Numerical Methods
Exercise Sheet 6

HS 16
M. Chrzaszcz
D. van Dyk

I. Bezshyiko, A. Pattori
http://www.physik.uzh.ch/en/teaching/PHY233/HS2016.html

Issued: 27.10.2016
Due: 02.11.2016

In this exercise sheet you will be asked to implement different algorithms to approximately solve the linear system $A \cdot \vec{x} = \vec{b}$ using iterative methods. The matrices $A, \vec{b}$ are defined as:

$$A = \begin{bmatrix} 60 & -8 & -4 & 8 & -5 & -9 & -9 & -8 & -8 & 4 \\ -1 & -40 & -5 & 5 & -5 & 9 & 5 & -4 & 4 & 1 \\ 4 & 1 & -30 & -6 & -9 & -6 & 7 & 7 & -3 & 0 \\ -1 & -4 & -3 & -30 & 7 & -7 & -6 & -7 & 7 & 8 \\ -6 & -8 & 4 & -6 & -30 & -2 & 7 & -7 & -3 & -6 \\ -6 & -3 & -4 & -2 & 9 & 30 & -7 & 4 & 1 & 5 \\ 3 & -3 & -4 & -6 & -9 & 4 & 20 & -4 & 8 & -3 \\ 3 & -7 & 5 & -7 & 1 & 6 & 0 & -40 & -3 & -5 \\ 3 & -8 & 9 & -4 & 1 & -7 & 2 & 2 & -70 & -8 \\ 5 & 4 & -6 & 3 & -6 & -2 & 6 & -4 & 8 & -20 \end{bmatrix} ; \quad \vec{b} = \begin{bmatrix} -36 \\ 103 \\ 48 \\ -18 \\ -82 \\ 149 \\ -43 \\ -35 \\ 87 \\ -76 \end{bmatrix} .$$

You can copy a .txt file with the above matrices at: https://git.physik.uzh.ch/gitbucket/pattori/num-meth/blob/master/linear_system_class6.txt.

Exercise 1: Jakobi method (50 Pts.)

a) From A one can immediately define the matrices L, U, D as:

$$L = \begin{cases} A_{ij} & i > j \\ 0 & \text{otherwise} \end{cases} \qquad U = \begin{cases} A_{ij} & i < j \\ 0 & \text{otherwise} \end{cases} \qquad D = \begin{cases} A_{ij} & i = j \\ 0 & \text{otherwise} \end{cases}$$

From such matrices, find the Jakobi matrix: $M_J = -D^{-1}(L + U)$. (15 points)

b) One should in principle check that M_J generates a converging algorithm. The necessary and sufficient condition is $\rho(M_J) < 1$, where $\rho(M_J)$ is the modulus of the biggest eigenvalue of M_J. In this exercise, you can just take as granted that M_J actually satisfy such condition.

However, can you think of any simple necessary condition on M_J that can be easily checked? If yes, implement it. (10 points)

c) The iterative step of the Jakobi method is:

$$\vec{x}^{(k+1)} = M_J \cdot \vec{x}^{(k)} + D^{-1} \cdot \vec{b} \, ,$$

where you can take an arbitrary guess for $\vec{x}^{(0)}$ (as in the Newton-Raphson method). Implement such iteration, and interrupt it when $||\vec{x}^{(k+1)} - \vec{x}^{(k)}|| < \epsilon$. You may take $\epsilon = 10^{-5}$. Find the approximate solution to the given linear system. (15 points)

d) Evaluate the rate of convergence r_J^k:

$$r_J^k = \frac{||\vec{x}^{(k+1)} - \vec{x}^{(k)}||}{||\vec{x}^{(k)} - \vec{x}^{(k-1)}||}$$

and numerically study the limit: $\lim_{k\to\infty} r_J^k$. (10 points)

Exercise 2: Gauss-Seidle method and SOR method (70 Pts.)

a) Implement the iterative step for the Gauss-Seidle method. This simply reads:

$$x_i^{(k+1)} = \frac{1}{A_{ii}} \left(b_i - \sum_{j<i} A_{ij} x_j^{(k+1)} - \sum_{j>i} A_{ij} x_j^{(k)} \right) . \tag{1}$$

(70 points)

b) Start the iteration beginning from an arbitrary guess $\vec{x}^{(0)}$. Interrupt it when $||\vec{x}^{(k+1)} - \vec{x}^{(k)}|| < \epsilon$. You may take $\epsilon = 10^{-5}$. Find the approximate solution to the given linear system.

(10 points)

c) Improve your code to the SOR (Successive Over-Relaxation) method. You just need to substitute Eq. (1) with:

$$x_i^{(k+1)} = x_i^{(k)} + \frac{\omega}{A_{ii}} \left(b_i - \sum_{j<i} A_{ij} x_j^{(k+1)} - \sum_{j\geq i} A_{ij} x_j^{(k)} \right) ,$$

where ω is called relaxation factor. (10 points)

d) Evaluate the rate of convergence r_J^k:

$$r_J^k = \frac{||\vec{x}^{(k+1)} - \vec{x}^{(k)}||}{||\vec{x}^{(k)} - \vec{x}^{(k-1)}||}$$

and numerically study the limit: $\lim_{k\to\infty} r_J^k$. (10 points)

e) Run the SOR algorithm for different values of the relaxation factor $\omega \in [0.2, 1.5]$ (at least 10 different values). Study how the evaluation time change with ω and draw your conclusions.

(20 points)

Maximum number of points for mandatory tasks on 02.11 : 100
Maximum possible number of points for tasks on 02.11 : 120