MTH 373/573:
Scientific Computing
Monsoon 2018

Homework 1

IIIT-Delhi
Kaushik Kalyanaraman

Due: August 16, 2018 by 11.55 pm **Total: 300 points.**

Before you start, please read General Submission Guidelines on ??.

Problem 1: Floating point hiccups

(50 points)

For computing the midpoint m of an interval $[a, b]$, which of the following two formulas is preferable in floating-point arithmetic?

(i) $m = (a + b)/2.0$

(ii) $m = a + (b - a)/2.0$

Why? When? (*Hint*: Devise examples for which the "midpoint" given by the formula lies *outside* the interval $[a, b]$.)

In addition, provide the following in your writeup.

(a) Specify the floating point system you are using.

(b) Specify values of a and b in floating-point representation (that is, if you choose a base-2 system, your numbers should be binary) within your floating point system.

(c) Write down all intermediate results for both cases in your chosen floating-point representation.

(d) Point out the steps where the problem occurs.

Problem 2: Bessel recurrence in floating point

(50 + 50 + 25 + 25 = 150 points)

Bessel functions $J_n(\cdot)$ obey the recurrence relation

$$J_{n+1}(z) = (2n/z)J_n(z) - J_{n-1}(z).$$

This means that, given $J_0(z)$ and $J_1(z)$ for a fixed $z \in \mathbb{R} \setminus \{0\}$, $J_n(z)$ for integer $n \geq 2$ can be computed – at least in theory.

You can evaluate Bessel functions using `scipy.special.jv(n,z)`.

(a) Write a Python program that tests the accuracy to which the values returned from `scipy` obey the Bessel recurrence relation. Print the left and right hand side values for each n. Also print the relative (to left hand side) error between the left and right hand sides for each $n = 2, 3, \ldots, 50$. For your experiments, fix $z = 20$. (This should yield roughly machine precision in each case.)

(b) Write a Python program that uses the values for $J_0(z)$ and $J_1(z)$ (as obtained from `scipy`) and the recurrence relation to compute the values of $J_2(z), \ldots, J_{50}(z)$. Print these recurrence computed values for each n, and also print the error relative to the value returned by `scipy` as the true value. Again, for your experiments, fix $z = 20$.

(c) You should find that the results from the recurrence relation rapidly start losing precision at around $n = 30$. Provide a reason for this loss of precision.

(d) Observe that the recurrence relation can be rearranged to compute J_{n-1} from J_{n+1} as follows:

$$J_{n-1}(z) = (2n/z)J_n(z) - J_{n+1}(z).$$

Do you believe using this modified recurrence relation to compute $J_0, \dots, J_{48}$ from J_{49} and J_{50} will encounter loss of precision? Why or why not? (Run the experiment if you are not sure!)

Problem 3: Floating point systems

(50 points)

Consider a floating point system with parameters $(\beta, p, L, U) = (2, 4, -3, 3)$. Assume that it is an ***unnormalized*** system. That is, a typical number is $b_0 \cdot b_1 b_2 b_3 \times 2^e$ where b_0 can be 0 or 1. (Of course the other bits b_1, b_2 and b_3 are also 0 or 1.)

Give examples (one in each category) of nonzero numbers that have unique, two, three, and four representations, respectively. Write the numbers in the forms $b_0 \cdot b_1 b_2 b_3 \times 2^e$. Write down all the representations in each category.

Problem 4: Basic linear algebra

(50 points)

Consider a 4×4 matrix B. We want to perform the following 6 operations on B in sequence:

1. Swap rows 1 and 4
2. Add 2 times column 3 to column 1
3. Swap columns 2 and 3
4. Add 5 times row 2 to row 4
5. Delete column 2
6. Delete row 3

(a) Express these operations as a sequence of matrix multiplications. Your final answer should be written as a product of 7 matrices, one of which will be B. Do not carry out the actual matrix product in this part.

(b) By doing the appropriate matrix multiplications express your answer as the matrix product ABC. That is, what A and C will yield an ABC which is identical to the matrix obtained by the sequence of operations listed above?

General Submission Guidelines

- **This assignment should be answered individually.**
- **You will be penalized for copying or any plagiarism with an automatic zero.**
- **Start working on your solutions early and don't wait until close to due date.**
- The points for each problem is roughly indicative of effort needed for answering that problem. For instance, 50 points can be taken to mean about 50 minutes of *focussed work*. This can include reading and thinking if the problem is theoretical. For machine problems, this will include time spent in coding, debugging and producing output assuming some Python competency. Your mileage may vary!
- IIIT-Delhi academic policies on honesty and integrity apply to all HWs. This includes not copying from one another, from the internet, a book, or any other online or offline source. A repeat offense will be reported to academic administration.
- You can ask questions about concepts, ideas and basics of coding on the Piazza discussion group for the class. You can also ask the instructor questions during office hours. You should however provide your own solutions.
- Any technical question pertaining to the HW emailed to me will be ignored.
- If you discuss or read secondary sources (other than class notes), please list all your discussion partners and/or secondary sources in your writeup. Failure to do so will constitute violation of honor code.
- Each problem should have a clear and concise write-up.
- Please clearly show all steps involved in theoretical problems.
- The answer to all theoretical problems, and output, figures and analyses for computer problems should be submitted in a PDF file or turned in handwritten.
- Handwritten submissions can be turned in during class, in person in my office or under my office door before the submission deadline. I will be present in my office during and for 5 minutes after every submission deadline to collect submissions.
- No late submission will be allowed unless I explicitly permit it.
- All Python files for computer problems should be submitted through Github Classroom.
- However, if your Python code generates an output figure or table, please provide all such results in a single PDF file along with your code submission.
- You will need to write a separate Python code for each problem and sometimes for each subproblem as well. You should name each such file as `problem_`n`.py` where n is the problem number. For example, your files could be named `problem_3a.py`, `problem_3b.py` and `problem_4.py`.
- You should import Python modules as follows:

```python
from __future__ import division
import numpy as np
import scipy as sp
import matplotlib.pyplot as plt
import numpy.linalg as npla
import scipy.linalg as spla
```

  Every code you write will have one or more of these import statements.