Introducción:

GUI's Con Matlab

Luis Pinedo Caro
E-mail: bizantino2@hotmail.com

Índice:

Prefacio:

Este documento pretende servir de guía para iniciarse en la construcción de interfaces graficas con Matlab. Habida cuenta de la dificultad de encontrar algo así, en español y sobre este tema y a este nivel, me he animado a escribir esta introducción para tapar el hueco en la medida de lo posible. Este documento esta fuertemente orientado al campo donde realizo mis estudios, es decir, la Economía, y por tanto los ejemplos prácticos y las utilidades explicadas en las ultimas secciones están enteramente referidas a problemas particulares de índole económica.
Por otra parte he escrito este pequeño manual teniendo en cuenta que quien lo va a leer carece totalmente de conocimientos en la construcción de GUI's con Matlab, no así de conocimientos de programación en Matlab. Estos últimos no son estrictamente necesarios para seguir estas notas, ni se requiere que sean altos, pero serán bastante útiles para entender propiamente los ejemplos finales.

1-¿Qué es una GUI? ¿Para que sirven?

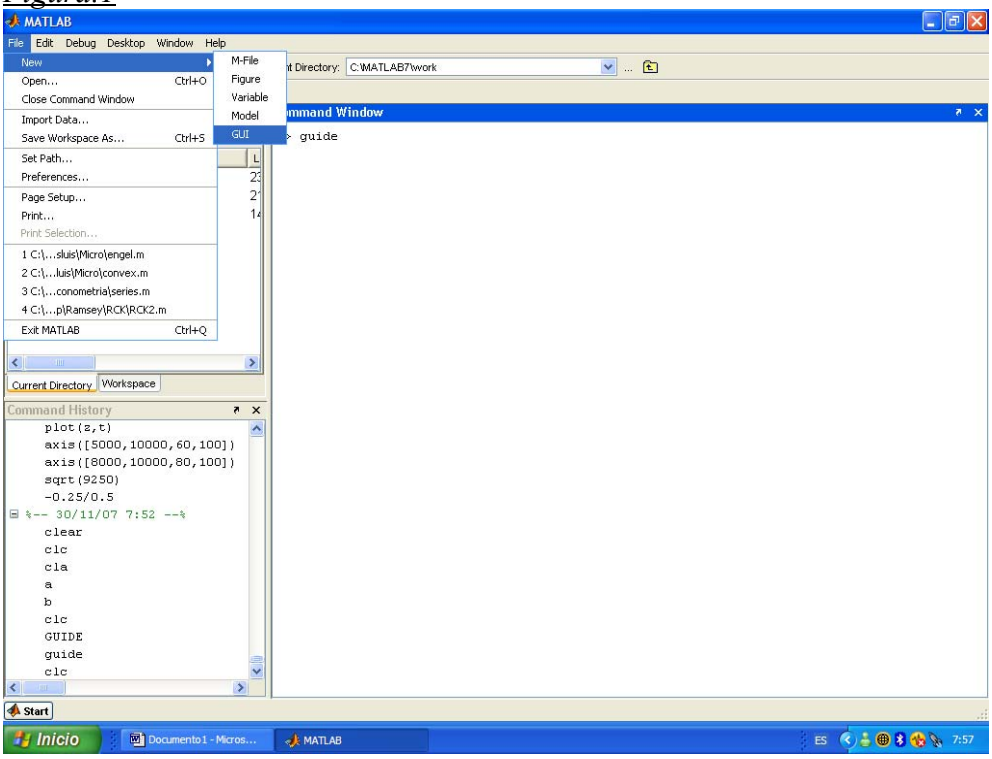
En primer lugar una GUI (graphical user interface) es una herramienta que nos permite operar con un programa previamente construido en Matlab de manera que tanto el autor del mismo como otros posibles usuarios finales puedan tener una rápida e interactiva visión del problema.

En segundo lugar el hecho de querer construir una interfaz es cuestión de las necesidades de cada persona. Es un hecho que existe un alto coste de oportunidad, especialmente en lo referido a tiempo invertido si se quiere llegar a usar con facilidad esta herramienta, por lo que antes de seguir leyendo el usuario debería tener claras sus preferencias. Una vez que se decide seguir adelante es preciso conocer las ventajas que nos ofrece la herramienta GUIDE de Matlab. Como desgraciadamente no todos los economistas ni la mayor parte de otras profesiones conocen o saben manejarse con comodidad en Matlab puede llegar a ser difícil comunicar tus ideas, conclusiones o investigaciones sino es mandando un grafico o unas líneas con el output obtenido del programa. A veces esto no es suficiente, bien porque no solo el fin importa , o bien porque la aplicación será utilizada multitud de veces nos gustaría tener un programa mas interactivo que nos permita ir mas allá del simple grafico. He aquí donde GUIDE aparece como una potente herramienta, no solo por su sencillez de uso, sino por el grado de dinamismo que añade a tu programa. Uno de sus puntos fuertes es el de acotar las posibilidades que tendrá el usuario a la hora de manejar el programa, evitando así que alguien con poca experiencia o con poco tiempo se meta de lleno en las tripas de un programa que podría tener cientos de líneas de código. Pero ya va siendo hora que conozcamos mas a fondo la aplicación donde nos vamos a mover.

2-Conociendo la aplicación GUIDE

Vamos a abrir un rato Matlab para conocer GUIDE, para ello puedes escribir en la pantalla principal de matlab >>guide ,o bien puedes acceder a traves File->New->Gui tal como aparece indicado en la figura 1.

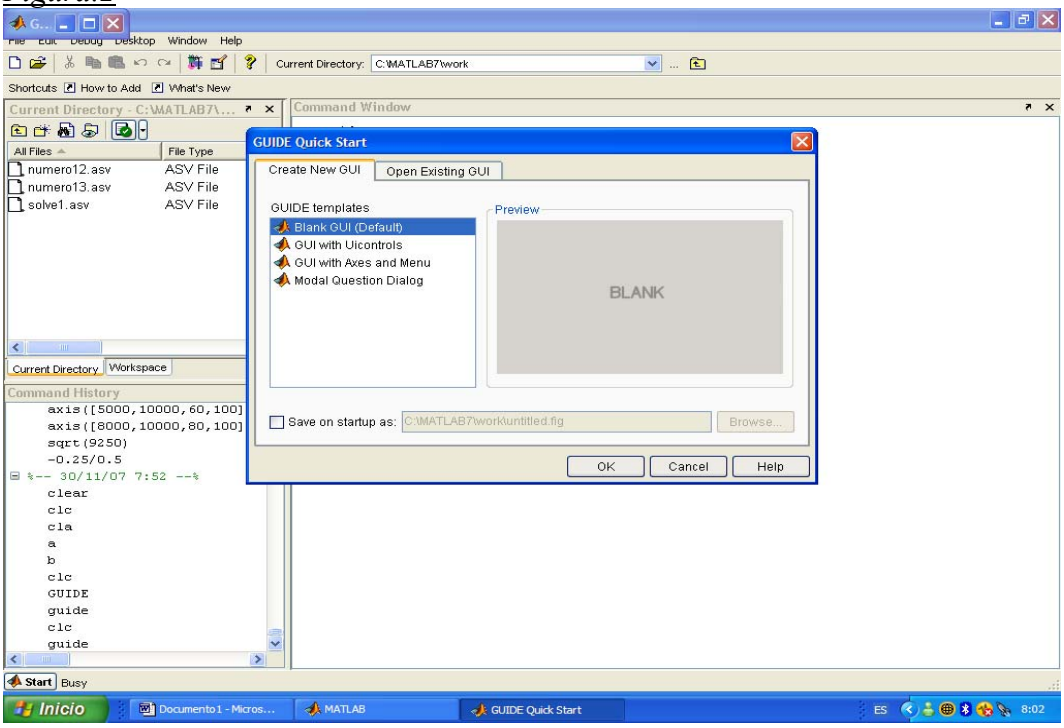
Figura.1



Automáticamente te aparecerá una pequeña pantalla con dos pestañas (Figura.2), una para comenzar una gui desde cero, y otra para abrir una gui que ya existe. También nos da la opción de usar plantillas prediseñadas con Uicontrols, ejes o diálogos. Nada de esto nos interesa por ahora, por lo que elegiremos Blank Gui(default). También tenemos la opción de salvar el documento antes de empezar. Esto es opcional, ya que es indiferente salvarlo antes o después de haber construido el diseño. Ahora solo queda hacer clic en OK.

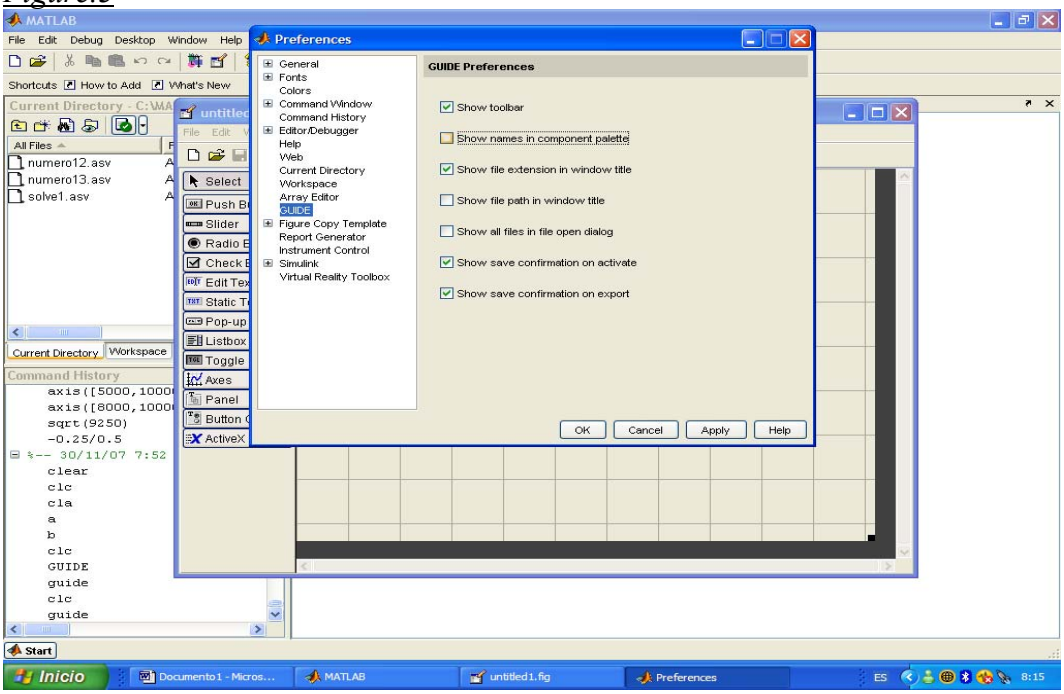
Lo siguiente que veremos es nuestro entorno de trabajo. Como podéis ver hay una serie de botones en la parte izquierda que son los instrumentos que vamos a usar para configurar nuestra gui. A primera vista puede parecer una amalgama de dibujitos sin sentido, así que para darle algo mas de sentido vamos a hacer click en file, preferences y allí vamos a picar en GUIDE, ahora a la derecha de este pantalla de opciones veremos "show names in component palette"(Figura.3) la señalamos, aceptamos y ahora también podemos empezar a familiarizarnos con los componentes de la gui. Para insertar cada uno de estos componentes en la gui basta con picar en el componente que queramos y arrastrarlo hacia la pantalla en blanco. Es recomendable insertar varios de estos componentes para sentirse mas cómodo en su uso. Por ejemplo *Axis, Panel, Text, Push Button,* son algunos de los mas usados, mas tarde profundizaremos en su uso y en su configuración, por el momento es suficiente ver como son y como aparecen en pantalla.

Figura.2



Ahora viene una de las partes mas importantes a la hora de saber funcionar con GUIDE, y es entender la correspondencia entre lo que nosotros dibujamos en el editor de guis y el código que matlab genera al mismo tiempo.

Figure.3

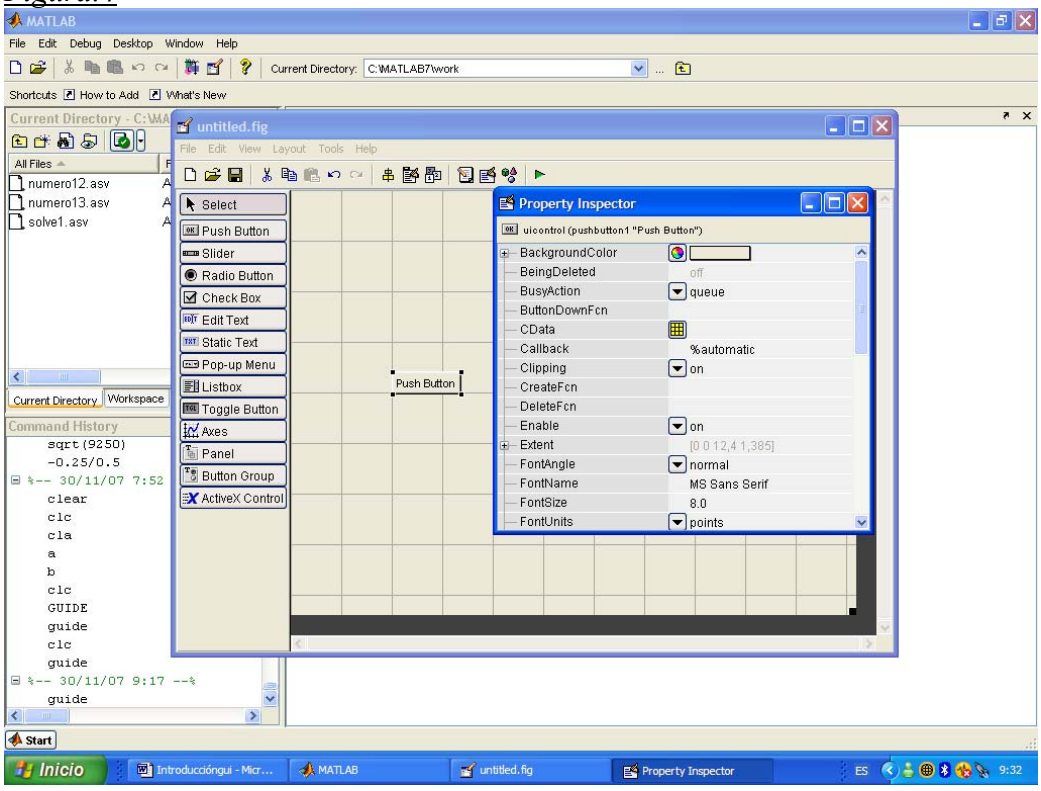


Debemos saber que toda GUI conlleva 2 archivos, un archivo .m y otro .fig. Digamos que el .fig es el antifaz y el .m es la maquina que hace que lo que vemos en el antifaz funcione. Nosotros por el momento estamos trabajando en el .fig, o dicho con otras palabras, estamos dándole forma a la mascara. Es aconsejable que, antes de diseñar nada

en Matlab lo hagamos nosotros a mano previamente. Hay 2 razones que hacen de lo anterior un punto básico; La primera es aclarar nuestras ideas, nuestro objetivo y la forma en que queremos obtenerlo. La segunda es mas pragmática, y es que una vez guardado el diseño, Matlab crea la maquina básica que lo va a hacer funcionar. Esa maquina tiene todo lo necesario para configurar la gui actual, pero NO esta garantizado que Matlab añada nuevas estructuras si introducimos nuevos elementos y los guardamos posteriormente por lo que podemos encontrarnos en serios problemas.

Antes de saltar a la caja negra que Matlab genera vamos a entender que opciones tenemos para configurar la parte visual de la gui. Vamos a introducir varios ejemplos, para empezar hagamos clic en *Push Button* y lo arrastramos dentro del espacio de diseño tal como esta en la Figura.4. Este tipo de botones se caracterizan por ejecutar algunos comandos cuando es presionado (ej. Ejecutar, Ayuda, Salir, Salvar).Si hacemos doble clic en el botón se nos aparece el denominado Revisor de Propiedades (Property Inspector),antes de volvernos locos con la cantidad de opciones que aparecen vamos a centrarnos en las que nos son mas útiles a la hora de programar una gui,y a las partes vitales que nos ayuden a entender que hace Matlab. En primer lugar vemos *Background color* , es decir Matlab nos permite elegir un color de fondo para nuestro botón, podéis hacer clic en el circulo que aparece a la derecha o si conocéis los colores RGB podéis elegir una combinación haciendo clic en el + situado a la izquierda de Backgroundcolor.

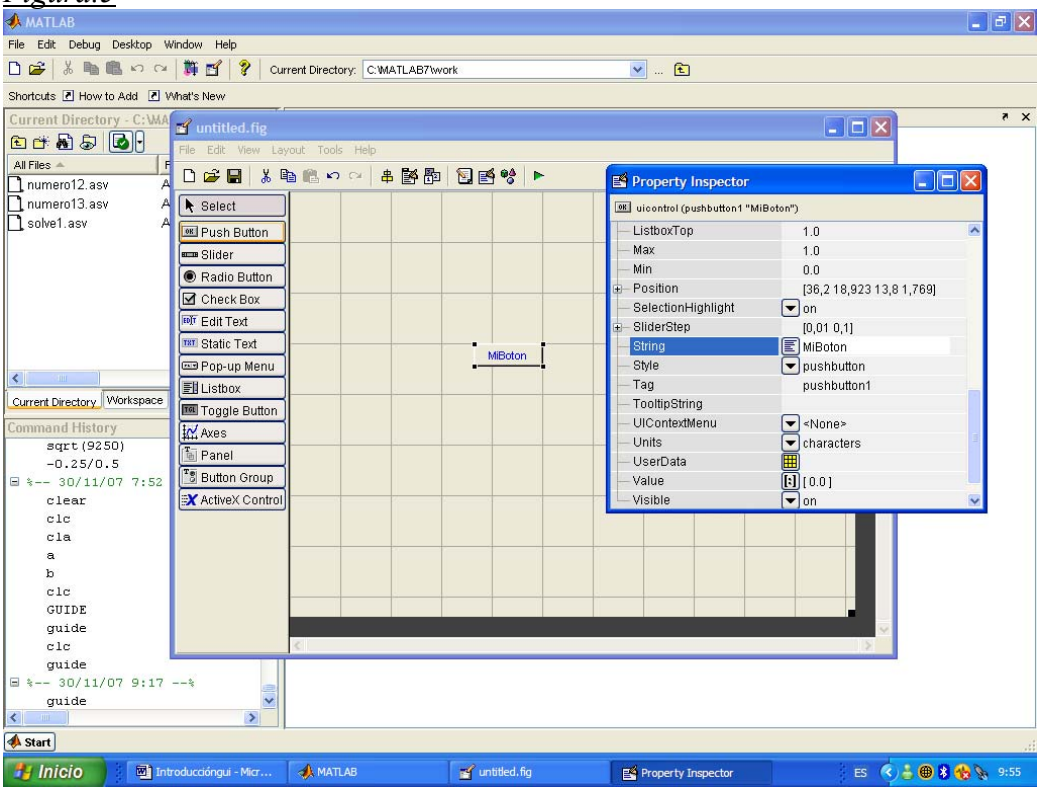
Figura.4



La siguiente opción en la que me detendré se llama Callback, esta propiedad es el enlace entre la parte visual y la maquinaria que se ejecuta detrás. Como veis aparece %Automatic, es decir una vez que guardáis el programa Matlab creara la estructura del callback de forma automática sin que hagáis nada mas. Mas tarde veremos como se configuran por ahora basta con conocer los usos. Mas abajo encontrareis Font Angle, FontName, FontUnits FontWeigth FontSize, Foregroundcolor todos ellos están

referidos a la configuración del tipo de letra que queráis elegir, así podéis elegir entre letra normal, cursiva, negrita ,tamaño, color o fuente. Simplemente cambiar algunos y ver que es lo que hacen es la mejor forma de habituarse a estas opciones. Lo que vamos a ver a continuación son dos opciones clave. La primera se llama *String* y contiene lo que se va a ver en la interfaz. Como ejemplo hacer clic en string y escribir MiBoton, veréis como cambia el nombre del botón si volvéis la vista a la gui.

Figura.5



La segunda opción es el Tag, esta es aun si cabe mas importante que la anterior. Es el nombre que Matlab va a asignar al callback. Por ejemplo si escribis boton1 Matlab creara boton1_callback. ¿Por qué es importante? Imaginaos una gui con 8 Push Buttons, que se llaman boton1 boton2... nadie es capaz de trabajar con eso sin volverse loco, por lo que se debe poner un nombre que os diga que hace ese botón en concreto sin que sea excesivamente largo, un ejemplo, imaginar que este botón ejecuta el programa, poniendo run en el tag veriais en algun lado del archivo .m que Matlab va a crear después run_callback, lo cual es mas que suficiente para darse uno cuenta de donde esta.

Podéis ver que cada tipo de botón contiene algunas diferencias en su editor de propiedades, pero analizarlos uno a uno es bastante tedioso, mas teniendo en cuenta que la estructura es parecida y después podremos verlos aplicados a algún problema concreto con lo que las dudas serán resueltas.

Antes de empezar a construir nuestra primera gui comentare algun ejemplo mas sobre los diferentes botones.

Edit text: Nos permite insertar objetos dentro de matlab.(Ej: Numeros relativos a parámetros de un modelo). Tambien nos permite visualizar outputs si asi se especifica.

Panel: Nos sirve para agrupar componentes de la gui, simplemente para mejorar su visualización.

Axes: Construye un eje donde podemos sacar plots del programa.

Static text: Al igual que panel produce un efecto únicamente visual. Este nos permite escribir lo que queramos para que aparezca en pantalla.

Radio button: Nos sirve para decidir entre varios opciones, parecido al slider.

Check box: En cierta manera cumple una funcion parecida a los radio button y los slider, pero en este caso podemos seleccionar mas de una variable a la vez.

La mayoría de estas opciones aparecerán mas tarde o mas temprano en los ejemplos que se muestran en las siguientes secciones, por lo que su configuración se explicara posteriormente. Por el momento el objetivo es entrar en contacto con las diversas posibilidades y soluciones que nos ofrece GUIDE.

3-Construyendo la primera GUI, pasos a seguir

Ahora que nos hemos familiarizado un poco con el entorno GUIDE vamos a construir una GUI lo mas simple posible, pero con las suficientes opciones como para que aprendamos algo de ella. Los pasos y los requisitos son los siguiente:

1º Tener un programa que funcione.
2º Tener un diseño en mente.
3º Configurar el diseño de forma que sea entendible para Matlab.

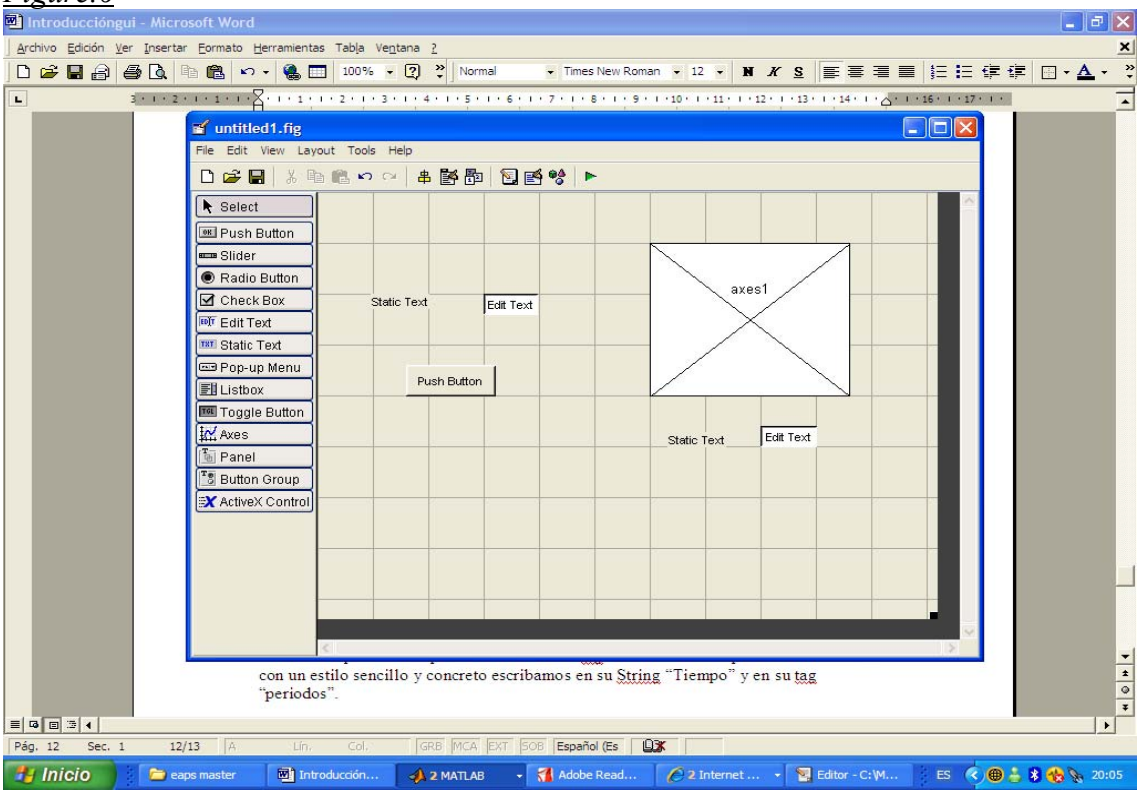
El primer paso lo vamos a cumplimentar con un sencillo programa que construye una serie numérica alrededor de un valor. Vamos a construir una interfaz que nos permita elegir la longitud de la serie y a su vez vamos a pedirle a Matlab que nos devuelva la media del proceso. De esta forma podremos visualizar de forma rápida como el proceso tiende a su media cuando su longitud tiende a infinito. Por supuesto la perturbación se distribuye normal con media 0 y varianza 1,o lo que es lo mismo, he usado el comando randn(*ver help randn). El código del programa es el siguiente:

```
%Programa que crea una secuencia de numeros alrededor del valor 10.
clear
clc
cla
%Programa
T   =   20;
for t   =   1:T
   x(t)=   10+randn;
end
media=mean(x)
plot(1:T,x,'k',1:T,media,'b:*')
title('Serie numerica alrededor del 10')
```

Este programa funciona correctamente y ya hemos analizado en el párrafo anterior lo que queremos de nuestra interfaz, a saber, que nos de un plot de la serie, así como su media, dándonos opciones de cambiar T, o lo que es lo mismo la longitud de la serie. Vamos a por el tercer paso y para ello retomemos Matlab, como vimos anteriormente teclear guide en la ventana de comandos y pedirle una gui nueva. Es el momento de usar lo aprendido anteriormente, vamos a insertar un eje(para nuestro plot), dos edit text(para introducir el valor de T y para recibir la media) dos static text (para facilitar la comprensión de los edit text) asi como un pushbutton, que será el que ejecute el programa. Veamos en la figura.6 el resultado de insertar todo lo anterior en Matlab.

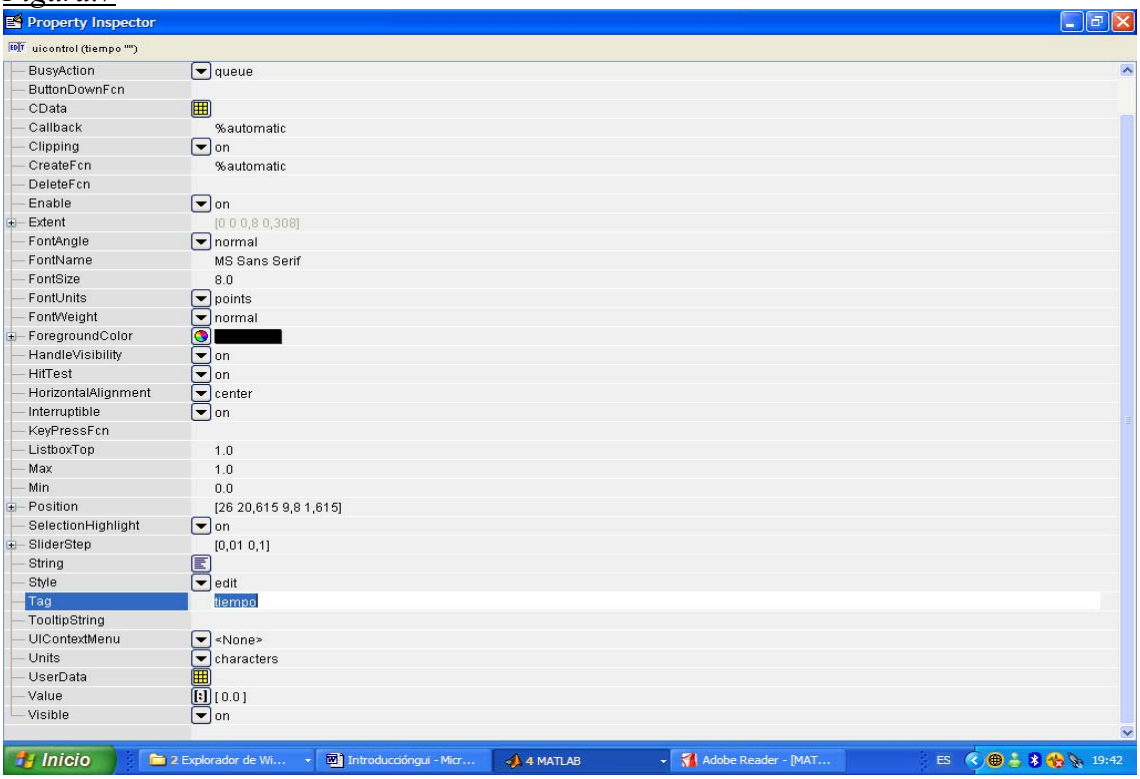
Lo siguiente que vamos a hacer es preparar los elementos que hemos insertado. Empezamos haciendo doble clic en un edit text y nos aparecerá el property inspector. Vallamos directamente al string y dejémoslo en blanco. Después pasemos al tag, como os comente antes este será el nombre del callback, por lo que usaremos algo que de verdad nos diga algo acerca de su naturaleza. Escribir “tiempo” en el tag y pulsar intro. El resultado debería ser parecido al mostrado en la figura 7.

Figure.6



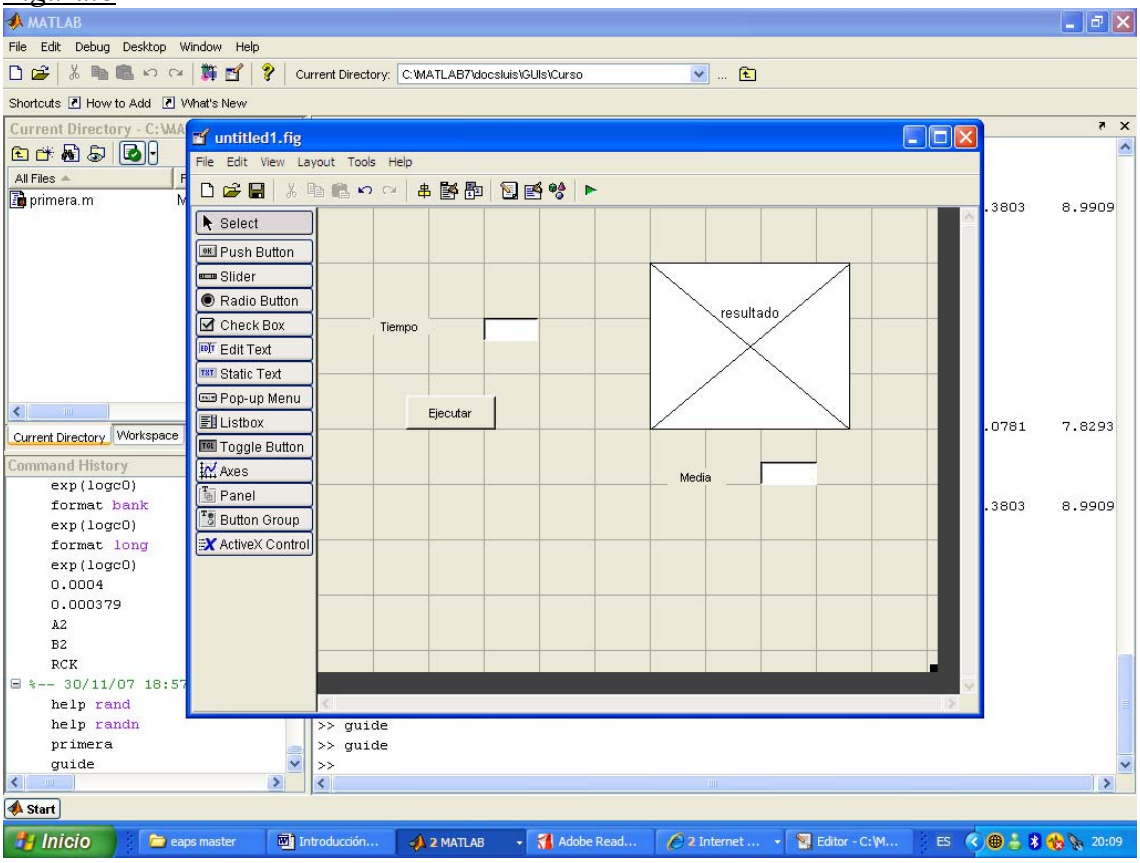
Pasemos a configurar uno de los static text. Al ser un boton de naturaleza meramente descriptiva no es tan importante lo que escribamos en su tag. De todas formas para ser coherentes con un estilo sencillo y concreto escribamos en su String "Tiempo" y en su tag "periodos".

Figura.7



El otro edit text será el que nos devuelva el valor de la media que ha calculado el programa. En este caso es obligatorio dejar en blanco su string, y llamaremos a su tag “media”. Al igual que antes denominamos al static text como Media(en su string) y “mean” en su tag. Ahora nos toca configurar el pushbutton, escribamos en su tag “run” y en su string “Ejecutar”. Y por ultimo doble clic sobre el eje de coordenadas, buscamos el tag (un poco mas complicado por la cantidad de opciones que nos ofrece Matlab) y escribimos “resultado”. De esta manera nuestra gui debería parecerse a la que vemos en la figura.8.

Figura.8

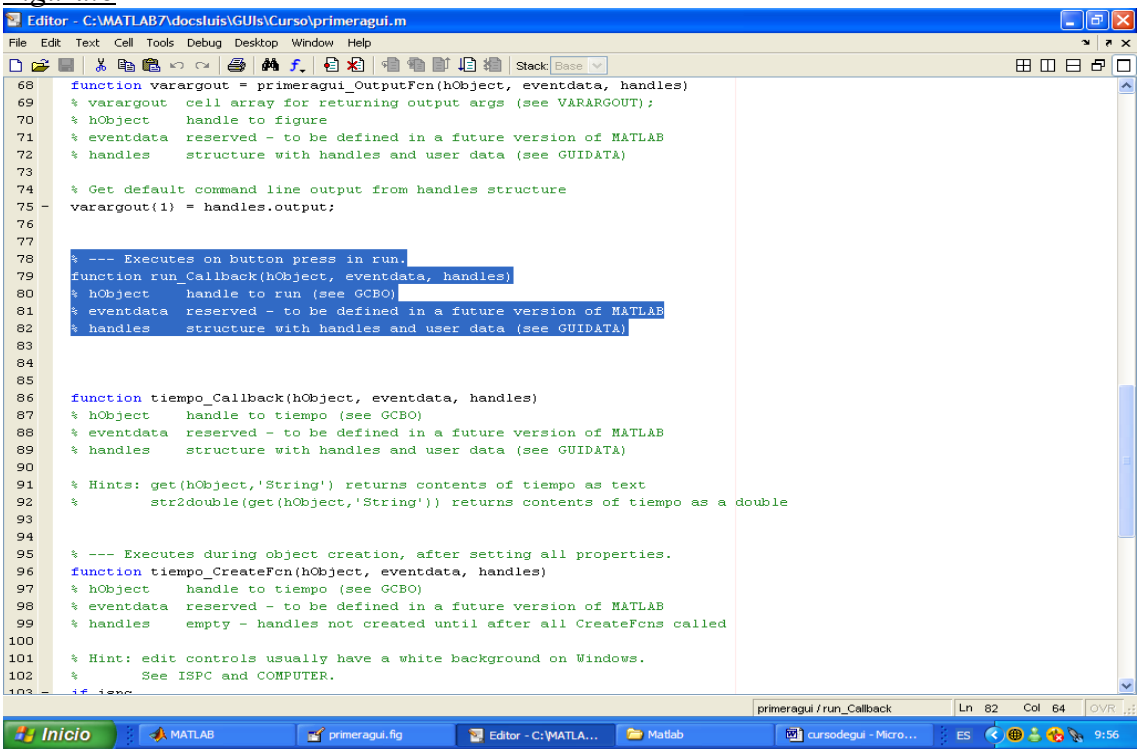


Por fin hemos acabado de configurar la mascara de la gui, por lo que hemos acabado la primera parte del trabajo. Para ver el resultado vamos a guardarla. La llamaremos primeragui ,y Matlab creara automáticamente dos ficheros, primeragui.m y primeragui.fig. Veréis como os salta automáticamente un fichero de matlab, acabáis de conocer a la caja negra donde en breve acabaremos de configurar nuestra gui, pero de momento no toquéis nada. Ahora hacer clic en el triangulo verde de la parte superior, que ejecuta la gui,(aunque obviamente nada sucederá ya que no tiene maquinaria configurada detrás)y podréis ver el resultado final.

Es el momento de echar un vistazo a la caja negra que ha creado Matlab. Su nombre (si habéis seguido las instrucciones) es primeragui.m. Lo primero al ver las 135 líneas de código que se han creado es no salir corriendo. Lo segundo ir conociendo las zonas clave donde tenemos que insertar el codigo extra que Matlab necesita para operar según nuestros deseos. Si recordais, anteriormente os mencione que los callbacks son la union entre las dos partes de la gui; Pues bien, esos callbacks son precisamente los que

tendremos que configurar, asi que de momento olvidar todo lo demas. En la figura 8 vereis en azul el callback asignado al pushbutton que hemos denominado “run”.

Figura.8

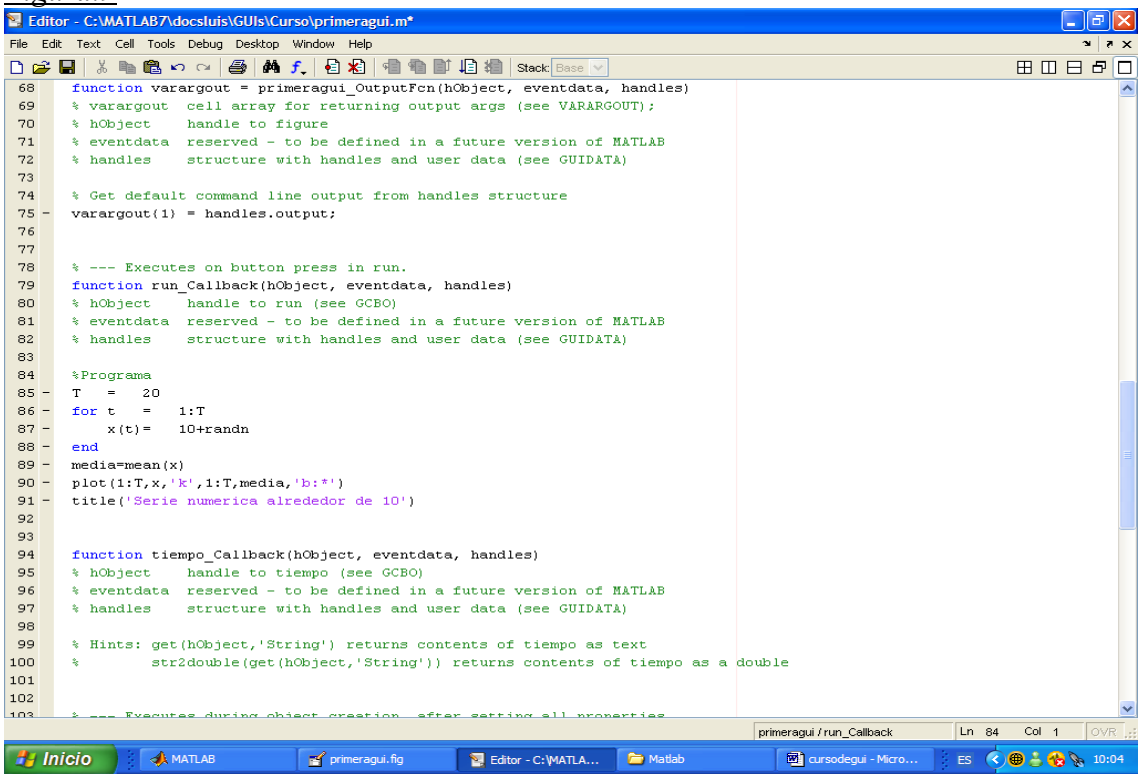


Es justo debajo del código function donde escribiremos las instrucciones. Y es aquí donde insertaremos nuestro programa. Básicamente lo que Matlab entiende es lo siguiente: Nosotros pulsamos Ejecutar, entonces Matlab a través del callback busca la subfunción dentro del fichero .m llamada run_callback. En ese momento Matlab encuentra con que esa subfuncion tiene unas instrucciones(que hemos añadido nosotros) y que no es nada mas ni nada menos que el programa. ¿Cómo sabe Matlab cual es el valor de T que hemos asignado?¿Cómo sabe donde mandar el plot? Eso es algo que también deberemos especificar pero algo mas tarde. Por el momento copiemos el programa original y peguémoslo debajo de run_callback como viene indicado en la figura 9. Si os fijais con detenimiento encontrareis algo en ese programa que falta con respecto a la versión original. Efectivamente no estan los comandos clear,clc,cla(*ver help clc,cla,clear) que si estaban en el programa original. Esto es basico, ya que si Matlab lee clear dentro del programa se le olvidaran los valores e instrucciones que le hayamos dado previamente y no obtendremos nada.

Como prueba de que de verdad estamos dando instrucciones a Matlab podéis volver a la parte visual de la interfaz(guardando previamente el cambio que habéis hecho), hacer de nuevo clic en el triangulo verde y comprobar que mágicamente ¡aparece vuestro plot!. Esto mas que una regla es una excepción. Es decir si solo tenéis un plot y un axes, Matlab hará la asignación automática y podréis verlo sin darle otra instrucción. Pero como las cosas no son tan simples la mayoría de las veces vamos a escribir el código con propiedad. Para ello vamos a añadir la siguiente frase justo encima de plot: “axes(handles.resultado)”. Esto hace lo que Matlab había realizado de forma automática, y es asignar un plot a un eje. En el caso anterior, al haber solo 1 plot y un eje la asignación fue automática. Además vamos a añadir debajo de plot el siguiente

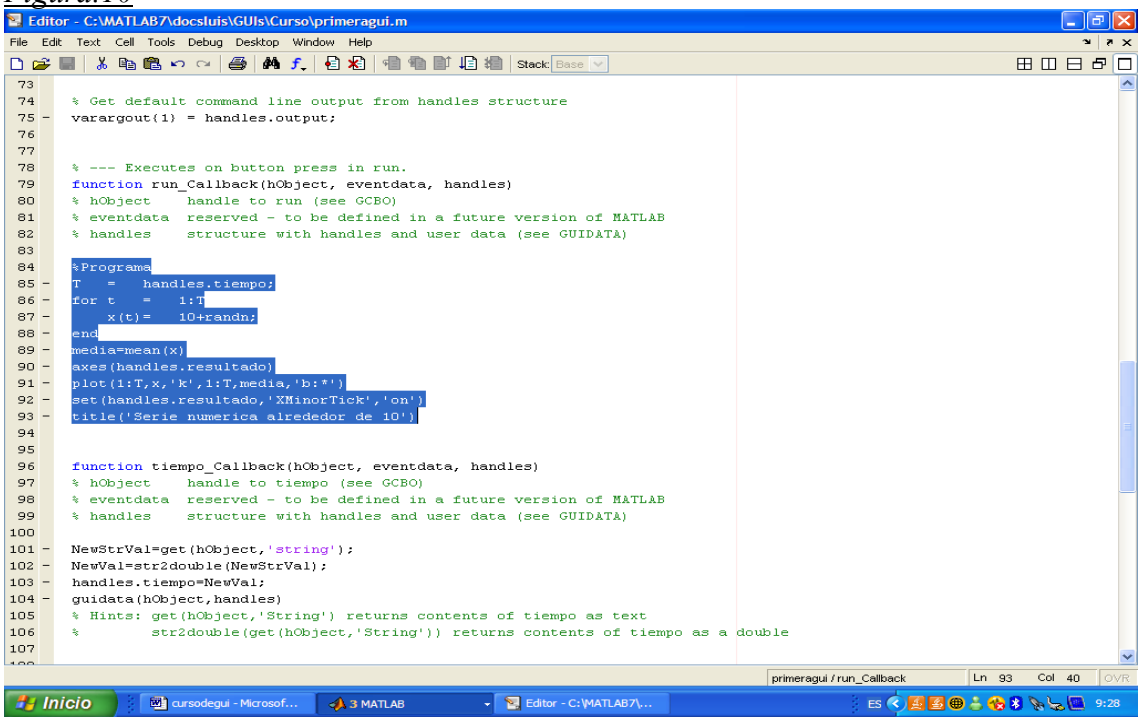
codigo: “set(handles.resultado,'XMinorTick','on')” esto no estrictamente necesario pero nos ayudara a seguir conociendo opciones. En este caso el comando set

Figura.9



(*ver help set) implementa una serie de propiedades a un objeto. La propiedad es Xminortick y on es el estado en que queremos esa propiedad. Podemos ver el resultado final en la figura 10 resaltado en azul.

Figura.10



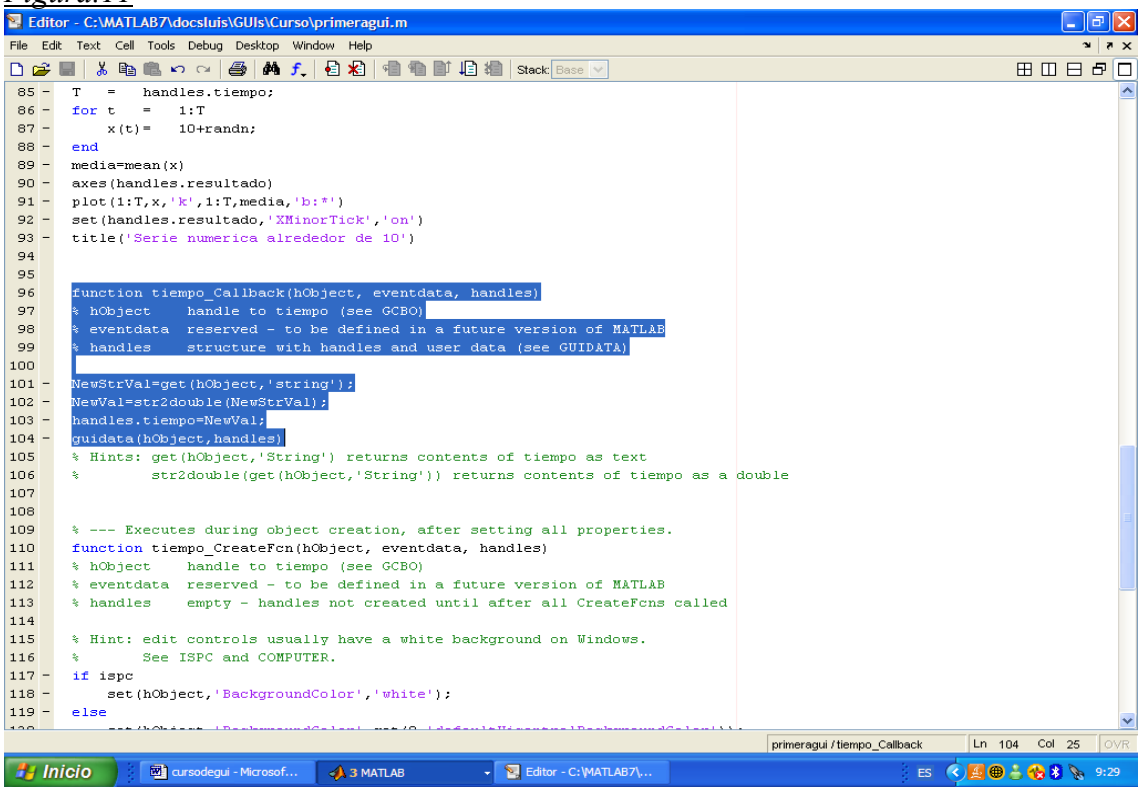
Ya hemos acabado de configurar nuestro primer callback. Ahora vamos a configurar un callback que nos permita especificarle a Matlab una longitud para la serie numérica. Este proceso se realiza de la siguiente manera:

1º Se captura el string de un edit text (en este caso del edit text tiempo).
2ºSe convierte en un numero (para Matlab en principio solo es codigo).
3ºSe guardan los datos.
4ºSe sustituye el valor inicial del programa por el nuevo capturado.

Este esquema es importante seguirlo paso a paso. Para ello situémonos en tiempo_Callback (figura.11) y pongamos debajo de function:

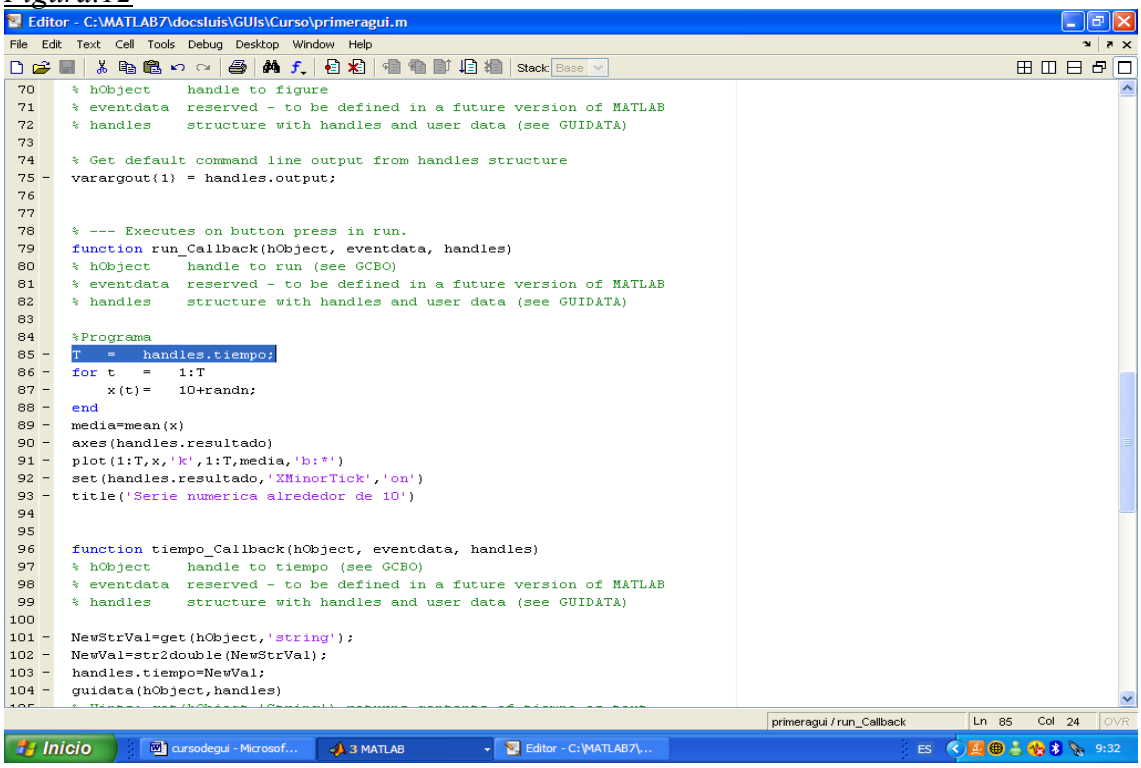
```
NewStrVal=get(hObject,'string');
NewVal=str2double(NewStrVal);
handles.tiempo=NewVal;
guidata(hObject,handles)
```

Figura.11



Vamos a explicar lo anterior brevemente; La primera frase esta creando un objeto compuesto por el string que asignamos al edit text. La segunda convierte el objeto en un numero. La tercera denomina de una forma especifica al nuevo valor, mientras que la cuarta guarda el nuevo handles. Ahora ya tenemos dentro de la caja negra el valor que ingresamos en la gui, pero,¿sabe Matlab donde asignarlo? Aun no, por lo que debemos especificárselo. Volvamos a run_callback. Alli veremos que T tiene asignado el valor 20. Pero nosotros queremos que tenga handles.tiempo. Nada mas facil, sustituir 20 por handles.tiempo de la misma forma que veis en la figura.12. Ya van quedando menos cosas para terminar la gui. Lo siguiente sera decirle a Matlab que nos de el valor de la

Figura.12

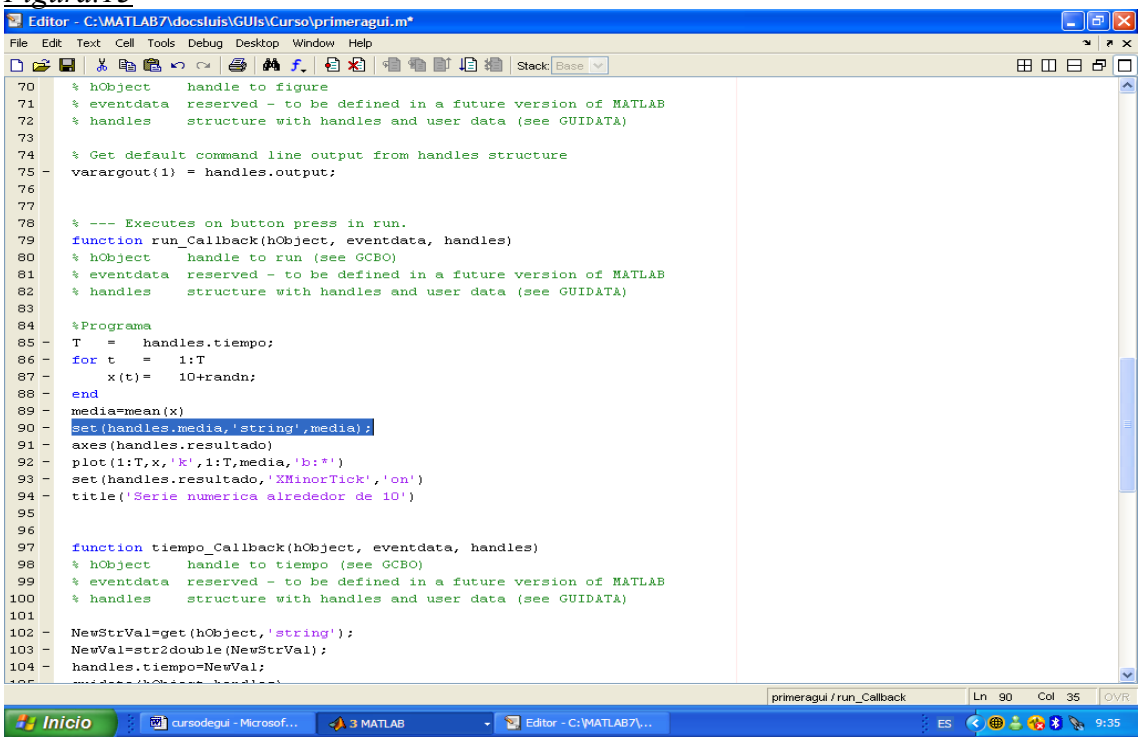


media. Para ello vamos a usar el comando set de la siguiente forma:

set(handles.media,'string',media);

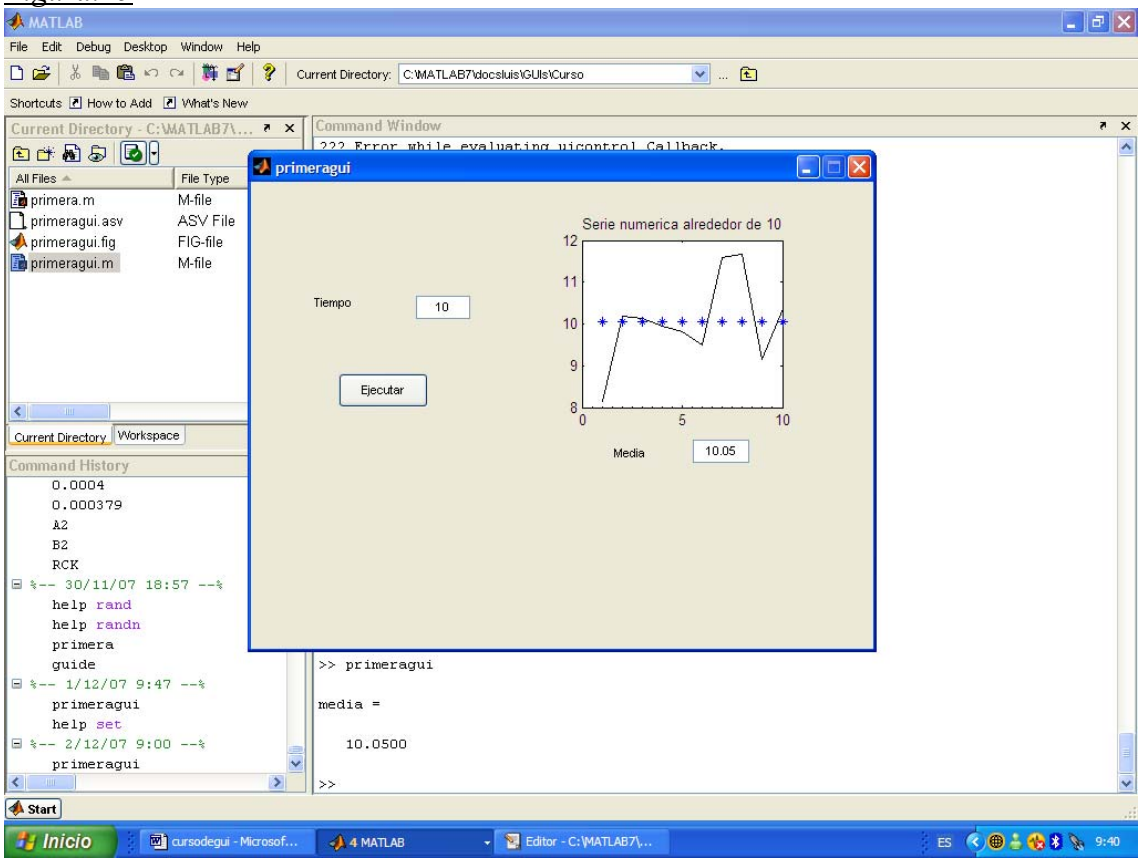
En este codigo se le dice a Matlab que coloque en el string de handles.media el valor de la media que se ha obtenido en el programa.

Figura.13



Esto lo vamos a añadir justo debajo del calculo de la media, para que los objetos comunes estén lo mas cerca posible. El resultado lo podéis ver en la figura 13. Por fin ha llegado la hora de la verdad, guardar el archivo, aseguraos de que el .fig y el .m están en el mismo fichero, entrar en matlab y escribir primeragui. El resultado debería parecerse al de la figura.15 donde si os dais cuenta aparece la media del proceso donde antes no había nada.

Figura.15



Una vez que se os aparezca vuestra gui escribir un numero en el espacio reservado para tiempo (y que debería estar correctamente indicado por un static text) y presionar Ejecutar. El resultado ya lo podéis ver arriba, bastante pésimo, pero esta es la primera de muchas, por lo que habrá tiempo de hacer numerosas mejoras.

*Tanto este codigo como el de todas las demas gui's se suministra al final de las notas. Cualquier duda que surja acerca de alguna configuración de comandos deberia comprobarse alli.