Deep Dive: Final Cumulative Problem Set

Loan Nguyen, William Nguyen, Waiyego Maina

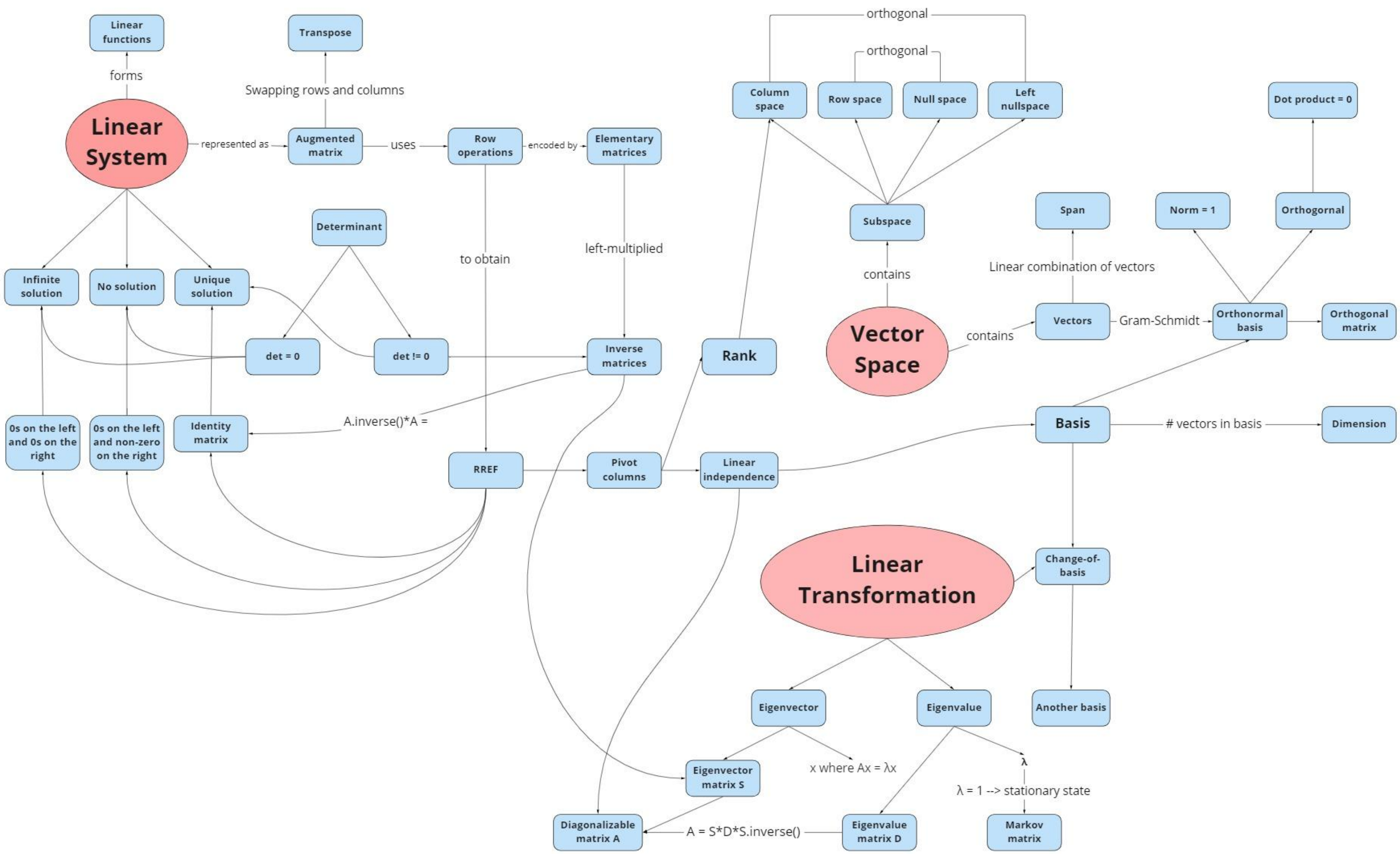
CS113 Spring 2022

Minerva University

Concept exploration

Concept map

Because the concept map is wide and contains multiple nodes, it is difficult to squeeze the map legibly into one page of this document. To view a higher-resolution version of the concept map, please visit [this link].



Glossary

Linear Functions

Linear functions are functions where variables are raised to the first power (or zeroth power for constants).

Linear System

A linear system is a set of linear functions.

Consistent Systems

A consistent system is a system of equations that have at least one solution. It can also have an infinite number of solutions.

Inconsistent Systems

An inconsistent system is a system of equations that do not have any solutions.

Augmented Matrices

In an augmented matrix, the left side represents the coefficients of the linear system while the right side represents the right-hand side of each equation in the linear system.

Row Operations

Row Operations are different arithmetic and algebraic operations that are applied to the rows of a matrix. There are three basic row operations: swapping rows, multiplying a row by a scalar, and addition/subtraction between rows.

Gaussian Elimination

A technique to systematically eliminate all but one variable from the coefficient side of a matrix. Start with the top left and use that entry and row operations to eliminate (generate 0s) everywhere below it.

RREF Matrix

In a Row Reduced Echelon Form (RREF) Matrix, all pivots are 1s, while below and above every pivots are 0s. If we can find pivots in each column, then RREF will be an identity matrix. Otherwise, the RREF has non-pivot columns and is not an identity matrix.

Matrix Multiplication

Given an m*n matrix A and an n*k matrix B, then each entry of the product A*B follows the format $c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \ldots + a_{in}b_{nj}$ where a_{i1} means the entry at row i and column 1 of matrix A, b_{1j} means the entry at row 1, column j of matrix B, etc.

Elementary Matrices

When we left-multiply an elementary matrix with a matrix, a row operation as indicated in the elementary matrix is conducted. In addition, multiplying elementary matrices together on the left will generate the inverse matrix of the original matrix.

Inverse Matrix

A matrix is invertible when its RREF is an identity matrix. When we multiply an inverse matrix with the original matrix (regardless of the side of multiplication), an identity matrix is produced. If invertible, we can find the inverse of a matrix by augmenting the matrix with an identity matrix and reducing it to RREF.

Transpose Matrix

The transpose of a matrix is obtained by swapping rows and columns.

Determinant

The determinant of an n*n matrix A, denoted as det(A), is given by $det(A) = \sum_{j=1}^{n} (-1)^{i+j} a_{ij} |M_{ij}|$, in which a_{ij} is the entry of matrix A at row i and column j, and M_{ij} is the cofactor of a_{ij}. The determinant can describe the number of solutions in the linear system represented by matrix A. If det(A) is not 0, there is a unique solution. If det(A) = 0, there is either no solution or infinite number of solutions.

Geometric Vectors

A geometric vector is an object that has both magnitude and direction.

Vector Space

For a non-empty set V and a field of scalars K, V is a vector space if it follows the following properties for all u, v, w in V and b,c in K:

1. Closed under addition: u+v still in V
2. Closed under scalar multiplication: b*u in V
3. Existence of 0 vector: u + 0 = u
4. Existence of additive inverse: u + (-u) = 0
5. Existence of multiplicative identity: 1u = u for scalar 1
6. Addition is commutative: u+v = v+u
7. Addition is associative:

   (u+v)+w = u+(v+w)

   b(cu) = (bc)u
8. Holds the distribution property: b(u+v) = bu+bv

Subspace

Let W be a subset of a vector space V. W is a subspace of V if W satisfies three following conditions:

1. W is closed under addition: w_1 + w_2 still in W for all w_1, w_2 in W
2. W is closed under scalar multiplication: alpha*w in W for all w in W and alpha in R
3. W contains zero vector

Span

For a set of vectors S, the span of S, span(S) is the set of all linear combinations of the vectors in S.

Linear Independence

A set of vectors are linearly independent if no vector in this set can be expressed as a linear combination of other vectors.

Basis

For a vector space V and one of its subset S, S is called the basis of the vector space V if

a. S is a linearly independent set

b. span(S) covers all of V

Dimension

The dimension of a vector space is the number of vectors in its basis.

Coordinates of a vector

For a given basis S, a vector v can be expressed as the linear combination of the basis vectors in S. The coefficients of the linear combination are called the coordinates of v in basis S.

Dot Product

The dot product between two vectors are defined by multiplying corresponding components of each vector and adding all of them together to get a scalar number.

Vector Projection

If $\vec{w}$ is the projection of $\vec{v}$ onto $\vec{a}$, $\vec{w} = \frac{\vec{v}\vec{a}}{||\vec{a}||^2}\vec{a}$

Orthonormal Basis

A basis is orthonormal if all the vectors in the basis have a magnitude of 1 and orthogonal to each other.

Gram-Schmidt Algorithm

An algorithm to orthonormalize a set of vectors.

Orthogonal Matrix

If the column of a matrix creates an orthonormal basis for a vector space, the matrix is called an orthogonal matrix

Rank

The rank of a matrix is the number of pivot columns it has in its RREF. In other words, the number of linearly independent columns or rows in a matrix is the rank of the matrix.

Fundamental Subspaces

1. Column Space

The vector space spanned by all linearly independent columns of a matrix A is called the column space of A, col(A). For a m*n matrix, Col(A) lies in R^m.

2. Row Space

   The vector space spanned by all linearly independent rows of a matrix A is called the row space of A, row(A). For a m*n matrix, row(A) lies in R^n.

3. Null Space

   The vector space which contains all possible solution(s) of $\vec{x}$ for the system: $A\vec{x} = \vec{0}$ for a matrix A is called the Null space of A, N(A). For a m*n matrix, N(A) lies in R^m.

4. Left Null Space

   The vector space which contains all possible solution(s) of $\vec{y}$ for the system: $A^T\vec{y} = \vec{0}$ for a matrix A is called the Left-Null space of A, N(A^T). For a m*n matrix, N(A^T) lies in R^m.

Change-of-basis Matrix

If we can change the coordinates of vector v in a basis B_1 to its coordinates in another basis B_2 by left multiplying a matrix with v's initial coordinates (in basis B_1), then that matrix is called the change-of-basis matrix from basis B_1 to basis B_2.

Mapping

A map between vector spaces is a function that takes as input a vector from one vector space and outputs a vector from another vector space.

Linear Mapping

A linear map between vector space V and W is a map L: V → W such that for u, v in V and scalar k:

- L(u+v) = L(u) + L(v)
- L(ku) = kL(u)

Eigenvector

Eigenvector is a vector that does not change direction after a linear transformation.

Eigenvalue

Eigenvector v does not change direction after a linear transformation, but its length can change. $T(v) = \lambda v$. λ is the eigenvalue of eigenvector v. One eigenvalue can be associated with multiple eigenvectors.

Markov Matrix

Each column in the Markov (transition) matrix adds up to 100% probability. This means that, given a current state, we know how the probabilities will be distributed in the next time step.

Stationary distribution

The properly normalized eigenvector associated with eigenvalue = 1 is the stationary distribution of the Markov process.

Diagonalization

If an n*n matrix A has n linearly independent eigenvectors $v_1, v_2, \ldots, v_n$, then for $S = [v_1 \mid v_2 \mid \ldots \mid v_n]$, $D = S^{-1}AS$. The entries along the main diagonal of D equal the corresponding eigenvalues in the same order as the eigenvectors in S. We call S the eigenvector matrix and A is a diagonalizable matrix.

The Spectral Theorem

Let A be an n*n symmetric matrix. Then there exists an orthonormal basis of eigenvectors of A

Deep Dive Problems

Question 1: A Bit of Algebra

(a)

Additive identity: 0 will be the identity of both 0 and 1 because the value still stays unchanged when plus 0

Additive inverse: 0 is the additive inverse for bit 0 and 1 is the additive inverse for bit 1 because in terms of bit, $0 + 0 = 0$ and $1 + 1 = 0$.

(b)

a)

$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0\times 1 + 0\times 1 + 1\times 1 \\ 1\times 1 + 1\times 1 + 1\times 1 \end{bmatrix} = \begin{bmatrix} 0 + 0 + 1 \\ 1 + 1 + 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

b)

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1\times 0 + 1\times 1 & 1\times 1 + 1\times 1 \\ 1\times 0 + 0\times 1 & 1\times 1 + 0\times 1 \\ 0\times 0 + 1\times 1 & 0\times 1 + 1\times 1 \end{bmatrix}$$

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

$$\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^{-1}$$

•) To find the inverse, we augment identity matrix

$$\left[\begin{array}{cc|cc} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{array}\right]$$

•) Finding elementary matrices that output the RREF of the original matrix

$$\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \xrightarrow{R1 = R1 + R2} \underbrace{\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}}_{M_1} \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \xrightarrow{R2 = R2 + R1} \underbrace{\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}}_{M_2} \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

•) We know that

$$\left[\begin{array}{c|c} M_2 M_1 A & M_2 M_1 I \end{array}\right]$$

$$= \left[\begin{array}{c|c} I & A^{-1} \end{array}\right]$$

$$\Rightarrow A^{-1} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^{-1} = M_2 M_1 I$$

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

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

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

Check by SAGE:

In [1]

```
A = matrix(GF(2), [[0, 0, 1], [1, 1, 1]])
B = vector(GF(2), [1, 1, 1])
A*B
```

Run Code

Out [1]

```
(1, 1)
```

In [2]

```
A = matrix(GF(2), [[1, 1], [1, 0], [0, 1]])
B = matrix(GF(2), [[0, 1], [1, 1]])
A*B
```

Run Code

Out [2]

```
[1 0]
[0 1]
[1 1]
```

In [3]
```
A = matrix(GF(2), [[0, 1], [1, 1]])
A.inverse()
```

Run Code

Out [3]
```
[1 1]
[1 0]
```

(c) Linear Equations

In [5]
```
A = matrix(GF(2), [[1, 1, 1], [1, 1, 0], [1, 0, 1]]).augment(vector(GF(2), [0, 1, 1]), subdivide = True)
A.rref()
```

Run Code

Out [5]
```
[1 0 0|0]
[0 1 0|1]
[0 0 1|1]
```

When we reduce the matrix to the RREF form, we can easily figure out this system of equations has a unique solution where $x_1 = 0, x_2 = 1, x_3 = 1$

(d) Correcting Codes

i. ***Repetition Code:***

(1) $< 0, 0, 0 >$**:** The interpreter might believe that there are no flips here because all entries are exactly the same, thus making the interpreter easily infer that the sent message here is 0.

(2) $< 1, 0, 1 >$: The interpreter might believe that the majority of components refers to the true sent message. Therefore, here, the interpreter will guess that there is only one flip and the sent message is 1 (because 1 is the majority)

(3) $< 0, 0, 1 >$**:** Likewise, the majority of components here is 0, which causes the interpreter to think that the sent message is 0 and there is one flip.

In repetition code, with 3 repeated components of the message and the assumption that the true message is the majority of components, the interpreter can recognize at most only 1 flip because with 2 flips, the interpreter might still assume only 1 flip and infer the wrong message.

ii. Parity Code:

(1) $< 1, 1, 1, 1, 0 >$: Currently, everything makes sense with $x_1 = x_2 = 1$ and it is true that $x_1 + x_2 = 1 + 1 = 0$, which follows the addition rule of Bit. Therefore, the interpreter can believe that there is no flip here, and the originally sent message is (1, 1).

(2) $< 0, 1, 0, 0, 0 >$**:** Here at positions of x_2, we both have 0s. Therefore, the interpreter might believe that $x_2 = 0$. Then, for the last position to be 0, x_1 has to be 0. As a result, it can be inferred that there is one flip here, and the original message is (0, 0)

(3) $< 1, 0, 0, 0, 1 >$**:** Both positions of x_2 imply that $x_2 = 0$. The last position is currently 1. If the last position is true, then $x_1 = 1 \rightarrow$ there is one flip here at the second position. If the last position is flipped, then the first position of x_1 is also flipped $\rightarrow x_1 = x_2 = x_1 + x_2 = 0$

(4) $< 0, 1, 1, 1, 1 >$**:** It is consistent that both positions of $x_2 = 1$. With the similar reasoning, if the last position is 1 without any flips, then there will be 1 flip here at one position of x_1. Otherwise, the interpreter can also believe that $x_1 + x_2 = 0$ which means $x_1 = x_2 = 1$ and there is one flip at position of x_1

(5) $< 0, 1, 0, 0, 1 >$**:** The same reasoning applies when the interpreter can trust that $x_2 = 0$. He might infer there is only one flip which is at the first position because x_1 is supposed to be 1 so that $x_1 + x_2 = 1 + 0 = 1$. But at the same time, there can be two flips because if the last position is actually 0 then there is another flip at the second position because $x_1 = 0$

(6) $< 0, 1, 0, 1, 0 >$**:** The extreme case here is both positions of x_1 and x_2 are inconsistent. Therefore, there has to be at least 2 flips. If the last position is indeed 0, then the second and fourth OR the first and third positions are flipped. But if the last position is flipped, then flips also occur to the first and fourth OR the second and third positions.

After this in-depth analysis, the interpreter can only correct 3 flips at maximum for parity codes.

(e) Linear code

- For Repetition Code, we send a 1-bit code as a 3-bit code, so the generator matrix will be a 3*1 matrix. In the Repetition Code, because we repeat the exact message 3 times, our matrix will have entries being 1 in every row.
- For Parity Code, we send a 2-bit code as a 5-bit code, so the generator matrix will be a 5*2 matrix. The first two rows of the matrix will preserve values of x_1 and the next two rows preserve values of x_2. The last row will just sum two values of $x_1 + x_2$.

Therefore, the two matrices manifesting the Generator can be shown here:

```
In [1]
repetition = matrix([[1], [1], [1]])
parity = matrix([[1, 0], [1, 0], [0, 1], [0, 1], [1, 1]])
repetition, parity
```

Run Code

```
Out [1]
(
     [1 0]
     [1 0]
[1]  [0 1]
[1]  [0 1]
[1], [1 1]
)
```

(f)

i. The encoding message of $x = < 0, 1, 0, 1 >$ is the resulting product of multiplying the generator matrix with the original code.

→ We have the following encoding message:

```
H = matrix(GF(2), [[1, 0, 0, 1], [0, 1, 1, 0], [0, 0, 1, 1], [0, 1, 1, 1], [1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 0, 1]])
x = vector(GF(2), [0, 1, 0, 1])
print(H*x)

```

Run Code

```
Out [2]    (1, 1, 1, 0, 0, 1, 1)
```

ii. Pseudocode to decode a codeword:

The decoding code is the solution x of the system Hx = y with H being the generator matrix and y being the codeword.

1. Construct a system of equations with the generator matrix H augmented by the vector y.
2. Reduce the augmented matrix to the RREF form so that we can understand the consistency as well as number of solutions in the system of equations
3. Based on the RREF and the number of solutions, conclude about the decoding message:
    a. If the system of equations has a unique solution, that is the message we are looking for and we successfully decode it!
    b. If the system of equations is inconsistent, the message received is not decodable!

Now we are going to decode $y_1 = \ < 1,\ 1,\ 1,\ 0,\ 0,\ 1,\ 1 >$

```
# ii.
y = vector(GF(2), [1, 1, 1, 0, 0, 1, 1])
M_ii = matrix(GF(2), H.augment(y, subdivide = True))
M_ii.rref()
```

Run Code

```
Out [6]
[1 0 0 0|0]
[0 1 0 0|1]
[0 0 1 0|0]
[0 0 0 1|1]
[0 0 0 0|0]
[0 0 0 0|0]
[0 0 0 0|0]
```

Here we can see that the system of equations has a unique solution because all columns have a pivot. Therefore, we can conclude that we did decode our message and the original one is $x = \ < 0,\ 1,\ 0,\ 1 >$, which is consistent with what we did in question f(i).

iii. Decode the new message $y_2 = < 0, 1, 1, 0, 0, 1, 1 >$.

```
# iii.
y = vector(GF(2), [0, 1, 1, 0, 0, 1, 1])
M_iii = matrix(GF(2), H.augment(y, subdivide = True))
M_iii.rref()
```

Run Code

```
Out [11]
[1 0 0 0|0]
[0 1 0 0|0]
[0 0 1 0|0]
[0 0 0 1|0]
[0 0 0 0|1]
[0 0 0 0|0]
[0 0 0 0|0]
```

Here, the system of equations is inconsistent because in the fifth row, all entries on the left are 0 but the right-side entry is non-zero. Therefore, this system of equations has no solutions and our message is not decodable.

iv.

Under the assumption of at most one flip, the standard basis vector makes sense because e_i is the i-th standard basis vector where the i-th position is 1 and other positions are all 0s. We notice that entries in the vector can only be 0 or 1. Both values, when added by 1, will change to the other one (1 + 1 = 0, 0 + 1 = 1). Therefore, if there is one flip occurring at y_2, that flipped element must have been added by 1 (at the i-th position of the standard basis vector). In other words, because y_1 and y_2 only differ at one element, and the former one is decodable while the other is not, there should exist a standard basis vector that makes y_1 become y_2.

v.

We have just proven that $y_2 = y_1 + e_i$

$\Leftrightarrow y_2 + e_i = y_1 + e_i + e_i$ (Add standard basis vectors to both sides)

$\Leftrightarrow y_2 + e_i = y_1$ ($e_i + e_i = 0$ because only the i-th position of e_i is 1 and other positions are all 0s. $1+1 = 0$, and $0 + 0 = 0$, so $e_i + e_i$ will result in zero vector)

$\rightarrow y_1 = y_2 + e_i$

vi.

We know that there is a standard basis vector that when added to y_2, makes it equal to y_1, which is decodable. Therefore, we just have to experiment with all standard basis vectors possible to find which standard basis vector corrects this non-decodability of our message. We use a loop in our algorithm to fix this problem.

```
# vi.
H = matrix(GF(2), [[1, 0, 0, 1], [0, 1, 1, 0], [0, 0, 1, 1], [0, 1, 1, 1], [1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 0, 1]])
y2 = vector(GF(2), [0, 1, 1, 0, 0, 1, 1])
for i in range(7):
    e = vector(GF(2),[0, 0, 0, 0, 0, 0, 0])
    e[i] = 1
    y_new = y2 + e
    print('The standard basis vector with 1 entry at the', i + 1, 'position')
    print(H.augment(y_new, subdivide = True).rref())
```

Run Code

Out [25]

```
The standard basis vector with 1 entry at the 1 position
[1 0 0 0|0]
[0 1 0 0|1]
[0 0 1 0|0]
[0 0 0 1|1]
[0 0 0 0|0]
[0 0 0 0|0]
[0 0 0 0|0]
The standard basis vector with 1 entry at the 2 position
[1 0 0 0|0]
[0 1 0 0|0]
[0 0 1 0|0]
[0 0 0 1|0]
[0 0 0 0|1]
[0 0 0 0|0]
[0 0 0 0|0]
```

```
Out [25]  The standard basis vector with 1 entry at the 3 position
[1 0 0 0|0]
[0 1 0 0|0]
[0 0 1 0|0]
[0 0 0 1|0]
[0 0 0 0|1]
[0 0 0 0|0]
[0 0 0 0|0]
The standard basis vector with 1 entry at the 4 position
[1 0 0 0|0]
[0 1 0 0|0]
[0 0 1 0|0]
[0 0 0 1|0]
[0 0 0 0|1]
[0 0 0 0|0]
[0 0 0 0|0]
The standard basis vector with 1 entry at the 5 position
[1 0 0 0|1]
[0 1 0 0|1]
[0 0 1 0|0]
[0 0 0 1|1]
[0 0 0 0|0]
[0 0 0 0|0]
[0 0 0 0|0]
```

```
The standard basis vector with 1 entry at the 6 position
[1 0 0 0|0]
[0 1 0 0|0]
[0 0 1 0|0]
[0 0 0 1|0]
[0 0 0 0|1]
[0 0 0 0|0]
[0 0 0 0|0]
The standard basis vector with 1 entry at the 7 position
[1 0 0 0|0]
[0 1 0 0|0]
[0 0 1 0|0]
[0 0 0 1|0]
[0 0 0 0|1]
[0 0 0 0|0]
[0 0 0 0|0]
```

After experimenting with all possible standard basis vectors, we figure out that there are 5 standard basis vectors that cannot make the decodability of y_2 possible. Instead, with 2 standard basis vectors where the 1-entry falls into the first and the fifth position, we can successfully decode y_2

→ From y_2, we can successfully decode $x = \langle 0, 1, 0, 1 \rangle$ $or\, x = \langle 1, 1, 0, 1 \rangle$

vii. Adjusted Pseudocode

The adjustment will only start within the condition statement where we find that our original system of equations is inconsistent.

1. Construct a system of equations with the generator matrix H augmented by the vector y.
2. Reduce the augmented matrix to the RREF form so that we can understand the consistency as well as number of solutions in the system of equations
3. Based on the number of solutions, conclude about the decodability of our message received.
    a. If the system of equations has a unique solution, that is the message we are looking for and we successfully decode it!
    b. If the system of equations is inconsistent, the message received is not decodable:
        i. Here, we need to find a vector that resembles almost all components with vector y. There should only be one difference between these two vectors. And the new vector should be decodable.
        ii. We will now loop over all possible standard basis vectors representing noise in the basis of vector y. We add this noise to vector y so that it can correct the decodability.
        iii. Whichever system of equations now gives us a unique solution, that unique solution is a potential original message.

Question 2: Inside Information

(a) Show that the dot product on R^n fulfills the five conditions of inner product

Suppose $\vec{v}\,(v_1, v_2, v_3, ..., v_n) \in V$ with $v_1, v_2, v_3, ..., v_n \in R$

$\vec{u}\,(u_1, u_2, u_3, ..., u_n) \in V$ with $u_1, u_2, u_3, ..., u_n \in R$

$\vec{w}\,(w_1, w_2, w_3, ..., w_n) \in V$ with $w_1, w_2, w_3, ..., w_n \in R$

- Positivity: $\vec{v} \cdot \vec{v} = v_1^2 + v_2^2 + v_3^2 + ... + v_n^2$

  $v_1^2, v_2^2, v_3^2, ..., v_n^2 \geq 0$ because each is a square of a number in R

  $\rightarrow v_1^2 + v_2^2 + v_3^2 + ... + v_n^2 \geq 0$

  $\rightarrow \vec{v} \cdot \vec{v} \geq 0$

  → Dot product fulfills the positivity condition.
- Definiteness:

  As seen above, $\vec{v} \cdot \vec{v} \geq 0$ for all $\vec{v} \in V$

  $\vec{v} \cdot \vec{v} = 0 \Leftrightarrow v_1^2 = v_2^2 = v_3^2 = ... = v_n^2 = 0$

  $\Leftrightarrow v_1 = v_2 = v_3 = ... = v_n = 0$

  $\Leftrightarrow \vec{v}$ is zero vector

  → Dot product fulfills the definiteness condition
- Additivity:

- $\vec{u} \cdot \vec{w} = u_1 w_1 + u_2 w_2 + ... + u_n w_n$
- $\vec{v} \cdot \vec{w} = v_1 w_1 + v_2 w_2 + ... + v_n w_n$
- $\vec{u} + \vec{v} = (u_1 + v_1, u_2 + v_2, u_3 + v_3, ..., u_n + v_n)$
- $(\vec{u} + \vec{v}) \cdot \vec{w} = (u_1 + v_1)w_1 + (u_2 + v_2)w_2 + ... + (u_n + v_n)w_n$

  $= u_1 w_1 + v_1 w_1 + u_2 w_2 + v_2 w_2 + ... + u_n w_n + v_n w_n$

  $= (u_1 w_1 + u_2 w_2 + ... + u_n w_n) + (v_1 w_1 + v_2 w_2 + ... + v_n w_n)$

  $= \vec{u} \cdot \vec{w} + \vec{v} \cdot \vec{w}$

→ Dot product fulfills the additivity condition

- Homogeneity
  - $a\vec{v} = (av_1, av_2, av_3, ..., av_n)$ for all $a \in R$
  - $a\vec{v} \cdot \vec{w} = av_1 w_1 + av_2 w_2 + ... + av_n w_n$

    $= a(v_1 w_1 + v_2 w_2 + ... + v_n w_n)$

    $= a(\vec{v} \cdot \vec{w})$

→ Dot product fulfills the homogeneity condition

- Symmetry
  - $\vec{v} \cdot \vec{w} = v_1 w_1 + v_2 w_2 + ... + v_n w_n$
  - $\vec{w} \cdot \vec{v} = w_1 v_1 + w_2 v_2 + ... + w_n v_n$

$= v_1 w_1 + v_2 w_2 + ... + v_n w_n$ (We can swap the position because $v_1, v_2, ..., v_n$ and $w_1, w_2, ..., w_n$ are just numbers in R that have commutative property)

$= \vec{v} \cdot \vec{w}$

$\rightarrow$ Dot product fulfills the symmetry condition

(b) Determine whether the following operations are inner products in R^2

i. $\langle \vec{v}, \vec{w} \rangle = |v_1 w_1| + |v_2 w_2|$

This operation is not an inner product. One counter example is $\vec{v}(1, 4)$, $\vec{w}(3, 7)$, $\vec{u}(-2, 1)$

- $\langle \vec{u}, \vec{w} \rangle = |(-2) \times 3| + |1 \times 7| = |-6| + |7| = 6 + 7 = 13$
- $\langle \vec{v}, \vec{w} \rangle = |1 \times 3| + |4 \times 7| = |3| + |28| = 3 + 28 = 31$

  $\rightarrow \langle \vec{u}, \vec{w} \rangle + \langle v, \vec{w} \rangle = 13 + 31 = 44$
- $\vec{u} + \vec{v} = (-1, 5)$
- $\langle \vec{u} + \vec{v}, \vec{w} \rangle = |(-1) \times 3| + |5 \times 7| = |-3| + |35| = 38 \neq 44$

$\rightarrow \langle \vec{u} + \vec{v}, \vec{w} \rangle \neq \langle \vec{u}, \vec{w} \rangle + \langle v, \vec{w} \rangle$

$\rightarrow$ The additivity condition of inner product is not satisfied by this operation

$\rightarrow \langle \vec{v}, \vec{w} \rangle = |v_1 w_1| + |v_2 w_2|$ is not an inner product in R^2

ii. $\langle \vec{v}, \vec{w} \rangle = v_1 w_1 - (v_1 w_2 + v_2 w_1) + 2 v_2 w_2$

This operation is an inner product in R^2. Suppose $\vec{v}(v_1, v_2)$ with $v_1, v_2 \in R$

$\vec{u}(u_1, u_2)$ with $u_1, u_2 \in R$

$\vec{w}\,(w_1,\ w_2)$ with $w_1,\ w_2 \in R$

- Positivity: $\langle \vec{v},\ \vec{v} \rangle = v_1^2 - (v_1v_2 + v_2v_1) + 2v_2^2$

  $= v_1^2 - 2v_1v_2 + 2v_2^2$

  $= (v_1 - v_2)^2 + v_2^2 \geq 0$ for all v_1 and v_2 in R because each term is a square of a number in R

→ The operation fulfills the positivity condition

- Definiteness: $\langle \vec{v},\ \vec{v} \rangle = 0 \Leftrightarrow (v_1 - v_2)^2 = v_2^2 = 0 \Leftrightarrow v_1 = v_2 = 0 \Leftrightarrow \vec{v}$ is a zero vector

→ The operation fulfills the definiteness condition

- Additivity:
  - $\langle \vec{u},\ \vec{w} \rangle = u_1w_1 - (u_1w_2 + u_2w_1) + 2u_2w_2$
  - $\langle \vec{v},\ \vec{w} \rangle = v_1w_1 - (v_1w_2 + v_2w_1) + 2v_2w_2$

→ $\langle \vec{u},\ \vec{w} \rangle + \langle \vec{v},\ \vec{w} \rangle = (u_1w_1 - (u_1w_2 + u_2w_1) + 2u_2w_2) + (v_1w_1 - (v_1w_2 + v_2w_1) + 2v_2w_2)$

$= u_1w_1 + v_1w_1 - (u_1w_2 + u_2w_1 + v_1w_2 + v_2w_1) + 2u_2w_2 + 2v_2w_2$

$= (u_1 + v_1)w_1 - ((u_1 + v_1)w_2 + (u_2 + v_2)w_1) + 2(u_2 + v_2)w_2$

  - $\vec{u} + \vec{v} = (u_1 + v_1,\ u_2 + v_2)$
  - $\langle \vec{u} + \vec{v},\ \vec{w} \rangle = (u_1 + v_1)w_1 - ((u_1 + v_1)w_2 + (u_2 + v_2)w_1) + 2(u_2 + v_2)w_2$

→ $\langle \vec{u} + \vec{v},\ \vec{w} \rangle = \langle \vec{u},\ \vec{w} \rangle + \langle \vec{v},\ \vec{w} \rangle$

→ The operation fulfills the additivity condition

- Homogeneity:
  - $a\vec{v} = (av_1,\ av_2)$ for all $a \in R$

- $\langle a\vec{v}, \vec{w}\rangle = av_1w_1 - (av_1w_2 + av_2w_1) + 2av_2w_2$

  $= a(v_1w_1 - (v_1w_2 + v_2w_1) + 2v_2w_2)$

  $= a\langle \vec{v}, \vec{w}\rangle$

→ The operation fulfills the homogeneity condition

- Symmetry:
  - $\langle \vec{v}, \vec{w}\rangle = v_1w_1 - (v_1w_2 + v_2w_1) + 2v_2w_2$
  - $\langle \vec{w}, \vec{v}\rangle = w_1v_1 - (w_1v_2 + w_2v_1) + 2w_2v_2$

    $= v_1w_1 - (v_1w_2 + v_2w_1) + 2v_2w_2$ (We can swap the position because v_1, v_2 and w_1, w_2 are just numbers in R that have commutative property)

    $= \langle \vec{v}, \vec{w}\rangle$

→ The operation fulfills the symmetry condition

(c) Determine whether the following operations are inner products in M_{2x2}

i. $\langle A, B\rangle = tr(A^TB)$ where the $tr(M) = \sum_{i=1}^{n} [M]_{ii}$ is the trace of M

This operation is an inner product in M_{2x2} because it satisfies all 5 conditions of inner product

- Positivity

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \Rightarrow A^T = \begin{bmatrix} a & c \\ b & d \end{bmatrix}$$

$$A^T A = \begin{bmatrix} a^2 + c^2 & ab + cd \\ ab + cd & b^2 + d^2 \end{bmatrix}$$

$\Rightarrow tr(A^T A) = a^2 + c^2 + b^2 + d^2 \geq 0$ with $a, b, c, d \in \mathbb{R}$

(Because each term is a square of a number in $\mathbb{R}$)

$\boxed{\Rightarrow \text{ Positivity condition satisfied}}$

- Definiteness

$$tr(A^T A) = a^2 + b^2 + c^2 + d^2$$

$$tr(A^T A) = 0 \iff a^2 = b^2 = c^2 = d^2 = 0$$
$$\iff a = b = c = d = 0$$

$$\boxed{\Rightarrow \text{Definiteness condition satisfied}}$$

- Additivity

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \quad B = \begin{bmatrix} e & f \\ g & h \end{bmatrix}, \quad C = \begin{bmatrix} t & u \\ v & w \end{bmatrix}$$

•) $\langle A, C \rangle = tr(A^T C)$

$$A^T C = \begin{bmatrix} at + cv & au + cw \\ bt + dv & bu + dw \end{bmatrix}$$

$$\Rightarrow tr(A^T C) = at + cv + bu + dw$$

•) Similarly, we have:

$$\langle B, C \rangle = tr(B^T C) = et + gv + fu + hw$$

•) $A + B = \begin{bmatrix} a+e & b+f \\ c+g & d+h \end{bmatrix}$

$$\begin{aligned}
&\langle A+B, C \rangle \\
&= tr((A+B)^T C) \\
&= (a+e)t + (c+g)v + (b+f)u + (d+h)w \\
&= (at + cv + bu + dw) + (et + gv + fu + hw) \\
&= \langle A, C \rangle + \langle B, C \rangle
\end{aligned}$$

$$\Rightarrow \langle A+B, C \rangle = \langle A, C \rangle + \langle B, C \rangle$$

$\boxed{\Rightarrow \text{Additivity condition satisfied}}$

- Homogeneity

For $\alpha \in \mathbb{R}$:

$$\alpha A = \begin{bmatrix} \alpha a & \alpha b \\ \alpha c & \alpha d \end{bmatrix} \Rightarrow (\alpha A)^T = \begin{bmatrix} \alpha a & \alpha c \\ \alpha b & \alpha d \end{bmatrix}$$

$$\begin{aligned} \langle \alpha A, C \rangle &= tr\left((\alpha A)^T C\right) \\ &= \alpha at + \alpha cv + \alpha bu + \alpha dw \\ &= \alpha (at + cv + bu + dw) \\ &= \alpha \langle A, C \rangle \end{aligned}$$

$\Rightarrow$ Homogeneity condition satisfied

- Symmetry

$\cdot)$ $\langle A, C \rangle = at + cv + bu + dw$

$\cdot)$ $\langle C, A \rangle = tr(C^T A)$

$$C^T = \begin{bmatrix} t & v \\ u & w \end{bmatrix} \Rightarrow C^T A = \begin{bmatrix} at + cv & bt + dv \\ au + cw & bu + dw \end{bmatrix}$$

$$\Rightarrow tr(C^T A) = at + cv + bu + dw$$

$$\Rightarrow \langle C, A \rangle = \langle A, C \rangle$$

$\Rightarrow$ Symmetry condition satisfied

ii. $\langle \mathbf{A}, \mathbf{B} \rangle = \sum_{i=1}^{n} \sum_{j=1}^{n} [AB]_{ij}$

This operation is not an inner product in M_{2x2}. One counterexample is A = $\begin{bmatrix} 1 & -10 \\ 1 & -2 \end{bmatrix}$

$$\langle A, A \rangle = \sum_{i=1}^{n} \sum_{j=1}^{n} [AA]_{ij}$$

$$AA = \begin{bmatrix} 1 \times 1 + (-10) \times 1 & 1 \times (-10) + (-10) \times (-2) \\ 1 \times 1 + (-2) \times 1 & 1 \times (-10) + (-2) \times (-2) \end{bmatrix}$$

$$= \begin{bmatrix} -9 & 10 \\ -1 & -6 \end{bmatrix}$$

$$\Rightarrow \langle A, A \rangle = (-9) + 10 + (-1) + (-6)$$

$$= -6 < 0$$

$\Rightarrow$ Positivity condition is not satisfied

$\Rightarrow$ The operation is not an inner product in $M_{2 \times 2}$

(d) Optional Problem: Orthogonal Polynomials

d) Optional

i $\langle p, q \rangle = \int_0^1 p(x)\, q(x)\, dx$

→ Positivity : $\int_0^1 [p(x)]^2\, dx \geq 0 \quad \forall x \in R$

because with $[p(x)]^2$, all the region under the curve will be above the Ox line

⇒ The entire area under the curve of $[p(x)]^2$ is positive

→ Definiteness :

$\int_0^1 [p(x)]^2\, dx \geq 0 \quad \forall x \in R$

⇒ $\int_0^1 [p(x)]^2\, dx = 0$ only when $p(x) = 0$

Intuitively, if there is a line or a curve created by function $[p(x)]^2$ then the area cannot be 0

when there is no elimination of two opposite parts

•) Additivity:

$$\int_0^1 (p(x) + q(x)) f(x) dx = \int_0^1 (p(x) f(x) + q(x) f(x)) dx$$

$$= \int_0^1 p(x) f(x) dx + \int_0^1 q(x) f(x) dx$$

=> SATISFIED

•) Homogeneity:

$$\int_0^1 c \cdot q(x)\, p(x)\, dx = c \int_0^1 q(x)\, p(x)\, dx$$

(property of integrals)

=> SATISFIED

•) Symmetry:

$$\int_0^1 \underset{\downarrow \atop u}{p(x)}\ \underset{\downarrow \atop dv}{q(x)} = \int_0^1 \underset{\downarrow \atop dv}{q(x)}\ \underset{\downarrow \atop u}{p(x)}$$

=> Using integral by parts, our result will still be the same

=> SATISFIED

$\Rightarrow \int_0^1 p(x)\,q(x)\,dx$ is indeed an inner product on $P(n)$

ii) First, lets check the ~~to~~ norm of each standard basis vectors. They should be equal to 1

$$\int_0^1 \underset{u}{1} \cdot \underset{dv}{1} \; dx = x \Big|_0^1 - \int_0^1 0$$

$$= 1 - 0 - 0$$

$\Rightarrow du = 0$ $\quad = \boxed{1}$

$v = x$

$$\int_0^1 x \cdot x \, dx$$

$u = x \rightarrow du = 1$

$dv = x \rightarrow v = \frac{x^2}{2}$

$$\Rightarrow \int_0^1 x \cdot x \, dx = \frac{x^3}{2} \Big|_0^1 - \int_0^1 \frac{x^2}{2}$$

$$= \frac{1}{2} - 0 = \frac{x^3}{6} \Big|_0^1$$

$$= \frac{1}{2} - \frac{1}{6} = \frac{1}{3}$$

$\Rightarrow$ NOT SATISFIED

$\Rightarrow$ This standard basis is not an orthonormal basis

iii. Gram Schmidt :

① Pick the first vector $v_0 = 1$

This vector is already normalized (proven above)

$$\Rightarrow e_0 = 1$$

② get the second vector $v_1 = x$

$$\Rightarrow e_1 = v_1 - \langle v_1, e_0 \rangle e_0$$

$$= x - \int_0^1 x \, dx$$

$$= x - \frac{1}{2}$$

③ Normalize: $e_1 = \frac{e_1}{\|e_1\|} = \frac{x - \frac{1}{2}}{\left[\int_0^1 (x - \frac{1}{2})^2\right]^{0.5}}$

$$\int_0^1 (x - \frac{1}{2})^2 = \int_0^1 (x^2 - x + \frac{1}{4}) = \frac{1}{12}$$

$$\rightarrow e_1 = \sqrt{12}\,x - \frac{\sqrt{12}}{2} = \sqrt{12}\,x - \sqrt{3}$$

④ get the third vector $v_2 = x^2$

$$\Rightarrow e_2 = v_2 - \langle v_2, e_0 \rangle e_0 - \langle v_2, e_1 \rangle e_1$$

$$= x^2 - \int_0^1 x^2\,dx - \int_0^1 x^2(\sqrt{12}\,x - \sqrt{3})\,dx\,(\sqrt{12}x - \sqrt{3})$$

$$= x^2 - \frac{x^3}{3}\Big|_0^1 - (\sqrt{12}\,x - \sqrt{3})\left(\sqrt{12}\,\frac{x^4}{4} - \sqrt{3}\,\frac{x^3}{3}\right)\Big|_0^1$$

$$= x^2 - \frac{1}{3} - (\sqrt{12}\,x - \sqrt{3})\left(\frac{\sqrt{12}}{4} - \frac{\sqrt{3}}{3}\right)$$

$$= x^2 - \frac{1}{3} - \left(3x - 2x - \frac{3}{2} + 1\right)$$

$$= x^2 - x - \frac{1}{3} + \frac{3}{2} - 1$$

$$= x^2 - x + \frac{1}{6}$$

⑤ Normalize e_2.

$$e_2 = \frac{e_2}{\|e_2\|} = \frac{x^2 - x + \frac{1}{6}}{\sqrt{\int_0^1 (x^2 - x + \frac{1}{6})^2 dx}}$$

$$\int_0^1 (x^2 - x + \frac{1}{6})^2 = \int_0^1 (x^4 + x^2 + \frac{1}{36} - 2x^3 + \frac{1}{3}x^2 - \frac{1}{3}x)dx$$

$$= \frac{x^5}{5} + \frac{x^3}{3} + \frac{1}{36}x - \frac{x^4}{2} + \frac{x^3}{9} - \frac{x^2}{6} \Big|_0^1$$

$$= \frac{1}{5} + \frac{1}{3} + \frac{1}{36} - \frac{1}{2} + \frac{1}{9} - \frac{1}{6}$$

$$= \frac{1}{180}$$

$$\Rightarrow \quad e_2 = \sqrt{180}\,(x^2 - x + \frac{1}{6})$$

$$= \sqrt{5}\,(6x^2 - 6x + 1)$$

⑥ get the vector $v_3 = x^3$

$$\Rightarrow e_3 = v_3 - \langle v_3, e_0 \rangle e_0 - \langle v_3, e_1 \rangle e_1 - \langle v_3, e_2 \rangle e_2$$

$$\Rightarrow e_3 = x^3 - \int_0^1 x^3\,dx - \int_0^1 x^3(\sqrt{12}\,x - \sqrt{3})\,dx\,(\sqrt{12}\,x - \sqrt{3})$$

$$- \int_0^1 x^3(\sqrt{5}\,(6x^2 - 6x + 1))\,dx\ \sqrt{5}\,(6x^2 - 6x + 1)$$

$$= x^3 - \frac{x^4}{4}\Big|_0^1 - (\sqrt{12}\,x - \sqrt{3})\left(\sqrt{12}\,\frac{x^5}{5} - \sqrt{3}\,\frac{x^4}{4}\right)\Big|_0^1$$

$$- \sqrt{5}\,(6x^2 - 6x + 1)\left[6\sqrt{5}\,\frac{x^6}{6} - 6\sqrt{5}\,\frac{x^5}{5} + \sqrt{5}\,\frac{x^4}{4}\right]\Big|_0^1$$

$$x^3 - \frac{1}{4} - (\sqrt{12}\,x - \sqrt{3})\left(\frac{\sqrt{12}}{5} - \frac{\sqrt{3}}{4}\right)$$

$$- \sqrt{5}\,(6x^2 - 6x + 1)\left(\frac{\sqrt{5}}{\cancel{4}} - \frac{6\sqrt{5}}{5} + \frac{\sqrt{5}}{4}\right)$$

$$x^3 - \frac{1}{4} - \left(\frac{12}{5}x - \frac{3}{2}x - \frac{6}{5} + \frac{3}{4}\right)$$

$$- (6x^2 - 6x + 1)\left(\frac{\cancel{3}}{\cancel{4}}5 - 6 + \frac{5}{4}\right)$$

$$= x^3 - \frac{1}{4} - \frac{9}{10}x + \frac{9}{20} - \frac{1}{4}\left(6x^2 - 6x + 1\right)$$

$$= x^3 - \frac{3}{2}x^2 + \frac{3}{5}x - \frac{1}{20}$$

(7) Normalize e_3 =

$$e_3 = \frac{e_3}{\|e_3\|} = \frac{x^3 - \frac{3}{2}x^2 + \frac{3}{5}x - \frac{1}{20}}{\left[\int_0^1 \left(x^3 - \frac{3}{2}x^2 + \frac{3}{5}x - \frac{1}{20}\right)^2\right]^{0,5}}$$

In [24]

```
# CS113 OPTIONAL

integrate((x^3 - (3/2)*x^2 + (3/5)*x - 1/20)^2, (x, 0, 1))
```

Run Code

Out [24]

```
1/2800
```

$$\Rightarrow e_3 = \frac{x^3 - \frac{3}{2}x^2 + \frac{3}{5}x - \frac{1}{20}}{\sqrt{\frac{1}{2800}}}$$

$$\Rightarrow e_3 = \sqrt{7}(20x^3 - 30x^2 + 12x - 1)$$

$\Rightarrow$ Our standard basis $\{1;\ \sqrt{12}x - \sqrt{3};\ \sqrt{5}(6x^2 - 6x + 1);$

$\sqrt{7}(20x^3 - 30x^2 + 12x - 1)\}$

iv) $P_2(x) = \dfrac{2.2-1}{2} x P_1(x) - \dfrac{1}{2} P_0(x)$

$= \dfrac{3}{2} x^2 - \dfrac{1}{2}$

$P_3(x) = \dfrac{3.2-1}{3} x P_2(x) - \dfrac{2}{3} P_1(x)$

$= \dfrac{5}{3} \left(\dfrac{3}{2} x^2 - \dfrac{1}{2} \right) x - \dfrac{2}{3} x$

$= \dfrac{5}{2} x^3 - \dfrac{5}{6} x - \dfrac{2}{3} x = \dfrac{5}{2} x^3 - \dfrac{3}{2} x$

In [25]

```
p0 = 1
p1 = x
p2 = (3/2) * x^2 - 1/2
p3 = (5/2) * x^3 - 3/2 * x

# orthogonality: supposed to be equal to 0 all
integrate(p0*p1, (x, -1, 1)), integrate(p0*p2, (x, -1, 1)), integrate(p0*p3, (x, -1, 1)), integrate(p1*p2, (x, -1, 1)), integrate(p1*p3, (x, -1, 1)), integrate(p2*p3, (x, -1, 1))
```

Run Code

Out [25]

```
(0, 0, 0, 0, 0, 0)
```

We can see that orthogonality holds here! However, the normality of each individual is not satisfied yet. Therefore, we have to normalize our basis vectors to achieve an orthonormal basis.

```
# normalize
p0_norm = p0 / (integrate(p0*p0, (x, -1, 1)))**0.5
p1_norm = p1 / (integrate(p1*p1, (x, -1, 1)))**0.5
p2_norm = p2 / (integrate(p2*p2, (x, -1, 1)))**0.5
p3_norm = p3 / (integrate(p3*p3, (x, -1, 1)))**0.5
print(p0_norm)
print(p1_norm)
print(p2_norm)
print(p3_norm)
#verify if the norm is all 1
round(integrate(p0_norm*p0_norm, (x, -1, 1))), round(integrate(p1_norm*p1_norm, (x, -1, 1))), round(integrate(p2_norm*p2_norm, (x, -1, 1))), round(integrate(p3_norm*p3_norm, (x, -1, 1)))
```

Run Code

Out [26]

```
0.707106781186547
1.22474487139159*x
2.37170824512628*x^2 - 0.790569415042095
4.67707173346743*x^3 - 2.80624304008046*x

(1, 1, 1, 1)
```

Legendre's differential equations:

- With $n = 0$, $P0(x) = 1$:

$\rightarrow \frac{dP}{dx} = \frac{dP^2}{dx} = 0$

$\rightarrow L = 0$

- With $n = 1,\ P1(x) = x$:

$\rightarrow \frac{dP}{dx} = 1$ and $\frac{dP^2}{dx} = 0$

We plug it in the differential equations to derive the value for L

$-2x = Lx \Leftrightarrow L = -2$

- With $n = 2,\ P2(x) = \frac{3}{2}x^2 - \frac{1}{2}$:

$\rightarrow \frac{dP}{dx} = 3x$ and $\frac{dP^2}{dx} = 3$

$\rightarrow 3(1 - x^2) - 2x * 3x = L(\frac{3}{2}x^2 - \frac{1}{2})$

$\Leftrightarrow -9x^2 + 3 = L(\frac{3}{2}x^2 - \frac{1}{2})$

$\Leftrightarrow L = -6$

- With $n = 3,\ P3(x) = \frac{5}{2}x^3 - \frac{3}{2}x$

$\rightarrow \frac{dP}{dx} = \frac{15}{2}x^2 - \frac{3}{2}$ and $\frac{dP^2}{dx} = 15x$

$\rightarrow (1 - x^2)\, 15x - 2x(\frac{15}{2}x^2 - \frac{3}{2}) = L(\frac{5}{2}x^3 - \frac{3}{2}x)$

$\Leftrightarrow -30x^3 + 18x = L(\frac{5}{2}x^3 - \frac{3}{2}x)$

$\Leftrightarrow L = -12$

From 4 observations, we derive the patterns and concluded that the eigenvalue should be $-n(n + 1)$

Question 3: Animation Studios

(a) Describe why linear transformations cannot perform translations.

Linear transformation maps a vector in a domain R^n to a codomain R^m, which maps the zero vector to the zero vector. The zero vector will change from n components of 0 entries to m components of 0s. However, this is not the case for translation. Translation moves a vector up / down, right / left by a certain number of units and amount of angels, thus making the head and tail of the vector not necessarily remain unchanged. As a result, the translated zero vector will no longer remain a zero vector.
→ Linear transformation cannot perform translations

(b) Find the matrix A and the vector $\vec{b}$ that translates a vector one unit to the right.

Because this vector should only be moved to one unit to the right, no rescaling is needed. Therefore, the matrix A should keep the vector unchanged, and the vector $\vec{b}$ will be responsible for moving the vector to the right by 1 unit. To compute vector $\vec{b}$, we understand that x-coordinate will navigate the vector in the direction of left / right and y-coordinate has nothing to do here because it mainly navigates up / down.

In [3]

```
# As a result, our matrix A and the vector b will be:
A = matrix([[1, 0], [0, 1]])
b = vector([1, 0])
A, b
```

Run Code

Out [3]

```
(
[1 0]
[0 1], (1, 0)
)
```

- Testing with $\vec{v}$ (1, -2). The result should be the vector (2, -2)

In [4]
```
# As a result, our matrix A and the vector b will be:
A = matrix([[1, 0], [0, 1]])
b = vector([1, 0])
# translated vector
v = vector([1, -2])
A*v + b
```

Run Code

Out [4]
```
(2, -2)
```

(c) Combine transformation

Suppose our original vector is (x_0, y_0). When we rotate this vector by 45 degrees counterclockwise, we will get a new vector (x_1, y_1). The length of our vector stays the same, so we can use trigonometric functions with hypotenuse to construct the relationship between the two vectors.

Suppose our vector length is r and the original vector shapes an angle of $alpha$ with the horizontal axis

$\rightarrow\ x_0 = r * cos(alpha)$ and $y_0 = r * sin(alpha)$

We know that the rotated vector will shape an angle of $alpha + 45^0$ with the horizontal axis

$\rightarrow\ x_1 = r * cos(alpha + 45^0)$ and $y_0 = r * sin(alpha + 45^0)$

$\rightarrow\ x_1 = r * cos(alpha)\, cos(45^0) - r * sin(alpha)\, sin(45^0) = \frac{\sqrt{2}}{2} x_0 - \frac{\sqrt{2}}{2} y_0$

$\rightarrow y_1 = r * sin(alpha)\, cos(45^0) + r * cos(alpha)\, sin(45^0) = \frac{\sqrt{2}}{2} x_0 + \frac{\sqrt{2}}{2} y_0$

In [5]
```
# from the above analysis, our matrix A will be
A = matrix([[sqrt(2) / 2, - sqrt(2) / 2], [sqrt(2) / 2, sqrt(2) / 2 ]])
A
```

Run Code

Out [5]
```
[ 1/2*sqrt(2) -1/2*sqrt(2)]
[ 1/2*sqrt(2)  1/2*sqrt(2)]
```

The vector b will remain $< 1,\ 0 >$ to translate the vector to the right by 1 unit.

In [7]
```
# from the above analysis, our matrix A will be
A = matrix([[sqrt(2) / 2, - sqrt(2) / 2], [sqrt(2) / 2, sqrt(2) / 2 ]])
b = vector([1, 0])
v = vector([1, -2])
A*v + b
```

Run Code

Out [7]
```
(3/2*sqrt(2) + 1, -1/2*sqrt(2))
```

The new vector after affine transformation will be $< \frac{3\sqrt{2}}{2} + 1,\ \frac{-\sqrt{2}}{2} >$. This vector will be visualized in questions below.

(d) Order matters

Because we move the vector to the right by 1 unit first and then rotate it by 45 degrees, we will add $\vec{b}$ to $\vec{v}$ first and then left-multiply the result of this addition with $matrix\ A$.

$$\rightarrow the\ new\ vector\ =\ A * (\vec{v}\ +\ \vec{b})\ =\ A * \vec{v}\ +\ A * \vec{b}$$

We can see that the matrix of translation still remains A. However, the vector $\vec{b}$ now is going to be the product of matrix A and $\vec{b}$. This makes sense because $\vec{b}$ is a vector with 2 elements. Matrix A's dimension is 2x2. Multiplying A by a 2x1 matrix representing $\vec{b}$ will output a new 2x1 matrix that corresponds to a new 2-element vector.

In [11]

```
# from the above analysis, our matrix A will be
A = matrix([[sqrt(2) / 2, - sqrt(2) / 2], [sqrt(2) / 2, sqrt(2) / 2 ]])
b = vector([1, 0])
v = vector([1, -2])
rotate_first = A*v + b
move_first = A*v + A*b
# compare matrix and vector in the affine transformation of two orders
A, b, A*b
```

Run Code

Out [11]

```
(
[ 1/2*sqrt(2) -1/2*sqrt(2)]
[ 1/2*sqrt(2)  1/2*sqrt(2)], (1, 0), (1/2*sqrt(2), 1/2*sqrt(2))
)
```

$\rightarrow$ the new vector = $A\vec{v}\ +\ A\vec{b}$ with A = $\begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{-\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix}$, and $A\vec{b} = (\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})$

In [8]:

```
# the new order of affine transformation
A*(v + b), A*v + A*b
```

Run Code

Out [8]:

```
((2*sqrt(2), 0), (2*sqrt(2), 0))
```

The vector (1, -2) after the new affine transformation will be $(2\sqrt{2},\ 0)$

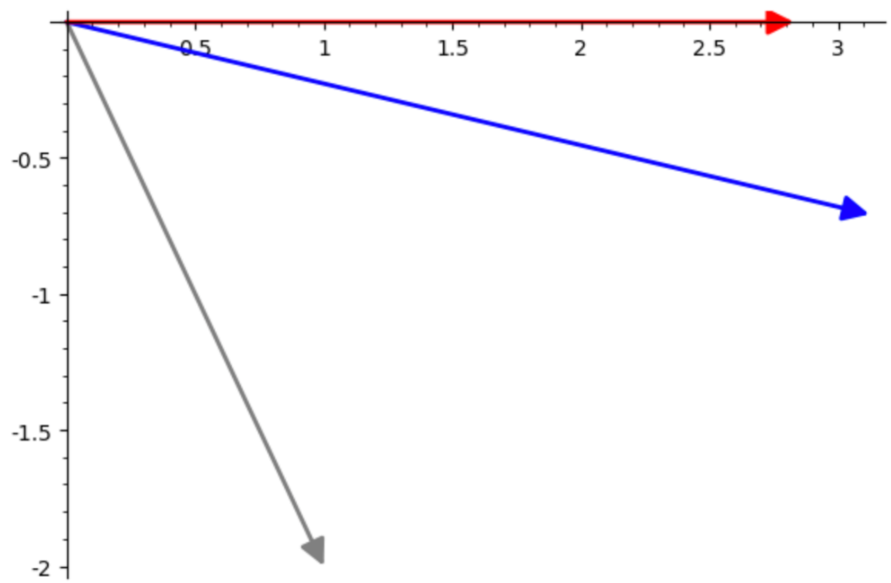
(e)

The two affine transformations of reverse order share the matrix of rotation A, but the $\vec{b}$ is different (the one on part c is $\vec{b}$, while the one on part d is $A\vec{b}$. We can also visualize the two transformed vectors and the original one to see the difference more clearly. Note that here we still stick things to the origin of the whole space which is $(0,\ 0)$ and only report the changed head of the vector.

```
# compare matrix and vector in the affine transformation of two orders
A, b, A*b
arrow((0, 0), v, color = 'gray') + arrow((0, 0), rotate_first, color = 'blue') + arrow((0, 0), move_first, color = 'red')
```

Run Code

Out [14]



(f)

Suppose that $\vec{v}$ has n components, then the new combined vector V will be an $(n + 1) \times 1$ matrix. Matrix A, for matrix multiplication to be possible, will be an $n \times n$ matrix, and $\vec{b}$ has n components. Therefore, the resulting matrix M will be an $(n + 1) \times (n + 1)$ matrix. When we multiply M with vector V to get the result after affine transformation, it will give out a vector with $n + 1$ elements (the product of an $(n + 1) \times (n + 1)$ matrix and an $(n + 1) \times 1$ matrix).

(g)

With $\vec{v}(1, -2)$, we have V = [1]

[-2]

[1]

- For part b, A = [1 0], and $\vec{b} = (1, 0)$

  [0 1]

  $\rightarrow$ M = [1 0 1]

  [0 1 0]

  [0 0 1]

- For part c,

```
In [15]  1 # resulting matrix M from A and b
         2 M = matrix([[sqrt(2)/2, - sqrt(2)/2, 1], [sqrt(2)/2, sqrt(2)/2, 0], [0, 0, 1]])
         3 M
```

Run Code

```
Out [15]
          [ 1/2*sqrt(2) -1/2*sqrt(2)           1]
          [ 1/2*sqrt(2)  1/2*sqrt(2)           0]
          [           0            0           1]
```

Now, we will multiply the matrix M with the combined vector V to verify if this trick works on affine transformation.

```
In [19]
# resulting matrix M from A and b
M = matrix([[sqrt(2)/2, - sqrt(2)/2, 1], [sqrt(2)/2, sqrt(2)/2, 0], [0, 0, 1]])
M
# verify
v = vector([1, -2])
V = vector([1, -2, 1])
M*V, A*v + b
```

Run Code

```
Out [19]
((3/2*sqrt(2) + 1, -1/2*sqrt(2), 1), (3/2*sqrt(2) + 1, -1/2*sqrt(2)))
```

It turns out that $n - 1$ first components of the product of matrix M and vector V will be exactly the translated vector when we apply affine transformation on it originally.

- For part d,

```
In [23]
# resulting matrix M from A and A*b
M = matrix([[sqrt(2)/2, - sqrt(2)/2, sqrt(2)/2], [sqrt(2)/2, sqrt(2)/2, sqrt(2)/2], [0, 0, 1]])
M
# verify
v = vector([1, -2])
V = vector([1, -2, 1])
M*V, A*v + A*b

```

Run Code

```
Out [23]
((2*sqrt(2), 0, 1), (2*sqrt(2), 0))
```

We can confirm that this resulting matrix is really helpful in giving the accurate result with only one step of matrix multiplication.

Question 4: A Valuable Factor

(a) Consider the 2*3 matrix A = [2 0 1]

[3 1 -6]

i. Show that AA^T and A^TA are both symmetric matrices.

A symmetric matrix is a square matrix that is equal to its transpose. From the code below, we can see that AA^T and A^TA are square matrices that remain the same after transposing. Therefore, AA^T and A^TA are both symmetric matrices.

```
In [1]: A = matrix([[2,0,1], [3,1,-6]])
        At = A.transpose()
        matrix_1 = A*At
        matrix_2 = At*A

        matrix_1, matrix_1.transpose(), matrix_2, matrix_2.transpose()

Out[1]: (
                                   [ 13   3 -16]  [ 13   3 -16]
        [ 5  0]  [ 5  0]  [  3   1  -6]  [  3   1  -6]
        [ 0 46], [ 0 46], [-16  -6  37], [-16  -6  37]
        )
```

ii. (Optional) With mxn matrices, show that MM^T and M^TM are both symmetric matrices

Here, we need to make use of a property $(AB)^T = B^TA^T$.

By definition of matrix multiplication, $(AB)_{ab} = \sum_{k=1}^{n} a_{ak} b_{kb} \rightarrow (AB)^T_{ab} = (AB)_{ba} = \sum_{k=1}^{n} a_{bk} b_{ka}$

$(B^T A^T)_{ab} = \sum_{k=1}^{n} (B^T)_{ak} (A^T)_{kb} = \sum_{k=1}^{n} (B_{ka} A_{bk}) = \sum_{k=1}^{n} b_{ka} a_{bk}$

→ We proved that $(AB)_{ab} = (B^T A^T)_{ab}$

Here, we have $(MM^T)^T = (M^T)^T * M^T = MM^T \rightarrow MM^T$ is a symmetric matrix!

Likewise, $(M^T M)^T = M^T * (M^T)^T = M^T M \rightarrow M^T M$ is also a symmetric matrix!

iii. **By the spectral theorem, we can find orthogonal matrices U and V such that $AA^T = U*D_1*U^T$ and $A^TA = V*D_2*V^T$. Find U, V, D_1, and D_2. How are D_1 and D_2 related?**

- Finding U and D_1
  - To find D_1, we find the eigenvalues of AA^T. The eigenvalues are the diagonal entries of D_1. All other entries outside the main diagonal are zero because D_1 is a diagonal matrix.
  - To find U, we make the eigenvectors of AA^T become the column vectors of the matrix U. The order of the eigenvectors is the same as the order of the corresponding eigenvalues in D_1. Since U is an orthogonal matrix, we also need to normalize the eigenvectors to finalize U.

```
#Get the eigenvectors of the matrix
matrix_1.eigenvectors_right() # this is the A*At matrix
```

```
[(46,
  [
  (0, 1)
  ],
  1),
 (5,
  [
  (1, 0)
  ],
  1)]
```

According to Sage, AA^T has 2 eigenvalues: 46 and 5 → D = [46 0]

[0 5]

The corresponding eigenvectors are (0, 1) and (1, 0). These eigenvectors are already orthonormal

→ D_1=[0 1]

[1 0]

- To confirm that U and D_1 are correct, we check the product $U*D_1*U^T$ to see if it equals AA^T

```
U = matrix([[0,1],[1,0]])
```

```
#the diagonals are formed by the eigenvalues of the matrix
D1 = matrix([[46,0], [0,5]])

check = U*D1*U.transpose()
#checks if the matrix U*the diagonal* the transpose of U yield the same result as the original matrix
print(matrix_1 == check)
D1,check, matrix_1
```

```
True

(
[46  0]  [ 5  0]  [ 5  0]
[ 0  5], [ 0 46], [ 0 46]
)
```

- Finding V and D_2:
  - Similarly, D_2 has the diagonal entries being the eigenvalues of A^TA, and V has column vectors being the corresponding normalized eigenvectors.

```
matrix_2.eigenvectors_right() # this is the At*A matrix
```

```
[(46,
  [
  (1, 1/3, -2)
  ],
  1),
 (5,
  [
  (1, 0, 1/2)
  ],
  1),
 (0,
  [
  (1, -15, -2)
  ],
  1)]
```

```
v1 = vector([1,1/3,-2])
v2 = vector([1,0,1/2])
v3 = vector([1,-15,-2])

#normalize the vectors to form an orthonormal basis
norm_v1 =v1.normalized()
norm_v2 =v2.normalized()
norm_v3 =v3.normalized()

#the eigenvectors form the column vectors of matrix V
V = matrix([norm_v1, norm_v2, norm_v3]).transpose()
V
```

```
[   3/46*sqrt(46)      2/5*sqrt(5)  1/230*sqrt(230)]
[   1/46*sqrt(46)                0  -3/46*sqrt(230)]
[  -3/23*sqrt(46)      1/5*sqrt(5) -1/115*sqrt(230)]
```

$\rightarrow V = [\frac{3\sqrt{46}}{46} \quad \frac{2\sqrt{5}}{5} \quad \frac{1\sqrt{230}}{230}]$

$[\frac{1\sqrt{46}}{46} \quad 0 \quad \frac{-3\sqrt{230}}{46}]$

$[\frac{-3\sqrt{46}}{23} \quad \frac{1\sqrt{5}}{5} \quad \frac{-1\sqrt{230}}{115}]$

$D_2 =$ [46 0 0]

[0 5 0]

[0 0 0]

- Check if $A^TA = V*D_2*V^T$

```
D2 = matrix([[46,0,0],[0,5,0],[0,0,0]]) #the eigenvalues are form the diagonal of the matrix
check = V*D2*V.transpose()
matrix_2,V, D2

print(check==matrix_2)
check,matrix_2, D2
```

```
True

(
[ 13   3 -16]  [ 13   3 -16]  [46  0  0]
[  3   1  -6]  [  3   1  -6]  [ 0  5  0]
[-16  -6  37], [-16  -6  37], [ 0  0  0]
)
```

- How are D_1 and D_2 related?

The similarity between D_1 and D_2 is that they share the eigenvalues 46 and 5. The main difference is that D_2 is a 3x3 matrix whereas D_1 is a 2x2 matrix. D_2's third eigenvalue is zero, making the last row of D_2 all zeros.

v. **This suggests an approach. Try factoring with U and V instead. Compute $\Sigma = U^TAV$. What form does Σ take? How is Σ related to D_1 and D_2 above?**

- Using Sage to compute sigma

```
sigma = U.transpose()*A*V
sigma
```

```
[sqrt(46)        0        0]
[       0  sqrt(5)        0]
```

- From the Sage result, sigma has the same number of rows as D_1 and the same number of columns as D_2. The entries on the main diagonal are the square roots of the non-zero eigenvalues that were used to construct D_1 and D_2.

(b) Find singular value decompositions for matrices B, C and D.

- To find the singular value decomposition of any m*n matrix (Y), we will go through the following steps:
  1. Calculate the transpose of matrix Y.
  2. Compute the product YY^T and Y^TY.

3. Calculate the eigenvalues and eigenvectors of YY^T and Y^TY. For each of the matrices, if the eigenvectors are not normal (magnitude != 1), normalize them to form an orthogonal matrix.
4. The normalized eigenvectors of the YY^T will form the column vectors of matrix U, and the eigenvectors of Y^TY will form the column vectors of matrix V.
5. To find the value of sigma, we compute the product $\Sigma = U^T*Y*V$
6. Sigma is the matrix used to form the singular value decomposition. $Y = U*\Sigma*V^T$

- **Matrix B = [1 -4 2]**

```
#step 1
B = matrix([1,-4,2])
Bt = B.transpose()
#step 2
B_Bt = B*Bt
Bt_B = Bt*B
B_Bt, Bt_B
# step 3 part A
B_Bt.eigenvectors_right()
```

```
[(21,
  [
  (1)
  ],
  1)]
```

```
u1 =vector([1])
U =matrix([u1]).transpose()
#step3 partB
Bt_B.eigenvectors_right()
```

```
[(21,
  [
  (1, -4, 2)
  ],
  1),
 (0,
  [
  (1, 0, -1/2),
  (0, 1, 2)
  ],
  2)]
```

```
v1= vector([1,-4,2])
v2 = vector([1,0,-1/2])
v3 = vector([0,1,2])
#normalize the vectors to form orthornormal eigenbasis
v1_norm = v1.normalized()
v2_norm = v2.normalized()
v3_norm = v3.normalized()

V = matrix([v1_norm, v2_norm, v3_norm]).transpose()
```

```
#calculation to find sigma
sigma = U.transpose()*B*V
print(U), print(sigma), print(V.transpose())
#confirmation check
B==U*sigma*V.transpose()
```

```
[1]
[sqrt(21)         0         0]
[ 1/21*sqrt(21) -4/21*sqrt(21)  2/21*sqrt(21)]
[    2/5*sqrt(5)              0    -1/5*sqrt(5)]
[              0     1/5*sqrt(5)     2/5*sqrt(5)]

True
```

- **Matrix C = [0 2]**

  [-1 0]

```
#step 1
C= matrix([[0,2],[-1,0]])
Ct = C.transpose()
#step 2
C_Ct = C*Ct
Ct_C = Ct*C
#step 3
print(C_Ct.eigenvectors_right())
```

```
[(4, [
(1, 0)
], 1), (1, [
(0, 1)
], 1)]
```

```
u1 = vector([1,0])
u2 = vector([0,1])
U = matrix([u1,u2]).transpose()
print(Ct_C.eigenvectors_right())
```

```
[(4, [
(0, 1)
], 1), (1, [
(1, 0)
], 1)]
```

```
#column vectors of V
v1 = vector([0,1])
v2 = vector([1,0])

V = matrix([v1, v2]).transpose()
```

```
#calculation to find sigma
sigma = U.transpose()*C*V
#singular value decomposition
print(U),print(sigma), print(V.transpose())

#confirmation check
C==U*sigma*V.transpose()
```

```
[1 0]
[0 1]
[ 2  0]
[ 0 -1]
[0 1]
[1 0]

True
```

- **Matrix D = [-4 1]**

  [-2 -4]

  [0 -2]

  [2 -2]

```
D = matrix([[-4,1],[-2,-4],[0,-2],[2,-2]])
Dt = D.transpose()
D_Dt = D*Dt
Dt_D = Dt*D
D_Dt.eigenvectors_right()
```

```
[(25,
  [
  (1, -4, -2, -2)
  ],
  1),
 (24,
  [
  (1, 1/2, 0, -1/2)
  ],
  1),
 (0,
  [
  (1, 0, -3/2, 2),
  (0, 1, -3, 1)
  ],
  2)]
```

```
u1 = vector([1,-4,-2,-2])
u2 = vector([1,1/2,0,-1/2])
u3 = vector([1,0,-3/2,2])
u4 = vector([0,1,-3,1])

u1_norm = u1.normalized()
u2_norm = u2.normalized()
u3_norm = u3.normalized()
u4_norm = u4.normalized()

U = matrix([u1_norm, u2_norm, u3_norm, u4_norm]).transpose()
```

```
Dt_D.eigenvectors_right()
```

```
[(25,
  [
  (0, 1)
  ],
  1),
 (24,
  [
  (1, 0)
  ],
  1)]
```

```
v1 = vector([0,1])
v2 = vector([1,0])
V = matrix([v1,v2]).transpose()
```

```
#finding sigma
sigma = U.transpose()*D*V
#singular value decomposition
print(U), print(sigma), print(V.transpose())
#confirmation
D== U*sigma*V.transpose()
```

```
[            1/5  2/3*sqrt(3/2)  2/29*sqrt(29)              0]
[           -4/5  1/3*sqrt(3/2)              0  1/11*sqrt(11)]
[           -2/5              0 -3/29*sqrt(29) -3/11*sqrt(11)]
[           -2/5 -1/3*sqrt(3/2)  4/29*sqrt(29)  1/11*sqrt(11)]
[            5             0]
[            0 -4*sqrt(3/2)]
[            0             0]
[            0             0]
[0 1]
[1 0]

True
```

Reflection

Problem-solving HCs

#algorithms:

We have applied this HC so well multiple times in this assignment. The typical application can be found in our pseudocode to decode our message in Question 1. Also, in Question 1 where we have to find which appropriate standard basis vector should be added, we chose to use a loop to effectively compute the results. This strategy is good because we can see all possible options at the same time. Furthermore, in Question 2, specifically the optional one, where we have to find the patterns in eigenvalues for the Legendre's differential equation, we also try values from 0 to 3 step-by-step to construct the relationship between the eigenvalues and the power of the function. Also in this Question, #algorithms help us in applying the Gram-Schmidt method as we have to conduct complicated calculations to construct the orthonormal basis. In question 4, we came up with a step-by-step algorithm that shows how to calculate the singular value decomposition of any given matrix. The algorithm is clear and concise and handles all types of matrices. The algorithm also terminates in a finite number of steps (6 steps in total). This algorithm was used to calculate the singular value decompositions for the matrices in 4b.

#breakitdown

In Question 2c, we broke down the original problem into five individual sub-problems whose answers came together to form the final answer. In order to check if the operation is an inner product of the matrix, we performed separate checks for each of the conditions:

Positivity, Definiteness, Additivity, Homogeneity, and Symmetry. Breaking down this problem in this way is the most efficient way to arrive at the final answer. Furthermore, the whole process of Gram-Schmidt algorithm in the optional problem in Question 2 requires so much care. Therefore, #breakitdown really helps when we choose to deal with subproblems meticulously with each value of n and then after finishing the process overall, we believe that no mistakes were made along the way. We also stay careful about each step in affine transformation in Question 3. Because the order really matters, we really want to ensure each step in each procedure. If each step is correct, then the whole problem will be correct. One approach that helps is to visualize the vector at all stages. For example, how it looks at the beginning, after being rotated by 45 degrees and then moved to the right and the reverse process. This really helps in clarifying our understanding and verifying the accuracy of our work for other groupmates.

Group-work HCs

#composition

Throughout the paper, we utilized clear, concise language, with useful diagrams and symbols that help the reader understand the paper easily. The calculations are written in a homogenous style that makes them easy to follow. Each of us have different writing styles but we worked together to homogenize the work and adopted a similar style that allowed us to create the smooth flow of the paper. We each succinctly explained our calculations, making it easy to proof-read each other’s work. This really helps because this assignment is a very long one, and we also try to be careful about everything to not make mistakes.

#differences

We evaluated each of the group members' strengths and assigned the problems accordingly. We checked in regularly to ensure that everyone was progressing well on their assigned tasks. The regular check-ins allowed us to discuss the problems that were difficult for some group members. With that strategy, we were able to accomplish our tasks effectively. William's technical skills were useful in proofreading the proofs and attempting the optional questions. Loan's editing and formatting skills allowed us to create a well-formatted paper. She was also very careful in ensuring that all of our calculations were correct, which really helped in such a long assignment like this. Waiyego was familiar with computational tools and visualizing vectors. That really helped when some problems required a lot of coding to solve more efficiently. Overall, the blend of people's strengths in our group really creates such a wonderful collaboration.