

# **Un sistema computacional interactivo que simula la aritmética de punto flotante**

**Taller.m versión 1.0**

Lic. Fernando Saldaña Jiménez  
Escuela de Matemáticas  
Universidad Autónoma de Coahuila

Dr. Jesús López Estrada  
Facultad de Ciencias  
Universidad Nacional Autónoma de México

Dr. Humberto Madrid de la Vega  
Centro de Investigación en Matemáticas Aplicadas  
Universidad Autónoma de Coahuila

Marzo 2001

# Í n d i c e

<b>Prefacio</b>	<b>ii</b>
<b>1 MANUAL DEL USUARIO</b>	<b>1</b>
1.1 Presentación . . . . .	1
1.2 Objetivo . . . . .	3
1.3 Descripción general del Taller . . . . .	3
1.4 Antecedentes . . . . .	4
1.5 Temas . . . . .	5
1.6 Estructura modular del Taller . . . . .	6
1.7 Guía para la utilización del Taller mediante un ejemplo . . . . .	9
<b>2 MANUAL DE REFERENCIA</b>	<b>17</b>
2.1 Funciones de Matlab para crear objetos de interfaz . . . . .	18
2.2 Funciones que simulan la aritmética de punto flotante . . . . .	22
2.3 Funciones para la creación de las ventanas del Taller . . . . .	26

## Prefacio

Sabemos que actualmente las computadoras son utilizadas en una gran diversidad de actividades cotidianas que van desde la prestación de servicios -en bancos, hoteles, hospitales, terminales aéreas y terrestres- hasta el diseño y control de procesos industriales, pero es importante señalar que fue la necesidad de resolver problemas matemáticos que involucraban una gran cantidad de cálculos numéricos, la razón que propició su invención.

En el ámbito científico la computadora es extensamente usada para resolver modelos matemáticos que representan alguna situación física o fenómeno social. Los problemas de los que derivan estos modelos provienen de los más diversos campos como: Astronomía, biología, economía, física, ingeniería, etc.

Cuando un modelo matemático es resuelto satisfactoriamente por la computadora, éste es aprovechado como patrón para resolver problemas similares al original e incluso para simular algunas variantes del mismo y hacer predicciones.

Prácticamente todas las ciencias recurren al uso de la computadora tanto para resolver los modelos matemáticos de sus problemas como para la simulación de variantes en éstos últimos. La gran ventaja de ello es que resulta mucho más económico ensayar varios diseños o situaciones en la computadora que construirlos y probarlos físicamente. De esa manera se puede simular p. ej. el efecto del aire sobre una ala de avión; hacer variaciones en el modelo puede significar que se están probando diferentes diseños de ala de avión.

Sin embargo, siempre que se formula un modelo matemático y se resuelve en la computadora se cometen errores cuyas causas son inevitables. Las principales fuentes de error son: Error de modelación, incertidumbre en los datos, error de redondeo y error de discretización.

*Error de modelación.* Cuando un problema consta de muchísimos aspectos o factores a considerar se omiten o aproximan aquellos que puedan tener poco efecto en la solución a fin de obtener un modelo lo más sencillo posible y factible de ser resuelto. La omisión o aproximación de tales factores constituye el error de modelación. Por ejemplo hay modelos de crecimiento económico o dinámica poblacional que asumen la disponibilidad de recursos ilimitados.

*Incertidumbre en los datos.* Esto se refiere a los datos de entrada con que

cuenta el modelo. Cuando esta información consta de mediciones u observaciones, siempre existe la posibilidad de que se cometan errores debido a la naturaleza finita de los instrumentos de medición.

*Error de redondeo.* Muchos números requieren de una cantidad infinita de cifras para su representación numérica exacta. Tal es el caso de números como  $\pi$  ó  $\frac{1}{3}$ . Sin embargo las computadoras trabajan con una cantidad finita de cifras por lo que dichos números nunca pueden ser introducidos exactamente en la computadora sino que se tiene una aproximación de ellos. La diferencia entre el valor numérico exacto y la aproximación que la computadora ha hecho del mismo es el error de redondeo.

Esta situación también se da aunque un proceso de solución inicie con números cuya representación en la computadora sea exacta, pues el desarrollo de los cálculos provocará que eventualmente se cometa el redondeo. Por ejemplo es posible que dos números tengan sus valores exactos en la computadora pero el cociente de ellos requiera de una infinidad de cifras para su representación exacta.

*Error de discretización.* Otra forma en que se pone de manifiesto la naturaleza finita de las computadoras es la necesidad de reemplazar problemas continuos por discretos. Por ejemplo, la integral de una función continua requiere conocer el integrando y todo el intervalo de integración, mientras que una aproximación numérica a la integral usará valores del integrando en un número finito de puntos dentro del intervalo de integración.

Salvo casos triviales, el error de discretización afecta la solución numérica de todos los problemas continuos.

El área que concierne a la solución de modelos matemáticos con la ayuda de la computadora se conoce como cómputo científico, el cual se apoya en dos ciencias que interactúan entre sí: La ciencia computacional y las matemáticas. La primera estudia y describe las capacidades y limitaciones de la computadora mientras que la segunda provee el lenguaje para la formulación de los modelos y proporciona las bases teóricas para determinar si un modelo tiene o no solución y en caso de que exista si ésta es única o no.

La interacción entre estas dos ciencias ha propiciado el desarrollo de técnicas numéricas cada vez más refinadas y siempre sustentadas teóricamente que permiten resolver modelos cada vez más complejos de manera satisfactoria.

Este conjunto de técnicas numéricas y teorías matemáticas es conocido como *análisis numérico* y constituye la mayor parte del cómputo científico.

Dado que el error de redondeo es inevitable al momento de efectuar cálculos en la computadora, uno de los objetivos del análisis numérico es implementar algoritmos numéricos que inhiban al máximo el crecimiento del error por redondeo.

Para brindar una idea del trabajo interno de una computadora cuando efectúa cálculos numéricos se ha elaborado el *Taller de Análisis Numérico* (Taller.m versión 1.0), un sistema computacional interactivo concebido como una colección de problemas matemáticos cuya solución numérica ilustra algunas de las dificultades de trabajar en aritmética de precisión finita, así como alternativas para superarlas gracias al estudio del análisis numérico.

En el presente documento, el capítulo 1 constituye el *manual del usuario*, en el cual se hace una presentación y descripción general del Taller, se menciona su objetivo, los temas que contempla así como una guía para su utilización.

El capítulo 2 conforma el *manual de referencia*: En el mismo se presenta una lista de las principales funciones de Matlab así como de aquellas escritas por un servidor que fueron requeridas para la creación del Taller.

El Taller de Análisis Numérico fue implementado en Matlab v.5.3 y tiene el propósito de servir como apoyo didáctico para introducir a los estudiantes de las carreras de ciencias exactas e ingenierías a sus cursos de análisis numérico o métodos numéricos.

Las principales características del Taller son:

- Es un sistema interactivo de fácil manejo.
- Se ha diseñado en el concepto de ventanas de interfaz.
- Los temas que contempla se imparten en los cursos de análisis numérico.
- No requiere conocimientos previos en la materia.
- El usuario tiene acceso a ejemplos y/o ejercicios.

## **R e c o n o c i m i e n t o :**

La creación del sistema computacional **Taller.m** fue primeramente motivada por la lectura del sobresaliente artículo de George Forsythe: *Pitfalls in Computation, or Why a Math Book Isn't Enough*.

# Capítulo 1

## MANUAL DEL USUARIO

### 1.1 Presentación

Dentro de los últimos 15-20 años se ha presentado un fenómeno impactante a nivel mundial: El abatimiento cada vez más evidente en el costo de las computadoras a la vez que ha aumentado sorprendentemente la capacidad y velocidad de las mismas para realizar las más diversas y sofisticadas operaciones. Más recientemente, la sociedad en general ha sido bombardeada y cautivada por el mercado de la más moderna generación de computadoras, la computadora personal (PC por sus siglas en inglés).

Las circunstancias anteriores han propiciado enormemente el uso extensivo de estas máquinas electrónicas digitales. En el ámbito científico este uso se acentúa debido a la necesidad cotidiana de resolver problemas matemáticos cada vez más complejos y con volumen creciente de operaciones, pues ello da lugar a la tarea ineludible de efectuar una gran cantidad de cálculos numéricos, constituyéndose la computadora para esta labor en una herramienta de trabajo indispensable.

Como consecuencia, el apoyo que brinda la computadora en dicho renglón ya no sólo es aprovechado por los investigadores especializados sino también por profesionistas que recién ingresan al campo científico e incluso por estudiantes de ciencias exactas e ingenierías quienes le confieren a la máquina sus algoritmos y la encomienda de que realice los cientos, miles o millones de operaciones aritméticas que deben hacerse como etapa previa a la solución de sus problemas matemáticos.

Lo alarmante es que muchas de estas personas usan la computadora en forma irreflexiva, pensando que por el simple hecho de programar una fórmula se encontrará la solución correcta al problema. Ello repercute en que tales usuarios acepten sin cuestionamiento alguno todo resultado obtenido computacionalmente.

Por otra parte, la mitificación que induce el auxiliarse de la computadora ha llevado también en muchos casos a la situación lamentable de considerar un resultado emitido por la computadora como indefectiblemente correcto, llegando incluso a anteponer el valor o los valores numéricos exhibidos en el monitor como la prueba matemática de que el problema ha sido resuelto.

Ante esta realidad, es preciso alertar a quienes utilizan el cómputo científico como parte de su trabajo acerca de la posibilidad siempre latente de que una computadora proporcione resultados incorrectos y hasta absurdos a pesar de que la formulación matemática para la solución de algún problema haya sido programada inequívocamente. Más aún, este hecho es susceptible de ocurrir tan sólo con la ejecución de unas pocas operaciones aritméticas y mediante sencillos cálculos numéricos.

Con el fin de brindar una idea del trabajo interno de una computadora al hacer su aritmética y mostrar algunos de los factores que sutilmente provocan la obtención de resultados incorrectos o incongruentes, se ha elaborado un taller computacional que ilustra la solución de problemas matemáticos mediante el cómputo científico.

El *Taller de Análisis Numérico* (Taller.m versión 1.0) es un sistema computacional implementado en Matlab v.5.3 que muestra una simulación de la Aritmética de Punto Flotante, que es la aritmética que toda computadora realiza al efectuar cálculos numéricos. En el mismo no sólo se abordan las dificultades de trabajar en aritmética de precisión finita sino también formas de superarlas mediante el análisis numérico.

En el Taller las operaciones aritméticas de punto flotante se realizan en base 10 pues es el sistema de numeración con el que todos estamos más familiarizados. Se señala este detalle porque si bien la mayoría de las computadoras trabajan internamente en base 2, los aspectos o fenómenos numéricos que se pretenden destacar en el Taller no son propios de una base en particular.



## 1.2 Objetivo

El Taller de Análisis Numérico se ofrece como material introductorio y complementario para los cursos de análisis numérico o métodos numéricos que se imparten en las carreras de ciencias exactas e ingenierías.

Como material introductorio, la interacción por parte de los alumnos con algunos temas del Taller en un centro de cómputo puede constituir la actividad predominante de las primeras sesiones de la materia de análisis numérico.

De esta forma será posible presentar de manera directa y atractiva a los estudiantes las características y limitaciones de la aritmética de punto flotante y familiarizarlos de inmediato con los fundamentos del cómputo científico como: Precisión de la computadora, error de redondeo, unidad de redondeo, además de enseñarles la notación para describir a los números de punto flotante.

Conforme se avanza en el curso de análisis numérico y se enseñan los fenómenos numéricos que más suelen presentarse como consecuencia de trabajar en aritmética de precisión finita, la utilización del Taller como material complementario permitirá corroborar de manera rápida y sencilla la solución de algún problema que se analizó en clase y así confrontar al estudiante con el fenómeno numérico en cuestión utilizando la computadora, así como brindarle una alternativa para evitar dicha situación desfavorable y mejorar aceptablemente el resultado; de esa manera se le hace ver al alumno que la posibilidad de resolver modelos matemáticos de manera satisfactoria puede ser muy amplia si se conocen y utilizan técnicas numéricas con sustento matemático, alentando así el interés por el estudio del análisis numérico.

## 1.3 Descripción general del Taller

El Taller es un sistema interactivo, amigable y de fácil manejo pues se ha diseñado sobre la moderna plataforma de ventanas de interfaz, es decir, mediante la disposición de una barra de menús, botones de activación y una agradable presentación de cuadros con texto que proporcionan las instrucciones necesarias e información general, el usuario es introducido a través de la exposición de ejemplos especialmente seleccionados para cada uno de los temas a las sorprendentes patologías numéricas que tienen lugar cuando se efectúan cálculos en aritmética de punto flotante.

Los ejemplos que se han preparado son muchos de ellos elementales (al-

gunos incluso corresponden a temas conocidos desde el nivel académico medio básico y medio superior), requieren de relativamente pocas operaciones aritméticas y se trabajan con pocos dígitos de precisión. Pero dichos ejemplos son representativos ya que ilustran lo que realmente sucede cuando la computadora trabaja con problemas matemáticos grandes que involucran muchísimas operaciones aritméticas (miles o millones) con algunas cifras más de precisión.

Los ejemplos muestran cómo es resuelto en la computadora un problema aplicando directamente un algoritmo tal vez estudiado previamente en el salón de clases o proveniente de un texto de matemáticas; de esta forma se pretende lograr que el alumno descubra que un procedimiento o formulación matemática que es infalible en el pizarrón, puede ser poco o nada eficaz al momento de programarse y ejecutarse en la computadora.

Asimismo la realización de los ejemplos destaca la presentación de algoritmos alternativos a los conocidos y que pueden proporcionar la solución 'aceptablemente' correcta al problema planteado.

Una característica interesante de este sistema es que le brinda al usuario la opción de experimentar con otros ejemplos del tema en que se encuentra pero introduciendo él mismo los valores que se solicitan. Ello lo puede hacer antes o después de trabajar el ejemplo ya elaborado.

Las ventajas de diseñar el Taller en el concepto de ventanas de interfaz son:

- Es una presentación acorde con los recientes sistemas de interacción usuario-computadora.
- Al disponer de menús y botones que con la ayuda del cursor y el ratón se activan para ejecutar las acciones deseadas, desaparece el trabajo de validar datos de entrada que se solicitarían para tales propósitos.
- Permite al usuario realizar muchos experimentos numéricos en poco tiempo eximiéndolo de la tediosa labor de hacer aritmética con lápiz y papel.
- También le evita la tarea de elaborar, editar y compilar programas.

## 1.4 Antecedentes

Se cuenta con versiones precedentes del Taller elaboradas en BASIC y FORTRAN; en tales sistemas el medio de interacción entre el usuario y los progra-

mas es el teclado, particularmente la utilización de las teclas correspondientes a los dígitos ya que para tener acceso a algún tema, ejemplo u otra actividad es necesario proporcionar como dato de entrada el valor numérico con que se ha designado a la opción elegida.

El propósito de implementar el Taller en Matlab es darle una nueva y mejor presentación, acorde con el moderno sistema de interacción entre el usuario y la computadora por medio de menús y botones de activación, aprovechando las enormes facilidades que ofrece este lenguaje para el diseño de ventanas de interfaz además de su poderoso ambiente para la realización de cómputo numérico y graficación.

## 1.5 Temas

Los temas que se contemplan en el Taller de Análisis Numérico v.1.0 son los siguientes:

1. *'Mi computadora'*. Aquí se dan a conocer las características básicas del sistema de números de punto flotante de la computadora anfitrión. Se determina la base, la precisión, los exponentes mínimo y máximo y la unidad de redondeo.
2. *Operaciones básicas en aritmética de punto flotante*. Se ilustra que en aritmética de punto flotante no siempre se cumplen las propiedades de los números reales. Se exhibe y analiza la asociatividad de la suma y el producto de 3 números, la propiedad del elemento neutro aditivo y la cerradura del producto.
3. *Cancelación numérica o catastrófica*. Este fenómeno ocurre cuando en la computadora se realiza la operación de sustracción entre 2 números muy cercanos entre sí.
4. *Sumas*. Por medio de la presentación de conocidas series de números positivos se hace notar que el cálculo elemental  $\sum_{i=1}^n x_i$ , imprescindible en gran parte del cómputo científico, no es de ningún modo trivial en la máquina.
5. *Resolución de la ecuación de segundo grado*. Con este tema conocido desde el nivel medio básico se destaca nuevamente la repercusión del

fenómeno de cancelación numérica, por lo que la obtención de las raíces de una ecuación cuadrática se debe trabajar con mucho cuidado en aritmética de punto flotante.

6. *Resolución de sistemas lineales  $2 \times 2$* . Utilizando distintos métodos conocidos desde el nivel medio superior (Regla de Cramer, Eliminación Gaussiana con y sin pivoteo) se muestra la trascendencia del error de redondeo -situación inevitable al trabajar en aritmética de precisión finita- sobre la solución del sistema de ecuaciones más pequeño posible.

## 1.6 Estructura modular del Taller

La organización de los primeros módulos del Taller es la siguiente:

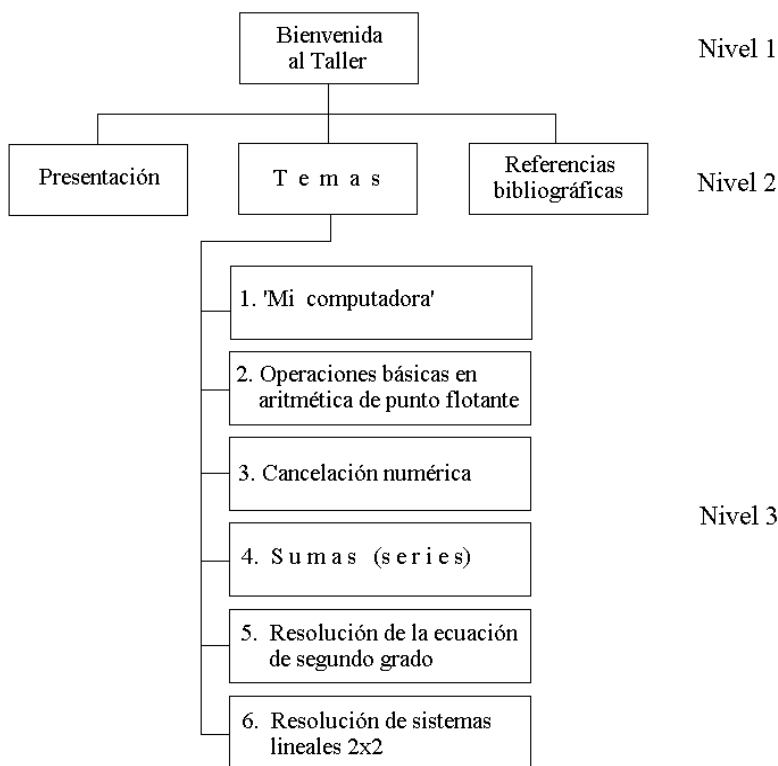


figura 1.1. Módulos del Taller correspondientes a los primeros tres niveles.

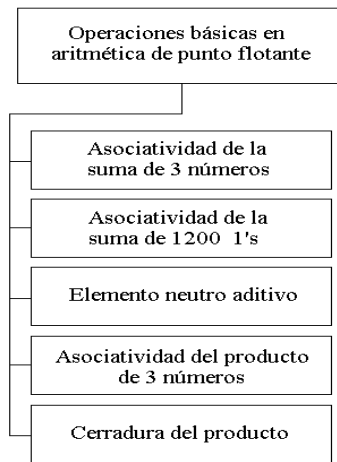


figura 1.2. Ejemplos del tema 'Operaciones básicas en aritmética de punto flotante'.

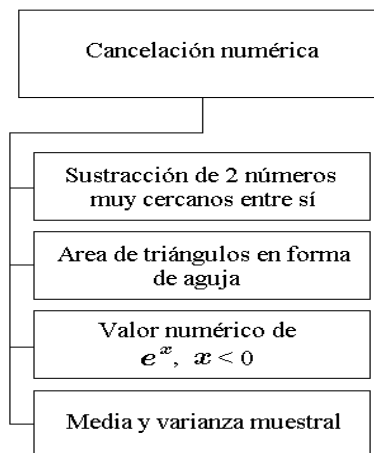


figura 1.3. Ejemplos del tema 'Cancelación numérica'.

En el nivel 4 están los ejemplos que se proporcionan para los temas 2 al 6. Los diagramas correspondientes a dicho nivel se muestran en las figuras 1.2 a la 1.6.

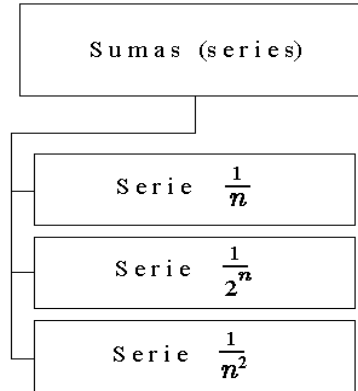


figura 1.4. Ejemplos del tema 'S u m a s'.

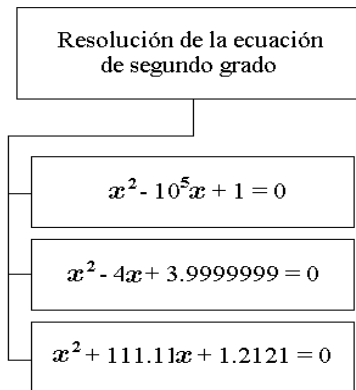


figura 1.5. Ejemplos del tema 'Resolución de la ecuación de 2o. grado'.

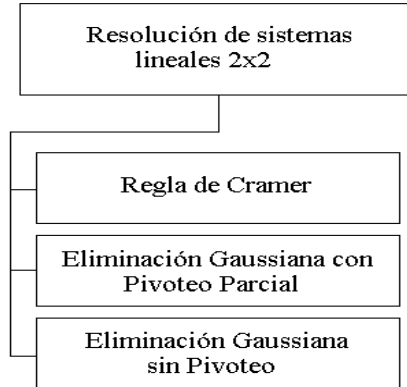


figura 1.6. Ejemplos del tema 'Resolución de sistemas lineales  $2 \times 2$ '.

## 1.7 Guía para la utilización del Taller mediante un ejemplo

Los programas que conforman el Taller ocupan en conjunto poco menos de 0.4 megabytes de memoria, prácticamente la cuarta parte de un disquete de 3.5 pulgadas de alta densidad. Suponga que dentro de la unidad C en la computadora se encuentra la carpeta **ananum** (que significa análisis numérico) la cual contiene una copia de los programas del Taller. Estando en Matlab localizamos y abrimos dicha carpeta escribiendo

```
>> cd c:\ananum
```

oprimiendo enseguida la tecla de entrada. De esta forma estamos listos para trabajar con el sistema. Al escribir la palabra

```
>> taller
```

y dar entrada aparece una ventana de bienvenida al Taller (vea fig. 1.7).

La ventana cuenta con botones en el lado derecho que se activan haciendo click con el ratón cuando el cursor o flecha está sobre alguno de ellos.

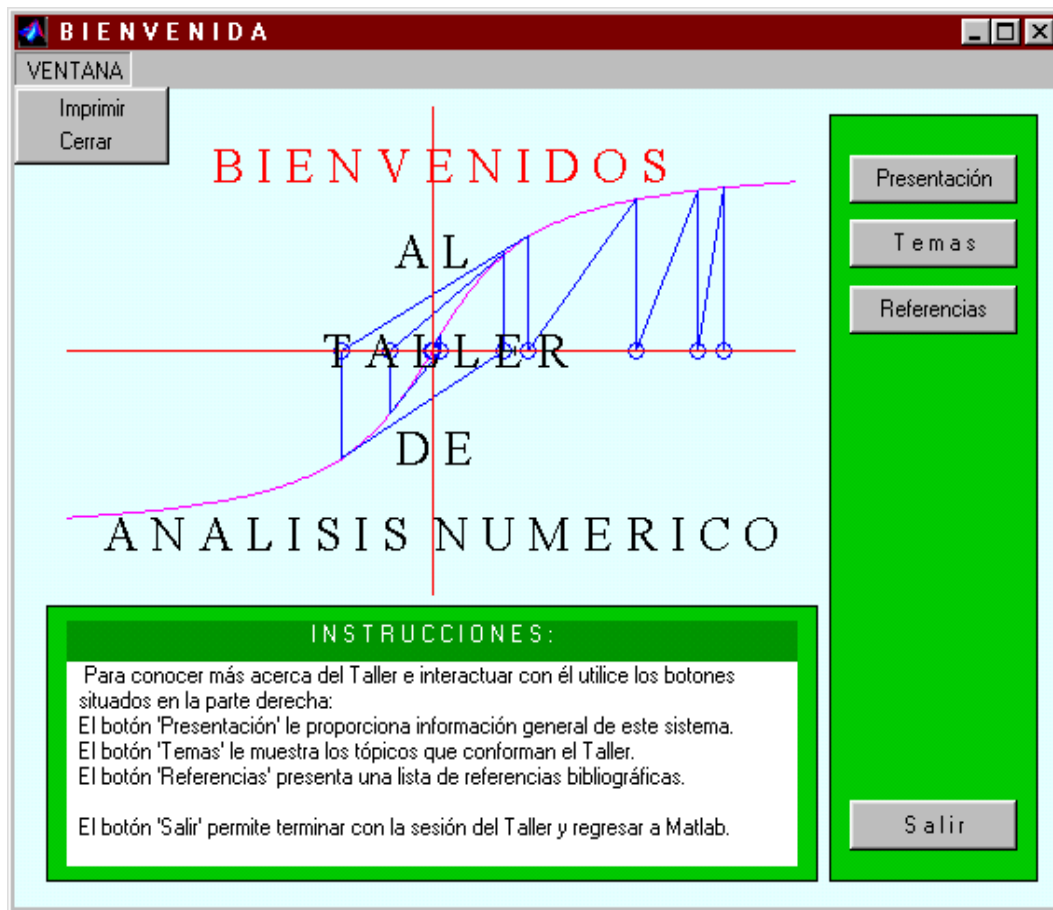


figura 1.7. Ventana de Bienvenida al Taller.

Al activar el botón '**Presentación**' se abre una nueva ventana que contiene información general acerca del Taller como: Conceptualización y propósitos, descripción de su diseño y breve crónica de su evolución. El botón '**Referencias**' abre una ventana que muestra una lista de referencias bibliográficas que proporcionaron bases teóricas y ejemplos numéricos para el Taller.

Las ventanas que se abren con cada uno de los botones anteriores disponen a su vez de un botón para cerrar dichas ventanas, regresando así a la de bienvenida.

El botón '**Temas**' abre una ventana que muestra los temas que contempla el Taller. El botón '**Salir**' hace desaparecer la ventana de bienvenida y se utiliza en cada ventana del Taller para terminar con la sesión y regresar a la



ventana principal de Matlab.

En la parte superior de la ventana, debajo del rótulo 'BIENVENIDA' se dispone de una barra con el menú 'VENTANA' el cual ofrece las opciones

- 'Imprimir'. Si la computadora en que se trabaja está conectada a una impresora, mediante esta opción puede obtenerse una copia en papel de los objetos que se encuentran dentro de la ventana actual del Taller.
- 'Cerrar'. Tiene la misma función que el botón 'Salir'.

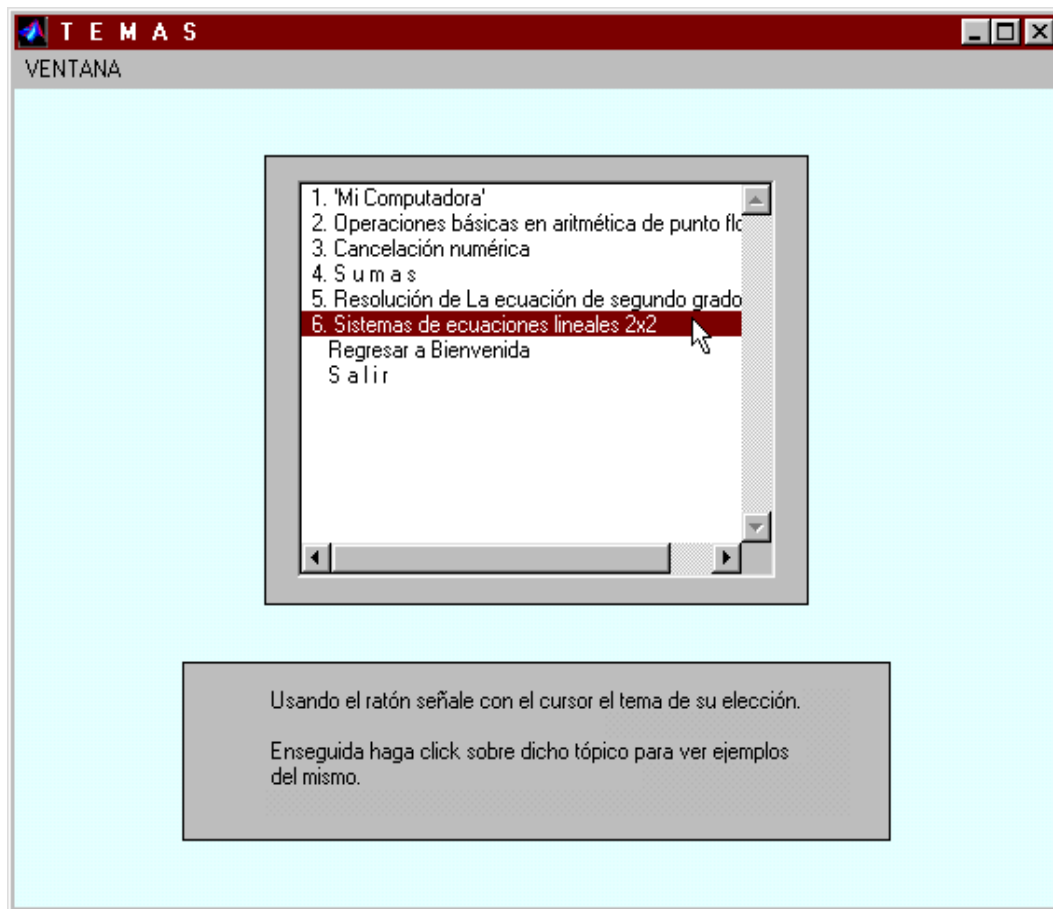


figura 1.8. Ventana con los temas del Taller.

Si se oprime el botón 'Temas' aparece la ventana mostrada en la figura 1.8, desapareciendo la de bienvenida.

En el recuadro de la parte inferior se le pide al usuario que utilizando el ratón señale con el cursor el tema de su elección haciendo enseguida click. Aprecie en la lista que además de los temas se ofrecen las opciones 'Regresar a Bienvenida' y 'Salir'.

Supongamos que se elige el tema 'Sistemas de ecuaciones lineales 2x2'. La ventana que aparece entonces es

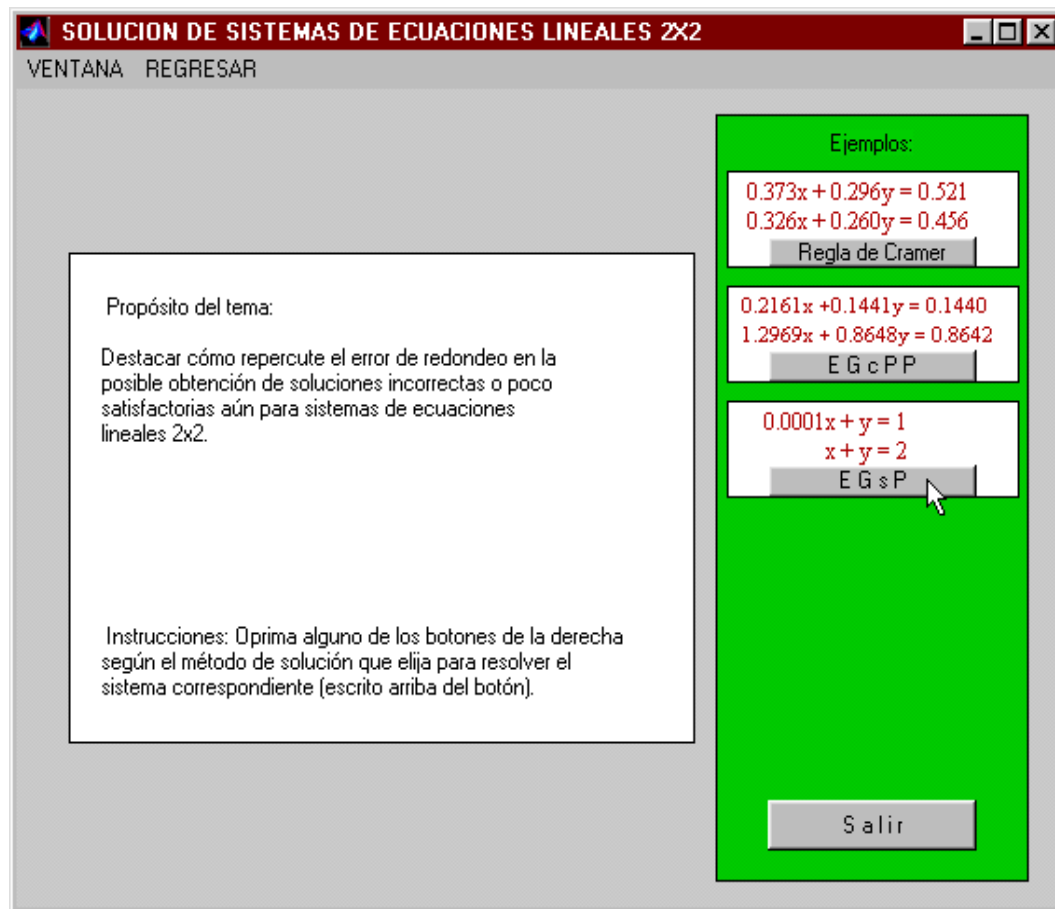


figura 1.9. Ventana con ejemplos de sistemas de ecuaciones  $2 \times 2$ .

En la parte izquierda de esta ventana se muestra el propósito de trabajar este tema y en el lado derecho se presentan 3 ejemplos de sistemas de ecuaciones  $2 \times 2$ . Debajo de cada sistema se encuentra un botón el cual al activarse ofrece resolver el ejemplo correspondiente por el método escrito en el botón. Los

métodos de solución son 'Regla de Cramer', 'EGcPP' (Eliminación Gaussiana con Pivoteo Parcial) y 'EGsP' (Eliminación Gaussiana sin Pivoteo).

Al oprimir el botón 'EGsP' se presenta la ventana

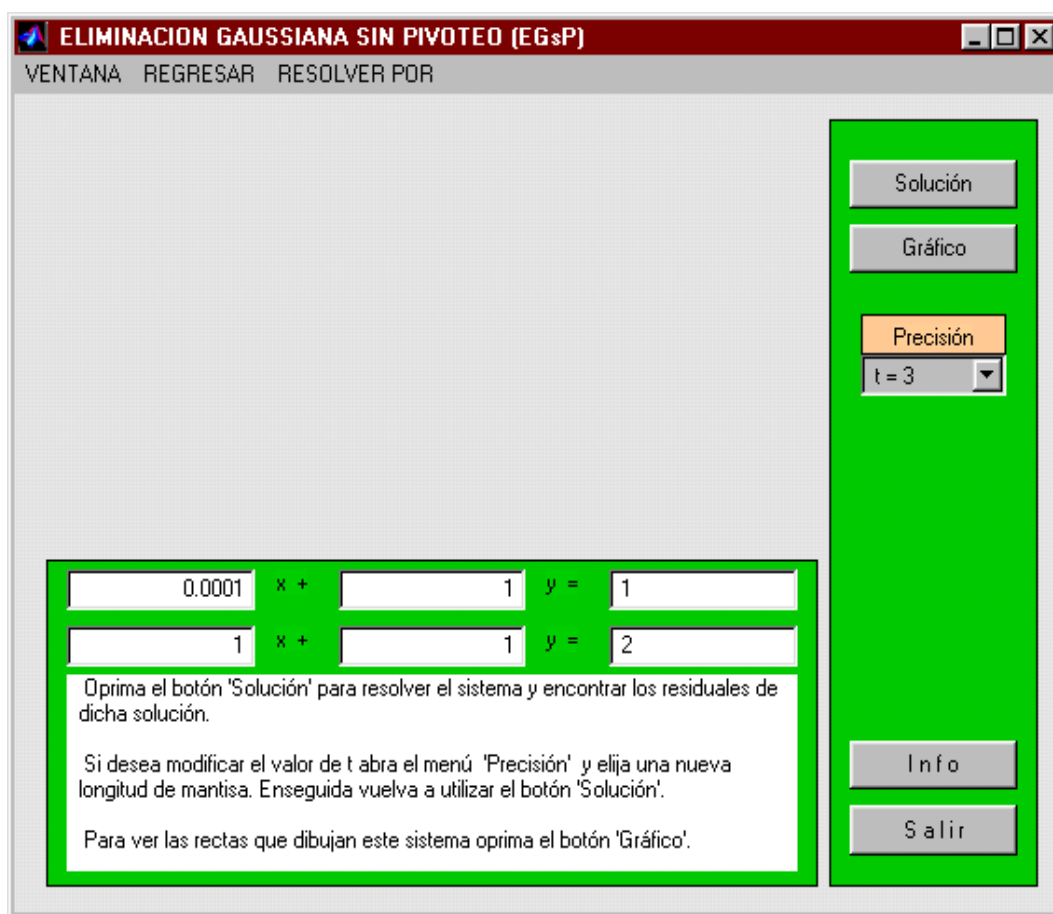


figura 1.10. Ventana que resuelve un sistema  $2 \times 2$  mediante EGsP.

la cual consta en el lado derecho de 3 botones y un menú, además del botón 'Salir'.

El botón 'Solución' proporciona la solución numérica del sistema utilizando el método de EGsP.

El botón 'Gráfico' muestra la representación gráfica del sistema.

El menú 'Precisión' despliega los distintos valores  $t$  de longitud de man-

tisa.

El botón 'Info' abre una ventana que ofrece una breve explicación sobre el comportamiento numérico del ejemplo en cuestión.

Note en la ventana anterior que los coeficientes del sistema están escritos dentro de espacios editables. Esto significa que colocando el cursor en dichos espacios es posible borrar los valores actuales e introducir nuevos coeficientes lo que hace posible la realización de ejercicios por parte del usuario.

En la parte superior de la ventana se tiene la barra con los menús:

- 'VENTANA'. El cual se ha comentado anteriormente.
- 'REGRESAR'. Este menú permite volver a módulos anteriores del Taller. A partir de la ventana de la figura 1.10 es posible regresar:

Al menú de sistemas de ecuaciones  $2 \times 2$ , ó  
al menú de temas, o bien  
a la ventana de bienvenida.

- 'RESOLVER POR'. Mediante este menú es posible resolver el sistema en cuestión aplicando otro método distinto. En este caso las opciones disponibles son

'Regla de Cramer', ó  
'EGcPP'.

Al oprimir el botón 'Solución' aparece en la misma ventana sobre el sistema de ecuaciones

$$\begin{array}{rcl} 0.0001x & + & 1y = 1 \\ 1x & + & 1y = 2 \end{array}$$

un recuadro con fondo color blanco en el que se exhibe un atractivo texto en color azul la solución numérica  $(X_G, Y_G)$  así como los residuales  $r1$  y  $r2$  en notación de punto flotante decimal (vea fig. 1.11).

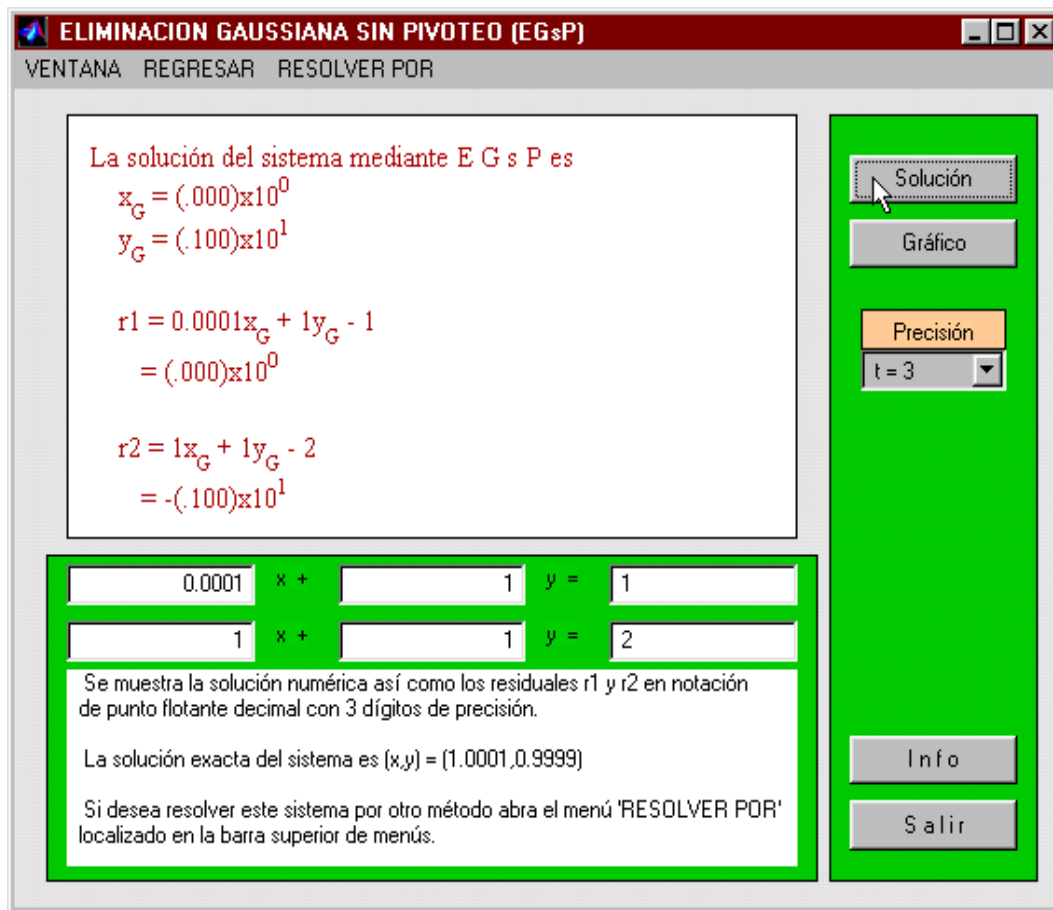


figura 1.11. Aplicación de la EGsP para resolver el sistema en cuestión.

En este ejemplo se simula una computadora que trabaja con tres dígitos de precisión. Luego de efectuar la EGsP se obtiene como resultado la solución  $(x_G, y_G) = (0, 1)$ . El usuario puede verificar el nivel de exactitud de la respuesta numérica ya que en el texto situado debajo del sistema de ecuaciones se presenta la solución exacta del mismo.

Si se oprime el botón 'Gráfico' aparece en la misma ventana sobre el sistema de ecuaciones la representación gráfica del mismo (vea fig. 1.12). Cerca de la esquina inferior izquierda del gráfico se escriben las ecuaciones con un color diferente cada una. Dichos colores se utilizan para identificar la recta correspondiente a cada ecuación.

Puede verse en este ejemplo que la intersección de las rectas se localiza cerca

del punto (1,1) y la solución exacta del sistema es mencionada nuevamente en el texto situado en la parte inferior.

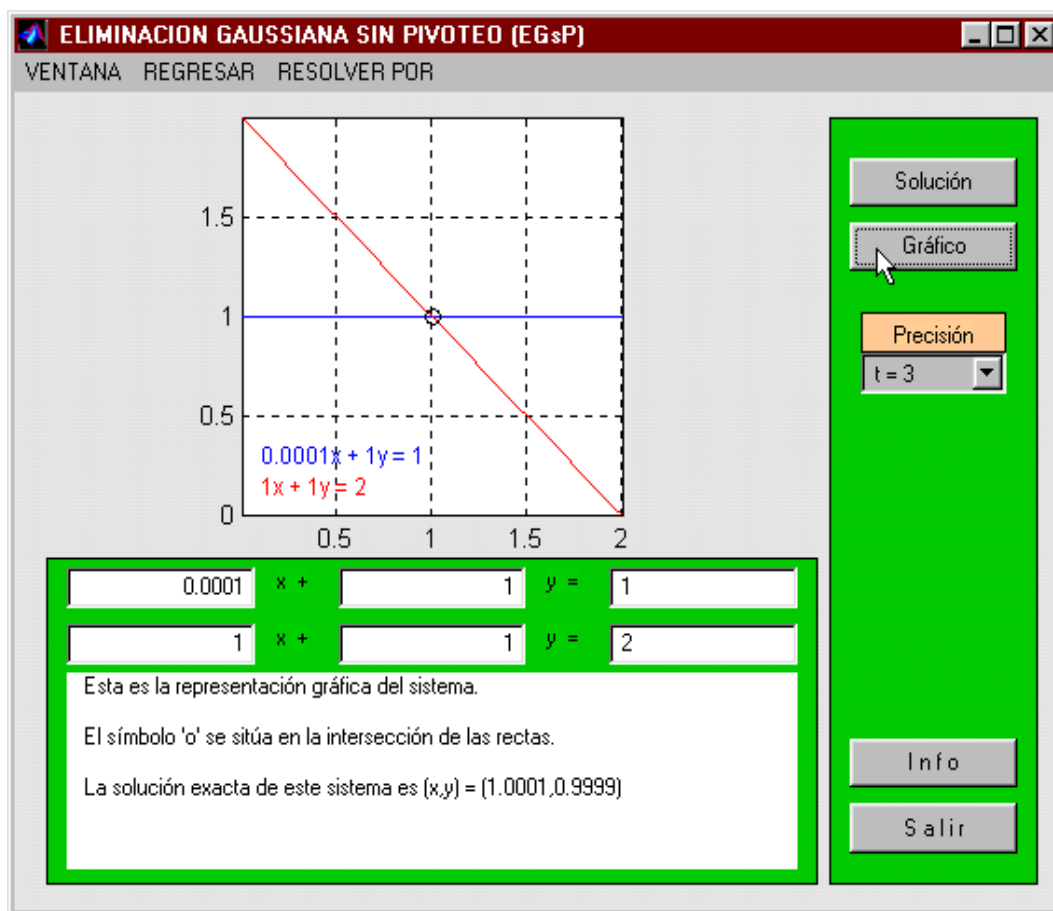


figura 1.12. Ventana que muestra el gráfico del sistema en cuestión.

El usuario puede seguir trabajando con este ejemplo abriendo el menú 'Precisión' para modificar la longitud de mantisa y oprimir nuevamente el botón 'Solución' o bien utilizar el menú 'RESOLVER POR' si desea aplicar la Regla de Cramer o el método de EGcPP. También es posible realizar ejercicios escribiendo otros sistemas de ecuaciones si se coloca el cursor en los espacios editables para borrar los valores actuales e introducir nuevos coeficientes o si el usuario lo desea puede volver a ventanas anteriores del Taller utilizando el menú 'REGRESAR' o bien terminar con la sesión y regresar a Matlab directamente con el botón 'Salir'.

## Capítulo 2

# MANUAL DE REFERENCIA

El sistema computacional **Taller.m** versión 1.0 fue desarrollado en Matlab v.5.3 y se utilizaron funciones propias de este lenguaje para la creación de ventanas y demás objetos de interfaz. También en Matlab se escribieron cuatro funciones para simular la aritmética de punto flotante, y treinta funciones para diseñar todas las ventanas del Taller.

A continuación se describe cada una de estas funciones mencionando su nombre, su propósito, su sintaxis o modo de invocarla, sus parámetros de entrada y de salida y en algunos casos, un ejemplo.

Para facilitar la localización de las funciones en este capítulo, se han formado tres grupos:

- Funciones de Matlab para crear objetos de interfaz.
- Funciones que simulan la aritmética de punto flotante.
- Funciones para crear las ventanas del Taller.

Además dentro de cada grupo las funciones se mencionan en orden alfabético.

## 2.1 Funciones de Matlab para crear objetos de interfaz

**axes.m**

**Propósito:**

Crea una porción del plano cartesiano con los ejes coordenados  $x$  y  $y$ .

**Sintaxis:**

```
axes('c1','v1','c2','v2',...,'cn','vn');
```

donde los datos de entrada se introducen por pares: Cada  $c_i$  hace referencia a una característica o propiedad de los ejes y cada  $v_i$  especifica el valor o asignación que puede recibir dicha característica o propiedad.

**Ejemplo:**

```
axes('Units','normalized','Position',[.05 .05 .45 .45]);
```

crea el cuadro  $[0,1]$  cerca de la esquina inferior izquierda de una ventana.



## **figure.m**

### **Propósito:**

Crea una ventana.

### **Sintaxis:**

```
figure('c1','v1','c2','v2',...,'cn','vn');
```

donde los datos de entrada se introducen por pares: Cada  $c_i$  hace referencia a una característica o propiedad de la ventana y cada  $v_i$  especifica el valor o asignación que puede recibir dicha característica o propiedad.

### **Ejemplo:**

```
figure('Color','b','Menubar','none');
```

crea una ventana con fondo color azul y sin la barra de menús.

## uicontrol.m

### Propósito:

Crea diferentes objetos de control como botones, cuadros de texto, espacios editables entre otros.

### Sintaxis:

```
uicontrol('c1','v1','c2','v2',...,'cn','vn');
```

donde los datos de entrada se introducen por pares: Cada  $c_i$  hace referencia a una característica o propiedad del objeto de control y cada  $v_i$  especifica el valor o asignación que puede recibir dicha característica o propiedad. Comúnmente  $c_1$  es la propiedad 'Style' la cual puede recibir alguna de las siguientes asignaciones dependiendo del tipo de objeto de control a crear:

- 'Pushbutton', para botones de activación
- 'text', para cuadros de texto
- 'edit', para espacios editables
- 'frame', para rectángulos
- 'popupmenu', para menús.

### Ejemplo:

```
uicontrol('Style','pushbutton','String','Salir',...  
          'Callback','close');
```

crea cerca de la esquina inferior izquierda de una ventana el botón con el rótulo 'Salir' el cual al oprimirse permite cerrar dicha ventana.

## **uimenu.m**

### **Propósito:**

Crea menús y submenús en la barra situada en la parte superior de una ventana.

### **Sintaxis:**

```
uimenu('c1','v1','c2','v2',...,'cn','vn');
```

donde los datos de entrada se introducen por pares: Cada  $c_i$  hace referencia a una característica o propiedad de la ventana y cada  $v_i$  especifica el valor o asignación que puede recibir dicha característica o propiedad.

### **Ejemplo:**

```
uimenu('Label','Malla','Callback','grid on');
```

produce el menú 'Malla' colocado enseguida del menú 'Help'. Al activarse aquel aparece sobre la ventana la porción  $[0,1]$  del plano cartesiano con un cuadriculado.

## 2.2 Funciones que simulan la aritmética de punto flotante

### `guardar.m`

#### **Propósito:**

Simula el guardar un número dado en una palabra cuya longitud especifica el usuario.

#### **Sintaxis:**

$$v = \text{guardar}(x, t)$$

donde los parámetros de entrada de esta función son un número real  $x$  y un valor entero  $t$  entre 1 y 16 inclusive con el que se especifica la longitud de mantisa o precisión.

El resultado es un vector renglón  $v$  el cual representa una palabra (la unidad de memoria para guardar un número) de longitud  $t+1$ . Los lugares de 1 a  $t$  conforman la mantisa y en ellos se colocan las primeras  $t$  cifras del número  $x$  mientras que en el lugar  $t+1$  se guarda el valor del exponente conociendo que el punto decimal se ha recorrido al principio de la mantisa

#### **Ejemplo:**

Guardar el número  $\pi$  en una palabra cuya mantisa dispone de 8 cifras:

```
>> v = guardar(pi,8)
v =
    3    1    4    1    5    9    2    7    1
```

esto significa que el vector  $v$  está representando la palabra de longitud 9:

+3	1	4	1	5	9	2	7	+1
----	---	---	---	---	---	---	---	----

cuyos primeros 8 lugares constituyen la mantisa y el noveno lugar es para el exponente.

## notacion.m

### Propósito:

Presenta un número anteriormente guardado, en notación de punto flotante decimal.

### Sintaxis:

```
c = notacion(v)
```

donde el parámetro de entrada `v` es el vector proveniente de la función `guardar` o de la función `op_float` (la cual se explica en la siguiente página).

El resultado es el conjunto de caracteres `c` que muestra al número guardado en notación de punto flotante decimal.

### Ejemplo:

Expresar en notación de punto flotante decimal con 5 dígitos el número  $-\frac{2000}{7}$ :

```
>> v = guardar(-2000/7,5);
>> c = notacion(v)
c =
    -(.28571)x10^{3} .
```

Si se utiliza la función `notacion` en la ventana principal de Matlab aparece el símbolo `^` que indica potencia así como las llaves `{ }` que encierran el valor del exponente.

Los caracteres anteriores son necesarios ya que en la implementación del Taller se utiliza para la exhibición de resultados la función `text` de Matlab la cual emplea la misma sintaxis para el reconocimiento de algunos símbolos de la misma forma en que lo hace el programa para elaborar documentos científicos L<sup>A</sup>T<sub>E</sub>X.

De modo que dentro del Taller el ejemplo anterior aparece

```
c = - (.28571)x103.
```

## **op\_flot.m**

### **Propósito:**

Efectúa la operación aritmética indicada por alguno de los símbolos: +, -, \* ó /, entre dos números de punto flotante.

### **Sintaxis:**

```
z = op_flot(u,v,'op')
```

donde los parámetros de entrada **u** y **v** son dos vectores renglón de igual longitud resultantes de aplicar la función **guardar** y **op** debe ser alguno de los 4 caracteres: +, -, \*, /, encerrado entre comillas simples para denotar la operación aritmética correspondiente que se efectuará con los números 'de punto flotante' guardados en **u** y **v**.

El resultado es también un vector renglón **z** de la misma longitud que **u** y **v**.

### **Ejemplo:**

Efectuar la suma  $e + \sqrt{2}$  en aritmética de punto flotante con 6 dígitos de precisión:

```
>> u = guardar(exp(1),6);
>> v = guardar(sqrt(2),6);
>> z = op_flot(u,v,'+')
z =
    4 1 3 2 4 9 1
```

### **Observación:**

En las operaciones de suma y multiplicación es indistinta la colocación de los vectores **u** y **v** pero en el caso de la resta el primer parámetro debe ser el minuendo, y el segundo parámetro el sustraendo. Igualmente para la división el primero y segundo datos deben ser respectivamente el numerador y el denominador.

## **valornum.m**

### **Propósito:**

Emite el valor numérico de un número de punto flotante.

### **Sintaxis:**

```
x = valornum(v)
```

donde el parámetro de entrada *v* es un vector renglón resultante de aplicar la función **guardar** u **op\_flot**.

Como resultado se obtiene el número *x* que consta solamente de las cifras que tenía la mantisa en *v*.

### **Ejemplo:**

Obtener el valor numérico de  $\sqrt{5}$  considerando 7 cifras:

```
>> v = guardar(sqrt(5),7)
v =
    2 2 3 6 0 6 8 1
>> x = valornum(v)
x =
    2.236068
```

## 2.3 Funciones para la creación de las ventanas del Taller

### acetato.m

#### Propósito:

Se utiliza para redactar hasta 9 líneas de texto en color azul. Mediante esta función se exhiben los resultados numéricos de los ejemplos.

#### Sintaxis:

`acetato(r1,r2,r3,r4,r5,r6,r7,r8,r9)`

donde cada parámetro  $r_i$ ,  $i = 1, 2, \dots, 9$  es un renglón o cadena de caracteres, escrito entre comillas simples.

#### Ejemplo:

La instrucción

```
>> acetato('La serie','',...  
           '\Sigma_{n=1}^{\infty} 1/n','',...  
           'es divergente')
```

crea un cuadro con fondo color blanco sobre el que aparece el texto:

La serie

$\sum_{n=1}^{\infty} 1/n$

es divergente



## **campo.m**

### **Propósito:**

Crea la ventana que presenta el tema de las propiedades de los números reales:

- Asociatividad de la suma de tres números.
- Asociatividad de la suma de 1200 1's.
- Propiedad del elemento neutro aditivo.
- Asociatividad del producto de tres números.
- Cerradura del producto.

### **Modo de invocarla:**

o simplemente

```
>> campo('iniciar')
```

```
>> campo
```

## **cancenun.m**

### **Propósito:**

Crea la ventana que reúne los ejemplos con los que se ilustra el fenómeno de cancelación numérica:

- Sustracción de dos números muy cercanos entre sí.
- Area de triángulos en forma de aguja.
- Exponencial de  $x$ ,  $x < 0$ .
- Media y varianza.

### **Modo de invocarla:**

```
>> cancenum('iniciar')
```

o simplemente

```
>> cancenum
```

## **cercanos.m**

### **Propósito:**

Crea la ventana que presenta el ejemplo de la sustracción  $a - b$  con los números:

$$\begin{array}{rcl} a & = & 63.221473 \\ b & = & 63.221486 \end{array}$$

### **Modo de invocarla:**

```
>> cercanos('iniciar')
```

o simplemente

```
>> cercanos
```

En la ventana el usuario puede proporcionar sus propios números  $a$  y  $b$ .

## **cramer.m**

### **Propósito:**

Crea la ventana que presenta el sistema de ecuaciones

$$0.373x + 0.296y = 0.521 \quad (1)$$

$$0.326x + 0.260y = 0.456 \quad (2)$$

el cual es resuelto por la Regla de Cramer.

### **Modo de invocarla:**

```
>> cramer('iniciar')
```

o simplemente

```
>> cramer
```

En la ventana el usuario puede proporcionar sus propios coeficientes.

## **crradura.m**

### **Propósito:**

Crea la ventana que ilustra el fenómeno de desbordamiento por exceso del exponente (*overflow*) mediante la multiplicación de los números

$$\begin{aligned}a &= 2 \times 10^{200} \\ b &= 3 \times 10^{150}\end{aligned}$$

### **Modo de invocarla:**

```
>> crradura('iniciar')
```

o simplemente

```
>> crradura
```

En la ventana el usuario puede proporcionar sus propios números  $a$  y  $b$ .

## **e\_taylor.m**

### **Propósito:**

Crea la ventana con la que se efectúa el cálculo numérico de

$$e^{-5.5}$$

ilustrando así el fenómeno de cancelación numérica.

### **Modo de invocarla:**

```
>> e_taylor('iniciar')
```

o simplemente

```
>> e_taylor
```

En la ventana el usuario puede proporcionar sus propios valores de  $x$ .

## **ec2gr1.m**

### **Propósito:**

Crea la ventana en la que se resuelve la ecuación:

$$x^2 - 10^5 + 1 = 0$$

mediante la fórmula general y el método de Vieta.

### **Modo de invocarla:**

```
>> ec2gr1('iniciar')
```

o simplemente

```
>> ec2gr1
```

En la ventana el usuario puede proporcionar sus propios coeficientes  $a$ ,  $b$  y  $c$ .

## **ec2gr2.m**

### **Propósito:**

Crea la ventana en la que se resuelve la ecuación:

$$x^2 - 4 + 3.9999999 = 0$$

mediante la fórmula general y el método de Vieta.

### **Modo de invocarla:**

```
>> ec2gr2('iniciar')
```

o simplemente

```
>> ec2gr2
```

En la ventana el usuario puede proporcionar sus propios coeficientes  $a$ ,  $b$  y  $c$ .



## **ec2gr3.m**

### **Propósito:**

Crea la ventana en la que se resuelve la ecuación:

$$x^2 + 111.11x + 1.2121 = 0$$

mediante la fórmula general y el método de Vieta.

### **Modo de invocarla:**

```
>> ec2gr3('iniciar')
```

o simplemente

```
>> ec2gr3
```

En la ventana el usuario puede proporcionar sus propios coeficientes  $a$ ,  $b$  y  $c$ .

## **ec2grado.m**

### **Propósito:**

Crea la ventana que reúne los ejemplos de las ecuaciones de segundo grado presentadas en `ec2gr1`, `ec2gr2` y `ec2gr3`.

### **Modo de invocarla:**

```
>> ec2grado('iniciar')
```

o simplemente

```
>> ec2grado
```

## **egcpp.m**

### **Propósito:**

Crea la ventana que presenta el sistema de ecuaciones

$$0.2161x + 0.1441y = 0.1440 \quad (1)$$

$$1.2969x + 0.8648y = 0.8642 \quad (2)$$

el cual es resuelto mediante Eliminación Gaussiana con Pivoteo Parcial.

### **Modo de invocarla:**

```
>> egcpp('iniciar')
```

o simplemente

```
>> egcpp
```

En la ventana el usuario puede proporcionar sus propios coeficientes.

## **egsp.m**

### **Propósito:**

Crea la ventana que presenta el sistema de ecuaciones

$$0.0001x + 1y = 1 \quad (1)$$

$$1x + 1y = 2 \quad (2)$$

el cual es resuelto mediante Eliminación Gaussiana sin Pivoteo.

### **Modo de invocarla:**

```
>> egsp('iniciar')
```

o simplemente

```
>> egsp
```

En la ventana el usuario puede proporcionar sus propios coeficientes.

## **explicac.m**

### **Propósito:**

Se utiliza para redactar la presentación del Taller así como las referencias bibliográficas. Esta función permite escribir hasta tres hojas de texto.

### **Sintaxis:**

```
explicac(titulo,texto1,texto2,texto3)
```

donde el parámetro `titulo` es una cadena de caracteres que consta de una palabra o de un renglón y los demás parámetros son textos que pueden contener entre 20-25 líneas de redacción.

## **infotext.m**

### **Propósito:**

Se utiliza para redactar hasta 19 líneas de texto. Con esta función se redactan las explicaciones que aparecen al oprimirse el botón '**Info**' en las ventanas de los ejemplos.

### **Sintaxis:**

`infotext(r1,r2,...,r19)`

donde cada parámetro  $r_i$ ,  $i = 1, 2, \dots, 19$  es un renglón o cadena de caracteres, escrito entre comillas simples. Esta función trabaja del mismo modo que `acetato`.

## **medyvar.m**

### **Propósito:**

Crea la ventana que calcula la media y varianza de los datos

$$\begin{aligned}x_1 &= 10,000 \\x_2 &= 10,001 \\x_3 &= 10,002\end{aligned}$$

comparando los métodos de un paso y de dos pasos.

### **Modo de invocarla:**

```
>> medyvar('iniciar')
```

o simplemente

```
>> medyvar
```

En la ventana el usuario puede proporcionar sus propios datos.

## **mimqn.m**

### **Propósito:**

Crea la ventana que presenta los parámetros del sistema de números de punto flotante de la computadora en que se trabaja: La base, precisión, unidad de redondeo y exponentes mínimo y máximo.

### **Modo de invocarla:**

```
>> mimqn('iniciar')
```

o simplemente

```
>> mimqn
```



## **neutroad.m**

### **Propósito:**

Crea la ventana que muestra la forma en que un número distinto de cero puede comportarse como el elemento neutro aditivo en aritmética de punto flotante. Se trabaja con los números

$$\begin{aligned}a &= 10^5 \\ b &= 0.125\end{aligned}$$

### **Modo de invocarla:**

```
>> neutroad('iniciar')
```

o simplemente

```
>> neutroad
```

En la ventana el usuario puede proporcionar sus propios números  $a$  y  $b$ .

## **prod3num.m**

### **Propósito:**

Crea la ventana en la que se realiza el producto de los números

$$\begin{aligned}a &= 1050 \\b &= -25.55 \\c &= 0.3333\end{aligned}$$

asociándolos de tres formas diferentes.

### **Modo de invocarla:**

```
>> prod3num('iniciar')
```

o simplemente

```
>> prod3num
```

En la ventana el usuario puede proporcionar sus propios números  $a$ ,  $b$  y  $c$ .

## **s1\_2alan.m**

### **Propósito:**

Crea la ventana en la que se calcula el valor numérico de la serie

$$\sum_{n=1}^{\infty} \frac{1}{2^n}.$$

### **Modo de invocarla:**

```
>> s1_2alan('iniciar')
```

o simplemente

```
>> s1_2alan
```

## **s1\_n\_al2.m**

### **Propósito:**

Crea la ventana en la que se calcula el valor numérico de la serie

$$\sum_{n=1}^{\infty} \frac{1}{n^2}.$$

### **Modo de invocarla:**

```
>> s1_n_al2('iniciar')
```

o simplemente

```
>> s1_n_al2
```

## **serie1\_n.m**

### **Propósito:**

Crea la ventana en la que se calcula el valor numérico de la serie

$$\sum_{n=1}^{\infty} \frac{1}{n}.$$

### **Modo de invocarla:**

```
>> serie1_n('iniciar')
```

o simplemente

```
>> serie1_n
```

## **sistemas.m**

### **Propósito:**

Crea la ventana que reúne los sistemas de ecuaciones  $2 \times 2$  contemplados en las funciones `cramer`, `egcpp` y `egsp`.

### **Modo de invocarla:**

```
>> sistemas('iniciar')
```

o simplemente

```
>> sistemas
```

## **sum3num.m**

### **Propósito:**

Crea la ventana en la que se realiza la suma de los números

$$\begin{aligned}a &= 0.1234567 \\b &= 4711.325 \\c &= -4711.325\end{aligned}$$

asociándolos de tres formas diferentes.

### **Modo de invocarla:**

```
>> sum3num('iniciar')
```

o simplemente

```
>> sum3num
```

En la ventana el usuario puede proporcionar sus propios números  $a$ ,  $b$  y  $c$ .

## **suma1s.m**

### **Propósito:**

Crea la ventana en la que se efectúa la suma

$$\sum_1^{1200} 1$$

asociando los 1's de tres formas diferentes.

### **Modo de invocarla:**

```
>> suma1s('iniciar')
```

o simplemente

```
>> suma1s
```



## **sumas.m**

### **Propósito:**

Crea la ventana que reúne las series que se contemplan en las funciones **s1\_2a1a**n, **s1\_n\_a12** y **serie1\_n**.

### **Modo de invocarla:**

```
>> sumas('iniciar')
```

o simplemente

```
>> sumas
```

## **taller.m**

### **Propósito:**

Crea la ventana de bienvenida al Taller. Jerárquicamente es la función principal.

### **Modo de invocarla:**

	<code>&gt;&gt; taller('inicia')</code>
ó	
	<code>&gt;&gt; taller('regresa')</code>
o simplemente	
	<code>&gt;&gt; taller</code>

## **taller.mat**

### **Propósito:**

Es un archivo que contiene la información acerca del gráfico de la función arcotangente el cual aparece detrás del rótulo

BIENVENIDOS  
AL  
TALLER  
DE  
ANALISIS NUMERICO

en la ventana de bienvenida al Taller.

La función `taller.m` manda llamar con el comando `load` este archivo.

## **temas.m**

### **Propósito:**

Crea la ventana que presenta los temas que contempla el Taller.

### **Modo de invocarla:**

```
>> temas('iniciar')
```

o simplemente

```
>> temas
```

## **triangul.m**

### **Propósito:**

Crea la ventana en la que se calcula el área de un triángulo en forma de aguja. Se compara la fórmula de Herón con el método de Kahan.

### **Modo de invocarla:**

```
>> triangul('iniciar')
```

o simplemente

```
>> triangul
```

En la ventana el usuario puede proporcionar sus propias medidas de los lados del triángulo.