R. Hiptmair
S. Hosseini
C. Pagliantini

Spring Term 2014

Numerical Analysis I

ETH Zürich
D-MATH

Homework Problem Sheet 3

Introduction. This problem sheet is devoted to the solution of linear systems of equations and the concept of the condition number of a matrix.

Problem 3.1 Properties of the Matrix Condition

In this problem we study some properties of the p-condition number $\kappa_p(\mathbf{A}) := \|\mathbf{A}\|_p\|\mathbf{A}^{-1}\|_p$ for $1 \leqslant p \leqslant \infty$ of matrices $\mathbf{A} \in \mathbb{R}^{n\times n}$, that is the condition number of a matrix based on the matrix norm induced by the ℓ^p vector norm as defined in [NMI, Eq. (0.6)].

(3.1a) Refresh you knowledge of vector norms and matrix norms as introduced in [NMI, Sect. 0.6] and [NMI, Sect. 0.7].

(3.1b) Prove that for a regular matrix $\mathbf{A}$, the p-condition number $\kappa_p(\mathbf{A})$ can be written as

$$\kappa_p(\mathbf{A}) = \frac{\max_{\|\mathbf{x}\|_p=1} \|\mathbf{A}\mathbf{x}\|_p}{\min_{\|\mathbf{x}\|_p=1}\|\mathbf{A}\mathbf{x}\|_p}. \tag{3.1.1}$$

(3.1c) Appealing to Equation 3.1.1, show that for the 2-condition number of a symmetric positive definite matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, we have

$$\kappa_2(\mathbf{A}) = \frac{\lambda_{\max}(\mathbf{A})}{\lambda_{\min}(\mathbf{A})}, \tag{3.1.2}$$

where $\lambda_{\max}(\mathbf{A})$ and $\lambda_{\min}(\mathbf{A})$ are the largest and the smallest eigenvalues of $\mathbf{A}$.

HINT: The proof is easy when you rely on the spectral decomposition theorem for symmetric matrices [NMI, Corollary 0.61].

(3.1d) Now show that for the 2-condition number of a regular, symmetric but not necessarily positive definite matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, we have

$$\kappa_2(\mathbf{A}) = \frac{|\lambda|_{\max}(\mathbf{A})}{|\lambda|_{\min}(\mathbf{A})},$$

where $|\lambda|_{\max}(\mathbf{A})$ and $|\lambda|_{\min}(\mathbf{A})$ are the maximum and minimum of the absolute values of the eigenvalues of $\mathbf{A}$.

(3.1e) Assume that $\mathbf{A} \in \mathbb{R}^{n\times n}$ is regular. Use subproblem (3.1b) to show that the following statements hold true:

(i) $\kappa_p(\mathbf{A}) \geq 1$, for $1 \leqslant p \leqslant \infty$.

(ii) For all $\alpha \in \mathbb{R} \setminus \{0\}$: $\kappa_p(\alpha\mathbf{A}) = \kappa_p(\mathbf{A})$.

Problem 3.2 Partitioned Matrix

In this problem we examine the LU-decomposition of a matrix with a special structure, sometimes called "arrow matrix". We will see that the LU-factors for such matrices are easily determined; it is possible to "see" the LU-decomposition thanks to a consideration elaborated in the first sub-problem.

(3.2a) Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be a matrix with LU-decomposition $\mathbf{A} = \mathbf{LU}$. Show that, with MATLAB notation used to designated sub-matrices, the LU-factors of $\mathbf{A}(1:l, 1:l)$ are $\mathbf{L}(1:l, 1:l)$ and $\mathbf{U}(1:l, 1:l)$ for all $l \in \{1, \ldots, n\}$.

For the remainder of the problem, let the matrix $\mathbf{A} \in \mathbb{R}^{n+1,n+1}$ possess the following block structure:

$$\mathbf{A} = \begin{pmatrix} \mathbf{R} & \mathbf{v} \\ \mathbf{u}^\top & 0 \end{pmatrix}, \tag{3.2.1}$$

where $\mathbf{v} \in \mathbb{R}^n$, $\mathbf{u} \in \mathbb{R}^n$, and $\mathbf{R} \in \mathbb{R}^{n\times n}$ is upper triangular and regular.

(3.2b) Find the LU-decomposition of $\mathbf{A}$.

(3.2c) Show that $\mathbf{A}$ is regular if and only if $\mathbf{u}^\top\mathbf{R}^{-1}\mathbf{v} \neq 0$.

(3.2d) Implement an *efficient* MATLAB function `x = solvelse(R,v,u,b)` that solves the linear system of equations $\mathbf{Ax} = \mathbf{b}$ with $\mathbf{A}$ from (3.2.1), using *only* elementary operations on matrices and vectors (no \-operator!). The arguments correspond to the matrices and vectors in (3.2.1) in an natural way.

In the problem statement, "efficient" means that your implementation should achieve an asymptotic computational effort $O(n^p)$ for $n \to \infty$ with the best possible power p.

HINT: Use the decomposition from subproblem (3.2b) and remember that $\mathbf{R}$ is upper triangular.

(3.2e) How many elementary floating point operations does your implementation of of `solvelse` take depending on n?

Problem 3.3 Gaussian Elimination with Pivoting for Structured Sparse Linear System

If the coefficient matrix of an $n \times n$ linear system of equations has special properties, Gaussian elimination can often be carried out with a computational effort that is much lower that the $\frac{2}{3}n^3 + \mathcal{O}(n^2)$ as counted in [NMI, Sect. 2.2]. In particular, we can usually benefit from *sparsity* of the matrix, that is, the property that only $\mathcal{O}(n)$ of its entries do not vanish. We saw this in the case of banded matrices in [NMI, Sect. 2.8] and in this problem we will come across banded matrices in disguise.

We consider a block partitioned linear system of equations

$$\mathbf{Ax} = \mathbf{b} \quad , \quad \mathbf{A} := \begin{pmatrix} \mathbf{D}_1 & \mathbf{C} \\ \mathbf{C} & \mathbf{D}_2 \end{pmatrix} \in \mathbb{R}^{2n\times 2n}, \tag{3.3.1}$$

where all the $n \times n$-matrices $\mathbf{D}_1$, $\mathbf{D}_2$, and $\mathbf{C}$ are diagonal. Hence, the matrix $\mathbf{A}$ can be described through three n-vectors $\mathbf{d}_1$, $\mathbf{c}$, and $\mathbf{d}_2$, which provide the diagonals of the matrix blocks. These vectors will be passed as arguments `d1`, `c`, and `d2` to the MATLAB codes below.

(3.3a) Write an *efficient* MATLAB function `function y = multA(d1,c,d2,x)` that returns $\mathbf{y} := \mathbf{A}\mathbf{x}$ in the column vector `y`. The argument `x` passes a column vector $\mathbf{x} \in \mathbb{R}^{2n}$.

(3.3b) Count the number of elementary floating point operations for a call to `multA` as you have implemented it in sub-problem (3.3a).

(3.3c) Assuming that the LU-decomposition of $\mathbf{A}$ exists, compute the matrices $\mathbf{L}$ and $\mathbf{U}$ of the factorization.

HINT: Remember Problem 3.2.

(3.3d) Write an *efficient* MATLAB function

```
function x = solveA(d1,c,d2,b)
```

that solves $\mathbf{A}\mathbf{x} = \mathbf{b}$ with Gaussian elimination *with partial pivoting* (see [NMI, Sect. 2.5]). The MATLAB \-operator must not be used.

HINT: Test your code with arguments `d1 = (1:n)', d2 = -d1, c = ones(n,1), b = [d1;d1]`. Compare with reference solution obtained in MATLAB by `x = A \ b`.

(3.3e) Analyze the asymptotic computational effort of your implementation of `solveA` in terms of the problem size parameter $n \to \infty$?

(3.3f) Runtime measurements in MATLAB can be carried out by means of the commands `tic` and `toc`. Please study their documentation.

(3.3g) Determine the asymptotic complexity of the algorithm `solveA` in a numerical experiment using the MATLAB routines `tic` and `toc`. As test case use `d1 = (1:n)', d2 = -d1, c = ones(n,1), b = [d1;d1]` for $n = 2^2, \ldots, 2^{12}$. Create a suitable plot of the runtime versus n that allows to read off the asymptotic complexity.

(3.3h) Find a permutation matrix $\mathbf{P} \in \mathbb{R}^{n\times n}$ such that $\mathbf{P}\mathbf{A}\mathbf{P}^\top$ becomes a banded matrix (see [NMI, Def. 2.40]) with bandwidth 3.

Problem 3.4 Lyapunov Equation

Any linear equation with a finite number of unknowns can be written in the "canonical form" $\mathbf{A}\mathbf{x} = \mathbf{b}$. However, it may be given in a different form and it may not be obvious how to extract the system matrix. This task gives an intriguing example and also presents an important matrix equation, the so-called Lyapunov Equation.

Given $\mathbf{A} \in \mathbb{R}^{n\times n}$, write a MATLAB function `X = solveLyapunov(A)` that determines $\mathbf{X} \in \mathbb{R}^{n\times n}$ such that

$$\mathbf{A}\mathbf{X} + \mathbf{X}\mathbf{A}^\top = \mathbf{I} \tag{3.4.1}$$

HINT: : Identify $\mathbf{X}$ with a vector $\mathbf{x} \in \mathbb{R}^{n^2}$ by stacking its columns on top of each other. Then rewrite (3.4.1) as a linear system of equations with unknown $\mathbf{x}$ and an $n^2 \times n^2$ coefficient matrix. You may solve this system by simply using the \-operator (though this is not the most efficient way to do it). Find out how to use the MATLAB `reshape` and `kron` functions.

Problem 3.5 Cramer's Rule vs. Gaussian Elimination

Define

$$\mathbf{A}(\phi) := \begin{pmatrix} \frac{1}{\sqrt{2}} & \cos\phi \\ \frac{1}{\sqrt{2}} & \sin\phi \end{pmatrix} \quad , \quad \mathbf{b}(\phi) = \mathbf{A}(\phi)\begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

The 2×2 linear system of equations $\mathbf{A}(\phi)\mathbf{x} = \mathbf{b}(\phi)$ with exact solution $\mathbf{x} = (x_1\ x_2)^\top = (1\ 1)^\top$ can be solved in two ways:

1. You may use Gaussian elimination employing the $\backslash$-operator in MATLAB.
2. You may use Cramer's rule, that is $x_i = \det(\mathbf{A}_i)/\det(\mathbf{A})$ where $\mathbf{A}_i$ is obtained by replacing the i-th column of $\mathbf{A}$ by the vector $\mathbf{b}$. For the 2×2 system $\mathbf{A}\mathbf{x} = \mathbf{b}$, one has

$$x_1 = \frac{b_1 a_{22} - b_2 a_{12}}{a_{11}a_{22} - a_{21}a_{12}} \ , \qquad x_2 = \frac{a_{11} b_2 - a_{21} b_1}{a_{11}a_{22} - a_{21}a_{12}} \ . \tag{3.5.1}$$

(3.5a) Implement both solution strategies in MATLAB in order to obtain numerical solutions $\widetilde{\mathbf{x}}(\phi)$ (affected by roundoff) to $\mathbf{A}(\phi)\mathbf{x} = \mathbf{b}(\phi)$ for $\phi = \pi/4 - 10^{-9} : 10^{-11} : \pi/4 + 10^{-9}$. Plot in a suitable scale the condition number of $\mathbf{A}(\phi)$ in the 2-norm, the relative residual

$$\frac{\|\mathbf{b}(\phi) - \mathbf{A}(\phi)\widetilde{\mathbf{x}}(\phi)\|_2}{\|\mathbf{b}(\phi)\|_2}$$

as function of ϕ for the two approaches and the relative (forward) error

$$\frac{\|\widetilde{\mathbf{x}}(\phi) - \mathbf{x}\|_2}{\|\mathbf{x}\|_2}$$

vs. the function ϕ for both strategies.

(3.5b) Write down what you would say, if you had to step in front of the class and explain the observations made in (3.5a). Which solution strategy should be preferred?

Published on March 10, 2014.

To be submitted on March 18/19, 2014.

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.

Written exercises: Hand-in the solutions during the exercise class or in the labeled boxes in HG G 53.x.

References

[NMI] Lecture Slides for the course "Numerical Analysis I".

Last modified on March 10, 2014