# APPLIED LINEAR ALGEBRA HOMEWORK 1

TOMMENIX YU
STAT 31430
DUE WEDNESDAY, OCT. 12, 8PM

## 1. Written Assignment

**Exercise 1.1.** *Suppose $A \in M_2(\mathbb{C})$ is a unitary 2 × 2 matrix with complex-valued entries. Show that if* $\det(A) = 1$ *then there exist two complex numbers $u, v \in \mathbb{C}$ such that*

$$|u|^2 + |v|^2 = 1$$

*and*

$$A = \begin{bmatrix} u & v \\ -\bar{v} & \bar{u} \end{bmatrix}.$$

*Proof.* Let $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$. Since A is unitary, $A^*A = I$, we get

$$\begin{bmatrix} \bar{a} & \bar{c} \\ \bar{b} & \bar{d} \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} a\bar{a} + c\bar{c} & \bar{a}b + \bar{c}d \\ a\bar{b} + c\bar{d} & b\bar{b} + d\bar{d} \end{bmatrix} = I$$

which implies

$$\begin{cases} |a|^2 + |c|^2 = 1 \\ |b|^2 + |d|^2 = 1 \\ a\bar{b} + c\bar{d} = 0 \end{cases}$$

Now, $\det(A) = 1$, along with the third line above, gives us

$$\begin{cases} ad - bc = 1 \\ a\bar{b} + c\bar{d} = 0 \end{cases} \Rightarrow \begin{cases} ad\bar{b} - bc\bar{b} = \bar{b} \\ ad\bar{b} + c\bar{d}d = 0 \end{cases} \Rightarrow \bar{b} + bc\bar{b} = -c\bar{d}d$$

$$\Rightarrow \bar{b} = -c(|b|^2 + |d|^2) = -c$$

Combining $\bar{b} = -c$ with $a\bar{b} + c\bar{d} = 0$ we get $a = \bar{d}$, which means $A = \begin{bmatrix} a & -\bar{c} \\ c & \bar{a} \end{bmatrix}$.

Letting $a = u, c = -\bar{v}$ and we are done. □

**Exercise 1.2.**

*(a) Fix $k, l \geq 1$. Let $A \in \mathcal{M}_k(\mathbb{R})$ and $B \in \mathcal{M}_l(\mathbb{R})$ be such that both $A$ and $B$ are invertible. Show that for every $C \in \mathcal{M}_{k,l}(\mathbb{R})$, the block matrix*

$$D = \begin{bmatrix} A & C \\ 0 & B \end{bmatrix}$$

*is invertible, with inverse*

$$D^{-1} = \begin{bmatrix} A^{-1} & -A^{-1}CB^{-1} \\ 0 & B^{-1} \end{bmatrix}$$

*(b) Use (a) to show that the inverse of an invertible $n \times n$ upper triangular matrix is upper triangular. (Hint: Argue by induction on n.)*

*Proof.* (a): We know from class that for block matrices of this type (diagonal blocks are square matrices) we can just do matrix multiplication as if the blocks are regular entries. Thus, direct computation tells us

$$\begin{bmatrix} A^{-1} & -A^{-1}CB^{-1} \\ 0 & B^{-1} \end{bmatrix} \begin{bmatrix} A & C \\ 0 & B \end{bmatrix} = \begin{bmatrix} A^{-1} & A^{-1}C - A^{-1}CB^{-1}B \\ 0 & B^{-1}B \end{bmatrix} = \begin{bmatrix} I_k & 0 \\ 0 & I_l \end{bmatrix} = I_{k+l}$$

which means

$$D^{-1} = \begin{bmatrix} A^{-1} & -A^{-1}CB^{-1} \\ 0 & B^{-1} \end{bmatrix}.$$

(b): We use induction on the dimension n of the upper triangular matrix.

When $n = 1$, the inverse of $[a_{11}^{-1}]$ is just $[a_{11}^{-1}]$ (since invertible, $a_{11} \neq 0$). So the inverse is still upper triangular.

Suppose the condition holds for $n = N - 1$, now for $n = N$, by (a) we have that the upper triangular matrix $U_N = \begin{bmatrix} a_{11} & C \\ 0 & B \end{bmatrix}$ where $B$ is an $(N-1)$ by $(N-1)$ upper triangular matrix. Since $\det(U_N) = a^{11}\det(B)$ so $\det(b) \neq 0$ as $U_N$ is invertible (the determinant result is proven in exercise 3, whose proof did not use result in this exercise), which further means that $B$ is invertible. By IH $B_{-1}$ is upper triangular.

Again, by (a) we get that

$$U_N^{-1} = \begin{bmatrix} a_{11}^{-1} & -a_{11}^{-1}CB^{-1} \\ 0 & B^{-1} \end{bmatrix}$$

is upper triangular, and so we are done. □

**Exercise 1.3.**

(a) *Fix $n \geq 1$ and let $A \in \mathcal{M}_n(\mathbb{R})$ be given. Show that for all $m \geq 1$, the $(n+m)\times(n+m)$ matrices given in block form by*

$$\begin{bmatrix} I & 0 \\ 0 & A \end{bmatrix} \text{ and } \begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix}$$

*both have determinants equal to $\det(A)$, where $I$ is the $m \times m$ identity matrix. (Hint: For the first matrix, repeatedly use the familiar recursive definition of determinant via "expansion by minors" along the first row, and the first rows of the resulting submatrices, until you reach $\det(A)$. For the second matrix, a similar argument applies, starting from the last row(s).)*

(b) *Fix $k, l \geq 1$ and let $A \in \mathcal{M}_k(\mathbb{R})$, $B \in \mathcal{M}_{k,l}(\mathbb{R})$, and $D \in \mathcal{M}_l(\mathbb{R})$ be given. Show that*

$$\begin{bmatrix} A & B \\ 0 & D \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & D \end{bmatrix} \begin{bmatrix} I & B \\ 0 & I \end{bmatrix} \begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix}$$

*where each instance of $I$ denotes an identity matrix of suitable size.*

(c) *Use (a) and (b) to show that for $A, B, D$ as above,*

$$\det \begin{bmatrix} A & B \\ 0 & D \end{bmatrix} = \det(A)\det(D)$$

*Proof.*

(a) We use induction on $n$ to prove that $\det \begin{bmatrix} I & 0 \\ 0 & A \end{bmatrix} = \det(A)$ .

For $m = 1$, $\det \begin{bmatrix} 1 & 0 \\ 0 & A \end{bmatrix} = 1 \cdot A = \det(A)$.

Assume this holds for $m = N - 1$, for $m = N$ we have

$$\det \begin{bmatrix} I_N & 0 \\ 0 & A \end{bmatrix} = \det \begin{bmatrix} 1 & 0 \\ 0 & B \end{bmatrix} = 1 \cdot \det(B) = \det(B) \text{ where } B = \begin{bmatrix} I_{N-1} & 0 \\ 0 & A \end{bmatrix}$$

and by IH, $\det(B) = \det(A)$, so we are done.

We do the exact same thing on the other matrix:

For $m = 1$, $\det \begin{bmatrix} A & 0 \\ 0 & 1 \end{bmatrix} = (-1)^{(m+1)+(m+1)} 1 \cdot A = \det(A)$.

Assume this holds for $m = N - 1$, for $m = N$ we have

$$\det \begin{bmatrix} A & 0 \\ 0 & I_N \end{bmatrix} = \det \begin{bmatrix} B & 0 \\ 0 & 1 \end{bmatrix} = (-1)^{(m+1)+(m+1)} 1 \cdot \det(B) = \det(B) \text{ where } B = \begin{bmatrix} A & 0 \\ 0 & I_{N-1} \end{bmatrix}$$

and by IH, $\det(B) = \det(A)$, so we are done.

(b)

$$\begin{bmatrix} I & 0 \\ 0 & D \end{bmatrix} \begin{bmatrix} I & B \\ 0 & I \end{bmatrix} \begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & D \end{bmatrix} \begin{bmatrix} A & B \\ 0 & I \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & D \end{bmatrix}$$

by direct computation. We can do direct computation due to theory covered in class (and because each diagonal block is square).

(c)

$$\det\begin{bmatrix} I & B \\ 0 & I \end{bmatrix} = \sum_{\sigma\in S_{n+m}}\left[\varepsilon(\sigma)\prod_{i=1}^{n+m} a_{i,\sigma(i)}\right] = 1$$

since the only term not containing the left bottom zero part is with $\sigma = id$. So we have by (a) and (b)

$$\begin{aligned}\det\begin{bmatrix} A & B \\ 0 & D \end{bmatrix} &= \det\left(\begin{bmatrix} I & 0 \\ 0 & D \end{bmatrix}\begin{bmatrix} I & B \\ 0 & I \end{bmatrix}\begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix}\right) \\ &= \det\begin{bmatrix} I & 0 \\ 0 & D \end{bmatrix}\det\begin{bmatrix} I & B \\ 0 & I \end{bmatrix}\det\begin{bmatrix} A & 0 \\ 0 & I \end{bmatrix} \\ &= \det(A)\cdot\det(D) = \det(A)\det(D).\end{aligned}$$

□

**Exercise 1.4.** *Read the excerpt from S. Weintraub, A Guide to Advanced Linear Algebra, Section 2.3 "Change of basis" posted on Canvas (under Files Other Resources).*

*Let* $\boldsymbol{B} = (1,2),(3,4) \subset \mathbb{R}^2$ *and* $\boldsymbol{C} = (7,3),(4,2)$ *be two sets of basis vectors for* $\boldsymbol{R}^2$*. Set* $v = (1,0) \in \mathbb{R}^2$*. Find* $[v]_B$ *and* $[v]_C$ *. Find the change of basis matrix* $P_{C\leftarrow B}$ *and verify that the identity* $[v]_C = P_{C\leftarrow B}$ *holds in this case.*

**Solution:**Let $[v]_B = (x,y)^T$, then we have $\begin{cases} x+3y=1 \\ 2x+4y=0 \end{cases} \Rightarrow [v]_B = \begin{bmatrix} -2 \\ 1 \end{bmatrix}$.

Let $[v]_C = (x,y)^T$, then we have $\begin{cases} 7x+4y=1 \\ 3x+2y=0 \end{cases} \Rightarrow [v]_C = \begin{bmatrix} 1 \\ -2/3 \end{bmatrix}$.

$$P_{C\leftarrow B} = \begin{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix}_C & \begin{bmatrix} 3 \\ 4 \end{bmatrix}_C \end{bmatrix} = \begin{bmatrix} -3 & -5 \\ 11/2 & 19/2 \end{bmatrix}.$$

And indeed,

$$P_{C\leftarrow B}[v]_B = \begin{bmatrix} 1 \\ -2/3 \end{bmatrix} = [v]_C$$

Below is question 5, I write one problem per exercise.

**Exercise 1.5.** *(2.2) The goal of this exercise is to define Matlab functions returning ma- trices having special properties that we shall exploit in the upcoming exercises. All variables will be initialized by rand. Note that there are several possible answers, as it is often the case for computer programs.*

(1) *Write a function (called SymmetricMat(n)) returning a real symmetric matrix of size $n \times n$.*
(2) *Write a function (called NonsingularMat(n)) returning a real nonsingular matrix of size $n \times n$.*
(3) *Write a function (called LowNonsingularMat(n)) returning a real nonsingular lower triangular matrix of size $n \times n$.*
(4) *Write a function (called UpNonsingularMat(n)) returning a real nonsingular upper triangular matrix of size $n \times n$.*
(5) *Write a function (called ChanceMat$(m, n, p)$) returning a real matrix of size $m \times n$ whose entries are chosen randomly between the values $-p$ and $p$.*
(6) *Write a function (called BinChanceMat(m,n)) returning a real matrix of size $m \times n$ whose entries are chosen randomly equal to 0 or 1.*
(7) *Write a function (called HilbertMat(m,n)) returning the so-called Hilbert matrix $H \in \mathcal{M}_{m,n}(\mathbb{R})$ defined by its entries:*

$$H_{i,j} = \frac{1}{i + j - 1}.$$

**Solution:**

(1) Write a function (called SymmetricMat(n)) returning a real symmetric matrix of size n $\times$ n.

```
function M = SymmetricMat(n)
%This function returns a real symmetric random matrix with size nxn
M = rand([n,n]);
M = M + M.';
end
```

(2) Write a function (called NonsingularMat(n)) returning a real nonsingular matrix of size n $\times$ n.

```
function M = NonsingularMat(n)
%       This function returns a real nonsingular matrix of size nxn.
    while true
        M = rand([n,n]);
        if rank(M) == n
            break
        end
    end
end
```

(3) Write a function (called LowNonsingularMat(n)) returning a real nonsingular lower triangular matrix of size n $\times$ n.

```
function M = LowNonsingularMat(n)
%   This function returns a real nonsingular lower trig matrix of nxn
    while true
        M = tril(rand([n,n]));
        if rank(M) == n
            break
        end
    end
end
```

(4) Write a function (called UpNonsingularMat(n)) returning a real nonsingular upper triangular matrix of size n $\times$ n.

```
function M = UpNonsingularMat(n)
%   This function returns a real nonsingular upper trig matrix of nxn
    while true
        M = triu(rand([n,n]));
        if rank(M) == n
            break
        end
    end
end
```

(5) Write a function (called ChanceMat($m, n, p$)) returning a real matrix of size m $\times$ n whose entries are chosen randomly between the values $-p$ and $p$.

```
function M = ChanceMat(m,n,p)
%This function returns a mxn real matrix with entrices randomly from -p to
%p
M = -p+ 2*p.*rand([m,n]);
end
```

(6) Write a function (called BinChanceMat(m,n)) returning a real matrix of size m × n whose entries are chosen randomly equal to 0 or 1.

```
function M = BinChanceMat(m,n)
%This function returns a real symmetric matrix with size mxn whose entries
%are random in {0,1}.
M = randi([0,1],[m,n]);
end
```

(7) Write a function (called HilbertMat(m,n)) returning the so-called Hilbert matrix $H \in \mathcal{M}_{m,n}(\mathbb{R})$ defined by its entries:

$$H_{i,j} = \frac{1}{i+j-1}.$$

```
function H = HilbertMat(m,n)
%This function returns a Hilbert matrix of size mxn
    for i = 1:m
        for j = 1:n
            H(i,j) = 1/(m+n-1);
        end
    end
end
```

**Exercise 1.6.** *(2.5) Vary n from 1 to 10 and*

*(1) determine the rank of a matrix A defined by A=rand(8,n)*rand(n,6). What is going on?*
*(2) Same question for A=BinChanceMat(8,n)*BinChanceMat(n,6).*
*(3) Justify your observations.*

**Solution:**

```
clear all;
for n = 1:10
    A = rand([8,n])*rand([n,6]);
    list(1,n) = rank(A);
end


for n = 1:10
    A = BinChanceMat(8,n)*BinChanceMat(n,6)
    list(2,n) = rank(A);
end
list
```

The output of the above code (list) is this:

list =

1 2 3 4 5 6 6 6 6 6

1 2 3 3 5 5 6 6 6 6

The first row solves the first question, the second solves the second.

As for the explanations it is this:

(3): For the first one, since there's very little chance that the random decimal numbers can have linearly independent columns, it is the case that the rank of both matrices is determined by the smaller entry, where as the rank of the multiplication depends on the matrix with smaller rank. That is, the rank of both matrix is n when n smaller than 6, since the second matrix has rank 6 when n larger, the multiplication cannot yield a matrix with rank larger.

For the second one, the only difference is that the BinChanceMat is more likely to give us linearly independent columns (as we can see, when n=4 and n=6), so the rank might be samller than the first. Otherwise the explanation is the same.

**Exercise 1.7.** *(2.6) We fix the dimension $n = 5$*

(1) *For any integer $r$ between 1 and 5, initialize (with rand) $r$ vectors $u_i$ and define a square matrix $A = \sum_{i=1}^{r} u_i u_i^T$. Compare the rank of $A$ with $r$.*
(2) *Same question for vectors generated by BinChanceMat.*
(3) *Justify your observations.*

**Solution:**

```
clear all;
for r = 1:5
    list(1,r) = r;
    A = zeros(r);
    B = zeros(r);
    for i = 1:r
        u = rand([r,1]);
        v = BinChanceMat(r,1);
        A = A + u*u.';
        B = B + v*v.';
    end
    list(2,r) = rank(A);
    list(3,r) = rank(B);
end
list
```

with the result

```
>> ex2_6

list =

     1     2     3     4     5
     1     2     3     4     5
     1     1     2     4     4

fx >>
```

The first row is just a counting of r.

The second row is the first question, the third row the second question.

The result of the first question is that the $\sum_{i=1}^{r} u_i u_i^t$ should have rank(r) as long as the $u_i$ are linearly independent. This is because that $u_i u_i^t \cdot v = 0$ iff the vector $v$ is orthogonal to $u_i$, yet this is saying that $Av = 0$ means v is independent to $u_i u_i^t$ for all $i$. Such $v$ exists (non-zero) when the span of the $u_i$ is not the full space, i.e., $u_i$ is not linearly independent.

Since for random vectors, it is really rare that they are linearly independent, where as for random vectors with value only in $\{0, 1\}$, it's relatively possible for this to happen, as we can tell in the third line, it happens when $r = 2, 3, 5$.

**Exercise 1.8.** *(2.7) Write a function (called MatRank(m,n,r)) returning a real matrix of size* $m \times n$ *and of fixed rank* $r$*.*

**Solution:**

```
function M = MatRank(m,n,r)
%   This function returns a real matrix of size mxn and of fixed rank r.
    I = zeros(m,n);
    for i = 1:r
        I(i,i)=1;
    end
    Right = NonsingularMat(n);
    Left = NonsingularMat(m);
    M = Left*I*Right;
end
```

**Exercise 1.9.** *(2.9) Let A be a matrix defined by A=MatRank(n,n,r) with r $\leq$ n and Q a matrix defined by Q=null(A'), that is, a matrix whose columns form a basis of the null space of $A^T$. Let u be a column of Q, compute the rank of $A + uu^t$. Prove the observed result.*

**Solution:**

```
clear all;
n=10;
r=4;
A = MatRank(n,n,r);
Q = null(A');
for i = 1: size(Q,2)
    u = Q(:,i);
    %rank(A)
    %A+u*u.'
    listrank(1,i) = rank(A+u*u.');
end
listrank
```

with output

```
>> ex2_9

listrank =

     5     5     5     5     5     5

>>
```

This is easy to explain. Since $u$ is in the null space of $A$, $A + uu^t$ should have one more rank since the direction $u$ is not a dimension in the kernel now.

In my case, I chose $\dim(A) = 10$, $\operatorname{rank}(A) = 4$, and thus by our theory above, the rank of $A + uu^t$ should be $4 + 1 = 5$. Indeed it is.

**Exercise 1.10.** *(2.16) For various values of n, compare the spectra of the matrices A and $A^t$ with A=rand(n,n). Justify the answer.*

**Solution:**

```
clear all;
for n = 1:10
    A = rand(n,n)
    eig(A)-eig(A.')
end
```

My output to this question is the difference between two columns of eigenvalues. In theory, transpose does not change the eigenvalues so the output should be a list of 0. This is because for the kernel really is the same, thus the direct sum of all eigenspaces.

Well in practice it is not.

The result is pretty good when n is small, as in ($n \leq 4$):

```
ans =

     0


A =

    0.3164    0.4987
    0.9591    0.7386


ans =

     0
     0


A =

    0.0128    0.8074    0.9024
    0.6054    0.6550    0.1522
    0.5765    0.8782    0.1926


ans =

   1.0e-15 *

    0.4441
    0.2220
    0.0278
```

But for large n......results starts to behave wildly (n=10):

```
A =

    0.4479    0.6595    0.8418    0.3011    0.4690    0.1951    0.6800    0.5091    0.7277    0.5476
    0.6512    0.2948    0.1309    0.4956    0.0873    0.7054    0.5150    0.2468    0.6510    0.3951
    0.1695    0.9504    0.1892    0.2582    0.8287    0.1805    0.5221    0.0454    0.6646    0.3983
    0.5314    0.6943    0.1536    0.7329    0.6859    0.5223    0.1029    0.8417    0.9388    0.7513
    0.6338    0.2068    0.0289    0.1168    0.2673    0.2962    0.9969    0.0482    0.5351    0.5224
    0.0141    0.5548    0.0091    0.7460    0.9695    0.4628    0.3590    0.3163    0.3984    0.4904
    0.4704    0.8793    0.5965    0.8098    0.1838    0.9252    0.6252    0.7834    0.6705    0.0887
    0.8863    0.5579    0.6090    0.7452    0.2999    0.2159    0.3934    0.9724    0.4405    0.2509
    0.1140    0.7523    0.9189    0.3371    0.4112    0.0010    0.0077    0.5865    0.1329    0.4476
    0.4425    0.8949    0.7336    0.5843    0.2365    0.9066    0.5453    0.7780    0.4392    0.6380
```

```
ans =

   0.0000 + 0.0000i
  -1.0766 + 0.1711i
  -1.0766 - 0.1711i
   1.0766 - 0.1711i
   1.0766 + 0.1711i
  -0.3202 - 0.4500i
   0.0000 + 0.9000i
  -0.0311 - 0.6766i
   0.0000 + 0.4533i
   0.3513 - 0.2266i
```

Possible reasons are:

1. the function I use to return eigenvalue, eig(A), does not return the same eigenvalue in the same order. As we can see, there’s some sort of repetition in the column, so maybe it is just because of that.

2. Maybe computing the eigenvalue of a large matrix is not accurate, but I don’t think this is the case since 10 by 10 is not large engough.

**Exercise 1.11.** *(2.17)Fix the dimension n. For u and v two vectors of* $\boldsymbol{R}^n$ *chosen randomly by rand, determine the spectrum of* $I_n + uv^t$. *What are your experimental observations? Rigorously prove the observed result.*

**Solution:**

```
clear all;
n=5;
for count = 1:10
    u=rand(n,1);
    v=rand(n,1);
    eig(eye(n)+u*v.')
end
```

And the solution is:

```
ans =

    1.0000
    2.0746
    1.0000
    1.0000
    1.0000

>>
```

Now this makes sense because the rank of $uv^t$ is 1 and by adding it we add (as long as we don't randomly get standard basis) 1 more rank to the identity matrix. This is reflected in one more different eigenvalue, as it gives a new eigenspace with dimension 1.

As for why it's the second eigenvalue... It is related to how matlab labels the eigenvalues. What I can guess is to repeat one eigenvalue to the number of the dimension of it is related, do it for all eigenvalues, then dump additional multiplicities below.