ACM 104 **October 11, 2023**

Problem Set 1

Problem 1. (10 Points) Matrix Multiplication

Let $A \in \mathbb{M}_{m\times p}$ and $B \in \mathbb{M}_{p\times n}$ be $m \times p$ and $p \times n$ matrices respectively and let $C = AB$. Let $c_1, c_2, ...c_n$ denote the n columns of C. Note that $c_j = Ab_j$ where b_j is the jth column of B. So we have that $c_j = \sum_{k=1}^{p} a_k b_{kj}$ where a_k is the kth column in matrix A and b_{kj} is the entry of matrix B in the kth row and jth column. Hence we have that:

$$C = \begin{bmatrix} \sum_{k=1}^{p} a_k b_{k1} & \sum_{k=1}^{p} a_k b_{k2} & ... & \sum_{k=1}^{p} a_k b_{kn} \end{bmatrix} = \sum_{k=1}^{p} \begin{bmatrix} a_k b_{k1} & a_k b_{k2} & ... & a_k b_{kn} \end{bmatrix} = \sum_{k=1}^{p} a_k b^k$$

Therefore $AB = \sum_{k=1}^{p} a_k b^k$.

Problem 2. (10 points) Nilpotent Matrices

Let $A \in \mathbb{M}_{n\times n}$ be a strictly upper triangular matrix. Let A_{ij} denoted the entry in the ith row and jth column of matrix A.

Claim: $A^k_{ij} = 0$ for $i > j - k$
Proof by induction: For the base case we have $k = 1$ so $A_{ij} = 0$ for $i > j - 1$. Note that this is true by definition of a strictly upper triangular matrix. For the inductive hypothesis, assume that $A^{k-1}_{ij} = 0$ for $i > j - (k - 1)$. For the inductive step we consider the value of A^k_{ij} for $i > j - k$. So we have the following:

$$A^k_{ij} = (AA^{k-1})_{ij} = \sum_{u=1}^{n} A_{iu} A^{k-1}_{uj} = \sum_{u=1}^{i} A_{iu} A^{k-1}_{uj} + \sum_{w=i+1}^{n} A_{iw} A^{k-1}_{wj}$$

Because A is a strictly upper triangular matrix then $A_{iu} = 0$ for $i \geq u$ therefore we have:

$$A^k_{ij} = (AA^{k-1})_{ij} = \sum_{u=1}^{i} 0 \cdot A^{k-1}_{uj} + \sum_{w=i+1}^{n} A_{iw} A^{k-1}_{wj} = \sum_{w=i+1}^{n} A_{iw} A^{k-1}_{wj}$$

Now from the inductive hypothesis we have that $A^{k-1}_{wj} = 0$ for $w > j - (k - 1)$. Therefore for $i > j - k$ we have that $i + 1 > j - k + 1 \Rightarrow w > j - (k - 1)$ for the summation shown above. Hence, for $i > j - k$ we have that:

$$A^k_{ij} = (AA^{k-1})_{ij} = \sum_{w=i+1}^{n} A_{iw} \cdot 0 = 0$$

Thus the claim $A^k_{ij} = 0$ for $i > j - k$ is true.

Now if $A^k_{ij} = 0$ for $i > j - k$ then $A^k = 0$ for $k \geq n$ because $i > j - k$ would be true for any $i, j \in [1, n]$ when $k \geq n$. Thus A is a nilpotent matrix.

Problem 3. (10 points) Permuted LU Factorization

Let A_n be the $n \times n$ tridiagonal matrix with all 2's along the main diagonal, and all -1's along the sub and super diagonals. Let P_n, L_n, U_n be the factors in the permuted LU factorization of A_n.

Claim: $P_n = I_n$ (the identity matrix). Let L_{ij} be the entry in the ith row and jth column in matrix L. L_n is the $n \times n$ matrix with entries $L_{ij} = 1$ for $i = j$, $L_{ij} = -\frac{(i-1)}{i}$ for $i = j + 1$ and $L_{ij} = 0$ for every other entry. Let U_{ij} be the entry in the ith row and jth column of U. U_n is the $n \times n$ matrix with entries $U_{ij} = \frac{i+1}{i}$ for $i = j$, $U_{ij} = -1$ for $i = j - 1$ and $U_{ij} = 0$.

Proof: We show that $P_n A_n = L_n U_n$ using the values of the matrix from the claim above.

$$P_n = I_n \Rightarrow A_n = L_n U_n$$

Let A_{ij} be the entry in the ith row and jth column of matrix A. Now we consider the different cases for the values of i and j. For $i = j = 1$ (the first diagonal of A) we have:

$$A_{11} = \sum_{k=1}^{n} L_{1k} U_{k1} = L_1 1 U_1 1 + \sum_{k=2}^{n} L_{1k} U_{k1} = 1 \cdot 2 + 0 = 2$$

For $i = j$ where $i > 1$ (the rest of the diagonals of A):

$$A_{ii} = \sum_{k=1}^{n} L_{ik} U_{ki} = \sum_{k=1}^{i-2} L_{ik} U_{ki} + L_{i(i-1)} U_{(i-1)i} + L_{ii} U_{ii} + \sum_{k=i+1}^{n} L_{ik} U_{ki}$$

$$A_{ii} = 0 + \frac{-(i-1)}{i} \cdot -1 + 1 \cdot \frac{i+1}{i} + 0 = \frac{i - 1 + i + 1}{1} = \frac{2i}{i} = 2$$

For $i = j + 1 \Rightarrow j = i - 1$ (sub diagonal entries of A):

$$A_{i(i-1)} = \sum_{k=1}^{n} L_{ik} U_{k(i-1)} = \sum_{k=1}^{i-2} L_{ik} U_{k(i-1)} + L_{i(i-1)} U_{(i-1)(i-1)} + \sum_{k=i}^{n} L_{ik} U_{k(i-1)}$$

$$A_{i(i-1)} = 0 + \frac{-(i-1)}{i} \cdot \frac{i}{i-1} + 0 = -1$$

For $i = j - 1 \Rightarrow j = i + 1$ (super diagonal entries of A):

$$A_{i(i+1)} = \sum_{k=1}^{n} L_{ik} U_{k(i+1)} = \sum_{k=1}^{i-1} L_{ik} U_{k(i+1)} + L_{ii} U_{i(i+1)} + \sum_{k=i+1}^{n} L_{ik} U_{k(i+1)} = 0 + 1 \cdot (-1) + 0 = -1$$

For every other case where $i \neq j$, $i \neq j - 1$, and $i \neq j + 1$ we have that $A_{ij} = \sum_{k=1}^{n} L_{ik} U_{kj}$. This is because for $k \in [1, n]$ either L_{ik} or U_{kj} (or both values) will be zero given that $i \neq j$, $i \neq j - 1$, and $i \neq j + 1$. Therefore, the claim above is correct (the factors as described above are correct).

Problem 4. (10 points) Orthogonal Matrices

Part (a)
Let $P \in \mathbb{M}_{n\times n}$ be a permutation matrix. By the definition of matrix multiplication and transpose of a matrix we have:

$$(PP^T)_{ij} = \sum_{k=1}^{n} P_{ik}(P^T)_{kj} = \sum_{k=1}^{n} P_{ik}P_{jk}$$

Note that the rows of P are the standard basis vectors of $\mathbb{R}^n$, since P is a permutation matrix. So for any particular row P_i of P we have that there must exist only one value P_iw in the row P_i such that $P_iw = 1$ and where all the other values in the row are 0. So for $i = j$ we have:

$$(PP^T)_{ij} = (PP^T)_{ii} = \sum_{k=1}^{n} P_{ik}P_{ik} = \sum_{k=1}^{w-1} P_{ik}P_{ik} + P_{iw}P_{iw} + \sum_{k=w+1}^{n} P_{ik}P_{ik} = 0 + 1 + 0 = 1$$

Now for $i \neq j$ we have that $(PP_T)_{ij} = \sum_{k=1}^{n} P_{ik}P_{jk} = 0$ because no two different rows in P can have their entry be 1 on the same column since P is a permutation matrix and each row pertains to a distinct standard basis vector of $\mathbb{R}^n$.
Hence we have that $PP^T = I_n \Rightarrow P^T = P^{-1}$. Therefore any permutation matrix P is orthogonal.

Part (b)
An orthogonal matrix is not necessarily a permutation matrix. Consider the following example:

$$A = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \Rightarrow A^T = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

Observe that $A(A^T) = I_2$. Hence we have that $A^{-1} = A^T$, so A is an orthogonal matrix. In other words, A is an orthogonal matrix but it is not a permutation matrix as it has -1 in one of its entries.

Problem 5. (10 points) Skew-Symmetric Matrices

Let $A \in \mathbb{M}_{n\times n}$.
Claim: A can be written as a sum of a symmetric matrix S and a skew-symmetric matrix J ($A = S + J$ such that $S^T = S$ and $J^T = -J$.

Proof:
Let $S \in \mathbb{M}_{n\times n}$ such that $S_{ij} = A_{ij}$ for $i = j$, and $S_{ij} = S_{ji} = \frac{A_{ij}+A_{ji}}{2}$ for $i \neq j$.
Let $J \in \mathbb{M}_{n\times n}$ such that $J_{ij} = 0$ for $i = j$, and $J_{ij} = -J_{ji} = \frac{A_{ij}-A_{ji}}{2}$ for $i \neq j$

For $i = j$ we have that $S_{ij} + J_{ij} = A_{ij} + 0 = A_{ij}$
For $i \neq j$ we have $S_{ij} + J_{ij} = \frac{A_{ij}+A_{ji}}{2} + \frac{A_{ij}-A_{ji}}{2} = A_{ij}$

Therefore for any i, j we have that $S_{ij}+J_{ij} = A_{ij}$. Clearly S is a symmetric matrix and J is a skew-symmetric matrix. Thus we have proven that A can be written as a sum of a symmetric matrix S and a skew-symmetric matrix J (A = S +J).

Problem 6. (10 points) Determinants are Expensive to Compute

Problem 7. (10 points) Solving Linear Systems

Part (a)
The rank of B is 2. Let b_j be the the jth column of B. Observe that any column in B is the linear combination of two of its column vectors. More specifically we have that

$$b_j = \begin{bmatrix} 1 \\ n+1 \\ \cdot \\ \cdot \\ \cdot \\ n^2-n+1 \end{bmatrix} + (j-1) \begin{bmatrix} 1 \\ 1 \\ \cdot \\ \cdot \\ \cdot \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ n+1 \\ \cdot \\ \cdot \\ \cdot \\ n^2-n+1 \end{bmatrix} + (j-1) \left(\begin{bmatrix} 2 \\ n+2 \\ \cdot \\ \cdot \\ \cdot \\ n^2-n+2 \end{bmatrix} - \begin{bmatrix} 1 \\ n+1 \\ \cdot \\ \cdot \\ \cdot \\ n^2-n+1 \end{bmatrix} \right) = b_1 + (j-1)(b_2 - b_1)$$

Therefore B only has 2 linearly independent columns, hence the rank of B is 2.

Part (b)
The only non-zero component is in the last entry of x. In particular $x_n = 0.01$ where n is the dimension of x.

```
N = 10;
A = diag(2*ones(1,N)) + diag(-1*ones(1,N-1),1) + diag(-1*ones(1,N-1),-1);
[L,U,P] = lu(A);
```

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.

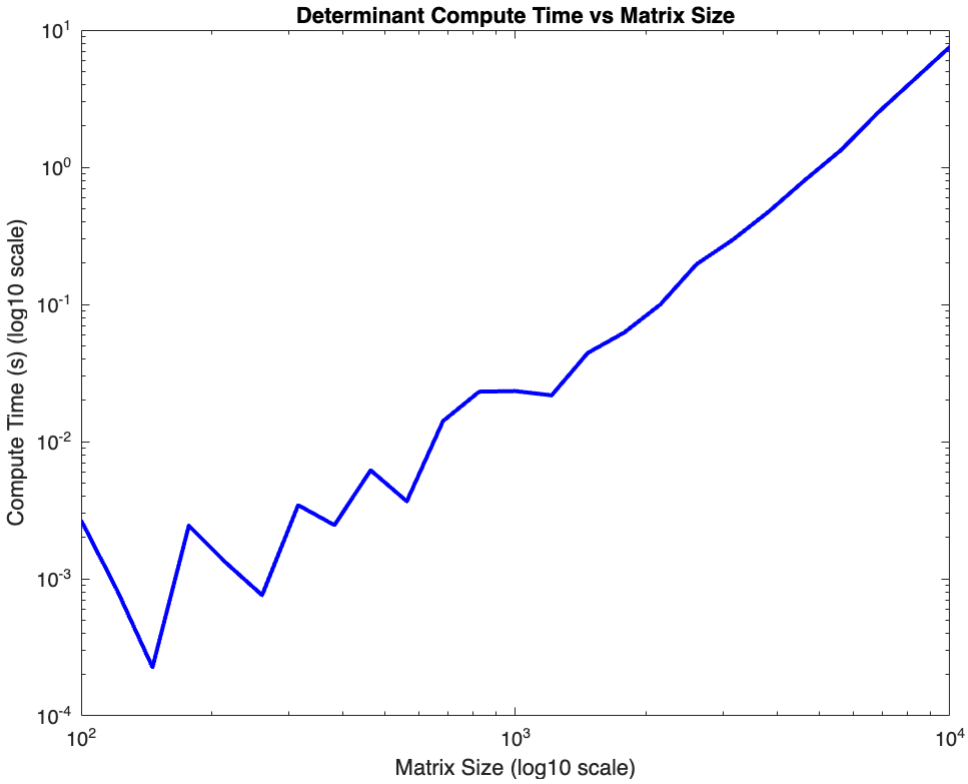
Problem 6 (10 points) Determinants are Expensive to Compute

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(1, 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
    d = 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 (log10 scale)");
ylabel("Compute Time (s) (log10 scale)");
title("Determinant Compute Time vs Matrix Size");
```



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 $\text{rank}(B)$. 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
    % TODO (one-liner)
    B = [B ; B(i-1, n) + 1 : B(i-1, n) + n];
end
r = rank(B); % check your answer here
```

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 $[n - \text{rank}(B)]$ 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;
c = c';

%{
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)
```

```
V = 100×98
   -0.3457    0.2392   -0.1498    0.0242    0.1751   -0.1583   -0.4067   -0.0931 ···
   -0.1408   -0.2431   -0.0637    0.4665    0.2635   -0.1300    0.0860    0.0428
    0.0080   -0.0918    0.0407   -0.5383    0.3701   -0.0161   -0.0286   -0.0025
   -0.0349    0.0377    0.0527    0.0064   -0.0890   -0.0435   -0.0828    0.0762
   -0.2189    0.3906    0.0220    0.0277   -0.0629   -0.0031    0.2418   -0.0842
    0.3323    0.3331    0.0613    0.1586    0.1356    0.2603   -0.1818   -0.1139
   -0.0574    0.1939    0.0237   -0.0137   -0.0453    0.1052    0.0520    0.0641
    0.4032    0.0392   -0.0076   -0.0808   -0.0955   -0.2418   -0.0208   -0.1190
   -0.0429   -0.2063    0.1916   -0.0541   -0.1194    0.4895   -0.2384   -0.1217
    0.0741    0.0233    0.0026   -0.0685    0.0809    0.1592    0.0195    0.1674
    ⋮
```

```
%{
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
```

```
k = 98
```

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
-> Hint2: Use backslash
%}
alpha = V(1:k, :) \ (-1 * 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
 0.0000
 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
    0.0100
```

```
%{
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);
```

```
   9.0801e-14
```

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