

Istanbul Technical University

Department of Computer Engineering

BLG 202E – Numerical Methods

Assignment 2

Solutions

**Solution 1**

syms x

x0 = 2; %%Initial x

xn = 0; %%Next value of x

i = 0; %%Iteration step

err = 1; %% Absolute relative approximate error

fprintf('Initial x = %f\n', x0)

while i < 15

f = inline(10/(x^3-1), 'x'); %%g(x)

xn = f(x0); %%Fixed Point Iteration

err = (xn-x0)/xn\*100;

x0 = xn;

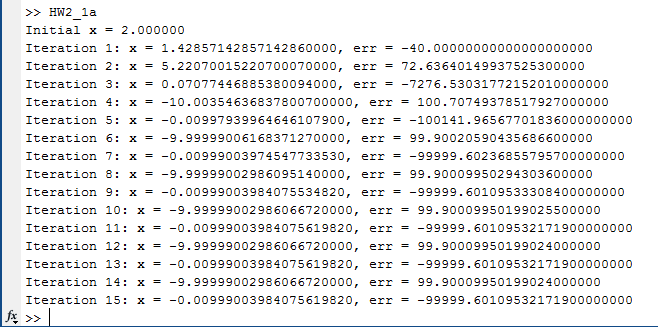
i = i +1;

fprintf('Iteration %d: x = %.20f, err = %.20f\n', i, xn, err)

end

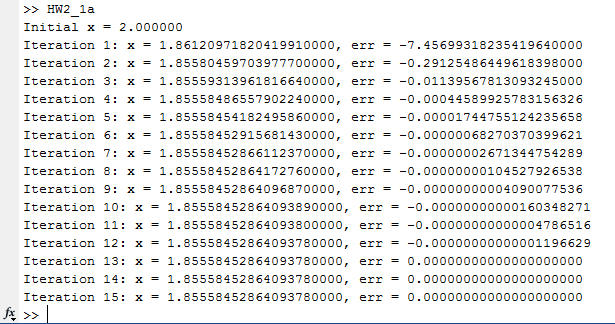
Above Matlab Code is an example of fixed point iterative method which iterate fifteen times. In 8th lines, definition of is updated for each given .

**Solution 1.a**



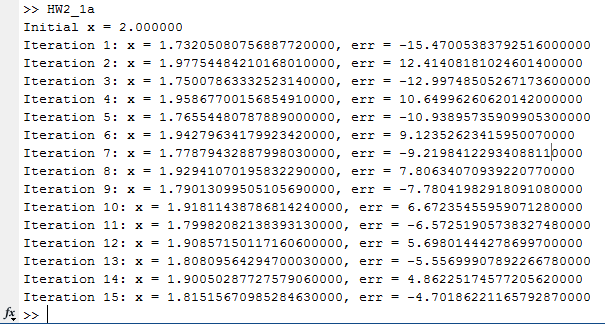
For , equation does not give a root.

**Solution 1.b**



For equation give a root as 1.85558452864093780000 in 12th step.

**Solution 1.c**

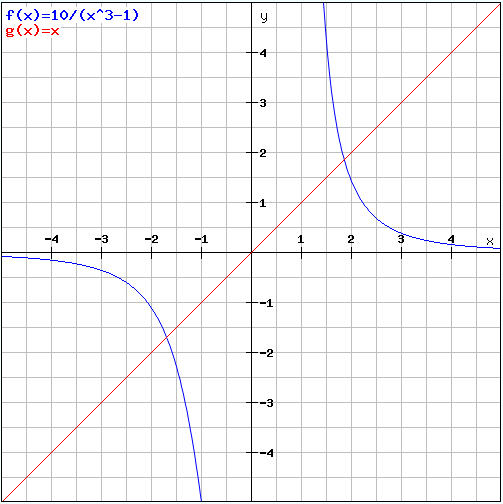
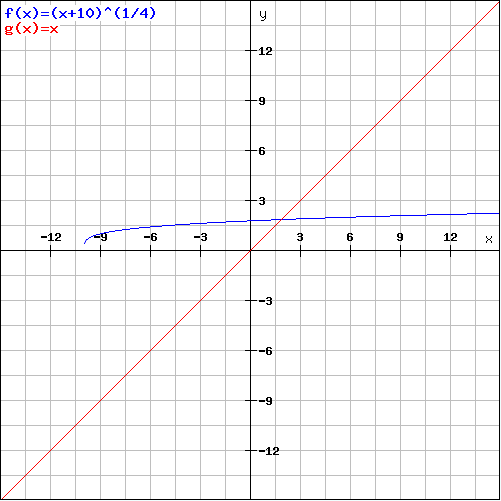


**.**

**.**

**.**

For , equation give a root as 1.85550098207740180000 in 91th step.

**Solution 1.d**

In area **I and II**, is a rising curve

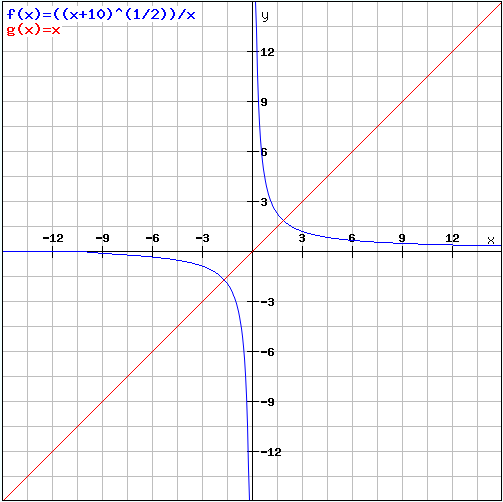
. The iteration converges. It give a root.

In area **I**, is a falling curve

. The iteration diverges. It does not give any root.

In area **III**, is a rising curve

. The iteration diverges. It does not give any root.



In area **I** intersection of , and , give a root same as with another convergence , Finally, converge.

**Solution 2**

The matrix that be evaluated is:

First, we need to compute and matrices. To find upper triangular matrix

* Multiply the first row by and add the result to the second tow to eliminate the term.
* Then, multiply the first row by and add the result to third row to eliminate the term. The result is
* Multiply the second row by and add the result to third row to eliminate the term. Final result for is like following,
* To find lower triangular matrix for , can be used coefficients where is found in order to observe . Inverse of blue multipliers will be element for , , respectively.

, so that

* , is repeated for each column.

The first column of the matrix inverse can be determined by using the

= and it is solved via forward substitution,

can be found following . This vector is used as the right-hand side of the upper triangular system,

=

which can be solved by back substitution and the result will be the first column of the invers matrix .

To determine the second column,

= and it is solved via forward substitution,

can be found following . And the results are used with

to determine via back substitution. The second column of the inverse of ,

Finally, using same steps, third column of the inverse of can be found like fallowing.

**Solution 3**

.

**Solution 4**

**“sample.png”** which is 256x256 pixel image corresponds to a 256 by 256 matrix **“imageMatrix”**.

The SVD of as .

The approximations for various values of k (k = 3, 10, 20, 100) can be shown in MATLAB Code.

r1 = 3;

r2 = 10;

r3 = 20;

r4 = 100;

colormap('gray')

imageMatrix = imread('sample.png');

figure(1)

image(imageMatrix);

imageMatrix = double(imageMatrix);

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

figure(2)

colormap('gray')

image(U(:,1:r1)\*S(1:r1,1:r1)\*V(:,1:r1)')

figure(3)

colormap('gray')

image(U(:,1:r2)\*S(1:r2,1:r2)\*V(:,1:r2)')

figure(4)

colormap('gray')

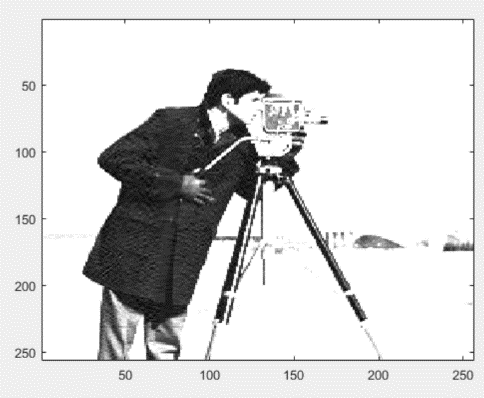
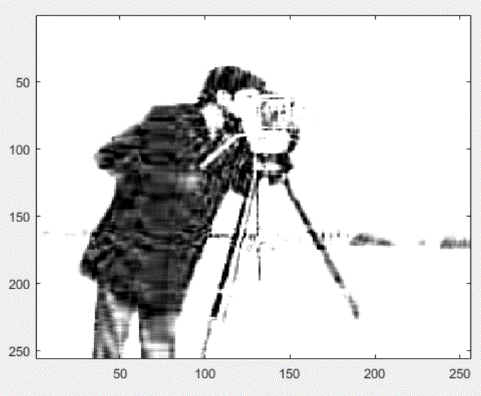
image(U(:,1:r3)\*S(1:r3,1:r3)\*V(:,1:r3)')

figure(5)

colormap('gray')

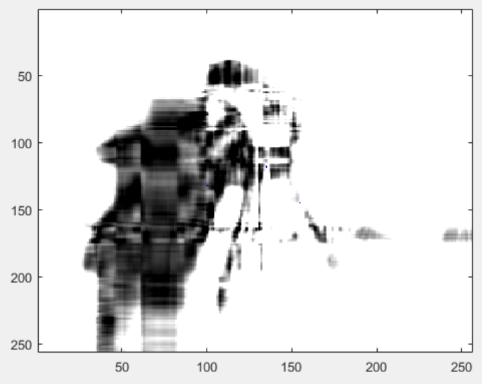
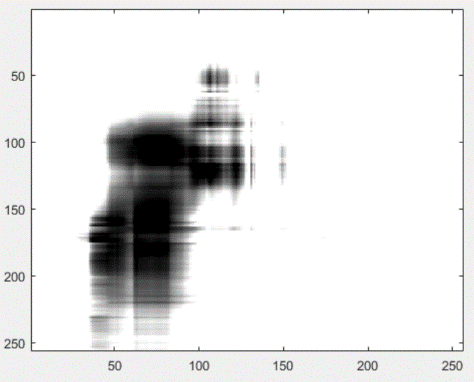
image(U(:,1:r4)\*S(1:r4,1:r4)\*V(:,1:r4)')

And corresponding output will be:



r = 20

r = 100



r = 3

r = 10

**Solution 5**

load('A.mat');

A;

AtA = A'\*A;

total = sum(AtA(:));

average = total/4;

centeredAtA = AtA - average

M = [1;0]; %%Initial matrix

index = 1;

while index < 10 %%Iterate 10 times

iterationM = centeredAtA \* M; %%A\*v = v\_next

n = norm(iterationM);

iterationM = iterationM / n; %%norm of iteration eigenvector

M = iterationM;

index = index + 1;

end

K = eig(centeredAtA); %%All eigenvalues

eigenValueWithSVD = K(2,1) %%dominant eigenvalue found in SVD

B = centeredAtA\*iterationM; %%AX = lamda\*X

x = B(1,1);

y = iterationM(1,1);

eigenValueWithPIM = x / y %% dominant eigenvalue found in PIM

%% Both are equal

In MATLAB Code, it can be seen that dominant eigenvalues are equal in found by SVD and PIM.

In power iteration method, initial matrix is .

After the power iteration, dominant eigenvector will be and corresponding dominant eigenvalue is *.*