself is from a previous AEA report (in 2019, available here). A transition in the web developer responsible for the website's maintenance prevents us from having numbers for 2020 and 2021. The number in 2022 is estimated by taking the number of registered users in 2022 through December 1 (8255), and then adding the monthly average growth between 2018 and 2022 (100).

```
years <- c(2014,2016,2018,2021+11/12,2022)
cum_reg_users <- c(744,1778,3472,8255,8255)
# adjust the partial 2022 number
avg = (cum_reg_users[4] - cum_reg_users[3])/(3*12+11)

df <- tibble(Year = years, Registered.users = cum_reg_users) %>%
  mutate(Registered.users = if_else(Year==2022,
                                    round(Registered.users + avg,0),
                                    Registered.users),
         label = if_else(Year==2022,
                                    "",
                                    as.character(Registered.users)))

registered.users.estimate = df %>% filter(Year==2022) %>% pull(Registered.users)
print(registered.users.estimate)
```

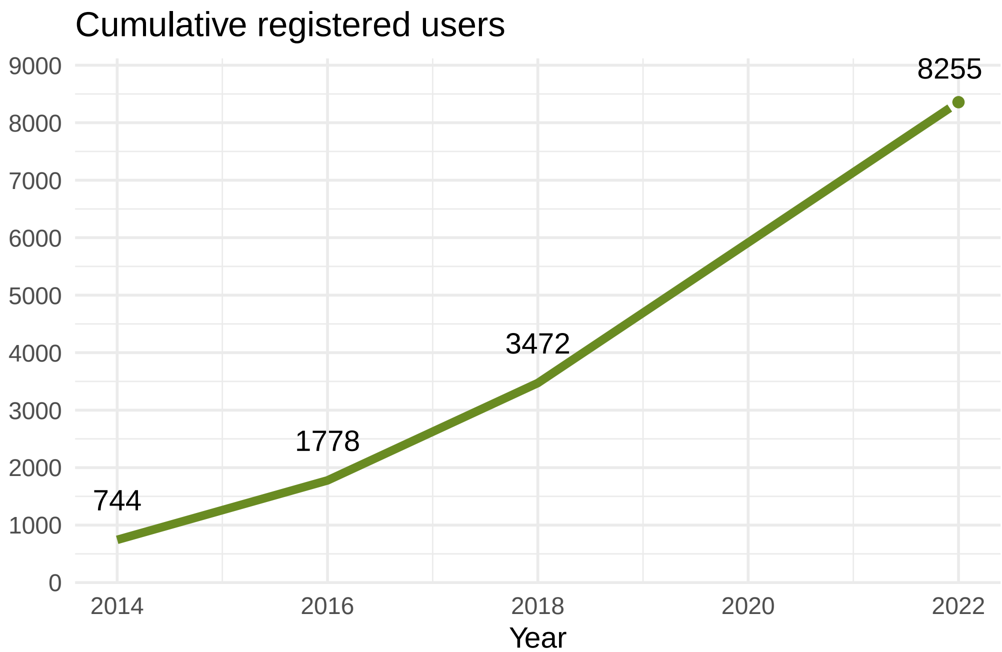
```
[1] 8357
```

```
registered.users.rounded = floor(registered.users.estimate/100)*100

update_latexnums("registeredusers",registered.users.rounded)
```

```
Updating existing field registeredusers
```

```
b <- ggplot(data=df, aes(x=Year,y=Registered.users))+
  geom_line(data = subset(df, Year < 2022),
            aes(y = Registered.users, group = 1),
            color = "olivedrab4", linetype = 1, size=1.5) +
  geom_point(data = subset(df, Year ==2022),
            aes(y = Registered.users, group = 1),
            color = "olivedrab4") +
  ggtitle("Cumulative registered users") +
  geom_text(y= df$Registered.users,
            label= df$label, vjust = -1.4) +
  scale_y_continuous(expand=c(0.1, 0),
                     n.breaks = 8) +
  labs(y= "", x= "Year") +
```

```
  theme(text = element_text(size = 20)) +
  theme_minimal()
b
```




```
ggsave(file.path(outputs,"registered_users.pdf"), b,width=3.5,height=3.5,units="in")
ggsave(file.path(outputs,"registered_users.png"), b,width=3.5,height=3.5,units="in")
```

As of this year, there are 8357 registered researchers across all registrations.

Getting the number of active users

Here active users are defined as those that either:

- Have a trial that is currently active (before its end date) on the registry
- Have a trial that they have updated within the last year on the registry

This will be a rough number, because there is heterogeneity in how PI's names are spelled from one trial to another (and so may therefore be a slight overcount). Efforts are taken to homogenize, but there may still be some duplicate PIs in the final count.

```r
### Getting number of active users:
# Defined by # of PIs with active projects or on registrations that have been updated in t

## First getting the right subset
aea_sm <- select(aea_orig, c(Title, Last.update.date,End.date,Primary.Investigator,Other.P

aea_sm$Last.update.date <- as.Date(aea_sm$Last.update.date, format = "%B %d, %Y")
aea_sm$End.date <- as.Date(aea_sm$End.date)

today <- Sys.Date()

aea_sm <- filter(aea_sm, End.date >= today | Last.update.date >= "2022-01-01")


### Then we separate out the PIs:
Oth <- select(aea_sm, -c(Primary.Investigator))

Oth <- better_sep(Oth,Oth$Other.Primary.Investigators,"; ","PIs")
```

```
Warning: Expected 15 pieces. Missing pieces filled with `NA` in 1877 rows [1, 2,
3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

```r
Oth_long <- reshape_long(Oth,"PIs")

Prim <- aea_sm %>%
  dplyr::rename(PI = Primary.Investigator) %>%
  select(PI)

Oth <- Oth_long %>%
  dplyr::rename(PI = PIs) %>%
  select(PI)

fin <- rbind(Prim,Oth)

fin$PI <- str_trim(fin$PI)
fin <- filter(fin, PI != "")

fin$PI <- gsub("\\s*\\([^\\)]+\\)","",as.character(fin$PI))

fin$PI <- rm_email(fin$PI)
fin$PI <- str_trim(fin$PI)
```

```r
fin$PI <- sub("^(\\S*\\s+\\S+).*", "\\1", fin$PI)

fin$PI <- tolower(fin$PI)
fin <- distinct(fin,PI)
fin <- fin[order(fin$PI),]

print(paste0("The number of active registered users is: ",length(fin)))
```

```
[1] "The number of active registered users is: 3660"
```

```r
num_activeusers <- length(fin)

update_latexnums("activeusers",num_activeusers)
```

```
Updating existing field activeusers
```

In the past year, there were 3660 researchers associated with actively updated registrations.

```r
# write out all the latex numbers

source(file.path(basedir,"Scripts","99_write_nums.R"))
```