SRQM: Programs

This document lists the programs included with the Statistical Reasoning and Quantitative Methods (SRQM) course run at Sciences Po by François Briatte and Ivaylo Petev.

All commands were written to assist students in completing their research projects. This file describes all commands in more detail than necessary to run them in class.

`burd`

The series of ten `scheme-burd` files contain reversed versions of the `RdBu` ColorBrewer theme, as well as a replacement for the `s2color` scheme. Used to scheme the course plots. Some example plots appear on the course wiki.

`properl`

Sets the labels of a variable to their proper capitalization, using code kindly provided by William A. Huber as an answer to a StackOverflow question. Occasionally used for data management.

`repl`

Creates a replication folder out of a do-file. The folder will contain the do-file, log and all plots that were assigned a `name()` in the code. It will also contain any file exported by the do-file and a short file manifest in a `README` file.

`require`

Checks whether a given list of commands are currently installed in Stata, and if not, tries to install the corresponding package at the SSC archive. Occasionally used for debugging.

`srqm`

The `srqm` utilities rely on the architecture of the `SRQM` folder, a.k.a the 'Teaching Pack', which contains the course material. All `srqm` commands should be run with this folder set as the working directory.

The `srqm` utilities require one command and optionally one subcommand to execute:

```
srqm command [subcommand] [, nolog]
```

The commands and subcommands form three blocks. The first of them, `setup`, is called from the `profile.do` file of the `SRQM` folder and is used to set up computers for the course:

```
srqm setup
srqm setup folder
srqm setup packages
```

To minimize trouble with working directory errors, the setup creates a mock symbolic link to the `SRQM` folder and performs a quick folder integrity check at startup. A few more similar checks are available:

```
srqm check
srqm check folder
srqm check packages
srqm check course
```

Finally, the mock symbolic link to the `SRQM` folder can be erased at the end of the semester, and a few more cleanup utilities can be used for testing purposes:

```
srqm clean
srqm clean folder
srqm clean packages
```

Unless the additional `nolog` option is specified, all commands send moderately verbose output to a log in order to help users report issues.

`srqm setup`

Tries to permanently set up a few options like screen breaks and scrollback buffer size in Stata, including `memory` on software versions older than Stata 12. Also sets the `burd` scheme (documented above).

`srqm setup folder`

Tries to tell Stata to automatically run from the `SRQM` folder by copying its path to the global macro `$srqm_wd`. The macro is saved into a `profile.do` file located in the Stata application folder.

The `profile.do` file saved to the Stata application folder acts as a symbolic link that also runs the `profile.do` file from the SRQM folder, to perform further integrity checks on the course setup.

This command requires to run Stata as administrator on Windows Vista and 7, or it will fail due to a system restriction in recent versions of Windows.

`srqm setup packages`

Installs the additional Stata packages used in the course do-files. Requires Internet access to execute properly. Usually runs in less than five minutes.

The subcommand will try to run as quickly as possible by skipping packages that are already installed. This behaviour can be overriden by passing the `forced` option.

`srqm check`

Produces a report on the current setup, covering the basic system options obtained with `query` and the list of installed packages obtained with `ado dir`.

`srqm check folder`

Checks the existence of the `Datasets` and `Replication` folders used in class. This subcommand is automatically run with the `nolog` option every time Stata loads with the `SRQM` working directory.

`srqm check packages`

Checks whether the additional packages installed from the SSC server by the `srqm setup packages` subcommand are up to date.

`srqm check course`

Runs the whole course (weekly sessions and draft assignments) as a single sequence to test the executability of the course code. Usually runs in less than ten minutes.

`srqm clean`

Cleans up the `SRQM` folder by erasing temporary workfiles produced by the course do-files. The workfiles consist of logs and folders with the `-files` suffix.

`srqm clean folder`

Unlinks Stata from the `SRQM` folder by erasing the `profile.do` file that is installed in the application folder as part of the setup for the course. Used at the end of the semester.

`srqm clean packages`

Uninstalls course packages. Used for testing purposes.

stab

Produces summary statistics and correlation matrix tables in the format required for course research projects. The command was built on top of the `tsst` command to improve student reports of summary statistics.

The basic syntax for `stab` is:

```
stab using week8 [aw,fw], su(v1 v2) corr fr(v3 v4) ttest prtest by() [f(0) replace]
```

The `su` option is meant produces five-number summaries out of continuous variables. The `fr` option produces frequencies out of categorical variables. Both results are combined in a single table.

The `by(varname)` option can create multiple tables based on the categories of `varname`. The `corr` option adds a correlation matrix of the continuous variables specified in `su`. The `ttest` option adds *t*-tests for all groups.

tsst

Produces a simple table of summary statistics as required for the course research project. The code draws on a tutorial by Ben Jann.

The command is very barebones and means ‘tabbed summary statistics table’ because it produces tab-separated values in plain text format, for maximum compatibility with text and spreadsheet editors.

The basic syntax for `tsst` is:

```
tsst using stats.txt [aw,fw,pw], su(v1 v2) fr(v3 v4) [f(0) replace]
```

The `su` option is meant produces five-number summaries out of continuous variables. The `fr` option produces frequencies out of categorical variables. Both results are combined in a single table.

Far more sophisticated output options appear in packages like `estout` or `tabout`. Tamás Bartus is also developing the `publish` command, which comes close to the spirit of the `tsst` command.

Students are referred to the documentation of their text or spreadsheet editor to learn how to import a tab-separated values document, or how to convert tab-delimited text into tabular output.