Gauss-Seidel Method

In the course of numerical modeling, we often need to solve linear systems of algebraic equations. Consider the simple 3 × 3 linear system:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \end{Bmatrix}$$

which is equivalent to:

$$\begin{array}{rcrcrcl} a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & a_{23}x_3 & = & b_2 \\ a_{31}x_1 & + & a_{32}x_2 & + & a_{33}x_3 & = & b_3 \end{array}$$

In lectures, several techniques for solving linear systems have been discussed. In the Gauss-Seidel method, the equations are rearranged such that the first equation is in terms of x_1, the second in terms of x_2, and so on and so forth.

$$x_1 = \frac{b_1 - a_{12}x_2 - a_{13}x_3}{a_{11}} \qquad x_1^{i+1} = \frac{b_1 - a_{12}x_2^i - a_{13}x_3^i}{a_{11}}$$

$$x_2 = \frac{b_2 - a_{21}x_1 - a_{23}x_3}{a_{22}} \qquad x_2^{i+1} = \frac{b_2 - a_{21}x_1^{i+1} - a_{23}x_3^i}{a_{22}}$$

$$x_3 = \frac{b_3 - a_{31}x_1 - a_{32}x_2}{a_{33}} \qquad x_3^{i+1} = \frac{b_3 - a_{31}x_1^{i+1} - a_{32}x_2^{i+1}}{a_{33}}$$

Beginning with an initial guess of zero for all x_i, these equations are used, in an iterative manner, to estimate improved values. The point to note is that the new estimates of x_i are calculated using values of x_i from the same iteration, as well as the previous iteration. It therefore converges faster than the Jacobi Method, which only uses values from the previous iteration.

Like other iterative methods, an approximate error can be computed. To avoid division by zero when calculating the approximate error, it can be calculated as the absolute difference between two consecutive estimates, as oppose to the percentage difference:

$$\varepsilon_{a,j} = \left| x_j^{i+1} - x_j^i \right|$$

where $\varepsilon_{a,j}$ is the approximate error in x_j at iteration $i + 1$. This is not ideal, as the x_i may be of different orders of magnitude, but it does makes life easier!

Your Task

Your task is to write a MATLAB function that performs the Gauss-Seidel method to solve a linear system of the form $[A]\{x\}=\{b\}$. It must work irrespective of the dimensions of the system. While it would be preferable to include pivoting, it is not expected and you should not attempt to include it, unless you have finished.

Your MATLAB function should stop either when the approximate error has fallen below a certain limit or a pre-defined maximum number of iterations is reached. The value of all x_i should be written to the screen at each iteration.

The matrix $[A]$ and vector $\{b\}$, error limit and maximum number of iterations should be passed to the function at the command line.

For example

```
>> A = [12 6 3; 3 13 2; 2 4 8]; b = [171; 173; 84];
>> miner = 0.000001; maxit = 100;
>> gseidel(A,b,miner,maxit)
```

or in a more efficient manner

```
>> A = [12 6 3; 3 13 2; 2 4 8]; b = [171; 173; 84];
>> gseidel(A,b,0.000001,100)
```

More Details

1. To perform the Gauss-Seidel method the dimensions of $[A]$ and $\{b\}$ must be compatible. In addition, all the diagonal elements of $[A]$ must be non-zero. Include error checks in your script to ensure that this is true.

2. The simplest way to proceed is to use three nested for loops: an outer loop to loop over the iterations, a middle loop to loop over all of the equations and an inner loop to sum the $a_{ij}x_j$ terms for each equation.

3. You can reduce the number of loops required to two, by using the MATLAB function `diag()`, along with colon notation. Note how, the sum of the $a_{ij}x_j$ terms never includes the diagonal a_{ij} values.

4. Formatting output is difficult when the number of values to write out can vary. Try using the MATLAB function `repmat()` to enable your script to create a format statement of correct length, to put into your `fprintf` statement.

In the early stages, it may be useful to calculate intermediate values either by hand or using Excel, to compare with those output by your program.

Testing Your MATLAB Script

Use your script to solve the linear system

$$\begin{bmatrix} 12 & 6 & 3 \\ 3 & 13 & 2 \\ 2 & 4 & 8 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 171 \\ 173 \\ 84 \end{Bmatrix}$$

Use an error limit of 0.000001 and set the maximum number of iterations to 100.

The exact solution is

$$\{x\} = \begin{Bmatrix} 8 \\ 11 \\ 3 \end{Bmatrix}$$

You should not expect your script to give exact values, but similar.

In addition, you should not expect your program to converge for all possible $[A]$,as convergence is only guaranteed when $[A]$ is diagonally dominant.