Aschaffenburg University of Applied Sciences
Faculty for Engineering
Prof. Dr. Michael Möckel

Email: Michael.Moeckel@th-ab.de
Telephone: +49 (0)6021 / 4206 - 507

Numerical Methods and Simulation
Module MT 03
Winter term 2022/23

`https://moodle.th-ab.de/course/view.php?id=624`

Problem Set 5

Solving systems of coupled linear equations

Coursework

Exercise 1: Gaussian elimination algorithm (without pivoting)

The Gauss elimination algorithm is a method for solving a system of linear equations. The linear equations are subjected to equivalence transformations, i.e. they are manipulated by adding them to each other and multiplying them with scalar values. The first equation remains unchanged. Then the factors are chosen such that in every row $i > 1$ another variable x_{i-1} gets eliminated from the i^{th} equation. This results in an equivalent system of linear equations which is represented by an upper triangular matrix and can be solved by using back substitution: The last row then reads $a_{nn}x_n = b_n$ and can be solved easily. With x_n at hand, the second last row can be solved for x_{n-1} and so forth (backward substitution).

(1) Perform the Gaussian algorithm by hand to solve the system of coupled linear equations $A\vec{x} = \vec{b}$

with the coefficient matrix $$A = \begin{pmatrix} 3 & 0 & 3 \\ 0 & -1 & 3 \\ 1 & 3 & 0 \end{pmatrix} \quad \text{and} \quad \vec{b} = \begin{pmatrix} 12 \\ 7 \\ 7 \end{pmatrix}$$

How many solutions exist?

(2) Read and run the following Matlab code for a very basic version of the Gaussian elimination algorithm. Delete the semicolons inside the loop to study the stepwise operation.

Code for naive Gaussian elimination

Let A and b be an example matrix and an example vector. For this version of the Gaussian elimination to work A must not contain zero or small elements on its diagonal.

```
function [ Ab ] = Naive_Gaussian( A,b )

   [m,n] = size(A);
   if m ~= n
      error('Matrix A must be square');
   end
   if length(b)~= n
      error('Matrix and vector must have same number of rows');
   end

   Ab = [A b];   % Augmented coefficient matrix for equivalence transforms

   % Forward elimination
   for i = 1: n-1                      % all rows treated
       for j = i+1:n                   % all elements below diagonal in ith column set 0
           l = Ab(j,i) / Ab(i,i);      % No pivoting (see Exercise 4)
           Ab(j, :) = Ab(j,:) - l*Ab(i,:);
       end
   end
end
```

Exercise 2: Forward and backward substitution

Upper (lower) triangular matrices are characterized by vanishing matrix elements below (above) the matrix diagonal. If a system of linear equations can be represented by an upper (lower) triangular matrix a simple backward (forward) substitution scheme can be applied to determine the solution.

(1) Write a MATLAB routine which performs backward substitution for upper triangular matrices of arbitrary quadratic shape (n,n), e.g.

$$\begin{pmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

(2) Write a MATLAB routine which performs forward substitution for lower triangular matrices.

$$\begin{pmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

(3) Determine the numerical complexity of forward (backward) substitution by counting the number of divisions, additions and multiplications w.r.t. the size of the square matrix U or L.

(4) Integrate the routine into naive Gaussian elimination code from Exercise 1.

Exercise 3: Solving linear equations in MATLAB

MATLAB has a very powerful built-in routine for solving systems of coupled linear equations. It is motivated by multiplying the linear system with the inverse of its coefficient matrix from left $A\vec{x} = \vec{b} \mid \cdot A^{-1}$ such that $A^{-1}A\vec{x} = E_n\vec{x} = \vec{x} = A^{-1} \cdot \vec{b}$ This operation is denoted with the 'left division operator' \ for the unmodified matrix A and vector b. While mathematically this operation only works if the inverse matrix exists, the MATLAB command calls a more general method for solving coupled linear equations.

(1) Call `x = A \ b` for matrix A and $\vec{b}$ in Exercise 1.

(2) Check the command also for the linear system $T\vec{x} = \vec{b}$ with matrix `T = [ 1 2 3 ; 4 6 9]` and vector `b = [1;1]`. Check the result. What does it mean?

Exercise 4: Pivoting

Using the Gauss algorithm may lead to the problem that diagonal matrix elements become zero. Then the Gauss algorithm cannot be continued naively.

(1) Try to solve an arbitrary system of linear equations with the following coefficient matrix by hand and using the naive Gauss algorithm:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 8 \\ -1 & 2 & -2 \end{pmatrix}$$

(2) A solution of the problem is given by swapping lines such that the maximum absolute value of the currently treated column is on the diagonal. These additional manipulations of the matrix A need to be applied directly to the vector $\vec{b}$ or recorded in an permutation matrix P.

(3) Extend the naive Gaussian elimination code from Exercise 1 by a suitable line swapping routine.