Call Stata from Python

Zhao Xu

StataCorp LLC

2021 UK Stata Conference

09-09-2021

PyStata - Stata and Python Integration

In Stata 16, we introduced **PyStata integration** allowing users to call Python from Stata:

1. Invoke Python code interactively
2. Embed Python code in do-files and ado-files
3. Run Python script files within Stata

In Stata 17, we introduced **PyStata module** allowing users to call Stata from Python:

1. Access Stata and Mata conveniently in an IPython kernel-based environment, such as Jupyter Notebook and console, Jupyter Lab and console
2. Call Stata and Mata from environments that support the IPython kernel, such as Spyder IDE and PyCharm IDE
3. Call Stata and Mata when accessing Python from command-line environments, such as Windows Command Prompt, macOS terminal, and Unix terminal

The pystata Python module

The **pystata** package includes two sets of tools for interacting with Stata from Python:

1. Three IPython magic commands: **stata**, **mata**, and **pystata**
2. A suite of API functions

The magic commands, or magics, are handly commands used in an IPython (interactive Python) environment to perform particular tasks.

The **pystata** package includes three magic commands to access Stata and Mata within the IPython environment.

The API functions can be used to interact with Stata and Mata from within both IPython and non-IPython environments. It's particular useful when using Stata to perform automating task within a shell environment.

Configuration

The **pystata** package is shipped with Stata and is located in **STATA_SYSDIR/utilities/pystata** directory.

It is placed there for convenientce, to avoid conflicts between official updates to Stata and updates to the **pystata** Python package.

When you try to import this package, it will error out claming "**no module named 'pystata'**" because Python cannot locate it from Python's module search path (**sys.path**).

Four ways are provided to configure and locate this pacakge within Python.

Method 1: the stata_setup module

The **stata_setup** module is used to find Stata's installation path and the pystata pacakge. This module is hosted on PyPI (https://pypi.org/project/stata-setup/). Users can install it using

```
$ pip install stata_setup

or

$ pip install stata-setup
```

After it is installed successfully, type

```
>>> import stata_setup
>>> stata_setup.config('STATA_SYSDIR', 'STATA_EDITION')
```

The first argument is Stata's installation path and the second one is the edition to use.

Method 2: adding pystata to sys.path

Add **pystata**'s location in Python's module search path and then call the **init** function of the **config** module in the **pystata** pacakge:

```
>>> import sys
>>> sys.path.append('STATA_SYSDIR/utilities')
>>> from pystata import config
>>> config.init('STATA_EDITION')
```

Method 3: changing your current working dicrectory

In the Python environment, the current working directory is automatically on the module search path, so you can also locate the **pystata** package by changing your current working directory to **STATA_SYSDIR/utilities**:

```
>>> import os
>>> os.chdir('STATA_SYSDIR/utilities')
>>> from pystata import config
>>> config.init('STATA_EDITION')
```

Method 4: editing PYTHONPATH

PYTHONPATH is a Python environment variable storing a list of paths that are added to the default module search path when the Python environment is initialized. You can add **STATA_SYSDIR/utilities** to **PYTHONPATH** to locate the **pystata** package directly. More importantly, this step only needed to do once.

For Linux and Mac OS X users, you can set it permanently in your ~/.bashrc or ~/.bash_profile file,

```
$ export PYTHONPATH=STATA_SYSDIR/utilities:$PYTHONPATH
```

or in your ~/.cshrc file,

```
$ setenv PYTHONPATH STATA_SYSDIR\utilities:${PYTHONPATH}
```

```
>>> from pystata import config
>>> config.init('STATA_EDITION')
```

Configuration

```
  ___  ____  ____  ____  ____ ©
 /__    /   ____/   /   ____/      17.0
___/   /   /___/   /   /___/       SE—Standard Edition

 Statistics and Data Science       Copyright 1985-2021 StataCorp LLC
                                   StataCorp
                                   4905 Lakeway Drive
                                   College Station, Texas 77845 USA
                                   800-STATA-PC        https://www.stata.com
                                   979-696-4600        stata@stata.com

Stata license: Single-user  perpetual
Serial number: 100
  Licensed to: zxx
               StataCorp LLC

Notes:
      1. Unicode is supported; see help unicode_advice.
      2. Maximum number of variables is set to 5,000; see help set_maxvar.
```

Magic Commands

The **pystata** Python package provides three magic commands to interact with Stata from within the IPython environment:

1. **stata**: Execute Stata commands
2. **mata**: Execute Mata code
3. **pystata**: Configure and display current system information

The stata magic

The **stata** magic can be used as both a cell magic **%%stata** and a line magic **%stata**.

When the cell magic is used, a block of Stata commands can be specified and executed all at once. This is similar to executing a series of commands from a do-file. When the line magic is used, a one-line Stata command can be specified and executed, as it would be in Stata's Command window.

```
%%stata [-d DATA] [-f DFLIST|ARRLIST] [-force]
 [-doutd DATAFRAME] [-douta ARRAY] [-foutd FRAMELIST] [-fouta FRAMELIST]
 [-ret DICTIONARY] [-eret DICTIONARY] [-sret DICTIONARY] [-qui] [-nogr]
 [-gw WIDTH] [-gh HEIGHT]

stata_cmd1
stata_cmd2
....
```

```
%stata stata_cmd
```

The mata magic

The **mata** magic is used to execute Mata code. It can be used as both a cell magic **%%mata** and a line magic **%mata**. When used as a line magic, it allows you to either execute a single-line Mata statement or enter Mata's interactive environment.

```
%%mata [-m ARRAYLIST] [-outm MATLIST] [-qui] [-c]

mata_istmt1
mata_istmt2
....
```

```
%mata istmt
```

```
%mata [-c]
---------------------------------------------- mata (type end to exit) ------
:
```

The pystata magic

The **%pystata** line magic is used to configure and display current system information and settings.

```
%pystata status

    Display current system information and settings.

%pystata set graph_show True|False [, perm]

    Set whether Stata graphics are displayed.

%pystata set graph_size w #[in|px|cm] [, perm]
%pystata set graph_size h #[in|px|cm] [, perm]
%pystata set graph_size w #[in|px|cm] h #[in|px|cm] [, perm]

    Set dimensions for Stata graphs.

%pystata set graph_format svg|png|pdf [, perm]

    Set the graphic format used to display Stata graphs.
```

API functions

The **pystata** Python package also provides two modules, **config** and **stata**, to interact Stata with Python.

1. **config**: configure the system and display current system information and settings.
2. **stata**: contain core functions to run Stata commands, pass data between Stata and Python, access Stata's returned results, etc.

The API functions can replicate what the magic commands do, and they provide more functionalities.

The config module

init(*edition*)

Initialize Stata's environment within Python

status()

Display current system information and settings

```
System information
  Python version          3.8.5
  Stata version           Stata 17.0 (SE)
  Stata library path      C:\Program Files\Stata17\se-64.dll
  Stata initialized       True
  sfi initialized         True

Settings
  graphic display         True
  graphic size            width = default, height = default
  graphic format          svg
```

The config module

set_graph_format(*gformat* [, *perm*])

Set the file format used to export and display graphs

set_graph_size([*width*, *height*, *perm*])

Set the graph size for images to be exported and displayed

set_graph_show(*show* [, *perm*])

Set whether the graphs generated by Stata are to be exported and displayed

The stata module

run(*cmd* [, *quietly*, *echo*, *inline*])

Run a single line or a block of Stata commands.

nparray_to_data(*arr* [, *prefix*, *force*])
pdataframe_to_data(*df* [, *force*])
nparray_to_frame(*arr*, *stfr* [, *prefix*, *force*])
pdataframe_to_frame(*df*, *stfr* [, *force*])
nparray_from_data([*var*, *obs*, *selectvar*, ...])
pdataframe_from_data([*var*, *obs*, *selectvar*, ...])
nparray_from_frame(*stfr* [, *var*, *obs*, ...])
pdataframe_from_frame(*stfr* [, *var*, *obs*, ...])

Transfer data between Stata and Python.

get_return()
get_ereturn()
get_sreturn()

The sfi (Stata Function Interface) module

The **sfi** module allows users to interact Python's capabilities with core features of Stata:

1. **Characteristics**
2. **Current dataset**
3. **Frames**
4. **Date and time**
5. **Macros**
6. **Scalars**
7. **Stata and Mata Matrices**
8. **Value labels**
9. **Missing values**

This module can be used together with the magic commands and the API functions. It provides more functionalities to access Stata's data and results.