Notes by Carolyn Kao

In this MATLAB code demonstration, I will showcase one application of the SVD. This firstly decomposes a matrix A into U, V (both orthogonal matrices) and Sigma (a rectangular diagonal matrix whose diagonal represents singular values). Then, I will use scatterplot to represent it graphically.

```
clear all; close all; clc;

%generate 10000 points in 2D
N = 10000;
```

I firstly generate 2D Data with correlation using scatterplot

```
X = mvnrnd([0 0], [1 .5;.5 1],N);
figure
colorForGraph = X(:,1)+X(:,2); %for visual reference
scatter(X(:,1),X(:,2),20,colorForGraph,'filled')
axis image
title('2D Data with Correlation');
pause
```

Then, by using the svd function built in MATLAB, I take the svd of the matrix. By the formula, A = U*Sigma*V_transpose, I firstly display the two singular values, which are on the diagonal of the diagonal matrix Sigma.

```
[U,S,V] = svd(1/sqrt(N)*X,0);
disp(diag(S));
pause
```

There are two singular vectors, each of length 2.
The first one, v1, which is the red one on the figure, points to the larger direction (dir1) of variance.
The second one, v2, which is the green one on the figure, points to the smaller direction (dir2) of the variance.
These two singular vectors are orthogonal, and are both rescaled by singular value (which is the largest absolute value among one’s elements)

```
figure
scatter(X(:,1),X(:,2),20,colorForGraph,'filled')
hold on
dir1 = V(:,1)*S(1,1);
plot([0 dir1(1)],[0 dir1(2)],'r','LineWidth',2)
dir2 = V(:,2)*S(2,2);
plot([0 dir2(1)],[0 dir2(2)],'g','LineWidth',2)
axis image
title('Directions of V with Magnitude S')
pause
```

By the formula, A = U*Sigma*V_transpose;
A*V = U*Sigma
To make the dataset uncorrelated, I multiply X by V. By the Geometric SVD Theorem, now V and U now represents orthoNORMAL bases. The two singular vectors, again, point to the larger and smaller direction of the variance, but now on the uncorrelated, renormalized dataset.

```
figure
newData = X*V;
scatter(newData(:,1),newData(:,2),20,colorForGraph,'filled');
```

```
hold on
plot([0 S(1,1)],[0 0],'r','LineWidth',2)
plot([0 0],[0 S(2,2)],'g','LineWidth',2)
title('Renormalized Data XV')
axis image
```

This is another MATLAB code demonstration, which illustrates one more application of the SVD. This firstly takes in an image called cameraman, and performs SVD. Since the picture size is 256*256, as designated, I will represent the image using different amount of singular vectors and show how much information one singular vector contains.

```
I = double(imread('cameraman.tif'));
N = size(I,1);
figure
imagesc(I)
axis image
colormap pink
pause

[U,S,V] = svd(I);
```

This plots the singular values (the diagonal elements in the matrix Sigma) in ascending order (from the biggest one). We can see that it’s decaying. The diminishing effect can be interpreted that each additional singular vector describes less information.

```
figure
plot(diag(S),'LineWidth',2)
title('Singular Values')
xlabel('i');f
ylabel('\sigma_i');
pause
```

Now, by the formula A = sum of ((u_i)*(sigma_i)*(v_i_transpose)), I show the effect of making A by summing up different amount of singular vectors. I illustrate the image using 4, 8, 16, 32, 64, 128 and 256 singular vectors. We can clearly see that as we sum up more singular vectors, the representation of the original image becomes more accurate. Reasonably, using 256 singular vectors gives out exactly the original image, as expected.

```
for i=2:log2(N)
    numKept = 2^i;
    Itmp = U(:,1:numKept)*S(1:numKept,1:numKept)*V(:,1:numKept)';
    imagesc(Itmp)
    axis image
    colormap pink
    title([num2str(numKept) ' Singular Vectors']);
    pause
end
```

Lastly, we see how the image would look like if we only use the first singular vector to represent it. That is, A = sigma(1)*u(1)*v(1)_transpose.
It indicates that the first singular vector (which corresponds to the biggest eigenvalues (in absolute value)) contains a fairly large amount of information, as the result is not too bad.

```
numKept = 1;
Itmp = U(:,1:numKept)*S(1:numKept,1:numKept)*V(:,1:numKept)';
imagesc(Itmp)
```