

UofT GASPS Workshop on Analysing Trade with the Structural Gravity Model

Pacha
v. 2022-12-16 12:50

This work is licensed under a [Creative Commons “Attribution 4.0 International”](#) license.



Contents

1 Disclaimer	1
2 Getting the most out of this material	1
3 Packages	1
4 Data	1
5 OLS estimates	3
6 Poisson Pseudo Maximum Likelihood	7
References	9

1 Disclaimer

The views and opinions expressed in this course are solely those of the author and do not necessarily reflect the official position of any unit of the OECD, the University of Toronto or the Pontifical Catholic University of Chile.

This workshop is based on Borchert et al. (2021), Gurevich and Herman (2020) and Yotov et al. (2016) and assumes basic R knowledge.

2 Getting the most out of this material

You can clone the GitHub repository to obtain the editable R files:

```
git clone https://github.com/pachadotdev/uoft-gasps-gravity-2022.git
```

Please read <https://happygitwithr.com/> if you have questions about git or GitHub.

3 Packages

Required packages for this workshop:

```
library(usitcgravity) # data
library(dplyr) # data cleaning/transformation/aggregation
library(tidyr) # nest/unnest data
library(purrr) # iteration
library(fixest) # regression
library(broom) # tidy regression results
library(duckdb) # additional SQL operations
```

usitcgravity has to be installed from GitHub. One option to install it is by running:

```
install.packages("remotes")
install_github("pachadotdev/usitcgravity")
```

4 Data

We are going to read directly from SQL. We can open a connection by using a function in usitcgravity.

```
con <- usitcgravity_connect()
```

Let's create a panel for the period 1986-2006 in intervals of 4 years. The required steps are:

1. Aggregate trade at sector level (4 sectors)
2. Add sector names
3. Add structural gravity variables

4. Move the data into R
5. Log trade and distance + create pair variable

Let's have a look at the tables:

```
dbListTables(con)
```

```
## [1] "country_names" "gravity"      "industry_names" "metadata"
## [5] "region_names"  "sector_names"  "trade"
```

Then check the `sector_names` table:

```
tbl(con, "sector_names") %>% collect()
```

```
## # A tibble: 4 x 2
##   broad_sector_id broad_sector
##             <int> <chr>
## 1             1 Agriculture
## 2             2 Manufacturing
## 3             3 Mining and Energy
## 4             4 Services
```

In order to create four panels (one per sector), we can create vectors to filter the years and sectors, iterate over `broad_sector_id`, and proceed with the rest of the steps.

In this particular case I transform the variables at the end, because it is more flexible to transform data in R than in SQL.

By using `purrr` we can create a list of tibbles which shall be used to estimate four models (i.e., specific sector effects) for the price of one.

This can be done in one chunk:

```
yrs <- seq(1986L, 2006L, 4)

sctrs <- tbl(con, "sector_names") %>%
  pull(broad_sector_id)

gravity <- map(
  sctrs,
  function(s) {
    tbl(con, "trade") %>%
      filter(year %in% yrs, broad_sector_id == s) %>%
      select(year, exporter_iso3, importer_iso3, broad_sector_id, trade) %>%
      group_by(year, exporter_iso3, importer_iso3, broad_sector_id) %>%
      summarise(trade = sum(trade, na.rm = T)) %>%

    left_join(
      tbl(con, "sector_names")
    ) %>%
```

```

select(year, exporter_iso3, importer_iso3, broad_sector, trade) %>%

left_join(
  tbl(con, "gravity") %>%
    select(year,
           exporter_iso3 = iso3_o, importer_iso3 = iso3_d, distance,
           contiguity, common_language, colony_ever
    )
) %>%

collect() %>%

mutate(
  log_trade = log(trade),
  log_distance = log(distance),
  pair = paste(exporter_iso3, importer_iso3, sep = "_"),
)
}
)

dbDisconnect(con, shutdown = T)

```

5 OLS estimates

Consider a simple specification of the structural gravity model of trade:

$$\log X_{ij,k,t} = \beta_0 + \beta_1 DIST_{i,j} + \beta_2 CNTG_{i,j} + \beta_3 LANG_{i,j} + \beta_4 CLNY_{i,j} + \varepsilon_{ij,t} \quad (1)$$

We need to remove 0 flows. If we don't do this, `lm` fails because $\log(0) = -\text{Inf}$.

```

map(
  seq_along(gravity),
  function(s) {
    d <- gravity[[s]] %>%
      filter(exporter_iso3 != importer_iso3, trade > 0)

    fit <- feols(log_trade ~ log_distance + contiguity + common_language +
                 colony_ever,
                 data = d
    )

    return(tidy(fit))
  }
)

```

```

## [[1]]
## # A tibble: 5 x 5

```

```
## term estimate std.error statistic p.value
## <chr> <dbl> <dbl> <dbl> <dbl>
## 1 (Intercept) 1.78 0.132 13.5 1.97e- 41
## 2 log_distance -0.328 0.0151 -21.7 8.83e-104
## 3 contiguity 1.74 0.0715 24.4 6.07e-131
## 4 common_language -0.141 0.0256 -5.52 3.47e- 8
## 5 colony_ever 1.97 0.0684 28.9 3.38e-182
##
## [[2]]
## # A tibble: 5 x 5
## term estimate std.error statistic p.value
## <chr> <dbl> <dbl> <dbl> <dbl>
## 1 (Intercept) 8.11 0.137 59.3 0
## 2 log_distance -0.964 0.0155 -62.0 0
## 3 contiguity 2.13 0.0898 23.7 6.89e-124
## 4 common_language -0.112 0.0251 -4.46 8.02e- 6
## 5 colony_ever 3.33 0.0876 38.0 4.00e-314
##
## [[3]]
## # A tibble: 5 x 5
## term estimate std.error statistic p.value
## <chr> <dbl> <dbl> <dbl> <dbl>
## 1 (Intercept) 0.982 0.221 4.43 9.31e- 6
## 2 log_distance -0.306 0.0258 -11.9 1.88e-32
## 3 contiguity 2.18 0.109 20.1 5.21e-89
## 4 common_language -0.174 0.0470 -3.71 2.11e- 4
## 5 colony_ever 1.13 0.108 10.4 1.80e-25
##
## [[4]]
## # A tibble: 5 x 5
## term estimate std.error statistic p.value
## <chr> <dbl> <dbl> <dbl> <dbl>
## 1 (Intercept) 6.95 0.378 18.4 1.81e-73
## 2 log_distance -0.527 0.0451 -11.7 3.37e-31
## 3 contiguity 0.827 0.222 3.72 1.98e- 4
## 4 common_language 1.30 0.102 12.8 7.16e-37
## 5 colony_ever 2.19 0.250 8.79 1.96e-18
```

It is good practice to compute clustered standard errors, so we use the “pair” variable previously created.

```
map(
  seq_along(gravity),
  function(s) {
    d <- gravity[[s]] %>%
      filter(exporter_iso3 != importer_iso3, trade > 0)

    fit <- feols(log_trade ~ log_distance + contiguity + common_language +
```

```

    colony_ever,
    data = d,
    cluster = ~pair
  )

  return(tidy(fit))
}
)

```

```

## [[1]]
## # A tibble: 5 x 5
##   term          estimate std.error statistic  p.value
##   <chr>          <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept)      1.78      0.250      7.13 1.07e-12
## 2 log_distance    -0.328     0.0285    -11.5 1.85e-30
## 3 contiguity       1.74      0.130     13.4 7.31e-41
## 4 common_language -0.141     0.0477     -2.96 3.10e- 3
## 5 colony_ever      1.97      0.128     15.5 1.45e-53
##
## [[2]]
## # A tibble: 5 x 5
##   term          estimate std.error statistic  p.value
##   <chr>          <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept)      8.11      0.246     33.0 8.48e-236
## 2 log_distance    -0.964     0.0278    -34.7 5.04e-259
## 3 contiguity       2.13      0.154     13.8 1.80e- 43
## 4 common_language -0.112     0.0440     -2.55 1.07e- 2
## 5 colony_ever      3.33      0.144     23.2 8.39e-118
##
## [[3]]
## # A tibble: 5 x 5
##   term          estimate std.error statistic  p.value
##   <chr>          <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept)      0.982     0.360      2.72 6.46e- 3
## 2 log_distance    -0.306     0.0420     -7.29 3.36e-13
## 3 contiguity       2.18      0.178     12.2 3.20e-34
## 4 common_language -0.174     0.0780     -2.23 2.56e- 2
## 5 colony_ever      1.13      0.184      6.16 7.66e-10
##
## [[4]]
## # A tibble: 5 x 5
##   term          estimate std.error statistic  p.value
##   <chr>          <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept)      6.95      0.409     17.0 1.25e-62
## 2 log_distance    -0.527     0.0499    -10.6 9.88e-26
## 3 contiguity       0.827     0.238      3.47 5.22e- 4
## 4 common_language  1.30      0.110     11.9 6.70e-32

```

```
## 5 colony_ever          2.19      0.246      8.90 7.98e-19
```

It is (extremely) important to conduct a misspecification test. Some of the models pass the misspecification test.

```
map(
  seq_along(gravity),
  function(s) {
    d <- gravity[[s]] %>%
      filter(exporter_iso3 != importer_iso3, trade > 0)

    fit <- feols(log_trade ~ log_distance + contiguity + common_language +
      colony_ever,
      data = d,
      cluster = ~pair
    )

    d <- augment(fit, newdata = d) %>%
      mutate(.fitted2 = .fitted^2)

    fit_reset <- feols(log_trade ~ log_distance + contiguity + common_language +
      colony_ever + .fitted2,
      data = d,
      cluster = ~pair
    )

    return(
      tidy(fit_reset) %>%
        filter(term == ".fitted2")
    )
  }
)
```

```
## [[1]]
## # A tibble: 1 x 5
##   term      estimate std.error statistic p.value
##   <chr>      <dbl>    <dbl>    <dbl>    <dbl>
## 1 .fitted2 -0.0526    0.0353    -1.49    0.137
##
## [[2]]
## # A tibble: 1 x 5
##   term      estimate std.error statistic p.value
##   <chr>      <dbl>    <dbl>    <dbl>    <dbl>
## 1 .fitted2 -0.0767    0.0115    -6.65 3.07e-11
##
## [[3]]
## # A tibble: 1 x 5
##   term      estimate std.error statistic p.value
```

```
##      <chr>      <dbl>      <dbl>      <dbl>      <dbl>
## 1 .fitted2    -0.0861      0.0709      -1.21      0.225
##
## [[4]]
## # A tibble: 1 x 5
##   term      estimate std.error statistic  p.value
##   <chr>      <dbl>      <dbl>      <dbl>      <dbl>
## 1 .fitted2   -0.312      0.0415      -7.52 6.50e-14
```

6 Poisson Pseudo Maximum Likelihood

The PPML model is written in multiplicative form and allows zero flows.

$$X_{ij,k,t} = \exp[\beta_0 + \beta_1 DIST_{i,j} + \beta_2 CNTG_{i,j} + \beta_3 LANG_{i,j} + \beta_4 CLNY_{i,j}] \times \varepsilon_{ij,t} \quad (2)$$

We evaluate the model as we did before, the code changes are minimal.

```
map(
  seq_along(gravity),
  function(s) {
    d <- gravity[[s]] %>%
      filter(exporter_iso3 != importer_iso3)

    fit <- fepois(trade ~ log_distance + contiguity + common_language +
      colony_ever,
      data = d,
      cluster = ~pair
    )

    return(tidy(fit))
  }
)
```

```
## [[1]]
## # A tibble: 5 x 5
##   term      estimate std.error statistic  p.value
##   <chr>      <dbl>      <dbl>      <dbl>      <dbl>
## 1 (Intercept)    3.60      0.806      4.47 7.65e- 6
## 2 log_distance  -0.199     0.0928     -2.14 3.23e- 2
## 3 contiguity     1.98      0.317      6.25 4.03e-10
## 4 common_language -0.126     0.138     -0.915 3.60e- 1
## 5 colony_ever    1.08      0.210      5.13 2.93e- 7
##
## [[2]]
## # A tibble: 5 x 5
##   term      estimate std.error statistic  p.value
##   <chr>      <dbl>      <dbl>      <dbl>      <dbl>
```



```
## 1 (Intercept)      8.98      0.834      10.8 5.14e-27
## 2 log_distance    -0.493      0.0962     -5.12 2.98e- 7
## 3 contiguity       2.08       0.365       5.69 1.26e- 8
## 4 common_language -0.417      0.145      -2.88 4.00e- 3
## 5 colony_ever      1.29       0.277       4.68 2.88e- 6
##
## [[3]]
## # A tibble: 5 x 5
##   term          estimate std.error statistic  p.value
##   <chr>          <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept)      3.78      0.980      3.86 0.000115
## 2 log_distance    -0.0574    0.112     -0.511 0.610
## 3 contiguity       1.83      0.466      3.92 0.0000889
## 4 common_language -0.288     0.172     -1.67 0.0940
## 5 colony_ever      1.05      0.294      3.59 0.000337
##
## [[4]]
## # A tibble: 5 x 5
##   term          estimate std.error statistic  p.value
##   <chr>          <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept)      7.92      0.599     13.2 6.37e-40
## 2 log_distance    -0.226     0.0749     -3.02 2.50e- 3
## 3 contiguity       1.03      0.284      3.64 2.73e- 4
## 4 common_language  0.0906     0.169      0.538 5.91e- 1
## 5 colony_ever      1.26      0.332      3.80 1.45e- 4
```

References

- Borchert, Ingo, Mario Larch, Serge Shikher, and Yoto V. Yotov. 2021. “The International Trade and Production Database for Estimation (ITPD-E).” *International Economics* 166 (August): 140–66. <https://doi.org/10.1016/j.inteco.2020.08.001>.
- Gurevich, Tamara, and Peter Herman. 2020. “The Dynamic Gravity Dataset: 1948-2016.” 2018-2. U.S. International Trade Commission. <https://www.usitc.gov/data/gravity/dgd.htm>.
- Yotov, Yoto V., Roberta Piermartini, José-Antonio Monteiro, and Mario Larch. 2016. *An Advanced Guide to Trade Policy Analysis: The Structural Gravity Model*. WTO iLibrary. <https://doi.org/10.30875/abc0167e-en>.