How to Work with Pseudo-Data

Banco de Portugal's Microdata Research Laboratory (BPLIM)

2024-06-01

Contents

1 BPLIM Guide to working with pseudo-data

This document serves as a guide for researchers working with pseudo-data prepared by **BPLIM**. Pseudo-data here refers to data that is generated randomly but respects the metadata, the links, and the time structure of the original data. Its purpose is solely to help researchers prepare the Stata scripts for surrogate access (a.k.a. remote execution). Please read this article for a brief introduction to **BPLIM**'s workflow for working with confidential data for research.

1.1 Data structure and links

To prepare the pseudo-data BPLIM must work with the researchers to identify all original datasets to use in the project along with the linking variables. For each original dataset, BPLIM will create a metafile and a file with a sample of the id (linking) variables. These files are needed to create the pseudo-data. As mentioned in the article, **BPLIM** does not provide the actual pseudo-data, but instead provides a set of *Stata* do-files that will generate the pseudo-data.

1.2 The package

BPLIM will prepare and send to the external researchers a compressed file with all the necessary files. For illustrative purposes, let's assume that the name of the project is **pxxx_BPLIM** and that the researcher identified two original files to work with:"CB_2006" and "CB_2007". The researcher will receive a zipped file, for example, *pxxx_BPLIM_pseudo.zip*. The researcher only has to unzip that file. This will create the following directory/file structure:

```
.../package

   ados/

   dos/
       generate_dummy_dos.do
       master.do
       CB_2006_dummy.do
       ...

   ids/
       CB_2006_ID.dta
       CB_2007_ID.dta
       ...

   metadata/
       CB_2006_meta.xlsx
```

```
    CB_2007_meta.xlsx
    ...

pxxx_BPLIM/
     ...
```

All the files in the first four directories - *ados*, *dos*, *ids*, and *metadata* - are needed to generate the pseudo datasets. The directory *pxxx_BPLIM* is the project folder, where the researcher is supposed to develop his/her work. Below we will further detail the structure of the project folder.

1.3 Creating the pseudo data

After unzipping the file, the researcher should open the file *master.do* in **Stata** (located in the *dos* folder). Please make sure that the **Stata** working directory is *…/package/dos* (where the *master.do* file is located) because we use relative paths in this script to reference other necessary files. Running *master.do* will create the pseudo datasets and place them in the project directory *pxxx_BPLIM* under the folder *initial_dataset*. The project directory has the following structure:

```
.../package/pxxx_BPLIM/

   initial_dataset/
      CB_D_2006.dta
      CB_D_2007.dta
      ...

   results/
      ...

   tools/
      ...

   work_area
       profile.do
       template.do
```

where we show the pseudo-data files that were created. These files will contain "*_D_*" in their names to reflect the fact that they are "dummy" data. After creating the pseudo data, you may relocate the project directory wherever you wish. However, the structure of the project directory must remain the same, because it mirrors the structure that BPLIM (or an internal user) has to replicate the code using original data. You should not copy any additional data files to the folder *initial_dataset*.

1.4 Setting up the project

1.4.1 Modifying the profile.do

After creating the pseudo data, the researcher is ready to start coding. The researcher should place all scripts in the *work_area* directory. In this area, the researcher is free to organize the code as it pleases. However, you may have noticed that this folder already contains two files: "*template.do*" and "*profile.do*". As the name suggests, *template.do* is a template that you should use to prepare your scripts. But before you can start coding you must edit *profile.do*. This do-file sets all the configurations (globals, paths, etc.) and must be placed in the same directory as the script file that you are executing. **Stata** will run this file first automatically. For example, suppose that you decide to place your project directory under the folder *C:/Users/Jane/*. In that case you edit *profile.do* and adjust the global `root_path`, to "C:/Users/Jane/pxxx_BPLIM", the path to the project directory.

```
*************************************************************
*            Initialization
*************************************************************
version 18
clear all
program drop _all
set more off
set rmsg on
set matsize 10000
set linesize 255
capture log close
*************************************************************
*               Define globals                              *
*************************************************************
**** Path for replication ****
* Root path
global root_path "C:/Users/Jane/pxxx_BPLIM"    /* changed here  */
* Base path for replications
global path_rep "${root_path}/work_area"

**** Paths for data ****
* Set the path for non perturbed data source
global path_source "${root_path}/initial_dataset"

**** Globals for type of modified dataset
* Dummy
global M1 "D"
/*******************************************************************************
Example: pseudo data
```

```
use "${path_source}/SLB_${M1}_YBNK_20102018_OCT20_QA1_V01.dta"
*********************************************************************/

**** Path for project specific ado files ****
adopath ++ "${root_path}/tools"

* Change ados path
sysdir set PLUS "${root_path}/tools"
local places "SITE PERSONAL OLDPLACE"
foreach p of local places {
  capture adopath - `p'
}
```

Note that the globals `path_rep` and `path_source` are built based on `root_path`. Global `M1` is used to reference the type of dataset being used. This means that when writing your scripts you must always use the global `M1` when referring to the datasets. For example, when reading the data do not write:

```
use "${path_source}/CB_D_2006.dta"
```

but instead use the global in the name of the file:

```
use "${path_source}/CB_${M1}_2006.dta"
```

1.4.2 Installing tools for your project

All the external user-written ados must be installed in the *tools* directory of your project. We recommend using adoinstall for this purpose. Fo example, to install `reghdfe` in the *tools* directory, you need to type the following in Stata:

```
ssc install adoinstall
adoinstall reghdfe, to("C:/Users/Jane/package/pxxx_BPLIM/tools")
```

The scripts that you prepare should only use user-written commands that are placed in the *tools* directory of your project.

1.5 Preparing your code

It is really important to follow the above guidelines to ensure that results can be safely replicated on the original data. If the code is well organized and follows the guidelines then BPLIM staff (or an internal co-author) can easily change the configuration file *profile.do* to rerun the analysis on the original data. The template file that we provide shows examples

on how the researcher should code, based on the settings defined in *profile.do*. Below we show the contents of *template.do*:

```
* Project      :  pxxx_BPLIM
* Author(s)    :
* Date         :
* Description  :
* Dependencies :
* Modifications: (add date, author and change)

* Run profile (usually not needed, but just to be sure)
capture run "profile.do"

* Change to work path - global `path_rep` defined in profile.do
cd "${path_rep}"

/* You may create a `results` folder inside `path_rep` to save outputs (this
is optional since there is already a `results` folder outside `path_rep`)
Always use capture when creating directories in scripts*/
capture mkdir results
* You may create the structure that you want, adding sub-diectories to `results`
capture mkdir results/tables
capture mkdir results/figures

/*
When defining globals for paths (if you do not want to use relative paths), remember to
include the global `path_rep`. This is the path where the analysis should run. See the
two examples below, where we define two globals for separate results folders
*/
global results_tables "${path_rep}/results/tables"
global results_figures "${path_rep}/results/figures"

* Creating a log file in the work area, where "logexample" is the log requested for ext
log using "logexample.log", replace

************************************************************
*                  Open data files                         *
************************************************************
/*
Please note the VERY IMPORTANT use of global `M1`, ${M1},
in the file names of the modified data. Failing to use the
globals when working with modified data will cause the
REPLICATION TO FAIL.
*/
```

```
* Example on how to read a dummy data file provided by BPLIM:
use "${path_source}/CB_${M1}_2006", clear

************************************************************
*            Start data analysis                           *
************************************************************
*
* ...
* YOUR STATA CODE GOES HERE
* ...
* You may call other do-files, just be sure to use the
* globals defined in the profile and eventually in this
* file

************************************************************
*            Saving the results                              *
************************************************************

* Saving an intermediate dataset. You may save it in your
* work area, if you wish to preserve it in order to analyze
* it later. However, in a replication, if these datasets are
* not needed, you can delete them. As an example, see below:

* Create intermediate data directory
cap mkdir intermediate_data

* Add global for directory
global path_data "${path_rep}/intermediate_data"

* Save datasets
compress
save "${path_data}/filename1.dta", replace
save "${path_data}/filename2.dta", replace

* Remove intermediate data after analysis is finished
local files: dir "${path_data}" files "*.dta", respectcase
foreach file of local files {
  rm "${path_data}/`file'"
}

****** Results Examples ******
* Creating a graph and saving to the results area (figures)
graph export "${results_figures}/mygraph.png", replace
* Creating a table and saving to the results area (tables)
```

```
esttab using "${results_tables}/myfile.tex", cells("cell1 cell2 ...")

*************************************************************
*                  Close the log file                       *
*************************************************************
log close

***************** IMPORTANT *****************

* BPLIM reserves the right to decline to send log files that are
* exceptionally large. For really long scripts, remember that not
* every line of code and its output needs to be in the log file.
* Take the following example:
/*
log using "mylog.log", replace

quietly {
  sysuse auto, clear
  summarize price
  noisily display "Mean Price: `r(mean)'"
  drop if foreign == 1
  keep price mpg rep78
  noisily reg price mpg i.rep78
}

log close
*/

* Only the outputs of the third and sixth lines will appear in the
* log file. This is a good strategy to follow if your log files are
* too cluttered with code and output, which are only intermediate steps
* to final outputs
```

Notice the first lines of the template, where we run the configuration file - `capture run "profile.do"` - and change directory to the work area folder - `cd "${path_rep}"`. Note that global `path_rep` is defined in the configuration file. We also create new folders in our working directory and set globals in order to reference them later:

```
capture mkdir results
* You may create the structure that you want, adding sub-diectories to `results`
capture mkdir results/tables
capture mkdir results/figures
...
global results_tables "${path_rep}/results/tables"
```

```
global results_figures "${path_rep}/results/figures"
```

The remaining lines of the template contain standard examples of how to generate outputs and organize your code.

1.5.1 Create a master script

Researchers should create a master script that runs their analysis from top to bottom. This file should be based on *template.do* and must create a log file that proves that the code ran without errors. Still, researchers are free to organize code in the *work_area* as they please. For example, the following structure would be acceptable:

```
.../package/pxxx_BPLIM/work_area/

  profile.do
  master.do

  01_data_management/
      01_data_manipulation.do
      ...

  02_exploratory/
      01_tables.do
      02_figures.do
      ...

  03_regressions/
       01_regressions.do
       ...
```

Then, within *master.do*, the researcher only has to run the dependencies:

```
...
cd ${path_rep}
...

do 01_data_management/01_data_manipulation.do

do 02_exploratory/01_tables.do
do 02_exploratory/02_figures.do

do 03_regressions/01_regressions.do

log close
```

1.6 Creating the replicability package

After completing your analysis, you must prepare your replication package to send to **BPLIM**. With that package, **BPLIM** wil be able to replicate your results using the original data.

Preparing the package is a simple procedure. All you need to do is run ado `archive_rep`, which will zip all the necessary files for the replication. It also generates a list of all the files used and creates a requirements file with all the dependencies (ados). Please note that only script files (ados, dos, etc) are copied to the archive. In this case, we would run the following in Stata:

```
adopath + "C:/Users/Jane/pxxx_BPLIM/tools"
archive_rep, rep(1) path("C:/Users/Jane/pxxx_BPLIM")
```

The output of the command is a folder named **Rep001** with the folders and files (scripts) needed for the replication. This folder, and the zip file *Rep001.zip* (zip file of the folder) are created in the *work_area* directory. The researcher only has to send the zip file to **BPLIM**.

1.7 Recap

It is important that the researcher follows the guidelines and the proper workflow:

1. Researcher identifies the datasets and tools for the project
2. **BPLIM** staff prepares the package and sends it to the researcher in a zip file
3. Researcher unpacks the file and creates the project structure
4. Researcher runs the do-file to create the pseudo data in the project initial dataset folder
5. Researcher adapts *profile.do*, namely the global `root_path`, so that it points to the correct directory in her computer
6. Researcher creates a *master.do* file based on *template.do* (it can be copied) and organizes the code as she desires
7. After the analysis is finished, the researcher runs the ado `archive_rep` and sends the output zip file to **BPLIM**. Researchers should clearly indicate the output files needed for the analysis.
8. **BPLIM** checks that the analysis runs from top to bottom. In case that it does, the staff will run the analysis on the original data
9. **BPLIM** checks the output files requested by the researchers and emails the results if they respect the output control rules.

Important note:

- Intermediate datasets must be created during the analysis. Remember that the researcher is only allowed to send scripts and logs for replications, so **BPLIM** only has access to datasets in the *initial_dataset* folder. Trying to reference intermediate data that is not created during the analysis will cause the replication to **fail**.