Apuntes de la materia Programación Numérica

Edgar Arce

23 de julio de 2004

Índice General

1 Tutorial de MatLab

1.1 ¿Qué es MatLab?

MatLab significa 'MATrix LABoratory' (LABORATORIO DE MATRICES). MATLAB es un medio computacional técnico, con un gran desempeño para el cálculo numérico computacional y de visualización. MATLAB integra análisis numérico, matrices, procesamiento de señales y gráficas, todo esto en un ambiente donde los problemas y soluciones son expresados tal como se escriben matemáticamente.

Escrito inicialmente como auxiliar en la programación de cálculo con matrices. MATLAB fue escrito originalmente en fortran, actualmente está escrito en lenguaje C. MATLAB es un lenguaje de programación amigable al usuario con características más avanzadas y mucho más fáciles de usar que los lenguajes de programación como basic, pascal o C. MATLAB cuenta con paquetes de funciones especializadas llamadas toolboxes algunas de las cuales se mencionan a continuación:

- Control system Toolbox, Robust Control Toolbox.
- Frequency Domain System Identification Toolbox.
- Fuzzy Logic Toolbox.
- Higher Order Spectral Analisys Toolbox.
- Image Processing Toolbox.
- Model Predective Control Toolbox.
- Mu Analisis and Synthesis Toolbox.
- NAG Foundation Toolbox.
- Neural Network Toolbox.
- Nonlinear Control Design Toolbox.
- Optimization Toolbox.
- Quantitative Feedback Theory Toolbox.
- Signal Processing Toolbox.
- SIMULINK, SIMULINK Real Time Workshop.
- Spline Toolbox.
- Statistics Toolbox.
- Symbolic Math Toolbox.
- System Identification Toolbox.

1.2 Inicio de MATLAB

MATLAB se inicia directamente desde Windows. Al invocarse MATLAB aparecerá la pantalla de comandos, algunas sugerencias y el símbolo >> , el cual indica la entrada de instrucciones para ser evaluadas.

>>

>> Comando o instrucción a evaluar < enter >

Para hacer la suma de dos números, escribimos :

>> 5 + 5 < enter > Presionamos la tecla entrar.

ans =

10

El resultado es desplegado y se guarda en la variable ans (answer).

El comando *help* proporciona una lista de todos los tópicos que MATLAB puede proporcionar ayuda. help 'comando' proporciona ayuda sobre el comando especificado.

help sqrt

proporciona ayuda sobre la instrucción sqrt., ejemplo:

» help sqrt

SQRT Square root.

SQRT(X) is the square root of the elements of X. Complex

results are produced if X is not positive.

See also SQRTM

1.3 ¿Cómo funciona MATLAB?

MATLAB puede almacenar información en variables tales como :

a = 100 " para evaluar la celda "

Cada vez que capturamos información en MATLAB y presionamos <ENTER> ésta es desplegada inmediatamente, pero si ponemos un punto y coma al final de la instrucción MATLAB omite el desplegado de información.

Por ejemplo :

b = 50 ;

Si se quiere saber el valor de alguna variable capturada sólo se tiene que poner el nombre de la variable y <ENTER> y MATLAB lo despliega. Estas variables residen en el espacio de trabajo de MATLAB.

b

Las variables son sensibles a las mayúsculas, por lo que las siguientes variables son diferentes :

Variable = 1

variable = 1

Las variables pueden contener hasta 19 caracteres. Éstas deben empezar con una letra, seguida por cualquier número de letras, dígitos o guiones de subrayado; los caracteres de puntuación no son permitidos en las variables.

Cuando se trabaja con muchas variables estas son difícil de recordar. El comando who muestra un desplegado de todas aquellas variables que se han estado utilizando.

who

whos Muestra las variables con información adicional.

1.3.1 Caracteres especiales

- [ ] Son usados para formar vectores y matrices [1 2 3 ; 4 5 6].
- ( ) Usados para expresiones matemáticas. sqrt(2).
- = Usado para hacer asignaciones: x = 5.
- ' Transpuesta de una matriz: A'.
- (espacio) Usada para separar texto: 'texto'.
- . Punto decimal: 3.1415.
- ... Al final de una línea indican que continua 2,3,4,5,6 ....
  - en el siguiente renglón. 7,8,9,10].
- , Para separar elementos [1,2,3,4].
- ; Para separar filas en las matrices. [1 2; 3 4].
- % Para hacer comentarios % este programa,etc.
- ! Para ejecutar un comando del Ms-dos: !dir.

1.3.2 Operaciones básicas

- SUMA C = a + b
- RESTA d = a - b
- MULTIPLICACION e = a * b
- DIVISION F = a / b, F = a \ b
- POTENCIA a ^2; como este último cálculo no tenía variable asignada, la respuesta se guarda en la variable ans (answer).

1.3.3 Borrado de variables

Para borrar el valor de una variable simplemente escribimos

clear a

borra la variable " a ".
Para borrar más de una variable, escribimos

clear a b c

Para borrar todas las variables se escribe

clear

1.3.4 Funciones trigonométricas

Si escribimos la siguiente instrucción en MatLab

$$\sin(0.5)$$

se estará evaluando el seno de 0.5 radianes.

Así mismo, existen, entre otras, las siguientes funciones trigonométricas:

- cos (x)
- tan (x)
- asin (x)
- acos (x)
- atan (x)
- sinh(x)
- cosh (x)
- tanh (x)

- asinh (x)
- acosh(x)
- atanh(x)
- atan2(x,y) inversa de la tangente en los cuatro cuadrantes.

1.3.5 Logaritmos

- log (x) Logaritmo natural.
- log10 (x) Logaritmo decimal.

1.3.6 Funciones matemáticas especiales

- abs (-3) Valor absoluto o magnitud de un número complejo.
- ceil (123.123123) Redondea hacia más infinito.
- floor (x) Redondea hacia menos infinito.
- fix (x) Redondea hacia cero.
- round (x) Redondea hacia el entero más próximo.
- imag (30 - 5j) Parte imaginaria de un número complejo.
- reak (x) Parte real de un número complejo.
- angle (x) Angulo de un número complejo.
- conj (x) Complejo conjugado.
- sign (-5) Función signo : Devuelve el signo del argumento (1 si es positivo, -1 si es negativo).
- exp (1) Exponencial : e (x).
- rem (x,y) Resto después de la división (x / y).
- sqrt (2) Raíz cuadrada.

1.3.7 Operaciones Lógicas

En MATLAB se pueden hacer operaciones lógicas, por ejemplo.

$$1 < 2$$

como 1 es menor que 2, la respuesta es cierta por lo que obtenemos un 1.
Para

$$1 < 1$$

obtenemos un 0, porque 1 no es menor que 1.

Como se puede observar las únicas respuestas posibles con las operaciones lógicas son :

$$\text{Cierto} = 1 \text{ y Falso} = 0.$$

1.3.8 Operadores relaciónales

- > Mayor que.
- < Menor que.
- >= Mayor o igual a.
- <= Menor o igual a.
- == Igual a.
- ~= No igual a.

Existen tres operadores lógicos :

- AND ó &
- OR ó |
- NOT ó ~

Para que la operación AND sea verdadera las dos relaciones deben ser verdaderas.

Recordemos la tabla de verdad de la operacion AND:

operando 1	operando 2	resultado	
0	0	0	$falso$
0	1	0	$falso$
1	0	0	$falso$
1	1	1	$verdadero$

Ejemplo

$$(\ 1 < 2\ )\ \&\ (\ 2 < 3) \text{ Verdadero.}$$
$$(\ 1 < 2)\ \&\ (\ 2 < 1\ ) \text{ Falso.}$$

Para la operación OR :

operando 1	operando 2	resultado	
0	0	0	$falso$
0	1	0	$verdadero$
1	0	0	$verdadero$
1	1	1	$verdadero$

Ejemplo:

(1 < 2) | (2 < 1) Verdadero.

Para la operación NOT :

operando	resultado	
~0	1	$verdadero$
~1	0	$falso$

Ejemplo:

~(2 < 1) Verdadero.

1.3.9 La variable NaN (Not a Number)

Cuando en un lenguaje de programación como basic, pascal o C, se da una situación que el programa no pueda manejar, como una división como 0/0 el programa se detiene, marcando un error.

Cuando en MATLAB se presenta una situación similar el programa no se detiene, sólo da una pequeña advertencia de que se presentó una división entre cero. Y el resultado es un NaN, que es una variable interna no es un número), por ejemplo si se escribe 0/0.

Ejemplo: defina a=[1 2 0] y b=[1 2 0] ahora pida la división elemento a elemento (comando "./")

a ./ b

1.3.10 Solución de ecuaciones de segundo grado

MATLAB se puede resolver fácilmente ecuaciones del tipo $ax^2 + bx + c = 0$, haciéndolo como si fuera una sola instrucción. La formula para resolver una ecuación de segundo grado si tenemos los valores para $a = 1, b = 2, c = 3$ es :

Escribimos la formula para $x1$:

$$x1 = (-b + sqrt(b\hat{}2 - 4 * a * c))/2 * a$$

Para $x2$:

$$x2 = (-b - sqrt(b\hat{}2 - 4 * a * c))/2 * a$$

Podemos hacer la comprobación para $x1$.

$$a * x1\hat{}2 + b * x1 + c$$

1.3.11 Arreglos (Arrays) ó Vectores

Si se desea calcular el seno de " 0 a 1 " con incrementos de 0.25, se pueden capturar los valores y después mandar llamar el seno de la función.

- x = [0, 0.25, 0.5, 0.75, 1]

Se pueden omitir las comas cuando se capturan los números. Con los números capturados, se obtiene el seno de la variable x escribiendo simplemente :

- sin (x)

MATLAB opera en radianes, donde $2\pi = 360$ grados.

Ahora se requiere obtener el coseno de cero a uno con incrementos de 0.01; lo que equivale a capturar 101 elementos. Para evitar capturarlos a mano, MATLAB nos permite crear un vector de la siguiente manera :

Variable = (Valor inicial : Con incrementos de : Valor final)

Ejemplo:

$$\mathrm{R} = (0 : 0.01 : 1)$$

Ahora se puede obtener el coseno de la variable R:

$$\cos(R)$$

Hagamos el siguiente vector :

$$Y = (0 : 1 : 10)$$

Si queremos saber cual es el cuarto elemento del vector ponemos :

$$Y(4)$$

Si nos interesan los elementos 5 al 10 :

$$Y(5 : 10)$$

Otras opciones son :

- $Y(1 : 2 : 9)$ Toma los elementos del 1 al 9 con incrementos de 2.
- $Y([1, 3, 7, 10])$ Toma los elementos 1, 3, 7 y 10 del arreglo.

1.3.12 Modificaciones de los arreglos

Si el noveno elemento del arreglo debió ser el número 20 en vez de 8, corregimos de la siguiente manera :

$$Y(9) = 20$$

Otra forma de hacer arreglos, es con linspace :

Linspace (Valor inicial , Valor final , Número de elementos)

Regresando al ejemplo del coseno de 0 a 1 con incremento de 0.01 escribimos :

$$Z = linspace(0, 10, 101)$$

note el uso de comas (#, #, #). Linspace describe una relación lineal de espaciado entre sus elementos.

Otra forma de crear arreglos es :

- $x1 = 1 : 5$ Arreglo de 1 a 5, con incremento de 1.
- $x2 = 10 : 5 : 100$ Arreglo de 10 a 100, con incrementos de 5.

Si se quiere concatenar $x1$ y $x2$

$$C = [x1\ x2]$$

1.3.13 Matemáticas con arreglos

- a = 1 : 6 Define un vector de seis elementos con incrementos de 1.
- b = 1 : 2 : 12 Vector de seis elementos con incremento de 2

Arreglos con escalares. Se le puede sumar o multiplicar un número a todo el arreglo, por ejemplo

$a + 10$ Suma de un escalar con una rreglo
$a * 10$ Multiplicación de un escalar con un arreglo

Operaciones con arreglos. Para hacer la suma de los arreglos a y b, solamente escribimos :

$$a + b$$

Se pueden hacer operaciones como :

$$Z = 100 - 2 * a + b$$

La multiplicación de arreglos se hace con (. *), ya que cuando se utiliza el asterisco sin punto indica multiplicación matricial, y además provoca un error.

$$Z = a. * b$$

La división también lleva un punto antes del signo, porque sino se utiliza el punto nos referimos a la división matricial que es muy diferente.

$$Z = a./b$$

La siguiente operación obtiene el cuadrado del arreglo " a ".

$$Z = a.\hat{}2$$

Orientación de arreglos Si separamos cada elemento del arreglo con punto y coma tenemos un arreglo de una sola columna :

$$a = [1; 2; 3; 4; 5; 6]$$

Es necesario usar los corchetes, porque si no los usamos obtenemos el último valor que capturamos :

$$d = 1; 2; 30; 40; 50; 600; 1000$$

Para crear una columna con 20 elementos hacemos lo siguiente :

$$d = (1 : 1 : 20)$$

y trasponemos el renglón a columna, es decir buscamos la transpuesta. (')

$$e = d\prime$$

¿ Que pasa si hacemos lo siguiente : ?

$$e\prime$$

1.3.14 Matrices

Se utiliza el punto y coma (;) para construir una matriz. Por ejemplo, para formar la matriz

$$\begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 2 & 1 & 3 \end{bmatrix}$$

escribimos :

$$A = [1\ 2\ 3;\ 3\ 2\ 1;\ 2\ 1\ 3]$$

Con MATLAB se pueden resolver sistemas de ecuaciones simultáneas fácilmente, por ejemplo para resolver el siguiente sistema de ecuaciones.

$$\begin{aligned} 2x + 0y + 5z &= 100 \\ 3x + 5y + 9z &= 251 \\ 1x + 5y + 7z &= 301 \end{aligned}$$

Capturamos los valores de x, y, z ; formando una matriz.

$$A = [205; 359; 157]$$

Después capturamos el valor al cual están igualadas las ecuaciones en otra matriz.

$$b = [100; 251; 301]$$

Una forma de solucionar las ecuaciones es obteniendo el inverso de la matriz, es decir : A -1 (menos uno)

$$c = inv(A) * b$$

el asterisco indica multiplicación matricial.

Otra forma de resolverlo, es utilizando la división matricial.

$$c = A \backslash b$$

Es también posible obtener la determinante de una matriz.

$$det(A)$$

Operaciones con Matrices Definamos las siguientes matrices ' g ' y ' h '.

$$\begin{aligned} g &= [1\ 2\ 3;\ 4\ 5\ 6;\ 7\ 8\ 9] \\ h &= [1\ 0\ 2;\ 11\ 2\ 3; 3\ 5\ 12] \end{aligned}$$

La suma de las matrices g y h se muestra enseguida :

$$k = g + h$$

Multiplicación de dos matrices:

$$k = g * h$$

Para calcula la factorización LU de la matriz cuadrada k

$$[L, U] = lu(k)$$

Para calcula la factorización QR de la matriz k.

$$[d, e] = qr(k)$$

Modificación de las matrices. Sea

$$A = [1\ 2\ 3;\ 4\ 5\ 7;\ 7\ 8\ 9]$$

Si nos equivocamos al capturar la matriz, por ejemplo si el número 7 del segundo renglón, tercer columna debió ser 6 en vez de 7, tendríamos que capturar de nuevo la matriz, pero con MATLAB es posible modificarla de la siguiente manera :

$$A(2, 3) = 6$$

Es decir *Variable (renglón, columna)= nuevo valor*

Si tenemos la matriz identidad de 4 x 4 :

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A = [1\ 0\ 0\ 0;\ 0\ 1\ 0\ 0;\ 0\ 0\ 1\ 0;\ 0\ 0\ 0\ 1]$$

Pero por algún error la matriz identidad debió de haber sido de 5 x 5.¿ Hay que capturar de nuevo la matriz ? La respuesta es ¡no!

$$A(5,5) = 1$$

Agregamos un 1 en el renglón 5 columna 5, y como este no existían previamente, las columnas y renglones se completan agregando ceros.

¿ Que pasa ahora si queremos sólo una matriz identidad de 3 x 3 y tenemos capturada una de 5 x 5. Podemos utilizar :

Matriz ("Renglón" inicio : Fin , "Columna" inicio : Fin)

$$B = A(1:3, 1:3)$$

Poner dos puntos (:) indica que se deben tomar todas las columnas, por ejemplo

$$C = A(:, [1\ 3\ 5])$$

toma todos los renglones, pero sólo toma las columnas 1, 3 y 5.

Si creamos las siguientes matrices A y B :

A = [1 2 3 4 5; 1 2 3 4 5; 1 2 3 4 5; 1 2 3 4 5]
B = [6 7 8; 6 7 8; 6 7 8; 6 7 8]

Podemos construir una matriz C uniendo las dos anteriores

$$c = [A\ B]$$

A partir de la matriz A queremos tomar las columnas 1, 2 y 5, y de la matriz B queremos tomar las columnas 1 y 3, para formar una matriz D.

D = [A(:,[1 2 5]) B(:, [1 3])]
D(:,1) = [] Elimina la columna número uno.

Matrices especiales

- ones(2) Hace una matriz de unos, de 2 x 2.
- zeros(5,4) Hace una matriz de ceros, de 5 x 4.
- rand(3) Hace una matriz de 3 x 3,
- eye(4) Hace una matriz identidad de 4 x 4.

1.3.15 Gráficas

En MATLAB se pueden crear gráficas tan simples como :

D = [1 2 3 5 4 7 6 8 9 8 6 3 1 3];
plot (D)

o se pueden crear gráficas tan complejas como :

cplxroot(3,10) Superficie de una raíz cubica.

Como se vió en el primer ejemplo es posible gráficar una serie de puntos y MATLAB automáticamente ajusta los ejes donde se gráfica. Por ejemplo, para gráficar la función seno se pueden crear un rango de valores

x = 0 : 0.1 : 20; (x = vector de cero a veinte con incrementos de 0.1)
y = sin(x); (Seno del vector (x))
plot (x,y) (Gráfica del seno)
z = cos(x); (Coseno del vector anterior)
plot (x,z) (Gráfica del coseno de x.)
plot (x,y,x,z) (Gráfica del seno y coseno en la misma pantalla)
plot (x,z,'*') (Gráfica del coseno con los signos ' * ')

Hace la gráfica en azul, y los signos ' + ', intercambiando los ejes.

plot (z, x,'b+')

Como se ve es posible gráficar en Matlab con símbolos y además escoger el color, tal como se muestra en la siguiente tabla.

Símbolo	Color	Símbolo	Estilo de línea
y	*amarillo*	.	*punto*
m	*magenta*	*o*	*circulo*
c	*cían*	*x*	*equis*
r	*rojo*	+	*más*
g	*verde*	*	*asterisco*
b	*azul*	−	*menos*
w	*blanco*	:	*dos puntos*

Es posible agregar un cuadriculado a la gráfica, para tener más precisión, con el comando.

grid

Se pueden agregar títulos a las gráficas y etiquetas en los ejes con los comandos siguientes.

title(' Gráfica del coseno de x')

Para ponerle etiquetas a los ejes se puede utilizar los comandos

```
ylabel ('etiqueta')
xlabel('etiqueta')
```

Desaparece los ejes.

```
axis off
```

El comando *subplot* nos permite desplegar en pantalla varias gráficas.

```
subplot(m,n,a)
```

'm' y 'n' son una matriz que representa la cantidad de gráficas que se van desplegar; 'a' indica el lugar que ocuparía la gráfica en el subplot.

Hagamos la gráfica de los siguientes puntos. La desplegaremos en cuatro puntos diferentes en pantalla para ver las características de subplot.

```
a=[ 1 ,2 ,3 9 ,8 ,7 ,4, 5, 6, 8, 7, 5];
plot (a)
```

Vamos hacer una matriz de 2 x 2 para gráficar, cuatro posibles ventanas o gráficas. Si queremos que la primera gráfica ocupe el lugar (1,1) de la matriz. entonces escribimos.

```
subplot(2,2,1) ,plot(a)
subplot(2,2,2) , plot(a)
subplot(2,2,4), plot(a)
```

CLF borra todos los objetos de la gráfica.

CLF RESET Borra todo lo que hay en la gráfica y resetea todas las propiedades de la figura.

Gráficas en tres dimensiones. El comando plot se puede extender a 3 dimensiones con el comando plot3. El siguiente ejemplo hace una gráfica de una espiral en tres dimensiones.

```
t=0:pi/50:10*pi;
plot3(sin(t),cos(t),t)
zlabel ('etiqueta')
```

La última insrucción se utiliza para dar etiquetas al eje z, en las gráficas en tres dimensiones.

Gráficos de malla y superficie.

```
z = peaks(10)
```

El comando peaks crea un conjunto de valores que al ser gráficados, se ven de la siguiente manera.

```
plot(z)
```

Se tomará como base la gráfica anterior para demostrar algunas funciones de graficación en tres dimensiones.

```
mesh(z)
contour(z,10)
surf(z)
```

Es posible cambiar el sentido de orientación de las gráficas con el comando view(x,y)

```
view(0,0)
view(90,0)
```

Gráficas en el plano complejo Ahora vamos a crear un conjunto de valores para gráficar en el plano complejo, en tres dimensiones.

```
z= cplxgrid(5)
cplxmap(z,z)
cplxmap(z,z.^z)
```

Se pueden crear gráficos en coordenadas polares con el comando

```
polar (t,r,s)
```

donde t es el vector en ángulos en radianes, r es el radio del vector y s es la cadena de caracteres que describe , color, símbolo del estilo del línea.

```
t=0:0.1:2*pi;
r = sin(2*t).*cos(2*t);
polar(t,r)
```

El comando

$$gtext(texto)$$

se utiliza para colocar texto en una gráfica, con la ayuda del mouse. Simplemente se ejecuta el comando y con el mouse se selecciona la coordenada deseada y se presiona el botón derecho del mouse, quedando fijo el texto en la pantalla.

Copiar una gráfica Cuando se quiera realizar algún reporte formal en un procesador de palabras como en este caso Word, es posible copiar las gráficas hechas en Matlab por medio de la orden copy to bitmap.

El procedimiento sería :

- En Matlab, en el menú de la ventana principal de la gráfica, se escoge el menú edit y de este se escoge copy to bitmap;

- Se minimiza Matlab y se pasa al procesador de palabras escogido;
- Se localiza la posición en la cual estará la gráfica, y del menú edit se escoge paste o pegar.

La gráfica aparecerá en el procesador de palabras. Existe un pequeño inconveniente ya que la gráfica aparecerá sobre un fondo de color negro que Matlab tiene por default, si se imprime este documento obviamente la gráfica aparecerá sobre un fondo negro lo cual hará que la impresora gaste tinta en exceso; para remediar esto se puede cambiar el color de fondo de las gráficas a blanco con el comando.

$$Whitebg$$

después se hace procedimiento mencionado anteriormente.

Imprimir una gráfica. Se puede imprimir una gráfica directamente desde el menú de la ventana de la gráfica, seleccionando la opción print.

1.3.16 Otros comandos

- what : Listado de todos los archivos *.m en el directorio actual.
- dir : Lista todos los archivos en el directorio actual.
- type nombre_archivo : Lista el programa, (Programas con terminación **.m*).
- which nombre_archivo : Da el path en el cual esta el archivo.

1.3.17 Como hacer un programa en MATLAB

Es posible realizar un programa en Matlab tal como se hace en otros lenguajes como el basic, pascal o el lenguaje C. Es necesario utilizar un editor para escribir el código.

Para cargar un editor, se puede hacer desde la ventana options, escogiendo editor preference, y cargando el editor que se desee utilizar.

Para escribir código, requerimos crear un archivo **.m*. Para esto necesitamos abrir new *m.file* en la ventana file. Ahora escribimos el código y salvamos el archivo utilizando la terminación *archivo.m*.

Se puede correr el programa desde Matlab simplemente escribiendo el nombre del archivo que fue creado.

Es posible abrir programas con la terminación **.m* desde Matlab, en el menú file, open *m.file*.

Bucles For Tal como en otros programas de programación en Matlab es posible crear programas con estructura con ciclos for.

```
for x = Número incial : número final
      Instrucción
end
```

Por ejemplo,

```
for x = 1 : 10
      x = x + 1
end
```

También se pueden hacer operaciones como la siguiente :

```
matriz = [ 1 2 3 4; 1 2 3 4; 1 2 3 4; 1 2 3 4]
for x = matriz
      x = n(1)*n(2)*n(3)*n(4)
end
```

Bucles while While permite que ciertas instrucciones sean repetidas un número indefinido de veces bajo el control de una condición lógica.

Por ejemplo, ¿ Cual es primer entero n para el cual n! (factorial) es un número de 100 dígitos ?.

```
n = 1;
while prod(1:n) < 1e100
      n = n + 1;
end
n
```

IF ELSE END Se pueden utilizar estructuras como:

```
If expresión (verdadero)
      acción
end.
If expresión (verdadero)
      acción 1
else (Falso)
      acción 2
end.
If expresión (verdadero)
      acción 1
elseif expresión (verdadero)
      acción 2
. . .
else (Falso)
      acción "n"
end
```

1.3.18 Análisis de datos.

En Matlab podemos hacer análisis de datos estadísticamente o probabilisticamente. Entre estos análisis están cálculos de medias, máximos, mínimos, desviaciones estándar, etc.

Inventemos un conjunto de datos, los cuales podremos analizar.

x =[9 1 ;23 34; 16 28 ;12 33 ;5 7; 9 4 ;12 34 ;5 14 ;43 6 ;3 6 ;12 9; 2 30 ;3 2; 2 4]

Si escribimos el comando *plot (x)*, tenemos la representación gráfica de los puntos anteriores.

A continuación se hace una análisis de los datos presentados, habrá dos respuestas porque tenemos dos columnas.

media = mean(x) (Obtención de la media)
max(x) (El máximo de los valores.)
min(x) (El mínimo de los todos los valores)
std(x) (La desviación estándar)
hist(x) (Histograma)

1.3.19 Interpolación

Matlab tiene varios comandos que nos permiten hacer interpolaciones, uno de los métodos es por medio de mínimos cuadrados.

Mínimos cuadrados Se crean varios puntos.

x = [0 .1 .2 .3 .4 .5 .6 .7 .8 .9 1];
y =[0.09 .12 .24 .27 .4 .45 .61 .67 .71 .63 .59];

Se muestra los puntos a los cuales se les va a interpolar

plot (x,y,'')*

Se utiliza una aproximación de segundo orden, porque la función es no lineal.

n=2 ; (Segundo orden.)
p = polyfit(x,y,n) (Crea los elementos del polinomio que hará la interpolación.)

El polinomio es del tipo $ax^2 + bx + c = 0$

f = linspace(0, 1, 100); (Formamos una serie de puntos para gráficar)
z = polyval(p,f); (Evaluación polinomial)
plot(x,y,'',x,y,f,z,':')* (Hacemos la gráfica de la interpolación)

Podemos ver que la interpolación es pobre; ahora, tratemos de hacerla con un polinomio de quinto grado. El procedimiento es el mismo que el anterior.

$$n = 5\ ;$$
$$p = polyfit(x,y,n)$$
$$z = polyval(p,f);$$
$$plot(x,y,'*',x,y,f,z,':')$$

Otra forma de interpolar, es con el comando interp1.

$$g=interp1(x,y,f)$$

Se puede observar en la gráfica resultante, que parece como una aproximación lineal entre cada punto.

$$plot(x,y,'*',f,g)$$

Para una aproximación más suave es recomendable usar el comando spline, que hace una interpolación tipo cubic spline.

$$g=spline(x,y,f)$$
$$plot(x,y,'*',f,g)$$

1.3.20 Polinomios

MATLAB puede sacar ls raíces de un polinomio. Para capturar el polinomio de abajo, solamente ponemos el valor de cada variable, respetando su lugar. Como no hay termino x1 ,de todos modos se captura como cero.

$$x^3 + 5x^2 - 2$$

$$p = [1\ 5\ 0\ -2]$$

Para sacar las raíces escribimos.

$$r=roots(p)$$

1.3.21 Tips de memoria.

Para obtener la máxima velocidad en Matlab debemos tratar de vectorizar los algoritmos, por ejemplo :

```
a = 0
for a = 0:.0.1:10
    a = a + 1;
    y(a)=sin(t)
end
```

La versión vectorizada sería :

```
t= 0:0.01:10;
y = sin(t)
```

El primer ejemplo en MATLAB toma aproximadamente 15 segundos, mientras que el segundo toma sólo 0.6 segundos.

2 Solución de ecuaciones no lineales

2.1 Método de bisección

En esta sección estudiaremos el problema de la búsqueda de raíces de una función:

$$f(x) = 0$$

En este método suponemos que f es una función continua definida en el intervalo $[a, b]$ con $f(a)$ y $f(b)$ con signos diferentes. Tomando en cuenta el teorema del valor medio, existe un número $p \in [a, b]$ tal que $f(p) = 0$.

Por razones de simplicidad, asumamos que la raíz en este intervalo es real y única. El método requiere dividir varias veces a la mita el intervalo $[a, b]$ y en cada paso localizar la mitad que contenga a p.

Para iniciar, supongamos que $a_1 = a$ y $b_1 = b$, y sea p_1 el punto medio:

$$p_1 = \frac{(a_1 + b_1)}{2}$$

Si $f(p_1) = 0$, entonces $p_1 = p$; de no ser así, si $f(p_1)$ tiene el mismo signo que $f(a_1)$, entonces $p \in (p_1, b_1)$; si $f(p_1)$ tiene el mismo signo que $f(b_1)$, entonces $p \in (a_1, p_1)$; y aplicamos el mismo procedimiento a este nuevo intervalo.

Algoritmo 1 *Bisección*

Entradas: *extremos a, b; número de iteraciones ni; tolerancia tol.*

1. $p = a$; $i = 1$; $eps = 1$;
2. *Mientras* $f(p) \neq 0$ *y* $i \leq ni$ *y* $eps < tol$
   - **2.1** $pa = p$;
   - **2.2** $p = (a + b)/2$;
   - **2.3** *Si* $f(p) * f(a) > 0$ *entonces* $a = p$;
   - **2.4** *Sino, si* $f(p) * f(b) > 0$ *entonces* $b = p$;
   - **2.5** $i = i + 1$; $eps = |p - pa|/p$

Ejemplo 2 *Ejemplo,* $f(x) = \sqrt{x} - \cos x$*; en el intervalo* $[0, 1]$ *(ver figura 1):*

(mostrar software en matlab).
Ejercicios:

- Desarrolle un programa de acuerdo al algoritmo presentado y
- Aplique el programa para resolver los ejercicios 7, 8, 9 y 17 en [1]. En el ejercicio 17 explicar cada elemento del la expresión del volumen (V).

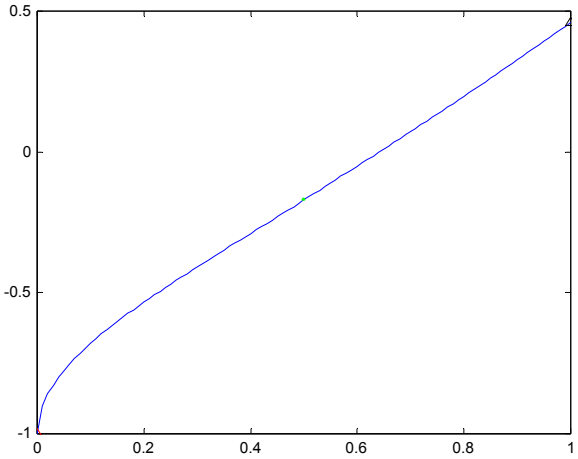

Figura 1: Función $f(x) = \sqrt{x} - \cos x$.

2.2 Iteración de punto fijo

Un punto fijo de una función $g(x)$ es un número p tal que $g(p) = p$. Los problemas de búsqueda de raíces y los de punto fijo son equivalentes en el siguiente sentido:

Dado un problema $f(x) = 0$, se puede definir una función $g(x)$ con un punto fijo en p de diferentes maneras; por ejemplo, $g(x) = x - f(x)$ o como $g(x) = x + 3f(x)$. De tal forma que, si la función $g(x)$ tiene un punto fijo en p, entonces la función definida como $F(x) = x - g(x)$ tiene un valor de cero en p:

$$\begin{aligned} F(p) &= p - g(p) \\ &= p - p = 0 \end{aligned}$$

así

$$F(p) = p - p + f(p) = 0$$

o

$$F(p) = p - p - 3f(p) = 0$$

lo que implica que $f(p) = 0$, y por lo tanto p es una raíz de $f(x)$.

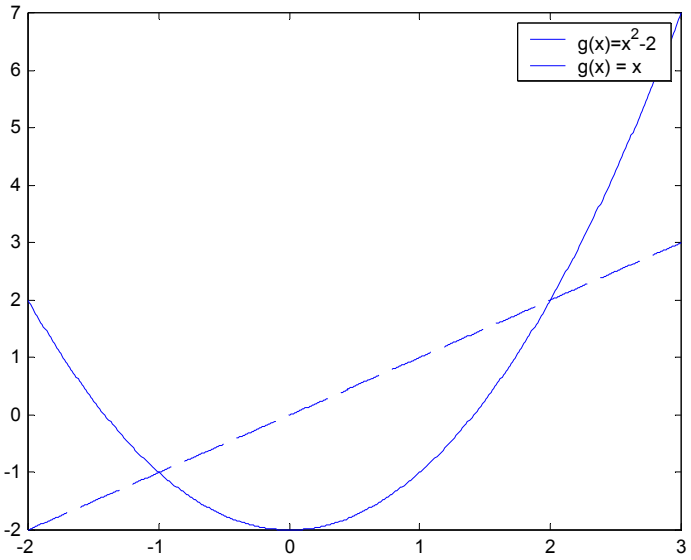

Figura 2: Función $g(x) = x^2 - 2$.

2.2.1 Punto fijo

La función $g(x) = x^2 - 2$ (Figura 2), para $-2 \leq x \leq 3$, tiene puntos fijos en $x = -1$ y $x = 2$, ya que:

$$\begin{aligned} g(-1) &= 1 - 2 = -1 \\ g(2) &= 4 - 2 = 2 \end{aligned}$$

El siguiente teorema describe la existencia y unicidad del punto fijo:

Teorema 3 *Existencia y unicidad de un punto fijo.*

a) *Si* $g(x) \in C[a,b]$ *y* $g(x) \in [a,b]$ *para toda* $x \in [a,b]$, *entonces* $g(x)$ *tiene un punto fijo en* $[a,b]$.

b) *Y si además* $g'(x)$ *existe en* (a,b) *y una constante positiva* $k < 1$ *existe con*

$$|g'(x)| \leq k, \text{ para toda } x \in (a,b)$$

entonces el punto fijo en $[a,b]$ *es único.*

Veamos un ejemplo. Sea $g(x) = \left(x^2 - 1\right)/3$ en $[-1,1]$, Figura tal 3. El valor

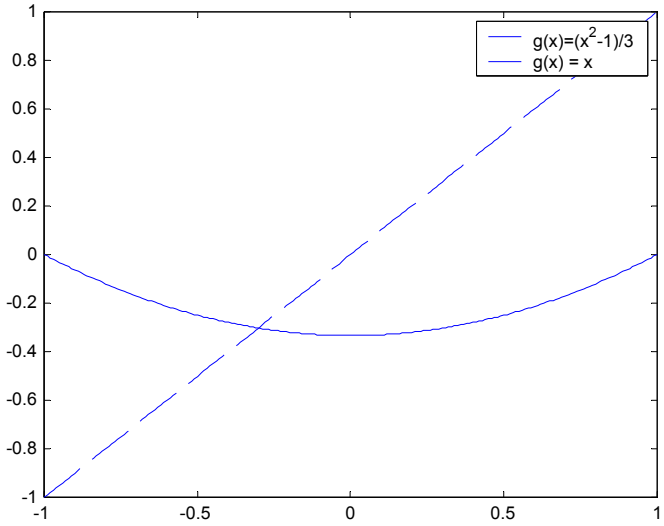

Figura 3: Función $g(x) = \left(x^2 - 1\right)/3$

mínimo de esta función en dicho intervalo es:

$$g(0) = -\frac{1}{3}$$

y además

$$g(\pm 1) = 0$$

es decir $0, -1/3 \in [-1, 1]$, y además

$$|g'(x)| = \left|\frac{2x}{3}\right| \leq \frac{2}{3} \text{ para toda } x \in [-1, 1]$$

En este ejemplo el punto fijo puede determinarse analíticamente.

$$p = g(p) = \frac{p^2 - 1}{3}, \text{ por lo tanto } 3p - p^2 + 1 = 0$$

donde una de sus raíces es

$$p = \frac{1}{2}\left(3 - \sqrt{13}\right) \approx -0.302$$

Para aproximar el punto fijo de una función $g(x)$, escogemos una aproximación inicial p_0 y generamos la sucesión $\{p_n\}_{n=0}^{\infty}$, haciendo $p_n = g(p_{n-1})$ para cada $n \geq 1$. Si la secuencia converge en p y $g(x)$ es continua, tenemos:

$$p = \lim_{n\to\infty} p_n = \lim_{n\to\infty} g(p_{n-1}) = g\left(\lim_{n\to\infty} p_{n-1}\right) = g(p)$$

Algoritmo 4 *Punto Fijo*

Entradas: *aproximación inicial p_0; número de iteraciones ni; tolerancia tol.*

1. $i = 1$; $p = g(p_0)$
2. *Mientras* $i \leq ni$ *y* $|p - p_0| > tol$
   (a) $p_0 = p$;
   (b) $p = g(p_0)$;
   (c) $i = i + 1$;

Ejemplo 5 *Encontrar las raíces de la ecuación* $x^3 + 4x^2 - 10 = 0$ *en el intervalo* $[1, 2]$.

Primero hay que convertir esta ecuación a al forma $x = g(x)$. *Existen varias formas de hacerlo pero se puede hacer mediante un simple manejo algebraico:*

$$\begin{aligned} 4x^2 &= 10 - x^3 \\ x^2 &= \frac{10 - x^3}{4} \\ x &= \pm\frac{1}{2}\left(10 - x^3\right)^{1/2} \end{aligned}$$

por lo que $g(x) = 1/2\left(10 - x^3\right)^{1/2}$ *(si escogemos la parte positiva).*

(mostrar software en matlab).

El problema es establecer la función $g(x)$ que converja a un punto fijo en una solución de un problema de búsqueda de una raíz para una función $f(x)$.

Para responder a esta respuesta tiene que ver con el siguiente teorema:

Teorema 6 *Teorema del punto fijo*

Sea $g(x) \in [a, b]$ *tal que* $g(x) \in C[a, b]$ *para toda* $x \in [a, b]$. *Además suponga-amos que existe* $g'(x) \in (a, b)$ *y una constante positiva* $k < 1$ *tal que*

$$|g'(x)| \leq k \text{ para toda } x \in (a, b)$$

Entonces, para cualquier número p_0 *en* $[a, b]$, *la sucesión definida por*

$$p_n = g(p_{n-1}), n \geq 1$$

converge en el único punto fijo $p \in [a, b]$.

Ejemplo 7 *Para* $g(x) = x - x^3 - 4x^2 + 10$, *tenemos que su derivada toma valores entre* $[-1, 1]$ *en el rango* $(-2.9, -2.68)$, *como puede verse en la Figura 4. En este rango se aprecia claramente que la función* $g(x) \notin (-2.9, -2.68)$, *al compararse dicha función con la recta* $y = x$, *por lo que no se asegura la convergencia a un punto fijo.*

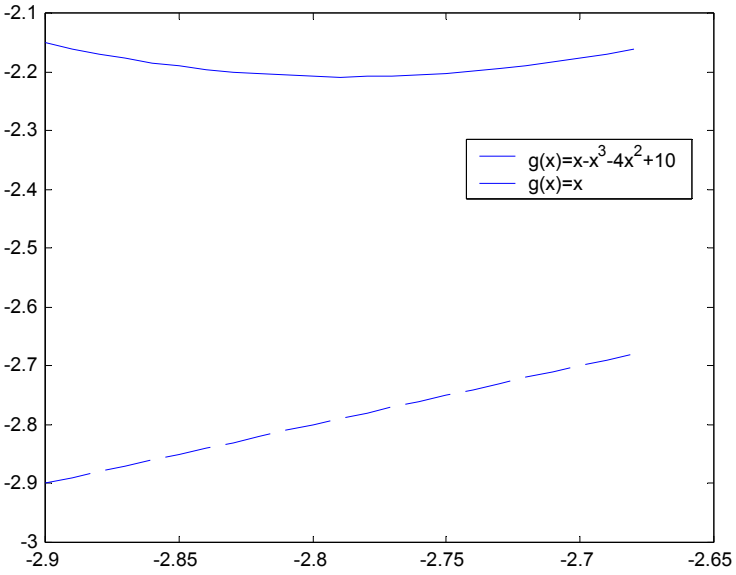

Figura 4: Gráficas de la función $g(x) = x - x^3 - 4x^2 + 10$, su derivada y de la identidad $y = x$.

Ejercicios:

- Desarrolle un programa de acuerdo al algoritmo presentado y
- Aplique el programa para resolver los ejercicios 1, 2, 6 y 7 de la página 63 en [1]. Nota: para el ejercicio 7 aplique el teorema de "Existencia y unicidad de un punto fijo".

2.3 Método de Newton-Raphson

Supongamos que $f \in C^2[a,b]$. Sea $\overline{x} \in [a,b]$ una aproximación de p (una raíz de f) tal que $f'(x) \neq 0$ y $|\overline{x} - p|$ es pequeño. Considerando el primer polinomio de la expansión de Taylor de $f(x)$ alrededor de $\overline{x}$,

$$f(x) = f(\overline{x}) + (x - \overline{x}) f'(\overline{x}) + \frac{(x-\overline{x})^2}{2} f''(\xi(x))$$

donde $\xi(x)$ está entre x y $\overline{x}$. Dado que $f(p) = 0$, entonces al sustituir en esta ecuación $x = p$ se tiene

$$0 = f(\overline{x}) + (p - \overline{x}) f'(\overline{x}) + \frac{(p-\overline{x})^2}{2} f''(\xi(p))$$

El método de Newton supone que $|p - \overline{x}|$ es muy pequeño, por lo que $(p-\overline{x})^2$ es mucho menor y por lo tanto

$$0 \approx f(\overline{x}) + (p - \overline{x}) f'(\overline{x})$$

que al despejar p obtenemos

$$p \approx \overline{x} - \frac{f(\overline{x})}{f'(\overline{x})}$$

Esto nos prepara para introducir el método de Newton-Raphson, el cual comienza con una aproximación inicial p_0 y genera la sucesión $\{p_n\}$, definida como

$$p_n = p_{n-1} - \frac{f(p_{n-1})}{f'(p_{n-1})} \text{ para } n \geq 1 \tag{1}$$

Algoritmo 8 *Método de Newton-Raphson*

Entradas *aproximación inicial p_0; número de iteraciones ni; tolerancia tol.*

1. $i = 1; eps = 1$
2. *Mientras* $i \leq ni$ *y* $|eps| > tol$
   a. $p = p_0 - f(p_0)/f'(p_0)$;
   b. $eps = p_0 - p$;

c. $p = p_0$;
d. $i = i + 1$;

(mostrar software en matlab).

Este método a menudo se usa para refinar las respuestas obtenidas con otras técnicas, como el método de la bisección. Dado que el método de Newton requiere una buena aproximación inicial (p_0), pero por lo general da una buena convergencia rápida, sirve perfectamente para el propósito antes mencionado.

Ejercicios:

- Demuestre que se puede llegar a la misma expresión 1 calculando el punto de intersección de la recta con pendiente $f'(p_{n-1})$ y que pasa por el punto $(p_{n-1}, f(p_{n-1}))$ con el eje de las x.
- Implemente el algoritmo de Newton-Raphson y realice los ejercicios 5 y 6 de la página 75 de [1].

2.4 Ceros de Polinomios

Un polinomio de grado n tiene la forma

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_1 x + a_0$$

donde las a_i son los coeficientes constantes de $P(x)$ y $a_n \neq 0$.

Si queremos localizar los ceros aproximados de u polinomio $P(x)$ con el procedimiento de Newton-Raphson, se necesita evaluar $P(x)$ y su derivada en valores específicos de x. Una forma de evaluar un polinomio de grado n eficientemente es por medio del método de Horner, el cual requiere sólo de n multiplicaciones y n sumas.

2.4.1 Método de Horner

Sea

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_1 x + a_0$$

si $b_n = a_n$ y

$$b_k = a_k + b_{k+1} x_0, \text{ para } k = n-1, n-2, ..., 1, 0$$

Y si

$$Q(x) = b_n x^{n-1} + b_{n-1} x^{n-2} + ... + b_2 x + b_1$$

entonces

$$P(x) = (x - x_0) Q(x) + b_0$$

Demostración (pedir que la hagan de tarea): Según la definición de $Q(x)$.

$$\begin{aligned}(x-x_0)Q(x)+b_0 &= (x-x_0)\left(b_nx^{n-1}+b_{n-1}x^{n-2}+...+b_2x+b_1\right)+b_0\\ &= \left(b_nx^n+b_{n-1}x^{n-1}+...+b_2x^2+b_1x\right)-\\ &\quad \left(b_nx_0x^{n-1}+b_{n-1x_0}x^{n-2}+...+b_2x_0x+b_1x_0\right)+b_0\\ &= b_nx^n+x^{n-1}(b_{n-1}-b_nx_0)+x^{n-2}(b_{n-2}-b_{n-1}x_0)+\\ &\quad x(b_1-b_2x_0)+(b_0-b_1x_0)\end{aligned}$$

según la definición de b_k tenemos que

$$(x-x_0)Q(x)+b_0=a_nx^n+a_{n-1}x^{n-1}+...+a_1x+a_0$$

y por lo tanto $(x-x_0)Q(x)+b_0=P(x)$ y $P(x_0)=b_0$.

Así, para evaluar $P(x_0)$ se necesita solamente calcular las b_k para $k=n-1,...,0$.

Ejemplo 9 *Aplicar el método de Horner para evaluar* $P(x)=2x^4-3x^2+3x-4$ *para* $x_0=-2$.

	$a_4=2$	$a_3=0$	$a_2=-3$	$a_1=3$	$a_0=-4$
$x_0=-2$	$b_4=2$	$b_3=-4$	$b_2=5$	$b_1=-7$	$b_0=10$

Por lo que $P(x)=(x+2)\left(2x^3-4x^2+5x-7\right)+10$ *y* $P(-2)=10$

Una ventaja de usar este método consiste en que, como $P(x)=(x-x_0)Q(x)+b_0$, la derivadas con respecto a x es

$$P'(x)=(x-x_0)Q'(x)+Q(x)$$

y el resultado de evaluar esta expresión en x_0 es por lo tanto $P'(x_0)=Q(x_0)$.

Cuando usamos el método de Newton-Raphson para encontrar un cero aproximado de un polinomio, podemos evaluar al mismo tiempo $P(x)$ y $P'(x)$, aplicando el método de Horner para calcular ambas expresiones.

Utilizando el ejemplo anterior, tenemos

	$a_4=2$	$a_3=0$	$a_2=-3$	$a_1=3$	$a_0=-4$
	$b_4=2$	$b_3=-4$	$b_2=5$	$b_1=-7$	$b_0=10$
	$a'_3=2$	$a'_2=-4$	$a'_1=5$	$a'_0=-7$	
$x_0=-2$	$b'_3=2$	$b'_2=-8$	$b'_1=21$	$b'_0=-49$	

De tal forma que $P(-2)=10$ y $P'(-2)=-49$.

Algoritmo 10 *Evaluación de un polinomio de grado n y su derivada utilizando el método de Horner. (pedir que lo desarrollen y lo expliquen en clase)*

Entradas *Grado del polinomio n, coeficientes* $a_n,a_{n-1},...,a_1,a_0$ *y* x_0.

1. $b=a_n$; *(en la variable b se almacenará al final* $P(x_0)$*)*

2. $c = a_n$; *(en la variable c se almacenará al final $P'(x_0)$)*
3. *Para* $i = n - 1 : 1$
   (a) $b = b * x_0 + a_i$;
   (b) $c = c * x_0 + b$;
4. $b = b * x_0 + a_0$;

Si en la iteración N, x_N es una aproximación cero de P en el procedimienton de Newton-Raphson, entonces

$$P(x) = (x - x_N) Q(x) + b_0 = (x - x_N) Q(x) + P(x_n) \approx (x - x_N) Q(x)$$

Suponiendo que $\widehat{x}_1 = x_N$ sea al cero aproximado de P y que $Q_1(x) \equiv Q(x)$ sea el factor aproximado, se tiene que

$$P(x) \approx (x - \widehat{x}_1) Q_1(x).$$

Si aplicamos el método de Newton-Raphson a $Q_1(x)$, podemos encontrar un segundo cero aproximado de P. Si $P(x)$ es un polinomio de grado n con n ceros reales, se aplica varias veces este procedimiento para finalmente obtener por resultado $(n-2)$ ceros aproximados de P y un factor aproximado $Q_{n-2}(x)$. En esta etapa, se pude resolver $Q_{n-2} = 0$ mediante una fórmula cuadrática para obtener los dos últimos ceros aproximados. Note que en este método, para obtener todos los ceros aproximados, se basa en el uso repetido de aproximaciones y puede generar resultados imprecisos.

Proyecto Calcule con el procedimiento descrito las raíces del polinomio

$$P(x) = x^3 + 3x^2 - 1$$

cuya gráfica se muestra en la Figura 5.para ello, cada que encuentre un nuevo factor aproximado $Q_n(x)$, para $n = 1..n - 3$, grafique el polinomio correspondiente y su raíz.

Ejercicios:

- Resuelva el problema 11 de la página 101 de [1].

3 Solución de Ecuaciones Lineales

3.1 Sistemas de ecuaciones lineales

Los sistemas de ecuaciones lineales se utilizan en muchos problemas de ingeniería y ciencias. La idea principal es encontrar los valores de las incógnitas $x_1, x_2, ..., x_n$, si es que existen, de un sistema de ecuaciones lineales en los que se involucran dichas incógnitas. Cuando el número de ecuaciones es igual al número de incógnitas, generalmente existe una solución.

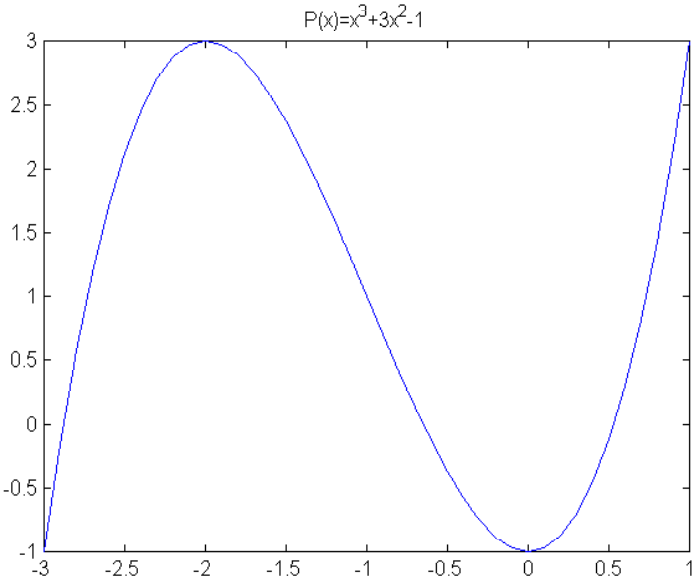

Figura 5: Gráfica del polinomio $P(x) = x^3 + 3x^2 - 1$.

Una manera de visualizar este problemas, desde el punto de vista geométrico, es que cada ecuación representa un hiperplano en el espacio *n-dimensional*, y el objetivo es encontrar un punto o conjunto de puntos que satisfagan las ecuaciones.

Veamos el siguiente ejemplo: encontrar los valores de x, y, z que satisfacen las ecuaciones

$$\begin{aligned} x + 2y + 3z &= 6 \quad (2) \\ 2x + 5y + 2z &= 4 \quad (3) \\ 6x - 3y + z &= 2 \end{aligned}$$

Si graficamos las dos primeras ecuaciones (planos en xyz), tenemos lo que se muestra en la Figura 6. Como se puede ver en la figura los dos planos se intersectan a lo largo de una línea; ahora, si para este sistema existiera un solución única, esta línea debería intersectar a su vez el plano correspondiente a la tercera ecuación en un solo punto, lo que se muestra en la Figura 7.

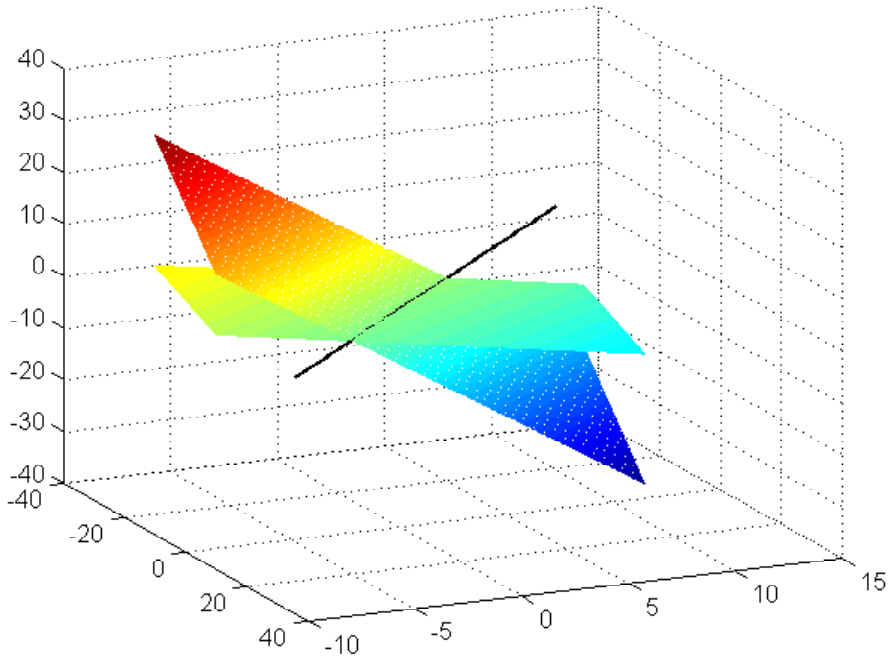

Figura 6: Intersección de dos planos

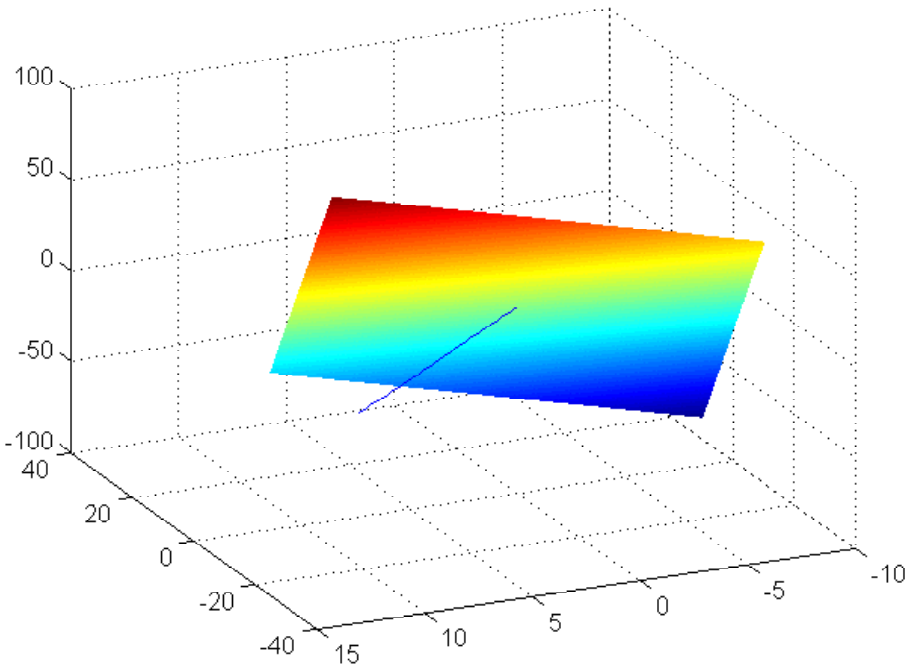

Figura 7: Intersección con el tercer plano.

3.2 Métodos Directos de Solución de Sistemas de Ecuaciones Lineales

En esta sección examinaremos métodos directos para resolver sistemas de ecuaciones lineales:

$$\begin{array}{rcl} E_1 & : & a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = b_1 \\ E_2 & : & a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = b_2 \\ & & . \quad . \quad . \quad . \\ & & . \quad . \quad . \quad . \\ & & . \quad . \quad . \quad . \\ E_n & : & a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n = b_n \end{array}$$

donde $x_1, x_2, ..., x_n$ son desconocidas, dadas las a_{ij} para $i, j = 1, 2, ..., n$ y b_i para $i = 1, 2, ..., n$.

En el desarrollo de los métodos de solución de estas ecuaciones lineales podremos realizar las siguientes tres operaciones sin que se altere de ninguna forma el sistema:

1. La ecuación E_i puede multiplicarse por una constante λ distinta de cero y la ecuación resultante se emplea en lugar de E_i. Esta operación se denota por $(\lambda E_i) \rightarrow E_i$.

2. La ecuación E_j puede multiplicarse por cualquier constante λ, y sumarse a la ecuación E_i; la ecuación resultante se emplea en lugar de la ecuación E_i. Esta operación se denota por $(\lambda E_j + E_i) \rightarrow E_i$.

3. El orden de las ecuaciones E_i y E_j puede intercambiarse. Esta operación se denota por $E_i \longleftrightarrow E_j$.

Con las operaciones mencionadas podemos transformar un sistema lineal en otro que pueda ser resuelto fácilmente con las mismas soluciones.

Ejemplo 11 *Simplifique y resuelva el siguiente sistema de ecuaciones.*

$$\begin{array}{rrcrcrcrcr} E_1: & x_1 & + & x_2 & & & + & 3x_4 & = & 4 \\ E_2: & 2x_1 & + & x_2 & - & x_3 & + & x_4 & = & 1 \\ E_3: & 3x_1 & - & x_2 & - & x_3 & + & 2x_4 & = & -3 \\ E_4: & -x_1 & + & 2x_2 & + & 3x_3 & - & x_4 & = & 4 \end{array}$$

El primer paso que realizaremos será eliminar la incógnita x_1 en las ecuaciones E_2, E_3, E_4 al efectuar las operaciones $(E_2 - 2E_1) \rightarrow E_2, (E_3 - 3E_1) \rightarrow E_3, (E_4 + E_1) \rightarrow E_4$, obteniendo el siguiente sistema:

$$\begin{array}{rrcrcrcrcr} E_1: & x_1 & + & x_2 & & & + & 3x_4 & = & 4 \\ E_2: & & - & x_2 & - & x_3 & - & 5x_4 & = & -7 \\ E_3: & & - & 4_{x2} & - & x_3 & - & 7x_4 & = & -15 \\ E_4: & & & 3x_2 & + & 3x_3 & - & 2x_4 & = & 8 \end{array}$$

Con este nuevo sistema, utilizamos ahora E_2 para eliminar x_2 de las ecuaciones E_3, E_4 al efectuar las operaciones $(E_3 - 4E_2) \rightarrow E_3, (E_4 + 3E_2) \rightarrow E_4$, obteniendo:

$$\begin{array}{lrcrcrcr} E_1: & x_1 & + & x_2 & & + 3x_4 & = & 4 \\ E_2: & & - & x_2 & - \ x_3 & - 5x_4 & = & -7 \\ E_3: & & & & 3x_3 & + 13x_4 & = & 13 \\ E_4: & & & & & - \ 13x_4 & = & -13 \end{array}$$

Este sistema presenta ahora una forma triangular superior que puede resolverse utilizando un proceso de sustitución hacia atrás:

$$\begin{aligned} x_4 &= 1 \\ x_3 &= \frac{13 - 13x_4}{3} = 0 \\ x_2 &= -(-7 + 5x_4) = 2 \\ x_1 &= 4 - x_2 - 3x_4 = -1 \end{aligned}$$

por lo tanto la solución es $x_1 = -1, x_2 = 2, x_3 = 0, x_4 = 1$.

Al realizar los cálculos anteriores no es necesario escribir todo el sistema de ecuaciones (las incógnitas y sus coeficientes), es suficiente trabajar con los coeficientes y realizar las operaciones antes descritas con ellos. Por tal razón, a menudo un sistema lineal se reemplaza por una matriz que contiene toda la información sobre el sistema necesaria para determinar su solución.

La notación para determinar una matriz $n \times m$ es la siguiente:

$$A = \begin{bmatrix} a_{11} & a_{12} & ... & a_{1m} \\ a_{21} & a_{22} & ... & a_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & ... & a_{nm} \end{bmatrix}$$

En el caso de la matriz $1 \times n$

$$A = [a_{11} \ \ a_{12} \ \ ... \ a_{1n}]$$

recibe el nombre de vectror renglón o vector columna cuando se escriba

$$A = \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{n1} \end{bmatrix}$$

En ambos casos es posible omitir los subíndeices innecesario, escribiendo

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Una matriz de $n \times (n+1)$ puede utilizarse para remplazar un sistema lineal de la forma

$$A = \begin{bmatrix} a_{11} & a_{12} & ... & a_{1m} \\ a_{21} & a_{22} & ... & a_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & ... & a_{nm} \end{bmatrix} \text{ y } b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

y luego combinando estas matrices para formar la matriz aumentada

$$[A, b] = \left[\begin{array}{cccc:c} a_{11} & a_{12} & ... & a_{1m} & b_1 \\ a_{21} & a_{22} & ... & a_{2m} & b_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & ... & a_{nm} & b_n \end{array}\right]$$

Al repetir las operaciones descritas en el ejemplo anterior, en matriz aumentada

$$\left[\begin{array}{cccc:c} 1 & 1 & 0 & 3 & 4 \\ 2 & 1 & -1 & 1 & 1 \\ 3 & -1 & -1 & 2 & -3 \\ -1 & 2 & 3 & -1 & 4 \end{array}\right]$$

se obtiene

$$\left[\begin{array}{cccc:c} 1 & 1 & 0 & 3 & 4 \\ 0 & -1 & -1 & -5 & -7 \\ 0 & -4 & -1 & -7 & -15 \\ 0 & 3 & 3 & 2 & 8 \end{array}\right] \text{ y } \left[\begin{array}{cccc:c} 1 & 1 & 0 & 3 & 4 \\ 0 & -1 & -1 & -5 & -7 \\ 0 & 0 & 3 & 13 & 13 \\ 0 & 0 & 0 & -13 & -13 \end{array}\right]$$

Finalmente esta última matriz se puede transformar en su correspondiente sistema lineal y obtener las soluciones para x_1, x_2, x_3, x_4. A este proceso se le denomina **eliminación Gaussiana con sustitución hacia atrás**.

El procedimiento general de eliminación Gaussiana que se aplica al sistema lineal

$$\begin{array}{rcl} E_1 & : & a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = b_1 \\ E_2 & : & a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = b_2 \\ & & \cdot \quad \cdot \quad \cdot \quad \cdot \\ & & \cdot \quad \cdot \quad \cdot \quad \cdot \\ & & \cdot \quad \cdot \quad \cdot \quad \cdot \\ E_n & : & a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n = b_n \end{array}$$

se forma, primero construyendo la matriz aumentada

$$\widetilde{A} = [A, b] = \left[\begin{array}{ccccc:c} a_{11} & a_{12} & ... & a_{1m} & \vdots & a_{1n+1} \\ a_{21} & a_{22} & ... & a_{2m} & \vdots & a_{2n+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & ... & a_{nm} & \vdots & a_{nn+1} \end{array}\right]$$

Siempre que $a_{ii} \neq 0$ las operaciones para eliminar los coeficientes a_{ji} para $j = i + 1, .., n$ corresponden a

$$\left(E_j - \frac{a_{ji}}{a_{ii}} E_i\right) \rightarrow E_j$$

La matriz resultante de este proceso debe quedar en forma triangular superior

$$\widetilde{\widetilde{A}} = \left[\begin{array}{ccccc:c} a_{11} & a_{12} & ... & a_{1n} & \vdots & a_{1n+1} \\ 0 & a_{22} & ... & a_{2n} & \vdots & a_{2n+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & ... & 0 & a_{nn} & \vdots & a_{nn+1} \end{array}\right]$$

note que aunque hemos utilizado la misma notación a_{ij} para la matriz resultante $\widetilde{\widetilde{A}}$, no implica que los coeficientes sean los mismos que la matriz original A debido al proceso de eliminación Gaussiana.

Finalmente, para obtener los valores de las incógnitas x_{ij}, realizamos la sustitución hacia atrás, iniciando con la ecuación n-ésima para x_n

$$x_n = \frac{a_{nn+1}}{a_{nn}}$$

Al resolver para la $(n-1)$-ésima ecuación tenemos

$$x_{n-1} = \frac{a_{n-1n+1} - a_{n-1n} x_n}{a_{n-1n-1}}$$

para la $(n-2)$-ésima ecuación

$$x_{n-2} = \frac{a_{n-2n+1} - a_{n-2n-1} x_{n-1} - a_{n-2n} x_n}{a_{n-2n-2}}$$

Generalizando para las n incógnitas se tiene que

$$x_i = \frac{a_{in+1} - \sum_{j=i+1}^{n} a_{ij} x_j}{a_{ii}}$$

El procedimiento de eliminación Gaussiana puede ser descrito con mayor precisión, aunque esto supone mayor complejidad, formando una sucesión de

matrices aumentadas $\widetilde{A}^{(1)}, \widetilde{A}^{(2)}, ..., \widetilde{A}^{(k)}$, donde $\widetilde{A}^{(1)}$ es la matriz original aumentada y $\widetilde{A}^{(k)}$ para $k = 2, ..., n$ se tienen los elementos de $a_{ij}^{(k)}$:

$$a_{ij}^{(k)} = \begin{cases} a_{ij}^{(k-1)} \text{ para } i = 1, 2, ..., k-1 \text{ y } j = 1, 2, ..., n \\ 0 \text{ para } i = k, k+1, ..., n \text{ y } j = 1, 2, ..., k-1 \\ a_{ij}^{(k-1)} - \frac{a_{ik-1}^{(k-1)}}{a_{k-1k-1}^{(k-1)}} a_{k-1j}^{(k-1)} \text{ para } i = k, k+1, ..., n \text{ y } j = k, k+1, ..., n+1 \end{cases}$$

por lo tanto

$$\widetilde{A}^{(k)} = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & a_{13}^{(1)} & ... & a_{1k-1}^{(1)} & a_{1k}^{(1)} & ... & a_{1n}^{(1)} & a_{1n+1}^{(1)} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} & ... & a_{2k-1}^{(2)} & a_{2k}^{(2)} & ... & a_{2n}^{(2)} & a_{2n+1}^{(2)} \\ & & & & \vdots & \vdots & \vdots & \vdots & \vdots \\ & & & & a_{k-1k-1}^{(k-1)} & a_{k-1k}^{(k-1)} & ... & a_{k-1n}^{(k-1)} & a_{k-1n+1}^{(k-1)} \\ & & & & 0 & a_{kk}^{(k)} & ... & a_{kn}^{(k)} & a_{kn+1}^{(k)} \\ & & & & \vdots & \vdots & \vdots & \vdots & \vdots \\ & & & & 0 & a_{nk}^{(k)} & ... & a_{nn}^{(k)} & a_{nn+1}^{(k)} \end{bmatrix}$$

representa el sistema lineal equivalente para el cual la variable x_{k-1} acaba de eliminarse para las ecuaciones $E_k, E_{k+1}, ..., E_n$.

El procedimiento fallará si uno de sus elementos $a_{11}^{(1)}, a_{22}^{(2)}, ..., a_{nn}^{(n)}$ es igual a cero, ya que en el paso

$$\left(E_i - \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} E_k \right) \rightarrow E_i$$

no puede efectuarse o si no es posible realizar la sustitución hacia atrás cuando $a_{nn}^{(n)} = 0$. Lo cual no significa necesariamente que el sistema no tiene solución, sino que debe modificarse el método para obtenerla.

Ejemplo 12 *Considere el siguiente sistema lineal:*

$$\begin{bmatrix} E_1: & x_1 & - x_2 & + 2x_3 & - x_4 & = & -8 \\ E_2: & 2x_1 & - 2x_2 & + 3x_3 & - 3x_4 & = & -20 \\ E_3: & x_1 & + x_2 & + x_3 & & = & -2 \\ E_4: & x_1 & - x_2 & + 4x_3 & + 3x_4 & = & 4 \end{bmatrix}$$

La matriz aumentada es

$$\widetilde{A} = \widetilde{A}^{(1)} = \begin{bmatrix} 1 & -1 & 2 & -1 & -8 \\ 2 & -2 & 3 & -3 & -20 \\ 1 & 1 & 1 & 0 & -2 \\ 1 & -1 & 4 & 3 & 4 \end{bmatrix}$$

y al efectuar las operaciones

$$(E_2 - 2E_1) \rightarrow E_2, (E_3 - E_1) \rightarrow E_3 \ y \ (E_4 - E_1) \rightarrow E_4$$

se obtiene

$$\widetilde{A}^{(2)} = \begin{bmatrix} 1 & -1 & 2 & -1 & -8 \\ 0 & 0 & -1 & -1 & -4 \\ 0 & 2 & -1 & 1 & 6 \\ 0 & 0 & 2 & 4 & 12 \end{bmatrix}$$

Dado que $a_{22} = 0$, llamado elemento pivote, no podemos seguir con el procedimiento antes descrito. Lo que se puede hacer en estos casos es intercambiar las ecuaciones, por ejemplo en este caso se podría hacer $E_2 \leftrightarrow E_3$, quedando

$$\widetilde{A}^{(3)} = \begin{bmatrix} 1 & -1 & 2 & -1 & -8 \\ 0 & 2 & -1 & 1 & 6 \\ 0 & 0 & -1 & -1 & -4 \\ 0 & 0 & 2 & 4 & 12 \end{bmatrix}$$

con lo cual podemos seguir el procedimiento. Finalmente, aplicando $(E_4 + 2E_3) \rightarrow E_4$ se tiene

$$\widetilde{A}^{(4)} = \begin{bmatrix} 1 & -1 & 2 & -1 & -8 \\ 0 & 2 & -1 & 1 & 6 \\ 0 & 0 & -1 & -1 & -4 \\ 0 & 0 & 0 & 2 & 4 \end{bmatrix}$$

Para obtener la solución al sistema, se realiza la sustitución hacia atrás:

$$\begin{aligned} x_4 &= \frac{4}{2} = 2 \\ x_3 &= \frac{-4 + x_4}{-1} = 2 \\ x_2 &= \frac{6 + x_3 - x_4}{2} = 3 \\ x_1 &= -8 + x_2 - 2x_3 + x_4 = -7 \end{aligned}$$

En el ejemplo anterior se mostró que hacer cuando $a_{kk}^{(k)} = 0$ para alguna $k = 1, 2, ..., n-1$. Lo que se hace en este caso es buscar alguna $a_{pk}^{(k)} \neq 0$ para alguna $p, k+1 \leq p \leq n$, y se efectúa la operación $E_k \rightarrow E_p$ para obtener $\widetilde{A}^{(k+1)}$. Si no existe alguna $a_{pk}^{(k)} \neq 0$ se puede demostrar lineal no tiene solución única y el procedimiento se interrumpe. En el caso de que $a_{nn}^{(n)} = 0$, el sistema tampoco tiene solución única.

El algoritmo completo de eliminación Gaussiana con sustitución hacia atrás se presenta a continuación.

Algoritmo 13 *Eliminación Gaussiana con sustitución hacia atrás.*

Para resolver el sistema lineal $n \times n$

$$\begin{array}{rcl} E_1 & : & a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = a_{1n+1} \\ E_2 & : & a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = a_{2n+1} \\ & & . \quad . \quad . \quad . \\ & & . \quad . \quad . \quad . \\ & & . \quad . \quad . \quad . \\ E_n & : & a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n = a_{nn+1} \end{array}$$

Entradas: número de incógnitas y ecuaciones n*; matriz aumentada* $A = (a_{ij})$*, donde* $i = 1, 2, ..., n$ *y* $j = 1, 2, ..., n+1$*;*

1. *Para* $i = 1, .., n-1$
   (a) *Sea* p *el entero más pequeño en valor absoluto, con* $i \leq p \leq n$ *y* $a_{pi} \neq 0$*; Si no existe parar ya que no existe solución única,*
   (b) *Si* $p \neq i$ *realizar* $E_i \longleftrightarrow E_p$*;*
   (c) *Para* $j = i+1, ..., n$
   c.1 *hacer* $m = a_{ji}/a_{ii}$*;*
   c.2 *hacer* $(E_j - mE_i) \rightarrow E_j$*;*
2. *Si* $a_{nn} = 0$ *parar ya que no existe solución única.*
3. $x_n = a_{nn+1}/a_{nn}$
4. *Para* $i = n-1, ..., 1$
   (a) $x_i = \left[a_{in+1} - \sum_{j=i+1}^{n} a_{ij}x_j\right]/a_{ii}$*;*

Aplicando el método de eliminación Gaussiana a el sistema de ecuaciones dado en (2) (Figuras 6 y 7) se obtiene el siguiente sistema equivalente

$$\begin{array}{rcl} x + 2y + 3z & = & 6 \\ 1y - 4z & = & -8 \\ -77z & = & -154 \end{array}$$

La gráfica de los planos de las dos primeras ecuaciones se muestran en al Figura 8. Observe que el segundo plano corresponde a una recta en el plano yz que se extiende a lo largo del plano xy. Finalmente la línea que intersecta estos dos planos, a su vez intersecta al plano $z = 2$ en el punto $(0, 0, 2)$, como se observa en la Figura 9.

Exercise 14 *Resolver los siguientes sistemas lineales utilizando el algoritmo anteriormente descrito descrito.*

$$\begin{array}{rcrcrcl} x_1 & + & x_2 & + & x_3 & = & 4 \\ 2x_1 & + & 2x_2 & + & x_3 & = & 6 \\ x_1 & + & x_2 & + & 2x_3 & = & 6 \end{array} \quad y \quad \begin{array}{rcrcrcl} x_1 & + & x_2 & + & x_3 & = & 4 \\ 2x_1 & + & 2x_2 & + & x_3 & = & 4 \\ x_1 & + & x_2 & + & 2x_3 & = & 6 \end{array}$$

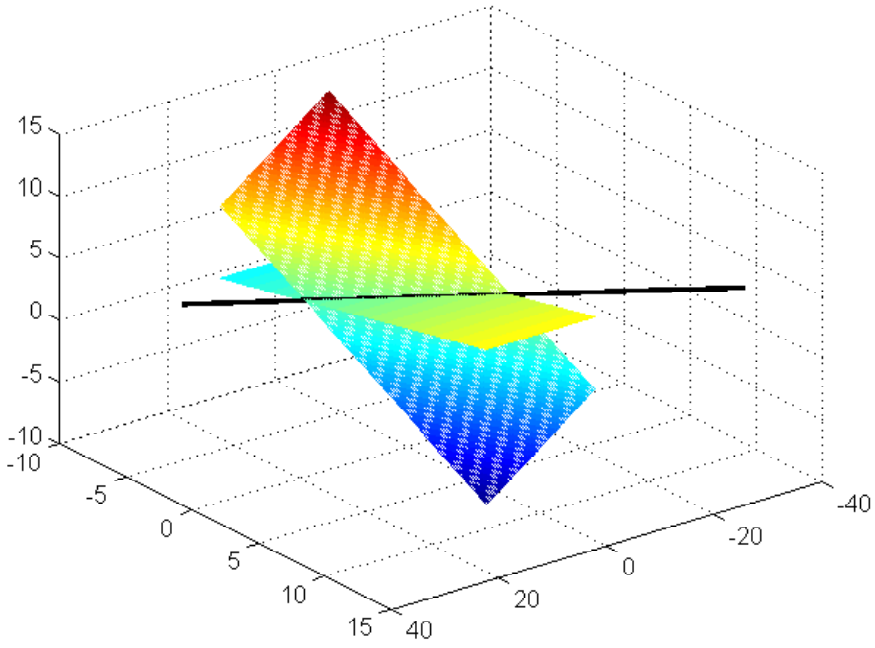

Figura 8: Primeros dos planos del sistema reducido.

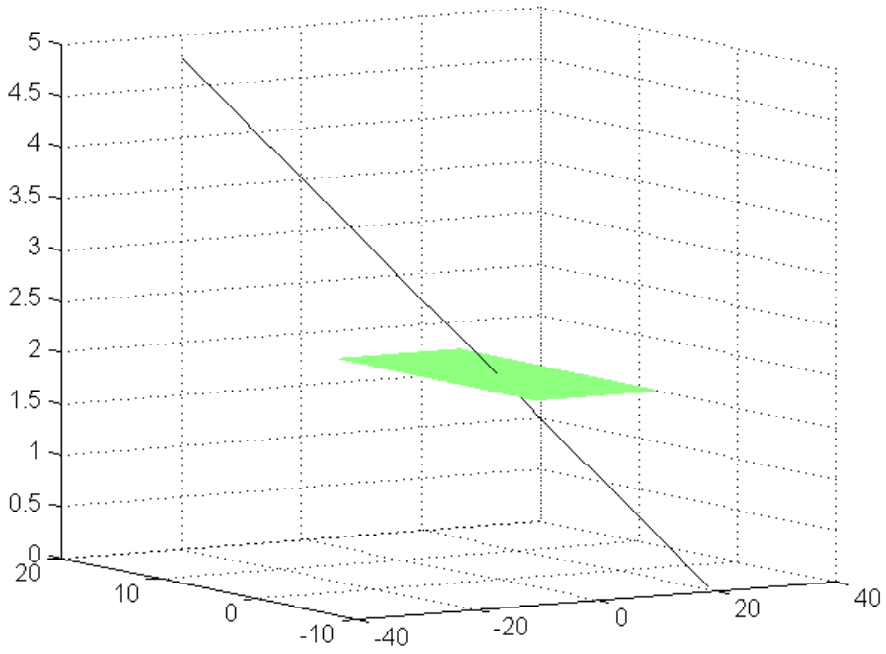

Figura 9: Intersección de los tres planos en el punto $(0, 0, 2)$.

3.3 Algebra Lineal e Inversa de Matrices

Antes de explicar como calcular la inversa de una matriz por medio de métodos numéricos, es necesario recordar algunas operaciones elementales de las matrices.

- Dos matrices A y B son iguales si tienen el mismo tamaño y si $a_{ij} = b_{ij}$, para cada $i = 1, ..., n$ y $j = 1, ..., m$.
- Si A y B son ambas matrices de $n \times m$, entonces la suma de ellas $A + B$ es la matriz cuyos elementos son $a_{ij} + b_{ij}$.
- Si A es un matriz de $n \times m$ y si λ es un número real, entonces la multiplicación $\lambda\ A$, es la matriz de $n \times m$ cuyos elementos son λa_{ij}.

Sean A, B y C matrices de $n \times m$ y sean λ y μ escalares. Se aplican las siguientes propiedades de suma y multiplicación por escalar:

- $A + B = B + C$
- $(A + B) + C = A + (B + C)$
- $A + O = O + A = A$
- $A + (-A) = -A + A = O$
- $\lambda\,(A + B) = \lambda A + \lambda B$
- $(\lambda + \mu)\,A = \lambda A + \mu A$
- $\lambda\,(\mu A) = (\lambda\mu)\,A$
- $1A = A$

Sea A una matriz de $n \times m$ y B una matriz de $m \times p$. El producto de AB, es una matriz C de $n \times p$ cuyos elementos c_{ij} están dados por

$$c_{ij} = \sum_{k=1}^{m} a_{ik} b_{kj}$$

podemos considerar el cálculo de c_{ij} como el producto del *i-ésimo* renglón de la matriz A y el *j-ésima* columna de la matriz B:

$$[a_{i1}, a_{i2}, ..., a_{im}] \begin{bmatrix} b_{1j} \\ b_{2j} \\ . \\ . \\ . \\ b_{mj} \end{bmatrix} = c_{ij}$$

(pedir que realicen el algoritmo de multiplicación de matrices).

Algunas matrices especiales son definidas a continuación.

- Una matriz cuadrada tiene la misma cantidad de renglones que de columnas. Una matriz diagonal es una matriz cuadrada $D = (d_{ij})$ con $d_{ij} = 0$ cuando $i \neq j$. La matriz identidad de orden n, $I_n = (\delta_{ij})$, es una matriz diagonal con elementos

$$\delta_{ij} = \begin{cases} 1 \text{ si } i = j \\ 0 \text{ si } i \neq j \end{cases}$$

- Una matriz triangular superior de $n \times m$, $U = (u_{ij})$ tiene los elementos

$$u_{ij} = 0, \text{ para toda } i = j+1, j+2, ..., n$$

y una matriz triangular inferior $L = (l_{ij})$

$$l_{ij} = 0, \text{ para toda } i = 1, 2, ..., j-1$$

Sea A una matriz de $n \times m$, B una matriz de $m \times k$, C una matriz de $k \times p$, D una matriz de $m \times k$ y λ un escalar. Las siguientes propiedades son aplicables:

- $A(BC) = (AB)C$
- $A(B + D) = AB + AD$
- $IB = B$ y $BI = B$
- $\lambda(AB) = (\lambda A)B = A(\lambda B)$

- La traspuesta de una matriz A de $n \times m$, $A = (a_{ij})$ es una matriz A^t, donde cada i-ésima columna de A^t es la misma que el i-ésimo renglón de A, es decir $A^t = (a_{ji})$, por ejemplo:

$$A = \begin{bmatrix} 7 & 2 & 0 \\ 3 & 5 & -1 \\ 0 & 5 & -6 \end{bmatrix}, A^t = \begin{bmatrix} 7 & 3 & 0 \\ 2 & 5 & 5 \\ 0 & -1 & -6 \end{bmatrix}$$

$$B = \begin{bmatrix} 2 & 5 & 7 \\ 3 & -5 & -1 \end{bmatrix}, B^t = \begin{bmatrix} 2 & 3 \\ 5 & -5 \\ 7 & -1 \end{bmatrix}$$

Las propiedades de la traspuesta de una matriz son las siguientes:

- $(A^t)^t = A$
- $(A + B)^t = A^t + B^t$
- $(AB)^t = B^t A^t$
- Si A^{-1} existe, $\left(A^{-1}\right)^t = (A^t)^{-1}$

3.3.1 Inversa de una matriz

Podemos considerar el sistema lineal

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n &= b_2 \\ \cdots\cdots\cdots\cdots &= \vdots \\ a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n &= b_n \end{aligned}$$

como la ecuación matricial

$$Ax = b$$

donde

$$A = \begin{bmatrix} a_{11}\ a_{12}\ ...\ a_{1n} \\ a_{21}\ a_{22}\ ...\ a_{2n} \\ \vdots \\ a_{n1}\ a_{n2}\ ...\ a_{nn} \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

La inversa de una matriz está relacionada con los sistemas lineales.

Se dice que una matriz A de $n \times n$ es no singular, se existe una matriz A^{-1} de $n \times n$ tal que $AA^{-1} = A^{-1}A = I$. A la matriz A^{-1} se le denomina la inversa de A. A una matriz que no tiene inversa se le da el nombre de singular.

Las siguientes propiedades son aplicables a una matriz no singular

- A^{-1} es única.
- $\left(A^{-1}\right)^{-1} = A$.
- Si B también es no singular de $n \times n$, entonces

$$(AB)^{-1} = B^{-1}A^{-1}$$

Si tenemos la inversa de A, es fácil resolver un sistema lineal de la forma $Ax = b$ ya que

$$\begin{aligned} A^{-1}Ax &= A^{-1}b \\ Ix &= A^{-1}b \\ x &= A^{-1}b \end{aligned}$$

Supongamos que tenemos el siguiente ejemplo

$$\begin{aligned} x_1 + 2x_2 - x_3 &= 2 \\ 2x_1 + x_2 &= 3 \\ -x_1 + x_2 + 2x_3 &= 4 \end{aligned} \tag{4}$$

tenemos que esto puede ser escrito como

$$\begin{bmatrix} 1 & 2 & -1 \\ 2 & 1 & 0 \\ -1 & 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix}$$

Si multiplicamos ambos lados por la inversa de A

$$\begin{bmatrix} -\frac{2}{9} & \frac{5}{9} & -\frac{1}{9} \\ \frac{4}{9} & -\frac{1}{9} & \frac{2}{9} \\ -\frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix} \begin{bmatrix} 1 & 2 & -1 \\ 2 & 1 & 0 \\ -1 & 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -\frac{2}{9} & \frac{5}{9} & -\frac{1}{9} \\ \frac{4}{9} & -\frac{1}{9} & \frac{2}{9} \\ -\frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix} \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \frac{7}{9} \\ \frac{13}{9} \\ \frac{5}{3} \end{bmatrix}$$

A fin de desarrollar un método para calcular A^{-1}, suponiendo su existencia, consideremos la multiplicación de matrices. Sea B_j la *j-ésima* columna de la matriz B de $n \times n$.

$$B_j = \begin{bmatrix} b_{1j} \\ b_{2j} \\ \vdots \\ b_{nj} \end{bmatrix}$$

si $AB = C$, entonces la j-ésima columna de C está dada por

$$\begin{bmatrix} c_{1j} \\ c_{2j} \\ \vdots \\ c_{nj} \end{bmatrix} = C_j = AB_j = \begin{bmatrix} a_{11}\ a_{12}\ ...\ a_{1n} \\ a_{21}\ a_{22}\ ...\ a_{2n} \\ \vdots \\ a_{n1}\ a_{n2}\ ...\ a_{nn} \end{bmatrix} \begin{bmatrix} b_{1j} \\ b_{2j} \\ \vdots \\ b_{nj} \end{bmatrix} = \begin{bmatrix} \sum_{k=1}^{n} a_{1k}b_{kj} \\ \sum_{k=1}^{n} a_{2k}b_{kj} \\ \vdots \\ \sum_{k=1}^{n} a_{nk}b_{kj} \end{bmatrix}$$

supongamos que A^{-1} existe y que $A = B = (b_{ij})$, entonces si $AB = I$ tenemos que

$$AB_j = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \text{ donde el valor de 1 aparece en el } j\text{-}\acute{e}simo \text{ renglón}$$

de tal forma que para encontrar B se pueden resolver n sistemas lineales (por ejemplo utilizando eliminación Gaussiana) donde la *j-ésima* columna de la inversa es la solución del sistema lineal que en el lado derecho tiene la *j-ésima* columna de I.

Exercise 15 *Utilice el programa que realiza el método de eliminación Gaussiana para determinar la inversa de la matriz del sistema dado en* (4).

Exercise 16 *La alacena de ingredientes mágicos de una bruja contiene 10 onzas de trébol de cuatro hojas molido y 14 onzas de raíz de mandrágora en polvo. La alacena se resurte automáticamente siempre que ella use justo todo lo que tiene. Una porción de amor requiere* $3\frac{1}{13}$ *onzas de trébol y* $2\frac{2}{13}$ *onzas de mandrágora. Una receta de una bruja muy conocida para curar el resfriado común requiere* $5\frac{5}{13}$ *onzas de trébol y* $10\frac{10}{13}$ *onzas de mandrágora.¿Qué cantidad de la poción de amor y del remedio para el resfriado debe hacer la bruja para usar toda la reserva de la alacena?*

3.3.2 Factorización de matrices

En esta sección veremos que los pasos que se siguieron para resolver un sistema de la forma $Ax = b$, también también pueden servir para factorizar una matriz en un producto matricial. La factorización es muy útil cuando se presenta de la forma $A = LU$, donde L es una triangular inferior y U es triangular superior. No todas las matrices pueden ser factorizadas de este modo, pero es posible hacerlo con un gran número de las que se presentan con frecuencia en las aplicaciones.

Si en el sistema $Ax = b$, se puede factorizar $A = LU$, entonces podemos encontrar x más fácilmente empleando un proceso de dos pasos. Primero, se sustituye $Ux = y$ y se resuelve el sistema $Ly = b$. Una vez que se conocen los valores de y se usan para encontrar los valores de x al resolver el sistema $Ux = y$. Debido a que la matriz L es triangular superior, los valores de y son fácilmente encontrados al hacer una sustitución hacia adelante; de igual forma, para encontrar los valores de x, una vez obtenido los de y, se obtienen realizando una sustitución hacia atrás ya que la matriz U es triangular superior.

Pos supuesto existe un costo computacional al factorrizar la matriz A, pero una vez determinada esta factorización, se pueden resolver de forma más simplificada cualquier sistema que contenga la matriz A.

Para poder realizar la factorización LU, supongamos primero que se puede realizar dicha factorización, por lo tanto tenemos que

$$
\begin{aligned}
A &= \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix} = LU \\
&= \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ l_{21} & 1 & 0 & \dots & 0 \\ l_{31} & l_{32} & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ l_{n1} & l_{n2} & \dots & \dots & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} & \dots & u_{1n} \\ 0 & u_{21} & u_{23} & \dots & u_{2n} \\ 0 & 0 & u_{33} & \dots & u_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & \dots & u_{nn} \end{bmatrix}
\end{aligned}
$$

se podría haber escogido también que, en vez de que la diagonal de la matriz L tuviera unos la tuviera la matriz U.

Para calcular los elementos de L y U, observemos lo siguiente: el primer renglón de U y la primera columna de L son fácilmente calculados ya que

$$\begin{aligned} L_1U &= \begin{bmatrix} u_{11} & u_{12} & u_{13} & ... & u_{1n} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & ... & a_{1n} \end{bmatrix} \\ u_{1k} &= a_{1k}, \textit{ para } k = 1, ...n \end{aligned}$$

la primera columna de L, se pude calcular

$$\begin{aligned} L_k u_{11} &= l_{k1} u_{11=} a_{k1}, \textit{ para } k = 2, ..., n \\ l_{k1} &= \frac{a_{k1}}{u_{11}}, \textit{ para } k = 2, ..., n \end{aligned}$$

observe que si $u_{11} = 0$ no se puede realizar la factorización.

Para los renglones y columnas de $2, .., n$, se va utilizando los elementos ya calculados, como se describe a continuación: para el i-ésimo renglón de U

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj}$$

para el i-ésima columna de L

$$l_{ij} = \frac{1}{u_{ii}} \left[a_{ji} - \sum_{k=1}^{i-1} l_{jk} u_{kj} \right]$$

Si al final, se tiene que $u_{nn} = 0$, se puede factorizar $A = LU$ pero A es singular.

El siguiente algoritmo factoriza la matriz A de $n \times n = (a_{ij})$ en el producto de una matriz triangular inferior $L = (l_{ij})$ y en la matriz triangular superior $U = (u_{ij})$, donde la diagonal principal de L o U son unos.

Algoritmo 17 *Factorización LU*

Entrada: la dimensión n; los elementos a_{ij}, $1 \leq i, j \leq n$ *de A, la diagonal* $l_{11} = ... = l_{nn} = 1$ *de L o la diagonal* $u_{11} = ... = u_{nn} = 1$ *de U.*

1. *Seleccione* $l_{11} = a_{11}$ *o* $u_{11} = a_{11}$*, según haya decidido si L o U contiene en su diagonal principal unos.*
   (a) *Si* $l_{11} u_{11} = 0$ *PARE (factorización imposible)*
2. *Para* $j = 2..n$
   (a) $u_{1j} = a_{1j}/l_{11}$; *(primer renglón de U)*
   (b) $l_{j1} = a_{j1}/u_{11}$; *(primera columna de L)*
3. *Para* $i = 2, ..., n-1$
   (a) *Seleccione* $l_{ii} = a_{ii} - \sum_{k=1}^{i-1} l_{ik} u_{ki}$ *o* $u_{ii} = a_{ii} - \sum_{k=1}^{i-1} l_{ik} u_{ki}$*, según haya decidido si L o U contiene en su diagonal principal unos.*

(b) *Si* $l_{ii}u_{ii} = 0$ *PARE (factorización imposible).*

(c) *Para* $j = i + 1, ..., n$

$$u_{ij} = \frac{1}{l_{ii}}\left[a_{ij} - \sum_{k=1}^{i-1} l_{ik}u_{kj}\right]; \text{ (i-ésimo renglón de U)}$$

$$l_{ji} = \frac{1}{u_{ii}}\left[a_{ji} - \sum_{k=1}^{i-1} l_{jk}u_{ki}\right]; \text{ (j-ésima columna de L)}$$

4. *Seleccione* $l_{nn} = a_{nn} - \sum_{k=1}^{n-1} l_{nk}u_{kn}$ *o* $u_{nn} = a_{nn} - \sum_{k=1}^{n-1} l_{nk}u_{kn}$, *según haya decidido si L o U contiene en su diagonal principal unos (nota: si* $l_{nn}\ u_{nn} = 0$, *entonces* $A = LU$ *pero A es singular).*

Una vez terminada la factorización, la solución de un sistema lineal de la forma $Ax = b$ se obtiene haciendo $y = Ux$ y luego determinando y a partir de las ecuaciones

$$y_1 = \frac{b_1}{l_{11}}$$

y, para $i = 2, 3, ..., n$

$$y_i = \frac{1}{l_{ii}}\left[b_i - \sum_{j=1}^{i-1} l_{ij}y_j\right]$$

Una vez que se tiene y, se resuelve el sistema triangular superior $Ux = y$ para x por medio de la sustitución hacia atrás.

Exercise 18 *Implemente el algoritmo para factorizar la matriz* $A = LU$ *y que resuelva el sistema* $Ax = b$ *utilizando el método anteriormente explicado.*

3.4 Métodos iterativos en el álgebra lineal

Referencias

[1] Richard L. Burded. Análisis Numérico (6a. edición), International Thomson Editors.

[2] Stanley I. Grossman. Álgebra lineal, quinta edición, Mc. Graw-Hill.