Project Two Template

MAT-350: Applied Linear Algebra

Gigi Morales

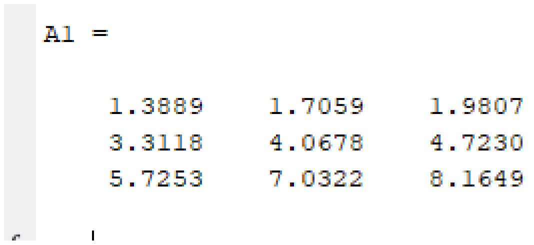
06/20/21

Problem 1

Use the svd() function in MATLAB to 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:

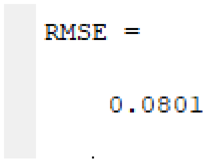
```
vA = [1 2 2; 3 4 5; 6 7 8]
[U, S, V] = svd(vA)
A1 = U(:,1:1)*S(1:1,1:1)*V(:,1:1)'
```



```
A1 =

    1.3889    1.7059    1.9807
    3.3118    4.0678    4.7230
    5.7253    7.0322    8.1649
```

```
E = vA-A1
RMSE = norm(E, 'fro')/(3*3)
```



```
RMSE =

    0.0801
```

Problem 2

Use the svd() function in MATLAB to 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:

```
A2 = U(:,1:2)*S(1:2,1:2)*V(:,1:2)'
```

```
A2 =

    1.0100    1.8213    2.1469
    2.9907    4.1656    4.8639
    6.0029    6.9476    8.0431
```

```
E1 = vA-A2
RMSE = norm(E1, 'fro')/(3*3)
```

```
RMSE =

    0.0359
```

Explain: The approximation for A2 is closer to the target value and is therefore a better approximation.

Problem 3

For the 3×3 matrix A, the singular value decomposition is $A = USV'$ where $U = [\mathbf{u}_1\ \mathbf{u}_2\ \mathbf{u}_3]$. Use MATLAB to **compute** the dot product $d_1 = dot(\mathbf{u}_1, \mathbf{u}_2)$.

Also, use MATLAB to **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:

```
U1 = U(:,1)
U2 = U(:,2)
U3 = U(:,3)
```

```
U1 =

   -0.2055
   -0.4900
   -0.8471


U2 =

   -0.6658
   -0.5644
    0.4880


U3 =

   -0.7172
    0.6643
   -0.2103
```

```
dot1 = dot(U1, U2)
crss = cross(U1, U2)
dot2 = dot(crss, U3)
```

```
dot1 =

  -8.3267e-17


crss =

   -0.7172
    0.6643
   -0.2103


dot2 =

     1
```

Explain: With an almost neglible amount being attributed to the dot product of U1 and U2, it becomes apparent that they are orthogonal to each other also making them perpendicular by definition. Furthermore, it makes sense that the dot2 = 1 because the cross product is equal to U3.

Problem 4

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

Solution:

```
U = [-0.2055 -0.6658 -0.7172; -0.4900 -0.5644 0.6643; -0.8471 0.4880 -0.2103]
[rowsred, pivots] = rref(U)
```

```
U =

   -0.2055   -0.6658   -0.7172
   -0.4900   -0.5644    0.6643
   -0.8471    0.4880   -0.2103


r =

     1     0     0
     0     1     0
     0     0     1


pivots =

     1     2     3
```

Explain: By obtaining the row reduced echelon form of U, it becomes possible to visually check whether or not the columns of U span R^3 by simply checking to see if the number of pivots in rrefU and rank are equal to the span of R^3.

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:

```
imshow(A)
```

```
[U, S, V] = svd(double(A))
[m, n] = size(A) %3072x4608
CR = 2
%solve given CR equation for missing variable k with given vals
k = round((m*n)/(CR*(m+n+1)))
%k-rank approx.
Ak = U(:,1:k)*S(1:k, 1:k)*V(:,1:k)'
%convert to orig. format
Ak = uint8(round(Ak))
```

Explain: After loading img, must be converted to double in order to do single value decomposition. Then, solve given compression rate equation for unknown variable k, yielding 921.

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(Ak)
```

```
E2 = double(A) - double(Ak)
RMSEk = norm(E2, 'fro')/(m*n)
```

```
RMSEk =

   4.0658e-04
```

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:

```
CR = 10
k2 = round((m*n)/(CR*(m+n+1)))
Ak2 = U(:, 1:k2)*S(1:k2, 1:k2)*V(:,1:k2)'
Ak2 = uint8(round(Ak2))
E3 = double(A) - double(Ak2)
RMSEk2 = norm(E3, 'fro')/(m*n)
```

```
RMSEk2 =

   9.1735e-04
```

```
imshow(Ak2)
```

```
CR = 25
k3 = round((m*n)/(CR*(m+n+1)))
Ak3 = U(:, 1:k3)*S(1:k3, 1:k3)*V(:,1:k3)'
Ak3 = uint8(round(Ak3))
E3 = double(A) - double(Ak3)
RMSEk3 = norm(E3, 'fro')/(m*n)
```

```
RMSEk3 =

    0.0018
```

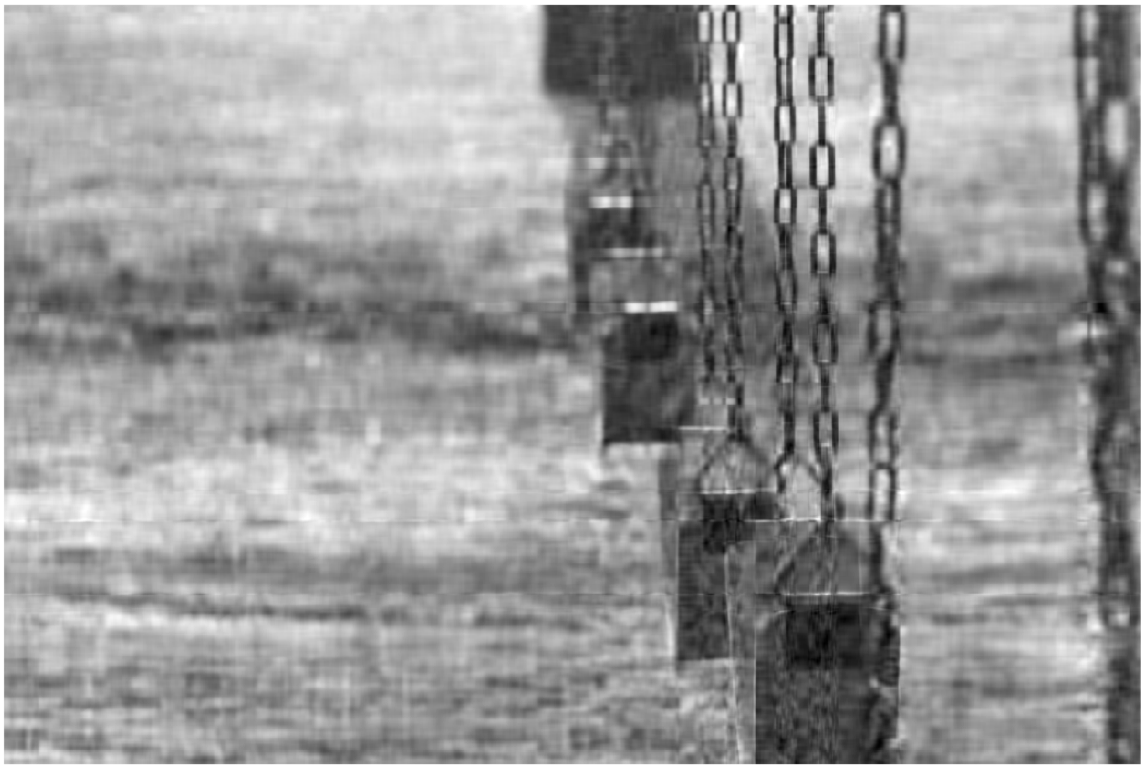
```
imshow(Ak3)
```

```
CR = 75
k4 = round((m*n)/(CR*(m+n+1)))
Ak4 = U(:, 1:k4)*S(1:k4, 1:k4)*V(:,1:k4)'
Ak4 = uint8(round(Ak4))
E4 = double(A) - double(Ak4)
RMSEk4 = norm(E4, 'fro')/(m*n)
```

```
RMSEk4 =

    0.0039
```

```
imshow(Ak4)
```



Explain: Using the same process as problems 5 and 6, we can come to see a pattern in the relationship between the compression of the pixels and the quality of the image. As compression increases and RMSE increases, the quality of the image starts to drastically decrease especially as we hit the 75 mark. On the other hand, the opposite is also true where the more true to size the pixels and image are, the clearer the image becomes.