Middle East Technical University
Department of Mechanical Engineering
ME 310 Numerical Methods
Spring 2023 (Dr. Cuneyt Sert)
Study Set 3

For Homework 3 submit the answers of questions 3, 4 and 5. Their grade percentages are not known at this point. It will be decided later.

Assigned: 17/04/2023 – Due: 29/04/2023, 23:59

Homework Rules and Suggestions:

- This is an individual assignment. Everything in your report should be the result of your own work. You are allowed to discuss its solution with your classmates and teaching staff up to a certain detail on ODTUClass. You are not allowed to use a solution manual. Put the following honor pledge at the top of your homework report and behave accordingly.

  "I understand that this is an individual assignment. I affirm that I have not given or received any unauthorized help on this assignment, and that this work is my own."

  If you have exchanged ideas with other students outside ODTUClass, you need to put their names and the extent of your discussion at the beginning of your report.

- Homework submission will be allowed until 5 minutes past the due time. Late submission is not allowed unless you have a valid excuse. In such a case, it is better if you let us know about this before the submission deadline.

- Upload your report as a PDF document (not a Word document) together with all other files (such as codes) to ODTUClass. Name your MATLAB files properly. Follow MATLAB file naming rules such as "File names cannot start with a number", "They cannot contain special characters or spaces", etc.

- Also put your MATLAB codes in your report, but in doing that make sure that you format them properly to avoid line wrapping. Codes with wrapped long lines become unreadable and we cannot understand them. If the codes are very long, you can shorten them by omitting noncritical parts.

- In writing your codes, follow good programming practices such as "use explanatory header lines", "explain inputs and outputs of functions", "use self-explanatory variable names", "use comments", "use empty lines and spaces for readability", "use indentation for code blocks", etc.

- Pay attention to the format of your report. Your report should look like a serious academic work, not like a high school student work. Font types and sizes, page margins, empty spaces on pages, equations, figures, tables, captions, colors, etc. are all important to give the desired "academic work feeling". Language used is also important. Reports with poor use of English cannot get a good grade.

- Do not provide an unnecessarily long report, with useless details or wasted spaces in pages. The shorter your report, the better it is, as long as it answers the questions properly. There are about 100 students, and we can spend only about 10 minutes to grade each report. For this, your report should be easy to read and understand. Also we should be able to find the results and judge their correctness easily. We should not get lost in your report. The more we struggle to understand your report, the lower your grade will be. Use figures and tables cleverly for this purpose.

- Reports with only figures, tables and codes, but no comments or discussions will not get a good grade. Even when a question does not specifically ask for a discussion, you better write some comments on its key points and your key learnings.

- Figures and tables should be numbered and should have captions (at the bottom for figures and at the top for tables). Their titles should be self-explanatory, i.e., we should be able understand everything about the table or figure just by reading its title. They should all be referred properly in the written text (such as "… as shown in Fig. 3" or "… (See Table 2)").

- Do not use Appendices in your report. Do not put your codes in Appendices.

- You can have a numbered reference list at the end of your report. In that case, you need to refer to the references in the text.

- If you are inexperienced in programming, converting an algorithm into a code and writing it in a bug-free way can be time consuming and frustrating. This is not something that can be done at the last minute. You are advised to start working on the assignments as soon as they are assigned.

Reading Assignments:

Self-learning is an important skill. Not everything can be discussed in lectures. You need to learn certain things by yourself.

For those of you who haven't got a chance to buy the textbook yet, I extracted the pages of the following reading assignments and uploaded them to ODTUClass.

R1) Read section 9.7 Gauss-Jordan (page 277 of 8th edition) to learn this method, which is very similar to Gauss elimination.

R2) Read section 10.3.3 Iterative Refinement (page 301 of 8th edition) to learn about a technique that can sometimes be used to reduce round-off errors of direct methods.

R3) Read section 11.1. Tridiagonal Systems (page 306 of 8th edition) to learn how the tridiagonal systems are solved effectively using a specially designed LU decomposition known as the Thomas algorithm.

R4) Read the Epilogue section of Part 3 (page 347 of 8th edition). Part 3 includes 4 chapters and its epilogue is at the end of Chapter 12. What we call "Chapter 3 Linear Algebraic Equation Systems" in our lectures is this 3rd part of the textbook.

R5) Quoting from the Epilogue of Part 3 of our textbook "Aside from $N \times N$ sets of equations, there are other systems where the number of equations, M, and number of unknowns, N, are not equal. Systems where $M < N$ are called underdetermined. In such cases, there can be either no solution or more than one. Systems where $M > N$ are called overdetermined. For such situations, there is in general no exact solution. However, it is often possible to develop a compromise solution that attempts to determine answers that come closest to satisfying all the equations simultaneously. A common approach is to solve the equation in a least-squares sense. Alternatively, one can use linear programming methods where the equations are solved in an optimal sense by minimizing some objective function".

I believe that you've already studied some of these concepts in ME 210. Review your notes or do a brief reading from other sources. We will briefly discuss such problems in the Optimization chapter, which is the next one.

Questions:

Q1. (This question is not related to numerical methods. It is for those who want to do some extra MATLAB coding for fun. Download Cikolatali_gofret.m code and add the following and similar functionalities to it.

- Generate random target locations each time the space bar is pressed.
- It is a good idea to give the player an audio feedback by playing a sound when a target is reached. Download a simple sound from a web site such as https://opengameart.org , read it into MATLAB using the `audioread` command, and play it using the `sound` command.
- Show the whole path traversed by the tip of the arm after all 3 targets are reached. This will give us an idea about how 'clean' or 'dirty' the arm is moved overall. For this, store the coordinates of the tip during its motion and draw the path at the end.

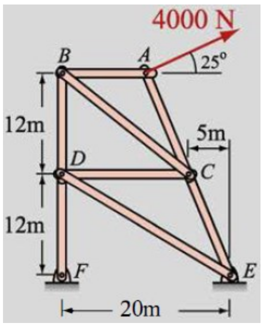
Q2. a) Discuss the advantages and disadvantages of direct and iterative methods used to solve linear algebraic equations.

b) What is matrix decomposition (factorization) and why do we use it for the solution linear algebraic equation systems?

c) Especially for what type of matrices does it become meaningful to use scaling while performing row pivoting?

Q3. Consider the shown truss that is made of 6 joints and 8 members. Each member carries an axial force, such as F_{AB} of the member between joints A and B. The following 8x8 system needs to be solved to determine these forces.

$$\begin{bmatrix} 0 & 0.9231 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -0.3846 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0.8575 & 0 \\ 1 & 0 & -0.7809 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.3846 & -0.7809 & 0 & -1 & 0.3846 & 0 & 0 \\ 0 & 0.9231 & 0.6247 & 0 & 0 & -0.9231 & 0 & 0 \\ 0 & 0 & 0.6247 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -0.5145 & -1 \end{bmatrix} \begin{bmatrix} F_{AB} \\ F_{AC} \\ F_{BC} \\ F_{BD} \\ F_{CD} \\ F_{CE} \\ F_{DE} \\ F_{DF} \end{bmatrix} = \begin{bmatrix} 1690 \\ 3625 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$



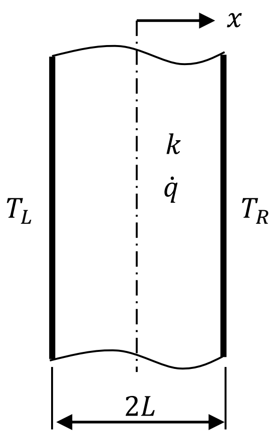
a) Solve the system using the `linsolve` function of MATLAB. Provide the result.

b) Solve the system using the provided NaiveGE.m code. Is the result the same as part (a)?

c) Implement partial row pivoting to the provided NaiveGE.m code. Do NOT use scaling. Solve the system again. Is the result the same as part (a)?

d) Which member of the truss carries the highest tensile force? Which member carries the highest compressive force? Does the result make sense?

Q4. As we've discussed at the beginning of Chapter 3, many engineering problems that are governed by a differential equation can be discretized into a linear algebraic equation system.



Consider the steady, one-dimensional heat conduction over the slab wall shown on the right. Left and right sides of the wall are kept at constant temperatures of T_L and T_R. Thermal conductivity of the wall is k. There is uniform heat generation, $\dot{q}$, inside the wall.

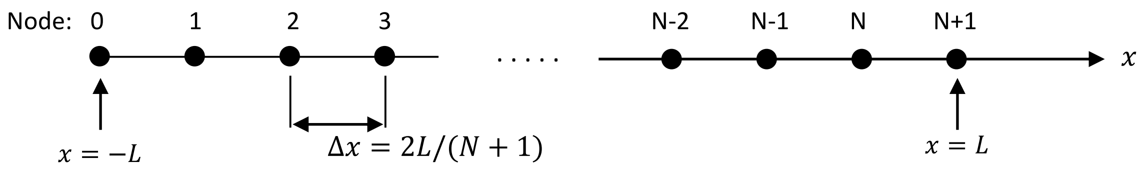
The governing differential equation for the unknown temperature distribution is

$$\frac{d^2T}{dx^2} + \frac{\dot{q}}{k} = 0$$

which has the following analytical (exact) solution

$$T_{exact} = -\frac{\dot{q}}{2k}x^2 + C_1 x + C_2 \quad , \quad C_1 = \frac{T_R - T_L}{2L} \quad \text{and} \quad C_2 = \frac{\dot{q}}{2k}L^2 + \frac{T_L + T_R}{2}$$

We do not need to perform a numerical solution for this simple problem, but let's do it for the sake of practice. For this purpose, we divide the 1D problem domain into a set of discrete nodes of spacing Δx, and we calculate the temperature values at these nodes. Consider the following grid of N+2 nodes.



Temperatures at nodes 0 and N+1 are given as T_L and T_R. To solve for the remaining N unknown temperatures, we need to discretize the governing differential equation like we did in our lectures for the heat transfer over a fin problem. Consider node i with neighbors $i-1$ and $i+1$.

$$\text{Node i:} \quad \left.\frac{d^2T}{dx^2}\right|_i + \frac{\dot{q}}{k} = 0$$

Replace the derivative with an approximation such as the following

$$\text{Node i:} \quad \frac{T_{i-1} - 2T_i + T_{i+1}}{(\Delta x)^2} + \frac{\dot{q}}{k} = 0$$

Arrange the equation

$$\text{Node i:} \quad T_{i-1} - 2T_i + T_{i+1} = -\frac{\dot{q}(\Delta x)^2}{k}$$

Write this equation for all the nodes from $i = 1$ to N to get the following system. Note that for $i = 1$ and $i = N$ one neighbor temperature is known as either T_L or T_R, and this known value is transferred to the right hand side of the system as shown below.

$$\begin{bmatrix} -2 & 1 & & & & & \\ 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & 1 & -2 & 1 \\ & & & & & 1 & -2 \end{bmatrix} \begin{Bmatrix} T_1 \\ T_2 \\ T_3 \\ \cdot \\ \cdot \\ \cdot \\ T_{N-1} \\ T_N \end{Bmatrix} = -\frac{\dot{q}(\Delta x)^2}{k} \begin{Bmatrix} 1 \\ 1 \\ 1 \\ \cdot \\ \cdot \\ \cdot \\ 1 \\ 1 \end{Bmatrix} + \begin{Bmatrix} -T_L \\ 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ 0 \\ -T_R \end{Bmatrix}$$

The empty entries of the coefficient matrix are zero. This is a tridiagonal system.

a) (This part is related to the reading assignment R3) An efficient way of solving tridiagonal systems is the Thomas algorithm, also known as the Tridiagonal Matrix Algorithm (TDMA). It is a special type LU decomposition. Convert its pseudocode given in the textbook into a MATLAB code. Do not store the coefficient matrix as a square matrix, but instead only store the 3 diagonal arrays that have the non-zeros. Use your code to solve the problem and provide the result. Plot the calculated solution and compare it with the exact solution. Use the following parameters.

$$L = 0.1\ \text{m}\ , \qquad T_L = 50\ ℃\ , \qquad T_R = 100\ ℃\ , \qquad k = 80\ \frac{\text{W}}{\text{mK}}\ , \qquad \dot{q} = 4 \times 10^5 \frac{\text{W}}{\text{m}^3}\ , \qquad N = 15$$

b) The coefficient matrix of this problem is symmetric and positive-definite, for which Cholesky decomposition is also a suitable technique. First read Handout 3 and then write a MATLAB code that performs Cholesky decomposition to find a lower triangular matrix L. Use it to perform a forward and a backward substitution to solve the system. Compare the result with that of part (a).

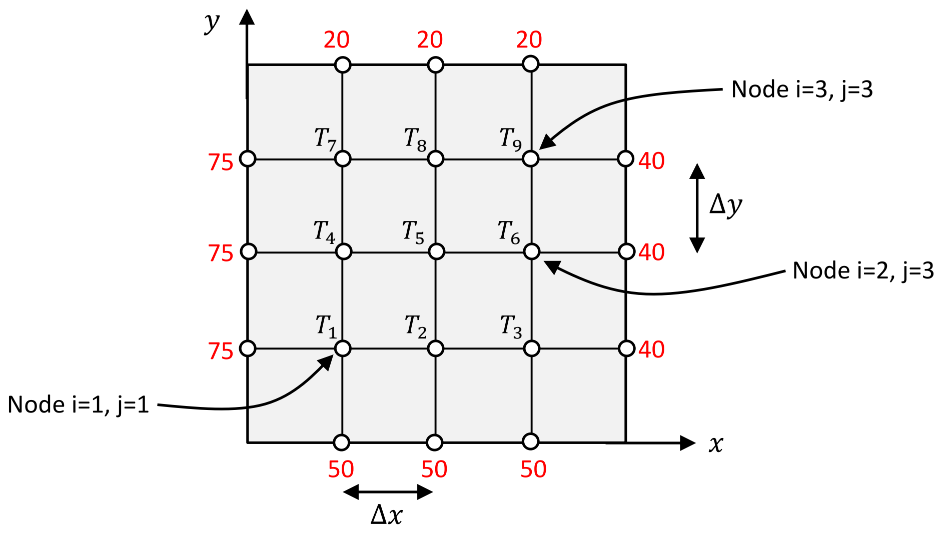
c) Is the equation system of this problem diagonally dominant?

d) Write a MATLAB code that can solve the problem using Gauss-Seidel with relaxation. Solve the system with no relaxation, and with different values of under- and over-relaxation. Use an average temperature of (50+100)/2 = 75 °C for the initial guess at all nodes. Stop the iterations when the Euclidean norm of the residual vector becomes less than 1e-7. Report how the relaxation parameter λ affects convergence. Plot the number of iterations necessary for convergence vs. λ and discuss.

Q5. Consider a 2D version of the previous problem. This time the problem domain is a square plate of size $L \times L$. All four sides of the plate are kept at different temperatures. Without any internal heat generation, the temperature distribution inside the plate is given by the following Laplace equation, which is a partial differential equation.

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0$$

Consider the following discretization of the domain. Boundary temperatures are given in °C in red color.



There are 12 boundary nodes with known temperatures and 9 inner nodes with unknown temperatures. Unknown temperatures can be obtained using the finite difference method, as explained in the previous question. Use the following approximate derivatives to discretize the Laplace equation for any inner node (i, j)

$$\left.\frac{\partial^2 T}{\partial x^2}\right|_{i,j} \cong \frac{T_{i-1,j} - 2T_{i,j} + T_{i+1,j}}{(\Delta x)^2} \quad , \qquad \left.\frac{\partial^2 T}{\partial y^2}\right|_{i,j} \cong \frac{T_{i,j-1} - 2T_{i,j} + T_{i,j+1}}{(\Delta y)^2}$$

Using these in the Laplace equation will result in an equation with 5 unknowns; $T_{i,j}, T_{i+1,j}, T_{i-1,j}, T_{i,j-1}, T_{i,j+1}$. For nodes next to the boundary, one of these will be known and need to be transferred to the right hand side of the equation.

a) Construct the 9x9 system that can be used to solve for the 9 unknown temperatures. Use $L = 1$ m and $\Delta x = \Delta y = 1/4$. Provide $[A]$ and $\{b\}$.

b) Solve the 9x9 system and obtain the temperature values using any method you want. You can use your own codes or one of MATLAB's built-in functions. Provide the result. Does the result make sense?

c) Is there an analytical solution for this problem?

Q6. Coefficient matrices of many engineering problems turn out to be sparse, such as the matrices of the previous two questions. As explained in Handout 5, MATLAB has a command called `gallery` that can be used to create well-known test matrices. For example, matrices of the previous two questions can be generated as follows.

`A = gallery('tridiagonal', 15)`	Creates the 15x15 coefficient matrix of Q4.
`A = gallery('poisson', 3)`	Creates the 9x9 coefficient matrix of Q5.

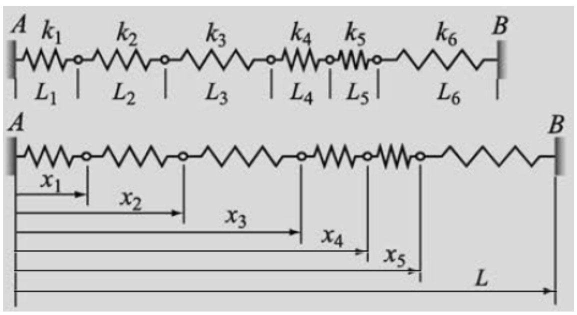
These matrices are created and stored in a special sparse format. Read Handout 5 for details. The amount of sparsity and the importance of sparse storage become more important as the system size increases. Direct methods need to be used carefully with these matrices because they spoil the sparsity structure and result in lower or upper triangular matrices that are not as sparse as matrix A. This is not desired, due to both storage and solution time. Specially designed factorization methods that do not the spoil the sparsity need to be used for such cases.

a) Generate the 10000 x 10000 Poisson matrix (use 100, not 10000, in `gallery` command). Use MATLAB's nnz command to find out the number of non-zero entries of it.

b) Use MATLAB's `chol` command to perform Cholesky factorization and determine the corresponding lower triangular matrix L. Determine the number of its non-zeros. Discuss. If you want, you can perform this test with larger matrices.

c) There is a special Cholesky factorization known as incomplete Cholesky. It determines an approximate L matrix that has the same sparsity pattern as A. In other words, when a non-zero L entry is calculated at a location where A has a zero, that entry is discarded and not stored. Use MATLAB's `ichol` command to perform incomplete Cholesky factorization and determine the corresponding lower triangular matrix L. Determine the number of its non-zeros and compare it with the values obtained in parts (a) and (b). Discuss.

Q7. Six springs with different spring constants k_i and un-stretched lengths L_i are attached to each other in series. The endpoint B is then displaced such that the distance between points A and B is L = 1.5 m. Following tridiagonal system needs to be solved to determine the positions $x_1, x_2, \ldots, x_5$ of the endpoints of the springs.



$$\begin{bmatrix} k_1+k_2 & -k_2 & 0 & 0 & 0 \\ -k_2 & k_2+k_3 & -k_3 & 0 & 0 \\ 0 & -k_3 & k_3+k_4 & -k_4 & 0 \\ 0 & 0 & -k_4 & k_4+k_5 & -k_5 \\ 0 & 0 & 0 & -k_5 & k_5+k_6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} k_1L_1 - k_2L_2 \\ k_2L_2 - k_3L_3 \\ k_3L_3 - k_4L_4 \\ k_4L_4 - k_5L_5 \\ k_5L_5 + k_6L - k_6L_6 \end{bmatrix}$$

Solve this system using the following values

Spring no:	1	2	3	4	5	6
k (kN/m):	8	9	15	12	10	18
L (m):	0.18	0.22	0.26	0.19	0.15	0.30