Lecture 9: Synthetic Control Methods

James Sears*

AFRE 891 SS 24

Michigan State University

*Parts of these slides are adapted from **"Econometrics III"** by Ed Rubin.

Table of Contents

Prologue

Prologue

This lecture is focusing on **Synthetic Control Methods**, which will let us solve several of the issues that can affect methods we discussed last lecture.

Part 1

- The Fundamental Problem of Causal Inference
- Matching
- Canonical Synthetic Control

Part 2

- Synthetic Diff-in-Diff
  - Uniform Adoption
  - Staggered Adoption
- Partially Pooled Synthetic Control

Prologue

Packages we'll use today:

```r
# If not installed, add in packages from GitHub not on CRAN
if (!require("augsynth")) remotes::install_github("ebenmichael/augsynth")
```

```r
if (!require("pacman")) install.packages("pacman")
pacman::p_load(augsynth, fixest, synthdid, tidysynth, tidyverse)
```

As well, let's load the event study data from **Sears et al. (2023)** we finished last lecture with:

```r
sah <- readRDS("data/sah_es.rds")
```

Causal Inference

Let's chat about the **fundamental problem of causal inference** for a moment.

Consider unit i's **potential outcomes:**

- $\mathbf{Y}_{1i}$: the outcome for unit i under the treatment
  - Treatment assignment $D_i = 1$
- $\mathbf{Y}_{0i}$: the outcome for unit i absent the treatment
  - Treatment assignment $D_i = 0$

1. We want/need to know $\tau_i = \mathbf{Y}_{1i} - \mathbf{Y}_{0i}$.
2. We cannot simultaneously observe *both* $\mathbf{Y}_{1i}$ *and* $\mathbf{Y}_{0i}$.

Most (all?) empirical strategies boil to estimating $\mathbf{Y}_{0i}$ for treated individuals — the **unobservable counterfactual** for the treatment group.

Causal Inference

Last lecture gave an overview of regression methods that make different assumptions about that **unobservable counterfactual**.

1. RCT + Random Assignment

- The average in the control group is what the average in the treatment group would have been absent the treatment
- Regress outcome on treatment dummy and you're good to go
  - Maybe add some control variables to improve precision of estimator

2. Difference-in-Differences + Event Study

- The **change over time** in the control group is what the change in the treatment group would have been absent the treatment

Causal Inference

All of these estimates are identified under a variation of the **Conditional Independence Assumption (CIA)**†

$$\{Y_{0i}, Y_{1i}\} \perp\!\!\!\perp D_i \mid \mathbf{X}_i$$

Conditional on $\mathbf{X}_i$[1], potential outcomes (Y_{0i}, Y_{1i}) are independent of treatment status (D_i).

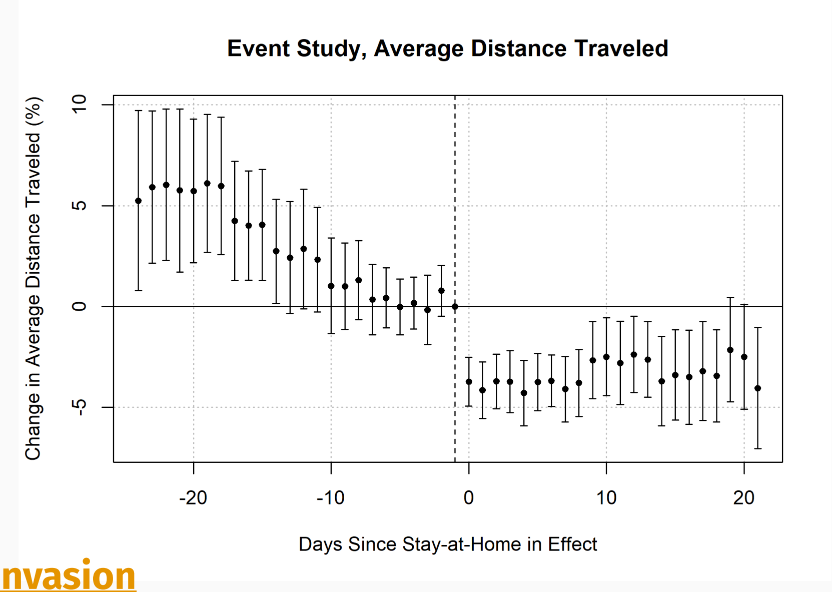
† AKA "selection on observables".

1. Or *it* if we're in a panel setting

But there are times when **CIA fails**[2].

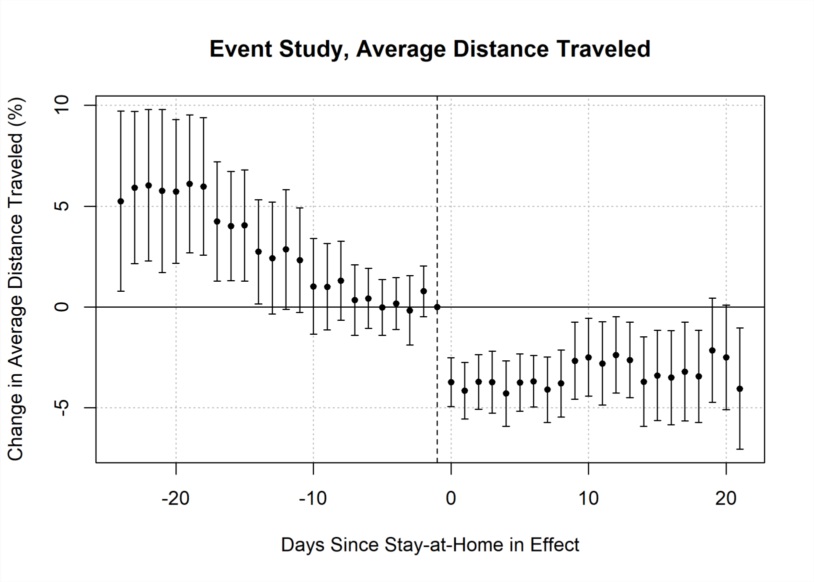
In the case of Diff-in-Diff and Event Study, this is often due to a failure of **parallel trends**.

For example, recall the event study for mobility responses to stay-at-home mandates from last lecture:



2. See the Bay of Pigs Invasion

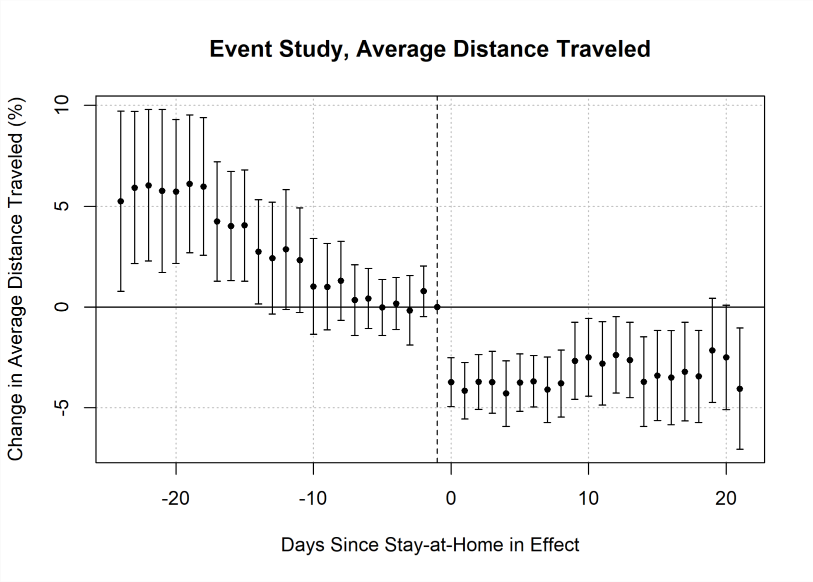
Causal Inference



In this case, the states that never adopted stay-at-home mandates might not be *valid counterfactuals* for the states that did adopt.

However, there might be a way to *construct* a valid counterfactual from the set of control units...

Causal Inference



... but before we get into that, let's chat briefly about **matching estimators**.

Matching

Matching

Matching Estimators provide an alternate way of coming up with the unobservable counterfactual for the treatment group.

The gist:

- Match untreated observations to treated observations using $\mathbf{X}_i$
  - *i.e.* calculate a $\widehat{\mathbf{Y}_{0i}}$ for each $\mathbf{Y}_{1i}$, based upon "matched" untreated individuals with (nearly) identical values of $\boldsymbol{X}_i$
- If CIA holds, then we can just calculate a bunch of treatment effects conditional on $\mathbf{X}_i$
  - *i.e.*

$$\tau(x) = E[\mathrm{Y}_{1i} - \mathrm{Y}_{0i} \mid \mathrm{X}_i = x]$$

Matching

More formally:

We want to construct a counterfactual for each individual with $\mathbf{D}_i = 1$.

CIA: The counterfactual for i should only use individuals that match $\mathbf{X}_i$.

Let there be N_T treated individuals and N_C control individuals. We want

- N_T sets of weights
- with N_C weights in each set:
  $w_i(j) \;\; (i = 1, \ldots, N_T;\, j = 1, \ldots, N_C)$

Assume $\sum_j w_i(j) = 1$. Our estimate for the counterfactual of treated i is

$$\widehat{\mathrm{Y}_{0i}} = \sum_{j \in (D=0)} w_i(j) \mathrm{Y}_j$$

Weight for it

So all we need is those weights and we're done.

Q: Where does one find these handy weights?

A: You've got options, but you need to choose carefully/responsibly.

E.g. if $w_i(j) = \frac{1}{N_C}$ for all (i, j), then we're back to a difference in means.
This weighting doesn't abide by our conditional independence assumption.

The plan: choose weights $w_i(j)$ that indicate ***how close*** $\mathbf{X}_j$ is to $\mathbf{X}_i$.

Weight for it

Some common choice of weights:

- **Nearest neighbor:**

$$\mathrm{d}_{i,j} = \left(\mathrm{X}_i - \mathrm{X}_j\right)' \left(\mathrm{X}_i - \mathrm{X}_j\right)$$

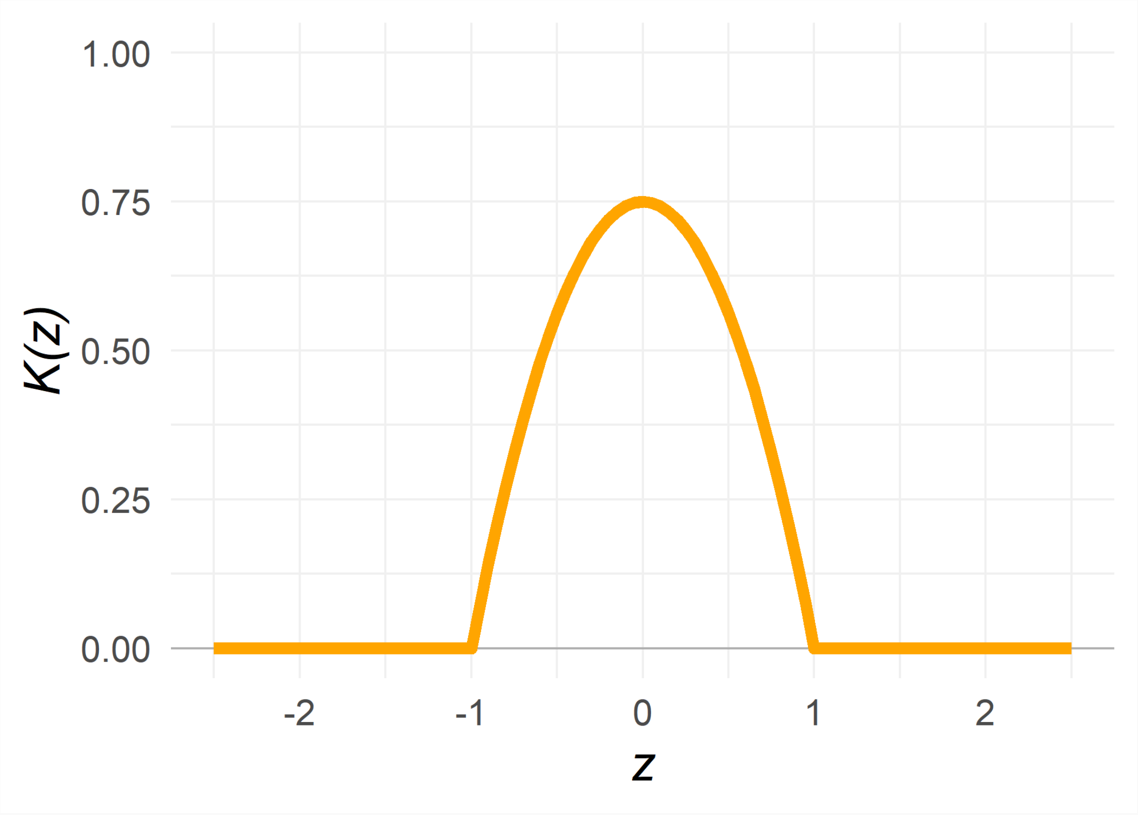
- **Kernel Matching** for **bandwidth** h and **kernel function** $K(\cdot)$:

$$w_i(j) = \frac{K\left(\dfrac{\mathrm{X}_j - \mathrm{X}_i}{h}\right)}{\sum_{j \in (D=0)} K\left(\dfrac{\mathrm{X}_j - \mathrm{X}_i}{h}\right)}$$

Kernels

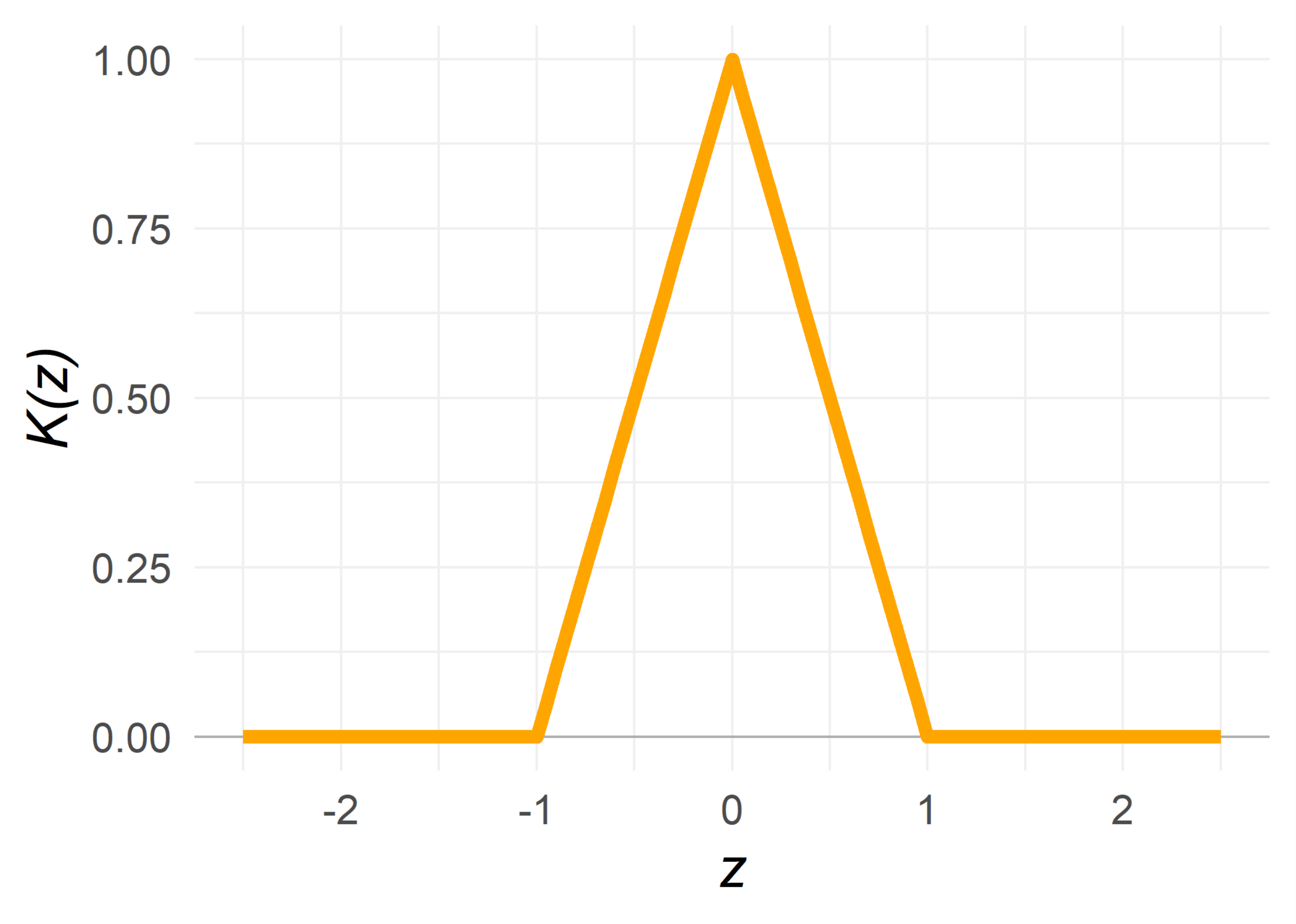
For example, the *Epanechnikov kernel* is defined as

$$K(z) = \frac{3}{4}\left(1 - z^2\right) \times \mathbb{I}(|z| < 1)$$



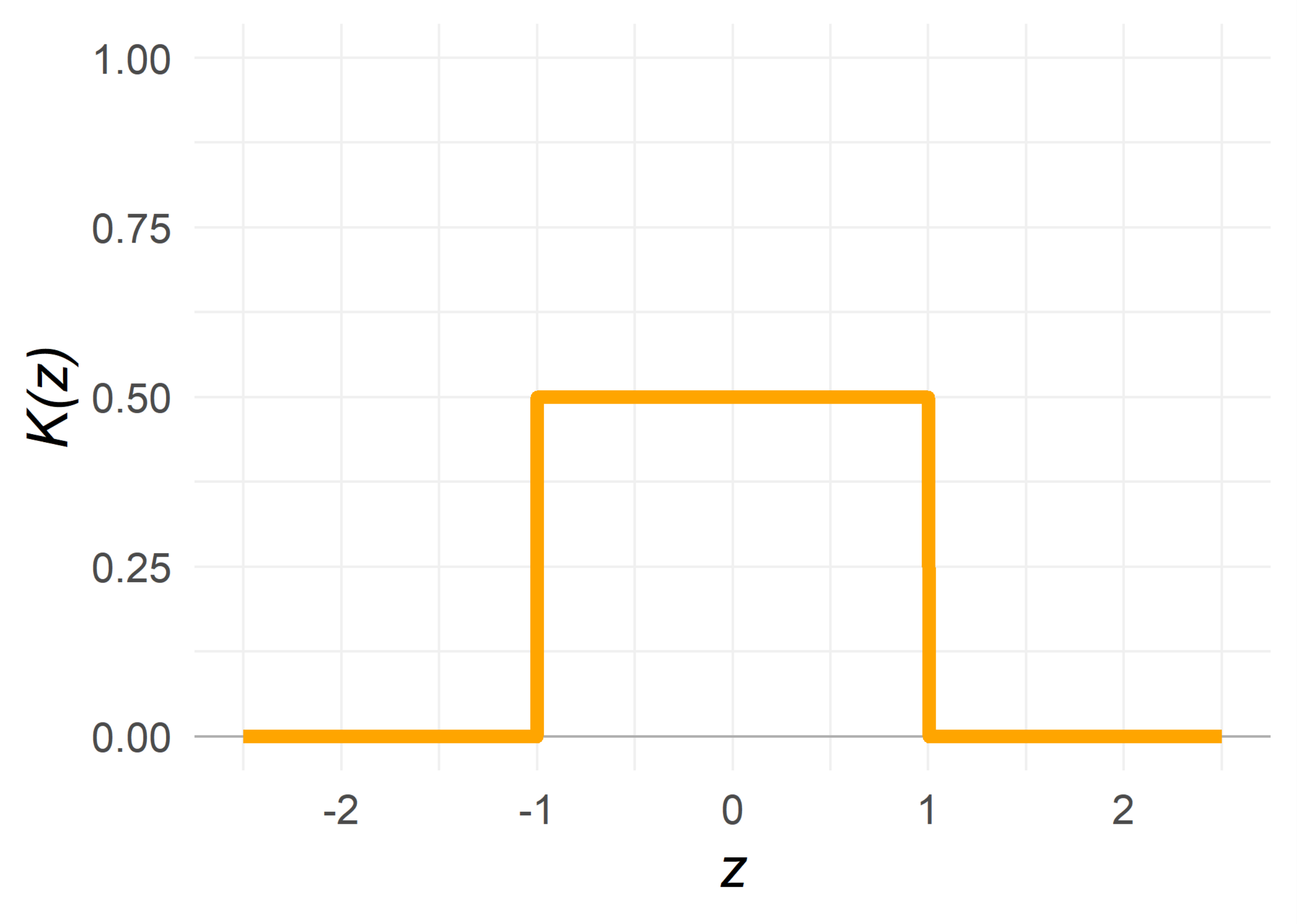
Kernels

And the *triangular kernel* can be expressed as
$K(z) = (1 - |z|) \times \mathbb{I}(|z| < 1)$



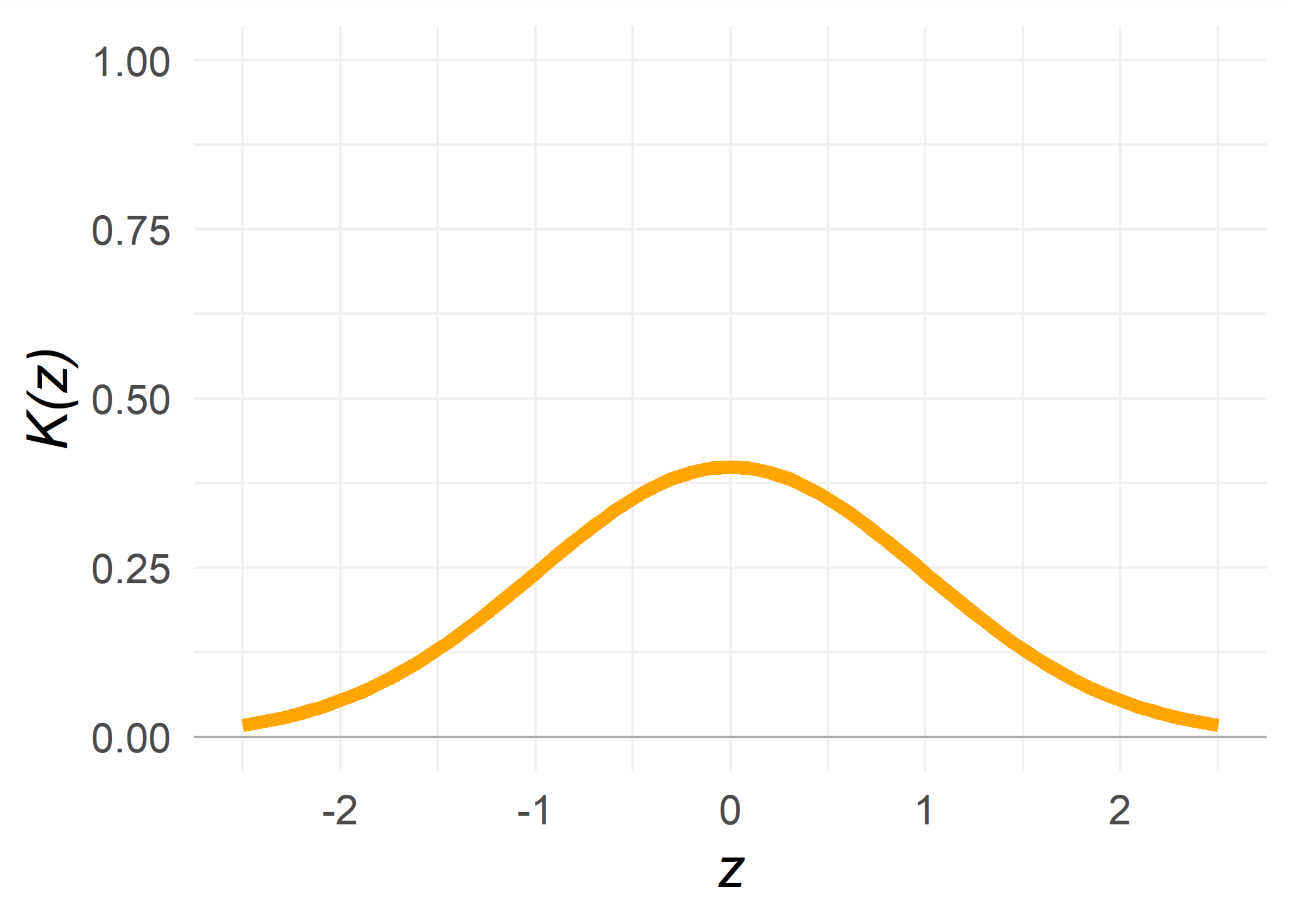
Kernels

And the *uniform kernel* with $K(z) = \frac{1}{2} \times \mathbb{I}(|z| < 1)$



Kernels

Or the *Gaussian kernel* $K(z) = (2\pi)^{-1/2} \exp\left(-z^2/2\right)$



Aside on Kernels

Kernel functions are good for more than just matching.

You will most commonly see/use them smoothing out densities—providing a smooth, moving-window average.

E.g. R's (`ggplot2`'s) smooth, density-plotting function `geom_density()`.

`geom_density()` defaults to `kernel = "gaussian"`, but you can specify many other kernel functions (including `"epanechnikov"`).

You can also change the `bandwidth` argument. The default is a bandwidth-choosing function called `bw.nrd0()`.

Canonical Synthetic Control

Canonical Synthetic Control

The **canonical synthetic control method** feels a lot like if **event study** and a **matching estimator** got together and had a kid.

- Originated in **Abadie and Gardeazabal (2003) AER**, refined and extended in **Abadie, Diamond, and Hainmueller (2010) JASA**
- Developed for **comparative case studies:** one aggregate unit exposed to treatment/intervention

The gist: compare post-treatment outcome evolution in treated group to a **synthetic control** unit constructed to match on

- Pre-trends in the outcome variable $\mathbf{Y}_i$
- Covariates $\mathbf{X}_i$ or $\mathbf{X}_{it}$

Canonical Synthetic Control

Synthetic control overcomes one of the key issues of studying aggregate units: **few, poor-counterfactual controls**

- Policy interventions often happen at an aggregate level (i.e. state, country)
- Aggregate/macro data are often easy to obtain

However,

- Finding a valid counterfactual with coarse, aggregate units can be difficult
- Control group selection is ad hoc, leading to *researcher degrees of freedom*

Canonical Synthetic Control

Formally:

- Suppose you have data for $J + 1$ units
  - Treated unit: $j = 1$
  - "Donor Pool": all $j = 2, \ldots, J + 1$ units
- Data span T periods, with T_0 periods prior to treatment

For each unit, we observe

1. The outcome of interest Y_{jt}
2. A set of k predictors of the outcome, $X_{1j}, \ldots, X_{kj}$
   - May include pre-intervention values of Y_{jt}
   - Must be *unaffected by the intervention*

Canonical Synthetic Control

Formally:

For each unit j, let Y_{jt}^{N} be the potential response without intervention, with Y_{1t}^{I} the potential response under intervention for the exposed unit

- For unit "one" with $t > T_0$, we have $Y_{1t} = Y_{1t}^{I}$

Under this setup, the effect of the policy in period t is given by

$$\tau_{1t} = Y_{1t}^{I} - Y_{1t}^{N}$$

Policy Evaluation Challenge: how to estimate Y_{1t}^{N}, the unobserved counterfactual?

Canonical Synthetic Control

A: Construct a **synthetic control** as a weighted average of units in the donor pool.

Let $W = (\omega_2, \ldots, \omega_{J+1})'$ be a $J \times 1$ vector of weights.

For a given W, the synthetic control counterfactual is

$$\hat{Y}_{1t}^{N} = \sum_{j=1}^{J+1} \omega_j Y_{jt}$$

and

$$\hat{\tau}_{1t} = Y_{1t}^{I} - \hat{Y}_{1t}^{N}$$

Choosing Weights

Weights are designed to **avoid extrapolation**

- $\omega_j \geq 0 \;\; \forall \, j$
- $\sum_{j=1}^{J+1} \omega_j = 1$
- Ensures synthetic control is located within the convex hull of donor units (based purely on observed data)

We will choose the ω_j so that the synthetic control best matches **pre-intervention values for the treated unit of predictors for the outcome variable**.

Choosing Weights

That is, choose weights W^* that minimize

$$||X_1 - X_0W|| = (\sum_{h=1}^{k} \nu_h (X_{h1} - \omega_2 X_{h2} - \ldots - \omega_{J+1} X_{hJ+1})^2)^{1/2}$$

- Positive constants $\nu_1 \ldots \nu_k$ reflect the **relative importance** put on predictors $1, \ldots k$
- Abadie, Diamond, and Hainmueller (2010): select $\nu_1 \ldots \nu_k$ to minimize mean square prediction error (MSPE) for some set of pre-intervention periods
- Abadie, Diamond, and Hainmueller (2015): select $\nu_1 \ldots \nu_k$ via out-of-sample validation
  1. Divide pre-intervention period into .hi-medgrn[training] and .hi-purple[validation] periods
  2. Select a value of $V^* = \nu_1^* \ldots \nu_k^*$ that yields a small MSPE in the validation period

German Reunification

Let's load in some state-by-year data on GDP and other economic conditions:

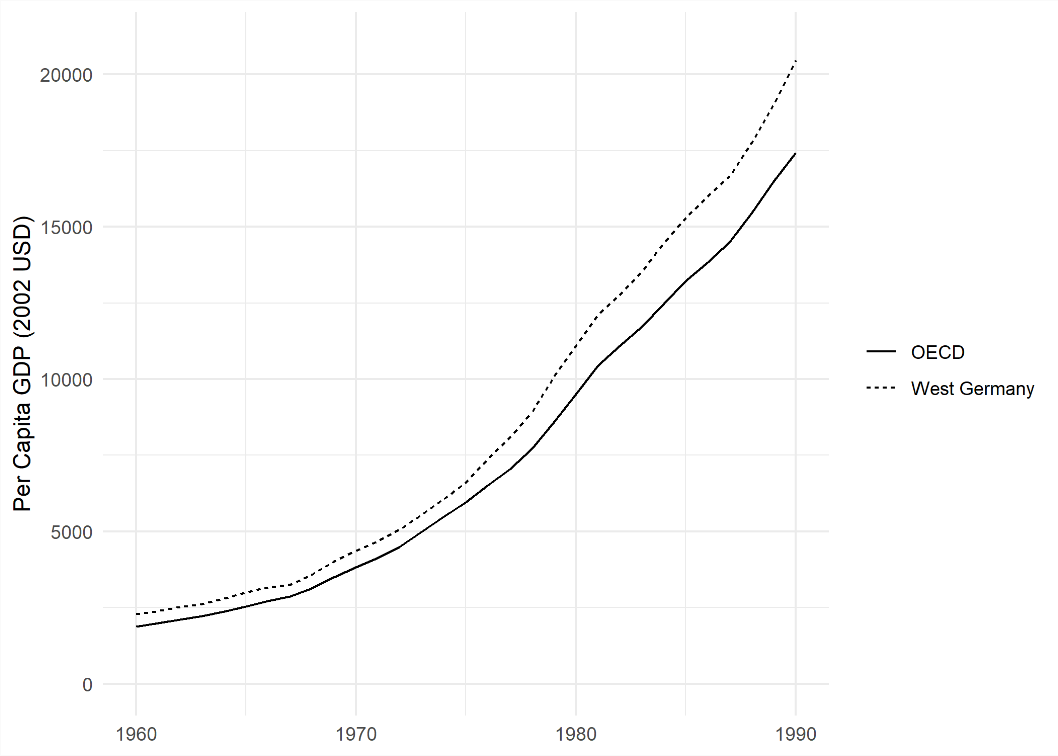
```
deu <- haven::read_dta("data/repgermany.dta") %>%
    mutate_at(vars(year, gdp, infrate, trade, schooling,
                   invest60, invest70, invest80,
                   industry),
              as.numeric) %>%
    mutate_at(vars(index, country), as.factor)

deu <- haven::read_dta("data/repgermany.dta") %>%
    mutate_at(vars(index, year, gdp, infrate, trade, schooling,
                   invest60, invest70, invest80,
                   industry),
              as.numeric) %>%
    mutate_at(vars(country), as.character)
head(deu)
```

```
## # A tibble: 6 × 11
##   index country  year   gdp infrate trade schooling invest60 invest70 invest
```

German Reunification

How did West Germany GDP compare to OECD countries prior to reunification?

- *Spoiler*: that gap looks to be growing



German Reunification

What if we construct a "synthetic" West Germany to match on pre-unification predictors of economic growth?

- GDP (average for 1980-1990)
- Trade openness: Exports + Imports as % of GDP (average for 1980-1990)
- Inflation Rate (average for 1980-1990)
- Industry share of value-added (average 1981-1989)
- Schooling: % of secondary school attained in the age 25+ population (average 1980 and 1985)
- Investment rate: ratio of real domestic investment (private + public) to real GDP (average 1980-84)

Let's use the **tidysynth** package to do this in a *tidy* workflow

German Reunification

First, let's set up the synthetic control object with `synthetic_control()`

```
synth_wg <- deu %>%
    synthetic_control(
      outcome = gdp,
      unit = country,
      time = year,
      i_unit = "West Germany", # treated unit
      i_time = 1990, # treatment year
      generate_placebos = T # whether to generate placebos for inference
    )
```

German Reunification

Next, add the predictors with `generate_predictor()`

- Choose a time period for matching
- Choose the variables to use
- Choose the summary method

```
synth_wg <- synth_wg %>%
  generate_predictor(time_window = 1981:1990,
                     gdp_81_90 = mean(gdp, na.rm = T),
                     trade81_90 = mean(trade, na.rm = T),
                     infrate81_90 = mean(infrate, na.rm = T)
  ) %>%
  generate_predictor(time_window = 1971:1980,
                     industry_71_80 = mean(industry, na.rm = T)) %>%
  generate_predictor(time_window =c(1970, 1975),
                     schooling_70_75 = mean(schooling, na.rm = T)) %>%
  generate_predictor(time_window = 1980,
                     invest_80 = invest80)
```

German Reunification

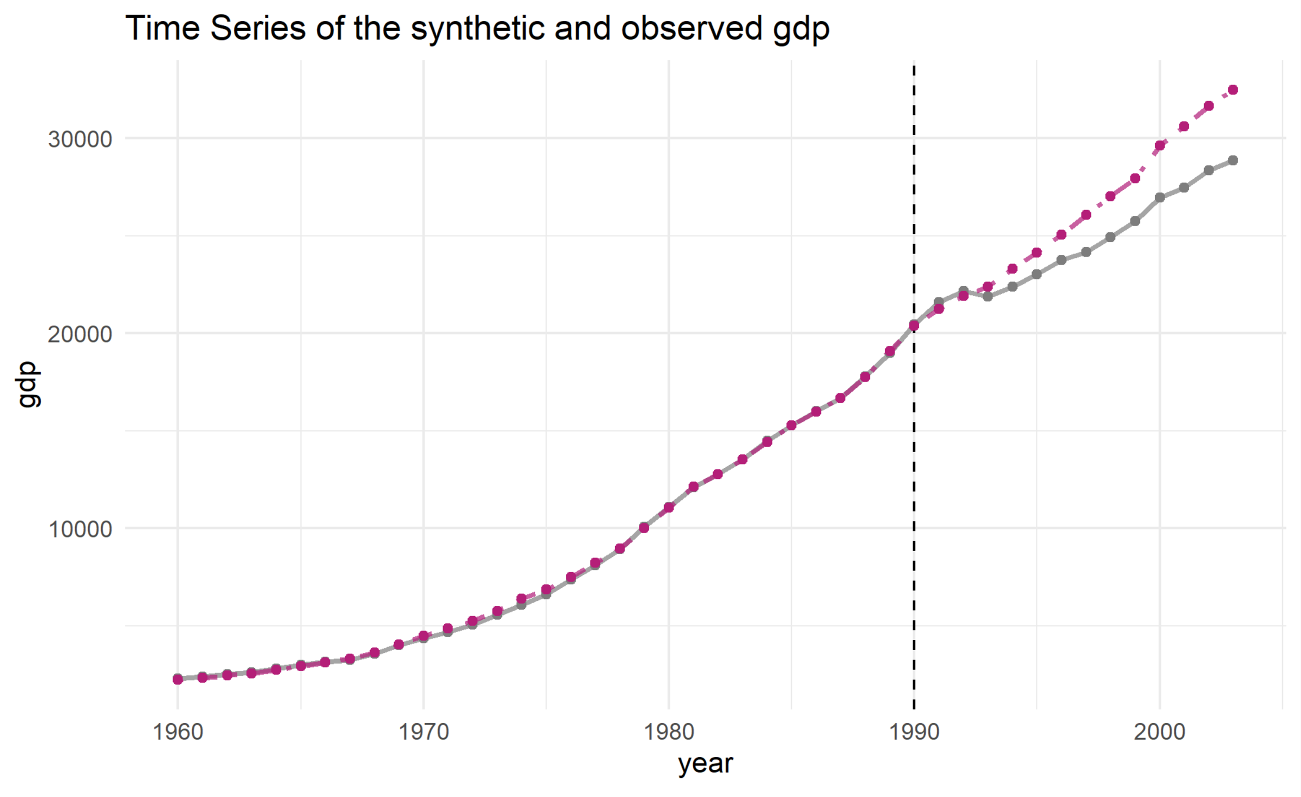
Next, generate weights with `generate_weights()`

```
wts <- synth_wg %>%
  generate_weights(optimization_window = 1981:1990)

# get variable weights
wt_vec <- wts[[7]][[1]] %>%
  select(weight) %>% as.vector() %>% unlist()
```

German Reunification

Finally, estimate the synthetic control and plot it

```
synth_control ← generate_control(wts)

plot_trends(synth_control)
```



German Reunification

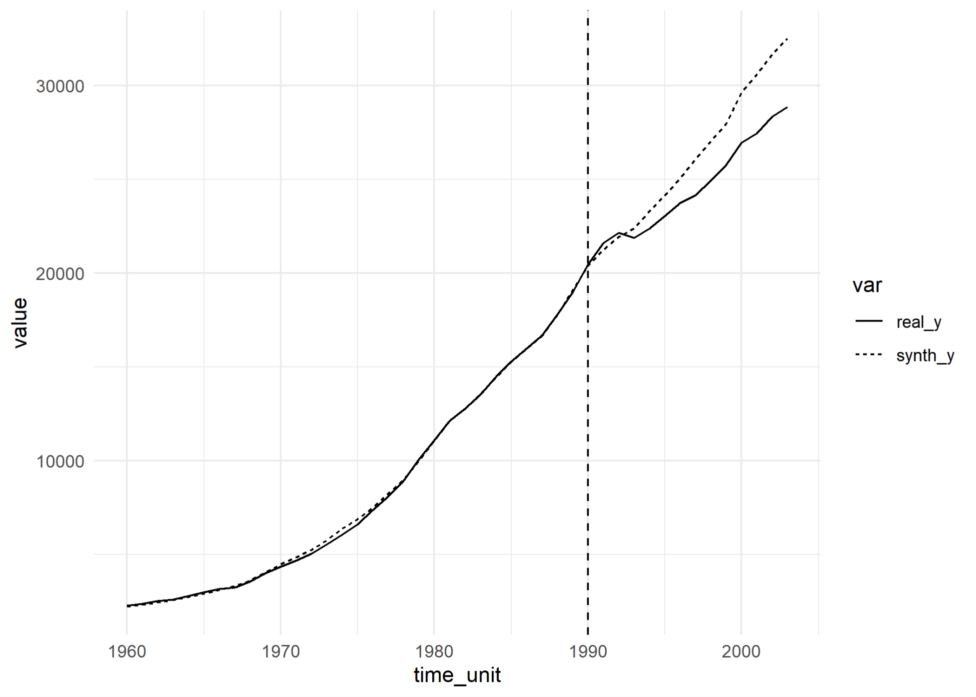
Alternatively, extract the synthetic control + treated unit values for plotting:

```
grab_synthetic_control(synth_control) %>% head()
```

```
## # A tibble: 6 × 3
##   time_unit real_y synth_y
##       <dbl>  <dbl>   <dbl>
## 1      1960   2284   2238.
## 2      1961   2388   2319.
## 3      1962   2527   2457.
## 4      1963   2610   2565.
## 5      1964   2806   2751.
## 6      1965   3005   2918.
```

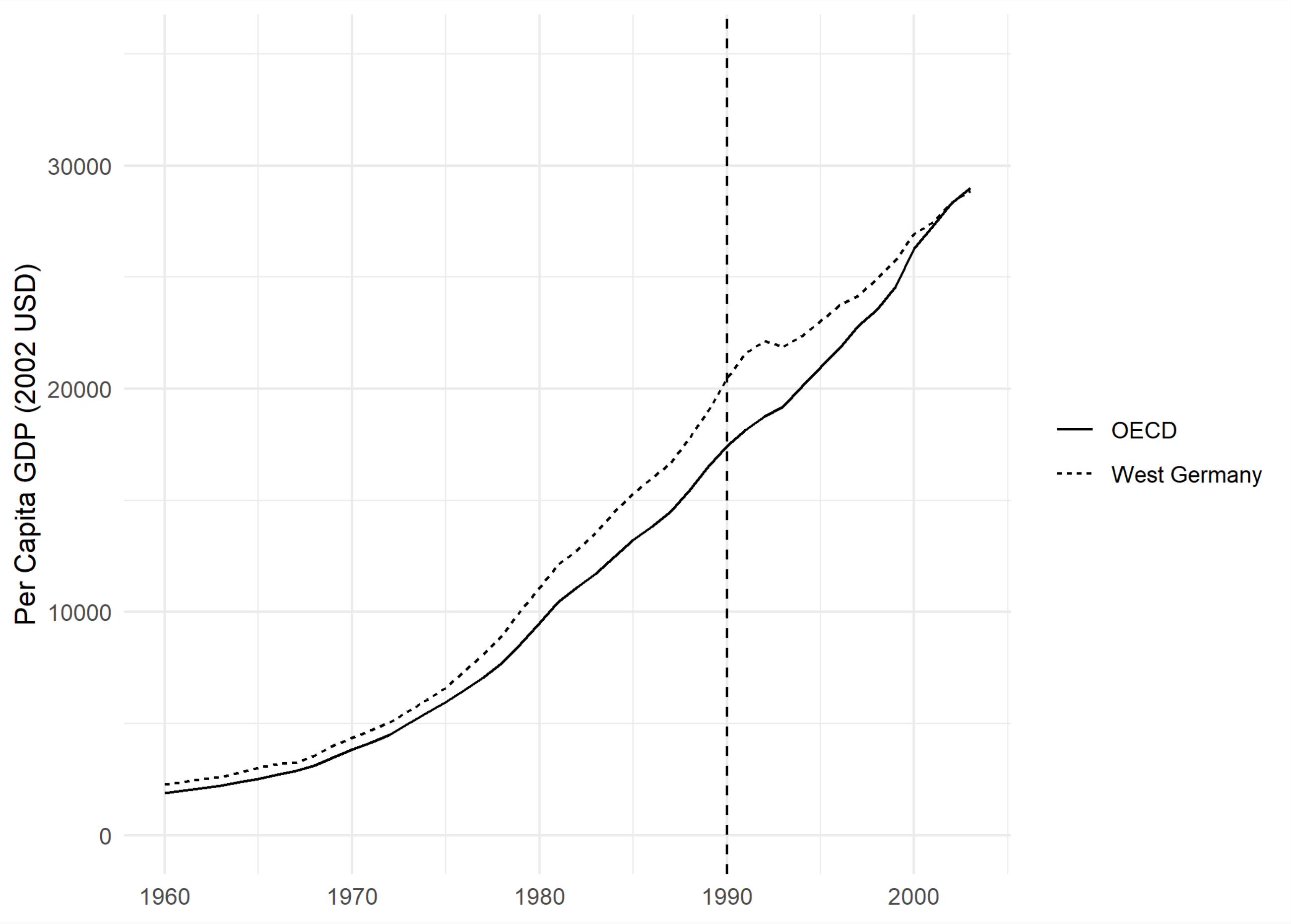
German Reunification

```
grab_synthetic_control(synth_control) %>%
  pivot_longer(cols = ends_with("y"), names_to = "var") %>%
  ggplot(aes(x = time_unit)) +
  geom_line(aes(y = value, linetype = var)) +
    geom_vline(aes(xintercept = 1990), linetype = "dashed") +
  theme_minimal()
```



German Reunification

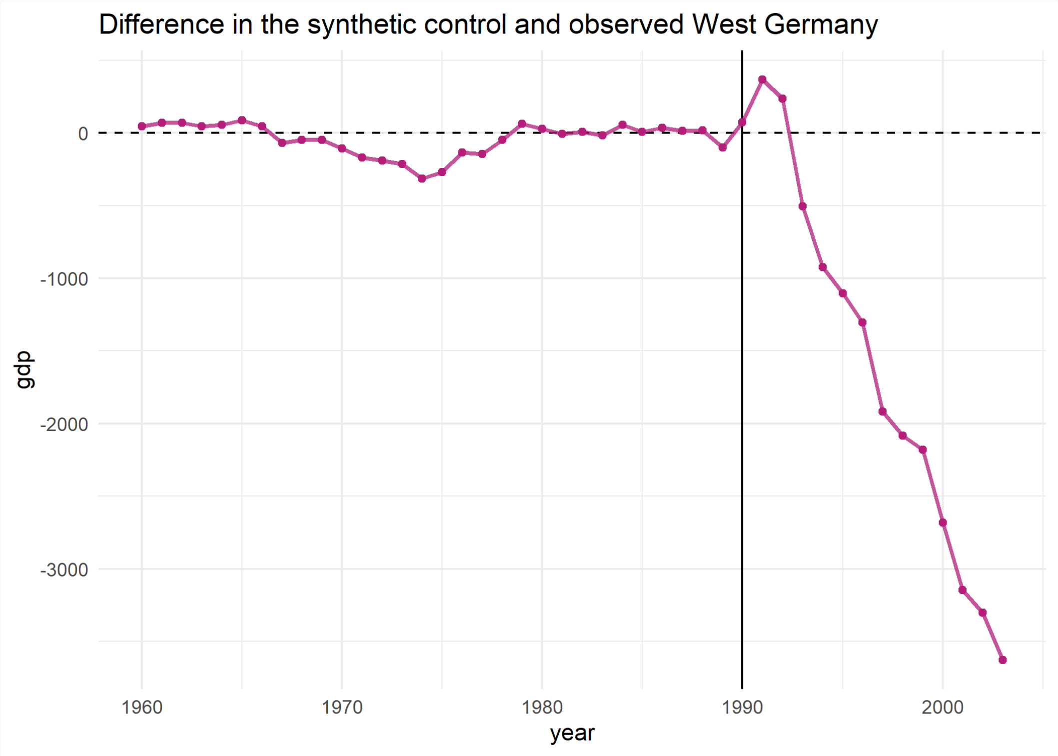
Comparing to the raw mean of OECD countries:



German Reunification

Alernatively we can plot the **difference** between West Germany and its synthetic control:

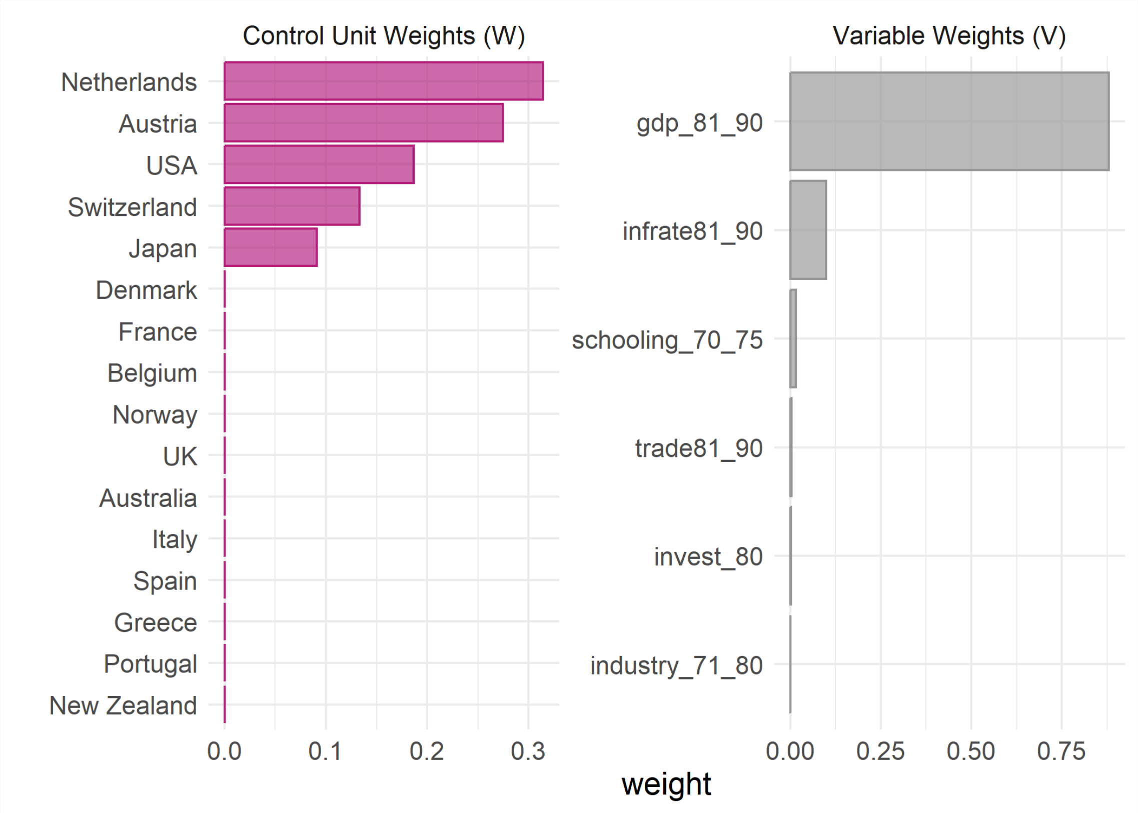
```
plot_differences(synth_control)
```



German Reunification

Looking at the weights:

```
synth_control %>%
  plot_weights()
```



German Reunification

Checking balance of real West Germany vs. Synthetic West Germany vs. Mean of OECD Countries:

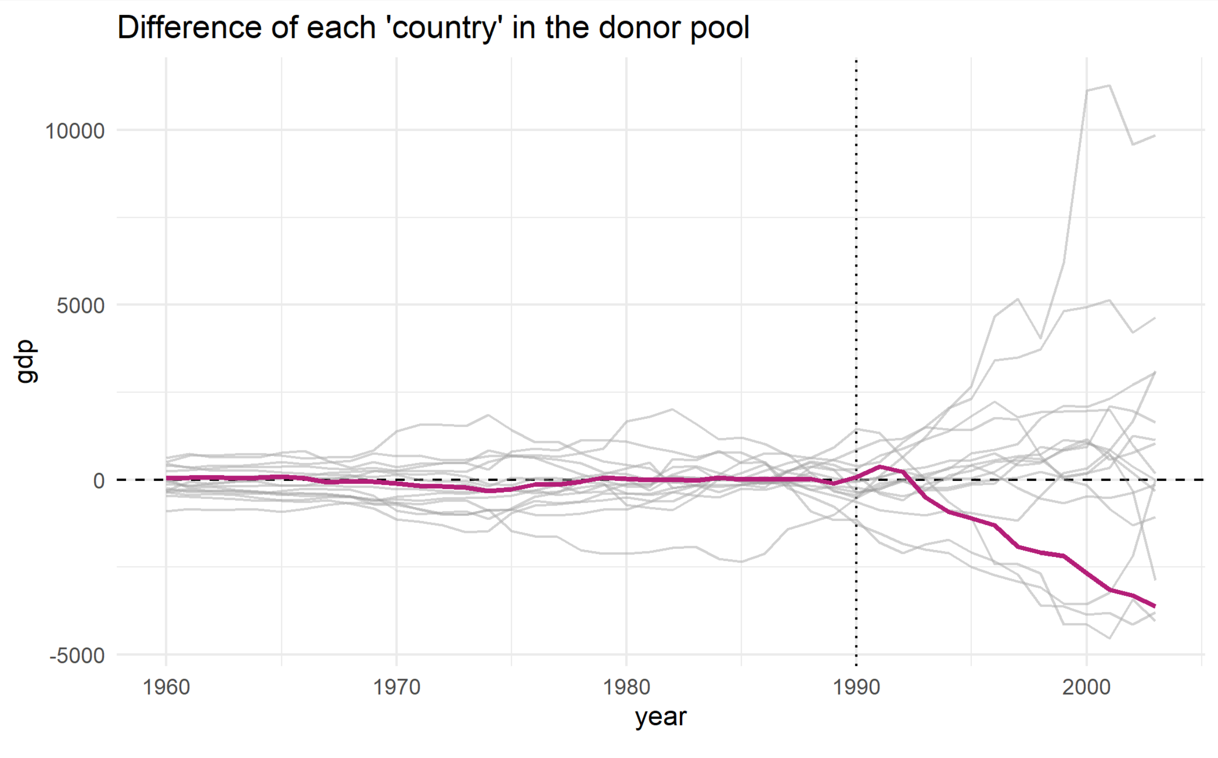
```
synth_control %>%
  grab_balance_table()
```

```
## # A tibble: 6 × 4
##   variable         `West Germany` `synthetic_West Germany` donor_sample
##   <chr>                     <dbl>                    <dbl>        <dbl>
## 1 gdp_81_90              15809.                   15800.       13669.
## 2 infrate81_90               2.59                     3.30         7.62
## 3 trade81_90                56.8                     69.1         59.8
## 4 industry_71_80            43.9                     37.1         36.9
## 5 schooling_70_75           51.9                     43.1         32.5
## 6 invest_80                 27.0                     25.7         25.9
```

German Reunification

For inference, we repeat the same process as before with every unit in the donor pool.

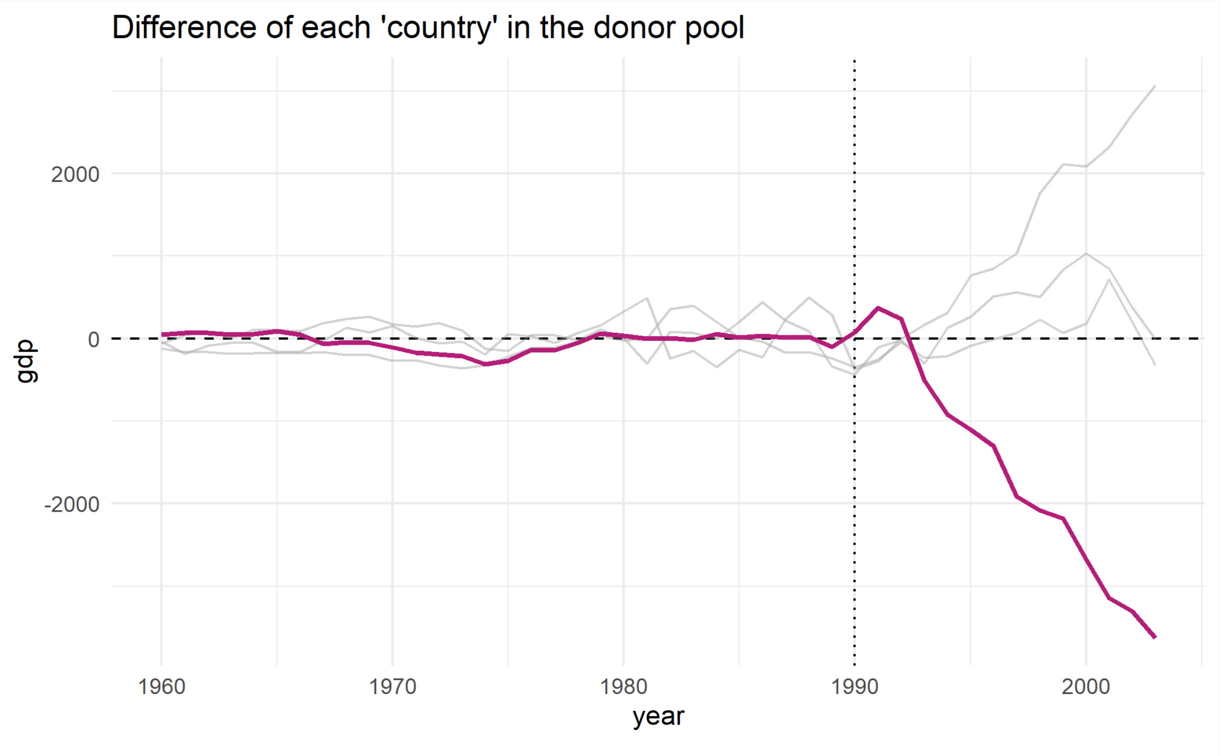
```
synth_control %>%
  plot_placebos(prune = FALSE)
```



German Reunification

By default, `plot_placebos()` hides the placebo controls with large MSPEs (here we only get 3)

```
synth_control %>%
  plot_placebos()
```



Inference

Finally, looking at inference:

```
wg_inf ← synth_control %>%
  grab_significance()
wg_inf
```

```
## # A tibble: 17 × 8
##    unit_name     type     pre_mspe post_mspe mspe_ratio  rank fishers_exact_pv
##    <chr>         <chr>       <dbl>     <dbl>      <dbl> <int>                <
##  1 West Germany Treated    12771.   4474135.     350.       1               0.
##  2 Norway        Donor    583094. 42826838.      73.4       2               0.
##  3 Australia     Donor     44373.   2816702.     63.5       3               0.
##  4 USA           Donor    331001. 12654685.      38.2       4               0.
##  5 New Zealand   Donor    388407.   9052988.     23.3       5               0.
##  6 Greece        Donor    410485.   6625701.     16.1       6               0.
##  7 Spain         Donor    122462.   1569470.     12.8       7               0.
##  8 Italy         Donor    297045.   2505509.      8.43      8               0.
##  9 Denmark       Donor     34078.    280217.      8.22      9               0.
## 10 Switzerland   Donor   1399771.   8117976.      5.80     10               0.
## 11 Netherlands   Donor    230567.   1094375.      4.75     11               0.
```

Inference

```
colnames(wg_inf)
```

```
## [1] "unit_name"              "type"                   "pre_mspe"
## [4] "post_mspe"              "mspe_ratio"             "rank"
## [7] "fishers_exact_pvalue" "z_score"
```

Inference with synthetic controls is based on the difference between pre and post-intervention MSPE values.

Idea: if the synthetic control fits the observed data well (low pre-intervention MSPE), and diverges in the post-period (high post-period MSPE), then the intervention had a meaningful effect.

- If the intervention had *no* effect, the pre and post-period MSPE should be similar, with a ratio around 1
- If placebos fit the data as well as the treated unit, we can't reject the null of no treatment effect

Inference

Fisher's exact P-value is generated by first ranking ratios then dividing the rank of the case over the total

```
unique_countries <- unique(deu$country) %>% length()

# Fisher's P calculated as rank/total, so for West Germany (rank 1):
1/unique_countries
```

```
## [1] 0.05882353
```

Z-score is then the standardized RMSE ratios for all cases.

- Captures degree to which a particular case's RMSE ratio deviates from the placebo distribution

Choice of Predictors

One challenge remaining for the researcher is the **definition of predictors**

- Which predictors to use

- Which years to match on

Ferman, Pinto, and Possebom (2020) go into great detail regarding how to properly select specifications of synthetic controls. Their punchline:

- Models including more pre-treatment outcome lags as predictors are better at controlling for unobserved confounders
- The possibilities for "specification searching" are higher with more pre-treatment periods used for matching
- **Best:** present multiple results under common specifications
  - If the result is robust to these different predictor choices, then the preferred specification isn't cherry-picked!

Table of Contents