Lab Session 7

MA-571 : Numerical Linear Algebra Lab 2022 R. Alam

---

1. This problem illustrates SVD based image compression algorithm. The MATLAB commdand `A = imread('photo.jpg')` reads an image (color as well as black and white) and convets it into a matrix `A`. The command `AG = rgb2gray(A)` converts the image to a grayscale image `AG`. On the other hand, the command `imshow('X')` displays the image (color or gray) stored in a matrix `X`.

   A compressed image of the grayscale image `AG` is computed as follows. Compute the SVD $AG = U\Sigma V^\top$ and the best k rank approximation $A_k := U \begin{bmatrix} \mathrm{diag}(\sigma_1, \ldots, \sigma_k) & 0 \\ 0 & 0 \end{bmatrix} V^T$ for a chosen value of $k \leq \mathrm{rank}(A)$. Then A_k represents a compressed image. The compressed image can be displayed by the command `imshow(`A_k`)`. Use the following commands:

   `[U, S, V] = svd(AG);` ; Ak = U(:, 1:k)*S(1:k, 1:k)*V(:,1:k)'; `imshow(Ak)`

   Write a MATLAB function or an `m-file` that takes an image 'photo.jpg' and rank k as input and returns a compressed image as output.

   The storage required for A_k is $k(m+n) = (m+n)k$ whereas the storage required for the full image is mn. Therefore, $\frac{(m+n)k}{mn}$ gives the compression ratio for the compressed image. Also the error in the representation is $\frac{\sigma_{k+1}}{\sigma_1}$.

   Choose a photo (jpg, png or any format) and run the above commands for various choices of k and make a table that records the relative errors and compression ratios for each choice.

   How does the choice of approximating rank k affect the visual qualities of the images? There are no precise answers here. Your results will depend upon the images you choose and the judgments you make.

   Color image: For an m-by-n color image the command `X = imread('photo.jpg');` produces a three-dimensional $m \times n \times 3$ array X with m-by-n integer subarrays for the red, green, and blue intensities. It would be possible to compute three separate m-by-n singular value decompositions of the three colors, i.e., SVD of `X(:, :, r)` for $r = 1, 2, 3$. Then rank k approximation of the three matrices can be used to compress the image.

   An alternative that requires less work involves altering the dimensions of X with `X = reshape(X,m,3*n);` and then computing one m-by-3n SVD.

   Choose a color image and perform SVD based image compression.

2. Singular value decomposition (SVD) of an m-by-n matrix A can be computed as follows.

   1. Compute spectral decomposition $A^*A = V\mathrm{diag}(\sigma_1^2, \ldots, \sigma_r^2, 0, \ldots, 0)V^*$, where $r = \mathrm{rank}(A)$ and $\sigma_j \geq \sigma_{j+1}$.
   2. Set $V_1 := V(:, 1:r)$, the first r columns of V, and set $\Sigma_r := \mathrm{diag}(\sigma_1, \cdots, \sigma_r)$.
   3. Compute spectral decomposition $AA^* = Z\mathrm{diag}(\sigma_1^2, \ldots, \sigma_r^2, 0, \ldots, 0)Z^*$ and set $U_2 := Z(:, r+1:m)$, the last $m-r$ columns of Z.

4. Compute $U_1 := AV_1\Sigma_r^{-1}$.
5. Set $U := [U_1, U_2]$ and $\Sigma := \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{C}^{m\times n}$. Then $A = U\Sigma V^*$ is an SVD of A.

Numerical rank determination is a tricky business (more on this later). To determine the rank of A you may consider `r = rank(A'*A)`. However, numerically this is likely to give inaccurate rank of A for some matrices. MATLAB determines rank of a matrix by setting a singular value σ_j to zero when $\sigma_j/\sigma_1 < \mathtt{max(m,n)} * \mathtt{eps}$, where `eps` is the machine epsilon. Type `help eps` to know more about `eps`.

Write a MATLAB function `[U, S, V] = mysvd(A)` that implements the above method. Use two different methods as outlined above for determining the rank of a matrix to see their influence on the computed SVD.

Your function may look like this

```
function [U, S, V] = mysvd(A)
% [U, S, V] = mysvd(A) produces a diagonal matrix S of singular
% values of nonsingular matrix A and two unitary matrices U and V
% whose columns are the corresponding left and right singular
% vectors so that AV = US.
```

Experiment: Your task is to compare `mysvd` with the MATLAB function `svd` by applying these functions on some test matrices. For this purpose, compute $\|V^*V - I\|_2$, $\|U^*U - I\|_2$ and $\|AV - US\|_2$ when U, S and V are obtained by your functions as well as by `svd(A)`. Plot these results and conclude which method is better and reliable. Plot the singular values and compare the smallest singular value obtained by all three methods. Consider the following test matrices.

1. Consider the magic squares `magic(n)` for $n = 10, 12, 14$.
2. Consider the Hilbert matrix `hilb(n)` for $n = 9, 11, 13$.
3. The frank matrix `F = gallery('frank',n)` for $n = 10, 12, 14$.
4. Finally, consider a matrix A with singular values $\sigma_j := 10^{-2j}$ for $j = 1:8$. Define A as follows.
   ```
   >> rand('state', 100); X = rand(8); [U, R] = qr(X);
   >> rand('state', 50); Y = rand(8); [V, R] = qr(Y);
   ```
   Define $A = U\mathrm{diag}(\sigma_1, \cdots, \sigma_8)V^*$. Let $\hat{\sigma}_j, j = 1:8$, be the computed singular values of A. For $\ell = 3, 5, 7, 8$ compute the relative errors $|\sigma_\ell - \hat{\sigma}_\ell|/\sigma_\ell$ for the methods `mysvd` and `svd`. Which method is better?.

*** End ***