finalG15-16resuelto.pdf

Kryzux

Laboratorio de Computación Científica

1º Grado en Física

Facultad de Ciencias Físicas
Universidad Complutense de Madrid

EXAMEN DE LABORATORIO DE COMPUTACIÓN CIENTÍFICA 2 Febrero 2015 **Grupo G1-EX1**

ALUMNO/DNI:_____

1. (3 puntos) El comportamiento de un sistema físico está gobernado por un sistema Ax=b de 10 ecuaciones con 10 incógnitas, con las siguientes características:

- Los elementos diagonales de la matriz del sistema A están comprendidos entre 1 y 3
- Los elementos no diagonales de la matriz A están comprendidos entre -1 y 1
- El vector de términos independientes b está comprendido entre 0 y 1

(a) Programa una función que genere matrices A y vectores b (ambos aleatorios) de estas características

El siguiente código define matrices aleatorias con las características requeridas

```
A=2*rand(10)-1;                          %Matriz aleatoria entre -1 y 1
L=A-triu(A);   U=A-tril(A);              %Submatrices triangulares
D=diag(1+2*rand(10,1));                  %Diagonal aleatoria entre 1 y 3
A=L+D+U;
b=rand(10,1);                            %Vector de términos independientes
```

(b) Estima la probabilidad de que los métodos de Jacobi y Gauss-Seidel (sin amortiguar) converjan para matrices de estas características, calculando el radio espectral para 1000 realizaciones distintas de A y b.

Incorporamos el código anterior a un bucle con 1000 realizaciones, en el que también añadimos instrucciones para el cálculo del radio espectral. Ejecutando este código podemos ver la probabilidad de que ambos métodos converjan:

```
conjac=0; congauss=0;
for i=1:1000
    A=2*rand(10)-1;                      %Matriz aleatoria entre -1 y 1
    L=A-triu(A);   U=A-tril(A);          %Submatrices triangulares
    D=diag(1+2*rand(10,1));              %Diagonal aleatoria entre 1 y 3
    A=L+D+U;
    b=rand(10,1);                        %Vector de términos independientes
    HJAC=-inv(D)*(L+U);                  %Matriz de convergencia de Jacobi
    HGAUSS=-inv(D+L)*U;            %Matriz de convergencia de Gauss-Seidel
    rjac=max(abs(eig(HJAC)));
    rgauss=max(abs(eig(HGAUSS)));
    if rjac<1    conjac=conjac+1; end
    if rgauss<1  congauss=congauss+1; end
 end
fprintf('Probabilidad Jacobi: %f\n',conjac/1000);
fprintf('Probabilidad Gauss-Seidel: %f\n',congauss/1000);
```

Ejecutando este script obtenemos las siguientes probabilidades:

>> testarconvergencia

Probabilidad Jacobi: 0.542000

Probabilidad Gauss-Seidel: 0.649000

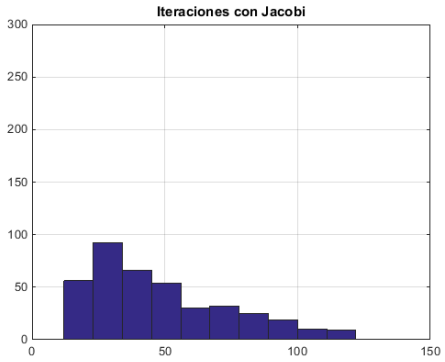
(c) Dibujar dos histogramas con la distribución del número de iteraciones requerido para la convergencia con cada uno de estos métodos con solución inicial cero y tolerancia 0.001. Para evitar sistemas con

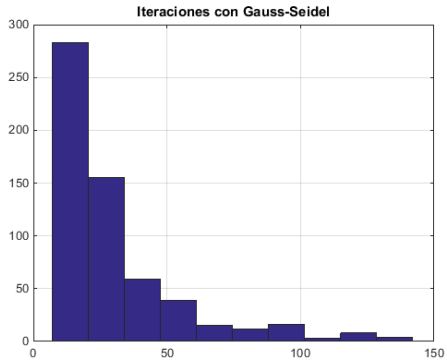
convergencia lenta, considera sólo aquellos casos en los que el radio espectral sea menor que 0.95. A partir de los resultados obtenidos, ¿qué método te parece más apropiado para este tipo de matrices?

Cuando el radio espectral es menor que 0.95 llamamos en el bucle a nuestras funciones de Jacobi y Gauss-Seidel y almacenamos en un vector el número de iteraciones requerido.

```
conjac=0; congauss=0;
itjac=[]; itgauss=[];
for i=1:1000
    A=2*rand(10)-1;                    %Matriz aleatoria entre -1 y 1
    L=A-triu(A);   U=A-tril(A);        %Submatrices triangulares
    D=diag(1+2*rand(10,1));            %Diagonal aleatoria entre 1 y 3
    A=L+D+U;
    b=rand(10,1);                      %Vector de términos independientes
    HJAC=-inv(D)*(L+U);                %Matriz de convergencia de Jacobi
    HGAUSS=-inv(D+L)*U;           %Matriz de convergencia de Gauss-Seidel
    rjac=max(abs(eig(HJAC)));
    rgauss=max(abs(eig(HGAUSS)));
    if rjac<1    conjac=conjac+1; end
    if rgauss<1  congauss=congauss+1; end
    if rjac<0.95
        [x,it]=jacobi(A,b,zeros(size(b)),0.001);
        itjac=[itjac it];
    end
    if rgauss<0.95
        [x,it]=gseidel(A,b,zeros(size(b)),0.001);
        itgauss=[itgauss it];
    end
end
fprintf('Probabilidad Jacobi: %f\n',conjac/1000);
fprintf('Probabilidad Gauss-Seidel: %f\n',congauss/1000);
```

Tras ejecutar el bucle podemos dibujar ambos histogramas. El método de Gauss-Seidel es claramente superior. No sólo tiene mayor probabilidad de convergencia, sino que requiere en general un número menor de iteraciones (para poder comparar las figuras, hemos tomado los mismos límites en ambas)





Se adjuntan también las funciones de Jacobi y Gauss-Seidel utilizadas:

```
function [x,it,rad]=jacobi(A,b,x0,tol)
D=diag(diag(A)); L=tril(A)-D; U=triu(A)-D;
error=2*tol; it=0;
while error>tol
```

QUE LO DIFÍCIL SEA ELEGIR EL COCHE.
HAZLO FÁCIL

TEÓRICAS ONLINE EN DIRECTO Y PRESENCIALES
EXÁMEN EN ALCALA DE HENARES = MÁS FÁCIL
LA AUTOESCUELA MÁS RECOMENDADA POR SUS CLIENTES

Autoescuelas 2000

autoescuela2000.com

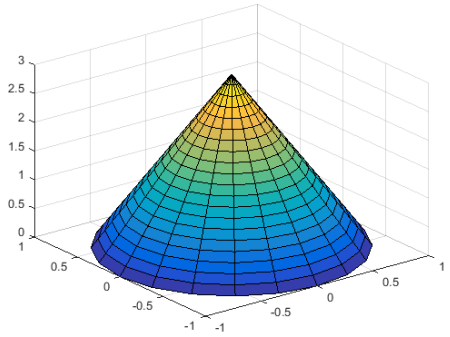
```
    x=inv(D)*(b-(U+L)*x0);
    error=norm(x-x0);
    x0=x; it=it+1;
end

function [x,it,rad]=gseidel(A,b,x0,tol)
D=diag(diag(A)); L=tril(A)-D; U=triu(A)-D;
error=2*tol; it=0;
while error>tol
    x=inv(D+L)*(b-U*x0);
    error=norm(x-x0);
    x0=x; it=it+1;
end
```

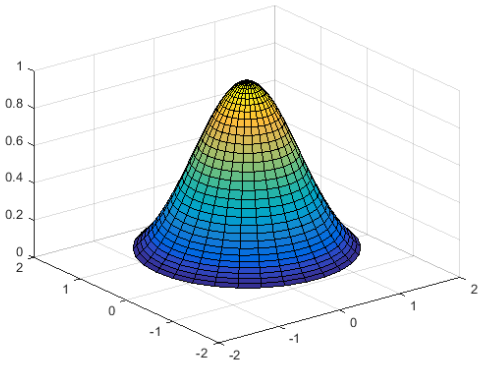
2. (2 puntos) Programar una función que reciba como argumentos de entrada una cadena de caracteres que almacena una expresión matemática y un vector, y dibuje la figura de revolución que genera la curva indicada por la ecuación, para los valores del radio indicados en el vector.

Ejemplos:

revolucion('3*(1-x)',[0:0.05:1]); %produce un cono girando la recta z=3*(1-x) alrededor del eje z

revolucion('exp(-x.^2)',[0:0.05:1.5]); %produce un pico gaussiano girando la curva z=exp(-x^2)





La siguiente función produce las figuras indicadas. La función crea un vector de ángulos equiespaciado entre 0 y 2π, y una malla en coordenadas polares usando meshgrid. Posteriormente se convierte a las coordenadas cartesianas usadas para dibujar. Debido a la simetría de revolución la altura de la superficie sólo depende de los valores de r.

```
function revolucion(formula,radvec)
theta=linspace(0,2*pi,length(radvec));  %Tantos puntos en theta como en r
[rg,thetag]=meshgrid(radvec,theta);
xg=rg.*cos(thetag);
yg=rg.*sin(thetag);
zg=feval(inline(formula),rg);  %Evaluar funcion
surf(xg,yg,zg);
```

3. (2.5 puntos) En un observatorio meteorológico se miden los valores máximos diarios de la temperatura y se almacenan en una matriz, cuya primera columna es el día del año (entre 1 y 365), y la

segunda la medida de la temperatura máxima observada. Cuando ocasionalmente no se tienen medidas (por ejemplo, debido a fallo de los instrumentos) se almacena para ese día el valor T=-100, que se usa como código de control para indicar falta de medida.
Al acabar el año, se requiere detectar los días en los que no se dispuso de medida, sustituyendo el código de control por una estimación obtenida interpolando los valores de días adyacentes. Programar una función *Mcorr*=**corregirdatos**(*M*) que a partir de una matriz *M* como la indicada (365x2):
a) Extraiga dos vectores dia y temp, con los valores de los días en los que existen medidas (es decir, eliminando los días en los que la temperatura registrada es T=-100)
b) Reconstruya la columna de temperatura de la matriz original a partir de los valores anteriores, usando interpolación por splines cúbicos.

```
function Mcorr=corregirdatos(M)
%Extraer medidas reales
dia=[]; temp=[];
for i=1:365
    if M(i,2)~=-100  %dato real
        dia=[dia M(i,1)];
        temp=[temp M(i,2)];
    end
end
%Construir matriz corregida
Mcorr(:,1)=M(:,1);  %la primera columna (días) es igual
Mcorr(:,2)=interp1(dia,temp,Mcorr(:,1),'splin');  %interpolar temperaturas
```

4. (**2.5 puntos)** La energía potencial de un sistema físico viene determinada por la siguiente expresión:

$$V(x) = 2 - 2sen^3(x) + sen(x) - \cos(x) \quad julios$$

siendo $x \epsilon [0, \pi]$ un parámetro que caracteriza al estado del sistema.
(a) Calcular los estados de equilibrio del sistema, indicando si se trata de equilibrios estables (mínimos relativos de la energía potencial) o inestables (máximos relativos). No es necesario que calcules la derivada usando cálculo simbólico, puedes derivar también a mano.
(b) Si el sistema parte del reposo desde su estado de máxima energía potencial, ¿cuál es la energía cinética máxima que puede alcanzar de forma conservativa (conservando la energía total)?

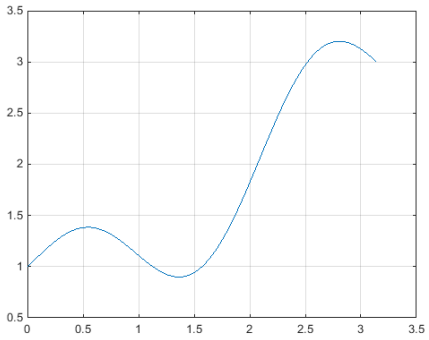
Por comodidad derivamos usando cálculo simbólico, aunque también podríamos hacerlo a mano:

```
>> syms x
>> V=2-2*sin(x)^3+sin(x)-cos(x)
V =
sin(x) - cos(x) - 2*sin(x)^3 + 2

>> Vprima=diff(V)
Vprima =
cos(x) - sin(x) + 6*sin(x)^3 - 12*cos(x)^2*sin(x)
```

Podemos crear funciones inline a partir de estas definiciones como sigue:

```
>> V=inline(V);  Vprima=inline(Vprima);
```

Alternativamente, podríamos haber definido ambas funciones a mano. En la figura adjunta, se muestra la energía potencial del sistema en el intervalo [0, π]:




```
>> x=linspace(0,pi);
>> plot(x,V(x))
```

A partir de esta figura, podemos ver que existen dos equilibrios inestables, para $x \approx 0.5$ y para $x \approx 2.7$, y un único punto de equilibrio estable para $x \approx 1.3$. Para hallar estos valores con más precisión, calculamos las raíces de la derivada usando la función fzero. Esto da:

```
>> x1=fzero(Vprima,0.5)
x1 =
   0.5434

>> x2=fzero(Vprima,1.3)
x2 =
   1.3631

>> x3=fzero(Vprima,2.7)
x3 =
   2.8059
```

La máxima energía cinética que puede adquirir el sistema corresponde a la energía potencial perdida al evolucionar desde su estado de máxima al de mínima energía, que calculamos como:

```
>> ecin=feval(V,x3)-feval(V,x2)
ecin =
   2.3036
```