Métodos iterativos para sistemas lineales: su implementación con Matlab

Mª Pilar de las Heras, José Luis Fernández
Dept. de Matemáticas y Computación
Area Matemática Aplicada
Universidad de Burgos
09006 Burgos
e-mail: pilarh@ubu.es jlfernan@ubu.es

1. Resumen

En la mayoría de los estudios de Informática se incluye una asignatura de Ampliación de Matemáticas, bien en el 2º cuatrimestre de 1º, bien en 2º curso.
En sus estudios de Informática de Gestión, la Universidad de Burgos, incluye una asignatura en 2º curso, llamada Laboratorio Computacional, que tiene 6 créditos, 3 teóricos y 3 prácticos. Esta asignatura es impartida por el área de Matemática Aplicada del Departamento de Matemáticas y Computación.
En ella, es interesante incluir un tema sobre los métodos numéricos del álgebra lineal.
Este trabajo se presenta con el objetivo de ser una posible práctica para impartir los créditos prácticos de parte de este tema.
Concretamente la parte del álgebra numérica a la que nos referimos son los métodos iterativos para resolver sistemas lineales: método iterativo de Jacobi y método iterativo de Gauss-Seidel. Además se exponen dos teoremas para estudiar la convergencia de ambos métodos.

2. Créditos prácticos. Metodología.

El alumno

Los créditos prácticos se imparten de la siguiente forma: los alumnos programan los algoritmos explicados en clase, en nuestro caso, usando Matlab.
Las razones de impartir así los créditos prácticos son varias; citemos algunas.
1. - Nuestros alumnos son de 2º curso y han tenido en 1º una asignatura troncal de 9 créditos llamada Fundamentos de Programación. Implementar los algoritmos numéricos con Matlab les servirá para que tengan oportunidad de practicar sus habilidades en la computación científica.
2. - Su formación como programadores debe incluir el conocimiento y manejo de algunos de los más importantes programas de cálculo simbólico hoy utilizados. Un ejemplo es el paquete Matlab.

El profesor

Si es el alumno el que programa ¿Qué papel le corresponde al profesor? Entre las muchas formas que puede haber de ayudar al alumno, hemos escogido dos para exponer a continuación.
1ª forma: Dejar, desde el principio de la clase, que el alumno programe. El profesor resuelve dudas individuales a medida que vayan surgiendo. A esta forma le llamaremos Método 1.
2ª forma: Podemos hacer una clase mas guiada con el objetivo de que a la hora de programar con Matlab, el alumno no tenga dificultades. A este método le llamaremos Método 2 ó Método propuesto porque es el que vamos a exponer con detalle y después es el que desarrollaremos para el tema que nos ocupa: métodos iterativos para sistemas lineales.

Esquema del método propuesto

Se divide en 6 pasos. Explicamos cada uno de ellos.
1. - El profesor presenta los comandos básicos que tiene Matlab de álgebra lineal necesarios para implementar los métodos iterativos.
Nota.- La primera clase es siempre una introducción al programa Matlab, pero la experiencia nos dice que no esta de más, al comienzo de cada práctica volver a comentar algunas instrucciones, aunque eso sí, sólo las que posiblemente sean necesarias en la práctica que nos ocupa.
2. - El profesor plantea sencillos ejercicios que requieran la utilización de dichos comandos.
El alumno resuelve estos ejercicios utilizando sus conocimientos del álgebra lineal. Dicha resolución le

servirá para afianzar la comprensión de los comandos expuestos en 1 y su posterior utilización al programar.
3.- El profesor hace un pequeño resumen de los métodos iterativos de Jacobi y Gauss-Seidel y de resultados afines a ellos. Sólo será un resumen, porque la explicación detallada se ha visto con antelación en la clase de teoría.
4. - Se deja tiempo al alumno para que implemente los algoritmos con Matlab. El profesor resuelve dudas individuales cuando surjan.
5. - Los alumnos verifican los programas, por ejemplo con los problemas hechos por el profesor en el apartado3.
6. - Se comparan los distintos programas y se mejoran en lo posible: Sencillez, programación estructurada, complejidad computacional, etc.

Comparación de ambos métodos

	Puntos fuertes	Puntos débiles
Metodo 1	Mas abierto a la iniciativa del alumno (uso de la ayuda de Matlab)	Tiempo insuficiente para terminar la práctica
Metodo propuesto	Ordenado en nivel de dificultad. Todos las alumnos trabajan desde el principio	No se adapta bien a todas las prácticas.

En general, hemos observado que el método propuesto ha sido aceptado por la mayoría de los alumnos, que están mas animados a trabajar.
Para el profesor requiere mas trabajo. Esta práctica pretende ser una ayuda para todos aquellos profesores que quieran probar este método.

3. Descripción detallada del método propuesto para la práctica: métodos iterativos para sistemas lineales

1.- Comandos básicos

length(x)	longitud del vector x
max(x)	componente máxima de x
size(A)	dimensión de la matriz A
eig(A)	valores propios de A
diag(A)	extrae las entradas diagonales de A
abs(x)	valor absoluto de x ó módulo de un número complejo x
A(i,j)	extrae la entrada (i,j) de la matriz A
A(:,j)	extrae la columna j-ésima de A
A(i,:)	extrae la fila i-ésima de A
A(3:5,:)	extrae las filas 3 a 5 de A (se puede generalizar)
A(:, 1:3)	extrae las columnas 1 a 3 de A (se puede generalizar)

2 .- Ejercicios

Introducir la matriz

- Calcular su dimensión.
- Mostrar la entrada (2,1).
- Mostrar la 2ª fila
- Mostrar la 3ª columna
- Restar a la 2ª fila la 1ª fila
- ¿Qué resultado obtienes al hacer diag (A)?
- ¿Y al hacer diag(diag(A))?
- Calcular los valores propios de A.

Introducir el vector x = (1,2,3,6,8,9,47)
- Calcular su longitud
- Entresacar las 4 primeras componentes
- Entresacar las 3 últimas componentes
- Calcular su máxima componente.

3.- Métodos iterativos para sistemas lineales. Resumen
Los métodos iterativos para resolver el sistema lineal

se utilizan sólo para grandes

sistemas dispersos y estructurados, donde las técnicas iterativas son eficaces tanto en términos de almacenamiento en computadora como de tiempo de cálculos.
Las técnicas iterativas involucran un proceso que convierte el sistema en un sistema equivalente de la forma para alguna matriz T y un vector c.

Se parte de un vector $x^{(0)}$, primera aproximación a la solución x y se genera la sucesión , calculando

de manera que dicha sucesión converja a x.
Vamos a exponer brevemente el método de Jacobi, el método de Gauss-Seidel y dos teoremas de convergencia, así como algún ejemplo.
Método de Jacobi para resolver el sistema $Ax = b$
Escribimos A como diferencia de dos matrices $A=N-P$ donde y nuestro sistema anterior se convierte en .

Ahora, partiendo de un iterante inicial $x^{(0)}$, podemos definir como ,es decir

Esto se puede escribir

(1)

Ejemplo:
Consideramos el sistema de ecuaciones

Empezando por ,calculamos los siguientes iterantes usando (1)

y así sucesivamente.

Al cabo de 19 iteraciones ,

,

Método de Gauss-Seidel
Es el método de Jacobi con la modificación de que las variables son utilizadas tan pronto como sean calculadas
La implementación del método será la siguiente:

Ejemplo: Resolver el sistema anterior por el método de Gauss-Seidel y mismo iterante inicial.

y así sucesivamente.

Tras 10 iteraciones , ,

Notas.1) Para que los métodos estén bien definidos es necesario que *N* sea inversible, es decir que la diagonal de *A* no tenga ningún elemento nulo.
2) Criterio de parada:
Dado que el iterante que usamos es un vector usamos como criterio de parada:

3) Se puede probar que si reordenamos el sistema del ejemplo, cambiando fila 1 por fila 3 ,con el mismo iterante inicial, los métodos divergen.
Se hace necesario disponer de algún criterio que determine si los métodos iterativos convergen.

Teorema 1: Si *A* es diagonalmente dominante, es decir, , entonces los métodos convergen para cualquier iterante inicial.

Teorema 2: Sea el sistema . Si es la matriz de iteración del método iterativo y es el iterante inicial, la sucesión converge a la solución del sistema si y sólo donde

Concluidos los pasos 1º,2º y 3º, en los pasos 4º y 5º trabaja el alumno.
Se le pide:

a) Implementar el método de Jacobi.
b) Implementar el método de Gauss-Seidel.
c) Hacer un pequeño programa para verificar el teorema 2 de condición necesaria y suficiente de convergencia.

Para concluir el paso 6º a continuación se exponen los programas.

```
function y=gausei(A,B,x,tol)
% resolución de sistemas lineales por
%Gauss-Seidel
% A matriz del sistema
% B término independiente
% x iterante inicial
% tol tolerancia
 [m,n]=size(A);
 if m~=n
   'matriz no cuadrada'
   return
 end
for i=1:m
  if A(i,i)==0
  'elementos diagonales nulos'
  return
  end
end
 y=x;
% iteración
for k=1:100
   y(1)=(B(1)-A(1,2:n)*x(2:n))/A(1,1);
   for i=2:n-1
    y(i)=(B(i)-A(i,1:i-1)*y(1:i-1)-
      A(i,i+1:n)*x(i+1:n))/A(i,i);
```

```
    end
       y(n)=(B(n)-A(n,1:n-1)*y(1:n-
            1))/A(n,n);
    if max(abs(y-x))<tol
       'nº de iteraciones',k
        return
    end
    x=y;
end
```

'100 iteraciones no han sido suficientes'

```
function y=jacobi(A,B,x,tol)
% resolución de sistemas lineales por
%Jacobi
% A matriz del sistema
% B término independiente
% x iterante inicial
% tol tolerancia
 [m,n]=size(A);
for i=1:m
 if A(i,i)==0
  'elementos diagonales nulos'
  return
 end
end
% iteración
N=diag(diag(A));
P=N-A;
N=N^(-1);
for i=1:100
  x1=N*(P*x+B);
       if max(abs(x1-x))<tol
         y=x1;
          'nº de iteraciones',i
         return
       end
  x=x1;
end

'100 iteraciones no han sido
suficientes'
y=x1;
```

```
function y=teorema( )
%Verificación del teorema de
%convergencia para el método de Jacobi
A=input('introduce la matriz de
coeficientes:');
disp(A);
[n,m]=size(A);
for i=1:n
 for j=1:n
  if i==j
    M(i,j)=0
    else M(i,j)=-A(i,j)/A(i,i);
  end
 end
end
v=eig(M);
n=length(v);
  for i=1:n
   l(i)=abs(v(i));
  end
radio=max(l);
if radio<1,
  disp('el método converge')
  else
  disp('el método diverge')
end
```

4- Referencias

[1] John H. Mathews, Kurtis D. Fink, *Métodos Numéricos con Matlab.* Prentice Hall 2.000

[2] Bernard Kolman, *Algebra Lineal con aplicaciones y Matlab.* Prentice Hall 1.999

[3] Richard L. Burden, J. Douglas Faires, *Análisis Numérico.* Grupo Editorial Iberoamérica 1.993