QUESTION – 1

We know singular values of are the square root of the eigenvalues of and.

We know since it is a diagonal matrix. We also know and since they are orthogonal matrices.

Thus equation

can be obtained which concludes that SVD decomposition gives square root of eigenvalues of which are the singular values of .

Thus equation

can be obtained which again concludes that SVD decomposition gives square root of eigenvalues of which are the singular values of .

QUESTION – 2

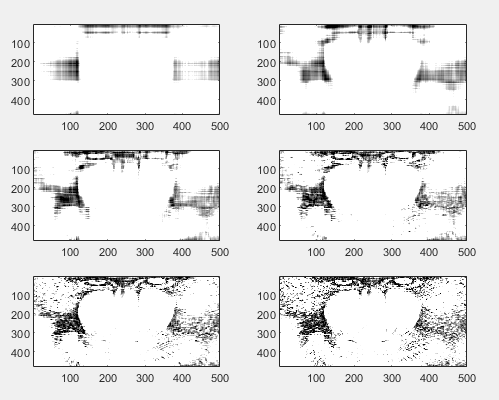
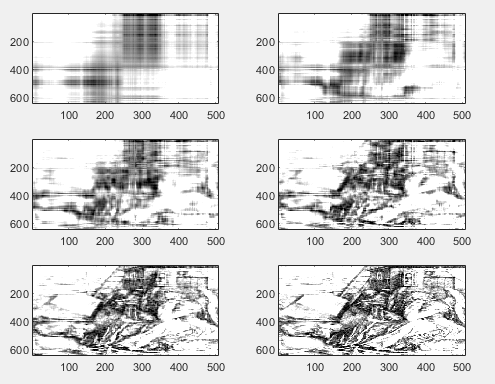
a-)

Figure 1: Mandrill image compressed with truncated SVD, with r values ranging from to

Figure 1: Drawing of Albrecht Dürer compressed with truncated SVD, with r values ranging from to

close all

figure()

colormap(gray);

load mandrill;

[U,S,V] = svd(X);

for i = 1:6

r = 2.^i;

subplot(3,2,i), image(U(:,1:r)\*S(1:r,1:r)\*V(:,1:r)');

end

figure()

colormap(gray);

load durer;

[U,S,V] = svd(X);

colormap(gray)

for i = 1:6

r = 2.^i;

subplot(3,2,i), image(U(:,1:r)\*S(1:r,1:r)\*V(:,1:r)');

end

b-) For both images until r = 32 image looks way too blurry to understand anything from it. For higher r values drawing of Dürer looks better and more detailed than the Mandrill image since original image contains more details. For lower r values Mandrill looks better since it does not have many details, thus required singular values to make a reasonable image is much less than the singular values required by Dürer’s drawing.

Original Mandrill image is a 480x500 image which takes 1 byte per pixel thus takes bytes to store. After compression the storage that is required can be calculated by

Original Durer image is a 648x509 image thus takes bytes to store. Required storage can be calculated by

QUESTION – 3

function vk = power\_method(inputMatrix, initialGuess, iterationCount)

vk = initialGuess;

for i = 1:iterationCount

tmp = inputMatrix \* vk;

vk = tmp / norm(tmp);

disp(-vk);

end

end

A = [-2 1 4; 1 1 1; 4 1 -2];

v0 = [1; 2; -1];

vt = [1; 2; 1];

power\_method(A,v0,20);

power\_method(A,vt,5);

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Iteration | 1 | 2 | 3 | 4 | 5 |
|  | 0.4364  -0.2182  -0.8729 | -0.8083  -0.1155  0.5774 | 0.6448  -0.0586  -0.7621 | -0.7356  -0.0294  0.6768 | 0.6922  -0.0147  -0.7216 |
|  | -0.5774  -0.5774  -0.5774 | -0.5774  -0.5774  -0.5774 | -0.5774  -0.5774  -0.5774 | -0.5774  -0.5774  -0.5774 | -0.5774  -0.5774  -0.5774 |

For power method found the eigenvector on the first iteration. seems like converging but it requires more steps.

[V, D] = eig(A)

V = 0.7071 0.4082 -0.5774

0 -0.8165 -0.5774

-0.7071 0.4082 -0.5774

D = -6.0000 0 0

0 0.0000 0

0 0 3.0000