



Introducción a Series de Tiempo Univariadas

Usando STATA

Javiera Vásquez
12/31/2010

Este curso tiene por objetivo lograr un análisis estadístico y econométrico de series de tiempo univariadas, determinar el proceso estadístico que sigue una serie de tiempo, y a partir de la estimación del modelo realizar pronósticos que sean relevante para tomar decisiones de negocios, política, etc.

Primero se hará un breve repaso del software STATA, los conceptos y funciones básicas para el desarrollo correcto de este curso.

0. Introducción STATA

STATA es una aplicación completa e integrada, basada en comandos, que tiene todos los elementos necesarios para realizar análisis estadístico, manejo de datos estadísticos y gráficos. Las versiones más nuevas de STATA (a partir de la versión 8.0) posee una forma más fácil de utilizar, que consiste simplemente en hacer clic en ventanas con las opciones de análisis y procesamiento de datos, además tiene la opción “antigua” mediante los comandos. El programa posee una ayuda en línea, es un programa fácil y rápido de utilizar.

¿Cómo se ve STATA?

Cuando abrimos el programa, inmediatamente podemos distinguir 4 ventanas:

- Review: en esta ventana aparecen los comandos que han sido utilizados durante la sección en turno.
- Results: muestra los resultados de la aplicación de los comandos, sólo los resultados más recientes son visibles en esta ventana
- Variables: en esta ventana se presenta el listado de variables que se encuentran en la base de datos que se está trabajando
- Commands: corresponde a la ventana donde introducen los comandos para obtener el resultado deseado. Sirve para utilizar STATA en forma interactiva.

Los íconos de la parte superior tienen los siguientes usos:



Abrir una base de datos



Guardar una base de datos, una vez que ha sido modificada en el programa




Imprimir los resultados de la ventana de resultados (STATA Results)



Comenzar o abrir un archivo **log**. Estos archivos tienen un formato de texto y permiten ir guardando todos los resultados.



Abrir el editor de do-file. Los archivos **do** son archivos con esta extensión que nos permiten en forma ordenada escribir todo lo que queremos hacer en nuestra base de datos: cambiar la base

de datos, sacar estadísticas, etc..., y luego presionando  correr dicho do y obtener los resultados.¹



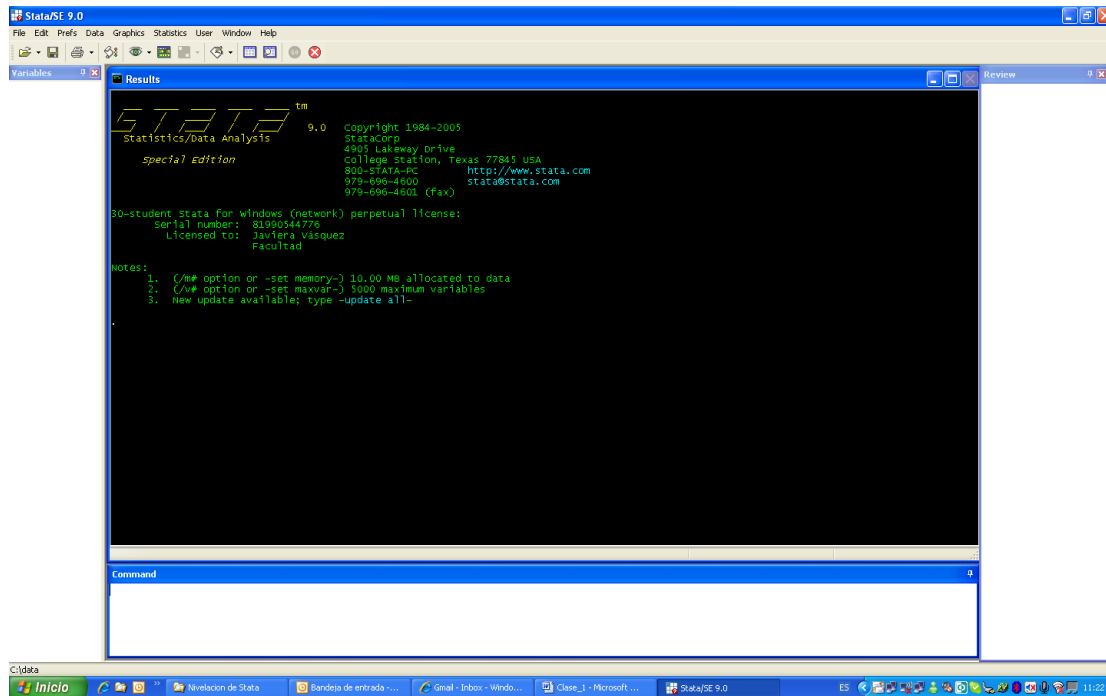
Permite ver y editar la Base de Datos.



Es igual al EDITOR, pero no permite eliminar variables ni observaciones.



Es para detener la ejecución de un comando.



0.1 Como organizar el trabajo en STATA

Cuando se abre STATA es importante saber donde se esta trabajando, es decir, en que carpeta se están guardando los resultados o desde que carpeta vamos a llamar la base de datos, etc. Si no se sabe la carpeta o directorio donde STATA esta ubicado podemos averiguarlo escribiendo el comando **pwd**:

pwd

C:\data → Este resultado nos indica que estamos ubicados en el disco C del computador en la carpeta data

¹ Lo ideal es combinar la utilización de un **do** y un **log**; el primero permite tener en forma ordenada todos los comandos que se están utilizando y todas las instrucciones que se quieren ejecutar, mientras que el segundo guarda en un archivo de texto todos los resultados que surgen de este archivo do.

Para cambiar el directorio o carpeta se debe realizar lo siguiente:

```
cd C:\Nivelacion_Stata
```

Utilizo el comando **cd** y entrego la nueva ruta. En este caso le estoy indicando al programa que se ubique en la carpeta “Nivelacion_Stata” que se encuentra en el disco C del computador.

La ventaja de indicar desde un comienzo en que carpeta del computador se esta trabajando, es que evita indicar la ruta completa de los archivos cada vez que queramos abrir o guardar una base de datos, o abrir o guardar un log. Obviamente esto tiene sentido cuando para un trabajo específico tenemos todos los archivos necesarios en la misma carpeta.

Por ejemplo, si estamos trabajando con información de tres bases de datos distintas, y queremos dejar la información relevante para el estudio en una sola base datos (más adelante veremos como hacer esto), lo ideal es trabajar en una sola carpeta, “Nivelacion_Stata”, y no tener las tres bases de datos repartidas en carpetas distintas. Si no están en la misma carpeta no es útil indicarle el directorio al comienzo, ya que igual cuando llamemos a cada una de las bases de datos, al estar en carpetas distintas, tendremos que cambiar la ruta.

Importante: los sistemas operativos más nuevos permiten que las carpetas tengan nombres con espacio en blanco, por ejemplo, “Nivelacion Stata”. Sin embargo, STATA no va a reconocer una carpeta que tenga **espacios en blanco** en el nombre, a no ser que se indique la ubicación de esta carpeta entre comillas. Por este motivo, se debe **evitar** llamar a una carpeta con la que van a trabajar en STATA con nombres que contengan espacios en blanco.²

Supongamos que la carpeta en que vamos a tratar se llama “Nivelacion Stata”, en la primera línea del siguiente cuadro podemos apreciar que al entregar la ubicación de la carpeta utilizando el comando **cd**, el programa nos entrega un error “invalid syntax”, esto se debe a que el nombre de la carpeta tiene espacios en blanco. Si agregamos comillas a la ruta no se produce el error.

```
. cd c:\Nivelacion Stata
invalid syntax
r(198);

. cd "C:\Nivelacion Stata"
C:\Nivelacion Stata
.
```

² Este problema es común cuando trabajan en el Escritorio del computador, ya que la carpeta en este caso es C:\Documents and Settings\...., tiene espacios en blanco.

En resumen, para trabajar ordenadamente en STATA es conveniente crear una carpeta para cada trabajo independiente, esta carpeta debe tener una ruta que no contenga espacios en blanco en los nombres.

0.2 Cargar una base de datos en STATA

Las bases de datos en formato Stata tienen extensión `.dta`. Las versiones antiguas del software no se pueden abrir bases de datos que han sido trabajadas y guardadas en una versión más moderna, cuando intentemos hacer esto el programa entregará un error indicando que la base no tiene formato Stata.

Antes de abrir una base de datos se tienen que cumplir dos condiciones:

- 1- El programa debe estar limpio, sin ninguna base de datos ya cargada. Para limpiar el programa de otras bases de datos se debe utilizar el comando **clear**. Si he estado trabajando una base de datos previamente la cual se ha modificado y no he guardado estas modificaciones, al intentar abrir una nueva base de datos sin limpiar antes arrojará el siguiente error:

```
no; data in memory would be lost
```

- 2- El programa debe tener suficiente memoria. Para entregarle memoria a Stata se debe utilizar el comando **set mem**. Por ejemplo, si la base de datos que deseamos cargar pesa 100 MB, en la ventana **Stata Command** debemos tipear:

```
set mem 100m
```

Si Ud. no agrega memoria y los 10 MB que vienen asignados al abrir el programa no son suficientes, el programa arrojará el siguiente error:

```
no room to add more observations
```


Esto también puede suceder cuando se ha trabajado en la base de datos y se han creado muchas variables: en un momento el programa se puede quedar sin memoria. En este caso se debe limpiar el programa (borrar la base de datos) utilizando el comando **clear**; entregarle más memoria al programa utilizando **set mem**; abrir la base de datos y realizar todo nuevamente. Por esta razón es fundamental que Ud., cuando comience a trabajar, asigne la memoria necesaria para todas las variables que espera generar.

El comando general para entregar memoria a Stata es:

```
set mem #[b|k|m|g] [, permanently]
```

con la opción “permanently” la cantidad de memoria ingresada se mantendrá cada vez que se inicie nuevamente el programa.

Existen distintas formas de cargar una base de datos:

1- Utilizando una base ya grabada con la extensión de STATA, es decir, disponer de la base de datos como **nombre.dta**. En este caso podemos apretar el icono  y buscar la ubicación de la base de datos. También podemos hacerlo dirigiéndonos a File/Open...

2- Otra forma es tipear en **Stata Command** use “[disco en que la guardaremos] \ [ruta de acceso] \ [nombre de archivo.dta]”, clear. Por ejemplo:

```
use "C:\Nivelacion_Stata\ingreso.dta",
```

o simplemente

```
use ingreso.dta, clear
```

si ya le hemos indicado previamente a Stata que vamos a trabajar en la carpeta Nivelacion_Stata del disco C.

Notar que en ambos casos el comando incorpora la opción “, clear”, esto nos garantiza que la base de datos sea abra si es que ya existe otra base de datos previa en el programa, esta opción ahorra el paso previo de ejecutar el comando **clear** antes de abrir la base de datos.

Recuerde que si la carpeta en la que esta trabajando tiene espacios en blanco, debo poner comillas al llamar la base de datos, de lo contrario aparecerá el siguiente error:


```
. use C:\Nivelacion Stata\ingreso.dta, clear
invalid 'Stata'
r(198);
```

Esto porque Stata cree que el nombre de la carpeta es simplemente Nivelacion. Si utilizamos comillas no se produce el error.

Ahora si el nombre de la carpeta o el nombre de la base esta mal ingresado en el comando se produce el siguiente error:

```
. use "C:\NivelacionStata\ingreso.dta", clear  
file C:\NivelacionStata\ingreso.dta not found  
r(601);
```

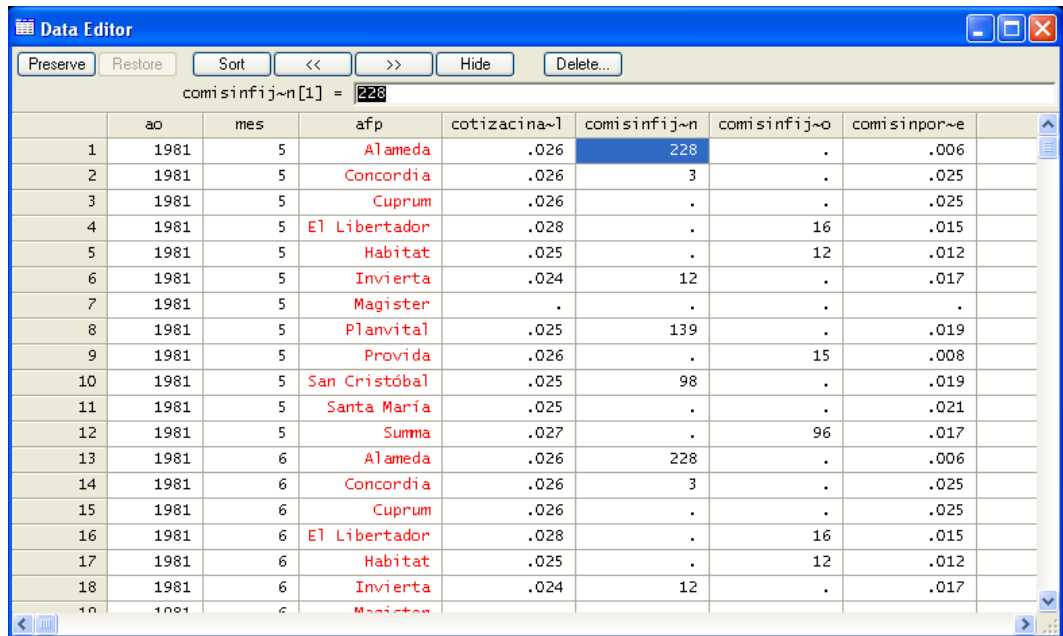
3- Traspasar los datos de un archivo Excel o similar copiando la información de este archivo al EDITOR de STATA.

Esto se hace copiando en el archivo Excel las columnas (variables) que queremos traspasar como base de datos a STATA (Ctrl+C). Luego nos dirigimos a STATA abrimos el EDITOR  y pegamos la información (Ctrl+V). Obviamente antes de hacer esto se debe haber limpiado Stata con el comando **clear**.

Algunos aspectos relevantes antes de copiar los datos de Excel a Stata:

- Para Stata, como para cualquier otro software norteamericano, el separador de miles es la coma (,), y el separador de decimales es el punto (.); Si el computador en el que esta trabajando no esta configurado de esta forma, debe dirigirse a inicio → Panel de Control → Opciones regionales, de idioma, y de fecha y hora → Configuración regional y de idioma → Opciones regionales, picar personalizar, aquí se puede cambiar la configuración numérica indicando que el símbolo decimal es "." Y el símbolo de separación de miles ",".
- Todas las variables que son numéricas, deben estar en formato numérico antes de ser exportadas.

El siguiente cuadro muestra lo que resulta de pasar la base de datos base.xls a Stata:



	ao	mes	afp	cotizacina~l	comisinfinj~n	comisinfinj~o	comisinpor~e
1	1981	5	Alameda	.026	228	.	.006
2	1981	5	Concordia	.026	3	.	.025
3	1981	5	Cuprum	.026	.	.	.025
4	1981	5	El Libertador	.028	.	16	.015
5	1981	5	Habitat	.025	.	12	.012
6	1981	5	Invierta	.024	12	.	.017
7	1981	5	Magister
8	1981	5	Planvital	.025	139	.	.019
9	1981	5	Provida	.026	.	15	.008
10	1981	5	San Cristóbal	.025	98	.	.019
11	1981	5	Santa María	.025	.	.	.021
12	1981	5	Summa	.027	.	96	.017
13	1981	6	Alameda	.026	228	.	.006
14	1981	6	Concordia	.026	3	.	.025
15	1981	6	Cuprum	.026	.	.	.025
16	1981	6	El Libertador	.028	.	16	.015
17	1981	6	Habitat	.025	.	12	.012
18	1981	6	Invierta	.024	12	.	.017

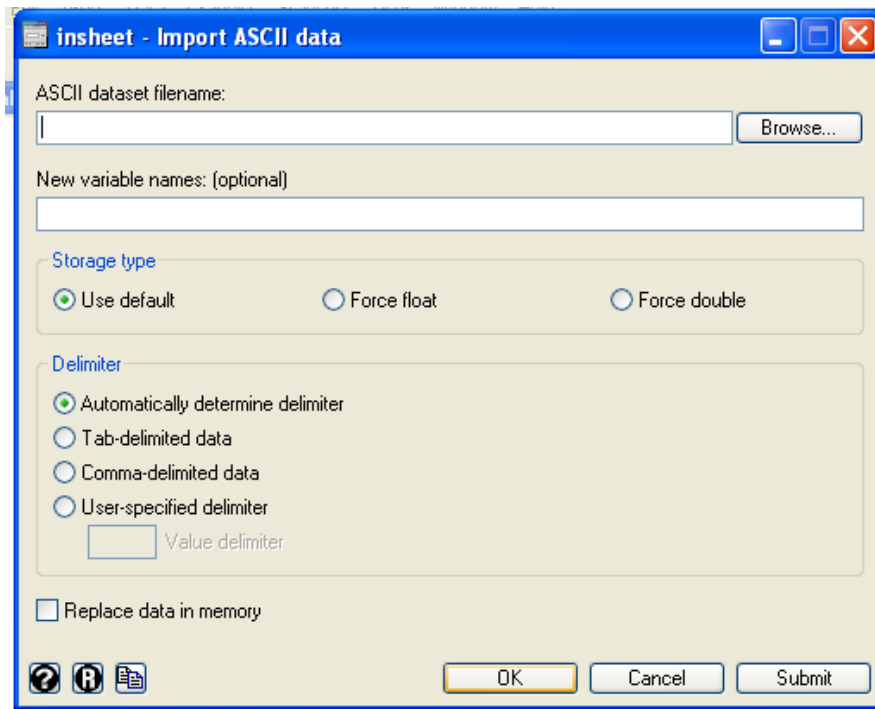
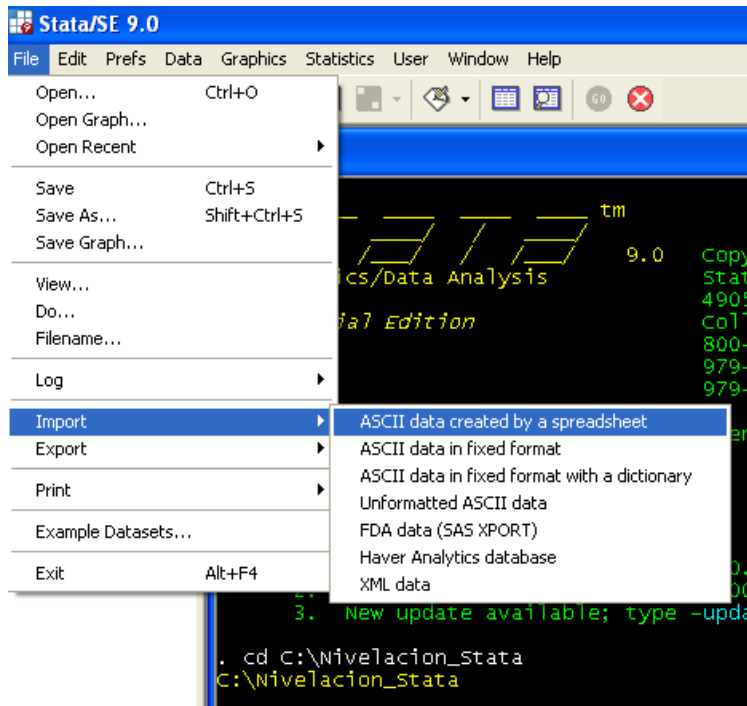
El color rojo indica que la variable no es numérica.

Generalmente las bases de datos muy grandes no vienen en excel, ya que este programa es limitado en cuanto al número de filas (observaciones) y número de columnas (variables). El número máximo de filas es de 65.536, y el número máximo de columnas es de 256.

4- Otra forma de cargar bases de datos es mediante el comando **insheet**, este comando permite cargar bases de datos en formato ASCII (texto) mediante el siguiente comando:

```
insheet using C:\Nivelacion_Stata\junio05.txt
```

o alternativamente:



Cuando las bases de datos vienen el texto y son muy grandes no se pueden ver utilizando un block de notas, en estos casos se recomienda utilizar el programa TextPad que puede ser descargado gratuitamente (www.textpad.com). Siempre es recomendable inspeccionar la base de datos en texto antes de ser traspasada a Stata.

5- Si la base de datos tiene otro formato, por ejemplo, SPSS (.sav), dbase (.dbf), Access (.mdb), etc; existe un software llamado **Stat Transfer**, que permite transformar base de datos desde y a diversos formatos.

Luego para guardar la base de datos utilizamos el comando **save**:

1- Si quiere reescribir la base de datos antigua:

```
save C:\Nivelacion_Stata\ingresos.dta, replace
```

Es importante escribir **replace**, sino el programa les enviara un error diciendo que la base de datos ya existe.

2- Si quiere guardar la base de datos con un nuevo nombre no es necesario tipear replace:

```
save C:\Nivelacion_Stata\ingresos_new.dta
```

Una vez que los datos han sido cargados, se puede optimizar el espacio que estos ocupan utilizando el comando **compress**, este comando comprime la base de datos. Es muy útil cuando trabajamos con bases de datos grandes.

Hasta ahora hemos aprendido como cargar una base de datos en Stata, en lo que sigue se verán los comandos básicos para analizar una base de datos.

Entonces, con los comandos recién estudiados, comencemos por abrir la base de datos:


```
cd C:\Nivelacion_Stata  
set mem 100m  
use ingreso.dta, clear
```

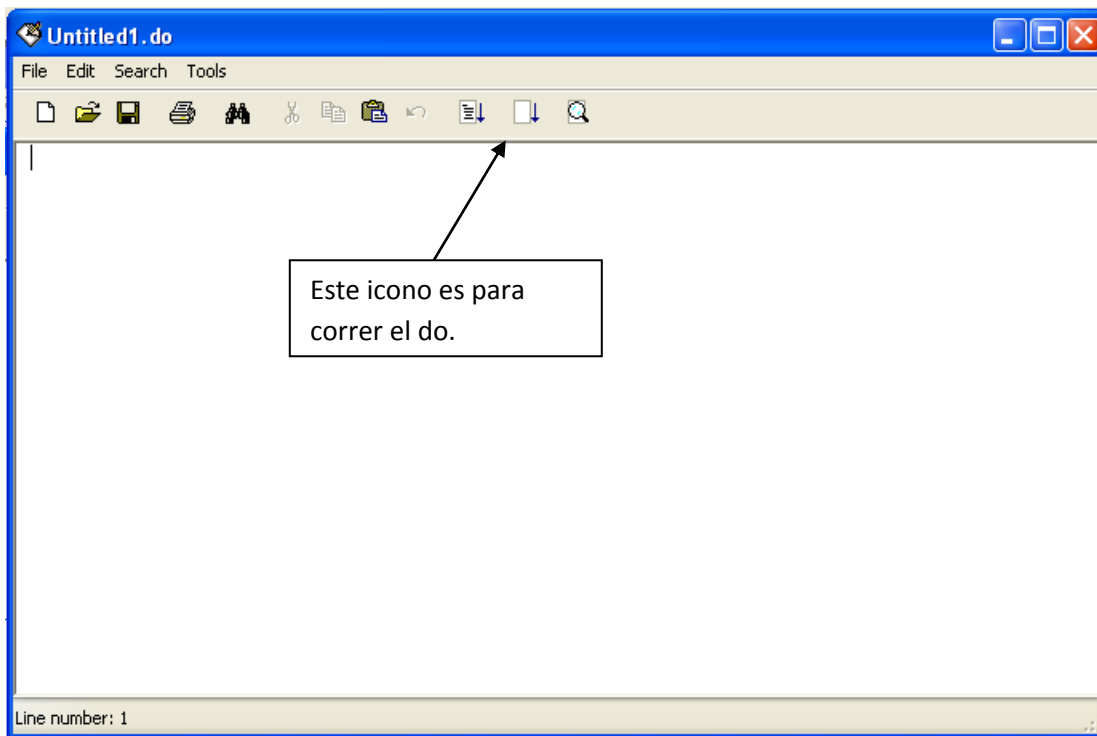
0.3 Trabajar de manera ordenada en STATA: do-file

Como se ha mencionado en clases anteriores, existen dos formas de trabajar en STATA, en forma interactiva y en forma programada. La primera forma consiste en ir ejecutando los comandos directamente en la ventana de comando, los resultados se obtienen inmediatamente en la ventana de resultados. Al trabajar de esta forma, la única manera de ir registrando todo lo realizado es mediante los archivos log. Sin embargo, esta forma de trabajar tiene la desventaja de que una vez que uno ha realizado varias modificaciones a la base de datos y uno quiere volver atrás, se pierde todo lo realizado y hay que volver a reconstruir todo con ayuda del log.

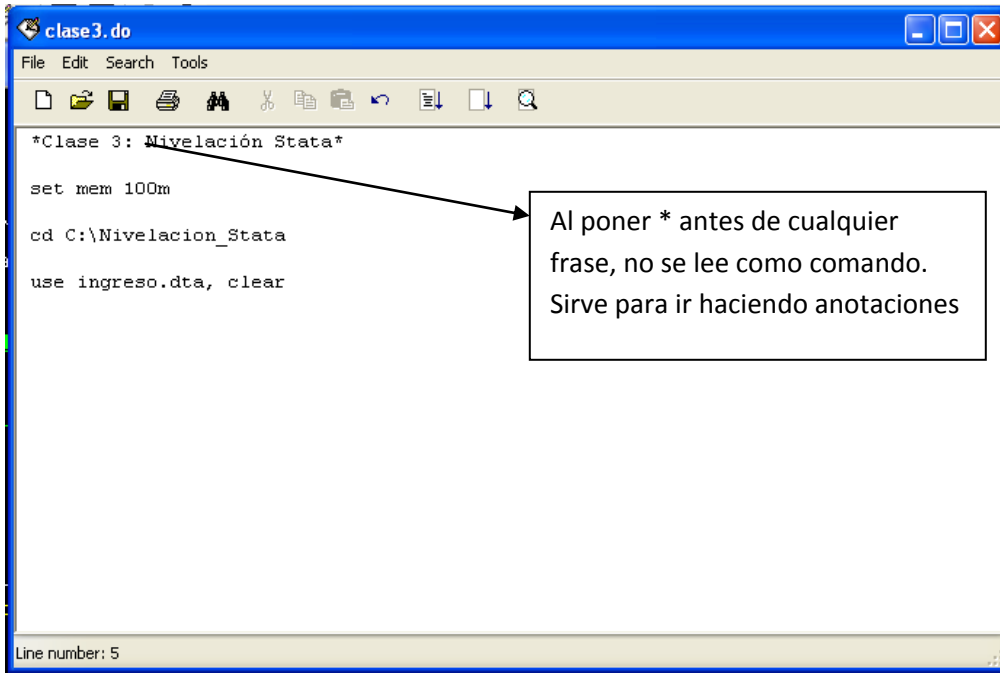
La manera más ordenada de trabajar en STATA cuando se requiere hacer varias modificaciones a la base de datos y obtener varias estadísticas de ella, es programar todos los comandos en un archivo do.

El archivo **do** no es mas que un archivo de texto que permite escribir las instrucciones para la ejecución de comandos en Stata.

Para abrir el archivo **do** debemos pinchar el icono , y se abrirá la siguiente ventana:



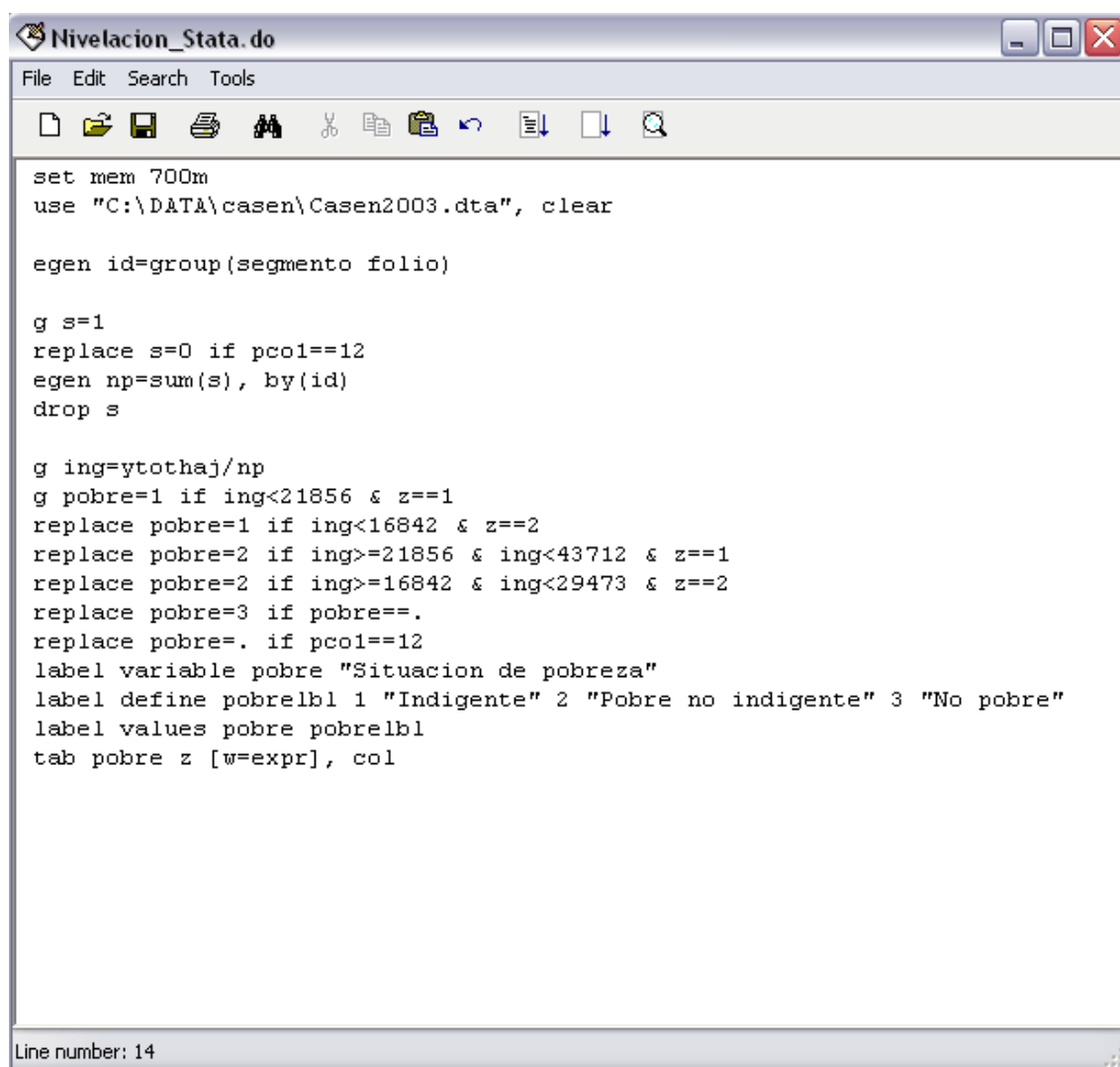
Por ejemplo, la forma típica de comenzar un **do** sería la siguiente:



Con esto ya he abierto la base de datos. A continuación puedo empezar a escribir los comandos para transformar la base de datos, para obtener estadísticas, etc. Exactamente de la misma forma que lo haría en la ventana de comandos pero ahora en forma más ordenada.

Es importante constantemente ir corriendo el **do** para detectar los errores que se están cometiendo.

En el cuadro siguiente observamos el archivo do file correspondiente al ejemplo visto en la sección 1;



```
set mem 700m
use "C:\DATA\casen\Casen2003.dta", clear

egen id=group(segmento folio)

g s=1
replace s=0 if pcol==12
egen np=sum(s), by(id)
drop s

g ing=ytotahj/np
g pobre=1 if ing<21856 & z==1
replace pobre=1 if ing<16842 & z==2
replace pobre=2 if ing>=21856 & ing<43712 & z==1
replace pobre=2 if ing>=16842 & ing<29473 & z==2
replace pobre=3 if pobre==.
replace pobre=. if pcol==12
label variable pobre "Situacion de pobreza"
label define pobrelbl 1 "Indigente" 2 "Pobre no indigente" 3 "No pobre"
label values pobre pobrelbl
tab pobre z [w=expr], col
```

Line number: 14

I. Formato de tiempo en STATA

Por definición los datos tienen una frecuencia temporal, la que puede ser mensual, trimestral, anual, etc. Lo primero que debemos hacer es indicarle a STATA que estaremos trabajando en formato de serie de tiempo lo que se hace a través del comando `tsset`.

Sin embargo, previo a esto debemos tener en nuestra base de datos una variable que indique la temporalidad o frecuencia de los datos. Por ejemplo, en la base de datos `imacec.dta` podemos ver las primeras 10 observaciones:

```
cd "C:\Javiera\Cursos_Externos\SUBPESCA"
use "bases\imacec.dta", clear
list if _n<=10
```

	año	mes	imacec
1.	1986	1	36.1797
2.	1986	2	33.6536
3.	1986	3	38.755
4.	1986	4	38.8662
5.	1986	5	38.2517
6.	1986	6	37.6241
7.	1986	7	36.9222
8.	1986	8	36.0124
9.	1986	9	35.9231
10.	1986	10	40.0556

En este caso no disponemos de una única variable que identifique que los datos tienen frecuencia mensual, por lo cual lo primero que debemos hacer es generar esta variable.

```
g fecha=ym(año,mes)
```

```
list if _n<=10
```

	año	mes	imacec	fecha
1.	1986	1	36.1797	312
2.	1986	2	33.6536	313
3.	1986	3	38.755	314
4.	1986	4	38.8662	315
5.	1986	5	38.2517	316
6.	1986	6	37.6241	317
7.	1986	7	36.9222	318
8.	1986	8	36.0124	319
9.	1986	9	35.9231	320
10.	1986	10	40.0556	321

Mediante la función `ym(.,.)` hemos generado una nueva variable llamada `fecha` que tiene un código que STATA entiende que el número 317 significa Junio de 1986. Si en vez de que me muestre esos números prefiero que me muestre la fecha a la cual corresponde, debo ejecutar el siguiente comando:

```
format fecha %tm
```

```
list if _n<=10
```

	año	mes	imacec	fecha
1.	1986	1	36.1797	1986m1
2.	1986	2	33.6536	1986m2
3.	1986	3	38.755	1986m3
4.	1986	4	38.8662	1986m4
5.	1986	5	38.2517	1986m5
6.	1986	6	37.6241	1986m6
7.	1986	7	36.9222	1986m7
8.	1986	8	36.0124	1986m8
9.	1986	9	35.9231	1986m9
10.	1986	10	40.0556	1986m10

Para datos con otras frecuencias se deben utilizar los siguientes comandos:

❖ **Diaria**

```
g fecha=mdy(mes,dia,año)
format fecha %td
```

❖ **Semanal**

```
g fecha=yw(año,semana)
format fecha %tw
```

❖ **Mensual**

```
g fecha=ym(año,mes)
format fecha %tm
```

❖ **Trimestral**

```
g fecha=yq(año,trimestre)
format fecha %tq
```

❖ **Semestral**

```
g fecha=yh(año,semestre)
format fecha %th
```

Entonces, una vez creada una variable única que contenga la frecuencia de los datos, en nuestro ejemplo la variable `fecha`, debemos indicarle a STATA que trabajaremos con formato de datos en series de tiempo:

```
tsset fecha
      time variable:  fecha, 1986m1 to 2010m12
            delta:    1 month
```

II. Operadores de series de tiempo

Debido a que las series tiempo tienen por naturaleza un orden temporal, con frecuencia no sólo nos interesa o queremos hacer referencia al valor de la serie en el momento t , sino por ejemplo al valor rezagado de la serie ($t-1$), o la serie en diferencia (valor en t menos el valor en $t-1$), etc. STATA posee operadores de series de tiempo que nos ayudan a obtener dichos valores de manera mucho más fácil que crearlos de manera manual.

II.1 Operador de rezagos

En series de tiempo se define el operador de rezago L tal que:

$$Lx_t = x_{t-1}$$

$$L^2x_t = L(Lx_t) = x_{t-2}$$

Por ejemplo, si queremos crear una variable que contenga el primer rezago de la variable IMACEC:

```
g imaceclag1=L.imacec
list fecha imacec imaceclag1 if _n<=10
```

	fecha	imacec	imacec~1
1.	1986m1	36.1797	.
2.	1986m2	33.6536	36.1797
3.	1986m3	38.755	33.65361
4.	1986m4	38.8662	38.75497
5.	1986m5	38.2517	38.86623
6.	1986m6	37.6241	38.25167
7.	1986m7	36.9222	37.62409
8.	1986m8	36.0124	36.92221
9.	1986m9	35.9231	36.01239
10.	1986m10	40.0556	35.9231

De manera análoga, podemos generar una variable con el segundo rezago de la variable IMACEC a través del siguiente comando:

```
g imaceclag2=L2.imacec
```

II.2 Operador forward

También podemos ocupar el operador **F** para adelantar datos, es decir, generar una variable con las observaciones en $t+1$:

```
g imacecforward1=F.imacec
list fecha imacec imacecforward1 if _n<=10
```

	fecha	imacec	imace~d1
1.	1986m1	36.1797	33.65361
2.	1986m2	33.6536	38.75497
3.	1986m3	38.755	38.86623
4.	1986m4	38.8662	38.25167
5.	1986m5	38.2517	37.62409
6.	1986m6	37.6241	36.92221
7.	1986m7	36.9222	36.01239
8.	1986m8	36.0124	35.9231
9.	1986m9	35.9231	40.05564
10.	1986m10	40.0556	38.80005

II.3 Operador de diferencias:

A menudo más que estar interesados en el valor de la serie en cada instante t , podemos estar interesados en los cambios de la serie en el tiempo. Por ejemplo, usualmente no nos interesa el valor en un mes puntual del IMACEC, sino como ha cambiado con respecto al mes anterior.

El operador de diferencias Δ se define de la siguiente manera:

$$\Delta x_t = x_t - x_{t-1}$$

$$\Delta^2 x_t = \Delta \Delta x_t = \Delta(x_t - x_{t-1}) = x_{t-1} - x_{t-2}$$

Mediante el siguiente comando generamos la primera diferencia de la serie IMACEC:

```
g difimacec=D.imacec
list fecha imacec difimacec if _n<=10
```

	fecha	imacec	difimacec
1.	1986m1	36.1797	.
2.	1986m2	33.6536	-2.526096
3.	1986m3	38.755	5.101368
4.	1986m4	38.8662	.1112518
5.	1986m5	38.2517	-.6145592
6.	1986m6	37.6241	-.6275787
7.	1986m7	36.9222	-.7018814
8.	1986m8	36.0124	-.9098206
9.	1986m9	35.9231	-.0892906
10.	1986m10	40.0556	4.132545

II.4 Operador de diferencias estacional

Tomando nuevamente el ejemplo del IMACEC, las variaciones relevantes para calcular el crecimiento de este índice son las variaciones en 12 meses, ya que estas eliminan los efectos estacionales propios de la actividad económica.

El operador de diferencias estacionales Δ_s se define de la siguiente manera:

$$\Delta_s x = x_t - x_{t-s}$$

De la siguiente manera creamos la variación en 12 meses del imacec:

```
g imacec12=S12.imacec
list fecha imacec imacec12 if _n>=13 & _n<=20
```

	fecha	imacec	imacec12
13.	1987m1	39.1044	2.924744
14.	1987m2	37.3693	3.715729
15.	1987m3	43.2659	4.510948
16.	1987m4	41.0376	2.171413
17.	1987m5	40.9902	2.738529
18.	1987m6	40.6597	3.035564
19.	1987m7	38.9137	1.991463
20.	1987m8	37.7142	1.701832

II.5 Cambios porcentuales

Muchas variables económicas se trabajan, piensan o presentan en cambios porcentuales, es decir:

$$\Delta\%x = 100 \cdot \frac{\Delta x}{x_{t-1}} = 100 \cdot \frac{x_t - x_{t-1}}{x_{t-1}}$$

Mediante los siguientes comandos podemos crear la variación porcentual mensual del IMACEC, y la variación porcentual en 12 meses:

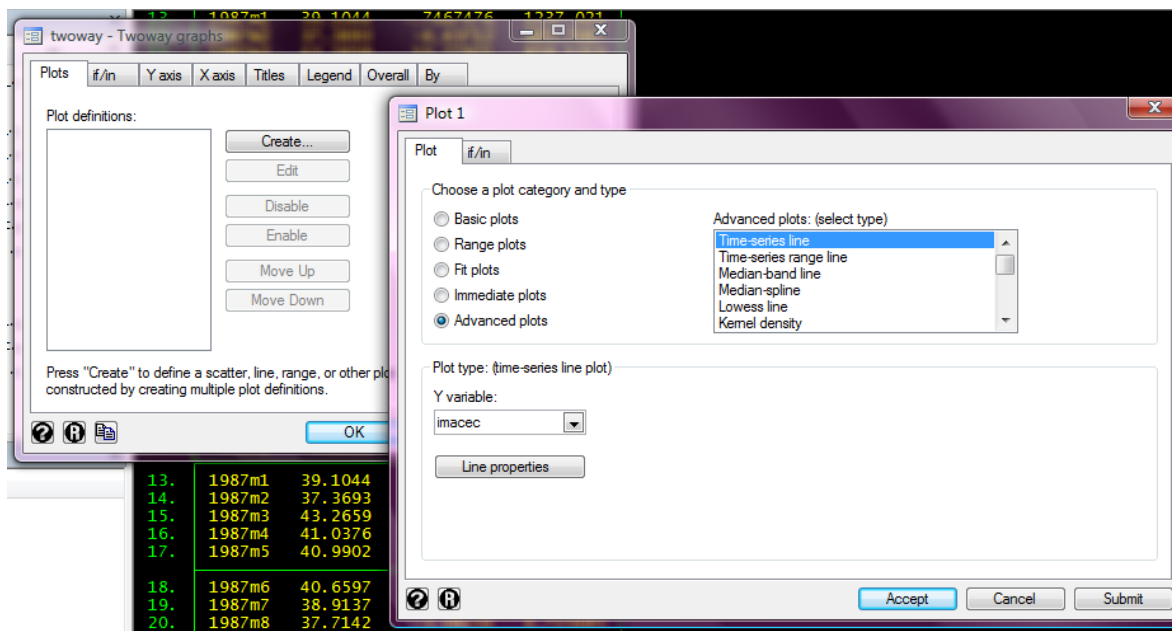
```
g porctimacec=100*(imacec-L.imacec)/L.imacec
g porctimacec12=100*(imacec-L12.imacec)/L12.imacec
list fecha imacec porctimacec porctimacec12 if _n>=13 &
_n<=20
```

	fecha	imacec	porctim~c	porct~12
13.	1987m1	39.1044	-.7467476	8.083935
14.	1987m2	37.3693	-4.43712	11.0411
15.	1987m3	43.2659	15.77921	11.63966
16.	1987m4	41.0376	-5.150203	5.58689
17.	1987m5	40.9902	-.1156095	7.159242
18.	1987m6	40.6597	-.8063965	8.068141
19.	1987m7	38.9137	-4.294142	5.393672
20.	1987m8	37.7142	-3.08234	4.725685

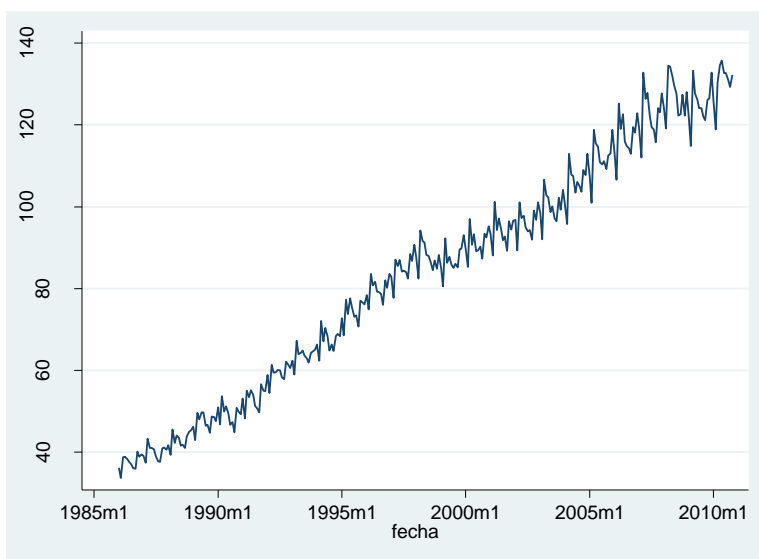
III. Gráficos

Siempre es útil comenzar un análisis estadístico de los datos con una inspección gráfica de los mismos, especialmente cuando trabajamos con series de tiempo.

Por ejemplo, podemos ver la evolución en el tiempo del IMACEC:

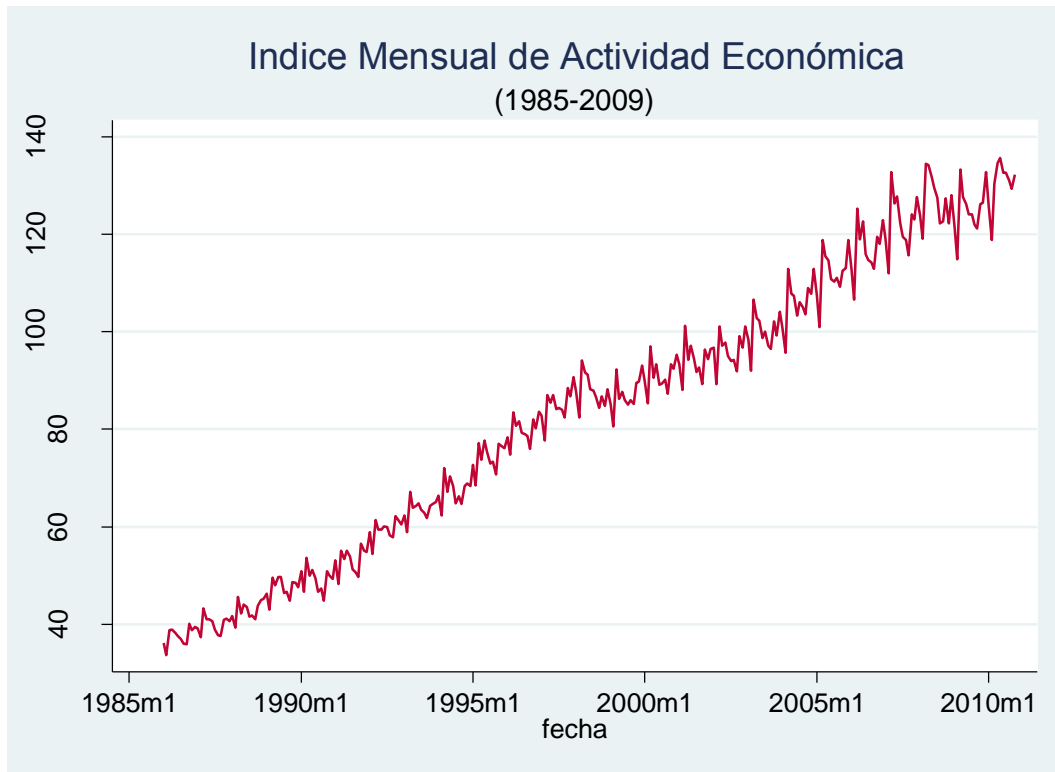


```
twoway (tsline imacec)
```



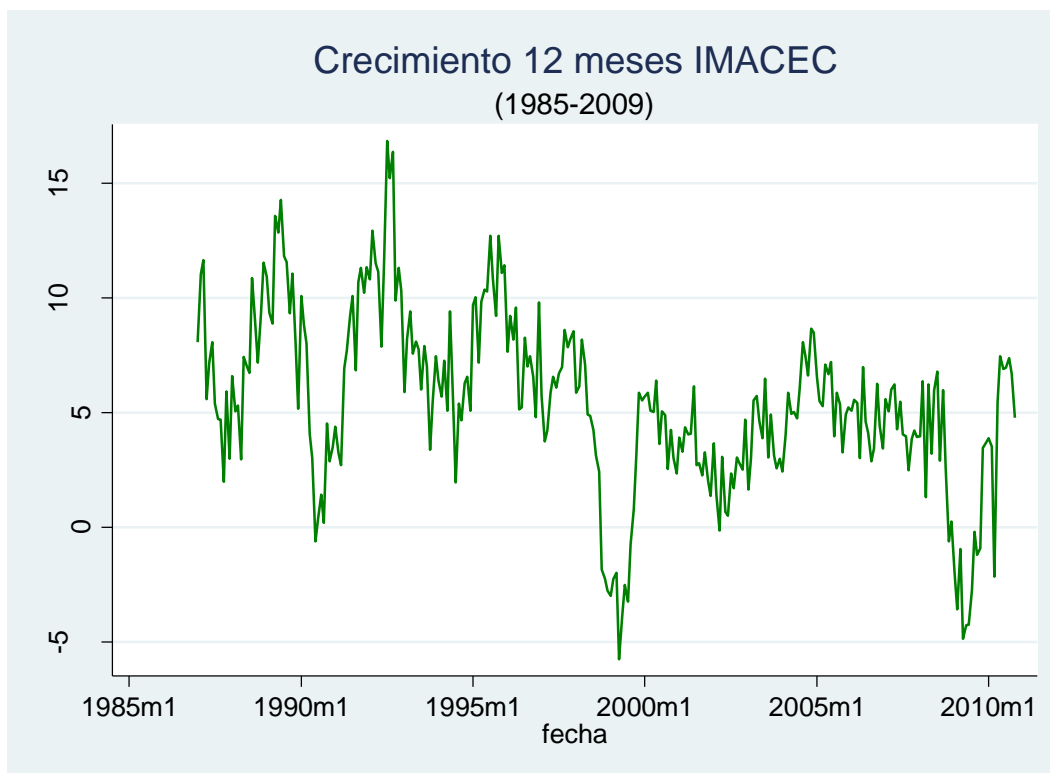
Podemos modificar los colores de la línea, títulos, etc:

```
twoway (tsline imacec, lcolor(cranberry)), title(Indice Mensual de Actividad  
Económica) subtitle((1985-2009))
```



También podemos graficar el crecimiento en 12 meses del IMACEC en color verde mediante el siguiente comando:

```
twoway (tsline porctimacec12, lcolor(green)), title(Indice Mensual de  
Actividad Económica) subtitle((1985-2009))
```

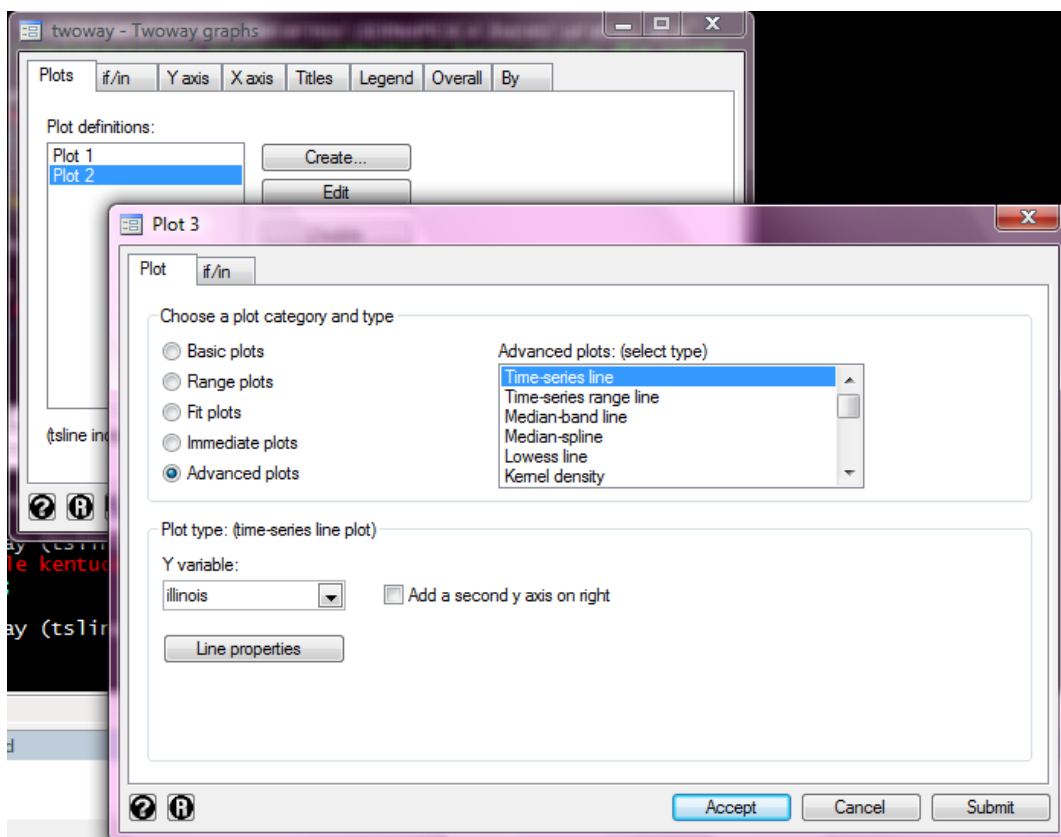


Por otra parte podemos graficar varias series en un mismo gráfico. Por ejemplo, la base de datos `urates.dta` tiene datos de tasas de desempleo para diferentes estados de Estados Unidos.

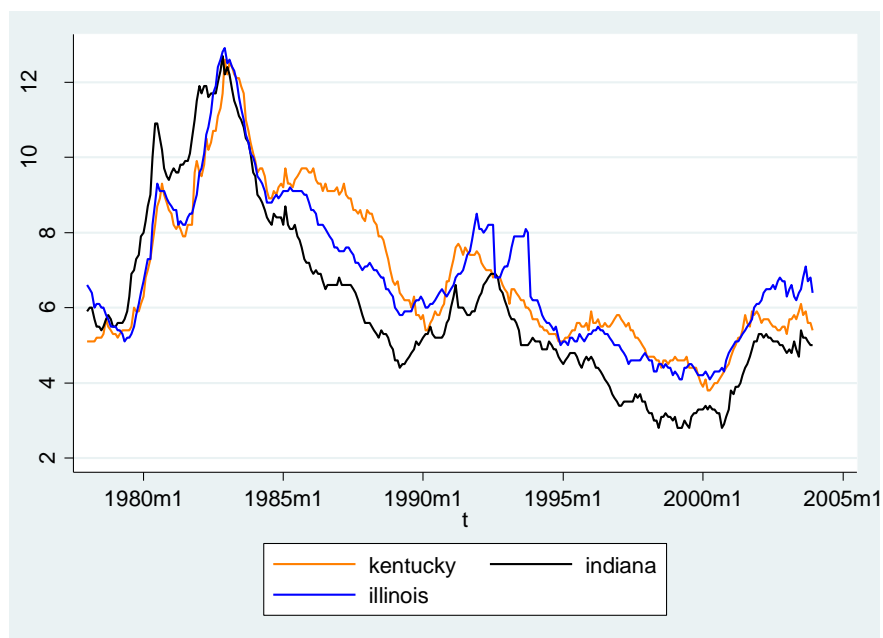
```
use bases/urates.dta, clear

tsset t
    time variable: t, 1978m1 to 2003m12
        delta: 1 month
```

Dentro de la misma ventana de gráfico `twoway` voy agregando más gráficos (`plot`):

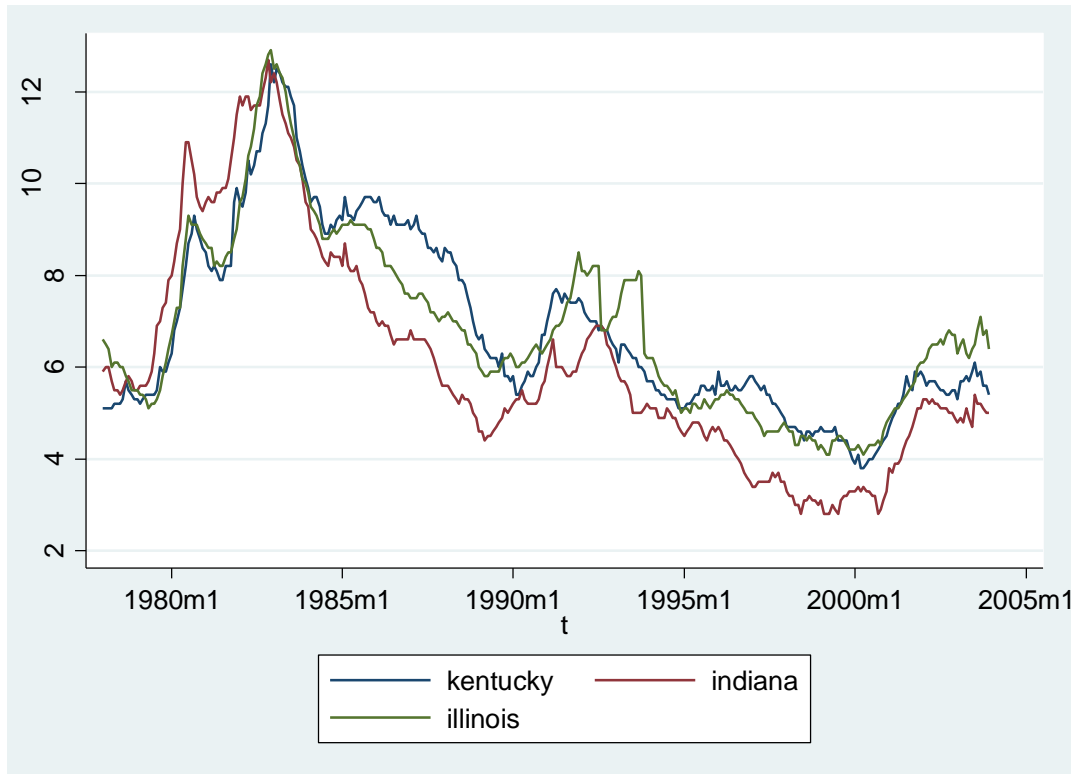


```
twoway (tsline kentucky, lcolor(orange)) (tsline indiana, lcolor(black))
      (tsline illinois, lcolor(blue))
```



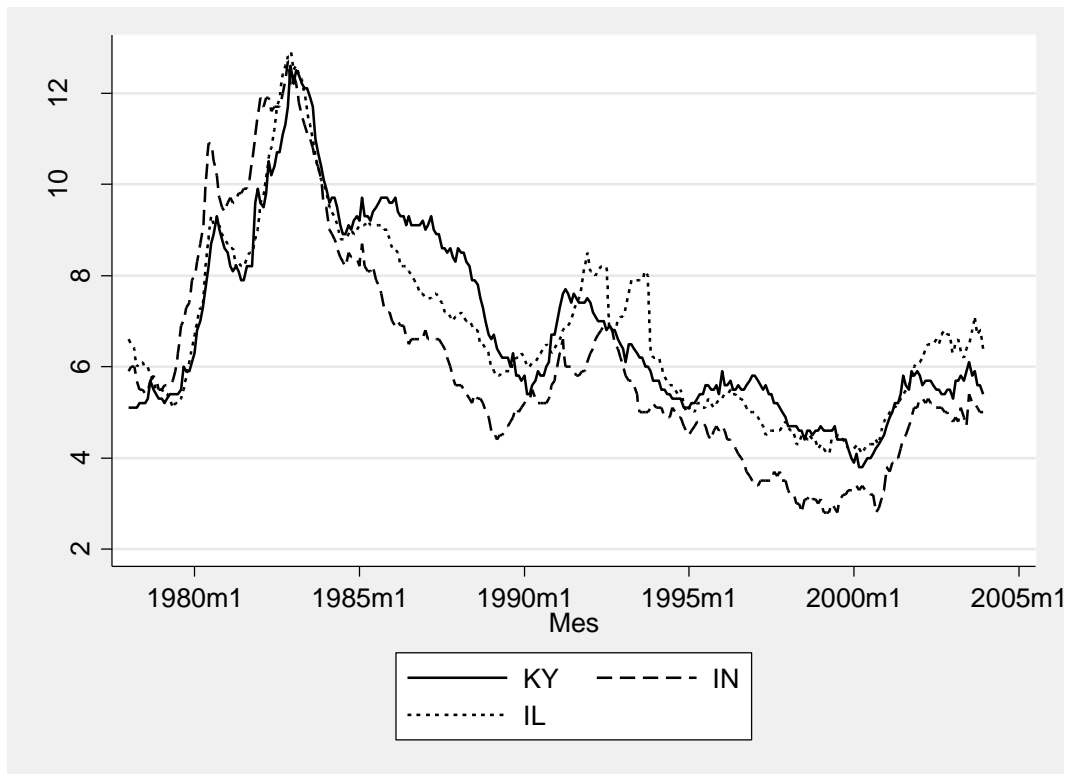
Estos mismos gráficos se pueden realizar simplemente con el comando `tsline`:

```
tsline kentucky indiana illinois
```



Podemos cambiar la leyenda de las series, y podemos pedir que no ocupe colores en las series sino diferentes tonalidades de negro, esto último es útil cuando los resultados deben ser presentados en blanco y negro. También notemos que al ser tres series no pone nombre al eje y, en el siguiente gráfico le pondremos nombre a ambos ejes.

```
tsline kentucky indiana illinois, legend(label(1 "KY") label(2 "IN") label(3  
"IL")) xtitle(Mes) ytitle(Tasa de desempleo) scheme(s2mono)
```

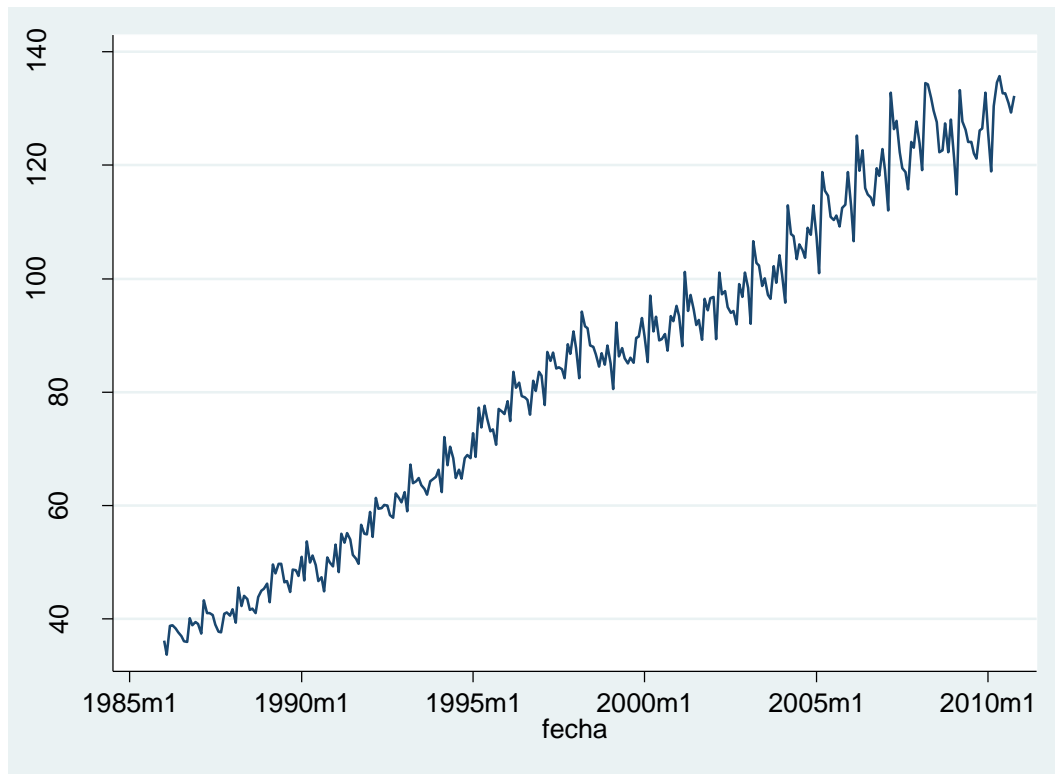
IV. Suavización y técnicas de predicción

Cuando analizamos series de tiempo, normalmente nos interesará eliminar ruidos o componentes idiosincráticos, de manera que las características de los datos sean más visibles.

IV.1 Cuatro componentes de una serie de tiempo

Recordemos el gráfico del IMACEC:

```
use bases/imacec  
  
g fecha=ym(año,mes)  
  
format fecha %tm  
  
tsset fecha  
    time variable: fecha, 1986m1 to 2010m12  
        delta: 1 month  
  
tsline imacec
```



¿Qué podemos observar en el comportamiento de esta serie?

Primero notamos una clara tendencia positiva en este índice, segundo podemos ver que este va creciendo pero con un comportamiento zig-zag indicando que dentro de un año hay meses en que este índice es mayor y otros meses en que es menor. Finalmente, después del año 1998 se observa una caída en este índice, su valor pareciera estar por debajo de lo que se esperaría para estos años.

Este ejemplo nos muestra que la mayoría de las series de tiempo poseen cuatro componentes:

- ❖ Tendencia: tendencia a gran escala a la cual la serie cambia en el tiempo. Series con una tendencia positiva, y con una tendencia negativa la serie va decreciendo en el tiempo.
- ❖ Componente estacional: representa la tendencia con la cual la serie varía dentro de un año, por clima, vacaciones, etc.
- ❖ Componente cíclico: se refiere a las fluctuaciones asociadas al comportamiento agregado de la economía o ciclos económicos.
- ❖ Componente idiosincrático: son todos los factores distintos a la tendencia, estacionalidad, y ciclos. Es como el término de error en una regresión lineal.

IV.2 Medias Móviles

En los análisis de corte transversal se usa la media muestral para obtener una característica general de los datos, es decir, eliminar el ruido de las observaciones individuales y enfocarse en la tendencia central. Siguiendo este mismo principio, se pueden promediar un número de observaciones en torno al momento t para obtener una medida de tendencia central de la serie en el momento t . Al tomar el promedio se reduce el ruido idiosincrático en la serie.

Si X_t denota el valor de la serie X en el momento t , la media móvil M_t se define como:

$$M_t = \frac{1}{F + L + 1} \sum_{i=-L}^F X_{t+i}$$

Donde L denota el número de valores rezagados incluidos en el promedio, y F el número de valores adelantados incluidos en el promedio.

Por ejemplo, una media móvil “simétrica” de cinco periodos tiene $L=2$ y $F=2$, y:

$$M_t = \frac{1}{5} (X_{t-2} + X_{t-1} + X_t + X_{t+1} + X_{t+2})$$

De acuerdo a esta fórmula, no se puede computar la media móvil para las primeras dos y últimas dos observaciones. Sin embargo, se puede ignorar la falta de información y hacer el siguiente cálculo:

$$M_1 = \frac{1}{3} (X_1 + X_2 + X_3)$$

$$M_T = \frac{1}{3} (X_{T-2} + X_{T-1} + X_T)$$

Esto es justamente lo que hace el comando `tssmooth ma` de STATA.

Normalmente el número de términos en la media móvil es el número de periodos de tiempo en un año. Por ejemplo, si los datos son trimestrales se escogen cuatro datos, si los datos son mensuales 12 datos, etc. Ahora supongamos que tenemos datos trimestrales y debemos escoger 4 observaciones para calcular la media móvil, estas cuatro observaciones se pueden escoger de varias maneras, por ejemplo:

$$M_{1t} = \frac{1}{4} (X_{t-2} + X_{t-1} + X_t + X_{t+1})$$

$$M_{2t} = \frac{1}{4} (X_{t-1} + X_t + X_{t+1} + X_{t+2})$$

Bowerman y O’Connell (1993) sugieren que para calcular una media móvil centrada en t se debe promediar M_{1t} y M_{2t} .

Mediante los siguientes comandos podemos crear nuevas variables que contienen el promedio móvil de la serie imacec original:

```
tssmooth ma smoothimacec1=imacec, window(11 1)
```

The smoother applied was

```
(1/12)*[x(t-11) + x(t-10) + x(t-9) + x(t-8) + x(t-7) + x(t-6) + x(t-5) +  
x(t-4) + x(t-3) + x(t-2) + x(t-1) +  
...; x(t)= imacec
```

```
tssmooth ma smoothimacec2=imacec, window(6 0 6)
```

The smoother applied was

```
(1/12)*[x(t-6) + x(t-5) + x(t-4) + x(t-3) + x(t-2) + x(t-1) + 0*x(t) +  
x(t+1) + x(t+2) + x(t+3) + x(t+4) +  
x(t+5) + ...; x(t)= imacec
```

```
tssmooth ma smoothimacec3=imacec, window(6 1 6)
```

The smoother applied was

```
(1/13)*[x(t-6) + x(t-5) + x(t-4) + x(t-3) + x(t-2) + x(t-1) + 1*x(t) +  
x(t+1) + x(t+2) + x(t+3) + x(t+4) +  
x(t+5) + ...; x(t)= imacec
```

En el primer ejemplo se crea la variable smoothimacec1 que tiene el promedio móvil de 11 periodos atrás más el periodo actual. En el segundo ejemplo, se crea la variable smoothimacec2 con el promedio móvil tomando 6 valores atrás de t y 6 valores delante de t, pero sin tomar el valor en t. Por último, en el tercer ejemplo, se crea la variable smoothimacec3 con el promedio móvil de imacec tomando 6 observaciones antes de t, 6 observaciones después de t, y la observación en t.

```
. list fecha imacec smoothimacec1 smoothimacec2 smoothimacec3 if _n<=12
```

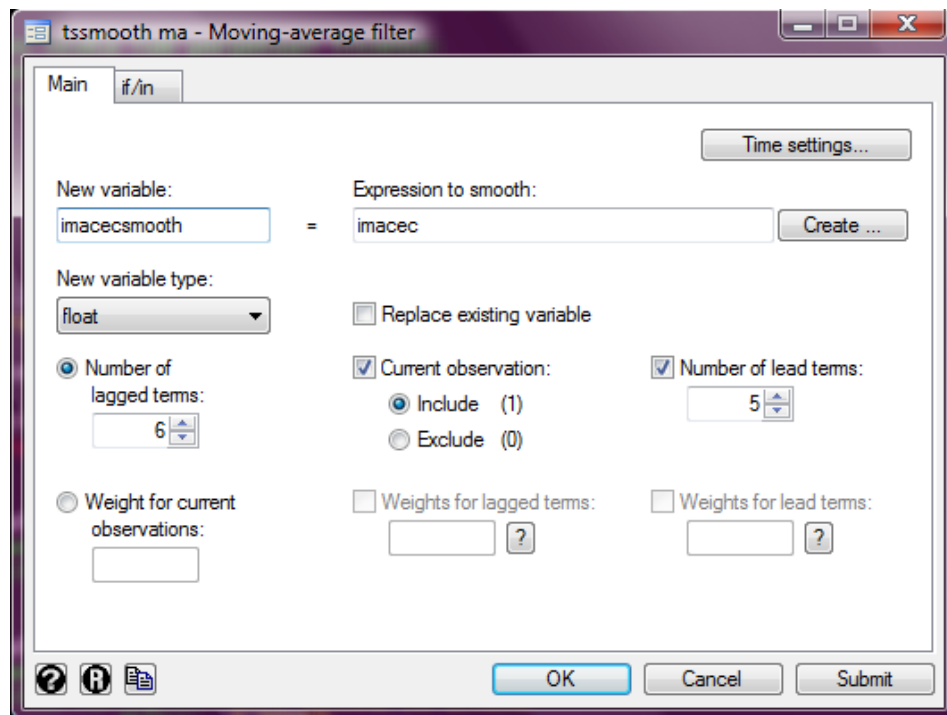
	fecha	imacec	smooth~1	smooth~2	smooth~3
1.	1986m1	36.1797	36.1797	37.34546	37.17892
2.	1986m2	33.6536	34.91666	37.51589	37.03311
3.	1986m3	38.755	36.19609	36.67912	36.90977
4.	1986m4	38.8662	36.86363	37.04193	37.22436
5.	1986m5	38.2517	37.14124	37.2792	37.3676
6.	1986m6	37.6241	37.22171	37.52893	37.53686
7.	1986m7	36.9222	37.17892	37.71871	37.65744
8.	1986m8	36.0124	37.03311	37.89367	37.74895
9.	1986m9	35.9231	36.90977	38.70213	38.48836
10.	1986m10	40.0556	37.22436	38.54798	38.66395
11.	1986m11	38.8001	37.3676	38.82961	38.82733
12.	1986m12	39.3987	37.53686	38.98039	39.01256

La sintaxis general del comando es la siguiente:

```
tssmooth ma newvar=exp [if] [in], window(#L [#C [#F]])
```

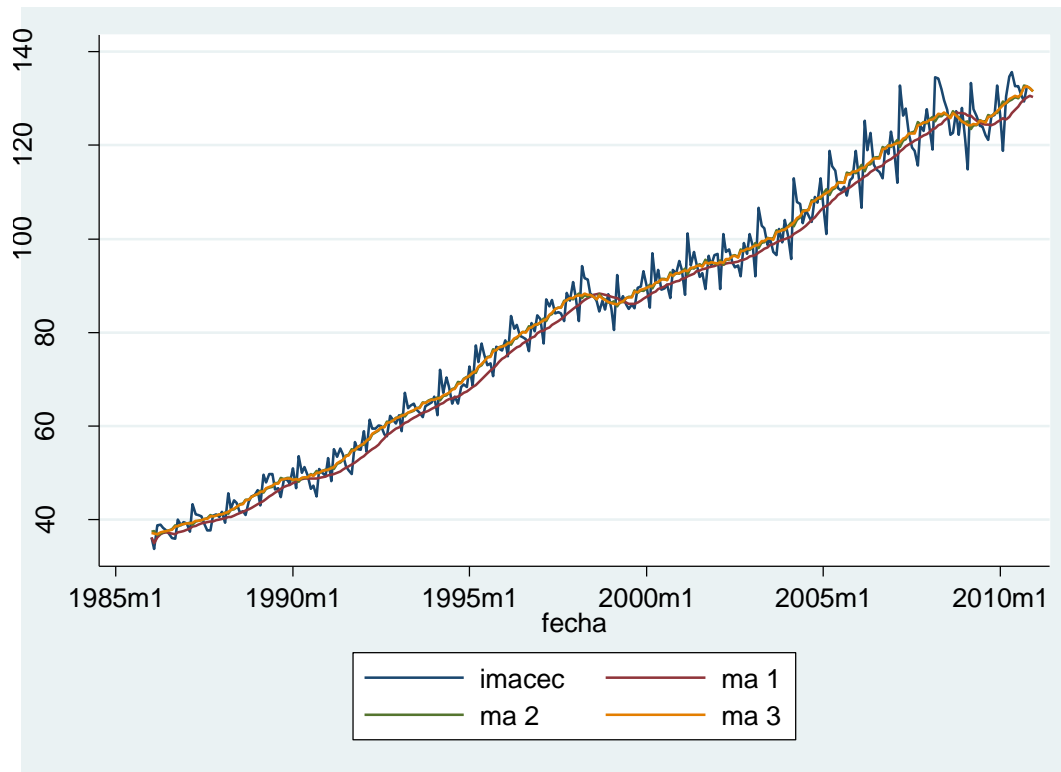
También podemos realizar esto mediante la ventana de comandos en:

Statistics>>Time Series>>Smoothers/univariate forecasters>>Moving average filters



Podemos graficar las series filtradas con los tres tipos de media móvil:

```
tsline imacec smoothimacec1 smoothimacec2 smoothimacec3, legend(label(1  
"imacec") label(2 "ma 1") label(3 "ma 2") label(4 "ma 3"))
```



IV.3 Medias móviles ponderadas

Los filtros de medias móviles hasta ahora calculados le dan igual peso a cada una de las observaciones, por ejemplo, al calcular una media móvil de cuatro periodos cada observación recibía un peso de $\frac{1}{4}$.

Sin embargo, la definición de media móvil puede fácilmente ser generalizada a la siguiente expresión:

$$M_t = \frac{\sum_{i=-L}^F w_i X_{t+i}}{\sum_{i=-L}^F w_i}$$

Donde w_i son los ponderadores que recibe cada una de las observaciones.

Por ejemplo, podemos calcular una media móvil ponderada simétrica de cinco periodos dándole un peso de 3 a la observación en t , de 2 a las observaciones distanciadas un periodo y de 1 a las observaciones distanciadas dos periodos:

$$M_t = \frac{X_{t-2} + 2X_{t-1} + 3X_t + 2X_{t+1} + X_{t+2}}{1 + 2 + 3 + 2 + 1}$$

El comando `tssmooth` ma contiene la opción de indicar ponderadores para las observaciones:

```
tssmooth ma newvar = exp [if] [in], weights([numlist_1] <#c>
[numlist_f])
```

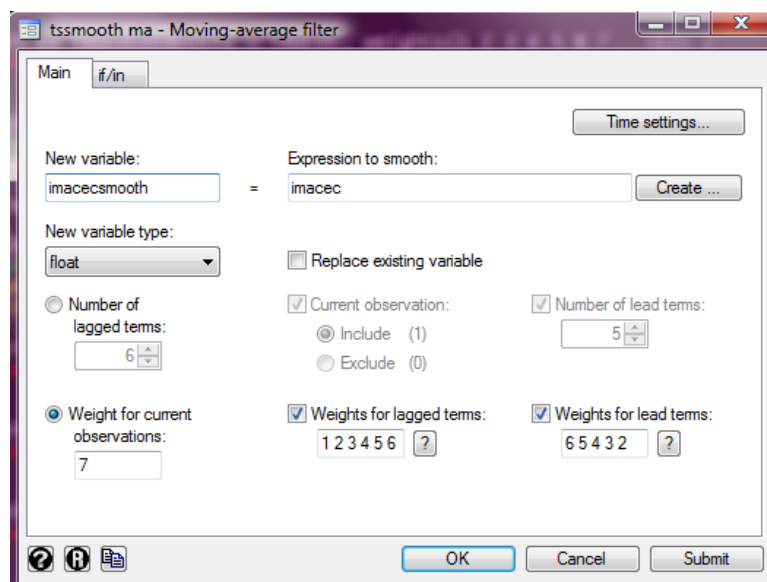
Notemos que en esta sintaxis se ha cambiado la opción `window()` por la opción `weigh()`, ya que en este caso a través de la opción `weight()` indicaremos el número de rezagos, valores adelantados, y si se incluye o no el valor en t , además de sus respectivos ponderadores.

Por ejemplo, calculemos la media móvil ponderada de la serie IMACEC, tomando el valor actual, 6 rezagos, y 5 valores adelantados de la serie dando un ponderador de 7 a la observación actual, y disminuyendo en una unidad el ponderador en la medida que se aleja de la observación actual.

```
tssmooth ma imacecsmooth1 = imacec, weights(1 2 3 4 5 6 <7> 6 5 4 3 2)

The smoother applied was
by año : (1/48)*[1*x(t-6) + 2*x(t-5) + 3*x(t-4) + 4*x(t-3) + 5*x(t-2) +
6*x(t-1) + 7*x(t) + 6*x(t+1) +
5*x(t+2) + 4*x(t+3) + 3*x(t+4) + ...; x(t)= imacec
```

Lo que también se puede realizar mediante la ventana de comandos:



IV.4 Suavización exponencial

IV.4.1 Suavización exponencial simple

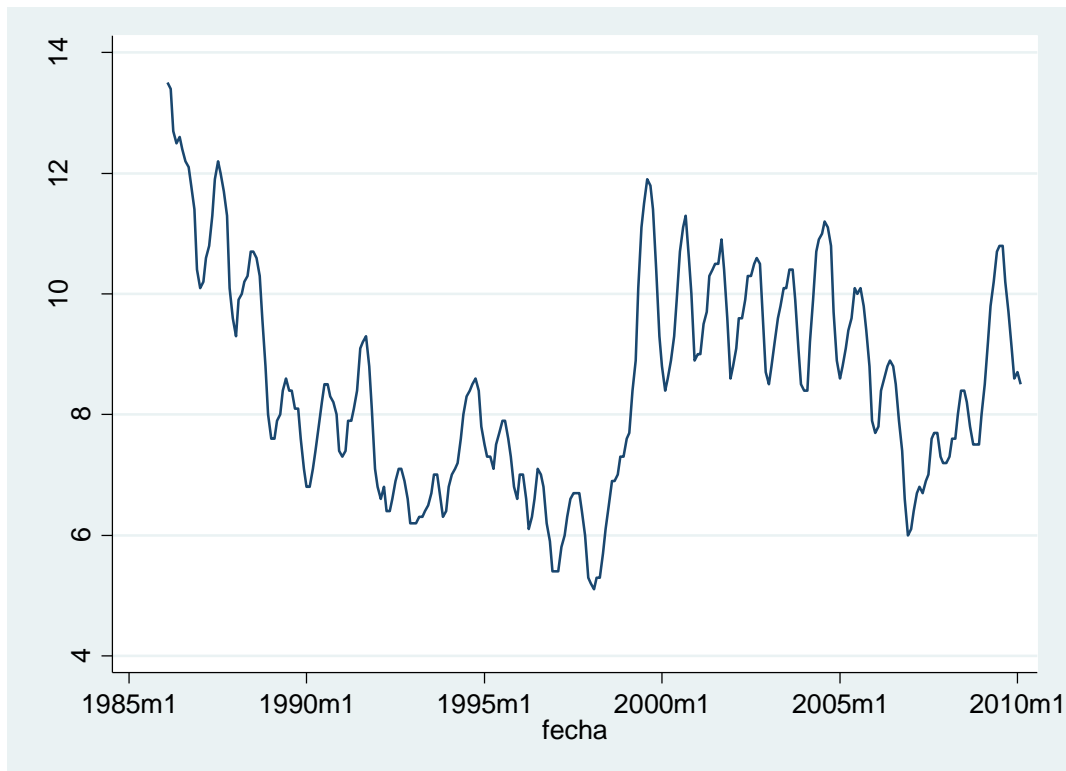
En la base de datos desempleo.dta se encuentra la tasa de desempleo mensual de Chile desde febrero de 1982 a febrero de 2010 (Fuente: INE).

Primero carguemos la base de datos y grafiquemos la serie:

```
use desempleo.dta

tsset fecha
      time variable:  fecha, 1986m2 to 2010m2
              delta:  1 month

twoway (tsline desempleo), ytitle(Tasa de desempleo)
```

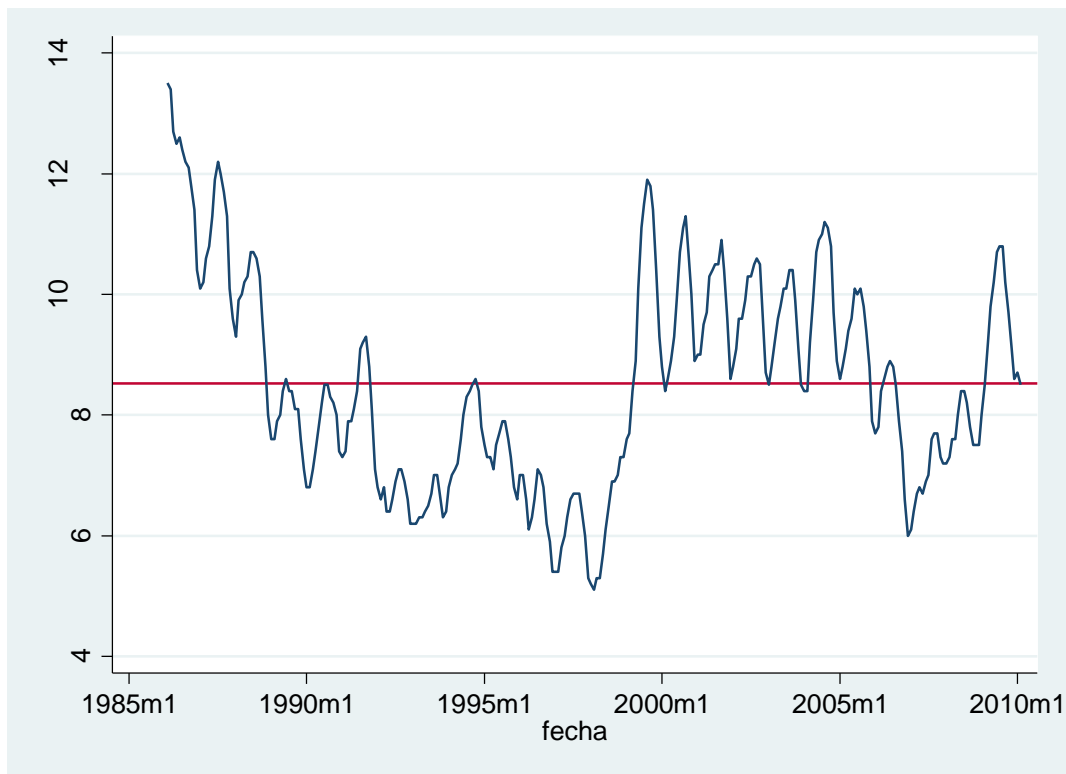


A pesar de que desde 1986 al año 1998 se observa una caída en la tasa de desempleo, y luego un alza, no existe una tendencia clara positiva o negativa de esta serie en el tiempo. Podríamos decir que esta serie fluctúa en torno a la media. Al gráfico anterior se le puede agregar una línea equivalente al promedio de la tasa de desempleo en este periodo:


```
sum desempleo
```

Variable	Obs	Mean	Std. Dev.	Min	Max
desempleo	289	8.523183	1.763081	5.1	13.5

```
tsline desempleo, yline(`=r(mean)')`
```



De esta manera, podemos modelar la tasa de desempleo en función de una constante y un término de error idiosincrático:

$$desempleo_t = X_t = \beta_0 + \varepsilon_t$$

El estimador MCO de b_0 es simplemente el promedio de la tasa de desempleo en la muestra disponible de datos. Sin embargo, sería más precisa la estimación si se considera que esta media puede ir variando en el tiempo, y no es necesariamente la misma para todo el periodo.

$$X_t = \beta_{0t} + \varepsilon_t$$

El método de suavización exponencial simple es adecuado cuando se tiene una serie de tiempo que no presenta una tendencia creciente o decreciente, y con una media que cambia en el tiempo.

Suponga que tenemos el valor inicial S_0 de la versión suavizada de la serie desempleo (luego discutiremos cómo obtener S_0). Luego, el filtro exponencial simple S_t se define como:

$$S_t = \alpha X_t + (1 - \alpha)S_{t-1}$$

Donde α es un parámetro de suavización entre 0 y 1 que se debe escoger, y S_{t-1} corresponde al promedio de las primeras $t-1$ observaciones de la serie X_t . De esta forma, el filtro exponencial simple actualiza la media en cada periodo t utilizando una fracción del valor observado de la serie en t .

S_t puede ser vista como la predicción para la serie en X_{t+1} , este filtro es bastante popular entre las personas que se dedican a hacer predicciones, por su simplicidad y ya que generalmente funciona bastante bien.

Podemos ordenar los términos de la ecuación anterior de la siguiente manera:

$$S_t = \alpha(X_t - S_{t-1}) + S_{t-1}$$

Donde $(X_t - S_{t-1})$ representa el error de predicción en el periodo t . En economía esta ecuación se conoce como el modelo de expectativas adaptativas o modelo de ajuste parcial.

Volviendo a la primera ecuación esta puede ser escrita de otra manera:

$$S_t = \alpha X_t + \alpha(1 - \alpha)X_{t-1} + \alpha(1 - \alpha)^2 S_{t-2}$$

$$S_t = \alpha \sum_{i=0}^{t-1} (1 - \alpha)^i X_{t-i} + (1 - \alpha)^t S_0$$

Lo que muestra que el filtro exponencial simple tiene la forma de media móvil, donde los ponderadores son decrecientes en el tiempo, y además se incluyen todas las observaciones.

Podríamos ocupar el promedio de la serie en todo el periodo de tiempo disponible como valor para S_0 , pero en series donde se distingue claramente cambios en la media en el tiempo, esta no será la mejor aproximación de S_0 . Una mejor alternativa consiste en utilizar la primera parte de las observaciones. El comando `tssmooth exponential` de STATA utiliza la primera mitad de los datos, pero si se dispone de mucha información utilizar el 10% o 25% de las primeras observaciones también podría ser una buena aproximación.

La sintaxis del comando de STATA para obtener la suavización exponencial simple se una serie es el siguiente:

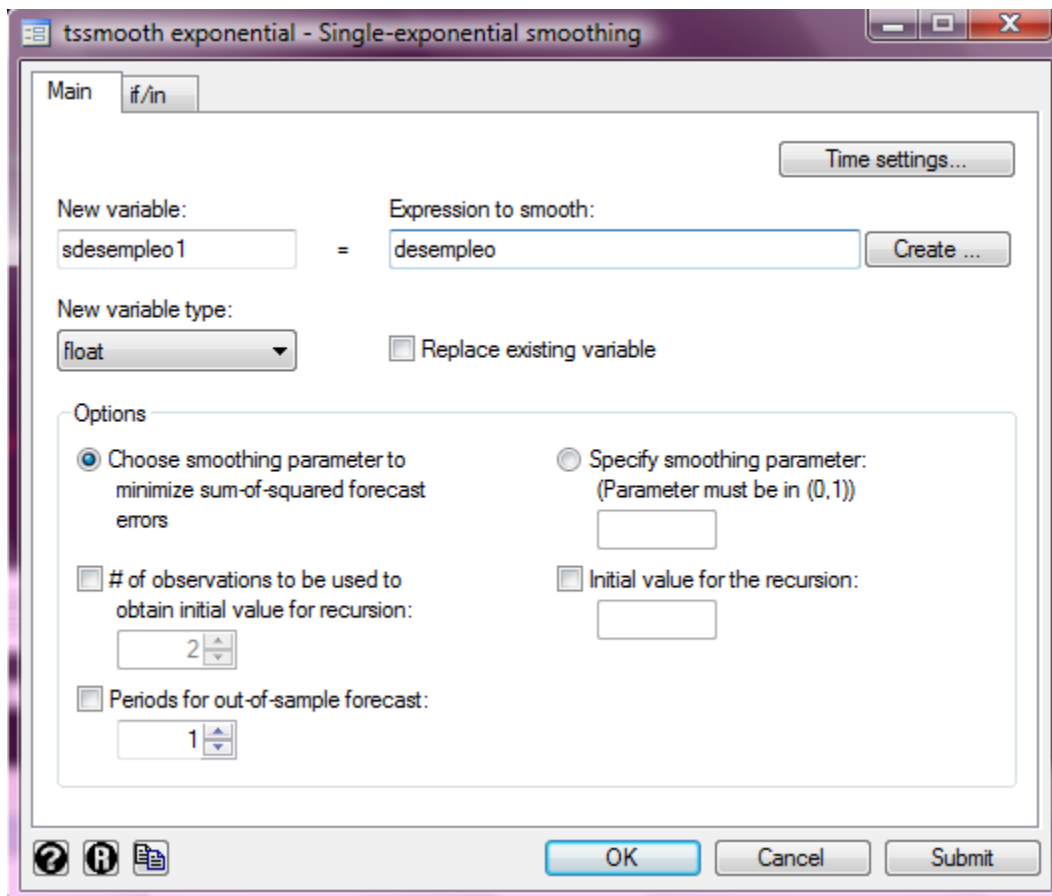
```
tssmooth exponential [type] newvar = exp [if] [in] [, options]
```

Dentro de las opciones se encuentra:

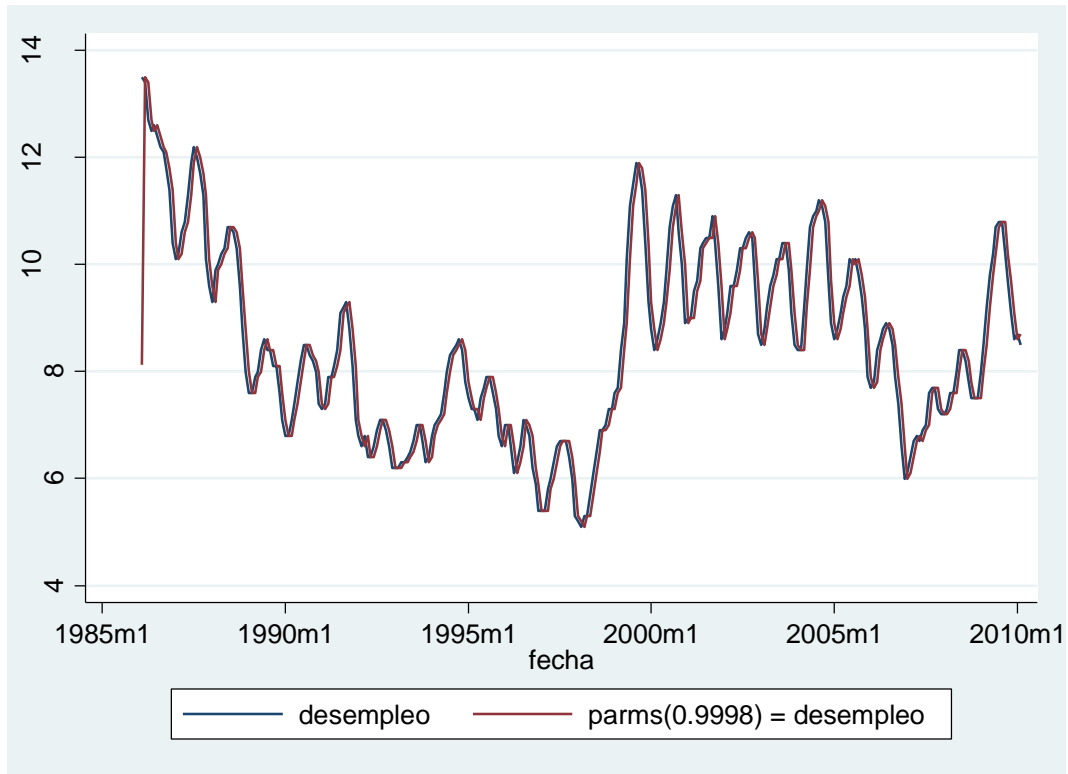
- ❖ `replace`: reemplaza la con variable que se esta creando (serie suavizada) si es que esta ya existe.

- ❖ `parms(#a)`: se especifica el parámetro a utilizado en la suavización; $0 < \#a < 1$. Si `parms(#a)` no es especificado este es escogido de forma tal de minimizar la suma al cuadrado de los errores de predicción.
- ❖ `samp0(#)` and `s0(#)` son dos maneras mutuamente excluyentes para especificar S_0 .
- ❖ `samp0(#)` indica que el valor de S_0 se obtiene de promediar las primeras $\#$ observaciones.
- ❖ `s0(#)` especifica el valor de S_0 que debe ser utilizado.
- ❖ Si ninguna de las dos opciones es especificada, por defecto utiliza el promedio de la primera mitad de las observaciones.
- ❖ `forecast(#)` indica el número de observaciones que se desea predecir fuera de muestral., donde $0 < \# < 500$. Si no se especifica, el defecto es `forecast(0)`.

Dado que el filtro exponencial simple con frecuencia es utilizado para hacer predicciones, este comando genera esta nueva variable con el valor de S_{t-1} en la posición t , el que corresponde al valor suavizado de la serie en $t-1$, por lo cual si queremos obtener la serie suavizada en t (S_t) debemos utilizar el operador forward.

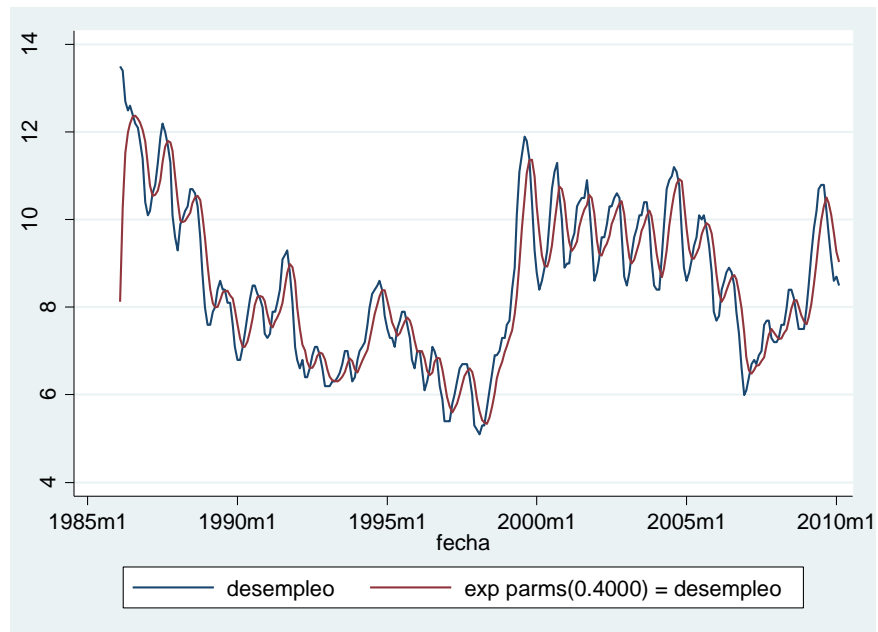


```
tssmooth exponential sdesempleo1 = desempleo  
  
computing optimal exponential coefficient (0,1)  
  
optimal exponential coefficient = 0.9998  
sum-of-squared residuals = 77.522763  
root mean squared error = .5179236  
  
tsline desempleo sdesempleo1
```



```
tssmooth exponential sdesempleo2 = desempleo, parms(0.4)  
  
exponential coefficient = 0.4000  
sum-of-squared residuals = 173.89  
root mean squared error = .77568  
  
tsline desempleo sdesempleo2
```

December 31, 2010

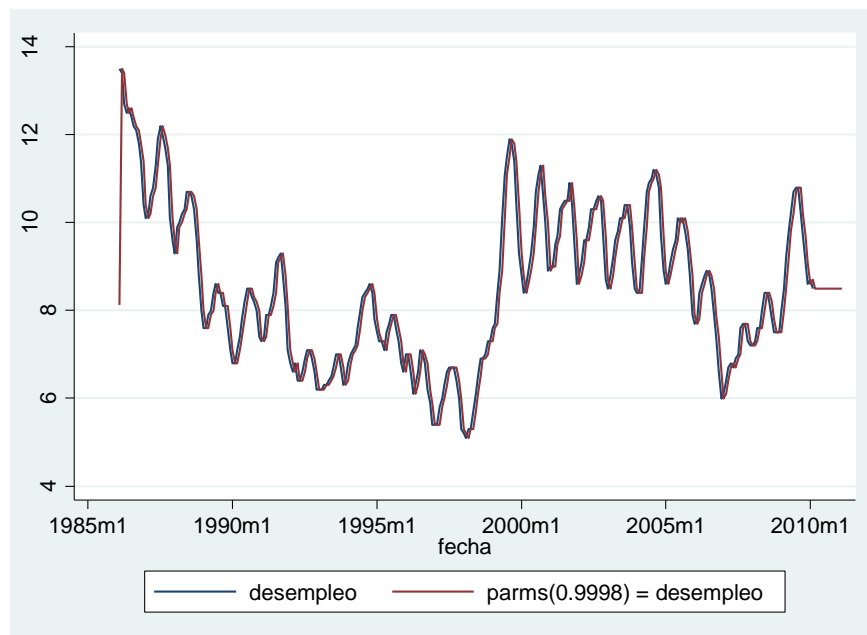


```
tssmooth exponential sdesempleo3 = desempleo, forecast(12)

computing optimal exponential coefficient (0,1)

optimal exponential coefficient =      0.9998
sum-of-squared residuals      =      77.522763
root mean squared error      =      .5179236

tsline desempleo sdesempleo3
```

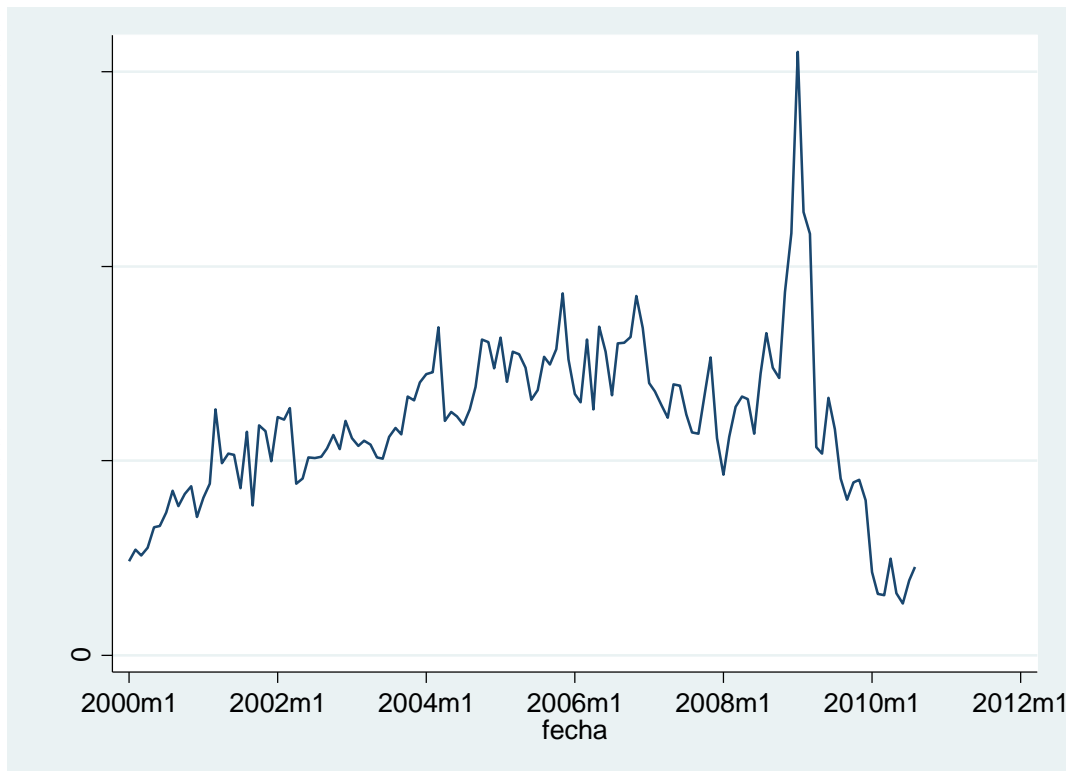


En el siguiente gráfico se presenta la cantidad de salmón del atlántico cosechado entre Enero de 2000 y agosto de 2010:

```
use sa, clear

tsset fecha
    time variable:  fecha, 2000m1 to 2011m8
    delta: 1 month

tsline sa, ytitle(salmón atlántico (ton))
```

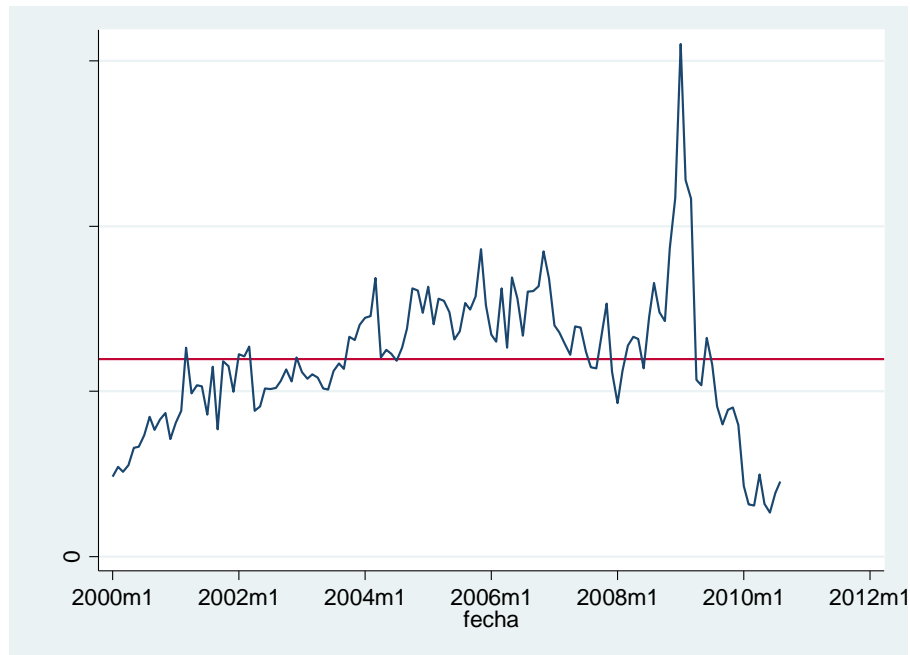


Esta serie tampoco presenta una tendencia creciente o decreciente en el tiempo y más bien tiende a fluctuar en torno a un valor medio.

```
sum sa
```

Variable	Obs	Mean	Std. Dev.	Min	Max
sa	128	23900.18	8479.046	5352.166	62028.66

```
tsline sa, ytitle(salmón atlántico (ton)) yline(`=r(mean)')
```



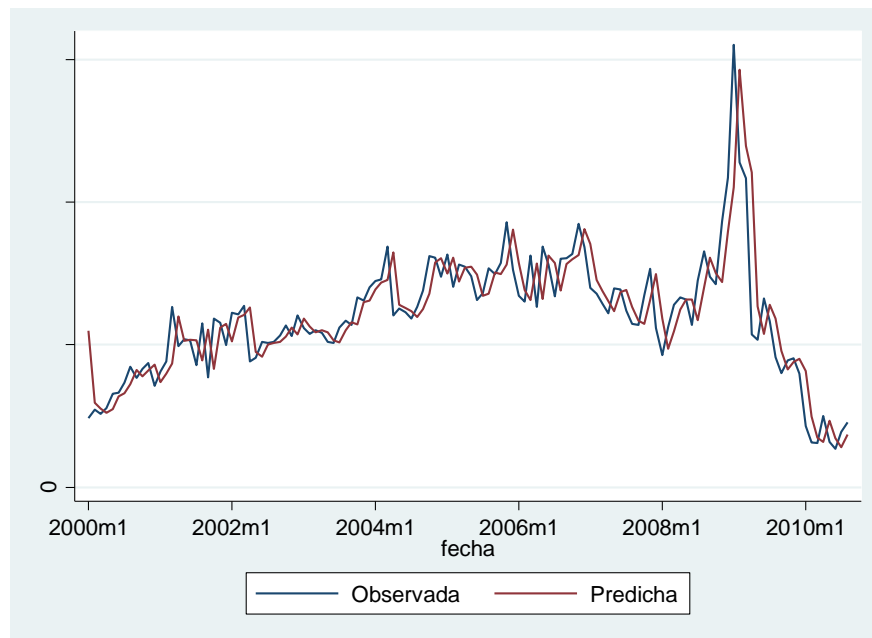
Mediante el comando `tssmooth exponential` podemos obtener la serie suavizada de la cosecha de salmón atlántico:

```
tssmooth exp sa_s1=sa

computing optimal exponential coefficient (0,1)

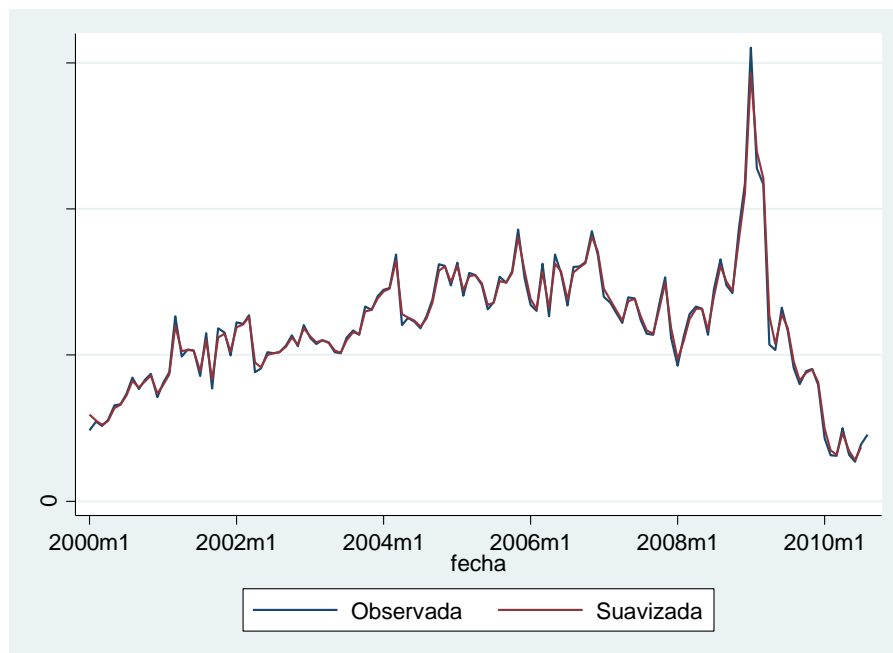
optimal exponential coefficient =      0.8258
sum-of-squared residuals      = 2755288384
root mean squared error      = 4639.5787

tsline sa sa_s1, legend(label(1 "Observada") label(2 "Predicha"))
```



Pero recordemos que la variable creada por el comando corresponde a la predicción de la serie en t , es decir, la serie suavizada pero en $t-1$. Por lo cual si queremos graficar la serie suavizada, debemos hacer lo siguiente:

```
g sa_s2=F.sa_s1  
(1 missing value generated)  
tsline sa sa_s2, legend(label(1 "Observada") label(2 "Suavizada"))
```



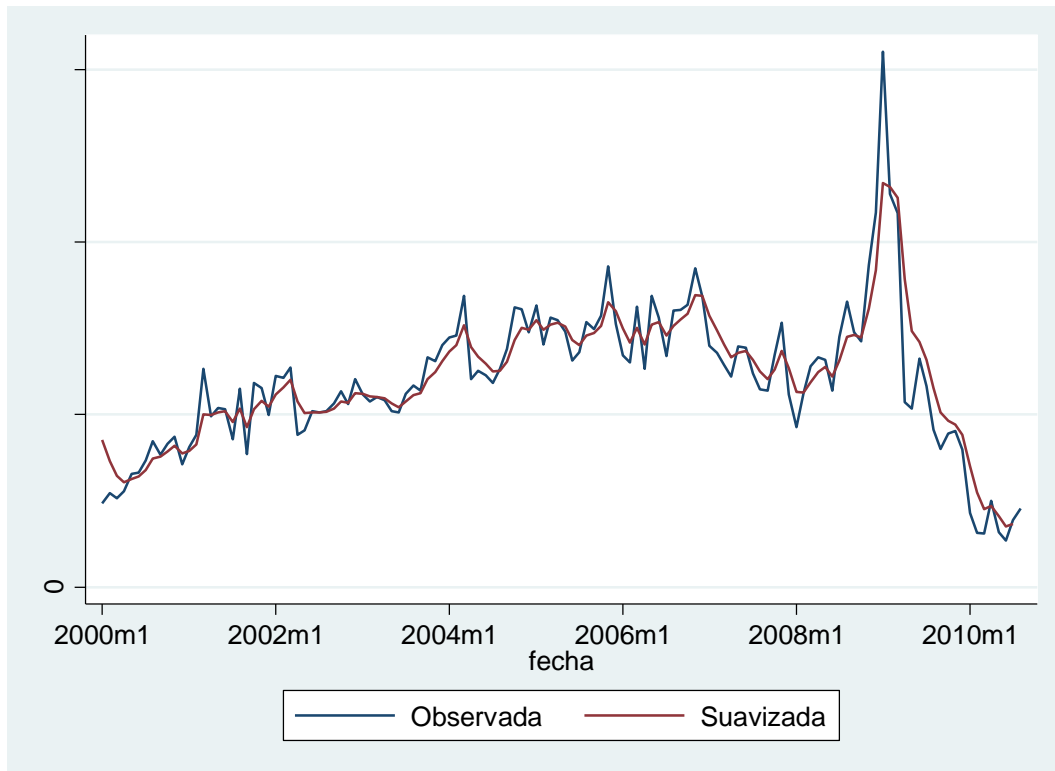
Se puede observar que la serie suavizada es bastante similar a la serie observada, esto porque el parámetro utilizado es alto. Si escogemos un parámetro más pequeño la serie se tenderá a suavizar más:

```
tssmooth exp sa_s3=sa, parms(0.4)

exponential coefficient =      0.4000
sum-of-squared residuals = 3263407124
root mean squared error =    5049.3

g sa_s4=F.sa_s3
(1 missing value generated)

tsline sa sa_s4, legend(label(1 "Observada") label(2 "Suavizada"))
```



Ahora, realicemos una predicción a 24 meses utilizando ambos parámetros:

```

tssmooth exp sa_s5=sa, forecast(24)

computing optimal exponential coefficient (0,1)

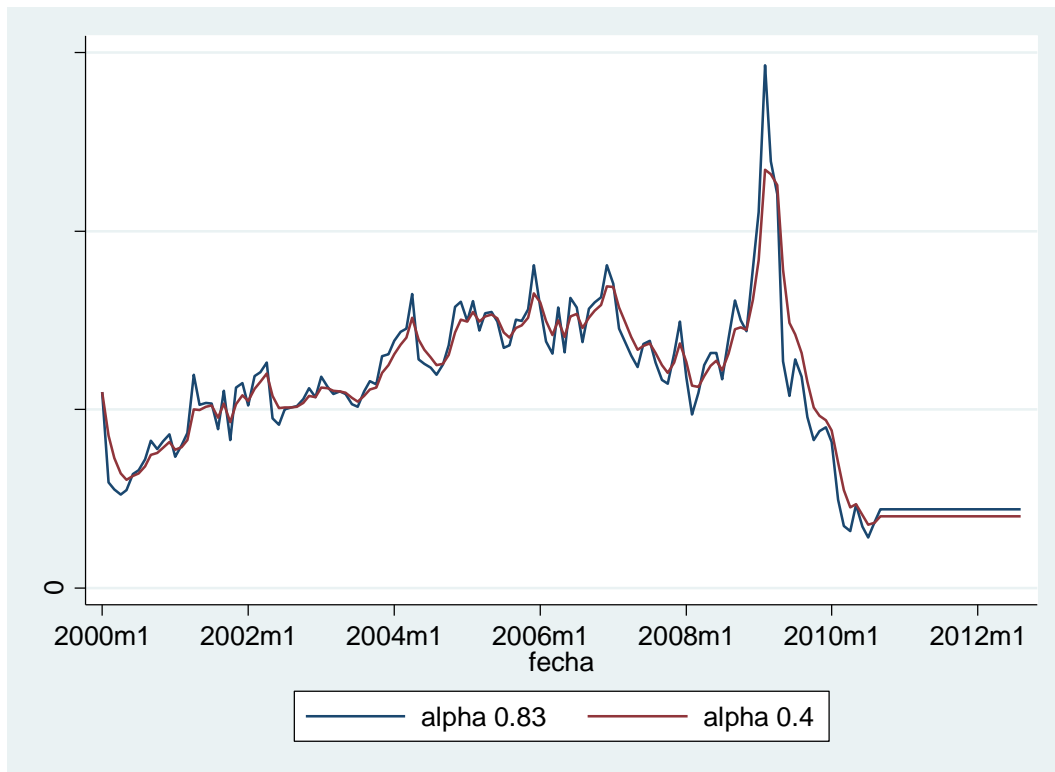
optimal exponential coefficient =      0.8258
sum-of-squared residuals      = 2755288384
root mean squared error      = 4639.5787

tssmooth exp sa_s6=sa, forecast(24) parms(0.4)

exponential coefficient =      0.4000
sum-of-squared residuals = 3263407124
root mean squared error = 5049.3

tsline sa_s5 sa_s6, legend(label(1 "alpha 0.83") label(2 "alpha 0.4"))

```



IV.4.1 Suavización exponencial doble

Ahora suponga que tiene una serie de tiempo cuya media cambia en el tiempo y además presenta una tendencia. La presencia de esta tendencia hace el filtro exponencial simple no sea apropiado en este caso.

En este caso la serie X_t puede ser modelada de la siguiente manera:

$$X_t = \beta_{0t} + \beta_{1t}t + e_t$$

Lo que permite que la media de X_t dependa de t y se incremente en la medida que pasa el tiempo.

La sintaxis del comando STATA para hacer suavización exponencial doble es la siguiente:

```
tssmooth dexpontial [type] newvar = exp [if] [in] [, options]
```

Con las mismas opciones del comando `tssmooth exponential`.

Por ejemplo, a través de los siguientes comandos podemos generar la serie suavizada mediante filtro exponencial doble de cosecha de salmón atlántico:

```
use sa, clear

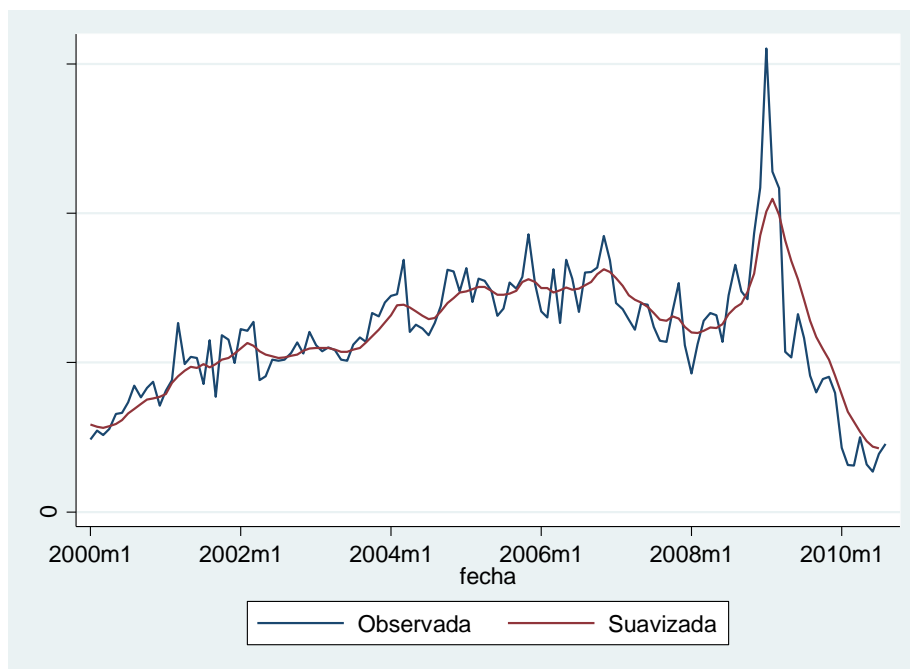
tsset fecha
    time variable:  fecha, 2000m1 to 2010m8
    delta:  1 month

tssmooth dexp sa_s1=sa
computing optimal double-exponential coefficient (0,1)

optimal double-exponential coefficient =      0.3693
sum-of-squared residuals               =    2969621125
root mean squared error                 =    4816.655

g sa_s2=F.sa_s1
(1 missing value generated)

tsline sa sa_s2, legend(label(1 "Observada") label(2 "Suavizada"))
```



A continuación podemos comparar las predicciones realizadas por ambos tipos de suavización exponencial:

```
tssmooth dexp sa_s3=sa, forecast(24)
computing optimal double-exponential coefficient (0,1)

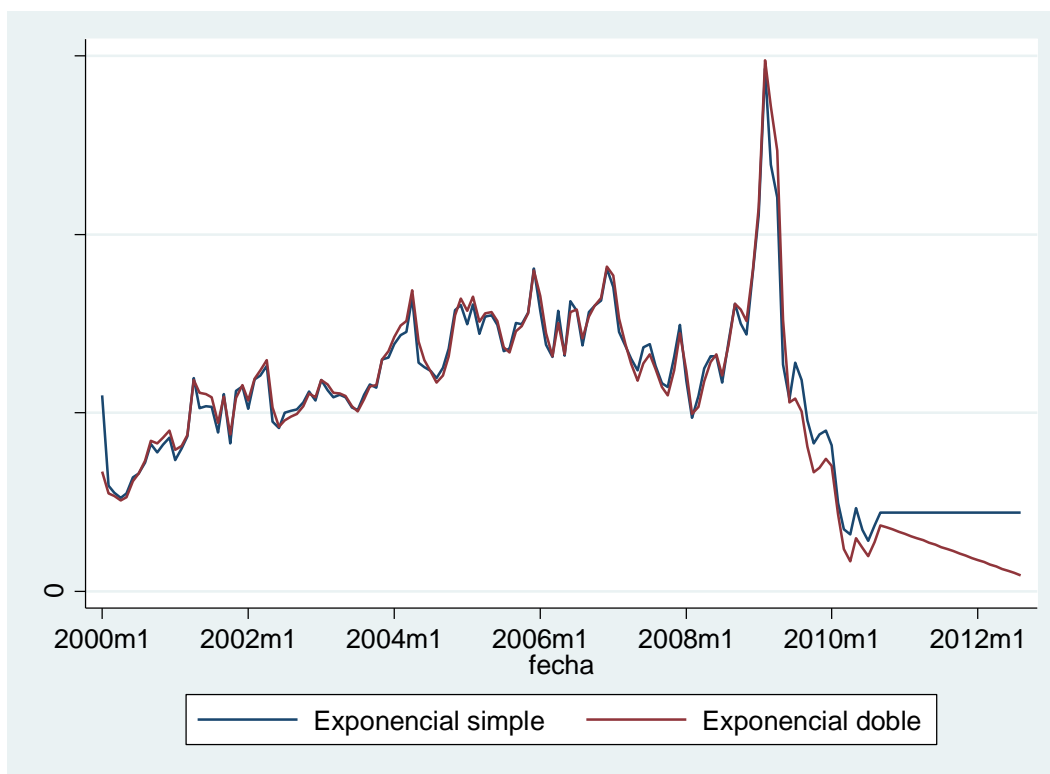
optimal double-exponential coefficient =      0.3693
sum-of-squared residuals                = 3138130066
root mean squared error                 =  4951.4282

tssmooth exp sa_s4=sa, forecast(24)

computing optimal exponential coefficient (0,1)

optimal exponential coefficient =      0.8258
sum-of-squared residuals        = 2755288384
root mean squared error         =  4639.5787

tsline sa_s4 sa_s3 , legend(label(1 "Exponencial simple") label(2
"Exponencial doble"))
```



IV.5 Filtro Holt-Winters

IV.5.1 Sin estacionalidad

El método de suavización exponencial doble, utilizaba el mismo parámetro α , en las dos suavizaciones realizadas a la serie. El método Holt-Winters libera este supuesto permitiendo que el parámetro de suavización del primer filtro sea diferente al del segundo.

La sintaxis del comando STATA para realizar este filtro es:

```
tssmooth hwinters [type] newvar = exp [if] [in] [, options]
```

Con las mismas opciones de los comandos anteriores.

Veamos que resulta de utilizar el filtro Holt-Winters en la serie de tiempo cosecha de salmón atlántico.

```
use sa, clear

tsset fecha
      time variable: fecha, 2000m1 to 2010m8
            delta: 1 month

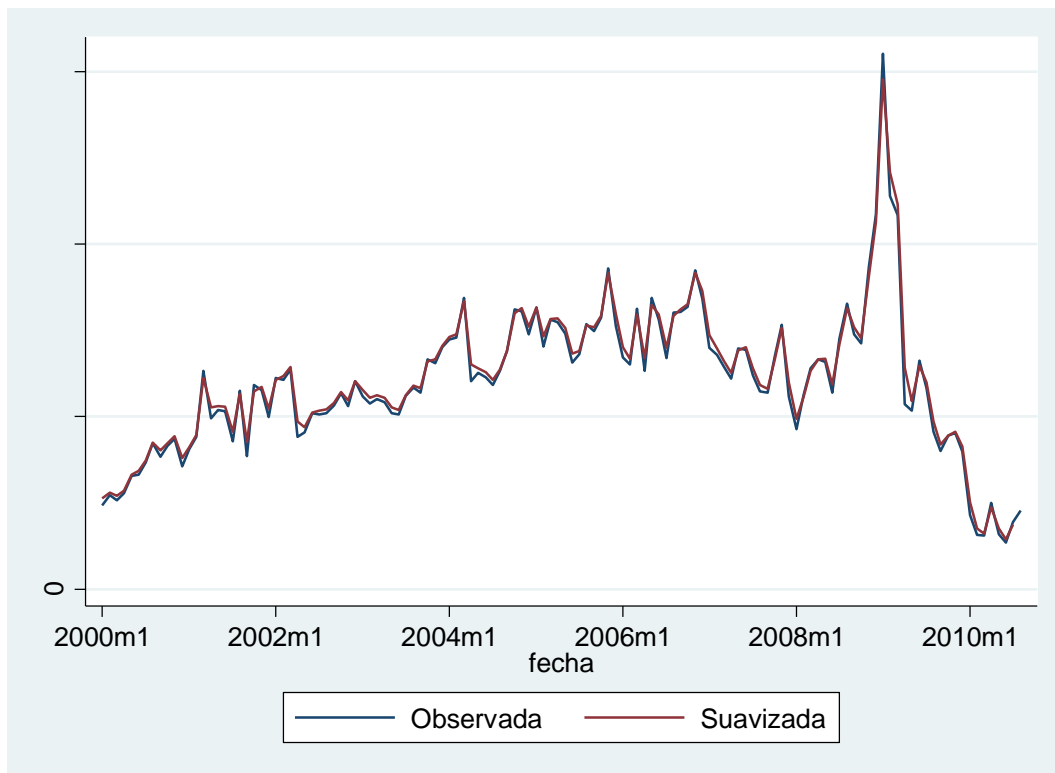
tssmooth hwinters sa_s1=sa, samp0(30)
computing optimal weights

Iteration 0:   penalized RSS = -3.740e+09   (not concave)
Iteration 1:   penalized RSS = -2.703e+09
Iteration 2:   penalized RSS = -2.639e+09
Iteration 3:   penalized RSS = -2.638e+09
Iteration 4:   penalized RSS = -2.638e+09
Iteration 5:   penalized RSS = -2.638e+09
Iteration 6:   penalized RSS = -2.638e+09

Optimal weights:
                        alpha = 0.8271
                        beta  = 0.0063
penalized sum-of-squared residuals = 2.64e+09
      sum-of-squared residuals = 2.64e+09
      root mean squared error = 4539.824

g sa_s2=F.sa_s1
(1 missing value generated)

tsline sa sa_s2, legend(label(1 "Observada") label(2 "Suavizada"))
```



Comparemos ahora las predicciones realizada por los tres filtros:

```
tssmooth hwinters sa_s3=sa, samp0(30) forecast(24)

Optimal weights:
               alpha = 0.8271
               beta  = 0.0063
penalized sum-of-squared residuals = 2.64e+09
sum-of-squared residuals = 2.64e+09
root mean squared error = 4539.824

tssmooth dexp sa_s4=sa, samp0(30) forecast(24)

computing optimal double-exponential coefficient (0,1)

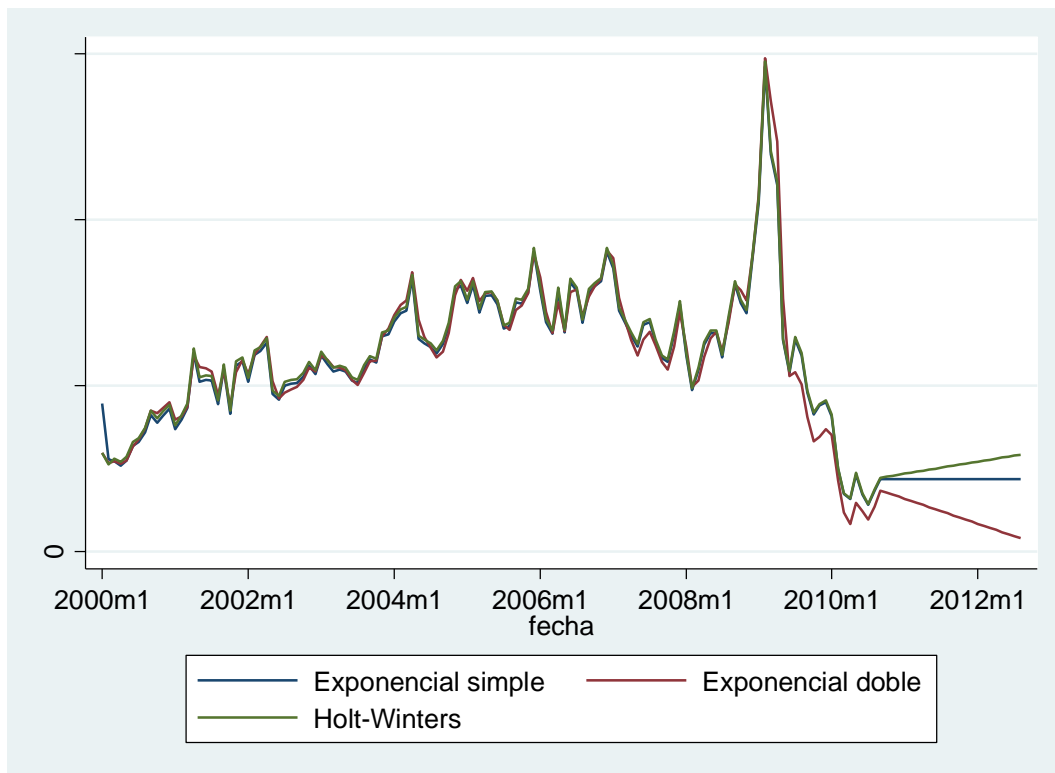
optimal double-exponential coefficient = 0.3681
sum-of-squared residuals = 3125155446
root mean squared error = 4941.1817

tssmooth exp sa_s5=sa, samp0(30) forecast(24)

computing optimal exponential coefficient (0,1)

optimal exponential coefficient = 0.8211
sum-of-squared residuals = 2671721663
root mean squared error = 4568.6787

tsline sa_s5 sa_s4 sa_s3 , legend(label(1 "Exponencial simple") label(2
"Exponencial doble") label(3 "Holt-Winters"))
```



Podemos notar que el parámetro β correspondiente al segundo filtro es bastante cercano a cero, de esta forma el filtro exponencial simple y Holt-Winters (HW) son bastante similares. Adicionalmente, los errores cuadráticos medios (RMSE) son menores en el filtro exponencial simple y HW que en el filtro exponencial doble.

Dado que creemos que los datos presentan tendencias locales, nos deberíamos inclinar por utilizar HW. Sin embargo, dado que el coeficiente β es tan cercano a cero, aparentemente no existe ventaja alguna de ocupar este filtro versus el exponencial simple.

IV.5.2 Con estacionalidad

Los tres filtros exponenciales hasta ahora revisados, no consideran la posibilidad de estacionalidad en los datos. A continuación se presenta una versión modificada de la suavización Holt-Winters que incorpora la presencia de estacionalidad en los datos. En este caso tenemos un parámetro adicional, que corresponde al componente estacional.

La sintaxis del comando STATA para obtener la serie suavizada mediante Holt-Winters con estacionalidad es la siguiente:

```
tssmooth shwinters [type] newvar = exp [if] [in] [, options]
```

Que tiene las mismas opciones de los filtros exponenciales antes revisados, pero además se le agrega la opción `period(#)`, que permite ingresar número de periodos en la estacionalidad (frecuencia). Si no se indica, por defecto toma la frecuencia en los datos señalada en la opción `tsset`.

Tomemos la tasa de desempleo del estado de Kentucky, y obtengamos la serie filtrada utilizando HW estacional:

```
use urates.dta, clear

tssmooth shwinters kentucky_s1 =kentucky

computing optimal weights

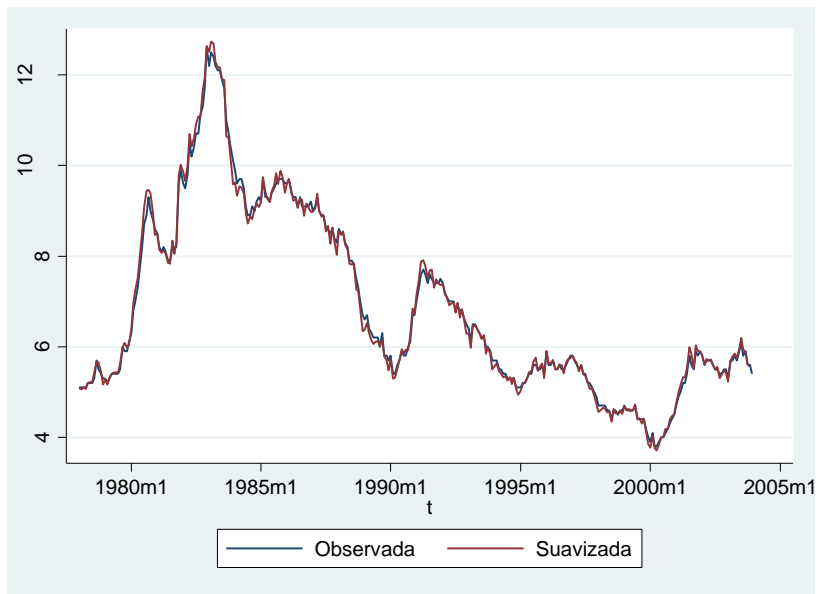
Iteration 0:   penalized RSS = -36.028124   (not concave)
Iteration 1:   penalized RSS = -14.570149   (not concave)
Iteration 2:   penalized RSS = -14.460323   (not concave)
Iteration 3:   penalized RSS = -14.433905
Iteration 4:   penalized RSS = -14.408993
Iteration 5:   penalized RSS = -14.386176
Iteration 6:   penalized RSS = -14.38517
Iteration 7:   penalized RSS = -14.385166
Iteration 8:   penalized RSS = -14.385166

Optimal weights:
                        alpha = 0.8879
                        beta  = 0.2473
                        gamma  = 0.1244
penalized sum-of-squared residuals = 14.38517
      sum-of-squared residuals = 14.38517
      root mean squared error  = .2147238

g  kentucky_s2=F.kentucky_s1
(1 missing value generated)

tsline  kentucky kentucky_s2, legend(label(1 "Observada") label(2
"Suavizada"))
```


December 31, 2010



Ahora comparemos las predicciones realizadas por los cuatro filtros para esta serie:

```
tssmooth shwinters kentucky_s1 =kentucky

Optimal weights:
                alpha = 0.8879
                beta  = 0.2473
                gamma = 0.1244
penalized sum-of-squared residuals = 14.38517
sum-of-squared residuals = 14.38517
root mean squared error = .2147238

tssmooth hwinters kentucky_s4=kentucky, forecast(24) samp0(30)

Optimal weights:
                alpha = 0.8667
                beta  = 0.2544
penalized sum-of-squared residuals = 14.09736
sum-of-squared residuals = 14.09736
root mean squared error = .2125649

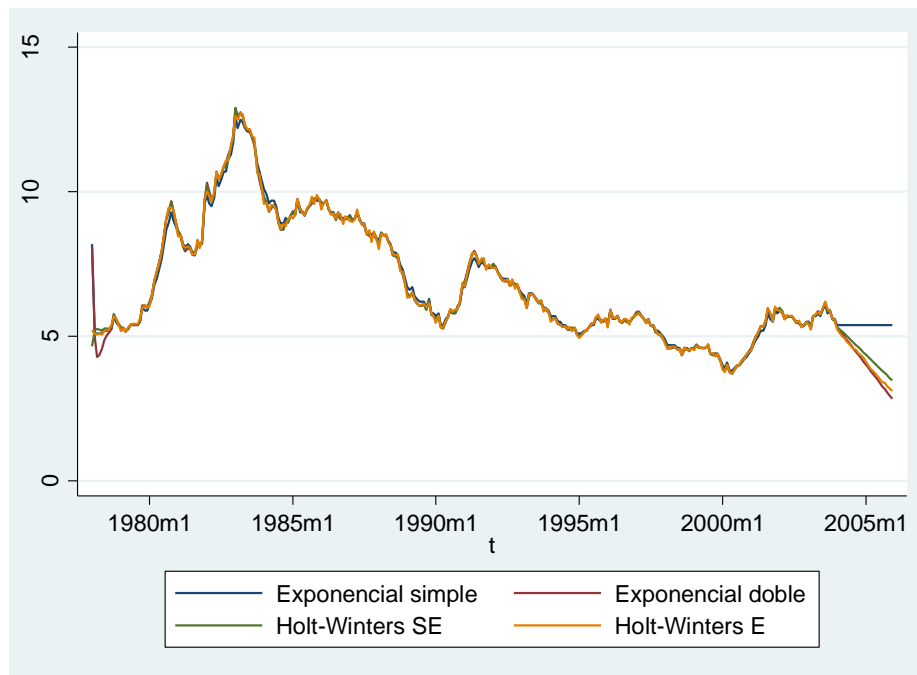
tssmooth dexp kentucky_s5=kentucky, forecast(24)

optimal double-exponential coefficient =      0.5235
sum-of-squared residuals              =      24.698182
root mean squared error                =      .28135536

tssmooth exp kentucky_s6=kentucky, forecast(24)

optimal exponential coefficient =      0.9998
sum-of-squared residuals        =      23.937495
root mean squared error         =      .2769887

tsline  kentucky_s6 kentucky_s5 kentucky_s4 kentucky_s3, legend(label(1
"Exponencial simple") label(2 "Exponencial doble") label(3 "Holt-Winters SE")
label(4 "Holt-Winters E"))
```



En este caso, el filtro HW sin estacionalidad es el que tiene menor RMSE.

V. Análisis Descriptivo de Series de Tiempo

Cuando se trabaja con datos de corte transversal, usualmente la media y la desviación estándar describen la tendencia central de la variable y su dispersión, y se utiliza la covarianza o correlación para describir en qué grado dos o más variables se mueven juntas.

Con datos de series de tiempo debemos ser más cuidadosos, ya que la característica típica de las series de tiempo es que las observaciones cercanas en el tiempo típicamente estarán correlacionadas. Esto es el valor x_t a menudo estará correlacionado con x_{t-1} , x_{t-2} , x_{t+1} . Por otra parte, un número importante de series de tiempo presentan tendencias. Y al hacer una regresión entre una variable con tendencia y otra variable con tendencia, las dos variables pueden parecer altamente correlacionadas, incluso cuando de hecho no están económicamente relacionadas. Tercero, cuando usamos datos de corte transversal, asumimos que la media muestral es un estimador de la media poblacional, pero en datos de series de tiempo puede que la media poblacional no exista.

V.1. Conceptos

Tal como en datos de series de tiempo podemos definir la media de la serie x_t como:

$$E(x_t) = \mu = \int_{-\infty}^{+\infty} x_t f(x_t) dx_t$$

Donde $f(x_t)$ representa la función de densidad de la variable x_t .

El análogo muestral de la media es:

$$\bar{x} = \frac{1}{T} \sum_{t=1}^T x_t$$

La varianza de la serie es:

$$E(x_t - \mu)^2 = \sigma^2 = \int_{-\infty}^{+\infty} (x_t - \mu)^2 f(x_t) dx_t$$

Y la varianza muestral es:

$$s^2 = \frac{1}{T} \sum_{t=1}^T (x_t - \bar{x})^2$$

Por otra parte, la covarianza entre dos variables x_t e y_t se define como:

$$\text{cov}(x_t, y_t) = E[(x_t - \mu_x) \cdot (y_t - \mu_y)]$$

Pero en series de tiempo la variable y_t puede ser considerada como la misma variable x en otro momento del tiempo, por ejemplo, x_{t+1} .

Entonces se puede definir la covarianza en t de la serie con su valor un periodo más adelante como:

$$\gamma_1 = \text{cov}(x_t, x_{t+1}) = E[(x_t - \mu_x) \cdot (x_{t+1} - \mu_x)]$$

Y en términos más generales, se puede definir la covarianza de la serie x en t con su valor j periodos de diferencia:

$$\gamma_j = \text{cov}(x_t, x_{t+j}) = E[(x_t - \mu_x) \cdot (x_{t+j} - \mu_x)]$$

La contraparte muestral de la covarianza γ_j es la siguiente:

$$\hat{\gamma}_j = \frac{1}{T} \sum_{t=1}^T (x_t - \bar{x})(x_{t+j} - \bar{x})$$

Finalmente, el coeficiente de correlación de la serie para valores de ellas distanciados j periodos es:

$$\rho_j = \frac{\gamma_j}{\gamma_0}$$

V.2. Estacionariedad

Una serie de tiempo es estacionaria, cuando su media, varianza y covarianza no depende de t . Para trabajar con las series de tiempo y estimar modelo se requiere que esta sea estacionaria. Si la serie no es estacionaria, es difícil hacer predicciones sobre su comportamiento futuro.

V.3. Procesos autoregresivos y de medias móviles**V.3.1. Procesos de media móvil**

Un proceso de media móvil de primer orden MA(1) esta descrito por la siguiente ecuación:

$$y_t = \mu + e_t + \theta e_{t-1}$$

Donde e_t es un ruido blanco, sigue una distribución $N(0,1)$.

La media de este proceso MA(1) es:

$$E(y_t) = \mu + E(e_t) + \theta E(e_{t-1}) = \mu$$

La varianza es igual a:

$$V(y_t) = E[(y_t - \mu)^2] = (1 + \theta^2)\sigma^2$$

La primera auto-covarianza es:

$$\gamma_1 = E[(y_t - \mu)(y_{t-1} - \mu)] = \theta\sigma^2$$

Y la auto-covarianza de segundo orden y superior son iguales a cero.

En términos más generales, la siguiente ecuación describe un proceso MA(q):

$$y_t = \mu + e_t + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \cdots + \theta_q e_{t-q}$$

El cual también tiene media igual a μ , y su varianza es igual a:

$$E[(y_t - \mu)^2] = \left(\sum_{i=0}^q \theta_i^2 \right) \sigma^2$$

Y las auto-covarianzas:

$$\gamma_1 = (\theta_1 + \theta_1\theta_2 + \theta_2\theta_3 + \cdots + \theta_{q-1}\theta_q)$$

$$\gamma_2 = (\theta_2 + \theta_1\theta_3 + \theta_2\theta_4 + \cdots + \theta_{q-2}\theta_q)$$

$$\vdots$$

$$\gamma_q = \theta_q\sigma^2$$

$$\gamma_m = 0 \quad \forall m > q$$

Dado que la media, varianza y auto-covarianzas son independientes de t , se puede concluir que un proceso $MA(q)$ es estacionario.

Si q tiende a infinito es decir, tenemos un proceso $MA(\infty)$ para que el proceso siga siendo estacionario se debe cumplir que:

$$\sum_{i=0}^{\infty} \theta_i^2 < \infty \quad o \quad \sum_{i=0}^{\infty} |\theta_i| < \infty$$

V.3.2. Proceso autoregresivo:

Un proceso autoregresivo de primer orden $AR(1)$ es de la forma:

$$y_t = \beta + \phi y_{t-1} + e_t$$

La media de este proceso estará dada por:

$$E(y_t) = \beta + \phi E(y_{t-1}) + E(e_t)$$

$$\mu = \beta + \phi \mu$$

$$\mu = \frac{\beta}{1 - \phi}$$

Para llegar a esto se asume que y_t es estacionaria.

¿Qué se requiere para que el proceso $AR(1)$ sea estacionario?

Partiendo de la ecuación inicial y reemplazando de manera recursiva se obtiene que el proceso $AR(1)$ puede ser escrito como un $MA(\infty)$:

$$y_t = \frac{\beta}{1 - \phi} + e_t + \phi e_{t-1} + \phi^2 e_{t-2} + \dots$$

Luego, la condición de estacionariedad se resume a que:

$$\sum_{i=0}^{\infty} |\phi^i| < \infty$$

Condición que se cumple siempre que $|\phi| < 1$.

Se puede demostrar que las autocorrelaciones de un proceso $AR(1)$ son:

$$\rho_j = \phi^j$$

En términos más generales, tenemos el siguiente proceso AR(p):

$$y_t = \beta + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \dots + \phi_p y_{t-p} + e_t$$

Podemos utilizar el operador de rezagos para escribir el proceso AR(p) de la siguiente manera:

$$(1 - \phi L - \phi^2 L^2 - \dots - \phi^p L^p) y_t = e_t$$

Para obtener el valor esperado de esta serie debemos encontrar la inversa de $(1 - \phi L - \phi^2 L^2 - \dots - \phi^p L^p)$. Si esta inversa existe, el proceso AR(p) se transforma en un proceso MA(∞) el cual será estacionario.

Si se encuentran $\lambda_1, \lambda_2, \lambda_3, \dots, \lambda_p$ tal que:

$$(1 - \phi L - \phi^2 L^2 - \dots - \phi^p L^p) = (1 - \lambda_1 L) \cdot (1 - \lambda_2 L) \cdots (1 - \lambda_p L)$$

Es posible escribir el proceso AR(p) de la siguiente manera:

$$y_t = \alpha + \frac{1}{(1 - \lambda_1 L)} \cdot \frac{1}{(1 - \lambda_2 L)} \cdots \frac{1}{(1 - \lambda_p L)} \cdot e_t$$

Luego, tendremos que:

$$\frac{1}{(1 - \lambda_i L)} = 1 + \lambda_i L + \lambda_i^2 L^2 + \dots$$

Siempre y cuando se cumpla que $|\lambda_i| < 1$. Es decir, se concluye que el proceso AR(p) es estacionario siempre y cuando $\lambda_1, \lambda_2, \lambda_3, \dots, \lambda_p$ sean menores a 1.

V.4. Procesos mixtos autoregresivos y de medias móviles ARMA(p,q)

El proceso ARMA(p,q) tiene la siguiente forma:

$$y_t = \mu + \phi_1 y_{t-1} + \dots + \phi_p y_{t-p} + e_t + \theta_1 e_{t-1} + \dots + \theta_q e_{t-q}$$

Será estacionario en la medida que las $\lambda_1, \lambda_2, \lambda_3, \dots, \lambda_p$ sean menores a 1.

La ventaja de los modelos ARMA es que nos permiten capturar la dinámica de los datos usando menor cantidad de parámetros que si utilizáramos un proceso AR puro o MA puro, lo que es particularmente ventajoso en datos de series de tiempo.

V.5. Función de autocorrelación muestra y autocorrelación parcial

Tal como notamos en las secciones previas, un factor distintivo entre los procesos AR y los MA es como los shocks afectan las futuras realizaciones de la serie. En un proceso MA(q) un shock en el periodo t no tiene efectos desde el periodo t+q+1 en adelante. Sin embargo, en un proceso AR(p)

el efecto de un shock en t decae de manera gradual en el tiempo. Este será un elemento que nos permitirá distinguir entre ambos procesos.

V.5.1. Simulación de procesos ARMA(p,q)

La mejor forma de familiarizarse con el mecanismo para determinar el orden de un proceso ARMA es primero concentrarnos en series que han sido simuladas por nosotros, de manera tal que tenemos conocimiento exacto del verdadero procesos que determina los datos.

Para esto utilizaremos el comando `sim_arma`, el que debe ser instalado a través de los siguientes comandos:

```
net from http://www.stata.com/users/jpitblado

-----
http://www.stata.com/users/jpitblado/
Materials by Jeff Pitblado, StataCorp
-----

Packages identified by (version #) use tools that are not available prior to
Stata #.

DIRECTORIES you could -net cd- to:
    ..                back to other contributors

PACKAGES you could -net describe-:
    boston05svytalk    Survey talk for the 2005 NASUG (version 9)
    .
    .
    .
    reg_ss             Sum of Squares Tables for regression (version 7)
    sim_arma           Simulate autoregressive moving average data (version 8)
    xtline2            modified version of xtline (version 10)
    ztest              Proportion comparison tests (version 7)
    zval              Standardize variables (version 7)
-----

net install sim_arma

checking sim_arma consistency and verifying not already installed...
installing into c:\ado\plus\...
installation complete.
```

Ahora, si queremos generar 100 observaciones para una serie que sigue el siguiente proceso:

$$y_t = 0.7y_{t-1} - 0.3y_{t-2} + e_t + 0.4e_{t-1} + 0.2e_{t-2} \quad e_t \sim N(0,9)$$

Se debe ejecutar el siguiente comando:

```
sim_arma y, ar(0.7 -0.3) ma(0.4 0.2) sigma(3) nobs(100)
```

Este comando genera una serie y con 100 observaciones, una variable $_t$ con el indicador de tiempo de la serie, y deja ya indicado el formato de serie de tiempo mediante el comando `tsset`.

V.5.2 Función de autocorrelación muestral:

En la sección anterior definimos la función de autocorrelación de orden j para los procesos MA y AR. Típicamente para inspeccionar los datos y determinar el proceso que sigue una serie a través de la función de autocorrelación muestral graficaremos las autocorrelaciones de orden $j=0,1,\dots,40$.

Primero, generemos una muestra de tamaño 1,000 de un proceso MA(1):

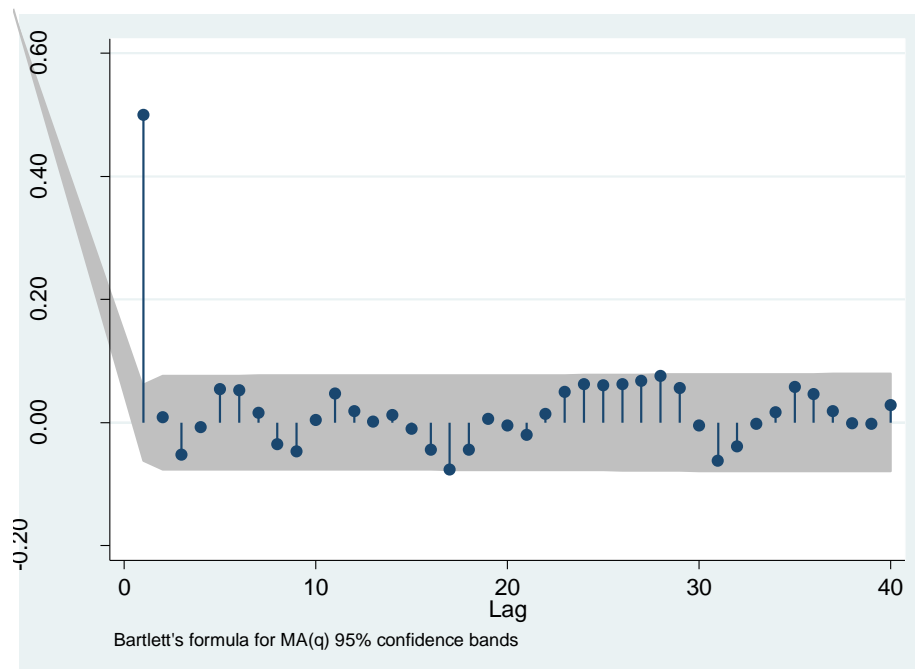
$$y_t = e_t + 0.7e_{t-1}$$

Donde e_t es un ruido blanco:

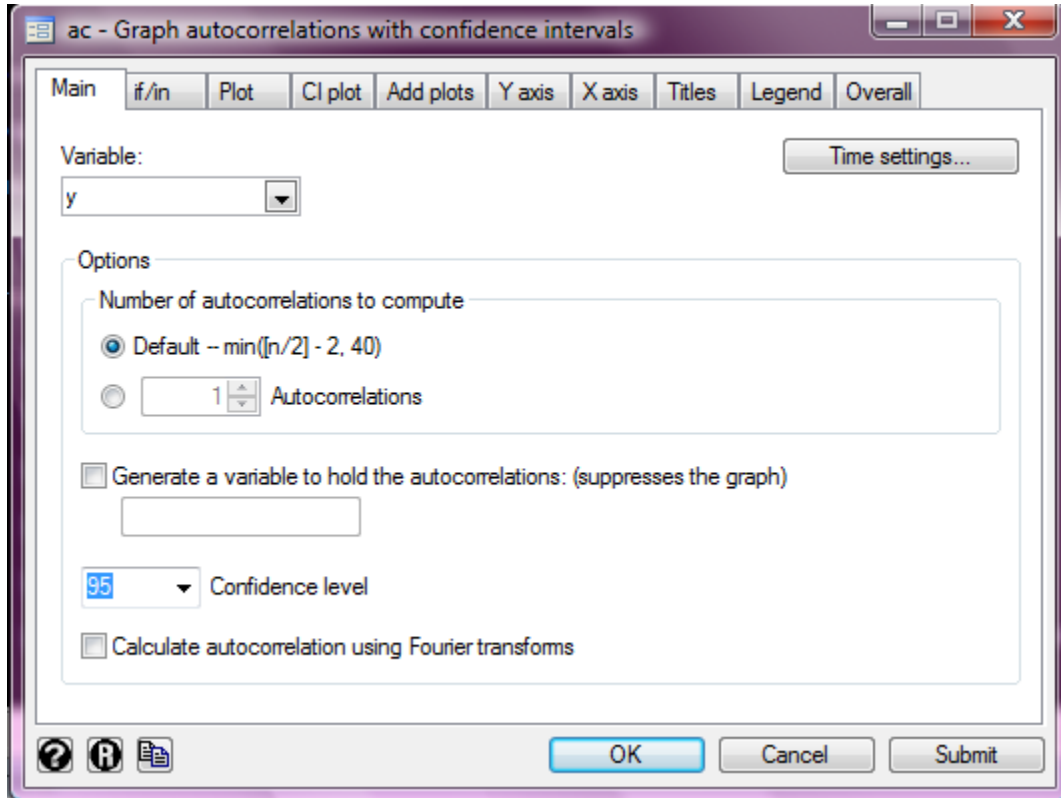
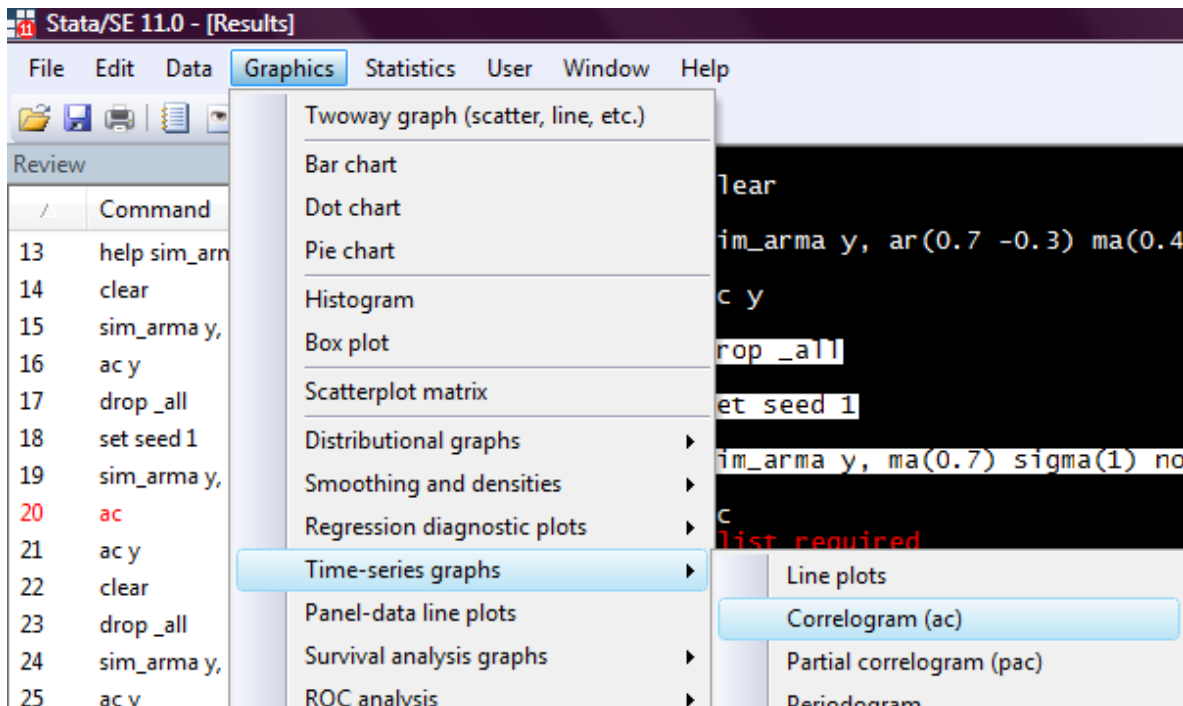
```
drop _all
sim_arma y, ma(0.7) sigma(1) nobs(1000)
```

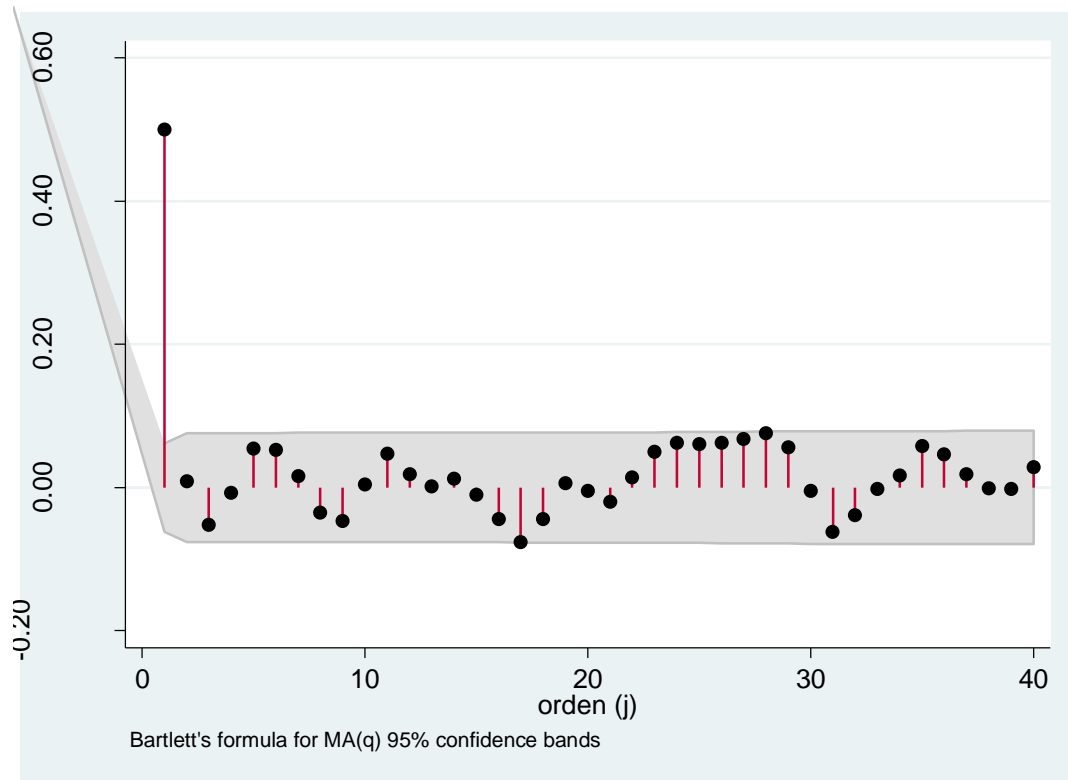
Para graficar las autocorrelaciones en STATA se puede utilizar el comando `ac`:

```
ac y
```



Podemos encontrar este gráfico dentro de las opciones de gráficos de series de tiempo, modificando colores, y títulos:



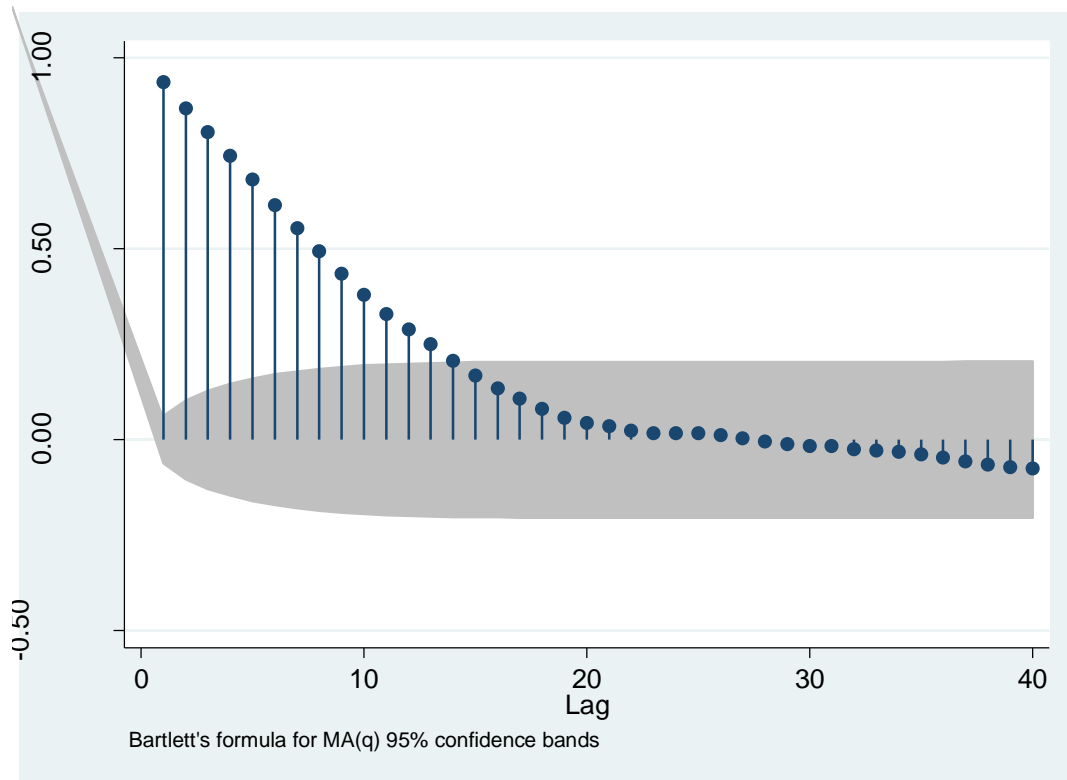


La autocorrelación de orden 1 es aproximadamente 0.47 que corresponde al valor teórico para el proceso generado. Las restantes autocorrelaciones no son estadísticamente diferentes de cero.

Ahora consideremos un proceso AR(1) de la forma:

$$y_t = 0.95y_{t-1} + e_t$$

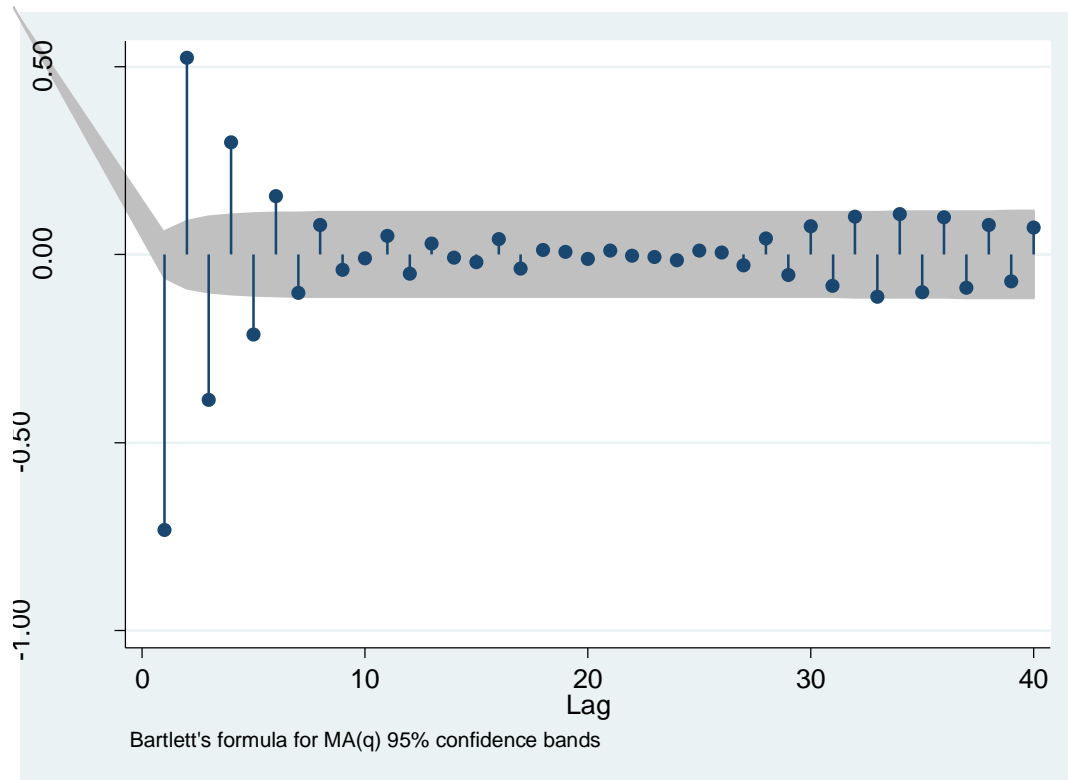
```
drop _all
sim_arma y, ar(0.95) sigma(1) spin(5000) time(t) nobs(1000)
ac y
```



Lo que se observa es que un shock no sólo afecta un periodo como en los procesos MA, sino que el efecto perdura en el tiempo, siendo cada vez menor.

Ahora veamos que sucede si el coeficiente autoregresivo es negativo:

```
drop _all
sim_arma y, ar(-0.75) sigma(1) spin(5000) time(t) nobs(1000)
ac y
```



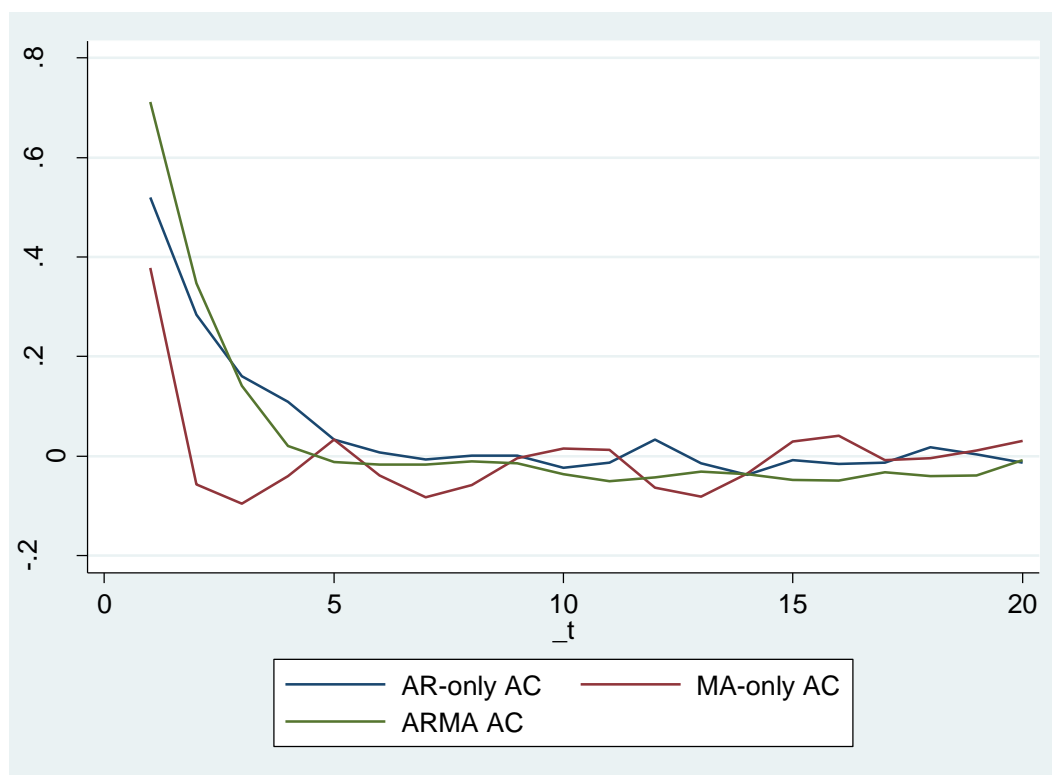
El efecto de un shock decae pero de manera oscilante.

Veamos ahora que sucede con un proceso ARMA, y como se compara con los procesos AR y MA puros. Suponga el siguiente proceso ARMA(1,1):

$$y_t = 0.5y_{t-1} + e_t + 0.5e_{t-1} \quad e_t \sim N(0,1)$$

```
drop _all
set seed 1
sim_arma arvar, ar(0.5) spin(2000) nobs(1000)
sim_arma mavar, ma(0.5) spin(2000) nobs(1000)
sim_arma armavar, ar(0.5) ma(0.5) spin(2000) nobs(1000)
ac arvar, gen(arac)
ac mavar, gen(maac)
ac armavar, gen(armaac)
label variable arac "AR-only AC"
label variable maac "MA-only AC"
label variable armaac "ARMA AC"

tsline arac maac armaac in 1/20
```



La función de autocorrelación del proceso ARMA comienza con valores más altos, pero decae de manera más rápida que el proceso AR. Se puede notar que la función de autocorrelación del proceso AR(1) decae geométricamente ($0.5, 0.5^2, 0.5^3, \dots$), en cambio, en el proceso ARMA la primera autocorrelación es cercana a 0.7, y luego van disminuyendo a una tasa más alta que la geométrica.

V.5.3 Función de autocorrelación parcial:

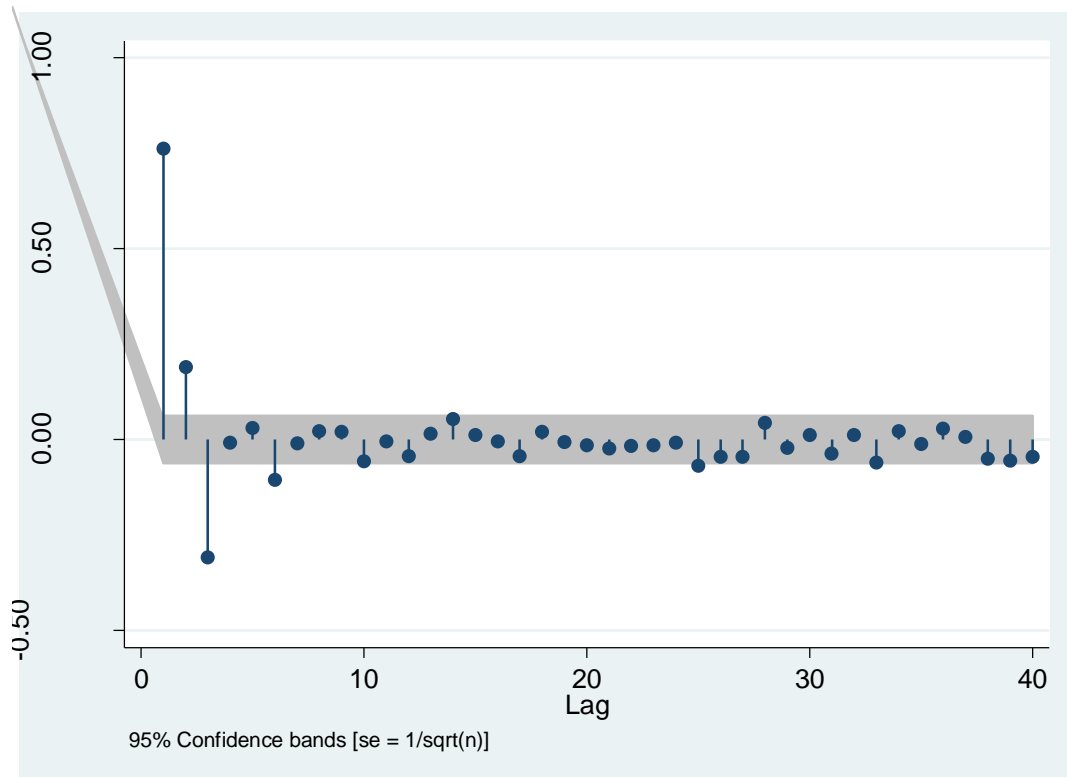
Pudimos notar que la función de autocorrelación muestral de un proceso autoregresivo decae de manera gradual, pero esta función no es capaz de darnos señales sobre el orden de este proceso, es decir, si es un AR(1) o un AR(5).

Para esto se utiliza la función de autocorrelación parcial, la idea es que si una serie sigue un proceso AR(p) la autocorrelación parcial p+1 y superior debiese ser cero. La función de autocorrelación parcial mide la correlación entre y_t e y_{t+j} después de controlar por el efecto de $y_{t+1}, y_{t+2}, \dots, y_{t+j-1}$. Desde la perspectiva del análisis de regresión, las autocorrelaciones parciales $\phi_{11}, \phi_{22}, \dots, \phi_{jj}$ son los coeficientes en la siguiente ecuación:

$$(y_t - \mu) = \phi_{11}(y_{t-1} - \mu) + \phi_{22}(y_{t-2} - \mu) + \dots + \phi_{jj}(y_{t-j} - \mu) + u_t$$

El comando en STATA para obtener la función de autocorrelación parcial es `pac`.

```
drop _all  
set seed 1  
sim_arma y, ar(0.7 0.4 -0.3) spin(5000) nobs(1000)  
  
pac y
```



VI. Predicción: Modelos ARIMA y ARIMAX

En la sección previa aprendimos sobre los procesos AR, MA y ARMA. También aprendimos sobre la función de autocorrelación muestral (ac) y parcial (pac) y como usarlas para determinar el orden del proceso ARMA.

Una vez decidido el orden del proceso ARMA, es necesario estimar los parámetros involucrados para de esta manera hacer predicciones sobre la serie de interés. De esta forma, estamos **asumiendo** que la serie que nos interesa puede ser representada como un proceso ARMA.

VI.1. Ideas básicas

Recordemos que un proceso ARMA(1,1) puede ser escrito de la siguiente manera:

$$y_t = \alpha + \phi_1 y_{t-1} + e_t + \theta_1 e_{t-1}$$

Donde e_t es un ruido blanco.

Este modelo puede ser escrito a través de las siguientes ecuaciones:

$$y_t = \beta_0 + \mu_t$$

$$\mu_t = \phi_1 \mu_{t-1} + e_t + \theta_1 e_{t-1}$$

La primera ecuación se denomina ecuación estructural, y la segunda ecuación del error. En este sentido, estamos diciendo que y_t es igual a cierto nivel (β_0) más un error con media cero, el que puede ser descrito como un proceso ARMA.

Si asumimos que e_t se distribuye normal, podemos utilizar el método de máxima verosimilitud para estimar los parámetros de ambas ecuaciones, en STATA esto se hace a través del comando `arima`.

VI.2. Estacionariedad

La letra i en el proceso ARIMA indica el orden de integración de la serie, es decir, cuantas veces esta debe ser diferenciada para que sea una serie estacionaria.

VI.2.1 Test de raíz unitaria

Cuando revisamos los procesos AR(p) se estableció como condición para que el proceso fuera estacionario, que las raíces del polinomio de rezagos debería ser menores a 1. A continuación presentaremos test de hipótesis que nos permiten establecer si la serie presenta o no raíz unitaria, es decir, si es o no estacionaria.

Pensemos en un proceso AR(1):

$$y_t = \phi y_{t-1} + e_t$$

La serie no será estacionaria, o tendrá raíz unitaria, si es que $\phi=1$. Entonces, se podría testear la hipótesis de no estacionariedad simplemente planteado un test de hipótesis simple para este parámetro. Sin embargo, bajo el cumplimiento de la hipótesis nula la distribución del estadístico no sigue la distribución t conocida.

El proceso AR(1) se puede expresar de la siguiente forma:

$$\Delta y_t = (\rho - 1)y_{t-1} + e_t$$

$$\Delta y_t = \alpha y_{t-1} + e_t$$

El **test de Dickey-Fuller (1979)** es uno de los más utilizados para testear la hipótesis nula de raíz unitaria en la serie, es decir, $H_0: \alpha = 0$.

Si rechazamos para hipótesis nula, podemos decir que la serie es estacionaria.

Existen tres versiones de este test:

- ❖ Test de raíz unitaria (paseo aleatorio):

$$\Delta y_t = \alpha y_{t-1} + e_t$$

- ❖ Test de raíz unitaria con drift (paseo aleatorio con constante):

$$\Delta y_t = \delta + \alpha y_{t-1} + e_t$$

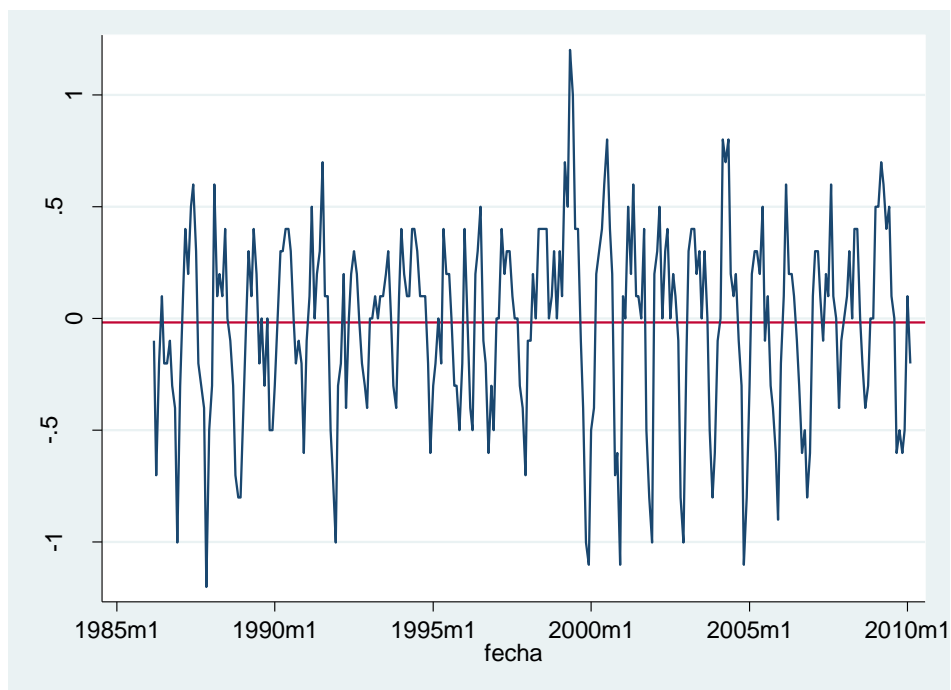
- ❖ Test de raíz unitaria con drift y tendencia determinística (paseo aleatorio con constante y tendencia):

$$\Delta y_t = \delta_0 + \delta_1 t + \alpha y_{t-1} + e_t$$

Para cada una de estas versiones del test los autores desarrollaron mediante simulación los valores críticos que permiten concluir sobre la hipótesis nula.

Por ejemplo, estudiemos la estacionariedad de la serie desempleo. Primero, grafiquemos la serie en diferencia para ver cuál de las tres versiones del test es apropiada utilizar:

```
use "desempleo.dta", clear
tsset fecha
sum D.desempleo
tsline D.desempleo, yline(`r(mean)')
```



La serie en diferencia se mueve en torno a la media, sin tendencia. Por lo cual deberíamos utilizar la primera o segunda versión del tests, sin tendencia. Con respecto a la constante, no esta claro si fluctúa en torno a cero o un valor inferior a cero, por lo cual vamos a utilizar la segunda versión del test que incluye constante o drift.

The screenshot shows the Stata 'Statistics' menu with 'Tests' selected. The main window displays the results of the Augmented Dickey-Fuller unit-root test for the variable 'desempleo'.

Test results for 'desempleo':

Test Statistic	1% Critical Value	5% Critical Value	10% Critical Value
-3.024	-3.457	-2.878	-2.570

Number of obs = 288

Interpolated Dickey-Fuller

P>|t| [95% Conf. Interval]

0.003	-.0676473	-.014306
0.005	.09976	.5640271

Augmented Dickey-Fuller unit-root test

DF-GLS test for a unit root

The screenshot shows the 'dfuller - Augmented Dickey-Fuller unit-roots test' dialog box. The 'Variable' is set to 'desempleo'. The 'Options' section includes:

- ☐ Suppress constant term in regression
- ☐ Include trend term in regression
- ☒ Include drift term in regression
- ☒ Display regression table
- Lagged differences: 0

Buttons: OK, Cancel, Submit

El comando ejecutado es el siguiente:

```
use "desempleo.dta", clear
tsset fecha
dfuller desempleo, drift regress lags(0)
```

Dickey-Fuller test for unit root Number of obs = 288

	Test Statistic	----- 1% Critical Value	Z(t) has t-distribution 5% Critical Value	----- 10% Critical Value
Z(t)	-3.024	-2.339	-1.650	-1.285

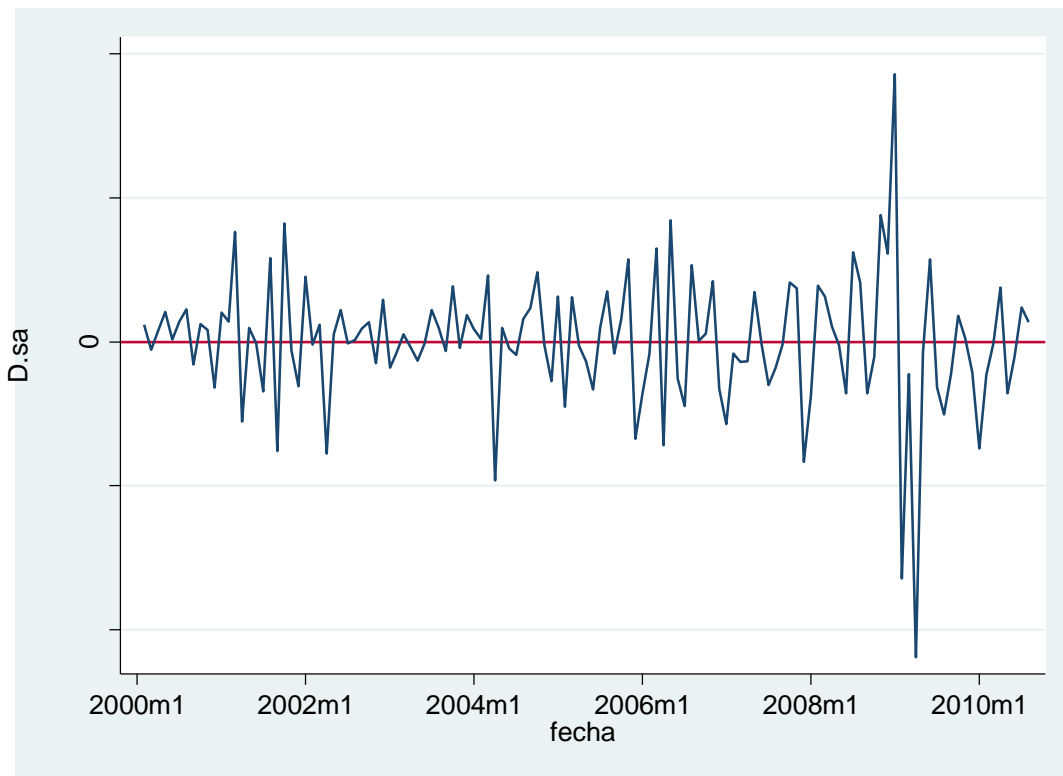
p-value for Z(t) = 0.0014

D.desempleo	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]
desempleo					
L1.	-.0409766	.0135501	-3.02	0.003	-.0676473 -.014306
_cons	.3318935	.1179364	2.81	0.005	.09976 .5640271

Observamos que el coeficiente que acompaña al rezago de la variable desempleo es estadísticamente diferente de cero, el p-value asociado a su test de significancia individual es menor a 0.05. Sin embargo, para este test no se puede utilizar la distribución t-student para hacer inferencia. La primera tabla muestra el estadístico calculado, los valores críticos, y el p-value. De esto podemos concluir que se rechaza la hipótesis de raíz unitaria en la serie desempleo.

Ahora veamos si la serie cosecha de salmones es estacionaria:

```
use "sa.dta", clear
sum D.sa
tsline D.sa, yline(`r(mean)')
```



Nuevamente, utilizaremos la segunda versión del test, ya que no está claro si tiene algún desvío (drift) de ruido blanco. Lo que si está claro, que la serie en primera diferencia no presenta tendencia.

```
dfuller sa, regress drift lags(0)
```

Dickey-Fuller test for unit root Number of obs = 127

	Test Statistic	----- Z(t) has t-distribution ----- 1% Critical Value	5% Critical Value	10% Critical Value
Z(t)	-3.165	-2.357	-1.657	-1.288

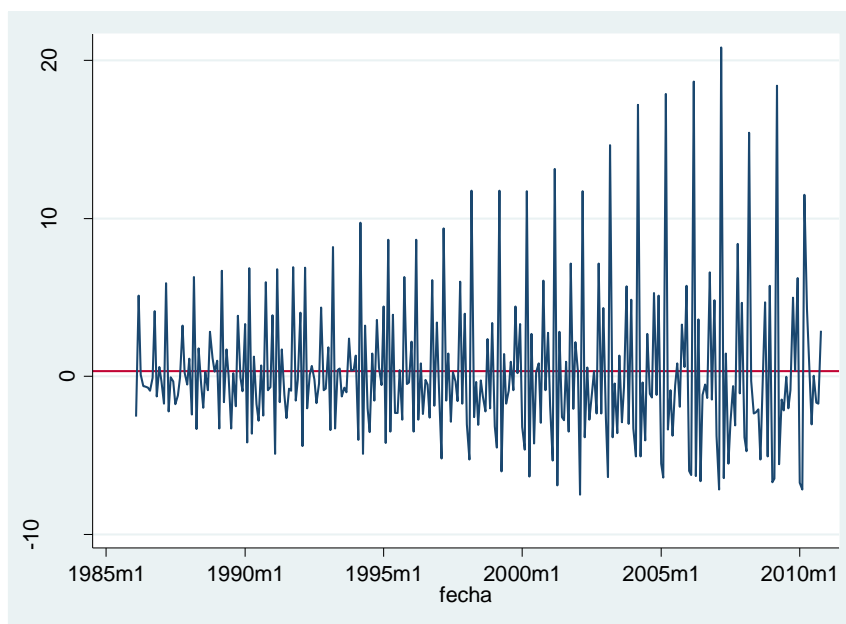
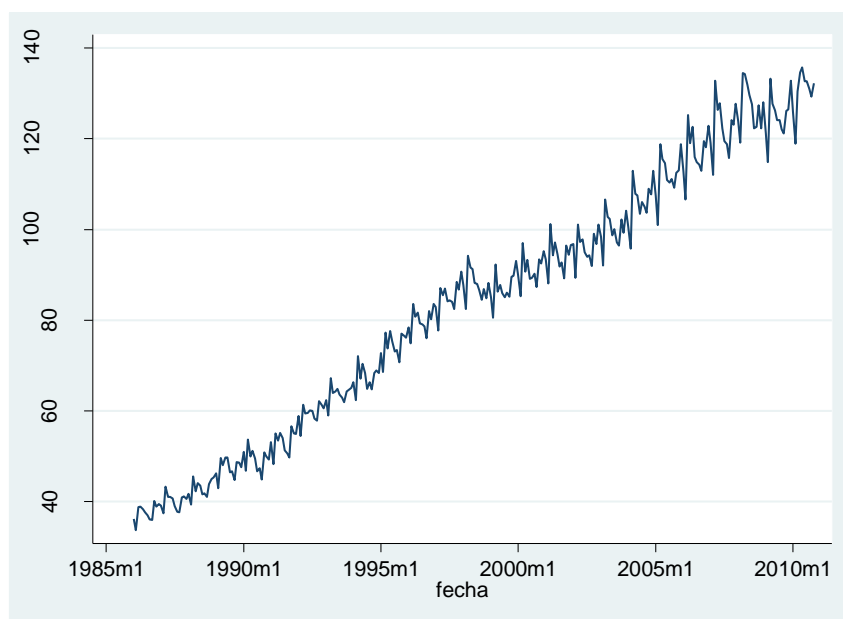
p-value for Z(t) = 0.0010

D.sa	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]
sa					
L1.	-.149419	.0472053	-3.17	0.002	-.2428441 - .0559939
_cons	3583.703	1200.686	2.98	0.003	1207.396 5960.011

En este caso, también se rechaza la hipótesis nula de que la serie cosecha de salmónes tenga una raíz unitaria, y podemos afirmar que la serie es estacionaria.

Por último, estudiemos la estacionariedad de la serie imacec:

```
use "imacec.dta", clear  
tsline imacec  
  
sum D.imacec  
tsline D.imacec, yline(`r(mean)')
```



Tal como habíamos notado sesiones anteriores, esta serie presenta una clara tendencia positiva, lo que hace que la serie se vea como no estacionaria, sin embargo, esta serie probablemente no tenga raíz unitaria, y el efecto de la no estacionariedad es simplemente por el hecho de tener una tendencia determinística. De esta forma, se debe distinguir entre series estacionarias en diferencia y estacionaria en tendencia. En el primer caso es necesario diferenciar la serie para que esta se vuelva una serie estacionaria, en el segundo caso se debe restar el componente tendencial de la serie para volverla estacionaria.

En este caso aplicaremos el tercer test de raíz unitaria, que considera la presencia de una tendencia determinística en la serie:

dfuller imacec, regress trend lags(0)

Dickey-Fuller test for unit root

Number of obs = 297

Test Statistic	----- 1% Critical Value	Interpolated Dickey-Fuller 5% Critical Value	----- 10% Critical Value
Z(t)	-11.783	-3.988	-3.428
			-3.130

MacKinnon approximate p-value for Z(t) = 0.0000

D.imacec	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]
imacec					
L1.	-.6411589	.0544149	-11.78	0.000	-.748251 - .5340667
_trend	.2132664	.018285	11.66	0.000	.1772803 .2492525
_cons	21.99825	1.896173	11.60	0.000	18.26645 25.73004

En este caso, se rechaza la hipótesis nula de que la serie imacec tenga raíz unitaria (una vez que hemos controlado por la tendencia presente en la serie).

Notemos que si hacemos el test de raíz unitaria sin considerar la tendencia, no es posible rechazar la hipótesis nula de raíz unitaria al 5% de significancia.

```
dfuller imacec, regress drift lags(0)
```

Dickey-Fuller test for unit root Number of obs = 297

	Test Statistic	----- 1% Critical Value	----- 5% Critical Value	----- 10% Critical Value
Z(t)	-1.386	-2.339	-1.650	-1.284

p-value for Z(t) = 0.0835

```
dfuller imacec, regress nocon lags(0)
```

Dickey-Fuller test for unit root Number of obs = 297

	Test Statistic	----- 1% Critical Value	----- 5% Critical Value	----- 10% Critical Value
Z(t)	0.660	-2.580	-1.950	-1.620

	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]
imacec					
L1.	-.0131869	.0095172	-1.39	0.167	-.0319171 .0055434
_cons	1.422699	.8394634	1.69	0.091	-.2293968 3.074795

	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]
imacec					
L1.	.002058	.0031184	0.66	0.510	-.004079 .008195

El test de Dickey-Fuller original está pensado para un proceso autoregresivo de orden superior se utiliza el **test Dickey Fuller Aumentado (ADF)**:

$$\Delta y_t = \delta_0 + \delta_1 t + \alpha y_{t-1} + \delta_1 \Delta y_{t-1} + \delta_2 \Delta y_{t-2} + \dots + \delta_p \Delta y_{t-p} + e_t$$

La cantidad de rezagos a considerar se puede escoger de manera óptima según criterios de información.

El **test de Phillips y Perron (1988)**, bastante popular en series de tiempo financieras, difiere del test ADF en la forma que se considera la presencia de heterocedasticidad y/o autocorrelación de orden superior. En particular el test ADF trata de incorporar la cantidad de rezagos necesaria para modelar este comportamiento, el test PP no incorpora rezagos sino que estima el modelo considerando la presencia de esto en el error.

Por ejemplo, para la serie desempleo:

```

use "desempleo.dta", clear

tsset fecha
      time variable:  fecha, 1986m2 to 2010m2
      delta: 1 month

pperron desempleo, regress

Phillips-Perron test for unit root                                Number of obs   =      288
                                                                Newey-West lags =       5

----- Interpolated Dickey-Fuller -----
              Test              1% Critical    5% Critical    10% Critical
              Statistic         Value          Value          Value
-----+-----+-----+-----+-----+-----+-----+-----+-----+
Z(rho)        -20.676          -20.330         -14.000         -11.200
Z(t)          -3.599           -3.457          -2.878          -2.570
-----+-----+-----+-----+-----+-----+
MacKinnon approximate p-value for Z(t) = 0.0058

-----+-----+-----+-----+-----+-----+-----+-----+-----+
desempleo |      Coef.   Std. Err.      t    P>|t|    [95% Conf. Interval]
-----+-----+-----+-----+-----+-----+-----+-----+
desempleo |
   L1. |   .9590234   .0135501    70.78   0.000    .9323527    .985694
   |
   _cons |   .3318935   .1179364     2.81   0.005    .09976    .5640271
-----+-----+-----+-----+-----+-----+-----+-----+

```

VI.3. Estimación de modelos ARIMA

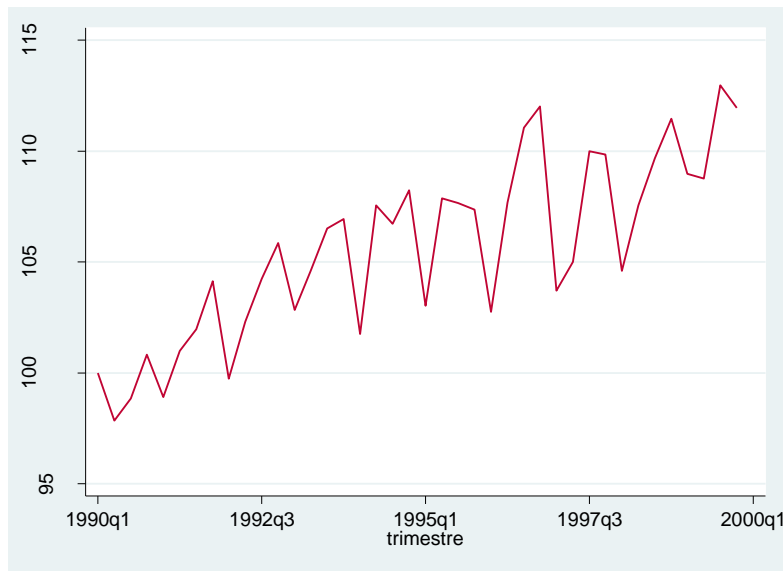
Utilicemos la base de datos turksales.dta, que contiene información trimestral de la venta de pavo en los noventas.

```

use "turksales.dta", clear
tsset t
twoway (tsline sales, lcolor(cranberry)), ytitle(venta de pavo)
ttitle(trimestre)

```

December 31, 2010



La serie claramente presenta una tendencia positiva, veremos si controlando por la tendencia la serie resulta ser estacionaria o no:

```
pperron sales, trend regress
```

Phillips-Perron test for unit root

Number of obs = 39

Newey-West lags = 3

		----- Interpolated Dickey-Fuller -----		
	Test Statistic	1% Critical Value	5% Critical Value	10% Critical Value
Z (rho)	-30.626	-24.292	-18.964	-16.272
Z (t)	-6.024	-4.251	-3.544	-3.206

MacKinnon approximate p-value for Z(t) = 0.0000

sales		Coef.	Std. Err.	t	P> t	[95% Conf. Interval]
sales						
l1.		.0127974	.1669043	0.08	0.939	-.3257003 .3512951
_trend		.2824702	.0582066	4.85	0.000	.1644218 .4005187
_cons		98.76293	16.67383	5.92	0.000	64.94685 132.579

Tenemos que la serie es estacionaria en tendencia, ya que al controlar por una tendencia se rechaza la hipótesis nula de que la serie tenga raíz unitaria.

Podemos quitar la tendencia de la serie, primero haciendo una regresión lineal, y luego tomando la diferencia entre la serie y el valor predicho (residuo):

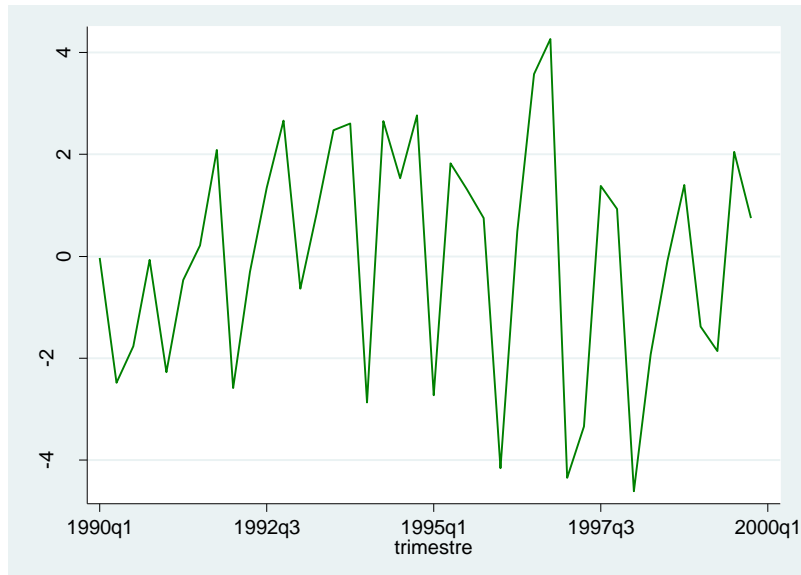
December 31, 2010

```
reg sales t
```

Source	SS	df	MS	Number of obs =	40
Model	436.707665	1	436.707665	F(1, 38) =	80.88
Residual	205.190671	38	5.39975449	Prob > F =	0.0000
Total	641.898336	39	16.4589317	R-squared =	0.6803
				Adj R-squared =	0.6719
				Root MSE =	2.3237

sales	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]
t	.286241	.031829	8.99	0.000	.2218065 .3506755
_cons	65.68723	4.455325	14.74	0.000	56.6679 74.70656

```
predict sales_t, resid
tsline sales_t
```



Podemos aplicar el test de raíz unitaria PP a esta serie:

```
pperron sales_t
```

Phillips-Perron test for unit root

Number of obs = 39
Newey-West lags = 3

----- Interpolated Dickey-Fuller -----				
Test Statistic	1% Critical Value	5% Critical Value	10% Critical Value	
Z(rho)	-30.630	-18.152	-12.948	-10.480
Z(t)	-6.130	-3.655	-2.961	-2.613

MacKinnon approximate p-value for Z(t) = 0.0000

Tenemos entonces, que para la serie sin tendencia se rechaza la hipótesis nula de raíz unitaria.

En caso de que la serie sea estacionaria en tendencia, como la que acabamos de analizar, se puede eliminar la tendencia también diferenciando la serie, no se genera error alguno. Por lo cual, muchos investigadores en vez de testear si la serie es estacionaria en diferencia o en tendencia, simplemente testean si la serie es estacionaria, y de no serlo toman primera diferencia para logarlo.

En el caso de las ventas de pavo, podríamos haber testeado simplemente la presencia de raíz unitaria, y si es que no se puede rechazar la hipótesis tomar primera diferencia de la serie para lograr la estacionariedad, en este caso la serie es integrada de orden 1:

```
pperron sales
```

Phillips-Perron test for unit root				Number of obs	=	39
				Newey-West lags	=	3
----- Interpolated Dickey-Fuller -----						
	Test	1% Critical	5% Critical	10% Critical		
	Statistic	Value	Value	Value		

Z (rho)	-7.784	-18.152	-12.948	-10.480		
Z (t)	-2.160	-3.655	-2.961	-2.613		

MacKinnon approximate p-value for Z(t) = 0.2212						

```
pperron D.sales
```

Phillips-Perron test for unit root				Number of obs	=	38
				Newey-West lags	=	3
----- Interpolated Dickey-Fuller -----						
	Test	1% Critical	5% Critical	10% Critical		
	Statistic	Value	Value	Value		

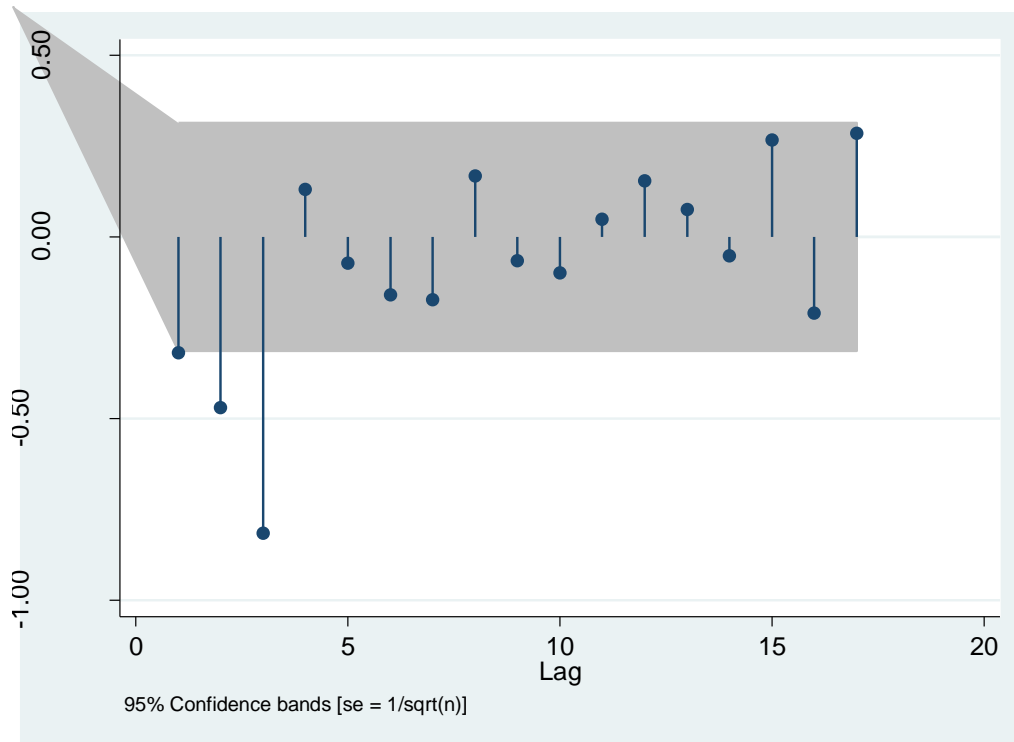
Z (rho)	-35.020	-18.084	-12.916	-10.460		
Z (t)	-17.707	-3.662	-2.964	-2.614		

MacKinnon approximate p-value for Z(t) = 0.0000						

Podemos apreciar que la serie en nivel (y sin controlar por tendencia determinística) tiene raíz unitaria, sin embargo, la primera diferencia de la serie es estacionaria. Por lo cual, podemos concluir que la serie sales es integrada de orden 1.

Ahora estudiemos la función de autocorrelación parcial para determinar el orden del proceso ARIMA:

```
pac D.sales
```



Se encuentra que las primeras tres autocorrelaciones parciales son estadísticamente diferentes de cero, indicando que al menos debemos considerar un proceso AR(3).

Mediante el comando `arima` podemos obtener los parámetros estimados por máxima verosimilitud de cualquier proceso ARIMA. Por ejemplo,

```
arima sales, arima(3,1,0)

ARIMA regression

Sample: 1990q2 - 1999q4                Number of obs   =      39
Wald chi2(3)                          =    125.92
Log likelihood = -75.53558              Prob > chi2      =    0.0000

-----+-----
      D.sales |               OPG
           Coef.   Std. Err.      z    P>|z|    [95% Conf. Interval]
-----+-----
sales
   _cons |   .3138001   .0780852    4.02   0.000    .1607558    .4668444
-----+-----
ARMA
      ar |
   L1. |  -.8190622   .0876331   -9.35   0.000   -1.000199   -.6473045
   L2. |  -.8174304   .1006355   -8.12   0.000   -1.014672   -.6201883
   L3. |  -.7738966   .0902346   -8.58   0.000   -.9507532   -.59704
-----+-----
      /sigma |   1.608579   .2642292    6.09   0.000    1.090699    2.126459
-----+-----
```

Hemos estimado un proceso AR(3) pero para la serie en primera diferencia, ya que hemos indicado que la serie es integrada de orden 1.

Hemos estimado la siguiente ecuación estructural y ecuación de los residuos:

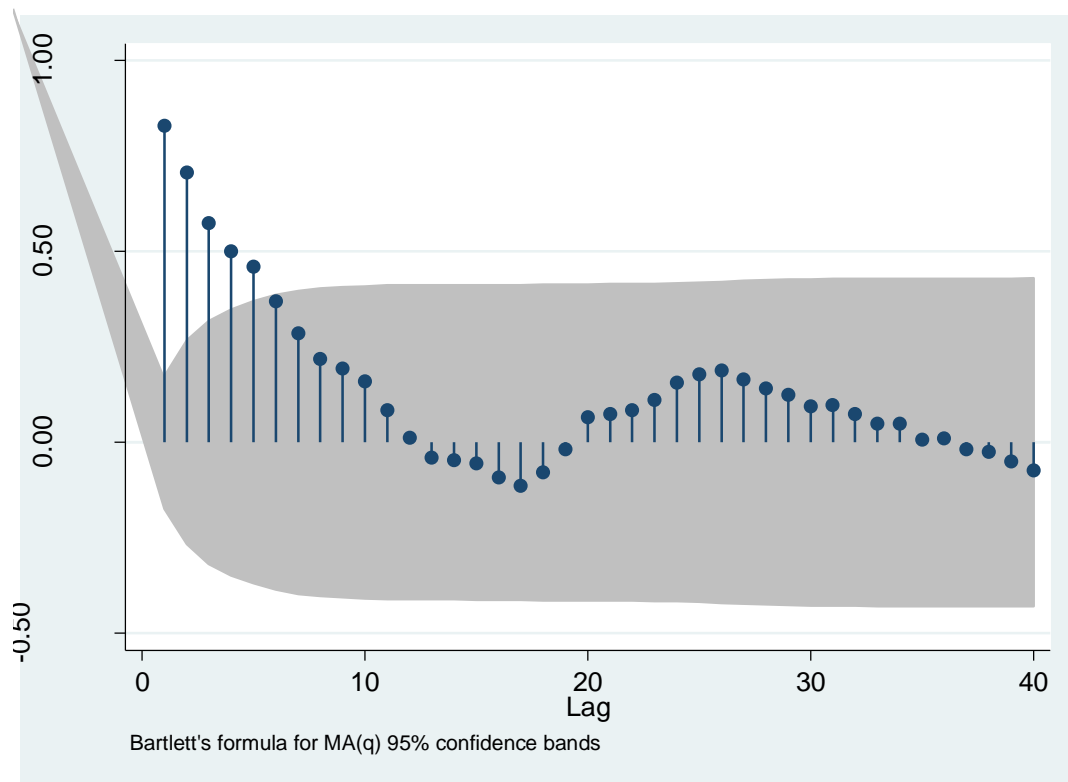
$$\Delta y_t = 0.314 + \mu_t$$

$$\mu_t = -0.819\mu_{t-1} - 0.817\mu_{t-2} - 0.774\mu_{t-3} + e_t$$

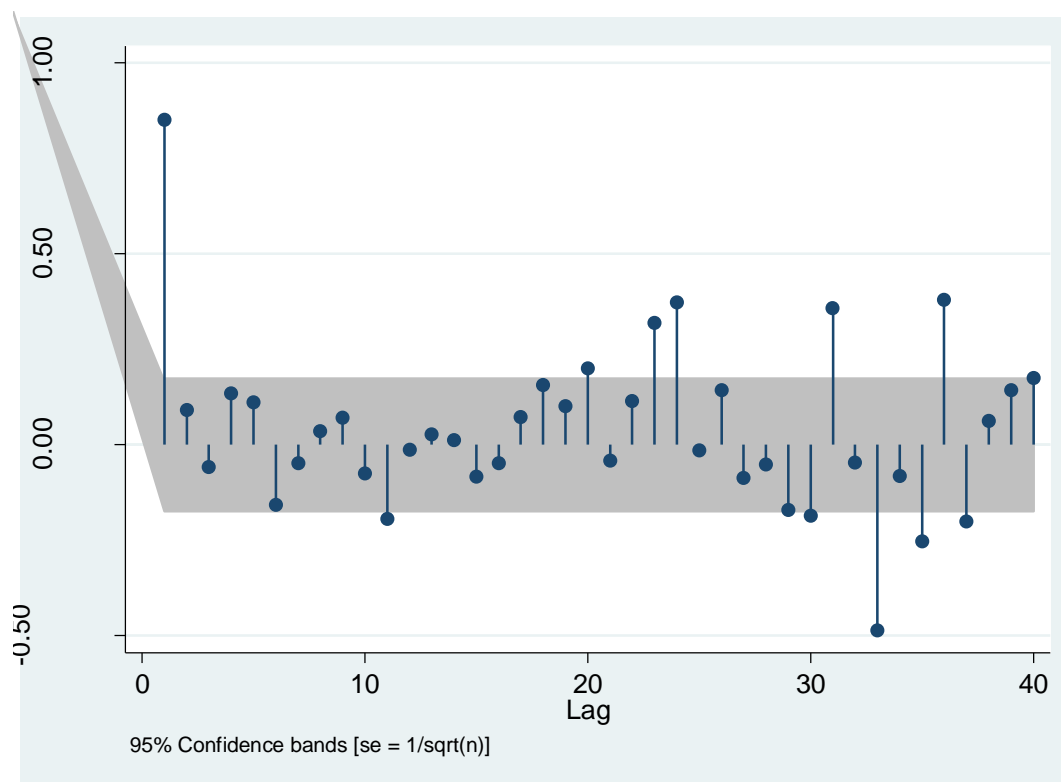
Ahora estimemos el proceso ARIMA para la serie cosecha de salmón. Ya habíamos testado la estacionariedad de la serie, rechazando la presencia de raíz unitaria, por lo cual no es necesario diferenciar la serie.

Podemos ver el gráfico con la función de autocorrelación muestral y la función de autocorrelación parcial:

ac sa



pac sa



También mediante el comando `corrgram` podemos visualizar ambas funciones:

```
. corrgram sa, lags(20)
```

LAG	AC	PAC	Q	Prob>Q	-1 [Autocorrelation]	0	1 [Partial Autocor]
1	0.8301	0.8506	90.294	0.0000	-----		-----
2	0.7072	0.0914	156.35	0.0000	-----		
3	0.5749	-0.0583	200.35	0.0000	-----		
4	0.5008	0.1346	234	0.0000	-----		
5	0.4595	0.1120	262.56	0.0000	-----		
6	0.3701	-0.1571	281.25	0.0000	-----		
7	0.2862	-0.0477	292.51	0.0000	-----		
8	0.2178	0.0350	299.09	0.0000	-----		
9	0.1932	0.0717	304.31	0.0000	-----		
10	0.1605	-0.0752	307.94	0.0000	-----		
11	0.0848	-0.1939	308.96	0.0000			
12	0.0122	-0.0131	308.98	0.0000			
13	-0.0392	0.0277	309.21	0.0000			
14	-0.0468	0.0115	309.53	0.0000			
15	-0.0540	-0.0829	309.96	0.0000			
16	-0.0924	-0.0479	311.22	0.0000			
17	-0.1140	0.0732	313.17	0.0000			
18	-0.0783	0.1562	314.1	0.0000			
19	-0.0172	0.1013	314.15	0.0000			
20	0.0667	0.2005	314.83	0.0000			

Las funciones indican que es un proceso AR(1). A continuación estimaremos un proceso ARIMA(1,0,0).

```

arima sa, ar(1)

ARIMA regression

Sample: 2000m1 - 2010m8      Number of obs   =      128
                             Wald chi2(1)         =      846.83
Log likelihood = -1257.472    Prob > chi2      =      0.0000

```

	sa	Coef.	OPG Std. Err.	z	P> z	[95% Conf. Interval]
sa						
_cons		22605.33	3116.448	7.25	0.000	16497.2 28713.45
ARMA						
ar						
L1.		.8629416	.029654	29.10	0.000	.8048208 .9210624
/sigma		4445.791	144.0753	30.86	0.000	4163.409 4728.174

Podemos distinguir entre dos sintaxis para la estimación de estos modelos:

❖ **Modelo ARMA:**

```
arima depvar, ar(numlist) ma(numlist)
```

❖ **Modelo ARIMA:**

```
arima depvar, arima(#p,#d,#q)
```

La primera opción nos permite indicar de manera precisa que rezagos incluir en el componente AR y MA.

VI.4. Predicción

VI.4.1. Predicción un paso adelante

Una vez estimado el modelo ARIMA podemos utilizar el comando `predict` para hacer predicciones.

Por ejemplo, volvamos a la estimación del modelo ARIMA(3,1,0) para la variable venta de pavos:

```

use " turksales.dta", clear
tsset t
arima sales, arima(3,1,0)

ARIMA regression

Sample: 1990q2 - 1999q4      Number of obs      =      39
                             Wald chi2(3)           =     125.92
Log likelihood = -75.53558   Prob > chi2       =      0.0000

-----+-----
      D.sales |               OPG               z   P>|z|   [95% Conf. Interval]
-----+-----
sales        |
  _cons      |   .3138001   .0780852   4.02   0.000   .1607558   .4668444
-----+-----
ARMA          |
  ar         |
    L1.      |  -.8190622   .0876331  -9.35   0.000   -.9908199  -.6473045
    L2.      |  -.8174304   .1006355  -8.12   0.000   -1.014672  -.6201883
    L3.      |  -.7738966   .0902346  -8.58   0.000   -.9507532  -.59704
-----+-----
  /sigma     |   1.608579   .2642292   6.09   0.000   1.090699   2.126459
-----+-----

predict y1

list sales y1 if _n<=10

+-----+
|      sales      y1 |
+-----+
1. |      100      .3138001 |
2. |  97.84603      .3138001 |
3. |  98.84029      1.100824 |
4. | 100.8275      1.126875 |
5. |  98.90981      .2967442 |
+-----+
6. | 100.9992      .2470322 |
7. | 101.9653     -.6115102 |
8. | 104.1229      .0550769 |
9. |  99.74297     -3.103686 |
10. | 102.3116      2.146297 |
+-----+

```

El comando realiza una predicción sobre la variable dependiente del modelo recién estimado, en este ejemplo como la serie era estacionaria en diferencia, la variable dependiente no es la serie en nivel sino la serie en diferencia. Si queremos hacer una predicción sobre la variable original, debemos hacer lo siguiente:

December 31, 2010

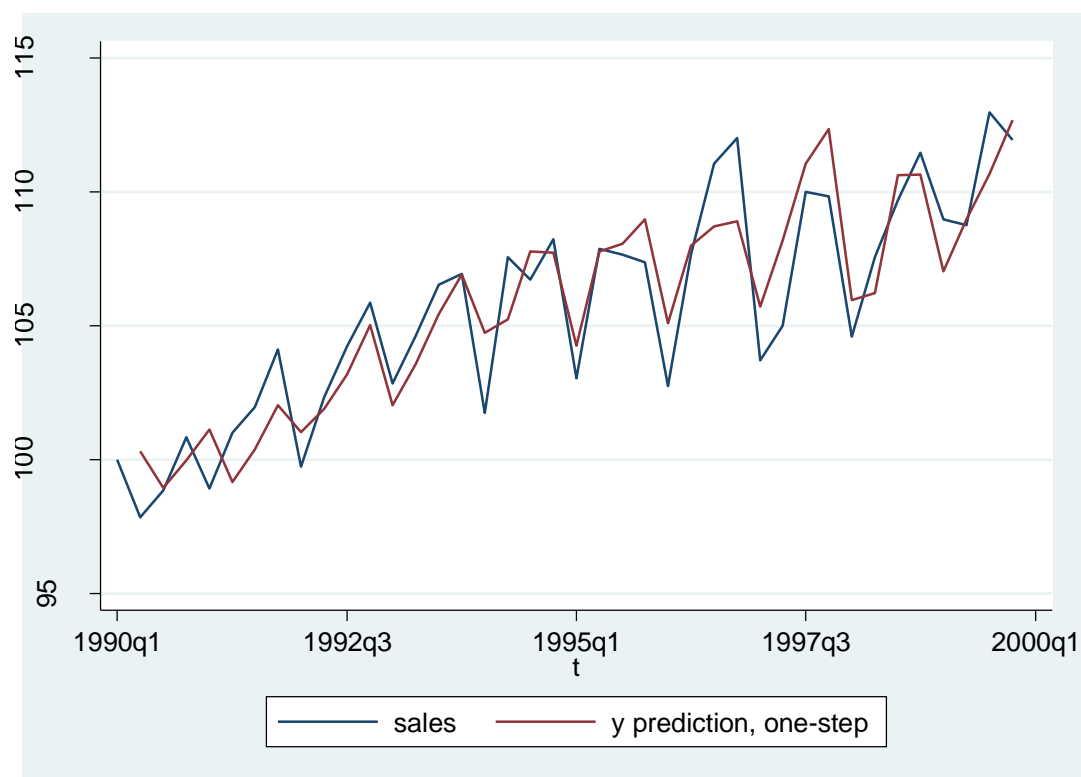
```
predict y2, y
(1 missing value generated)

list sales y1 y2 if _n<=10
```

	sales	y1	y2
1.	100	.3138001	.
2.	97.84603	.3138001	100.3138
3.	98.84029	1.100824	98.94685
4.	100.8275	1.126875	99.96716
5.	98.90981	.2967442	101.1242
6.	100.9992	.2470322	99.15684
7.	101.9653	-.6115102	100.3877
8.	104.1229	.0550769	102.0203
9.	99.74297	-3.103686	101.0192
10.	102.3116	2.146297	101.8893

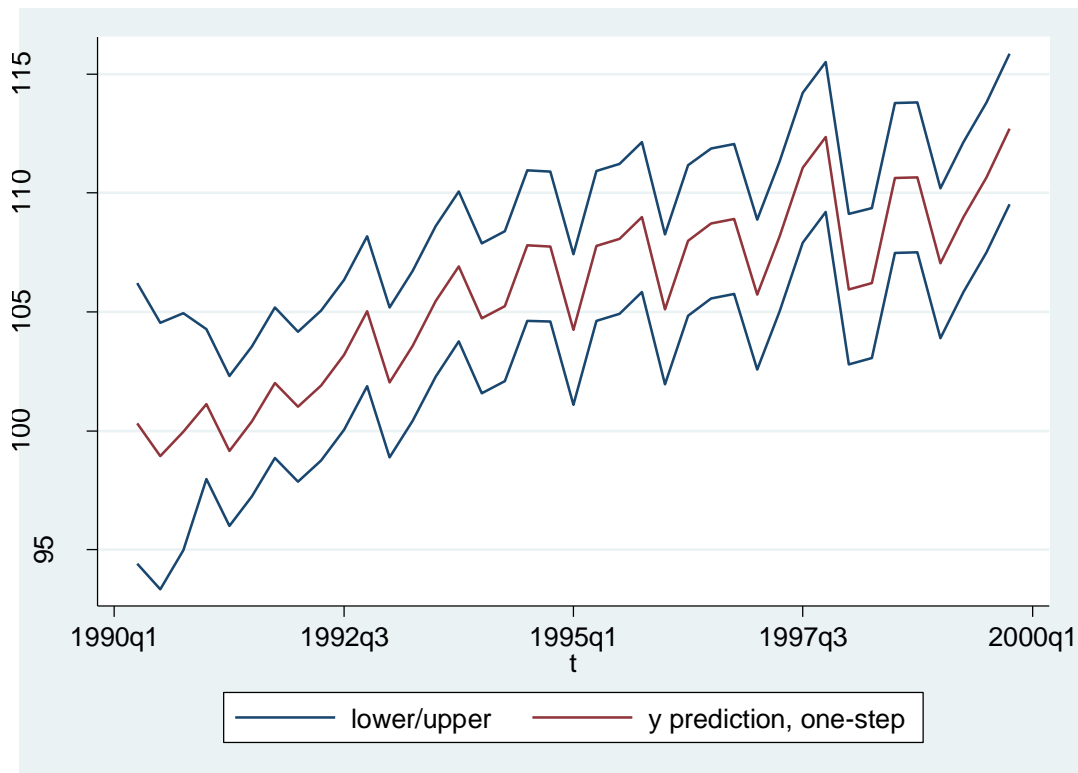
Podemos graficar la serie original de ventas y la predicha:

```
tsline sales y2
```



También podemos obtener el error cuadrático medio de la predicción, y con esto construir el intervalo de confianza:

```
predict meansqerr, mse  
  
g upper=y2+1.96*sqrt( meansqerr)  
g lower=y2-1.96*sqrt( meansqerr)  
  
tsrline lower upper || tsline y2
```



Este intervalo de confianza es sólo una aproximación ya que está basado en la varianza residual, y no considera la variabilidad de la estimación de los parámetros.

VI.4.2. Predicción Dinámica

La predicción un paso adelante lo que hace es utilizar la información disponible en $t-1$ para hacer la predicción para el periodo t . Sin embargo, con este mecanismo de predicción no podrá hacer predicciones más allá del periodo $T+1$.

La predicción dinámica lo que hace es considerar el valor predicho de la variable para poder hacer predicciones más allá de un paso hacia adelante.

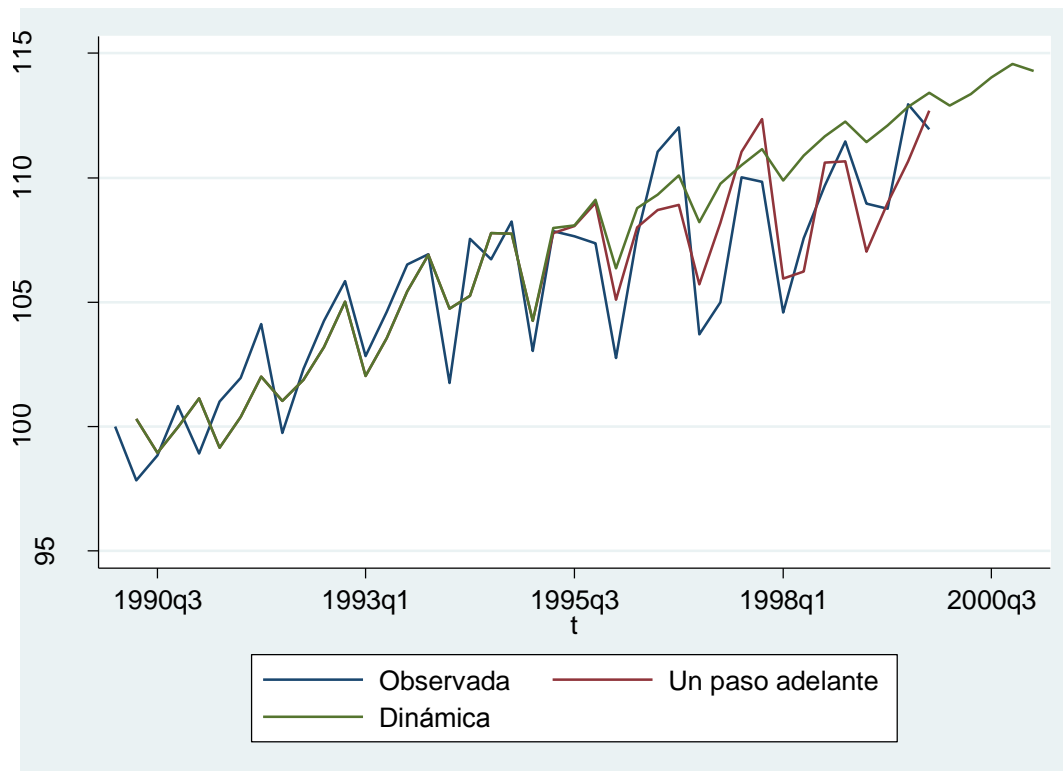
Volviendo a los datos de venta de pavos, tenemos información disponible hasta el cuarto trimestre del año 1999, si queremos hacer predicciones para 5 trimestres más, debemos ejecutar el siguiente comando:

```
tsappend, add(5)  
  
predict ydyn, y dynamic(q(1995q1))  
(1 missing value generated)
```

La primera línea de comandos lo que hace es agregar 5 observaciones más (vacías) a la base de datos. La segunda línea, lo que hace es hacer la predicción de la variable en nivel, ocupando predicción dinámica a partir del primer trimestre de 1995.

El siguiente gráfico nos muestra la serie original, la serie predicha un paso adelante, y la serie predicha de manera dinámica:

```
tsline sales y2 ydyn, legend(label(1 "Observada") label(2 "Un paso adelante")  
label(3 "Dinámica"))
```



Podemos notar que antes de 1995 ambas predicciones coinciden, luego estas se comienzan a diferenciar.

VI.5. Criterios de bondad de ajuste

Tal como en el modelo de regresión lineal existe la medida de bondad de ajuste denominada R^2 y de evaluar el desempeño del mismo. En modelos de series de tiempo, la medida que nos permite evaluar una predicción es el error cuadrático medio (MSE).

Adicionalmente, existen dos medidas que permiten estudiar la bondad de ajuste del modelo ARIMA estimado y elegir entre dos o más modelos anidados. Estos son dos criterios de información ampliamente utilizados como el criterio de Akaike y el criterio bayesiano o de Schwartz:

$$AIC = -2\ln L + 2p$$

$$BIC = -2\ln L + p\ln T$$

En STATA estos criterios pueden ser reportados de manera muy sencilla una vez estimado el modelo ARIMA:

```

arima sales, arima(3,1,0)

estat ic
-----
      Model |      Obs      ll(null)      ll(model)      df          AIC          BIC
-----+-----
          . |       39          .      -75.53558         5      161.0712      169.389
-----
Note:  N=Obs used in calculating BIC; see [R] BIC note

```

Ahora podemos estimar nuevamente el modelo pero incorporando un componente MA:

```

arima sales, arima(3,1,1)

estat ic
-----
      Model |      Obs      ll(null)      ll(model)      df          AIC          BIC
-----+-----
          . |       39          .      -75.18403         6      162.3681      172.3494
-----
Note:  N=Obs used in calculating BIC; see [R] BIC note

```

En este segundo modelo ambos valores de los criterios de información son mayores, por lo cual nos deberíamos quedar con el modelo que no incorpora el componente media móvil.

VI.6. ¿Cómo escoger la cantidad de componentes AR y MA en el modelo?

Primero hay que decir que escoger el modelo ARIMA “correcto” es un arte más que una ciencia. Debemos tener presente que la función de autocorrelación muestral nos ayudará a determinar el número de componentes MA en el modelo, y la función de autocorrelación parcial, el número de

componentes AR. El problema es que típicamente las series son una combinación entre modelos AR y MA, es decir, modelos ARMA donde sabemos que la función de autocorrelación decae más rápido o más lento que un proceso AR puro.

El consejo es comenzar con un modelo general, con cuatro componentes AR y 4 componentes MA, luego estimar diferentes combinaciones, y quedarse con el modelo con menor criterio de información.

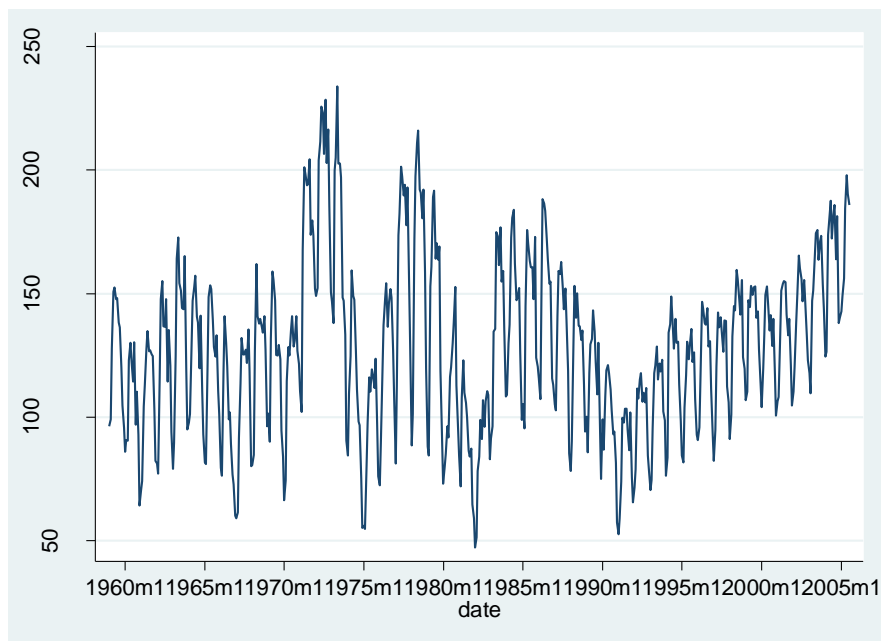
VI.7. Modelos ARIMA con estacionalidad

Recordemos que si la magnitud de la fluctuación estacional no es proporcional al nivel de la serie, el efecto estacional es aditivo, por el contrario si la fluctuación estacional es proporcional al nivel de la serie, la estacionalidad es multiplicativa.

VI.7.1. Estacionalidad aditiva

Utilizaremos la base de datos `hstarts.dta` que contiene datos mensuales de construcciones nuevas, debido a los meses de invierno o frío, el inicio de construcciones tiene un fuerte componente estacional. A pesar de que la estacionalidad normalmente es multiplicativa en este caso una estacionalidad aditiva tampoco parece tan extraña.

```
use "hstarts.dta", clear
tsset date
tsline starts
```



En un modelo ARIMA estacional, se utiliza un segundo proceso ARMA para modelar las fluctuaciones estacionales. Por ejemplo, para los datos mensuales de construcción de nuevas viviendas, podemos usar un proceso ARMA para modelar las fluctuaciones mes a mes de la serie, y un segundo proceso ARMA para modelar las variaciones estacionales:

$$h_t = \alpha + \phi_1 h_{t-1} + \phi_{12} h_{t-12} + e_t + \theta_1 e_{t-1} + \theta_{12} e_{t-12}$$

En este caso estamos utilizando un proceso ARMA(1,1) para las fluctuaciones mensuales, y también un proceso ARMA(1,1) para las fluctuaciones estacionales.

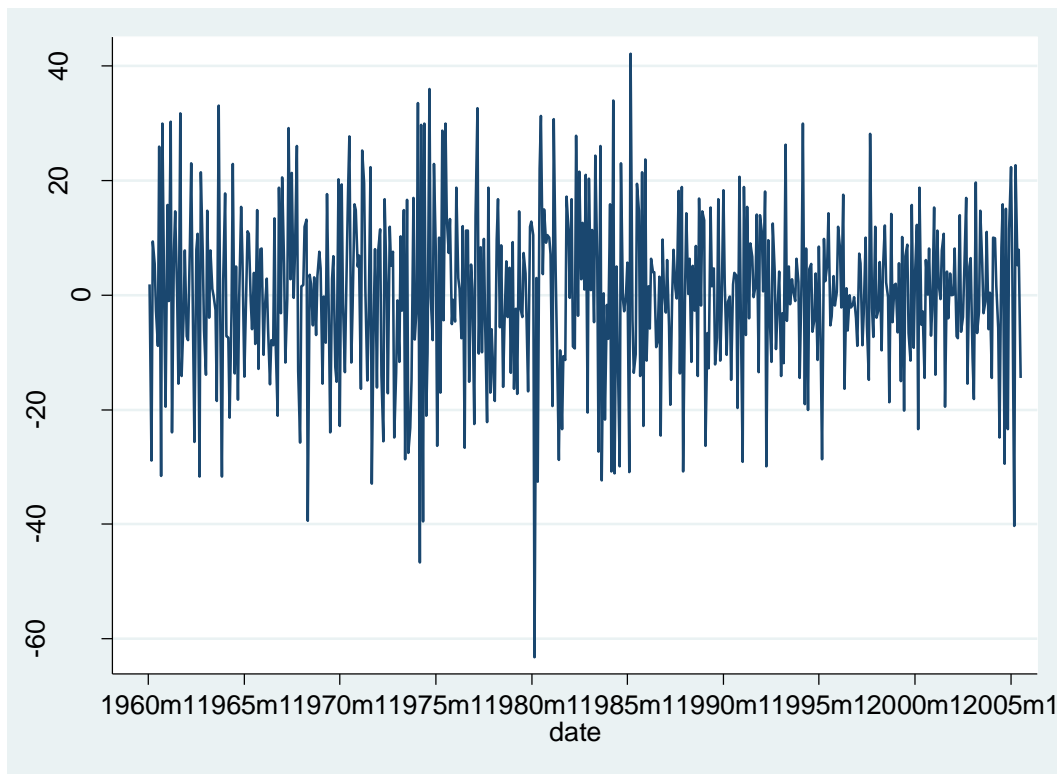
Si utilizáramos un proceso ARMA(2,2) para el componente estacional, tendríamos:

$$h_t = \alpha + \phi_1 h_{t-1} + \phi_{12} h_{t-12} + \phi_{24} h_{t-24} + e_t + \theta_1 e_{t-1} + \theta_{12} e_{t-12} + \theta_{24} e_{t-24}$$

Nuevamente, es clave estudiar la función de autocorrelación muestral y autocorrelación parcial para determinar el número de rezagos a incluir.

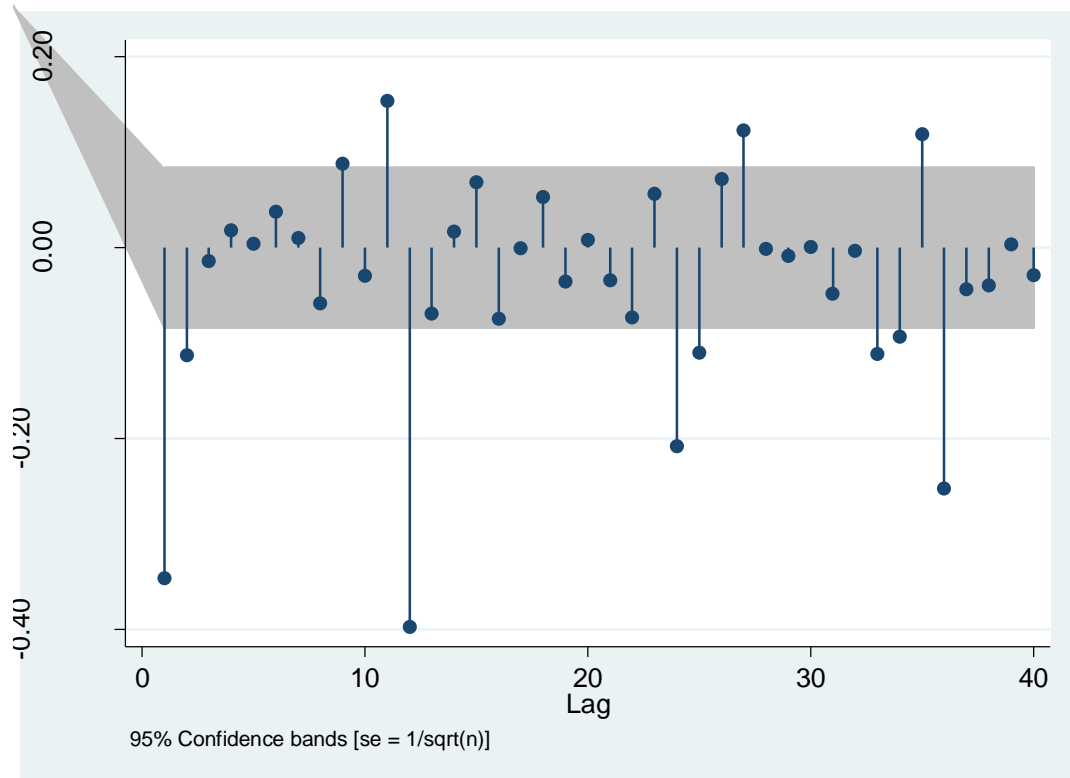
Recordemos que lo primero que se requiere es que la serie sea estacionaria, si la serie tiene un componente estacional, debemos trabajar con la serie diferenciada en primera diferencia y con respecto al componente estacional.

```
tsline DS12.starts
```



Ahora podemos ver la función de autocorrelación parcial:

```
pac DS12.starts
```



Podemos ver claramente que los rezagos 1 y 12 son relevante, también los rezagos 24 y 36, indicando que quizás debemos incorporar tres componentes autoregresivos.

Estimaremos diferentes modelos y utilizaremos los criterios de información para elegir entre ellos:

```

use "hstarts.dta", clear
tsset date

arima DS12.starts, ar(1 12) ma(1 12)
estimates store arima1

arima DS12.starts, ar(1 12 24) ma(1 12 24)
estimates store arima2

arima DS12.starts, ar(1 12 24 36) ma(1 12 24 36)
estimates store arima3

arima DS12.starts, ar(1 12)
estimates store arima4

arima DS12.starts, ar(1 12 24)
estimates store arima5

arima DS12.starts, ar(1 12 24 36)
estimates store arima6

estimates table arima1 arima2 arima3 arima4 arima5 arima6, stat(aic, bic)
b(%7.3g) p(%4.3f)

```

Variable	arima1	arima2	arima3	arima4	arima5	arima6
starts						
_cons	.0134	.0147	.015	.034	.0324	.0286
	0.771	0.790	0.791	0.919	0.909	0.896
ARMA						
ar						
L1.	-.288	-.257	-.267	-.28	-.267	-.232
	0.000	0.000	0.000	0.000	0.000	0.000
L12.	.122	-.327	-.217	-.393	-.478	-.544
	0.006	0.001	0.148	0.000	0.000	0.000
L24.		.103	.0783		-.195	-.344
		0.022	0.515		0.000	0.000
L36.			-.124			-.286
			0.014			0.000
ma						
L1.	.00621	-.058	-.0523			
	0.796	0.106	0.098			
L12.	-.931	-.554	-.718			
	0.000	0.000	0.000			
L24.		-.517	-.445			
		0.000	0.000			
L36.			.0579			
			0.728			
sigma						
_cons	10.9	10.3	9.94	12.7	12.4	11.9
	0.000	0.000	0.000	0.000	0.000	0.000
Statistics						
aic	4197	4188	4186	4336	4317	4272
bic	4222	4222	4229	4354	4338	4298

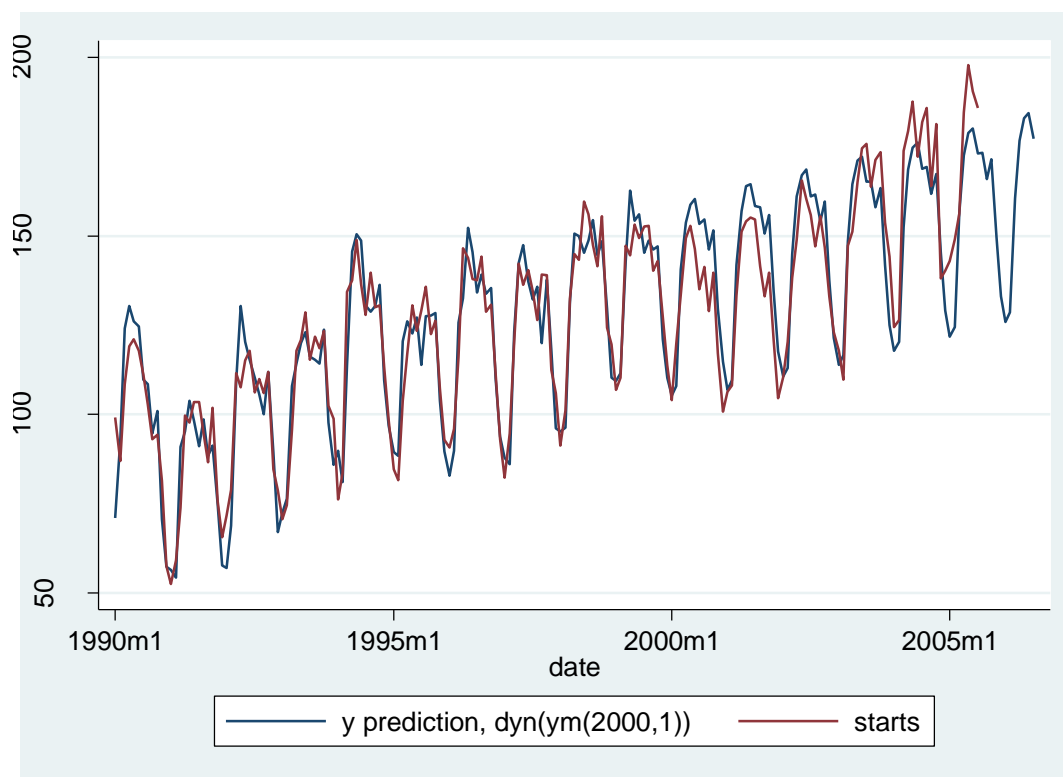
legend: b/p

Podemos apreciar que los modelos sólo con componente autoregresivo tienen mayores criterios de información que los modelos que además incluyen un componente de media móvil. Ahora, dentro de los modelos ARMA deberíamos optar por el primero o el segundo según el criterio BIC, y por el segundo según el criterio AIC. Dado que en el segundo modelo casi todos los coeficientes resultan ser estadísticamente significativos a excepción del coeficiente asociado al componente MA(1) que lo es sólo al 10.6%, optaremos por quedarnos con el proceso ARMA(1,1) para las fluctuaciones mes a mes, con estacionalidad aditiva que sigue un proceso ARMA(2,2).

$$h_t = \alpha + \phi_1 h_{t-1} + \phi_{12} h_{t-12} + \phi_{24} h_{t-24} + e_t + \theta_1 e_{t-1} + \theta_{12} e_{t-12} + \theta_{24} e_{t-24}$$

Entonces estimado el modelo podemos hacer predicciones:

```
tsappend, add(12)
predict aditiva, y dynamic(ym(2000,1))
tsline aditiva starts if date>=ym(1990,1)
```



VI.7.2. Estacionalidad multiplicativa

En la sección anterior modelamos el componente estacional de la serie de manera aditiva, entonces un proceso que sigue un ARMA(1,1) para las variaciones mensuales y ARMA(1,1) para el componente estacional, se podía expresar de la siguiente manera:

$$y_t = \phi_1 y_{t-1} + \phi_{12} y_{t-12} + e_t + \theta_1 e_{t-1} + \theta_{12} e_{t-12}$$

Lo que se puede expresar utilizando operadores de rezagos, de la siguiente manera:

$$(1 - \phi_1 L - \phi_{12} L^{12}) y_t = (1 + \theta_1 L + \theta_{12} L^{12}) e_t$$

Sin embargo, típicamente el factor estacional funciona de manera multiplicativa, es decir, depende del nivel de la serie. El proceso análogo a un ARMA(1,1) con componente estacional ARMA(1,1) pero multiplicativo sería de la siguiente manera:

$$(1 - \phi_1 L)(1 - \phi_{12} L^{12}) y_t = (1 + \theta_1 L)(1 + \theta_{12} L^{12}) e_t$$

En términos generales podemos escribir un proceso ARIMA con estacionalidad multiplicativa, permitiendo diferenciar mes a mes la serie y por estacionalidad, de la siguiente manera:

$$\begin{aligned} (1 - \phi_1 L - \dots - \phi_p L^p)(1 - \phi_{s,1} L^s - \phi_{s,2} L^{2s} - \dots - \phi_{s,p} L^{ps}) \Delta^d \Delta_s^D y_t \\ = (1 + \theta_1 L + \dots + L^p)(1 + \theta_{s,1} L^s + \theta_{s,2} L^{2s} + \dots + \theta_{s,p} L^{ps}) e_t \end{aligned}$$

Este modelo se denota como $(p, d, q) \times (P, D, Q)_s$, donde p es el número de componente autoregresivos, y q el número de componentes de media móvil a ser incluidos en el modelo básico. s denota la estacionalidad, por ejemplo s=4 en datos trimestrales, y s=12 en datos mensuales. P y Q son la cantidad de componentes autoregresivos y de medias móvil a ser incluidos en la parte estacional del modelo. Finalmente, d y D denotan el número de veces que la serie debe ser diferenciada, la serie básica y por estacionalidad respectivamente.

Tal como antes, la función de autocorrelación muestral y función de autocorrelación parcial son útiles para determinar el orden de cada uno de estos componentes del modelo. Se ha demostrado que graficar la función de autocorrelación muestral y parcial luego de haber diferenciado la serie para eliminar cualquier tendencia o no estacionariedad, y diferenciar el componente estacional ayudan mucho a la identificación del proceso.

Volviendo a los datos de construcción de nuevas viviendas, asumamos el siguiente proceso $(1,1,1) \times (1,1,1)_{12}$, es decir, el mismo proceso anterior pero ahora la estacionalidad es multiplicativa. Ocuparemos la opción `sarima(P,D,Q,s)` para indicarle a STATA que estamos trabajando con estacionalidad multiplicativa.

```

arima starts, arima(1,1,1) sarima(1,1,1,12)
estimates store arima7

arima starts, arima(1,1,1) sarima(2,1,2,12)
estimates store arima8

arima starts, arima(1,1,1) sarima(3,1,3,12)
estimates store arima9

estimates table arima7 arima8 arima9, stat(aic, bic) b(%7.3g) p(%4.3f)
-----
Variable | arima7      arima8      arima9
-----+-----
starts   |
  _cons  |      .0129      .0129      .0128
         |      0.742      0.739      0.759
-----+-----
ARMA     |
  ar     |
    L1.  |     -.0366     -.016     -.00576
         |      0.765      0.899      0.965
         |
    ma    |
    L1.  |     -.291      -.306      -.308
         |      0.011      0.009      0.012
-----+-----
ARMA12   |
  ar     |
    L1.  |      .162      -.649      .855
         |      0.001      0.004      0.000
    L2.  |             .168      -1.05
         |             0.001      0.000
    L3.  |             .109
         |             0.060
         |
    ma    |
    L1.  |     -.942      -.245      -1.64
         |      0.000      0.275      0.000
    L2.  |             -.857      1.59
         |             0.000      0.000
    L3.  |             -.863
         |             0.000
-----+-----
sigma    |
  _cons  |      10.8      10.2      10.7
         |      0.000      0.000      0.000
-----+-----
Statistics |
  aic    |      4188      4190      4184
  bic    |      4213      4224      4227
-----+-----

```

legend: b/p

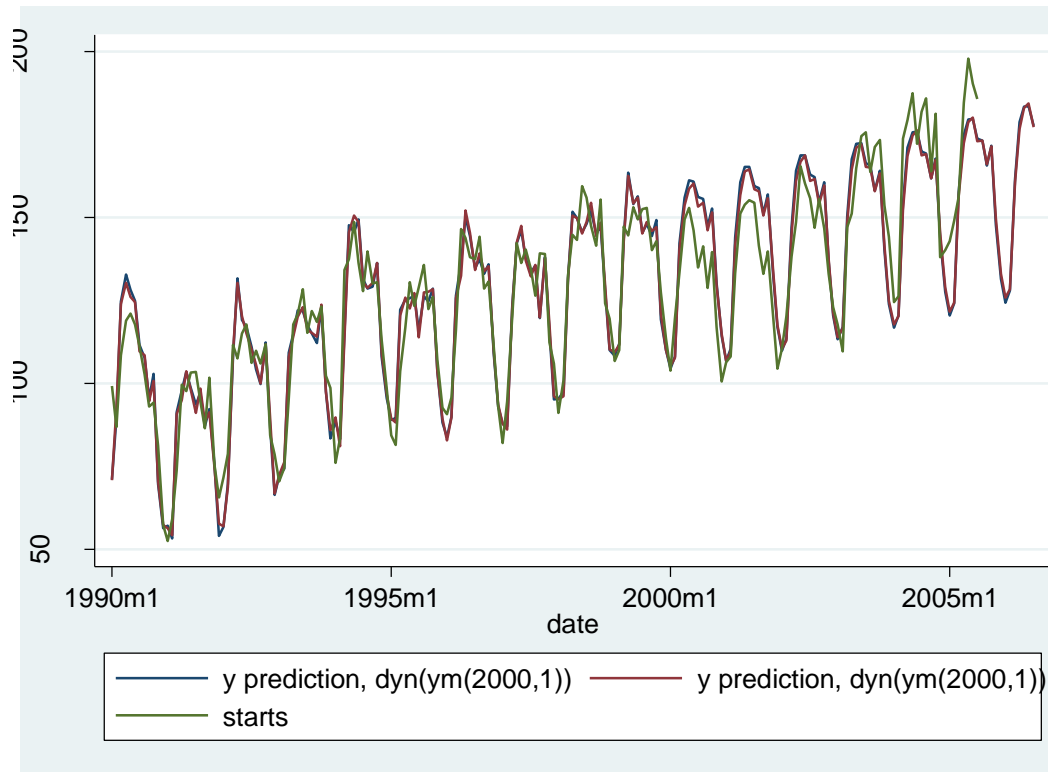
Según el criterio de información bayesiano nos debemos quedar con el modelo más parsimonioso, es decir, el modelo ARIMA(1,1,1) con componente estacional multiplicativo ARIMA(1,1,1).

Ahora podemos utilizar este modelo para hacer predicción:

```

arima starts, arima(1,1,1) sarima(1,1,1,12)
predict multi, y dynamic(ym(2000,1))
tsline multi aditiva starts if date>=ym(1990,1)

```



En este caso en particular, las predicciones con estacionalidad multiplicativa y aditiva para el modelo ARIMA son prácticamente iguales. Revisando los criterios de información del modelo con estacionalidad multiplicativa versus el modelo con estacionalidad aditiva, nos deberíamos quedar con el modelo con ARIMA(1,1,1) con estacionalidad multiplicativa ARIMA(1,1,1).

La sugerencia es siempre comenzar con estacionalidad multiplicativa, y sólo en caso que esta no logre ajustar bien los datos intentar con estacionalidad aditiva.

VI.8 Modelos ARMAX

Hasta ahora hemos tratado de determinar el comportamiento de una serie de tiempo y_t en función de valores pasados de la misma serie, y del término de error. Sin embargo, pueden haber otros factores medidos a través de otras variables que también ayuden y complementen la parte explicara por el modelo ARIMA. Esto son los denominados modelos ARIMAX.

En STATA podemos incorporar estas otras variables, y estimar un modelo ARIMAX, con el mismo comando antes utilizado, pero después del nombre de nuestra serie de interés se agregan las variables explicativas que consideremos relevantes.

Por ejemplo, en el modelo anterior que buscamos estudiar el comportamiento de las construcciones de viviendas nuevas podemos incorporar la tasa de interés como un regresos o variable explicativa.

```

arima starts irate, arima(1,1,1) sarima(1,1,1,12)
estimates store arimax
estimates table arima2 arima7 arimax, stat(aic, bic) b(%7.3g) p(%4.3f)

```

Variable	arima2	arima7	arimax
starts			
irate			
DS12.			-2.12
			0.080
_cons	.0147	.0129	.0101
	0.790	0.742	0.790
ARMA			
ar			
L1.	-.257	-.0366	.0008
	0.000	0.765	0.995
L12.	-.327		
	0.001		
L24.	.103		
	0.022		
ma			
L1.	-.058	-.291	-.329
	0.106	0.011	0.003
L12.	-.554		
	0.000		
L24.	-.517		
	0.000		
sigma			
_cons	10.3	10.8	10.8
	0.000	0.000	0.000
ARMA12			
ar			
L1.		.162	.158
		0.001	0.001
ma			
L1.		-.942	-.943
		0.000	0.000
Statistics			
aic	4188	4188	4187
bic	4222	4213	4217

legend: b/p

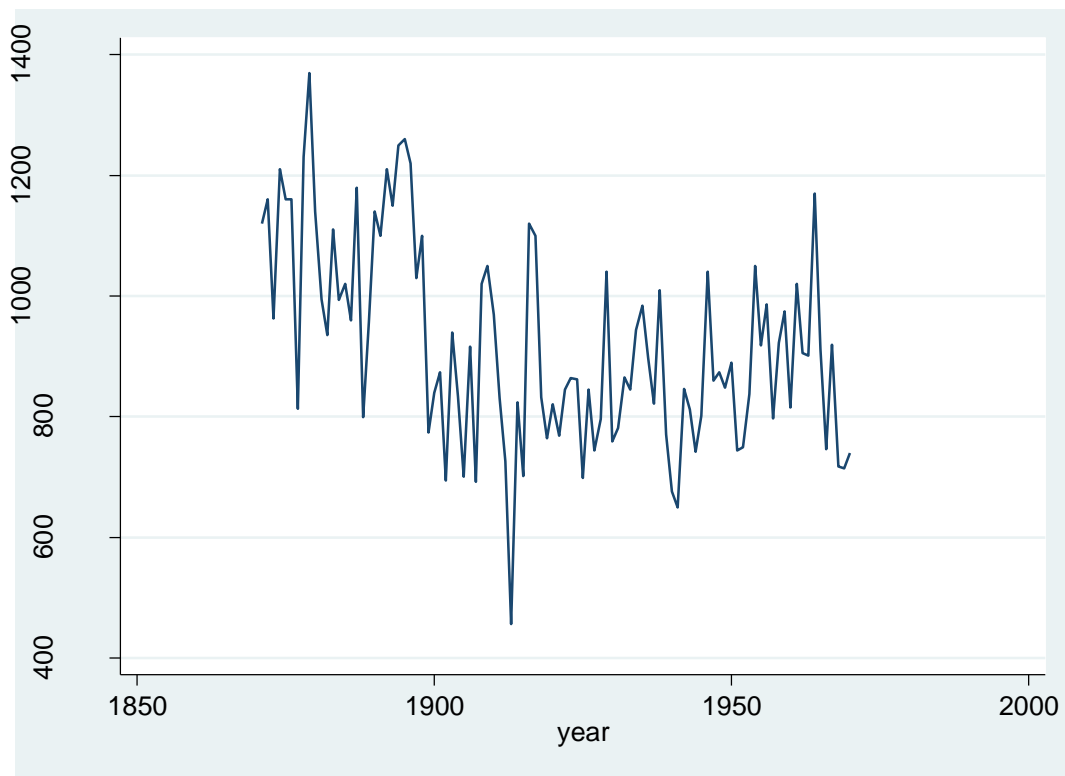
Observamos que la variable de tasa de interés resulta ser estadísticamente significativa pero sólo a un 10% de significancia, adicionalmente, el criterio de información bayesiano es mayor en el modelo que incorpora esta variable explicativa, por lo cual nos deberíamos quedar con el modelo anterior.

VI.9. Outliers

Cuando nuestra serie de tiempo tiene observaciones aisladas o interrupciones en el proceso que genera los datos podemos generar predicciones sesgadas.

Observemos los datos sobre cantidad de agua descargada por el río Nilo desde 1871 a 1970. Este río ha sido estudiado de manera extensa por climatólogos, así como altamente utilizado para el estudio en series de tiempo de procesos ARIMA. Ver Cobb (1978) y Balke (1993).

```
use nile  
tsset year  
tsline discharge
```



Podemos observar que la serie es estacionaria, pero para asegurar que así sea podemos realizar el test de raíz unitaria:

```
pperron discharge
```

```
Phillips-Perron test for unit root
```

```
Number of obs = 99
```

```
Newey-West lags = 3
```

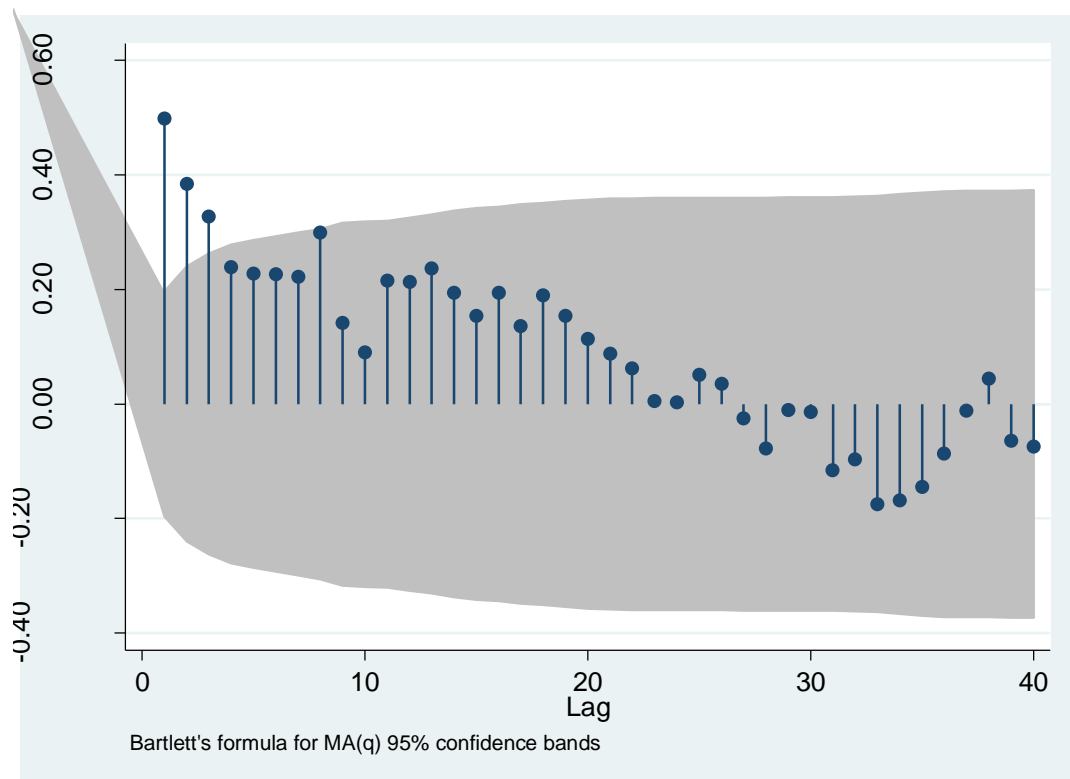
----- Interpolated Dickey-Fuller -----				
	Test Statistic	1% Critical Value	5% Critical Value	10% Critical Value
Z (rho)	-48.815	-19.782	-13.692	-10.994
Z (t)	-5.654	-3.511	-2.891	-2.580

```
MacKinnon approximate p-value for Z(t) = 0.0000
```

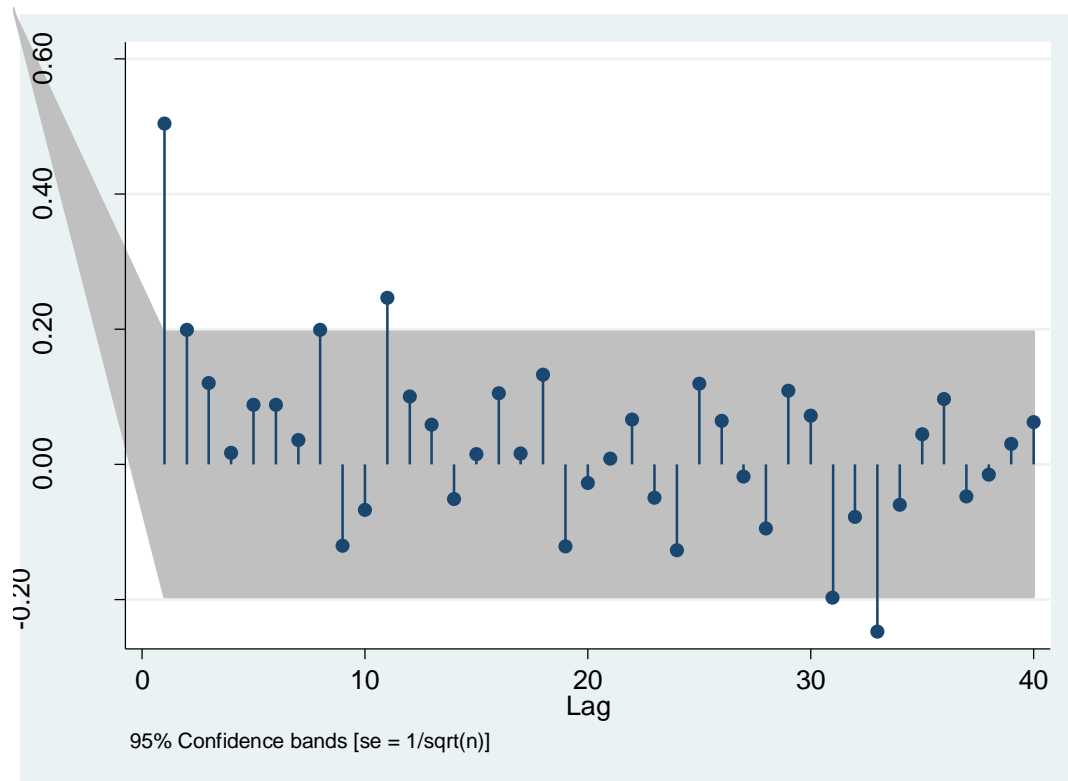
Se rechaza la hipótesis nula de raíz unitaria en la serie, por lo cual podemos decir que la serie es estacionaria o integrada de orden 0.

Ahora podemos observar la función de autocorrelación muestral y parcial para determinar aproximadamente que tipo de proceso sigue esta serie:

```
ac discharge
```



```
pac discharge
```



Por la forma en que decae la función de autocorrelación muestral el proceso tiene un componente de AR, y por la forma de la función de autocorrelación parcial, este es de orden 1. Con respecto al componente de media móvil pareciese también tener pero debemos estimar varios modelos para determinar el mejor proceso.


```

arima discharge, arima(1,0,1)
estimates store arima1

arima discharge, arima(1,0,2)
estimates store arima2

arima discharge, arima(1,0,3)
estimates store arima3

estimates table arima1 arima2 arima3, stat(aic, bic) b(%7.3g) p(%4.3f)
-----
      Variable | arima1      arima2      arima3
-----+-----
discharge     |
  _cons       |      921      930      935
              |      0.000      0.000      0.000
-----+-----
ARMA          |
  ar         |
    L1.       |      .861      .954      .966
              |      0.000      0.000      0.000
              |
  ma         |
    L1.       |     -.518     -.605     -.619
              |      0.000      0.000      0.000
    L2.       |              -.146     -.0941
              |              0.141      0.388
    L3.       |              -.0778
              |              0.469
-----+-----
sigma         |
  _cons       |      141      140      140
              |      0.000      0.000      0.000
-----+-----
Statistics    |
  aic         |     1282     1283     1284
  bic         |     1292     1296     1300
-----+-----
                                legend: b/p

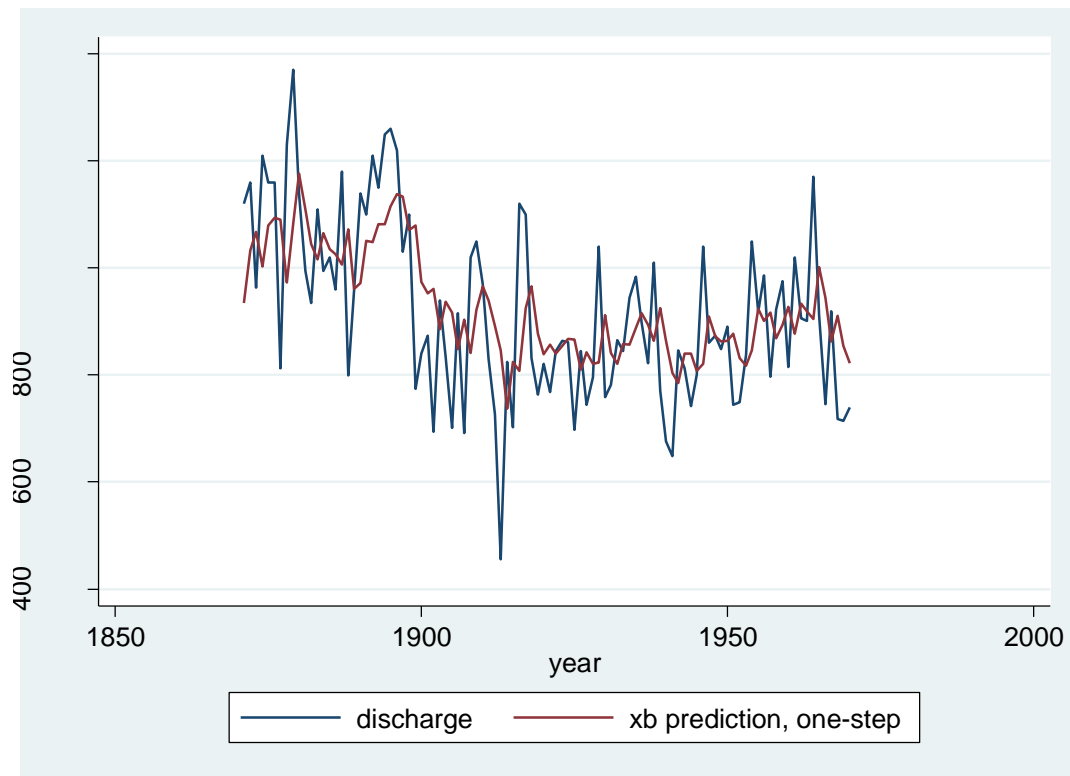
```

Por lo cual, escogemos el modelo ARIMA(1,0,1) para este proceso. Ahora podemos hacer la predicción un paso adelante, y graficar:

```

predict discharge_p1
tsline discharge discharge_p1

```



Los valores predichos para la serie parecen ajustarse de manera razonable, pero existe variabilidad que no está siendo capturada. La desviación estándar del error estimada es de 141.04.

Si miramos el gráfico notamos que en las primeras observaciones la media de la serie pareciese ser más alta que después. En efecto, a comienzos de 1899 se construyó la primera represa, y como resultado el nivel de agua en el Nilo disminuyó. Este factor debe ser considerado en los datos para efectos de estimar mejor el modelo, y obtener parámetros correctos para hacer predicciones.

Entonces vamos a generar una variable explicativa binaria (dummy) que tome valor uno desde 1899 en adelante, y cero para los años previos.

```

g d1899=0
replace d1899=1 if year>=1899
arima discharge d1899, arima(1,0,1)

```

		OPG					
discharge		Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
discharge							
	d1899	-248.969	32.95188	-7.56	0.000	-313.5535	-184.3845
	_cons	1098.466	27.31565	40.21	0.000	1044.928	1152.003
ARMA							
	ar						
	L1.	.0357627	.7375229	0.05	0.961	-1.409756	1.481281
	ma						
	L1.	.1275965	.689086	0.19	0.853	-1.222987	1.47818
	/sigma	124.6959	8.539466	14.60	0.000	107.9588	141.4329

Lo que hace este modelo es estimar una constante distinta para el periodo previo a 1899. Pero observamos que los componentes AR y MA ahora no son significativos.

Debemos ajustar la estimación del modelo, para eso estimamos modelos AR(1) y MA(1) incorporando esta variable explicativa, y escogemos el modelo con menor criterio de información. En este caso ambos modelos tienen igual criterio de información, pero al estimar el modelo AR(1), el coeficiente autoregresivo resulta significativo al 14%, en el modelo MA(1) el coeficiente es significativo al 9%, por lo cual nos quedaremos con el modelo MA(1). Veamos como se comparan las predicciones con el modelo anterior.

```

arima discharge d1899, arima(1,0,1)
estimates store arima4
arima discharge d1899, arima(1,0,0)
estimates store arima5
arima discharge d1899, arima(0,0,1)
estimates store arima6
estimates table arima1 arima4 arima5 arima6, stat(aic, bic) b(%7.3g) p(%4.3f)

```

Variable	arima1	arima4	arima5	arima6

discharge				
d1899		-249	-249	-249
		0.000	0.000	0.000
_cons	921	1098	1099	1098
	0.000	0.000	0.000	0.000

ARMA				
ar				
L1.	.861	.0358	.16	
	0.000	0.961	0.141	
ma				
L1.	-.518	.128		.162
	0.000	0.853		0.091

sigma				
_cons	141	125	125	125
	0.000	0.000	0.000	0.000

Statistics				
aic	1282	1259	1257	1257
bic	1292	1272	1267	1267

REFERENCIAS

Akaike, H. 1973. Information theory and an extension of the maximum likelihood principle. In *Second International Symposium on Information Theory*, B. N. Petrov and F. Csaki (eds.), 267-281. Budapest: Akailseoniai-Kiudo.

Balke, N. S. 1993. Detecting level shifts in time series. *Journal of Business and Economic Statistics*, 11, 81-92.

Bollerslev, T. 1986. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31, 307-327.

Bollerslev, T., R. Y. Chou, and K. F. Kroner. 1992. ARCH modeling in finance. *Journal of Econometrics*, 52, 5-59.

Bollerslev, T., R. F. Engle, and D. B. Nelson. 1994. ARCH models. In *Handbook of Econometrics, Volume IV*, R. F. Engle & D. L. McFadden (Eds.) New York: Elsevier.

Bowerman, B. L. and R. T. O'Connell. 1993. *Forecasting and Time Series: An Applied Approach*, 3rd ed. Pacific Grove, CA: Duxbury.

Box, G. E. P., G. M. Jenkins, and G. C. Reinsel. 1994. *Time Series Analysis: Forecasting and Control*, 3rd ed. Upper Saddle River, NJ: Prentice-Hall.

Breusch, T. S. 1978. Testing for autocorrelation in dynamic linear models. *Australian Economic Papers*, 17, 334-355.

Brockwell, P. J. and R. A. Davis. 2002. *Introduction to Time Series and Forecasting*, 2nd ed. New York: Springer-Verlag.

Chatfield, C. 2004. *The Analysis of Time Series*, 6th ed. Boca Raton, FL: Chapman & Hall/CRC.

Cobb, G. W. 1978. The problem of the Nile: Conditional solution to a changepoint problem. *Biometrika*, 65, 243-251.

Dickey, D. A. and W. A. Fuller. 1979. Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American Statistical Association*, 74, 427-431.

Durbin, J. 1970. Testing for serial correlation in least-squares regression when some of the regressors are lagged dependent variables. *Econometrica*, 38, 410-421.

Durbin, J. and G. S. Watson. 1950. Testing for serial correlation in least squares regression I. *Biometrika*, 37, 409-428.

Durbin, J. and G. S. Watson. 1951. Testing for serial correlation in least squares regression II. *Biometrika*, 38, 159-178.

Durbin, J. and G. S. Watson. 1971. Testing for serial correlation in least squares regression III. *Biometrika*, 58, 1-19.

Elliott, G., T. J. Rothenberg, and J. H. Stock. 1996. Efficient tests for an autoregressive unit root. *Econometrica*, 64, 813-836.

Engle, R. F. 1982. Autoregressive conditional heteroskedasticity with estimates of the variance of UK inflation. *Econometrica*, 50, 987-1008.

Engle, R. F. 2001. GARCH 101: The use of ARCH/GARCH models in applied econometrics. *Journal of Economic Perspectives*, 15(4), 157-168.

Engle, R. F. and C. W. J. Granger. 1987. Co-integration and error correction: representation, estimation, and testing. *Econometrica*, 55, 251-276.

Evans, G. B. A. and N. E. Savin. 1981. Testing for unit roots: 1. *Econometrica*, 49, 753-779.

Godfrey, G. L. 1978. Testing against general autoregressive and moving average error models when the regressors include lagged dependent variables. *Econometrica*, 46, 1293-1302.

Granger, C. W. J. and P. Newbold. 1974. Spurious regressions in econometrics. *Journal of Econometrics*, 2, 111-120.

Hamilton, J. D. 1994. *Time Series Analysis*. Princeton, NJ: Princeton University Press.

Harvey, A. C. 1989. *Forecasting, Structural Time Series Models and the Kalman Filter*. Cambridge: Cambridge University Press.

Johnston, J. and J. DiNardo. 1997. *Econometric Methods*, 3rd ed. New York: McGraw-Hill.

Mackinnon, J. G. 1994. Approximate asymptotic distribution functions for unit-root and cointegration tests. *Journal of Business and Economic Statistics*, 12, 167-176.

McCullagh, P. and J. A. Nelder. 1989. *Generalized Linear Models*, 2nd ed. London: Chapman & Hall.

McCullough, B. D. 1998. Algorithm choice for (partial) autocorrelation functions. *Journal of Economic and Social Measurement*, 24, 265-278.

Montgomery, D. C., L. A. Johnson, and J. S. Gardiner. 1990. *Forecasting and Time Series Analysis*, 2nd ed. New York: McGraw-Hill.

Newey, W. K. and K. D. West. 1987. A simple, positive semi-definite, heteroskedasticity and autocorrelation consistent covariance matrix. *Econometrica*, 55, 703-708.

Newton, H. J. 1988. *Timeslab: A Time Series Analysis Laboratory*. Pacific Grove, CA: Wadsworth & Brooks/Cole Publishing.

Schwartz, G. 1978. Estimating the dimension of a model. *Annals of Statistics*, 6, 461-464. Tsay, R. S. 1988. Outliers, level shifts, and variance changes in time series. *Journal of Forecasting*, 7, 1-20.

Phillips, P. C. B. 1986. Understanding spurious regressions in econometrics. *Journal of Econometrics*, 33, 311-340.

Phillips, P. C. B. and P. Perron. 1988. Testing for a unit root in time series regression. *Biometrika*, 75, 335-346.

Savin, N. E. and K. J. White. 1977. The Durbin-Watson test for serial correlation with extreme sample sizes or many regressors. *Econometrica*, 45, 1989-1996.