

QM - Computer Lab 4

In this computer lab we will have to achieve the following tasks/learning outcomes:

- import data
- understand some important features of the dataset
- select a subset of data
- estimate a Diff-in-Diff estimator to evaluate the effectiveness of a sugar tax

The example is the example discussed in Demonstration Class 2 and comes from the Doing Economics Project 3 which is based on the work in this paper:

These questions refer to the work presented in:

Silver LD, Ng SW, Ryan-Ibarra S, Taillie LS, Induni M, Miles DR, et al. (2017) Changes in prices, sales, consumer spending, and beverage consumption one year after a tax on sugar-sweetened beverages in Berkeley, California, US: A before-and-after study. PLoS Med 14(4): e1002283. <https://doi.org/10.1371/journal.pmed.1002283>

Preparing your workfile

We add the basic libraries needed for this week's work:

```
library(tidyverse)    # for almost all data handling tasks
library(ggplot2)      # to produce nice graphics
library(stargazer)    # to produce nice results tables
library(haven)        # to import stata file
library(AER)          # access to HS robust standard errors
library(readxl)       # enable the read_excel function
library(plm)          # enable panel data methods
```

You should also save the separately supplied `stargazer_HC.r` file in your working directory. This will make it straightforward to estimate and compare regressions with robust standard errors. Once you have done that you should include the following line into your code which basically makes this function available to you.

```
source("stargazer_HC.r") # includes the robust regression
```

We will first look at price data from the treatment group (stores in Berkeley) to see what happened to the price of sugary and non-sugary beverages after the tax.

- Download the data from the Global Food Research Program's website, and select the 'Berkeley Store Price Survey' Excel dataset. Then upload the dataset into R.
- The first tab of the Excel file contains the data dictionary. Make sure you read the data description column carefully, and check that each variable is in the Data tab.

Data Import and tidying

The authors use various data sources. The data used here are the data from the store survey in which prices for a range of beverages are observed in stores in Berkeley at three different points in time.

This is an `xlsx` file and we use the `readxl` package to import this file. We also install (if you haven't done so yet) and load the `tidyverse` library as this includes packages that we will use later (piping and graphing). There are two worksheets, one with some info on the variables ("Data Dictionary") and the other with the data ("Data"). It is useful to import both into R so that you have the additional data info available while you are working without having to go back into EXCEL.

Either use the R help function or google to find out how to use the `sheet =` option inside the `read_excel` function.

```
var_info <- XXXX("sps_public.xlsx", XXXX="Data Dictionary")
dat <- XXXX("sps_public.xlsx", XXXX="Data")
```

Let's look at the datatypes this import process has assigned to the respective variables.

```
str(XXXX)
```

Confirm that all variables which should be numbers, in particular prices, are indeed `num` variables. But then there are variables that may be useful to have as categorical (factor in R speak) variables. Namely `type`, `taxed`, `supp`, `store_id`, `store_type`, `type2` and `product_id`. All these variables have been expressed as numbers but really are categorical variables. So let's convert them to factor variables.

The following code converts all but the `store_type` variable. Where the data are initially `num` we supply a `label` option which gives sensible names to the different levels.

```
dat$type <- factor(dat$type)
dat$taxed <- factor(dat$taxed,
                    labels=c("not taxed", "taxed"))
dat$supp <- factor(dat$supp,
                    labels=c("Standard", "Supplemental"))
dat$store_id <- factor(dat$store_id)
dat$type2 <- factor(dat$type2)
dat$product_id <- factor(dat$product_id)
```

Before converting the `store_type` variable, let's see what information we have on this variable

```
var_info[var_info$`Variable Name` == "store_type",]
```

```
## # A tibble: 1 x 3
##   `Variable Name` Type Description
##   <chr>           <chr> <chr>
## 1 store_type      Num   Store Type: 1(Large Supermarket), 2(Small Supermarket),~
```

Let's change the `store_type` variable to a factor.

```
dat$store_type <- XXXX(dat$store_type,
                       labels=c("Large Supermarket",
                                XXXX,
                                XXXX,
                                XXXX))
```

There is another variable `time` which is characterised as a `chr` variable. This stands for characters as in letters and numbers. It may be useful to change this also into a factor variable. Before we do this we will check which values this variable takes by using the `unique` command.

```
unique(dat$time)
```

```
## [1] "DEC2014" "JUN2015" "MAR2015"
```

If you read the Silver et al. (2016) paper, and in particular if you have looked at the timeline, then you will have realised that the third survey wasn't in March 2015 but in March 2016. So the data have been labelled incorrectly. We shall therefore change all the values "MAR2015" to "MAR1016".

```
dat$time[XXXX == "MAR2015"] <- XXXX
```

We can now change the time variable into a factor variable.

```
dat$time <- factor(dat$time)
```

Understanding the data

How many different stores and how many different products have been included in the survey? There are different ways to achieve this. One way is to use the `unique` function in combination with the `length` function.

Confirm the following:

```
## [1] "Number of Stores: 26"
```

```
## [1] "Number of Products: 247"
```

Execute the following command and interpret its outcome. You may want to use the help function or google to understand what the `tally()` function does in this pipe.

```
table1 <- dat %>% group_by(store_type,taxed,time) %>%  
  tally() %>%  
  spread(time,n) %>%  
  print(n=Inf)
```

Now use the `tally` function to produce a frequency table showing the number of each product type (`type`), with product type (`type`) as the row variable and time period (`time`) as the column variable.

```
table2 <- XXXX %>% group_by(XXXX,XXXX) %>%  
  tally() %>%  
  spread(XXXX,n) %>%  
  print(n=Inf)
```

```
## # A tibble: 13 x 4  
## # Groups:   type [13]  
##   type      DEC2014 JUN2015 MAR2016  
##   <fct>      <int>    <int>    <int>  
## 1 ENERGY          56         58         49  
## 2 ENERGY-DIET     49         54         35  
## 3 JUICE            70         64         52  
## 4 JUICE DRINK      19         17          6  
## 5 MILK             63         61         53  
## 6 SODA            239        262        215  
## 7 SODA-DIET       128        174        127  
## 8 SPORT           11         16         12  
## 9 SPORT-DIET        2          2         NA  
## 10 TEA            52         45         41  
## 11 TEA-DIET         6          6          8  
## 12 WATER          48         38         34  
## 13 WATER-SWEET      1          1          1
```

Can you confirm that we have, altogether 177 observations for milk?

To explore the data a bit further let's look at all observations for the "TEA-DIET" category. In particular we want to see the `product_id`, `store_id` and `time` for these observations and only those from either "DEC2014" or "MAR2016" (should be 14). It would also be best to sort the data according to `store_id`.

```
dat %>% XXXX(type == XXXX & time %in% c(XXXX, XXXX)) %>%
  select(store_id, product_id, time) %>%
  arrange(XXXX) %>%
  print()
```

For how many stores do you have price observations in that product category for both these periods? Only two stores (`store_id` 14 and 24) have observations from both periods, and only in store 24 did we record the price for the same `product_id`.

In order to establish whether prices change following the tax introduction we want to restrict our data to only those products (identified by `product_id`) for which we have price observations for all three periods (DEC2014, JUN2015 and MAR2016) from the same store. It is best to create a new variable `period_test` which takes the value 1 (or TRUE) if we have observations for all periods for a particular product in a particular store and 0 (FALSE) otherwise. We call variables like this Boolean variables.

It turns out that this is not a task that is straightforward and the easiest way to achieve this is with a loop. Execute the code below. It is not crucial to understand everything now, but do, later, take some time to understand what is going on.

```
dat$period_test <- NA

sid_list = unique(dat$store_id) # list of all store ids
pid_list = unique(dat$product_id) # list of all product ids

for (s in sid_list) {
  for (p in pid_list) {
    temp <- subset(dat, product_id == p & store_id == s)
    temp_time <- temp$time
    # test will take value TRUE if we have obs from all three periods, FALSE otherwise
    test <- (any(temp_time == "DEC2014") & any(temp_time == "JUN2015") & any(temp_time == "MAR2016"))

    dat$period_test[dat$product_id == p & dat$store_id == s] <- test
  }
}
```

Now we are in a position to remove all products that have not been observed in all three periods. We define a new data frame `dat_c` with only products that are observed in all three periods in a particular store. Also, we will only select observations for which the variable `supp` takes the value “Standard”. This is what we then call a balanced panel.

```
dat_c <- dat %>% XXXX(period_test == XXXX & supp == XXXX)
```

You should confirm that you are left with 939 observations.

Now we can calculate the price averages (for `price_per_oz_c`, which is the price per ounce in cents) by grouping the data according to `store_type`, `taxed` and `time`. One way to achieve this is to use piping:

```
table_res <- dat_c %>%
  group_by(taxed, store_type, time) %>%
  summarise(n = length(price_per_oz_c), avg.price = mean(price_per_oz_c)) %>%
  spread(time, avg.price) %>%
  print()
```

```
## # A tibble: 8 x 6
## # Groups:   taxed, store_type [8]
##   taxed   store_type           n DEC2014 JUN2015 MAR2016
```

	<fct>	<fct>	<int>	<dbl>	<dbl>	<dbl>
## 1	not	taxed Large Supermarket	36	11.2	11.5	11.7
## 2	not	taxed Small Supermarket	70	13.7	13.8	13.4
## 3	not	taxed Pharmacy	18	15.2	16.1	15.4
## 4	not	taxed Gas Station	12	16.9	17.0	17.0
## 5	taxed	Large Supermarket	36	15.6	16.9	16.7
## 6	taxed	Small Supermarket	101	15.9	16.0	15.5
## 7	taxed	Pharmacy	18	18.2	19.1	18.6
## 8	taxed	Gas Station	22	19.4	20.3	19.2

Calculating the DiD estimator

We now want to treat these as a panel dataset. You may remember that we need to define a panel dataset and let R know what variables identifies an “individual” and what variable defines the time dimension. Here an “individual” or the “unit of observation” is a “particular product in a particular store”. There is no variable which can serve as such an index, but we can create it by concatenating the `store_id` and `product_id` variables.

```
dat_c$sp_id <- paste0(dat_c$store_id,"_",dat_c$product_id)
```

Have a look at values of the new variable to understand what happened. We will also have to create the interaction variable between `time` and `taxed`.

```
dat_c$did <- (dat_c$time == "MAR2016") * (dat_c$taxed == "taxed")
```

This variable will take the value of 1 for all observations which are from “MAR2016” and for taxed products. All other observations will have a value of 0.

Let’s restrict the analysis to large supermarket data and the two time periods “DEC2014” and “MAR2016”.

```
dat_c_ls <- XXXX %>% XXXX(XXXX == XXXX & XXXX %in% XXXX)
```

You should be left with 144 observations.

Then we can use this new variable `sp_id` together with the `time` variable in the definition of the panel data frame:

```
p_dat_c_ls <- pdata.frame(dat_c_ls, index = c("sp_id","time"))
```

Now we specify the regression model from which we estimate the DiD estimator.

$$y_{it} = \beta_0 + \delta d_t + \tau T_{it} + a_i + u_{it} \quad (1)$$

where y_{it} is `price_per_oz_c`, d_t is `time`, T_{it} is `did` and a_i is our indicator variable for the individual (store-product combination), `sp_id`. This is estimated in “first-differenced” form (`model = "fd"`) using the panel estimation function `plm`.

```
mod1 <- plm(XXXX ~ XXXX + XXXX + XXXX, data = XXXX, model = "fd")
stargazer_HC(mod1)
```

```
##
## =====
##                               Dependent variable:
##                               -----
##                               price_per_oz_c
##                               -----
## did                           0.557*
##                               (0.316)
```

```
##
## Constant                0.510***
##                        (0.121)
##
## -----
## Observations            72
## R2                      0.021
## Adjusted R2             0.007
## F Statistic             1.532 (df = 1; 70)
## =====
## Note:                   *p<0.1; **p<0.05; ***p<0.01
##                        Robust standard errors in parenthesis
```

You can see that the estimated coefficient is 0.557, which implies that taxed products have, on average, seen price increases 0.6p/ounze higher than non-taxed products. But this difference is not statistically significantly different.

Can you find the equivalent result in the Silver et al. paper?

Exporting Regression Output to Word

For your project you are likely to want to use some tables like the above. Here is the way to do that properly. You will have to add one option to the stargazer function.

```
stargazer_HC(mod1, out = "Table1.doc")
```

You will find a new file called “Table1.doc” in your working directory. However, copying this into another document may result in lost formatting.

In fact, in Word the Table will look even better if you change the type to be a html table (using `type_out = "html"`). In fact that creates a table in word which copies easily across other word documents.

```
stargazer_HC(mod1, type_out = "html", out = "Table1.doc")
```

Sometimes you will want to add information (say the mean of the dependent variable) or remove information (coefficient estimates that are not being discussed). You can then easily do these changes in word.