MATLAB PROJECT 3

Please include this page in your Group file, as a front page. Type in the group number and the names of all members WHO PARTICIPATED in this project.

GROUP # \_\_\_\_3\_\_\_\_\_

FIRST & LAST NAMES (UFID numbers are NOT required):

\_1.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Kevin Nguyen \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_2.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Chamara Gunaratne\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ­­­

\_3.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Cayle Gao\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_4.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Carl Lin\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_5.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Nicolas Miller\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_6.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**By signing your names above, each of you had confirmed that you did the work and agree with the work submitted**.

Project 3

% Exercise 1

type rowspace

function [] = rowspace(A,B)

%ROWSPACE Determines whether Row A and Row B are subspaces of the same space.

% If the two matrices are of the same space it then determines

% if the dimensions of the two matrices are the same. If that is

% true then it determines if the two matrices are row equivelent.

format compact

format rat

m=size(A,2);

n=size(B,2);

k=rank(A);

p=rank(B);

fprintf('dim of Row A is k = %i\n', k)

fprintf('dim of Row B is p = %i\n', p)

if (k == m)

fprintf('k = m (%i=%i) Row A is all R^%i\n',k,m,m)

else

fprintf('k ~= m (%i~=%i) Row A is not all R^%i\n',k,m,m)

end

if (p == n)

fprintf('p = n (%i=%i) Row B is all R^%i\n',p,n,n)

else

fprintf('p ~= n (%i~=%i) Row B is not all R^%i\n',p,n,n)

end

if m~=n

fprintf('Row A and Row B are subspaces of different spaces')

else

if (k ~= p)

fprintf('k ~ = p (%i~=%i) the dimensions of Row A and Row B are different',k,p)

else

if rank([A;B]) ~= k

fprintf('k = p (%i=%i) but Row A ~= Row B',k,p)

else

fprintf('k = p (%i=%i) and Row A = Row B',k,p)

end

end

end

end

C=[2 6 2 4 -6;-4 -9 -7 -2 3;-2 -5 -3 -2 3;3 8 9 -1 4];

A=C; B=rref(A);

rowspace(A,B)

dim of Row A is k = 3

dim of Row B is p = 3

k ~= m (3~=5) Row A is not all R^5

p ~= n (3~=5) Row B is not all R^5

k = p (3=3) and Row A = Row B

% Matrix B is the reduced echelon form of matrix A, and because

% row operation do not affect the row space of a matrix,

% Row A and Row B are of the same space and equivalent.

A=C';, B=B';

rowspace(A,B)

dim of Row A is k = 3

dim of Row B is p = 3

k ~= m (3~=4) Row A is not all R^4

p ~= n (3~=4) Row B is not all R^4

k = p (3=3) but Row A ~= Row B

% By transposing both matrices you are effectively working with

% the columns of each matrix. Row operations alter the column

% space of a matrix, but preserve the linear relationship

% between rows. This is why the dimension of both matrices are

% the same, but both matrices are not row equivelent.

% Row operations do not affect the row space of a matrix.

% Row operations affect the column space of a matrix.

A=eye(3);B=eye(4);

rowspace(A,B)

dim of Row A is k = 3

dim of Row B is p = 4

k = m (3=3) Row A is all R^3

p = n (4=4) Row B is all R^4

Row A and Row B are subspaces of different spaces

A=magic(4);B=eye(4);

rowspace(A,B)

dim of Row A is k = 3

dim of Row B is p = 4

k ~= m (3~=4) Row A is not all R^4

p = n (4=4) Row B is all R^4

k ~ = p (3~=4) the dimensions of Row A and Row B are different

A=magic(4);B=eye(3);

rowspace(A,B)

dim of Row A is k = 3

dim of Row B is p = 3

k ~= m (3~=4) Row A is not all R^4

p = n (3=3) Row B is all R^3

Row A and Row B are subspaces of different spaces

A=magic(5);B=eye(5);

rowspace(A,B)

dim of Row A is k = 5

dim of Row B is p = 5

k = m (5=5) Row A is all R^5

p = n (5=5) Row B is all R^5

k = p (5=5) and Row A = Row B

%Exercise 2

type shrink

function B = shrink(A)

[~,pivot] = rref(A);

B= A(: , pivot);

end

format rat

A=magic(4), R=rref(transpose(A)), M=colspace(sym(A))

A =

16 2 3 13

5 11 10 8

9 7 6 12

4 14 15 1

R =

1 0 0 1

0 1 0 3

0 0 1 -3

0 0 0 0

M =

[ 1, 0, 0]

[ 0, 1, 0]

[ 0, 0, 1]

[ 1, 3, -3]

%The colspace function uses the pivot rows of the row reduced transpose of the input matrix to give a basis for Col A

A=[ 2 -4 -2 3; 6 -9 -5 8; 2 -7 -3 9; 4 -2 -2 -1; -6 3 3 4]

A =

2 -4 -2 3

6 -9 -5 8

2 -7 -3 9

4 -2 -2 -1

-6 3 3 4

B=shrink(A), M=colspace(sym(A))

B =

2 -4 3

6 -9 8

2 -7 9

4 -2 -1

-6 3 4

M =

[ 1, 0, 0]

[ 0, 1, 0]

[ 0, 0, 1]

[ 0, 1, -1]

[ -2, -1, 2]

%Both B and M form bases with dimension 3 because they have three columns

rref(transpose(B)), rref(transpose(M)), rref(transpose(A))

ans =

Columns 1 through 4

1 0 0 0

0 1 0 1

0 0 1 -1

Column 5

-2

-1

2

ans =

[ 1, 0, 0, 0, -2]

[ 0, 1, 0, 1, -1]

[ 0, 0, 1, -1, 2]

ans =

Columns 1 through 4

1 0 0 0

0 1 0 1

0 0 1 -1

0 0 0 0

Column 5

-2

-1

2

0

%This method demonstrates that the columns of the matrices all reduce to the same matrix.

%As a result, matrices B and M are both column spaces of A.

%The column reduction of matrix A has an extra row of zeroes becuase it is not a basis and therefore does not have linearly independent columns.

format compact

type noll

function C = homobasis(A)

m = size(A,1);

n = size(A,2);

format rat

red\_ech\_form = rref(A);

[~, pivot\_c] = rref(A);

S = 1:n;

P = setdiff(S,pivot\_c);

q = length(P);

p\_rows = any(red\_ech\_form,2);

B = -red\_ech\_form(p\_rows,P);

if q == 0

C = zeros(n,1);

else

C(pivot\_c,:) = B;

C(P,:) = eye(q);

end

end

C=noll(A)

C =

1/3

-1/3

1

0

Z=null(A,'r')

Z =

1/3

-1/3

1

0

%The two outputs match. Both methods give the same basis.

M=colspace(sym(transpose(A)))

M =

[ 1, 0, 0]

[ 0, 1, 0]

[ -1/3, 1/3, 0]

[ 0, 0, 1]

p=size(C,2)

p =

1

%This is the dimension of the null space of A

q=size(M,1)

q =

4

%This is the dimension of the row space of A

format compact

rank(A) + p == size(A,2)

ans =

1

%Since the sum of the number of pivot positions and the dimension of the null space

%is equal to the number of columns in A the Rank Theorem holds for matrix A

transpose(M)\*C

ans =

0

0

0

%This method multiplies the columns of M (the row space)

%by the column of C (the null space) and essentially computes the

%dot product for each pair of vectors between the two sets.

%Since all the values are 0 in the result, the null space and

%row space are orthogonal complements.

%It was shown previously that their dimensions summed to the number of columns in A.

% Exercise 3

type closetozeroroundoff

function B=closetozeroroundoff(A)

[m,n]=size(A);

for i=1:m

for j=1:n

if abs(A(i,j))<10^(-7)

A(i,j)=0;

end

end

end

B=A;

end

type polyspace

function P = polyspace(B, Q, r)

format rat

u=sym2poly(B(1));

n=length(u);

C=zeros(n);

for i=1: n

C( : , i) = transpose(sym2poly(B(i)));

end

P=closetozeroroundoff(C);

if rank(P) ~= n

sprintf('the polynomials do not form a basis for P%d',n-1)

fprintf('the reduced echelon form of P is \n')

A=rref(P)

return

else

q = transpose(sym2poly(Q));

y = inv(P)\*q;

fprintf('the coordinates of Q with respect to the basis B is \n')

y = closetozeroroundoff(y)

z = P\*r;

fprintf('the polynomial, whose B-coordinates form the vector r, is \n')

R=poly2sym(z)

end

end

syms x

% (A)

B=[x^3+3\*x^2,10^(-8)\*x^3+x,10^(-8)\*x^3+4\*x^2+x,x^3+x] ,

B =

[ x^3 + 3\*x^2, x^3/100000000 + x, x^3/100000000 + 4\*x^2 + x, x^3 + x]

Q=10^(-8)\*x^3+x^2+6\*x, r =[2;3;-1;1]

Q =

x^3/100000000 + x^2 + 6\*x

r =

2

3

-1

1

P=polyspace(B, Q, r)

ans =

'the polynomials do not form a basis for P3'

the reduced echelon form of P is

A =

1 0 0 1

0 1 0 7/4

0 0 1 -3/4

0 0 0 0

P =

1 0 0 1

3 0 4 0

0 1 1 1

0 0 0 0

% (B)

B=[x^3-1,10^(-8)\*x^3+2\*x^2,10^(-8)\*x^3+x,x^3+x]

B =

[ x^3 - 1, x^3/100000000 + 2\*x^2, x^3/100000000 + x, x^3 + x]

P=polyspace(B, Q, r)

the coordinates of Q with respect to the basis B is

y =

0

1/2

6

0

the polynomial, whose B-coordinates form the vector r, is

R =

3\*x^3 + 6\*x^2 - 2

P =

1 0 0 1

0 2 0 0

0 0 1 1

-1 0 0 0

% (C)

B=[x^4+x^3+x^2+1,10^(-8)\*x^4+x^3+x^2+x+1,10^(-8)\*x^4+x^2+x+1, 10^(-8)\*x^4+x+1,10^(-8)\*x^4+1],

B =

[ x^4 + x^3 + x^2 + 1, x^4/100000000 + x^3 + x^2 + x + 1, x^4/100000000 + x^2 + x + 1, x^4/100000000 + x + 1, x^4/100000000 + 1]

Q=x^4+x^3+1, r=randi(10,5,1)

Q =

x^4 + x^3 + 1

r =

9

10

2

10

7

P=polyspace(B, Q, r)

the coordinates of Q with respect to the basis B is

y =

1

0

-1

1

0

the polynomial, whose B-coordinates form the vector r, is

R =

9\*x^4 + 19\*x^3 + 21\*x^2 + 22\*x + 38

P =

1 0 0 0 0

1 1 0 0 0

1 1 1 0 0

0 1 1 1 0

1 1 1 1 1

% Exercise 4

type closetozeroroundoff

function B = closetozeroroundoff(A)

[m,n]=size(A);

for i=1:m

for j=1:n

if abs(A(i,j))<10^(-7)

A(i,j)=0;

end

end

end

B=A;

end

type quer

function [] = quer(A)

% The purpose of this function is to

% verify that the input gets factorization

% and also that Q is unitary and R is

% upper triangular.

[Q,R]=qr(A)

S = inv(Q);

if closetozeroroundoff(A-Q\*R)==0

disp('the product of Q and R forms a QR decomposition of A')

else

disp('No, it cannot be true!')

end

if isequal(S,transpose(Q)) && isequal(R,triu(A))

disp('Q\*R forms an orthogonal-triangular decomposition of A')

else

disp('Something is wrong with this picture?')

end

end

% A

A=randi(10,3,4)

A =

9 10 3 10

10 7 6 2

2 1 10 10

quer(A)

Q =

-0.6617 0.7427 0.1031

-0.7352 -0.6157 -0.2835

-0.1470 -0.2633 0.9534

R =

-13.6015 -11.9105 -7.8668 -9.5578

0 2.8532 -4.0998 3.5617

0 0 8.1428 9.9981

the product of Q and R forms a QR decomposition of A

Something is wrong with this picture?

% B

A=ones(5,3)

A =

1 1 1

1 1 1

1 1 1

1 1 1

1 1 1

quer(A)

Q =

-0.4472 0.8944 -0.0000 -0.0000 -0.0000

-0.4472 -0.2236 -0.5000 -0.5000 -0.5000

-0.4472 -0.2236 0.8333 -0.1667 -0.1667

-0.4472 -0.2236 -0.1667 0.8333 -0.1667

-0.4472 -0.2236 -0.1667 -0.1667 0.8333

R =

-2.2361 -2.2361 -2.2361

0 -0.0000 -0.0000

0 0 0

0 0 0

0 0 0

the product of Q and R forms a QR decomposition of A

Something is wrong with this picture?

% C

A=ones(4,4)

A =

1 1 1 1

1 1 1 1

1 1 1 1

1 1 1 1

quer(A)

Q =

-0.5000 -0.5000 -0.5000 -0.5000

-0.5000 0.8333 -0.1667 -0.1667

-0.5000 -0.1667 0.8333 -0.1667

-0.5000 -0.1667 -0.1667 0.8333

R =

-2 -2 -2 -2

0 0 0 0

0 0 0 0

0 0 0 0

the product of Q and R forms a QR decomposition of A

Something is wrong with this picture?

% D

A=diag([1,2,3,4])

A =

1 0 0 0

0 2 0 0

0 0 3 0

0 0 0 4

quer(A)

Q =

1 0 0 0

0 1 0 0

0 0 1 0

0 0 0 1

R =

1 0 0 0

0 2 0 0

0 0 3 0

0 0 0 4

the product of Q and R forms a QR decomposition of A

Q\*R forms an orthogonal-triangular decomposition of A

% E

A=triu(magic(4))

A =

16 2 3 13

0 11 10 8

0 0 6 12

0 0 0 1

quer(A)

Q =

1 0 0 0

0 1 0 0

0 0 1 0

0 0 0 1

R =

16 2 3 13

0 11 10 8

0 0 6 12

0 0 0 1

the product of Q and R forms a QR decomposition of A

Q\*R forms an orthogonal-triangular decomposition of A

% F

A=tril(magic(4))

A =

16 0 0 0

5 11 0 0

9 7 6 0

4 14 15 1

quer(A)

Q =

-0.8230 0.4356 -0.0336 0.3632

-0.2572 -0.5144 0.8107 0.1093

-0.4629 -0.1690 -0.1382 -0.8591

-0.2057 -0.7191 -0.5679 0.3436

R =

-19.4422 -8.9496 -5.8635 -0.2057

0 -16.9087 -11.8001 -0.7191

0 0 -9.3476 -0.5679

0 0 0 0.3436

the product of Q and R forms a QR decomposition of A

Something is wrong with this picture?

% G

A=triu(tril(rand(6),1),-1)

A =

0.9572 0.7922 0 0 0 0

0.4854 0.9595 0.7577 0 0 0

0 0.6557 0.7431 0.2769 0 0

0 0 0.3922 0.0462 0.0344 0

0 0 0 0.0971 0.4387 0.4456

0 0 0 0 0.3816 0.6463

quer(A)

Q =

-0.8919 0.2733 -0.0800 -0.3065 0.1210 0.1218

-0.4523 -0.5390 0.1577 0.6045 -0.2386 -0.2402

0 -0.7967 -0.1341 -0.5142 0.2029 0.2043

0 0 0.9751 -0.1936 0.0764 0.0769

0 0 0 -0.4886 -0.6149 -0.6190

0 0 0 0 -0.7094 0.7048

R =

-1.0732 -1.1405 -0.3427 0 0 0

0 -0.8231 -1.0005 -0.2206 0 0

0 0 0.4023 0.0079 0.0336 0

0 0 0 -0.1988 -0.2210 -0.2177

0 0 0 0 -0.5378 -0.7325

0 0 0 0 0 0.1797

the product of Q and R forms a QR decomposition of A

Something is wrong with this picture?

% H

A=[1 1 4;0 -4 0;-5 -1 -8]

A =

1 1 4

0 -4 0

-5 -1 -8

quer(A)

Q =

-0.1961 0.1887 0.9623

0 -0.9813 0.1925

0.9806 0.0377 0.1925

R =

-5.0990 -1.1767 -8.6291

0 4.0762 0.4529

0 0 2.3094

the product of Q and R forms a QR decomposition of A

Something is wrong with this picture?

%Exercise 5

% (Part I:)

%(A)

A=[1 1 4;0 -4 0;-5 -1 -8]

A =

1 1 4

0 -4 0

-5 -1 -8

[L,U,invA] = eluinv(A)

Yes, I have got LU factorization

U is an echelon form of A

Yes, LU factorization works for calculating the inverses

L =

-0.2000 -0.2000 1.0000

0 1.0000 0

1.0000 0 0

U =

-5.0000 -1.0000 -8.0000

0 -4.0000 0

0 0 2.4000

invA =

-0.6667 -0.0833 -0.3333

0 -0.2500 0

0.4167 0.0833 0.0833

% (B)

A = magic(4)

A =

16 2 3 13

5 11 10 8

9 7 6 12

4 14 15 1

[L,U,invA] = eluinv(A)

Yes, I have got LU factorization

U is an echelon form of A

ans =

'A is not invertible'

L =

1.0000 0 0 0

0.3125 0.7685 1.0000 0

0.5625 0.4352 1.0000 1.0000

0.2500 1.0000 0 0

U =

16.0000 2.0000 3.0000 13.0000

0 13.5000 14.2500 -2.2500

0 0 -1.8889 5.6667

0 0 0 0.0000

invA =

[]

% (c)

A=[2 1 -3 1;0 5 -3 5;-4 3 3 3;-2 5 1 3]

A =

2 1 -3 1

0 5 -3 5

-4 3 3 3

-2 5 1 3

[L,U,invA] = eluinv(A)

Yes, I have got LU factorization

U is an echelon form of A

ans =

'A is not invertible'

L =

-0.5000 0.5000 0 1.0000

0 1.0000 0 0

1.0000 0 0 0

0.5000 0.7000 1.0000 0

U =

-4.0000 3.0000 3.0000 3.0000

0 5.0000 -3.0000 5.0000

0 0 1.6000 -2.0000

0 0 0 0

invA =

[]

% (d)

A=magic(3)

A =

8 1 6

3 5 7

4 9 2

[L,U,invA] = eluinv(A)

Yes, I have got LU factorization

U is an echelon form of A

Yes, LU factorization works for calculating the inverses

L =

1.0000 0 0

0.3750 0.5441 1.0000

0.5000 1.0000 0

U =

8.0000 1.0000 6.0000

0 8.5000 -1.0000

0 0 5.2941

invA =

0.1472 -0.1444 0.0639

-0.0611 0.0222 0.1056

-0.0194 0.1889 -0.1028

% (Part II:)

%(A)

A=[1 1 4;0 -4 0;-5 -1 -8], B=randi(10,3,4)

A =

1 1 4

0 -4 0

-5 -1 -8

B =

10 2 8 1

5 5 10 9

9 10 7 10

[X1,X2] = msystem(A,B)

The solutions are the same

X1 =

-10.0833 -5.0833 -8.5000 -4.7500

-1.2500 -1.2500 -2.5000 -2.2500

5.3333 2.0833 4.7500 2.0000

X2 =

-10.0833 -5.0833 -8.5000 -4.7500

-1.2500 -1.2500 -2.5000 -2.2500

5.3333 2.0833 4.7500 2.0000

%(b)

A=magic(3), B=[magic(3),eye(3)]

A =

8 1 6

3 5 7

4 9 2

B =

8 1 6 1 0 0

3 5 7 0 1 0

4 9 2 0 0 1

[X1,X2] = msystem(A,B)

The solutions are the same

X1 =

1.0000 0 -0.0000 0.1472 -0.1444 0.0639

0 1.0000 0 -0.0611 0.0222 0.1056

0 0.0000 1.0000 -0.0194 0.1889 -0.1028

X2 =

1.0000 0 0 0.1472 -0.1444 0.0639

0 1.0000 0 -0.0611 0.0222 0.1056

0 0 1.0000 -0.0194 0.1889 -0.1028

%(c)

A=magic(5), B=randi(10,5,3)

A =

17 24 1 8 15

23 5 7 14 16

4 6 13 20 22

10 12 19 21 3

11 18 25 2 9

B =

7 2 1

8 8 9

8 1 7

4 3 4

7 1 10

[X1,X2] = msystem(A,B)

The solutions are the same

X1 =

0.1200 0.3708 0.2463

0.0277 -0.1772 -0.2594

0.1046 0.0128 0.3954

-0.0185 0.0612 -0.1748

0.2892 -0.0369 0.2694

X2 =

0.1200 0.3708 0.2463

0.0277 -0.1772 -0.2594

0.1046 0.0128 0.3954

-0.0185 0.0612 -0.1748

0.2892 -0.0369 0.2694

%Exercise 6

type ststate

function q=ststate(P,x0)

[~,n]=size(P);

S=sum(P);

if any(closetozeroroundoff(S-ones(1,n)))

disp('Error: P is not stochastic')

q=[];

return

else

Q = null(P-eye(n),'r');

c=sum(Q);

q=(1/c)\*Q;

end

x1 = P\*x0;

k = 1;

while norm(x1-q)<10^-7

x0 = x1;

x1 = P\*x0;

k = k + 1;

end

x1

k

end

%(a)

P=[.3 .3;.5 .7], x0=[.4;.6]

P =

0.3000 0.3000

0.5000 0.7000

x0 =

0.4000

0.6000

q=ststate(P,x0)

Error: P is not stochastic

q =

[]

%(b)

P=[.5 .3;.5 .7]

P =

0.5000 0.3000

0.5000 0.7000

q=ststate(P,x0)

x1 =

0.3800

0.6200

k =

1

q =

0.3750

0.6250

%(c)

P=[.9 .2;.1 .8], x0=[.12;.88]

P =

0.9000 0.2000

0.1000 0.8000

x0 =

0.1200

0.8800

q=ststate(P,x0)

x1 =

0.2840

0.7160

k =

1

q =

0.6667

0.3333

x0=[.14;.86]

x0 =

0.1400

0.8600

q=ststate(P,x0)

x1 =

0.2980

0.7020

k =

1

q =

0.6667

0.3333

x0=[.86;.14]

x0 =

0.8600

0.1400

q=ststate(P,x0)

x1 =

0.8020

0.1980

k =

1

q =

0.6667

0.3333

%the initial vector x0 does not have an impact on q.

%x0 does not impact k

%(d)

P=[.90 .60 .30; .05 .30 .40;.05 .10 .30], x0=[.6;.3;.1]

P =

0.9000 0.6000 0.3000

0.0500 0.3000 0.4000

0.0500 0.1000 0.3000

x0 =

0.6000

0.3000

0.1000

q=ststate(P,x0)

x1 =

0.7500

0.1600

0.0900

k =

1

q =

0.8257

0.1009

0.0734

%(e)

P=[.90 .60 .30; .06 .30 .40;.05 .10 .30]

P =

0.9000 0.6000 0.3000

0.0600 0.3000 0.4000

0.0500 0.1000 0.3000

q=ststate(P,x0)

Error: P is not stochastic

q =

[]