Iterative Methods

Anand Kamble

26th September 2023

1 Introduction

In this assignment, we implement the Jacobi and Gauss-Seidel algorithms to solve the $Ax = b$. These are the iterative methods for determining the solutions of a strictly diagonally dominant system of linear equations.

2 Program

2.1 User Input

To read input from the user and store it in the variables `n` and `niter` we are using the function `scanf`, which is from the C library `stdio`.

2.2 Memory Allocation

According to the input provided by the user, we need to allocate memory dynamically to hold the matrices. For this purpose, we use the function `malloc` provided by the C library. We are using this function to allocate memory for three matrices named A, B, and X. `stdlib`.

2.3 Error handling

While allocating the required memory using the above function, it is possible that it cannot be allocated. In this case, the malloc function will return 0, that is `NULL`. Hence, we are checking if any of the pointers to the matrices A, B, and X are `NULL`. If so, we are calling the function `free` to deallocate the dynamically allocated memory and printing a message `Failed to allocate memory.`, followed by gracefully exiting the program with a function `exit`.

2.4 Matrix Initialization

To initialize the matrices with the given conditions we are using the function `initializeArrays`. In this function, we are filling the memory allocated for the matrices with 0 and updating the matrices by using the following conditions, For matrix A,

$$A(i,i) = 2.0,\ where\ i = 0, ..., (n-1)$$

$$A(i,i+1) = -1.0;\ A(i+1,i) = -1.0; i = 0, \ldots, (n-2)$$

For matrix B, all the elements will be zero, except the first and last which will be 1.0

$$B(0) = 1.0; B(n-1) = 1.0$$

For matrix X, since we are using two methods that will solve for the same A and B. We can use only one matrix, with two columns for storing values of X from both methods. Hence X will be,

$$X = \begin{bmatrix} 0 & 0 \\ . & . \\ . & . \\ . & . \\ 0 & 0 \end{bmatrix}$$

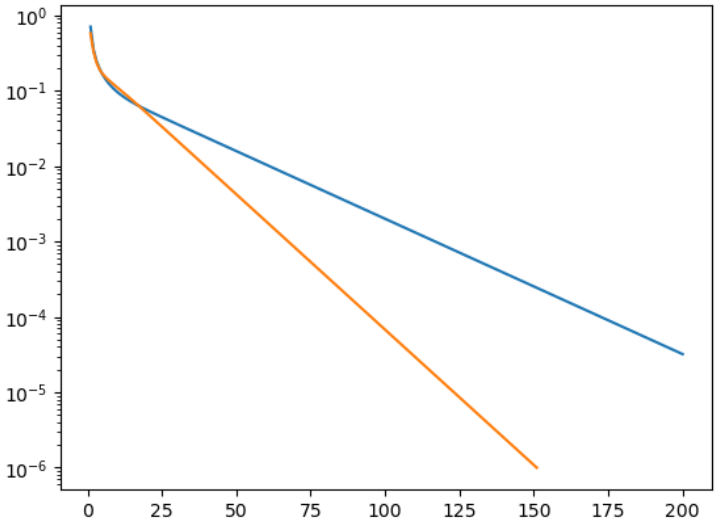
This method is also used by the Lapack library for solving linear systems. This function can also be used to reset the matrices.

2.5 Error

The solution error and the L2 norm of the error of the functions are calculated at iteration 20. And if we change the value of n, we get the following results,

n	Jacobi	Gauss Seidel
10	0.0962991	0.110722
100	0.096299	0.083265
1000	0.096299	0.083265

If we plot the errors and n, with log-log scale, we get the following results.



2.6 Profiling

To measure the time required for the program to execute all the calculations when `niter` is set to 200, we are using the `clock_t` which is the data type used to represent processor time.

n	Time in sec.
10	0.001000
100	0.024000
1000	2.008000

2.7 Output

The output of the program of the matrix X when n = 10 and niter = 30.

Jacobi	Gauss-Seidel
0.962742	0.962742
0.931390	0.931390
0.907964	0.907964
0.893697	0.893697
0.888999	0.888999
0.893485	0.893485
0.906073	0.906073
0.925120	0.925120
0.948602	0.948602
0.974301	0.974301

References

[1] MathWorks: Gauss-Seidel Method, Jacobi Method
`https://www.mathworks.com/matlabcentral/fileexchange/63167-gauss-seidel-method-jacobi-method`.

[2] Chat-GPT: scanf explaination
`https://chat.openai.com/share/14cfe5be-777f-4d0f-bef2-1fa0f1f5f345`.