Linear Algebra

Assignment

Session: Jan-May 2020

Name : Tanishq Vyas

SRN : PES1201800125

Section : H

List of Programs

1. Gaussian Elimination

2. Inverse of a Matrix by the Gauss- Jordan Method

3. The LU Decomposition

4. The Span of Column Space of a Matrix

5. The Four Fundamental Subspaces,

6. The Gram-Schmidt Orthogonalization

7. Eigen values and Eigen Vectors of a Matrix

8. Projections by Least Squares

9. The Largest Eigen Value of a Matrix by the Power Method.

1. Gaussian Elimination

Source code

clc;clear;close;

A=[6,5,2;8,-4,6;6,2,-1], b = [14;-3;-4]

A\_aug = [A b]

a = A\_aug

n = 3

for i=2:n

for j=2:n+1

a(i,j)=a(i,j)-a(1,j)\*a(i,1)/a(1,1);

end

a(i,1)=0;

end

for i=3:n

for j=3:n+1

a(i,j)=a(i,j)-a(2,j)\*a(i,2)/a(2,2);

end

a(i,2)=0;

end

x(n)=a(n,n+1)/a(n,n); for i=n-1:-1:1

mysum = 0

for k=i+1:n

mysum = mysum + a(i,k)\*x(k);

end

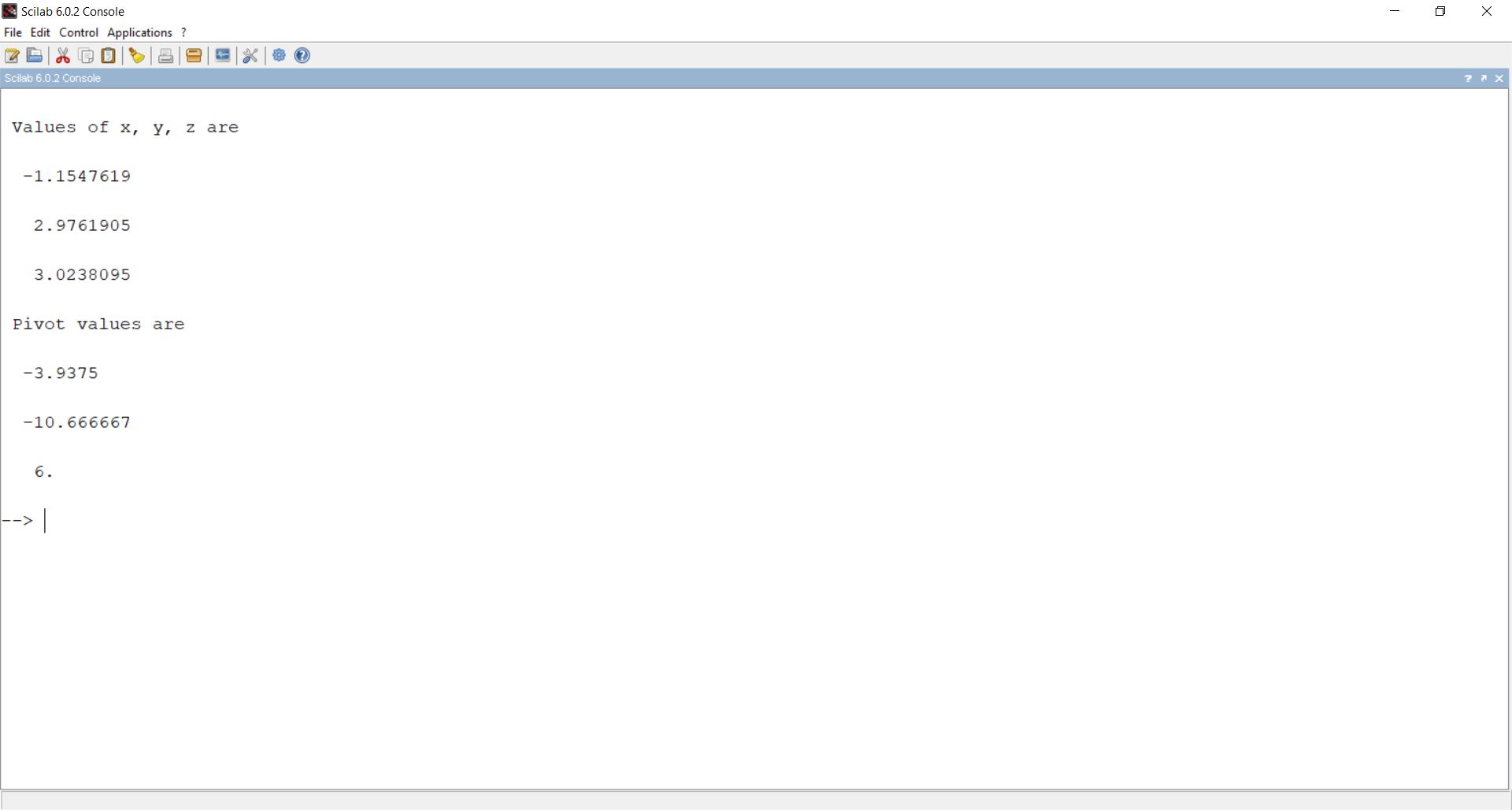
x(i) = (a(i,n+1) - mysum)/a(i,i);

end

disp(x(3), x(2), x(1), "Values of x, y, z are");

disp(a(1,1), a(2,2), a(3,3), "Pivot values are");

Sample Outputs



1. Inverse of a Matrix by the Gauss- Jordan Method

Source Code :

clc;clear;

A = [1 6 7; 3 6 -2; 8 5 3];

n = length ( A (1, :));

Aug = [A, eye(n,n)];

for j = 1:n-1

for i = j+1:n

Aug(1,j:2\*n) = Aug(i,j:2\*n) - Aug(i,j)/Aug(j,j)\*Aug(j,j:2\*n);

end

end

for j = n: -1:2

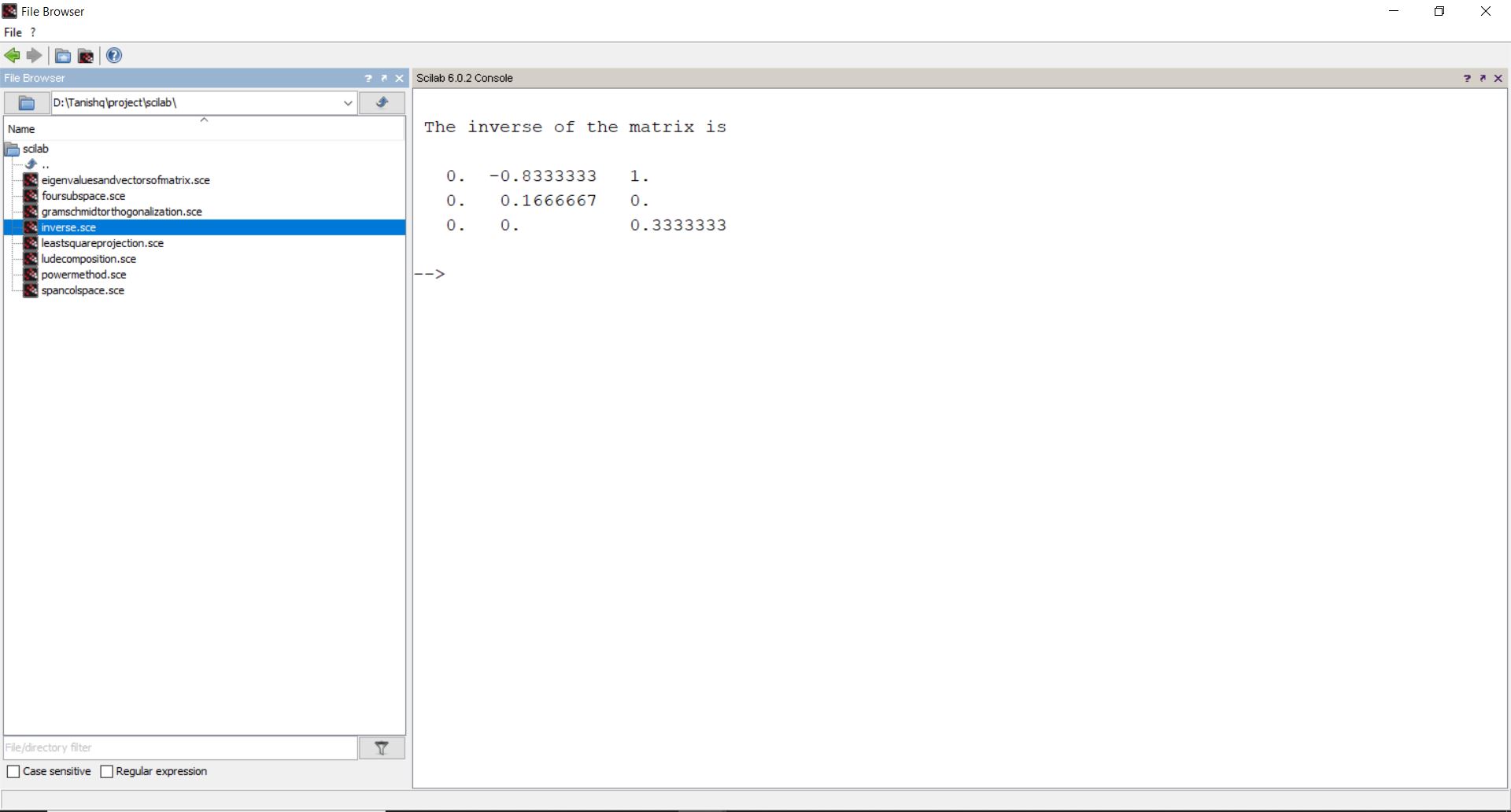
Aug(j,:) = Aug(j,:)/Aug(j,j);

end

B=Aug(:,n+1:2\*n);

disp(B,"The inverse of the matrix is")

Sample Output :



3. The LU Decomposition

Source Code :

A = [6,3,4;0.1, 4, -0.3; 9, 7, 10];

U = A;

disp(A,"The given Matrix is")

m = det(U(1,1));

n = det(U(2,1));

a = n/m;

U(2,:) = U(2,:) - U(1,:)/(m/n);

n = det(U(3,1));

b = n/m;

U(3,:) = U(3,:) - U(1,:) / (m/n);

m = det(U(2,2));

n = det(U(3,2));

c = n/m;

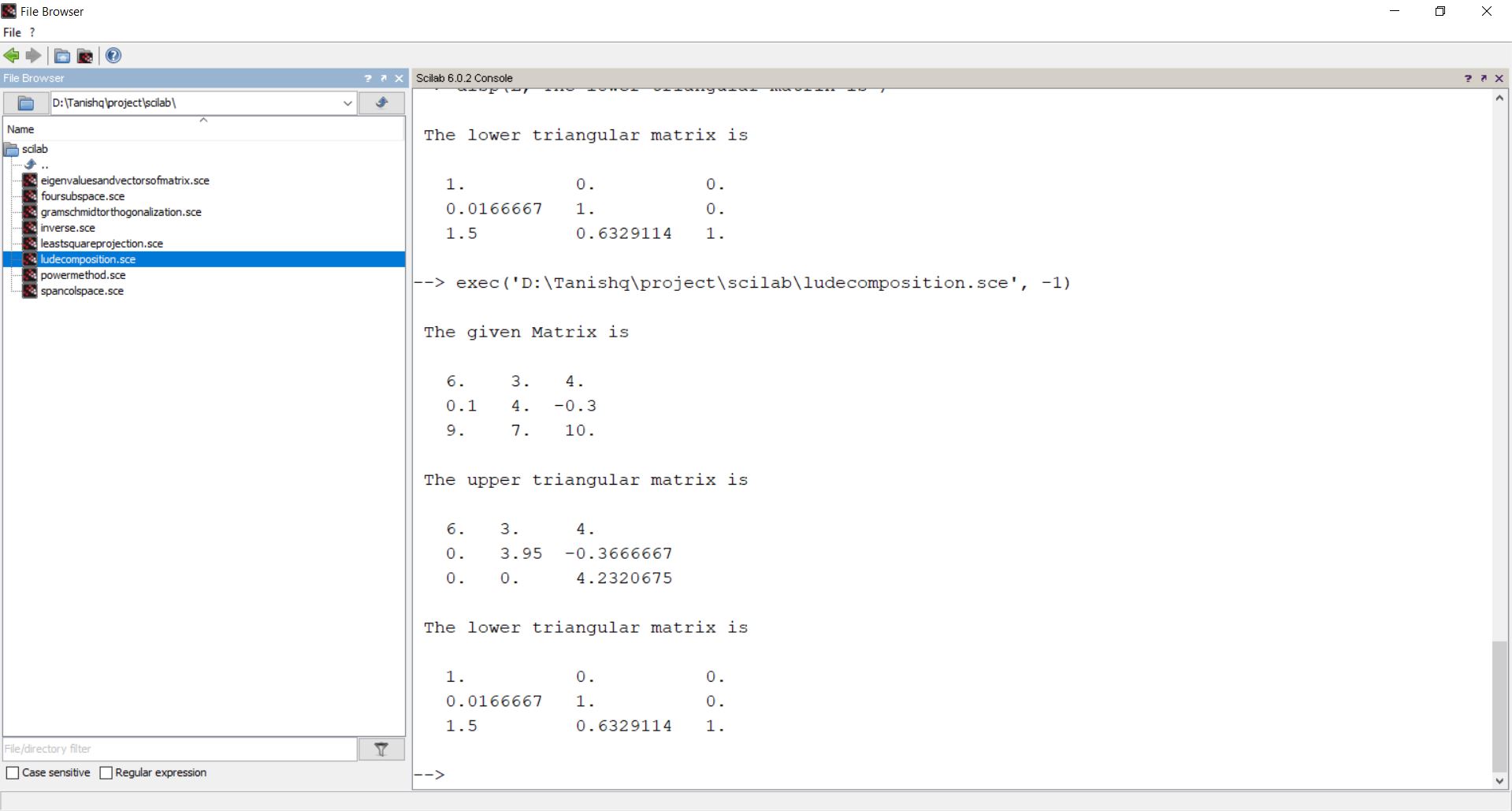
U(3,:) = U(3,:) - U(2,:) / (m/n);

disp(U,"The upper triangular matrix is")

L = [1,0,0; a,1,0;b,c,1];

disp(L,"The lower triangular matrix is")

Sample output



4. The Span of Column Space of a Matrix

Source Code

clc;clear;close;

a = [4 7 2 -2; 3 2 1 7;3 8 4 -3]

a(2,:)=a(2,:)-(a(2,1)/a(1,1))\*a(1,:)

a(3,:)=a(3,:)-(a(3,1)/a(1,1))\*a(1,:)

disp(a)

a(3,:)=a(3,:)-(a(3,2)/a(2,2))\*a(2,:)

disp(a)

a(1,:)=a(1,:)/a(1,1)

a(2,:)=a(2,:)/a(2,2)

disp(a)

for i=1:3

for j=i:4

if(a(i,j) <> 0)

disp("Is a pivot column",j,"Columns")

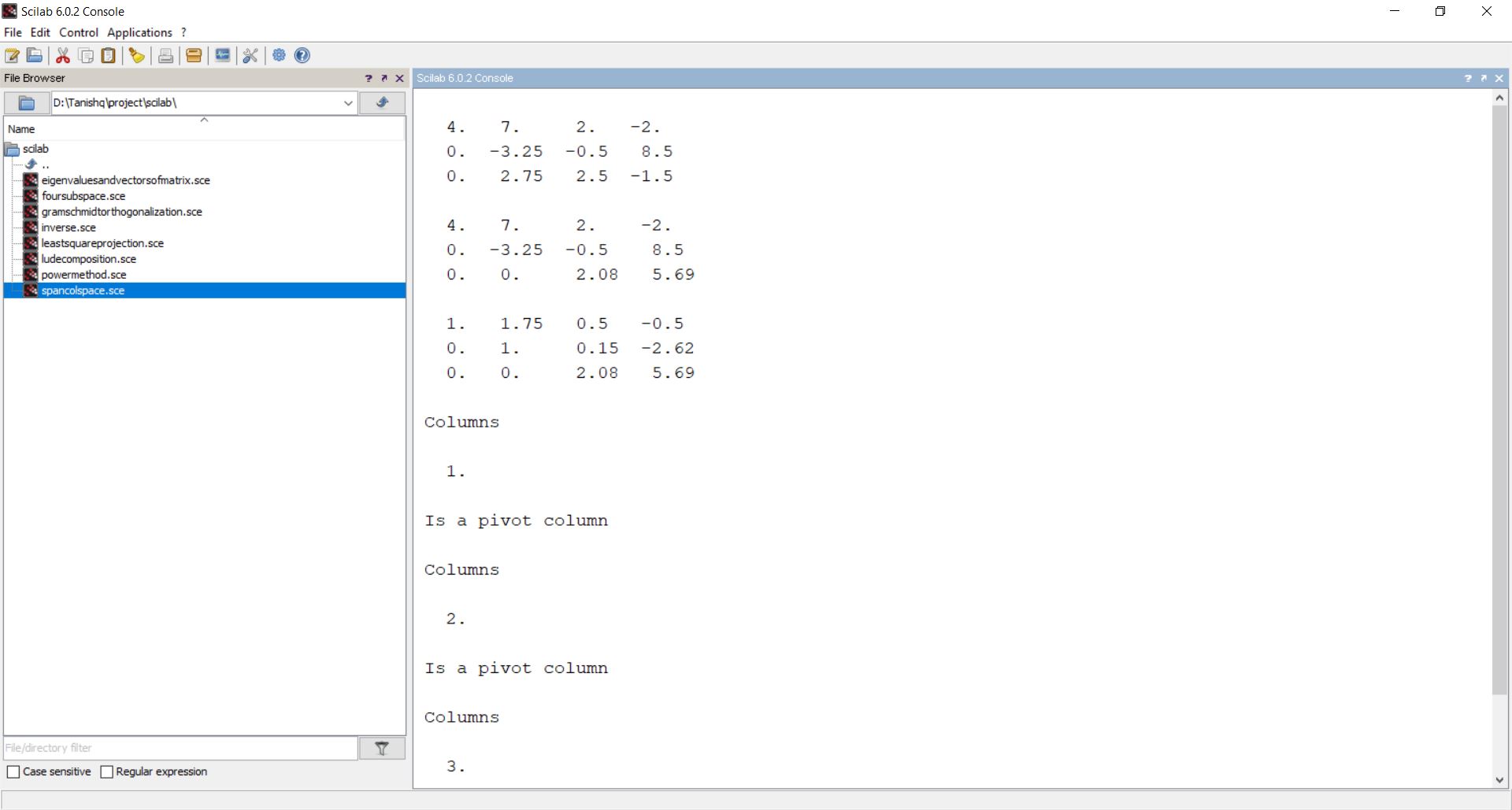
break

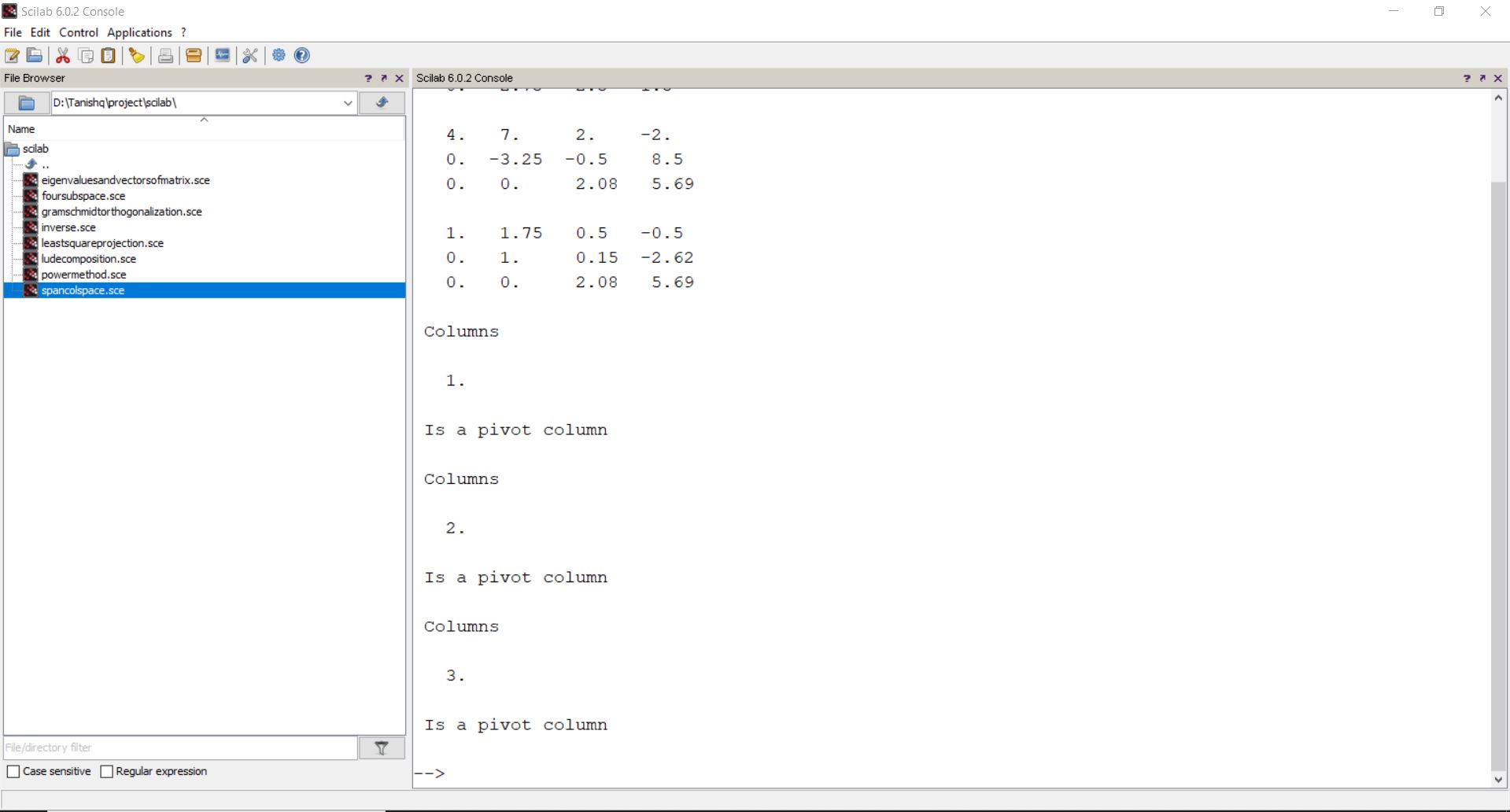
end

end

end

Sample Output





5. The Four Fundamental Subspaces

Source Code

clear;close;clc;

A = [3 1 0;5 3 1;0 0 0];

disp(A,"A=");

[m,n] = size(A);

disp(m,"m=");

disp(n,"n=");

[v, pivot] = rref(A);

disp(rref(A));

disp(v);

r=length(pivot);

disp(r,"Rank =")

cs=A(:,pivot);

disp(cs,"column space =");

ns=kernel(A);

disp(ns,"null space =");

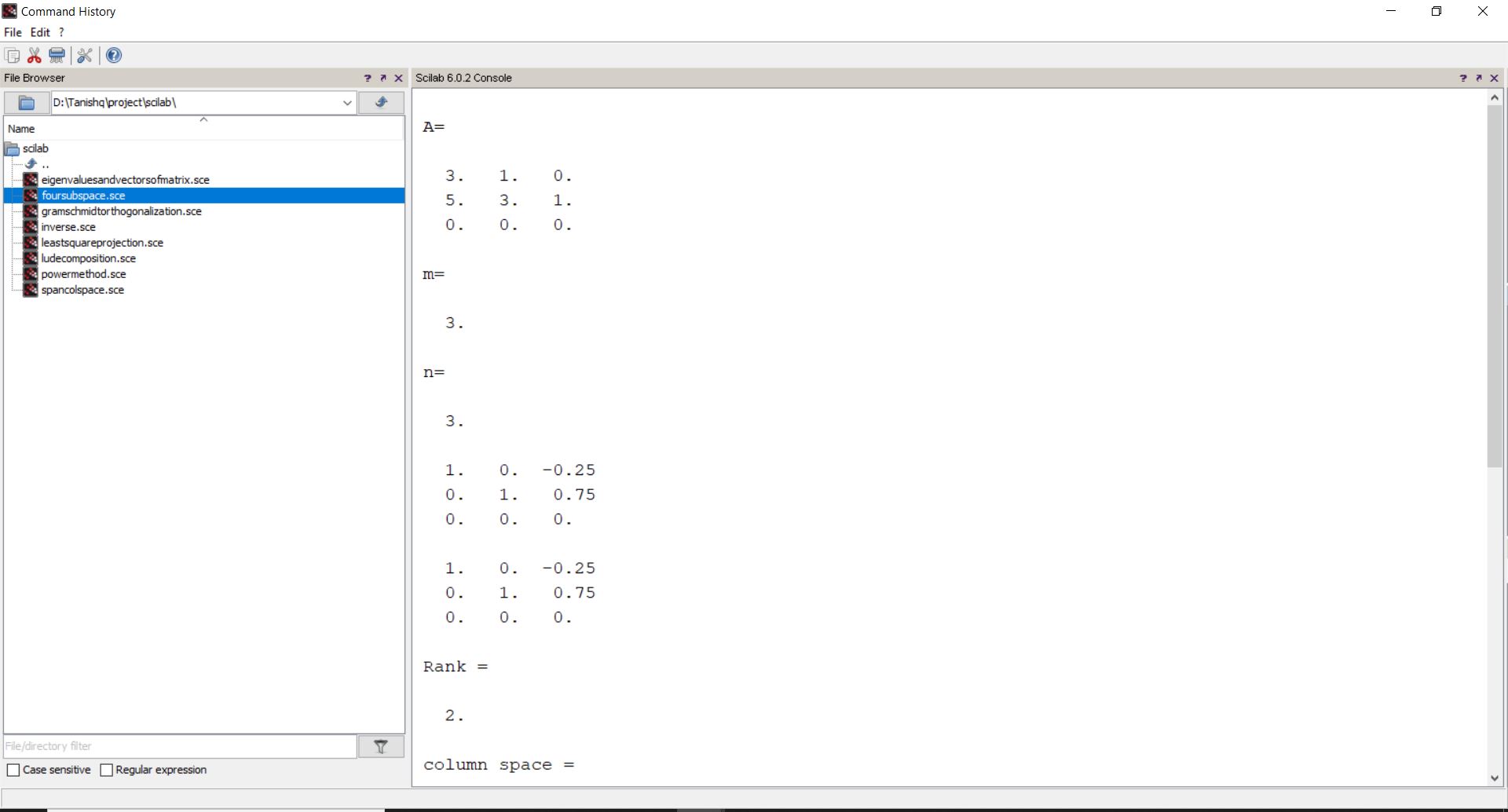
rs=v(1:r,:)';

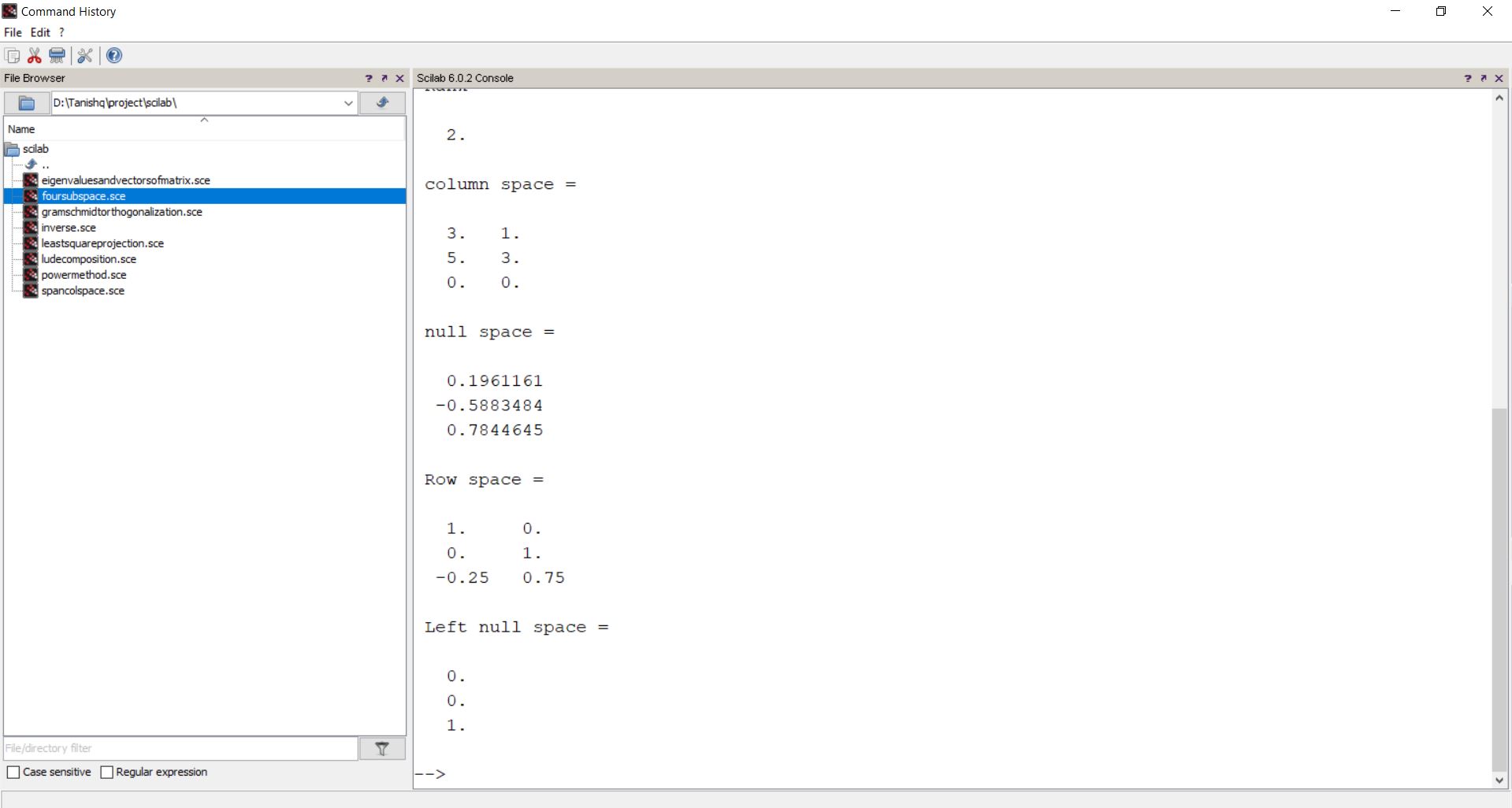
disp(rs,"Row space =");

lns=kernel(A');

disp(lns,"Left null space =");

Sample Output





6. The Gram-Schmidt Orthogonalization

Source Code

clear;close;clc;

A=[3 6 1;1 9 3;8 1 0];

disp(A,"A=");

[m,n]=size(A);

for k=1:n

V(:,k)=A(:,k);

for j=1:k-1

R(j,k)=V(:,j)'\*A(:,k);

V(:,k)=V(:,k)-R(j,k)\*V(:,j);

end

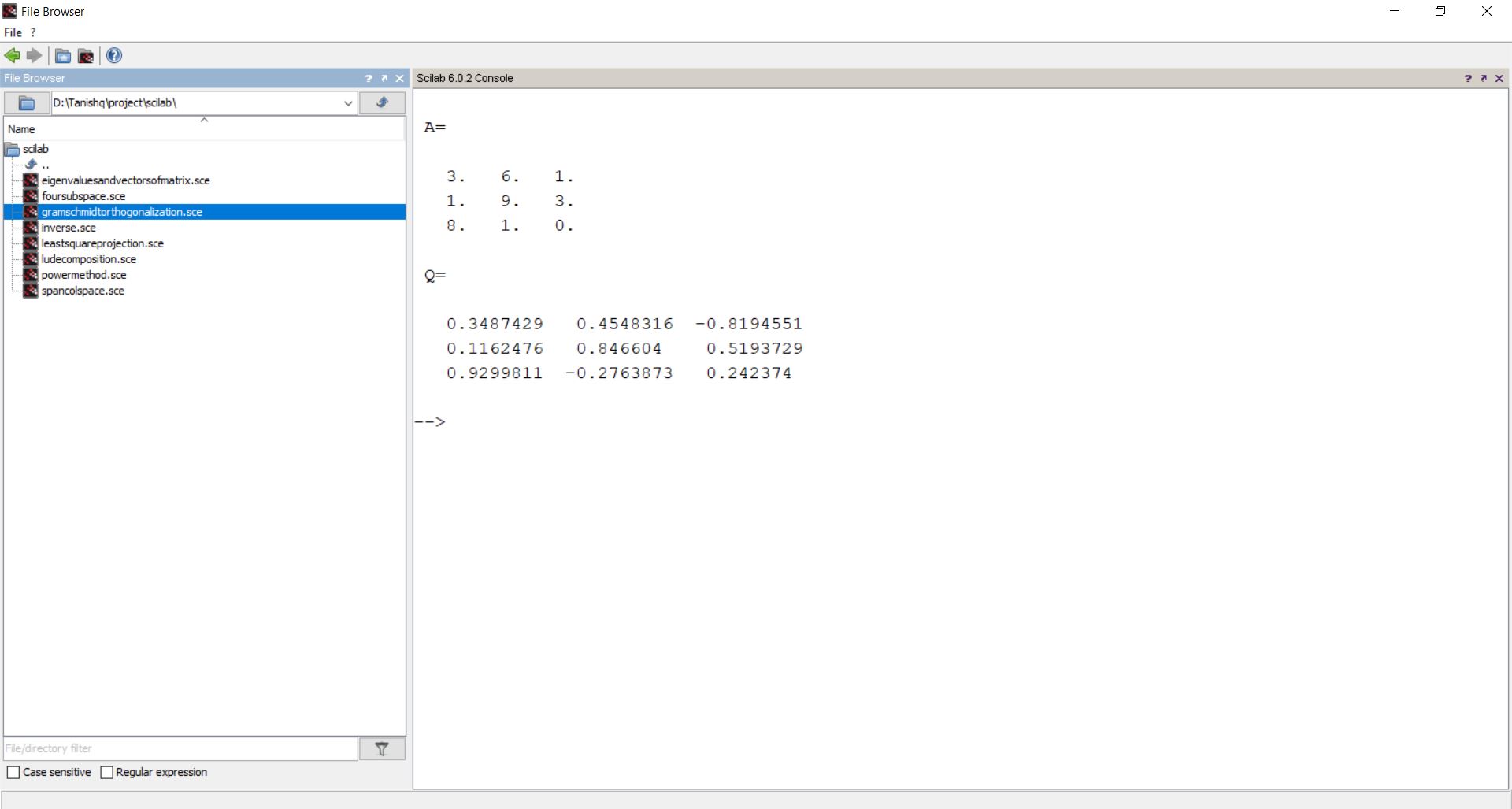
R(k,k)=norm(V(:,k));

V(:,k)=V(:,k)/R(k,k);

end

disp(V,"Q=");

Sample Output



7. Eigen values and Eigen Vectors of a Matrix

Source Code

clc;close;clear;

A=[6,-1,7;-7,3,5;7,1,4]

lam=poly(0,'lam')

lam=lam

charMat=A-lam\*eye(3,3)

disp(charMat,"the characteristic Matrix is")

charPoly=poly(A,"lam")

disp(charPoly,"the characteristic polynomial is")

lam=spec(A)

disp(lam,"the eigne values of A are")

function[x,lam]=eigenvectors(A)

[n,m]=size(A);

lam=spec(A)';

x=[];

for k=1:3

B=A-lam(k)\*eye(3,3);

C=B(1:n-1,1:n-1);

b=-B(1:n-1,n);

y=C\b;

y=[y;1];

y=y/norm(y);

x=[x y];

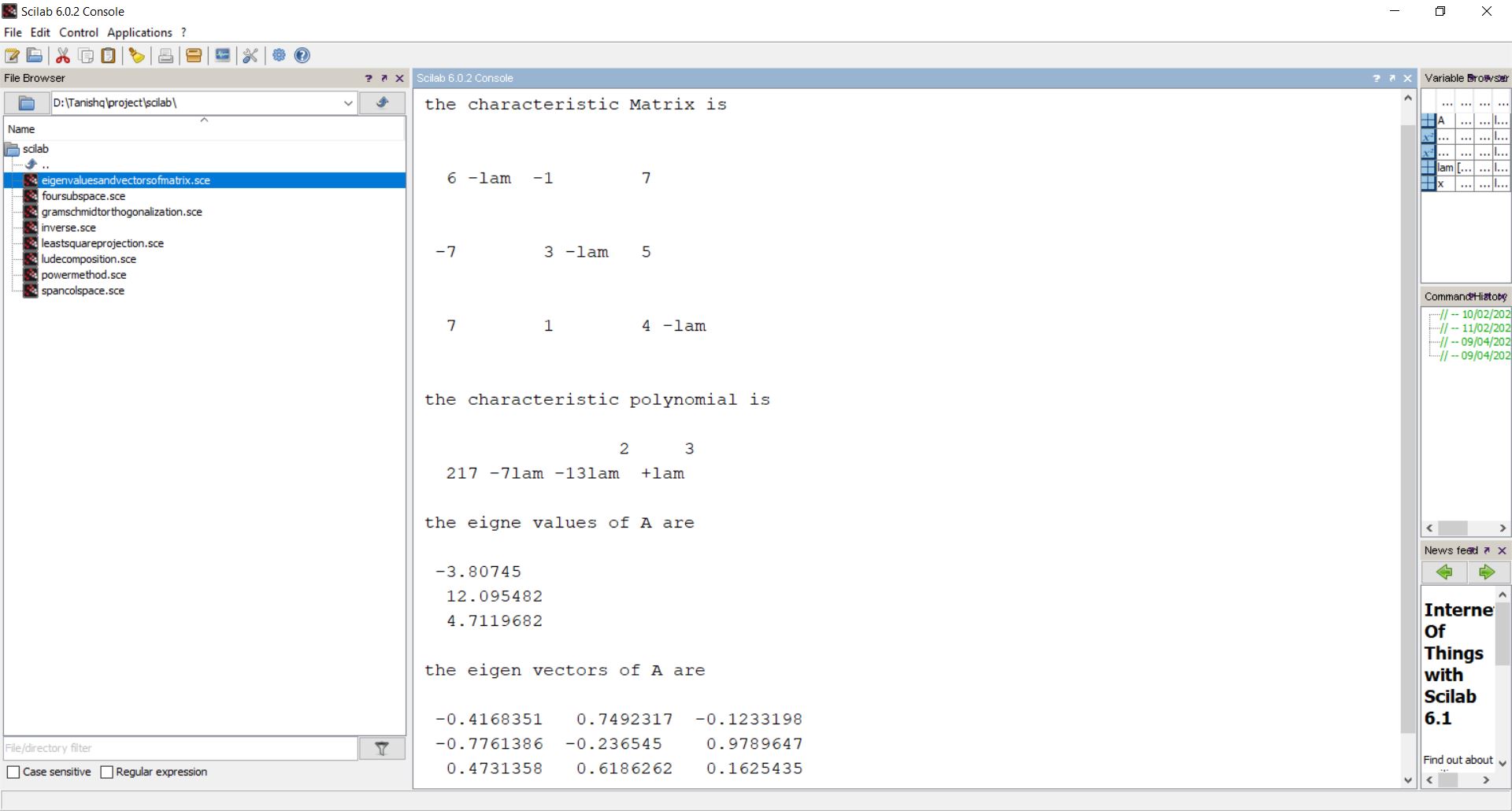
end

endfunction

[x,lam] = eigenvectors(A)

disp(x,"the eigen vectors of A are")

Sample Output



8. Projections by Least Squares

Source Code

clear;close;clc;

A=[3 -3;1 1;1 2];

disp(A,'A=');

b=[1;1;3];

disp(b,"b=");

x=(A'\*A)\(A'\*b);

disp(x,"x=");;

C=x(1,1);

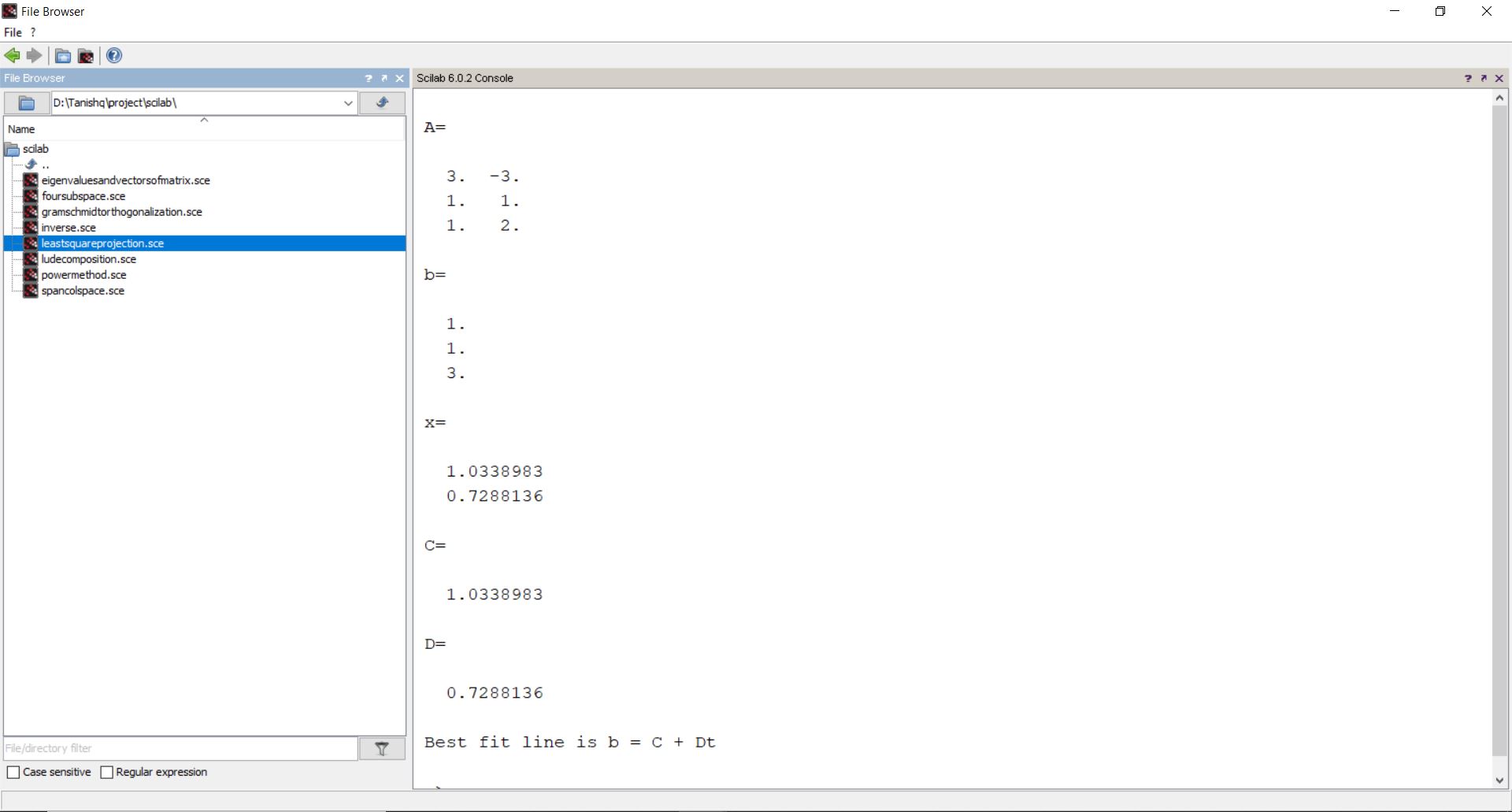
D=x(2,1);

disp(C,"C=");

disp(D,"D=");

disp("Best fit line is b = C + Dt")

Sample Output



9. The Largest Eigen Value of a Matrix by the Power Method.

Source Code

clear;clc;close();

A=[6 3 5;2 7 2;7 2 9];

disp(A,"The given matrix is")

u0=[1 1 1]';

disp(u0,"The initial vector is")

v=A\*u0;

a=max(u0);

disp(a,"First approximation to eigen value is");

while abs(max(v)-a)>0.002

disp(v,"Current eigen vector is");

a = max(v);

disp(a,"Current eigen value is")

u0=v/max(v);

v=A\*u0;

end

format('v',4);

disp(max(v),"the largest eigen value is")

format('v',5)

disp(u0,"The corresponding Eigen Vector is")

Sample Output

