Math 318 Homework 2

(1) **Permutation Matrices**

(6.1 #34) Consider the following *permutation matrix*:

$$P = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Let's examine where the name of P comes from and the eigenvalues and eigenvectors of P.

(a) What is the effect of multiplying the vector $\mathbf{x} = (x_1, x_2, x_3, x_4)^\top$ by the matrix P, i.e., what is $P\mathbf{x}$?
(b) Write down the permutation matrix of the permutation 1324.
(c) Compute all eigenvalues of the matrix P shown above.
(d) For each eigenvalue find a corresponding eigenvector. You shouldn't have to do any tedious computations such as finding nullspaces.

(2) **Product and sum of eigenvalues**

Let A be an $n \times n$ matrix with eigenvalues $\lambda_1, \ldots, \lambda_n$ (maybe not all distinct).

(a) (i) (6.2 #20) Suppose A is diagonalizable. Then show that $\det(A) = \lambda_1 \lambda_2 \cdots \lambda_n$.
(ii) (6.1 #16) Argue that for any matrix A, $\det(A) = \lambda_1 \lambda_2 \cdots \lambda_n$.
Hint: Recall that $\det(A - \lambda I) = (-1)^n (\lambda - \lambda_1)(\lambda - \lambda_2) \cdots (\lambda - \lambda_n)$

(b) (6.2 #20, #21) The *trace* of a square matrix is the sum of its diagonal entries. For example, if $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$, then $\text{trace}(A) = a + d$.
(i) Argue that $\text{trace}(PQ) = \text{trace}(QP)$ for any two $n \times n$ matrices P and Q.
Hint: To prove the general case, write down $P = (p_{ij})$ and $Q = (q_{ij})$ symbolically and compute the trace of PQ and QP symbolically. The trace expression will be a sum of sums. You then need to argue that these sums are the same. You might first check the statement for two symbolic 2×2 matrices to understand what is happening.
(ii) Whether you can do (a) or not, use the result to show that if A is diagonalizable, then the trace of A is the sum of the eigenvalues of A.
(iii) *In general, the trace of* any $n \times n$ *matrix is the sum of its eigenvalues. (No need to argue this.)*
Show that the trace of *any* 3×3 matrix is the sum of its eigenvalues. (This requires starting with a symbolic 3×3 matrix.)
Hint: Here is a hint for general n that you can also use for the $n = 3$ case. Recall that the characteristic polynomial $p(\lambda) = (-1)^n (\lambda - \lambda_1) \cdots (\lambda - \lambda_n)$ and also $p(\lambda) = \det(A - \lambda I)$. Think about where you might see the sum of the eigenvalues of A if you expanded the product and where you might see it if you computed $\det(A - \lambda I)$ using permutations.

(3) † (6.2 #25) Recall that the column space of a $n \times n$ matrix A, denoted $\text{Col}(A)$, is the span of the columns of A. Suppose A is a non-zero $n \times n$ matrix such that $A^2 = A$.
(a) Show that $\lambda = 1$ is an eigenvalue of A with eigenspace equal to $\text{Col}(A)$.
(b) What is the eigenspace of $\lambda = 1$ if A is invertible? Is A diagonalizable in this case? If yes, write its diagonalization.

(c) If A is not invertible, what are its eigenvalues and their eigenspaces? Is A diagonalizable in this case? If yes, write its diagonalization. **Note:** If you can write its diagonalization, it will not be with explicit vectors, but rather a general diagonalization in terms of vectors coming from the various eigenspaces that you have found.

There will be a series of problems in this class with † that all relate to projections.

(4) (6.2 #24, #29)

(a) Consider the set of all 4×4 matrices that are diagonalized by the same eigenvector matrix U:

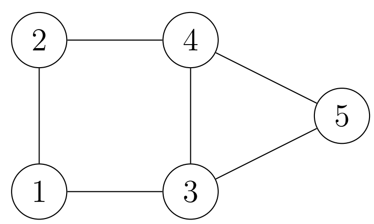
$$S_U = \left\{ U \Lambda U^{-1} \in \mathbb{R}^{4\times 4} : \Lambda \text{ is a diagonal matrix} \right\}.$$

Show that S_U is a subspace of $\mathbb{R}^{4\times4}$. (Check the properties of a subspace.)

(b) What is S_I where I is the identity matrix?

(c) Suppose the same U diagonalizes A and B, i.e., $A = U\Lambda_1 U^{-1}$ and $B = U\Lambda_2 U^{-1}$. Argue that $AB = BA$. (Recall that in general matrix multiplication is not commutative, i.e., $AB \neq BA$.)

(5) * **How to find a triangle in a graph.** A *graph* G is a collection of *nodes* labeled $1, 2, \ldots, n$ and *edges* which are pairs of nodes. Shown below is a graph with 5 nodes labeled $1, \ldots, 5$ and edges $\{1,2\}, \{1,3\}, \{2,4\}, \{3,4\}, \{3,5\}, \{4,5\}$. A triangle in a graph G is a triple of nodes i, j, k such that all edges $\{i,j\}, \{i,k\}, \{j,k\}$ are present in G. For example $3, 4, 5$ forms a triangle in the graph below. Two nodes i and j are *neighbors* in G if $\{i,j\}$ is an edge in G.



If the graph G is very large it becomes hard to decide if there is a triangle in it by simply looking at the graph. In this problem we will see that linear algebra can be used to decide if a graph contains a triangle.

(a) The *adjacency matrix* A of a graph G with n nodes is a $n \times n$ matrix defined as follows. The rows and columns of A are indexed by the node labels $1, \ldots, n$ and the (i,j)-entry of A is 1 if the pair $\{i,j\}$ is an edge in G and 0 otherwise. (An example can be seen in Problem 2.4A on page 76 in Strang's book. See also Chapter 3 of the lecture notes.)
Write down the adjacency matrix of the graph G shown above.

(b) Let $B = A^2$ where A is the adjacency matrix of G. The entry b_{ij} in B is the dot product of two vectors sitting in A. Which vectors are they? In our example, how is b_{23} formed?

(c) For three nodes i, j, k from G, argue that $a_{ik}a_{kj}$ is 1 exactly when k is a common neighbor of i and j.

(d) Using the above, what does b_{ij} count?

(e) The nodes i, j, k form a triangle if and only if i and j are neighbors and k is a common neighbor of i and j. If i, j, k is a triangle what property must a_{ij} and b_{ij} have?

(f) Putting all this together can you construct an algorithm that takes as input two indices i, j, and determines whether or not there exists a triangle with the edge between i and j as one of the sides . Justify your algorithm. **Think of an algorithm that uses A and B. If you use only A you will likely have a less efficient algorithm.** You don't need to write computer code, just writing out the steps in words is enough.

(g) (optional) If you know about running times of algorithms, do you see how fast this algorithm runs? Is it faster than checking all triples of nodes in G?