Homework 4
APPM 5650 Fall 2021
Randomized Algorithms

Due date: Monday, Sept 20 2021 at 10:20 AM
Theme: Large files, Randomized Sketches

Instructor: Prof. Becker
Revision date: 10/22/2021

Instructions Collaboration with your fellow students is allowed and in fact recommended, although direct copying is not allowed. Please write down the names of the students that you worked with. The internet is allowed for basic tasks only, not for directly looking for solutions.

An arbitrary subset of these questions will be graded.

Problem 1: [MATH] Let A be a $m \times n$ matrix with $m \geq n$, and $\sigma_1 = \sigma_{\max} = \|A\|$ the maximum singular value of A, and $\sigma_n = \sigma_{\min}$ the minimum singular value of A (which may be zero). Prove that for all vectors $x, y \in \mathbb{R}^n$ that

$$\sigma_{\min}\|x - y\|_2 \leq \|Ax - Ay\|_2 \leq \sigma_{\max}\|x - y\|_2$$

(and note that there exist x, y such that these inequalities are tight, so they cannot be improved). Note: the upper bound is still true even if $m < n$.

Problem 2: [PROGRAMMING] Using the 10×10^9 matrix U that is stored in the file `/rc_scratch/stbe1590/data_U.mat` on the research computing (rc) filesystem, compute its spectral norm $\|U\|$. The file was saved in the "version 7.3" Matlab file format, which uses a variant of HDF5. The file is about 72 GB, so I suggest *not* transferring it to your computer.

Alternatively load the file `/rc_scratch/stbe1590/data_U.h5` which was created using `h5create` and `h5write` in Matlab; this allows us to turn off compression, so operations are faster (between 2 and 10 times faster in my experience). This file format is pure HDF5.

Deliverable: What is $\|U\|$? And how much time did your code spend reading the data from disk, and how much time did it take to actually do the computations?

Hints: on the rc login node, you must request a compute node. You should have access to the 2 special nodes on the Blanca cluster that the applied math department owns. To access these nodes, run `module load slurm/blanca` and then request an interactive session on one core of one of these nodes with the command `sinteractive --account blanca-appm`. Now you have command line access on a compute node. If you want to run Matlab, it's simplest to do it without the GUI. First run `module load matlab` and then run `matlab -nodisplay` to launch a terminal version of Matlab (or `matlab -batch "scriptname"` to use it non-interactively). You can edit script files, and then call them from the Matlab command lines. If you use Python, you can use JupyterHub to run your Jupyter notebook so that the notebook runs in your web browser but the computation is done on the Blanca node; see docs and jupyter.rc.colorado.edu (a more flexible jupyterhub interface is at tutorials-jupyter.rc.colorado.edu/hub/login).

In Matlab, you can load just parts of the `.mat` file by using the `matfile` command, rather than the `load` command. Please read the documentation of `matfile`. Or if you use the `.h5` file, you can see info via `h5disp` and actually load portions of it via `h5read`.

In Python, install the h5py package `https://www.h5py.org/` to be able to read the .mat file. The `scipy.io.loadmat` package will not work.

There is a smaller file `/rc_scratch/stbe1590/data_U_medium.mat` (and a `.h5` version as well) which holds a 10×10^7 matrix; you might find this helpful for checking whether your code is correct.

Problem 3: [PROGRAMMING] **Are random projections that good?** We often use random projections precisely because they are *agnostic* to the data. What if we custom tailer a projection to the data?

Load the dataset `/rc_scratch/stbe1590/MNIST_subsampled.mat`, which is a sub-sampled version of the MNIST dataset. The file contains two variables, `X` which contains a 784×3000 matrix of the data, and `labels` which contains 3000 labels. Each label is a digit 0–9, and each column of the data matrix is a 784 length vector that represents a hand-drawn digit. You can reshape each column to be a 28×28 matrix if you'd like to graphically see what the look like. This .mat file was not saved in "version 7.3" Matlab format, so it is easy to read in Python using `scipy.io.loadmat`.

Perform the following three types of dimensionality reduction methods:

a) Write your own code to run PCA on the matrix, using either SVD or eigenvalue routines. Don't forget to center your variables first. Keep the top 2 or 3 principal components.

b) Use a Gaussian matrix to project the data to 2 or 3 dimensions.

c) Run the tSNE algorithm on the data to reduce it to 2 dimensions. You can download Matlab and Python code that runs tSNE from `https://lvdmaaten.github.io/tsne/`, or it is in newer versions ($\geq$ R2017a) of the Matlab "Statistics and Machine Learning Toolbox" under the name `tsne`; alternatively, it is implemented in the Python scikit-learn package in `sklearn.manifold.TSNE` in case you already have that installed. To learn more about tSNE and variants like UMAP, see *Minimum-Distortion Embedding* by Agrawal, Ali and Boyd (2021, Foundations and Trends in Machine Learning) which places these methods relative to others like PCA and sketches (in particular, min-distortion methods seek to minimize the *average* distortion, in contrast to the *worst-case* distortion).

Deliverables: For each dimensionality reduction method, plot your data in 2 or 3 dimensions. You should color code it according to the true labels (or adjust the marker shape, etc.). Make some short concluding remarks about the effectiveness of each method (for your own benefit, you might want to make a 3D plot and then interact with it).

Problem 4: [JUST FOR FUN] (Not required) **The Waiting Time Paradox**. Buses arrive at a bus stop at random times according to a Poisson process with intensity λ. I arrive at time t and ask how much time $\mathbb{E}(W)$ on average I will have to wait for the next bus. There are two contradictory arguments.

a) "The lack of the memory of the Poisson process implies that the distribution of my waiting time should not depend on my arrival time. In this case $\mathbb{E}(W) = 1/\lambda$."

b) "The time of my arrival is chosen at random in the interval between two consecutive buses, and for reasons of symmetry, my expected waiting time should be half the time the expected time between two consecutive buses, that is $\mathbb{E}(W) = 1/(2\lambda)$."

Which answer is correct? Carefully explain why the other answer is incorrect. [exercise taken from `https://statweb.stanford.edu/~candes/acm116/Hw/hw4.pdf`; you can look on the internet *after* attempting the problem, and there are many good explanations and simulations and comparison with real data]