

Stata/SQL/Python integration to emulate prospective cohort studies from big register data

Matteo Marrazzo

Nicola Orsini

Karolinska Institutet

2019 Nordic and Baltic Stata Users Group meeting
Stockholm | 30 August

Available sources

- Data registers
 - Big dimensions
 - Covering long periods of time
 - Necessity to develop solid designs

Design valid epidemiological studies

- Prospective cohorts
- Measuring exposures
- Defining outcomes
- Including confounders and effect modifiers
- Replication in different points in time

Relational Databases

- Structured data
- SQL language
- Key data processing
- ODBC

ODBC Stata Integration

```
. odbc list
```

Data Source Name	Driver
------------------	--------

odbc1	ODBC Driver
-------	-------------

```
odbc query "odbc1"  
odbc load, exec("`query'")
```

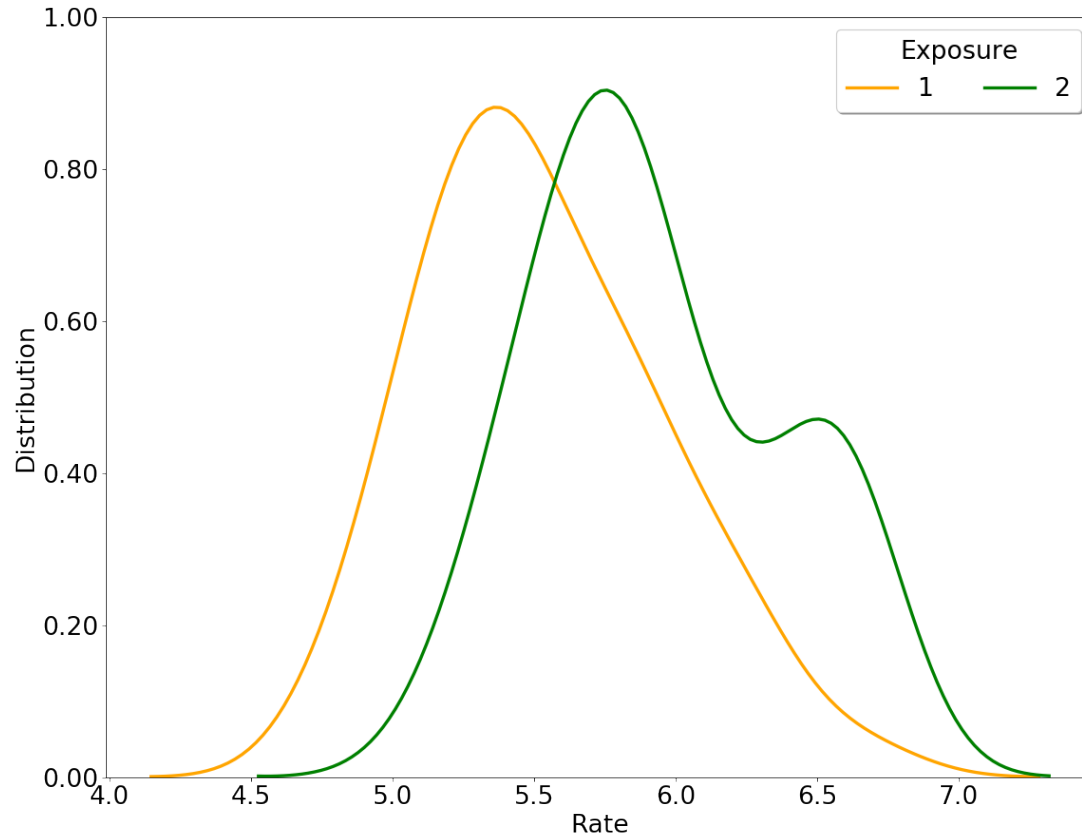
Statistical Analysis

- Poisson regression models to predict rates (**poisson**)
Exposure + effect modifiers + confounders
- Predictive margins (**margins**)
Adjusted rates by exposure

Python integration for visualization

```
use "C:\rates.dta", clear
python:
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os
os.environ['QT_QPA_PLATFORM_PLUGIN_PATH'] = "C:\\Anaconda3\\Library\\plugins"
from sfi import Data
X = np.array(Data.get("exposure rate"))
df = pd.DataFrame({'Exposure': X[:, 0], 'Rate': X[:, 1]})
fig, ax1 = plt.subplots()
colorset = ["orange", "green"]
for i in range(0, 2):
    sns.distplot(df.loc[df['Exposure'] == i, "Rate"], color = colorset[i], label=i, hist=False)
plt.ylim(0, 1)
plt.legend(title = 'Exposure', loc='upper right', ncol=2, fancybox=True, shadow=True)
plt.xlabel('Rate')
plt.ylabel('Distribution')
plt.show()
end
```

Distribution of rates by exposure



Animations with python: scatterplot

Import libraries and create dataframe from Stata

```
python:  
import numpy as np  
import pandas as pd  
import matplotlib  
import matplotlib.pyplot as plt  
import matplotlib.animation as animation  
import seaborn as sns  
import os  
os.environ['QT_QPA_PLATFORM_PLUGIN_PATH'] = "C:\\\\Anaconda3\\\\Library\\\\plugins"  
from sfi import Data  
X = np.array(Data.get("day exposure rate"))  
df = pd.DataFrame({'day': X[:, 0], 'exposure': X[:, 1], 'rate': X[:, 2]})
```

Animations with python

Create the basic plot figure and the function to get x and y

```
fig, ax = plt.subplots(figsize=(16, 9), dpi = 90)
ax.set_xlim(0,24)
ax.set_xlabel('Month')
ax.set_ylabel('Rate')
ax.set_ylim(4, 8)
ax.set_title('')
colorset = ["orange","green"]

def get_data(day=0, exposure=0):
    x = df.loc[(df['exposure'] == exposure) & (df['day'] == day), "day"]
    y = df.loc[(df['exposure'] == exposure) & (df['day'] == day), "rate"]
    return x,y
```

Animations with python

Create initialization and animation functions

```
# initialization function
def init():
    for j in range(2):
        x,y= get_data(day=0,exposure=j)
        sc = ax.scatter(x,y, c=colorset[j], s=10)
    return sc,

# animation function
def animate(i):
    for j in range(2):
        x,y= get_data(day=i,exposure=j)
        sc = ax.scatter(x,y, c=colorset[j], s=10)
    return sc,
```

Animations with python

Run the animation and save the file ('ffmpeg' required)

```
Writer = animation.writers['ffmpeg']  
writer = Writer(fps=5, metadata=dict(artist='Example'), bitrate=1800)  
  
ani = matplotlib.animation.FuncAnimation(fig, animate, init_func=init, frames=25,  
interval=5000, blit=True, repeat = True)  
  
ani.save("Animation.mp4", writer=writer)  
end
```

Animations with python



Conclusions

- We have shown how it's possible to integrate Stata with relational databases and python
- The design, implementation, analysis and visualization can be simplified by taking the best of every software
- The new python integration in Stata 16 works efficiently and provides a solid base to expand Stata capabilities
- This integration can provide solutions to increasingly complex research questions