Homework 2 — Numerical Analysis Fall 2023

Instructions:

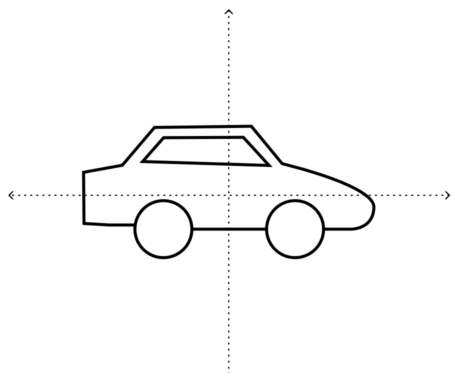
- Due 09/29 at 5:00pm on Gradescope.
- You must follow the submission policy in the syllabus

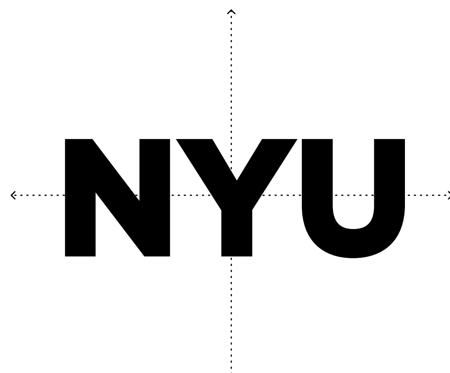
Problem 1. Consider the matrix

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -2 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \pi/4 & -\sin \pi/4 \\ \sin \pi/4 & \cos \pi/4 \end{bmatrix}$$

(a) Find a SVD of $\mathbf{A}$. Hint: think about why the given factorization is not an SVD.

(b) Draw what $\mathbf{A}$ does to the following points (here a point is anything shown in black, and the dotted lines represent the x and y axes.):





Problem 2.

(a) Suppose $\mathbf{X}$ is a $n \times m$ matrix. How does $\|\mathbf{X}\|_\mathsf{F}$ relate to $\|\mathbf{X}^\mathsf{T}\|_\mathsf{F}$?

(b) Suppose $\mathbf{X}$ is a $n \times m$ matrix. Write $\|\mathbf{X}\|_\mathsf{F}$ in terms of the column-norms $\|[\mathbf{X}]_{:,i}\|_2$.

(c) Suppose $\mathbf{X}$ is a $n \times m$ matrix and $\mathbf{U}$ is a $n \times n$ orthogonal matrix ($\mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{I}$). Show that $\|\mathbf{U}\mathbf{X}\|_\mathsf{F} = \|\mathbf{X}\|_\mathsf{F}$. Hint: use (b) and show that $\|\mathbf{U}\mathbf{x}\|_2 = \|\mathbf{x}\|_2$ for any vector $\mathbf{x}$.

(d) Let $\mathbf{A}$ be a $n \times m$ matrix with SVD $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\mathsf{T}$ and assume $n \geq m$. Prove that $\|\mathbf{A}\|_\mathsf{F} = \sqrt{\sigma_1^2 + \cdots + \sigma_m^2}$, where σ_i are the singular values of $\mathbf{A}$

Problem 3. Download the numpy data file from this link: `https://drive.google.com/file/d/18Xf0029XXm3ENWfe3Z0a--Cf6tjyCViz/view?usp=drive_link`

Use the following code to import the file into numpy.

```
import numpy as np
import matplotlib.pyplot as plt

im = np.load('change this path/CIMS.npy')
```

If you are using google colab, you can copy the `CIMS.npy` file to your own drive and then

```
from google.colab import drive
drive.mount('/content/gdrive')

im = np.load('gdrive/MyDrive/change this path/CIMS.npy')
```

In both cases, forma matrix from the image data.

```
A = np.mean(im,axis=2)
```

Here we obtain $\mathbf{A}$ by averaging the red, green, and blue channels of the image. This results in a black and white image.

(a) Plot the image using `plt.imshow`. You may want to use the colormap `'Greys_r'` so that it looks like a greyscale image.

(b) Compute the reduced SVD of $\mathbf{A}$. You can use `full_matrices=False` to get the reduced SVD. This will be much faster than computing the full SVD.

For each $k = 1, 10, 100, 200$, make a plot of the best rank-k approximation $\mathbf{A}_k$ to $\mathbf{A}$ (i.e. via truncated SVD). Label each plot with the rank k as well as the relative error $\|\mathbf{A} - \mathbf{A}_k\|_\mathsf{F} / \|\mathbf{A}\|_\mathsf{F}$

(c) Remark on the quality of the plots.

How many numbers are required to store $\mathbf{A}$? How many numbers are required to store the rank-k truncated SVD (as a factorization)?

Problem 4. Computing the SVD is expensive, but randomization can help us!

(a) Randomized numerical linear algebra (RandNLA) is the study of the use of randomness in numerical linear algebra algorithms. One of the most famous randNLA algorithms is the randomized SVD. A simple version for approximating the SVD of a $m \times n$ matrix $\mathbf{A}$ can be described in several lines:
   - Choose a $n \times k$ matrix $\mathbf{R}$ with standard normal random entries
   - Compute $\mathbf{X} = \mathbf{AR}$
   - Compute $\mathbf{Q}, \_, \_ = \textsc{reduced-svd}(\mathbf{X})$
   - $\hat{\mathbf{U}}, \hat{\mathbf{\Sigma}}, \hat{\mathbf{V}}^\mathsf{T} = \textsc{reduced-svd}(\mathbf{Q}^\mathsf{T}\mathbf{A})$:
   - Return approximate SVD of $\mathbf{A}$: $(\mathbf{Q}\hat{\mathbf{U}})\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^\mathsf{T}$

   Implement this algorithm with the same matrix $\mathbf{A}$ as in Problem 3. To generate the random matrix, you can use `np.random.randn(n,k)`.

   Again make sure to use `full_matrices=False` when computing the SVD of $\mathbf{Q}^\mathsf{T}\mathbf{A}$. Compare this to long the whole randomized SVD took (all of the steps) with $k = 100$ against the time to compute the exact SVD in the previous problem.

(b) Prove that the factors $\mathbf{Q}\hat{\mathbf{U}}$, $\hat{\mathbf{\Sigma}}$ and $\hat{\mathbf{V}}^\mathsf{T}$ have the same properties as a SVD; i.e. $\mathbf{Q}\hat{\mathbf{U}}$ and $\hat{\mathbf{V}}$ have orthonormal columns and $\hat{\mathbf{\Sigma}}$ is diagonal with non-negative entires.

(c) Make a plot of the rank $k = 100$ truncated SVD (from problem 3) and the $k = 100$ randomized SVD. Show the relative errors $\|\mathbf{A} - (\mathbf{Q}\hat{\mathbf{U}})\hat{\mathbf{\Sigma}}\hat{\mathbf{V}}^\mathsf{T}\|_\mathsf{F}/\|\mathbf{A}\|_\mathsf{F}$ for each.

(d) How long did this algorithm take to run vs. the reduced SVD in problem 3? Why was it so much faster? Hint: what are the dimensions of the matrices which we take the SVD of using this approach?