

# Documentos de Políticas Públicas

**ECONOMETRÍA PARA LA EVALUACIÓN DE POLÍTICAS PÚBLICAS CON  
STATA: INTRODUCCIÓN Y ANÁLISIS DE DATOS**

Carlos Giovanni González Espitia

POLICY PAPER 2011 - 001

**POLIS**

Observatorio  
de Políticas  
Públicas



**DOCUMENTOS DE POLÍTICAS PÚBLICAS  
POLIS**

ISSN: 2011 -5903

2011 - 001 Cali, Enero de 2011.

Comité Editorial

Blanca Cecilia Zuluaga  
John James Mora  
Julio Cesar Alonso  
Luciana Manfredi  
Vladimir Rouvinski  
Silvana Godoy Mateus  
Ximena Dueñas Herrera

Observatorio de Políticas Públicas – POLIS  
Teléfono: (2) 555 23 34 Ext. 8400 Fax: (2) 555 17 06  
Calle 18 N° 122 – 135 Cali –Colombia  
Correo electrónico: [polis@icesi.edu.co](mailto:polis@icesi.edu.co)  
[www.icesi.edu.co/polis](http://www.icesi.edu.co/polis)

# ECONOMETRÍA PARA LA EVALUACIÓN DE POLÍTICAS PÚBLICAS CON STATA: INTRODUCCIÓN Y ANÁLISIS DE DATOS<sup>1</sup>

**Carlos Giovanni González Espitia**

[cggonzalez@icesi.edu.co](mailto:cggonzalez@icesi.edu.co)

Profesor Tiempo Completo

Departamento de Economía

Universidad Icesi

Cali - Colombia

## Resumen

En econometría es necesario el uso continuo de *software* especializado tanto en la docencia como en la investigación aplicada. El objetivo de este documento es introducir al lector en el manejo de *Stata*, posiblemente el *software* econométrico más popular y con las herramientas predefinidas más adecuadas de cálculo automatizado para la docencia y la investigación en economía. Este escrito está dirigido a todos los estudiantes (en especial a los del curso de Econometría II de la Universidad Icesi), profesores e investigadores en economía deseosos de empezar a usar el programa o profundizar sus conocimientos en esta herramienta. El documento se basa en la versión de *Stata* 10.

**Palabras clave:** Econometría, *software* econométrico, *Stata*.

**Clasificación JEL:** C01, C87.

---

<sup>1</sup> Las opiniones expresadas en este documento comprometen únicamente a su autor y no a las entidades involucradas. El autor agradece a los estudiantes del curso de Econometría II (promoción 2009-2) de la Universidad Icesi, por comentar una versión preliminar.

## Contenido

1. Introducción .....	3
1.1 Introducción “ <i>Stattransfer</i> ” .....	5
2. Introducción a <i>Stata</i> y comandos básicos.....	7
2.1 Algunas modificaciones (preferencias) de la ventana inicial.....	10
2.2 Ayudas de <i>Stata</i> .....	11
2.3 Modificar la memoria.....	11
2.4 Estructura de los comandos.....	12
2.5 Comando para usar bases de datos que vienen con <i>Stata</i> .....	15
3. Procesamiento de datos .....	16
3.1 Abrir, guardar y salir .....	16
3.2 Describir, listar e inspeccionar .....	19
3.3 Seleccionar y eliminar variables .....	21
3.4 Cambiar nombres de las variables y hacer etiquetas .....	22
3.5 Creación y modificación de variables: .....	23
3.6 Combinación de bases de datos.....	27
3.7 Archivos log, do y ado .....	29
4. Estadísticas descriptivas .....	33
5. Gráficos .....	37
6. Referencias .....	43

## 1. Introducción

Este documento es el primero de una serie de documentos de carácter académico sobre el uso en econometría del *software Stata*. Este programa no es libre, y tampoco es gratuito por lo que es necesario acceder a una licencia para su uso legal. La empresa que desarrolla y comercializa el *software* es *StataCorp* ([www.stata.com](http://www.stata.com)). El programa supera prácticamente todos los *test* de fiabilidad ([www.stata.com/support/cert/](http://www.stata.com/support/cert/)), por esta razón, su rigurosidad y el manejo de dos ambientes de trabajo, es uno de los programas econométricos más utilizados.

El primero, es el tradicional ambiente de trabajo por ventanas (*Windows*), que parte del menú principal y de la barra de herramientas donde se despliegan todas las opciones posibles que tiene *Stata* predefinidas. El segundo ambiente, es trabajar con comandos predefinidos, que se complementa con un potente lenguaje de programación, esta opción permite utilizar rutinas para ejecutar programas previamente hechos sin necesidad de tener que empezar de nuevo. Posiblemente, ésta es una de las ventajas más reconocidas del por qué usar *Stata* y no otro *software*.

El documento está dirigido a todos los estudiantes, profesores e investigadores en economía deseosos de empezar a usar el programa, o profundizar sus conocimientos en esta herramienta. En especial, a los alumnos del curso de Econometría II de la Universidad Icesi, quienes realizan la parte práctica del curso en *Stata*.

Este escrito es una herramienta de apoyo y no sustituye los manuales de *Stata*. Lo que se presenta es una breve introducción dejando la parte avanzada para los siguientes documentos de la serie, donde se profundizará no sólo en temas específicos sino en el uso avanzado de los comandos de este primer documento. Los siguientes documentos de la serie tratarán los temas: (i) Análisis de regresión lineal múltiple, (ii) Análisis microeconómico, (iii) Análisis de series de tiempo y (iv) Econometría de evaluación de impacto de políticas públicas y programas.

Este primer documento no pretende enseñar *Stata* como una caja negra. Por el contrario, intenta acercar a los interesados en la econometría a uno de los pasos fundamentales de todo análisis econométrico, es decir, el procesamiento y comprensión de los datos con los que se está trabajando. Es importante recordar que la econometría sigue una metodología ampliamente aceptada por los econométricos de todo el mundo, y se puede resumir en los siguientes pasos:

1. **Especificación** del modelo econométrico
2. **Estimación** del modelo econométrico
3. **Contrastes**
4. **Proyecciones**

Antes de realizar la estimación de cualquier modelo, el econométrico se enfrenta a descargar los datos a un *software* en el cual pueda trabajar. De ahí, que la primera parte del documento gire alrededor del *stattransfer* y de cómo abrir y cargar datos en *Stata*. Posteriormente una vez se han cargado los datos, el paso siguiente es el procesamiento de los mismos. Este paso es de vital importancia, ya que, por lo general, los investigadores no cuentan con la suerte de tener procesada la información (bases de datos) a la que se puede acceder. El siguiente paso es obtener algunas estadísticas descriptivas de la base de datos, para analizar fácilmente el comportamiento de las variables que se usaran en las regresiones. Finalmente, es vital poder hacer gráficos que permiten, en muchos casos, hacer un análisis intuitivo del comportamiento de las variables y de los datos.

En suma, el documento se estructura en seis secciones. La primera de ellas es esta introducción donde se presenta la motivación para escribir el documento y un preámbulo a *Stata* y *Stattransfer*, este último es uno de los complementos de mayor utilidad para la conversión de datos en la investigación económica y del mismo *Stata*. La segunda sección, hace una breve introducción a *Stata* haciendo énfasis en la interface y cómo se debe manejar el programa al inicio de cada sesión de trabajo. La tercera, introduce los comandos básicos que trae el programa. La cuarta, introduce al lector en el procesamiento de datos, incluyendo los temas de *merge* y *matching*. En la sesión quinta, se trabajan algunos de los

comandos que permiten hacer estadísticas descriptivas. El documento termina con una sección de referencias utilizadas para la realización del mismo.

Finalmente, *Stata* no sería lo mismo si no contáramos con *Stattransfer*. Por lo tanto, el documento presenta un breve comentario antes de pasar a la primera sección sobre el uso de *Stata*.

### 1.1 Introducción “*Stattransfer*”

Un excelente complemento para los usuarios de *Stata* es el software *Stattransfer* ([www.stattransfer.com](http://www.stattransfer.com)). Este programa es de gran utilidad para la conversión de bases de datos de un formato a otro. Por ejemplo, si se tiene acceso a unos datos en SPSS, Epi-info, SAS, Excel, o datos en otro formato; con *Stattransfer* se pueden convertir a archivos de *Stata* con mucha facilidad. El programa no sólo permite pasar a *Stata*, sino que convierte los datos que se tengan en cualquier otro formato, si se desea. Es importante aclarar que *Stattransfer* no tiene todos los formatos en los que se puede trabajar una base de datos, pero si tiene una amplia gama de programas estadísticos y econométricos que son convencionalmente usados en la docencia y la investigación económica.

El cuadro 1 presenta la salida inicial de *Stattransfer* 9. A continuación se hace una breve descripción de cómo usar este programa para convertir bases de datos a diferentes extensiones. El programa tiene un menú de opciones que no es necesario saber usar para convertir una base de datos de una extensión o formato a otra. Este menú no se trabajará ya que son opciones para expertos y la idea es sólo introducir al lector en el uso de este programa.

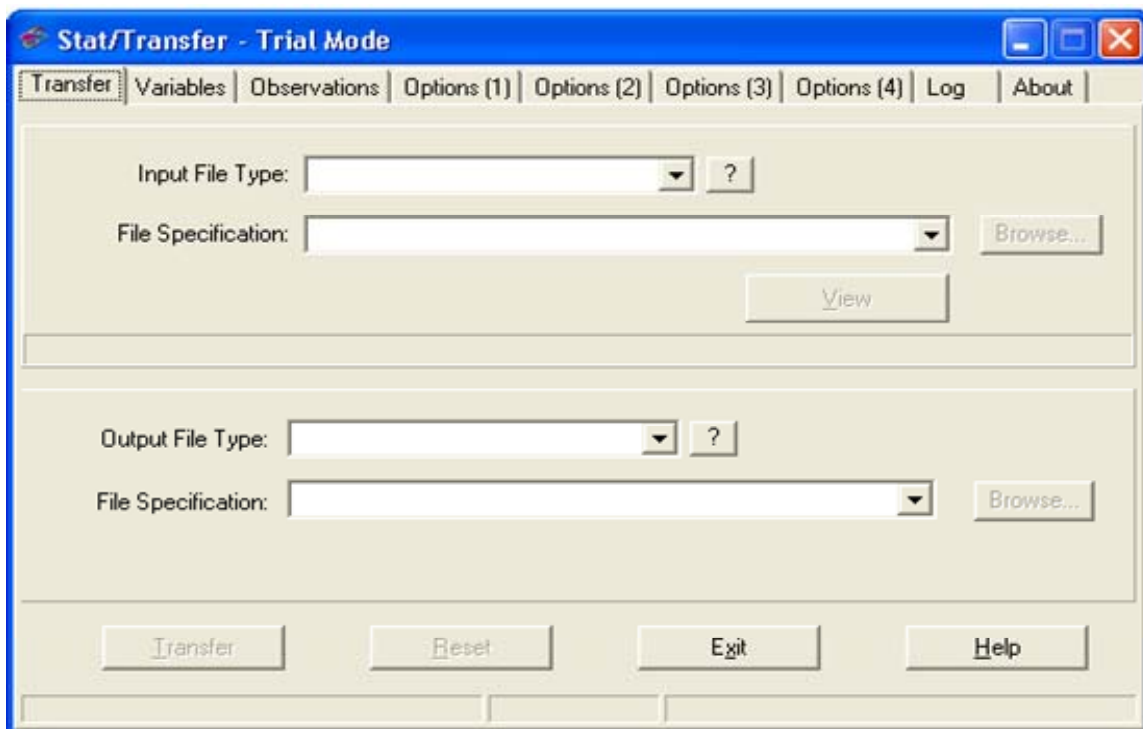
El programa tiene dos opciones: tipo de datos de entrada (*Input File Type*) y tipo de datos de salida (*Output File Type*), en la primera de ellas se selecciona la extensión o programa en el que está la base de datos, e inmediatamente en la parte de abajo se selecciona la ubicación donde se encuentra el archivo (puede seleccionar el archivo con

*browse* o *view*), así, con el paso anterior el programa identifica el archivo y formato o programa en el cual está originalmente los datos.

Ahora bien, el siguiente paso es seleccionar la extensión o programa en el que se desea tener los datos, a través de la opción tipo de datos de salida. Una vez seleccionado un formato en la parte inferior elegimos el folder o carpeta donde queremos guardar la nueva base de datos convertida a la extensión seleccionada. Tenga en cuenta que *Stattransfer* le dará automáticamente la misma ubicación al nuevo archivo donde se encuentran los datos originales.

Después de haber seleccionado el tipo de datos de entrada, el tipo de datos de salida y sus respectivas ubicaciones, se activará la opción transfer, con darle click, el programa empezará a transformar los datos a la nueva extensión. Una vez termina, podemos ver el nuevo archivo creado con la extensión que se ha predefinido; también es posible iniciar otro proceso de conversión con la opción *reset* o salir del programa con *exit*.

**Cuadro 1.** Salida inicial de *Stattransfer* 9



## 2. Introducción a *Stata* y comandos básicos

*Stata* es un programa de fácil instalación con una interface muy amigable. En el cuadro 2, se puede observar la salida inicial del programa. En la parte superior de la interfase de salida de *Stata*, vemos el menú principal con todas sus opciones (*File, Edit, Data, Graphics, Statistics, User, Windows y Help*), así como una barra de herramientas con once iconos, distribuidos en el siguiente orden (*Open, Save, Print, Log, New viewer, Graph, New do-file editor, Data editor, Data browser, Clear y Break*). En la misma salida observamos cuatro grandes ventanas, (i) *Results*, (ii) *variables*, (iii) *Review* y (iv) *Command*. A continuación se describe cual es el uso de cada una de estas ventanas:

### a. *Results*

En esta ventana encontramos el logo de *Stata*, la versión del programa, y la memoria disponible para cargar los datos y el número de variables. En esta ventana de resultados, como su nombre lo indica, tendremos todo tipo de resultados sobre los comandos que se ejecuten; también aparecen mensajes de lo que se está haciendo o lo que está ejecutando el programa, así como mensajes si cometemos algún tipo de error. Los colores de las palabras indican si son resultados de un comando, si es un comando o si es un error. Esta opción de los colores está predefinida pero se puede cambiar. Con *click* derecho sobre la ventana podemos copiar, imprimir el texto, o establecer preferencias.

### b. *Variables*

Esta ventana, que aparece a un lado de la ventana de resultados, muestra las variables que contiene el archivo que actualmente tenemos en memoria, también nos muestra información de las etiquetas de cada una de las variables; así como la información relevante de cada variable (*Name, Label, Type y Format*). Puede utilizarse para introducir los nombres de las variables en la ventana de comandos haciendo *click* sobre la variable. Igualmente, situándonos en la variable y haciendo *click* derecho se puede introducir

comentarios a esa variable. Para ver las notas de un archivo se puede usar el comando *.notes list*.

### ***c. Review***

La ventana de revisión muestra la lista de los comandos que recientemente se han ejecutado desde que se abrió algún tipo de archivo, sea de datos (*\*.dta*), un *do-file* (*\*.do*) o un *log* (*\*.log*). Desde esta ventana se pueden incluir comandos tecleados con anterioridad en la ventana *command*, con hacer *click* una vez en el mismo. Y haciendo *click* dos veces se copia y ejecuta.

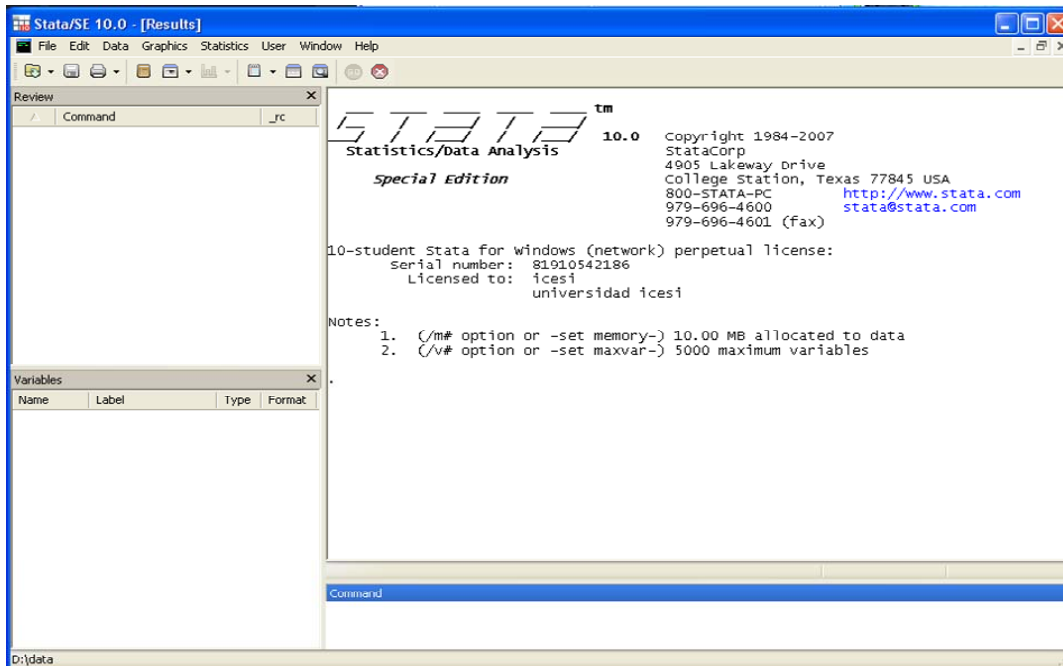
### ***d. Command***

En esta ventana se deben escribir los comandos que se desean ejecutar. Se pueden recuperar comandos escritos anteriormente o posteriormente con la tecla *RePág* o *AvPág*, hasta llegar al comando buscado, o también podemos buscarlos en la ventana *Review* y dándole *click*. Otra ayuda importante en esta ventana es la opción de completar automáticamente el nombre de una variable tecleada parcialmente con la tecla *Tab*.

En general, *Stata* permite ser usado por medio de ventanas con la opción del menú principal. Una vez se selecciona una opción del menú, se puede buscar y seleccionar la opción de lo que se desea hacer (inmediatamente el programa abre otra ventana, donde se define la tarea que deseamos que realice). Ejemplo: *File*→*Open*, con la opción *archivo (file)* se puede proceder a abrir un archivo e inmediatamente aparece una ventana adicional para abrir (*open*) un archivo.

La otra opción es por comandos o programación, que se debe hacer directamente en la ventana de comandos o creando un *do-file*. Ejemplo: se escribe directamente la opción *use or open*.

Cuadro 2. Salida inicial de *Stata* 10



Muchos de los comandos de *Stata* se pueden abreviar a las primeras letras. Ejemplo: el comando *inspect* se puede escribir *ins*, teniendo el mismo resultado.

También, hay que tener cuidado porque *Stata* es muy sensible al uso de mayúsculas y minúsculas. Ejemplo: para *Stata* no es lo mismo *inspect* que *INSPECT* o que *Inspect*. El comando correcto es *inspect*. Todos los comandos de *Stata* se escriben en minúsculas, mientras que los nombres de las variables dependen del creador del archivo de datos y pueden ir en mayúsculas, minúsculas o combinaciones.

Por otra parte, los tipos de archivos que podemos usar en *Stata* son los que se describen en el cuadro 3.

**Cuadro 3.** Extensión y tipos de archivos en *Stata*

<b>Extensión</b>	<b>Tipo de archivo</b>
<i>.dta</i>	Archivos de datos
<i>.do</i>	Archivos de comandos
<i>.ado</i>	Programas
<i>.hlp</i>	Archivos de ayuda
<i>.gph</i>	Gráficos
<i>.dct</i>	Archivos diccionarios
<i>.smcl</i>	Archivos log
<i>.raw</i>	Archivos de datos ASCII/text
<i>.dct</i>	Archivos de instrucciones ASCII

A continuación, se presentan algunos temas que es necesario conocer antes de empezar a usar el programa.

### **2.1 Algunas modificaciones (preferencias) de la ventana inicial**

El programa tiene unas preferencias establecidas para su uso. Sin embargo, estas pueden cambiar desde el menú principal. Siguiendo la ruta: *Edit* → *Preferences*

Algunas de las preferencias que podemos modificar son:

*Edit* → *Preferences* → *General preferences*

En las preferencias generales se pueden cambiar las preferencias de la ventana inicial de *Stata*. Por ejemplo, se pueden modificar los colores de la ventana de resultados, hay tres colores negro (predefinido), blanco y azul. En esta misma ventana, se pueden hacer cambios sobre los colores de las letras. Por ejemplo, el programa tiene predefinidos mensajes en letras rojas para errores. Aquí también se puede definir el tamaño de la ventana

y si se quiere minimizar o maximizar la ventana: *Edit* → *Preferences* → *General preferences* → *Manage preferences*.

## 2.2 Ayudas de *Stata*

*Stata* es un programa muy completo y en ocasiones es difícil saber para qué sirven todos los comandos predefinidos. El menú desplegable *Help*, permite de forma muy intuitiva buscar información sobre estadísticas, gráficos, manejo de datos, programación, etc., así como descargar de la red las últimas actualizaciones disponibles, programas realizados por analistas y puestos a disposición de todos los usuarios del *software*.

Para situaciones en las cuáles necesitamos ayuda podemos acceder fácilmente al menú de ayuda del programa. Igualmente, si tenemos dudas sobre qué hace un comando o cómo se utiliza, podemos acceder a la información tecleando simplemente:

*.help* nombre del comando

O, por ventanas, podemos acceder por *Help Stata* → *Comand* → “nombre del comando”.

Si no sabemos cómo se escribe el comando o desconocemos si el programa tiene predefinido un comando para lo que se desea hacer, se puede usar el comando:

*.search* nombre del comando

## 2.3 Modificar la memoria

Cuando abrimos un archivo de datos en *Stata* el programa mantiene los datos en la memoria. Así, si se está trabajando con un archivo de datos y se desea cargar otro archivo, es necesario remover el archivo de la memoria y para esto usamos el comando:

*.clear*

Tecleando este comando borramos todos los datos que se encontraban en la memoria.

En otras ocasiones cuando se intenta abrir un archivo puede salir el siguiente mensaje:

*.no room to add more observations*

*.r (901);*

Significa que la memoria asignada por defecto por el programa es insuficiente para cargar los datos o ejecutar lo que solicitamos. Se puede averiguar cuanta memoria hay asignada y cuanta se tiene libre tecleando:

*.memory*

Y se puede ampliar la memoria asignada, por ejemplo a 10Mb tecleando:

*.set memory 10m*

Se puede asignar una memoria permanente diferente a la predefinida siempre que se abra el programa, agregando el comando *permanently*.

*. set memory 10m, permanently*

La memoria asignada por defecto es insuficiente para la mayoría de trabajos en investigación y algunos en docencia. Por lo que es mejor siempre al inicio de cada sesión de trabajo ampliar la memoria.

## 2.4 Estructura de los comandos

La estructura general de la sintaxis de los comandos de *Stata* es la siguiente:

*[by varlist:] Command [varlist] [=exp] [if exp] [in range] [weight] [using filename] [, options]*

Donde los corchetes indican partes que tienen o no que añadirse según el comando específico del que se trate y lo que se desee hacer. A continuación se describirán brevemente cada una de las opciones entre corchetes.

**[by varlist:]**

Esta opción hace que el comando se ejecute por separado, dentro de cada grupo de variables del archivo o de algunas variables especificadas. El archivo tiene que estar ordenado por esas variables (para ordenar variables usamos el comando *.sort*):

*.sort sex age*

Este comando ordena los datos por sexos y dentro de cada sexo, por edades.

*.by sex age: regress y x1 x2 x3*

Este otro, estima una regresión de y sobre x1, x2 y x3 dentro de cada grupo de sexo y edad.

**[varlist:]**

Esta opción especifica las variables para las que se pide el comando. Por ejemplo:

- var1                      Variable var1.
- var1 var 2 var3        Variables var1, var2, var3.
- hh\*                        Variables que empiezan con hh.
- var1-var6                Desde la variable var1 hasta var6.

**[=exp]**

Esta opción sirve para expresiones matemáticas o lógicas. Especifica el valor asignado a una variable. Los operadores que pueden incluirse son los siguientes:

**Cuadro 4.** Operadores básicos de expresiones en *Stata*

Aritméticos	Lógicos	Relacionales (variables numéricas y de cadena)
+ suma	~ no	> mayor que
- resta	o	< menor que
* multiplicación	& y	>= mayor o igual que
/ división		<= menor o igual que
^ potencia		== igual que
+ encadenamiento de cadenas		~= no igual que

Es importante aclarar que las expresiones lógicas generan dos posibles resultados para *Stata*, según las va evaluando, observación a observación:

Verdadero=1, si son ciertas para esas observaciones, o bien,

Falsas=0, si no son ciertas para esa observación.

Otras expresiones que pueden utilizarse son las variables del sistema, que son variables internas de *Stata* cuyos nombres empiezan por “\_”. Por ejemplo, dos variables del sistema que pueden ser útiles son:

\_n → número de la observación, según el orden actual del fichero.

\_N → número total de observaciones (coincide con el \_n de la última observación)

### **[if exp]**

Las condiciones *if* se utilizan para restringir el campo de actuación de un comando a sólo las observaciones que cumplen la condición especificada. Por ejemplo:

```
.list var1 var2 if var1>20
```

```
.list var1 var2 if var1>20 & var1<30
```

```
.list var1 var2 if var1>20 | var1<10
```

### **[in range]**

Especifica las observaciones para las que ha de ejecutarse el comando. Ejemplo:

<i>in</i> 5	Observación número 5
<i>in</i> 1/100	Las 100 primeras observaciones
<i>in</i> f/100	Las 100 primeras observaciones
<i>in</i> 100/200	Observaciones desde la 100 hasta la 200
<i>in</i> -70/-1	Las 70 últimas observaciones (-1)
<i>in</i> 2300/1	Observaciones desde la 2300 hasta la final (1)

### **[weight]**

Esta opción la usamos para indicarle a *Stata* que tiene que utilizar unas determinadas ponderaciones al ejecutar el comando. Básicamente *Stata* acepta cuatro tipos de *weights*:

- ***fweight***: *frequency weights*, indica el número de casos que representa realmente cada observación muestral. La variable debe contener enteros positivos.
- ***pweight***: *sampling weights*, indica la inversa de la probabilidad de selección muestral de cada observación. Han de ser positivos, pero no necesariamente enteros.
- ***aweight***: *analytic weights*, indica los pesos inversamente proporcionales a la varianza de cada observación. Un uso típico de este tipo de ponderación es cuando las observaciones son medias y el peso representa el número de elementos que generan la media. Han de ser positivos, pero no necesariamente enteros.
- ***iweight***: *importance weights*, no tienen definición estadística formal, representan de alguna forma la importancia que se atribuye a cada observación. Cada comando que los acepta explica cómo los utiliza. Puede tener cualquier forma.

Estas variables de ponderación se incorporan a los comandos así (recuerde que los *weights* se especifican siempre entre corchetes):

`.command ponderación [weighttype=varname]`

**[using filename]**

Se utiliza sólo en algunos comandos, como *infile* o *outfile*.

**[, options]**

En la sintaxis de cada comando, que puede verse en las ayudas, se especifican las opciones disponibles. Las opciones se escriben siempre detrás de una coma, pero no es necesario poner comas entre las distintas opciones.

## 2.5 Comando para usar bases de datos que vienen con *Stata*

Otra de las herramientas básicas con las que podemos contar para practicar y perfeccionar nuestro entrenamiento en *Stata* es un conjunto de bases de datos que vienen incorporadas

automáticamente con el programa. El comando para abrir éstas bases de datos es `.sysuse` nombre de la base de datos. Un ejemplo es: `.sysuse auto`

Inmediatamente vemos cómo el programa carga un archivo de datos llamado `auto`. Alternativamente por ventanas podemos hacer lo mismo así: `File`→ `Example datasets`. Con esta opción podemos acceder a todas las bases de datos o podemos acceder a los ficheros por temas de estudio, regresión lineal, series de tiempo, datos de panel, etc.

En la siguiente sección veremos algunos de los comandos básicos para empezar a usar el programa.

### **3. Procesamiento de datos**

Después de descargar los datos a un *software* amigable como *Stata*, el econométra se enfrenta al procesamiento de los datos, variables y observaciones para realizar su análisis econométrico. A continuación se describen algunos de los comandos básicos para realizar un buen procesamiento de la base de datos.

#### **3.1 Abrir, guardar y salir**

##### **a. Abrir archivos de *Stata***

*Stata* permite varias opciones para abrir o leer ficheros de datos. Así, si ya tenemos grabados los datos en formato *Stata* (\*.dta), podemos abrir el archivo directamente con el comando `.use`

Algunas de las posibilidades que permite el programa para usar este comando son: si los datos están en una carpeta específica entonces,

```
.use "C:\carlos\2009\"
```

```
.use t12009
```

O también, para obtener el mismo resultado

```
.use "C:\carlos\2009\t12009.dta"
```

No es preciso indicar el directorio si los datos están en el directorio de trabajo actual (el directorio de trabajo lo da *Stata* en la línea inferior de la pantalla). Por defecto, el directorio de trabajo es C:\DATA. En este caso debería usar `.use t12009`

Además, no es preciso señalar la extensión `.data`, ya que el programa lo asume por defecto. También podemos abrir datos parciales de un archivo `.use x1 x2 using t12009`

Con la opción anterior sólo se abrirán las variables `x1` y `x2` del archivo `t12009`. Además, si queremos seleccionar las diez primeras observaciones, entonces usamos `.use t12009 in 1/10`

Las dos anteriores opciones se pueden combinar seleccionando de un archivo de datos sólo algunas variables y algunas observaciones. Cuando trabajamos al mismo tiempo con varias bases de datos debemos tener cuidado, ya que si tenemos en la memoria unos datos en los que hemos hecho cambios no grabados e intentamos abrir un archivo de datos nuevo, nos dará un mensaje de error con el aviso de que no hemos guardado los cambios. Para evitar esto debemos incluir al final el comando `.clear`, de la siguiente forma: `.use t12009, clear`

También podemos abrir una archivo de *Stata* por medio de ventanas: *File*→ *Open* (o directamente con el icono *Open*).

## **b. Abrir archivos en otro formato**

Para abrir datos que todavía no están en formato *Stata* se utilizan los comandos `.insheet`, `.infile` o `.infix`, dependiendo del formato y disposición de los datos

`.infile`

Este comando permite la lectura de archivos sin formato o con formato ASCII

*.infix e insheet*

Estos comandos sirven para leer datos desde un fichero auxiliar de formato fijo y hacer lectura recursiva de algún archivo, respectivamente.

### **c. Guardar**

Para grabar un archivo de datos se utiliza el comando

*.save nombre del archivo*

*Stata* graba el nuevo archivo en el directorio de trabajo y le añade automáticamente la extensión *.dta*, si deseamos que lo guarde en el directorio de trabajo predeterminado lo guardara por defecto siempre en *c:\data*. Debemos escribir *.save* nombre del archivo.

Ahora bien, si se tiene el archivo con ese nombre y se quiere guardar una nueva versión del mismo, con algunas modificaciones, como por ejemplo, nuevas variables que hemos creado, se añade el comando *.replace*.

*.save, replace*

Si se desea grabar en otro directorio distinto de trabajo se especifica:

*.save C:\econometria\nombre del archivo*

### **d. Salir**

El comando para salir del programa es *.exit*

También se puede hacer por *File*, del menú principal, o dando *click* en la x de la esquina superior derecha. Hay que tener en cuenta que si se tienen los datos sin grabar, pedirá confirmación.

### 3.2 Describir, listar e inspeccionar

#### a. Describir datos

Para facilitar la explicación de los comandos a partir de ahora se trabajará con los datos *auto.dta*, que vienen predefinidos como (*data examples*) en *Stata*. Se abren los datos:

```
.sysuse auto.dta
```

Una vez se tienen cargados los datos se puede empezar a trabajar con ellos. Lo primero que se hace es una descripción básica de la base de datos. El comando que se usa para describir la base de datos es:

```
.describe
```

O simplemente (**d**). Esta opción permite ver cuántas observaciones y cuántas variables había cuando se creó el archivo y qué tamaño tiene; así como una descripción de cada una de las variables con las que se puede trabajar. El cuadro 5 muestra la salida en la ventana de resultados de *Stata*.

**Cuadro 5.** Resultados del comando `.describe`

```
. describe
Contains data from C:\Archivos de programa\Stata10\ado\base/a/auto.dta
  obs:          74          1978 Automobile Data
  vars:         12          13 Apr 2007 17:45
  size:        3,478 (99.9% of memory free)  (_dta has notes)
```

variable name	storage type	display format	value label	variable label
make	str18	%-18s		Make and Model
price	int	%8.0gc		Price
mpg	int	%8.0g		Mileage (mpg)
rep78	int	%8.0g		Repair Record 1978
headroom	float	%6.1f		Headroom (in.)
trunk	int	%8.0g		Trunk space (cu. ft.)
weight	int	%8.0gc		Weight (lbs.)
length	int	%8.0g		Length (in.)
turn	int	%8.0g		Turn Circle (ft.)
displacement	int	%8.0g		Displacement (cu. in.)
gear_ratio	float	%6.2f		Gear Ratio
foreign	byte	%8.0g	origin	Car type

```
Sorted by:  foreign
```

En el siguiente apartado se explica otro de los comandos importantes para conocer la base de datos con la que se trabaja en un estudio econométrico.

## **b. Inspeccionar datos**

*.inspect*

Proporciona un resumen básico del tipo de valores que tiene una variable numérica y un histograma de: valores positivos, nulos, negativos, enteros y no enteros de la variable, así como los valores *missing* que existen. Estos últimos están codificados en *Stata* como un punto (.) en las variables numéricas y un espacio ( ) en las variables de texto. Así si deseamos inspeccionar una sola variable debemos escribir *.inspect* nombre de la variable; pero si queremos inspeccionar todas las variables escribimos sólo *.inspect* o *.inspect \_all*. Otros dos comandos que sirven para inspeccionar los datos son: *.edit* y *.browse*

*.edit*

Permite editar la información contenida en la base de datos. Nos abre la ventana del *data editor*.

*.browse*

Nos permite ver y revisar los datos que tenemos cargados en *Stata*.

## **c. Listar datos**

Igualmente, si se quiere obtener un listado de los valores que tienen las observaciones de una o más variables, se usa el comando *.list*

Si se desea un listado para todas las variables se usa el comando sólo, pero si se quiere hacer un listado solo de los valores de una variable o de algunas, se escriben los nombres de las variables después del comando: *.list x1 x2*.

Podemos filtrar la lista, a continuación se presentan tres ejemplos:

*.list x1 x2 in 1/5*

Lista de las primeras 5 observaciones de las variables indicadas

*.list x1 in -5/-1*

Lista de las últimas 5 observaciones de las variables seleccionadas

*.list x1 if x2 <0*

Lista todas las observaciones de la variable x1 para las que se cumple que el valor de x2 es negativo.

Es importante aclarar que el resultado puede ser muy largo. Cuando ocurra esto podemos cortar el resultado usando la letra **(q)** en la ventana de comandos. Automáticamente se cortará la acción que está ejecutando el programa.

Otros dos comandos que permiten comprender los datos con los que se está trabajando son:

*.count,*

Permite conocer cuál es el tamaño de la muestra.

*.codebook*

Muestra una descripción de los datos, algunas estadísticas descriptivas como la media y una distribución de los datos por percentiles.

### **3.3 Seleccionar y eliminar variables**

#### **a. Comando para seleccionar de una base de datos las variables de interés**

En la práctica de la econometría en ocasiones se tienen bases de datos con un gran número de variables. Para evitar que el trabajo sea engorroso por tener cargadas en el

programa unas variables que no son de interés, hay dos opciones básicas: i). seleccionar las de interés o, ii) eliminar las que no son de interés de la memoria.

### *.keep*

Este comando es el de la primera opción, permite seleccionar las variables de la base de datos con las que se desea trabajar. El comando *.keep* guarda en la memoria las variables u observaciones indicadas, eliminando las restantes. Algunos ejemplos del uso de este comando son:

- . keep make price*      Conserva las variables *make* y *price*, eliminando el resto.
- . keep in 1/100*        Conserva las 100 primeras observaciones y elimina el resto.
- . keep if price >= 0*    Conserva las observaciones con precio positivo y elimina el resto.

### **b. Comando para eliminar variables**

### *.drop*

Este comando permite borrar de la memoria las variables u observaciones de la base de datos, conservando las restantes. Algunos ejemplos pueden ser:

- . drop price*                      Elimina la variable *price*
- . drop in 1/15*                    Borra las 15 primeras observaciones
- . drop if price < 0*                Borra las observaciones en las que el precio es negativo

Cuando se usa uno de estos comandos, eliminar o seleccionar, *Stata* comunica que es lo que ha hecho con las variables o la muestra.

### **3.4 Cambiar nombres de las variables y hacer etiquetas**

Una vez seleccionadas (filtramos) las variables y la muestra de interés, se pueden cambiar los nombres “acortándolos” o poniendo nombres más intuitivos para facilitar el trabajo con las variables. También se pueden crear etiquetas (*label*) a cada una de las variables para que ayuden a identificar qué es en si la variable. A continuación se describe una serie de comandos útiles:

### a. Cambiar nombre de variables

*.rename*

Cambiar el nombre de la variable “*price*” por “precio”.

*. rename price precio*

Se puede hacer este cambio de nombre de la variable, con todas si es necesario.

### b. Hacer etiquetas a las variables

*. label*

Es un comando que sirve para crear una etiqueta para cada variable. La etiqueta es una palabra o una serie de palabras que ayuda a identificar qué es cada variable. Así por ejemplo, la variable *Price*, que ahora es precio, es el precio de mercado de los carros (precio de mercado nominal en dólares). Se puede crear la etiqueta para las variables a fin de identificar de forma rápida qué es cada variable.

*. label variable precio “precio de mercado nominal en dólares”*

Entre comillas va la etiqueta que deseamos para la variable.

Los cambios de nombres (*. rename*), etiquetas (*. label*) y también formatos (*. format*) de las variables se pueden hacer fácilmente desde el *Data Editor* (*Data* → *Data Editor*). Situando el cursor encima de la variable y pulsando dos veces, aparece en pantalla la información sobre el nombre, etiqueta y formato actual de esa variable, y se pueden introducir ahí los cambios. Al salir del editor pedirá confirmación de los cambios si no se han marcado explícitamente (*, preserve*) antes de abandonarlo. También es posible poner etiquetas a las variables desplegando el menú *Data* → *Labels*.

## 3.5 Creación y modificación de variables

A continuación se presentan algunas formas básicas de crear y modificar variables entre las que se destacan: generar, recodificar, renombrar y organizar variables.

## a. Generar

El comando más simple y directo para crear nuevas variables es *generate*. La sintaxis básica es:

```
. generate newvar = expresión
```

Es importante recordar, que la expresión puede incluir cualquier operación aritmética, de relación o funcional (ver cuadro 4). También es importante tener en cuenta que este comando como muchos otros de *Stata* pueden restringirse a determinados casos usando las condiciones (*if - in*), este último para un rango de observaciones.

Una extensión del comando *generate* es el comando *egen*.

```
. egen
```

Este comando puede utilizarse con determinadas funciones que se señalan a continuación. Dependiendo de la función, los argumentos se refieren a una expresión, una lista de variables, etc. Las opciones también dependen de la función utilizada. A continuación se da un ejemplo sencillo de cómo utilizar este comando.

Se crea un archivo con cuatro observaciones:

```
.set obs 4
```

Se generan tres variables var1, var2 y var3:

```
.generate var1=2
```

```
.generate var2=4
```

```
.generate var3=var1 + var2
```

Se ha creado una base de datos con tres columnas (variables) y con cuatro observaciones, cada una. A continuación, se crean dos nuevas variables var4 y var5.

Var4 será la suma acumulada de cada observación de var3 (va sumando renglón tras renglón, el dato de la observación fila por fila): `.generate var4=sum(var3)`

Y var5, será la suma acumulada final de la variable var3: `.generate var5=sum(var3)`

Para generar una variable con el número de una observación en la base de datos se puede usar: `.generate id=_n`

Otra opción que permite este comando es generar una variable *dummy*: `.generate dummy1=sexo==2`

Con este comando se crea una variable *dummy* que es igual a 1 si sexo=2 y cero en otro caso.

También se puede crear de esta otra forma:

```
.generate dummy1=0  
.generate dummy1=1 if sexo==2
```

En cualquiera de estos casos se debe tener cuidado si hay valores *missing*, puesto que se estaría asignando un cero a algo que en realidad es un *missing*. Para arreglar este problema tenemos: `.replace dummy1= if sexo== .`

## **b. Recodificar**

Este comando puede cambiar el valor de una variable determinada.

```
.recode var1 2=0
```

Algunas encuestas vienen con los *missing* codificados, estos se pueden convertir nuevamente en *missing*.

`.recode val 999=.`

También se puede recodificar una variable. Por ejemplo; recodificando la edad en rangos: `.recode edad 15/25=1 26/35=2 36/45=3 46/55=4 56/65=5 *=9`

Otros ejemplos son:

`.recode edad 0/25=1 25/50=2 50/max =3, gen(edad_agrupada)`

`.recode x (1 2 3=1) (4 5 6 =2), gen(n_x)`

### c. Renombrar

Cambia el contenido de una variable ya existente.

`.replace`

Así si deseamos que en la variable edad todas las observaciones que tienen una edad mayor o igual a 65 tomen el valor de 65, entonces se usa el siguiente comando: `.replace edad=65 if edad>=65`

### d. Renombrando variables

Un comando muy útil para renombrar algunas variables ya existentes es: `.rename`. Este comando puede cambiar los nombres de las variables para que se ajusten a nuestro interés, hacer más cortos los nombres o más familiares.

### e. Otros comandos: *sort*, *gsort*

Es muy frecuente la necesidad de organizar una base de datos según una o varias variables, dando prioridad al orden de la variable que se pone en primer lugar y así sucesivamente. Por defecto este comando ordena las observaciones de menor a mayor. Si se especifican varias variables *Stata* ordena por cada una dentro de la anterior.

`.sort var1`

El comando anterior ordena las observaciones de menor a mayor según var1

`.sort var1 var2`

El comando anterior ordena las observaciones por la variable 1 y dentro var1, se ordena de menor a mayor según la variable 2.

Un comando que permite elegir cómo se debe ordenar cada variable, y a su vez, permite cambiar entre descendente o ascendente para cada variable es: `.gsort +var1 -var2`

Con este comando ordenamos var1 de menor a mayor y var2 de mayor a menor.

### **3.6 Combinación de bases de datos**

#### **a. Comando *merge***

Este comando sirve para añadir variables a un archivo de datos que tenemos abierto. Esto es pegar datos de forma horizontal, o sea, añadir variables a las observaciones existentes. Hay que tener en cuenta que no es necesario que los dos archivos de datos tengan exactamente las mismas observaciones. Este comando es muy apropiado cuando se tienen datos de los individuos que participan en una encuesta y se reciben datos de un segundo modulo de la encuesta. Por ejemplo, la Encuesta Nacional de Hogares (ENH), Encuesta Continua de Hogares (ECH) y la Gran Encuesta Integrada de Hogares (GEIH), realizada por el Departamento Administrativo Nacional de Estadísticas (DANE).

Para poder llevar a cabo con éxito esta orden, ambos conjuntos de datos deben estar ordenados con base en las mismas variables y en el mismo orden. Entonces, este comando se usa con el siguiente orden. Primero se abre el archivo de datos (`.use`), en segundo lugar se ordenan las variables, en este caso se supone que las variables ordenadas son var1 y var2 (`.sort`) y en tercer lugar se hace el *Merge*:

`. merge var1 var2 using ECH`

Noten que *merge* crea una variable adicional *\_merge*, esta variable puede tomar tres valores que nos sirven para revisar si estamos trabajando correctamente con la base de datos. Esta variable toma los siguientes valores:

*\_merge==1* para las observaciones del archivo “*master*”  
*\_merge==2* para las observaciones del archivo “*using*”  
*\_merge==3* para las observaciones presentes en ambos archivo

Recuerde que en ocasiones se puede tener una nueva versión de algunas variables. Que pueden combinarse con las anteriores con la opción *update replace*.

`.merge using filename, update replace`

### **b. Comando *append***

El comando *append* es muy útil para unir archivos de datos. Esto es pegar datos de forma vertical en una base de datos. Al contrario de *merge* con el que agrega variables, este comando agrega observaciones.

`. append`

Este comando añade un archivo de datos con formato *Stata* al final del archivo que se tiene abierto. Es una combinación vertical de bases de datos, añadiendo al final

del archivo que está abierto las observaciones. Para el uso de este comando no se requiere que los dos archivos de datos tengan exactamente las mismas variables.

### **c. Comando *joinby***

Con este comando se pueden crear diferentes combinaciones entre varias bases de datos. Este comando crea un archivo de datos con todas las parejas entre archivos. Lo primero es abrir el archivo de datos (*.use*) y posteriormente

*. joinby idenh using varfam*

Con este comando combinamos usando una variable de identificación del hogar usando *varfam*, variables familiares.

Otros comandos importantes para la combinación de bases de datos son: *.cross* y *.fillin*.

*. cross*

Este comando crea todas las posibles combinaciones entre ambos ficheros. Y el comando *.fillin*

Rellena con observaciones todas las posibles combinaciones de un listado de variables. Los valores de otras variables aparte de las que definen el relleno se asignan a *missing*. Esta opción es buena cuando se trabajan datos longitudinales para balancear datos de panel.

### **3.7 Archivos *log*, *do* y *ado***

Primero se describe el papel de los archivos *log file*; para posteriormente pasar a los *do file* y finalmente a los *ado file*. Se observa que en la ventana de resultados no se muestran todos los resultados de la sesión, sólo los últimos, y puede ocurrir que tras ejecutar una orden que genera un *output* especialmente largo sólo se tenga “en memoria” la última parte. Por ello, la forma normal de trabajar es abrir un *archivo log* al principio de la sesión. Los *archivos log* contienen los comandos y los resultados del análisis (no los gráficos), este tipo de archivos se pueden abrir por *Stata* o por un procesador de texto. También, pueden crearse los archivos log (por defecto), en un formato de *Stata* (*.smcl*). Algunos ejemplos de este tipo de archivos son:

## a. Archivos *log file*

*.log*

Al principio de cada sesión de trabajo se debe crear un archivo *log*.

*. log using* nombre del archivo

Por defecto tiene formato *smcl* (*Stata markup control language*).

Si se desea que el archivo se pueda abrir en un procesador de texto. El comando sería:

*. log using* nombre del archivo *.log*

Este último tiene formato ASCII, y se puede abrir en el block de notas, como un archivo de texto *.txt*.

Con los comandos *log* podemos usar dos nuevos comandos (*,append* y *,replace*).

*. log using* carlos.*log*, *append*

Este comando es similar a (*. log using* nombre del archivo) y si existe ya otro fichero con el mismo nombre, continua grabando encima del archivo. Mientras que el comando (*,replace*) lo que hace es reemplazar el archivo existente por el nuevo.

*. log using* carlos.*log*, *replace*

Para ver el *log file*, usamos el comando (*. view*).

*. view* carlos.*log*

También se puede abrir o convertir el *log file* a un formato de texto y poder abrirlo en cualquier procesador de texto.

*. translate* carlos.*smcl* to hwl.*txt*

Igual resultado se obtiene, con el siguiente comando:

*. translate* carlos.*smcl* to hwl.*log*

Cuando se está en una sesión Stata con un *log* abierto, se puede interrumpir o reanudar temporalmente la grabación en el *log* tecleando los comandos:

*. log off*

Cierra el *log*, o lo desactiva.

*. log on*

Abre el *log*, o lo activa.

Es posible introducir comentarios en el *log* a través de líneas que empiezan por \*. Por ejemplo:

\*INTRODUCCION Y ANALISIS DE DATOS\*

\*\*\*\*\*INTRODUCCION Y ANALISIS DE DATOS\*\*\*\*\*

Basta con un asterisco al principio, pero con muchos (\*) el comentario resalta más y es de más ayuda para leer fácilmente el *log*.

El *log* se cierra automáticamente al salir de Stata pero también se puede cerrar en cualquier momento de la sesión o al final de la sesión con los siguientes comandos:

*. log close*

*. cmdlog close*

## **b. Archivos *do file***

Los *do files* son archivos que contienen una lista ordenada de instrucciones de Stata, que se ejecutan de una sola vez. Se crean, graban, cargan, modifican y ejecutan utilizando el *do file Editor*, aunque también pueden utilizarse otros editores de texto para escribir las instrucciones (se recomienda el block de notas). Son especialmente útiles cuando se desean utilizar los mismos comandos de forma repetida sobre muestras distintas, o bien reproducir los resultados con algunos cambios. A menudo incorporan la utilización de programas definidos por el usuario, así como instrucciones para crear los oportunos ficheros *Log* en los que se graban los resultados.

Los comandos se pueden escribir en cualquier editor de texto o en el editor de texto de *Stata (Do-file editor)*.

Los comandos se pueden separar por *enter* o *return*. En general un archivo *do file* sería:

```

Clear
Set memory 1g
Cd "C:\data\carlos\"
capture log close
log using carlos.log, replace
set more off
log close

```

El único comando que se desconoce del ejemplo del *do file* es (*. set more off*). Este comando es útil cuando la extensión de los resultados (en la ventana de resultados) supera una página, entonces *Stata* pausa el proceso y pregunta si se desea continuar con la opción (*more*). El comando *set more off*, nos permite hacer el *do file* sin pausas. La opción contraria es *.set more on*.

Para ejecutar un archivo *do* y mostrar los resultados, se utiliza el siguiente comando:

```
. do carlos.do
```

Mientras que para ejecutar un archivo *do* y no mostrar los resultados:

```
.run carlos.do
```

El uso real de *Stata* se apoya normalmente en la construcción y ejecución de archivos *do file*, más que en la forma interactiva de trabajo. Por eso, la importancia de estos comandos, *log*, *do* y *ado*. Recordemos que los *do files* son archivos de comandos, mientras que los *ado* son macro archivos de programación.

### **c. Archivos *ado file***

Un archivo *ado file (automatic do file)*, es como un archivo *do*, o sea que es un archivo que contiene una serie de líneas de programación *Stata*. Pero a diferencia de un archivo *do*,

tiene que estar archivado en determinados directorios y se ejecuta de la misma forma que los demás comandos de *Stata*. Los archivos *ado* se pueden buscar con el comando *sysdir*, y los encontraremos en el directorio preestablecido previamente por *Stata* o por el autor del archivo *ado*.

*. sysdir*

Cuando se encuentra el archivo *ado* lo siguiente sería ejecutarlo. Este tipo de archivos se ejecuta igual que cualquier comando de *Stata*. Si no lo se ha creado o lo estamos creando lo primero es guardarlo en un directorio predefinido que esté usando *Stata*. Luego se usa la opción *do*.

*.do nombre del ado*

Este comando le dice a *Stata* que lea el archivo *do*, luego se debe ejecutar dándole el nombre del *ado*. Por otra parte, algunos econométricos han escrito programas *ado* que hoy en día se pueden encontrar publicados en *Stata Technical Bulletin*, o directamente en la página web de *Stata* en forma de *ado files*. Como ya se mencionó estos programas *ado* se pueden descargar desde Internet de manera permanente de tal forma que *Stata* los reconozca como si fueran comandos internos (predefinidos) del programa. Estos archivos *ado* en su mayoría vienen acompañados de un archivo que explica exactamente la sintaxis y el funcionamiento del nuevo comando. Entre los *ado files* que pueden descargarse de Internet se encuentran las actualizaciones oficiales de *Stata*.

Para buscar e instalar a *do files* sobre, por ejemplo, desigualdad, hacemos: *Help* → *Search* → *Search net resources* → *inequality* (=palabra clave). Genera una lista de *ados*, con una pequeña descripción. Se entra en los que interesan y, si se desea instalarlos, se siguen las instrucciones (*click here to install*).

#### **4. Estadísticas descriptivas**

En econometría después de obtener los datos y organizarlos es necesario realizar algunas estadísticas básicas para familiarizarnos con las variables de la base de datos. Es por ello

que en esta sección se presentan algunos de los comandos básicos que tiene predefinido Stata para realizar estadísticas descriptivas.

### **a. Estadísticas descriptivas básicas**

El comando *summarize* obtiene las estadísticas descriptivas básicas de las variables que tengamos en la base de datos.

```
.summarize
```

El comando anterior permite obtener algunas estadísticas descriptivas como la media, la desviación estándar, el mínimo, el máximo y el número de observaciones para todas las variables de la base de datos.

```
.summarize var1
```

Este comando obtiene las estadísticas descriptivas de la variable *var1*. Se puede usar el comando para un conjunto de variables a la vez. Igualmente, se puede entrar en detalle de las estadísticas descriptivas usando el comando *detail*.

```
.summarize var1, detail
```

El comando anterior obtiene las estadísticas descriptivas detalladas para la variable (*var1*).

### **b. Tablas**

Con frecuencia es necesario realizar tablas con algunas estadísticas como las frecuencias entre otras. Por esto, en esta sección se introduce el tema de cómo realizar algunos tipos de datos que nos permiten obtener información estadística y demás. El comando que nos permite obtener una tabla con la frecuencia de una o dos variables es:

```
. tabulate
```

Este comando obtiene la frecuencia de los datos. Por defecto, aparecen las frecuencias absolutas, las porcentuales y las porcentuales acumuladas. Algunas opciones adicionales a este comando son:

```
. tabulate var1, plot
. tabulate var1, nolabel
```

Con la primera opción se obtienen las frecuencias absolutas junto con un pequeño gráfico de barras. Y con la segunda opción, se obtienen en la tabla, los valores de las variables, en lugar de las etiquetas de esos valores, eso sí, cuando tenemos asignada una etiqueta a esos valores.

Con este mismo comando (*tabulate*), se pueden obtener tablas de cruce de variables (o tablas de doble entrada). Así por ejemplo, si se desea cruzar la edad con el sexo, se puede hacer con este comando.

```
.tabulate var1 var2
```

Por defecto con este comando aparecen solo las frecuencias absolutas. Algunas de las opciones que se pueden especificar son:

<i>.tabulate var1 var2, row</i>	frecuencias relativas horizontales
<i>.tabulate var1 var2, col</i>	frecuencia relativas verticales
<i>.tabulate var1 var2, cell</i>	frecuencias relativas totales
<i>.tabulate var1 var2, nofreq</i>	no presenta las frecuencias absolutas
<i>.tabulate var1 var2, nolabel</i>	idéntico a <i>.tabulate var1 var2</i>
<i>.tabulate var1 var2, missing</i>	frecuencias con el porcentaje de valores <i>missing</i>
<i>.tabulate var1 var2, chi</i>	frecuencias con el estadístico <i>chi2</i>

Con la opción *chi2* junto con el comando *tabulate* calcula el coeficiente de chi cuadrado de *Pearson* para la hipótesis de que las filas y columnas en una tabla de dos variables, son independientes.

Existen dos extensiones de este comando que resultan frecuentemente muy útiles: *tab1* y *tab2*.

Con la primera opción:

```
.tab1 var1 var2 var3
```

Se obtienen tablas de frecuencias separadas de una lista de variables.

Con la segunda opción, se pueden obtener tablas de frecuencias para cada variable:

```
.tab2 var1 var2 var3
```

Otras dos opciones para realizar tablas que permitan analizar las estadísticas descriptivas de la base de datos son: *.tabstats* y *.table*. La primera es una opción muy recurrida para construir tablas con estadísticos:

```
.tabstat
```

Este comando puede calcular las estadísticas que se desean de las variables. Por ejemplo con el mínimo, la mediana, la media, el máximo, el tamaño de la muestra y coeficiente de variación.

```
.tabstat var1 var2 (min median mean max n cv)
```

Por otro lado, con el comando *.table* se pueden crear tablas de estadísticos controlando el contenido de cada casilla. Por ejemplo, crear una tabla cruzada de dos variables *var1* y *var2*, y que este controlada por la media de la *var3*.

```
.table var1 var2, cont(mean var3)
```

### **c. Correlaciones**

Otro estadístico relativamente importante en econometría es el análisis de correlaciones. El comando para realizar correlaciones es:

`.correlate var1 var2`

Esta opción es para cuando se tienen dos variables, Pero también puede utilizarse cuando se tienen varias variables:

`.correlate var1 var2 var3 var4`

Así con el comando anterior se calcula el coeficiente de correlación de Pearson y aparecerá una matriz de correlaciones. Igual que con todos los comandos anteriores podemos usar filtros para usar sólo una parte de la muestra o usar alguna variable de control.

#### **d. Algunos *tests* de comparación de medias**

En ocasiones es necesario hacer una comparación de medias antes de pasar a hacer la estimación del modelo. Utilizando el comando:

`.test`

Se obtiene un *test* de comparación de medias, donde se puede contrastar la hipótesis de que las medias de la variable (var1) son iguales y no dependen de la variable (var2).

`.ttest var1, by(var2)`

Para que sean significativas las diferencias es necesario observar si el valor p de la  $H_a: \mu_1 \neq \mu_2$ , sea cercano a cero (0,000). (0.05; indica un t de 1,96).

## **5. Gráficos**

Realizar algunos gráficos es esencial en econometría por lo cual esta sección se dedica a explicar brevemente como realizar gráficos en *Stata*. En ocasiones los gráficos permiten hacer análisis intuitivo del comportamiento de algunas variables o de cómo se relacionan entre ellas. *Stata* ofrece numerosas posibilidades y opciones para realizar gráficos.

Algunos de los comandos para realizar gráficos son:

*.graph twoway*  
*.graph matrix*  
*.graph bar*  
*.graph dot*  
*.graph box*  
*.graph pie*

Se puede obtener ayuda para cada uno de estos gráficos con el comando *.help*.

*.help twoway*

Así se puede hacer con todas las opciones antes descritas. *help graph\_matrix,...*etc.

Algunos comandos que permiten grabar gráficos y vuelven a dibujar gráficos previamente guardados son:

*.graph save*  
*.graph use*  
*.graph display*  
*.graph combine*

Los comandos que permiten imprimir gráficos son:

*.graph print*  
*.graph printcolor*  
*.graph export*

Los comandos que permiten realizar operaciones con los gráficos que están en la memoria son:

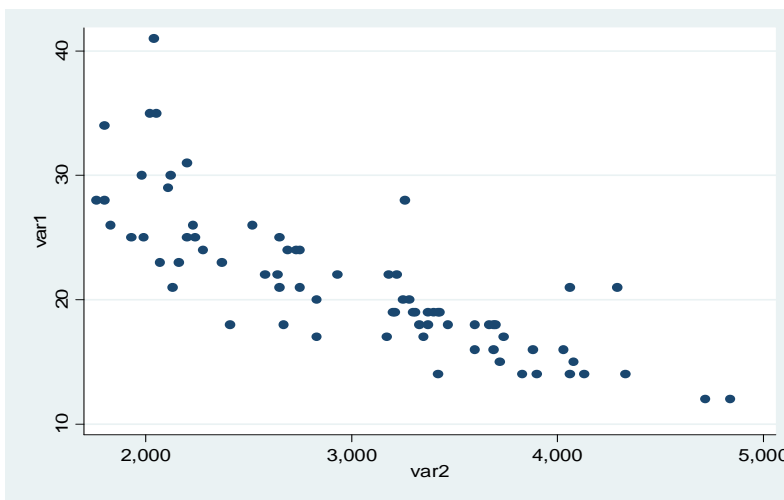
*.graph display*  
*.graph dir*  
*.graph rename*  
*.graph copy*  
*.graph drop*

**a. Diagrama de dispersión (*scatter*)**

Uno de los gráficos más comunes en econometría es el diagrama de dispersión. Con el comando *scatter* se pueden representar las observaciones en una nube de puntos. Con las opciones *line* se unen las observaciones, y la opción *connected* une las observaciones representadas por los puntos.

```
.scatter var1 var2
```

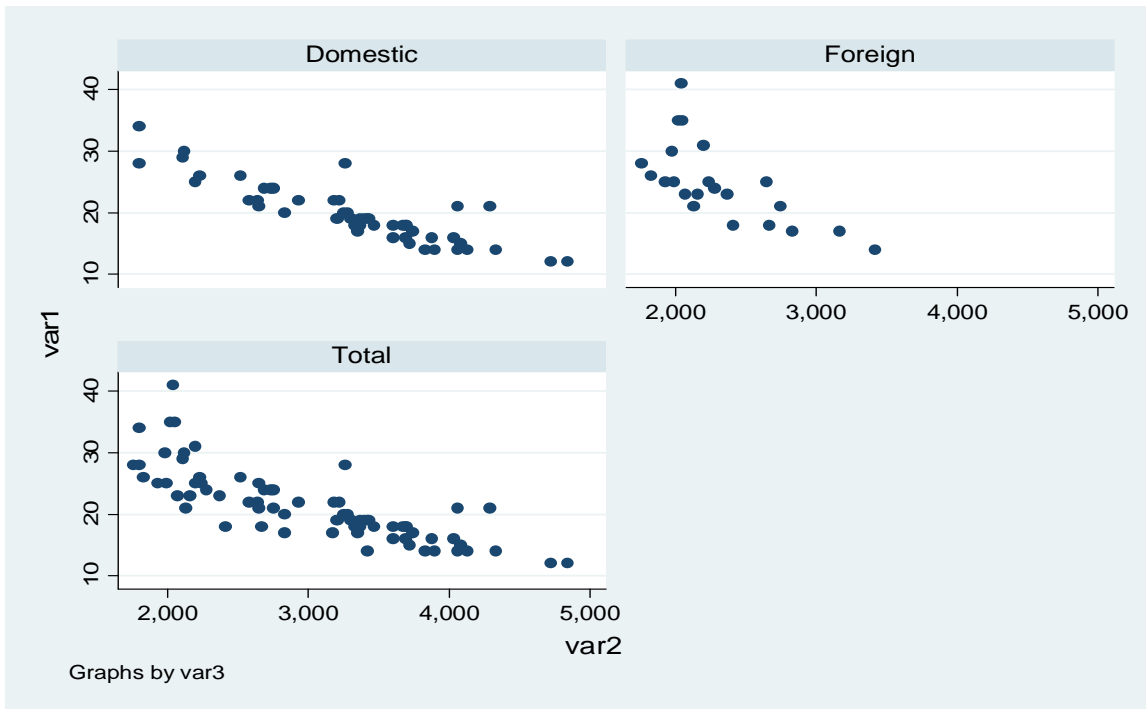
**Gráfico 1.** Diagrama de dispersión entre dos variables



Con la opción anterior podemos hacer un solo gráfico, pero también podemos hacer varios gráficos.

```
.scatter var1 var2, by (foreign, total)
```

**Gráfico 2.** Diagramas de dispersión (varios diagramas en un solo gráfico)



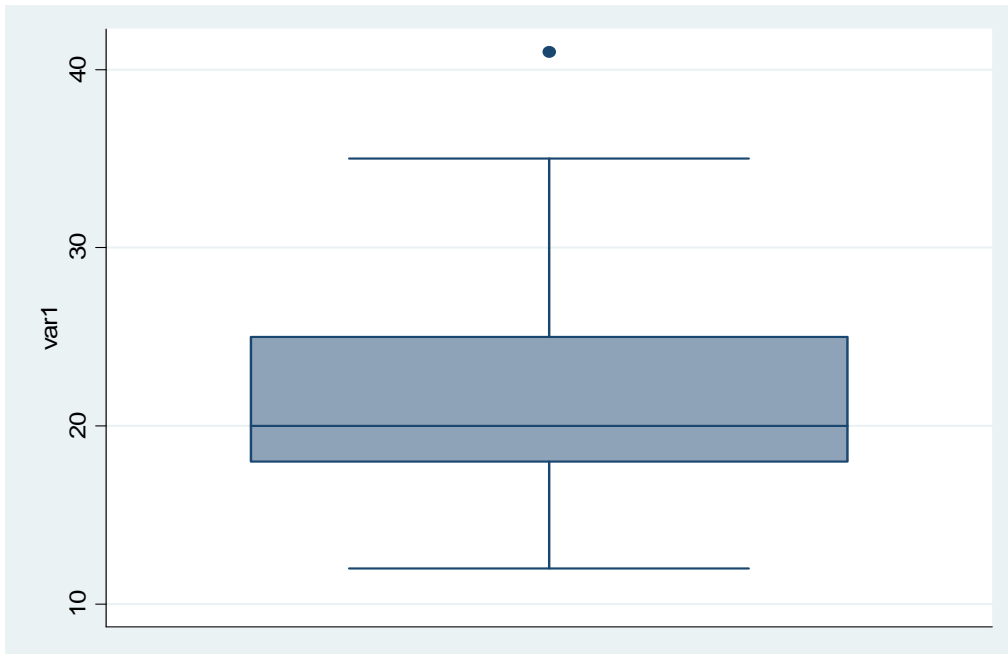
En este tipo de gráficos también se puede agregar la línea de regresión. Ya que con el comando *scatter* sólo aparece la nube de puntos, se debe agregar el comando *line*.

### b. Gráficos de cajas (*Box Plots*)

Los gráficos de cajas proporcionan información básica sobre la distribución de las variables; además permite observar la asimetría y los *outliers*.

```
.graph box var1
```

**Gráfico 3.** Gráfico de cajas (*Box Plots*)



### c. Gráficos de barras

Ésta es una de las opciones más comunes de gráficos que se pueden encontrar para analizar tanto la evolución de una variable como su comportamiento. Con esta opción se pueden realizar gráficos de barras en presentación vertical (*.graph bar*) y también horizontal (*.graph hbar*). Hay que tener cuidado pues el eje Y es la variable numérica y el eje X es la variable categórica.

También es importante recordar que la media de la variable numérica puede ser sustituida por cualquier estadístico (*mean, median, min, max, sum, count, ....etc*). A continuación se presenta un ejemplo.

```
.graph bar var1, over(var2)
```

Se puede cambiar el comando para realizar el gráfico de forma horizontal.

```
.graph bar var1
```

También se puede hacer el gráfico con las estadísticas descriptivas:

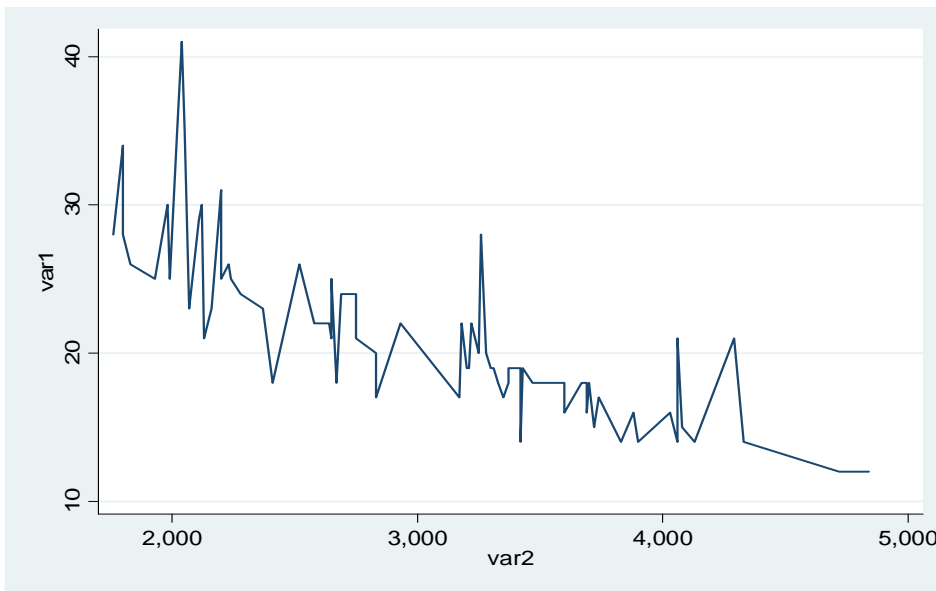
`.graph bar (mean) var1, over(var2)`

#### d. Gráficos de dos variables

Estos son los gráficos más comunes, se pueden hacer gráficos de líneas, áreas, barras entre otros muchos para una combinación de dos variables. Por ejemplo, la evolución de un precio en el tiempo. El comando es:

`. twoway`

**Gráfico 4.** Gráfico de líneas entre dos variables

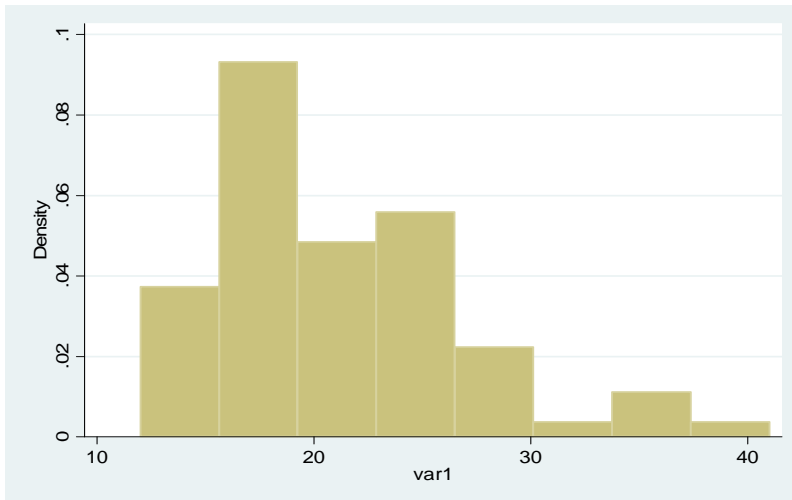


#### e. Histogramas

Con este comando se pueden hacer histogramas de variables continuas y discretas. Permite la opción *fweight* y la opción *by*. La opción por defecto sobrepone al histograma el gráfico de una distribución normal.

`. histogram var1`

**Gráfico 5.** Histograma de una variable



## 6. Referencias

Adkins L.C y Carter R. (2008). Using Stata for Principles of Econometrics. Wiley.

Cameron A.C y Trivedi P.K (2009). Microeconometrics using Stata. Stata Press

Hamilton, L.C. (2009). Statistics with STATA 8. Belmont, CA: Duxbury Press

Kohler, U. y Kreuter, F. (2009). Data Analysis Using Stata. College Station, TX: Stata Press

Rabe-Hesketh, S. y Everitt, B. (2004). A Handbook of Statistical Analysis Using STATA. London: Chapman & Hall/CRC Press

Stata Corp (2008). User's Guide, Reference Manual Release 10. Stata Press.

Algunos recursos en Internet para usuarios Stata:

- <http://www.ats.ucla.edu/stat/stata/>
- <http://econpapers.hhs.se/paper/bocbocoec/531.htm>

- <http://fmwww.bc.edu/ec/res.info.php>
- <http://ideas.repec.org/s/boc/bocins.html>
- <http://ideas.repec.org/s/boc/bocode.html>

## Documentos de Políticas Públicas - POLIS

### Artículos Publicados

<b>Policy Paper Número</b>	<b>Autor(es)</b>	<b>Título</b>	<b>Fecha</b>
2008 - 001	Pablo Sanabria Natalia Solano	Seguimiento a las Finanzas Públicas del Valle del Cauca: 2004 – 2006	Julio de 2008
2008-002	Juan Pablo Milanese	Relaciones ejecutivo-legislativo en la actual coyuntura política colombiana, un análisis desde la lógica de los veto players	Octubre de 2008
2008-003	Juanita Villaveces	Política de tierra en Colombia: Enfoques y perspectivas de política pública	Noviembre de 2008
2009-001	Jhon James Mora Carlos Giovanni González	Desaceleración de la economía y las políticas activas de empleo: una estrategia común para la creación activa de empleo para la ciudad de Cali – Colombia	Julio de 2009
2009-002	Juan Esteban Carranza Romero Carlos Giovanni González	Consideraciones casi obvias sobre la tasa de cambio en Colombia	Diciembre de 2009
2010-001	Jaime Andrés Collazos Pedro Luis Rosero	¿Posee el Valle Del Cauca una economía trasformadora de importaciones orientadas a la exportación?	Marzo de 2010
2010-002	Juan Esteban Carranza Romero Carlos Giovanni González Ximena Dueñas Herrera	Lo dicen los datos: La violencia homicida en Colombia es un resultado del ciclo económico	Abril de 2010
2010-003	Pedro Pablo Sanabria Natalia Solano Juan Sebastián Corrales M.	Seguimiento a las Finanzas Públicas de Cali: 2007-2008	Abril de 2010

2010-004	María del Pilar López R.	Valle del Cauca: Gran Subsidiador del Gasto Público Territorial en Colombia. 2002-2008	Junio de 2010
2010-005	Pedro Pablo Sanabria	¿Quiénes dirigen a los vallecaucanos? Caracterización de los altos funcionarios públicos del Valle del Cauca y Cali – 2008	Noviembre de 2010
2011-001	Carlos Giovanni González Espitia	Econometría para la Evaluación de Políticas Públicas con Stata: Introducción y Análisis de Datos	Enero de 2011

# POLIS

[www.icesi.edu.co/polis](http://www.icesi.edu.co/polis)

## ¿Qué es **POLIS**?

Una unidad académica y de coordinación de la Universidad Icesi que tiene por objeto hacerle seguimiento y evaluación a hechos y decisiones de carácter político y a políticas públicas de interés general o consideradas estratégicas para el desarrollo del Valle del Cauca.



UNIVERSIDAD  
**ICESI**

**Teléfono:** 555 2334 **Ext.:** 400 | **Fax:** (572) 555 1706

Calle 18 No. 122 - 135 Cali - Colombia

Correo electrónico: [polis@icesi.edu.co](mailto:polis@icesi.edu.co)

[www.icesi.edu.co/polis](http://www.icesi.edu.co/polis)

[www.icesi.edu.co/polis](http://www.icesi.edu.co/polis)