Table of Contents


```
%%%% MATLAB PROJECT 4, MATH461, LINEAR ALGEBRA FOR SCIENTIST AND ENGINEERS
%%%% Romeo Perlstein, 5/1/2023
%%%% UID: 118030685, section 0123
```

Question 1

a)

```
format rat
syms t
Epsilon1_1 = {1, t, t^2, t^3}
B1_1 = {1, 1+2*t, 2-t+3*(t^2), 4-t+t^3}
C1_1 = {1+3*t+t^3, 2+t, 3*t-t^2+4*t^3, 3*t}

%%%% PB->E is [[b1]E [b2]E ...]
P1_1 = transpose([1 0 0 0; 1 2 0 0; 2 -1 3 0; 4 -1 0 1]) %%%% I tranpose here
 just to make it in the correct format, without having to completely rewrite
 it
```

```
Q1_1 = transpose([1 3 0 1; 2 1 0 0; 0 3 -1 4; 0 3 0 0])
```

b)

```
R = inv(Q1_1)*P1_1
```

c)

```
%%%% c1(1+3t+t^3) + c2(2+t) + c3(3t-t^2+4t^3) + c4(3t) = t^3
%%%% c1 + 3t(c1) + t^3(c1) + 2c2 + t(c2) + 3t(c3) - t^2(c3) + 4t^3(c3) +
 3t(c4) = t^3
%%%% (c1+2c2) + (3tc1 + tc2 +3tc3 + 3tc4) + (-t^2(c3)) + (t^3c1 + 4t^3c3) =
 t^3
temp_trix_1 = [1 2 0 0 0; 3 1 3 3 0; 0 0 -1 0 0; 1 0 4 0 1]
temp_res_1 = rref(temp_trix_1)
C_coord_vect_1 = [temp_res_1(1,5) ; temp_res_1(2,5); temp_res_1(3,5);
 temp_res_1(4,5)]

%%%% sanity check
(C_coord_vect_1(1,1)*(1+3*t
+t^3))+(C_coord_vect_1(2,1)*(2+t))+(C_coord_vect_1(3,1)*(3*t-
t^2+4*t^3))+(C_coord_vect_1(4,1)*3*t)
```

d)

```
pt_B_1 = transpose([0 1 2 3])
pt_C_1 = R*pt_B_1
```

e)

```
%%%% convert p of b coords back to p coords. since we are given p_B,
%%%% multiply each element of p_B with B to get original polynomial
pt_1 = 0*1 + 1*(1 + 2*t) + 2*(2-t+3*t^2) + 3*(4-t+t^3)


Epsilon1_1 =

  1×4 cell array

    {[1]}    {[t]}    {[t^2]}    {[t^3]}


B1_1 =

  1×4 cell array

    {[1]}    {[2*t + 1]}    {[3*t^2 - t + 2]}    {[t^3 - t + 4]}


C1_1 =
```

```
  1×4 cell array

    {[t^3 + 3*t + 1]}    {[t + 2]}    {[4*t^3 - t^2 + 3*t]}    {[3*t]}


P1_1 =

       1             1             2             4
       0             2            -1            -1
       0             0             3             0
       0             0             0             1


Q1_1 =

       1             2             0             0
       3             1             3             3
       0             0            -1             0
       1             0             4             0


R =

       0             0            12             1
       1/2           1/2          -5             3/2
       0             0            -3             0
      -1/6           1/2         -23/3          -11/6


temp_trix_1 =

       1             2             0             0             0
       3             1             3             3             0
       0             0            -1             0             0
       1             0             4             0             1


temp_res_1 =

       1             0             0             0             1
       0             1             0             0            -1/2
       0             0             1             0             0
       0             0             0             1            -5/6


C_coord_vect_1 =

       1
      -1/2
       0
      -5/6


ans =
```

```
t^3


pt_B_1 =

       0
       1
       2
       3


pt_C_1 =

      27
      -5
      -6
     -61/3


pt_1 =

3*t^3 + 6*t^2 - 3*t + 17
```

Question 2

```
format short
A_2 = [0 1 2; 3 4 5; 6 7 8]
```

a)

```
eig(A_2)
```

b)

```
B_2 = [A_2(2,:); A_2(1,:); A_2(3,:)]
C_2 = [7*A_2(1,:); A_2(2,:); A_2(3,:)]
D_2 = [A_2(1,:); A_2(2,:); A_2(3,:) + 3*A_2(1,:)]

%%%% re-adding eig(A) for comparison)
eig(A_2)
eig(B_2)
eig(C_2)
eig(D_2)
```

c)

The Eigenvalue that is shared between each of the matrices A, B, C, D is 0. This is because A, B, C, D are all non-invertible matrices, so they have to have 0 as an eigenvalue. 0 can only be an eigenvalue of a matrix if and only if the matrix is non-invertible.

```
A_2 =

     0     1     2
     3     4     5
     6     7     8


ans =

   13.3485
   -1.3485
   -0.0000


B_2 =

     3     4     5
     0     1     2
     6     7     8


C_2 =

     0     7    14
     3     4     5
     6     7     8


D_2 =

     0     1     2
     3     4     5
     6    10    14


ans =

   13.3485
   -1.3485
   -0.0000


ans =

   12.7082
    0.0000
   -0.7082


ans =

   18.0000
   -6.0000
```

```
   -0.0000


ans =

   18.4868
   -0.4868
   -0.0000
```

Question 3

```
A_3 = [7 6 -2; -8 -7 2; 2 1 -1]
```

a)

```
[P_3,D_3] = eig(A_3)

A_3_check = P_3*D_3*inv(P_3)
A_3_check_compare = int64(real(A_3_check))
```

b)

```
if(A_3 == A_3_check_compare)
    fprintf("A_3 and PDP^-1 are in fact equal! Woohoo!\n")
end
```

c)

Using the results from the previous part, we can extract the Eigenvalues and the Eigenvectors of A. The Eigenvalues are diagonal elements of of D, and the Eigenvectors are are the set of vectors in P, if P was a set (change the matrix to a set and bada bing bada boom the Eigenvectors are each element in the set. knowing this, we have the following

The Eigenvalues of A are i, -i, and -1. to check this, we can also do eig(A)

The Eigenvectors of A are: [-0.6324 ; 0.6325 ; -0.3162 + 0.3162i] for i [-0.6324 ; 0.6325 ; -0.3162 - 0.3162i] for -i [0.3333 ; -0.6667 ; -0.6667 - for -1

now, we'll do a quick check:

```
eig(A_3)


A_3 =

     7     6    -2
    -8    -7     2
     2     1    -1


P_3 =

  -0.6325 + 0.0000i  -0.6325 + 0.0000i   0.3333 + 0.0000i
```

```
   0.6325 + 0.0000i   0.6325 - 0.0000i  -0.6667 + 0.0000i
  -0.3162 + 0.3162i  -0.3162 - 0.3162i  -0.6667 + 0.0000i


D_3 =

  -0.0000 + 1.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i  -0.0000 - 1.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i  -1.0000 + 0.0000i


A_3_check =

   7.0000 - 0.0000i   6.0000 - 0.0000i  -2.0000 - 0.0000i
  -8.0000 + 0.0000i  -7.0000 + 0.0000i   2.0000 + 0.0000i
   2.0000 - 0.0000i   1.0000 - 0.0000i  -1.0000 - 0.0000i


A_3_check_compare =

  3×3 int64 matrix

    7    6   -2
   -8   -7    2
    2    1   -1

A_3 and PDP^-1 are in fact equal! Woohoo!

ans =

  -0.0000 + 1.0000i
  -0.0000 - 1.0000i
  -1.0000 + 0.0000i
```

Question 4

```
A_4 = [3 1 ; 0 3]
```

a)

```
[P_4,D_4] = eig(A_4)
```

b)

```
A_4_check = P_4*D_4*inv(P_4)
A_4_check_compare = int64(real(A_4_check))
if (A_4_check_compare == A_4)
    fprintf(" The are equal! Everything was A-OK\n")
else
    fprintf("Uh oh, they weren't equal, what is MATLAB project 4 trying to
 show me?\n")
```

```
end
```

c)

```
In_4 = [1 0; 0 1]
lambda_4 = 3
temp_trix_4 = A_4-lambda_4*In_4
% only one vector in the eigenspace (weird looking matrix jeez)
basis_for_eig_space_of_A_4 = {[1;0]}
```

d)

no, there is no basis for R2 consisting of the eigenvectors of A. This is because the set of the eigenvectors of A are linearly dependent, meaning the set really only contains one element, and as such cannot be a basis for R2 (dim of set is 1, when it would need to be 2 and linearly independant). I think this does explain why something went wrong in part b. In this case, A_4 is not diagonizable, which is why part B didn't work. And to continue, the theorem states that a matrix is only diagonizable if and only if the dimension of it's eigenspace is equal to n, or the number of dimensions of the original matrix, and if the dimension of the eigenspace for each eigenvalue is equal to it's multiplicity. In this case, 3 is our only eigenvalue, with multiplicity 2, yet the dimension of it's eigenspace is only 1 (that was a lot, I wanted to include everything, my bad)

```
A_4 =

     3     1
     0     3


P_4 =

    1.0000   -1.0000
         0    0.0000


D_4 =

     3     0
     0     3


A_4_check =

     3     0
     0     3


A_4_check_compare =

  2×2 int64 matrix

   3   0
   0   3
```

```
Uh oh, they weren't equal, what is MATLAB project 4 trying to show me?

In_4 =

     1     0
     0     1


lambda_4 =

     3


temp_trix_4 =

     0     1
     0     0


basis_for_eig_space_of_A_4 =

  1×1 cell array

    {2×1 double}
```

5

```
A_5 = [3 -1 -1; 2 1 0; 0 2 5 ; 3 1 3]
```

a)

```
dot(A_5(:,1), A_5(:,2))
dot(A_5(:,3), A_5(:,3))
```

b)

```
transpose(A_5)*A_5
```

c)

the relationship between the dot product of the columns and the values of A^T*A are that, the dot product of column 1 and do appear in place 1-2 and 2-1, respectively in the matrix, and the dot product of column 3 onto itself appears in the spot 3-3. To me, this implies that the dot product of the columns will appear in the place that it was computed from (wording is a little weird here). As in, the dot product of Column 1 onto itself would appear in spot 1-1 in the matrix, and the dot product of column 1 and 3 would appear in the spots 3-1 and 1-3.

d)

the relationship between the dot product of vectors associated with A and A*A^t is that the dot product of the ROWs of A will not appear in a certain place, and that A*A^T is composed of the dot product of the rows of A, as in, the

dot product of row 1 and row 1, the dot product of row 2 and row 4, etc etc. These values of these dot procducts will manifest in the coordinates that the dot product is made of, as in, the dot product of row 1 and row 1 will appear in 1-1, and the dot product of row 2 and row 4 will appear in 2-4 and 4-2

e)

supporting evidence:

```
A_5_second_Tpose = A_5*transpose(A_5)
A_5_check1 = dot(A_5(1,:), A_5(1,:))
A_5_check2 = dot(A_5(2,:), A_5(4,:))
A_5_check3 = dot(A_5(3,:), A_5(2,:))

if (A_5_check1 == A_5_second_Tpose(1,1))
    fprintf("First check is all good!\n")
end
if(A_5_check2 == A_5_second_Tpose(2,4) && A_5_check2 == A_5_second_Tpose(4,2))
    fprintf("Second checked passed as well! We are SO close \n")
end
if(A_5_check3 == A_5_second_Tpose(3,2) && A_5_check3 == A_5_second_Tpose(2,3))
    fprintf("Third check is all good mannn, we are golden!\n")
end
```

f)

```
Q_5 = [1/sqrt(14) 1/sqrt(3) 5/sqrt(42) ; 2/sqrt(14) 1/sqrt(3) -4/sqrt(42) ; 3/
sqrt(14) -1/sqrt(3) 1/sqrt(42)]
transpose(Q_5)*Q_5
```

g)

The computation above shows that Q_5 is an othornormal set because Q_5^T* Q_5 results in a matrix that has the form: [u1^T*u1 u1^T*u2 ... u1^T*un ; u2^T*u1 u2^T*u2 ... u2^T*un ; ...] for each row in the matrix. This checks every combination of the dot product between the different columns of the matrix, since it does the vector times it's transpose (which is the same as the dot product). You'll notice that the diagonal of the matrix is the product of the vector and it's tranpose, which results in 1. so, if the diagonal is all 1s, and the rest are zeros then you have an orthonormal set!

h)

the rows of Q also form a orthonormal set because Q is orthagonal, and as such the rows of an orthagonal matrix are orthonormal

```
A_5 =

     3    -1    -1
     2     1     0
     0     2     5
     3     1     3
```

```
ans =

     2


ans =

    35


ans =

    22     2     6
     2     7    14
     6    14    35


A_5_second_Tpose =

    11     5    -7     5
     5     5     2     7
    -7     2    29    17
     5     7    17    19


A_5_check1 =

    11


A_5_check2 =

     7


A_5_check3 =

     2

First check is all good!
Second checked passed as well! We are SO close
Third check is all good mannn, we are golden!

Q_5 =

    0.2673    0.5774    0.7715
    0.5345    0.5774   -0.6172
    0.8018   -0.5774    0.1543


ans =

    1.0000         0         0
         0    1.0000   -0.0000
```

```
         0   -0.0000    1.0000
```

Published with MATLAB® R2022b