Student: Leidong Xu
Journal: Fall 2018
Class: Applied Math I
Instructor: Pietro Poggi-Corradini

Week 1: This week, I attended all the three lecture of Applied Math I and the major topic was going over linear algebra.

References or papers consulted:

- *Convergence Analysis of the Nonlinear Coarse-Mesh Finite Difference Method for One-Dimensional Fixed-Source Neutron Diffusion Problem*
- *A Comparison of Coarse Mesh Rebalance (CMR) and Coarse Mesh Finite Difference (CMFD) Acceleration Methods for the Neutron Transport Calculations*

Things I learned:

Went Over

- Eigenvalue and Eigenvector
- Finite element method
- Matrix Factorization
- Gauss Elimination
- LU factorization Jordan decomposition and Schur decomposition $A = QUQ^{-1}$

New

- Condition number
  We compute $f(x+\delta x) - f(x) \approx |\delta x| \cdot |f'(x)|$ and $|f'(x)|$ is absolute condition number.
  We compute $\frac{|f(x+\delta x)-f(x)|}{|f(x)|} \approx \frac{|\delta x|}{|x|} \cdot \frac{|f'(x)|\cdot|x|}{|f(x)|}$ and $|f'(x)|$ is relative condition number.
- Absolute error and relative error: $|y - \bar{y}|$ and $\frac{|y-\bar{y}|}{|y|}$
- Backward error δx could be defined by $alg(x) = f(x + \delta x)$. This error is used for verify the stability of an algorithm.
- Round-off effects The results could lose its digits by a bad algorithm,etc switch rows before using Gauss elimination.
- Floating Point Algorithm A float number stored in computer included sign, fraction, base and exponent. An IEEE double precision has 64 total bits.
- overflow/underflow threshold are the maximum and minimum value of an certain standard float number. etc, the underflow of IEEE float is 2^{1022} and overflow is 2^1024
- machine epsilon ϵ Machine epsilon could be define as $fl(a) = a(1+\delta)$ $|\delta| \leq \epsilon$ when all the numbers are not beyond the overflow/underflow.

Software Code:

- Create a repository on Github for all the assignments and Journals in the rest of the semester.

Student: Leidong Xu
Journal: Fall 2018
Class: Applied Math I
Instructor: Pietro Poggi-Corradini

Also, only for this week: Personal Intro

Hi, my name is Leidong Xu. And I have been studying mechanical and nuclear engineering in K-state for five years. This academic year will be the second year for my master degree. My thesis is related to using numerical method solving engineering and science problems on Computational Fluid Dynamics and Nuclear systems, which include preconditioner(sparse matrix) and geometric multigrid method.

I did not take any pure mathematic class in the past 3 years, and almost all the courses I took last year were using programming to solve PDEs and ODEs. Things get boring when I continue stayed in finite difference all the time. This also the main reason I come to this class, I hope I can have a better understanding of linear algebra and have a better understanding of those numerical methods in a view of applied math.

Student: Leidong Xu
Journal: Fall 2018
Class: Applied Math I
Instructor: Pietro Poggi-Corradini

Week 2: This week, I studied the rest parts of Chapter 1 which included vector and matrix norm and also a good start of Chapter

References or papers consulted:

- *Linear Independence Oracles and Applications toRectangular and Low Rank Linear Systems*

Things I learned:

- Condition number
  We compute $f(x + \delta x) - f(x) \approx |\delta x| \cdot |f'(x)|$ and $|f'(x)|$ is absolute condition number.

Software Code:

- Finished programming hw 1. The questions I picked are Problem.32 Most nonzero elements in row and Problem 17. Find all elements less than 0 or greater than 10 and replace them with NaN.
- These problems are pretty easy, and good to know we can define NaN by float('nan').

```
'''
Problem 32. Most nonzero elements in row
Given the matrix a, return the index r of the row with the most nonzero
elements. Assume there will always be exactly one row that matches this
criterion.
'''

import numpy as np


def mostnonzero(matrix):
    row = []
    max_zero = 0
    for i in range(matrix.shape[0]):
        N_nonzero = 0
        for j in range(matrix.shape[1]):
            if matrix[i, j] != 0.:
                N_nonzero += 1
        if max_zero < N_nonzero:
            row = [i]
            max_zero = N_nonzero
        elif max_zero == N_nonzero:
            row.append(i)
    row = np.array(row) + 1
    print('the' + ' ' + str(row) + 'th row has the maxinum nonzero element')
    return row


'''
Find all elements less than 0 or greater than 10 and replace them with NaN
```

```
'''


def replacenan(matrix):
    for i in range(matrix.shape[0]):
        for j in range(matrix.shape[1]):
            if matrix[i, j] > 10 or matrix[i, j] < 0:
                matrix[i, j] = float('nan')
    return matrix
```