Laboratorio 1 MA-33A 2007:

Comandos Básicos Matlab y Algebra Lineal Computacional

Gonzalo Hernández - Gonzalo Rios

UChile - Departamento de Ingeniería Matemática

1 Primeros Comandos (10 min)

En esta sesión aprenderemos los comandos básicos de Matlab, aplicados a resolver Sistemas de Ecuaciones Lineales de forma precisa. Para esto, nos concentraremos en el sistema:

$$\begin{bmatrix} 100 & 1 & 10 \\ 101 & 1 & 9 \\ 102 & 1 & 9 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 10 \\ 1 \\ 0 \end{bmatrix}$$

Cuya solución exacta es:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -1 \\ 30 \\ 8 \end{bmatrix}$$

Veamos como ingresamos este sistema a Matlab:

1.1 Creación de un vector

b=[a_1 a_2 ... a_n]; //b es un vector fila
b=[a_1; a_2; ... ; a_n]; //b es un vector columna

En nuestro ejemplo:

```
>> b=[10;1;0]

b =

    10
     1
     0
```

O de forma equivalente:

```
>> b=[10 1 0]

b =

    10     1     0

>> b=b'
```

```
b =

    10
     1
     0
```

Como pudo apreciar, el comando "'" significa transposición.

1.2 Creación de una matriz

A=[f_1 ; f_2 ; f_3 ;...; f_n]; //donde los f_k son filas de la forma $f_k = a_1\ a_2 ...\ a_n$

En nuestro ejemplo:

```
>> A=[100 1 10; 101 1 9; 102 1 9]

A =

   100     1    10
   101     1     9
   102     1     9
```

O de forma equivalente:

```
>> A=[100 1 10
101 1 9
102 1 9
]

A =

   100     1    10
   101     1     9
   102     1     9
```

Existen muchas matrices predefinidas en Matlab, como por ejemplo (n es un natural, v es un vector):

```
>> pascal(n)
>> hilb(n)
>> magic(n)
>> vander(v)
>> compan(v)
```

Se invita al alumno a investigar que significa cada uno de estos comandos. Mas información se puede obtener del excelente "Help" de Matlab.

2 Comandos de Álgebra Lineal (25 min)

ACT1: Ud. deberá probar en Matlab los siguientes comandos y describir brevemente en el informe el resultado que obtiene. Considere i,n,m$\in \mathbb{N}$.

1) A'
2) inv(A)
3) A^n
4) A.^n
5) I=eye(n)
6) Z=zeros(n)
7) D1=diag(b)
8) D2=diag(A)
9) U=triu(A)
10) L=tril(A)
11) R=rand(n,m)
12) A(n,m)
13) A(:,m)
14) A(n,:)
15) A(i,n:m)
16) R*A
17) A.*U
18) A*n
19) A./D2
20) L+U
21) A+n
22) [A;L]
23) det(A)
24) x1=A\b
25) x2=inv(A)*b

3 Creación de una Función .m (15 min)

A continuación programaremos paso a paso el algoritmo de Gauss en Matlab:

3.1 Para crear un archivo .m

File→New→M-File

3.2 Encabezado de una función

function x = gauss(A,b)

// Mediante este comando se nombra la función como "gauss.m", que recibe como parámetros A y b, y devuelve x.

3.3 Método

Los comandos que ejecuta la función "gauss.m" son los siguientes:

n = length(b);

// Se guarda en la variable n el tamaño del vector b

for k = 1:(n-1)

// El ciclo for comienza en k=1, en cada iteración se suma 1 a k, y termina cuando k=n-1, incluyendo esa iteración

for i = k+1:n

// Comienza un ciclo for anillado al anterior entre i=k+1 y i=n

lambda = A(i,k)/A(k,k);

// En la variable lambda se guarda el valor de la división sin modificar la matriz A

A(i,k+1:n) = A(i,k+1:n) - lambda*A(k,k+1:n);

// Para ahorrarnos un ciclo, se toma de la fila i los elementos desde la columna k+1 hasta la n

b(i)= b(i) - lambda*b(k);

// Se hace el pivoteo en el vector b

end

// Finaliza el ciclo del for de variable i

end

// Finaliza el ciclo del for de variable k

for k = n:-1:1

// Comienza el ciclo de la sustitución en reversa, inicializando la variable k=n, en cada iteración se le suma -1 a k, y termina cuando k=1

x(k) = (b(k) - A(k,k+1:n)*b(k+1:n))/A(k,k);

// De forma similar, se ahorra un ciclo multiplicando la fila A(k,k+1:n) por la columna b(k+1:n)

end

// Finaliza el ciclo del for de variable k

end

// Finaliza la función "gauss.m"

3.4 Guardar la función

File→Save as

// Se escoge la carpeta donde guardaremos la función y el nombre que tendrá. Guardela en la carpeta "C:\Laboratorio-MA-33A", bajo el nombre de "gauss.m"

4 Utilización de funciones (10 min)

4.1 Current Directory

Para poder utilizar funciones que hemos creado, estas se deben visualizar en la ventana "Current Directory". Por ejemplo, si queremos utilizar la funcion "gauss.m", se debe cambiar el "Current Directory" a "C:\Laboratorio-MA-33A".

4.2 Llamar funciones

Dado que ya ingresamos las variables A y b, entonces podemos utilizar la funcion "gauss.m" de la siguiente forma:

```
>> x3=gauss(A,b)
```

Queda entonces guardada en la variable x3 el valor entregado por la funcion gauss.

ACT2: Escriba en el informe la solución dada por la función. ¿Es exacta la solución? Explique.

4.3 Llamar funciones desde otras funciones

Se puede programar una función que utiliza otra función, siempre y cuando este en el "Current Directory". Por ejemplo:

```
function [x,n] = gauss2(A,b)
n = length(b);
x=gauss(A,b);
end
```

ACT3: Escriba en el informe que es lo que hace esta función y como debe ser llamada.

5 Comparación de métodos (45 min)

Matlab incorpora varios métodos para resolver SEL. Incluyendo el que programamos, contamos con 3 métodos:

1) x1=A\b
2) x2=inv(A)*b
3) x3=gauss(A,b)

Veamos tres formas de comparar métodos.

5.1 Exactitud

En SEL, el error de la solución aproximada $\tilde{x}$, en comparación con la solución exacta x, se define de dos formas:

i) Error Absoluto: $E_{abs} = \|\tilde{x} - x\|$

ii) Error Relativo: $E_{rel} = \dfrac{\|\tilde{x} - x\|}{\|x\|}$

En donde la norma $\|\cdot\|$ se considerará como la norma infinito:

$$\|C\| = \max_{i=1\ldots m} \sum_{j=1}^{n} |c_{ij}|$$

En Matlab, esta norma se escribe como:

```
>> norm(A,inf)

ans =

   112
```

Ahora bien, como no siempre se sabe a priori a solución exacta x, estos errores se pueden expresar como:

i) Error Absoluto: $E_{abs} = \|A\tilde{x} - b\|$

ii) Error Relativo: $E_{rel} = \dfrac{\|A\tilde{x} - b\|}{\|b\|}$

ACT4: Programe una función en Matlab, llamada "errores.m", que reciba una matriz A y un vector b, y entrege el error absoluto y el error relativo de los 3 métodos para resolver un SEL. Escriba esta función en el informe, los resultados de la prueba con el sistema (A, b), y explique cual método es más exacto.

5.2 Robustez

Los SEL no siempre son exactos a los datos reales, por lo que se tienen pequeñas fluctuaciones en sus valores. Por ejemplo, consideremos el sistema pertubado de (A, b):

$$\begin{bmatrix} 100 & 1 & 10 \\ 101 & 1.01 & 9 \\ 102 & 1 & 9 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 10 \\ 1 \\ 0 \end{bmatrix} \Longrightarrow \begin{bmatrix} x \\ y \\ z \end{bmatrix} \approx \begin{bmatrix} -0.863\,636\,363\,636\,363\,636\,36 \\ 13.\,636\,363\,636\,363\,636\,364 \\ 8.\,272\,727\,272\,727\,272\,727\,3 \end{bmatrix}$$

En este caso, un pequeño cambio en el coeficiente a_{22} genera un cambio significativo en la solución del SEL.

ACT5: Resuelva el SEL pertubado con los 3 métodos conocidos. ¿Qué puede observar? Explique.

5.3 Tiempo

En Matlab, la instrucción cputime retorna el tiempo total en segundos que Matlab a usado la CPU. Por ejemplo:

```
>> t=cputime;gauss(A,b);e=cputime-t

e =

     0
```

Como el sistema es sólo de 3 por 3, el tiempo necesario para resolverlo es despreciable. Pero si resolvemos un sistema de mayor dimensión, por ejemplo, de 100 por 100, el tiempo será considerable.

ACT6: Programe una función en Matlab, llamada "tiempo.m", que reciba una matriz M y un vector N, y entregue el tiempo de ejecución de cada uno de los métodos para resolver un SEL. Haga una prueba con un sistema de 100 por 100, utilizando los comandos "M=rand(100)" y "N=rand(100,1)". Escriba en el informe esta función, y el resultado de la prueba.

5.4 Comparación usando tablas

En Matlab, un cell array es una matriz en donde sus elementos pueden ser constantes, enteros, reales, matrices, cadenas de caracteres, etc. Por ejemplo:

```
>> C={'','cadena','hola', 'mundo';'numeros', [3],[55],[5];'error',[3.764],[10],[pi]}

C =

          ''    'cadena'              'hola'    'mundo'
    'numeros'    [               3]    [  55]    [               5]
    'error'      [3.76400000000000]    [  10]    [3.14159265358979]
```

ACT7: Utilizando las funciones "errores.m" y "tiempo.m", programe una funcion llamada "tabla.m" que reciba una matriz A y un vector b, y devuelva un cell array con el error relativo y el tiempo de ejecución de los tres métodos. Escriba en el informe esta función, y el resultado de la prueba con "M=rand(100)" y "N=rand(100,1)".