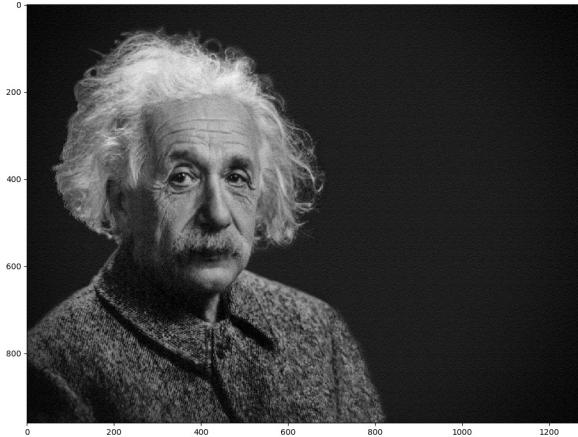


### Lab3b: Application of SVD to digital picture compression

An digital image of size  $m \times n$  is recorded as a matrix. SVD provides a method to reduce the size of the image by keeping a good quality. For example,



the original image is an  $960 \times 1280$  matrix  $A$ . We can use Matlab or Python to find the SVD decomposition of the matrix and then reconstruct the approximation of the image. (Codes of Matlab and Python in the end)

For a gray digital picture, in each pixel, it is a number in  $\{0, 1, 2, \dots, 255\}$

For color digital picture, in each pixel, it is a 3 dimensional vector  $(r, g, b)$  represented colors red green and blue.

Let  $M$  be the one of the matrix. Suppose  $M$  has a SVD decomposition

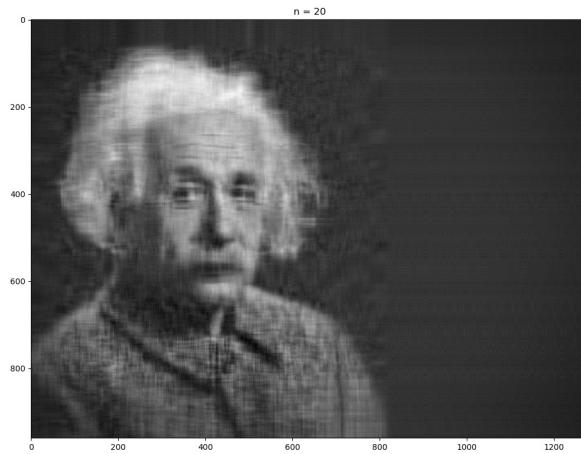
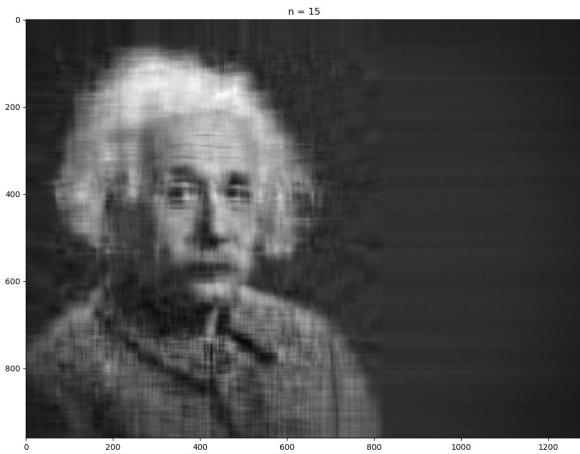
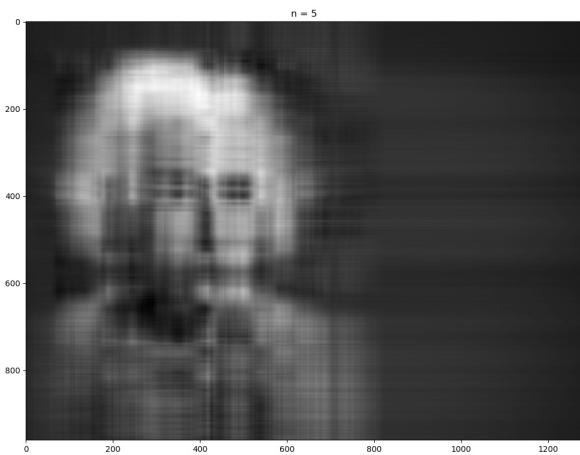
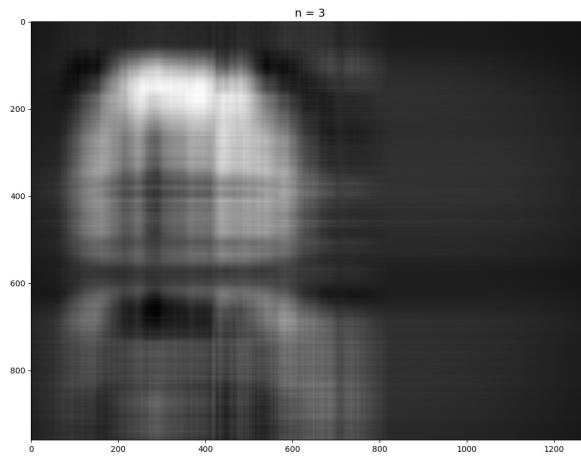
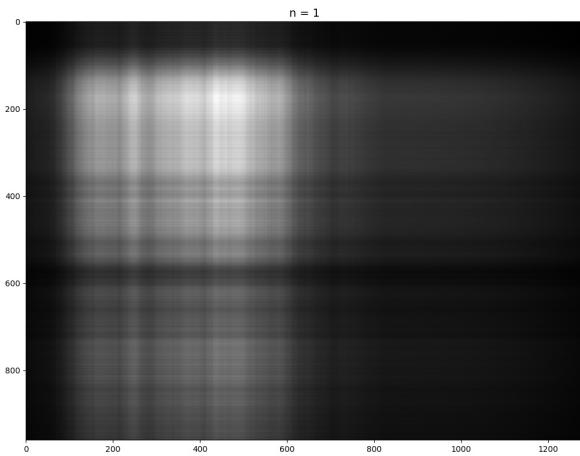
$$M = U\Sigma V^T = \sigma_1 \vec{u}_1 \vec{v}_1^T + \sigma_2 \vec{u}_2 \vec{v}_2^T + \cdots + \sigma_r \vec{u}_r \vec{v}_r^T$$

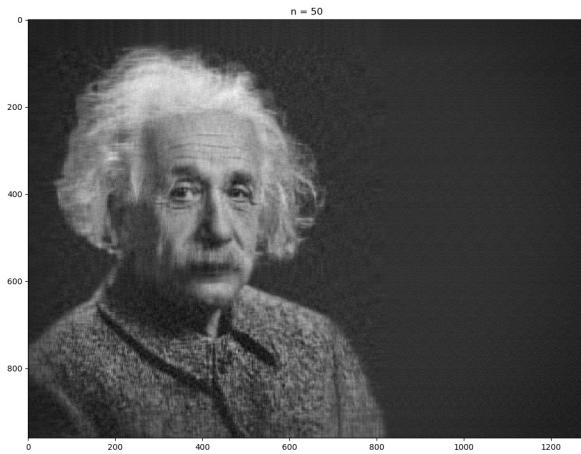
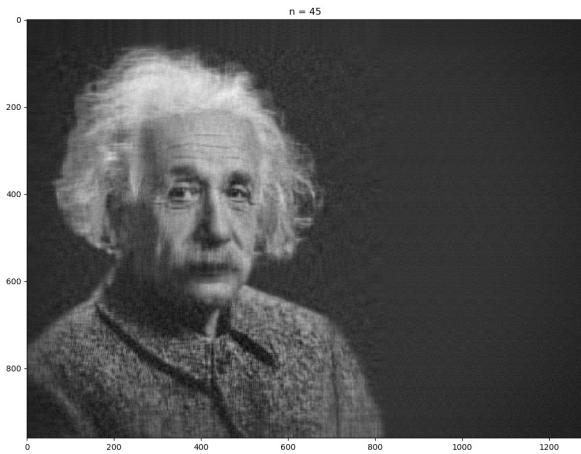
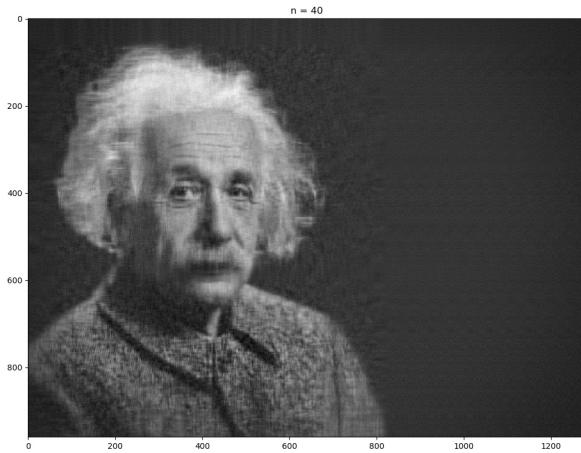
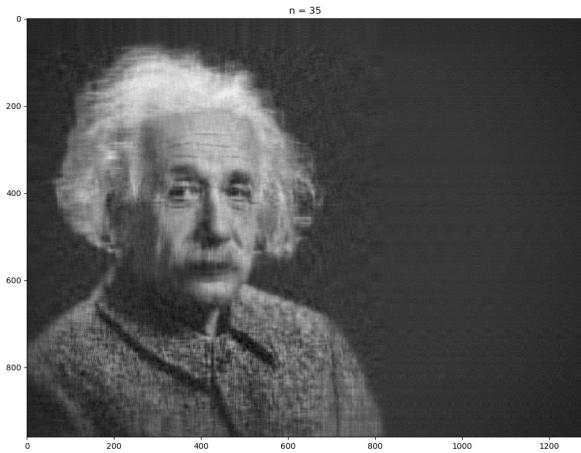
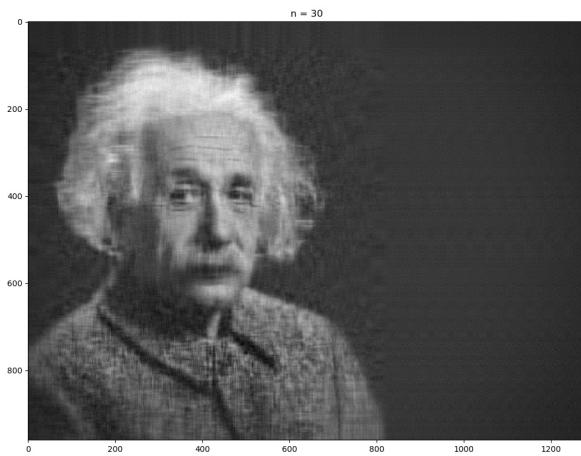
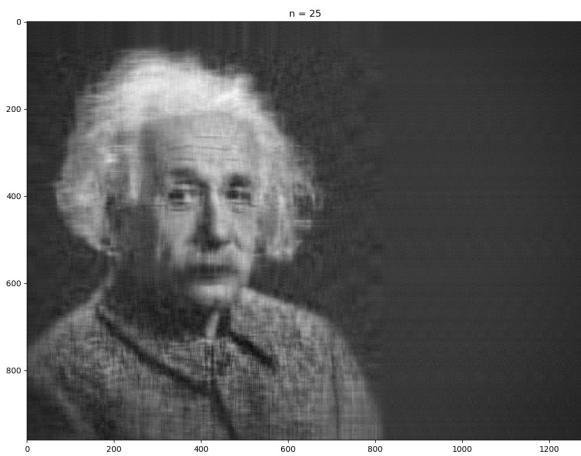
We can truncate the decomposition to be

$$M_n = \sigma_1 \vec{u}_1 \vec{v}_1^T + \sigma_2 \vec{u}_2 \vec{v}_2^T + \cdots + \sigma_n \vec{u}_n \vec{v}_n^T$$

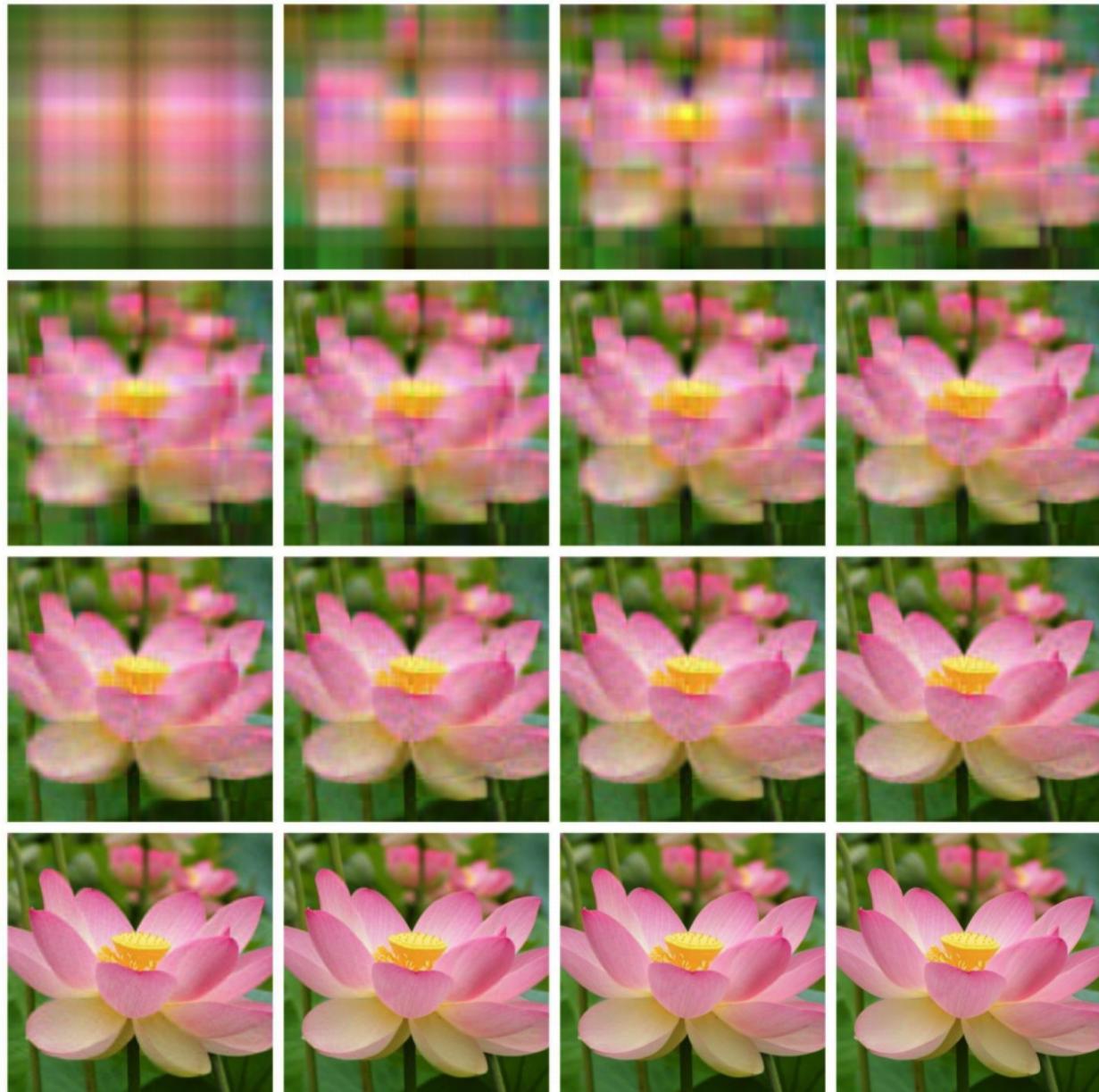
for  $n = 1, 2, \dots$

The original picture  $M$  includes  $960 \times 1280 = 1,228,800$  numbers. However, the truncated combination needs  $n \times (960 + 1280)$  numbers, which is much smaller. This is very useful for the image transmission.





For a color picture of size  $2000 \times 2000$ , with  $n = 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 25, 50, 75, 100$  and the last picture is the original picture.



Code for the Albert Einstein picture compression:

## 1. Matlab code:

```
1 clear
2 clc
3 %%comment: role of the following code?
4 a=imread('einstein.jpg');
5 [m,n,d]=size(a);
6 kmax=floor((m*n)/(m+n+1));
7 da=double(a);
8 U=zeros(m,m);S=zeros(m,n);V=zeros(n,n);e=zeros(kmax,d);cr=zeros(kmax,1);rmse=zeros(kmax,d)
9 ;
9 %%comment: SVD.
10 for i=1:d
11     [U(:,:,i),S(:,:,i),V(:,:,i)]=    %complete the code.
12 end
13
14 %%comment: role of the following code?
15 pv=20
16 for k=1:pv
17     ca=zeros(m,n,d);
18     cr(k)=m*n/(k*(m+n+1));
19     for i=1:d
20         cai=zeros(m,n,d);
21         [ca(:,:,i),cai(:,:,i)]=deal(U(:,1:k,i)*S(1:k,1:k,i)*V(:,1:k,i)');
22         e(k,i)=S(k+1,k+1,i)/S(1,1,i);
23         rmse(k,i)=sqrt(sum(sum(((da(:,:,i)-ca(:,:,i)).^2)))/(m*n));
24         imwrite(uint8(cai), sprintf('%dk%d.jpg',k,i));
25     end
26     imwrite(uint8(ca), sprintf('%dk.jpg', k));
27 end
28 figure
```

## 2. Python code:

```
1 #
2 get_ipython().run_line_magic('matplotlib', 'inline')
3 import matplotlib.pyplot as plt
4 import numpy as np
5 import time
6 from numpy import linalg as LA
7 from PIL import Image
8
9 #
10 img = Image.open('einstein.jpg')
11 imggray = img.convert('LA')
12 plt.figure(figsize=(12.8,9.6),dpi=100)
13 plt.imshow(imggray);
14
15 #
16 #
17 imgmat = np.array(list(imggray.getdata(band=0)), float)
18 imgmat.shape = (imggray.size[1], imggray.size[0])
19 imgmat = np.matrix(imgmat)
20 plt.figure(figsize=(12.8,9.6),dpi=100)
21 plt.imshow(imgmat, cmap='gray');
22 plt.savefig('/Users/hewang/Dropbox/Sec8.3/AE.jpg', bbox_inchesstr='tight')
23
24 #
25 U, sigma, V = np.linalg.svd(imgmat)
26
27 #
28 for i in range(1, 5):
29     reconstimg = np.matrix(U[:, :i]) * np.diag(sigma[:i]) * np.matrix(V[:i, :])
30     plt.figure(figsize=(12.8,9.6),dpi=100)
31     plt.imshow(reconstimg, cmap='gray')
32     title = "n = %s" % i
33     plt.title(title, fontsize=14)
34     plt.savefig('/Users/hewang/Dropbox/Sec8.3/AE'+str(i)+'.jpg', bbox_inchesstr='tight')
35     plt.show()
36
37 for i in range(5, 51, 5):
38     reconstimg = np.matrix(U[:, :i]) * np.diag(sigma[:i]) * np.matrix(V[:i, :])
39     plt.figure(figsize=(12.8,9.6),dpi=100)
40     plt.imshow(reconstimg, cmap='gray')
41     title = "n = %s" % i
42     plt.title(title)
43     plt.savefig('/Users/hewang/Dropbox/Sec8.3/AE'+str(i)+'.jpg', bbox_inchesstr='tight')
44     plt.show()
```