

## CS/ECE/ME532 Activity 13

*Estimated time: 25 minutes for Q1 and 35 minutes for Q2.*

1. We've previously considered several low rank approximations for matrices based on the SVD. Let an  $n$ -by- $p$  matrix  $\mathbf{X}$  with  $n \leq p$  be expressed as

$$\mathbf{X} = \sum_{i=1}^n \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

where  $\sigma_i$  is the  $i$ th singular value with left singular vector  $\mathbf{u}_i$  and right singular vector  $\mathbf{v}_i$ . The rank- $r$  approximation is

$$\mathbf{X}_r = \sum_{i=1}^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

where  $r \leq n$ . Define the error between  $\mathbf{X}$  and the rank- $r$  approximation as  $\mathbf{E}_r = \mathbf{X} - \mathbf{X}_r$ .

- a) Find the SVD of  $\mathbf{E}_r$  in terms of the  $\sigma_i$ ,  $\mathbf{u}_i$ , and  $\mathbf{v}_i$ .
  - b) Suppose  $\mathbf{X}$  is full rank. What is the rank of  $\mathbf{E}_r$ ?
  - c) Find the operator norm (which is also called the 2-norm of a matrix) of the error matrix  $\|\mathbf{E}_r\|_{op}$  in terms of the SVD parameters for  $\mathbf{X}$ .
  - d) Explain the conditions under which  $\mathbf{X}_r$  will be a “good” approximation to  $\mathbf{X}$ .
- 
2. **Image compression.** A digital image can be represented with a matrix, where each element of the matrix represents a pixel in the image. A low-rank approximation to the matrix is one way to compress the image, as explored in this problem. A data file contains a matrix  $\mathbf{A} \in \mathbb{R}^{600 \times 400}$  of grayscale values scaled to lie between 0 and 1. A helper script loads the data and displays the corresponding image. There are three lines of code that require completion before you can run the code: one in the section labeled “Bucky’s Singular Values” and two in the section labeled “Low-Rank Approximation”.
    - a) Take the SVD of  $\mathbf{A}$  by completing the code. Inspect the singular value spectrum. What do you conclude about the approximate rank of  $\mathbf{A}$ ? Why is it useful to plot the logarithm of the singular values?

- b) Approximate  $\mathbf{A}$  as a rank  $r$  matrix  $\mathbf{A}_r$  by only keeping the  $r$  largest singular values and making the rest zero. Try this for  $r \in \{10, 20, 50, 100\}$  and plot the corresponding low-rank images. Also find the fractional squared error

$$e = \frac{\|\mathbf{A} - \mathbf{A}_r\|_F^2}{\|\mathbf{A}\|_F^2}$$

Comment on the how the quality of the approximation changes as  $r$  increases.

- c) Compare the space required to store the full  $\mathbf{A}$  matrix with the space required to store the rank  $r$  SVD approximation of  $\mathbf{A}$ ; how many times smaller is the storage requirement for  $r \in \{10, 20, 50, 100\}$ ? You may assume that storage space requirements are proportional to the number of numbers that must be stored. e.g. a  $10 \times 10$  matrix contains 100 numbers.
- d) Use the last section of the code to find the rank of the low-rank approximation that minimizes the sum of the bias squared and variance for a noisy version of Bucky. Note that since the “Bias-Variance Tradeoff in Low-Rank Approximations” lecture assumes an  $N$ -by- $M$  matrix with  $N < M$ , we work with the transpose of  $\mathbf{A}$  so that  $M = 600$  and  $N = 400$ .
- i. Assume the variance of each row  $M\sigma_g^2 = \sum_{j=1}^M g_{ij}^2$  in the “Bias-Variance Tradeoff in Low-Rank Approximations” lecture is  $M\sigma_g^2 = 10$ .
  - ii. Assume the variance of each row  $M\sigma_g^2 = \sum_{j=1}^M g_{ij}^2$  in the “Bias-Variance Tradeoff in Low-Rank Approximations” lecture is  $M\sigma_g^2 = 50$ .
- e) **Optional.** Simulate the noisy case by performing low-rank approximations to a noisy version of  $\mathbf{A}$ . You may create this using the command `Anoise = A + np.sqrt(sigma2/600)* np.random.randn(np.shape(A)[0], np.shape(A)[1])` in Python, where `sigma2` corresponds to  $\sigma_g^2$ . Note that the division by  $M = 600$  is necessary because `randn` creates random matrices where each element (not the row) has unit variance.