MTH 373/573: Scientific Computing	Homework 1	IIIT-Delhi, Monsoon 2021 Kaushik Kalyanaraman
Due by: September 2, 2021 (Thursday) by 23:59 IST		Total: 125 points
Late submission by: September 4, 2021 (Saturday) by 23:59 IST		Total: 75 points

Guidelines

- Please consult with and carefully read through **IIIT-Delhi's Academic Dishonest Policy** at this link.
- *This assignment should be answered individually.* You are not allowed to discuss specifics of your solutions, solution strategies, or coding strategies with any other student in this class. There are only two exceptions:
  1. You may post queries relating to any HW question to the `#homeworks` channel of the class's Discord server.
  2. You may reach out directly to either me or one of the Ph.D. TAs – Archana (archanaa@iiitd.ac.in) or Gaurav (gauravr@iiitd.ac.in) with your clearly phrased questions about any HW problem.
- Start working on your solutions early and don't wait until close to due date.
- Each problem should have a clear and concise write-up.
- Please clearly show all steps involved in your solution.
- A late submission can turned within an additional 2 days with grading rescaled to 0.6 of total points.
- You will need to write a separate code for each computational problem and sometimes for some subproblems too. You should name each such file as `problem_`*n*`.py` where *n* is the problem number. For example, your file names could be `problem_5.py` `problem_6a.py`, `problem_6b.py`, and so on.
- *Python tip:* You can import Python modules as follows:

```
import numpy as np
import scipy as sp
import matplotlib.pyplot as plt
import numpy.linalg as npla
import scipy.linalg as spla
```

- You can submit IPython Notebook files `*.ipynb` as well. Please use the same naming convention as above. If you choose to do this, please clean your output and also

provide an exported HTML file. You can go to a Notebook's File menu options and choose `Kernel -> Restart & Run All` for a clean output. For exporting to HTML, you carry out `File -> Download as -> HTML (.html)`.

Problem 1: Condition Number — 5 + 5 = 10 points

Recast the following problems as evaluation problems $y = f(x)$ for the input x, a fixed parameter a and output y, and compute their condition numbers $\kappa(x)$:

(a) $y - a^x = 0, \quad a > 0,$

(b) $x - y + 1 = 0.$

Problem 2: Vector Norms are Lipschitz Continuous — 30 points

Show that any vector norm is Lipschitz continuous, that is, it satisfies the inequality:

$$|\|x\| - \|y\|| \leq \|x - y\|, \qquad \text{for all } x, y \in \mathbb{R}^n.$$

Problem 3: p-norms and Tensor Product — 15 + 5 = 20 points

(a) Let $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$. Define the *tensor product* of x and y to be the following $\mathbb{R}^{mn}$ vector:

$$x \otimes y := \begin{bmatrix} xy_1 \\ \vdots \\ xy_m \end{bmatrix}_{mn}.$$

What is $\|x \otimes y\|_p$ in terms of $\|x\|_p$ and $\|y\|_p$ for each of the cases $p = 1, 2, \infty$? Demonstrate for each case via step-by-step calculations.

(b) Based on your solution, conjecture for the generalization to tensor product of two matrices: if A is a $\mathbb{R}^{m \times n}$ matrix and B is a $\mathbb{R}^{k \times \ell}$ matrix, their tensor product is the following $\mathbb{R}^{mk \times n\ell}$ matrix:

$$A \otimes B := \begin{bmatrix} AB_{11} & \dots & AB_{1\ell} \\ \vdots & \ddots & \vdots \\ AB_{k1} & \dots & AB_{k\ell} \end{bmatrix}.$$

Problem 4: Nilpotent Matrix is Singular — 10 points

Let $A \in \mathbb{R}^{n \times n}$ be a nilpotent matrix with index 2, that is $A^2 = \mathbb{O}$ where $\mathbb{O} \in \mathbb{R}^{n \times n}$ is the zero matrix. Using the definition of singularity in terms of linear independence of columns (rows) of a matrix, show that A is singular.

Problem 5: Condition Number of Matrices — 5 points

Show that for any nonsingular matrices $A, B \in \mathbb{R}^{n\times n}$, their condition number (in any subordinate matrix norm) satisfies: $\kappa(AB) \leq \kappa(A)\,\kappa(B)$.

Problem 6: Exploring IEEE Double Precision using Python — 15 points

(a) What does this code snippet do? Type it in an IPython terminal or Jupyter Notebook. Explain what you observe when you run it and why in no more than 2-3 sentences.

```
a = 1.
while a != 0:
    a /= 2
    print(a)
```

(b) What does this code snippet do? Type it in an IPython terminal or Jupyter Notebook. Explain what you observe when you run it and why in no more than 2-3 sentences.

```
a = 1.; eps = 1.; b = a + eps
while a != b:
    eps /= 2
    b = a + eps
print(eps)
```

(c) What does this code snippet do? Type it in an IPython terminal or Jupyter Notebook. Explain what you observe when you run it and why in no more than 2-3 sentences.

```
a = 1.
while a != inf:
    a *= 2
    print(a)
```

Warning: The decimal point "." after the `1` in the variable `a` is extremely important. Please read Submission Notes (on last page) for some additional information.

Problem 7: Understanding Round off Error — 20 + 15 = 35 points

(a) Recall that the number e is defined by $\lim_{n\to\infty}\left(1+\frac{1}{n}\right)^n$. Estimate the value of e by writing a Python code to compute the expression $\left(1+\frac{1}{n}\right)^n$ for $n = 10^k$, $k = 1, 2, \ldots, 15$. Using `plt.plot` from `matplotlib` (see Python tip in Guidelines on first page), plot the relative error in this computation for each value of k. For what value of k do you get the most accurate estimate for e using this limit formula? Why does this not match with

the mathematical expectation that this formula should become increasingly more accurate as n increases? Reason in terms of floating point arithmetic and its relation with machine epsilon ε_M.

(b) Now implement the Taylor series computation for e:

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \cdots.$$

First, using your understanding of ε_M, what stopping criterion should be used for determining where to truncate this Taylor series? Once you have determined this, implement this using an appropriate looping construct in Python. What is the relative error in the computation of e in this case?

Please read Submission Notes (on last page) for some additional information.

Submission Notes

1. The answer to all theoretical problems, output of Python code for computational problems including figures, tables and accompanying analyses should be provided in a single PDF file along with all your code files.
2. You can typeset (please consider using LaTeXif you choose to do so), or write by hand and scan/take photographs of solutions for theoretical questions. For scanning or photography, please put in the extra effort and provide a single PDF file of your submission.
3. For Problem 6, in your PDF submission of solutions, please provide the output from the code snippet (run via Terminal or Notebook) for each of the three parts (a)–(c). For each part, you will also explain what you observe and why. To make life easier (for you, for TAs!), here is an example of how to report what you observe.

```
a = 1.
while a <= 100:
    a = 2**a
    print(a)
```

Solution: The output I obtained by running the code snippet is below:

```
2
4
16
65536
```

The observed output is obtained because the code computes and prints the first few terms of the recurrence relation: $a(n+1) = 2^{a(n)}$ with $a(1) = 1$.

4. For Problem 7, please submit your code in two files: `problem_7a.py` and `problem_7b.py`. You can use the two Python files and the code within them as boilerplate. Please do read the comments within each of these files. The remarks are meant to clarifying or helpful but they may even contain some hints!