Introducción a Stata

Programación para el análisis de datos

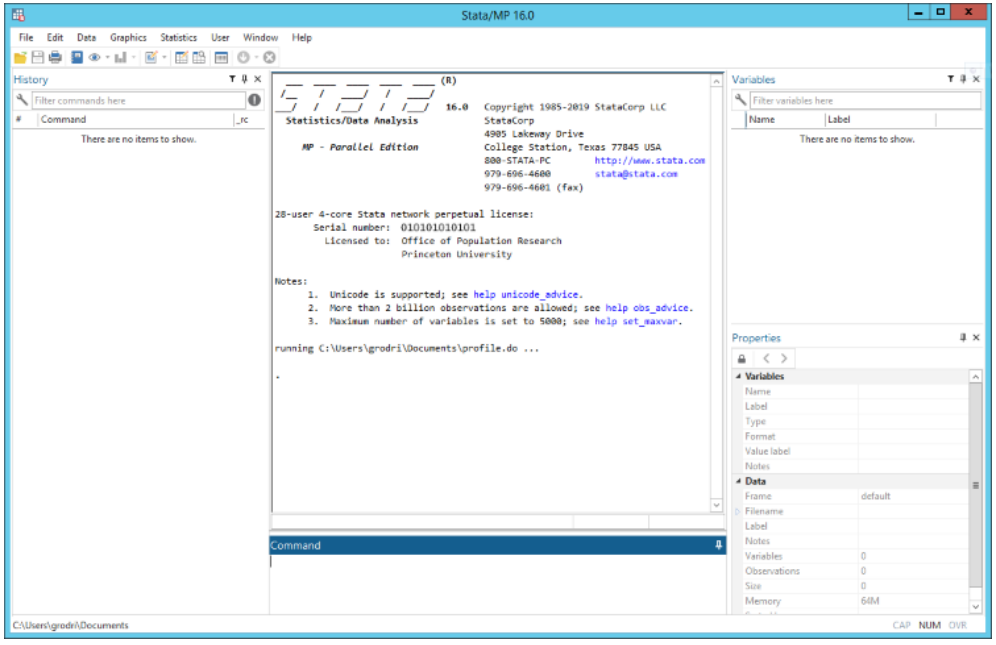
Departamento de Ciencias Sociales, UCU - Martín Opertti

Sobre Stata

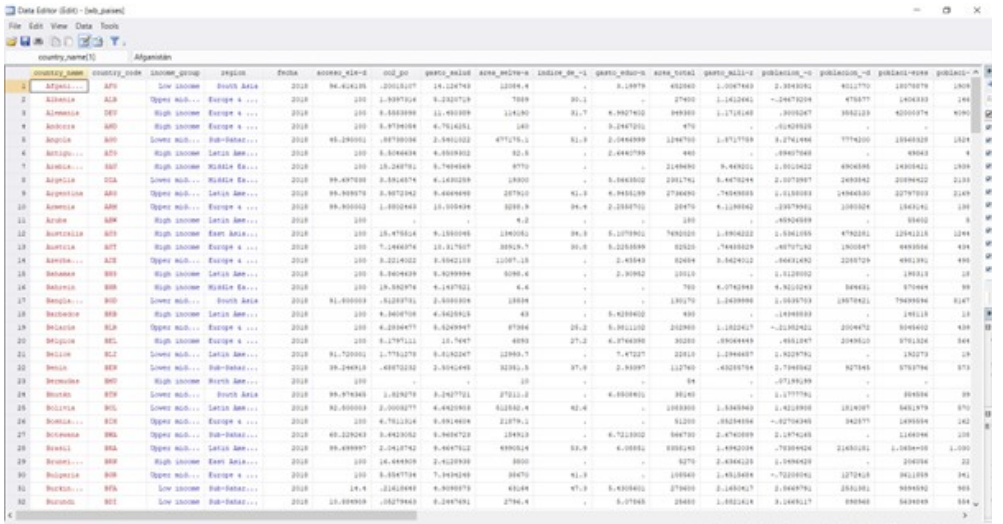
Introducción a Stata

- Stata es un paquete estadístico poderoso con muchas técnicas estadísticas disponibles.
- Es simple de usar y rápido.
- Es un software pago
- Tiene una importate comunidad de usuarios

Interfaz


Stata/MP 16.0
File Edit Data Graphics Statistics User Window Help
History
Filter commands here
Command _rc
There are no items to show.
(R)
16.0
Statistics/Data Analysis
MP - Parallel Edition
Copyright 1985-2019 StataCorp LLC
StataCorp
4905 Lakeway Drive
College Station, Texas 77845 USA
800-STATA-PC http://www.stata.com
979-696-4600 stata@stata.com
979-696-4601 (fax)
28-user 4-core Stata network perpetual license:
Serial number: 010101010101
Licensed to: Office of Population Research
Princeton University
Notes:
1. Unicode is supported; see help unicode_advice.
2. More than 2 billion observations are allowed; see help obs_advice.
3. Maximum number of variables is set to 5000; see help set_maxvar.
running C:\Users\grodri\Documents\profile.do ...
.
Command
Variables
Filter variables here
Name Label
There are no items to show.
Properties
Variables
Name
Label
Type
Format
Value label
Notes
Data
Frame default
Filename
Label
Notes
Variables 0
Observations 0
Size 0
Memory 64M
C:\Users\grodri\Documents
CAP NUM OVR

Editor de datos



Do file

- Un do file es un archivo de texto standard que Stata ejecuta si se escribe do *filename*
- Todo lo que hagamos en Stata para el curso debe ser hecho desde un do file.
- El uso de do files tiene varias ventajas:
  - Permite que el análisis sea reproducible
  - Permite hacer anotaciones
  - Permite encontrar errores en el código de forma más eficiente

Comandos básicos

- Las funciones o comandos en Stata son case sensitive, tener ciudado con las mayúsculas.
- Si escribimos `help mean` obtendremos información sobre el comando
- `clear` sirve para eliminar los datos cargados de la memoria de Stata
- `summarize` o `sum` sirve para obtener una descripción de los datos (particularmente útil para variables numéricas)
- `tabulate` o `tab` sirve para imprimir una tabla de una o dos variables
- `mean` sirve para estimar la media de una o más variables.

Comentarios

- Cuando escribimos código siempre es importante hacer comentarios y anotaciones tanto para que otros puedan entender y reutilizar el código así como para nosotros mismos.
- Stata no correrá las lineas de código que comienzen con `*`, `//` o `/*`.
- También se puede utilizar al final de una línea de código
- Se puede quebrar una línea utilizando `///`

Ayuda y recursos

- Para obtener ayuda sobre un comando se puede escribir `help nombre_del_comando`.
- La página de Stata tiene muchos tutoriales y videos.
- Statalist es un foro donde se puede hacer preguntas pero sobretodo consultas preguntas ya hechas sobre código en Stata
- UCLA tiene un portal de Stata donde usualmente hay muy buenos tutoriales
- Stata Cheat Sheets
- Stata Journal es una revista que publica articulos sobre estadística y análisis de datos y el uso efectivo de Stata.

Importar datos en Stata

Importar datos en Stata

- Nuestros do files siempre comenzarán diciéndole a Stata en qué carpeta queremos trabajar. Debemos definir la carpeta en la que se encuentran los datos que queremos utilizar y donde irán las tablas, gráficos o datos transformados que exportaremos más adelante.
- Para fijar el directorio utilizamos el comando `cd`
- Para importar los datos (si ya se encuentran en formato `.dta`), utilziamos la función `use`
- Supongamos que yo tengo una base de datos en formato stata llamada "mis_datos" dentro de una carpeta llamada "mi_carpeta". Para importarla correctamente a Stata debo correr:

```
cd "C:Usuarios/carpeta_general/mi_carpeta" // Fijo directorio

use "mis_datos.dta" // Importo base
```

Archivos en formato Stata (.dta)

- Los archivos de datos en formato Stata tienen la extensión .dta
- Pueden tener hasta 2000 variables (aproximadamente) y millones de observaciones.

Importar datos desde otros formatos

Podemos también importar en Stata archivos de datos en otros formatos:

```
import delimited "wb_paises.csv" // Archivos .csv

import excel "wb_paises.xlsx", firstrow // Archivos .xlsx
```

Ejercicio

Importar a Stata la base de datos wb_paises desde formato Stata (.dta) y excel (.xlsx)

Explorar datos

Summarize

Con summarize obtenemos un resumen a través de distintas estadísticas descriptivas sobre una o más variables. Ver documentación

```
summarize pob_mujeres pob_hombres pob_total
```

```
. sum poblacion_mujeres poblacion_hombres poblacion_total

    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
poblaci~eres |        193    1.95e+07    7.03e+07       47079    6.83e+08
poblaci~bres |        193    1.98e+07    7.46e+07       46419    7.20e+08
poblacio~tal |        216    3.51e+07    1.37e+08       10678    1.40e+09

.
end of do-file
```

List

Con `list` podemos listar los valores de una o más variables. Se usa frecuentemente para chequeos específicos o en bases de datos con pocas observaciones. Ver documentación

```
list country_name pob_total pob_mujeres pob_hombres if region == 5
```

```
. list country_name pob_total pob_mujeres pob_hombres if region == 5

     +-------------------------------------------------------------+
     | country_name   pob_total   pob_muj~s   pob_hom~s           |
     |-------------------------------------------------------------|
 23. | Bermudas         63919          .           .              |
 36. | Canadá        37065084    18677663    18387421              |
 62. | Estados Unidos 3.268e+08  1.651e+08   1.617e+08             |
     +-------------------------------------------------------------+
```

Tablas con tabulate o tab

Como vimos anteriormente podemos utilizar `tabulate` o `tab` para crear tablas de una o dos variables.

```
tabulate region
```

```
. tabulate region
```

region	Freq.	Percent	Cum.
East Asia & Pacific	37	17.05	17.05
Europe & Central Asia	58	26.73	43.78
Latin America & Caribbean	42	19.35	63.13
Middle East & North Africa	21	9.68	72.81
North America	3	1.38	74.19
South Asia	8	3.69	77.88
Sub-Saharan Africa	48	22.12	100.00
Total	217	100.00	

Tablas con table

- Con `table` podemos agregar estadísticos como la media a una tabla.
- Con la opción `contents` o su abreviación `c` podemos usar estadísticos. Por más detalles ver documentación.

```
table region, c(n tasa_desempleo mean tasa_desempleo sd tasa_desempleo min tasa_desempleo max tasa_desemple
```

```
. table region, c(n tasa_desempleo mean tasa_desempleo sd tasa_desempleo min tasa_desempleo max tasa_desempleo)
```

region	N(tasa_de~o)	mean(tasa~o)	sd(tasa_d~o)	min(tasa_~o)	max(tasa_~o)
East Asia & Pacific	29	4.0448276	3.386695	.1	14.8
Europe & Central Asia	49	7.6632653	4.649314	2.2	20.7
Latin America & Caribbean	31	7.816129	4.514576	1.7	19.2
Middle East & North Africa	21	10.219048	7.732116	.1	26.3
North America	2	4.85	1.343503	3.9	5.8
South Asia	8	5.0625	2.686441	2.4	11.2
Sub-Saharan Africa	47	8.4893617	7.162743	.5	26.9

.

Tablas con table

`table` permite también agregar estadísticos a tablas bivariadas. A su vez, `table` tiene otras opciones como row y column

```
table region income_group, c(mean tasa_desempleo) row
```

```
. table region income_group, c(mean tasa_desempleo) column
```

region	income_group				
	Low income	Lower middle income	Upper middle income	High income	Total
East Asia & Pacific	2.6	2.8	3.16	5.9363636	4.0448276
Europe & Central Asia		7.125	9.93125	6.4862069	7.6632653
Latin America & Caribbean		6.4833333	8.2176471	8.3857143	7.91
Middle East & North Africa	10.95	15.671429	15.4	2.675	10.219048
North America				4.85	4.85
South Asia	11.2	3.95	5.6		5.0625
Sub-Saharan Africa	5.5782609	9.2277778	17.433333		8.4893617

Ejercicio

Crear una tabla con la media de acceso a electrcididad y emisiones de co_2 según grupos de ingresos

Variables

Variables

- Los nombres de las variables pueden tener hasta 32 caracteres pero dentro de lo posible restringirlas a menos de 12 y no utilizar mayúsculas
- Cuando nombramos variables es mejor tratar de relacionarlas a su contenido. Por ejemplo supongamos que tenemos dos variables de edad una numérica con la edad en años y otra con cuatro categorías de edad (18 a 29; 30 a 49; 50 a 64 y 65 o más). Es mejor nombrarlas con algo que refiera a su contenido como `edad_num` y `edad_cat` que como `edad_1` y `edad_2` o aun peor `var_2` y `var_3`.
- Es una buena práctica asignar una etiqueta a la variable con una descripción. Por ejemplo nuestra variable `edad` puede tener una etiqueta "Edad de los encuestados (númerica, en años)" "Edad de los encuestados (en 4 categorías)"

Tipos de variables

- Los softwares estadísticos tienen distintos tipos de almacenamiento de datos para determinar cómo y qué datos guardar sobre una variable o valor.
- A grandes rasgos existen dos grandes tipos de almacenamiento: numérico y strings (cadenas o texto).
- El formato de texto lo usamos para variables que contienen texto no estructurado o identificadores.
- En general, las variables ordinales o categóricas en Stata las almacenamos como numéricas con su correspondientes etiquetas para cada valor, debido a que para realizar varios análisis estadísticos (como regresiones) Stata necesita que estén en ese formato en lugar de en formato texto. Es importate asignar etiquetas cuando almacenamos estas variables como numéricas.
- A su vez, dentro de cada tipo (*data type*) existen distintos formatos de almacenamiento (*storage type*), que determinar por ejemplo cuántos valores luego de la coma se almacenan ej. (byte, int, etc.)
- Por más información utilizar: `help datatype`

Cambiar el tipo de una variable

Un error frecuente al crear una variable es ingresar una letra por error en una variable que iba a ser numérica. En ese caso Stata transformará toda la variable a string (porque las strings pueden contener números pero las numéricas no pueden contener texto). Para resolver este tipo de situaciones Stata tiene comandos que transforman las variables de un tipo a otro.

De carácteres (string) a numérica:

```
destring varlist, replace
```

De numérica a string:

```
decode varlist, replace
```

Consultar tipos de datos, almacenamiento y etiquetas

- Con el comando `describe` obtenemos para cada variable (o las variables seleccionadas) el nombre, el tipo de almacenamiento y en caso de existir el nombre de las etiquetas (value label) y la etiqueta de la variable (variable label)
- Con el comando `codebook` obtenemos una descripción de cada variable que incluye el tipo de variable, cantidad de valores únicos, valores perdidos, ejemplos, advertencias y frecuencias con sus etiquetas o percentiles en caso de aplicar.
- Con el comando `labelbook` obtenemos un libro de códigos solo para las variables que tienen etiquetas en sus valores (value label) con sus correspondencias y datos adicionales.

Manipular etiquetas

Para etiquetar una variable:

```
label variable pob_total "Población total del país"
```

Se necesitan dos pasos en Stata para definir etiquetas. primero se crea la etiqueta que asocia codigos numéricos con etiquetas con el comando `label define` para luego asignar esa etiqueta creada (que lleva un nombre) a una variable ya existente, con el comando `label values`. Se puede aplicar la misma etiqueta a muchas variables cuando utilizamos las variables tienen la misma escala.

```
## Crear nueva variable (si el país tiene al menos 10 millones de habitantes)
generate pais_10m = 0
replace pais_10m = 1 if pob_total >= 10000000
ta pais_10m

## Primero creo las etiquetas de los valores con el nombre "pais_10m"
label define pais_10m 0 "Menos de 10M habitantes" 1 "Más de 10M habitantes", replace

## Luego pego las etiquetas de los valores a la variable
label values pais_10m pais_10m

## Asigno también etiqueta a la variable (descripción)
label variable pais_10m "Si el país tiene una población mayor o menor a 10 millones de habitantes"
```

Valores perdidos

Valores perdidos

- La mayoría de los softwares estatdísticos prevee y distingue a los casos perdidos (valores para los que no tenemos información), en Stata los valores perdidos son representados por un punto `.`
- En general Stata omite los casos perdidos al aplicar un comando pero es recomendable ver la documentación de cada comando si tenemos casos perdidos para estar seguros.
- Por ejemplo, si queremos crear una variable vacía, es decir, solo con valores perdidos:

```
generate newvar = .
```

- Los datos de encuestas muchas veces utilizan códigos como 88 o 99 para señalar los datos perdidos (ej. cuando los enceustados no quieren responder una pregunta). Tener cuidado de que definir en Stata que esos son casos perdidos.

```
replace varname = . if varname == 88
replace varname = . if varname == 99
```

Consultar valores perdidos

A la hora de consultar si nuestros datos tienen casos perdidos tenemos varias opciones.

- Usando `misstable` podemos chequear las observaciones perdidas por variable.
- Con `list` podemos chequear que observaciones tienen datos perdidos
- Con `tabulate`, podemos poner la opción missing para ver los datos perdidos

```
misstable summarize

list country_name if missing(income_group)

tabulate income_group
tabulate income_group, missing
```

Recodificar con valores perdidos

Tenemos que tener especial cuidado cuando recodificamos variables que tienen casos perdidos. Por ejemplo, si queremos recodificar la variable de `tasa_desempleo` en mayor o menor a 10 y que los casos perdidos por supuesto se mantengan como casos perdidos, este código no funcionará:

```
generate tasa_desempleo_rec = .
replace tasa_desempleo_rec = 1 if tasa_desempleo >= 10
replace tasa_desempleo_rec = 0 if tasa_desempleo < 10
```

Debemos agregar:

```
replace tasa_desempleo_rec = . if tasa_desempleo == .
```

Ejercicio

Con la base de datos de wb_paises identificar cuántos valores perdidos hay para la variable area_selvatica y cuáles son esos países

Guardar datos

Guardar datos

- Muchas veces cuando escribimos código en un do-file, exportamos lo que necesitamos (si es que queremos exportar algo) y es suficiente con tener la base de datos original y el do-file.
- Sin embargo, a veces queremos guardar una nueva base de datos con los cambios que le realizamos. En estos casos lo mejor suele ser no reemplazar la base de datos original, sino que guardar una nueva con otro nombre.
- Para guardar los datos modificamos usamos:

```
save "carpeta_ejemplo/data_nueva", replace
```

Sintáxis básica

Sintáxis básica

```
[by varlist:] command [varlist] [=exp] [if exp] [weight] [,options]
```

- **command** es usualmente un verbo y el único elemento imprescindible, generalmente viene acompañado del nombre de una o más variables. Muchas veces se pueden abreviar ej. `summarize` puede usarse como `sum`
- **varlist** esto refiere a la variable o las variables a las que se le aplica el comando/función. También se pueden abreviar. También cuando se quiere aplicar a un conjunto de variables por ejemplo que empiezen con determinados caracteres. Ej: `summarize pob_*` nos devolverá un resumen de todas las variables que comiencen con "pob_"

```
summarize pob_hombres pob_mujeres
```

Sintáxis básica

```
[by varlist:] command [varlist] [=exp] [if exp] [weight] [,options]
```

- **=exp** es para cuando nuestro comando genera una nueva variable y debemos definir que valor tendrá. En este caso generamos una variable nueva llamada `log_pob_total` que es el logaritmo de la variable ya existente `pob_total`, y otra donde no necesitamos usar un comando en la expresión simplemente dividimos la población por 1.000.000 para obtener esa variable en millones.

```
generate log_pob_total = log(pob_total)
generate pob_total_m = pob_total/1000000
```

Sintáxis básica

```
[by varlist:] command [varlist] [=exp] [if exp] [weight] [,options]
```

- **if exp** la acción de un comando puede aplicarse a una parte específica de nuestros datos. Por ejemplo, supongamos que quiero un resumen de la variable pob_total pero solo para los países de América Latina. Entonces debo fijarme el valor de América Latina en la variable región (3), y establecer el condicional luego del uso estandar del comando.

```
summarize pob_total if region == 3
```

```
. summarize poblacion_total if region == 3

    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
poblacio~tal |         42    1.52e+07    3.76e+07       29795    2.09e+08

.
end of do-file
```

Sintáxis básica

```
[by varlist:] command [varlist] [=exp] [if exp] [weight] [,options]
```

- **options** la mayoría de los comandos tienen opciones que se definen al final, luego de una coma. Por ejemplo, la variable summarize tiene una opción "detail" para que se agreguen más estadísticos al resumen

```
summarize pob_total, detail
```

```
. summarize poblacion_total, detail

                    poblacion_total
-------------------------------------------------------------
      Percentiles      Smallest
 1%         17911         10678
 5%         40895         11505
10%         84073         17911       Obs                 216
25%      766701.5         29795       Sum of Wgt.         216

50%       6572034                     Mean           3.51e+07
                        Largest       Std. Dev.      1.37e+08
75%      2.50e+07       2.68e+08
90%      6.65e+07       3.27e+08       Variance       1.89e+16
95%      1.26e+08       1.35e+09       Skewness       8.799001
99%      3.27e+08       1.40e+09       Kurtosis       85.46565

.
end of do-file
```

Sintáxis básica

```
[by varlist:] command [varlist] [=exp] [if exp] [weight] [,options]
```

- **weight** sirve para ponderar los datos, simplemente se define el tipo de ponderador y el nombre del mismo entre [] ej.

```
mean [aweight = ponderador]
```

Sintáxis básica

```
[by varlist:] command [varlist] [=exp] [if exp] [weight] [,options]
```

- **by** un comando muy importante es by, y sirve para decirle a Stata que repita un comando para distintos grupos de observaciones (definidos por los valores de la variable especificada luego del by). Para utilizar el comando by es necesario utilizar el comando sort anteriormente. Por ejemplo, si queremos calcular la media de `pob_total` según región podemos hacer:

```
sort income_group
by income_group: summarize pob_total
```

Sintáxis básica

```
. by income_group: summarize poblacion_total
```

-> income_group = High income

Variable	Obs	Mean	Std. Dev.	Min	Max
poblacio~tal	79	1.50e+07	4.18e+07	10678	3.27e+08

-> income_group = Low income

Variable	Obs	Mean	Std. Dev.	Min	Max
poblacio~tal	26	2.41e+07	2.44e+07	1874304	1.09e+08

-> income_group = Lower middle

Variable	Obs	Mean	Std. Dev.	Min	Max
poblacio~tal	55	5.89e+07	1.86e+08	112640	1.35e+09

Sintáxis básica

Para no tener que usar sort primero y by después, se puede utilizar bysort directamente y ahorrarse un paso:

```
bysort income_group: summarize pob_total
```