Prof. Ch. Schwab
B. Fitzpatrick
J. Zech

Spring Term 2016

Numerische Mathematik I

ETH Zürich
D-MATH

Homework Problem Sheet 0

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. Introduction to MATLAB, norms and topics form Chapter 0.

Problem 0.1 Introduction to MATLAB

On the course website, you find an introduction "Learning MATLAB by doing MATLAB". Work through this manual step by step (no submission).

Problem 0.2 Fibonacci Numbers

The sequence F_n ($n = 1, 2, \dots$) of Fibonacci numbers is defined as follows:

$$F_n = \begin{cases} 1 & \text{if } n = 1 \text{ or } n = 2, \\ F_{n-1} + F_{n-2} & \text{if } n > 2. \end{cases} \tag{0.2.1}$$

(0.2a) Write a MATLAB-function `fibonacci(n)` that takes a positive integer n as an input and returns the Fibonacci numbers $F_1, \dots, F_n$ in a vector. Print the result for $n = 10$.

(0.2b) Write a MATLAB-function `fiboQuot(n)` that takes a positive integer n as an input and returns the quotients $q_k = \frac{F_{k+1}}{F_k}$ for $k = 1, \dots, n$ in a vector. Print the result for $n = 10$.

(0.2c) Use the MATLAB-function `semilogy` and the function `fiboQuot` you came up with in subproblem (0.2b) to plot $|q_k - \phi|$ with $\phi = \frac{1+\sqrt{5}}{2}$ for $k = 1, \dots, N$ as a function of k in a graph with logarithmic y-axis. For this, set $N = 30$. Explain the behaviour of the graph and try to explain what happens for $N = 60$.

HINT: You can find the detailed documentation of `semilogy` in the MATLAB-Documentation or by typing `help semilogy` into the MATLAB-console.

Listing 0.1: Testcalls for Problem 0.2

```
F = fibo(10)

Q = fiboQuot(10)
```

Listing 0.2: Output for Testcalls for Problem 0.2

```
>> test_call

F =
```

```

     1
     3
     4
     7
    11
    18
    29
    47
    76
   123

Q =

    3.0000
    1.3333
    1.7500
    1.5714
    1.6364
    1.6111
    1.6207
    1.6170
    1.6184
    1.6179
```

Problem 0.3 Approximation of π

We discuss different methods to approximate the number π.

(0.3a) The *Monte-Carlo*-method is based on the fact that the area of the unit circle is π.

Write a MATLAB-function `calcPiMC(n)` in which you generate n pairs (x_i, y_i) of uniformly distributed pseudorandom numbers in $[0,1]^2$ using `rand` in a `for` loop. Determine how many pairs (x_i, y_i) lie inside the first quadrant of the unit circle: The ratio of that number by n is an approximation of $\frac{1}{4}\pi$. Let then the resulting approximation of π be the function's return value.

Test your function for $n = 10^i$, $i = 1, \dots, 7$ and list the results.

Now write a vectorized version `calcPiMCVec(n)` of the above function that does not use any explicit loops as e.g. `for`. In addition, write a script `execution_times.m` to measure the time for $n = 10^i$, $i = 1, \dots, 7$ it takes to calculate the approximation of π via `calcPiMC(n)` and `calcPiMCVec(n)`, respectively. Comment on your results.

(0.3b) Partial sums of known infinite sums for π generate approximations of π. For each of the following two sums, write a function `calcPiSumA(n)` and `calcPiSumB(n)`, that calculates an approximation of π on the first n terms of the sums.

$$\text{(A)} \qquad \frac{\pi}{4} = \sum_{k=0}^{\infty} \frac{(-1)^k}{2k+1} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots. \qquad (0.3.1)$$

$$\text{(B)} \qquad \frac{\pi^2 - 8}{16} = \sum_{k=1}^{\infty} \frac{1}{(4k^2-1)^2} = \frac{1}{3} + \frac{1}{15} + \frac{1}{35} + \dots. \qquad (0.3.2)$$

(0.3c) For $n = 10^i$, $i = 1, \dots, 4$, calculate the errors of the approximations comparing your results to the MATLAB-constant `pi` for all three methods. Plot the errors using `loglog` in one log-log diagram.

Which of the discussed methods converges the fastest? Can you analytically explain why?

Listing 0.3: Testcalls for Problem 0.3

```
execution_times
```

Listing 0.4: Output for Testcalls for Problem 0.3

```
>> test_call
i = 1: with for-loop:    Elapsed time is 0.207747 seconds.
i = 1: without for-loop: Elapsed time is 0.019630 seconds.
i = 2: with for-loop:    Elapsed time is 0.000708 seconds.
i = 2: without for-loop: Elapsed time is 0.000122 seconds.
i = 3: with for-loop:    Elapsed time is 0.008695 seconds.
i = 3: without for-loop: Elapsed time is 0.000291 seconds.
i = 4: with for-loop:    Elapsed time is 0.065324 seconds.
i = 4: without for-loop: Elapsed time is 0.001642 seconds.
i = 5: with for-loop:    Elapsed time is 0.661119 seconds.
i = 5: without for-loop: Elapsed time is 0.041397 seconds.
i = 6: with for-loop:    Elapsed time is 5.444969 seconds.
i = 6: without for-loop: Elapsed time is 0.106453 seconds.
i = 7: with for-loop:    Elapsed time is 50.537209 seconds.
i = 7: without for-loop: Elapsed time is 1.565055 seconds.

pi =

   2.000000000000000   3.200000000000000
   3.280000000000000   3.120000000000000
   3.148000000000000   3.080000000000000
   3.137200000000000   3.146800000000000
   3.138680000000000   3.145000000000000
   3.140108000000000   3.141520000000000
   3.141388400000000   3.141259600000000
```

Problem 0.4 L^1-Norm

Denote by $V := C^0([0,1])$ the space of continuous functions $f : [0,1] \to \mathbb{R}$. Recall that every $f \in V$ is Riemann summable on $[0,1]$. For $f \in V$ define via the Riemann integral

$$\|f\|_{L^1} := \int_0^1 |f(x)| \, \mathrm{d}x. \tag{0.4.1}$$

(0.4a) Prove that $\|\cdot\|_{L^1}$ constitutes a well-defined norm on V.

Solution: The map $[0,1] \ni x \mapsto |f(x)| \in \mathbb{R}$ is continuous as a composition of continuous functions. Hence $|f| \in V$ (which implies Riemann summability of $|f|$) and (0.4.1) is well-defined. Let us prove that $\|\cdot\|_{L^1}$ is a norm.

(N1): Let $f \in V$. We have $|f(x)| \geq 0$ for every $x \in [0,1]$ and thus $\|f\|_{L^1} \geq 0$. Let us assume that $\|f\|_{L^1} = 0$ and $f(\tilde{x}) = c \neq 0$ for some fixed $\tilde{x} \in [0,1]$.

(N2): We show $\|\alpha f\|_{L^1} = |\alpha| \|f\|_{L^1}$:

(N3): Finally let us prove the triangle inequality:

(0.4b) Is V complete with respect to this norm? Argue why/why not.

Solution: We show that this is not the case. For $n \in M := \{m \in \mathbb{N} : n \geq 3\}$, consider the functions $f_n : [0, 1] \to \mathbb{R}$ defined by

$$f_n(x) := \begin{cases} 0 & \text{if } x \in [0, \frac{1}{2} - \frac{1}{n}] \\ \frac{xn}{2} - \frac{n-2}{4} & \text{if } x \in [\frac{1}{2} - \frac{1}{n}, \frac{1}{2} + \frac{1}{n}] \\ 1 & \text{if } x \in [\frac{1}{2} + \frac{1}{n}, 1]. \end{cases}$$

Problem 0.5 Matrix p-Norms

(0.5a) Let $\mathbf{A} \in \mathbb{R}^{n\times n}$. Using the definition of the vector p-norm and the induced matrix p-norm, derive the following explicit formulas for $p \in \{1, 2, \infty\}$:

$$\|\mathbf{A}\|_1 = \max_{j=1,\dots,n} \sum_{i=1}^{n} |a_{ij}|, \quad \|\mathbf{A}\|_\infty = \max_{i=1,\dots,n} \sum_{j=1}^{n} |a_{ij}|, \quad \|\mathbf{A}\|_2 = \sqrt{\lambda_{\max}(\mathbf{A}^\top \mathbf{A})},$$

where $\lambda_{\max}(\mathbf{A}^\top\mathbf{A})$ denotes the maximum eigenvalue of $\mathbf{A}^\top\mathbf{A}$.

HINT: For $\|\mathbf{A}\|_2$ use the singular value decomposition of the matrix $\mathbf{A}$.

Solution: Let $1 \leqslant p \leqslant \infty$. The matrix norm induced by the vector p-norm $\|\cdot\|_p$ is defined by

$$\|\mathbf{A}\|_p := \max_{\mathbf{x}\neq\mathbf{0}} \frac{\|\mathbf{A}\mathbf{x}\|_p}{\|\mathbf{x}\|_p} = \max_{\|x\|_p=1} \|\mathbf{A}\mathbf{x}\|_p.$$

Let $p = 1$, we show that

$$\|\mathbf{A}\|_1 = \max_{j=1,\dots,n} \sum_{i=1}^{n} |a_{ij}|. \tag{0.5.1}$$

Let $\mathbf{x} \neq 0$, using the definition of vector 1-norm

$$\frac{\|\mathbf{A}\mathbf{x}\|_1}{\|\mathbf{x}\|_1} = \frac{\left\|\left(\sum_{j=1}^{n} a_{ij}x_j\right)_i\right\|_1}{\|\mathbf{x}\|_1} = \frac{\sum_{i=1}^{n}\left|\sum_{j=1}^{n} a_{ij}x_j\right|}{\sum_{j=1}^{n}|x_j|} \leq \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}|a_{ij}||x_j|}{\sum_{j=1}^{n}|x_j|} \leq \max_{j=1,\dots,n}\sum_{i=1}^{n}|a_{ij}|.$$

Let $S_j := \sum_{i=1}^{n} |a_{ij}|$ and $S_m = \max_{j=1,\dots,n} \sum_{i=1}^{n} |a_{ij}|$ for some $m \in \{1, \dots, n\}$.

Let now $p = \infty$ and take $\mathbf{x} \neq 0$. Using the definition of vector ∞-norm

$$\frac{\|\mathbf{A}\mathbf{x}\|_\infty}{\|\mathbf{x}\|_\infty} = \frac{\max\limits_{i=1,\dots,n} \left|\sum_{j=1}^n a_{ij} x_j\right|}{\max\limits_{j=1,\dots,n} |x_j|} \leq \frac{\max\limits_{i=1,\dots,n} \sum_{j=1}^n |a_{ij}| \max\limits_{j=1,\dots,n} |x_j|}{\max_{j=1}^n |x_j|} \leq \max_{i=1,\dots,n} \sum_{j=1}^n |a_{ij}| =: S_m.$$

For $p = 2$, we consider the singular value decomposition of the matrix $\mathbf{A}$ as given in Theorem 0.19 in the lecture notes, namely $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ with $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{n\times n}$ orthogonal.

(0.5b) Let the matrix $\mathbf{A} \in \mathbb{R}^{2\times 2}$ be defined as

$$\mathbf{A} = \begin{pmatrix} 1 & 2 \\ -1 & 1/2 \end{pmatrix}. \tag{0.5.6}$$

In view of the explicit formulas derived in subproblem (0.5a), compute the norm $\|\mathbf{A}\|_p$ for $p = \{1, 2, \infty\}$.

Solution:

(0.5c) Write a MATLAB function

```
function np = MatrixpNorm(A)
```

which takes as input a matrix A and returns a 3×1-vector np containing the p-norm of A for $p = 1, 2, \infty$. Compare the numerical results with the ones obtained in subproblem (0.5b) for $\mathbf{A}$ as in Equation 0.5.6.

HINT: Use the MATLAB built-in routines and refer to p. 16 of the lecture notes.

Published on February 19, 2016.

References

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

Last modified on February 18, 2016