Prof. Ch. Schwab
B. Fitzpatrick
J. Zech

Spring Term 2016

Numerische Mathematik I

ETH Zürich
D-MATH

Homework Problem Sheet 2

For some problems, parts of the solution are already given. Fill in the gaps and complete the proofs where you see a red band at the left margin.
Introduction. Landau notation, LU-decomposition

Problem 2.1 Landau-Notation

For the following exercises, use the definition in [NMI, Ch. 1.6].

(2.1a) For $f_i(x) = \mathcal{O}(g_i(x))$, $g_i(x) > 0$, $i = 1, 2$ and $x \to a$, $a \in \mathbb{R} \cup \{\pm\infty\}$, prove the following two rules:

$$f_1(x) + f_2(x) = \mathcal{O}(g_1(x) + g_2(x)) \tag{2.1.1}$$

$$\text{and} \quad f_1(x) f_2(x) = \mathcal{O}(g_1(x) g_2(x)). \tag{2.1.2}$$

(2.1b) Prove that for $s \geqslant 0$ and $n \to \infty$, we have $n! n^s = o(n^n)$.

HINT: Use the inequality

$$\sum_{k=1}^{n} \log k \leqslant n \log \frac{n+1}{2}, \tag{2.1.3}$$

a result from Jensen's inequality.

(2.1c) We always consider $n \to \infty$. Prove the following statements:

(i) $2^n = \mathcal{O}(3^{n-17})$ but $3^{n-17} \neq \mathcal{O}(2^n)$.

(ii) For all $\epsilon > 0$, we have $2^{n+\epsilon} = \mathcal{O}(2^n)$, but $2^{n(1+\epsilon)} \neq \mathcal{O}(2^n)$.

(iii) For all $\epsilon > 0$, we have $\log(2^{n(1+\epsilon)}) = \mathcal{O}(\log(2^n))$.

Problem 2.2 Forward and Backward Error of the LU-Decomposition

(2.2a) Write two MATLAB functions `forwardsub(A,b)` and `backwardsub(A,b)` that perform forward- and backward-substitution following [NMI, Alg. 2.1] and [NMI, Alg. 2.2] for a lower and an upper triangular matrix $\mathbf{A}$, respectively, and a vector $\mathbf{b}$, such that the output solves $\mathbf{Ax} = \mathbf{b}$.

(2.2b) Write a MATLAB function `lrsolve(A, b)` that solves a linear system $\mathbf{Ax} = \mathbf{b}$ via a LU-decomposition without pivoting. Use your functions from (2.2a) and the LU-decomposition `lr(A)` from the course website.

(2.2c) Write a MATLAB function `estimateBError(A)` that calculates the backward error of the LU-decomposition of a matrix $\mathbf{A}$ in the 2-norm with the help of

$$\|\Delta\mathbf{A}\|_2 \leqslant n(3\gamma_n + \gamma_n^2)\big\||\widehat{\mathbf{L}}||\widehat{\mathbf{U}}|\big\|_2 \quad \text{(compare [NMI, Thm. 2.15] and its norm representation).}$$

Use the *unit roundoff* $u = u(\mathbb{F})$ for *double* floating point numbers.

(2.2d) Implement a MATLAB function `calcMinBError(A, b)` that calculates the minimal possible backward error for the system $\mathbf{A}\mathbf{x} = \mathbf{b}$ using the residuum and following [NMI, Thm. 2.17].

(2.2e) Write a MATLAB script that plots the minimum and the estimate of the backward error in a logarithmic diagram dependent on the size n of the matrix, where $n \in \{4, \ldots, 20\}$. For this, let $\mathbf{A}$ be the Hilbert-Matrix of size n (MATLAB function `hilb(n)`) and let the right side $\mathbf{b}$ of the system be a vector with all entries equal to 1 (MATLAB function `ones(n,1)`).

In the same diagram, plot the exact forward error $\|\mathbf{x} - \hat{\mathbf{x}}\|_2 / \|\mathbf{x}\|_2$ that is given by the (provided) function `calcFError(n)`.

Compare the behaviour of the backward and the forward error. What do the curves imply for the accuracy of the calculated solution $\hat{\mathbf{x}}$ of $\mathbf{A}\mathbf{x} = \mathbf{b}$ and the product $\mathbf{A}\hat{\mathbf{x}}$?

Problem 2.3 LU-Decomposition

(2.3a) For the lower triangular matrices $\mathbf{L}_k \in \mathbb{R}^{n\times n}$, $k = 1, \ldots, n-1$, from [NMI, Eq. (2.5)], prove the following properties:

(i) $\mathbf{L}_k^{-1}$ is given by [NMI, Eq. (2.7)].

(ii) $\mathbf{L} = \mathbf{L}_1^{-1}\mathbf{L}_2^{-1} \cdot \ldots \cdot \mathbf{L}_{n-1}^{-1}$ is given by [NMI, Eq. (2.8)].

Solution:

(i) We write $\mathbf{L}_k$ and $\mathbf{L}_k^{-1}$ as

$$\mathbf{L}_k = \mathbf{I} - \mathbf{u}_k \cdot \mathbf{e}_k^\top, \quad \text{and} \quad \mathbf{L}_k^{-1} = \mathbf{I} + \mathbf{u}_k \cdot \mathbf{e}_k^\top,$$

where $\mathbf{u}_k = (0, \ldots, 0, l_{k+1,k}, \ldots, l_{n,k})^\top$ and $\mathbf{e}_k$ is the k^{th} unit vector. Their product gives

(ii) For the second property, we proceed as before:

(2.3b) Prove that the algorithm for the LU-decomposition without pivoting does not terminate for strictly row diagonally dominant matrices $\mathbf{A} \in \mathbb{R}^{n \times n}$.

HINT: A matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is said to be strictly row diagonally dominant if $|a_{ii}| > \sum_{j=1, j \neq i}^{n} |a_{ij}|$ for all $i = 1, \ldots, n$.

Solution: Let $\mathbf{A} := (a_{ij})_{i,j=1}^{n} \in \mathbb{C}^{n \times n}$ be strictly row diagonally dominant. We do the first step of the algorithm for the LU-decomposition without pivoting ([NMI, Alg. 2.8]):

$$\begin{pmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \ldots & a_{nn} \end{pmatrix} \rightsquigarrow \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ 0 & \tilde{a}_{22} & \ldots & \tilde{a}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \tilde{a}_{n2} & \ldots & \tilde{a}_{nn} \end{pmatrix}$$

We want to show that the resulting matrix $\tilde{\mathbf{A}} = (\tilde{a}_{ij})_{i,j=2}^{n}$ is strictly row diagonally dominant.

(2.3c) Given the matrix $\mathbf{A} \in \mathbb{R}^{4,4}$,

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 3 & 1 \\ 3 & 1 & 2 & 2 \\ 6 & 2 & 4 & 3 \\ 1 & 3 & 2 & 1 \end{pmatrix}$$

prove that the algorithm for the LU-decomposition without pivoting [NMI, Alg. 2.8] terminates at the 3rd elimination step.

Solution: First note that $\mathbf{A}$ is regular. In view of [NMI, Thm. 2.9], we need to check whether its leading principal minors $\mathbf{S}_k$ for $k = 1, \ldots, 3$ are regular.

Problem 2.4 Completing the LU-Decomposition

Fill in the template of the MATLAB function `lrsolve(A,b)` from the course website which calculates solutions of $\mathbf{Ax} = \mathbf{b}$ by an $\mathbf{LU}$ decomposition and subsequent backward substitution.

Solution: The template is given in Listing 2.1.

Listing 2.1: Solving $\mathbf{Ax} = \mathbf{b}$ by an $\mathbf{LU}$ decomposition

```
function x = lrsolve(A, b)
% Given a matrix  A  and a column vector  b, the function
%   constructs matrices  L, R such that
%       L is lower triangular
%       R is upper triangular
%       A = L * R  (up to roundoff)
%       diag(L) = [1; 1; ...; 1]
%       L, R have minimal generic size
%   and returns an approximate solution x to A x = b
%   using  L, R  by backsubstitution
%
%   Author:
%   Date:

```

```matlab
% Check if A is square
assert(all(size(A) == size(A')));
n = size(A, 2);

% Make default vector x and matrices L, R
L = eye(size(A));
R = A;
x = zeros(n, 1);

% ...
% TODO: Compute  L, R and c = L\b  explicitly
% ...

% Check integrity of the LR decomposition
assert(TestLR(A, L, R));
% and that indeed  c = L\b
assert(norm(L*c - b, 'inf') <= 1e-10 * norm(b,'inf'));

% ...
% TODO: Compute x=R\c by backsubstitution
% ...

% Check integrity of the solution
if (norm(A*x-b, 'inf') > 1e-9 * norm(b,'inf'))
    warning('Solution tolerance not met');
end
end

function ok = TestLR(A, L, R)
    ok = false;
    if (~all(all(L == tril(L))))
        warning('L must be lower triangular');
    elseif (max(abs(diag(L) - 1)))
        warning('L(i,i) must be all ones');
    elseif (~all(all(R == triu(R))))
        warning('R must be upper triangular');
    elseif ((norm(A-L*R, 'inf') > 1e-8 * norm(A, 'inf')))
        warning('L*R must approximate A');
    else
        ok = true;
    end
end
```

Explicit computation of L, R and $L^{-1}b$:

Compute $R^{-1}c$:

Problem 2.5 Solving A System of Linear Equations with Rounding

In [NMI, Sect. 2.5] it was demonstrated that roundoff can cause instability of Gaussian elimination, unless a suitable pivot policy is implemented. This problem examines this effect in detail for a small example, similar to [NMI, Ex. 2.13] and [NMI, Ex. 2.25]. You are advised to study these examples again before tackling this problem.

Let $\mathbf{A} \in \mathbb{R}^{2\times 2}$ and $\mathbf{b} \in \mathbb{R}^2$ be given by

$$\mathbf{A} = \begin{pmatrix} 0.005 & 1 \\ 1 & 1 \end{pmatrix} \quad \text{and} \quad \mathbf{b} = \begin{pmatrix} 0.5 \\ 1 \end{pmatrix}.$$

We will solve the system $\mathbf{A}\mathbf{x} = \mathbf{b}$ for $\mathbf{x}$ using Gaussian elimination in different ways:

(2.5a) Without rounding errors.

Solution:

By backward substitution, we see that the exact solution $\mathbf{x}$ has components

$$x_2 = \frac{99}{199} \approx 0.497487 \quad \text{and} \quad x_1 = \frac{1}{5 \cdot 10^{-3}} \left(0.5 - \frac{99}{199} \right) = \frac{100}{199} \approx 0.502512. \qquad (2.5.1)$$

(2.5b) Without pivoting, i.e. without interchanging rows or columns, in the floating-point arithmetic $\mathbb{F}(10, 3, -10, 10)$ up to three significant digits.

Solution:

Now, by backward substitution we obtain the solution $\mathbf{x}$ whose components are

$$x_2 = 0.49748... \approx 4.97 \cdot 10^{-1} = 0.497 \quad \text{and} \quad x_1 = 2 \cdot 10^2 \cdot (0.5 - 0.497) = 0.6. \qquad (2.5.2)$$

(2.5c) With pivoting in the floating-point arithmetic $\mathbb{F}(10, 3, -10, 10)$.

Solution:

Now, we can again directly compute the solutions and round to the demanded three significant digits to obtain

$$x_2 = 4.95 \cdot 10^{-1} / 9.95 \cdot 10^{-1} = 0.49748 \ldots \approx 0.497 \quad \text{and} \quad x_1 = 1 - 0.497 = 0.503. \quad (2.5.4)$$

(2.5d) Compare and comment on the above results.

Remark: Calculations in floating-point arithmetic $\mathbb{F}$ are meant as follows: the results of elementary operations from $\{+, -, \cdot, /\}$ are calculated exactly but rounded to a number in $\mathbb{F}$ before being used for further calculations, see [NMI, Ch. 1].

Solution: Comparing the respective solutions in (2.5.1), (2.5.2) and (2.5.4), we see that there are substantial differences in the calculated values for x_1 and x_2.

Published on March 3, 2016.

To be submitted on March 15, 2016.

MATLAB: Submit all files in the online system. Include the files that generate the plots. Label all your plots. Include commands to run your functions. Comment on your results.

References

[NMI] Lecture Notes for the course "Numerische Mathematik I".

Last modified on March 8, 2016