Project Two

MAT-350: Applied Linear Algebra

Tatiana Gabel

2/14/2023

Problem 1

Compute A_1, the **rank-1 approximation of** A**.** Clearly state what A_1 is, rounded to 4 decimal places. Also, **compute** the root-mean square error (RMSE) between A and A_1.

Solution

```
A = [1 2 3; 3 3 4; 5 6 7]
```

```
A = 3×3
     1     2     3
     3     3     4
     5     6     7
```

```
[U,S,V] = svd(A)
```

```
U = 3×3
   -0.2904    0.9504   -0.1114
   -0.4644   -0.2418   -0.8520
   -0.8367   -0.1957    0.5115
S = 3×3
   12.5318         0         0
         0    0.9122         0
         0         0    0.3499
V = 3×3
   -0.4682   -0.8261   -0.3136
   -0.5581    0.0012    0.8298
   -0.6851    0.5635   -0.4616
```

```
%Rank-1 approximation of A
A1 = U(:,1:1)*S(1:1,1:1)*(V(:,1:1).')
```

```
A1 = 3×3
    1.7039    2.0313    2.4935
    2.7243    3.2477    3.9867
    4.9087    5.8517    7.1832
```

```
%Root mean square error
RMSE1 = norm(A-A1,'fro')/3
```

```
RMSE1 = 0.3257
```

Problem 2

Compute A_2, the **rank-2 approximation of** A. Clearly state what A_2 is, rounded to 4 decimal places. Also, **compute** the root-mean square error (RMSE) between A and A_2. Which approximation is better, A_1 or A_2? Explain.

Solution:

```
%Rank-1 approximation of A
A2 = U(:,1:2)*S(1:2,1:2)*(V(:,1:2).')
```

```
A2 = 3×3
    0.9878    2.0324    2.9820
    2.9065    3.2474    3.8624
    5.0561    5.8515    7.0826
```

```
%Root mean square error
RMSE2 = norm(A-A2,'fro')/3
```

```
RMSE2 = 0.1166
```

```
%The A2 approximation has a lower average error
```

Explain:

Problem 3

Compute the dot product $d_1 = dot(\mathbf{u}_1, \mathbf{u}_2)$.

Compute the cross product $\mathbf{c} = cross(\mathbf{u}_1, \mathbf{u}_2)$ and dot product $d_2 = dot(\mathbf{c}, \mathbf{u}_3)$. Clearly state the values for each of these computations. Do these values make sense? **Explain**.

Solution:

```
% Dot product of vectors u1 and u2
% from this we know that u1 and u2 are not orthogonal
d1 = dot(U(:,1:1),U(:,2:2))
```

```
d1 = 1.6653e-16
```

```
% Cross product of vectors u1 and u2
% This results in another vector that
% is orthogonal to both u1 and u2
% C is also equal to u3
c = cross(U(:,1:1),U(:,2:2))
```

```
c = 3×1
   -0.1114
   -0.8520
    0.5115
```

```
% Dot product of C and u3
d2 = dot(c,U(:,3:3))
```

```
d2 = 1.0000
```

Explain: This makes sense, C and u3 are the same and both unit vectors, so their dot product would always be 1. u1,u2,u3 are derived from the singular values and are orthogonal to each other, hence a vector that is orthogonal to u1 and u2 must be in the same direction(or directly opposite direction), in this case it was the first scenario.

Problem 4

Using the matrix $U = [\mathbf{u}_1\ \mathbf{u}_2\ \mathbf{u}_3]$, d**etermine whether or not the columns of U span $\mathbb{R}^3$. Explain your approach.**

Solution:

```
% We form a plane between u1 and u2
% if we account for error due to how double's are stored in computer memory
% We can assume this is 0, I will use this logical for the following
% computations
% Since the dot product between u1 and u2, we know that they are orthogonal
% so they are not parallel and can form a plane
testu1u2plane = dot(U(:,1:1),U(:,2:2))
```

```
testu1u2plane = 1.6653e-16
```

```
% in order to be considered an orthogonal compliment u3 must be orthogonal to both.
% Since the dot product between u1 and u3 is zero they are orthogonal
testu1 = dot(U(:,1:1),U(:,3:3))
```

```
testu1 = -2.7756e-17
```

```
% Since the dot product between u2 and u3 is zero they are orthogonal
testu2 = dot(U(:,2:2),U(:,3:3))
```

```
testu2 = 8.3267e-17
```

Explain: We can define R3 as a direct sum of orthogonal complements. In other words, if a plane formed by two of the vectors is orthogonal to the 3rd vector, then we have a direct sum for R3.

Problem 5

Use the MATLAB imshow() function to load and display the image A stored in the image.mat file, available in the Project Two Supported Materials area in Brightspace. For the loaded image, **derive the value of k** that will result in a compression ratio of $CR \approx 2$. For this value of k, **construct the rank-*k* approximation of the image**.

Solution:

```
load("MAT 350 Project Two MATLAB Image (1).mat")
figure
imshow(A)
```

```
% Finding the value of k for CR=2
CR=2;
m = height(A);
n = width(A);
k = (m * n) / (CR * (m + n + 1));
%rounding
k = round(k)
```

```
k = 801
```

```

%using this value we find the rank-k approximation
imageA = im2double(A); %converts A to a matrix of doubles
[U,S,V] = svd(imageA);
A801 = U(:,1:801)* S(1:801, 1: 801) * V(:,1:801).';
```

Explain: We use the equation for the compression rate, to find k or the number of singular values/ columns we need to consider in our approximation. In this case we rounded it to be 801.

Problem 6

Display the image and compute the root mean square error (RMSE) between the approximation and the original image. Make sure to include a copy of the approximate image in your report.

Solution:

```
imshow(A801)
title('CR=2 approximation')
```

```
% Root mean square error between the original and the approximation
RMSE801 = norm((imgA-A801),'fro')/sqrt(m*n)
```

```
RMSE801 = 0.0124
```

Problem 7

Repeat Problems 5 and 6 for $CR \approx 10$, $CR \approx 25$, and $CR \approx 75$. **Explain** what trends you observe in the image approximation as CR increases and provide your recommendation for the best CR based on your observations. Make sure to include a copy of the approximate images in your report.

Solution:

```
% We repeat the calculation for a Compression rate of 10
CR = 10;
k = round((m * n) / (CR * (m + n + 1)))
```

```
k = 160
```

```
% We can reuse the svd values, as it refers to the same matrix
A160 = U(:,1:k)* S(1:k, 1: k) * V(:,1:k).'
```

```
A160 = 2583×4220
    0.1139    0.1139    0.1190    0.1125    0.1114    0.1012    0.1040    0.0991 ...
    0.1157    0.1141    0.1175    0.1108    0.1109    0.1028    0.1081    0.1024
    0.1020    0.1008    0.1042    0.0969    0.0967    0.0902    0.0950    0.0874
    0.1112    0.1120    0.1159    0.1073    0.1083    0.1011    0.1059    0.0970
    0.1118    0.1116    0.1150    0.1094    0.1122    0.1041    0.1095    0.1008
    0.1045    0.1048    0.1068    0.1017    0.1047    0.0995    0.1076    0.0993
    0.1087    0.1056    0.1073    0.1014    0.1030    0.0973    0.1053    0.0972
    0.1043    0.1031    0.1045    0.1015    0.1058    0.1015    0.1088    0.1000
    0.0999    0.0993    0.0995    0.0968    0.0996    0.0968    0.1027    0.0941
    0.0941    0.0918    0.0909    0.0897    0.0916    0.0891    0.0939    0.0864
      ⋮
```

```
imshow(A160)
title('CR=10 approximation')
```

```
% Root mean square error between the original and the approximation
%This is higher than CR=2, but still fairly small
RMSE160 = norm((imgA-A160),'fro')/sqrt(m*n)
```

```
RMSE801 = 0.0322
```

```
% We repeat the calculation for a Compression rate of 25
CR = 25;
k = round((m * n) / (CR * (m + n + 1)))
```

```
k = 64
```

```
% We can reuse the svd values, as it refers to the same matrix
A64 = U(:,1:k)* S(1:k, 1: k) * V(:,1:k).'
```

```
A64 = 2583×4220
    0.0980    0.0955    0.0863    0.0871    0.0861    0.0860    0.0904    0.0933 ···
    0.0988    0.0960    0.0873    0.0879    0.0878    0.0883    0.0925    0.0951
    0.0932    0.0902    0.0816    0.0830    0.0822    0.0839    0.0880    0.0908
    0.0954    0.0934    0.0850    0.0857    0.0855    0.0867    0.0908    0.0937
    0.0973    0.0942    0.0858    0.0866    0.0871    0.0878    0.0919    0.0952
    0.0946    0.0921    0.0833    0.0846    0.0851    0.0859    0.0892    0.0923
    0.0944    0.0910    0.0826    0.0837    0.0841    0.0851    0.0885    0.0915
    0.0984    0.0954    0.0872    0.0889    0.0903    0.0902    0.0931    0.0960
    0.0931    0.0898    0.0818    0.0831    0.0844    0.0844    0.0870    0.0905
    0.0874    0.0845    0.0762    0.0785    0.0797    0.0799    0.0819    0.0862
      ⋮
```

```
imshow(A64)
title('CR=25 approximation')
```

```
% Root mean square error between the original and the approximation
RMSE64 = norm((imgA-A64),'fro')/sqrt(m*n)
```

```
RMSE801 = 0.0483
```

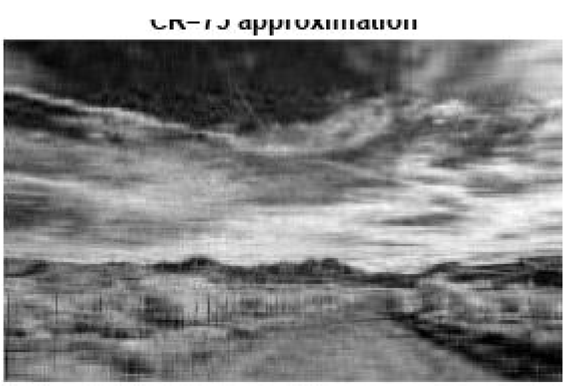
```
% We repeat the calculation for a Compression rate of 75
CR = 75;
k = round((m * n) / (CR * (m + n + 1)))
```

```
k = 21
```

```
% We can reuse the svd values, as it refers to the same matrix
A21 = U(:,1:k)* S(1:k, 1: k) * V(:,1:k).'
```

```
A21 = 2583×4220
    0.1028    0.1005    0.0980    0.0984    0.1002    0.1034    0.1075    0.1065 ···
    0.1040    0.1016    0.0991    0.0995    0.1015    0.1048    0.1088    0.1079
    0.1031    0.1007    0.0982    0.0988    0.1008    0.1042    0.1085    0.1076
    0.1061    0.1041    0.1016    0.1022    0.1040    0.1072    0.1112    0.1102
    0.1055    0.1029    0.1005    0.1008    0.1030    0.1061    0.1100    0.1093
    0.1043    0.1017    0.0994    0.0998    0.1022    0.1052    0.1091    0.1083
    0.1052    0.1026    0.1003    0.1006    0.1026    0.1054    0.1094    0.1084
    0.1068    0.1045    0.1022    0.1023    0.1046    0.1071    0.1108    0.1097
    0.1020    0.0995    0.0973    0.0972    0.0996    0.1019    0.1054    0.1044
    0.1073    0.1048    0.1026    0.1027    0.1050    0.1074    0.1110    0.1099
      ⋮
```

```
imshow(A21)
title('CR=75 approximation')
```



```
% Root mean square error between the original and the approximation
RMSE21 = norm((imgA-A21),'fro')/sqrt(m*n)
```

```
RMSE801 = 0.0716
```

Explain: As CR increases, the quality of the image decreases, and this vertical and horizontal pattern becomes apparent. I also noticed that because CR and k are proportional, when CR is changed by a certain factor, k is changed by that same factor. The RMSE however is not, as it's value appears to increase at a slow rate as the CR increases. For example from CR 2 to 25, the RMSE value increases by 0.0359, but the change from CR 25 to 75(which is a larger gap) is 0.0233. It would be interesting to see if this behavior continues or not.

My recommendation for the best CR would be CR2 or CR10. CR2 is less compressed, but looks better. C10 takes up less space but there is already some visible pixelation and depending on what the site is primarily used for this could drive users away.