Journal of Statistical Software

May 2021, Volume 98, Software Review 1. *doi: 10.18637/jss.v098.s01*

Reviewer: Anson T. Y. Ho, Kim P. Huynh, David T. Jacho-Chávez, Diego Rojas-Baez

---

Data Science in Stata 16: Frames, Lasso, and Python Integration

StataCorp LLC, 4905 Lakeway Drive, College Station, Texas, 77845-4512, United States of America. Price varies according to license.
https://www.stata.com/

Introduction

Stata (StataCorp 2019) is one of the most widely used software for data analysis, statistics, and model fitting by economists, public policy researchers, epidemiologists, among others. Stata's recent release of version 16 in June 2019 includes an up-to-date methodological library and a user-friendly version of various cutting edge techniques. In the newest release, Stata has implemented several changes and additions (see https://www.stata.com/new-in-stata/) that include lasso, multiple data sets in memory, meta-analysis, choice models, Python integration, Bayes-multiple chains, panel-data extended regression models, sample-size analysis for confidence intervals, panel-data mixed logit, nonlinear dynamic stochastic general equilibrium (DSGE) models, numerical integration.

This review covers the most salient innovations in Stata 16. It is the first release that brings along an implementation of machine-learning tools. The three innovations we consider in this review are: (1) Multiple data sets in Memory, (2) Lasso for causal inference, and (3) Python integration. The following three sections are used to describe each one of these innovations. The last section are the final thoughts and conclusions of our review.

Multiple frames

The new capability to work with multiple data sets (referred as 'frames' in the Stata 16) in memory is one of the most awaited improvements among current users. Historically, Stata had always worked under the modality of a single data table stored in memory at any time. Users had limited options for using multiple data sets. Typical examples include merging or appending data sets, managing temporal files in a sequential program, among others. It was not user-friendly because any incidental operation performed on a separate data set implies losing all progress not saved in the data frame. The result was an increase in the number of "incidental" files related to intermediate tasks and the execution time increased altogether.

The mechanics of using multiple frames is simple and can be implemented both through the user interface, or through scripting ("do" files). The first data set loaded into memory uses

the "default" data frame. On top of that, users can create additional data frames to perform other tasks. The number of frames one can create is bounded by the physical memory of the hosting machine, as Stata loads the entire data set into memory. At any time users are able to perform the following operations on the existing frames in memory: create new frames, switch frames, copy frames, drop frames, reset frames, and link frames.

Linking frames allows the user to create connections between frames based on a set of variables that serve as a key, in the same fashion a relational database would operate. This reduces memory overhead produced by merge and join operations that are commonly used in Stata. For advanced users, they can be incorporated multiple fame in both "ado" files and "Mata" (Stata's very own built-in matrix programming language). Legacy scripts dealt with several data sets using `preserve/restore` will be executed in Stata 16 using the new "frame" scheme. This decreases the complexity in execution because it avoids the creation and deletion of temporal data sets.

A simple example can illustrate the difference between releases. Define a task as *"load example; summarize v1 in example; get a count of rows in auxiliary; get the mean of v1 in example; get a mean of a1 in auxiliary."* The code snippets shown below allow us to compare the ways to approach the same task on the latest release and the previous one. While both code samples use the same number of lines, the previous release code has to load the auxiliary data set every time it is used. Stata 16 reduces the execution time by having all the frames loaded in memory.

Stata 15

```
1.  use example.dta, clear
2.  sum v1
3.  preserve
4.  use auxiliary.dta
5.  count
6.  restore
7.  mean v1
8.  preserve
9.  use auxiliary.dta
10.  mean a1
11.  restore
```

Stata 16

```
1.  use example.dta, clear
2.  sum v1
3.  frame create aux
4.  frame change aux
5.  use auxiliary.dta
6.  count
7.  frame change default
8.  mean v1
9.  frame change aux
10.  mean a1
11.  frame change default
```

Although multiple frames functionality is an improvement to deal with multiple data sets, this feature still does not offer the same level of convenience/flexibility offered by other competing open source data analysis software. For example, Stata has not worked out a parsimonious way to preserve the links when matching variables are modified. Regardless, moving towards a more efficient and user-friendly way of managing related data sets is an important step in the right direction.

Model selection for causal inference

Stata 16 introduces a module for regularized estimation of Generalized Linear models. For example, it allows users to implement LASSO, Elastic Net, and the Square-root LASSO to

perform estimations for Gaussian, Logit, Probit, and Poisson specifications. Stata 16 is the first official implementation, although user-written programs currently exist for such procedures.[1] The models are fitted through Stata's internal gradient coordinate descent algorithm, which performs similarly to other data analysis tools. The model selection procedures can be done by the usual cross validation[2] sampling methods for hyperparameter selection, or, by estimation of the penalty hyper parameters as in Belloni, Chen, Chernozhukov, and Hansen (2012). In addition, the module provides several unique features that has its own advantages with respect to alternative routines in other statistical software.[3] It allows users to implement Zou's (2006) Adaptive-Lasso. A new tool for estimating regularization parameters through the iterative plugin method is also introduced.

The most important feature in Stata 16's implementation of these popular models is the causal inference on a subset of parameters of interests, based on recent developments in post-selection inference by Belloni, Chernozhukov, and Hansen (2014a,b); Belloni *et al.* (2012); Chernozhukov, Chetverikov, Demirer, Duflo, Hansen, Newey, and Robins (2018). These modules provide several alternatives to achieve reliable post-selection inference even in the presence of "endogenous" covariates.[4]

The usual approach to inference on parameters after using regularized methods for model selection is to perform OLS on the set of regressors that survived. Several works, including Leeb and Pötscher (2008b,a), have showed that the statistical properties of tools like LASSO for prediction cannot always be extrapolated to the context of inference on parameters. Recent developments by Belloni *et al.* (2014a,b, 2012); Chernozhukov *et al.* (2018) take into account that the set of selected variables can be subject to errors. If the uncertainty in model selection is not taken into account when making inference on the parameters, as the case of vanilla OLS, the estimators could suffer from omitted variable bias. Hence, Stata's new addition brings a recent and powerful tool that merges the causal inference approach to the selection approach from machine learning.

While the capabilities this module offers are not unique to Stata 16 (see Chernozhukov, Hansen, and Spindler 2016, for the R package **hdm**), it provides a user-oriented implementation that brings tools at the edge of the field of machine learning applied to economics. Empiricists in other fields such as political sciences, public policy, among others could also greatly benefit from this. Considering the wide base of social scientists and policy makers that use Stata, this module is a useful addition to their toolbox. One possible setback in this implementation, as brought to our attention by a referee, is that the use of grid option to provide a user-specified list of parameters is somewhat inflexible. It only allows for equally spaced grids, and in the case of a single value, it is necessary to create a grid of length one.

Python integration

As the social sciences start to catch up with the recent surge in data availability and the techniques developed to analyze it, Stata 16 allows for the integration with Python, one of the most versatile programming languages around. This language has a broad audience ranging

[1] The user-written packages are described in Ahrens, Hansen, and Schaffer (2018a,b).

[2] For an example of an application in this setting please see Rojas, Estrada, Huynh, and Jacho-Chávez (2020).

[3] For a detailed description please see https://www.stata.com/manuals/lasso.pdf.

[4] For further details please visit: https://www.stata.com/manuals/lassolassoinferenceintro.pdf.

from arts, education, to software engineering, and it has been rapidly being adopted by data analysts.[5] Stata 16's Python module requires minimum Python 2.7 and all required Python dependencies installed on the local machine.

There are two ways users are able to interact with Python from Stata 16. First, one can use the `python` Stata command. This command allows users to embed Python code or run Python scripts from the command prompt, or in their do or ado[6] files. For example, if one can get the current time using Python in Stata 16 by typing at the Stata command prompt:

```
. python
-------------------- python (type end to exit) --------------------
>>> from datetime import datetime
>>> now= datetime.now()
>>> current_time = now. strftime("%H:%M:%S")
>>> print("Current Time =", current_time)
Current Time = 15:58:21
>>> end
------------------------------------------------------------------
.
```

If users need to use a user-written function stored as a Python script, the `python script` Stata command can be used instead. For example, take the following `v98s01.py` script:

```
import matplotlib.pyplot as plt
from matplotlib import style
style.use('fivethirtyeight')
import numpy as np
from scipy.stats import multivariate_normal

def plot_norm(M,Cov,n):
    rv = multivariate_normal(M, Cov, n)
    x = np.linspace(-10, 10, 500)
    y = np.linspace(-10, 10, 500)
    X, Y = np.meshgrid(x, y)
    pos = np.array([X.flatten(), Y.flatten()]).T
    fig = plt.figure(figsize=(5, 5))
    ax0 = fig.add_subplot(111)
    ax0.contour(rv.pdf(pos).reshape(500, 500))
    plt.show()

mean = [0, 0]
cov = [[7, 0.7], [0.7, 11]]

plot_norm(mean, cov, 300)
```

[5] In 2019, Python became the fourth more popular programming language in the world.

[6] See for example Estrada, Huynh, Jacho-Chávez, and Sánchez-Aragón (2020) for an integration of this sort applied to the estimation of peer effects in collaboration networks.

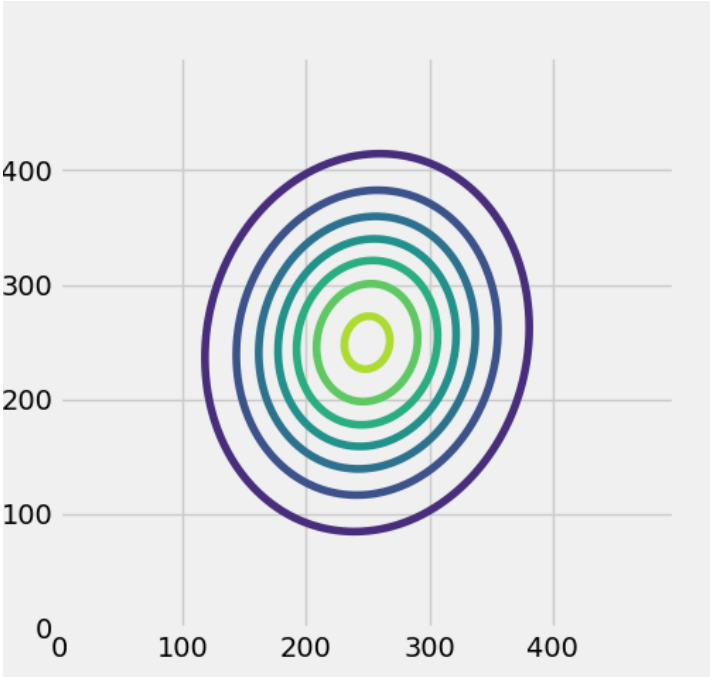

Figure 1: **Matplotlib** plot made by `python script` execution in Stata 16.

In order to use this script one simply writes at the Stata command prompt:

```
. python script v98s01.py
```

The plot in Figure 1 will be displayed on a **Matplotlib** instance. The `python script` Stata command can also pass arguments to the Python script for execution. This is accomplished by using a pre-built option `args`. To pass a set of arguments to our example script, one can use: `python script v98s01.py , args(..)`. As with most commands in Stata, the `python` command can be embedded in both ado and do files.[7]

The second way Stata 16 integrates with Python is through the `sfi` (*Stata Function Interface*) module. This module provides a set of Python classes that allow the Python instance to interact with objects on Stata, and vice-versa. In other words, this is the channel that allows information to flow from one tool to the other. This includes, and not limited to, the transfer of data sets, frames, macros, scalars, Mata matrices, and Stata matrices. This module also allows for the execution of Stata commands within Python with the `SFIToolkit` class. The following code illustrates a Stata 16 session that utilizes a K-means clustering routine from `scikit-learn` and plot the results using `pyplot`'s 3D capabilities in Figure 2. transfer of macros Consider the following example. We want to load the well known auto data set in Stata. Then through the `python` command we can connect to the Python prompt. We import the `sfi` module and then we first connect to the "default" data frame in memory. Then using the another function in the class frame we can copy a selection of variables into a python dictionary. We can then use K-means clustering from `scikit-learn` and plot the result using `pyplot` 3D capabilities. The result of this task is shown in Figure 2.

[7]For a more detailed description of how to use the `python` command please see: https://www.stata.com/manuals/ppython.pdf.

```
. sysuse  auto
(1978 Automobile Data)
.
. python
-------------------- python (type end to exit) --------------------
>>>
>>> from sfi import *
>>> import pandas as pd
>>> import numpy as np
>>> from sklearn.cluster import KMeans
>>> from sklearn.preprocessing import StandardScaler
>>> import matplotlib.pyplot as plt
>>> from mpl_toolkits.mplot3d import Axes3D
>>>
>>> f=Frame.connect("default")
>>> m=pd.DataFrame.from_dict( f.getAsDict(["mpg","weight","price"]))
>>>
>>> for var in list(m): m[var] = m[var].astype(float)
>>>
>>> X = StandardScaler().fit_transform(m)
>>>
>>> k=KMeans(n_clusters=4, random_state=0).fit(X)
>>>
>>> fig = plt.figure(figsize=(12,10))
>>> ax = fig.add_subplot(111, projection='3d')
>>> ax.scatter(m["mpg"],m["weight"],m["price"],
...             c=k.labels_, cmap='viridis',
...             edgecolor='k', s=40, alpha = 0.8)
...
<mpl_toolkits.mplot3d.art3d.Path3DCollection object at 0x0000000033E0A148>
>>> ax.set_title("Cars clustering")
Text(0.5, 0.92, 'Cars clustering')
>>> ax.set_xlabel("Miles_per_gallon")
Text(0.5, 0, 'Miles_per_gallon')
>>> ax.set_ylabel("Weight")
Text(0.5, 0, 'Weight')
>>> ax.set_zlabel("Price")
Text(0.5, 0, 'Price')
>>> ax.dist = 10
>>> plt.autoscale(enable=True, axis='x', tight=True)
>>> plt.show()
>>> end
------------------------------------------------------------------
.
```

The capability to interacting with Python from within Stata 16 is a relevant improvement that brings its users closer to the highly optimized tools in Python. Possibilities include: `bs4` for

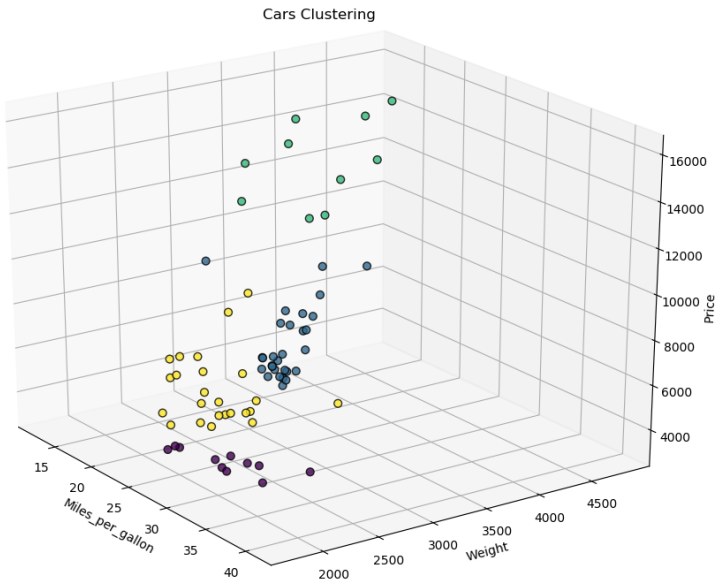

Figure 2: **Matplotlib**'s `pyplot` 3D plot of the K-means clustering in Python applied to Stata's classic 1978 automobile data set.

web scrapping, `scikit-learn` for machine learning tools, `sci-py` to deal with sparse matrix linear algebra, `multiprocessing` for process based parallelization, and `bokeh` for the creation of dashboards and other applications, etc.

Conclusions

Stata 16 brings its users closer to the latest tools in statistical analysis. It provides new features to the user interface, adds cutting edge tools to the set of available commands, and offers user the flexibility to utilize the specialized tools readily available in Python.

The addition of multiple frames to the Stata workspace gives users a more flexible way to interact with the program. More importantly, it also provides significant efficiency gains both in terms of user experience and performance. The cutting edge techniques for causal models with the possibility to obtain reliable inference for the estimated parameters is a novel implementation, to our knowledge. The most relevant addition, according to our judgment, is Stata 16's integration with Python. Stata provides their users with a (virtually unlimited) set of possibilities by allowing for this integration. This includes access to the latest tools for data analytics, data mining, spatial statistics, map rendering, plotting, application development, among others.

This review concludes by mentioning that the interfaces and modules analyzed here are not perfect. There is still room for improvement in every one of them. In the case of frames, the mechanics of use are still not as fluid as one might want. In the Python integration, the `sfi` module could, and should, be expanded in order to overcome the difference in schemes between programs. These, shortcomings are natural to first deployments, however, the additions reviewed here can be considered novel useful tools for the majority of users.

Acknowledgments and disclaimer

We thank David Drukker for helpful discussions about the ongoing developments of econometrics capabilities of Stata. We acknowledge the comments and suggestions of the editor and reviewers. We also acknowledge the efforts of Valérie Clermont, Mireille Lacroix, Phil Riopelle, and Nathalie Swift and the use of the Bank of Canada's Digital Analytical Zone Microsoft Azure Cloud. The views expressed in this review article are those of the authors. No responsibility for them should be attributed to the Bank of Canada. All remaining errors are the responsibility of the authors.

References

Ahrens A, Hansen CB, Schaffer ME (2018a). "**LASSOPACK**: Stata Module for Lasso, Square-Root Lasso, Elastic Net, Ridge, Adaptive Lasso Estimation and Cross-Validation." Statistical Software Components, Boston College Department of Economics. URL https://ideas.repec.org/c/boc/bocode/s458458.html.

Ahrens A, Hansen CB, Schaffer ME (2018b). "**PDSLASSO**: Stata Module for Post-Selection and Post-Regularization OLS or IV Estimation and Inference." Statistical Software Components, Boston College Department of Economics. URL https://ideas.repec.org/c/boc/bocode/s458459.html.

Belloni A, Chen D, Chernozhukov V, Hansen C (2012). "Sparse Models and Methods for Optimal Instruments with an Application to Eminent Domain." *Econometrica*, **80**(6), 2369–2429. doi:10.3982/ecta9626.

Belloni A, Chernozhukov V, Hansen C (2014a). "High-Dimensional Methods and Inference on Structural and Treatment Effects." *Journal of Economic Perspectives*, **28**(2), 29–50. doi:10.1257/jep.28.2.29.

Belloni A, Chernozhukov V, Hansen C (2014b). "Inference on Treatment Effects after Selection among High-Dimensional Controls." *The Review of Economic Studies*, **81**(2), 608–650. doi:10.1093/restud/rdt044.

Chernozhukov V, Chetverikov D, Demirer M, Duflo E, Hansen C, Newey W, Robins J (2018). "Double/Debiased Machine Learning for Treatment and Structural Parameters." *The Econometrics Journal*, **21**(1), C1–C68. doi:10.1111/ectj.12097.

Chernozhukov V, Hansen C, Spindler M (2016). "**hdm**: High-Dimensional Metrics." *The R Journal*, **8**(2), 185–199. doi:10.32614/RJ-2016-040.

Estrada J, Huynh KP, Jacho-Chávez DT, Sánchez-Aragón L (2020). "On the Identification and Estimation of Endogenous Peer Effects in Multiplex Networks." Unpublished Manuscript.

Leeb H, Pötscher BM (2008a). "Can One Estimate the Unconditional Distribution of Post-Model-Selection Estimators?" *Econometric Theory*, **24**(2), 338–376. doi:10.1017/s0266466608080158.

Leeb H, Pötscher BM (2008b). "Guest Editors' Editorial: Recent Developments in Model Selection and Related Areas." *Econometric Theory*, **24**(2), 319–322. doi:10.1017/s0266466608080134.

Rojas D, Estrada J, Huynh KP, Jacho-Chávez DT (2020). "Survival Analysis of Banknote Circulation: Fitness, Network Structure, and Machine Learning." *Advances in Econometrics*, **42**, 235–262. doi:10.1108/s0731-905320200000042018.

StataCorp (2019). *Stata Statistical Software: Release 16*. StataCorp LLC, College Station. URL http://www.stata.com/.

Zou H (2006). "The Adaptive Lasso and Its Oracle Properties." *Journal of the American Statistical Association*, **101**(476), 1418–1429. doi:10.1198/016214506000000735.

Reviewer:

Anson T. Y. Ho
Financial Stability Department
Bank of Canada
234 Wellington Ave., Ottawa ON K1A 0G9, Canada
E-mail: contact@atyho.info
URL: https://www.bankofcanada.ca/profile/anson-t-y-ho/

Kim P. Huynh
Currency Department
Bank of Canada
234 Wellington Ave., Ottawa ON K1A 0G9, Canada
E-mail: kim@huynh.tv
URL: https://www.bankofcanada.ca/profile/kim-huynh/

David Jacho-Chávez, Diego Rojas-Baez
Department of Economics
Emory University
Atlanta, GA 30322-2240, United States of America
E-mail: djachocha@emory.edu, drojasb@emory.edu
URL: http://www.davidjachochavez.org/,
http://economics.emory.edu/home/people/grad-students/rojas-baez-diego.html

Journal of Statistical Software http://www.jstatsoft.org/
published by the Foundation for Open Access Statistics http://www.foastat.org/
May 2021, Volume 98, Software Review 1 *Published:* 2021-05-31
doi:10.18637/jss.v098.s01