12-10-2016

**Métodos de Computación Científica**

Alumno:Stessens,Alejandro

LU:83262

**LABORATORIO Nro 2**

Ejercicio 1)

Implemento una función que calcula el método SOR

function [x] = sor()

disp('POR FAVOR,INGRESE LA MATRIZ QUE SE LE PIDA DE FORMA ORDENADA');

n=input('CUANTAS ECUACIONES:');

A=input('INGRESE LA MATRIZ A:');

b=input('INGReSE LA MATRIZ B:');

maxiter=input('CUANTAS ITERACIONES DESEA HACER:');

x0=input('INGRESE VECTOR APROXIMACION:');

w=input('INGRESE ACELERADOR W:');

tol= 0.00001;

[m n]= size(A);

if m~=n, error('Matriz del sistema no cuadrada'), end

if m~= length(b), error ('sistema no coherente'),end

x=zeros(size(b));

x2=x;

if any(abs(diag(A))<eps)

error('Metodo no valido. Elemento diagonal nulo')

end

for k=1:maxiter

for i=1:n

tem=(b(i)-sum(A(i,[1:i-1 i+1:n])\*x0([1:i-1 i+1:n])'))/A(i,i);

x(i)=tem\*w + (1-w)\*x0(i);

x0(i)=x(i);

end

if norm(x-x2)< tol

fprintf('\n SOR CONVERGE en %d iteraciones\n',k),

return

end

x2=x;

end

fprintf('\n SOR NO CONVERGE en %d iteraciones\n',maxiter)

Realizo los siguientes testeos:

CUANTAS ECUACIONES:3

INGRESE LA MATRIZ A:[ -5 -1 2; 2 6 -3; 2 1 7 ]

INGReSE LA MATRIZ B:[1 2 32]

CUANTAS ITERACIONES DESEA HACER:100

INGRESE VECTOR APROXIMACION:[0 0 0]

INGRESE ACELERADOR W:0.7

**SOR CONVERGE en 15 iteraciones**

**ans = 1.0000 2.0000 4.0000**

CUANTAS ECUACIONES:3

INGRESE LA MATRIZ A:[ -5 -1 2; 2 6 -3; 2 1 7 ]

INGReSE LA MATRIZ B:[1 2 32]

CUANTAS ITERACIONES DESEA HACER:100

INGRESE VECTOR APROXIMACION:[0 0 0]

INGRESE ACELERADOR W:0.9

**SOR CONVERGE en 10 iteraciones**

**ans = 1.0000 2.0000 4.0000**

CUANTAS ECUACIONES:3

INGRESE LA MATRIZ A:[ -5 -1 2; 2 6 -3; 2 1 7 ]

INGReSE LA MATRIZ B:[1 2 32]

CUANTAS ITERACIONES DESEA HACER:100

INGRESE VECTOR APROXIMACION:[0 0 0]

INGRESE ACELERADOR W:0.97

**SOR CONVERGE en 9 iteraciones**

**ans = 1.0000 2.0000 4.0000**

***Valor de relajación óptimo***

El mejor factor de relajación que encontré es

W=0.97

EJERCICIO 2)

Trial>> A=[1 1;1.001 1]

A =

1.0000 1.0000

1.0010 1.0000

Trial>> b=[1;2]

b =

1

2

Trial>> linsolve(A,b)

Resuelvo Matriz aumentada:

ans =

1.0e+03 \*

1.0000

-0.9990

Expreso como aumentada:

Trial>> ACalculada=[1 -0.999;1.001 -0.999]

ACalculada =

1.0000 -0.9990

1.0010 -0.9990

Trial>> total=[ACalculada,b]

total =

1.0000 -0.9990 1.0000

1.0010 -0.9990 2.0000

Trial>> cond(total)

ans =

6.2078

Hallo el determinante

Trial>> det(ACalculada)

ans =

9.9900e-04

Ejercicio 3)

A)

Dado un A con las características que me dice el enunciado, empleo Factorizacion LU y calculo sus incógnitas.

**Trial>> A = [U B]**

**A =**

**6 0 0 1**

**0 6 0 2**

**0 0 6 3**

**Trial>> rref(A)**

**ans =**

**1.0000 0 0 0.1667**

**0 1.0000 0 0.3333**

**0 0 1.0000 0.5000**

**Trial>> y=[0.1667;0.3333;0.5]**

**y =**

**0.1667**

**0.3333**

**0.5000**

**Calculo**

**Ly=B:**

**Trial>> B=[L y]**

**Expreso como aumentada el calculo Ly=B:**

**B = 1.0000 0 0 0.1667**

**0 1.0000 0 0.3333**

**0 0 1.0000 0.5000**

**Ahora calculo Ux=Y**

**Trial>> rref(B)**

**ans =**

**1.0000 0 0 0.1667**

**0 1.0000 0 0.3333**

**0 0 1.0000 0.5000**

**Incognitas:**

**X=0.1667**

**Y=0.3333**

**Z=0.5000**

**B)**

1) Ingresamos la matriz A y el vector de soluciones b. Hallamos la descomposición LU de A usando la función predefinida lu(A) y procedemos a resolver el sistema.

2) Aprovechamos que las matrices obtenidas son triangulares para resolver el sistema de forma más eficiente.

Para esto, usamos el comando linsolve(A,B,opts). Este comando nos permite

especificar la forma de resolución más apropiada según el formato de la matriz A modificando el valor del parámetro opts.

Primero resolvemos el sistema Ly = B, donde y = Ux

En este caso, L es triangular inferior. Por eso, establecemos el valor de LT.opts en true y

procedemos a resolver.

>> opts.LT = true;

>> r1 = linsolve(L,b,opts)

A continuación resolvemos Ux = y, cuyo resultado será el resultado definitivo. Establecemos

el valor de opts adecuadamente y resolvemos el sistema.

>> opts.UT = true;

>> opts.LT = false;

>> r1 = linsolve(U,r1,opts)

y en el r1 me hallará el valor del xi.

**C)**

**Trial>> A=[5 -5 0 0 0;-5 10 -5 0 0;0 -5 10 -5 0;0 0 -5 10 -5;0 0 0 -5 10]**

**A =**

**5 -5 0 0 0**

**-5 10 -5 0 0**

**0 -5 10 -5 0**

**0 0 -5 10 -5**

**0 0 0 -5 10**

**Trial>> b1=[0;1;0;0;0]**

**b1 =**

**0**

**1**

**0**

**0**

**0**

**Trial>> b2=[0;0;1;0;0]**

**b2 =**

**0**

**0**

**1**

**0**

**0**

**Trial>> b3=[0;0;0;1;0]**

**b3 =**

**0**

**0**

**0**

**1**

**0**

**Trial>> b4=[1;1;1;1;1]**

**b4 =**

**1**

**1**

**1**

**1**

**1**

**Trial>> [L,U] = lu(A)**

**L =**

**1 0 0 0 0**

**-1 1 0 0 0**

**0 -1 1 0 0**

**0 0 -1 1 0**

**0 0 0 -1 1**

**U =**

**5 -5 0 0 0**

**0 5 -5 0 0**

**0 0 5 -5 0**

**0 0 0 5 -5**

**0 0 0 0 5**

**Procedo a calcular los valores de X:**

**Trial>> opts.LT = true;**

**r1 = linsolve(L,b1,opts)**

**r1 =**

**0**

**1**

**1**

**1**

**1**

**Trial>> opts.UT = true;**

**opts.LT = false;**

**r1 = linsolve(U,r1,opts)**

**El valor de X1 es:**

**r1 =**

**0.8000**

**0.8000**

**0.6000**

**0.4000**

**0.2000**

**Trial>> opts.LT = true;**

**r1 = linsolve(L,b2,opts)**

**r1 =**

**0**

**0**

**1**

**1**

**1**

**Trial>> opts.UT = true;**

**opts.LT = false;**

**r1 = linsolve(U,r1,opts)**

**el valor de x2 es:**

**r1 =**

**0.6000**

**0.6000**

**0.6000**

**0.4000**

**0.2000**

**Trial>> opts.LT = true;**

**r1 = linsolve(L,b3,opts)**

**r1 =**

**0**

**0**

**0**

**1**

**1**

**Trial>> opts.UT = true;**

**opts.LT = false;**

**r1 = linsolve(U,r1,opts)**

**el valor de x3 es:**

**r1 =**

**0.4000**

**0.4000**

**0.4000**

**0.4000**

**0.2000**

**Trial>> opts.LT = true;**

**r1 = linsolve(L,b4,opts)**

**r1 =**

**1**

**2**

**3**

**4**

**5**

**Trial>> opts.UT = true;**

**opts.LT = false;**

**r1 = linsolve(U,r1,opts)**

**el valor de x4 es:**

**r1 =**

**3.0000**

**2.8000**

**2.4000**

**1.8000**

**1.0000**