MN2: Assignment 1

Taras Yarema

November 5, 2020

1 Bound of the norms B_J and B_{GS}

1.1 Norm of the Jacobi method iteration matrix

We know by Proposition 1.2.2 [1] that if A is strictly diagonal dominant by rows, then the Jacobi method is convergent.

We have that the diagonal of A is 5 for all i, and all the other elements are 0, 2 or 1, depending on the row. Note that all the elements are strictly positive, so we assume the absolute value in the following computations. Hence, A is strictly diagonal dominant. Now we compute the norm of B_J as follows

$$||B_J||_\infty = \max_{1\leq i\leq n} \sum_{1\leq j\leq n, j\neq i}^{n} \frac{a_{ij}}{a_{ii}} = \max\left\{\frac{3}{5}, \frac{2}{5}, \frac{4}{5}\right\} = \frac{4}{5} < 1 \Rightarrow ||B_J||_\infty = 0.8.$$

1.2 Norm of the Gauss-Seidel method iteration matrix

We know by Proposition 1.2.3 [1] that if A is strictly diagonal dominant by rows, then the Gauss-Seidel method is convergent.

As in (1.1) we know that A is strictly diagonal dominant by rows. In this method we may compute the norm of B_{GS} as follow

$$||B_{GS}||_\infty = \max_{||x||_\infty = 1} ||B_{GS}x||_\infty$$

As in the proof of Proposition 1.2.3, we can compute s_k and r_k:

$$s_k = \begin{cases} 3/5 & \text{if } k = 1 \\ 2/5 & \text{if } k > 1 \end{cases}, \quad 0 < s_k < 1 \qquad r_k = \begin{cases} 2/5 & \text{if } k < n \\ 3/5 & \text{if } k = n \end{cases}, \quad 0 < r_k < 1$$

Then,

$$||B_{GS}|| \leq \max_{1\leq k\leq n} \frac{r_k}{1 - s_k} < 1 \Rightarrow ||B_{GS}||_\infty < \frac{2}{3}.$$

1.3 Using the norm bounds in the stopping criterion with max error ϵ

As seen before (1.1, 1.2) we found bounds for the iteration matrix B of the Jacobi and Gauss-Seidel methods. In fact, for the Jacobi, the bound is an equality. Now, as we will be using the absolute value stopping criterion, we want to compute the following inequality:

$$\frac{||B||}{1 - ||B||}||\delta^{(k)}|| < \epsilon$$

For the Jacobi method we know that $||B_J|| = \frac{4}{5}$, hence

$$\frac{||B_J||}{1-||B_J||} = \frac{||B||}{1-||B||} \Rightarrow ||\delta_J^{(k)}|| < \frac{\epsilon}{4} = \epsilon_J$$

and for the Gauss-Seidel, knowing that $||B_{GS}|| < \frac{2}{3}$, we have

$$\frac{||B_{GS}||}{1-||B_{GS}||} \leq \frac{||B||}{1-||B||} \Rightarrow \frac{||B_{GS}||}{1-||B_{GS}||}||\delta_{GS}^{(k)}|| < \epsilon \iff ||\delta_{GS}^{(k)}|| < \frac{\epsilon}{2} = \epsilon_{GS}$$

As $\delta^{(k)} = x^{(k+1)} - x^{(k)}$ we will have this value in every iteration, hence we found errors bound ϵ_J and ϵ_{GS} that will impose $||e^{(k+1)}|| < \epsilon$ in both methods.

2 Matrix A manipulation

The matrix A manipulation details are stated in the last section (4).

3 SOR method explanation

We have found that the best value of ω for the SOR method is $\omega^* = 1.22$. To compute this value we firstly looped over all values of ω in the range $(0.1, 2)$ with a step of 0.1. We found that if $\omega \in [0.7, 1.6)$ the method needs less than 100 iterations to end. With this new range we re-computed with a smaller step 0.01 and found that for $\omega = 1.22$ the method needs 37 iterations to end.

As a comparison, we found a $\omega^* = 1.22$ that needs 19 less iterations than the Gauss-Seidel method to solve the system for the given tolerance.

Also, applying Theorem 1.2.3 [1] we have that $\rho(B_{SOR}(\omega^*)) \geq |\omega^* - 1|$, hence the whole spectral radius inequality is $0.22 \leq \rho(B_{SOR}(\omega^*)) < 1$.

4 Special notes and optimizations

We are given a system of the type $Ax = b$ where the matrix A is sparse, i.e. almost all its elements are zero. As stated in the final notes of [1], we do not want to store the whole matrix in memory. In fact we don not even want to store the b array.

As A, and b don't change during the Jacobi, Gauss-Seidel and SOR methods, we will use special utility functions to access the elements of A and b. This way we can highly improve the speed of the iterations by making them $O(n)$ per iteration, as multiplying the iteration matrix in the straightforward $O(n^2)$ is no more needed.

There are a set of functions in the source that given a i and j return the value m_{ij}, where m is the corresponding matrix of the method, hence having $O(1)$ access to the matrices elements. This way, when we compute the matrix multiplication in the iteration methods, we only need to loop over the rows of the independent vector values, which are also computed via a $O(1)$ method, and compute the, at max 2, elements of the iteration matrix. Hence we have linear complexity $O(n)$ for every iteration of every method.

References

[1] Joan Carles Tatjer. *Mètodes Numèrics II.* 2020.