# Applied Economics Research using R: Session 1

Seunghyun Lee (arslee@ucdavis.edu)

10/4/2021

## Contents

# 1 Outline of R scripts

## 1.1 Create setup files

```r
rscript_to_create <- c(
  "Code/000_master.R",
  "Code/001_packages.R",
  "Code/002_functions.R",
  "Code/003_parameters_public.R",
  "Code/004_parameters_private.R"
)

lapply(rscript_to_create, file.create)
```

## 1.2 master.R

```r
# clear
rm(list = ls())

# set directory
if (Sys.info()[["nodename"]] == "DESKTOP-8FJP3KC") {
  setwd("C:/Users/Seunghyun Lee/Dropbox/Teaching/ARE231/Rsession1/")
}

# setup files
source("Code/001_packages.R")
source("Code/002_functions.R")
source("Code/003_parameters_public.R")
source("Code/004_parameters_private.R")

# codes
source("Code/100_download and clean yield and acreage.R")
source("Code/110_construct annual county-level weather.R")
source("Code/120_exploratory data visualization.R")
source("Code/130_analysis.R")
```

### 1.3 packages.R

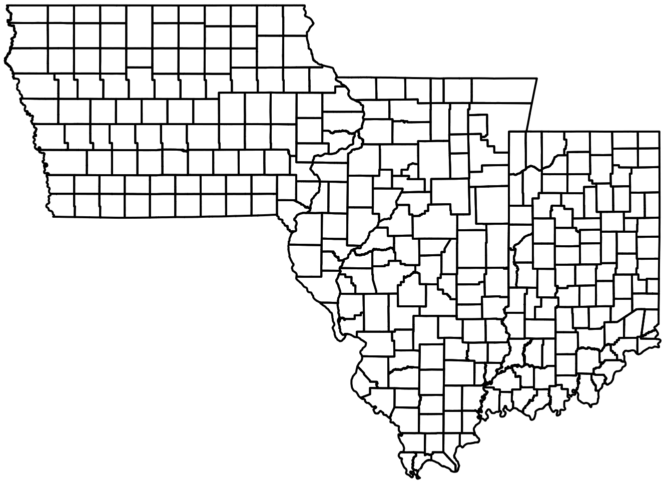
```r
library(pacman)
p_load(rnassqs, tidyverse, tigris, naniar, fixest, broom)
```

### 1.4 functions.R

```r
cb <- tigris::counties()
sb <- tigris::states()
```

I draw maps a lot. County boundaries and state boundaries are always in my `functions.R`.

```r
sf_3I <- cb %>% filter(STATEFP %in% 17:19)
plot(sf_3I[,"geometry"])
```



### 1.5 parameters_public.R

```
par <- list()

par$crops <- c("CORN", "SOYBEANS")
par$years <- 1981:2019
par$states <- c("IA", "IL", "IN")
par$fips <- cb %>%
  filter(STATEFP %in% 17:19) %>%
  pull(GEOID)
par$gs <- 4:9
```

`004_parameters_private.R` includes nothing but `par$apikey <- "my NASS api key"`

# 2 Crop yield model

## 2.1 To do list

1. Download annual county-level yield and acreage data for corn (for grain) and soybeans for all counties in the I states from the USDA NASS quickstats for the period 1981-2019. (Note: There are multiple ways of doing this. I would like your workflow to be reproducible and automated, meaning that

executing your script downloads all data you need at once. Please feel free to refer to Accessing Ag Data Using R)

2. Download weather data by clicking `Download All Monthly Data` from US County Weather once the app is fully loaded. (In a few weeks later, we are going to learn how to construct this data using the gridded daily temperature data from the PRISM Climate Group, county boundaries, and crop frequency map)

3. Using the data you downloaded, create a crop-county-year level panel dataset that contains columns of yield and weather variables necessary for your regressions. This will involve some data cleaning. (Tip: You can construct annual *gdd* or *hdd* by summing monthly degree days over the growing season (April to September))

4. Before running regressions, do some data exploration to check your data. You can compare your data with US Crops.

5. Run regressions for the following combinations.

   - crop : 1) corn, 2) soybeans
   - period : 1) full (1981-2019), 2) pre 2000 (1981-2000), 3) post 2000 (2001-2019),
   - regression weights : 1) no weight, 2) acreage
   - cluster standard errors by: 1) year, 2) state
   - time trend : 1) county-specific linear, 2) state-specific linear, 3) county-specific quadratic, 4) state-specific quadratic
   - weather variables : 1) (with precipitation) $prec$, $prec^2$, $gdd$ and $hdd$, 2) (without precipitation) $gdd$ and $hdd$

That is, you have $2 \times 3 \times 2 \times 2 \times 4 \times 2 = 192$ regression results.

6. Let's focus on $\beta_4$.

   - Obtain $\hat{\beta}_4$s and calculate their confidence intervals.
   - Compare $\hat{\beta}_4$s and their confidence intervals by changing one dimension at a time. (i.e., compare $\hat{\beta}_4$s for different crops 1) corn and 2) soybeans while keeping 1) for all the other dimensions).
   - Do you find any systematic differences between models in any dimensions? If so, what are they? Are they consistent with your intuition?

## 2.2 Cleaned Data

```
df_weather <- readRDS("Data/Processed/df_weather.rds")
head(df_weather)
```

```
## # A tibble: 6 x 6
## # Groups:   fips [1]
##    fips  year  prec  precsq   gdd   hdd
##   <int> <int> <dbl>   <dbl> <dbl> <dbl>
## 1 17001  1981  982. 964769. 1859. 11.1
## 2 17001  1982  675. 455230. 1772.  9.64
## 3 17001  1983  475. 225613. 1917. 94.2
## 4 17001  1984  554. 307120. 1789. 25.3
## 5 17001  1985  556. 308787. 1868. 12.5
## 6 17001  1986  726. 527551. 2013. 17.2
```

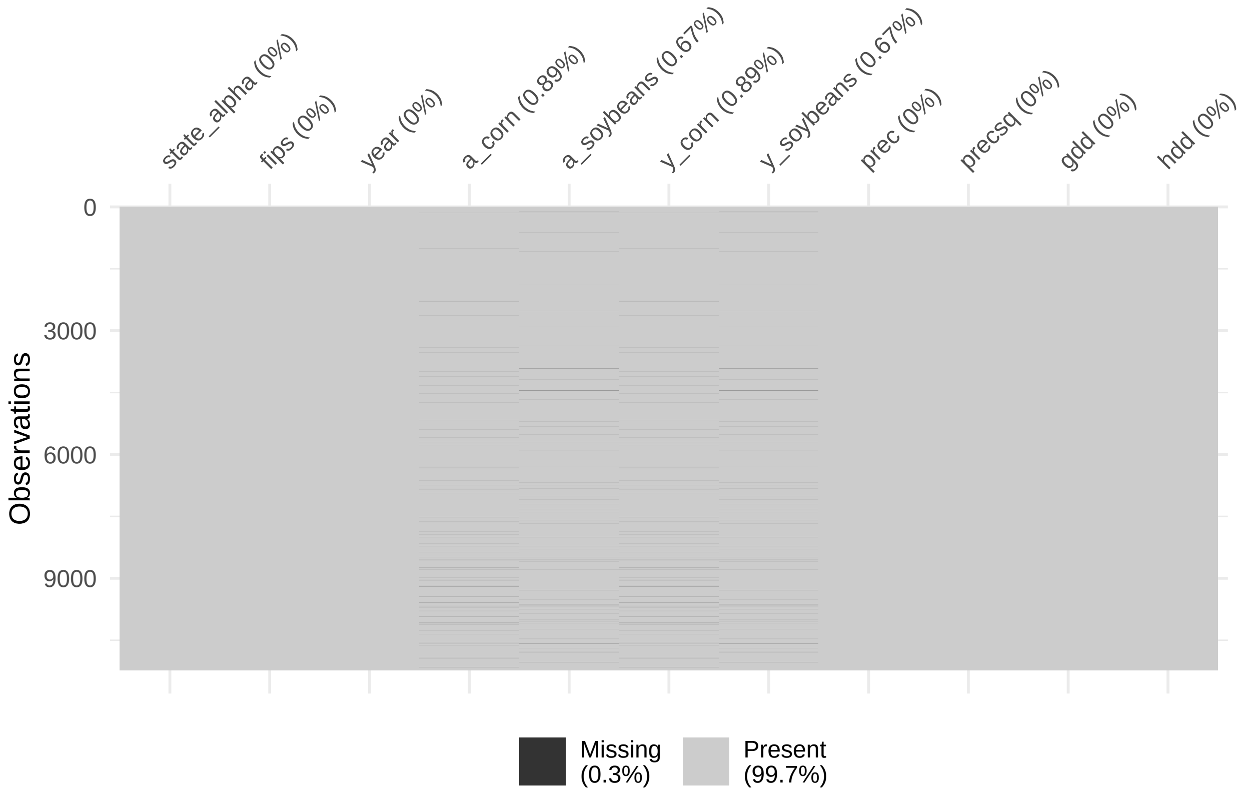
```r
df_ya <- readRDS("Data/Processed/df_ya.rds")
head(df_ya)
```

```
##   state_alpha  fips year a_corn a_soybeans y_corn y_soybeans
## 1          IA 19001 1981 120100      61400  121.0       43.0
## 2          IA 19001 1982 114900      61900  102.2       32.7
## 3          IA 19001 1983  73200      65200   77.3       35.1
## 4          IA 19001 1984 113500      62900   97.4       25.4
## 5          IA 19001 1985 111800      64000  127.9       39.5
## 6          IA 19001 1986 101900      69200  128.3       42.0
```

```r
df <- left_join(df_ya, df_weather, by = c("fips", "year"))
head(df)
```

```
##   state_alpha  fips year a_corn a_soybeans y_corn y_soybeans     prec   precsq
## 1          IA 19001 1981 120100      61400  121.0       43.0 614.4529 377552.4
## 2          IA 19001 1982 114900      61900  102.2       32.7 786.3799 618393.4
## 3          IA 19001 1983  73200      65200   77.3       35.1 504.3404 254359.3
## 4          IA 19001 1984 113500      62900   97.4       25.4 719.3647 517485.6
## 5          IA 19001 1985 111800      64000  127.9       39.5 536.9831 288350.8
## 6          IA 19001 1986 101900      69200  128.3       42.0 953.9710 910060.8
##        gdd      hdd
## 1 1722.762 12.43103
## 2 1625.951 12.68410
## 3 1813.734 58.95062
## 4 1660.162 29.37303
## 5 1708.155 16.59215
## 6 1825.161 12.66629
```

```r
vis_miss(df)
```

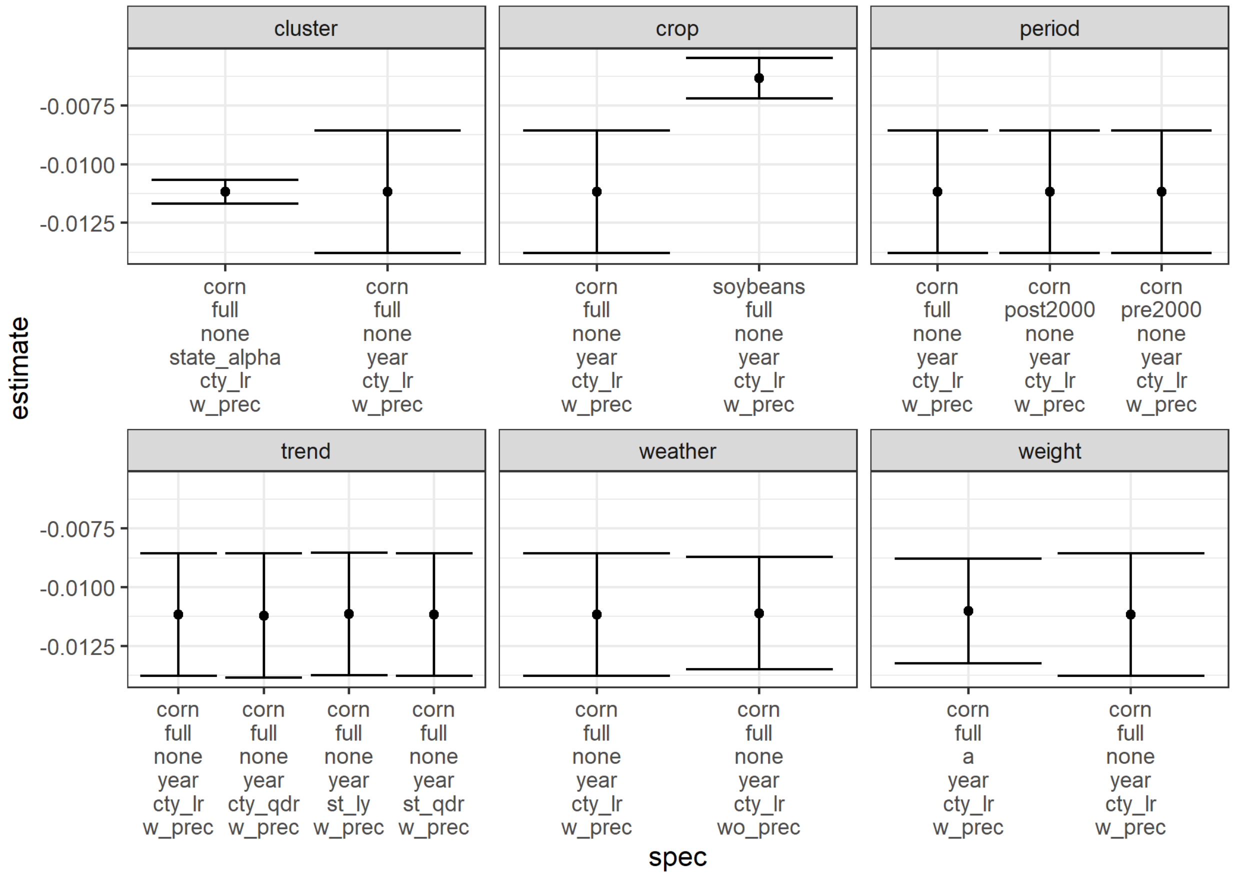


```
summary(df)
```

```
##  state_alpha             fips            year          a_corn
##  Length:11230       Min.   :17001   Min.   :1981   Min.   :  1000
##  Class :character   1st Qu.:17147   1st Qu.:1990   1st Qu.: 56300
##  Mode  :character   Median :18091   Median :2000   Median : 91100
##                     Mean   :18093   Mean   :2000   Mean   :100786
##                     3rd Qu.:19053   3rd Qu.:2009   3rd Qu.:138900
##                     Max.   :19197   Max.   :2019   Max.   :394000
##                                                    NA’s   :100
##    a_soybeans         y_corn        y_soybeans        prec
##  Min.   :   700   Min.   : 19.0   Min.   :13.0   Min.   : 211.8
##  1st Qu.: 48800   1st Qu.:118.5   1st Qu.:37.1   1st Qu.: 543.4
##  Median : 77700   Median :140.0   Median :43.6   Median : 633.4
##  Mean   : 82663   Mean   :140.3   Mean   :43.5   Mean   : 639.6
##  3rd Qu.:108950   3rd Qu.:164.9   3rd Qu.:50.0   3rd Qu.: 727.3
##  Max.   :328400   Max.   :246.7   Max.   :80.4   Max.   :1276.5
##  NA’s   :75       NA’s   :100     NA’s   :75
##      precsq            gdd            hdd
##  Min.   :  44850   Min.   :1074   Min.   :  0.07818
##  1st Qu.: 295247   1st Qu.:1590   1st Qu.:  6.71760
##  Median : 401220   Median :1734   Median : 13.53991
##  Mean   : 429281   Mean   :1745   Mean   : 18.86617
##  3rd Qu.: 528924   3rd Qu.:1886   3rd Qu.: 24.33231
##  Max.   :1629447   Max.   :2485   Max.   :128.11010
##
```

## 2.3 Result

**$\hat{\beta}s$ and their CIs**



## 2.4 Code

### 2.4.1 100_download and clean yield and acreage.R

```
#---------------[   Purpose    ]--------------------
#
# To download annual county-level acreage and yield data for 3 I states from 1981 to 2019
#
#---------------[   Sys Info   ]--------------------
#
#  Date   : Fri Oct 01 21:17:19 2021
#  Author: Seunghyun Lee
#  OS     : Windows
#  Node   : DESKTOP-8FJP3KC
#
#---------------[ Pinned Notes ]--------------------
#
#
#
```

```
#---------------[   Process    ]--------------------



# Test --------------------------------------------------------------------

nassqs_auth(key = par$apikey)

params <- list(
  commodity_desc = "CORN",
  source_desc = "SURVEY",
  agg_level_desc = "COUNTY",
  state_alpha = par$states[1],
  year = 2012,
  statisticcat_desc = "YIELD"
)

df <- nassqs(params)

class(df)
sapply(df, class)
head(df)



# 1. Loop --------------------------------------------------------------------

## grid  -------------------------------------
grid <- expand_grid(
  cr = par$crops,
  yr = par$years,
  st = par$states,
  var = c(
    "CORN, GRAIN - YIELD, MEASURED IN BU / ACRE",
    "SOYBEANS - YIELD, MEASURED IN BU / ACRE",
    #----@ note: use harvested acres  @----
    "CORN, GRAIN - ACRES HARVESTED",
    "SOYBEANS - ACRES HARVESTED"
  )
)

## function  -------------------------------------
extract_nass <- function(cr, yr, st, var) {
  library(rnassqs)
  nassqs(
    commity_desc = cr,
    source_desc = "SURVEY",
    agg_level_desc = "COUNTY",
    year = yr,
    state_alpha = st,
    short_desc = var
  )
}
```

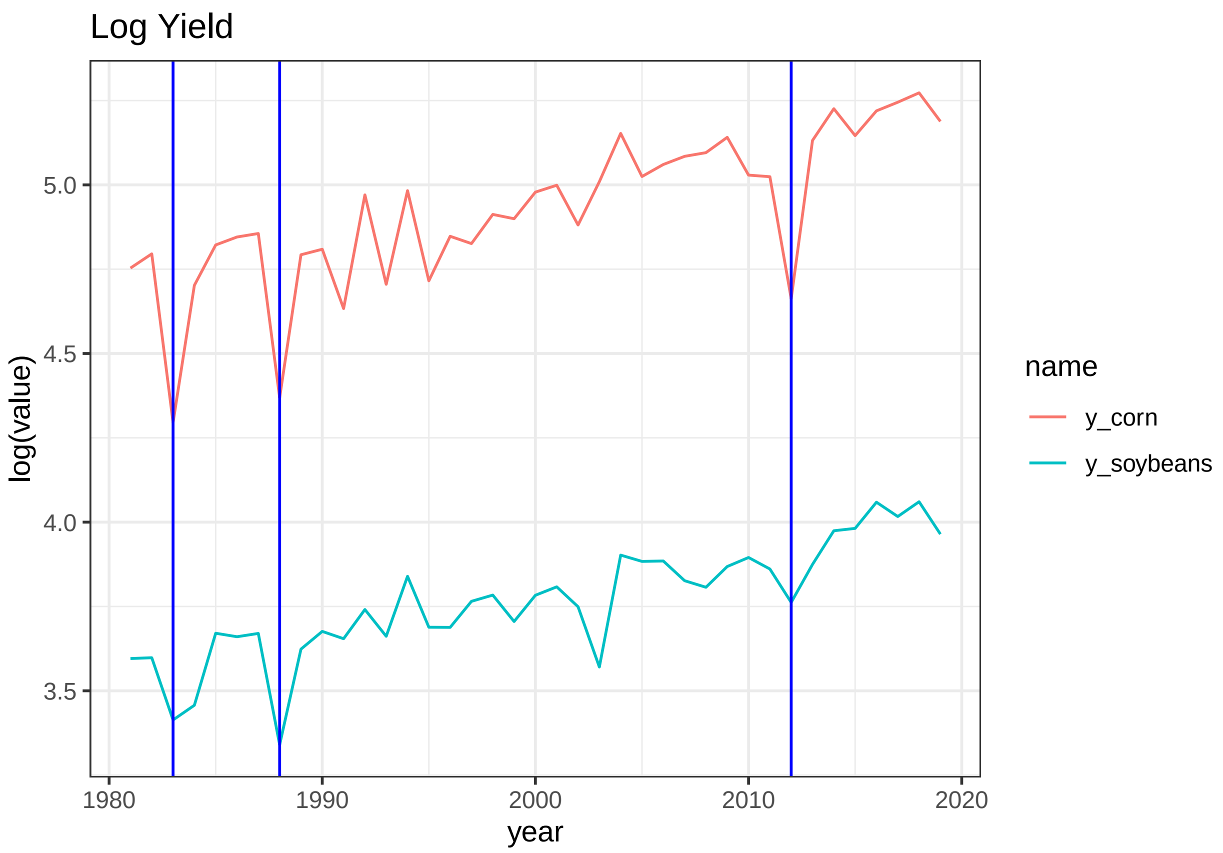
```r
## loop  -------------------------------------
df <- future_pmap(.progress = T, grid, extract_nass) %>% rbindlist(use.names = T)


# 2. clean ---------------------------------------------------------------------

df_ya <- df %>%
  filter(county_name != "OTHER (COMBINED) COUNTIES") %>%
  distinct() %>%
  mutate(
    Value = as.numeric(str_replace(Value, ",", "")),
    fips = as.integer(paste0(state_fips_code, county_code))
  ) %>%
  select(state_alpha, fips, year, short_desc, commodity_desc, statisticcat_desc, Value) %>%
  mutate(var = paste0(str_sub(statisticcat_desc, 1, 1), "_", commodity_desc) %>% tolower()) %>%
  select(state_alpha, fips, year, var, Value) %>%
  spread(var, Value)



vis_miss(df_ya)
summary(df_ya)


# 3. export ----------------------------------------------------------------------
#----@ output: Data/df_ya.rds @----
saveRDS(df_ya, "Data/df_ya.rds")
```

### 2.4.2 110_construct annual county-level weather.R

```r
#---------------[   Purpose    ]--------------------
#
# To construct annual county-level weather variables
#
#---------------[   Sys Info   ]--------------------
#
#  Date  : Fri Oct 01 23:26:43 2021
#  Author: Seunghyun Lee
#  OS    : Windows
#  Node  : DESKTOP-8FJP3KC
#
#---------------[ Pinned Notes ]--------------------
#
#
#
#---------------[   Process    ]--------------------


#----@ input: Raw/weather_monthly.csv @----
df <- fread("Data/Raw/weather_monthly.csv")
df_weather <- df %>%
  filter(year %in% par$years & state %in% par$states & month %in% par$gs) %>%
  select(fips, year, month, prec, dday10C, dday30C) %>%
  group_by(fips, year) %>%
  summarize_at(c("prec", "dday10C", "dday30C"), sum, na.rm = T) %>%
  mutate(
    precsq = prec^2,
    gdd = dday10C - dday30C,
    hdd = dday30C
  ) %>%
  select(!c(dday10C, dday30C))

#----@ output: Data/Processed/df_weather.rds @----
saveRDS(df_weather, "Data/Processed/df_weather.rds")
```

### 2.4.3 120_exploratory data visualization.R

```
#---------------[   Purpose    ]--------------------
#
# Exploratory data analysis only for yield
#
#---------------[   Sys Info   ]--------------------
#
#  Date  : Fri Oct 01 23:39:46 2021
#  Author: Seunghyun Lee
#  OS    : Windows
#  Node  : DESKTOP-8FJP3KC
#
#---------------[ Pinned Notes ]--------------------
#
#
#
#---------------[   Process    ]--------------------
df_ya <- readRDS("Data/Processed/df_ya.rds")

df_plot <- df_ya %>%
  select(year, contains("y_")) %>%
  pivot_longer(!year) %>%
  group_by(year, name) %>%
  summarise(value = mean(value, na.rm = T)) %>%
  ungroup()

df_plot %>%
  ggplot(aes(x = year, y = log(value), color = name)) +
  geom_line() +
  theme_minimal() +
  geom_vline(xintercept = c(1983, 1988, 2012), color = "blue") +
  theme_bw() +
  ggtitle("Log Yield")
```



### 2.4.4 130_analysis.R

```
#----------------[   Purpose    ]--------------------
#
# Run regressions and compare coef and confidence intervals depending on specifications
#
#----------------[   Sys Info   ]--------------------
#
#  Date   : Fri Oct 01 22:40:49 2021
#  Author: Seunghyun Lee
#  OS     : Windows
#  Node   : DESKTOP-8FJP3KC
#
#----------------[ Pinned Notes ]--------------------
#
#
#
#----------------[   Process    ]--------------------


# prep data --------------------------------------------------------------

df_weather <- readRDS("Data/Processed/df_weather.rds")
df_ya <- readRDS("Data/Processed/df_ya.rds")
```

```
df <- left_join(df_ya, df_weather, by = c("fips", "year"))



# primitive lists for grid ---------------------------------------------------------

period_list <- list(
  full = 1981:2019,
  pre2000 = 1981:2000,
  post2000 = 2001:2019
)

trend_list <- list(
  cty_lr = "year:factor(fips)",
  st_ly = "year:factor(state_alpha)",
  cty_qdr = "year:factor(fips)+year^2:factor(fips)",
  st_qdr = "year:factor(state_alpha)+year^2:factor(state_alpha)"
)

weather_list <- list(
  w_prec = "prec+precsq+gdd+hdd",
  wo_prec = "gdd+hdd"
)


# grid for regression -----------------------------------------------------

grid <- expand_grid(
  crop = tolower(par$crops),
  period = names(period_list),
  weight = c("none", "a"),
  cluster = c("year", "state_alpha"),
  trend = names(trend_list),
  weather = names(weather_list)
) %>%
  mutate(id = 1:n())


#  regression function ----------------------------------------------------

reg <- function(crop, period, weight, cluster, trend, weather, id) {
  print(y)
  print(id)

  #--- regression equation ---#
  y <- paste0("log(", "y_", crop, ")")
  fml <- paste0(y, "~", weather_list[[weather]], "+", trend_list[[trend]], "|fips") %>% formula()


  #--- run regressions ---#
  if (weight != "none") {
    w <- paste0(weight, "_", crop)
    output <- feols(fml, data = df, weights = df %>% pull(w), cluster = cluster)
  } else {
    output <- feols(fml, data = df, cluster = cluster)
  }

  #--- extract results of interest ---#
  output %>%
    tidy() %>%
    filter(term == "hdd")
}



# run regression and store results ----------------------------------------

df_result <- grid %>%
  mutate(result = pmap(., reg)) %>%
  unnest(result) %>%
  mutate(low = estimate - 1.96 * std.error, high = estimate + 1.96 * std.error)




# prep data for plot ------------------------------------------------------

columns <- names(grid)[names(grid) != "id"]
df_plot <- lapply(1:length(columns), function(col) {
  df_result %>%
    group_by(across(columns[col])) %>%
    slice(1) %>%
    # extract only first rows by the group
    mutate(dim = columns[col])
}) %>% bind_rows()


# plot --------------------------------------------------------------------

df_plot %>%
  mutate(spec = paste(crop, period, weight, cluster, trend, weather, sep = "\n")) %>%
  ggplot(aes(spec, estimate)) +
  geom_point() +
  geom_errorbar(aes(ymin = low, ymax = high)) +
  facet_wrap(~dim, scales = "free_x", ) +
  theme_bw()

ggsave("Figure/result.png", height = 5, width = 7)
```