PS 1

1.

$$\begin{bmatrix} a_{1,1} & \cdots & a_{1,p} \\ \vdots & & \\ a_{m,1} & & \end{bmatrix} \begin{bmatrix} b_{1,1} & \cdots & b_{1,n} \\ \vdots & & \\ b_{p,1} & & \end{bmatrix} = \begin{bmatrix} c_{1,1} & \cdots & c_{1,n} \\ \vdots & & \\ c_{m,1} & & \end{bmatrix}$$

We want to show that calculating $C=AB$ by taking the dot products of i^{th} row of A and j^{th} column of B is equivalent to summing up p $m \times n$ matrices. This is the same as showing that ~~[illegible]~~ the entries c_{ij} of C are made up of the same partial sums in both approaches.

~~Let's take an individual $a_k b^k$ and call it E_k~~

~~[illegible]~~

Let's call summation $AB = \sum_{k=1}^{p} a_k b^k = M$, and ~~[illegible]~~ an arbitrary $m \times n$ matrix in the series m_k, with $k \in [1, p]$. The k^{th} column of A and k^{th} row of B are matrices and so to find the ij^{th} entry of their product we can use the definition

$$c_{ij} = a^i \cdot b_j = \sum_{k=1}^{p} a_{ik} b_{kj}.$$

Then, we have $c_{ij} = a_k{}^i \cdot b_j{}^k = \underline{a_{ik} \cdot b_{kj}}$. a_{ik} and b_{kj} are ~~singulare~~ singular entries in A + B respectively and so the dot product is simply the product of the two entries.

~~[illegible]~~ Now we have an expression for the ij^{th} entry of an individual m_k. To get the ij^{th} entry of M, M_{ij} we sum up the ij^{th} components, getting

$$\sum_{k=1}^{p} a_{ik} \cdot b_{kj}$$

This is equivalent to our expression for the $(ij)^{th}$ entry using dot product. $\square$

1. $$\begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ & 0 & \cdots & a_{2n} \\ \vdots & \vdots & 0 & \vdots \\ & & & 0 \\ 0 & 0 & \cdots & 0 \end{bmatrix}^2$$

2. We start with upper triangular matrix A. We know that $a_{ij} = 0 \ \forall i, j$ s.t. $i \geq j$. Given that the (i, j) entry of A^k is $(a^k)_{ij}$, we will assert the following statement and prove it with induction.

$P(k): \forall j \leq i + (k-1), \ (a^k)_{ij} = 0.$

Base case

$k = 1 \Rightarrow \forall j \leq i, \ (a^k)_{ij} = 0$

This is obviously true because it is the definition

~~k+j>i+j~~ $j \le k+(n-1)$
$i+n \le k+$

of a strictly upper triangular matrix.

Inductive Step

Assume true for $K=n$ $P(n)$

$\forall j \le i+(n-1),\ (a^n)_{ij} = 0$

Show true for $k=n+1$

$P(n+1): \forall j \le i+n,\ (a^{n+1})_{ij} = 0$

$(a^{n+1})_{ij} = (a \cdot a^n)_{ij} = a^i \cdot (a^n)_j = \sum_{k=1}^{p} a_{ik}\ (a^n)_{kj}$

Let's apply condition $j \le i+n$ and then derive $(a^{n+1})_{ij}$.

There are two cases for comparison b/t i & k

$k \le i \Rightarrow$ Our base case $P(1)$ tells us $a_{ik} = 0$

$k > i \Rightarrow i+n < k+n \Rightarrow j \le i+n < k+n$
$j \le k+n$
$j \le k+(n-1)$
$\therefore (a^n)_{kj} = 0$

$\forall k \le p$ we have found that at least one of a_{ik} or $(a^n)_{kj} = 0$ and thus the series summation is 0.

So ~~for~~ $\forall j \le i+n,\ (a^{n+1})_{ij} = 0.$

$\square$

3. $$P_n = \overbrace{\begin{pmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{pmatrix}}^{n} \qquad L_n = \begin{pmatrix} 1 & & & & 0 \\ -\frac{1}{2} & \ddots & & & \\ & -\frac{2}{3} & \ddots & & \\ & & \ddots & \ddots & \\ 0 & & & -\frac{n+1}{n} & 1 \end{pmatrix}$$

$-\frac{n+1}{n}$

$$U_n = \begin{pmatrix} \frac{2}{1} & -1 & & & 0 \\ & \frac{3}{2} & \ddots & & \\ & & \frac{4}{3} & \ddots & \\ & & & \ddots & -1 \\ 0 & & & & \frac{n+1}{n} \end{pmatrix}$$

Since P_n is identity matrix, it suffices to show that $A_n = L_n U_n$.

Let us break down the possible values of L_n and U_n by cases so that our $(A_n)_{ij}$ entries are easier to formulate.

$L_n \Rightarrow L_{i,i} = 1 \quad L_{i,i-1} = -\frac{i+1}{i} \quad L_{else} = 0$

$U_n \Rightarrow U_{i,i} = \frac{i+1}{i} \quad U_{i,i+1} = -1 \quad U_{else} = 0$

- Now we check each of the tridiagonals in A_n to ensure we get the right values.

- $(a_n)_{ii} = (L_n U_n)_{ii} = \sum_{k=1}^{n} L_{ik} U_{ki}$

This can be expressed as $L_{ii} \cdot U_{ii} + L_{i,i-1} \cdot U_{i-1,i}$

since every other term has a 0 in it.

Then, $L_{ii} \cdot U_{ii} + L_{i,i-1} \cdot U_{i-1,i}$

$$= \frac{i+1}{i} + \frac{-(-i+1)}{i}$$

$$= \frac{2i}{i} = \boxed{2}$$

So, we have shown all elements of main diagonal $= 2$.

- $(a_n)_{i,i+1} = \sum_{k=1}^{n} L_{ik} U_{k,i+1}$

$$= L_{ii} \cdot U_{i,i+1} + L_{i,i-1} \cdot \underbrace{\cancel{U_{i-1,i+1}}}_{0} \quad U_{else} = 0$$

$$= 1 \cdot -1 = \boxed{-1}$$

So, all elements of super diagonal are -1.

be $\cancel{(a_n)_{i+1,2} = \sum_{k=1}^{n} L_{ik} U_{k,i+2}}$

- $(a_n)_{i+1,i} = \sum_{k=1}^{n} L_{i+1,k} U_{k,i}$

$$= L_{i+1,i+1} \underbrace{\cancel{U_{i+1,i}}}_{0} + L_{i+1,i} U_{i,i}$$

$$= \frac{-i}{i+1} \cdot \frac{i+1}{i} = \boxed{-1}$$

So, all elements of sub diagonal are -1.

Therefore, we have shown for each of the tridiagonals of $L_n U_n$, $(A_n)_{ij} = (L_n U_n)_{ij}$. The other entries not on the tridiagonal will always equal 0 because their dot products always have a factor of 0 from L_n or U_n. $\square$

4. a) If P is orthogonal, $P^{-1} = P^T$

$$PP^{-1} = PP^T$$
$$I = PP^T$$

So, we will prove $PP^T = I$.

We know by definition a permutation matrix has 1 one per row and per column. Because of the definition of a transpose matrix then, the 1 in the i^{th} row of P will correspond to the location of the 1 in the i^{th} column of P^T ($P_{in} = P^T_{ni} = 1$). Because every other entry in the dot product of the rows is 0, we get that when $i=j$, $(PP^T)_{ij} = 1$. By similar logic, we know that when $i \neq j$, the 1 entries will not correspond because there is only one unique matching per row & column.

We have proved
$$(PP^T)_{ij} = 1, \quad i=j$$
$$(PP^T)_{ij} = 0, \quad i \neq j$$
which is definition of Identity matrix.

$\square$

b) We can provide a counterexample.

$$A = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix} \quad A^{-1} = \frac{1}{-1}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix} = A^T$$

Thus, A is orthogonal but because our nonzero entries are not 1's, A is not a permutation matrix.

5. We will approach this with proof by construction, finding an expression for a symmetric matrix ~~A~~ S using A and an expression for skew-symmetric matrix J using A.

Given we have a square matrix A, we construct the following:

$S = A^T + A \Rightarrow S^T = (A^T + A)^T = A^T + A = S$ * S is symmetric

$J = -A^T + A \Rightarrow J^T = (-A^T + A)^T = A^T - A = -J$ * J is skew-symmetric

Symmetric and skew-symmetric matrices are closed under scalar multiplication so $\frac{1}{2}S$ and $\frac{1}{2}J$ retain properties.

$\frac{1}{2}S + \frac{1}{2}J = \frac{A^T + A}{2} + \frac{A - A^T}{2} = \frac{2A}{2} = A$ ✓

So, we have shown given a square matrix A, we can write it as sum of symmetric matrix S + skew-symmetric matrix J.

□

6. In Matlab file

7. a) Using Matlab, we guess that for $n \geq 2$, rank$(B) = 2$.

Pf

$$A = \begin{pmatrix} 1 & 2 & \cdots & n \\ n+1 & n+2 & & 2n \\ \vdots & \vdots & & \vdots \\ 1+n(n-1) & 2+n(n-1) & \cdots & n+n(n-1) \end{pmatrix}$$

* Perform row ops to create pivot col.

$R_2 \sim R_2 - (n+1)R_1$

$R_3 \sim R_3 - (2n+1)R_1$

$\vdots$

$R_n \sim R_n - (1+n(n-1))R_1$

Our matrix becomes

$$A = \begin{pmatrix} 1 & 2 & 3 & \cdots & n \\ 0 & -n & -2n & & -n(n-1) \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & -n(n-1) & -2n(n-1) & \cdots & -n(n-1)^2 \end{pmatrix}$$

Now we can see that $\forall i \in [2, n]$, A^i (i^{th} row of A) are all scalar multiples of one another.

$$R_3 = 2R_2$$
$$R_4 = 3R_2$$
$$\vdots$$
$$R_n = (n-1)R_2$$

Thus, when we row reduce the superfluous ~~multiples to~~ rows become 0 rows and only the first two rows are left. Therefore, rank(B) = 2.

b) In Matlab file

any vector in ker(A)

"linear combination of vectors in null space" = ker(A)

* find sol'n ~~x=~~ $x = x_0 + V\alpha$

$(100 \times 98) \times (98 \times 1) = 100 \times 1$

$V\alpha = -x_0$

ACM/IDS 104 - Problem Set 1 - MATLAB Problems

Before writing your MATLAB code, it is always good practice to get rid of any leftover variables and figures from previous scripts.

```
clc; clear; close all;
```

NOTE: As this is the first problem set (and many of you might be unfamiliar with MATLAB) we will provide some helper code. As the term progresses (and you become more experienced) we will omit this.

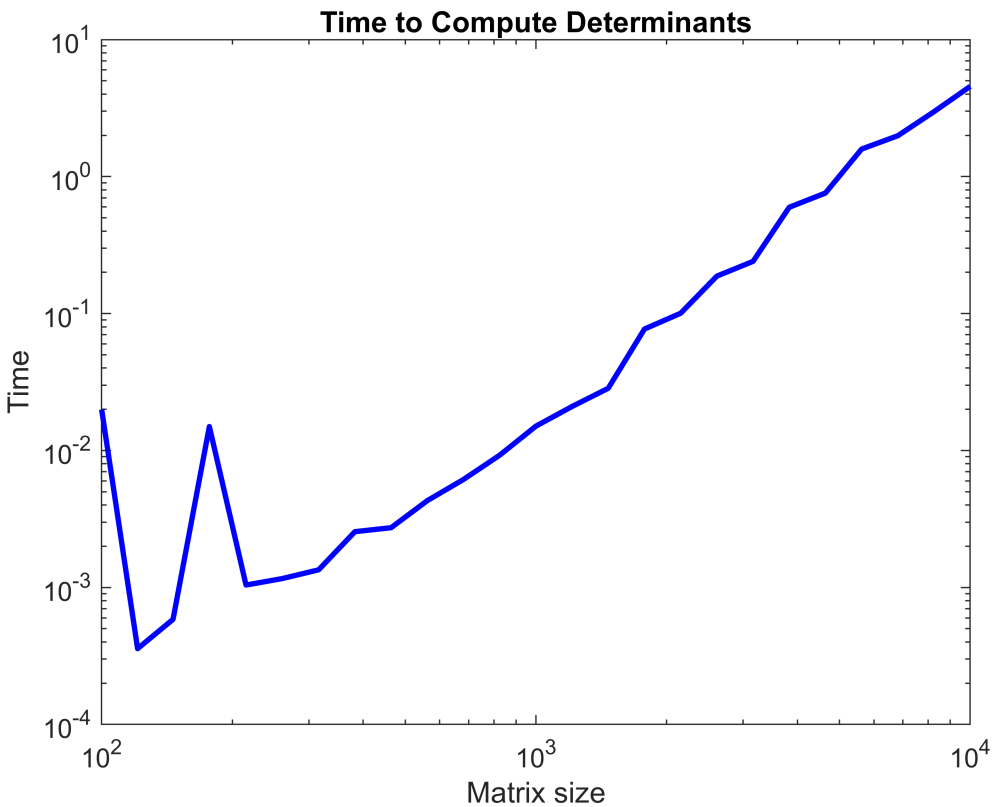
Problem 6 (10 points) Determinants are Expensive to Compute

In lecture 2, we discussed that computing determinants is a computationally expensive task. Let's see how long it takes MATLAB to compute determinants of large matrices (MATLAB uses LU factorization, not the Leibniz formula of course). To do this, we will:

1. Generate 25 values of n logarithmically spaced between 10^2 and 10^4.
2. For each n, generate a matrix A of size $n \times n$, with entries a_{ij} being sampled from the standard normal distribution. Use `randn()`.
3. Compute $\det(A)$ and measure the time MATLAB took to complete the computatation. Use `det()`, `tic` and `toc`.
4. Plot the times versus the values of n in the log-log scale. Label the axes and give a meaningful plot title.

```
% Setup is completed for you; make sure you understand what is happening
vals = 25;
n_vals = floor(logspace(2, 4, vals));
times = NaN(vals, 1);

% TODO
for i = 1 : vals

    % Define n as the i-th element of the n_vals array
    n = n_vals(i);

    % Create the matrix A as described in step 2
    A = randn(n, n);

    tic; % This tells MATLAB to start the timer
    % Compute the determinant of A
    det(A);

    times(i) = toc; % This tells MATLAB to stop the timer and store the time
end


%{
PLOTTING
Here is an elementary example of how to plot in MATLAB. Feel free
```

```
to explore and customize your plots to make them more attractive!
%}

figure; % Always tell MATLAB you are starting a new figure
loglog(n_vals, times, "b", "LineWidth", 2); % log-log scale plot
xlabel("Matrix size");
ylabel("Time");
title("Time to Compute Determinants");
```



Problem 7 (10 points) Solving Linear Systems

We have the matrix:

$$B = \begin{pmatrix} 1 & 2 & \cdots & n \\ n+1 & n+2 & \cdots & 2n \\ \vdots & \vdots & & \vdots \\ n^2-n+1 & n^2-n+2 & \cdots & n^2 \end{pmatrix}$$

Part (a) (5 points)

In this part, your task is to find $\mathrm{rank}(B)$. As mentioned in the problem set, MATLAB is not needed to obtain the answer. However, we can use MATLAB to make a right guess and check our answer. To do this, we first need to construct matrix B in MATLAB:

NOTE: Although you can check your answer here, you still need to justify and show your reasoning to obtain full credit :)

```
n = 100; % set n as specified in Part (b)
B = 1 : n;
for i = 2 : n
     B(i,:) = B(i-1,:) + n;

end
r = rank(B) % check your answer here
```

```
r = 2
```

Part (b) (5 points)

Set $n = 100$ and consider the system of linear equations $Bx = c$ where $c = (1 \quad 2 \quad \cdots \quad n)^T$. Find a solution x such that its first $[n - \mathrm{rank}(B)]$ components are zero. What are the non-zero components of x?

HINT: The backslash operator $B \backslash c$ issues a warning if B is nearly singular and raises an error condition if it detects exact singularity. In that case, use `pinv(B)*c` for finding a particular solution of $Bx = c$. The function `pinv(B)` returns the "pseudoinverse" of B (will discuss the Moore-Penrose pseudoinverse in lecture 16). Also, the following built-in function may be useful: `null`.

```
%{
Let us start by defining the column vector c as specified above.
Remember that in MATLAB we can use ' to transpose a vector.
%}
c = (1:n)';


%{
Now, we obtain a particular solution, x_0, as described above
%}
x_0 = pinv(B)*c;

%{
Use null(B) to define the matrix V, whose columns form an orthonormal
basis in the vector space of all solution of the homogeneous
system Bx=0.
%}
V = null(B);

%{
Use the rank, r, found in part (a) to define k = n - rank(B).
This is the number of free variables / dimension of the vector
space.
%}
k = n - r;
```

This is a good point to review what we have done so far. Recall that the desired solution of the system is:

$$x = x_0 + V\alpha \quad (\star)$$

where α is a $k \times 1$ vector. We can obtain α by solving the system:

$$x_0 + V\alpha = 0 \quad (\star\,\star)$$

```
%{
Find alpha by solving the described system (**).
-> Hint1: Remeber that alpha is a k*1 vector. Hence, you need to
restrict the sizes of x_0 and V
----------------------------------------We need to truncate the last two entries of
x_0 and last two rows of V?????????????????????????????
-> Hint2: Use backslash
%}
%x_0_restricted = x_0(1:k);
%V_restricted = V(1:k,:)
%neg_x_0_restricted = -1 * x_0_restricted
%alpha = V_restricted\neg_x_0_restricted

alpha = V(1:k,:)\-x_0(1:k);
%{
Finally, put everything together and find x using (*)
Use disp(x) to display your solution.
%}
x = x_0 + V * alpha;
disp(x)
```

```
   -0.0000
    0.0000
    0.0000
    0.0000
    0.0000
   -0.0000
   -0.0000
    0.0000
    0.0000
    0.0000
    0.0000
    0.0000
   -0.0000
    0.0000
   -0.0000
    0.0000
    0.0000
    0.0000
   -0.0000
   -0.0000
   -0.0000
    0.0000
   -0.0000
   -0.0000
   -0.0000
   -0.0000
    0.0000
    0.0000
   -0.0000
   -0.0000
```

```
-0.0000
 0.0000
 0.0000
 0.0000
 0.0000
 0.0000
-0.0000
-0.0000
 0.0000
-0.0000
 0.0000
 0.0000
 0.0000
-0.0000
 0.0000
-0.0000
-0.0000
-0.0000
-0.0000
 0.0000
 0.0000
-0.0000
 0.0000
      0
 0.0000
-0.0000
 0.0000
-0.0000
-0.0000
-0.0000
 0.0000
      0
-0.0000
 0.0000
 0.0000
 0.0000
 0.0000
-0.0000
 0.0000
 0.0000
-0.0000
-0.0000
-0.0000
 0.0000
-0.0000
-0.0000
-0.0000
 0.0000
-0.0000
 0.0000
 0.0000
 0.0000
 0.0000
 0.0000
-0.0000
 0.0000
-0.0000
 0.0000
 0.0000
-0.0000
 0.0000
 0.0000
-0.0000
-0.0000
```

```
    0.0000
    0.0000            The second to last entry of x is
   -0.0000            2.49 * 10^-14 which is 10^4
   -0.0000            times greater than the other
    0.0000            entries, so we can consider it
    0.0100            non-zero relatively.
```

```
%{
Now, let us see how x compares to the actual solution.
Un-comment the following 2 lines of code once you reach this part.
%}
error = norm(B*x - c);
disp(error);
```

```
  4.1439e-12
```

Don't forget to report the non-zero components of x!