# Jacobi Iterative Method

This is an iterative method used to solve n\*n matrix i.e. equation of the form : **Ax=b**, where A is an n\*n matrix having these three properties: **a**) *non-zero entries on diagonal,* **b)** *a diagonal entry of a row is greater than the absolute sum of other*  *elements in that row* (it implies that matrix should be diagonally dominant), **c)** *a system has unique solution.*

Below is the algorithm of this method:

Input a n\*n matrix **A,** vector **b (**which is the final form of matrix A after multiplying t by variable vector x), **XO** which is assumed to be equals to 0 vector,  **tol**  tolerance level which is also called *stopping criteria*, , maximum number of iterations **N**.

**Step 1:**

Set k=1, k is number of iterations.

**Step 2**:

while ( k<=N) do Steps 3-6

**Step 3**:

For i=0,1,2….n

**Step 4:** Step 4 If ||x-XO ||<tol , then OUTPUT (x1,x2,….xn );

STOP

**Step 5**:

Set k=k+1

**Step 6**:

For i=1,2,3…n

Set

**Step 7:**

Output (x1,x2…,xn);

stop

# The Gauss-Seidel Method

The conditions of a matrix are same as in Jacobi. But since the Jacobi method, the values of obtained in the kth iteration remain unchanged until the entire k+1th iteration has been calculated. With the Gauss-Seidel method, we use the new values as soon as they are known. For example, once we have computed from the first equation, its value is then used in the second equation to obtain the new and so on.

Here is the algorithm:

Input a n\*n matrix **A,** vector **b (**which is the final form of matrix A after multiplying it by variable vector x), **XO** which is assumed to be equals to 0 vector,  **tol**  tolerance level which is also called *stopping criteria*, , maximum number of iterations **N**.

**Step 1:**

Set k=1, k is number of iterations.

**Step 2**:

while ( k<=N) do Steps 3-6

**Step 3**:

For i=0,1,2….n

**Step 4:** Step 4 If ||x-XO ||<tol , then OUTPUT (x1,x2,….xn );

STOP

**Step 5**:

Set k=k+1

**Step 6**:

For i=1,2,3…n

Set

**Step 7:**

Output **(**x1,x2,…,xn);

Stop

|  |  |  |
| --- | --- | --- |
| Size of matrix n\*n | Number of iterations by my Jacobi Method | Number of iterations by my Gauss-Seidal Method |
| 100\*100 | 15 | 10 |
| 200\*200 | 14 | 9 |
| 400\*400 | 12 | 9 |

|  |  |  |
| --- | --- | --- |
| Size of matrix n\*n | Seconds taken by my Jacobi Method in 4.dp | Seconds taken by my Gauss-Seidal Method in 4.dp |
| 100\*100 | 0.4920 seconds | 0.0179 seconds |
| 200\*200 | 0.4234 seconds | 0.0160 seconds |
| 400\*400 | 1.1758 seconds | 0.0246 seconds |

The theoretical concept is that gauss is faster than Jacobi and approximately gauss takes half iterations than Jacobi do for a particular matrix, and my code verifies this.