Una breve guía para el uso de STATA

Mg. Monserrat Serio*†
Universidad Nacional de Cuyo

Versión: Setiembre, 2015

*Facultad de Ciencias Económicas, Universidad Nacional de Cuyo, Becaria CONICET, Mendoza, 5500 Argentina. E-mail: monserrat.serio@fce.uncu.edu.ar.

†Agradezco la valiosa colaboración del Ing. Andrés Eduardo Valero en la construcción de esta guía. Todos los errores u omisiones son exclusiva responsabilidad de la autora.

Índice

1. Introducción al software estadístico y de análisis de datos Stata

Stata brinda un entorno de programación que tiene lenguaje propio. Al momento de escribir este documento, la versión más reciente disponible del software que contiene las últimas actualizaciones, es Stata 14. Cuando trabajamos con Stata es importante considerar qué tipo de versión es necesaria para el análisis de datos que queramos realizar. No obstante, esta guía es compatible con versiones de Stata anteriores. A su vez existen diferentes productos de Stata de la misma versión que principalmente difieren en la cantidad de observaciones y variables que puede procesar. Según el análisis a realizar y el tipo de computadora en la que trabajemos se tendrán más ventajas con uno u otro producto de Stata. Stata/IC procesa como máximo número de variables 2.047 y 2.140 millones de observaciones. Mientras que Stata/SE para la misma cantidad de observaciones admite 32.767 variables. El Stata/MP es la versión más rápida y grande de Stata, procesa un máximo de 32.767 y más de 20.000 millones de observaciones. Usualmente para trabajos empíricos de economía que hacen uso de bases de microdatos es conveniente utilizar Stata/SE. También existe Small Stata que es una versión limitada para estudiantes que solo admite 99 variables y 1200 observaciones. Para uso en páginas web y desarrollo de aplicaciones exite Numerics by Stata que provee todo el poder de procesamiento del sofware estadístico STATA en entornos embebidos.

Hay que tener en cuenta que Stata utiliza la memoria RAM de la computadora. Una ventaja es que la ejecución de comandos es muy rápida, ya que no se accede al disco rígido. La desventaja es que el tamaño de los conjuntos de datos (data sets) que podemos procesar se encuentra limitado por la cantidad de memoria RAM de la PC. El mínimo de memoria RAM necesario para correr Stata es de 512MB sin embargo es recomendable utilizarlo en computadoras con mas de 1Gb de RAM. Si bien Stata funciona con cualquier procesador está idealmente diseñado para funcionar en procesadores multinucleo. Cada tipo de Stata esta diseñado para maximizar su funcionamiento en un tipo de procesador determinado.

Stata está disponible para diferentes versiones de sistemas operativos: Windows desde Vista hasta 8.1 o más moderno (32bits o 64 bits), Linux en sus diferentes distribuciones, incluso Mac con OS X 10.7 o más moderno. Es necesario considerar que si contamos con una versión de Stata para un sistema operativo las misma no funcionará en otro pero su licencia continuará siendo valida.

Para determinar el producto que necesitamos de Stata, cuál es el hardware y sistema operativo mínimo y cuál es el ideal para maximizar su funcionamiento debemos consultar la pagina Web de Stata **http://www.stata.com/products** en la sección **Supported platforms**

Una pregunta importante que debemos hacernos es ¿Cómo sé si Stata es adecuado para el análisis que quiero realizar? En algunos casos puede ser más conveniente utilizar MatLab, R, excel o algún otro software, todo dependerá del problema que se desee estudiar. En particular si trabajaremos con grandes bases de datos, en especial microdatos provenientes de encuestas, Stata presenta muchas ventajas que hacen que sea beneficioso su uso en estos casos.

2. Comenzando a utilizar Stata

Es aconsejable a la hora de utilizar Stata armar una carpeta de trabajo en la computadora, ya que Stata genera diversos tipos de archivos (.do, .log, .dta, .xls, étc.). Dentro de dicha carpeta armar

sub-carpetas para los archivos do, log, gráficos, resultados y bases de datos. La carpeta **bases de datos** contendrá las bases de datos a utilizar en Stata. Hay que considerar que las bases de datos deben estar en un formato adecuado para poder abrirlas.
Por ejemplo:

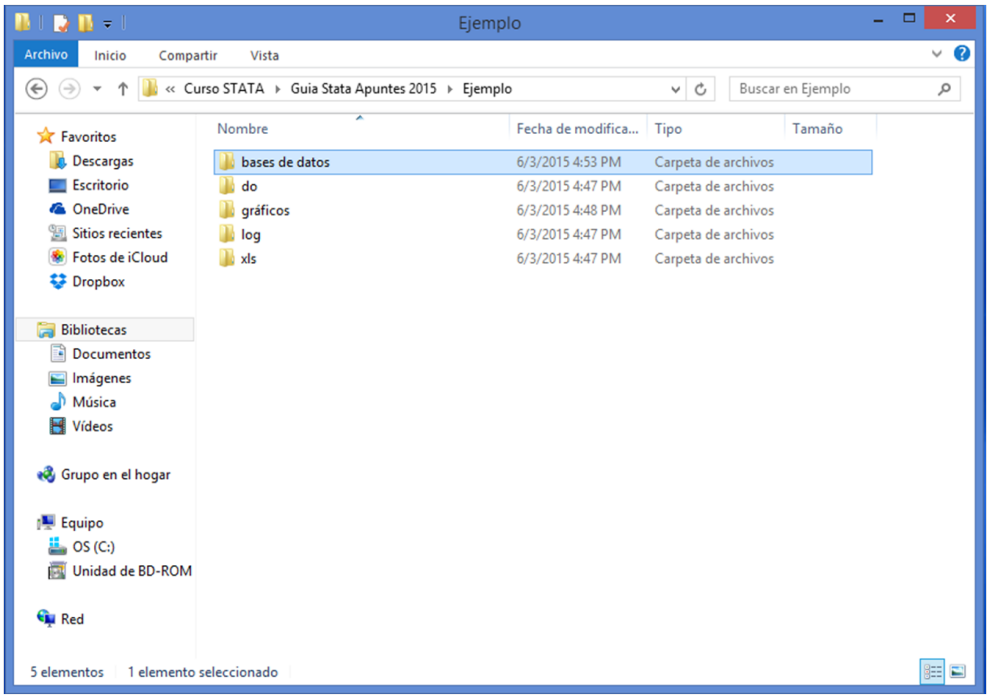

Figura 1

3. Interface

La interface de Stata muestra en el centro de la pantalla la **ventana de resultados** donde aparecerán los comandos y sus resultados. A la derecha de la pantalla encontraremos una lista de las variables de la base de datos (**Variables**) y debajo sus propiedades (**Properties**). En el lado izquierdo Stata muestra la lista de comandos utilizados (**Review**). Por último, en la parte inferior hay una ventana que nos permite trabajar con los comandos (**Command**). (Ver Figura 2).

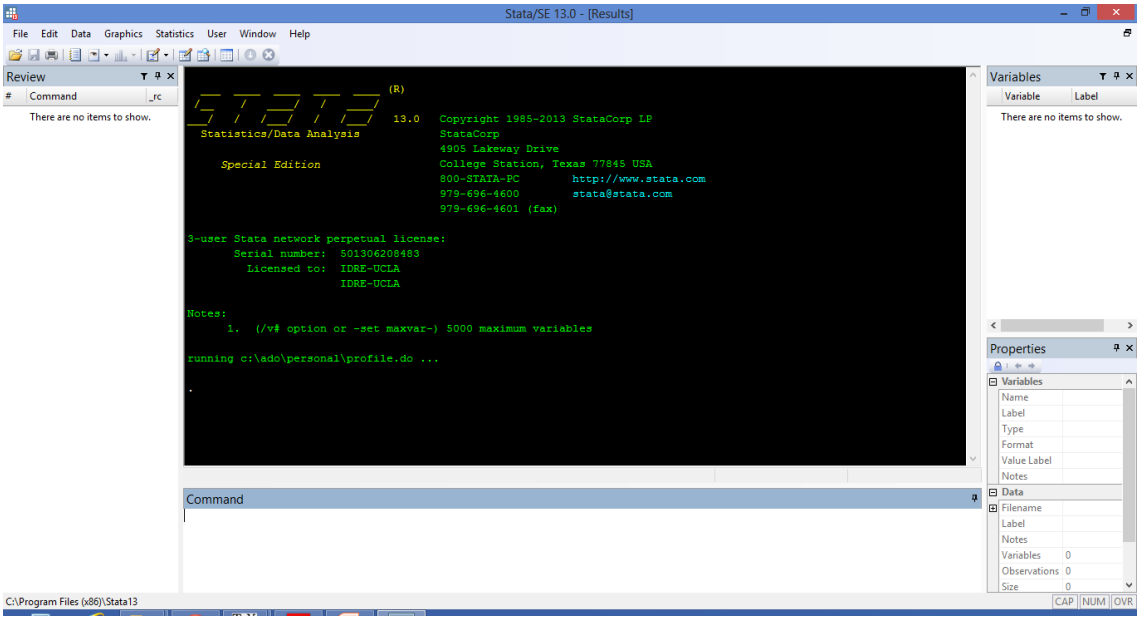

Figura 2

En cuanto a la barra de herramientas, será muy poco utilizada mientras trabajemos en Stata.

4. Archivos DO, LOG, DTA

Los archivos DO son aquellos donde se programan los comandos del trabajo. Es el entorno donde vamos a trabajar la mayoría del tiempo. Es preciso guardar regularmente los resultados encontrados en archivos LOG, que luego podrán ser abiertos desde cualquier editor de texto como por ejemplo block de notas (txt).

Si bien Stata puede abrir diferentes formatos de bases de datos como .csv, .xls, otros; el formato de bases de datos de Stata tiene la extensión .dta. Cabe señalar que Stata nos puede preguntar si realiza cambios en la base de datos a medida que vamos trabajando sobre ella. Nunca procederemos a guardar los cambios en dicha base de datos. Esto es porque el archivo do, podrá recrear los resultados siempre que tengamos la base de datos original. No obstante, es conveniente tener un back up de la base de datos original en caso que por error se altere la misma.

Existen otros tipos de archivos en Stata, conocidos como archivos "ado". Estos archivos son archivos de programación. Cualquier comando de Stata es un programa con la extensión ".ado". Estos pueden encontrarse dentro de la carpeta donde está instalado Stata, en la subcarpeta ado. También puede suceder que exista un comando fuera de Stata, que nosotros mismos u otra persona ha programado. En el caso de tener el programa de dicho comando, éste debe ser guardado como archivo .ado en la carpeta mencionada anteriormente según la letra inicial del comando. Cabe señalar que este tipo de archivos que no son nativos de Stata, pueden contener errores teóricos o de programación. Una revisión reiterada en cada caso será importante de hacer.

5. Comandos básicos

Un comando es una orden o instrucción a través de una palabra clave predefinida por el lenguaje de programación de Stata para realizar una acción determinada ej. una estimación, generación de variable, etc. Es importante resaltar que el idioma de Stata por defecto es el inglés y, por lo tanto, la mayoría de los nombres de los comandos son en inglés.

Lo primero que vamos a hacer es abrir un archivo **do** y vamos a guardar dicho archivo en la carpeta **do** con el nombre deseado como por ejemplo: Ejemplo.do. Es importante que cada archivo **do** esté lo mejor comentado posible, es decir que escribamos por qué utilizamos cada comando y qué queremos hacer en cada caso. Esto porque, en primer lugar, si otra persona quiere replicar nuestros resultados le será más fácil con estas notas de ayuda y, segundo, porque hasta nosotros mismos pasado un lapso de tiempo podemos olvidar fácilmente que pretendíamos hacer en cada caso.

Los comentarios en Stata se escriben con un asterisco, en caso que quisiéramos escribir un párrafo que ocupe varias líneas del **do**, en vez de colocar un asterisco en cada línea se pueden utilizar los siguientes operadores: /* y */. Ver archivo Ejemplo.do en el apéndice.
Tener en cuenta que si se utiliza un comando de una versión anterior de Stata que ha sido reemplazado y modificado en la nueva versión, Stata en algunos casos avisará al usuario mediante una frase en la ventana de resultados "(note: you are using old command syntax; see [D] command for new syntax)". En caso que el comando sea erróneo o no exista, Stata arroja **unrecognized command: "el comando erróneo"**.

En general los comandos presentan la siguiente sintaxis:

comando [varlist] [if] [in] [weight] [, options]

donde primero aparece el nombre del comando, luego la lista de variables con las cuáles vamos a trabajar, la posibilidad de trabajar con condicionales usando el *if* o el *in*. En algunos casos también podemos trabajar con observaciones ponderadas para ello tenemos que especificar a Stata cuál es la variable ponderadora *weight* y, por último, todo comando presenta distintas opciones que pueden ser especificadas luego de escribir al final de cada comando una coma.

5.1. Abrir archivos

Al empezar a trabajar debemos asegurarnos que Stata no esté por error trayendo información de otro proyecto de trabajo. Para ello utilizamos el comando **clear**. La primer línea del archivo do será:

clear all

Aquí le pedimos a Stata que si hay alguna información retenida que la borre y que comience de cero.

Muchas veces la cantidad de resultados que arroja Stata en la pantalla de resultados son demasiados, y sólo puede mostrar una parte de ellos. En estos casos aparecerá la siguiente leyenda "-more-" como se muestra en la Figura 3. Para continuar visualizando los datos tenemos que hacer click en dicha leyenda. Cuando se trabaja, en muchos casos, puede no ser eficiente clickear permanentemente para ver los resultados. Para ello se incluye en el **do** el comando **set more off**.

```
. des

Contains data from bases de datos\Individual_t414.dta
  obs:        60,959
 vars:           173                          19 Mar 2015 14:41
 size:    29,016,484
-------------------------------------------------------------------------------
              storage   display    value
variable name   type    format     label      variable label
-------------------------------------------------------------------------------
CODUSU          str8    %8s                   Código para distinguir viviendas
nro_hogar       byte    %1.0fc                Código para distinguir hogares
componente      byte    %2.0fc     componente
                                              Número de componente
h15             byte    %1.0fc     h15        Entrevista individual realizada
ano4            int     %4.0fc                Año de relevamiento
trimestre       byte    %1.0fc     trimestre
                                              Número de trimestre
region          byte    %2.0fc     region     Código de Región
mas_500         str1    %1s                   Aglomerados según tamaño
aglomerado      byte    %2.0fc     aglomerado
                                              Aglomerado
pondera         int     %6.0fc                Ponderación
ch03            byte    %2.0fc     ch03       Relación de parentesco
ch04            byte    %1.0fc     ch04       Sexo
--more--
```

Figura 3

Si bien no es necesario para comenzar a trabajar en Stata decirle en qué carpeta de trabajo estaremos trabajando, es sumamente conveniente cuando el proyecto de trabajo es complejo. Entonces vamos a indicar a Stata en qué lugar de la computadora vamos a estar trabajando, es decir vamos a indicar cuál es nuestro directorio de trabajo. Esto se hace con el comando **cd** de la siguiente manera:

cd "directorio de cada computadora personal"

Por ejemplo:

cd "C:..."

Distintos comandos son utilizados para abrir determinadas bases de datos. Si queremos abrir una base de datos .dta utilizamos el comando **use**, si fuese el caso de una base de datos en formato excel usaremos el comando **import** y si fuera el caso por ejemplo de una base de datos delimitada por comas (archivos con la extensión .csv) podemos utilizar el comando **insheet using**. Estos tres comandos nos permiten abrir distintas bases de datos de diferentes formatos.

Cada uno de los comandos podemos utilizarlos de la siguiente manera:

insheet using "archivo.csv", clear

import excel "archivo.xlsx", firstrow clear

use "archivo.dta", clear

Observemos que en todos los casos una de las opciones utilizada es **clear**, porque si ya existe una base de datos abierta, le pedimos a Stata que la borre y que abre la base de datos que le estamos

indicando. Observar que dado que ya le indicamos el directorio de trabajo, solo resta indicar el nombre del archivo que queremos abrir. No obstante, podría pasar que no hubiésemos indicado el directorio de trabajo o que la base de datos se encuentre en una subcarpeta del directorio que hemos indicado. Estos casos pueden ser fácilmente adaptados de la siguiente manera:

Caso 1: la base de datos se encuentra en una subcarpeta del directorio asignado
use "nombre subcarpeta\archivo.dta", clear

Caso 2: la base de datos se encuentra en otro directorio distinto al asignado
use "directorio donde se encuentra la base de datos distinto al asignado\archivo.dta", clear

Caso 3: no se asignó un directorio de entrada, luego hay que especificar el directorio completo hasta la carpeta donde el archivo se encuentra.
use "directorio donde se encuentra la base de datos\archivo.dta", clear

También existen comandos para exportar bases de datos como **export** o **outsheet**.

Cuando trabajamos con bases de datos puede pasar que las mismas ya estén ordenadas y armadas por alguna institución o colega. Sin embargo, en la mayoría de los casos esto no es así. Generalmente las bases de datos, en especial las referidas a encuestas de hogares, vienen en varios archivos. Por ejemplo tomemos el caso de la Encuesta Permanente de Hogares de Argentina (EPH) que realiza el INDEC. Si entramos a la página web y descargamos la encuesta de cualquier trimestre vamos a encontrar 2 archivos: un archivo que se llama "Individual" que contiene información de las personas encuestadas y un archivo "Hogar" que contiene información del hogar. Es muy posible que uno quiera trabajar con toda la información que provee la encuesta y no solo con uno de los archivos. En estos casos se deben juntar dichos archivos.

En Stata hay dos comandos importantes que nos permiten hacer esto: **append** y **merge**. El primero de ellos acumula filas en uno de los archivos que es tomado como base. El segundo agrega columnas en el archivo base.

Ambos comandos serán mostrados a partir de ejemplos. Supongamos que tenemos una base de datos de las exportaciones de 70 países, en este caso tendríamos un archivo con 70 filas (cada fila contiene información de cada país). Resulta que conseguimos la base de datos de 20 países más. Quisiéramos anexar esta última base de datos con la primera para ello utilizamos el comando **append**. Primero abrimos la primer base de datos y luego anexamos la segunda base de la siguiente manera:

use "basededatos70.dta", clear
append using "basededatos20.dta", generate(nuevabase)

Por otro lado, podemos tener el caso en que las dos bases de datos proveen información de las mismas unidades de análisis. Este es el caso de las EPH, la base de datos de Hogar provee información del hogar de cada uno de los individuos de la base de datos Individual. Para unir ambas bases de datos es necesario una variable codificada que permita unir cada individuo con su hogar respectivo. En las EPH esta variable recibe el nombre de CODUSU. La misma debe aparecer en ambas bases de datos y se debe llamar exactamente igual en ambas bases. Entonces lo que vamos a hacer es asignar a cada individuo información de su hogar, pero esto no es una tarea tan sencilla ya que no siempre hay una correspondencia 1 a 1. Supongamos un hogar de cuatro individuos, entonces

la correspondencia es 4 a 1, es decir a cuatro individuos se les asignará la misma información de un hogar pero no cualquier hogar solo la información del de ellos. Para esto se utiliza el comando **merge**. Una vez más tenemos primero abierto un archivo, por ejemplo la base de datos Individual y luego le agregamos la información del hogar de la siguiente manera. Le decimos a Stata que la variable que nos indica cómo unir el hogar con cada individuo es CODUSU. Verá el lector que para poder usar el **merge** la variable de codificación deberá estar ordenada en ambas bases de datos con el comando **sort** (ver ejemplo en el apéndice).

use "Individual t414.dta", clear

sort CODUSU

merge $m:m$ CODUSU using "Hogar t414.dta"

Por otro lado, el comando **merge** nos pide que expliquemos cómo es la correspondencia. La misma puede ser 1 por 1, o múltiples correspondencias como m por 1, 1 por m o m por m. Este comando genera una variable que llamada "merge" que nos indica cómo ha sido la unión, cuando todos los casos pudieron asociarse la variable sólo tomará el valor 3. En otro caso tomará valores 1 o 2 dependiendo de cómo la unión no se pudo concretar. Cada analista deberá analizar cada caso en particular. En la Figura 4 se presenta el merge de las bases de datos de la EPH del cuarto trimestre del año 2014.

```
. merge m:m CODUSU using "bases de datos\Hogar_t414.dta"
(label trimestre already defined)
(label region already defined)
(label aglomerado already defined)

    Result                           # of obs.
    -----------------------------------------
    not matched                             0
    matched                            60,959  (_merge==3)
    -----------------------------------------

. tab _merge

      _merge |      Freq.     Percent        Cum.
-------------+-----------------------------------
matched (3)  |     60,959      100.00      100.00
-------------+-----------------------------------
       Total |     60,959      100.00
```

Figura 4

En algunos casos podríamos desear guardar la nueva base de datos que surge al unir distintas bases de datos. Para guardar una base de datos utilizamos el comando **save** con la opción "replace", ya que si existe una base de datos con el nombre con la que estamos guardando la nueva base, Stata reemplaza el archivo existente por este.

save "EPH t414.dta", replace

Nuevamente es posible guardar la base de datos en el directorio de trabajo, en alguna subcarpeta del directorio o en otro directorio.

5.2. Archivo LOG

Como ya explicamos es posible que deseemos guardar todos los resultados que muestra la pantalla principal de Stata en un archivo para poder analizarlo en otro momento sin tener que correr el archivo **do**. Esto se logra generando un archivo **log** de la siguiente manera:

capture log close
log using "nombre del archivo log", replace

... Escribo el código ...

log close

La primer línea captura cualquier archivo **log** que se encuentre abierto en Stata y lo cierra. Esto es para no escribir encima de otro archivo **log** de otro proyecto de trabajo. La segunda línea crea el archivo **log**, usando un nombre para el archivo que se le quiera dar (permite seleccionar la carpeta donde lo va a guardar). La última línea es **log close** y se le indica a Stata que cierre el archivo **log**. Entre la segunda línea y esta última es que se debe armar el archivo **do** con todos los comandos cuyos resultados queramos guardar. Todo aquello que quede fuera no será guardado. Podemos abrir el archivo **log** en cualquier editor de texto como por ejemplo un block de notas.

5.3. Descripción base de datos

Stata admite bases de datos de corte transversal (cross-section), de tiempo y de panel. Por defecto considera la base de datos como cross-section, en caso de trabajar con series de tiempo hay que indicarle esto a Stata a través del comando **tsset**.

El comando **xtset** declara a Stata que los datos con los que se trabajará son datos de panel. Para este comando hay que designar cuál es la variable de panel y cuál es la variable temporal. Por ejemplo: supongamos que tenemos una base de datos de panel de países a lo largo de una década. En la base tenemos la variable países que hace referencia al país observado y una variable años que indica el período de los datos. En este caso escribimos,

xtset paises años

Una vez abierta la base de datos en Stata, se puede realizar una descripción detallada de la misma a partir del comando **describe**. En la pantalla de resultados este comando arroja información sobre la cantidad de observaciones, variables y tamaño de la muestra; así como información sobre cada una de las variables (ver Figura 5).

```
. des

Contains data from bases de datos\Individual_t414.dta
  obs:        60,959
 vars:           173                          19 Mar 2015 14:41
 size:    29,016,484
-------------------------------------------------------------------------------
              storage   display    value
variable name   type    format     label      variable label
-------------------------------------------------------------------------------
CODUSU          str8    %8s                   Código para distinguir viviendas
nro_hogar       byte    %1.0fc                Código para distinguir hogares
componente      byte    %2.0fc     componente
                                              Número de componente
h15             byte    %1.0fc     h15        Entrevista individual realizada
ano4            int     %4.0fc                Año de relevamiento
trimestre       byte    %1.0fc     trimestre
                                              Número de trimestre
region          byte    %2.0fc     region     Código de Región
mas_500         str1    %1s                   Aglomerados según tamaño
aglomerado      byte    %2.0fc     aglomerado
                                              Aglomerado
pondera         int     %6.0fc                Ponderación
ch03            byte    %2.0fc     ch03       Relación de parentesco
ch04            byte    %1.0fc     ch04       Sexo
--more--
```

Figura 5

A partir de la Figura 5 podemos observar para cada variable el nombre, el tipo de variable, el formato, y su etiqueta. A su vez cada variable puede ser numérica o de texto ("string"). Los tipos de formato de las variables numéricas son:

Tipo	Mínimo	Máximo	bytes
byte	-127	100	1
int	-32,767	32,740	2
long	-2,147,483,647	2,147,483,620	4
float	-1.70141173319*$10^3$8	1.70141173319*$10^3$8	4
double	-8.9884656743*$10^3$07	8.9884656743*$10^3$07	8

Fuente: StataCorp. 2013. Stata 13 Help Manual.

Las variables "string" por su parte son variables de texto que no pueden ser utilizadas para cálculos numéricos. En algunos casos variables que contienen datos con los cuales queremos trabajar, aparecen en la base de datos como "string". En este caso podemos convertirlas en variables numéricas a través del comando **destring**.

En cuanto el tamaño de la base de datos, para versiones de Stata anteriores a Stata 12 era necesario designar la cantidad de memoria que necesitaba Stata para abrir la base de datos con el comando **set memory**. En las últimas versiones ya no es necesario, porque Stata lo hace automáticamente al abrir la base de datos a utilizar. Cabe señalar que Stata utiliza la memoria de la computadora, con lo cual si la base de datos que se está abriendo es demasiado grande y la computadora no posee mucho espacio en su memoria puede ser posible que se vuelvan más lentos otros programas abiertos

en la computadora. Si nos interesa conocer el estado de uso de la memoria, podemos utilizar el comando **memory**.

Para observar los valores de los datos utilizamos el comando **browse**. Solo basta con escribir **browse** en el archivo **do** o en la ventana de comandos, y la base de datos se abrirá en una ventana diferente a las que estamos trabajando de la siguiente forma:

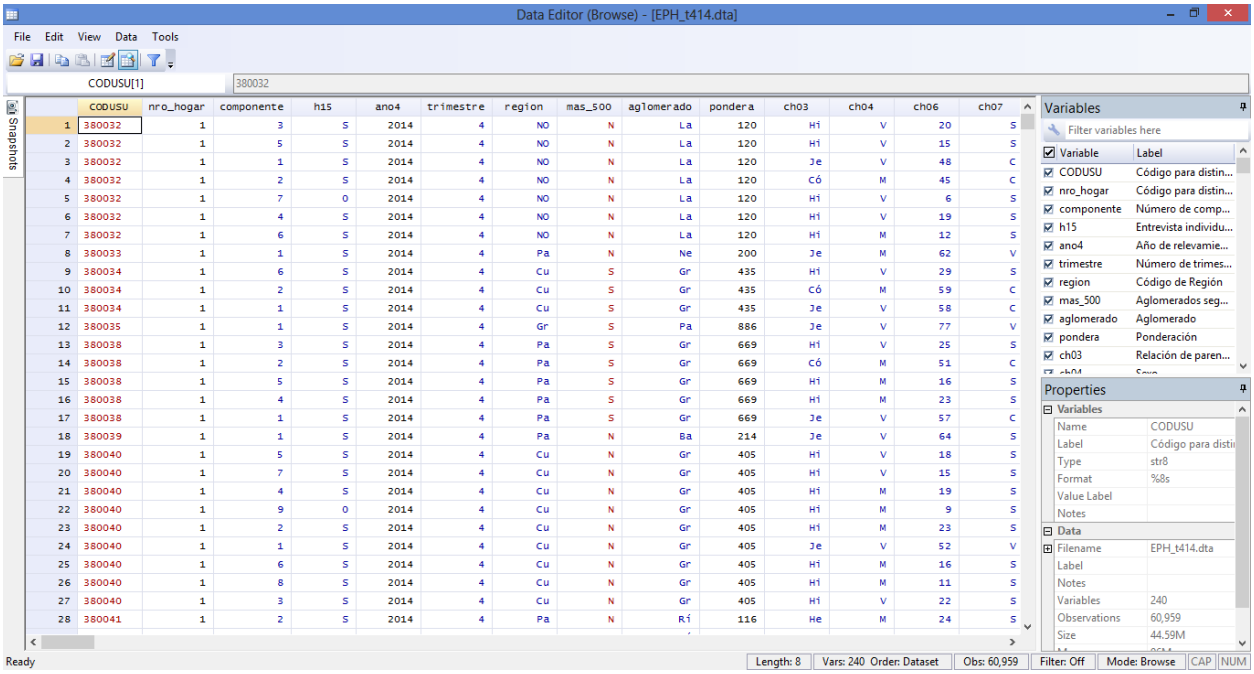

Figura 6

Observemos que los valores de las variables string están en rojo. También podemos disminuir el tamaño de la base de datos con el comando **compress** que comprime el tamaño de cada variable según el mejor tipo de formato que le corresponda en cada caso.

5.4. Generación de variables

La limpieza de bases de datos es el primer paso en un análisis empírico, la cual resulta ser un tarea muy importante ya que de esta dependerá la solidez de los resultados del trabajo. Dicha limpieza trae consigo la generación de nuevas variables o cambios de nombres de variables ya existentes en la base de datos, en todo estos casos los siguientes comandos resultan ser útiles. Para los ejemplos de esta sección trabajaremos con la base de datos de la EPH del cuarto trimestre de 2014.

El comando **generate**, que podemos acortar a **gen**, nos permite generar una variable nueva. Por ejemplo si por un lado tenemos la variable cantidad de miembros en el hogar menores de 10 años (ixmen10) y, por otro lado, una variable con la cantidad de miembros de 10 o más años (ixmayeq10) pero lo que necesitamos es una variable que sea cantidad de miembros del hogar. Hay que generar una nueva variable como la suma de dichas variables,

gen miembros=ixmayeq10 +ixmen10.

Podríamos comparar esta variable con la variable miembros total del hogar que provee la encuesta de la siguiente manera:

compare miembros ixtot.

También podemos renombrar una variable ya existente con el comando **rename**. Un comando que podemos precisar cuando se quiere generar una variable es el comando **replace**. Por ejemplo si generamos una variable de género binaria, que toma valor 1 cuando el individuo es hombre y 0 cuando es mujer, la podemos llamar hombre. En la base de datos de la EPH la variable ch04 es la variable sexo que toma valor 1 si es hombre y 2 si es mujer, entonces:

gen hombre=.

replace hombre=1 if ch04==1

replace hombre=0 if ch04==2.

Primero generamos la variable hombre y le damos el valor *missing*, es decir un punto. Stata toma este punto como infinito, con lo cual hay que tener cuidado cuando trabajamos con los operadores mayor (>) o menor (<). Luego le asignamos el valor 1 si la variable ch04 es 1 y el valor 0 si ch04 toma valor 2, para ello trabajamos con el operador if. Notar que para señalar un valor determinado al operador if debemos utilizar el doble igual (==). En algunos casos el comando **egen** es más útil que el comando **generate**.

Otro comando muy útil es **label**, este comando entre las acciones que permite puede asignar una etiqueta a una variable o al valor de la variable. Por ejemplo, asignemos una etiqueta a la variable hombre:

label var hombre "Género: valor 1 si es hombre"

si pedimos una descripción de la base con el comando **describe**, observaremos:

```
hombre          float   %9.0g      etiquetahombre
                                        Género: valor 1 si es hombre
```

Figura 7

Ahora asignemos etiquetas a los valores de la variable hombre.

label define etiquetahombre 0 "mujer" 1 "hombre"

label values hombre etiquetahombre

Si hacemos **browse** antes de asignar etiquetas nos encontramos que la variable hombre presenta solo ceros y unos:

	hombre
1	1
2	1
3	1
4	0
5	1
6	1
7	0
8	0
9	1
10	0
11	1
12	1
13	1
14	0
15	0
16	0
17	1
18	1

Figura 8

Al asignar etiquetas a los valores tenemos que la variable hombre muestra las etiquetas, pero continúa siendo una variable numérica ya que cada valor numérico está asociado a una etiqueta. Es decir, que podemos continuar utilizando la variable como cualquier variable numérica.

	hombre
1	hombre
2	hombre
3	hombre
4	mujer
5	hombre
6	hombre
7	mujer
8	mujer
9	hombre
10	mujer
11	hombre
12	hombre
13	hombre
14	mujer
15	mujer
16	mujer
17	hombre
18	hombre

Figura 9

Armar cuantiles es sencillo en Stata con los comandos **xtile** o **pctile**. El comando **pctile** genera los cuantiles y el comando **xtile** genera una variable que asigna a cada observación de la base de datos el cuantil al cual pertenece. Supongamos que queremos armar los quintiles (5 cuantiles) de la variable ingreso total familiar (itf), esto lo podemos hacer de la siguiente manera:

pctile qitf = itf, nq(5) genp(porcentaje)

xtile qxitf = itf, nq(5)

5.5. Descripción de datos

El comando **summarize** (que podemos abreviarlo como **sum**) arroja una breve descripción de la variable: la cantidad de observaciones de la variable, la media, el desvío estándar y el valor mínimo y máximo. Usualmente las bases de datos contienen una variable de ponderación (weight), en estos casos en los que queremos realizar la descripción de la muestra a partir de la muestra ponderada utilizamos [weight=ponderador]. En el caso de las EPH, la variable que contiene los ponderadores se llama *pondera*. Ahora supongamos que se desea dar una breve descripción estadística del ingreso total familiar (itf) de la encuesta EPH del cuarto trimestre de 2014,

```
. sum itf
```

Variable	Obs	Mean	Std. Dev.	Min	Max
itf	60959	12399.33	9898.248	0	330000

```
. * Describe a partir de la muestra ponderada
. sum itf [w=pondera]
(analytic weights assumed)
```

Variable	Obs	Weight	Mean	Std. Dev.	Min	Max
itf	60959	26725120	12571.07	9786.08	0	330000

Figura 10

Notar que al utilizar la muestra ponderada, tenemos las observaciones e inmediatamente al lado las observaciones ponderadas.

El comando tiene la opción “detail”, que nos arroja un análisis descriptivo más detallado.

```
. sum itf [w=pondera], detail
(analytic weights assumed)
```

Monto total de ingreso familiar percibido en ese mes

Percentiles		Smallest		
1%	960	0		
5%	2900	0		
10%	3876	0	Obs	60959
25%	6500	0	Sum of Wgt.	26725120
50%	10140		Mean	12571.07
		Largest	Std. Dev.	9786.08
75%	16000	220000		
90%	23200	330000	Variance	9.58e+07
95%	29200	330000	Skewness	4.01655
99%	46800	330000	Kurtosis	51.87721

Figura 11

Otro comando como **tabulate** nos permite hacer tablas descriptivas de la siguiente manera:

Género: valor 1 si es hombre	Freq.	Percent	Cum.
mujer	13,780,987	51.57	51.57
hombre	12,944,133	48.43	100.00
Total	26,725,120	100.00	

Figura 12

Aquí observamos que el 51,57 % son hombres y el resto mujeres. Por otro aldo, vamos a utilizar el comando **table** para armar tablas. Por ejemplo queremos armar una tabla etaria de individuos menores a 10 años. La variable ch06 contiene las edades de todos lo individuos encuestados en la EPH y le pedimos a Stata lo siguiente:

```
. table ch06 if ch06<=10
```

Edad en años cumplidos	Freq.
Menos de 1 año	914
1	980
2	945
3	933
4	961
5	969
6	1,011
7	992
8	914
9	1,002
10	966

Figura 13

Esta tabla muestra cuántos individuos tienen una determinada edad al momento de realizada la encuesta. También podemos hacer tablas cruzadas, por ejemplo analizar el promedio de edad en el caso de los varones y mujeres por separado así como otras estadísticas (nro. de observaciones, desvío estándar, mediana) de la siguiente manera:

Género: valor 1 si es hombre	N(ch06)	mean(ch06)	sd(ch06)	med(ch06)
mujer	31,778	35	22.6298	32
hombre	29,181	32	21.41873	28

Figura 14

5.6. Gráficos

Existen diversos tipos de gráficos en Stata. Aquí solo veremos algunos de ellos. A su vez cada tipo de gráfico presenta diferentes opciones para su construcción. Stata mostrará el gráfico respectivo en una ventana independiente y el mismo podrá ser exportado y guardado en la carpeta de **gráficos** en diferentes formatos jpg, png, etc.

Primero vamos a aprender cómo hacer un gráfico de dispersión con el comando **scatter**. Tenemos una base de datos de países con la esperanza de vida (lexp) y el producto bruto per cápita (gnppc) de cada país. Para hacer este gráfico le pedimos a Stata:

scatter lexp gnppc

y obtenemos:

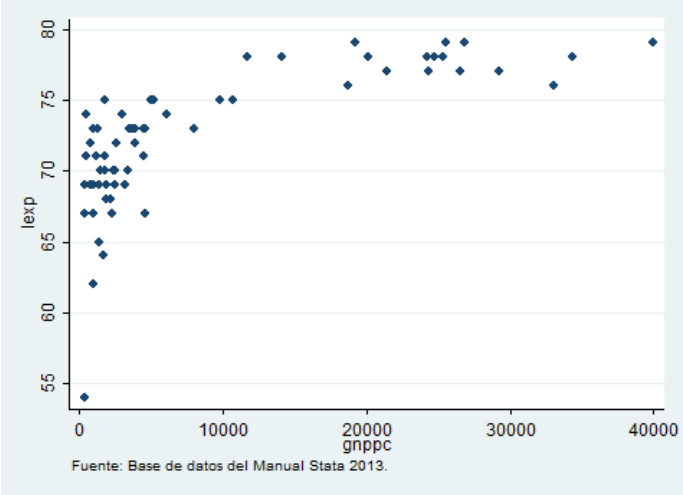

Figura 15

Si queremos cambiar la leyenda usamos la opción "legend", y si queremos cambiar el nombre de los ejes podemos usar ytitle o xtitle. Por ejemplo:

scatter lexp gnppc, ytitle("Esperanza de vida")

y obtenemos:

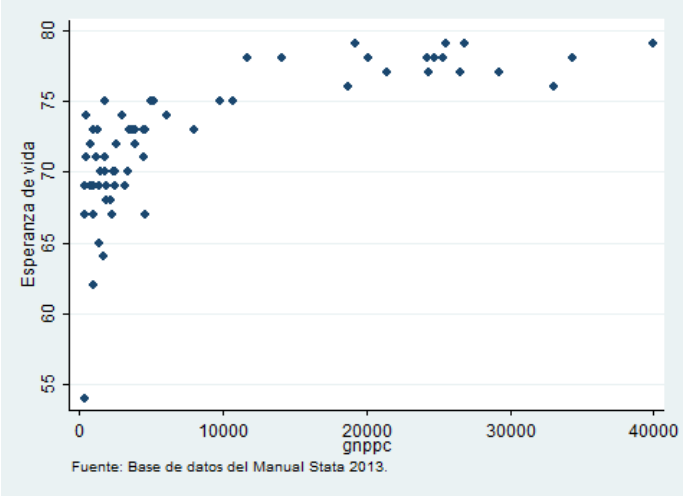

Figura 16

Ahora podríamos invertir el orden de las variables
scatter gnppc lexp , ytitle("Esperanza de vida")

de esta manera podemos obtener en el eje de abscisas la variable lexp y en el de ordenadas la variable gnppc. También podríamos querer tener varios gráficos de acuerdo a cada grupo, por ejemplo si en este caso quisiéramos dividir según la región, esto se hace con la opción "by" de la siguiente forma:

scatter lexp gnppc, by(region)

y obtenemos:

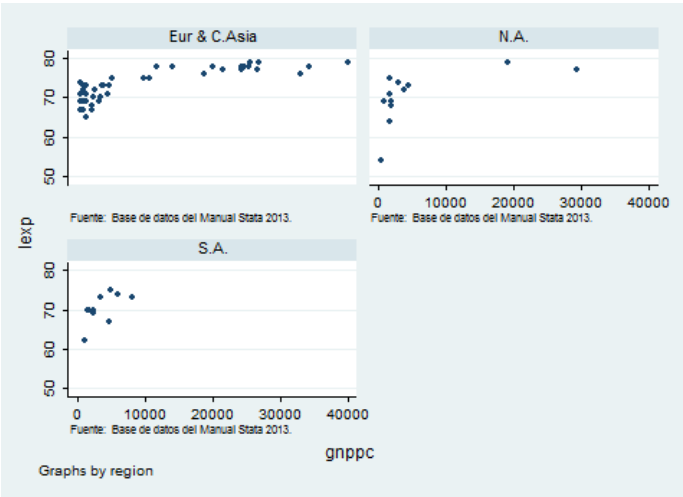

Figura 17

Podríamos también querer etiquetar cada punto del gráfico. Supongamos que solo nos interesan los países con producto bruto per cápita entre 10000 y 24000 dólares, entonces:

scatter lexp gnppc if (gnppc>10000 gnppc<24000), mlabel(country)

y obtenemos:

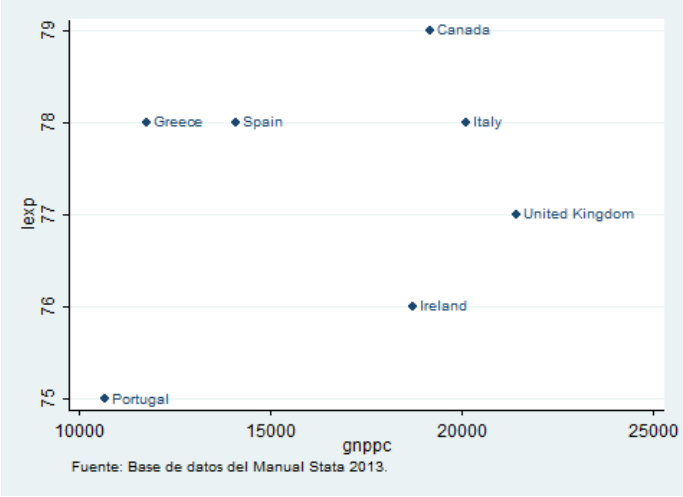

Figura 18

Para gráficos tipo scatter podemos agregar al gráfico una tendencia lineal estimada a partir del comando **lfit**. Es decir,

scatter lexp gnppc, msymbol(Oh) || **lfit** lexp gnppc

Notar que la doble línea vertical || indica que comienza un nuevo gráfico contenido en el anterior. De esta manera tenemos:

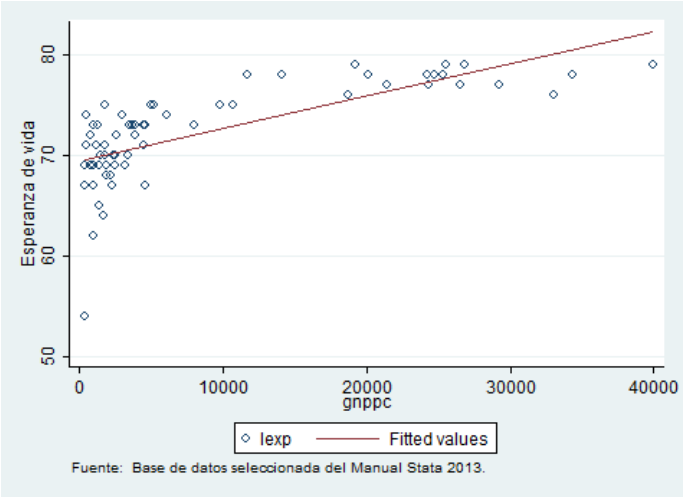

Figura 19

Podemos tener este mismo gráfico a partir del comando **twoway**, este comando lo usamos cuando queremos hacer varios gráficos en uno solo.

twoway (**scatter** lexp gnppc, msymbol(Oh)) (**lfit** lexp gnppc)

Existen otros tipos de gráficos como de área, de barras, spike, etc. Mostramos ahora cómo graficar un histograma. A partir de la base de datos de la EPH del cuarto trimestre de 2014, vamos a hacer un histograma de la variable de ingresos per cápita familiar (ipcf).

histogram ipcf

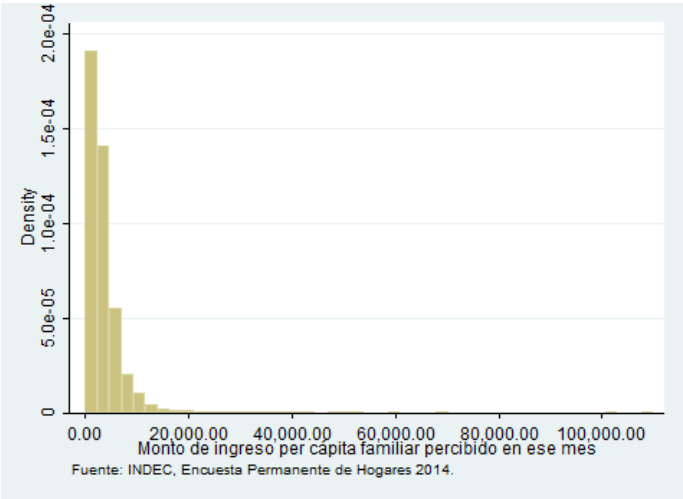

Figura 20

Si hacemos un histograma del logaritmo de dicha variable

histogram lnipcf

tenemos:

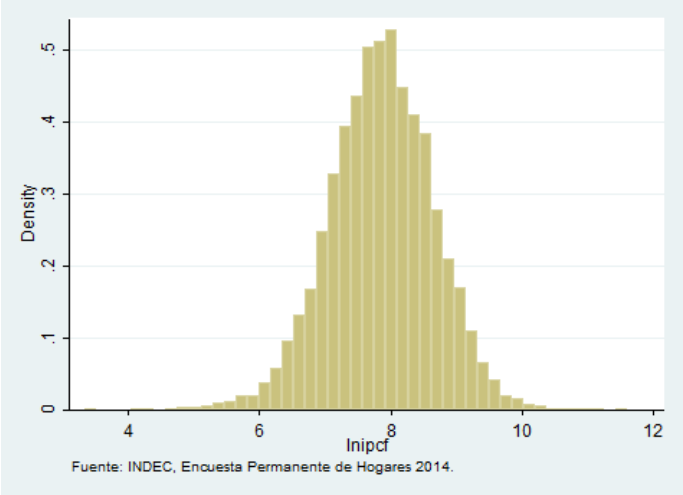

Figura 21

A través del uso de Kernels (un método de estimación no paramétrico) podemos graficar la respectiva función de densidad estimada por kernel, de esta manera:

kdensity lnipcf

genera:

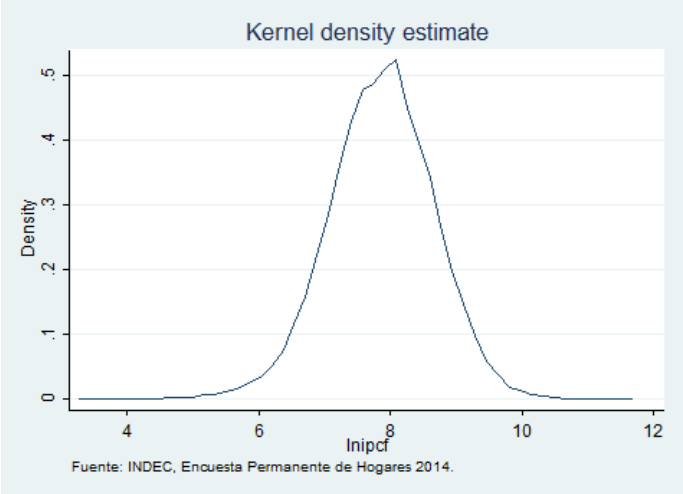

Figura 22

Por alguna razón, nos podría interesar tener ambos gráficos en uno solo. Como ya vimos con el comando **twoway** podemos hacerlo:

twoway (**histogram** lnipcf) (**kdensity** lnipcf)

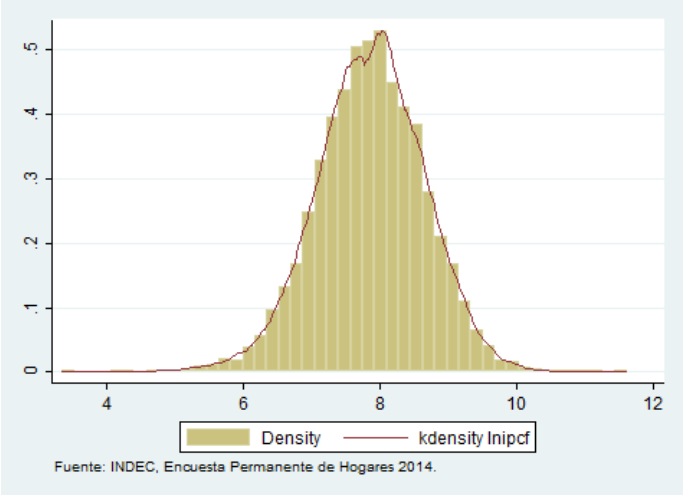

Figura 23

En todos estos casos los comandos de gráficos admiten condicionar a partir del if, ponderar la muestra con [w= ponderador] y diferentes opciones. Como se dijo anteriormente cada comando permite diferentes opciones ya sea sobre el título del gráfico, ejes, legenda, colores, formas, etc. Por ejemplo, en el siguiente gráfico vamos a graficar dos funciones de densidad del logaritmo del ingreso per cápita familiar uno para Mendoza y otro para el resto del país para ello utilizamos el operador condicional **if**. Además indicaremos el título del gráfico, la legenda, e incluiremos una nota con la fuente del gráfico.

twoway (**kdensity** lnipcf if mendoza==1) (**kdensity** lnipcf if mendoza==0), title("Ejemplo gráfico") legend(label(1 "Mendoza") label(2 "Resto del país")) note("Fuente: INDEC, Encuesta Permanente de Hogares 2014.")

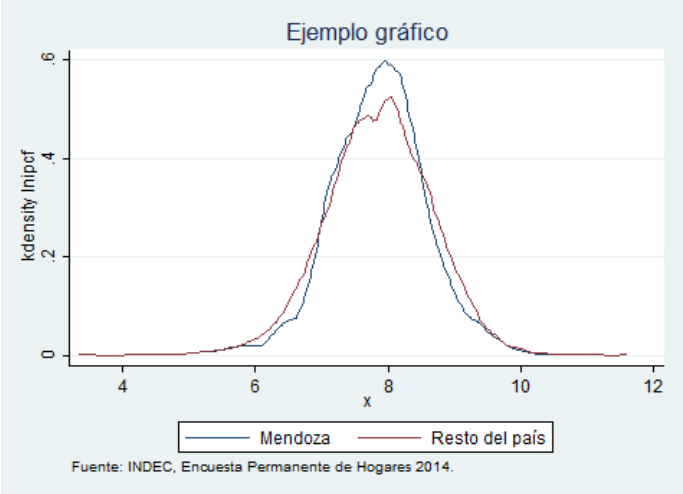

Figura 24

5.7. Regresiones

En cuanto a métodos econométricos Stata es un buen software para realizarlos. Sin embargo, en este punto cabe señalar que Stata (como cualquier otro paquete estadístico) es un software y la validez de los resultados dependerá netamente del investigador y no de Stata. Es muy importante tener en claro el problema que queremos investigar, las ventajas y limitaciones del método que utilizaremos, así como la base de datos disponible. Stata arroja determinados resultados, pero luego será trabajo del analista interpretar e identificar posibles problemas de estimación si es que los hubiera.

Para realizar una regresión por mínimos cuadrados ordinarios (MCO), utilizamos el comando **regress** o su abreviación **reg**. El mismo admite varias opciones entre las cuales está estimar errores estándar robustos con la opción vce(r). Ahora realicemos un ejemplo donde la variable dependiente es el ingreso per cápita familiar (**y**) y las variables explicativas (X_1, X_2, X_3, X_4) incluyen variables demográficas, educativas, geográficas y ocupacionales; y estimemos esta regresión por MCO:

reg y X_1 X_2 X_3 X_4 [w=pondera], vce(r)

En el caso que quisiéramos realizar una estimación por medio de Logit o Probit en Stata, usamos los comandos **logit** y **probit**. Para estimar los respectivos efectos marginales se utiliza el comando **margins**. En el caso de datos de panel tenemos el comando **xtreg** con las opciones que nos permiten estimar por efectos fijos, aleatorios y también los modelos between o within.

Podemos exportar los resultados de una regresión a una hoja de cálculo u a otro formato de una manera muy sencilla con el comando **outreg** o **outreg2**. Este comando debe ser usado inmediatamente después de realizada la estimación y uno le puede decir donde guardar dicha estimación (i.e. en qué parte del directorio de trabajo o en su defecto en otro directorio). Por ejemplo,

reg lnipcf edad hombre niveled pampeana cuyo noa patagonia nea [w=pondera], vce(r)

outreg2 using "ejemplo regresion.xls", dec(3) replace

En esta última línea le pedimos que genere un archivo de excel en la carpeta del directorio y al abrir

dicho archivo en excel nos encontramos que si bien en la pantalla de resultados de Stata tenemos:

```
. reg lnipcf edad hombre nivel_ed pampeana cuyo noa patagonia nea [w=pondera], vce(r)
(analytic weights assumed)
(sum of wgt is   2.6578e+07)

Linear regression                               Number of obs =   60706
                                                F(  8, 60697) = 1310.27
                                                Prob > F      =  0.0000
                                                R-squared     =  0.2166
                                                Root MSE      =  .70926
```

lnipcf	Coef.	Robust Std. Err.	t	P>|t|	[95% Conf.	Interval]
edad	.007394	.0001921	38.49	0.000	.0070174	.0077705
hombre	.0690459	.0088045	7.84	0.000	.0517891	.0863027
nivel_ed	.1537775	.002715	56.64	0.000	.1484561	.1590989
pampeana	-.025601	.0101022	-2.53	0.011	-.0454013	-.0058007
cuyo	-.1402423	.0125203	-11.20	0.000	-.1647821	-.1157024
noa	-.2661225	.01005	-26.48	0.000	-.2858206	-.2464244
patagonia	.3422901	.0121776	28.11	0.000	.3184219	.3661583
nea	-.3924968	.0123051	-31.90	0.000	-.4166149	-.3683788
_cons	7.172321	.0136807	524.26	0.000	7.145507	7.199135

Figura 25

en el archivo de excel tendremos:

	A	B
1		
2		(1)
3	VARIABLES	lnipcf
4		
5	edad	0.007***
6		(0.000)
7	hombre	0.069***
8		(0.009)
9	nivel_ed	0.154***
10		(0.003)
11	pampeana	-0.026**
12		(0.010)
13	cuyo	-0.140***
14		(0.013)
15	noa	-0.266***
16		(0.010)
17	patagonia	0.342***
18		(0.012)
19	nea	-0.392***
20		(0.012)
21	Constant	7.172***
22		(0.014)
23		
24	Observations	60,706
25	R-squared	0.217
26	Robust stand	
27	*** p<0.01, **	

Figura 26

El comando **predict** nos permite obtener el valor estimado a partir de la ecuación de regresión. Podemos obtener también los residuos de la regresión a partir de:

predict errores, resid

También podemos realizar pruebas de hipótesis con Stata. Algunos comandos son **ttest**, **sdtest**, **prtest**.

5.8. Comandos para series de tiempo

Como ya se mencionó usamos el comando **tsset** para definir las dimensiones de la base de datos. Cuando los datos son series de tiempo, en este caso la unidad de información es un momento en el tiempo (anual, mensual, semestral, etc.). Por ejemplo, supongamos una base de datos anual entonces:

tsset año, yearly

La periodicidad puede ser: daily (diaria), weekly (semanal), monthly (mensual), quarterly (trimestral), halfyearly (semestral), yearly (anual). Los operadores de series de tiempo son útiles para generar nuevas variables a partir de las variables de la base de datos. Estos operadores son L (lag), F (lead), D (difference), S (seasonal).

Operador	Definición
L.	un periodo hacia atrás $(x_t - 1)$
L2.	2-periodos hacia atrás $(x_t - 2)$
...	
F.	un periodo hacia delante $(x_t + 1)$
F2.	2-periodos hacia delante $(x_t + 2)$
...	
D.	diferencia $(x_t - x_t - 1)$
D2.	diferencia en diferencia $(x_t - x_t - 1) - (x_t - 1 - x_t - 2)$
...	
S.	diferencia con el primer periodo $(x_t - x_t - 1)$
S2.	diferencia con el segundo periodo $(x_t - x_t - 2)$
...	

Fuente: StataCorp. 2013. Stata 13 Help Manual.

6. Otros comandos útiles

Stata nos permite también usar macros. Las macros que utilizamos en Stata para programar son dos: local y global. Ambas macros son muy útiles en múltiples situaciones. Vamos a definir primero qué es una local porque son las más usadas. Las locales guardan información que luego se las puede "llamar" y acceder al contenido de la macro. Las locales se generan a partir del comando **local**. Cada local tiene un nombre y un valor.

Por ejemplo: queremos guardar el valor de la media de la variable *edad*. Entonces con el comando **summarize** obtenemos la media, Stata guarda los resultados de cada comando por un lapso de

tiempo breve hasta que se vuelva a correr un nuevo comando. El comando **summarize** guarda los resultados en una extensión **r**, podemos observar dichos resultados con la instrucción **return list**. Para guardar este valor generamos una local de la siguiente manera

local mediaedad=r(mean)

Luego, podemos llamar la local por el nombre que le hemos asignado, en este caso *mediaedad*, para trabajar con el valor de la media. Para llamar la local se utilizan la tilde y la comilla individual (ver en el ejemplo 5 del apéndice el uso de las comillas).

Es preciso señalar, que las variables que aparecen en las columnas de la base de datos toman distintos valores para cada unidad de observación. Sin embargo, un parámetro toma un único valor y es el mismo para todas las unidades observacionales. La local es útil para guardar dicho parámetro y no tener que generar una variable demandando más memoria que la necesaria. Un caso en que podemos usar las locales puede ser cuando queremos guardar los parámetros de un modelo.

```
. sum edad [w=pondera]
(analytic weights assumed)

    Variable |       Obs      Weight        Mean   Std. Dev.       Min        Max
-------------+--------------------------------------------------------------------
        edad |     60959    26725120    32.85311    22.00098         0         98

.
. * Vemos los resultados guardados en el breve lapso hasta no correr otro comando
. return list

scalars:
                  r(N) =  60959
              r(sum_w) =  26725120
               r(mean) =  32.85310879801475
                r(Var) =  484.0429336275271
                 r(sd) =  22.00097574262394
                r(min) =  0
                r(max) =  98
                r(sum) =  878003275

.
. * Guardo la media de la edad con una local y la llamo mediaedad
. local mediaedad=r(mean)

.
. * Le pido a Stata que me muestre la local mediaedad con el comando display
. display `mediaedad'
32.853109
```

Figura 27

Por otro lado, también podemos usar las globales de la misma forma que las locales. El contenido de las macros globales se lo indica con el comando **global**. Las macros global son utilizadas porque las mismas se mantienen en un programa o una sesión de STATA; por lo que es útil usarlas en programación de comandos. Otra diferencia con respecto a las macros "local" es la forma en cómo se referencian. Para las locales debemos utilizar las comillas mientras que para llamar las globales tenemos que utilizar el signo pesos $. Por ejemplo, armamos una global llamada V que contenga los nombres de tres variables edad, hombre, GBA. Podemos luego llamarlas juntas escribiendo $V.

```
. global V "edad hombre GBA"

.
. /* Hago una descripción de las variables edad, hombre y GBA, pero como tenemos
>  la global podemos directamente llamarla para ejecutar el comando */
. sum $V
```

Variable	Obs	Mean	Std. Dev.	Min	Max
edad	60959	33.25499	22.07877	0	98
hombre	60959	.4786988	.4995502	0	1
GBA	60959	.1750193	.3799867	0	1

Figura 28

Otros procesos útiles son los "bucles" o estructuras de procedimiento repetitivo. En particular, dos comandos muy utilizados son: **foreach** y **forvalues**. Cada uno de ellos nos permite realizar la misma tarea una y otra vez.

El comando **forvalues** nos permite trabajar con valores consecutivos, para ello el comando utiliza una local. La **local** toma el valor inicial indicado y luego lo reemplaza por los valores consecutivos elegidos. Este comando ejecuta todos los comandos que se encuentran entre las llaves para el valor inicial y los repite para los siguientes valores que toma la local. Por ejemplo: genero un bucle donde le pedimos que ejecute un **summarize** de la variable edad para cada una de las regiones (la variable región toma 6 valores desde 1 hasta 6 y cada valor representa una región). En este caso tenemos lo siguiente:

```
.  forvalues r=1/6{
  2.  sum edad [w=pondera] if region==`r'
  3.  }
```

Variable	Obs	Weight	Mean	Std. Dev.	Min	Max
edad	10669	13778196	33.06427	22.14815	0	98

Variable	Obs	Weight	Mean	Std. Dev.	Min	Max
edad	17517	6137750	34.0362	22.33971	0	98

Variable	Obs	Weight	Mean	Std. Dev.	Min	Max
edad	5697	1798056	32.7966	22.15096	0	98

Variable	Obs	Weight	Mean	Std. Dev.	Min	Max
edad	12997	2628696	30.96724	21.19942	0	98

Variable	Obs	Weight	Mean	Std. Dev.	Min	Max
edad	8055	990110	30.90616	20.5775	0	98

Variable	Obs	Weight	Mean	Std. Dev.	Min	Max
edad	6024	1392312	30.5661	20.80661	0	96

Figura 29

Por su parte, el comando **foreach** repite un conjunto de macros local a cada elemento de la lista indicada y ejecuta los comandos que se colocan entre llaves. A continuación mostramos algunos ejemplos,

```
. foreach num of numlist 1 4/8 13(2)21 103 {
  2.  display `num'
  3.  }
1
4
5
6
7
8
13
15
17
19
21
103
```

Figura 30

```
.  foreach v of global V{
  2.  sum `v'
  3.  }
```

Variable	Obs	Mean	Std. Dev.	Min	Max
edad	60959	33.25499	22.07877	0	98

Variable	Obs	Mean	Std. Dev.	Min	Max
hombre	60959	.4786988	.4995502	0	1

Variable	Obs	Mean	Std. Dev.	Min	Max
GBA	60959	.1750193	.3799867	0	1

Figura 31

7. Generación de números aleatorios

Hay varias formas de generar números aleatorios. En el caso de no tener una base de datos y quisiéramos armar una a partir de las variables que se van a generar, debemos primero especificar a Stata con cuántas observaciones vamos a trabajar. Esto se hace con el comando

set obs nro. obs.

Por otro lado, si deseamos replicar la simulación a partir de los mismos datos iniciales debemos especificar una semilla. De esta manera Stata cada vez que genere las variables va a comenzar de la misma manera. El comando que especifica la semilla se llama **set seed**.

Se pueden generar variables a partir de distribuciones uniforme, normal, normal estándar, entre otras. Algunos ejemplos son:

```
set obs 200
set seed 4
generate x1 = uniform()*10
generate x2 = uniform()*25
generate u=rnormal(0,1)
generate y = 15 + 2*x1 + 1/3*x2 + u
generate u=rnormal(2,5)
```

Otro ejemplo:

```
clear
set obs 100000
set more off
gen clase = 0
local d = 0
while 'd'<0.15 {
replace clase = uniform() if clase<0.15
sum clase
local d = r(min)
display " d = 'd' "
}
```

El lector podrá observar que hasta que d no tome el valor 0.15 los valores menores a 0.15 de la variable clase serán reemplazados por los valores de una distribución uniforme.

Por último, cabe señalar que la ayuda (es posible acceder a la misma con el comando **help**) así como el manual de Stata es muy completa y contiene diversos ejemplos que son de apoyo a la hora de utilizar un comando particular. Recomendamos al usuario que los utilice y sea la primer fuente de consulta a la hora de trabajar con Stata. Este documento solo intenta ser una primera guía de herramientas para el uso de Stata.

Apéndice

```
/*****************************************************************
            Ejemplo Guía uso de Stata


Autor: Monserrat Serio
Mails: monserrat.serio@fce.uncu.edu.ar
Actualizado:03/06/2015
*****************************************************************/

clear all
*set more off

*  Computadora personal
*  Directorio de trabajo:
cd "C:\Users\Monserrat\Monsi\Trabajo\Jefe de Trabajos Prácticos FCE Politicas Sociales\Curso STATA\Guia Stata Apuntes 2015\Ejemplo"


* Creo el archivo log
local log_file="log\Ejemplo1.log"
capture log close
log using "`log_file'", replace

* Abre base de datos original

* Abrir base de datos desde un archivo delimitado por comas
insheet using "bases de datos\lifeexp.csv", clear
des

* Abrir base de datos desde un archivo de excel
import excel "bases de datos\lifeexpect.xlsx", firstrow clear


* Abrir base de datos de Encuesta Permanente de Hogares en formato .dta
use "bases de datos\Individual_t414.dta", clear



* Unir bases de datos Encuesta Permanente de Hogares
use "bases de datos\Hogar_t414.dta", clear
sort CODUSU
save  "bases de datos\Hogar_t414.dta", replace

use "bases de datos\Individual_t414.dta", clear
sort CODUSU

merge m:m CODUSU using "bases de datos\Hogar_t414.dta"
tab _merge

*Guardo la base de datos unida
save "bases de datos\EPH_t414.dta", replace




log close
exit

```

```
/*****************************************************************
            Ejemplo Guía uso de Stata

            Base de datos: EPH 2014

Autor: Monserrat Serio
Mails: monserrat.serio@fce.uncu.edu.ar
Actualizado:03/06/2015
*****************************************************************/

clear all
set more off

*  Computadora personal
*  Directorio de trabajo:
cd "C:\Users\Monserrat\Monsi\Trabajo\Jefe de Trabajos Prácticos FCE Politicas Sociales\Curso STATA\Guia Stata Apuntes 2015\Ejemplo"


* Creo el archivo log
local log_file="log\Ejemplo2.log"
capture log close
log using "`log_file'", replace

* Abre base de datos original

* Abrir base de datos de Encuesta Permanente de Hogares en formato .dta
use "bases de datos\EPH_t414.dta", replace


*-------------- DESCRIPCIÓN DE BASE DE DATOS -------------*

* Describe la base de datos
describe

* Muestra la base de datos
browse


*---------------- GENERACIÓN DE VARIABLES ---------------*

/* Genero una variable de cantidad de miembros en el hogar que es la suma de
 las variables ix_mayeq10 (cantidad de miembros de 10 años o más) y ix_men10
 (cantidad de miembros menores a 10 años) */
gen miembros=ix_mayeq10 +ix_men10

*Comparo con la variable de cantidad de miembros total que provee la encuesta
compare miembros ix_tot

* Cambio el nombre de la variable ix_tot por miembros_tot
rename ix_tot miembros_tot

/* Genero la variable hombre que es una variable binaria que toma valor 1 si es
hombre y valor 0 si es mujer */
gen hombre=.
replace hombre=1 if ch04==1
replace hombre=0 if ch04==2

* Genero una etiqueta a la variable hombre
label var hombre "Género: valor 1 si es hombre"

* Genero una etiqueta a los valores de la variable hombre
label define etiquetahombre 0 "mujer" 1 "hombre"

label values hombre etiquetahombre

*Armo los quintiles de la variable ingreso total familiar
pctile qitf = itf, nq(5) genp(porcentaje)
xtile qxitf = itf, nq(5)


*---------------- DESCRIPCIÓN DE DATOS ------------------*

* Describir la variable ingreso total familiar (itf)
* Describe a partir de la muestra
```

```
75   sum itf
76   * Describe a partir de la muestra ponderada
77   sum itf [w=pondera]
78
79   * Describe en forma más detallada a partir de la muestra ponderada
80   sum itf [w=pondera], detail
81
82   * Describe a partir de la muestra ponderada de los hombres
83   sum itf [w=pondera] if hombre==1
84
85   * Tabla de frecuencia de hombre y mujer
86   tabulate hombre [w=pondera]
87
88   * Tabla de individuos por edad menores a 10 años
89   table ch06 if ch06<=10
90
91   * Tabla cruzada de género y edad
92   table hombre, c(n ch06 mean ch06 sd ch06 median ch06)
93
94
95   log close
96   exit
97
```

```
/*****************************************************************
            Ejemplo Guía uso de Stata


Autor: Monserrat Serio
Mails: monserrat.serio@fce.uncu.edu.ar
Actualizado:03/06/2015
*****************************************************************/

clear all
set more off

*  Computadora personal
*  Directorio de trabajo:
cd "C:\Users\Monserrat\Monsi\Trabajo\Jefe de Trabajos Prácticos FCE Politicas Sociales\Curso STATA\Guia Stata Apuntes 2015\Ejemplo"


* Creo el archivo log
local log_file="log\Ejemplo3.log"
capture log close
log using "`log_file'", replace



*----------------------- GRÁFICOS -----------------------*

* Abre base de datos
insheet using "bases de datos\lifeexp.csv", clear

* Distintos gráficos de dispersión (scatter)
scatter lexp gnppc

scatter lexp gnppc,  ytitle("Esperanza de vida")

scatter lexp gnppc, by(region)

scatter lexp gnppc if (gnppc>10000 & gnppc<24000), mlabel(country)

scatter lexp gnppc, msymbol(Oh) || lfit lexp gnppc, ytitle("Esperanza de vida")


* Dos gráficos en uno: un scatter y uno líneal (estimado)
twoway (scatter lexp gnppc, msymbol(Oh)) (lfit lexp gnppc)

* Otro tipo de gráficos: area, barras, spike, otros. Ej.
twoway spike safewater lexp


* Para agregar la fuente usar note
* scatter lexp gnppc,  note("Fuente: Base de datos del Manual Stata 2013.")



* Abre base de datos de Encuesta Permanente de Hogares en formato .dta
use "bases de datos\EPH_t414.dta", replace

* Genero la variable logaritmo del ingreso per cápita familiar
gen lnipcf=.
replace lnipcf=log(ipcf)

* Genero la variable Aglomerado de Mendoza
gen mendoza=.
replace mendoza=1 if aglomerado==10
replace mendoza=0 if aglomerado!=10

* Gráfico de un histograma
histogram ipcf

* Gráfico de un histograma
histogram lnipcf

* Gráfico de una función de densidad a través de kernel
kdensity lnipcf

```

```
75   twoway  (histogram lnipcf) (kdensity lnipcf)
76
77   * Dos funciones de densidad en un solo gráfico
78   twoway (kdensity lnipcf if mendoza==1) (kdensity lnipcf if mendoza==0), title("Ejemplo
     gráfico") legend(label(1 "Mendoza") label(2 "Resto del país")) note("Fuente: INDEC,
     Encuesta Permanente de Hogares 2014.")
79
80   * Agrego fuente con note
81   * twoway (kdensity lnipcf if mendoza==1) (kdensity lnipcf if mendoza==0), title("Ejemplo
     gráfico") legend(label(1 "Mendoza") label(2 "Resto del país")) note("Fuente: INDEC,
     Encuesta Permanente de Hogares 2014.")
82
83
84   log close
85   exit
86
```

```
/*****************************************************************
            Ejemplo Guía uso de Stata

            Base de datos: EPH 2014

Autor: Monserrat Serio
Mails: monserrat.serio@fce.uncu.edu.ar
Actualizado:03/06/2015
*****************************************************************/

clear all
set more off

*  Computadora personal
*  Directorio de trabajo:
cd "C:\Users\Monserrat\Monsi\Trabajo\Jefe de Trabajos Prácticos FCE Politicas Sociales\Curso STATA\Guia Stata Apuntes 2015\Ejemplo"


* Creo el archivo log
local log_file="log\Ejemplo4.log"
capture log close
log using "`log_file'", replace

* Abre base de datos original

* Abrir base de datos de Encuesta Permanente de Hogares en formato .dta
use "bases de datos\EPH_t414.dta", replace


*---------------- GENERACIÓN DE VARIABLES ---------------*

* Genero la variable hombre que es una variable binaria que toma valor 1 si
* es hombre y valor 0 si es mujer
/*  CH04  N(1):   sexo
            1= varón
            2= mujer  */

gen hombre=.
replace hombre=0 if ch04==2
replace hombre=1 if ch04==1
label var hombre "=1 si es hombre"

* Genero la variable edad
* Edad

/* CH06 N(2): ¿cuántos años cumplidos tiene? */

rename ch06 edad
replace edad=0 if edad==-1
replace edad=. if edad==99
label var edad "edad"


* Genero variable estado civil

/* CH07  N(1):   ¿Actualmente está...
            1=...unido?
            2=...casado?
            3=...separado/a ó divorciado/a?
            4=...viudo/a?
            5=...soltero/a?   */
gen soltero=.
replace soltero=1 if ch07==5
replace soltero=0 if (ch07==1 | ch07==2 | ch07==3 | ch07==4)
label var soltero "=1 si es soltero"



* Genero variables educativas
/* NIVEL-ED N(2): Nivel Educativo
            1= Primaria Incompleta
            2= Primaria Completa
            3= Secundaria Incompleta
            4= Secundaria Completa
```

```
75              5= Superior Universitaria Incompleta
76              6= Superior Universitaria Completa
77              7= Sin instrucción
78              9= Ns/ Nr        */
79
80  replace nivel_ed=. if nivel_ed==9
81  replace nivel_ed=1 if nivel_ed==7
82
83  * Genero variables regionales
84
85  /* REGION N(2):     1= Gran Buenos Aires
86                       40= NOA
87                       41= NEA
88                       42= Cuyo
89                       43= Pampeada
90                       44= Patagonia   */
91
92  /* Voy a generar una variable llamada region que toma valores de 1 a 6 donde
93                       1=GBA,
94                       2=Pampeana,
95                       3=Cuyo,
96                       4=NOA,
97                       5=Patagonia,
98                       6=NEA*/
99
100 replace region=2 if region==43
101 replace region=3 if region==42
102 replace region=4 if region==40
103 replace region=5 if region==44
104 replace region=6 if region==41
105 label var region "regiones de Argentina"
106
107 * Armo variables binarias para cada región
108
109 tabulate region, gen(regionindividual)
110
111 rename regionindividual1 GBA
112 rename regionindividual2 pampeana
113 rename regionindividual3 cuyo
114 rename regionindividual4 noa
115 rename regionindividual5 patagonia
116 rename regionindividual6 nea
117
118 * Genero la variable logaritmo del ingreso per cápita familiar (ipcf)
119 gen lnipcf=.
120 replace lnipcf=log(ipcf)
121
122
123
124
125 *---------------------- REGRESIONES ----------------------*
126 * Regresión por MCO (mínimos cuadrados ordinarios)
127 reg lnipcf edad hombre nivel_ed pampeana cuyo noa patagonia nea [w=pondera]
128
129 * Regresión con errores estándar robustos
130 reg lnipcf edad hombre nivel_ed pampeana cuyo noa patagonia nea [w=pondera], vce(r)
131
132 * Exporto los resultados a un archivo excel
133 outreg2 using "ejemplo regresion.xls", dec(3) replace
134
135
136 log close
137 exit
138
```

```
/*****************************************************************
            Ejemplo Guía uso de Stata

            Base de datos: EPH 2014

Autor: Monserrat Serio
Mails: monserrat.serio@fce.uncu.edu.ar
Actualizado:03/06/2015
*****************************************************************/

clear all
set more off

*  Computadora personal
*  Directorio de trabajo:
cd "C:\Users\Monserrat\Monsi\Trabajo\Jefe de Trabajos Prácticos FCE Politicas Sociales\Curso STATA\Guia Stata Apuntes 2015\Ejemplo"


* Creo el archivo log
local log_file="log\Ejemplo5.log"
capture log close
log using "`log_file'", replace

* Abre base de datos original

* Abrir base de datos de Encuesta Permanente de Hogares en formato .dta
use "bases de datos\EPH_t414.dta", replace


*---------------- GENERACIÓN DE VARIABLES ---------------*

* Genero la variable edad
* Edad

/* CH06 N(2): ¿cuántos años cumplidos tiene? */

rename ch06 edad
replace edad=0 if edad==-1
replace edad=. if edad==99
label var edad "edad"

* Genero variables regionales

/* REGION N(2):     1= Gran Buenos Aires
                    40= NOA
                    41= NEA
                    42= Cuyo
                    43= Pampeada
                    44= Patagonia   */

/* Voy a generar una variable llamada region que toma valores de 1 a 6 donde
                    1=GBA,
                    2=Pampeana,
                    3=Cuyo,
                    4=NOA,
                    5=Patagonia,
                    6=NEA*/

replace region=2 if region==43
replace region=3 if region==42
replace region=4 if region==40
replace region=5 if region==44
replace region=6 if region==41
label var region "regiones de Argentina"



*--------------- OTROS COMANDOS ÚTILES ------------------*


******************** LOCAL ******************

* Pido calcular la media con el comando sum
sum edad [w=pondera]
```

```
75
76  * Vemos los resultados guardados en el breve lapso hasta no correr otro comando
77  return list
78
79  * Guardo la media de la edad con una local y la llamo mediaedad
80  local mediaedad=r(mean)
81
82  * Le pido a Stata que me muestre la local mediaedad con el comando display
83  display `mediaedad'
84
85  * Tengo un modelo con un parámetro a que toma el valor 2, genero dicha macro
86  local a=2
87
88  * Luego genero una variable que sea el doble de la edad de los individuos
89  generate edad2=`a'*edad
90
91  ******************** GLOBAL ******************
92  * Genero la global
93  global V "edad hombre GBA"
94
95  /* Hago una descripción de las variables edad, hombre y GBA, pero como tenemos
96   la global podemos directamente llamarla para ejecutar el comando */
97  sum $V
98
99  * Genero una global de un escalar
100 global e "4"
101
102 * Le pido a Stata que me muestre la global
103 display $e
104
105
106 ******************** FORVALUES **********************
107
108 /* Genero el bucle: donde le pido que desde el valor 1 hasta el valor 5 ejecute
109  el comando display, y me muestre el  valor que toma la local i en cada caso */
110 forvalues i=1/5{
111 display `i'
112 }
113
114 /* Genero un bucle: donde le pido que ejecute un summarize de la edad para
115 cada una de las regiones (la variable region toma valor 1 hasta 6 y cada valor
116 representa una región) */
117
118  forvalues r=1/6{
119  sum edad [w=pondera] if region==`r'
120  }
121
122
123  *************** FOREACH ************************
124
125 foreach x in 1/10{
126 display `x'
127  }
128
129 foreach num of numlist 1 4/8 13(2)21 103 {
130  display `num'
131  }
132
133
134  foreach v of global V{
135  sum `v'
136  }
137
138
139 log close
140 exit
141
```

```
/*****************************************************************
            Ejemplo Guía uso de Stata


Autor: Monserrat Serio
Mails: monserrat.serio@fce.uncu.edu.ar
Actualizado:03/06/2015
*****************************************************************/

clear all


*-------------- GENERACIÓN DE NÚMEROS ALEATORIOS ---------------*

* Indico cantidad de observaciones
set obs 200


* Indico la semilla con el número 4
set seed 4

* Genero dos variables uniforme
generate x1 = uniform()*10
generate x2 = uniform()*25

* Genero una variable normal
generate u=rnormal(0,1)

* O una normal con media 2 y desvío estándar 5
generate z=rnormal(2, 5)

* Genero una variable a partir de una ecuación específica
generate y = 15 + 2*x1 + 1/3*x2 + u


* Otro ejemplo es:

clear
set obs 100000
set more off
gen clase = 0
local d = 0
while `d'< 0.15 {
replace clase = uniform() if clase<0.15
summ clase
local d = r(min)
display "d = `d'"
}

```