05.03.2016
Elif Ak
150130009

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 $g(x)$ is updated for each given $g(x)$.

Solution 1.a

```
>> HW2_1a
Initial x = 2.000000
Iteration 1: x = 1.42857142857142860000, err = -40.00000000000000000000
Iteration 2: x = 5.22070015220700070000, err = 72.63640149937525300000
Iteration 3: x = 0.07077446885380094000, err = -7276.53031772152010000000
Iteration 4: x = -10.00354636837800700000, err = 100.70749378517927000000
Iteration 5: x = -0.00997939964646107900, err = -100141.96567701836000000000
Iteration 6: x = -9.99999006168371270000, err = 99.90020590435686600000
Iteration 7: x = -0.00999003974547733530, err = -99999.60236855795700000000
Iteration 8: x = -9.99999002986095140000, err = 99.90009950294303600000
Iteration 9: x = -0.00999003984075534820, err = -99999.60109533308400000000
Iteration 10: x = -9.99999002986066720000, err = 99.90009950199025500000
Iteration 11: x = -0.00999003984075619820, err = -99999.60109532171900000000
Iteration 12: x = -9.99999002986066720000, err = 99.90009950199024000000
Iteration 13: x = -0.00999003984075619820, err = -99999.60109532171900000000
Iteration 14: x = -9.99999002986066720000, err = 99.90009950199024000000
Iteration 15: x = -0.00999003984075619820, err = -99999.60109532171900000000
>>
```

For $g_1(x) = \frac{10}{x^3-1}$, equation does not give a root.

Solution 1.b

```
>> HW2_1a
Initial x = 2.000000
Iteration 1: x = 1.86120971820419910000, err = -7.45699318235419640000
Iteration 2: x = 1.85580459703977700000, err = -0.29125486449618398000
Iteration 3: x = 1.85559313961816640000, err = -0.01139567813093245000
Iteration 4: x = 1.85558486557902240000, err = -0.00044589925783156326
Iteration 5: x = 1.85558454182495860000, err = -0.00001744755124235658
Iteration 6: x = 1.85558452915681430000, err = -0.00000068270370399621
Iteration 7: x = 1.85558452866112370000, err = -0.00000002671344754289
Iteration 8: x = 1.85558452864172760000, err = -0.00000000104527926538
Iteration 9: x = 1.85558452864096870000, err = -0.00000000004090077536
Iteration 10: x = 1.85558452864093890000, err = -0.00000000000160348271
Iteration 11: x = 1.85558452864093800000, err = -0.00000000000004786516
Iteration 12: x = 1.85558452864093780000, err = -0.00000000000001196629
Iteration 13: x = 1.85558452864093780000, err = 0.00000000000000000000
Iteration 14: x = 1.85558452864093780000, err = 0.00000000000000000000
Iteration 15: x = 1.85558452864093780000, err = 0.00000000000000000000
>>
```

For $g_2(x) = (x+10)^{1/4}$ equation give a root as 1.85558452864093780000 in 12th step.

Solution 1.c

```
>> HW2_1a
Initial x = 2.000000
Iteration 1: x = 1.73205080756887720000, err = -15.47005383792516000000
Iteration 2: x = 1.97754484210168010000, err = 12.41408181024601400000
Iteration 3: x = 1.75007863332523140000, err = -12.99748505267173600000
Iteration 4: x = 1.95867700156854910000, err = 10.64996260620142000000
Iteration 5: x = 1.76554480787889000000, err = -10.93895735909905300000
Iteration 6: x = 1.94279634179923420000, err = 9.12352623415950070000
Iteration 7: x = 1.77879432887998030000, err = -9.21984122934088110000
Iteration 8: x = 1.92941070195832290000, err = 7.80634070939220770000
Iteration 9: x = 1.79013099505105690000, err = -7.78041982918091080000
Iteration 10: x = 1.91811438786814240000, err = 6.67235455959071280000
Iteration 11: x = 1.79982082138393130000, err = -6.57251905738327480000
Iteration 12: x = 1.90857150117160600000, err = 5.69801444278699700000
Iteration 13: x = 1.80809564294700030000, err = -5.55699907892266780000
```

•

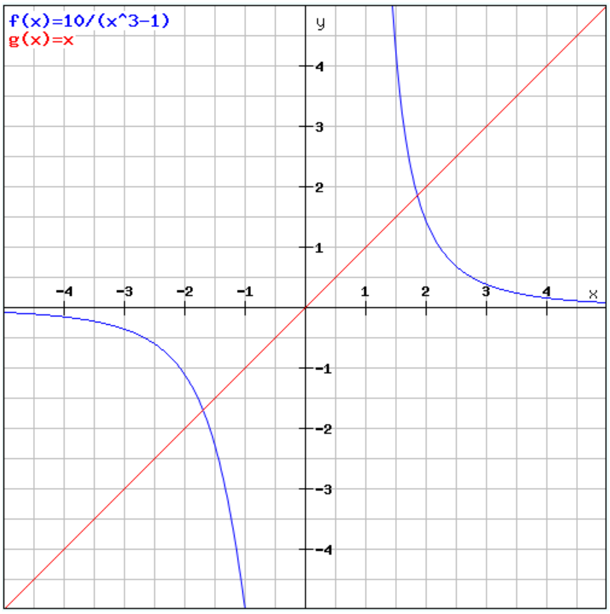
•

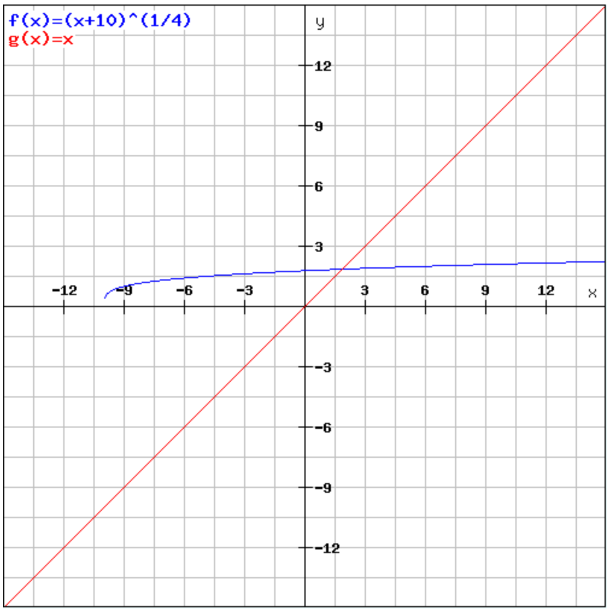
•

For $g_3(x) = \frac{(x+10)^{1/2}}{x}$, equation give a root as 1.85550098207740180000 in 91th step.

05.03.2016
Elif Ak
150130009

Solution 1.d





In area **I**, $g_1(x)$ is a falling curve

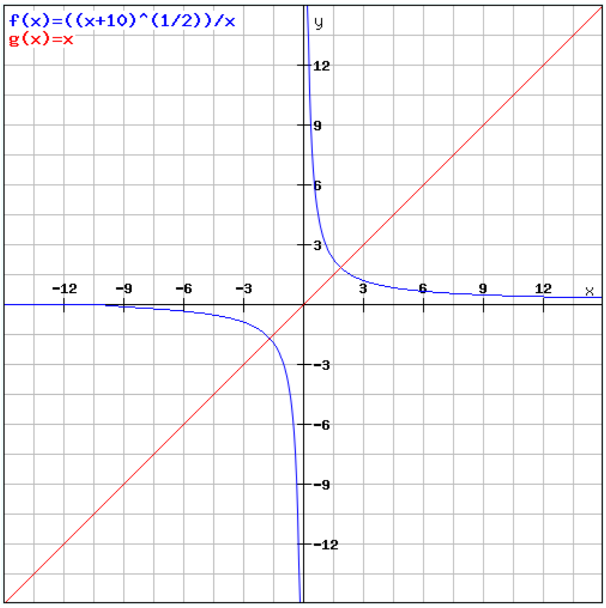
$g_1'(x) < -1$**.** The iteration diverges. It does not give any root.

In area **I and II**, $g_2(x)$ is a rising curve

$0 < g_2'(x) < 1$**.** The iteration converges. It give a root.

In area **III**, $g_1(x)$ is a rising curve

$g_1'(x) > 1$**.** The iteration diverges. It does not give any root.



In area **I** intersection of , $g_3(x)$ and , $y = x$ give a root same as with another convergence , $g_2(x)$. Finally, $g_3(x)$ converge.

05.03.2016
Elif Ak
150130009

Solution 2

The matrix that be evaluated is:

$$A = \begin{bmatrix} 10 & 2 & -1 \\ -3 & -6 & 2 \\ 1 & 1 & 5 \end{bmatrix}$$

First, we need to compute $[L]$ and $[U]$ matrices. To find upper triangular matrix $[U]$

- Multiply the first row by $\mathbf{3/_{10}}$ and add the result to the second tow to eliminate the a_{21} term.
- Then, multiply the first row by $\mathbf{{-1}/_{10}}$ and add the result to third row to eliminate the a_{31} term. The result is
    - $\begin{bmatrix} 10 & 2 & -1 \\ 0 & -5.4 & 1.7 \\ 0 & 0.8 & 5.1 \end{bmatrix}$
- Multiply the second row by ${-0.8}/_{5.4} = \mathbf{0.148148}$ and add the result to third row to eliminate the a_{32} term. Final result for $[U]$ is like following,
    - $U = \begin{bmatrix} 10 & 2 & -1 \\ 0 & -5.4 & 1.7 \\ 0 & 0 & 5.351852 \end{bmatrix}$
- To find lower triangular matrix for $[A]$, can be used coefficients where is found in order to observe $[U]$. Inverse of blue multipliers will be element for L_{21} , L_{31} , L_{32} respectively.
    - $L = \begin{bmatrix} 1 & 0 & 0 \\ -0.3 & 1 & 0 \\ 0.1 & -0.148148 & 0 \end{bmatrix}$
    - $A = LU = \begin{bmatrix} 1 & 0 & 0 \\ -0.3 & 1 & 0 \\ 0.1 & -0.148148 & 0 \end{bmatrix} \begin{bmatrix} 10 & 2 & -1 \\ 0 & -5.4 & 1.7 \\ 0 & 0 & 5.351852 \end{bmatrix}$

$AA^{-1} = I$, so that

- $L.z = c$, $U.x = z$ is repeated for each column.

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

$\begin{bmatrix} 1 & 0 & 0 \\ -0.3 & 1 & 0 \\ 0.1 & -0.148148 & 0 \end{bmatrix} . \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ and it is solved via forward substitution,

z can be found following $\begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0.3 \\ -0.055556 \end{bmatrix}$. This vector is used as the right-hand side of the upper triangular system, $U.x = z$.

$$\begin{bmatrix} 10 & 2 & -1 \\ 0 & -5.4 & 1.7 \\ 0 & 0 & 5.351852 \end{bmatrix} . \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0.3 \\ -0.055556 \end{bmatrix}$$

which can be solved by back substitution and the result will be the first column of the invers matrix A^{-1}.

$$[A]^{-1} = \begin{bmatrix} 0.110727 & ? & ? \\ -0.058824 & ? & ? \\ -0.010381 & ? & ? \end{bmatrix}$$

To determine the second column,

$$\begin{bmatrix} 1 & 0 & 0 \\ -0.3 & 1 & 0 \\ 0.1 & -0.148148 & 0 \end{bmatrix} . \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$ and it is solved via forward substitution,

z can be found following $\begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0.148148 \end{bmatrix}$. And the results are used with $U.x = z$

to determine $[x]$ via back substitution. The second column of the inverse of A,

$$[A]^{-1} = \begin{bmatrix} 0.110727 & 0.038062 & ? \\ -0.058824 & -0.176471 & ? \\ -0.010381 & 0.027682 & ? \end{bmatrix}$$

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

$$[A]^{-1} = \begin{bmatrix} 0.110727 & 0.038062 & 0.00692 \\ -0.058824 & -0.176471 & 0.058824 \\ -0.010381 & 0.027682 & 0.186851 \end{bmatrix}$$

05.03.2016
Elif Ak
150130009

Solution 3

$Let\ \{\sigma_i, v_i\} be\ set\ of\ n\ pairs\ (eigenvalue, eigenvector) corresponding\ A^T A.$

$The\ eigenvectors\ are\ orthonormal\ and\ go\ into\ columns\ of\ V. r\ non - zero\ eigenvalues.$

$A^T A v_i = \sigma_i v_i$

$(Av_i)^T (Av_i) = v_i^T A^T A v_i = v_i^T (A^T A v_i) = v_i^T (\sigma_i v_i) = \sigma_i v_i^T v_i = \sigma_i$

$\|Av_i\|^2 = \sigma_i, \ \ \sigma_i \geq 0$

$For\ the\ positive\ \sigma_i, \lambda_i = \sqrt{\sigma_i}\ and\ u_i = Av_i / \lambda_i$

$$u_i^T u_j = \frac{v_i^T A^T A v_j}{\lambda_i \lambda_j} = \frac{\sigma_j v_i^T v_j}{\lambda_i \lambda_j}$$

$(i, j) \quad U^T A V = u_i^T A v_j$

$Therefore\ the\ only\ non - zeros\ in\ the\ product\ U^T AV\ are\ the\ first\ r\ diagonal\ entries\ (which\ are\ \lambda_1, \lambda_2, \dots \lambda_r).$

$U^T A V = \Lambda, \qquad which\ means\ that\ A = U \Lambda V^T$

05.03.2016
Elif Ak
150130009

Solution 4

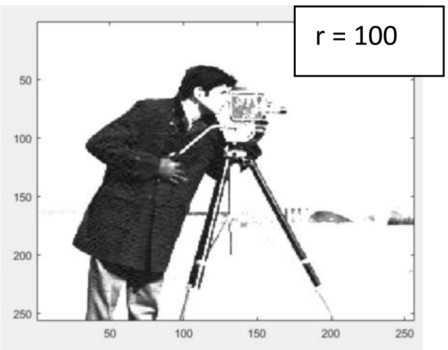
“sample.png” which is 256x256 pixel image corresponds to a 256 by 256 matrix **“imageMatrix”**.

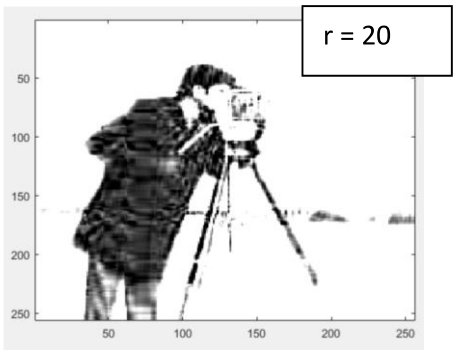
The SVD of A as $A = U\sum V^T$.

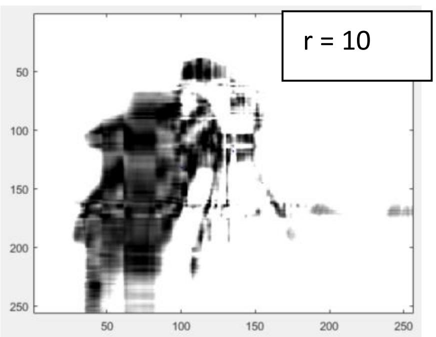
The approximations for various values of k (k = 3, 10, 20, 100) can be shown in MATLAB Code.

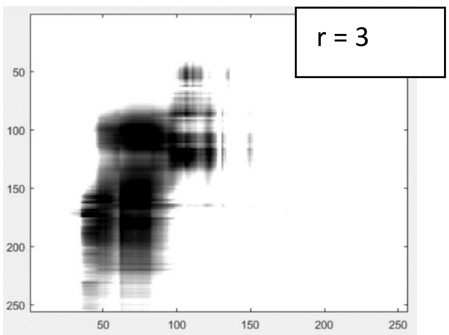
```
r1 = 3;
r2 = 10;
r3 = 20;
r4 = 100;
colormap('gray')
imageMatrix = imread('sample.png');
figure(1)
image(imageMatrix);
imageMatrix = double(imageMatrix);
[U,S,V] = svd(imageMatrix);
figure(2)
colormap('gray')
image(U(:,1:r1)*S(1:r1,1:r1)*V(:,1:r1)')
figure(3)
colormap('gray')
image(U(:,1:r2)*S(1:r2,1:r2)*V(:,1:r2)')
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:









05.03.2016
Elif Ak
150130009

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 $\begin{bmatrix}1\\0\end{bmatrix}$.

After the power iteration, dominant eigenvector will be $\begin{bmatrix}0.5787\\-0.8156\end{bmatrix}$ and corresponding dominant eigenvalue is 498.1708.