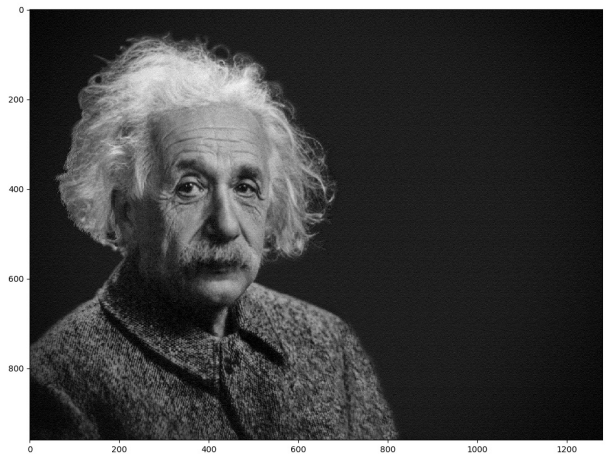
• Instructor: **He Wang** Email: **he.wang@northeastern.edu**

Lab: 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 960×1280 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 Mathlab and Python in the end)

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

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

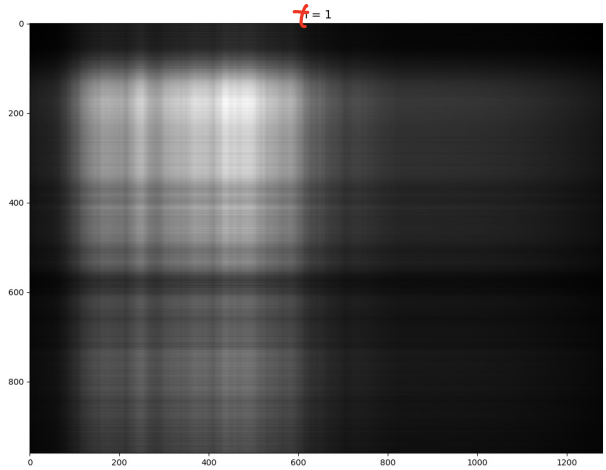
$$M = U\Sigma V^T = \sigma_1 \vec{u}_1 \vec{v}_1^T + \sigma_2 \vec{u}_2 \vec{v}_2^T + \cdots + \sigma_r \vec{u}_r \vec{v}_r^T$$

We can truncate the decomposition to be

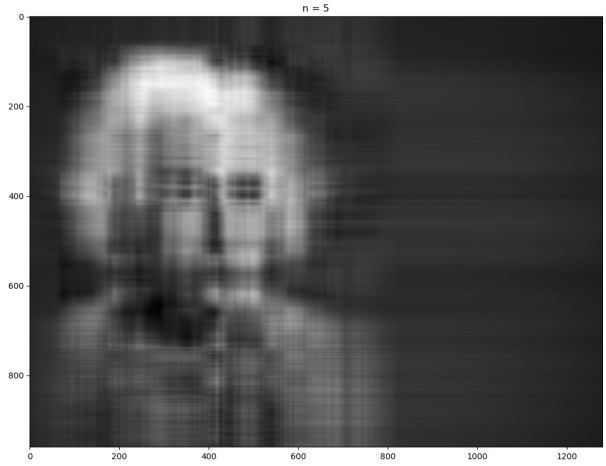
$$M_n = \sigma_1 \vec{u}_1 \vec{v}_1^T + \sigma_2 \vec{u}_2 \vec{v}_2^T + \cdots + \sigma_n \vec{u}_n \vec{v}_n^T$$

for $n = 1, 2, ...$

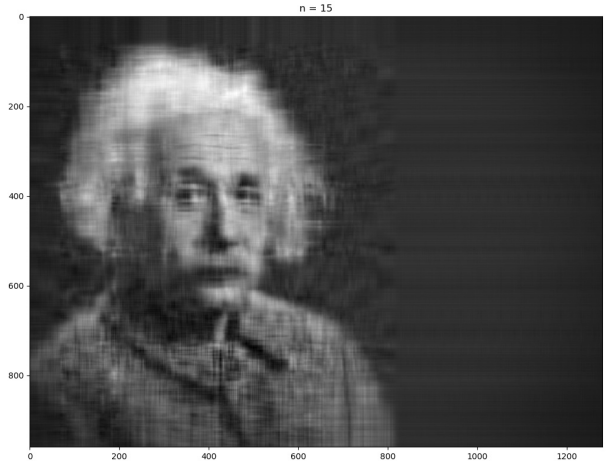
The original picture M includes $960 \times 1280 = 1,228,800$ numbers. However, the truncated combination needs $n \times (960 + 1280 + 1)$ numbers, which is much smaller. This is very useful for the image transmission.

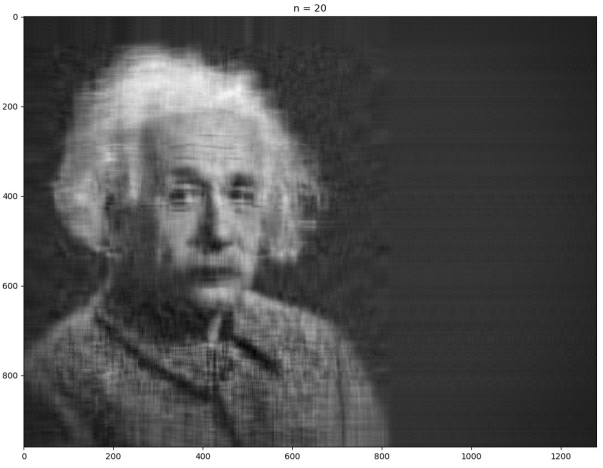

n = 1
0
200
400
600
800
0
200
400
600
800
1000
1200

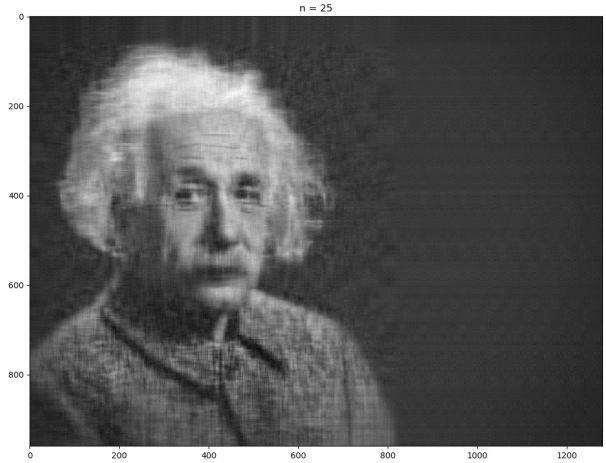

n = 3
0
200
400
600
800
0
200
400
600
800
1000
1200

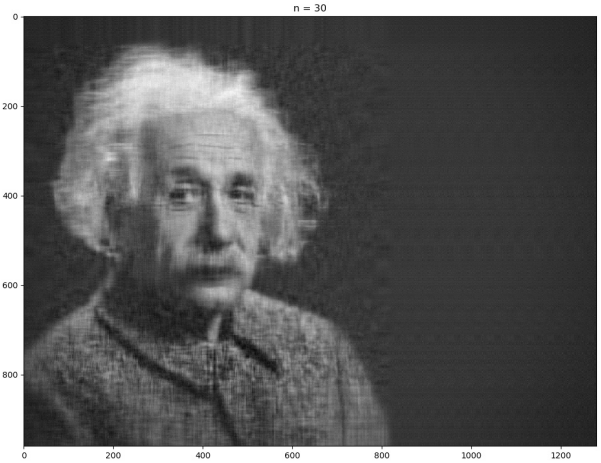

n = 5
0
200
400
600
800
0
200
400
600
800
1000
1200

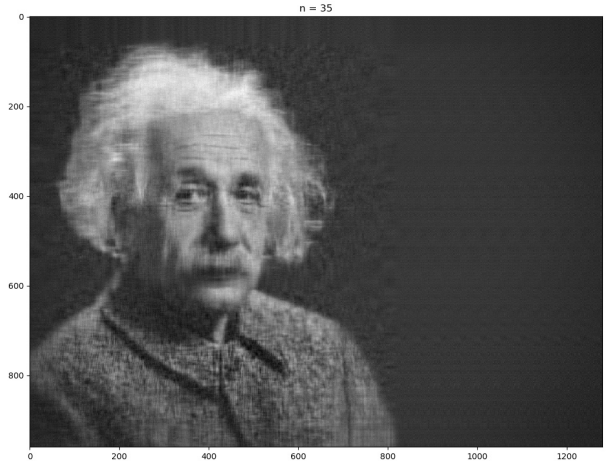

n = 10
0
200
400
600
800
0
200
400
600
800
1000
1200

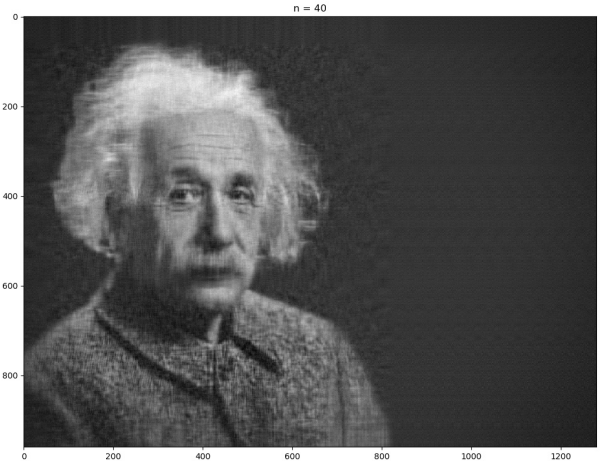

n = 15
0
200
400
600
800
0
200
400
600
800
1000
1200

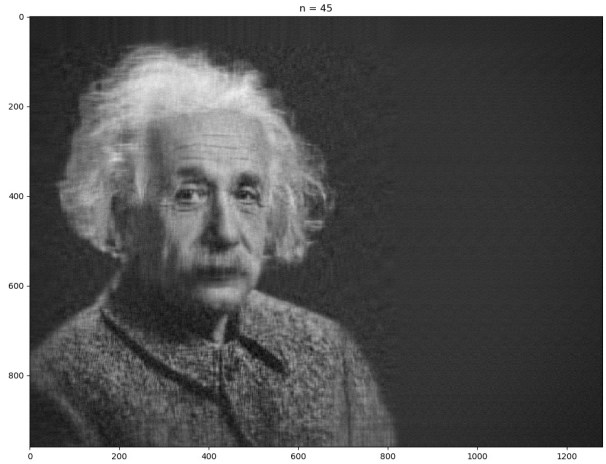

n = 20
0
200
400
600
800
0
200
400
600
800
1000
1200

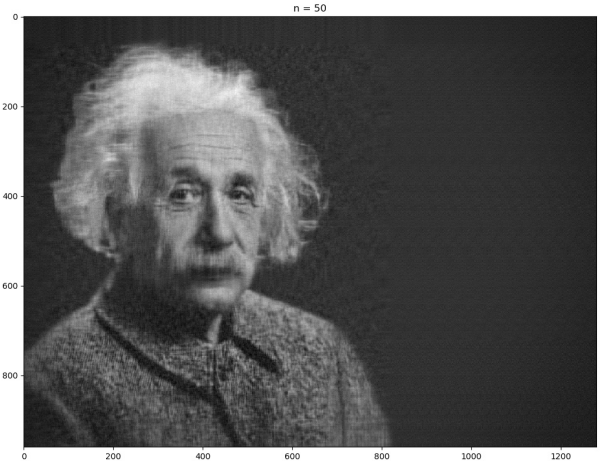

n = 25
0
200
400
600
800
0
200
400
600
800
1000
1200


n = 30
0
200
400
600
800
0
200
400
600
800
1000
1200

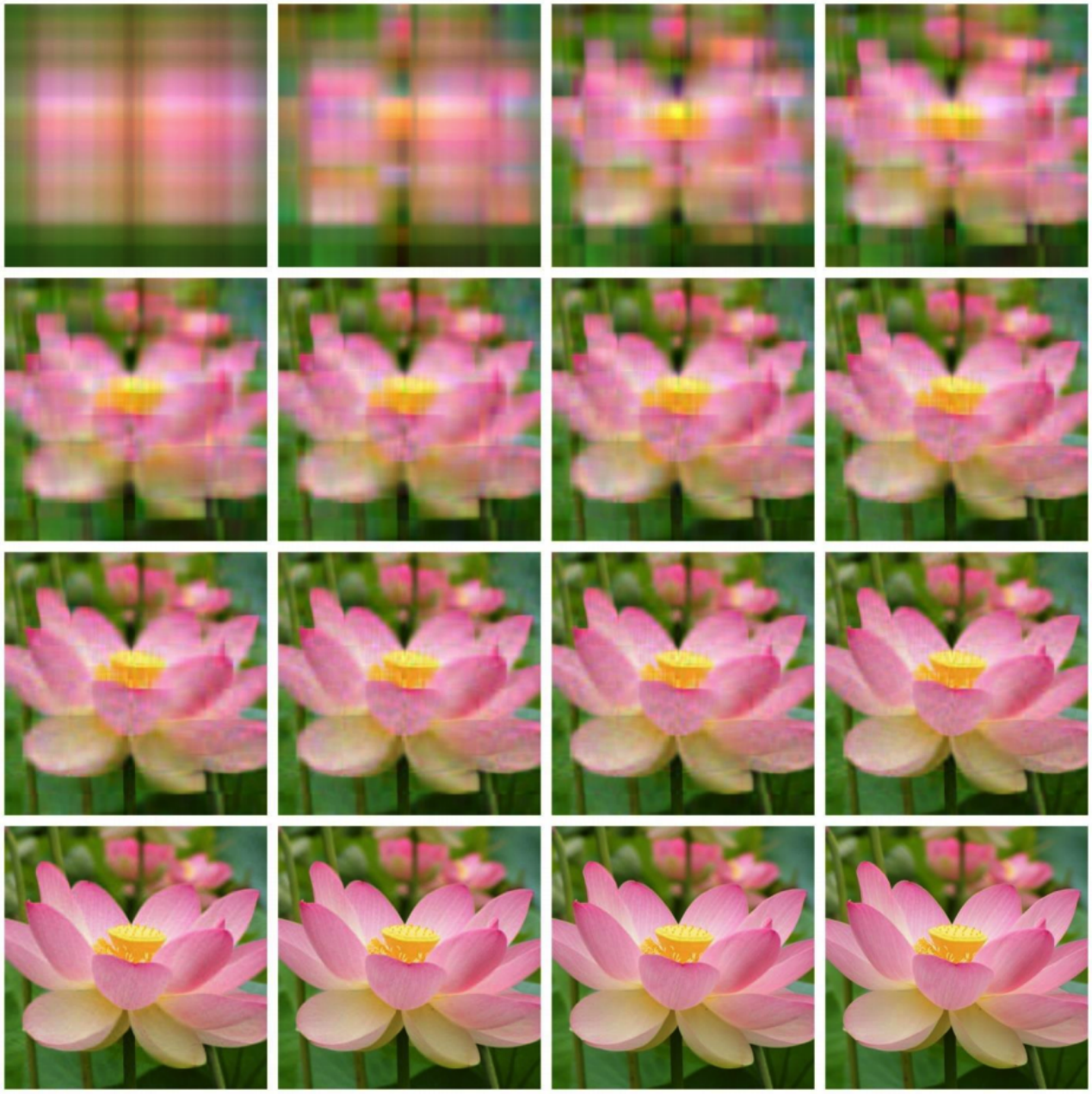

n = 35
0
200
400
600
800
0
200
400
600
800
1000
1200


n = 40
0
200
400
600
800
0
200
400
600
800
1000
1200


n = 45
0
200
400
600
800
0
200
400
600
800
1000
1200


n = 50
0
200
400
600
800
0
200
400
600
800
1000
1200

For a color picture of size 2000×2000, 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:

```
clear
clc
%%comment:  role of the following code?
a=imread('einstein.jpg');
[m,n,d]=size(a);
kmax=floor((m*n)/(m+n+1));
da=double(a);
U=zeros(m,m);S=zeros(m,n);V=zeros(n,n);e=zeros(kmax,d);cr=zeros(kmax,1);rmse=zeros(kmax,d)
    ;
%%comment: SVD.
for i=1:d
    [U(:,:,i),S(:,:,i),V(:,:,i)]=   %complete the code.
end

%%comment:  role of the following code?
pv=20
for k=1:pv
    ca=zeros(m,n,d);
    cr(k)=m*n/(k*(m+n+1));
    for i=1:d
        cai=zeros(m,n,d);
        [ca(:,:,i),cai(:,:,i)]=deal(U(:,1:k,i)*S(1:k,1:k,i)*V(:,1:k,i)');
        e(k,i)=S(k+1,k+1,i)/S(1,1,i);
        rmse(k,i)=sqrt(sum(sum(((da(:,:,i)-ca(:,:,i)).^2)))/(m*n));
        imwrite(uint8(cai), sprintf('%dk%d.jpg',k,i));
    end
    imwrite(uint8(ca), sprintf('%dk.jpg', k));
end
figure
```

Task for this lab 3b: (1) Complete the above comment and the code.

(2) Try the picture apollo.png and submit the 1st and the 10th truncation of the picture.

2. Python code:

```
#
get_ipython().run_line_magic('matplotlib', 'inline')
import matplotlib.pyplot as plt
import numpy as np
import time
from numpy import linalg as LA
from PIL import Image

#
img = Image.open('einstein.jpg')
imggray = img.convert('LA')
plt.figure(figsize=(12.8,9.6),dpi=100)
plt.imshow(imggray);


#
imgmat = np.array(list(imggray.getdata(band=0)), float)
imgmat.shape = (imggray.size[1], imggray.size[0])
imgmat = np.matrix(imgmat)
plt.figure(figsize=(12.8,9.6),dpi=100)
plt.imshow(imgmat, cmap='gray');
plt.savefig('/Users/hewang/Dropbox/Sec8.3/AE.jpg', bbox_inchesstr='tight')

#
U, sigma, V = np.linalg.svd(imgmat)

#
for i in range(1, 5):
    reconstimg = np.matrix(U[:, :i]) * np.diag(sigma[:i]) * np.matrix(V[:i, :])
    plt.figure(figsize=(12.8,9.6),dpi=100)
    plt.imshow(reconstimg, cmap='gray')
    title = "n = %s" % i
    plt.title(title, fontsize=14)
    plt.savefig('/Users/hewang/Dropbox/Sec8.3/AE'+str(i)+'.jpg', bbox_inchesstr='tight')
    plt.show()

for i in range(5, 51, 5):
    reconstimg = np.matrix(U[:, :i]) * np.diag(sigma[:i]) * np.matrix(V[:i, :])
    plt.figure(figsize=(12.8,9.6),dpi=100)
    plt.imshow(reconstimg, cmap='gray')
    title = "n = %s" % i
    plt.title(title)
    plt.savefig('/Users/hewang/Dropbox/Sec8.3/AE'+str(i)+'.jpg', bbox_inchesstr='tight')
    plt.show()
```