Guión de prácticas de los Temas 3 y 4 de Cálculo Numérico

José Luis Bravo Trinidad, Pedro Martín Jiménez

Preliminares

Antes de comenzar la práctica, descárgate los archivos de ejemplo de Avuex. Después, descomprime el fichero en un directorio y establece ese directorio como directorio de trabajo de Matlab.

3 Sistemas Lineales

3.1 Algebra de matrices

Recordemos cómo maneja Matlab las matrices. Define la siguiente matriz en Matlab:

```
A=[3,2,1
   4,6,5
   7,8,9]
```

Para calcular la matriz traspuesta, la inversa y el determinante:

```
A' % Matriz traspuesta
inv(A) % Matriz inversa
det(A) % Determinante
```

Comprueba que el producto de matrices no es conmutativo calculando:

```
(A')*A
A*(A')
```

Para sustituir una fila/columna por un vector dado, podemos hacerlo así:

```
v=[1,1,1] % Vector por el que queremos sustituir la tercera fila/columna
B=A; % Copiamos A para no modificarla
B(3,:)=v % Cambiamos la tercera fila
B=A; % Recuperamos A
B(:,3)=v' % Cambiamos la tercera columna
```

Para intercambiar dos filas:

```
B=A;
B([1,3],:)=B([3,1],:) % Intercambiamos la primera y la tercera fila.
```

3.2 Resolución de sistemas de ecuciones lineales: método de Gauss

La función *soldet.m* calcula la solución de un sistema a partir de los determinantes. Pruébala con el sistema cuya matriz completada es Ac (en *ejemplos.m*) y con

$$\begin{cases} x_1 - x_2 = 3 \\ 2x_1 + x_2 - x_3 + x_4 = 4 \\ 3x_1 + x_3 - x_4 = 5 \\ 4x_1 + x_4 - x_5 = 12 \end{cases}$$

Ejercicio 3.1. ¿Qué ocurre? ¿Por qué sucede esto?

La función *pivote.m* recibe la matriz ampliada de un sistema y devuelve un sistema equivalente pero de matriz triangular superior, mediante el método de Gauss con pivoteo parcial. La función *soltri.m* recibe la matriz ampliada de un sistema con matriz A triangular superior y lo resuelve.

Ejercicio 3.2. Calcula la solución de los sistemas anteriores usando Gauss con pivoteo parcial. ¿Es la misma? ¿Cómo comprobarías si un vector dado es solución del sistema?

Ejercicio 3.3. Podemos comparar el tiempo que tardan los distintos métodos directos, ejecutando en Matlab la siguiente secuencia de instrucciones.

```
clear all
n=20;
for i=1:n
  tam=i*10;
  Ac=rand(tam,tam+1); % Genera un s.l. aleatorio con i incógnitas
  tic;soldet(Ac);u(i)=toc; % Tiempo mediante los determinantes
  tic;soltri(pivote(Ac));v(i)=toc; % Tiempo con Gauss con pivote
end
plot(1:n,u,1:n,v); % Representamos ambos tiempos
```

¿Para qué tamaños de sistemas es más rápido el Gauss con pivote? (Para valientes) Calcula el orden de cada uno de los métodos.

Ejercicio 3.4. (Para valientes) Matlab y octave usan para calcular un determinante métodos mucho más eficientes que su cálculo directo. Crea una función que calcule el determinante de una matriz diréctamente y comprueba cómo afecta ésto al rendimiento.

Ejercicio 3.5. (Para valientes) Codifica el método de Gauss con pivoteo total.

3.3 Factorización LU. Factorización de Choleski

Matlab tiene implementada la factorización de Choleski en el comando *lu*. Para usarla:

```
[l,u,p]=lu(A);
```

Devolverá en l una matriz triangular inferior y en u una matriz superior, de modo que

$$l * u = p * A$$

Ejercicio 3.6. Resolver el sistema de *ejemplos.m* utilizando la factorización lu. ¿Qué secuencias de comandos habría que ejecutar?.

Nota: Puedes no tener en cuenta la matriz p (la solución será la correcta, aunque estén desordenadas las variables). Los valientes no deberían olvidarse de esa matriz.

3.4 Normas y análisis del error

En matlab la norma de una matriz se calcula con la función *norm*. Escribe "help norm" para obtener información de la sintaxis. La inversa de una matriz A se puede calcuar con $inv(A)$.

Ejercicio 3.7. Calcula el condicionamiento de la matriz del sistema de "ejemplos.m".

Acota el error relativo de las posibles soluciones en función del error en los datos.

Ejercicio 3.8. Supongamos que al medir los datos se ha cometido un error del 10% y los datos obtenidos son $(3.2, 3.9, 5.5, 5.6, 7.7)$. ¿Cómo afecta esto a las soluciones?

Ejercicio 3.9. Considera el sistema:

$$\begin{cases} x + 2y = 2.2 \\ 2x + 3y = 3.2 \end{cases}$$

¿Cuáles son sus soluciones? ¿Y si los términos independientes fueran 2.0 y 3.4 (es decir, un error del 10%?

3.5 Métodos iterativos

Vamos a aplicar el método de Gauss-Seidel al ejemplo visto en clase.

```
A=[6,2,0
   0,5,1
   2,0,3]
b=[0,1,0]
n=length(A)
L=eyel(n).*A
D=eye(n).*A
U=eyeu(n).*A
x=zeros(1,n)
x=soltril(D+L,(-U*x')'+b)
```

Para dar más pasos del método, basta repetir la última línea.

Ejercicio 3.10. Crea una función que reciba una matriz A, un vector b, un número de pasos m y aplique m pasos del método de Gauss-Seidel para la ecuación $Ax = b$, partiendo del vector inicial $x^0 = (0, 0, \ldots, 0)$.

4 Aproximación de funciones

4.1 Construcción del polinomio interpolador

Consideremos la tabla:

x	-1	0	1	2
y	-2	-2	0	4

Para obtener el polinomio interpolador, tenemos que obtener la matriz completada del sistema:

```
x=[-1,0,1,2]
n=length(x);
A=(ones(n,1)*x)' % matriz cuadrada tal que cada columna es el vector x
E=ones(n,1)*(0:n-1) % matriz de los exponentes a los que hay que elevar
A=A.^E % matriz del sistema
y=[-2,-2,0,4]
Ac=[A,y']
```

Ahora Ac es la matriz que hay que resolver para obtener los coeficientes.

Ejercicio 4.1. Obtén los coeficientes del polinomio interpolador de la tabla.

Ejercicio 4.2. Crea una función que reciba dos vectores, uno correspondiente a las coordenadas x y otro a las y y devuelva los coeficientes del polinomio interpolador.

Esta función está ya implementada en matlab como *polyfit*. Sin embargo esta función devuelve los coeficientes al revés (el primero es el de mayor grado). Para invertir los coeficientes de un vector v, se puede usar $v(length(v) : -1 : 1)$.

Ejercicio 4.3. Usa la función *polinomio.m* para calcular los valores del polinomio en los puntos $-1, 0, 1, 2$. ¿Qué relación hay entre el polinomio interpolador y los puntos de la tabla? (polinomio(a,x) devuelve los valores en x de el polinomio de coeficientes a).

Ejercicio 4.4. (Para valientes) Dibuja el polinomio interpolador. Puedes usar la función *polinomio.m* o polyeval, con la misma sintaxis, pero con los coeficientes ordenados de mayor a menor grado.

Para dibujar funciones, ayúdate del siguiente ejemplo:

```
a=0;
b=2;
h=0.01;
plot(a:h:b,sin(a:h:b))
```

4.2 Error del polinomio interpolador

Ejercicio 4.5. Vamos a comprobar que si "aumenta n", el polinomio interpolador puede no "parecerse" a la función que se desea interpolar. Para ello, usaremos la función $f(x) = 1/(1 + x^2)$ y su polinomio interpolador en puntos equiespaciados en el intervalo $[-5, 5]$.

Tomemos $n = 10$. Hacemos:

```
n=10;
x10=-5:10/n:5;
y10=1./(1+x10.^2);
```

Obtén el polinomio interpolador para ese ejemplo y para $n = 20$. Calcula la diferencia entre el polinomio interpolador y la función en el punto $x = 4.95$ en los dos casos. ¿En cuál se comete un error mayor?

(Para valientes) Represénta los polinomios interpoladores para $n = 10$ y $n = 20$ en los puntos $t = -5 : 0.0012 : 5$. Puedes pintar dos funciones f y g en la misma gráfica del siguiente modo:

```
t=-5:0.012:5;
plot(t,f(t),t,g(t));
```

¿Cuál de ellos tiene menos error absoluto con $f(t) = 1/(1 + x^2)$?

4.3 Métodos de Lagrange y Newton

Las funciones *lagran.m* y *newpoly.m* calculan los polinomios de interpolación utilizando los métodos de Lagrange y Newton, respectivamente. Devuelven los coeficientes comenzando por los de mayor grado.

Para implementar los métodos, se usan las funciones *poly* y *conv*. La función *poly(a)* devuelve los coeficientes del polinomio $x - a$ y la función *conv(P,Q)* calcula el producto de los polinomios de coeficientes P y Q.

Ejercicio 4.6. Calcula el valor en $x = 2$ del polinomio interpolador de

```
x = [0.0 0.4 0.8 1.2]
y = cos(x)
```

con el método de Lagrange y con el de Newton.

Ejercicio 4.7. (Para valientes) Dibuja el polinomio interpolador del ejercicio anterior y la función coseno en una gráfica.