Homework 1

Problem 2: SVD for Image compression

Question: As you saw in class SVD can be applied to compress image. Notes here. For this question you will,

1. Import your favorite image in MATLAB
2. Apply SVD and compute how many components are needed to represent 90% of variance
3. Reconstructuct the image using 4,8,16,32, 64 and 128 components.
4. Based on your work in parts 3 and 4, how will you compress your image? Quantify reduction in image size and loss of accuracy. Is this loss of accuracy acceptable?

```
clc
close all
clear all

A = imread('butterfly.jpg');
imshow(A);
```

```
Warning: Image is too big to fit on screen; displaying at 25%
```

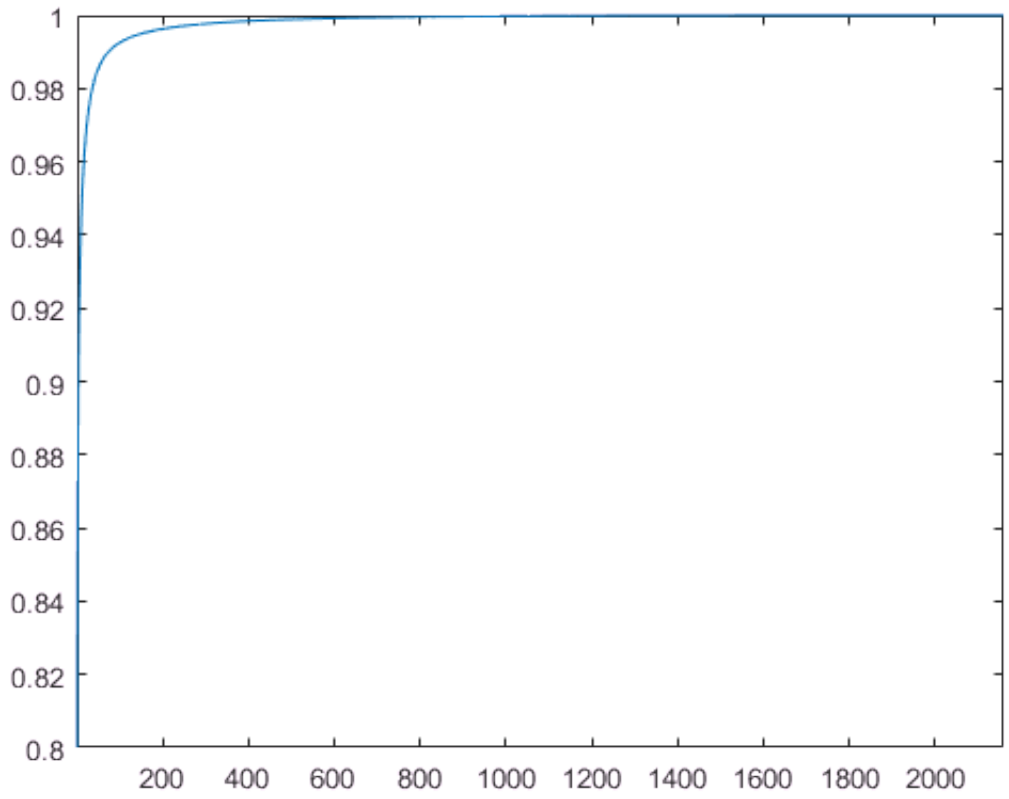
```
Ar = A(:,:,1);
Ag = A(:,:,2);
Ab = A(:,:,3);
[Ur,Sr,Vr] = svd(double(Ar));
```

```
[Ug,Sg,Vg] = svd(double(Ag));
[Ub,Sb,Vb] = svd(double(Ab));

eig_Ar = diag(Sr);
eig_Ag = diag(Sg);
eig_Ab = diag(Sb);

ep1_var = zeros(1,length(eig_Ar));
for i = 1:length(eig_Ar)
    ep1_var(i) = sum(eig_Ar(1:i).^2 + eig_Ag(1:i).^2 + eig_Ab(1:i).^2);
    ep1_var(i) = ep1_var(i)/sum(eig_Ar.^2+eig_Ag.^2+eig_Ab.^2);
end

figure;
plot(ep1_var)
axis([1 length(eig_Ar) .8 1])
```




```
ep1_var(4)
```

```
ans = 0.8924
```

```
ep1_var(5)
```

```
ans = 0.9073
```

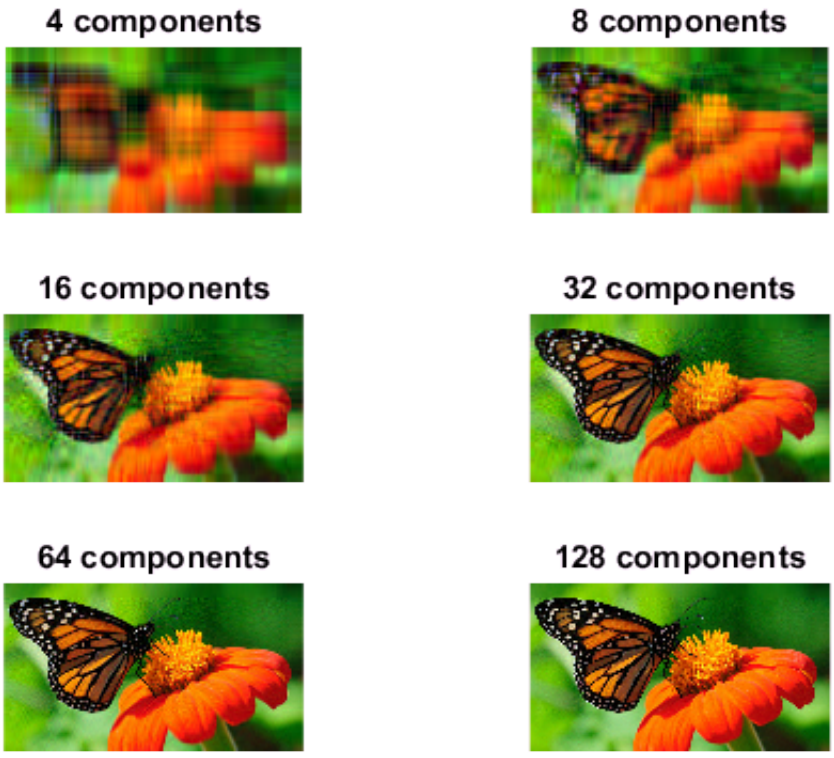
```

```

As we can see, at least 5 elements are required to have at least 90% of the image.

```
vec_red = [4,8,16,32,64,128];
figure;
for i=1:length(vec_red)
    ind = vec_red(i);
    Ar_reduced = Ur(:,1:ind)*Sr(1:ind,1:ind)*Vr(:,1:ind)';
    Ag_reduced = Ug(:,1:ind)*Sg(1:ind,1:ind)*Vg(:,1:ind)';
    Ab_reduced = Ub(:,1:ind)*Sb(1:ind,1:ind)*Vb(:,1:ind)';
    A_reduced = A;
    A_reduced(:,:,1)=Ar_reduced;
    A_reduced(:,:,2)=Ag_reduced;
    A_reduced(:,:,3)=Ab_reduced;
    subplot(3,2,i);
    imshow(A_reduced);
    title([num2str(vec_red(i)) ' components']);

end
```



Visually we can see that although we have nearly 90% of the image at 4 elements it is extremely blurry. I would therefore use the 64 or 128 component image compressions.

With 64 Component Compression

```
ind = 64;
```

```
Ar_red = Ur(:,1:ind)*Sr(1:ind,1:ind)*Vr(:,1:ind)';
Ag_red = Ug(:,1:ind)*Sg(1:ind,1:ind)*Vg(:,1:ind)';
Ab_red = Ub(:,1:ind)*Sb(1:ind,1:ind)*Vb(:,1:ind)';
A_red = A;
A_red(:,:,1)=Ar_red;
A_red(:,:,2)=Ag_red;
A_red(:,:,3)=Ab_red;


figure;
imshow(A);
```

```
Warning: Image is too big to fit on screen; displaying at 25%
```

```
imshow(A_red);
```

```
Warning: Image is too big to fit on screen; displaying at 25%
```

```
err_r = norm(double(Ar) - Ar_red)/norm(double(Ar));
err_g = norm(double(Ag) - Ag_red)/norm(double(Ag));
err_b = norm(double(Ab) - Ab_red)/norm(double(Ab));
err_avg = (err_r+err_g+err_b)/3*100;

size_diff = (2160*3840*3 - (2160 + 3840 + 64)*64*3)*100/(2160*3840*3);
```

```
fprintf('Reduced matrix represents a savings of %0.4f percent with a %0.4f percent reduction i
```

```
Reduced matrix represents a savings of 95.3210 percent with a 2.1338 percent reduction in accuracy
```

With 128 Component Compression

```
ind = 128;


Ar_red2 = Ur(:,1:ind)*Sr(1:ind,1:ind)*Vr(:,1:ind)';
Ag_red2 = Ug(:,1:ind)*Sg(1:ind,1:ind)*Vg(:,1:ind)';
Ab_red2 = Ub(:,1:ind)*Sb(1:ind,1:ind)*Vb(:,1:ind)';
A_red2 = A;
A_red2(:,:,1)=Ar_red;
A_red2(:,:,2)=Ag_red;
A_red2(:,:,3)=Ab_red;


figure;
```

```
imshow(A);
```

Warning: Image is too big to fit on screen; displaying at 25%

```
imshow(A_red);
```

Warning: Image is too big to fit on screen; displaying at 25%

```
err_r2 = norm(double(Ar) - Ar_red2)/norm(double(Ar));
err_g2 = norm(double(Ag) - Ag_red2)/norm(double(Ag));
err_b2 = norm(double(Ab) - Ab_red2)/norm(double(Ab));
err_avg2 = (err_r2+err_g2+err_b2)/3*100;
```

```
size_diff = (2160*3840*3 - (2160 + 3840 + 128)*128*3)*100/(2160*3840*3);
```

```
fprintf('Reduced matrix represents a savings of %0.4f percent with a %0.4f percent reduction i
```

```
Reduced matrix represents a savings of 90.5432 percent with a 1.1205 percent reduction in accuracy
```