6-6 DiD and Synthetic Control

April 03, 2024

```r
# Install packages
if (!require("pacman")) install.packages("pacman")
```

```
## Loading required package: pacman
```

```r
# We are using a package (augsynth) that is not on CRAN, R packages on CRAN have to pass
# some formal tests. Always proceed with caution if a packages is not on CRAN. Since the
# R package is not on CRAN, we needed to download and install the package directly from
# GitHub. Always use the CRAN version if there is one because it is most stable. However,
# if you need something that is currently in development, you might want to download from
# GitHub. I've commented out the workflow since I already have it on my computer:

#
# workflow to install a package from GitHub
# ----------------------------------------------------------------


# 1. install `devtools` if you don't already have it. Note that you might need to update the 'rlang' pa
# ----------
#install.packages("devtools") # download developer tools package
#library(devtools)            # load library

# 2. install the package ("augsynth"). you can find this path on the GitHub instructions
# ----------
#devtools::install_github("ebenmichael/augsynth")


# install libraries - install "augsynth" here since it is now on CRAN
pacman::p_load(# Tidyverse packages including dplyr and ggplot2
               tidyverse,
               ggthemes,
               augsynth)

#
# chunk options
# ----------------------------------------------------------------
knitr::opts_chunk$set(
  warning = FALSE            # prevents warning from appearing after code chunk
)


# set seed
set.seed(44)
```

Introduction

In this lab we will explore difference-in-differences estimates and a newer extension, synthetic control. The basic idea behind both of these methods is simple - assuming two units are similar in a pre-treatment period and one undergoes treatment while the other stays in control, we can estimate a causal effect by taking three differences. First we take the difference between the two in the pre-treatment period, then take another difference in the post-treatment period. Then we take a difference between these two differences (hence the name difference in differences). Let's see how this works in practice!

Basic DiD

We'll use the kansas dataset that comes from the **augsynth** package. Our goal here is to estimate the effect of the 2012 Kansas tax cuts on state GDP. Let's take a look at our dataset:

```r
# load data
data(kansas)

# summary statistics of kansas
summary(kansas)
```

```
##       fips            year           qtr           state          
##  Min.   : 1.00   Min.   :1990   Min.   :1.000   Length:5250       
##  1st Qu.:17.00   1st Qu.:1996   1st Qu.:1.000   Class :character  
##  Median :29.50   Median :2003   Median :2.000   Mode  :character  
##  Mean   :29.32   Mean   :2003   Mean   :2.486                     
##  3rd Qu.:42.00   3rd Qu.:2009   3rd Qu.:3.000                     
##  Max.   :56.00   Max.   :2016   Max.   :4.000                     
##                                                                   
##       gdp            revenuepop    rev_state_total  rev_local_total 
##  Min.   :  11509   Min.   : 1335   Min.   :  1668   Min.   :   550  
##  1st Qu.:  55151   1st Qu.: 3057   1st Qu.:  7026   1st Qu.:  3268  
##  Median : 130650   Median : 3628   Median : 13868   Median : 10041  
##  Mean   : 228237   Mean   : 3851   Mean   : 20813   Mean   : 17197  
##  3rd Qu.: 276303   3rd Qu.: 4365   3rd Qu.: 24405   3rd Qu.: 18774  
##  Max.   :2568986   Max.   :14609   Max.   :182530   Max.   :143137  
##                    NA's   :2250    NA's   :2850     NA's   :2850    
##   popestimate       qtrly_estabs_count month1_emplvl      month2_emplvl     
##  Min.   :  453690   Min.   :  15133    Min.   :  178737   Min.   :  178587  
##  1st Qu.: 1652585   1st Qu.:  48170    1st Qu.:  657056   1st Qu.:  663786  
##  Median : 3997978   Median : 108822    Median : 1675988   Median : 1684341  
##  Mean   : 5767107   Mean   : 161021    Mean   : 2482331   Mean   : 2494933  
##  3rd Qu.: 6611215   3rd Qu.: 188730    3rd Qu.: 2990530   3rd Qu.: 2993158  
##  Max.   :39250017   Max.   :1448488    Max.   :16600851   Max.   :16633834  
##                                                                             
##  month3_emplvl      total_qtrly_wages   taxable_qtrly_wages avg_wkly_wage   
##  Min.   :  181521   Min.   :8.811e+08   Min.   :0.000e+00   Min.   : 301.0  
##  1st Qu.:  667492   1st Qu.:5.403e+09   1st Qu.:0.000e+00   1st Qu.: 515.2  
##  Median : 1699044   Median :1.362e+10   Median :1.096e+09   Median : 658.0  
##  Mean   : 2510204   Mean   :2.402e+10   Mean   :3.776e+09   Mean   : 674.8  
##  3rd Qu.: 3016494   3rd Qu.:2.973e+10   3rd Qu.:4.177e+09   3rd Qu.: 804.0  
##  Max.   :16606038   Max.   :2.753e+11   Max.   :7.689e+10   Max.   :1792.0  
##                                                                             
##     year_qtr       treated             gdpcapita          lngdp       
##  Min.   :1990   Min.   :0.000000   Min.   :15029   Min.   : 9.351  
```

```
##  1st Qu.:1996   1st Qu.:0.000000   1st Qu.:27989   1st Qu.:10.918
##  Median :2003   Median :0.000000   Median :36449   Median :11.780
##  Mean   :2003   Mean   :0.003048   Mean   :37808   Mean   :11.754
##  3rd Qu.:2010   3rd Qu.:0.000000   3rd Qu.:45531   3rd Qu.:12.529
##  Max.   :2016   Max.   :1.000000   Max.   :84382   Max.   :14.759
##
##   lngdpcapita     revstatecapita  revlocalcapita   emplvl1capita
##  Min.   : 9.618   Min.   : 2021   Min.   : 883.6   Min.   :0.3249
##  1st Qu.:10.240   1st Qu.: 2903   1st Qu.:2012.4   1st Qu.:0.4113
##  Median :10.504   Median : 3380   Median :2428.3   Median :0.4356
##  Mean   :10.486   Mean   : 3742   Mean   :2480.2   Mean   :0.4368
##  3rd Qu.:10.726   3rd Qu.: 4048   3rd Qu.:2819.4   3rd Qu.:0.4621
##  Max.   :11.343   Max.   :20353   Max.   :7160.9   Max.   :1.0524
##                   NA's   :2850    NA's   :2850
##  emplvl2capita    emplvl3capita     emplvlcapita    totalwagescapita
##  Min.   :0.3251   Min.   :0.3289   Min.   :0.3269   Min.   : 1493
##  1st Qu.:0.4138   1st Qu.:0.4163   1st Qu.:0.4138   1st Qu.: 2941
##  Median :0.4378   Median :0.4406   Median :0.4378   Median : 3787
##  Mean   :0.4390   Mean   :0.4420   Mean   :0.4393   Mean   : 3869
##  3rd Qu.:0.4644   3rd Qu.:0.4676   3rd Qu.:0.4644   3rd Qu.: 4608
##  Max.   :1.0507   Max.   :1.0513   Max.   :1.0515   Max.   :10275
##
##  taxwagescapita   avgwklywagecapita  estabscapita         abb
##  Min.   :   0.0   Min.   : 301.0    Min.   :0.01992   Length:5250
##  1st Qu.:   0.0   1st Qu.: 515.2    1st Qu.:0.02553   Class :character
##  Median : 355.7   Median : 658.0    Median :0.02845   Mode  :character
##  Mean   : 728.8   Mean   : 674.8    Mean   :0.02928
##  3rd Qu.:1224.4   3rd Qu.: 804.0    3rd Qu.:0.03211
##  Max.   :5254.4   Max.   :1792.0    Max.   :0.07071
##
```

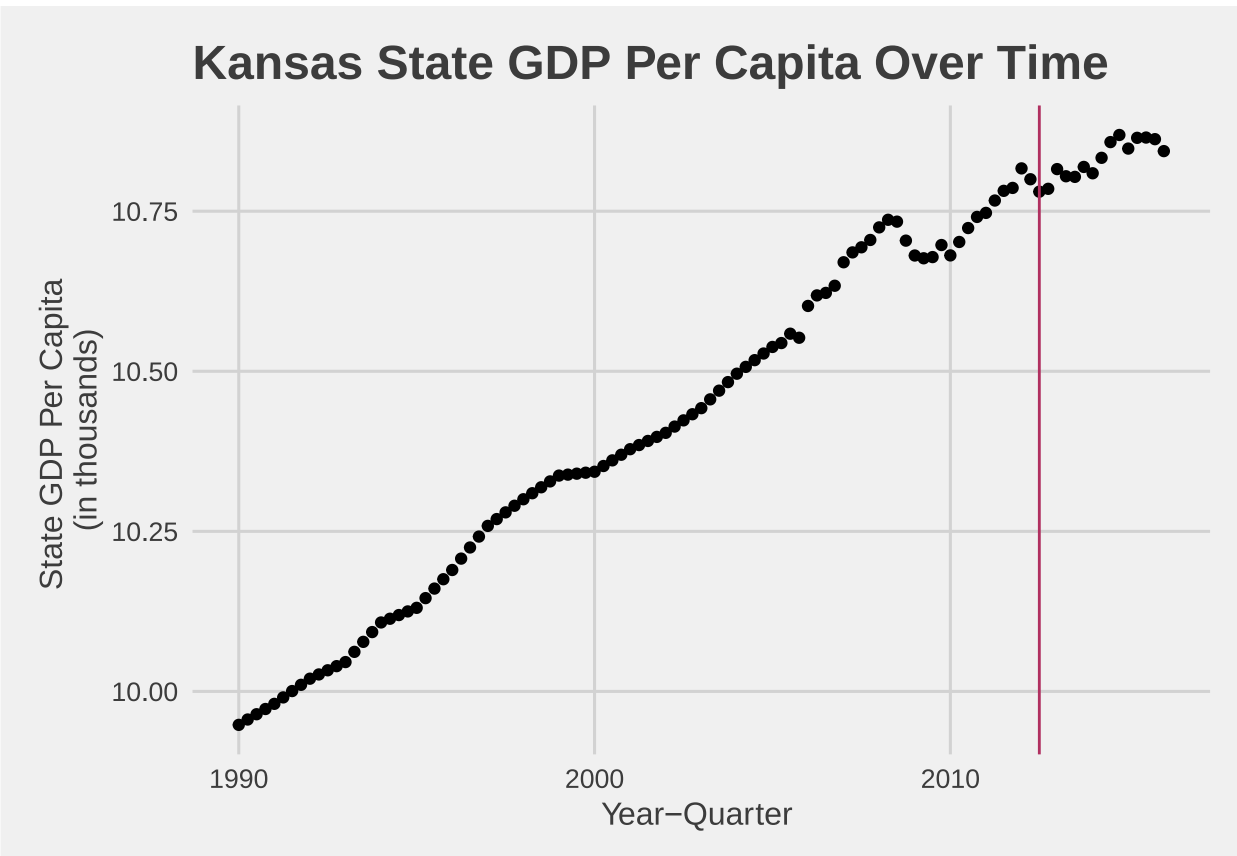
We have a lot of information here! We have quarterly state GDP from 1990 to 2016 for each U.S. state, as well as some other covariates. Let's begin by adding a treatment indicator to Kansas in Q2 2012 and onward.

```r
# create a treatment indicator
# ----------
kansas <-
  kansas %>%
  # select subset of variables
  select(year, qtr, year_qtr, state, treated, gdp, lngdpcapita, fips) %>%
  # create new treatment flag just to see
  mutate(treatment = case_when(state == "Kansas" & year_qtr >= 2012.5 ~ 1, # note this adds treatment i
                               TRUE ~ 0))
# view head
head(kansas)
```

```
## # A tibble: 6 x 9
##    year   qtr year_qtr state   treated    gdp lngdpcapita  fips treatment
##   <dbl> <dbl>    <dbl> <chr>     <dbl>  <dbl>       <dbl> <dbl>     <dbl>
## 1  1990     1    1990  Alabama       0 71610         9.78     1         0
## 2  1990     2    1990. Alabama       0 72718.        9.79     1         0
## 3  1990     3    1990. Alabama       0 73826.        9.80     1         0
## 4  1990     4    1991. Alabama       0 74935.        9.82     1         0
## 5  1991     1    1991  Alabama       0 76043         9.83     1         0
## 6  1991     2    1991. Alabama       0 77347.        9.84     1         0
```

One approach might be to compare Kansas to itself pre- and post-treatment. If we plot state GDP over time we get something like this:

```r
# visualize intervention in Kansas
# ----------
kansas %>%

  # processing
  # ----------
  filter(state == 'Kansas') %>%

  # ggplot
  # ----------
  ggplot() +
    # geometries
    geom_point(aes(x = year_qtr, y = lngdpcapita)) +
    geom_vline(xintercept = 2012.5, color = "maroon") + # color horizontal line red

    # themes
    theme_fivethirtyeight() +
    theme(axis.title = element_text()) +

    # labels
    labs(x = "Year-Quarter ",                                # x-axis label
           y = "State GDP Per Capita \n(in thousands)",      # y-axis label
         title = "Kansas State GDP Per Capita Over Time") # title
```

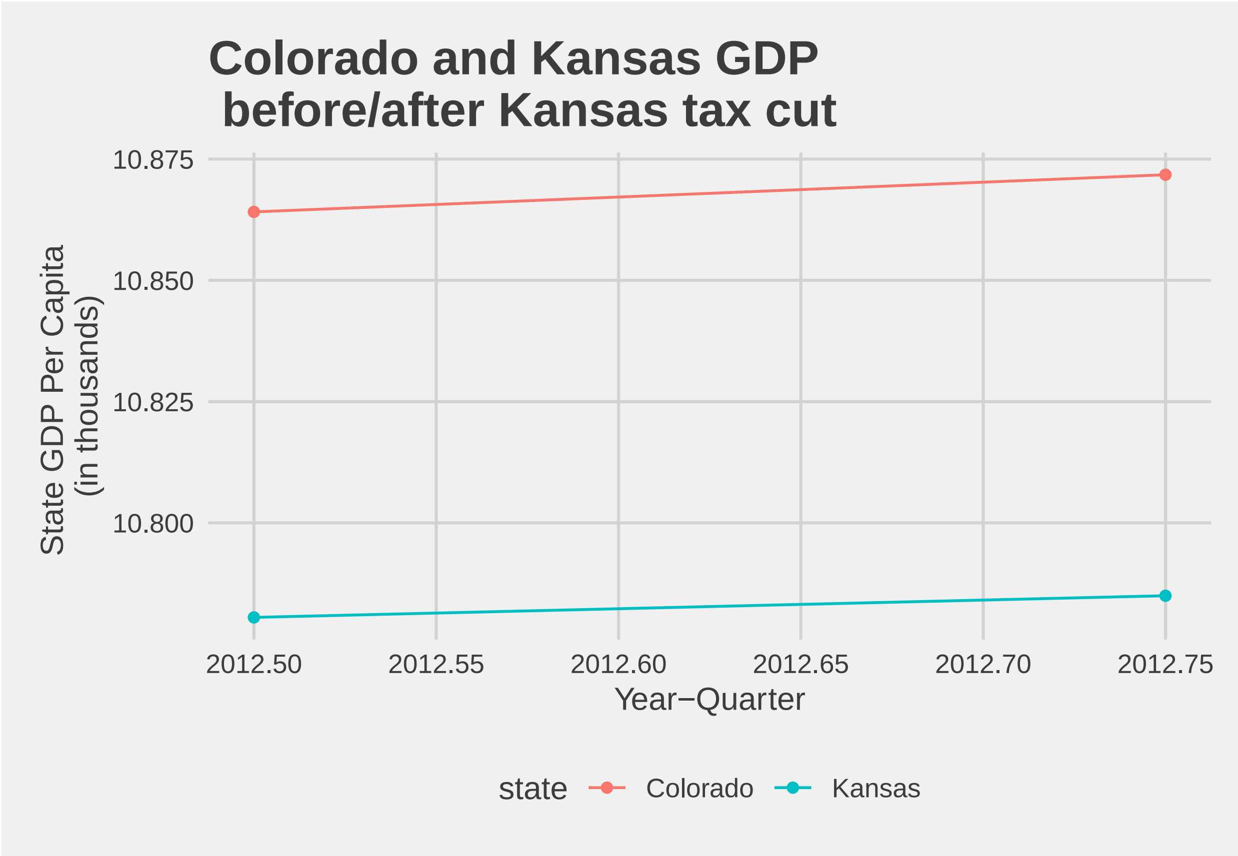


QUESTION: Looks like GDP went up after the tax cut! What is the problem with this inference?

ANSWER: It looks like GDP went up after the tax cut, but we have no way of telling whether it went up because of the tax cut or went up because it would have otherwise. In short, we need to compare the treated Kansas to a counterfactual for if taxes weren't cut.

Ideally, we would like to compare treated Kansas to control Kansas. Because of the fundamental problem of causal inference, we will never observe both of these conditions though. The core idea behind DiD is that we could instead use the fact that our treated unit was similar to a control unit, and then measure the differences between them. Perhaps we could choose neighboring Colorado:

```
# visualize intervention in Kansas
# ----------
kansas %>%
  # processing
  # ----------
  filter(state %in% c("Kansas","Colorado")) %>%  # use "%in% to filter values in a vector
  filter(year_qtr >= 2012.5 & year_qtr<= 2012.75) %>%
  #filter(between(year_qtr, 2012.5, 2012.75)) %>%   # same filtering but using between() instead which

  # plot
  # ----------
  ggplot() +
  # add in point layer
  geom_point(aes(x = year_qtr,
                 y = lngdpcapita,
                 color = state)) + # color by state
  # add in line
  geom_line(aes(x = year_qtr,
                y = lngdpcapita,
                color = state)) +

  # themes
  theme_fivethirtyeight() +
  theme(axis.title = element_text()) +

  # labels - PREFER TO USE labs() SO THAT IT IS ALL IN ONE ARGUMENT
  ggtitle('Colorado and Kansas GDP \n before/after Kansas tax cut') +
  xlab('Year-Quarter') +
  ylab('State GDP Per Capita \n(in thousands)')
```



This is basically what Card-Krueger (1994) did measuring unemployment rates among New Jersey and Pennsylvania fast food restaurants.

Challenge: Try writing a simple DiD estimate using `dplyr/tidyr` (use subtraction instead of a regression):

```
#
# DiD for: kansas-colorado
# ----------------------------------------------------------------
# create a dataset for kansas and colorado
kc <-
  kansas %>%
  filter(state %in% c("Kansas","Colorado")) %>%
  filter(year_qtr >= 2012.5 & year_qtr<= 2012.75)

# pre-treatment difference
# ----------
pre_diff <-
  kc %>%
  # filter out only the quarter we want
  filter(year_qtr == 2012.5) %>%
  # subset to select only vars we want
  select(state,
         lngdpcapita) %>%
  # make the data wide
  pivot_wider(names_from = state,
              values_from = lngdpcapita) %>%
  # subtract to make calculation
```

```
  summarise(Colorado - Kansas)

# post-treatment difference
# ----------
post_diff <-
  kc %>%
  # filter out only the quarter we want
  filter(year_qtr == 2012.75) %>%
  # subset to select only vars we want
  select(state,
         lngdpcapita) %>%
  # make the data wide
  pivot_wider(names_from = state,
              values_from = lngdpcapita) %>%
  # subtract to make calculation
  summarise(Colorado - Kansas)

# diff-in-diffs
# ----------
diff_in_diffs <- post_diff - pre_diff
diff_in_diffs
```

```
##   Colorado - Kansas
## 1       0.003193447
```

Looks like our treatment effect is about .003 (in logged thousands dollars per capita). Again this is the basic idea behind Card-Krueger.

QUESTION: Why might there still be a problem with this estimate?

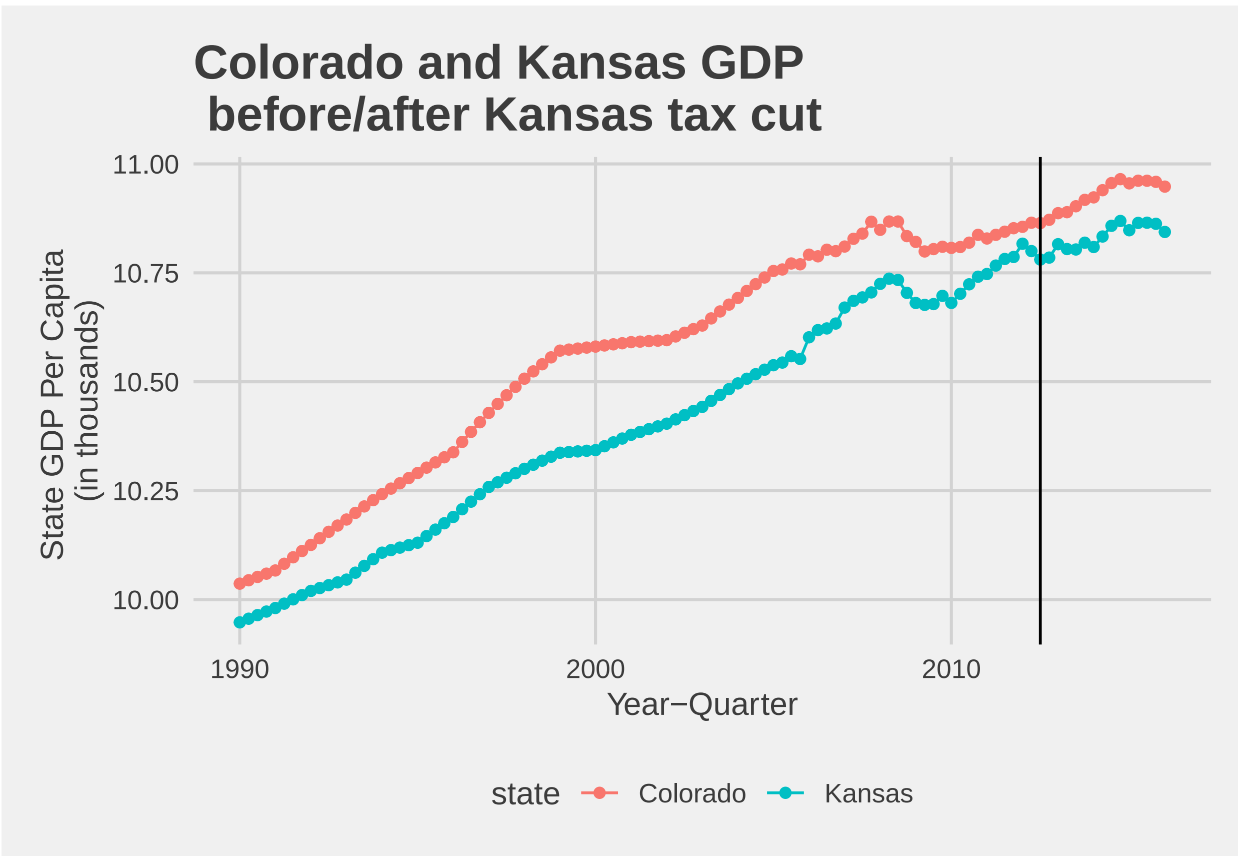
ANSWER: We just assumed that Colorado was similar to Kansas because they are neighbors - we don't really have evidence for this idea.

Parallel Trends Assumptions

One of the core assumptions for difference-in-differences estimation is the "parallel trends" or "constant trends" assumption. Essentially, this assumption requires that the difference between our treatment and control units are constant in the pre-treatment period. Let's see how Kansas and Colorado do on this assumption:

```
#
# parallel trends
# ----------------------------------------------------------------
kansas %>%

  # process
  # ---------
  filter(state %in% c("Kansas","Colorado")) %>%
  # plotting all of the time periods -- not filtering out any of them

  # plot
  # ---------
  ggplot() +
  # add in point layer
  geom_point(aes(x = year_qtr,
                 y = lngdpcapita,
```

```
                color = state)) +
  # add in line layer
  geom_line(aes(x = year_qtr,
               y = lngdpcapita,
               color = state)) +
  # add a horizontal line
  geom_vline(aes(xintercept = 2012.5)) +

  # themes
  theme_fivethirtyeight() +
  theme(axis.title = element_text()) +

  # labels
  ggtitle('Colorado and Kansas GDP \n before/after Kansas tax cut') +
  xlab('Year-Quarter') +
  ylab('State GDP Per Capita \n(in thousands)')
```

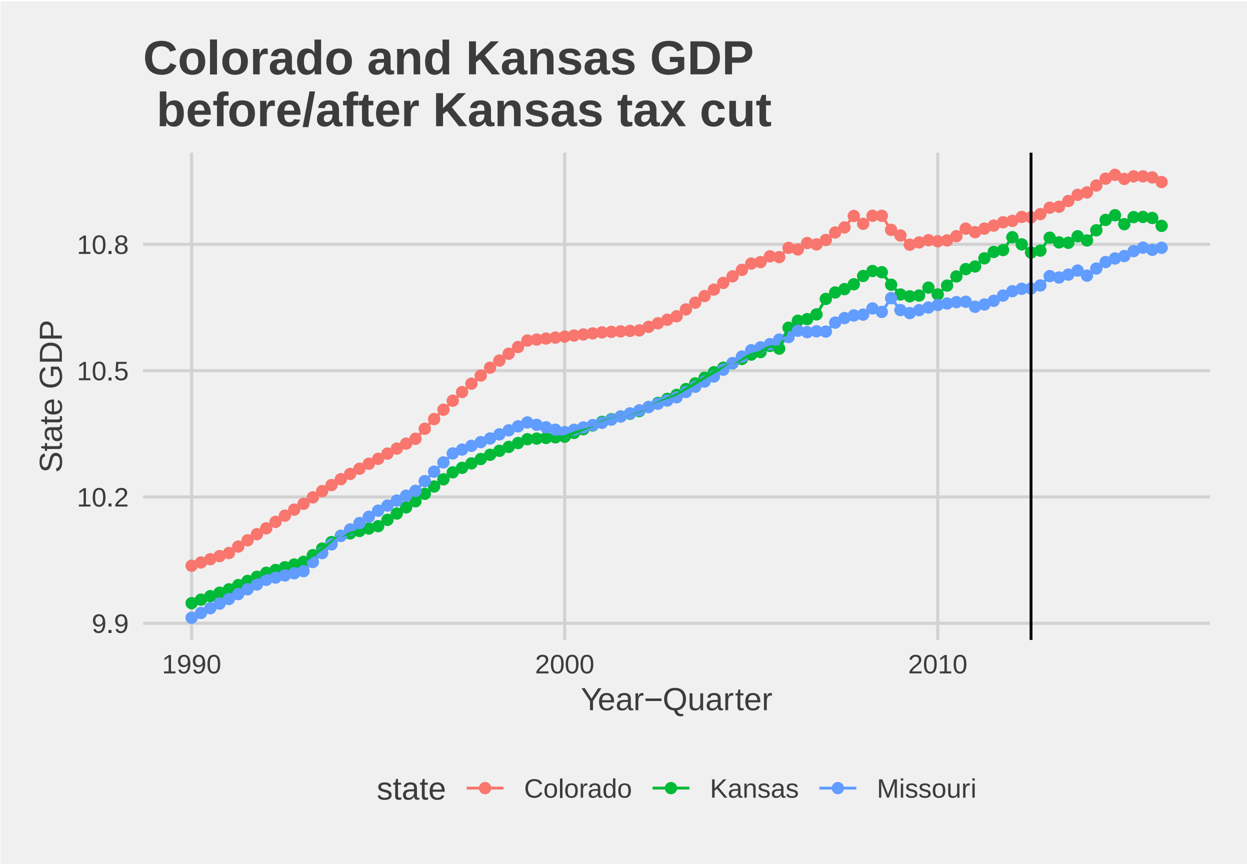


The two lines somewhat move together, but the gap does grow and shrink at various points over time. The most concerning part here is that the gap quickly shrinks right before treatment. What do we do if we do not trust the parallel trends assumption? Perhaps we pick a different state.

Challenge: Choose another state that you think would be good to try out, and plot it alongside Kansas and Colorado.

```
#
# parallel trends: add a third state
# ------------------------------------------------------------
```

```r
kansas %>%
  # process
  # ---------
  filter(state %in% c("Kansas",
                      "Colorado",
                      "Missouri")) %>%

  # plot
  # ---------
  ggplot() +
  geom_point(aes(x = year_qtr,
                 y = lngdpcapita,
                 color = state)) +
  geom_line(aes(x = year_qtr,
                y = lngdpcapita,
                color = state)) +
  geom_vline(aes(xintercept = 2012.5)) +

  # themes
  theme_fivethirtyeight() +
  theme(axis.title = element_text()) +

  # labels
  ggtitle('Colorado and Kansas GDP \n before/after Kansas tax cut') +
  xlab('Year-Quarter') +
  ylab('State GDP')
```



QUESTION: Would you pick Colorado or your choice? be the more plausible control unit in this case? Why?

ANSWER: There is a good argument for both of them (Missouri in this case). However, the gap between Colorado and Kansas closes quickly before the treatment period, and similarly it grows between between Kansas and Missouri at the same point.

Selecting comparative units this way can be hard to justify theoretically, and sometimes we do not have a good candidate. What can we do then? This is where synthetic control comes in.

Synthetic Control

Synthetic control is motivated by the problem of choosing comparison units for comparative case studies. It aims to create a "synthetic" version of the treatment unit by combining and weighting covariates from other units ("donors"). In this case, we would construct a synthetic Kansas by creating a weighted average of the other 49 U.S. states. Ideally, the synthetic unit would match the treatment unit in the pre-treatment periods.

For constructing a synthetic control, we are going to primarily rely on the `augsynth` library, since you can use the same library for augmented synthetic controls. The basic syntax for this library is:

`augsynth(outcome ~ trt, unit, time, t_int, data)`

augsynth library

This is a very flexible package that can handle both synthetic controls as well as augmentation and staggered adoption. It's a bit more clunky but will handle the heavy lifting of estimation. Here is a tutorial for simultaneous adoption.

Note that the ATT here varies slightly from the tutorial because we have specified 2012.5 as the first treatment quarter, whereas the tutorial specifies 2012.25 (the quarter in which the law was passed (May)).

```
# NOTE: when t_int is not specified (time when intervention took place), then the code will automatical
# Doesn't seem to run when try to specify t_int anyways

# synthetic control
# ---------
syn <-                              # save object
  augsynth(lngdpcapita ~ treatment, # treatment - use instead of treated bc latter codes 2012.25 as tre
                         state,     # unit
                         year_qtr,  # time
                         kansas,    # data
           progfunc = "None",       # plain syn control
           scm = T)                 # synthetic control
```

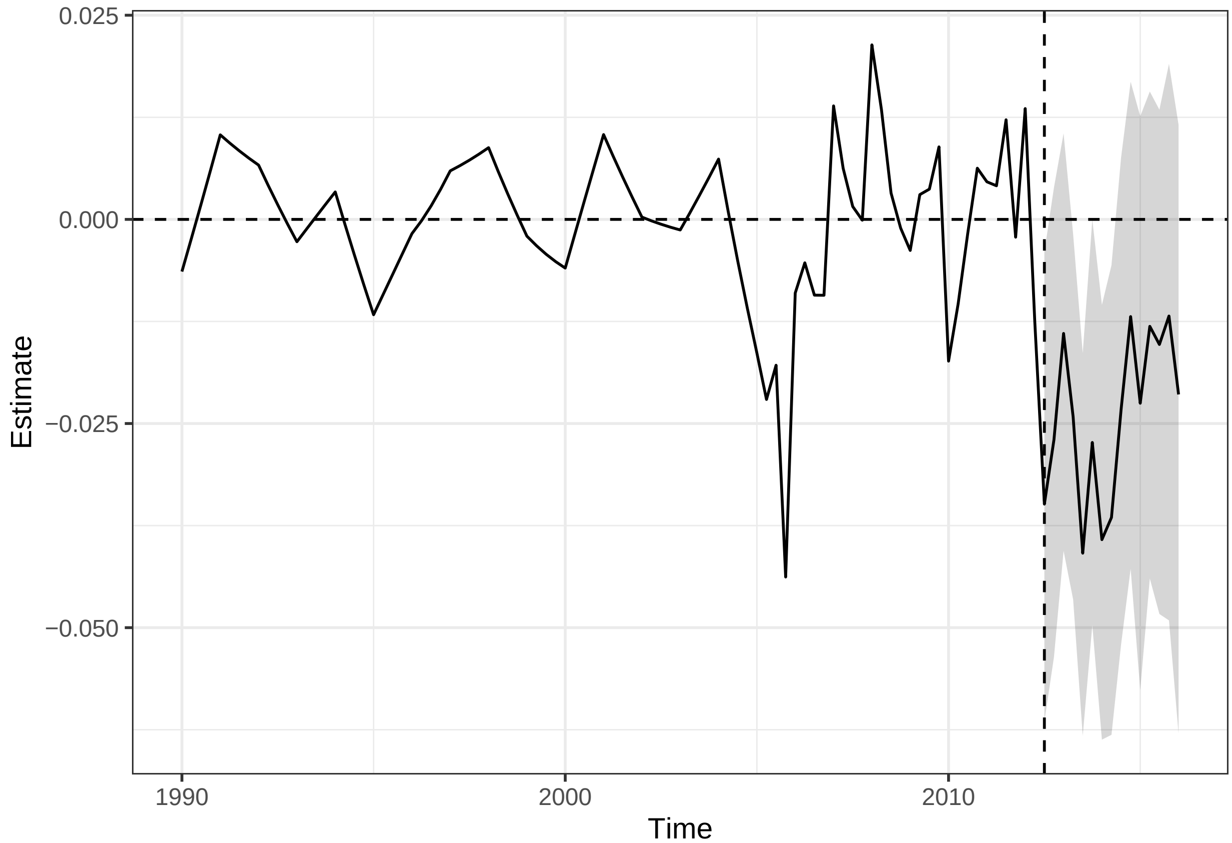
```
## One outcome and one treatment time found. Running single_augsynth.
```

```
# summary
summary(syn)
```

```
##
## Call:
## single_augsynth(form = form, unit = !!enquo(unit), time = !!enquo(time),
##     t_int = t_int, data = data, progfunc = "None", scm = ..2)
##
## Average ATT Estimate (p Value for Joint Null):  -0.0242   ( 0.28 )
## L2 Imbalance: 0.084
## Percent improvement from uniform weights: 79.1%
##
## Avg Estimated Bias: NA
##
## Inference type: Conformal inference
##
##     Time Estimate 95% CI Lower Bound 95% CI Upper Bound p Value
##  2012.50   -0.035             -0.059             -0.004   0.036
##  2012.75   -0.027             -0.054              0.004   0.052
##  2013.00   -0.014             -0.036              0.015   0.131
##  2013.25   -0.024             -0.047              0.005   0.047
##  2013.50   -0.041             -0.065             -0.012   0.016
##  2013.75   -0.027             -0.050             -0.005   0.046
##  2014.00   -0.039             -0.064             -0.015   0.025
##  2014.25   -0.037             -0.063             -0.008   0.018
##  2014.50   -0.023             -0.050              0.008   0.066
##  2014.75   -0.012             -0.043              0.019   0.311
##  2015.00   -0.023             -0.058              0.010   0.091
##  2015.25   -0.013             -0.044              0.016   0.243
##  2015.50   -0.015             -0.048              0.013   0.178
##  2015.75   -0.012             -0.047              0.019   0.303
##  2016.00   -0.021             -0.065              0.014   0.127
```

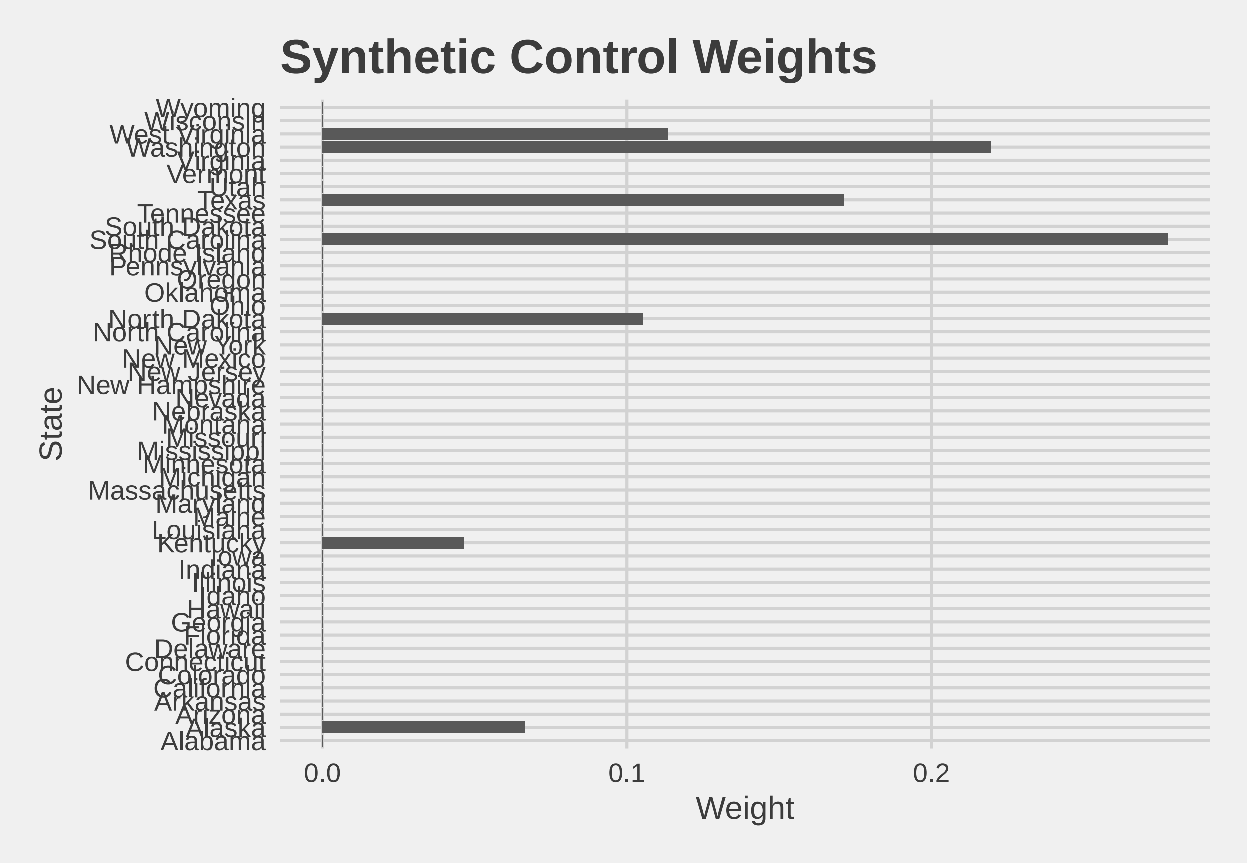
We can use the built in plot function to see how Kansas did relative to synthetic Kansas. The confidence intervals are calculated using Jackknife procedures (leave one out, calculate, and cycle through all).

```
# plot
plot(syn)
```



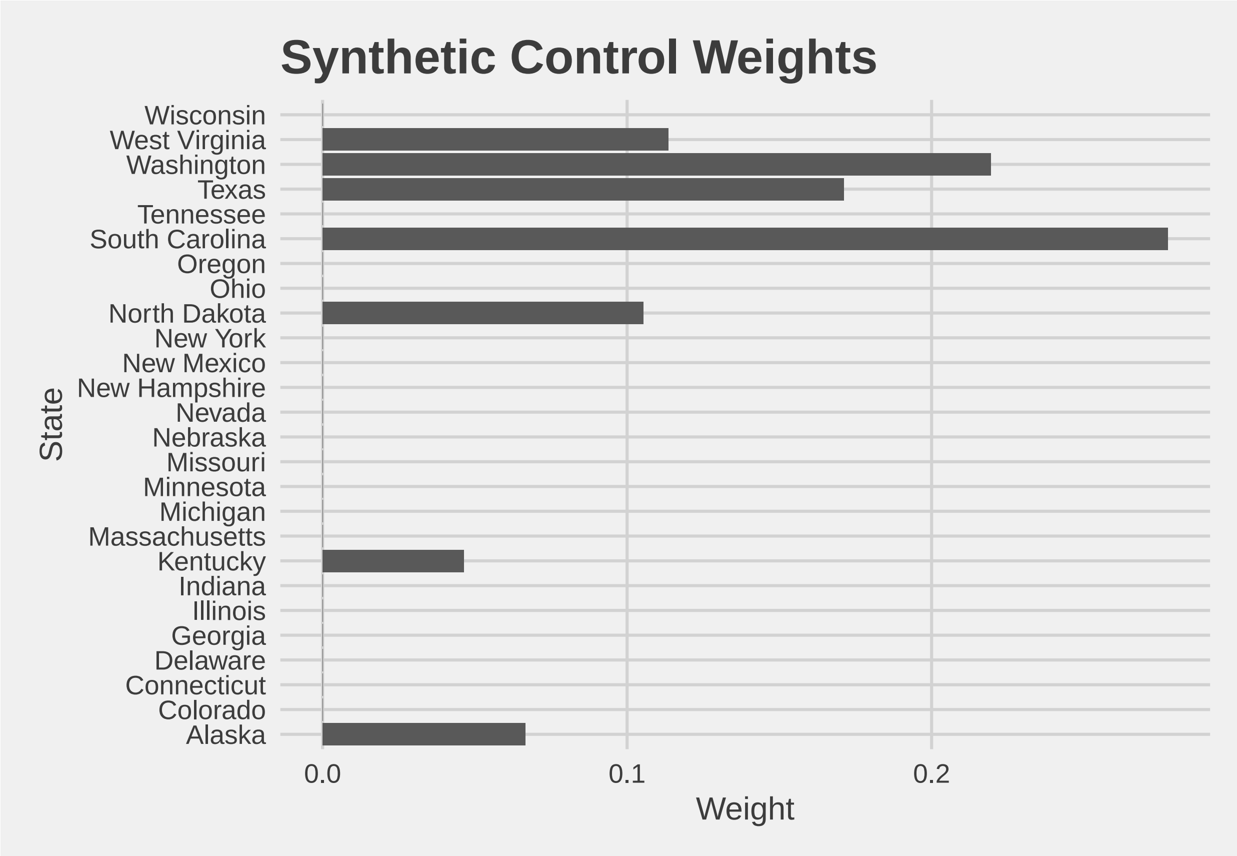
We can see which donors contributed the most to the synthetic Kansas:

```
# view each state's contribution
# ---------
data.frame(syn$weights) %>% # coerce to data frame since it's in vector form
  # process
  # ---------
  # change index to a column
  tibble::rownames_to_column('State') %>% # move index from row to column (similar to index in row as i
  # plot
  # ---------
  ggplot() +
  # stat = identity to take the literal value instead of a count for geom_bar()
  geom_bar(aes(x = State,
               y = syn.weights),
           stat = 'identity') +  # override count() which is default of geom_bar(), could use geom_col(
  coord_flip() +   # flip to make it more readable
  # themes
  theme_fivethirtyeight() +
  theme(axis.title = element_text()) +
  # labels
  ggtitle('Synthetic Control Weights') +
  xlab('State') +
  ylab('Weight')
```



Surprisingly, only a few units ended up contributing! Let's take a closer look at the ones that did:

```
# view each state's contribution, where weights are greater than 0
# ---------
data.frame(syn$weights) %>%
  # processing
  # ---------
  tibble::rownames_to_column('State') %>%
  filter(syn.weights > 0) %>% # filter out weights less than 0
  # plot
  # ---------
  ggplot() +
  geom_bar(aes(x = State,
               y = syn.weights),
           stat = 'identity') +
  coord_flip() +   # flip to make it more readable
  # themes
  theme_fivethirtyeight() +
  theme(axis.title = element_text()) +
  # labels
  ggtitle('Synthetic Control Weights') +
  xlab('State') +
  ylab('Weight')
```



tidysynth library

Before we move on, I want to talk about the `tidysynth` library, which is a new, `tidyverse`-friendly implementation of original `synth` package. As you will see, it is easy to use to visualize the parallel trends, but it cannot handle the augmentation functions we might want to implement and it doesn't have as much support for estimation, unlike `augsynth`. So, you should be aware of it, use it for visualization, but maybe use `augsynth` for estimation and augmentation. Here is a helpful tutorial by the package author as well as an another implementation that might be helpful.

```
#
# specifying a synthetic control using tidysynth
# ----------------------------------------------------------------
# install package
# install.packages('tidysynth')

# load library
library(tidysynth)

# specify synthetic control
kansas_out <-

  kansas %>%

  # initial the synthetic control object
  synthetic_control(outcome = lngdpcapita, # outcome
                    unit = state, # unit index in the panel data
```

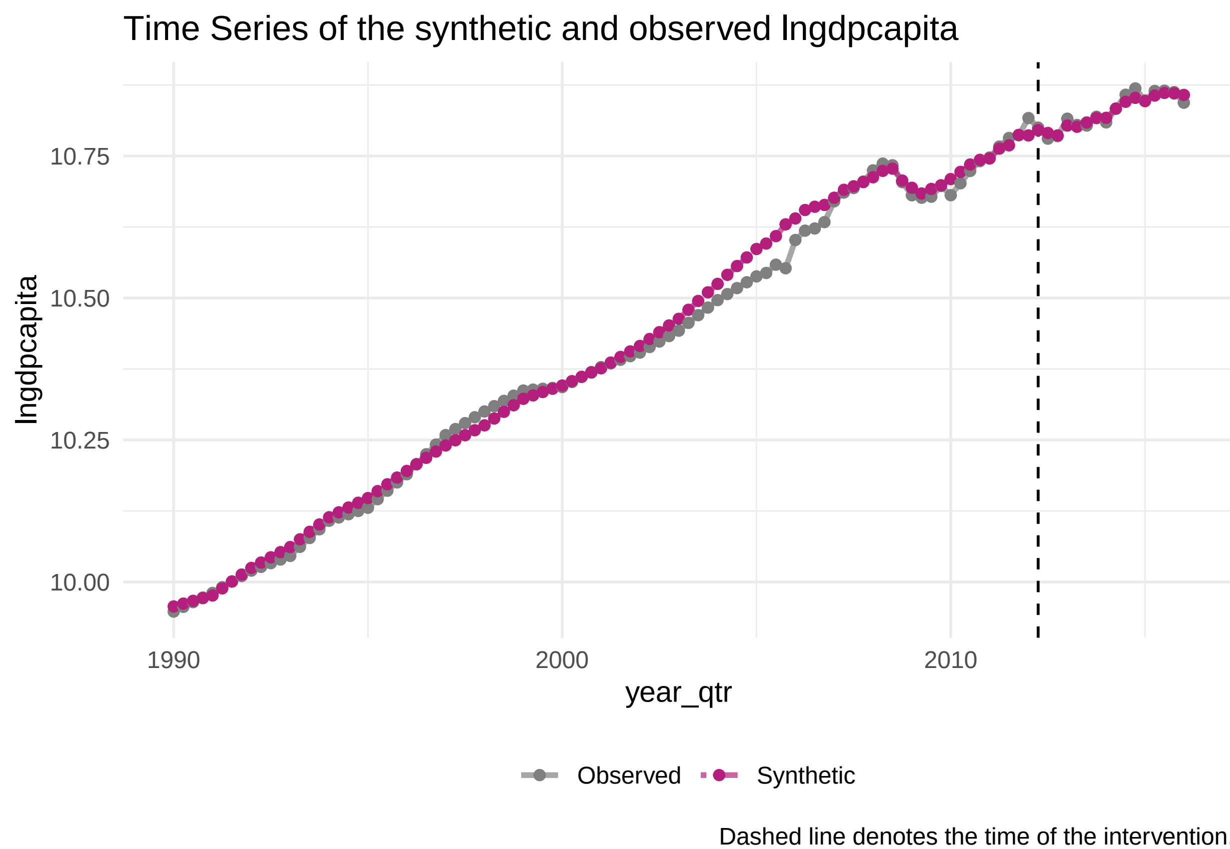
```
                    time = year_qtr, # time index in the panel data
                    i_unit = "Kansas", # unit where the intervention occurred (treatment in augsynth)
                    i_time = 2012.25, # time period when the intervention occurred #  (t_int variable i
                    generate_placebos=T # generate placebo synthetic controls (for inference)
                    ) %>%

  # GDP covariate
  generate_predictor(gdp = gdp) %>%

  # Generate the fitted weights for the synthetic control
  generate_weights(optimization_window = 1990.00:2012.25, # time to use in the optimization task
                   margin_ipop = .02,
                   sigf_ipop = 7,
                   bound_ipop = 6) %>% # optimizer options


  # Generate the synthetic control
  generate_control()
```

Now we can manually calculate a treatment effect (ATT) that approximates what we obtained using `augsynth` but is not exactly the same. For this reason, I might use `augsynth` for estimation.

```
#
# calculate the treatment effect manually
# ----------------------------------------------------------------
kansas_out %>%
  grab_synthetic_control(placebo = T) %>%  # specify placebo to be able to filter on .id variable
  filter(.id == "Kansas")%>%
  filter(time_unit >= 2012.5) %>%  # time period
  # sum all of the post-treatment effects
  mutate(estimate = synth_y - real_y) %>%
  summarize(ATT = sum(estimate)) %>%         # subtract difference to obtain treatment effect
  glimpse()
```

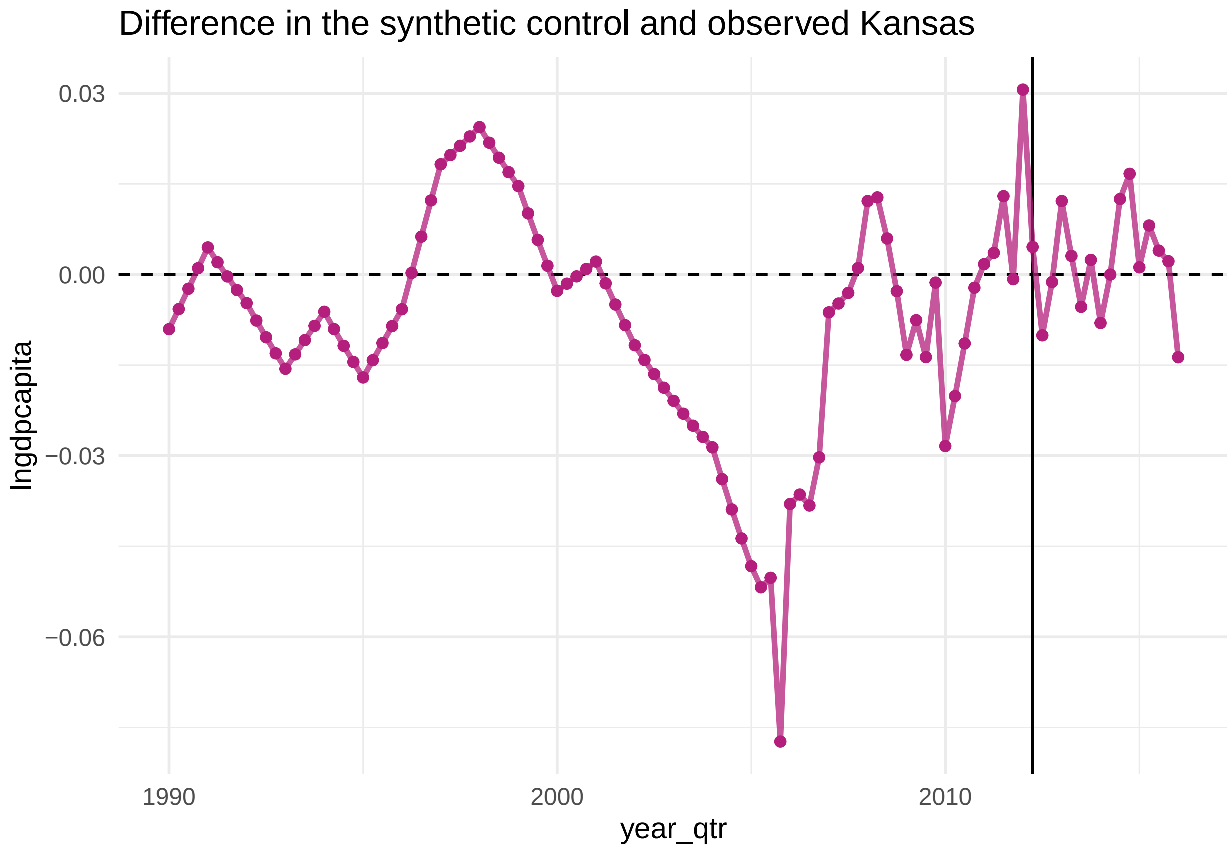
Plot trends. The key here is that we differences in synthetic Kansas more closely tracts Kansas than did Missouri in our DiD.

```
#
# plot parallel trends for synthetic Kansas vs observed Kansas
# ----------------------------------------------------------------
kansas_out %>% plot_trends()
```



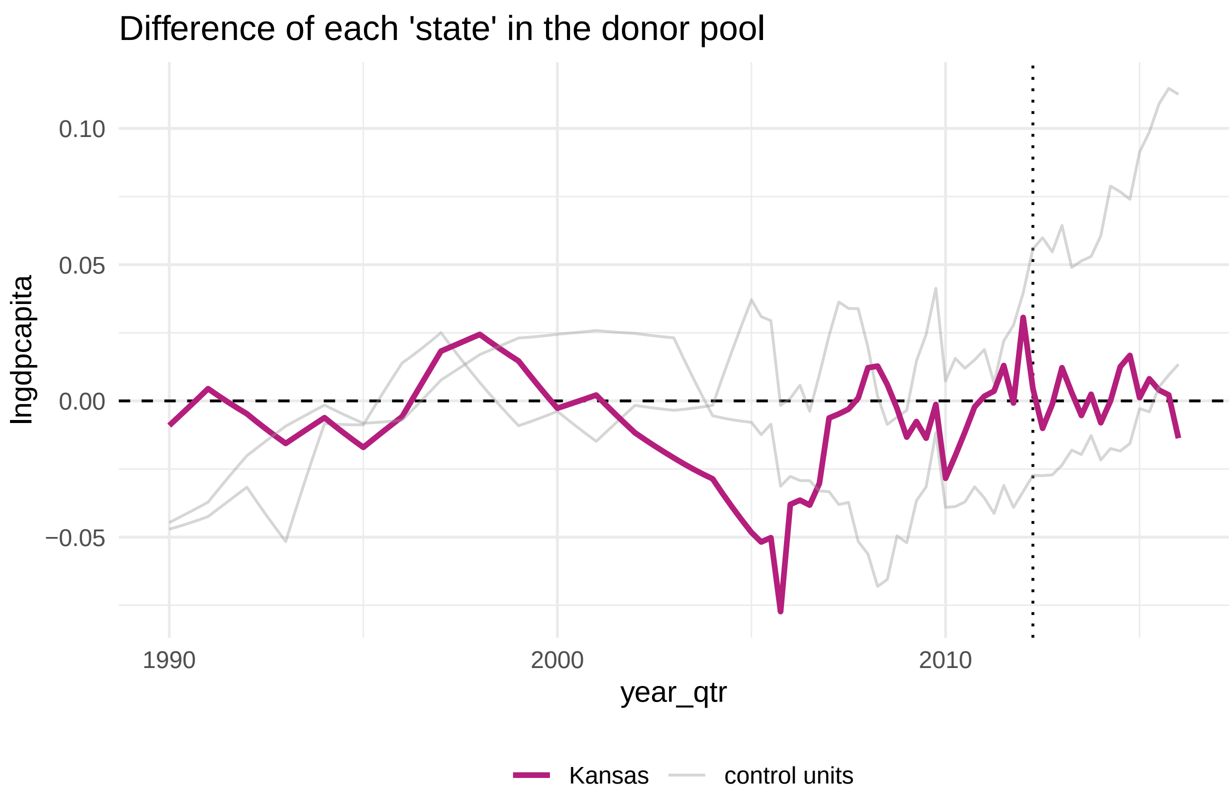
View the differences between Kansas and Synthetic Kansas.

```
#
# plot observed differences between synthetic Kansas vs observed Kansas
# ----------------------------------------------------------------
kansas_out %>% plot_differences()
```



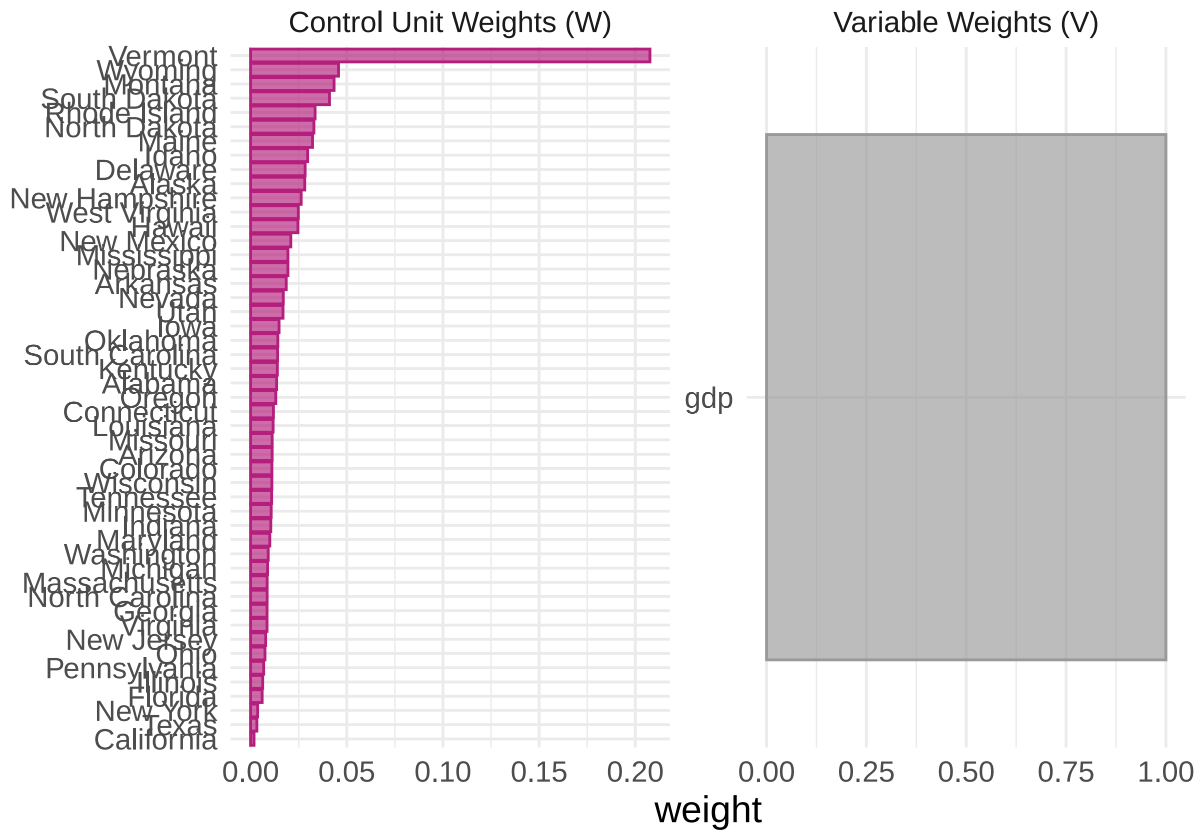
Differences in each state in the donor pool from Kansas. So this shows how much each state varies from Kansas.

```
#
# plot differences in trends for all other states that contribute to synethetic Kansas vs observed Kans
# ---------------------------------------------------------------
kansas_out %>% plot_placebos()
```



Pruned all placebo cases with a pre−period RMSPE exceeding two times the treated unit's pre−period RMSPE.

```
#
# plot control weights of each other state
# ----------------------------------------------------------------
kansas_out %>% plot_weights()
```



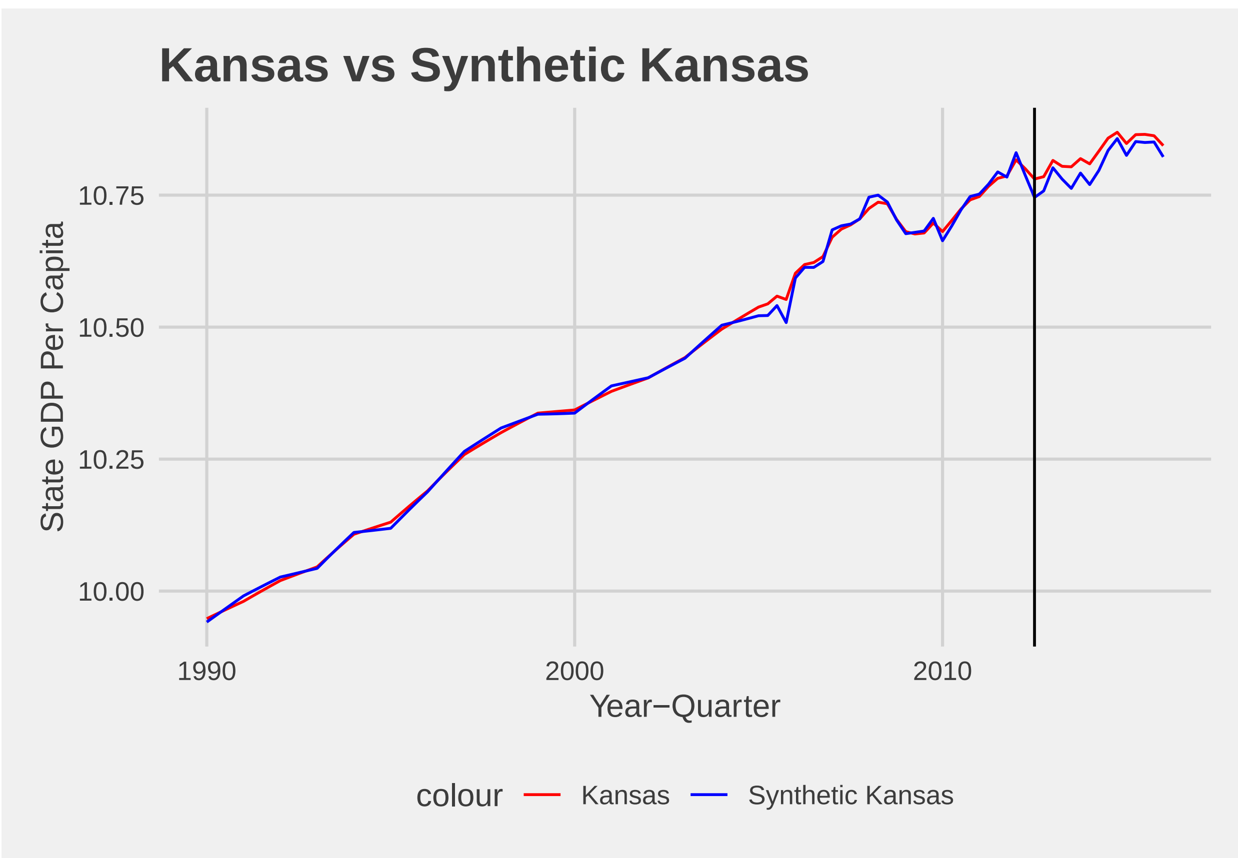
Synthetic Control Augmentation

The main advantage of the `asynth` package is that it allows for "augmented synthetic control". One of the main problems with synthetic control is that if the pre-treatment balance between treatment and control outcomes is poor, the estimate is not valid. Specifically, they advocate for using L2 imbalance, which he first encountered as the penalty that ridge regression uses. L2 uses "squared magnitude" of the coefficient to penalize a particular feature.

Parallel Trends

```
#
# plot parallel trends for synthetic Kansas vs observed Kansas (manually)
# ----------------------------------------------------------------

# Aniket's method for getting the underlying data
# ---------
syn_sum <- summary(syn)

# create synthetic Kansas
# ---------
kansas_synkansas <-
  # data
  kansas %>%
  # filter just Kansas
  filter(state == "Kansas") %>%
```

```
  # bind columns
  bind_cols(difference = syn_sum$att$Estimate) %>%    # add in estimate
  # calculate synthetic Kansas
  mutate(synthetic_kansas = lngdpcapita + difference) # adds the estimate to the observed Kansas to cre

# plot
# ---------
kansas_synkansas %>%
  ggplot() +
  # kansas
  # ---------
  geom_line(aes(x = year_qtr,
              y = lngdpcapita,
              color = 'Kansas')) +
  # synthetic kansas
  # ---------
  geom_line(aes(x = year_qtr,
              y = synthetic_kansas,
              color = 'Synthetic Kansas')) +
  scale_color_manual(values = c('Kansas' = 'red', 'Synthetic Kansas' = 'blue')) +
  geom_vline(aes(xintercept = 2012.5)) +
  theme_fivethirtyeight() +
  theme(axis.title = element_text()) +
  ggtitle('Kansas vs Synthetic Kansas') +
  xlab('Year-Quarter') +
  ylab('State GDP Per Capita')
```



QUESTION: How does pre-treatment matching between Kansas and Synthetic Kansas look here?

ANSWER: Pretty good! We may not need to augment this synthetic control, though let's try anyway.

Augmentation

Let's play a bit with the augmentation parameters that will adjust the weights to see if we can find better fits to create a synthetic control.

```
#
# recalculate with Ridge function that penalizes really high weights
# ------------------------------------------------------------------
ridge_syn <-
  augsynth(lngdpcapita ~ treatment,
                         state,
                         year_qtr,
                         kansas,
           progfunc = "ridge",  # specify
           scm = T)
```

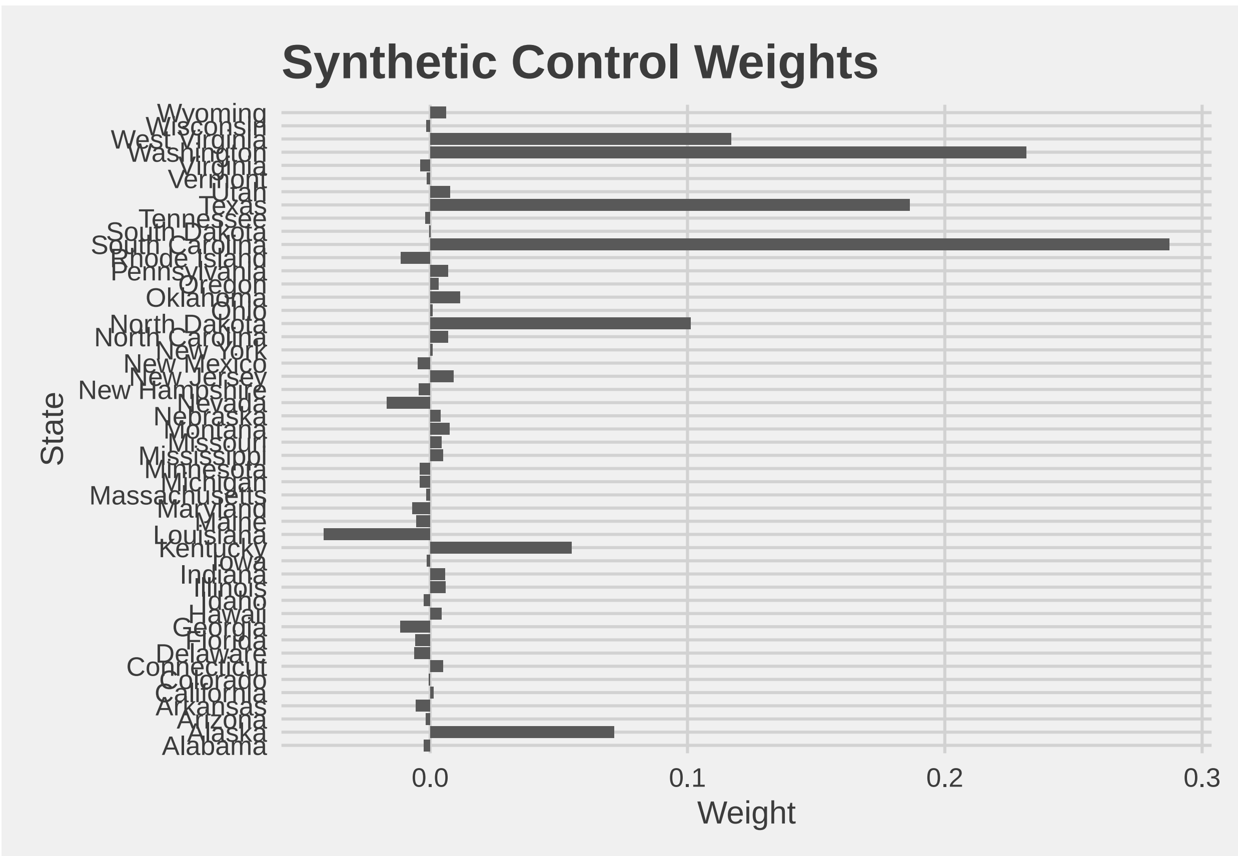
```
## One outcome and one treatment time found. Running single_augsynth.
```

```
summary(ridge_syn) # the lower the L2 balance, the better -- now 0.07 compared to ~0.08
```

```
##
## Call:
## single_augsynth(form = form, unit = !!enquo(unit), time = !!enquo(time),
##     t_int = t_int, data = data, progfunc = "ridge", scm = ..2)
##
## Average ATT Estimate (p Value for Joint Null):  -0.0298   ( 0.14 )
## L2 Imbalance: 0.070
## Percent improvement from uniform weights: 82.7%
##
## Avg Estimated Bias: 0.006
##
## Inference type: Conformal inference
##
##     Time Estimate 95% CI Lower Bound 95% CI Upper Bound p Value
##  2012.50   -0.038             -0.065             -0.013   0.023
##  2012.75   -0.031             -0.058             -0.004   0.036
##  2013.00   -0.019             -0.041              0.002   0.066
##  2013.25   -0.031             -0.055             -0.009   0.011
##  2013.50   -0.048             -0.075             -0.023   0.028
##  2013.75   -0.034             -0.058             -0.012   0.022
##  2014.00   -0.046             -0.073             -0.022   0.020
##  2014.25   -0.043             -0.072             -0.016   0.026
##  2014.50   -0.029             -0.061              0.000   0.055
##  2014.75   -0.017             -0.052              0.012   0.122
##  2015.00   -0.028             -0.065              0.004   0.055
##  2015.25   -0.019             -0.053              0.011   0.076
##  2015.50   -0.021             -0.055              0.009   0.099
##  2015.75   -0.017             -0.057              0.018   0.112
##  2016.00   -0.026             -0.069              0.006   0.053
```

Let's look at the weights:

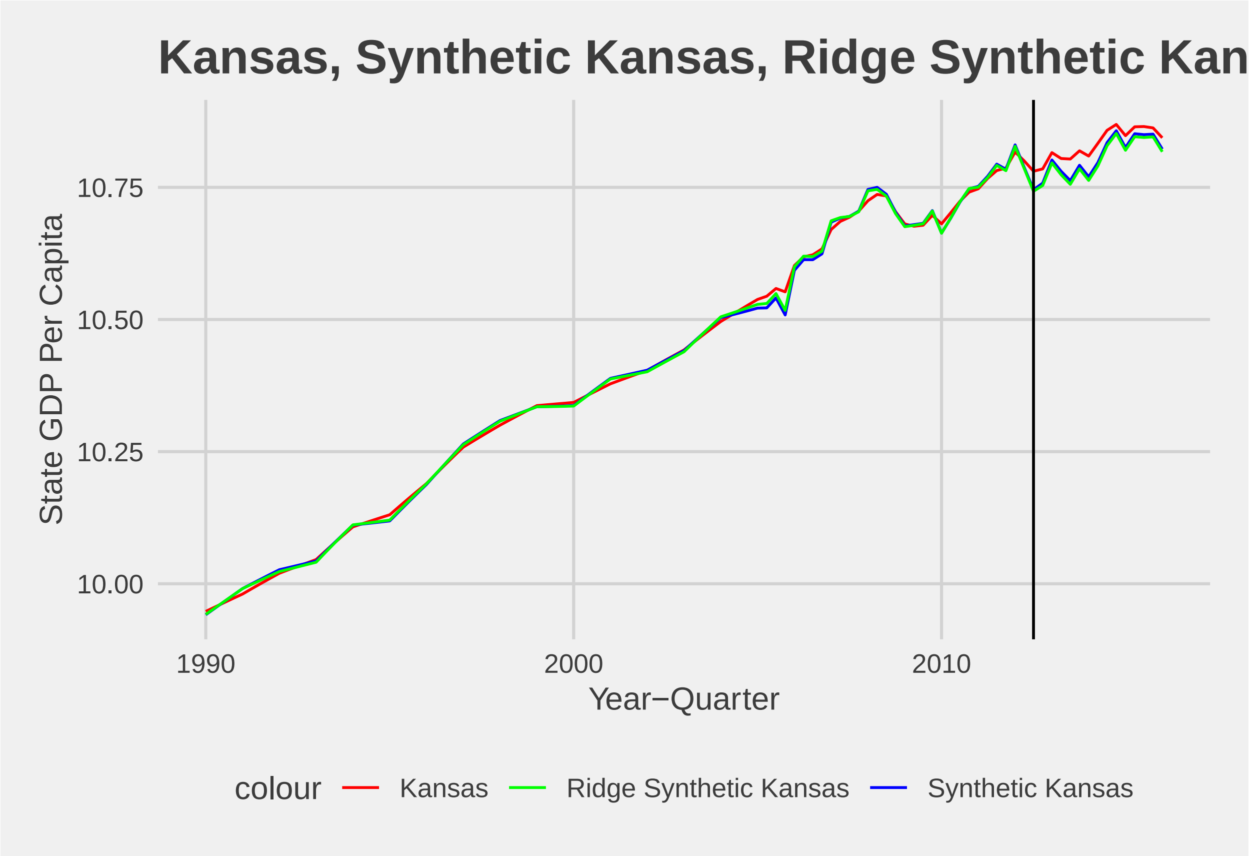
```r
#
# view weights - now we have negative weights as a result of Ridge
# ------------------------------------------------------------------
data.frame(ridge_syn$weights) %>%
  tibble::rownames_to_column('State') %>%
  ggplot() +
  geom_bar(aes(x = State, y = ridge_syn.weights),
           stat = 'identity') +
  coord_flip() + # coord flip
  theme_fivethirtyeight() +
  theme(axis.title = element_text()) +
  ggtitle('Synthetic Control Weights') +
  xlab('State') +
  ylab('Weight')
```



Notice how with the ridge augmentation, some weights are allowed to be negative now. Now let's go ahead and plot the ridge augmented synthetic Kansas alongside Kansas and synthetic Kansas:

```r
#
# plot parallel trends for observed Kansas vs synthetic Kansas vs ridge Kansas
# ------------------------------------------------------------------

# Aniket's method for getting the underlying data
# ---------
ridge_sum <- summary(ridge_syn)

# create synthetic Kansas
```

```r
# ---------
kansas_synkansas_ridgesynkansas <- kansas_synkansas %>%
  bind_cols(ridge_difference = ridge_sum$att$Estimate) %>%
  mutate(ridge_synthetic_kansas = lngdpcapita + ridge_difference)

# plot
# ---------
kansas_synkansas_ridgesynkansas %>%
  ggplot() +

  # kansas
  # ---------
  geom_line(aes(x = year_qtr,
                y = lngdpcapita,
                color = 'Kansas')) +
  # synthetic kansas
  # ---------
  geom_line(aes(x = year_qtr,
                y = synthetic_kansas,
                color = 'Synthetic Kansas')) +
  # ridge kansas
  # ---------
  geom_line(aes(x = year_qtr,
                y = ridge_synthetic_kansas,
                color = 'Ridge Synthetic Kansas')) +
  # use scale color manual to assign color values
  scale_color_manual(values = c('Kansas' = 'red',
                                'Synthetic Kansas' = 'blue',
                                'Ridge Synthetic Kansas' = 'green')) +
  geom_vline(aes(xintercept = 2012.5)) +
  # themes
  theme_fivethirtyeight() +
  theme(axis.title = element_text()) +
  # labels
  ggtitle('Kansas, Synthetic Kansas, Ridge Synthetic Kansas') +
  xlab('Year-Quarter') +
  ylab('State GDP Per Capita')
```



These all seem pretty good! Like we thought, augmentation did not necessarily improve the matches in this particular dataset. We can check the two L2 imbalances and see that we have reduced the overall imbalance a bit with our ridge model:

```
# print imbalances
# ---------
print(syn$l2_imbalance)
```

```
## [1] 0.083922
```

```
print(ridge_syn$l2_imbalance)
```

```
## [1] 0.0695046
```

Finally, we can add covariates to our model if we would like:

Adding covariates

```
#
# add in some covariates
# ----------------------------------------------------------------
data(kansas)

covsyn <- augsynth(lngdpcapita ~ treated | lngdpcapita + log(revstatecapita) +
                                           log(revlocalcapita) + log(avgwklywagecapita) +
                                           estabscapita + emplvlcapita,
                   fips,                # unit
                   year_qtr,            # time
```

```r
                   kansas,              # data
                   progfunc = "ridge",  # augmentation
                   scm = T)             # synthetic control
```

```
## One outcome and one treatment time found. Running single_augsynth.
```

```r
summary(covsyn)
```

```
##
## Call:
## single_augsynth(form = form, unit = !!enquo(unit), time = !!enquo(time),
##     t_int = t_int, data = data, progfunc = "ridge", scm = ..2)
##
## Average ATT Estimate (p Value for Joint Null):  -0.0609   ( 0.11 )
## L2 Imbalance: 0.054
## Percent improvement from uniform weights: 86.6%
##
## Covariate L2 Imbalance: 0.005
## Percent improvement from uniform weights: 97.7%
##
## Avg Estimated Bias: 0.027
##
## Inference type: Conformal inference
##
##     Time Estimate 95% CI Lower Bound 95% CI Upper Bound p Value
##  2012.25   -0.021             -0.044              0.000   0.085
##  2012.50   -0.047             -0.076             -0.019   0.035
##  2012.75   -0.050             -0.083             -0.007   0.025
##  2013.00   -0.045             -0.074             -0.012   0.044
##  2013.25   -0.055             -0.088             -0.022   0.024
##  2013.50   -0.071             -0.110             -0.033   0.016
##  2013.75   -0.058             -0.091             -0.025   0.022
##  2014.00   -0.081             -0.125             -0.037   0.020
##  2014.25   -0.078             -0.121             -0.019   0.026
##  2014.50   -0.065             -0.119             -0.006   0.033
##  2014.75   -0.057             -0.110             -0.008   0.038
##  2015.00   -0.075             -0.124             -0.037   0.032
##  2015.25   -0.063             -0.106             -0.014   0.025
##  2015.50   -0.067             -0.111             -0.019   0.024
##  2015.75   -0.063             -0.101             -0.009   0.017
##  2016.00   -0.078             -0.122             -0.019   0.030
```

Staggered Adoption

The last technique we'll look at is "staggered adoption" of some policy. In the original Hainmueller paper, states that already had similar cigarette taxes were discarded from the donor pool to create a synthetic California. But what if we were interested in the effect of a policy overall, for every unit that adopted treatment? The problem is, these units all choose to adopt treatment at different times. We could construct different synthetic controls for each one, or we can use a staggered adoption approach.

To explore this question, we'll continue using the `augsynth` package's vignette. This time we will load a dataset that examines the effect of states instituting mandatory collective bargaining agreements.

```r
# import data
collective_bargaining <- read_delim("https://dataverse.harvard.edu/api/access/datafile/:persistentId?pe
```

```
## Rows: 3723 Columns: 23
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr  (1): State
## dbl (22): year, Stateid, avgteachsal, YearCBrequired, CBstatusby1990, ppexpe...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# view head
head(collective_bargaining)
```

```
## # A tibble: 6 x 23
##    year State Stateid avgteachsal YearCBrequired CBstatusby1990 ppexpend
##   <dbl> <chr>   <dbl>       <dbl>          <dbl>          <dbl>    <dbl>
## 1  1899 AK          1          NA           1970              2       NA
## 2  1900 AK          1          NA           1970              2       NA
## 3  1904 AK          1          NA           1970              2       NA
## 4  1909 AK          1          NA           1970              2       NA
## 5  1910 AK          1          NA           1970              2       NA
## 6  1912 AK          1          NA           1970              2       NA
## # i 16 more variables: avginstrucsal <dbl>, agr <dbl>, perinc <dbl>,
## #   pnwht <dbl>, purban <dbl>, ESWI <dbl>, studteachratio <dbl>,
## #   nonwageppexpend <dbl>, lnppexpend <dbl>, lnavginstrucsal <dbl>,
## #   lnavgteachsal <dbl>, lnnonwageppexpend <dbl>, CBrequired_SY <dbl>,
## #   CBeverrequired <dbl>, South <dbl>, idmap <dbl>
```

The main variables we'll use here are:

The dataset contains several important variables that we'll use:

- `year`, `State`: The state and year of the measurement
- `YearCBrequired`: The year that the state adopted mandatory collective bargaining
- `lnppexpend`: Log per pupil expenditures in 2010 dollars

Let's do some preprocessing before we estimate some models. We're going to remove DC and Wisconsin from the analysis and cabin our dataset to 1959 - 1997. Finally, we'll add a treatment indicator `cbr` which takes a 1 if the observation was a treated state after it adopted mandatory collective bargaining, or a 0 otherwise:

```r
#
# create dataset
# ---------
collective_bargaining_clean <-
  # data
  collective_bargaining %>%
    # filter out two exceptions and subset to appropriate years
    filter(!State %in% c("DC", "WI"),
           year >= 1959,
           year <= 1997) %>%
    # create "treatment" - year collective bargaining was adopted
    mutate(YearCBrequired = ifelse(is.na(YearCBrequired),
                                   Inf, YearCBrequired),
           cbr = 1 * (year >= YearCBrequired))
```

We're ready to start estimating a model! To do this, we use the `multisynth()` function that has the following signature:

```
mutltisynth(outcome ~ treatment, unit, time, nu, data,  n_leads)
```

The key parameters here are `nu` and `n_leads`. Staggered adoption uses multi-synthetic control which essentially pools together similar units and estimates a synthetic control for each pool. `nu` determines how much pooling to do. A value of 0 will fit a separate synthetic control for each model, whereas a value of 1 will pool all units together. Leaving this argument blank with have `augsynth` search for the best value of `nu` that minimizes L2 loss. Determining this is more of an art—the hard and fast rule is DO NOT estimate more post-treatment periods than pre-treatment ones. `n_leads` determines how many time periods to estimate in the post-treatment period.

```
#
# implementing staggered adoption
# ----------------------------------------------------------------

#
# setting nu to 0.5
# ---------
ppool_syn <- multisynth(lnppexpend ~ cbr,
                        State,                          # unit
                        year,                           # time
                        nu = 0.5,                       # varying degree of pooling
                        collective_bargaining_clean,    # data
                        n_leads = 10)                   # post-treatment periods to estimate
# with default nu
# ---------
ppool_syn <- multisynth(lnppexpend ~ cbr,
                        State,                           # unit
                        year,                            # time
                        collective_bargaining_clean,     # data
                        n_leads = 10)                    # post-treatment periods to estimate

# view results
print(ppool_syn$nu)
```

```
## [1] 0.2618752
```

```
ppool_syn
```

```
##
## Call:
## multisynth(form = lnppexpend ~ cbr, unit = State, time = year,
##     data = collective_bargaining_clean, n_leads = 10)
##
## Average ATT Estimate: -0.010
```

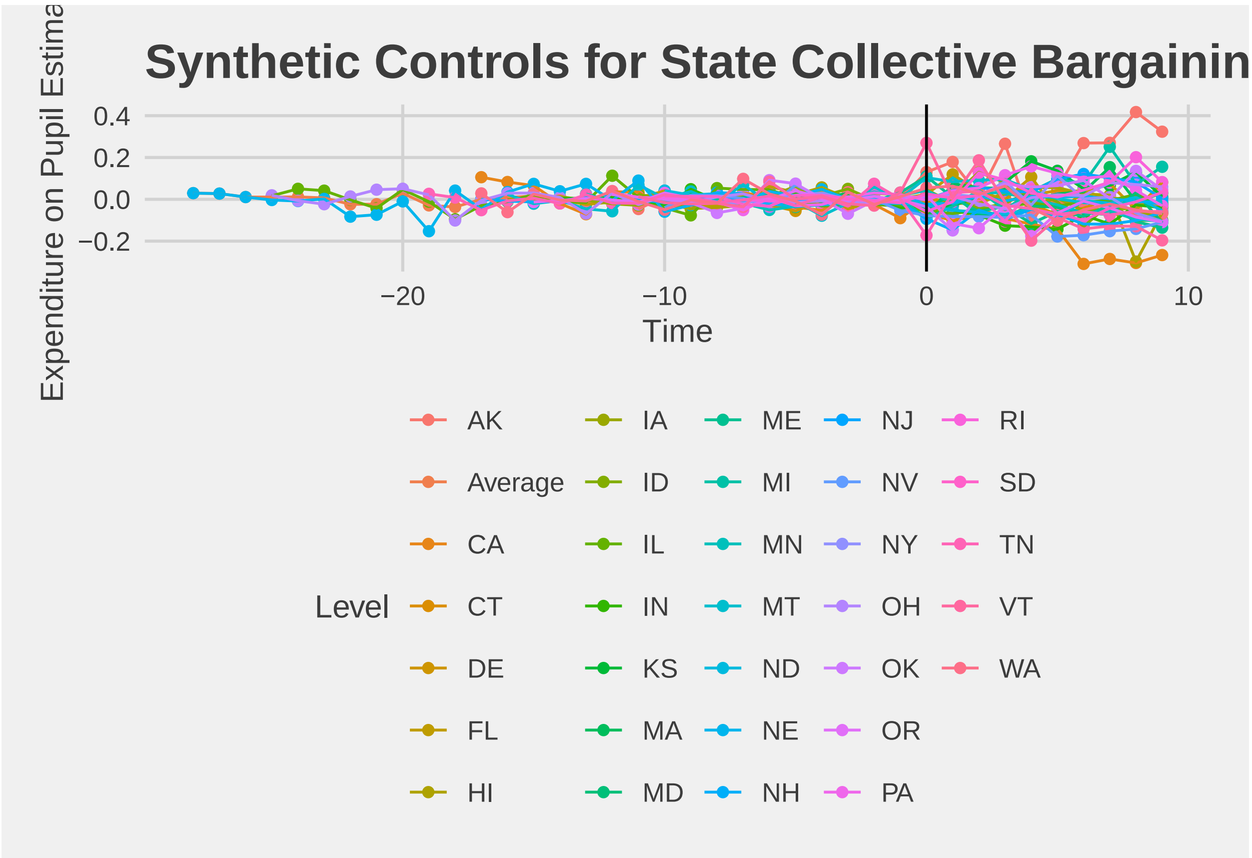
After you've fit a model that you like, use the `summary()` function to get the ATT and balance statistics.

```
# save ATT and balance stats
# ---------
ppool_syn_summ <- summary(ppool_syn)
```

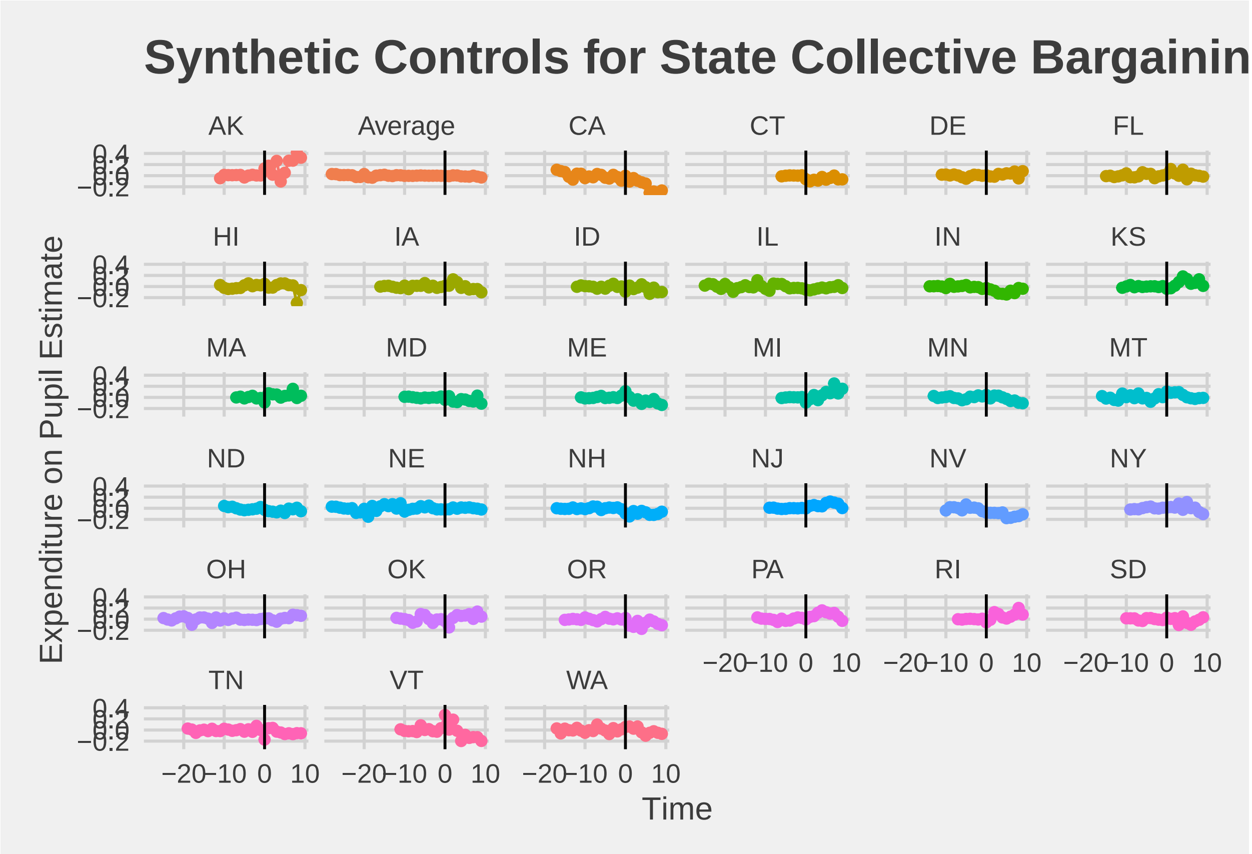
Next, plot the estimates for each state as well as the average average treatment effect (so average for all treated states). Try to do this with `ggplot()` instead of the built-in plotting function (hint: how did we get the dataframe with the estimates before?)

```
# plot actual estimates not values of synthetic controls
# ---------
ppool_syn_summ$att %>%
  ggplot(aes(x = Time, y = Estimate, color = Level)) +
```

```r
  geom_point() +
  geom_line() +
  geom_vline(xintercept = 0) +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(),
        legend.position = "bottom") +
  ggtitle('Synthetic Controls for State Collective Bargaining') +
  xlab('Time') +
  ylab('Expenditure on Pupil Estimate')
```




```r
# plot actual estimates not values of synthetic controls - use a facet_wrap for readability
# ----------
ppool_syn_summ$att %>%
  ggplot(aes(x = Time, y = Estimate, color = Level)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = 0) +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(),
        legend.position = 'None') +
  ggtitle('Synthetic Controls for State Collective Bargaining') +
  xlab('Time') +
  ylab('Expenditure on Pupil Estimate') +
  facet_wrap(~Level)    # facet-wrap by level (state in this case) for clearer presentation
```



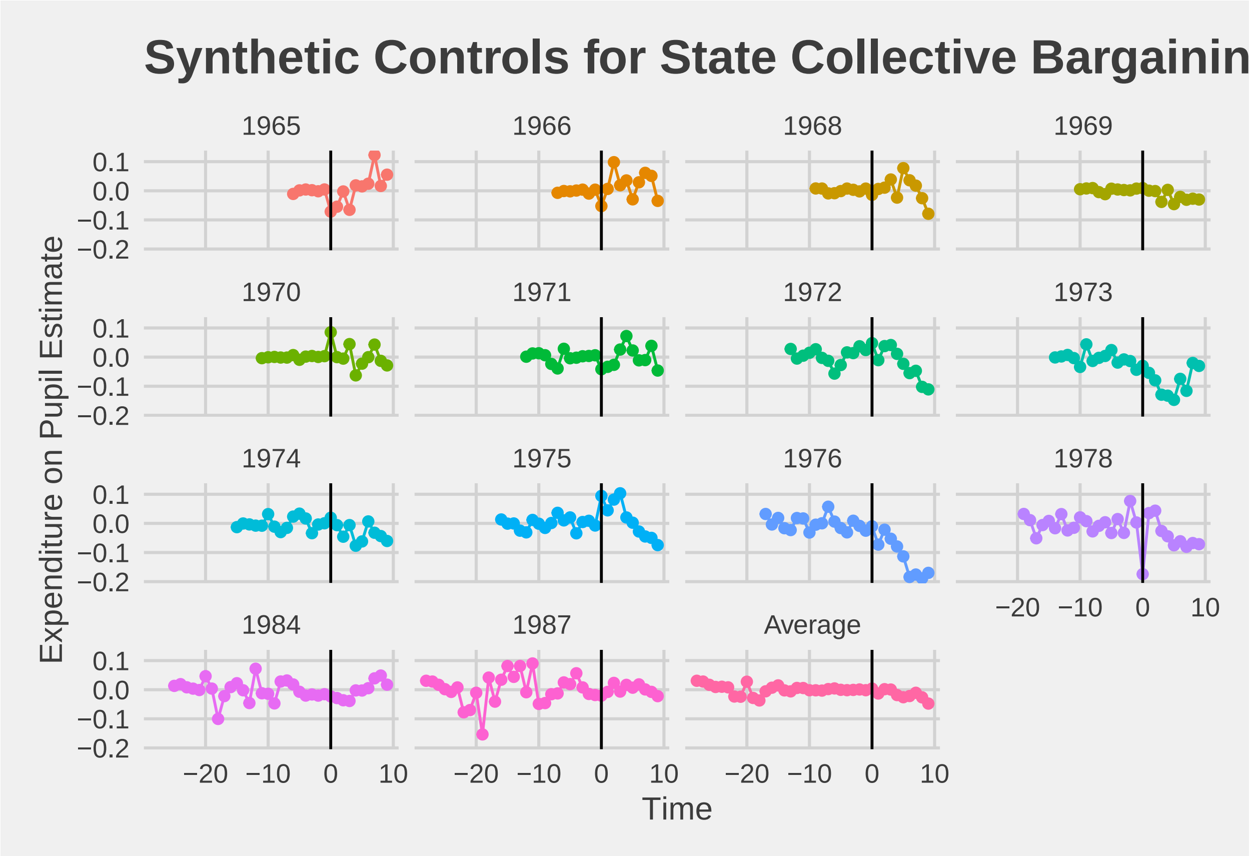
We can also combine our observations into "time cohorts" or units that adopted treatment at the same time. Try adding `time_cohort = TRUE` to your multisynth function and see if your estimates differ. Plot these results as well.

```
#
# break observations into time cohorts
# ---------------------------------------------------------------
ppool_syn_time <- multisynth(lnppexpend ~ cbr,
                             State,
                             year,
                             collective_bargaining_clean,
                             n_leads = 10,
                             time_cohort = TRUE)           # time cohort set to TRUE

# save summary
ppool_syn_time_summ <- summary(ppool_syn_time)

# view
ppool_syn_time_summ
```

```
##
## Call:
## multisynth(form = lnppexpend ~ cbr, unit = State, time = year,
##     data = collective_bargaining_clean, n_leads = 10, time_cohort = TRUE)
##
## Average ATT Estimate (Std. Error): -0.016  (0.022)
##
```

```
## Global L2 Imbalance: 0.005
## Scaled Global L2 Imbalance: 0.018
## Percent improvement from uniform global weights: 98.2
##
## Individual L2 Imbalance: 0.039
## Scaled Individual L2 Imbalance: 0.058
## Percent improvement from uniform individual weights: 94.2
##
##  Time Since Treatment   Level      Estimate  Std.Error lower_bound upper_bound
##                     0 Average  0.0038263026 0.02351018 -0.04499867  0.04785117
##                     1 Average -0.0130748834 0.02363226 -0.06096703  0.03279031
##                     2 Average  0.0018300044 0.02327762 -0.04069697  0.04755810
##                     3 Average  0.0005232868 0.02550527 -0.04805002  0.05254703
##                     4 Average -0.0184345032 0.02423198 -0.06451377  0.02593260
##                     5 Average -0.0258163688 0.02491757 -0.06977512  0.02194964
##                     6 Average -0.0217543090 0.02511451 -0.07064656  0.02701818
##                     7 Average -0.0105432314 0.03037188 -0.07004877  0.04814811
##                     8 Average -0.0262042318 0.03161621 -0.09045378  0.03363515
##                     9 Average -0.0476919393 0.03036504 -0.11007285  0.01089619
```

```r
# plot effect for each time period (local treatment effects)
# ---------
ppool_syn_time_summ$att %>%
  ggplot(aes(x = Time, y = Estimate, color = Level)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = 0) +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(),
        legend.position = 'None') +
  ggtitle('Synthetic Controls for State Collective Bargaining') +
  xlab('Time') +
  ylab('Expenditure on Pupil Estimate') +
  facet_wrap(~Level)
```



Finally, we can add in augmentation. Again augmentation essentially adds a regularization penalty to the synthetic control weights. In the multisynth context, you may especially want to do this when the pre-treatment fit is poor for some of your units. There are a couple of different options for augmentation. One is to specify `fixed_effects = TRUE` in the multsynth call, and this will estimate unit fixed effects models after de-meaning each unit. We can also specify a `n_factors =` argument (substituting an integer in) to use the `gsynth` method that uses cross-validation to estimate the weights for multi-synthetic control.

Try creating an augmented synthetic control model. How do your balance and estimates compare?

```
# likely need to install gsynth package do not need to load
# install.packages("gsynth")

# play with fixed effects and use cross validation
# ---------------------------------------------------------------
scm_gsyn <- multisynth(lnppexpend ~ cbr,
                      State,
                      year,
                      collective_bargaining_clean,
                      n_leads = 10,
                      fixedeff = T,    # fixed effect- for units in time
                      n_factors = 2)   # uses cross-validation to determine most appropriate weights

# save s
scm_gsyn_summ <- summary(scm_gsyn)
```

```
#
# plot multisynth
```

```r
# --------------------------------------------------------------
scm_gsyn_summ$att %>%
  ggplot(aes(x = Time, y = Estimate, color = Level)) +
  geom_point() +
  geom_line() +
  geom_vline(xintercept = 0) +
  theme_fivethirtyeight() +
  theme(axis.title = element_text(),
        legend.position = 'None') +
  ggtitle('Augmented Synthetic Controls for State Collective Bargaining') +
  xlab('Time') +
  ylab('Expenditure on Pupil Estimate') +
  facet_wrap(~Level)
```