NATIONAL INSTITUE OF TECHNOLOGY KARANATAKA, SURATHKAL

ASSIGNMENT 3

APPLIED COMPUTATIONAL METHODS IN MECHANICAL SCIENCES

RAJAT A CHANDAVAR – 16ME156

19-Aug-19

ASSIGNMENT ON JACOBI, GAUSS-SIDEL AND SOR ITERATIVE METHODS

Question 1 Answer:

$$\begin{bmatrix} 10 & -1 & 2 & 0 \\ -1 & 11 & -1 & 3 \\ 2 & -1 & 10 & -1 \\ 0 & 3 & -1 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 6 \\ 25 \\ -11 \\ 15 \end{bmatrix}$$

Program (C++)

```
#include<iostream>
#include<cmath>
#include<time.h>
using namespace std;
main()
{
    clock_t start=clock();
    int i,j,n=4,itrn=0,b[n]={6,25,-11,15},flag=1,choice=1;
    int a[n][n]={{10,-1,2,0},{-1,11,-1,3},{2,-1,10,-1},{0,3,-1,8}};
    float x0[n],x[n]={0},sum,omega=1,max_error;
    switch(choice)
    {
        case 1:cout<<"JOCOBI ITERATION METHOD";break;
        case 2:cout<<"GAUSS SIDEL ITERATION METHOD";break;
        case 3:cout<<"SUCCESSIVE OVER-RELAXATION METHOD\nOmega =
"<<omega;break;
    }
    for(i=0;i<n;++i)
    {   sum=0;
        for(j=0;j<n;j++)
        {
            if(j!=i)
                sum+=abs(a[i][j]);
        }
        if(sum<=a[i][i])
        {
            flag=0;
            break;
        }
    }
    if(flag)
        cout<<"\nMatrix is not diagonally dominant";
    else
    {
        do
        {
            ++itrn;
            for(i=0;i<n;++i)
                x0[i]=x[i];
            for(i=0;i<n;++i)
            {
                sum=0;
                for(j=0;j<n;++j)
                {
                    if(j==i)
                        continue;
                    if(choice==1)
                        sum+=a[i][j]*x0[j];
                    else
                        sum+=a[i][j]*x[j];
                }
```

```
                    if(choice==3)
                        x[i]=(1-omega)*x[i]+(omega/a[i][i])*(b[i]-sum);
                    else
                        x[i]=(b[i]-sum)/a[i][i];
                    if(!i)
                        max_error=abs(x[i]-x0[i]);
                    else if(abs(x[i]-x0[i])>max_error)
                        max_error=abs(x[i]-x0[i]);
                }
            }while(max_error>1e-4);
            cout<<"\nNo. of iterations:"<<itrn<<"\nSolution Vector:";
            for(i=0;i<n;i++)
                cout<<"\nx"<<i+1<<" = "<<x[i];
            clock_t stop=clock();
            double timespent = (double)(stop-start)/(double)CLOCKS_PER_SEC;
            cout<<"\nCPU Time:"<<timespent<<" seconds";
        }
    }
```

Output:

Choice 1: Jacobi

```
JOCOBI ITERATION METHOD
No. of iterations:13
Solution Vector:
x1 = 0.99999
x2 = 2.00002
x3 = -1.00001
x4 = 1.00002
CPU Time:0.004 seconds
```

Choice 2: Gauss Sidel

```
GAUSS SIDEL ITERATION METHOD
No. of iterations:6
Solution Vector:
x1 = 1.00001
x2 = 2
x3 = -1
x4 = 0.999999
CPU Time:0.003 seconds
```

Choice 3: SOR

```
SUCCESSIVE OVER-RELAXATION METHOD
Omega = 0.5
No. of iterations:18
Solution Vector:
x1 = 0.999881
x2 = 1.99988
x3 = -0.999917
x4 = 1.00013
CPU Time:0.004 seconds
```

```
SUCCESSIVE OVER-RELAXATION METHOD
Omega = 0.75
No. of iterations:10
Solution Vector:
x1 = 0.999973
x2 = 2
x3 = -0.999983
x4 = 1.00001
CPU Time:0.003 seconds
```

```
SUCCESSIVE OVER-RELAXATION METHOD
Omega = 0.85
No. of iterations:8
Solution Vector:
x1 = 1.00001
x2 = 2.00003
x3 = -1.00001
x4 = 0.999982
CPU Time:0.003 seconds
```

```
SUCCESSIVE OVER-RELAXATION METHOD
Omega = 0.95
No. of iterations:7
Solution Vector:
x1 = 1.00001
x2 = 2.00001
x3 = -1.00001
x4 = 0.999995
CPU Time:0.003 seconds
```

```
SUCCESSIVE OVER-RELAXATION METHOD
Omega = 0.99
No. of iterations:7
Solution Vector:
x1 = 1
x2 = 2
x3 = -1
x4 = 1
CPU Time:0.005 seconds
```

```
SUCCESSIVE OVER-RELAXATION METHOD
Omega = 1
No. of iterations:6
Solution Vector:
x1 = 1.00001
x2 = 2
x3 = -1
x4 = 0.999999
CPU Time:0.003 seconds
```

```
SUCCESSIVE OVER-RELAXATION METHOD
Omega = 1.1
No. of iterations:7
Solution Vector:
x1 = 0.999997
x2 = 2
x3 = -0.999999
x4 = 1
CPU Time:0.003 seconds
```

```
SUCCESSIVE OVER-RELAXATION METHOD
Omega = 1.05
No. of iterations:6
Solution Vector:
x1 = 1
x2 = 2
x3 = -1
x4 = 0.999999
CPU Time:0.003 seconds
```

```
SUCCESSIVE OVER-RELAXATION METHOD
Omega = 1.3
No. of iterations:10
Solution Vector:
x1 = 1.00003
x2 = 2.00001
x3 = -1.00001
x4 = 1.00001
CPU Time:0.003 seconds
```

```
SUCCESSIVE OVER-RELAXATION METHOD
Omega = 1.09
No. of iterations:6
Solution Vector:
x1 = 1.00001
x2 = 1.99999
x3 = -1
x4 = 1
CPU Time:0.003 seconds
```

```
SUCCESSIVE OVER-RELAXATION METHOD
Omega = 1.6
No. of iterations:23
Solution Vector:
x1 = 1.00002
x2 = 2.00001
x3 = -1.00001
x4 = 1.00002
CPU Time:0.003 seconds
```

OPTIMUM OMEGA = 1 to 1.09