Simulation of Gauss-Seidel Method

University of South Florida

United States of America

http://numericalmethods.eng.usf.edu/mws

NOTE: This worksheet demonstrates the use of Maple to illustrate the Gauss-Seidel Method, an iterative technique used in solving a system of simultaneous linear equations.

```
> restart;
```

Introduction

Gauss-Seidel method is used to solve a set of simultaneous linear equations, [A] [X] = [RHS], where $[A]_{nxn}$ is the square coefficient matrix, $[X]_{nx1}$ is the solution vector, and $[RHS]_{nx1}$ is the right hand side array. The equations can be rewritten as:

$$x[i] = \frac{(rhs[i] - Sum(a[i,j] \cdot x[j], i = 1..n)[i <> j])}{a[i,i]}$$

$$x_i = \frac{rhs_i - \left(\sum_{i=1}^{n} a_{i,j} x_j\right)_{i \neq j}}{a_{i,i}} \qquad (1.1)$$

In certain cases, such as when a system of equations is large, iterative methods of solving equations such as Gauss-Seidel method are more advantageous. Elimination methods, such as Gaussian Elimination, are prone to round-off errors for a large set of equations whereas iterative methods, such as Gauss-Seidel method, allow the user to control round-off error. Also if the physics of the problem are well known, initial guesses needed in iterative methods can be made more judiciously for faster convergence.

Complete details of how this formula is derived as well as pitfalls of the method can be found here.

Steps to apply Gauss-Seidel Method:

1) Make an initial guess for the solution vector [X]. This can be based on the physics of the problem. (*Note*: To begin, the initial guess will be considered X_{old}).

$$X_{old} = [x_1, x_2, x_3 .. x_n]$$

$$X_{old} = [x_1, x_2, x_3 .. x_n] \qquad (1.2)$$

2) Substitute the initial guess solution vector [X] in Equation (1.1).

$$x[i] = \frac{(rhs[i] - Sum(a[i,j]\cdot x[j], i = 1..n)[i <> j])}{a[i,i]}$$

$$x_i = \frac{rhs_i - \left(\sum_{i=1}^{n} a_{i,j}\, x_j\right)_{i \neq j}}{a_{i,i}} \tag{1.3}$$

3) The new x_i guess that is obtained will replace the previous guess in the [X] vector.

$$X_{old} = \left[x_{1\,new}, x_2, x_3..x_n\right]$$

$$X_{old} = \left[x_{new}, x_2, x_3..x_n\right] \tag{1.4}$$

[X] will then be used to calculate the next x_i value by repeating Step 2. This will be repeated n times until the new solution vector [X] is complete.

$$X_{new} = \left[x_{1\,new},\ x_{2\,new}, x_{3\,new}...x_{n\ new}\right]$$

$$X_{new} = \left[x_{new}, x_{2\,new}, x_{3\,new}..x_{n\,new}\right] \tag{1.5}$$

4) At this point, the first iteration is completed and the absolute relative approximate error (abs_ea) is calculated by comparing the new guess [X] with the previous guess [X_{old}].

$$abs\_ea[i] = abs\left(\frac{\left(x_{[i]new} - x_{[i]old}\right)}{x_{[i]new}}\right)\cdot 100$$

$$a\,b\,s\_ea_i = 100\left|\frac{x_{[i]\,new} - x_{[i]\,old}}{x_{[i]\,new}}\right| \tag{1.6}$$

The maximum of these errors is the absolute relative approximate error at the end of the iteration.

5) Repeat Steps 1-4, replacing the new solution vector with the old solution vector in Step 1. Repeat until you have conducted either the maximum number of iterations or met the pre-specified tolerance.

$$X_{old} = X_{new}$$

$$X_{old} = X_{new} \tag{1.7}$$

A simulation of Gauss-Seidel method follows.

Section I: Input Data

The following are the input parameters to begin the simulation. This is the only section that requires user input. Once values are entered, Maple will calculate the approximate solution to the system of equations in Section II, while demonstrating each iterative step taken to arrive at the solution.

n = number of equations
[A] = *n*x*n* coefficient matrix
[RHS] = *n*x1 right hand side array
[Xold] = *n*x1 initial guess of the solution vector
maxit = maximum number of iterations

```
> restart;
> n:=4;
  A:=Matrix([[10,3,4,5],[2,24,7,4],[2,2,34,3],[2,5,2,12]]);
  RHS:=[22,32,41,18];
  X:=[1,23,4,50];
  maxit:=5;
```

$$n := 4$$

$$A := \begin{bmatrix} 10 & 3 & 4 & 5 \\ 2 & 24 & 7 & 4 \\ 2 & 2 & 34 & 3 \\ 2 & 5 & 2 & 12 \end{bmatrix}$$

$$RHS := [22, 32, 41, 18]$$

$$X := [1, 23, 4, 50]$$

$$maxit := 5 \qquad (2.1)$$

Section II: Iterations

Below, *maxit* iterations are conducted and values of the previous approximations, present approximations, absolute relative percentage approximate error, and maximum absolute relative percentage approximate error are calculated at the end of each iteration.

```
> #Initializing the absolute relative approximate error array
  abs_ea:=Array(1..n);
```

$$abs\_ea := [0 \quad 0 \quad 0 \quad 0] \qquad (3.1)$$

Iterations

```
> #Defining the number of iterations to be conducted.
  for k from 1 by 1 to maxit do
  print("Iteration number");
  k;
  print("Previous iteration values of the solution vector");
  #Replacing the initial or old value of [X] with the value that was calculated in the previous
  iteration.
  Xold:=X;
```

```
#The following i loop generates the new solution vector:
for i from 1 by 1 to n do
    #Initializing the series sum of Equation (1.1) to zero.
    summ:=0.0:
    for j from 1 by 1 to n do
        #Only adding i<> terms.
        if (i<>j) then
            #Generating the summation term.
            summ:=summ+A[i,j]*X[j]:
        end if:
    end do:
    #Using Gauss-Seidel method Equation (1.1) to calculate new [X] term.
    X[i]:=(RHS[i]-summ)/A[i,i];
end do:
#Initializing the maximum absolute relative percentage approximate error to zero.
Max_abs_ea:=0.0:
#The following i loop generates the maximum absolute relative percentage approximate error
for the k^th step:
for i from 1 by 1 to n do
    #Calculating absolute relative percentage approximate error for each X_i.
    abs_ea[i]:=abs((X[i]-Xold[i])/X[i])*100.0;
        #Defining the maximum value of the relative approximate errors.
        if abs_ea[i]>Max_abs_ea then
           Max_abs_ea:=abs_ea[i]:
        end if:
end do:
print("New iterative values of the solution vector");
X;
print("Absolute relative percentage approximate error");
abs_ea;
print("Maximum absolute relative percentage approximate error");
Max_abs_ea;
print(``);
cat(' ___________________________________________________ ');
print(``);
end do;
>
```

"Iteration number"

1

"Previous iteration values of the solution vector"

Xold := [1, 23, 4, 50]

Max_abs_ea := 0.

"New iterative values of the solution vector"

[−31.30000000, −5.558333333, −1.037745098, 9.205596408]

"Absolute relative percentage approximate error"

[103.1948882 513.7931034 485.4511101 443.1478612]

"Maximum absolute relative percentage approximate error"

513.7931034

"Iteration number"

2

"Previous iteration values of the solution vector"

Xold := [−31.30000000, −5.558333333, −1.037745098, 9.205596408]

Max_abs_ea := 0.

"New iterative values of the solution vector"

[−.3202001650, 0.1284262662, 0.4049046641, 1.432371639]

"Absolute relative percentage approximate error"

[9675.135502 4428.034675 356.2936883 542.6821195]

"Maximum absolute relative percentage approximate error"

9675.135502

"Iteration number"

3

"Previous iteration values of the solution vector"

Xold := [−.3202001650, 0.1284262662, 0.4049046641, 1.432371639]

Max_abs_ea := 0.

"New iterative values of the solution vector"

[1.283324435, 0.8695638300, 0.9528561338, 0.7649849760]

"Absolute relative percentage approximate error"

[124.9508352 85.23095582 57.50621214 87.24180003]

"Maximum absolute relative percentage approximate error"

124.9508352

"Iteration number"

4

"Previous iteration values of the solution vector"

Xold := [1.283324435, 0.8695638300, 0.9528561338, 0.7649849760]

Max_abs_ea := 0.

"New iterative values of the solution vector"

[1.175495910, 0.8299614725, 1.020415597, 0.7881974687]

"Absolute relative percentage approximate error"

[9.173024260 4.771589864 6.620779161 2.945009800]

"Maximum absolute relative percentage approximate error"

9.173024260

"Iteration number"

5

"Previous iteration values of the solution vector"

Xold := [1.175495910, 0.8299614725, 1.020415597, 0.7881974687]

Max_abs_ea := 0.

"New iterative values of the solution vector"

[1.148746585, 0.8086169908, 1.021196484, 0.8014190757]

"Absolute relative percentage approximate error"

[2.328566226 2.639628148 0.07646785043 1.649774431]

"Maximum absolute relative percentage approximate error"

2.639628148

(3.2)

Section III: Exact Answer

The last iteration above generates an approximate value with the given number of maximum iterations. The exact answer, last iterative solution vector, absolute relative true error (abs_et), maximum absolute relative true error (Max_abs_et), and maximum absolute relative percentage approximate error are given below.

Note: If the coefficient matrix $[A]_{nxn}$ is diagonally dominant, convergence is ensured. Otherwise, the solution may or may not converge.

```
> #Initializing the linear algebra package
  with(linalg):
  #Solving for the exact solution vector [X]
  B:=evalf(linsolve(A,RHS)):
  #Defining the absolute relative true error array
  abs_et:=Array(1..n):
  for i from 1 by 1 to n do
      #Calculating absolute relative true error
```

```
    abs_et[i]:=abs((B[i]-X[i])/B[i])*100.0;
  end do:
  Max_abs_et:=0.0:
  for i from 1 by 1 to n do
    #Defining the maximum absolute relative true error
    if Max_abs_et<=abs_et[i] then
          Max_abs_et:=abs_et[i]:
      end if:
  end do:
  print("Exact Answer");
  print(B);
  print(``);
  cat(' _____________________________________________ ');
  print(``);
  print("Last Iterative Value");
  X;
  print(``);
  cat(' _____________________________________________ ');
  print(``);
  print("Absolute relative true error");
  abs_et;
  print(``);
  cat(' _____________________________________________ ');
  print(``);
  print("Maximum absolute relative true error");
  Max_abs_et;
  print(``);
  cat(' _____________________________________________ ');
  print(``);
  print("Maximum absolute relative percentage approximate error");
  Max_abs_ea;
```

```
Warning, the protected names norm and trace have been redefined
and unprotected
```

"Exact Answer"

[1.148801058 0.8063228286 1.020061382 0.8025550815]

_____

"Last Iterative Value"

[1.148746585, 0.8086169908, 1.021196484, 0.8014190757]

_____

"Absolute relative true error"

[0.004741726134 0.2845215488 0.1112778133 0.1415486396]

"Maximum absolute relative true error"

0.2845215488

"Maximum absolute relative percentage approximate error"

2.639628148 (4.1)

References

[1]Autar Kaw, *Holistic Numerical Methods Institute, http://numericalmethods.eng.usf.edu/mws,* See How does Gauss-Seidel Method work?

Conclusion

Maple helped us apply our knowledge of Gauss-Seidel method to solve a system of *n* simultaneous linear equations.

Question 1: Change the coefficient matrix to one that is not diagonally dominant and see if Gauss-Seidel method converges.

Question 2: See if you can get a set of equations with a coefficient matrix that is not diagonally dominant to converge by Gauss-Seidel method.