|| 4

SCHOOL OF ARTIFICIAL INTELLIGENCE

24MA602 Computational Mathematics for Data Science

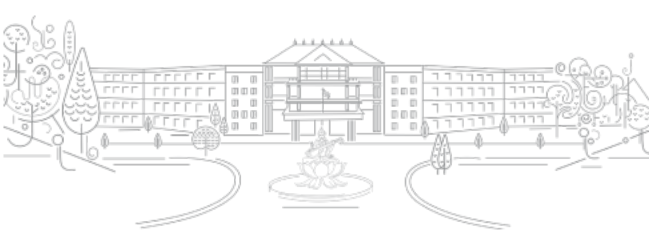
Assignment Set-4

Submitted by

Siju K S
Reg. No. CB.AI.R4CEN24003
Center for Computational Engineering & Networking

Submitted to

Prof. (Dr.) Soman K.P.
Professor & Dean
School of Artificial Intelligence
Amrita Vishwa Vidyapeetham



SEPTEMBER
2024

Contents

List of Tables

List of Figures

1 | Assignment 33 Eigen values of Anti-diagonal Matrices

1.1 Definition and Properties

Do thought/computational experiments and find patterns. The generic inference obtained from such experiments reduces time required for answering questions in examinations.

It is demonstrated through finding Eigenvalues and eigenvectors of anti-diagonal matrices.

1.1.1 Definition

Let A be a square matrix . An entry in A is an *anti-diagonal entry* if it is on the line going from the lower left corner of A to the upper right corner. If all entries in A are zero except on the anti-diagonal, then A is an *anti-diagonal matrix.*

$$A=\begin{bmatrix} 0 & 0 & 0 & a \\ 0 & 0 & b & 0 \\ 0 & c & 0 & 0 \\ d & 0 & 0 & 0 \end{bmatrix}$$

1.1.2 Properties

Properties of anti-diagonal matrices

If A and D are n×n anti-diagonal and diagonal matrices respectively, then AD, DA are anti-diagonal.

The product of two anti-diagonal matrices is a diagonal matrix.

For example

$$A*A\begin{bmatrix} 0 & 0 & 0 & a \\ 0 & 0 & b & 0 \\ 0 & c & 0 & 0 \\ d & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} 0 & 0 & 0 & a \\ 0 & 0 & b & 0 \\ 0 & c & 0 & 0 \\ d & 0 & 0 & 0 \end{bmatrix}=\begin{bmatrix} \mathrm{ad} & 0 & 0 & 0 \\ 0 & \mathrm{bc} & 0 & 0 \\ 0 & 0 & \mathrm{bc} & 0 \\ 0 & 0 & 0 & \mathrm{ad} \end{bmatrix}$$

1.1.3 Examples

Example 1. Creating 3 × 3 anti-diagonal matrix and finding eigen values and vectors.

```
>> A= fliplr(diag([1 2 3])) % or fliplr(diag(1:3))
```

A= $\begin{bmatrix} 0 & 0 & 1 \\ 0 & 2 & 0 \\ 3 & 0 & 0 \end{bmatrix}$

Through computational experiments by hand or using `matlab` we can find that, if matrix size is odd, then the central number is an eigenvalue. So here 2 is an

eigenvalue. Corresponding eigen vector is $\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$(or its scalar multiple)

Multiply (pairwise) the numbers around middle number and find square roots. Here multiplications (1 and 3) give 3 and its roots are. These are the remaining eigenvalues.

Corresponding eigen vectors can be computed as follows

EV$(\sqrt{3})$is the solution of $\begin{bmatrix} 0-\sqrt{3} & 0 & 1 \\ 0 & 2-\sqrt{3} & 0 \\ 3 & 0 & 0-\sqrt{3} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

Note,first and third row are dependent,So consider rows 1 and 2

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \perp \begin{bmatrix} 0-\sqrt{3} \\ 0 \\ 1 \end{bmatrix}; \text{also} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \perp \begin{bmatrix} 0 \\ 2-\sqrt{3} \\ 0 \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \sqrt{3} \end{bmatrix}$$

Note. It is created from $\begin{bmatrix} 0-\sqrt{3} \\ 0 \\ 1 \end{bmatrix}$. Interchange non-zero elements and change sign of one of the elements interchanged. (vertically flip and change one sign)

Also

EV$(\sqrt{3})$is the solution of$\begin{bmatrix} 0+\sqrt{3} & 0 & 1 \\ 0 & 2+\sqrt{3} & 0 \\ 3 & 0 & 0+\sqrt{3} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

Note,first and third row are dependent.So consider row 1 and 2

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \perp \begin{bmatrix} 0+\sqrt{3} \\ 0 \\ 1 \end{bmatrix}; \text{also} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \perp \begin{bmatrix} 0 \\ 0+\sqrt{3} \\ 0 \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ \sqrt{3} \end{bmatrix}$$

So,the set of Eigenvectors corresponding to eigen value 2,$\sqrt{3}$,-$\sqrt{3}$are

$$\left\{ \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ \sqrt{3} \end{bmatrix} \begin{bmatrix} -1 \\ 0 \\ \sqrt{3} \end{bmatrix} \right\}$$

Example 2. 5×5 anti-diagonal matrix and its eigen values

A= $\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 \end{bmatrix}$

Eigen values are 3,$\sqrt{4*2}$,-$\sqrt{4*2}$,$\sqrt{5*1}$,$\sqrt{5*1}$

Eigen vectors are

$$\left\{\begin{bmatrix}0\\0\\1\\0\\0\end{bmatrix},\begin{bmatrix}0\\2\\0\\\sqrt{8}\\0\end{bmatrix}\begin{bmatrix}0\\-2\\0\\\sqrt{8}\\0\end{bmatrix}\begin{bmatrix}1\\0\\0\\0\\\sqrt{5}\end{bmatrix}\begin{bmatrix}-1\\0\\0\\0\\\sqrt{5}\end{bmatrix}\right\}$$

Do you see anywhere a staircase pattern?

We can make one more change in the first eigenvector value so that you can create this matrix just by looking at values in upper part of A and eigen values. Take eigen vector set as :

$$\left\{\begin{bmatrix}0\\0\\3\\0\\0\end{bmatrix},\begin{bmatrix}0\\2\\0\\\sqrt{8}\\0\end{bmatrix}\begin{bmatrix}0\\-2\\0\\\sqrt{8}\\0\end{bmatrix}\begin{bmatrix}1\\0\\0\\0\\\sqrt{5}\end{bmatrix}\begin{bmatrix}-1\\0\\0\\0\\\sqrt{5}\end{bmatrix}\right\}$$

Example 3. 4×4 anti-diagonal matrix and its eigen values.

$$A=\begin{bmatrix}0&0&0&1\\0&0&2&0\\0&3&0&0\\4&0&0&0\end{bmatrix}$$

Eigenvalues are$\sqrt{2*3}$,-$\sqrt{2*3}$,$\sqrt{1*4}$,$-\sqrt{1*4}$=$\sqrt{6}$,-$\sqrt{6}$,2,-2

$$\left\{\begin{bmatrix}0\\2\\\sqrt{6}\\0\end{bmatrix},\begin{bmatrix}0\\-2\\\sqrt{6}\\0\end{bmatrix},\begin{bmatrix}1\\0\\0\\2\end{bmatrix},\begin{bmatrix}-1\\0\\0\\2\end{bmatrix}\right\}$$

1.2 Tasks

Assignment

1. Write one significant fact about the eigen values of each of the following.
   (a) A real symmetric matrix
   (b) A stable matrix ; all solutions to du/dt = Au approach zero.
   (c) An orthogonal matrix
   (d) A Markov matrix (stochastic matrix)
   (e) A defective matrix (nonzero eigenvalue repetition)

(f) A singular matrix

SOLUTION

(a) **Real Symmetric Matrix**:

- All eigenvalues of a real symmetric matrix are **real**.
- **Example**: Consider the matrix

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

The eigenvalues are 3 and 1, both real.

(b) **Stable Matrix** (solutions to $\frac{du}{dt} = Au$ approach zero):

- A matrix is stable if and only if all of its eigenvalues have **negative real parts**.
- **Example**: Consider the matrix

$$A = \begin{pmatrix} -1 & 0 \\ 0 & -2 \end{pmatrix}$$

The eigenvalues are -1 and -2, both negative, indicating stability.

(c) **Orthogonal Matrix**:

- The eigenvalues of an orthogonal matrix lie on the **unit circle**, meaning they have an absolute value (magnitude) of 1.
- **Example**: Consider the matrix

$$Q = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

The eigenvalues are i and $-i$, which lie on the unit circle in the complex plane.

(d) **Markov Matrix** (Stochastic Matrix):

- A Markov matrix has at least one eigenvalue equal to **1**, as the rows sum to 1 and represent probability distributions.
- **Example**: Consider the matrix

$$P = \begin{pmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{pmatrix}$$

The eigenvalues are 1 and 0.3, where 1 corresponds to the steady-state probability vector.

(e) **Defective Matrix** (nonzero eigenvalue repetition):

- A defective matrix has **fewer linearly independent eigenvectors** than its repeated eigenvalues, meaning it cannot be fully diagonalized.
- **Example**: Consider the matrix

$$A = \begin{pmatrix} 4 & 1 \\ 0 & 4 \end{pmatrix}$$

The eigenvalue 4 is repeated, but there is only one linearly independent eigenvector, making the matrix defective.

(f) **Singular Matrix**:

- A singular matrix has at least one eigenvalue equal to **0**, which is why it is non-invertible.
- **Example**: Consider the matrix

$$A = \begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix}$$

The eigenvalues are 0 and 5, indicating that the matrix is singular.

2. Compute Eigenvalues of

$$p1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \text{and } p2 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}.$$

SOLUTION

Matlab code and output for this task are given below.

```
% eigen values of p1
p1 = [0 1 0; 0 0 1; 1 0 0];
eigenvalues = eig(p1);
disp('The eigenvalues of matrix p1 are:');
```

```
The eigenvalues of matrix p1 are:
```

```
disp(eigenvalues);
```

```
  -0.5000 + 0.8660i
  -0.5000 - 0.8660i
   1.0000 + 0.0000i
```

```
% eigen values of p2
p2 = [0 1 0; 0 0 1; 1 0 0];
eigenvalues = eig(p2);
disp('The eigenvalues of matrix p2 are:');
```

```
The eigenvalues of matrix p2 are:
```

```
disp(eigenvalues);
```

```
  -0.5000 + 0.8660i
  -0.5000 - 0.8660i
   1.0000 + 0.0000i
```

3. Find the eigenvalues and eigen vectors of

$$A = \begin{bmatrix} 3 & 4 & 2 \\ 0 & 1 & 2 \\ 0 & 0 & 0 \end{bmatrix} \text{and } B = \begin{bmatrix} 0 & 0 & 2 \\ 0 & 2 & 0 \\ 2 & 0 & 0 \end{bmatrix}.$$

SOLUTION

Matlab code and output for this task are given below.

```
% eigen values of A
A = [3 4 2; 0 1 2; 0 0 0];
eigenvalues = eig(A);
disp('The eigenvalues of matrix A are:');
```

```
The eigenvalues of matrix A are:
```

```
disp(eigenvalues);
```

```
     3
     1
     0
```

```
% eigen values of B
B = [0 0 2; 0 2 0; 2 0 0];
eigenvalues = eig(B);
disp('The eigenvalues of matrix B are:');
```

```
The eigenvalues of matrix B are:
```

```
disp(eigenvalues);
```

```
    -2
     2
     2
```

RESULTS

1. Definition and basic properties of anti-diagonal matrices are reviewed.
2. Eigen values and eigen vectors of given anti-diagonal matrices are computationally found.

2 | Assignment 34 Connecting Geometry with Linear Algebra

2.1 Geometry to Algebra

Some 'plane' thoughts about plane passing through origin in 3Dimension

Summary

Many geometrical problems can be solved quickly if we express and visualize the problem in linear algebra perspective. This writeup and assignment enable us solve plane related problems in 3D space. We connect equation of a plane to null space of a matrix and its normal vector. Required quantities can be obtained in split second if we practice.

Equation of a 3D plane though **origin** is given by

From linear algebra point of view, it is a set of points (x, y, z) satisfying

$(a, b, c)\begin{bmatrix} x \\ y \\ z \end{bmatrix}$=0.It is of the form Ax = 0; $x = \left(x, y, z\right)^T$. It is set of points that defines the null space of matrix A. Here, A is a row vector. Any row vector is orthogonal to all vectors in (right) null space.

Hence Vector $\begin{bmatrix} a \\ b \\ c \end{bmatrix}$ is a normal to the plane given by $\mathrm{ax} + \mathrm{by} + c = 0$

2.1.1 Connection with points on the plane to null space

Dimension of (right) null space of our A is 2. So, it is spanned by 2 independent vectors. These 2 vectors are orthogonal to row vector in A.

How to get these basis vectors for null space.

Gauss procedure for solving Ax=**0** come to our rescue.

Here is it. We should be able to do in a split second in examination hall.

Let us take an example. Let $2x + 3y + 4z = 0$ be the plane.

As per 'Gauss Terminology' x is our pivot variable and others are free variables.

$$\underbrace{2\ \overset{}{x}}_{\text{pivot}} + \underbrace{3\ y}_{\text{free}} + \underbrace{4\ y}_{\text{free}} = 0$$

We express all variables in terms of free variables and immediately get basis set for null space . It is as follows

$x = -\frac{3}{2}$y + -2z

y = 1y + 0z $\Rightarrow \begin{bmatrix} x \\ y \\ z \end{bmatrix} =$ y $\begin{bmatrix} -\frac{3}{2} \\ 1 \\ 0 \end{bmatrix}$ + z $\begin{bmatrix} -2 \\ 0 \\ 1 \end{bmatrix}$

z = 0y + 1z

The set of vectors $\left\{ \begin{bmatrix} -\frac{3}{2} \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -2 \\ 0 \\ 1 \end{bmatrix} \right\}$span right null space .

Any point (x,y,z) in the plane is a linear combination of these two vectors.

To check , add these two vectors to get (x,y,z) and substitute in $2x + 3y + 4z$.

The result will be zero.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -\frac{3}{2} \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} -2 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -\frac{7}{2} \\ 1 \\ 1 \end{bmatrix}$$; and

$2 \text{ x } \underbrace{(-7/2)}_{x} + 3\text{x } \underbrace{(1)}_{y} + 4 \underbrace{(1)}_{z} = 0$

This process and the concept of **Gram-Schmidt** orthogonalization process allows you to tackle many geometric problems. Some time you may require the help of projection matrices.

Assignment Questions

1.1 Find a basis for the subspace **S** in $\mathbb{R}^4$ spanned by all solution of

$$x1 + x2 + x3 - x4 = 0$$

SOLUTION

Basis for the Subspace $S \subset \mathbb{R}^4$

Consider the linear equation:

$$x_1 + x_2 + x_3 - x_4 = 0.$$

This can be expressed in matrix form as:

$$\begin{pmatrix} 1 & 1 & 1 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = 0.$$

The corresponding augmented matrix is:

$$\left(\begin{array}{cccc|c} 1 & 1 & 1 & -1 & 0 \end{array} \right).$$

Using Gaussian elimination, we find that the matrix is already in row echelon form. We solve for x_1:

$$x_1 = -x_2 - x_3 + x_4.$$

Thus, the general solution can be expressed as:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = x_2 \begin{pmatrix} -1 \\ 1 \\ 0 \\ 0 \end{pmatrix} + x_3 \begin{pmatrix} -1 \\ 0 \\ 1 \\ 0 \end{pmatrix} + x_4 \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

Therefore, a basis for the subspace S is given by the set:

$$\left\{ \begin{pmatrix} -1 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \right\}.$$

These vectors are linearly independent and span the solution space of the equation. `Matlab` code and output for this task is given below.

```
A = [1 1 1 -1];
null_space = null(A, 'r');
disp('The basis for the subspace S is:');
```

```
The basis for the subspace S is:
```

```
disp(null_space);
```

```
    -1          -1           1
     1           0           0
     0           1           0
     0           0           1
```

1.2 Find a basis for the orthogonal complement $S^{\perp}$.

SOLUTION

Finding the Basis for the Orthogonal Complement $S^\perp$

Given the linear equation that defines the subspace S:

$$x_1 + x_2 + x_3 - x_4 = 0,$$

we identify the normal vector $\mathbf{n}$ associated with this equation:

$$\mathbf{n} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix}.$$

The orthogonal complement $S^\perp$ consists of all vectors $\mathbf{v} \in \mathbb{R}^4$ such that:

$$\mathbf{n} \cdot \mathbf{v} = 0.$$

This gives the condition:

$$1 \cdot x_1 + 1 \cdot x_2 + 1 \cdot x_3 - 1 \cdot x_4 = 0,$$

which simplifies to:

$$x_1 + x_2 + x_3 - x_4 = 0.$$

Rearranging this, we find:

$$x_4 = x_1 + x_2 + x_3.$$

We can express the general solution as:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_1 + x_2 + x_3 \end{pmatrix} = x_1 \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} + x_2 \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} + x_3 \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}.$$

Thus, the orthogonal complement $S^\perp$ is spanned by the normal vector:

$$S^\perp = \text{span} \left\{ \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix} \right\}.$$

The basis for the orthogonal complement $S^\perp$ is:

$$\left\{ \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix} \right\}.$$

`Matlab` code and output for this task are given below.

```
% Use the null space function to find the basis for the orthogonal
    complement
n = A'; % Normal vector as a column vector

% Display the normal vector
disp('Normal vector:');
```

```
Normal vector:
```

```
disp(n);
```

```
         1
         1
         1
        -1
```

2. Find an orthonormal basis for $\mathbb{R}^3$ starting from the vector (1 1 -1)

Hint: Get basis set for null space for A=[1 1 -1]. Apply orthogonalization process intelligently.

SOLUTION

Given the matrix $A = \begin{pmatrix} 1 & 1 & -1 \end{pmatrix}$ and the vector $\mathbf{b} = \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix}$, we are asked to project $\mathbf{b}$ onto the null space of A.

The null space of A, $\mathcal{N}(A)$, consists of all vectors $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$ such that:

$$x_1 + x_2 - x_3 = 0.$$

This implies that $x_1 = -x_2 + x_3$. Therefore, a basis for the null space is given by:

$$\mathcal{N}(A) = \text{span}\left\{ \begin{pmatrix} -1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \right\}.$$

Let $B = \begin{pmatrix} -1 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$ be the matrix whose columns are the basis vectors for $\mathcal{N}(A)$.
The projection of $\mathbf{b}$ onto the null space is given by:

$$\text{proj}_{\mathcal{N}(A)}(\mathbf{b}) = B(B^T B)^{-1} B^T \mathbf{b}.$$

First, compute $B^T B$:

$$B^T B = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}.$$

The inverse of $B^T B$ is:

$$(B^T B)^{-1} = \frac{1}{3} \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}.$$

Now, compute $B^T \mathbf{b}$:

$$B^T \mathbf{b} = \begin{pmatrix} -1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -1 \\ 2 \end{pmatrix}.$$

Thus,

$$(B^T B)^{-1} B^T \mathbf{b} = \frac{1}{3} \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} -1 \\ 2 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Finally, the projection is:

$$\text{proj}_{\mathcal{N}(A)}(\mathbf{b}) = B \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}.$$

Thus, the projection of **b** onto the null space of A is:

$$v = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}.$$

`Matlab` code and output for this task is given below.

```
format rational;
v1 = [1; 1; -1];
u1 = v1 / norm(v1);
e1 = [1; 0; 0];
u2 = e1 - (dot(u1, e1) / dot(u1, u1)) * u1;
u2 = u2 / norm(u2);
e2 = [0; 1; 0];
u3 = e2 - (dot(u1, e2) / dot(u1, u1)) * u1 - (dot(u2, e2) / dot(u2,
     u2)) * u2;
u3 = u3 / norm(u3);
disp('Orthonormal basis:');
```

```
Orthonormal basis:
```

```
disp('u1:');
```

```
u1:
```

```
disp(u1);
```

```
     780/1351
     780/1351
    -780/1351
```

```
disp('u2:');
```

```
u2:
```

```
disp(u2);
```

```
     881/1079
    -881/2158
     881/2158
```

```
disp('u3:');
```

```
u3:
```

```
disp(u3);
```

```
     0
   985/1393
   985/1393
```

3. What point on the plane $x + y - z = 0$ is closest to b=(2 1 0)

Hint project b on to null space (right) of A=[1 1 -1]. Get those vectors as columns of Matrix B and project b into column space of B.

```
% Given data
A = [1 1 -1];    % Matrix A
b = [2; 1; 0];   % Vector b
B = null(A, 'r');
proj_b = B * inv(B' * B) * (B' * b);
disp('The point on the plane closest to b is:');
```

```
The point on the plane closest to b is:
```

```
disp(proj_b);
```

```
     1
     0
     1
```

RESULTS

1. Connections of linear algebra with Geometry is revisited.
2. Solution of linear equations and its connection with the null space of transformation matrix is discussed.

3 | Assignment 35 Definiteness of Symmetric Matrices

3.1 Introduction

Symmetric matrices are associated with pure quadratic polynomials which in most two variable cases take the shape of parabolas with vertex at origin. The other shapes possible are 1) a channel with parabolic cross section and 2) saddle.

In case of two variables, quadratic polynomial is of the form

$\phi(x)=\phi\left(x_{1,}x_2\right)=$ ax_1^2 + 2bx_1x_{2+} cx_2^2

In matrix form $\phi(x) = (x_1\ x_2)\begin{bmatrix} a & b \\ b & c \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}=x^T$ Ax

Now matrix A characterizes the shape of the function. Depending on elements in A (to be more precise, its eigen values) , it can take following shapes.

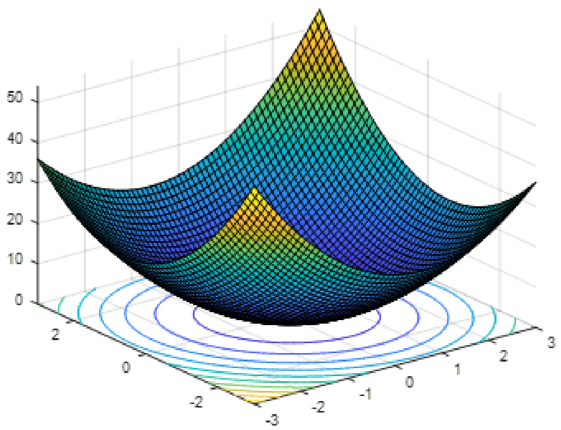


Figure 3.1: 3D visualization of quadratic form-1.

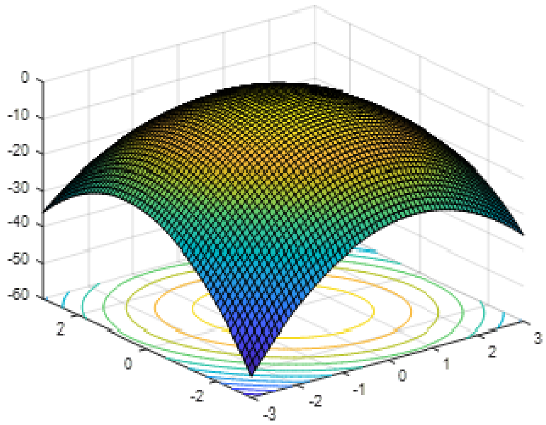


Figure 3.2: 3D visualization of quadratic form-2.

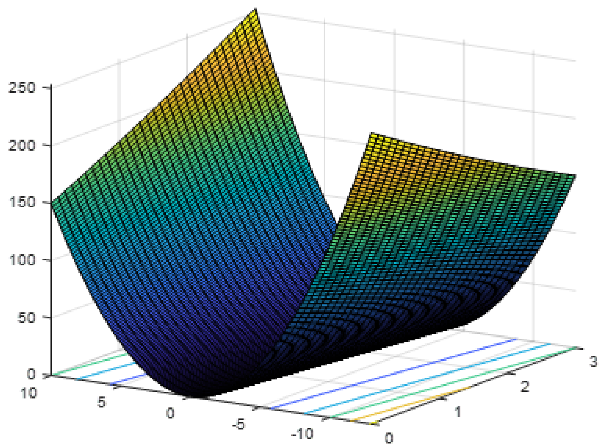


Figure 3.3: 3D visualization of quadratic form-3.

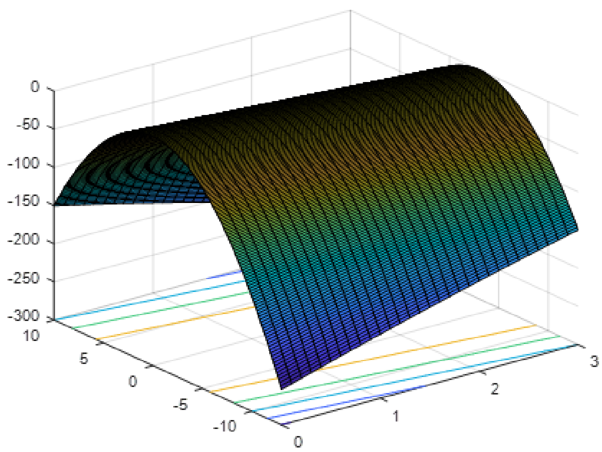


Figure 3.4: 3D visualization of quadratic form-4.

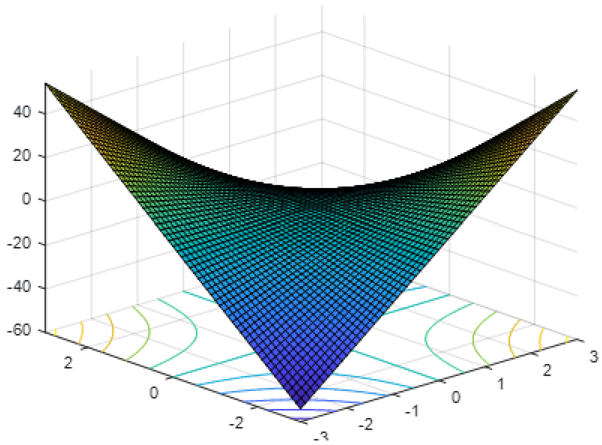


Figure 3.5: 3D visualization of quadratic form-5.

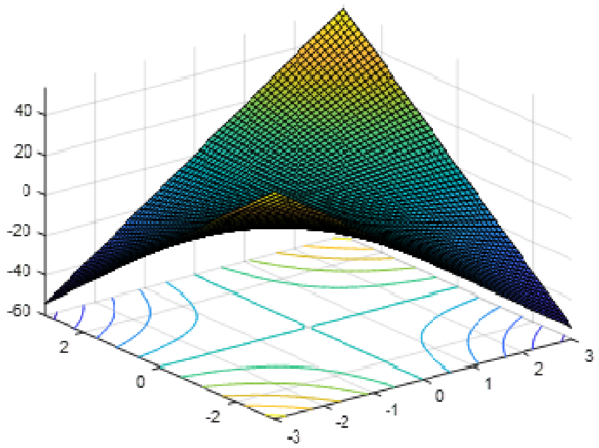


Figure 3.6: 3D visualization of quadratic form-6.

It is also associated with probability distributions like multivariate normal distributions in the form of covariance matrices.

Definition: Let A be nxn and symmetric (we assume elements of A are real),

Then if

1) x^T Ax > 0 $\forall$ $(x \neq 0) \in R^n$, then A is positivedefinite

2) x^TAx ≥ 0 $\forall$ $(x \neq 0)\in R^n$, then A is positive semidefinite

3) x^T Ax < 0 $\forall$ $(x \neq 0)\in R^n$, then A is negative definite

4) x^T Ax ≤ 0 $\forall$ $(x \neq 0)\in R^n$, then A is negative semidefinite

5) x^T Ax>0 for some x$\in$ R^n, and x^T Ax ≤ 0 for some other x$\in$ R^n A is inidefinite

Prove that if A is real, symmetric and x^T A_{nxn}x>0;$\forall x \neq 0$

$\Rightarrow$all eigenvalue $\lambda > 0$

Proof:

A is real and symmetric implies eigenvalues are **real** and eigen vectors are **real** and orthogonal.

We assume all eigenvalues are distinct

Let $x_1, x_2, \ldots\ldots x_n$ be unit norm eigen vectors corresponding to eigen values λ_1 ,λ_2.....,λn also , $x_i \perp x_j$, i$\neq j$ and they span vector space R^n.

Therefore, we can express any x as linear combination of eigenvectors of A

$$x \neq 0 \in \mathbb{R}^n \Rightarrow x = c_1 x_1 + c_2 x_2 + ... + c_r x_r,\ \text{each } c_r \in \mathbb{R}$$

$$Ax = A(c_1 x_1 + c_2 x_2 + \ldots + c_n x_n) = c_1 \lambda_1 x_1 + c_2 \lambda_2 x_2 + \ldots + c_n \lambda_n x_n$$

$$x^T Ax = (c_1 x_1 + c_2 x_2 + \ldots + c_n x_n)^T (c_1 \lambda_1 x_1 + c_2 \lambda_2 x_2 + \ldots + c_n \lambda_n x_n)$$

$$x^T Ax = c_1^2 \lambda_1 + c_2^2 \lambda_2 + \ldots + c_n^2 \lambda_n > 0 \Rightarrow \text{every } \lambda_i > 0 \text{ since } c_i^2 > 0\ \forall i$$

Creating 2x2 Matrices with desired eigen values

As per spectral theorem, square matrix A with distinct eigen values can be decomposed into $A = S\Lambda S^{-1}$

If A is real and symmetric, eigenvalues are real and eigenvectors are orthogonal (also real) .

Let us take S =$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$; then S^{-1}= $\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$

A = $\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}=\frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$

We use this to create a symmetric matrix with desired eigen values.

```
l1=3;

l2=-3;

B=[1 1; 1 -1];

A=(1/2)*B*diag([l1 l2])*B;

x1=-3:.1:3;

x2=-3:.1:3;

[X1,X2]=meshgrid(x1,x2);

fx=A(1,1)*X1.^2+A(2,2)*X2.^2+2*A(1,2)*X1.*X2;surfc(X1,X2,fx)
```

Creating 2x2 orthogonal matrices using QR decomposition of random matrices

We now A=QR.

Where Q is an orthogonal matix.

This can be employed for creating quadratic polynomials with different shapes.

```
close(gcf)

l1=-3;

l2=3;

B=randi(10,2,2);

[Q,R]=qr(B);

A=Q*diag([l1 l2])*Q';

x1=-3:.1:3;

x2=-3:.1:3;

[X1,X2]=meshgrid(x1,x2);

fx=A(1,1)*X1.^2+A(2,2)*X2.^2+2*A(1,2)*X1.*X2;

surfc(X1,X2,fx)
```

We already know, Orthogonal matrices can also be created using SVD.

Theorem

Every square matrix can be factored into a **Orthogonal matrix** and **a symmetric positive semidefinite matrix**.

If A is full rank (nonsingular) then the second matrix is positive definite.

$$A = U\Sigma V^T$$

$$A = U(V^T V)\Sigma V^T \text{ and } V^T V = I$$

$$A = \underbrace{(UV^T)}_{\text{orthogonal}} \underbrace{(V\Sigma V^T)}_{\text{positive semidefinite}}$$

Note that

$(UV^T)^T(UV^T) = VU^TUV^T = VV^T = I$ hence UV^T is an orthogonal matrix.
$V\Sigma V^T$ is positive semidefinite since Σ is.

Assignments

1. If A and B are nxn symmeyric positive definite matrices then prove that A+B is positive definite.

SOLUTION

To prove that if A and B are symmetric positive definite matrices, then $A+B$ is also positive definite, we proceed as follows:
Given that, A and B are $n \times n$ symmetric positive definite matrices.
A symmetric matrix M is positive definite if for all non-zero vectors $\mathbf{x} \in \mathbb{R}^n$, we have:

$$\mathbf{x}^T M\mathbf{x} > 0.$$

Since A and B are positive definite, we know that:

$$\mathbf{x}^T A\mathbf{x} > 0 \quad \text{and} \quad \mathbf{x}^T B\mathbf{x} > 0 \quad \text{for all} \, \mathbf{x} \in \mathbb{R}^n, \mathbf{x} \neq 0.$$

Now consider the matrix $A+B$. We need to show that $A+B$ is positive definite. For any non-zero vector $\mathbf{x} \in \mathbb{R}^n$, we have:

$$\mathbf{x}^T (A+B)\mathbf{x} = \mathbf{x}^T A\mathbf{x} + \mathbf{x}^T B\mathbf{x}.$$

Since A and B are positive definite, both terms on the right-hand side are strictly positive:

$$\mathbf{x}^T A\mathbf{x} > 0 \quad \text{and} \quad \mathbf{x}^T B\mathbf{x} > 0.$$

Therefore:

$$\mathbf{x}^T (A+B)\mathbf{x} = \mathbf{x}^T A\mathbf{x} + \mathbf{x}^T B\mathbf{x} > 0.$$

This shows that for all $\mathbf{x} \neq 0$, $\mathbf{x}^T (A+B)\mathbf{x} > 0$, which means that $A+B$ is positive definite.
Since we have shown that for any non-zero vector $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{x}^T (A+B)\mathbf{x} > 0$, we conclude that $A+B$ is positive definite.

$$A+B \text{ is positive definite.}$$

`Matlab` code and output to verify this theorem is given below.

```
n = 3;
R1 = randi(n,n);
R2 = randi(n,n);
A = R1' * R1;
B = R2' * R2;
disp("Matrix A:");
```

```
Matrix A:
```

```
disp(A);
```

```
    22    20    22
    20    19    20
    22    20    22
```

```
disp("Matrix B:");
```

```
Matrix B:
```

```
disp(B);
```

```
     6          10           9
    10          22          15
     9          15          14
```

```
C = A + B;
disp("Matrix sum, C=A+B");
```

```
Matrix sum, C=A+B
```

```
disp(C);
```

```
    28          30          31
    30          41          35
    31          35          36
```

```
isPositiveDefiniteA = all(eig(A) > 0);
isPositiveDefiniteB = all(eig(B) > 0);
isPositiveDefiniteC = all(eig(C) > 0);
fprintf('Matrix A is positive definite: %d\n', isPositiveDefiniteA)
   ;
```

```
Matrix A is positive definite: 1
```

```
fprintf('Matrix B is positive definite: %d\n', isPositiveDefiniteB)
   ;
```

```
Matrix B is positive definite: 1
```

```
fprintf('Matrix C = A + B is positive definite: %d\n',
   isPositiveDefiniteC);
```

```
Matrix C = A + B is positive definite: 1
```

2. If λ is an eigenvalue of nonsingular square matrix A, then $\frac{1}{\lambda}$ is an eigen value of A^{-1}.

SOLUTION

Proof

Let A be a nonsingular square matrix. Suppose λ is an eigenvalue of A. By definition, there exists a non-zero vector $\mathbf{v}$ such that:

$$A\mathbf{v} = \lambda\mathbf{v}.$$

We aim to show that $\frac{1}{\lambda}$ is an eigenvalue of A^{-1}.
1. Starting from the eigenvalue equation, we multiply both sides by A^{-1}:

$$A^{-1}(A\mathbf{v}) = A^{-1}(\lambda\mathbf{v}).$$

2. Using the property $A^{-1}A = I$ (where I is the identity matrix), we have:

$$I\mathbf{v} = \lambda A^{-1}\mathbf{v}.$$

3. This simplifies to:

$$\mathbf{v} = \lambda A^{-1}\mathbf{v}.$$

4. Rearranging gives:

$$A^{-1}\mathbf{v} = \frac{1}{\lambda}\mathbf{v}.$$

Thus, we have shown that the vector $\mathbf{v}$ is an eigenvector of A^{-1} corresponding to the eigenvalue $\frac{1}{\lambda}$. Therefore, we conclude that if λ is an eigenvalue of a nonsingular matrix A, then $\frac{1}{\lambda}$ is indeed an eigenvalue of A^{-1}.
Hence

$$\text{If } \lambda \text{ is an eigenvalue of } A, \text{ then } \frac{1}{\lambda} \text{ is an eigenvalue of } A^{-1}.$$

`Matlab` code and its output that verify this theorem is given below.

```
% Create a diagonal matrix with the desired eigenvalues
eigenvalues = [1; 2; 3];
D = diag(eigenvalues); % Diagonal matrix with eigenvalues on the
    diagonal

% Create a random orthogonal matrix to transform D
[Q, ~] = qr(rand(3)); % QR decomposition to get an orthogonal
    matrix

% Construct the matrix A with the desired eigenvalues
A = Q * D * Q'; % A is symmetric and has eigenvalues [1, 2, 3]
% Compute the eigenvalues of A
eigenvalues_A = eig(A);

% Compute the inverse of A
A_inv = inv(A);

% Compute the eigenvalues of A_inv
eigenvalues_A_inv = eig(A_inv);

% Expected eigenvalues for A^{-1}
expected_eigenvalues_inv = 1 ./ eigenvalues;

% Sort the eigenvalues for comparison
eigenvalues_A_inv_sorted = sort(eigenvalues_A_inv);
expected_eigenvalues_inv_sorted = sort(expected_eigenvalues_inv);

% Display results
fprintf('Eigenvalues of A: \n');
```

```
Eigenvalues of A:
```

```
disp(sort(eigenvalues_A)); % Sort for clarity
```

```
1
2
3
```

```
fprintf('Eigenvalues of A^{-1}: \n');
```

```
Eigenvalues of A^{-1}:
```

```
disp(eigenvalues_A_inv_sorted);
```

```
1/3
1/2
1
```

```
fprintf('Expected eigenvalues of A^{-1}: \n');
```

```
Expected eigenvalues of A^{-1}:
```

```
disp(expected_eigenvalues_inv_sorted);
```

```
1/3
1/2
1
```

```

% Check if the expected eigenvalues are close to those of A^{-1}
isClose = all(abs(eigenvalues_A_inv_sorted -
   expected_eigenvalues_inv_sorted) < 1e-10);
if isClose
    fprintf('The expected eigenvalues are indeed eigenvalues of A
       ^{-1}.\n');
else
    fprintf('The expected eigenvalues are NOT eigenvalues of A
       ^{-1}.\n');
end
```

```
The expected eigenvalues are indeed eigenvalues of A^{-1}.
```

1. If A is symmetric positive definite prove that A^2 and A^{-1} are also positive definite.

```
% Generate a random symmetric positive definite matrix A
n = 4; % Size of the matrix
A = randn(n);
A = A' * A; % A is now symmetric and positive definite

% Check if A is positive definite
disp('Eigenvalues of A:')
```

```
Eigenvalues of A:
```

```
eig_A = eig(A);
disp(eig_A); % Eigenvalues should all be positive
```

```
   39/5608
 1133/964
 2533/947
 1641/263
```

```

% Calculate A^2
A_squared = A^2;

% Check if A^2 is positive definite
disp('Eigenvalues of A^2:')
```

```
Eigenvalues of A^2:
```

```
eig_A_squared = eig(A_squared);
disp(eig_A_squared); % Eigenvalues should all be positive
```

```
   37/765048
  163/118
 4357/609
16001/411
```

```

% Calculate A^{-1}
A_inv = inv(A);

% Check if A^{-1} is positive definite
disp('Eigenvalues of A^{-1}:')
```

```
Eigenvalues of A^{-1}:
```

```
eig_A_inv = eig(A_inv);
disp(eig_A_inv); % Eigenvalues should all be positive (inverses of
    the eigenvalues of A)
```

```
  263/1641
  947/2533
  964/1133
 5608/39
```

```

% Verifying if both A^2 and A^{-1} are symmetric positive definite
is_A_squared_pos_def = all(eig_A_squared > 0);
is_A_inv_pos_def = all(eig_A_inv > 0);

if is_A_squared_pos_def
    disp('A^2 is positive definite.');
else
    disp('A^2 is not positive definite.');
end
```

```
A^2 is positive definite.
```

```

if is_A_inv_pos_def
    disp('A^{-1} is positive definite.');
else
    disp('A^{-1} is not positive definite.');
end
```

```
A^{-1} is positive definite.
```

1. If A is symmetric positive definite and C is nonsingular ,then $B = C^T AC$ is also symmetric positive definite.

SOLUTION

Proof

Let $A \in \mathbb{R}^{n\times n}$ be a symmetric positive definite (SPD) matrix. We will prove that both A^2 and A^{-1} are also positive definite matrices.

Proving that A^2 is Positive Definite:

Since A is symmetric positive definite, for any non-zero vector $x \in \mathbb{R}^n$, we have:

$$x^T Ax > 0.$$

Now, consider the matrix A^2. We need to show that for any non-zero vector $x \in \mathbb{R}^n$:

$$x^T A^2 x > 0.$$

Using the property of matrix multiplication:

$$x^T A^2 x = x^T AAx = (Ax)^T Ax.$$

Let $y = Ax$, where $y \neq 0$ because A is positive definite and $Ax \neq 0$ for all non-zero x. Thus, the expression becomes:

$$x^T A^2 x = y^T Ay.$$

Since A is positive definite, we know that:

$$y^T Ay > 0.$$

Therefore:

$$x^T A^2 x > 0.$$

This proves that A^2 is positive definite.

Proving that A^{-1} is Positive Definite:

We now show that the inverse of a symmetric positive definite matrix, A^{-1}, is also positive definite. To do this, we need to show that for any non-zero vector $x \in \mathbb{R}^n$:

$$x^T A^{-1} x > 0.$$

Let $y = A^{-1}x$. Then, multiplying both sides by A, we have $Ay = x$. Substituting this into the expression, we get:

$$x^T A^{-1} x = (Ay)^T A^{-1}(Ay) = y^T Ay.$$

Since A is positive definite, we know that:

$$y^T Ay > 0 \quad \text{for all } y \neq 0.$$

Thus, it follows that:

$$x^T A^{-1} x > 0.$$

Therefore, A^{-1} is positive definite.
`Matlab` code and its output verifying this theorem is given below.

```
A = rand(3);
A = A' * A;
C = rand(3);
B = C' * A * C;
tolerance = 1e-10;
is_symmetric = norm(B - B', 'fro') < tolerance;
eigenvalues_B = eig(B);
is_positive_definite = all(eigenvalues_B > 0);
disp('Eigenvalues of B:')
```

```
Eigenvalues of B:
```

```
disp(eigenvalues_B)
```

```
    5735/473
      32/2501
      44/16795
```

```

if is_symmetric
    disp('B is symmetric.')
else
    disp('B is not symmetric.')
end
```

```
B is symmetric.
```

```

if is_positive_definite
    disp('B is positive definite.')
else
    disp('B is not positive definite.')
end
```

```
B is positive definite.
```

RESULTS

1. Properties of eigen values of symmetric matrices are verified.
2. Method to create positive symmetric matrices is reviewed
3. Relationship between nature of definiteness and eigen values of a symmetric matrix is studied.

4 Assignment 36
Shifted QR Decomposition Algorithm

4.1 Uses of Matrix Decomposition

A=LU is mainly for solving linear system of simultaneous equations of the form, $Ax = b$

Spectral decomposition is mainly for computing A^k and e^A

Applications of Singular Value Decomposition (SVD) are plenty starting from compression to indexing and searching.

Many are wondering the use of QR Decomposition. Its main use is in Eigenvalue value computation of large matrices . Here is a code demonstrating how it is used for the task. Figure out how it working? Following `Matlab` code demonstrate the heart of the very useful *QR Algorithm*- iteratively extract eigen values without solving $AX = \lambda X$.

```
A=randi(10,2,2);
sort(eig(A))
```

```
ans = 2x1
   -2.8310
    8.8310
```

```
B=A;
for i= 1:10
    [Q R]=qr(B); % B=QR.
    % Now create a new B=RQ and repeat the computation.
    % B retains eigen values  of A but approaches to an upper
       triangular matrix
    % That is, diagonal elements of B approaches to eigen values of
        A
    B=R*Q;
end
sort(diag(B))
```

```
ans = 2x1
   -2.8309
    8.8309
```

Note that we will always get eig(A) and diag(B) as same set of values.

It is how eigenvalues are computed in practice.

This Computation can be faster using shifted QR decomposition.

4.2 Tasks

1. Write a note about shifted QR decomposition for eigenvalue computation.

SOLUTION

The **Shifted QR Algorithm** is an iterative method used to compute the eigenvalues of a square matrix. The method improves the convergence of the basic QR algorithm by applying a shift to the matrix in each iteration, typically using the bottom-right element of the current matrix. This shift accelerates convergence by reducing the off-diagonal elements of the matrix faster.

Let $A \in \mathbb{R}^{n\times n}$ be a real square matrix. The steps for the shifted QR algorithm are:

1. Start with the matrix A.
2. At each iteration k, choose a shift μ_k, usually the bottom-right element of the current matrix A_k.
3. Perform a QR decomposition on the matrix $A_k - \mu_k I$, where I is the identity matrix and μ_k is the shift.

$$A_k - \mu_k I = Q_k R_k$$

where Q_k is an orthogonal matrix and R_k is an upper triangular matrix.
4. Update the matrix:

$$A_{k+1} = R_k Q_k + \mu_k I$$

5. Repeat until A_k converges to an upper triangular matrix, from which the eigenvalues can be read off as the diagonal elements.

The algorithm for the simple shifted QR algorithm is:

Algorithm 1 Shifted QR Algorithm for Eigenvalue Computation

```
Input: Matrix A ∈ R^{n×n}
Output: Eigenvalues of A
Initialize A_0 = A, tolerance ϵ, and max iterations max_iter
for k = 0 to max_iter do
    Let μ_k = (A_k)_{n,n} {Choose shift as the bottom-right element of A_k}
    Compute QR decomposition: A_k − μ_k I = Q_k R_k
    Update: A_{k+1} = R_k Q_k + μ_k I
    if ‖A_{k+1} − A_k‖ < ϵ then
        break {Check convergence}
    end if
end for
Return: Diagonal elements of A_k as the eigenvalues
```

The algorithm terminates when the off-diagonal elements of the matrix A_k are sufficiently small (below a certain tolerance).

Convergence

For symmetric matrices, the shifted QR algorithm converges rapidly to the eigenvalues. The choice of shift μ_k affects the convergence speed; using the bottom-right element is a common and effective choice.

2. Write a `matlab` code for the same.

SOLUTION

`Matlab` code for executing the *shifted QR algorithm* and its output is given below. This computational approach verified the simple Shifted QR algorithm for symmetric matrices.

```
A = rand(4);
A = A + A';  % Making the matrix symmetric
% Parameters
n = size(A, 1);
tol = 1e-12;
max_iter = 500;
A_k = A;
shifted_eigenvalues = [];
for k = 1:max_iter
    mu_k = A_k(n, n);
    [Q_k, R_k] = qr(A_k - mu_k * eye(n));
    A_k = R_k * Q_k + mu_k * eye(n);
    if norm(tril(A_k, -1), 'fro') < tol
        break;
    end
end
shifted_eigenvalues = diag(A_k);
direct_eigenvalues = eig(A);
sorted_direct_eigenvalues = sort(direct_eigenvalues);
sorted_shifted_eigenvalues = sort(shifted_eigenvalues);
disp('Eigenvalues from the built-in eig function:');
```

```
Eigenvalues from the built-in eig function:
```

```
disp(sorted_direct_eigenvalues);
```

```
-1.3356
-0.5663
 0.2032
 4.7619
```

```

disp('Eigenvalues from the shifted QR algorithm:');
```

```
Eigenvalues from the shifted QR algorithm:
```

```
disp(sorted_shifted_eigenvalues);
```

```
-1.3356
-0.5663
 0.2032
 4.7619
```

```
if isequal(round(sorted_direct_eigenvalues, 8), round(
    sorted_shifted_eigenvalues, 8))
    disp('The eigenvalues computed by both methods match.');
else
    disp('The eigenvalues computed by both methods do not match
        .');
end
```

```
The eigenvalues computed by both methods match.
```

RESULTS

1. Various Matrix decompositions and their applications are revisited.

2. Eigen values of matrices are extracted using repeated use of $QR-$ decomposition on small matrices.

3. Shifted QR decomposition algorithm is discussed and computationally verified on symmetric matrices. On non-symmetric matrices, there are significant difference in eigen values computed through simple shifted QR decomposition algorithm.

5 | Assignment 37
Multivariate Optimization

5.1 Calculus Basics for Optimization

The calculus method of optimization involves finding the maximum or minimum values of functions, commonly known as extrema, by analyzing the behavior of their derivatives. This technique is frequently applied in computer science, engineering, economics, and various other fields to optimize resource allocation, compute costs, and perform similar tasks.

1. Single-variable Functions

For a single-variable function $f(x)$, the calculus-based optimization process follows these steps:

Algorithm 2 Optimization of Single-variable Function

Input: A single-variable function $f(x)$.
Compute the first derivative $f'(x)$.
Set $f'(x) = 0$ and solve for x to find the critical points.
Compute the second derivative $f''(x)$.
if $f''(x) > 0$ **then**
    The critical point is a local minimum.
else if $f''(x) < 0$ **then**
    The critical point is a local maximum.
else
    The second derivative test is inconclusive.
end if
If the function is defined on a closed interval, evaluate $f(x)$ at the endpoints and compare with values at critical points.
Output: Local maxima and minima of $f(x)$.

2. Bivariate Functions

For a bivariate function $f(x, y)$, the process involves partial derivatives and the Hessian determinant.

Algorithm 3 Optimization of Bivariate Function

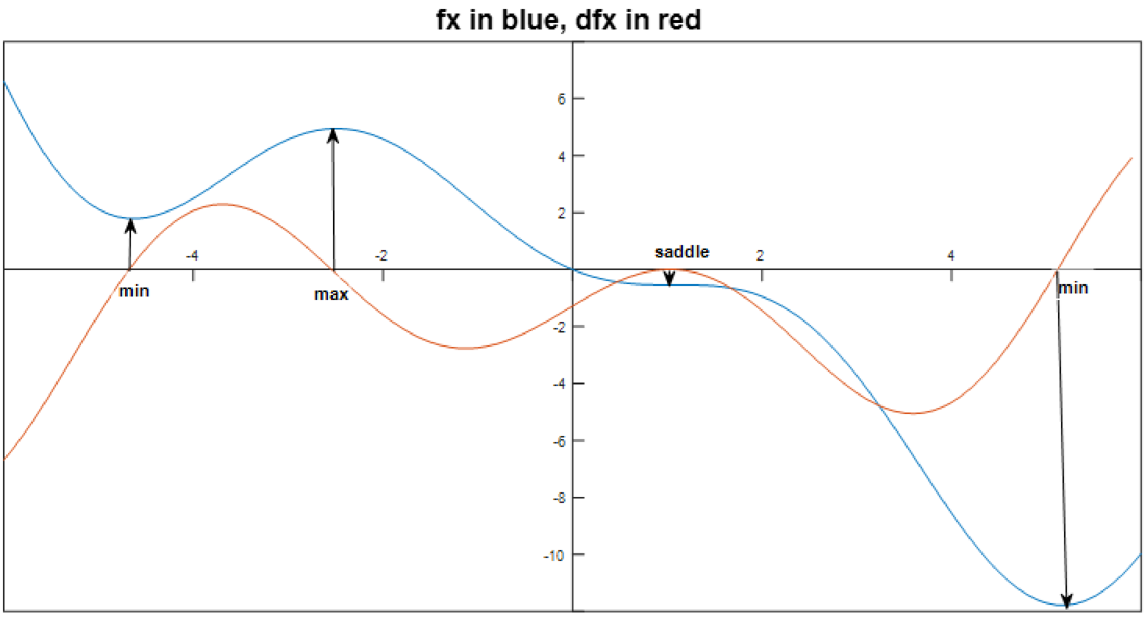
1: **Input:** A bivariate function $f(x,y)$.
2: Compute the partial derivatives $f_x = \frac{\partial f}{\partial x}$ and $f_y = \frac{\partial f}{\partial y}$.
3: Set $f_x = 0$ and $f_y = 0$ and solve for x and y to find the critical points.
4: Compute the second partial derivatives f_{xx}, f_{yy}, and f_{xy}.
5: Calculate the determinant of the Hessian matrix $D = f_{xx} f_{yy} - (f_{xy})^2$.
6: **if** $D > 0$ and $f_{xx} > 0$ **then**
7: The critical point is a local minimum.
8: **else if** $D > 0$ and $f_{xx} < 0$ **then**
9: The critical point is a local maximum.
10: **else if** $D < 0$ **then**
11: The point is a saddle point.
12: **else**
13: The test is inconclusive.
14: **end if**
15: If $f(x,y)$ is defined over a closed region, evaluate $f(x,y)$ along the boundary to ensure no extrema are overlooked.
16: **Output:** Local maxima, minima, or saddle points of $f(x,y)$.

Following is a plot of a function with different kinds of turning points.The function is given by $f(x) = x(-cos(1) - sin(1) + sin(x))$

Blue curve indicates the function

Red curve indicates derivative of the same function.

Note that where ever derivative function crosses x axis the original function attains maxima, minima or an inflection.

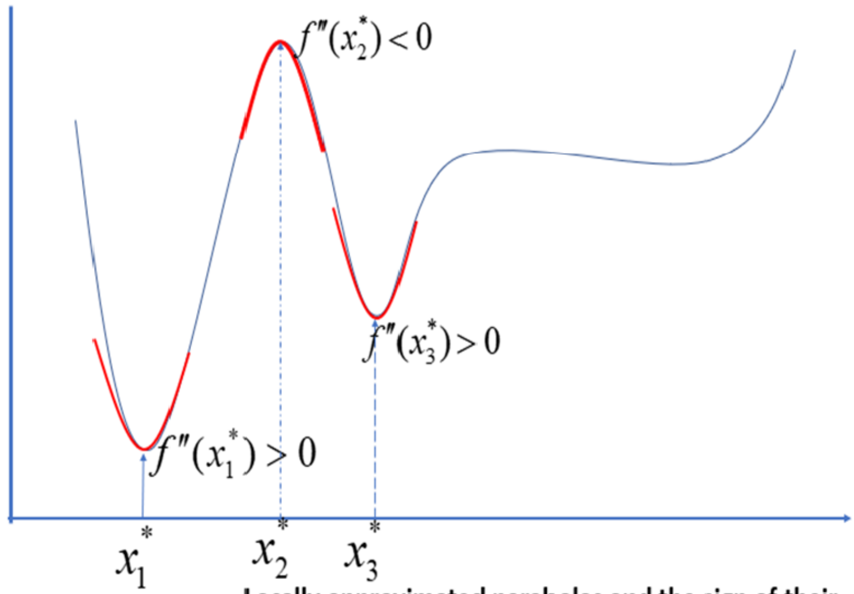


`Matlab` code for above diagram

```
\%one variable function with different inflection points
\%fx=x(-cos(1)-sin(1)+sin(x))
```

```
close(gcf)
h=0.1;
x=-6:h:6;
a=cos(1);
b=sin(1);
c=(-a-b)+sin(x);
fx=x.*c;
dfx=diff(fx)/h;
figure
plot(x,fx);
ax = gca;
ax.XAxisLocation = 'origin';
ax.YAxisLocation = 'origin';
hold on
plot(x(1:end-1),dfx);
caption = sprintf('fx in blue, dfx in red');
title(caption, 'FontSize', 20);
```

The following figure shows the sign of second derivatives at the turning points.



Locally approximated parabolas and the sign of their second derivatives at the turning points

Basically there are two kinds of algorithm for finding turning points.

1. a. gradient ascent.

For finding the maxima point, we move along gradient at the current location. Here the update rule is$x_{n+1} = x_n + \eta f'(x_n)$

Gradient (positive means function is increasing towards right, otherwise left) at any point indicate the direction as wells rate at which function is changing. Multiplier is step length. In many algorithms the quantity is reduced as iteration number is increased.

1).b gradient descent.

For finding the minima point, we move opposite to gradient direction at the current location. Here the update rule is $x_{n+1} = x_n - \eta f'(x_n)$

2. Newton method which move based on gradient (first order derivative) and curvature (second order derivative) at the current location.

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}$$

This just find turning point. It always move towards nearest turning point. It can be maxima, minima, or a saddle point.

Two variable function and its turning points. A function Exhibiting all kinds of turning points in $\mathbb{R}^2$.

$$f(x, y) = 25x^2 - 12x^4 - 6xy + 25y^2 - 24x^2y^2 - 12y^4$$

Turning points of the function

$$f(x, y) = 25x^2 - 12x^4 - 6xy + 25y^2 - 24x^2y^2 - 12y^4$$

```
close(gcf);
x=-1.2:.1:1.2;
y=x;
[X,Y]=meshgrid(x,y);
fxy=25*X.^2-12*X.^4-6*(X.*Y)+25*Y.^2-24*(X.^2).*(Y.^2)-12*Y.^4;
figure;
surfc(X,Y,fxy);
figure
contour(X,Y,fxy)
```

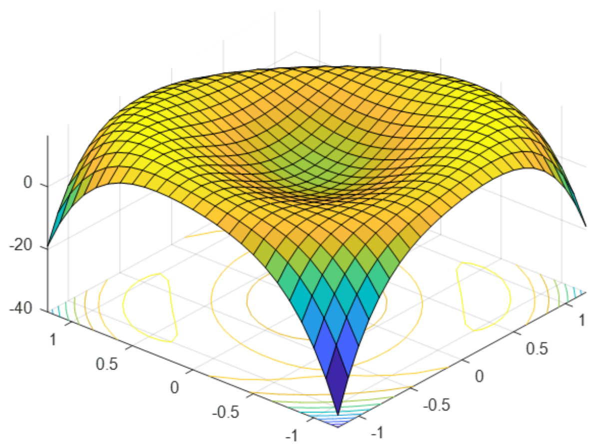


Figure 5.1: Surface plot of polynomial expression.

The contour plot showing the location and nature of critical points are shown in Figure 5.2. Figure 5.3 shows the direction of functional growth/ decay

```
close(gcf);
x=-0.9:.1:0.9;
y=x;
[X,Y]=meshgrid(x,y);
fxy=25*X.^2-12*X.^4-6*(X.*Y)+25*Y.^2-24*(X.^2).*(Y.^2)-12*Y.^4;
[px,py]=gradient(fxy,.5,.25);
contour(X,Y,fxy), hold on
quiver(X,Y,px,py), hold off, axis image
```

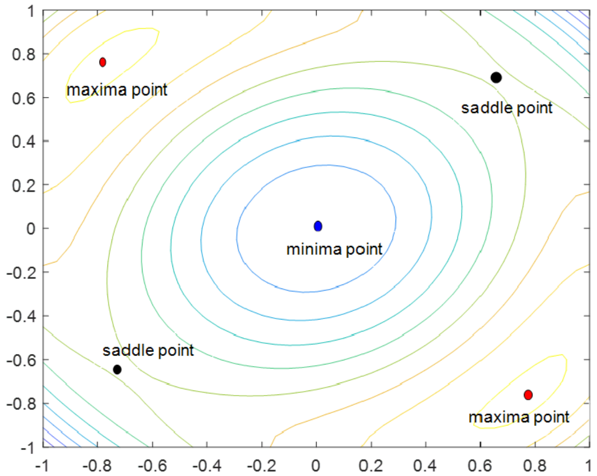


Figure 5.2: Location and nature of the critical points.

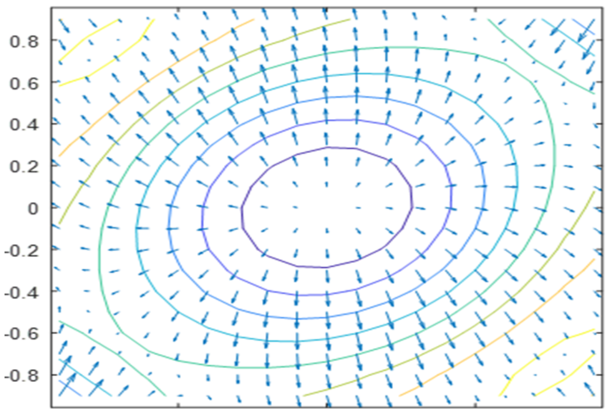


Figure 5.3: Direction of growth/ decay of $f(x, y)$ near the critical points.

How do we characterize the turning points.

Here we take the help of Madhava(Taylor) series expansion at the turning point.

$$f(x) \approx f(x^*) + (x - x^*)^T \nabla f(x^*) + \frac{1}{2}(x - x^*)^T \nabla^2 f(x^*)(x - x^*)$$

At the turning point, if we approximate the function up to second order, gradient vector $\nabla f(x^*)$is a zero vector. So we only have first and third term. The first term is a constant. The third term is a pure quadratic function. The quadratic term involving hessian matrix take a shape according to the curvature of function at the turning point. If it is a minimum point, the parabola will be 'upward open' (or convex). If it is a maximum point, the parabola will be 'downward open' (or concave).

This in turn is indicated by the 'definiteness' of the hessian matrix evaluated at the turning point.

The turning point x^* corresponds to a minimum point if $\nabla^2 f(x^*)$ is positive definite.

Mathematically, at x^* the function is a minimum if $\nabla^2 f(x^*) \succ 0$.
The turning point x^* corresponds to a maximum point if $\nabla^2 f(x^*)$ is negative definite.
Mathematically, at x^* the function is a maximum if $\nabla^2 f(x^*) \prec 0$.
The turning point x^* corresponds to a saddle point if $\nabla^2 f(x^*)$ is indefinite.

Assignment.

1. Determine the linear and quadratic approximation of the following function around the given point.

$$f\left(x, y\right) = 2x^2 + y^2 + \frac{1}{2x^2 + y^2} \text{ at } (2,2)$$

SOLUTION

The Taylor series expansion of a function $f(x, y)$ around the point (x_0, y_0) is given by:

$$f(x, y) \approx f(x_0, y_0) + \left.\frac{\partial f}{\partial x}\right|_{(x_0,y_0)} (x - x_0) + \left.\frac{\partial f}{\partial y}\right|_{(x_0,y_0)} (y - y_0)$$

$$+\frac{1}{2}\left[\left.\frac{\partial^2 f}{\partial x^2}\right|_{(x_0,y_0)} (x - x_0)^2 + 2\left.\frac{\partial^2 f}{\partial x \partial y}\right|_{(x_0,y_0)} (x - x_0)(y - y_0) + \left.\frac{\partial^2 f}{\partial y^2}\right|_{(x_0,y_0)} (y - y_0)^2\right].$$

To find the linear and quadratic approximations of $f(x, y)$, we will use the second-order Taylor series expansion around the point $(2,2)$.

we first evaluate $f(2,2)$:

$$f(2,2) = 2(2^2) + (2^2) + \frac{1}{2(2^2) + 2(2^2)} = 8 + 4 + \frac{1}{16} = \frac{193}{16}.$$

Next, we compute the first derivatives. The partial derivative with respect to x is given by

$$\frac{\partial f}{\partial x} = 4x - \frac{4x + y^2}{(2x^2 + xy^2)^2}.$$

Evaluating at $(2,2)$,

$$\frac{\partial f}{\partial x}\bigg|_{(2,2)} = 4(2) - \frac{4(2)+2^2}{(16)^2} = 8 - \frac{12}{256} = \frac{511}{64}.$$

The partial derivative with respect to y is

$$\frac{\partial f}{\partial y} = 2y - \frac{x^2 y}{(2x^2+xy^2)^2}.$$

Evaluating at $(2,2)$,

$$\frac{\partial f}{\partial y}\bigg|_{(2,2)} = 2(2) - \frac{(2^2)(2)}{(16)^2} = 4 - \frac{8}{256} = \frac{127}{32}.$$

Next, we find the second derivatives. The second derivative with respect to x is

$$\frac{\partial^2 f}{\partial x^2} = 4 - \frac{8x^2+y^2}{(2x^2+xy^2)^3}.$$

Evaluating at $(2,2)$,

$$\frac{\partial^2 f}{\partial x^2}\bigg|_{(2,2)} = 4 - \frac{36}{4096} = \frac{4087}{1024}.$$

The second derivative with respect to y is

$$\frac{\partial^2 f}{\partial y^2} = 2 - \frac{2x^2}{(2x^2+xy^2)^3}.$$

Evaluating at $(2,2)$,

$$\frac{\partial^2 f}{\partial y^2}\bigg|_{(2,2)} = 2 - \frac{8}{4096} = \frac{1023}{512}.$$

The mixed partial derivative is

$$\frac{\partial^2 f}{\partial x \partial y} = -\frac{x^2}{(2x^2+xy^2)^3} \cdot \frac{\partial}{\partial y}(2x^2+xy^2).$$

Calculating

$$\frac{\partial}{\partial y}(2x^2+xy^2) = 2xy,$$

and evaluating at $(2,2)$,

$$\frac{\partial^2 f}{\partial x \partial y}\bigg|_{(2,2)} = -\frac{4}{(16)^3}\cdot 8 = -\frac{32}{4096} = -\frac{1}{128}.$$

The Taylor series expansion about $(2,2)$ is given by

$$f(x,y) \approx f(2,2) + \frac{\partial f}{\partial x}\bigg|_{(2,2)}(x-2) + \frac{\partial f}{\partial y}\bigg|_{(2,2)}(y-2) + \frac{1}{2}\left[\frac{\partial^2 f}{\partial x^2}\bigg|_{(2,2)}(x-2)^2 + 2\frac{\partial^2 f}{\partial x \partial y}\bigg|_{(2,2)}(x-2)(y-2) + \frac{\partial^2 f}{\partial y^2}\bigg|_{(2,2)}(y-2$$

Substituting the computed values, we have

$$f(x,y) \approx \frac{193}{16} + \frac{511}{64}(x-2) + \frac{127}{32}(y-2) + \frac{1}{2}\left[\frac{4087}{1024}(x-2)^2 - \frac{1}{64}(x-2)(y-2) + \frac{1023}{512}(y-2)^2\right].$$

The linear approximation at $(2,2)$ is

$$L(x,y) = \frac{193}{16} + \frac{511}{64}(x-2) + \frac{127}{32}(y-2).$$

The quadratic approximation is

$$Q(x,y) = \frac{193}{16} + \frac{511}{64}(x-2) + \frac{127}{32}(y-2) + \frac{1}{2}\left[\frac{4087}{1024}(x-2)^2 - \frac{1}{64}(x-2)(y-2) + \frac{1023}{512}(y-2)^2\right].$$

2. The function $f\left(x,y\right) = (x-2)^2 + \left(x-2y^2\right)^2$ has 3 turning points . Prove analytically that these are at (1,0), (2,-1),(2,1).

SOLUTION

Given the function

$$f(x,y) = (x-2)^2 + (x-2y^2)^2,$$

we calculate the first partial derivatives:

$$f_x = 4x - 4 - 4y^2,$$
$$f_y = -8y(x-2y^2).$$

Setting the first partial derivatives to zero:

1. From $f_x = 0$:

$$4x - 4 - 4y^2 = 0 \implies x = 1 + y^2.$$

2. From $f_y = 0$:

$$-8y(x-2y^2) = 0 \implies y = 0 \text{ or } x = 2y^2.$$

Case 1: $y = 0$

Substituting into $x = 1 + y^2$:

$$x = 1 \implies (1,0).$$

Case 2: $x = 2y^2$

Substituting into $x = 1 + y^2$:

$$2y^2 = 1 + y^2 \implies y^2 = 1 \implies y = 1 \text{ or } y = -1.$$

For $y = 1$:

$$x = 2 \implies (2,1).$$

For $y = -1$:

$$x = 2 \implies (2,-1).$$

Summary of Critical Points

The critical points are: - $(1,0)$ - $(2,1)$ - $(2,-1)$

Second Partial Derivatives

Calculating the second partial derivatives:

$$
\begin{aligned}
f_{xx} &= 4, \\
f_{yy} &= 32y, \\
f_{xy} &= 16y.
\end{aligned}
$$

Summary table

SL.No	Critical Points	r	t	s	$D = rt - s^2$
1	$(1,0)$	4	0	0	0 (inconclusive)
2	$(2,1)$	4	32	16	-128 (saddle point)
3	$(2,-1)$	4	-32	-16	-384 (saddle point)

Table 5.1: Critical Points and Conclusions

3. Compute Hessian at these points and decide the nature of the points (minima,maxima,inflection)

SOLUTION

Hessian Matrix

The Hessian matrix is defined as:

$$
H = \begin{bmatrix} f_{xx} & f_{xy} \\ f_{xy} & f_{yy} \end{bmatrix}.
$$

The second partial derivatives are:

$$
\begin{aligned}
f_{xx} &= 4, \\
f_{yy} &= 32y, \\
f_{xy} &= 16y.
\end{aligned}
$$

Thus, the Hessian matrix becomes:

$$
H = \begin{bmatrix} 4 & 16y \\ 16y & 32y \end{bmatrix}.
$$

Evaluate the Hessian at Critical Points

1. **At** $(1,0)$:

$$
f_{yy} = 32 \cdot 0 = 0,
$$

$$
H(1,0) = \begin{bmatrix} 4 & 0 \\ 0 & 0 \end{bmatrix}.
$$

$\therefore$ H is positive semi-definite The determinant is

$$
\det(H) = 4 \cdot 0 - 0^2 = 0 \quad \text{(inconclusive)}.
$$

2. **At** $(2,1)$:

$$
f_{yy} = 32 \cdot 1 = 32,
$$

$$H(2,1) = \begin{bmatrix} 4 & 16 \\ 16 & 32 \end{bmatrix}.$$

$\therefore$ H is positive indefinite. The determinant is

$$\det(H) = 4 \cdot 32 - 16^2 = 128 - 256 = -128 \quad \text{(saddle point)}.$$

3. **At** $(2,-1)$:

$$f_{yy} = 32 \cdot (-1) = -32,$$

$$H(2,-1) = \begin{bmatrix} 4 & -16 \\ -16 & -32 \end{bmatrix}.$$

$\therefore$ H is positive indefinite The determinant is

$$\det(H) = 4 \cdot (-32) - (-16)^2 = -128 - 256 = -384 \quad \text{(saddle point)}.$$

RESULTS

1. Second partial derivative test is revisited.
2. Nature of exrtrema is identified with the nature of the Hessian matrix.
3. Role of Hessian matrix in identifying the nature of extrema is revisited.
4. Taylor series expansion is used to write the linear and quadratic approximation of a differentiable function about a point.
5. Nature of extrema of a given function is examined using the Calculus approach.

6 | Assignment 38 Constrained Optimization

6.1 Introduction

In optimization theory, constraints represent region of search. It is a set of points on which we evaluate a function called **objective function** and find out at which point(value of the variables) the function attains maxima or minima.

Usually we specify the search region by a set of equality and inequality constraints.

In two variable case we can use MATLAB to plot/color the region.

Let us plot the region defined by following inequalities

$$\begin{aligned} x + y &\leq 5 \\ x + y &\geq 0 \\ x &> -3 \\ y &> -3 \end{aligned}$$

`Matlab` code to plot the region bounded by these inequalities is given below.

```
v = -10:0.01:10;     % plotting range from -7 to 7
[x, y] = meshgrid(v,v);    % get 2-D mesh for x and y
conditions = (x+y<5) &(x+y>0)& (x> -3) & (y > -3);
cond = zeros(length(v)); % Initialize
cond(conditions) =NaN;
surf(x, y, cond)
%hold on
ax = gca;
ax.XAxisLocation = 'origin';
ax.YAxisLocation = 'origin';
view(0,90)
hold off
```

Output of the code is shown in Figure 6.1.

Assignment Q1

Figure out how the program works. Can you find out a way in `matlab` to change the black color to red. I do not know how to do it.

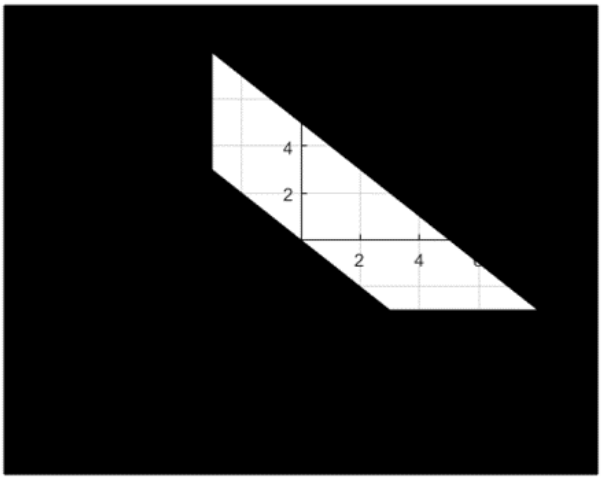


Figure 6.1: Common region bounded by the inequalities.

```
close(gcf)
v = -10:0.01:10;     % plotting range from -10 to 10
[x, y] = meshgrid(v,v);    % get 2-D mesh for x and y
conditions = (x+y<5) & (x+y>0) & (x > -3) & (y > -3);
cond = zeros(length(v)); % Initialize
cond(conditions) = NaN; % Set condition area to NaN (not plotted)

% Create the surface plot
h = surf(x, y, cond, 'FaceColor', 'flat', 'EdgeColor', 'none');

% Set CData for the colors
h.CData = cond;

% Set the colormap
colormap([1 0 0; 1 1 1]); % Red for NaN areas, White for 0 areas

% Manually set the color of the areas
h.CData(~isnan(cond)) = 1; % Set non-NaN areas to white (0)
h.CData(isnan(cond)) = 2;  % Set NaN areas to red (1)

% Adjust axes properties
ax = gca;
ax.XAxisLocation = 'origin';
ax.YAxisLocation = 'origin';
view(0, 90);
hold off;
```

Example 1.

Replace appropriate line in the above piece of code with the following

```
v = -5:0.01:5;
conditions = (x.^2+y.^2<20) \&(x.^2-y.^2<10);
```

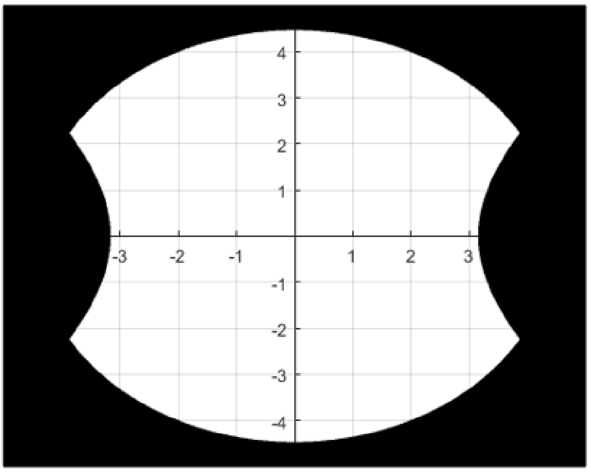


Figure 6.2: Revised plot with change in constraints.

```
v =  -5:0.01:5;    % plotting range from -10 to 10
[x, y] = meshgrid(v,v);   % get 2-D mesh for x and y
conditions = (x.^2 + y.^2 < 20) & (x.^2 - y.^2 < 10); % Define conditions for the region
cond = zeros(length(v)); % Initialize
cond(conditions) = NaN;  % Set NaN where the condition is met (red areas)

% Create the surface plot
h = surf(x, y, cond, 'EdgeColor', 'none');

% Set the colormap: white for background, red for the condition areas
colormap([1 1 1; 1 0 0]);  % White for NaN, red for areas where conditions hold

% Use AlphaData to make white regions transparent
h.AlphaData = ~isnan(cond); % 1 for non-NaN (red) areas, 0 (transparent) for others
h.FaceAlpha = 'flat';        % Apply per-vertex transparency

% Adjust axes properties
ax = gca;
ax.XAxisLocation = 'origin';  % Set X-axis at origin
ax.YAxisLocation = 'origin';  % Set Y-axis at origin
ax.XColor = 'k';              % Set X-axis color to black
ax.YColor = 'k';              % Set Y-axis color to black
ax.LineWidth = 1.0;           % Thicker axes lines for better visibility

% Bring the axis lines to the front
ax.Layer = 'top';  % This makes sure the axes are drawn over the surface plot

% Set the view to 2D
view(0, 90);

```

```
% Enable tick marks
ax.XTick = -5:1:5;            % Set X-axis tick positions
ax.YTick = -5:1:5;            % Set Y-axis tick positions
ax.XTickLabel = num2cell(-5:1:5);  % X-axis tick labels
ax.YTickLabel = num2cell(-5:1:5);  % Y-axis tick labels

% Ensure grid is visible
grid on;
set(gca, 'FontSize', 10);  % Set font size for better visibility of
     tick labels

hold on;

% Plot points (markers) on the axes at the tick positions
plot(ax.XTick, zeros(size(ax.XTick)), 'k.', 'MarkerSize', 1);  %
    Markers on X-axis
plot(zeros(size(ax.YTick)), ax.YTick, 'k.', 'MarkerSize', 1);  %
    Markers on Y-axis

hold off;
```

Assignment Q2

Plot the region enclosed by following constraints. Take x and y in the range -2 to 6

$$\left(x-\tfrac{3}{2}\right)^2+(y-5)^2 \leq 10$$
$$2x^2+3y^2 \leq 35$$
$$x \geq 0$$
$$y \geq 0$$

```
close(gcf)
v =  -2:0.01:6;    % plotting range from -2 to 6
[x, y] = meshgrid(v,v);    % get 2-D mesh for x and y

% Corrected the conditions statement (removed extra parenthesis)
conditions = ((x-3/2).^2 +(y-5).^2 <= 10) & (2*x.^2 - 3*y.^2 <= 35)
     & (x>=0) & (y>=0);

cond = zeros(length(v));    % Initialize
cond(conditions) = NaN;     % Set NaN where the condition is met (
    red areas)

% Create the surface plot
h = surf(x, y, cond, 'EdgeColor', 'none');

% Set the colormap: white for background, red for the condition
    areas
colormap([1 1 1; 1 0 0]);  % White for NaN, red for areas where
    conditions hold

% Use AlphaData to make white regions transparent
h.AlphaData = ~isnan(cond); % 1 for non-NaN (red) areas, 0 (
    transparent) for others
```

```
h.FaceAlpha = 'flat';        % Apply per-vertex transparency

% Adjust axes properties
ax = gca;
ax.XAxisLocation = 'origin';  % Set X-axis at origin
ax.YAxisLocation = 'origin';  % Set Y-axis at origin
ax.XColor = 'k';              % Set X-axis color to black
ax.YColor = 'k';              % Set Y-axis color to black
ax.LineWidth = 1.0;           % Thicker axes lines for better visibility

% Bring the axis lines to the front
ax.Layer = 'top';  % This makes sure the axes are drawn over the surface plot

% Set the view to 2D
view(0, 90);

% Enable tick marks
ax.XTick = -2:2:6;           % Set X-axis tick positions
ax.YTick = -2:2:6;           % Set Y-axis tick positions
ax.XTickLabel = num2cell(-2:2:6);  % X-axis tick labels
ax.YTickLabel = num2cell(-2:2:6);  % Y-axis tick labels

% Ensure grid is visible
grid on;
set(gca, 'FontSize', 10);  % Set font size for better visibility of tick labels

hold on;

% Plot points (markers) on the axes at the tick positions
plot(ax.XTick, zeros(size(ax.XTick)), 'k.', 'MarkerSize', 10);  % Markers on X-axis
plot(zeros(size(ax.YTick)), ax.YTick, 'k.', 'MarkerSize', 10);  % Markers on Y-axis

hold off;
```

Output of the above code is shown in Figure 6.3.

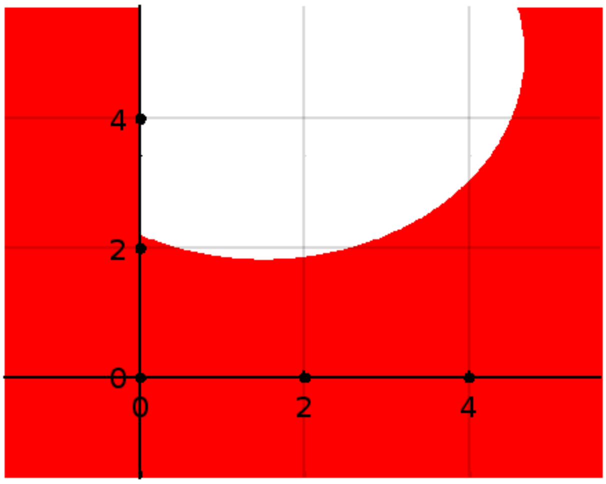


Figure 6.3: Revised plot with additional constraints.

Assignment Q3.

It is a general question. The answer must be found out using high-school geometric formulas. If r is the common radius of the circle, what is the common area enclosed. Note that the each circle pass through other's centre.

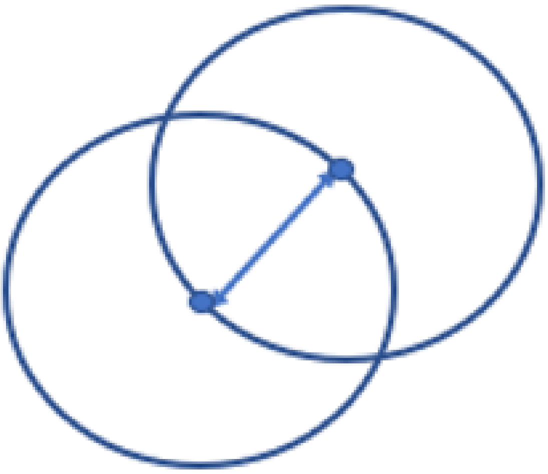


Figure 6.4: Figure for assignment question 3.

SOLUTION

Each circle has radius r, and the distance between the centers of the two circles is also r, since each passes through the center of the other.
The area of intersection is formed by two identical circular segments, each subtending an angle of 120° (or $\frac{2\pi}{3}$ radians) at the center of the respective circle.
The area of one circular segment is the difference between the area of the sector and the area of the triangle formed by the radii and the chord. This is given by the formula:

$$A_{\text{segment}} = A_{\text{sector}} - A_{\text{triangle}}$$

where:

$$A_{\text{sector}} = \frac{\theta}{2} \times r^2 = \frac{2\pi}{3} \times \frac{r^2}{2} = \frac{\pi r^2}{3}$$

and

$$A_{\text{triangle}} = \frac{1}{2}r^2 \sin\left(\frac{2\pi}{3}\right) = \frac{1}{2}r^2 \times \frac{\sqrt{3}}{2} = \frac{\sqrt{3}}{4}r^2$$

Thus, the area of one segment is:

$$A_{\text{segment}} = \frac{\pi r^2}{3} - \frac{\sqrt{3}}{4}r^2$$

Since there are two identical segments, the total common area is:

$$A_{\text{common}} = 2 \times \left(\frac{\pi r^2}{3} - \frac{\sqrt{3}}{4}r^2\right)$$

Simplifying:

$$A_{\text{common}} = \frac{2\pi r^2}{3} - \frac{\sqrt{3}}{2}r^2$$

Therefore, the total common area enclosed by the two intersecting circles is:

$$A_{\text{common}} = \frac{2\pi r^2}{3} - \frac{\sqrt{3}}{2}r^2$$

Constrained optimization is a field of optimization focused on finding the maximum or minimum of a function while adhering to specific restrictions, known as constraints. This approach is extensively applied in areas like engineering, economics, operations research, and machine learning, where practical problems often involve optimizing an objective subject to various limitations.
In constrained optimization, the goal is to find the optimal solution for an objective function $f(x)$ while satisfying a set of constraints. These constraints typically take the form of equality or inequality conditions on the variables involved.

Types of Constraints

- **Equality Constraints:** These constraints require that certain functions of the variables equal a specified value. They are written as:

  $$g_i(x) = 0 \quad \text{for } i = 1, \ldots, m.$$

- **Inequality Constraints:** These constraints impose limits that must not be exceeded, commonly expressed as:

  $$h_j(x) \leq 0 \quad \text{for } j = 1, \ldots, n.$$

Methods of Constrained Optimization

1. **Lagrange Multipliers:**
   The method of Lagrange multipliers is commonly used for problems with equality constraints. It involves defining an auxiliary function, called the Lagrangian, which combines the objective function and the constraints using a set of new variables known as Lagrange multipliers. For a function $f(x)$ with an equality constraint $g(x) = 0$, the Lagrangian $\mathscr{L}$ is defined as:

   $$\mathscr{L}(x, \lambda) = f(x) + \lambda g(x),$$

   where λ is the Lagrange multiplier. The solution is found by setting the partial derivatives of $\mathscr{L}$ with respect to each variable to zero.

2. **Karush-Kuhn-Tucker (KKT) Conditions:**
For problems with inequality constraints, the Karush-Kuhn-Tucker (KKT) conditions generalize the method of Lagrange multipliers. These conditions provide necessary conditions for a solution to be optimal, including feasibility and stationarity of the Lagrangian. The KKT conditions are fundamental in nonlinear programming and are widely used in operations research.

3. **Penalty and Barrier Methods:**
These methods transform a constrained optimization problem into an unconstrained one by adding a penalty or barrier term to the objective function. Penalty methods add a large penalty for constraint violations by points outside the feasible region, while barrier methods create a barrier within the feasible region to prevent the solution from approaching the constraint boundaries too closely.

RESULTS

1. Basic `matlab` commands for plotting (feasible) region are revisited.

2. Basics of constrained optimization is revisited.

7 Assignment 39
Generating Rowspace and Kernel Using Free Variables

7.1 Back to The Basics

7.1.1 Revisiting `rref()` as a main tool

Quickly creating rowspace basis set and right null space basis set using row elimination.

The treatment here is slightly different from those used in complete solution to $Ax = b$. But it is equivalent and produce same result.

Simplest first

Example 1

Let A=$\begin{bmatrix} 1 & 1 & 2 \\ 0 & 1 & -1 \end{bmatrix}$.Find rowspace basis set and right null space basis set.

Row vectors are independent. Therefore, row-space basis set is given by

$$\left\{ \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 2 & -1 \end{bmatrix} \right\}$$

To find right null space basis set, we consider,

$\begin{bmatrix} 1 & 1 & 2 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

This demand that solution vector to be orthogonal to row vectors.

That is $\begin{bmatrix} x \\ y \\ z \end{bmatrix} \perp \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} \& \begin{bmatrix} x \\ y \\ z \end{bmatrix} \perp \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix}$

Generally we go for row-elimination process to retrieve independent row-space vectors.

Row vectors are already in row-reduced format and independent. Therefore, here, row-elimination process is not required.

We now look for pivotal variables. (variables with first non-zero coefficient in each row)

$\underbrace{x}_{\uparrow}$ +y+2z=0

0x+ $\underbrace{y}_{\uparrow}$ -z=0

Arrow indicate, pivot location and variable. So x and y are pivot variables.

Now, Express, all variables in terms of free variables (non-pivot variable z) .

We start from free variables and bottom rows.

z = z

$$\text{y = z} \Rightarrow \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \text{z} \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

x = -3z

There fore, null space is spanned by the set $\left\{ \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix} \right\}$

Example 2.

$$A = \begin{bmatrix} 1 & 2 & 2 & 1 \\ 2 & 4 & 6 & 8 \\ 1 & 2 & 4 & 7 \end{bmatrix}$$.Find row space basis and right null space basis set.

On row-elimination

$$A = \begin{bmatrix} 1 & 2 & 2 & 1 \\ 2 & 4 & 6 & 8 \\ 1 & 2 & 4 & 7 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 2 & 1 \\ 0 & 0 & 2 & 6 \\ 1 & 2 & 4 & 7 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 2 & 1 \\ 0 & 0 & 2 & 6 \\ 0 & 0 & 2 & 6 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 2 & 1 \\ 0 & 0 & 2 & 6 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

So the row space is spanned by basis set

$$\left\{ \begin{bmatrix} 1 \\ 2 \\ 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 2 \\ 6 \end{bmatrix} \right\}$$

Let us find null space basis set vectors

$$\begin{bmatrix} 1 & 2 & 2 & 1 \\ 0 & 0 & 2 & 6 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

y and t are free variables. Express all variables in terms of free variables.

Start with free variables and then bottom rows to top rows

t = t

$$\text{y} = \text{y} \Rightarrow \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix} = \begin{bmatrix} -2y+5t \\ y \\ -3t \\ t \end{bmatrix} = \text{y} \begin{bmatrix} -2 \\ 1 \\ 0 \\ 0 \end{bmatrix} + \text{t} \begin{bmatrix} 5 \\ 0 \\ -3 \\ 1 \end{bmatrix}$$

z = -3t

x = -2y-2z-t=-2y+5t

The solution vector is Linear combination of two independent vectors. The right null space basis set , therefore, is

$$\left\{ \begin{bmatrix} -2 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 5 \\ 0 \\ -3 \\ 1 \end{bmatrix} \right\}$$

Note that row-space dimension is 2. It is a subspace of R^4.

Right nullsapce dimension is 4-2=2

Assignment

1. Find rowspace basis set and right null space basis set of the following matrix

$$A = \begin{bmatrix} 1 & 2 & 2 & 1 & 1 \\ 1 & 3 & 1 & 2 & 2 \\ 4 & 7 & 5 & 4 & 4 \end{bmatrix}$$

SOLUTION

Let A be the matrix

$$A = \begin{bmatrix} 1 & 2 & 2 & 1 & 1 \\ 1 & 3 & 1 & 2 & 2 \\ 4 & 7 & 5 & 4 & 4 \end{bmatrix}$$

We aim to find the basis for the row space and the right null space of the matrix A. To begin, we compute the reduced row echelon form (RREF) of A. Performing elementary row operations on A, we obtain:

$$rref(A) = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

From this reduced form, we observe that the pivot columns are the 1st, 2nd, and 3rd columns. Hence, the rank of A is 3. The 4th and 5th columns correspond to free variables.

The row space of A is spanned by the non-zero rows of $rref(A)$, which correspond to the non-zero rows of A. Therefore, the basis for the row space is given by the following vectors from the original matrix:

$$\left\{ \begin{bmatrix} 1 & 2 & 2 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 3 & 1 & 2 & 2 \end{bmatrix}, \begin{bmatrix} 4 & 7 & 5 & 4 & 4 \end{bmatrix} \right\}$$

Next, we solve for the right null space, which consists of all vectors $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{bmatrix}^T$ such that $A\mathbf{x} = 0$. The RREF form of A yields the following system of equations:

$$x_1 + x_4 = 0$$
$$x_2 - x_4 = 0$$
$$x_3 + x_5 = 0$$

From these equations, we express the leading variables x_1, x_2, and x_3 in terms of the free variables x_4 and x_5:

$$x_1 = -x_4$$
$$x_2 = x_4$$
$$x_3 = -x_5$$

Thus, the general solution for the null space is:

$$\mathbf{x} = x_4 \begin{bmatrix} -1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} + x_5 \begin{bmatrix} 0 \\ 0 \\ -1 \\ 0 \\ 1 \end{bmatrix}$$

The vectors corresponding to the free variables x_4 and x_5 form a basis for the null space:

$$\left\{ \begin{bmatrix} -1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ -1 \\ 0 \\ 1 \end{bmatrix} \right\}$$

Thus, the basis for the row space is:

$$\left\{ \begin{bmatrix} 1 \\ 2 \\ 2 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 3 \\ 1 \\ 2 \\ 2 \end{bmatrix}, \begin{bmatrix} 4 \\ 7 \\ 5 \\ 4 \\ 4 \end{bmatrix} \right\}$$

And the basis for the right null space is:

$$\left\{ \begin{bmatrix} -1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ -1 \\ 0 \\ 1 \end{bmatrix} \right\}$$

2. For the above matrix find column-space and left null space basis set. (Hint: For the Transposed matrix find column and right null space basis set.)

SOLUTION

Consider the matrix:

$$A = \begin{bmatrix} 1 & 2 & 2 & 1 & 1 \\ 1 & 3 & 1 & 2 & 2 \\ 4 & 7 & 5 & 4 & 4 \end{bmatrix}$$

We begin by calculating the transpose of A, denoted A^T:

$$A^T = \begin{bmatrix} 1 & 1 & 4 \\ 2 & 3 & 7 \\ 2 & 1 & 5 \\ 1 & 2 & 4 \\ 1 & 2 & 4 \end{bmatrix}$$

To determine the column space of A, the reduced row echelon form (RREF) of A^T is computed as:

$$rref(A^T) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

The pivot rows in the RREF of A^T correspond to the 1st, 2nd, and 3rd rows. Hence, the corresponding columns of A provide a basis for the column space of A.

Thus, the basis for the column space of A is given by the 1st, 2nd, and 3rd columns of A:

$$\left\{ \begin{bmatrix} 1 \\ 1 \\ 4 \end{bmatrix}, \begin{bmatrix} 2 \\ 3 \\ 7 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \\ 5 \end{bmatrix} \right\}$$

Next, the left null space of A is determined by finding the right null space of A^T, which corresponds to the free variables in $rref(A^T)$.

From the RREF of A^T, there are no free variables, the right null space of A^T is empty.

Thus, the basis for the left null space of A contains only the zero vector. This can be verified using the rank nullity theorem for A^T.

Therefore, the final results are:

- The basis for the column space of A is:

$$\left\{ \begin{bmatrix} 1 \\ 1 \\ 4 \end{bmatrix}, \begin{bmatrix} 2 \\ 3 \\ 7 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \\ 5 \end{bmatrix} \right\}$$

- The basis for the left null space of A is empty.

RESULTS

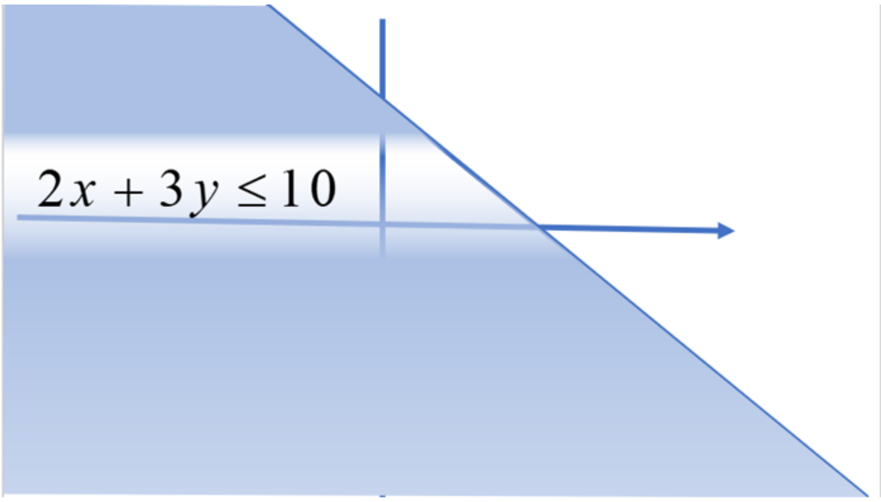
1. Basics of the fundamental subspaces is revisited.
2. Using free variables, basis for right null space is created.
3. Big picture of fundamental subspaces is used to find basis of all subspaces of the given matrix, A.

8 | Assignment 40 Plotting Linear Inequality Constraints

8.1 Plotting Basics

We are about to deal with inequality constraints in Optimization Theory.

Each inequality divides search space into two halves. One half where the condition is satisfied and the other half that does not satisfy. For example, linear inequality in 2D divides $\mathbb{R}^2$ space as in the following figure.



The intersection of all the constraints define a set of points where all the conditions are met. Our aim is draw such regions for 2D (two variable) and 3D problems.

8.1.1 `plotregion` for plotting regions

The `plotregion` function in MATLAB is used to plot convex closed regions in 2D or 3D. This function is particularly useful for visualizing regions defined by linear inequalities.

Let us start with plotting common regions enclosed by two variable linear constraints.

Plot the region enclosed by the following linear constraints.

$x + y \leq 10; y \leq 8; x \leq 8; y \geq 0; x \geq 0$

Note that *plotregion* requires that the formulation of the problem in the following format.

So we rewrite our constraints as follows

$\mathrm{x} = \mathrm{y} \leq 10 \Rightarrow -x - y \geq -10$

y$\leq 8 \Rightarrow -y \geq -8$

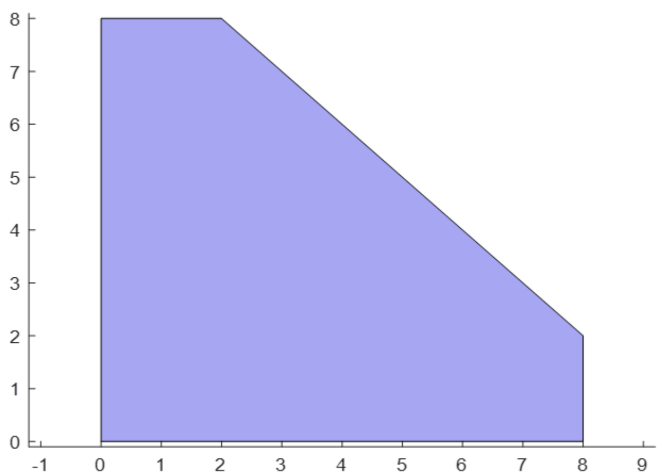
x$\leq 8 \Rightarrow -x \geq -8$

x≥ 0

y ≥ 0

Accordingly we obtain,

$$\begin{bmatrix} -1 & -1 \\ 0 & -1 \\ -1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$
$$\begin{bmatrix} x \\ y \end{bmatrix}$$
$$\geq \begin{bmatrix} -10 \\ -8 \\ -8 \\ 0 \\ 0 \end{bmatrix}$$



Code for above diagram

```
%inequality 2d simplex

close all

A=[-1 0 -1 1 0;-1 -1 0 0 1]';

b=[-10 -8 -8 0 0]';

lb=[];ub=[];

plotregion(A,b,lb,ub,[0.3,0.3,0.9],0.5);

axis equal
```

8.1.2 Ploting feasible region of an LPP

Assignment 1

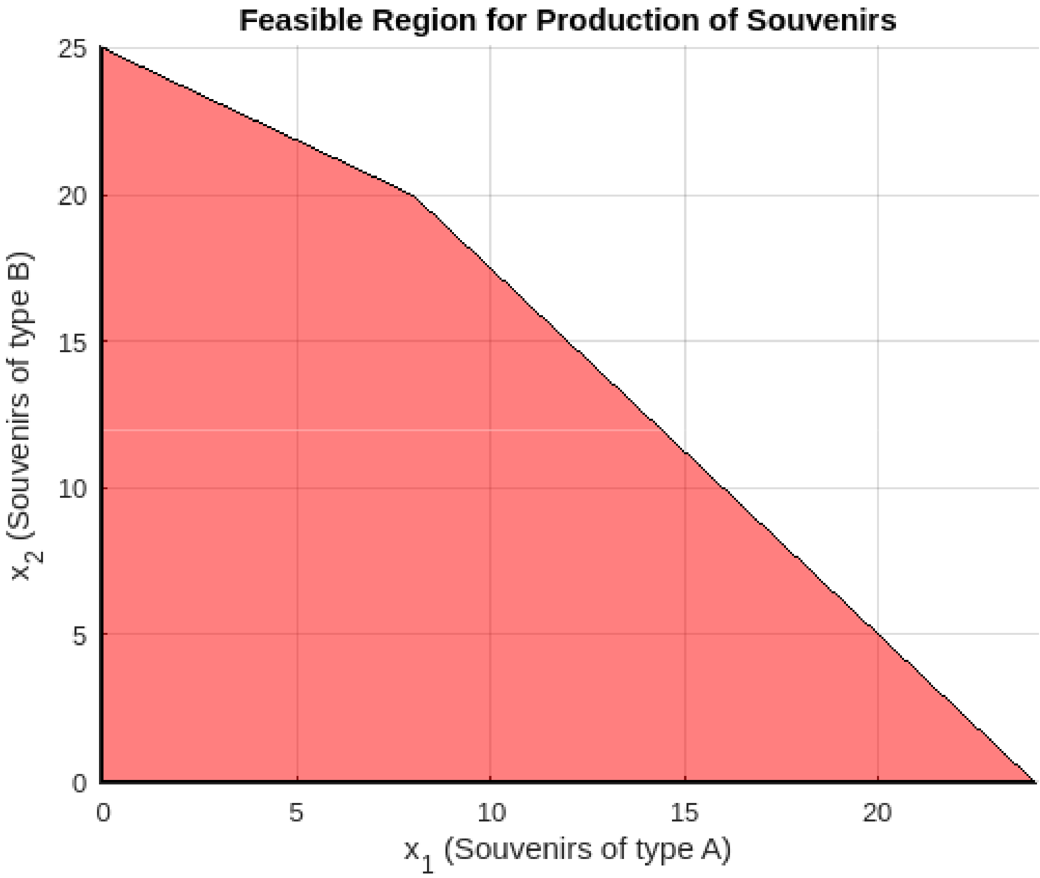
A company manufactures two types of novelty souvenirs made of plywood. Souvenirs of type A require 5 minutes each for cutting and 10 minutes each for assembling. Souvenirs of type B require 8 minutes each for cutting and 8 minutes

each for assembling. There are 3 hours 20 minutes available for cutting and 4 hours for assembling. Express the constraints on number of different types of souvenirs that can be produced. Plot it using plotregion.

SOLUTION

`Matlab` code for this task and the out of the code is given below.

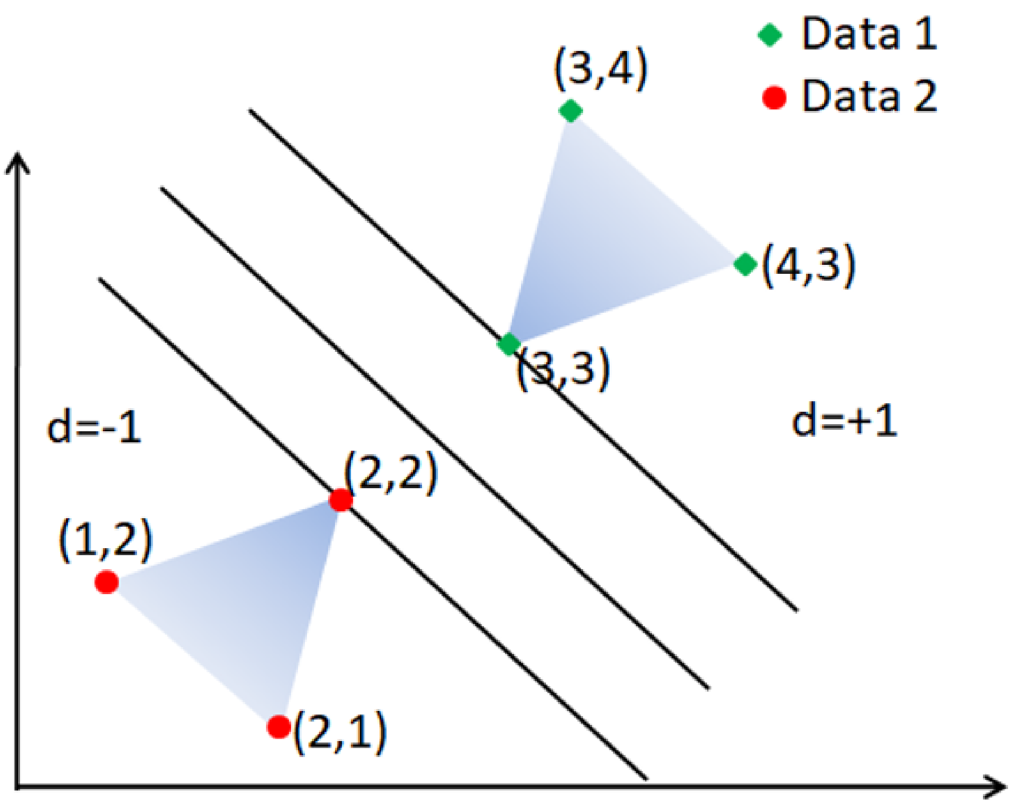
```
% Coefficients of inequalities
A = [-5 -8; -10 -8; 1 0; 0 1];
b = [-200; -240; 0; 0];

% Plot the feasible region
figure;
plotregion(A, b, [], [], 'r', 0.5);

% Add labels and title
xlabel('x_1 (Souvenirs of type A)');
ylabel('x_2 (Souvenirs of type B)');
title('Feasible Region for Production of Souvenirs');
grid on;
```



Linear inequalities and support vector machines

Support vector machines are a group of Algorithm developed in 1990s that marked second stage of developments of Machine learning algorithms. It could tackle the problem of overfitting the data much more elegant way than classical methods like Decision tress(DT) and Backpropagation neural networks(BPN). It achieved it through the concept of maximum margin separating planes. This gave good generalization capability to SVM classifiers.

Plot the range of parameters of the line, that define the set of all lines that separate the two groups of data



Let $w_1x_1 + w_2x_2 - \gamma = 0$ is a line (like the central line in the figure) that separate the two groups. There are infinitely many such lines with different (w_1, w_2, γ) parameters . Our aim is to plot set of all such (w_1, w_2, γ) triplets.

The set of points marked green should satisfy the constraint $w_1x_1 + w_2x_2 - \gamma \le 0$

Substituting point coordinates, we get following inequalities

$$\begin{aligned} 2w_1 + 2w_2 - \gamma \le 0 \\ 1w_1 + 2w_2 - \gamma \le 0 \\ 2w_1 + 1w_2 - \gamma \le 0 \end{aligned}$$

The set of points marked greenshould satisfy the constraint $w_1x_1 + w_2x_2 - \gamma \ge 0$

Substituting point coordinates, we get following inequalities

$$\begin{aligned} 3w_1 + 3w_2 - \gamma \ge 0 \\ 3w_1 + 4w_2 - \gamma \ge 0 \\ 4w_1 + 3w_2 - \gamma \ge 0 \end{aligned}$$

We can express every constraints in the form of same type of inequality (say, all greater than zero) by multiplying each inequality in the first set by (-1).

Thus in the matrix format, all constraints are expressed as

$$\begin{bmatrix} -2 & -2 & 1 \\ -1 & -2 & 1 \\ -2 & -1 & 1 \\ 3 & 3 & -1 \\ 3 & 4 & -1 \\ 4 & 3 & -1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \gamma \end{bmatrix} \ge \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}; \text{It is of the form } Az \ge 0$$

Let us plot the region specified by these constraints using plotregion.

Note that *plotregion* requires that the formulation of the problem in the following format.

Az ≥ 0;

$$1b \leq z \leq \text{ub}$$

```
% Coefficients of inequalities
A = [-2 -2 1; -1 -2 1; -2 -1,1; 3 3 -1; 3 4 -1; 4 3 -1];
b = [0;0;0;0;0;0];

% Plot the feasible region
figure;
plotregion(A, b, [], [], [0.3,0.3,0.9], 0.5);

% Add labels and title
xlabel('w_1');
ylabel('w_2');
title('SVM -support vectors');
grid on;
```

RESULTS

The `Matlab` function `plotregion` is used to visualize the feasible region generated by the constraint sets in an LPP.

9 | Assignment 41 Specifying Row-space/column-space by an Equation or by a Set of Equations.

9.1 General Representation for Fundamental Subspaces

9.1.1 Background thoughts

Problem: Row-space by definition is set of all LC of row-vectors. Also it can be defined as space spanned by any **row-space** basis set. All are identical.

We specify null space (right) of a matrix by saying that it is a set of all $x|Ax = 0$ vector (read x such that $Ax = 0$).

Another way of looking at is in terms of orthogonality. We can say, right null space is the set of all vectors that are orthogonal to row-space.

Similarly, we can define left null space by the set of all

y|y^T A=0 rowvector

y|A^T y=0 vector

Can we define row-space in a similar way? Yes.

We use basis set of right null space for the task.

Let B =null(A). Columns of B form a basis set for right null space.

Or rows of B^T form the basis set for right null space. x|B^T x=0 vector

Similarly let C = null(A^T) . Columns of C form basis set for left null space

So we define column space as y|C^Ty = 0.

Another important Note

All the spaces associated with a square matrix of size $n \times n$. if they exist , are subspaces of R^n.

For a rank-2 3x3 matrix , rowspace and columns space are planes passing through origin. Also left and right null space are a line of points through origin.

Can we plot these using the matlab tool 'plotregion'.

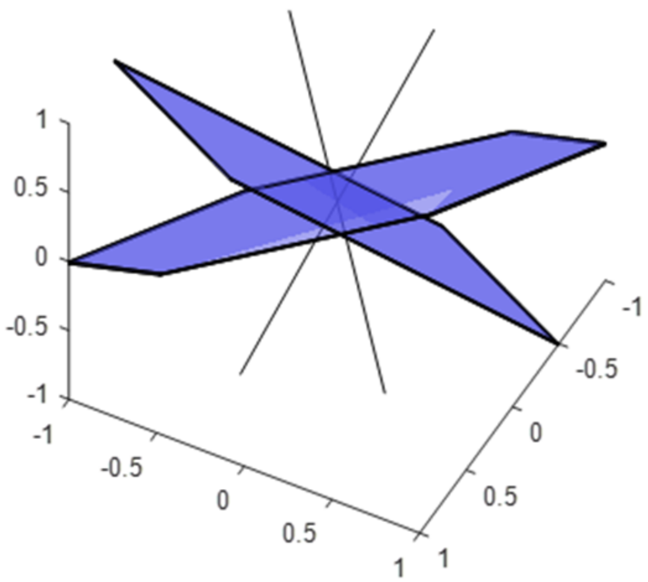
But note that 'plotregion' is basically meant for plotting 'common region' of inequality constraints and points.

Can we use it for the task of plotting planes and lines. Yes.

Following lines of code does the job. Can you figure out how?.

Read the previous assignment on plotting inequalities and thin planes.

```
close(gcf)
A=[1 2 3;3 1 1; 4 3 4];
B=(null(A))';
B=[B; -B];
b=[0;-0.01];
lb=[-1 -1 -1]';ub=[1 1 1]';
plotregion(B,b,lb,ub,[0.3,0.3,0.9],0.5,B');
hold on
D=A';
C=(null(D))';
C=[C; -C];
plotregion(C,b,lb,ub,[0.3,0.3,0.9],0.5,C');
axis equal
hold off
```



Assignment.

1. What is the equation for the row-space of the following matrix

$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \end{bmatrix}$.Use row-elimination to get right null-space vector.

SOLUTION

Given matrix is first transformed into its RREF as follows.

$$
\begin{aligned}
A &= \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \end{bmatrix} \\
&\sim \begin{bmatrix} 1 & 2 & 3 \\ 0 & -3 & -4 \end{bmatrix} \\
&\sim \begin{bmatrix} 1 & 0 & \frac{1}{3} \\ 0 & 1 & \frac{4}{3} \end{bmatrix} \\
&\text{in RREF}
\end{aligned}
$$

From the RREF of A, it is clear that pivot elements are in first and second columns and all the two rows are non-zero. Hence rank of A is 2. The null space is the solution of $Ax = 0$. Using the RREF, it is clear that last variable (say) x_3 is a free variable. In terms of x_3 solution of the reduced system $Ax = 0$ gives,

$$
\begin{aligned}
x_1 &= -\frac{1}{3}x_3 \\
x_2 &= -\frac{4}{3}x_3 \\
x_3 &= x_3
\end{aligned}
$$

Hence the right null space of A is spanned by $\left\{ \begin{bmatrix} -\frac{1}{3} \\ -\frac{4}{3} \\ 1 \end{bmatrix} \right\}$.

In Gilbert Strang's approach, once we know the basis of the column space and the row reduced Echelon form of A, then A can be decomposed as $A = CR \quad REF(A)$. Also using the Kronecker product, A can be written as sum of two matrices;

$$A = \sum_{j=1}^{2} A[:, j] \otimes \text{RREF}(A)[j,:]$$

Let $A = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$ Note that A is orthogonal matrix. Both column vectors and row vectors are pairwise orthogonal.

Let $B = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \end{bmatrix}$ Note that it is a submatrix of A. write down equation for right row-space of B, without doing any computation.

SOLUTION

Given that A is an orthogonal matrix and B is a submatrix of A, the right row-space of B can be described using the orthogonality property of A.

Since A is orthogonal, its rows form an orthonormal basis. Therefore, any submatrix B of A will have rows that are linear combinations of the rows of A. The right row-space of B is thus spanned by the corresponding rows of A.

In equation form, if B consists of rows $i_1, i_2, \ldots, i_k$ of A, then the right row-space of B is given by:

$$\text{RowSpace}(B) = \text{span}\{\mathbf{a}_{i_1}, \mathbf{a}_{i_2}, \ldots, \mathbf{a}_{i_k}\}$$

where $\mathbf{a}_{i_j}$ are the rows of A.

Plot all spaces associated with matrix $A = \begin{bmatrix} 1 & 1 \\ 2 & 3 \end{bmatrix}$

SOLUTION

Matlab code for this task and its output is given below.

```
A = [1 2; 2 3];
B = (null(A))';
B = [B; -B];
b = [0; -0.01];
lb = [-1 -1]';
ub = [2 3]';

% Ensure plotregion function is defined and works correctly
try
    plotregion(B, b, lb, ub, [0.3, 0.3, 0.9], 0.5, B');
    hold on;
catch ME
    disp(['Error in plotregion for B: ', ME.message]);
end

D = A';
C = (null(D))';
C = [C; -C];

try
    plotregion(C, b, lb, ub, [0.3, 0.3, 0.9], 1, C');
    axis equal;
    hold off;
catch ME
    disp(['Error in plotregion for C: ', ME.message]);
end
```

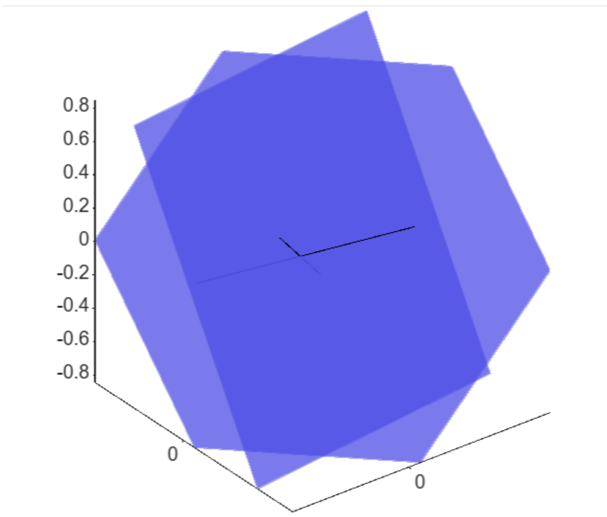


Figure 9.1: Visualization of Subspaces of A

RESULTS

1. Connection between fundamental subspaces and solution of equations is revisited.

2. Null space of matrices are visualized.

10 Assignment-42 Eigenvectors And Orthogonality Concepts

10.1 Core Concept Behind the Eigen Values and Eigen Vectors

10.1.1 From coordinate geometry to orthonormal vectors

Concept of Orthogonality of vectors is very important in Science and engineering. High-school co-ordinate geometry start with orthogonal axes (x-axis and y-axis) . High-school vector algebra begin with 3 orthogonal unit vectors $\hat{i}, \hat{j}, \hat{k}$. Complex numbers and associated concepts are again built over orthogonal axes.

10.1.2 Role of orthogonal decomposition in Engineering applications

All the classical signal decomposition techniques like Fourier and wavelet theory are based on orthogonality of functions. Many Communication Engineering techniques like OFDM depends on orthogonality. Electromagnetic wave itself is composed of orthogonal electrical and magnetic waves.

In nutshell, orthogonality concept is embedded everywhere. Your eyes (mind) should be trained to see it.

10.1.3 Matrix multiplication as scaling or rotation

In LA, eigen vector computation can be seen as finding orthogonal vectors.

Let us revise again. To give a mental picture about what eigenvalue and eigenvector, we interpret '*matrix vector'* multiplication Ax as a mapping. If A is a square matrix, it does a mapping (transformation) from to when acting (multiplying) on a vector. That means if you give x from to the 'operator' A , as input, it produces another vector y in . The ' y' in general is a scaled and rotated version of 'x'. There can be special directions for each A, in which, it does only scaling. Such independent directions represented as vectors are called eigenvectors. The scaling (multiplying) factor associated with eigen vectors are called eigenvalues.

The finding of such eigenvectors can be linked to the concept of orthogonality. It is a crucial change in view point.

Matrix A, which we considered as an operator, is in fact a set of ordered vectors (column vectors, or row vectors). So, the eigen vector must have some connection with these vectors.

We have

Ax = $\lambda x \Rightarrow (A - \lambda I)$x = 0 vector

To get a better view, let us consider 3x3 matrix

let A = $\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$

$(A-\lambda I)\,x = \begin{bmatrix} a-\lambda & b & c \\ d & e- & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

Now look for orthogonality concept that is hidden here.

The unknown vector x must be orthogonal to the rows of the new derived matrix $(A-\lambda I)$. That is eigenvector is orthogonal to rowspace of $(A-\lambda I)$.

For nonzero x to exist, one important thing should happen. There must be special λ values that makes rows of $(A-\lambda I)$ dependent. That is $(A-\lambda I)$ must be singular. If singular, $|A-\lambda I|$. That is how we compute eigenvalues for small matrices.

Once we know eigenvalues, computation of eigen vector can be seen as **finding orthogonal vectors**. Gaussian elimination on A -λI = 0 can easily give that vector.

For each λ we get one eigen vector

Let us start with 2x2 matrices

Let $A = \begin{bmatrix} 1 & 4 \\ 2 & -1 \end{bmatrix}$

For a 2×2 matrix, $A = \begin{matrix} a & b \\ c & d \end{matrix}$, the shortcut formula for the characteristic polynomial is

$$\lambda^2 - \text{trace of}(A) - |A| = 0$$

First Find eigen values

Each column sum is 3. Hence 3 is an eigen value. Based on trace, we find next eigen value is λ_1 - 3.

Consider

$A-\lambda_1 I$ = A - 3I= $\begin{bmatrix} 1-3 & 4 \\ 2 & -1-3 \end{bmatrix} = \begin{bmatrix} -2 & 4 \\ 2 & -4 \end{bmatrix}$

Note that second row is multiple of first row. Hence we need not consider it for finding an orthogonal vector to rowspace.

Now, null space vector that is orthogonal to first row is $\begin{bmatrix} 4 \\ 2 \end{bmatrix}$ or any multiple of $\begin{bmatrix} 2 \\ 1 \end{bmatrix}$ which can be inferred from first row.

Consider $\lambda_{2\,=-3}$

$A - \lambda_2 I = A + 3I \begin{bmatrix} 1+3 & 4 \\ 2 & -1+3 \end{bmatrix} = \begin{bmatrix} 4 & 4 \\ 2 & 2 \end{bmatrix}$

Note again that second row is multiple of first row.

null space vector is $\begin{bmatrix} 4 \\ -4 \end{bmatrix}$ or any multiple of $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$ which can be inferred from first row.

Thus in general, the moment you substitute $lambda$, rows become dependent and hence give space for right nullspace vector to appear.

An intelligent student always find general rule for saving time and memory.

(If you think that you are intelligent try doing it)

let $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ and λ_1,λ_2 be the eigenvalues of A.

Then eigen vector corresponding to λ_1 is $\begin{bmatrix} -b \\ a - \lambda_1 \end{bmatrix}$

Eigen vector corresponding to $\lambda_2 \begin{bmatrix} -b \\ a - \lambda_2 \end{bmatrix}$

Assignment Q1

Find eigenvalues and eigenvectors of

$$A = \begin{bmatrix} 2 & 4 \\ 3 & 3 \end{bmatrix}; B = \begin{bmatrix} 2 & 1 \\ 5 & 6 \end{bmatrix}; C \begin{bmatrix} 1 & -1 \\ -2 & 2 \end{bmatrix}$$

SOLUTION

1. Here given matrix is $A = \begin{bmatrix} 2 & 4 \\ 3 & 3 \end{bmatrix}$

   Eigen values are the solutions of $|A - \lambda I| = 0$. Using the shortcut formula,

$$\lambda^2 - \text{trace}(A) - |A| = 0$$

   , the characteristic polynomial is

$$\lambda^2 - 5\lambda - 6 = 0$$
$$(\lambda - 6)(\lambda + 1) = 0$$
$$\implies \lambda_1 = 6 \text{and } \lambda_2 = -1$$

   Using the shortcut method;

$$EV(\lambda_1 = 6) = \begin{bmatrix} 6-3 \\ 3 \end{bmatrix}$$
$$= \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$
$$= \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$
$$EV(\lambda_2 = -1) = \begin{bmatrix} -1-3 \\ 3 \end{bmatrix}$$
$$= \begin{bmatrix} -4 \\ 3 \end{bmatrix}$$

2. Here given matrix is $A = \begin{bmatrix} 2 & 1 \\ 5 & 6 \end{bmatrix}$

   Eigen values are the solutions of $|A - \lambda I| = 0$. Using the shortcut formula,

$$\lambda^2 - \text{trace}(A) - |A| = 0$$

   , the characteristic polynomial is

$$\begin{aligned} \lambda^2 - 8\lambda + 7 = 0 \\ (\lambda - 7)(\lambda - 1) = 0 \\ \implies \lambda_1 = 7 \text{and } \lambda_2 = 1 \end{aligned}$$

   Using the shortcut method;

$$\begin{aligned} EV(\lambda_1 = 7) &= \begin{bmatrix} 7-6 \\ 5 \end{bmatrix} \\ &= \begin{bmatrix} 1 \\ 5 \end{bmatrix} \\ EV(\lambda_2 = 1) &= \begin{bmatrix} 1-6 \\ 5 \end{bmatrix} \\ &= \begin{bmatrix} -5 \\ 5 \end{bmatrix} \\ &= \begin{bmatrix} -1 \\ 1 \end{bmatrix} \end{aligned}$$

3. Here given matrix is $A = \begin{bmatrix} 1 & -1 \\ -2 & 2 \end{bmatrix}$

   Eigen values are the solutions of $|A - \lambda I| = 0$. Using the shortcut formula,

$$\lambda^2 - \text{trace}(A) - |A| = 0$$

   , the characteristic polynomial is

$$\begin{aligned} \lambda^2 - 3\lambda + 0 = 0 \\ \lambda(\lambda - 3) = 0 \\ \implies \lambda_1 = 3 \text{and } \lambda_2 = 0 \end{aligned}$$

   Using the shortcut method;

$$\begin{aligned} EV(\lambda_1 = 3) &= \begin{bmatrix} 3-2 \\ -2 \end{bmatrix} \\ &= \begin{bmatrix} 1 \\ -2 \end{bmatrix} \\ EV(\lambda_2 = 0) &= \begin{bmatrix} 0-2 \\ -2 \end{bmatrix} \\ &= \begin{bmatrix} -2 \\ -2 \end{bmatrix} \\ &= \begin{bmatrix} 1 \\ 1 \end{bmatrix} \end{aligned}$$

```
% first question
A=[2 4; 3 3];
[Evec,Evals]=eig(A);
disp('Eigen values:')
```

```
Eigen values:
```

```
disp(diag(Evals));
```

```
      -1
       6
```

```
disp('Eigen vectors');
```

```
Eigen vectors
```

```
disp(Evec)
```

```
      -4/5          -985/1393
       3/5          -985/1393
```

```
%second question
A2=[2 1; 5 6];
[Evec,Evals]=eig(A2);
disp('Eigen values:')
```

```
Eigen values:
```

```
disp(diag(Evals));
```

```
       1
       7
```

```
disp('Eigen vectors');
```

```
Eigen vectors
```

```
disp(Evec)
```

```
    -985/1393     -1020/5201
     985/1393     -5100/5201
```

```
%third question
A3=[1 -1; -2 2];
[Evec,Evals]=eig(A3);
disp('Eigen values:')
```

```
Eigen values:
```

```
disp(diag(Evals));
```

```
       0
       3
```

```
disp('Eigen vectors');
```

```
Eigen vectors
```

```
disp(Evec)
```

```
    -985/1393      1292/2889
    -985/1393     -2584/2889
```

RESULTS

1. Revisited the intuition behind the eigen values and eigen vectors along with theoretical definitions
2. Eigen values and eigen vectors of given problems are evaluated and verified computationally

11 Assignment 43 Linking Optimization With Linear Algebra

11.1 Analytical Tools for Muti-variable Optimization

11.1.1 Role of Gradient and Hessian of a multivariate function

Given a smooth function $f : \mathbb{R}^n \to \mathbb{R}$, we can write a second order Taylor expansion in the form:

$$f(x+\Delta x) = f(x) + (\nabla f(x))^\top \Delta x + \frac{1}{2}(\Delta x)^\top Hf(x)\Delta x + O(|\Delta x|^3)$$

where $\nabla f(x)$ is the gradient of f at x , $Hf(x)$ is the Hessian matrix of f at x , and Δx is some small displacement. Note that since the Hessian matrix is symmetric, then it has n (possibly repeated) real eigenvalues, and in particular it is diagonalizable as a real matrix. This is the heart of multi-variate function optimization with strong roots in calculus.

Suppose the function has a critical point at $x = a$, so that $\nabla f(x) = 0$. Then the Taylor expansion looks like:

$$f(a+\Delta x) = f(a) + \frac{1}{2}(\Delta x)^\top Hf(a)\Delta x + O(|\Delta x|^3).$$

Thus, for small displacements Δx , the Hessian tells us how the function behaves around the critical point.

Consider a rectangular domain as shown in figure 1 with centre as origin.

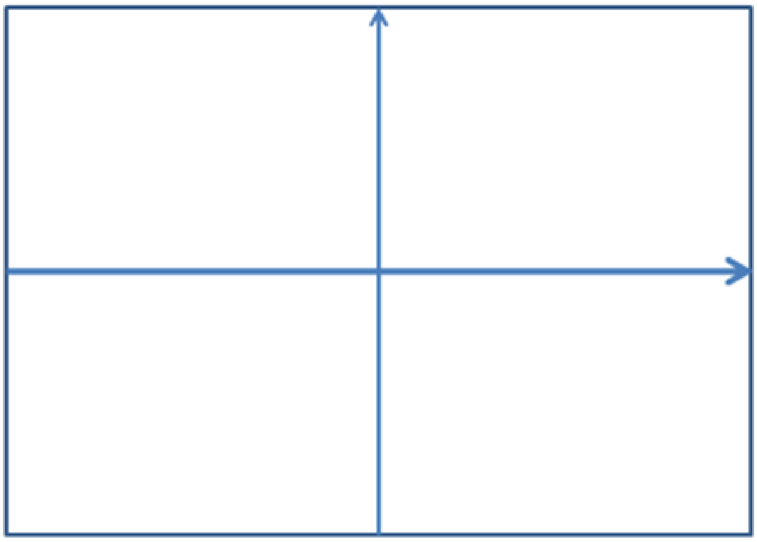


Figure 1. Domain of a function

Over the domain, we can define many functions such as $f(x, y) = x^2 + y^2$, or $f(x, y = 1/(|x| + |y| + 0.5))$. A function assign a numerical value at each point depending on its coordinates in the domain. Usually we plot that value in another dimension as in figure 2 and 3.

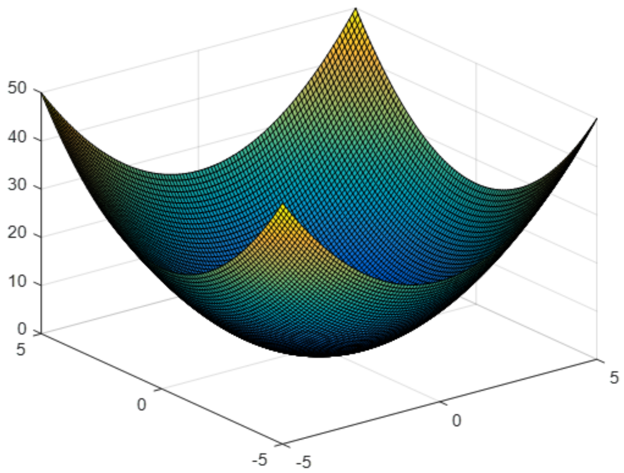


Figure 2 plot of $f(x, y) = x^2 + y^2$

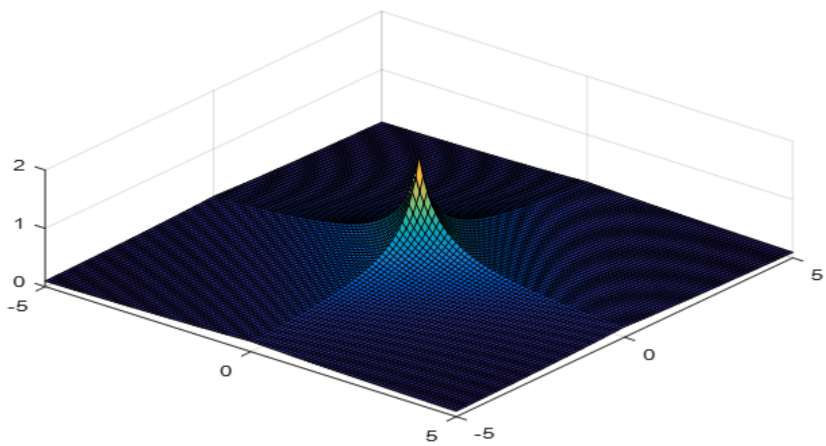


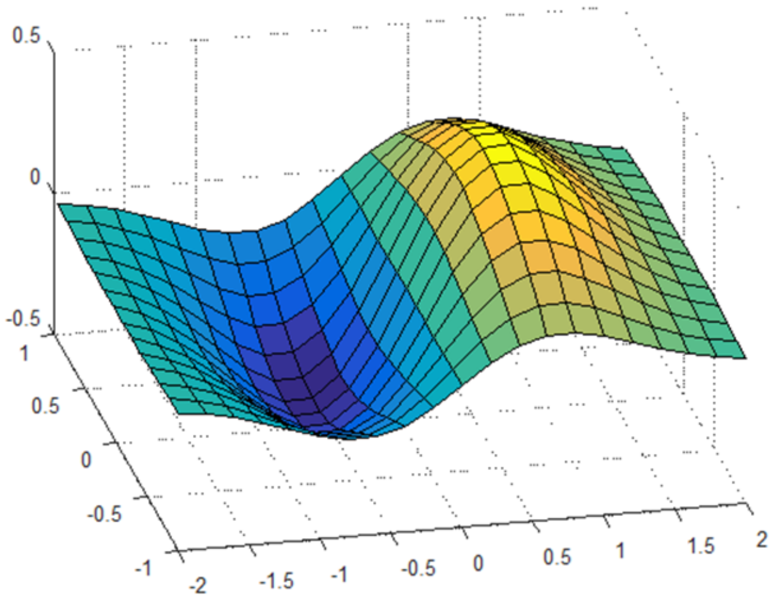
Figure 3 plot of $f(x, y = 1/(|x| + |y| + 0.5))$

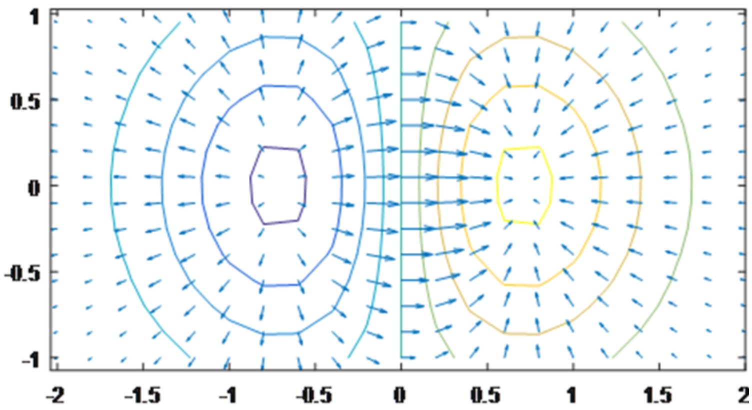
For human beings, the only dimension left is the third dimension. But we have another method that save a dimension. Use color intensity to designate the value as in figure 4. which gives an intuitive idea about the variation of the function in the domain

Figure 4. visualizing function in terms of colors.

Still another way to deal with function visualization is gradient (with contour) and hessian of a function at a given point. For 2D domain , we can plot the gradient but hessian is a matrix whose sign of eigenvalues tell about the type of second order curvature at the point.

```
[x,y] = meshgrid(-2:.2:2,-1:.15:1);

z = x .* exp(-x.^2 - y.^2);

figure

surf(x,y,z);

figure

[px,py] = gradient(z,.2,.15);

contour(x,y,z), hold on

quiver(x,y,px,py), hold off, axis image
```





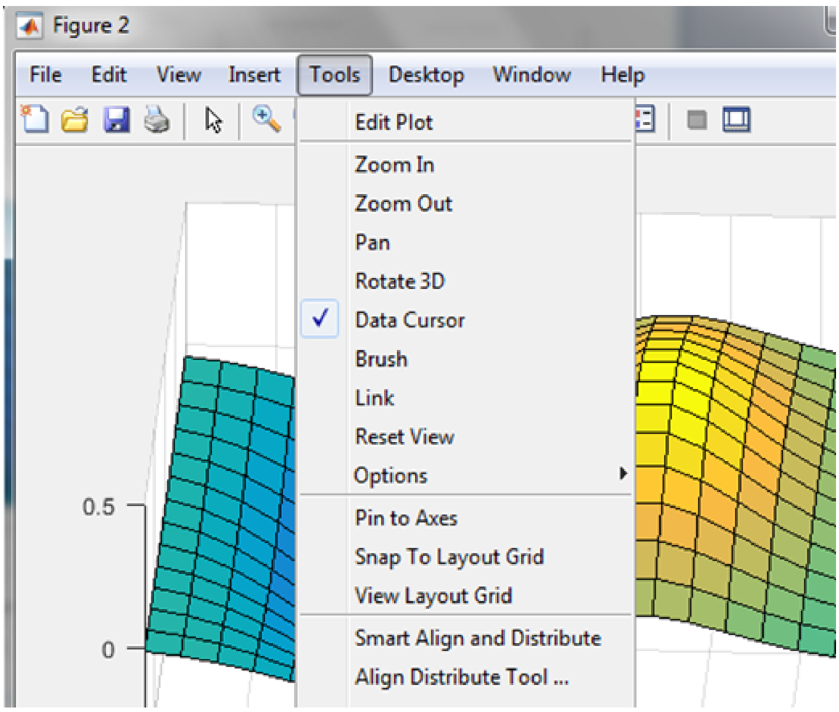
11.2 Geometrical Approach to Idenitify Nature of a Point on $z = f(x, y)$

Use of data curser

11.2.1 Use of surface plots and gradient

Under tools there is a utility that allows you to find data values(co-ordinates, gradient values, contour value etc)

Learn how to use this tool.

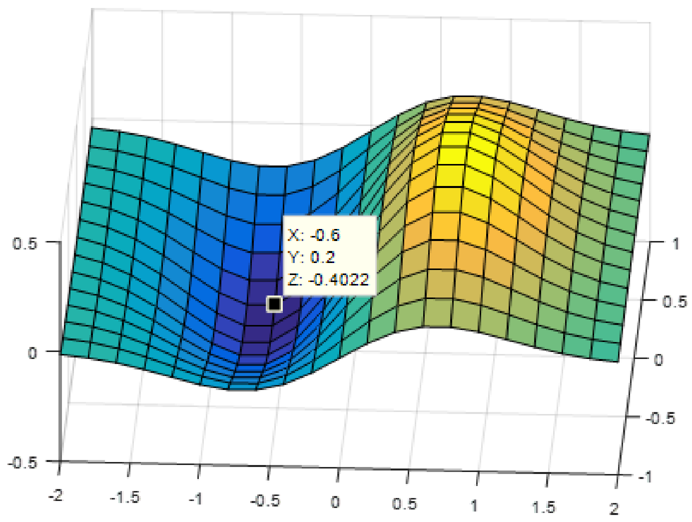


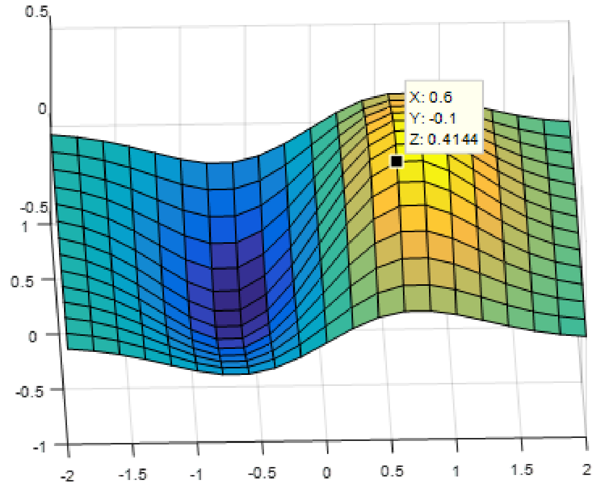
Assignment Question

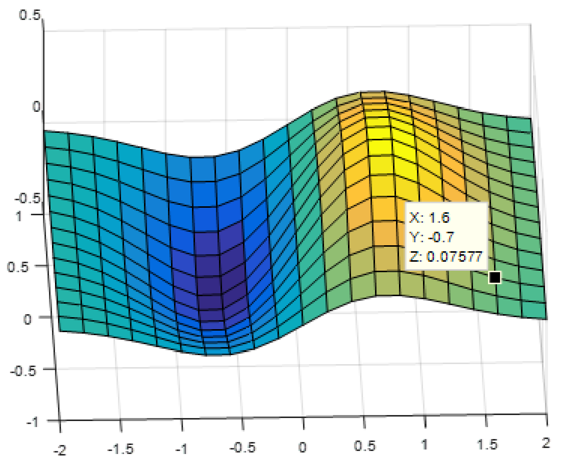
The function plotted below is $f(x, y) = xe^{-(x^2+y^2)}$

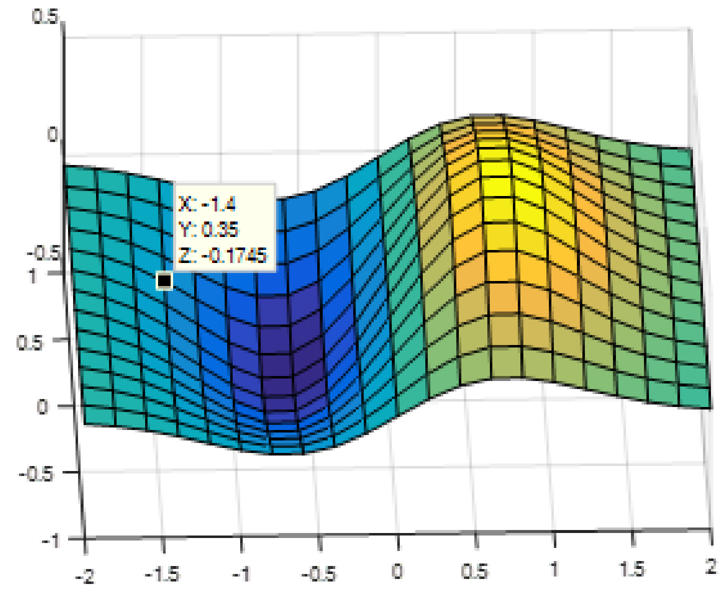
1. Write the General expression for gradient and Hessian for the function
2. Evaluate hessian at the points given in the following figures
3. Find eigenvalues of those hessian matrices and decide the nature of curvature at the points where it is evaluated. (curved upward, downward, neither)

4. Read the blog post on hessian (pdf attached)









SOLUTION

1. Let the function be $f(x, y) = xe^{-(x^2+y^2)}$.

   The gradient of $f(x, y)$ is given by:

$$\nabla f(x, y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

   First, compute the partial derivatives. The partial derivative with respect to x is:

$$\frac{\partial f}{\partial x} = \frac{\partial}{\partial x}\left(xe^{-(x^2+y^2)}\right) = e^{-(x^2+y^2)} + x \cdot \frac{\partial}{\partial x}\left(e^{-(x^2+y^2)}\right)$$

   Since $\frac{\partial}{\partial x}\left(e^{-(x^2+y^2)}\right) = -2xe^{-(x^2+y^2)}$, we get:

$$\frac{\partial f}{\partial x} = e^{-(x^2+y^2)} - 2x^2 e^{-(x^2+y^2)} = \left(1 - 2x^2\right) e^{-(x^2+y^2)}$$

   Next, compute the partial derivative with respect to y:

$$\frac{\partial f}{\partial y} = \frac{\partial}{\partial y}\left(xe^{-(x^2+y^2)}\right) = x \cdot \frac{\partial}{\partial y}\left(e^{-(x^2+y^2)}\right)$$

   Since $\frac{\partial}{\partial y}\left(e^{-(x^2+y^2)}\right) = -2ye^{-(x^2+y^2)}$, we have:

$$\frac{\partial f}{\partial y} = -2xye^{-(x^2+y^2)}$$

   Thus, the gradient is:

$$\nabla f(x, y) = \left(\left(1 - 2x^2\right) e^{-(x^2+y^2)}, -2xye^{-(x^2+y^2)}\right)$$

   Now, compute the Hessian matrix, $H(f)$, which consists of the second-order partial derivatives. First, the second partial derivative with respect to x:

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x}\left(\left(1 - 2x^2\right) e^{-(x^2+y^2)}\right) = \frac{\partial}{\partial x}\left(1 - 2x^2\right) e^{-(x^2+y^2)} + \left(1 - 2x^2\right) \frac{\partial}{\partial x}\left(e^{-(x^2+y^2)}\right)$$

The derivative of $1-2x^2$ with respect to x is $-4x$, and $\frac{\partial}{\partial x}\left(e^{-(x^2+y^2)}\right) = -2xe^{-(x^2+y^2)}$. Thus:

$$\frac{\partial^2 f}{\partial x^2} = -4xe^{-(x^2+y^2)} - 2x(1-2x^2)e^{-(x^2+y^2)} = -2x(3-2x^2)e^{-(x^2+y^2)}$$

Next, the mixed partial derivative $\frac{\partial^2 f}{\partial x \partial y}$ is:

$$\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial}{\partial y}\left(\left(1-2x^2\right)e^{-(x^2+y^2)}\right) = \left(1-2x^2\right)\cdot(-2y)e^{-(x^2+y^2)}$$

Thus:

$$\frac{\partial^2 f}{\partial x \partial y} = -2y(1-2x^2)e^{-(x^2+y^2)}$$

Finally, the second partial derivative with respect to y is:

$$\frac{\partial^2 f}{\partial y^2} = \frac{\partial}{\partial y}\left(-2xye^{-(x^2+y^2)}\right) = -2x\cdot\frac{\partial}{\partial y}\left(ye^{-(x^2+y^2)}\right)$$

The derivative of $ye^{-(x^2+y^2)}$ with respect to y is $(1-2y^2)e^{-(x^2+y^2)}$, so:

$$\frac{\partial^2 f}{\partial y^2} = -2x(1-2y^2)e^{-(x^2+y^2)}$$

Thus, the Hessian matrix is:

$$H(f) = \begin{pmatrix} -2x(3-2x^2)e^{-(x^2+y^2)} & -2y(1-2x^2)e^{-(x^2+y^2)} \\ -2y(1-2x^2)e^{-(x^2+y^2)} & -2x(1-2y^2)e^{-(x^2+y^2)} \end{pmatrix}$$

2. The functional evaluation, gradient, Hessian and eigenvalues of Hessian matrix at the given points are evaluated and shown in the following table.

(x,y)	$f(x,y)$	$\nabla f(x,y)$	$H(f)(x,y)$	$EV(H(f)(x,y)$
$(-0.6, 0.2)$	-0.4022	$(0.1877, 0.1609)$	$\begin{pmatrix} 1.834 & -0.0750 \\ -0.0750 & 0.7400 \end{pmatrix}$	$[0.7349, 1.8391]$
$(0.6, -0.1)$	0.4144	$(0.1934, 0.0829)$	$\begin{pmatrix} -1.8898 & 0.0386 \\ 0.0386 & -0.8123 \end{pmatrix}$	$[-1.8912, -0.8109]$
$(1.6, -0.7)$	0.0758	$(-0.1951, 0.1061)$	$\begin{pmatrix} 0.3212 & -0.2731 \\ -0.2731 & -0.0030 \end{pmatrix}$	$[-0.1585, 0.4768]$
$(-1.4, 0.35)$	-0.1745	$(-0.3639, 0.1221)$	$\begin{pmatrix} -0.3210 & 0.2547 \\ 0.2547 & 0.2634 \end{pmatrix}$	$[-0.4164, 0.3589]$

11.2.2 Nature of critical points using eigen values of Hessian matrix

3. Nature of the points can be investigated using the nature of Hessian matrix at these points. From the Taylor series approximation from multi-variable calculus the following results follows.

   - The Hessian $Hf(a)$ is positive definite if and only if $(\Delta x)^\top Hf(a)\Delta x > 0$ for $\Delta x \neq 0$. Equivalently, this is true if and only if all the eigenvalues of $Hf(a)$ are positive. Then no matter which direction you move away from the critical point, the value of $f(a+\Delta x)$ grows (for small $|\Delta x|$), so a is a local minimum.
   - Likewise, the Hessian $Hf(a)$ is negative definite if and only if $(\Delta x)^\top Hf(a)\Delta x < 0$ for $\Delta x \neq 0$. Equivalently, this is true if and only if all the eigenvalues of $Hf(a)$ are negative. Then no matter which direction you move away from the critical point, the value of $f(a+\Delta x)$ grows (for small $|\Delta x|$), so a is a local maximum.

- The Hessian $Hf(a)$ is indefinite if and only if it has at least one positive and at least one negative eigenvalue. Then $f(a+\Delta x)$ either increases or decreases depending on the direction you move away from the critical point so a is a saddle point.
- Finally, the test fails if we have a zero eigenvalue but the rest of the eigenvalues are either all larger than or smaller than 0. In this case we need to use higher order terms to decide if we have a saddle or local max/min.

Using this inter relationship between Calculus and algebra, we can decide the nature of $z = f(x, y)$ at the given points as follows.

Points	EV(H(f)(x,y)	Nature	Conclusion
(-0.6,0.2)	(0.739,1.839)	positive definite	local minimum
(0.6,-0.1)	(-1.189,-0.81)	negative definite	local maximum
(1.6,-0.7)	(-0.158,0.47)	indefinite	saddle point
(-1.4,0.35)	(-0.416,0.358)	indefinite	saddle point

Table 11.1: Nature of $z = f(x, y)$ at given points using eigen values of $H(f)(x, y)$

`Matlab` code and output for this task is given below.

```
syms x y;

% Define the function
f = x * exp(- (x^2 + y^2));

% Compute the gradient
grad_f = gradient(f, [x, y]);

% Compute the Hessian
hess_f = hessian(f, [x, y]);

% Points to evaluate
points = [ -0.6, 0.2;
           0.6, -0.1;
           1.6, -0.7;
           -1.4, 0.35 ];

% Preallocate arrays for results
f_vals = zeros(size(points, 1), 1);
grad_vals = zeros(size(points, 1), 2);
eigen_vals = zeros(size(points, 1), 2);  % Store eigenvalues

% Evaluate at each point
for i = 1:size(points, 1)
    point = points(i, :);
    f_vals(i) = double(subs(f, {x, y}, point));
    grad_vals(i, :) = double(subs(grad_f, {x, y}, point));
    hess_val = double(subs(hess_f, {x, y}, point));

    % Calculate eigenvalues of the Hessian matrix
    eigen_vals(i, :) = eig(hess_val);
end
```

```
% Create a table to display results
result_table = table(points(:,1), points(:,2), f_vals, ...
                     grad_vals(:,1), grad_vals(:,2), ...
                     eigen_vals(:,1), eigen_vals(:,2), ...
                     'VariableNames', {'x', 'y', 'f(x,y)', 'Grad_x', 'Grad_y', 'Eige

% Display the table
disp(result_table);
```

```
x       y       f(x,y)      Grad_x      Grad_y      Eigenvalue_1    Eigenvalue_2
----    ----    --------    --------    --------    ------------    ------------

-0.6     0.2    -0.40219     0.18769     0.16088       0.73491          1.8391
0.6     -0.1     0.41444     0.19341    0.082888       -1.8912        -0.81092
1.6     -0.7    0.075774    -0.19512     0.10608      -0.15854          0.4768
-1.4    0.35    -0.17447    -0.36389     0.12213      -0.41645         0.35887
```

Computational step to provide summary report of a more detailed analysis is given below.

```
syms x y;

% Define the function
f = x * exp(- (x^2 + y^2));

% Compute the gradient
grad_f = gradient(f, [x, y]);

% Compute the Hessian
hess_f = hessian(f, [x, y]);

% Points to evaluate
points = [ -0.6, 0.2;
           0.6, -0.1;
           1.6, -0.7;
           -1.4, 0.35 ];

% Preallocate arrays for results
f_vals = zeros(size(points, 1), 1);
grad_vals = zeros(size(points, 1), 2);
eigen_vals = zeros(size(points, 1), 2);  % Store eigenvalues

% Evaluate at each point
for i = 1:size(points, 1)
    point = points(i, :);
    f_vals(i) = double(subs(f, {x, y}, point));
    grad_vals(i, :) = double(subs(grad_f, {x, y}, point));
    hess_val = double(subs(hess_f, {x, y}, point));

    % Calculate eigenvalues of the Hessian matrix
    eigen_vals(i, :) = eig(hess_val);

    % Create a table for the Hessian matrix
```

```
    hessian_table = table(hess_val(1,1), hess_val(1,2), ...
                          hess_val(2,1), hess_val(2,2), ...
                          'VariableNames', {'Hess_11', 'Hess_12', 'Hess_21', 'Hess_22'});

    % Display the results
    fprintf('At point (%.2f, %.2f):\n', point(1), point(2));
    fprintf('f(x,y) = %.4f\n', f_vals(i));
    fprintf('Gradient = [%.4f, %.4f]\n', grad_vals(i, 1), grad_vals(i, 2));
    fprintf('Eigenvalues: [%.4f, %.4f]\n', eigen_vals(i, 1), eigen_vals(i, 2));
    disp('Hessian Matrix:');
    disp(hessian_table);
    fprintf('\n');
end
```

```
At point (-0.60, 0.20):
f(x,y) = -0.4022
Gradient = [0.1877, 0.1609]
Eigenvalues: [0.7349, 1.8391]
Hessian Matrix:
    Hess_11     Hess_12      Hess_21     Hess_22
    -------    ---------    ---------    -------

     1.834     -0.075076    -0.075076    0.74003
At point (0.60, -0.10):
f(x,y) = 0.4144
Gradient = [0.1934, 0.0829]
Eigenvalues: [-1.8912, -0.8109]
Hessian Matrix:
    Hess_11    Hess_12     Hess_21     Hess_22
    -------    --------    --------    -------

    -1.8898    0.038681    0.038681    -0.8123
At point (1.60, -0.70):
f(x,y) = 0.0758
Gradient = [-0.1951, 0.1061]
Eigenvalues: [-0.1585, 0.4768]
Hessian Matrix:
    Hess_11    Hess_12     Hess_21      Hess_22
    -------    --------    --------    ---------

    0.32128    -0.27317    -0.27317    -0.003031
At point (-1.40, 0.35):
f(x,y) = -0.1745
Gradient = [-0.3639, 0.1221]
Eigenvalues: [-0.4164, 0.3589]
Hessian Matrix:
    Hess_11     Hess_12    Hess_21    Hess_22
    --------    -------    -------    -------

    -0.32102    0.25472    0.25472    0.26344
```

RESULTS

1. Role of linear algebra in multi-variate function optimization is revisited
2. Nature of the given function $z = f(x, y)$ at the given points are investigated using Geometrical, Calculus plus Linear Algebraic tools

12 | Assignment 44 Left and Right Eigenvectors

12.1 Relationship Between Eigen Values of A and A^T

Eigen values of A and A^T will be always same.

Quickly finding eigenvectors of 3x3 matrix using Gaussian Elimination

AI and Data-science heavily uses concept of eigenvalues and eigenvectors. Soon you will learn a methodology called DMD (Dynamic Mode Decomposition) for spatio-temporal data analytics. It is very difficult to get a grip on it unless you master LA concepts.

Consider a matrix $A = \begin{bmatrix} 1 & 3 & 1 \\ 2 & 1 & 3 \\ 3 & 2 & 2 \end{bmatrix}$. One of its eigenvalue is 6 since all column sum is 6. Corresponding left eigenvector is [1 1 1]'.

You can verify it as follows.

$(1\ 1\ 1)\begin{bmatrix} 1 & 3 & 1 \\ 2 & 1 & 3 \\ 3 & 2 & 2 \end{bmatrix}$=6(1 1 1) That is y^T A = λ y^T

Let us find out corresponding right eigenvector.

$$\begin{bmatrix} 1-6 & 3 & 1 \\ 2 & 1-6 & 3 \\ 3 & 2 & 2-6 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \Rightarrow \begin{bmatrix} 5 & -3 & -1 \\ 2 & -5 & 3 \\ 3 & 2 & -4 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Last row become dependent on first two (this will always happen). Hence consider

$$\begin{bmatrix} 1 & -3/5 & -1/5 \\ 2 & -5 & 3 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & -3/5 & -1/5 \\ 0 & -19/5 & 17/5 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Note now that x and y are pivotal variable and z is a free variable. From second equation, we obtain,

$$(19/5)\, y = (17/5)\, z$$

We take z=1 (z is a free variable, it can take any non-zero value)

Choice of z will not affect the eigenvector direction.

This you should verify. You are holding a computer. Verify with it.

$$z = 1 \Rightarrow y = 17/19$$

We use first row to find x value.

$$x = (3/5)\,y + (1/5)\,z = \frac{3}{5} \cdot \frac{17}{19} + \frac{1}{5} = \frac{70}{95}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 70/95 \\ 17/19 \\ 1 \end{bmatrix}$$

Very important thing to note

1. Eigen values of A are same as eigenvalues of A^T.
2. Corresponding to a given eigenvalue, Left eigenvector of A is same as right eigenvector of A^T.
3. Right and left eigen vectors are different for a given eigenvalue (except for symmetric matrices)

Assignment

1. By hand calculation , find out the eigenvalues and right and left eigen vectors of following matrices.

$A = \begin{bmatrix} 1 & 3 \\ 6 & -2 \end{bmatrix}$;note row sum is same

SOLUTION For this 2×2 matrix, the characteristic polynomial is

$$\lambda^2 - \text{trace}(A)\lambda + |A| = 0$$

$$\lambda^2 + \lambda - 20 = 0$$
$$(\lambda + 5)(\lambda - 4) = 0$$

So the eigen values are $(-5, 4)$. The eigen vectors are the solutions of the equation $(A - \lambda I)x = 0$ for each λ. Using the shortcut method,

$$EV(\lambda) = \begin{bmatrix} \lambda - d \\ c \end{bmatrix}$$

we get

$$EV(\lambda = -5) = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$$

$$EV(\lambda = 4) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Hence the right eigen values and their corresponding eigen vectors are:
the right eigen values of A are $(-5, 4)$

$$REV(\lambda = -5) = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$$

$$REV(\lambda = 4) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Similarly we can find the left eigen values and eigen vectors of A by finding eigen values and eigen vectors of A^T.
The left eigen values and their corresponding eigen vectors are:
the Left eigen values of A are $(-5, 4)$

$$LEV(\lambda = -5) = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

$$LEV(\lambda = 4) = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

`Matlab` code to verify these results are given below.

```
A=[1 3; 6 -2];
[EV,Eval]=eig(A);
[EVL,EvalL]=eig(A');
disp('Right Eigen Values of A are:')
```

```
Right Eigen Values of A are:
```

```
disp(diag(Eval))
```

```
   4
  -5
```

```
disp('Right Eigen Vectors of A are:')
```

```
Right Eigen Vectors of A are:
```

```
disp(EV)
```

```
     985/1393      -1292/2889
     985/1393       2584/2889
```

```
disp('Left Eigen values of A are:')
```

```
Left Eigen values of A are:
```

```
disp(diag(EvalL));
```

```
   4
  -5
```

```
disp("Left Eigen Vectors of A are:")
```

```
Left Eigen Vectors of A are:
```

```
disp(EVL)
```

```
2584/2889       -985/1393
1292/2889        985/1393
```

$B = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}$;note column sum is same

For this 2×2 matrix, the characteristic polynomial is

$$\lambda^2 - \text{trace}(A)\lambda + |A| = 0$$

$$\lambda^2 - 3\lambda - 4 = 0$$
$$(\lambda - 4)(\lambda + 4) = 0$$

So the eigen values are $(4, -1)$. The eigen vectors are the solutions of the equation $(A - \lambda I)x = 0$ for each λ. Using the shortcut method,

$$EV(\lambda) = \begin{bmatrix} \lambda - d \\ c \end{bmatrix}$$

we get

$$EV(\lambda = 4) = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$
$$EV(\lambda = -1) = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

Hence the right eigen values and their corresponding eigen vectors are:
the right eigen values of A are $(4, 1)$

$$REV(\lambda = 4) = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$
$$REV(\lambda = -1) = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

Similarly we can find the left eigen values and eigen vectors of A by finding eigen values and eigen vectors of A^T.
The left eigen values and their corresponding eigen vectors are:
the Left eigen values of A are $(4, -1)$

$$LEV(\lambda = 4) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$
$$LEV(\lambda = -1) = \begin{bmatrix} -2 \\ 3 \end{bmatrix}$$

`Matlab` code to verify these results are given below.

```
A1=[2 3; 2 1];
[EV,Eval]=eig(A1);
[EVL,EvalL]=eig(A1');
disp('Right Eigen Values of A are:')
```

```
Right Eigen Values of A are:
```

```
disp(diag(Eval))
```

```
     4
    -1
```

```
disp('Right Eigen Vectors of A are:')
```

```
Right Eigen Vectors of A are:
```

```
disp(EV)
```

```
    1189/1429      -985/1393
    1369/2468       985/1393
```

```
disp('Left Eigen values of A are:')
```

```
Left Eigen values of A are:
```

```
disp(diag(EvalL));
```

```
     4
    -1
```

```
disp("Left Eigen Vectors of A are:")
```

```
Left Eigen Vectors of A are:
```

```
disp(EVL)
```

```
     985/1393     -1369/2468
     985/1393      1189/1429
```

2. By hand calculation, find right and left eigenvector of following matrix corresponding eigenvalue of 7

$$A = \begin{bmatrix} 2 & 3 & 2 \\ 1 & 3 & 3 \\ 3 & 2 & 2 \end{bmatrix}.$$

The eigen vectors are the solutions of the equation $(A - \lambda I)x = 0$ for each λ.
So the $LEV(\lambda = 7)$ is obtained by solving

$$-5x + y + 3z = 0$$
$$x - 4y + 3z = 0$$

Using the componento-dividento;

$$\frac{x}{17} = \frac{y}{17} = \frac{z}{17}$$

So the right eigen vector is

$$REV(\lambda = 7) = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Similarly the left eigen vector for $\lambda = 7$ is obtained by solving $(A^T - 7I) = 0$.

$LEV(\lambda = 7)$ is obtained by solving

$$-5x + y + 3z = 0$$
$$3x - 4y + 2z = 0$$

Using the componento-dividento;

$$\frac{x}{14} = \frac{y}{19} = \frac{z}{17}$$

Therefore the left eigen vector is

$$LEV(\lambda = 7) = \begin{bmatrix} 14 \\ 19 \\ 17 \end{bmatrix}$$

`Matlab` code to verify these results are given below.

```
A3=[2 3 2; 1 3 3; 3 2 2];
[EV,Eval]=eig(A3);
[EVL,EvalL]=eig(A3');
disp('Right Eigen Values of A are:')
```

```
 Right Eigen Values of A are:
```

```
disp(diag(Eval))
```

```
        7          +     0i
       -1/9297754069410056 +     1i
       -1/9297754069410056 -     1i
```

```
disp('Right Eigen Vectors of A are:')
```

```
 Right Eigen Vectors of A are:
```

```
disp(EV)
```

```
     780/1351    +     0i            -151/1664    + 1447/3357i      -151/1664    - 1447/3357i
     780/1351    +     0i            -491/941     -  478/1505i      -491/941     +  478/1505i
     780/1351    +     0i            2202/3347    +     0i          2202/3347    +     0i
```

```
disp('Left Eigen values of A are:')
```

```
 Left Eigen values of A are:
```

```
disp(diag(EvalL));
```

```
 7          +    0i
-1/4969489243995030 +    1i
-1/4969489243995030 -    1i
```

```
disp("Left Eigen Vectors of A are:")
```

```
Left Eigen Vectors of A are:
```

```
disp(EVL)
```

```
941/1955   +    0i          -985/1393   +    0i          -985/1393   +    0i
697/1067   +    0i          1189/3363   + 1189/3363i     1189/3363   - 1189/3363i
941/1610   +    0i          1189/3363   - 1189/3363i     1189/3363   + 1189/3363i
```

RESULTS

1. Concept of left and right eigen vectors of a matrix are revisited.
2. Relation between left and right eigen values and eigen vectors of A and A^T are identified.
3. Left and right eigen vectors of given matrices are found and computationally verified.

13 | Assignment 45 Multivariate Regression

13.1 Data Generation for Multivariate Regression

13.1.1 Common origin of multivariate data

Multivariate data contains, at each sample point, multiple scalar values that represent different simulated or measured quantities. Multivariate data can come from numerical simulations that calculate a list of quantities at each time step, or from medical scanning modalities such as MRI, which can measure a variety of tissue characteristics, or from a combination of different scanning modalities, such as MRI, CT, and PET.

13.1.2 General form of multivariable function

For univariate regression problem of the form $y = mx + c$, we assumed m and c and computed y for a range of x and added error. Now our problem is , we want to create data for multivariate linear regression (MLR). Equation is of the form

$$y = m_1 x_1 + m_2 x_2 + m_3 x_3 + \cdots + m_n x_n + c + \epsilon$$

13.1.3 Popularity of MLR

Many problems can be formulated as regression problem. A dependent variable is rarely explained by only one variable. In such cases, an analyst uses multiple regression, which attempts to explain a dependent variable using more than one independent variable. The model, however, assumes that there are no major correlations between the independent variables. Some popular examples are listed below.

1. Digital Filter design can be casted as MLR
2. Pattern classification problem can be casted as Multiple output MLR
3. Neural Network training is basically estimation of MNLR-multivariate non-linear regression

13.1.4 A Synthetic data generation for MLR

Like in single variable case, we will assume values for the coefficients $m_1, m_2, \dots, m_n, c$ and compute Y for various combination of $x_1, x_2, \dots, x_n$ values. Finally we add noise from some probability distribution.

For univariate regression problem of the form $y = \text{mx} + c + \text{error}$, we assumed m and c and computed y for a range of x and added error. We could create data in matlab without any loop and re-estimate the parameters.

This was how it was done .

```
X=(-5:5)';
n=length(X);
Y0=5*X+6  ;
Y=Y0+3*randn(n,1) ;
A= [X  ones(n,1)] ;       Coeff = pinv(A)*Y;
Ys=A*Coeff;
plot (X,Y,'*')
hold on
plot(X,Ys)
```

Now our problem is , we want to create data for multivariate linear regression (MLR).

Equation is of the form $Y = m_1\ X_1 + m_2 X_2 + \ldots.. + m_n X_n$+C+$\varepsilon$

Finally we add noise from some probability distribution.

Let us take up one at a time.

Problem 1

How shall we generate N say (10000), n-tuple (say 8-tuple) values for $X_{1,} X_2, \ldots.., X_n$ Here we have many choices , important ones are

$X_{1,} X_2, \ldots.., X_n$ are independent and follow some distribution

$X_{1,} X_2, \ldots.., X_n$ dependent and follow a multivariate normal distribution

So, code can be

```
X=rand(10000,8)
X=mvnrnd(mu, sigma, 10000)
X=[X   ones(10000,1)]
```

Problem 2

How shall we compute $Y = m_1\ X_1 + m_2 X_2 + \ldots.. + m_n X_n$+C without loop for above N combination of $X_{1,} X_2, \ldots.., X_n$ values.

First we assume values for $m_{1,} m_2, \ldots.., m_n$,C and put it as a row matrix, say

```
Rc=[ 1  2  3  4  5  4  7  3  10];  \% n=8, C=10
```

We use `repmat` command to duplicate the values to the size of X matrix. This will allow us to compute Y in one step for all N(=10000) data points.

```
Mx=repmat(Rc,10000,1);
```

Problem 3

Adding noise to Y

```
Y=Y+sigma*randn(10000,1)
```

Final Re-estimation problem

```
Xm=Y
m = pinv(X)*Y
```

Final matlab code

```
N=1000;
n=8;
Rc=[ 1 2 3 4 5 4 6 3 100];
X=randn(N,n);
X=[X  ones(N,1)];
Mx=repmat(Rc,N,1);
Y=sum(Mx.*X, 2);
sigma=10;
Y=Y+sigma*randn(N,1);
Rc\_esti=pinv(X)*Y;
Yesti=X* Rc\_esti;
```

Assignment Question 1

Generate multivariate data using above code and report the following

SOLUTION

```
N=1000;
n=8;
Rc=[ 1 2 3 4 5 4 6 3 100];
X=randn(N,n);
X=[X  ones(N,1)];
Mx=repmat(Rc,N,1);
Y=sum(Mx.*X, 2);
sigma=10;
Y=Y+sigma*randn(N,1);
Rc_esti=pinv(X)*Y;
Yesti=X* Rc_esti;
```

- Plot of the percentage deviation of the coefficient

SOLUTION

Matlab code for this task is given below.

```
percentage_deviation = ((Rc_esti - Rc') ./ Rc') * 100;
figure;
bar(percentage_deviation);
xlabel('Coefficient Index');
ylabel('Percentage Deviation (%)');
title('Percentage Deviation of Estimated Regression Coefficients');
grid on;
```

Output of the code is shown in Figure 13.1.

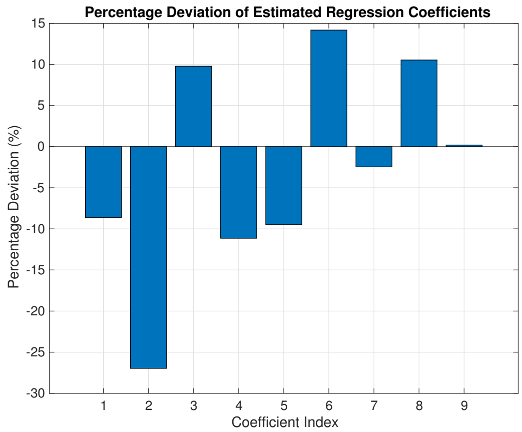


Figure 13.1: Percentage deviation of estimated regression coefficients

- get error vector `e = Y-Yesti` and check whether is orthogonal to all columns of X. If yes what is the reason?.

SOLUTION

`Matlab` code for checking orthogonality is given below. If dot product is zero, then the vectors are orthogonal.

```
e = Y - Yesti;
orthogonality_check = X' * e;
disp('Dot product of error vector with each column of X:');
disp(orthogonality_check);
```

Output of the orthogonality check is shown below.

```
1.0e-10 *

 0.2323
-0.2389
-0.0193
-0.1864
 0.0364
 0.0324
-0.5446
-0.3351
 0.1290
```

Since, the dot product is very small and while rounding it become zero. So the error vector is orthogonal to each columns of X.
Following `matlab` code check this and display the result.

```
if all(abs(orthogonality_check) < 1e-10)
    disp('The error vector is orthogonal to all columns of X.');
else
    disp('The error vector is not orthogonal to all columns of X.');
end
```

In this context, the result is:

```
The error vector is orthogonal to all columns of X.
```

- compute sum(e) and check whether it is close to zero or not

SOLUTION

Matlab code for this task is given below.

```
sum_e = sum(e);
disp('Sum of the error vector:');
disp(sum_e);
```

Output of the code is shown below.

```
Sum of the error vector:
   1.3031e-11
```

Following code will check whether the sum of error is zero or not.

```

if abs(sum_e) < 1e-10
    disp('The sum of the error vector is very close to zero.');
else
    disp('The sum of the error vector is not close to zero.');
end
```

Output of the code in this context is given below.

```
The sum of the error vector is very close to zero.
```

- explain why sum(e) need to be zero

SOLUTION

In OLS regression, the goal is to find the best-fit line (or hyperplane) that minimizes the sum of squared differences between the observed data points Y and the predicted values Y_{esti}. The least squares solution ensures that the residuals are distributed such that their sum is minimized. This balancing of positive and negative residuals naturally results in the sum of the residuals being approximately zero.

Assignment Question 2

Generate data for single variable nonlinear regression equation of the form $y = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n + \varepsilon$ for n=5 and re-estimate the coefficients using the same methodology . Assume x varies in the range -5 to 5

SOLUTION

Matlab code for this task is given below.

```
N = 1000;
range [-5, 5]
x = 10 * rand(N,1) - 5;
a_true = [1, -2, 3, -1, 0.5, 2];
X = [ones(N,1), x, x.^2, x.^3, x.^4, x.^5];
epsilon = 0.5 * randn(N, 1);
y = X * a_true' + epsilon;
```

```
a_estimated = pinv(X) * y;
disp('True coefficients:');
disp(a_true');
disp('Estimated coefficients:');
disp(a_estimated);
y_estimated = X * a_estimated;
e = y - y_estimated;
orthogonality_check = X' * e;
disp('Dot product of error vector with each column of X:');
disp(orthogonality_check);
sum_e = sum(e);
disp('Sum of the error vector:');
disp(sum_e);
if abs(sum_e) < 1e-10
    disp('The sum of the error vector is very close to zero.');
else
    disp('The sum of the error vector is not close to zero.');
end
```

Output of the code is given below.

```
True coefficients:
    1.0000
   -2.0000
    3.0000
   -1.0000
    0.5000
    2.0000
Estimated coefficients:
    1.0256
   -1.9503
    2.9966
   -1.0097
    0.5001
    2.0004
Dot product of error vector with each column of X:
   1.0e-06 *

   -0.0001
    0.0012
    0.0006
    0.0201
    0.0188
    0.3949

Sum of the error vector:
  -5.8134e-11

The sum of the error vector is very close to zero.
```

Following code will plot the true and estimated coefficients of the regression model.

```
figure;
bar([a_true' a_estimated]);
```

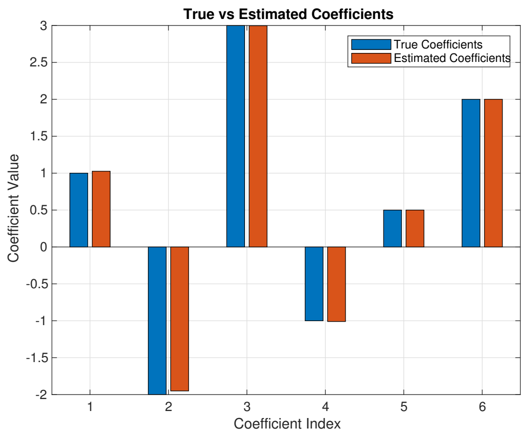


Figure 13.2: Comparison of predicted coefficents with true coefficients.

```
legend('True Coefficients', 'Estimated Coefficients');
xlabel('Coefficient Index');
ylabel('Coefficient Value');
title('True vs Estimated Coefficients');
grid on;
```

Output of the code is shown in Figure 13.2.
Following code will demonstrate the skill of the model.

```
x_smooth = linspace(-5, 5, 1000)';
X_smooth = [ones(length(x_smooth), 1), x_smooth, x_smooth.^2,
    x_smooth.^3, x_smooth.^4, x_smooth.^5];
y_smooth = X_smooth * a_estimated;
figure;
scatter(x, y, 10, 'b', 'filled');
plot(x_smooth, y_smooth, 'r', 'LineWidth', 2);
title('Scatter Plot of Actual Points and Fitted Polynomial Curve');
xlabel('x');
ylabel('y');
legend('Actual Data Points', 'Fitted Polynomial Curve');
grid on;
hold off;
```

Output of the above code is shown in Figure 13.3.

Assignment Question 3

Formulate the multivariate regression problem as

1. Linear algebra problem

SOLUTION

In multivariate linear regression, the model can be written as:

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\epsilon}$$

where:

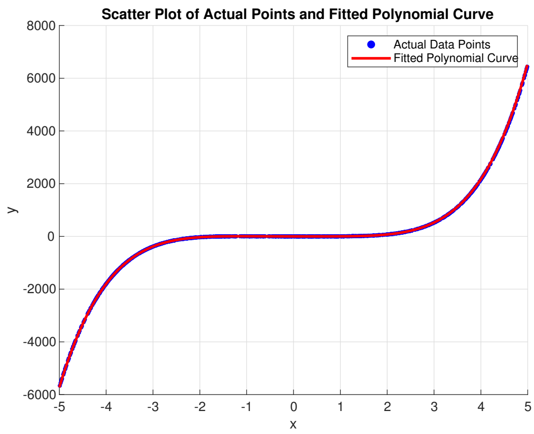


Figure 13.3: Skill of the polynomial regression.

- $\mathbf{X} \in \mathbb{R}^{N\times(n+1)}$ is the input matrix with N data points and $n+1$ features (including a column of ones for the intercept),
- $\mathbf{w} \in \mathbb{R}^{(n+1)}$ is the vector of regression coefficients,
- $\mathbf{y} \in \mathbb{R}^{N}$ is the vector of observed outputs,
- $\boldsymbol{\epsilon} \in \mathbb{R}^{N}$ is the vector of errors (noise).

To estimate $\mathbf{w}$, we solve the least-squares problem by minimizing the residuals between the observed values $\mathbf{y}$ and the predicted values $\mathbf{Xw}$. This leads to the normal equations:

$$\mathbf{X}^\top \mathbf{X}\mathbf{w} = \mathbf{X}^\top \mathbf{y}$$

The solution to these normal equations is:

$$\mathbf{w} = \left(\mathbf{X}^\top \mathbf{X}\right)^{-1} \mathbf{X}^\top \mathbf{y}$$

This is the closed-form solution for the least-squares estimate of $\mathbf{w}$, provided that $\mathbf{X}^\top \mathbf{X}$ is invertible.

2. Optimization problem

SOLUTION

In the optimization formulation, we aim to minimize the sum of squared errors between the observed outputs $\mathbf{y}$ and the predicted outputs $\mathbf{Xw}$. The objective function to minimize is:

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N} \left(y_i - \mathbf{x}_i^\top \mathbf{w}\right)^2$$

In matrix form, this can be written as:

$$J(\mathbf{w}) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2$$

where $\|\cdot\|_2^2$ is the squared L_2 norm, i.e., the sum of squared residuals.

13.1.5 Gradient of the Objective Function

The gradient of the objective function with respect to $\mathbf{w}$ is given by:

$$\nabla J(\mathbf{w}) = -\mathbf{X}^\top (\mathbf{y} - \mathbf{X}\mathbf{w})$$

13.1.6 Gradient Descent

To minimize $J(\mathbf{w})$, we can use the gradient descent algorithm, where the weights $\mathbf{w}$ are updated iteratively as follows:

$$\mathbf{w}_{\text{new}} = \mathbf{w} - \eta \nabla J(\mathbf{w})$$

Here, η is the learning rate, controlling the step size in the gradient descent algorithm. The iteration continues until the gradient is sufficiently small, indicating that the algorithm has converged.

RESULTS

1. Various methods of simulating data for multivariable regression is revisited
2. Percentage error for coefficient estimate is calculated and visually interpreted.
3. Multiple linear regression problem is modelled as both a linear algebra model (that provide a closed form solution) and an optimization model (that provide an approximate solution iteratively).

14 Assignment 46 Orthogonal Basis for Signal Processing

14.1 Linear Algebra for Signal Processing

What is Digital signal processing? What is the importance of the linear algebraic concept of 'Bases' in Digital signal processing.

For the purpose of defining usual signal that people have in mind, we limit our context. We define digital signal as sequence (time or space coordinate indexed or both) of numbers that carry some information. Mathematically, it is a function.

The 'information' here is the pattern.

14.1.1 A simple example

Our ECG waveform has a generic pattern. Given an ECG waveform, how far it is away from the expected pattern is information for the doctor.

To **quantify and characterize information** digitally, we use the concept of bases.

14.1.2 Common basis for signal processing

We express the digital signal (a function, now a vector) in terms of linear combination of other functions(vectors) selected and fixed in advance.

Commonly used bases are

1. Fourier bases (sines and cosines with integer wave numbers but of same length/duration)
2. Cosine bases (cosines of same length and with integer multiple of half-wave numbers)
3. Walsh Hadamard bases
4. Wavelet bases.

These base elements (vectors) are created to be orthogonal for computational reasons.

Linear independence and completeness (basis set should span entire signal space) is the criteria to be met by the basis set.

Choice of basis set depends on applications.

Once a signal is expressed in terms of some standard basis set, we can talk about the nature of the signal in terms of the magnitude of the coefficients of each base elements.

For example if

$$\underbrace{\begin{bmatrix} 1 \\ 2 \\ -2 \\ 3 \end{bmatrix}}_{\text{signals}} = 10* \underbrace{\begin{bmatrix} . \\ . \\ . \\ . \end{bmatrix}}_{\text{base1}} + 2* \underbrace{\begin{bmatrix} . \\ . \\ . \\ . \end{bmatrix}}_{\text{base2}} + (-1)* \underbrace{\begin{bmatrix} . \\ . \\ . \\ . \end{bmatrix}}_{\text{base3}} + 3* \underbrace{\begin{bmatrix} . \\ . \\ . \\ . \end{bmatrix}}_{\text{base4}}$$

we say the first component(base) is more in the signal,

We can further process the signal by modifying these coefficients and re-constructing the signal with the modified coefficients.

Suppose we are interested in analyzing a n- tuple signal.

Following `matlab` codes give basis set as **columns of a matrix**.

(For getting basis vectors in columns use inverse transform of identity matrix.

For getting basis vectors in rows use forward transform of identity matrix.)

We take n=4

1. B=idst(eye(n))

$$B = \begin{bmatrix} 0.2351 & 0.3804 & 0.3804 & 0.2351 \\ 0.3804 & 0.2351 & -0.2351 & -0.3804 \\ 0.3804 & -0.2351 & -0.2351 & 0.3804 \\ 0.2351 & -0.3804 & 0.3804 & -0.2351 \end{bmatrix}$$

2. B=idct(eye(n))

$$B = \begin{bmatrix} 0.5000 & 0.6533 & 0.5000 & 0.2706 \\ 0.5000 & 0.2706 & 0.5000 & -0.6533 \\ 0.5000 & -0.2706 & 0.5000 & 0.6533 \\ 0.5000 & -0.6533 & 0.5000 & -0.2706 \end{bmatrix}$$

3. Walsh Hadamard bases

$$\text{B} = \begin{bmatrix} 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & -0.5 & 0.5 & -0.5 \\ 0.5 & 0.5 & -0.5 & -0.5 \\ 0.5 & -0.5 & -0.5 & -0.5 \end{bmatrix}$$

4. Complex Fourier bases

B=1/sqrt(n)*(fft(eye(4)))'

$$B = \frac{1}{\sqrt{n=4}}$$

$$\begin{bmatrix} 1.00 & 1.00 & 1.00 & 1.00 \\ 1.00 & 1.00 & 1i & -1i \\ 1.00 & -1.00 & 1.00 & -1.00 \\ 1.00 & 0.00 & -1i & 1i \end{bmatrix}$$

5. Wavelet bases

$$B = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} \\ 0 & \frac{1}{\sqrt{2}} & 0 & \frac{-1}{\sqrt{2}} \end{bmatrix}$$

In addition to this, we can split signal based on other criteria.

Assignment Question

Express vector [1 2 -2 3]' using above five basis set vectors.

Answer should look like

$$\begin{bmatrix} 1 \\ 2 \\ -2 \\ 3 \end{bmatrix} = \text{coef1x} \underbrace{\begin{bmatrix} . \\ . \\ . \\ . \end{bmatrix}}_{\text{base1}} + \text{coef2x} + \underbrace{\begin{bmatrix} . \\ . \\ . \\ . \end{bmatrix}}_{\text{base2}} + \text{coef3x} \underbrace{\begin{bmatrix} . \\ . \\ . \\ . \end{bmatrix}}_{\text{base3}} +$$

$$\text{coef4x}$$

$$\underbrace{\begin{bmatrix} . \\ . \\ . \\ . \end{bmatrix}}_{\text{base4}}$$

SOLUTION

The task can be represented as follows. Let v be the vector in $\mathbb{R}^4$. For a given basis, $B = \{b_1, b_2, b_3, b_4\}$ the vector v can be written as:

$$\begin{aligned} v^T &= c_1 b_1 + c_2 b_2 + c_3 b_3 + c_4 b_4 \\ & CB \\ \therefore C &= B^{-1} v^T \end{aligned}$$

`Matlab` code and its output for this task is given below.

```
n=4;
v=[1 2 -2 3];
B1=idst(eye(n));
B2=idct(eye(n));
B3=[0.5 0.5 0.5 0.5; 0.5 -0.5 0.5 -0.5; 0.5 0.5 -0.5 -0.5; 0.5 -0.5
    -0.5 -0.5];
B4=1/sqrt(n)*(fft(eye(n)))';
B5=1/sqrt(2)*[1 0 1 0; 1 0 -1 0;0 1 0 1; 0 1 0 -1];
C1=pinv(B1)*v';
C2=pinv(B2)*v';
C3=pinv(B3)*v';
C4=pinv(B4)*v';
C5=pinv(B5)*v';
disp("The vector v=c_1b_1+c_2b_2+c_3b_3");
disp("The coefficients c_1,c_2,and c_3 is given for each Basis is
   given below:");
disp(Coefficients for Basis 1:);
disp(C1');
disp("Coefficients for Basis 2:");
disp(C2');
disp("Coefficients for Basis 3:");
disp(C3');
disp("Coefficients for Basis 4:");
disp(C4');
disp("Coefficients for Basis 5:");
disp(C5');
```

```
The vector v=c_1b_1+c_2b_2+c_3b_3

The coefficients c_1,c_2,and c_3 is given for each Basis is given below:
Coefficients for Basis 1:
    2.3511    0.4490    3.8042   -4.9798
Coefficients for Basis 2:
    2.0000   -0.2242    2.0000   -3.1543
Coefficients for Basis 3:
    4.0000   -5.0000   -1.0000    4.0000
Coefficients for Basis 4:
   2.0000 + 0.0000i   1.5000 - 0.5000i  -3.0000 - 0.0000i   1.5000 + 0.5000i
Coefficients for Basis 5:
    2.1213    0.7071   -0.7071   -3.5355
```

RESULTS

1. Importance of linear algebra in Signal Processing is revisited.
2. Common basis for signal processing is revisited.
3. A given vector (signal) is represented in terms of basis vectors for each basis set.

15 | Assignment 47 Lagrangian Multipliers

15.1 Solution of $Ax = b$ using Linear Algebra and Calculus

15.1.1 Two approaches but the same solution!

For millions of computer science people aspiring to become AI and data scientists, one of the stumbling block is optimization theory based on lagrangian multipliers. Kernel methods (SVM, SVR, Kernel PCA, Kernel CCA, Kernel ICA, Kernel DMD etc) and Modern control theory heavily depends on it. In fact entire convex optimization theory is built over concepts like lagrangian function and lagrangian duality. Ignorance of this techniques make you weep when you read latest papers on AI and Data Science. So better confront it at the earliest.

Problem 1.

x_1+2x_2=6.What is the least norm solution.

LA solution

$$[1\ 2]\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 6 \longrightarrow \text{Ax} = b$$

$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in R^2$.With respect to matrix $A = [1\ 2]$ $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in R^2$ has two orthogonal components. One in Rowspace and the other in Right nullspace .

So, $\begin{vmatrix} \\ x \\ \\ \end{vmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{vmatrix} \\ x_R \\ \\ \end{vmatrix} + \begin{vmatrix} \\ x_N \\ \\ \end{vmatrix}$

$A(x_R + x_N) = b \Rightarrow x_R$ provides least norm solution since $Ax_R = b$ & $Ax_N = 0$

x_R is rowspace component and so x_R is linear combination of rows. So let

$$x_R = c\begin{bmatrix} 1 \\ 2 \end{bmatrix}; c \in R$$

$$Ax_R = b \Rightarrow c[1\ 2]\begin{bmatrix} 1 \\ 2 \end{bmatrix} = 6 \Rightarrow c = 6/5$$

$x_R = \frac{6}{5}\begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 6/5 \\ 12/5 \end{bmatrix}$. Check x_1+2x_2=6; 6 / 5+2 * 12/5 = 6

Solution with lagrangian multiplier method.

The problem can be formulated as:

arg min $x_1{}^2$+$x_2{}^2$

Subjected to x_1+2x_2=6

This means which point on the given line has least square-norm.

Forming lagrangian f unction, we obtain

$L(x,\lambda) = x_1{}^2$+ $x_2{}^2$ - $\lambda\,(x_1 + 2x_2 - 6)$

Equation $\frac{\partial L}{\partial x}$=$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$& $\frac{\partial L}{\partial \lambda}$ = 0, we obtain

$$\frac{\partial L}{\partial x} = \begin{bmatrix} 2x_1 \\ 2x_2 \end{bmatrix} - \lambda \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \frac{\lambda}{2} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 6/5 \\ 12/5 \end{bmatrix}$$

$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$=$\begin{bmatrix} 6/5 \\ 12/5 \end{bmatrix}$is require solution vector

Check x_1+2x_2=6; 6 / 5+2 * 12/5 = 6

Problem 2. More than one equality constraints in higher dimension

In matrix form, the problem can be formulated as:

$$\arg\min_x X^T X = \arg\min_x ||x||_2^2$$

subject to $Ax = b$.

This means which point x that satisfy $Ax = b$ has least square-norm.

The problem in three variables and two constraints look like this,

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}; A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 1 & 2 \end{bmatrix}; b = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

LA solution. For the least norm, x has to be in row space. We assume rows of A are independent.

So let,

$x = A^T y \equiv$ linear combination of columns of $A^T \equiv$ linear combination of columns of A

$$\mathrm{Ax} = b \Rightarrow$$
$$\mathrm{AA}^T y = b \Rightarrow y = \left(\mathrm{AA}^T\right)^{-1} b$$

$$x^* = A^T y = A^T \left(\mathrm{AA}^T\right)^{-1} b$$

If Rows of A are dependent, we resort to `pinv`.

$$x^* = \text{pinv}(A) * \ b$$

You may check that $x^* = \text{pinv}(A) * \ b$ also give same solution.

Optimization method

$$L(x,\lambda) = L\left(x,\lambda_1,\lambda_{2,}\right) = \left(x_1^2, x_2^2, x_3^2\right) - \lambda_1\left(x_{1+2x_2+3_{x_3}} - 3\right) - \lambda_2\left(4x_{1+x_2+2_{x_3}} - 4\right)$$

$$\begin{bmatrix} \frac{\partial L}{\partial x_1} \\ \frac{\partial L}{\partial x_2} \\ \frac{\partial L}{\partial x_3} \end{bmatrix} = 2\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} - \lambda_1\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} - \lambda_2\begin{bmatrix} 4 \\ 1 \\ 2 \end{bmatrix} = 2x - \begin{bmatrix} 1 & 4 \\ 2 & 1 \\ 3 & 2 \end{bmatrix}\begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = 2x - A^T\lambda$$

$$\frac{\partial L}{\partial x} = 2x - A^T\lambda = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \Rightarrow x^* = \frac{1}{2}A^T y = \frac{1}{2}\begin{bmatrix} 1 & 4 \\ 2 & 1 \\ 3 & 2 \end{bmatrix}\begin{bmatrix} \lambda_1 \\ \lambda_2 \end{bmatrix} = \frac{\lambda_2}{2}$$

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} + \frac{\lambda_2}{2}\begin{bmatrix} 4 \\ 1 \\ 2 \end{bmatrix}$$

$x^* \equiv$ linear combination of columns of A

$$\frac{\partial L}{\partial \lambda_1} = x_{1+2x_2+3_{x_3}} - 3 = 0 \,\&\, \frac{\partial L}{\partial \lambda_2} = 4x_{1+x_2+2_{x_3}} - 4 = 0 \Rightarrow$$

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 1 & 2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} - \begin{bmatrix} 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow Ax^* = b$$

On substitution $Ax^* = \frac{1}{2}\text{AA}^T\lambda = b$

$$\lambda = 2\left(\text{AA}^T\right)^{-1}b$$

$$x^* = \frac{1}{2}A^T\lambda = \frac{1}{2}A^T * 2\left(\text{AA}^T\right)^{-1}b = A^T\left(\text{AA}^T\right)^{-1}b$$

Generic solution

min $x^T x$

$$s.t \ \text{Ax} = b;$$

$$L(x,\lambda) = x^T x - \lambda^T(\text{Ax} - b)\,;\text{note } \lambda \text{ is } a \text{ vector}$$

$$\frac{\partial L}{\partial x} = 2x - A^T\lambda = 0 \Rightarrow x^* = \frac{1}{2}A^T\lambda \Rightarrow \text{solution vector is in row} - \text{space}$$

$$\frac{\partial L}{\partial \lambda} = (\text{Ax} - b) = 0 \Rightarrow Ax^* = b$$

above two equations imply

$$x^* = A^T \left(\mathrm{AA}^T\right)^{-1} b$$

Solution assumes Rows of A are independent.

Assignment Questions

1 Find least norm solution to $x + 2y + 4z = 10$

SOLUTION

We want to find the least norm solution to the equation:

$$x + 2y + 4z = 10$$

This can be represented in matrix form as:

$$Ax = b$$

where

$$A = \begin{bmatrix} 1 & 2 & 4 \end{bmatrix}, \quad x = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, \quad b = \begin{bmatrix} 10 \end{bmatrix}$$

The least norm solution is given by the formula:

$$x = A^T (AA^T)^{-1} b$$

Calculating AA^T:

$$A^T = \begin{bmatrix} 1 \\ 2 \\ 4 \end{bmatrix}$$

$$AA^T = \begin{bmatrix} 1 & 2 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 4 \end{bmatrix} = 21$$

Now calculate $A^T (AA^T)^{-1} b$:

$$A^T (AA^T) b = \begin{bmatrix} 1 \\ 2 \\ 4 \end{bmatrix} 21^{-1} 10 = \begin{bmatrix} \frac{10}{21} \\ \frac{20}{21} \\ \frac{40}{21} \end{bmatrix}$$

The solution vector x can be represented as:

$$x = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \frac{10}{21} \\ \frac{20}{21} \\ \frac{40}{21} \end{bmatrix}$$

`Matlab` code for this task and its output are shown below.

```
A = [1; 2; 4];
b = [10];
v_least_norm_method2 = (A * inv(A' * A)* b);
disp('The least norm solution using A^T(AA^T)^(-1)b is:');
```

```
The least norm solution using A^T(AA^T)^(-1)b is:
```

```
disp(v_least_norm_method2);
```

```
        10/21
        20/21
        40/21
```

Consistency of the result is verified with the `x=pinv()*b` method as shown below.

```
A = [1 2 4];
b = 10;
v_least_norm = pinv(A) * b;
disp('The least norm solution [x, y, z] is:');
```

```
The least norm solution [x, y, z] is:
```

```
disp(v_least_norm);
```

```
        10/21
        20/21
        40/21
```

2 Find least norm solution to $x+2y+4z=10\& 4x+3y+3z=20$

SOLUTION

We seek the least-norm solution to the system:

$$x+2y+4z=10$$

$$4x+3y+3z=20$$

The system can be expressed in matrix form as:

$$A\mathbf{x}=\mathbf{b}$$

where

$$A=\begin{pmatrix}1 & 2 & 4\\ 4 & 3 & 3\end{pmatrix}, \quad \mathbf{x}=\begin{pmatrix}x\\ y\\ z\end{pmatrix}, \quad \mathbf{b}=\begin{pmatrix}10\\ 20\end{pmatrix}$$

The least-norm solution is given by:

$$\mathbf{x}=A^T(AA^T)^{-1}\mathbf{b}$$

First, compute A^T, the transpose of matrix A:

$$A^T=\begin{pmatrix}1 & 4\\ 2 & 3\\ 4 & 3\end{pmatrix}$$

Next, calculate AA^T:

$$AA^T=\begin{pmatrix}1 & 2 & 4\\ 4 & 3 & 3\end{pmatrix}\begin{pmatrix}1 & 4\\ 2 & 3\\ 4 & 3\end{pmatrix}=\begin{pmatrix}21 & 22\\ 22 & 34\end{pmatrix}$$

The inverse of AA^T is:

$$(AA^T)^{-1} = \frac{1}{230}\begin{pmatrix} 34 & -22 \\ -22 & 21 \end{pmatrix}$$

We now compute $A^T(AA^T)^{-1}$:

$$A^T(AA^T)^{-1} = \frac{1}{230}\begin{pmatrix} 1 & 4 \\ 2 & 3 \\ 4 & 3 \end{pmatrix}\begin{pmatrix} 34 & -22 \\ -22 & 21 \end{pmatrix} = \frac{1}{230}\begin{pmatrix} -54 & 62 \\ 2 & 19 \\ 70 & -25 \end{pmatrix}$$

Finally, multiplying by $\mathbf{b}$:

$$\mathbf{x} = \frac{1}{230}\begin{pmatrix} -54 & 62 \\ 2 & 19 \\ 70 & -25 \end{pmatrix}\begin{pmatrix} 10 \\ 20 \end{pmatrix} = \frac{1}{230}\begin{pmatrix} 700 \\ 400 \\ 200 \end{pmatrix} = \begin{pmatrix} \frac{70}{23} \\ \frac{40}{23} \\ \frac{20}{23} \end{pmatrix}$$

Thus, the least-norm solution is:

$$x = 3.04347, \quad y = 1.7391, \quad z = 0.8695$$

`Matlab` code for this task and its output is given below.

```
A=[1 2 4; 4 3 3];
b=[10;20];
x=A'*inv(A*A')*b;
x_i=pinv(A)*b;
disp("The least norm solution is:");
```

```
The least norm solution is:
```

```
disp(x)
```

```
70/23
40/23
20/23
```

```
disp("Solution using pseudo inverse:");
```

```
Solution using pseudo inverse:
```

```
disp(x_i)
```

```
70/23
40/23
20/23
```

```
if (x==x_i<1.010-12)
    disp("Both least norm solution and Pseudo inverse solution are
       equal")
else
    disp("The least norm solution and Pseudo inverse solution are
       different")
end
```

```
Both least norm solution and Pseudo inverse solution are equal
```

Solution using Lagrange multiplier approach is given below. We can also solve this system using Lagrange multiplier method. In this method the problem is converted as a minimization problem as steted below.
Minimize the Euclidean norm $\|\mathbf{v}\|^2 = x^2 + y^2 + z^2$, subject to the constraints, $Ax = b$. We define the Lagrange function as:

$$\mathscr{L}(x, y, z, \lambda_1, \lambda_2) = x^2 + y^2 + z^2 - \lambda_1(x + 2y + 4z - 10) - \lambda_2(4x + 3y + 3z - 20)$$

The critical points are found by equating the first order partial derivatives with respect to x, y, z and λ into zero.
Taking the partial derivatives of $\mathscr{L}$ with respect to x, y, and z, and setting them equal to zero gives:

$$\frac{\partial \mathscr{L}}{\partial x} = 2x - \lambda_1 - 4\lambda_2 = 0 \quad (3)$$

$$\frac{\partial \mathscr{L}}{\partial y} = 2y - 2\lambda_1 - 3\lambda_2 = 0 \quad (4)$$

$$\frac{\partial \mathscr{L}}{\partial z} = 2z - 4\lambda_1 - 3\lambda_2 = 0 \quad (5)$$

Solve for x, y, and z From equation (3):

$$x = \frac{\lambda_1 + 4\lambda_2}{2} \quad (6)$$

From equation (4):

$$y = \frac{2\lambda_1 + 3\lambda_2}{2} \quad (7)$$

From equation (5):

$$z = \frac{4\lambda_1 + 3\lambda_2}{2} \quad (8)$$

Substituting (6), (7), and (8) into the constraints (1) and (2) and simplifying leads to a system of equations that can be solved for λ_1 and λ_2.
After substitution and simplification, we find:

$$\lambda_1 = -\frac{20}{23}, \quad \lambda_2 = \frac{40}{23}$$

Now compute x, y, and z Substituting λ_1 and λ_2 into equations (6), (7), and (8):

$$x = \frac{\lambda_1 + 4\lambda_2}{2} = \frac{-\frac{20}{23} + 4 \cdot \frac{40}{23}}{2} = \frac{70}{23}$$

$$y = \frac{2\lambda_1 + 3\lambda_2}{2} = \frac{2 \cdot -\frac{20}{23} + 3 \cdot \frac{40}{23}}{2} = \frac{40}{23}$$

$$z = \frac{4\lambda_1 + 3\lambda_2}{2} = \frac{4 \cdot -\frac{20}{23} + 3 \cdot \frac{40}{23}}{2} = \frac{20}{23}$$

Thus, the Lagrangian solution is:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \frac{70}{23} \\ \frac{40}{23} \\ \frac{20}{23} \end{bmatrix}$$

3. Find Nearest point on $x + 3y + 4z = 10$ from origin (0,0,0)

SOLUTION

We need to minimize the distance from the origin to a point $\mathbf{p} = (x, y, z)$ on the plane. The distance is given by the norm:

$$d(\mathbf{p}) = \|\mathbf{p}\| = \sqrt{x^2 + y^2 + z^2}$$

subject to the constraint:

$$x + 3y + 4z = 10$$

We can solve this minimization problem using the method of Lagrange multipliers. The objective function to minimize is:

$$f(x, y, z) = x^2 + y^2 + z^2$$

and the constraint is:

$$g(x, y, z) = x + 3y + 4z - 10 = 0$$

The Lagrangian is:

$$\mathscr{L}(x, y, z, \lambda) = x^2 + y^2 + z^2 + \lambda(x + 3y + 4z - 10)$$

Taking partial derivatives of $\mathscr{L}$ with respect to x, y, z, and λ, and setting them to zero:

$$\frac{\partial \mathscr{L}}{\partial x} = 2x + \lambda = 0 \quad \Rightarrow \quad x = -\frac{\lambda}{2}$$

$$\frac{\partial \mathscr{L}}{\partial y} = 2y + 3\lambda = 0 \quad \Rightarrow \quad y = -\frac{3\lambda}{2}$$

$$\frac{\partial \mathscr{L}}{\partial z} = 2z + 4\lambda = 0 \quad \Rightarrow \quad z = -2\lambda$$

$$\frac{\partial \mathscr{L}}{\partial \lambda} = x + 3y + 4z - 10 = 0$$

Substitute the expressions for x, y, and z into the constraint equation:

$$-\frac{\lambda}{2} + 3\left(-\frac{3\lambda}{2}\right) + 4(-2\lambda) = 10$$

Simplifying:

$$-\frac{\lambda}{2} - \frac{9\lambda}{2} - 8\lambda = 10$$

$$-\left(\frac{\lambda}{2} + \frac{9\lambda}{2} + 8\lambda\right) = 10$$

$$-\left(\frac{10\lambda}{2} + 8\lambda\right) = 10$$

$$-5\lambda - 8\lambda = 10 \quad \Rightarrow \quad -13\lambda = 10 \quad \Rightarrow \quad \lambda = -\frac{10}{13}$$

Now substitute $\lambda = -\frac{10}{13}$ into the expressions for x, y, and z:

$$x = -\frac{-10}{2 \times 13} = \frac{5}{13}, \quad y = -\frac{3(-10)}{2 \times 13} = \frac{15}{13}, \quad z = -2 \times -\frac{10}{13} = \frac{20}{13}$$

Thus, the nearest point to the origin on the plane is:

$$\mathbf{p}_{\text{nearest}} = \left(\frac{5}{13}, \frac{15}{13}, \frac{20}{13}\right)$$

The same problem can be solved using projection of $(0, 0, 0)$ into the plane through the normal line. `Matlab` code for this task and its output are shown below.

```
format rational
n = [1; 3; 4];
d0 = 10
P0 = [0; 0; 0
P_nearest = P0 - (dot(n, P0) - d0) / (norm(n)^2) * n;
disp('The nearest point on the plane x + 3y + 4z = 10 from (0,0,0)
    is:');
disp(P_nearest);
```

```
The nearest point on the plane x + 3y + 4z = 10 from (0,0,0) is:
       5/13
      15/13
      20/13
```

1. Find Projection of (1, 2, 8) on to set of points defined by $x+3y+4z=10$.

SOLUTION

We aim to find the projection of the point $P(1,2,8)$ onto the plane defined by the equation:

$$g(x,y,z) = x+3y+4z-10 = 0$$

We want to minimize the distance from the point P to a point (x,y,z) on the plane. The squared distance is given by:

$$f(x,y,z) = (x-1)^2 + (y-2)^2 + (z-8)^2$$

We set up the Lagrangian function:

$$\mathscr{L}(x,y,z,\lambda) = (x-1)^2 + (y-2)^2 + (z-8)^2 + \lambda(x+3y+4z-10)$$

Now Compute the Partial Derivatives Taking the partial derivatives and setting them to zero gives:

$$\frac{\partial \mathscr{L}}{\partial x} = 2(x-1) + \lambda = 0 \quad (1)$$

$$\frac{\partial \mathscr{L}}{\partial y} = 2(y-2) + 3\lambda = 0 \quad (2)$$

$$\frac{\partial \mathscr{L}}{\partial z} = 2(z-8) + 4\lambda = 0 \quad (3)$$

$$\frac{\partial \mathscr{L}}{\partial \lambda} = x+3y+4z-10 = 0 \quad (4)$$

From equation (1):

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

From equation (2):

$$y = 2 - \frac{3\lambda}{2}$$

From equation (3):

$$z = 8 - 2\lambda$$

Using these values in (4),

$$\lambda = \frac{29}{13} \quad (5)$$

Substituting (5) in x, y and z, we get

$$\begin{aligned} x = 1 - \frac{29}{2\times 13} &= \frac{-3}{26} \\ y = 2 - \frac{3\times 29}{2\times 13} &= \frac{-35}{26} \\ z = 8 - 2\frac{29}{13} &= \frac{46}{13} \end{aligned}$$

The projection of the point $(1,2,8)$ onto the plane $x+3y+4z=10$ is:

$$\left(-\frac{3}{26}, -\frac{35}{26}, \frac{46}{13}\right)$$

`Matlab` code and its output for the same task but in projection method is shown below.

```
n = [1; 3; 4];
d0 = 10;
P0 = [1; 2; 8];
P_nearest = P0 - (dot(n, P0) - d0) / (norm(n)^2) * n;
disp('The projection of (1,2,8)in to x+3y+4z=10 is:');
disp(P_nearest);
```

```
The nearest point on the plane x + 3y + 4z = 10 from (1,2,8) is:
      -3/26
     -35/26
      46/13
```

1. Solve $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$ Use both LA and optimization approach

SOLUTION

1. **LA approach**
   Consider the system of equations represented as $Ax = b$, where:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 1 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

First, compute the transpose of A:

$$A^T = \begin{bmatrix} 1 & 4 \\ 2 & 1 \\ 3 & 2 \end{bmatrix}$$

Next, calculate AA^T:

$$AA^T = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 1 & 2 \end{bmatrix} \begin{bmatrix} 1 & 4 \\ 2 & 1 \\ 3 & 2 \end{bmatrix} = \begin{bmatrix} 14 & 12 \\ 12 & 21 \end{bmatrix}$$

The least norm solution can be found using the formula:

$$x = A^T (AA^T)^{-1} b$$

To find $(AA^T)^{-1}$

$$(AA^T)^{-1} = \begin{bmatrix} \frac{7}{50} & -\frac{2}{25} \\ -\frac{2}{25} & \frac{7}{75} \end{bmatrix}$$

Then:

$$x = \begin{bmatrix} 1 & 4 \\ 2 & 1 \\ 3 & 2 \end{bmatrix} \cdot \begin{bmatrix} \frac{13}{50} \\ -\frac{4}{75} \end{bmatrix}$$

Calculate this gives:

$$x = \begin{bmatrix} \frac{7}{150} \\ \frac{7}{15} \\ \frac{101}{150} \end{bmatrix}$$

The least norm solution is:

$$x = \begin{bmatrix} \frac{7}{150} \\ \frac{7}{15} \\ \frac{101}{150} \end{bmatrix}$$

2. **Lagrange Multiplier Method:**

We aim to solve the system $Ax = b$ using the Lagrange multiplier method, where:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 1 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

The objective function is to minimize the norm:

$$f(x) = \|x\|^2 = x_1^2 + x_2^2 + x_3^2$$

The constraints derived from the system are:

$$g_1(x) = 1x_1 + 2x_2 + 3x_3 - 3 = 0 \quad (1)$$

$$g_2(x) = 4x_1 + 1x_2 + 2x_3 - 2 = 0 \quad (2)$$

The Lagrangian is defined as:

$$\mathscr{L}(x_1, x_2, x_3, \lambda_1, \lambda_2) = x_1^2 + x_2^2 + x_3^2 - \lambda_1(1x_1 + 2x_2 + 3x_3 - 3) - \lambda_2(4x_1 + 1x_2 + 2x_3 - 2)$$

Taking the partial derivatives:

1. With respect to x_1:

$$\frac{\partial \mathscr{L}}{\partial x_1} = 2x_1 - \lambda_1 - 4\lambda_2 = 0 \quad (3)$$

2. With respect to x_2:

$$\frac{\partial \mathscr{L}}{\partial x_2} = 2x_2 - 2\lambda_1 - \lambda_2 = 0 \quad (4)$$

3. With respect to x_3:

$$\frac{\partial \mathscr{L}}{\partial x_3} = 2x_3 - 3\lambda_1 - 2\lambda_2 = 0 \quad (5)$$

From equations (3), (4), and (5): 1. From (3):

$$x_1 = \frac{\lambda_1 + 4\lambda_2}{2} \quad (6)$$

2. From (4):

$$x_2 = \frac{2\lambda_1 + \lambda_2}{2} \quad (7)$$

3. From (5):

$$x_3 = \frac{3\lambda_1 + 2\lambda_2}{2} \quad (8)$$

Substituting equations (6), (7), and (8) into constraints (1) and (2):

Substituting into (1):

$$\frac{\lambda_1 + 4\lambda_2}{2} + 2\left(\frac{2\lambda_1 + \lambda_2}{2}\right) + 3\left(\frac{3\lambda_1 + 2\lambda_2}{2}\right) = 3$$

Simplifying gives:

$$\frac{\lambda_1 + 4\lambda_2 + 4\lambda_1 + 2\lambda_2 + 9\lambda_1 + 6\lambda_2}{2} = 3$$

$$\frac{14\lambda_1 + 12\lambda_2}{2} = 3$$

$$14\lambda_1 + 12\lambda_2 = 6 \quad (9)$$

Substituting into (2):

$$4\left(\frac{\lambda_1 + 4\lambda_2}{2}\right) + 1\left(\frac{2\lambda_1 + \lambda_2}{2}\right) + 2\left(\frac{3\lambda_1 + 2\lambda_2}{2}\right) = 2$$

This simplifies to:

$$\frac{4\lambda_1 + 16\lambda_2 + 2\lambda_1 + \lambda_2 + 6\lambda_1 + 4\lambda_2}{2} = 2$$

$$\frac{12\lambda_1 + 21\lambda_2}{2} = 2$$

$$12\lambda_1 + 21\lambda_2 = 4 \quad (10)$$

Now we solve the system of equations (9) and (10): 1. From (9):

$$14\lambda_1 + 12\lambda_2 = 6$$

2. From (10):

$$12\lambda_1 + 21\lambda_2 = 4$$

Multiply the first equation by 3 and the second by 2:

$$42\lambda_1 + 36\lambda_2 = 18$$

$$24\lambda_1 + 42\lambda_2 = 8$$

Subtracting these gives:

$$(24 - 42)\lambda_1 + (42 - 36)\lambda_2 = 8 - 18$$

$$-18\lambda_1 + 6\lambda_2 = -10$$

$$3\lambda_2 - 9\lambda_1 = -5 \quad (11)$$

Now express λ_2 in terms of λ_1:

$$\lambda_2 = 3\lambda_1 - \frac{5}{3}$$

Substituting into (9):

$$14\lambda_1 + 12\left(3\lambda_1 - \frac{5}{3}\right) = 6$$

$$14\lambda_1 + 36\lambda_1 - 20 = 6$$

$$50\lambda_1 = 26$$
$$\lambda_1 = \frac{13}{25}$$

Now find λ_2:

$$\lambda_2 = 3\left(\frac{13}{25}\right) - \frac{5}{3} = -\frac{8}{75}$$

Now compute x_1, x_2, x_3 using equations (6), (7), and (8), substitute the values of λ_1 and λ_2 to find:

$$x_1 = \frac{\frac{13}{25} + 4\frac{-8}{75}}{2} = \frac{7}{150}, \quad x_2 = \frac{2\left(\frac{13}{25}\right) + \frac{-8}{75}}{2} = \frac{7}{15}, \quad x_3 = \frac{3\left(\frac{13}{25}\right) + 2\frac{-8}{75}}{2} = \frac{101}{150}$$

Thus, the solution x can be computed explicitly.

The final values are given by:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \frac{7}{150} \\ \frac{7}{15} \\ \frac{101}{150} \end{bmatrix}$$

`Matlab` code for this task is given below.

```
A=[1 2 3; 4 1 2];
b=[3;2];
x=pinv(A)*b;
disp("Solution of the system is:");
disp(x);
```

Output of the code is shown below.

```
Solution of the system is:
       7/150
       7/15
     101/150
```

Both analytical and computational approaches give the same result!

Hint$(x_1 - 1)^2 + (x_2 - 2)^2$+$(x_3 - 8)^2$;subject to Ax $= b$

Or min $(x-c)^T(x-c)$;subject to Ax $= b$ where $c = (128)^T$

5. Find solution to $x^T\left(A^T A\right)x$; subject to $x^T x = 1$

SOLUTION

To find the solution to the problem

$$\text{minimize } x^T(A^T A)x \quad \text{subject to } x^T x = 1,$$

we can use the method of Lagrange multipliers.
The Lagrangian function is given by:

$$\mathscr{L}(x, \lambda) = x^T(A^T A)x - \lambda(x^T x - 1),$$

where λ is the Lagrange multiplier.
To find the extrema, we need to compute the gradient of $\mathscr{L}$ and set it to zero:

$$\nabla_x \mathscr{L} = 2A^T Ax - 2\lambda x = 0.$$

This can be rearranged to:

$$A^T Ax = \lambda x.$$

The equation $A^T Ax = \lambda x$ suggests that x is an eigenvector of the matrix $A^T A$, and λ is the corresponding eigenvalue.
To minimize $x^T(A^T A)x$ under the constraint $x^T x = 1$, we are looking for the smallest eigenvalue of the matrix $A^T A$, along with its corresponding eigenvector.
The solution to the problem is to find the eigenvalues and eigenvectors of $A^T A$:
The minimum value of $x^T(A^T A)x$ subject to $x^T x = 1$ occurs at the eigenvector associated with the smallest eigenvalue of $A^T A$. - The constraint $xx^T = 1$ forces x to lie on the unit sphere in the feature space.

This is used in the derivation of SVD. Can you give geometrical interpretation to this problem, assuming rows of A are data points scattered around origin.

SOLUTION

This problem provides a way to understand the geometric distribution of data points and their inherent structure. By seeking the direction that minimizes the spread of data, we can uncover important characteristics about the data's arrangement in the space, which is crucial for dimensionality reduction techniques such as PCA and SVD. If the data points are tightly clustered around the origin, the direction of least variance (as found by this minimization) will point toward a direction where the points are closest to each other.

RESULTS

1. Solution of linear system is viewed in both algebraic and optimization approaches. It is found that, for large system of linear equations the least norm approach of Linear Algebra is more handy compared to Lagrange Multiplier method.

2. Given problems were solved in LA approach and optimization approach.

3. General constrained minimization problem $x^T(AA^T)x$ where $x^T x = 1$ is reformulated as an eigen value problem that leads to the SVD.

16 Assignment 48 Root of a Square Matrix and It's Applications

Root of a square matrix and one of it's application

Let A be $n \times n$ square matrix with distinct eigen values. Then by spectral decomposition

$A = S \wedge S^{-1}$. Let $B = S \wedge^{1/2} S^{-1}$.Then $B^2 = \left(S \wedge^{1/2} S^{-1}\right) S \wedge^{1/2} S^{-1} = S \wedge S^{-1} = A$

This means $B = \sqrt{A}$

Here $\Lambda = \begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \lambda_n \end{bmatrix}; \because \wedge^{1/2} = \begin{bmatrix} \pm\sqrt{\lambda_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \pm\sqrt{\lambda_n} \end{bmatrix} \Rightarrow \wedge^{1/2}$ is not unique

We have in general 2^nsolutions for B .

Generating multivariate Gaussian Random variable

How mvnrnd function in matlab works.

`mvnrand()` function is used for generating multivariate random variables with given mean vector and covariance matrix . Internally it generate random numbers from i.i.d (identically and independently distributed) standard Gaussian(normal) distributions and convert it into required random variables with given mean and covariance matrix. For convenience we assume mean vector is zero vector.

Note that covariance matrices are symmetric positive definite matrices.

(The reason you can deduce from formula for density function of multivariate normal distribution. Make a try)

It's eigen values are real and positive and eigen vectors are orthogonal. The principle with an Example

Let Y be required 2-tuple random variate with zero mean.

That is $Y = \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}. E(Y) = \begin{bmatrix} E(Y_1) \\ E(Y_2) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$.

$$\mathrm{Cov}(Y) = E\left(Y Y^T\right) = E \begin{bmatrix} {Y_1}^2 & Y_1 Y_2 \\ Y_2 Y_1 & {Y_2}^2 \end{bmatrix} = \begin{bmatrix} E\left({Y_1}^2\right) & E(Y_1 Y_2) \\ E(Y_2 Y_1) & E\left({Y_2}^2\right) \end{bmatrix} = \begin{bmatrix} V(Y_1) & \mathrm{Cov}(Y_1 Y_2) \\ \mathrm{Cov}(Y_2 Y_1) & V(Y_2) \end{bmatrix}$$

Let $\text{Cov}(Y) = \sum = \begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix}$

Compute eigen decomposition of $\sum = \begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix}$

$$\sum = S \wedge S^T$$

$$B = S\left(\sum\right)^{1/2} S^T$$

We generate two independent N(0,1)random variate. Denote that pair of random variable as X Generate X as two rows (say 2x100000 matrix) Each column is a two tuple. We have such 100000 samples .

Compute $Y = B * X$. It is a mapping that change independent random variates to dependent random variates with given covariance $\sum$

Proof:

E(X)=(0,0)'

Cov(X)=E(X*X')=[1 0; 0 1];

Let Y=B*X , B is symmetric

E(Y)=B*E(X)=(0,0)'

Cov(Y)=E(Y*Y')=B*E(X*X')B'=B*I*B'=B*B=$\sum$

```
close(gcf)
Sigma=[4 2;2 8];
[a b]=eig(Sigma);
b1=sqrt(b);
B=a*b1*a';
B   N=100000;
x1=randn(1,N);
x2=randn(1,N);
covx=(1/N)*X*X';
covy=(1/N)*Y*Y';
figure
plot(x1,x2,'*');
figure
plot(Y(1,:),Y(2,:),'o');
```

Assignment Questions

1. Find four square root matrices corresponding to matrix $A = \begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix}$

SOLUTION

Here the key concept is that, while taking square root of the eigen values, $\pm\sqrt{\lambda}$ will come. So for a 2×2 matrix, there are 4 such combinations. These different combinations can be created using a simple matrix multiplication:

$$c_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{16.1}$$

$$c_2 = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \tag{16.2}$$

$$c_3 = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \tag{16.3}$$

$$c_4 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{16.4}$$

$$\tag{16.5}$$

Using the matrix product, $b_i = c_i E$, where E is the diagonal matrix of eigen values of A. Now $B_i = a * b_i * a^{-1}$ define the i^{th} square root of A. `Matlab` code for and its output is given in below.

```
A=[2 1;1 4];
[a b1]=eig(A);
b1= sqrt(b1);
b2=[-1 0;0 -1].*b1;
b3=[1 0;0 -1].*b1;
b4=[-1 0;0 1].*b1;
B1=a*b1*inv(a);
disp("First square root of A:")
```

```
First square root of A:
```

```
disp(B1);
```

```
    1.3825    0.2976
    0.2976    1.9777
```

```
B1*B1 ;
B2=a*b2*inv(a);
disp("Second square root of A:")
```

```
Second square root of A:
```

```
disp(B2);
```

```
   -1.3825   -0.2976
   -0.2976   -1.9777
```

```
B2*B2;
B3=a*b3*inv(a);
disp("Third square root of A:")
```

```
Third square root of A:
```

```
disp(B3);
```

```
 0.7672   -1.1880
-1.1880   -1.6089
```

```
B3*B3 ;
B4=a*b4*inv(a);
B4*B4;
disp("Fourth square root of A:")
```

```
Fourth square root of A:
```

```
disp(B4);
```

```
-0.7672    1.1880
 1.1880    1.6089
```

2. Generate 100000 random variate from Gaussian distribution with following mean and covariance matrix by modifying above code

$$\mu = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}; \sum = \begin{bmatrix} 5 & 2 & 1 \\ 2 & 4 & 3 \\ 1 & 3 & 8 \end{bmatrix}$$

SOLUTION

Matlab code and its output for this task is given below.

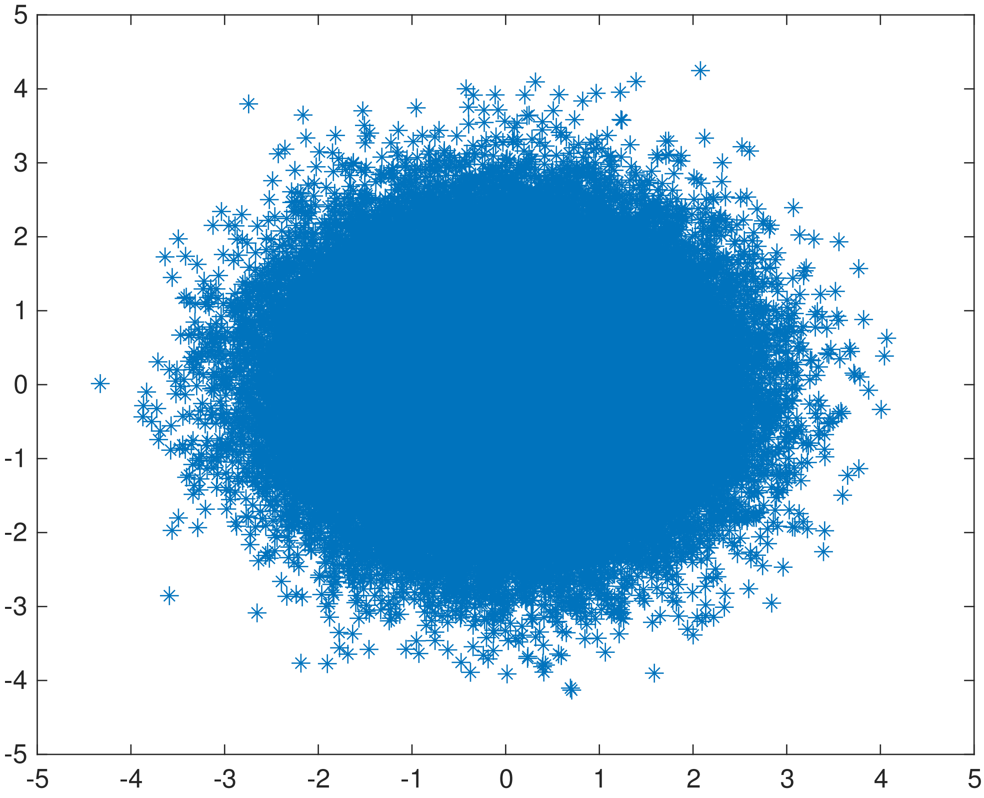
```
close(gcf)
Sigma=[5 2 1;2 4 3; 1 3 8];
[a b]=eig(Sigma);    % a is eigen vectors, diag(b) is eigen values
b1=sqrt(b); % find square root
B=a*b1*a'; % Generate transformation matrix
N=100000;
x1=randn(1,N); % independent Normal random variate
x2=randn(1,N);
x3=randn(1,N);
X=[x1; x2;x3]; % two-tuple values in colums
Y=B*X; % transform 2-tuple Xs to two-tuple Ys  % checking of
    Covariance of generated X samples
covx=(1/N).*(X*X')  % checking of Covariance of generated Y samples
```

```
covx = 3x3
    0.9970    0.0014   -0.0053
    0.0014    0.9963    0.0013
   -0.0053    0.0013    1.0011
```

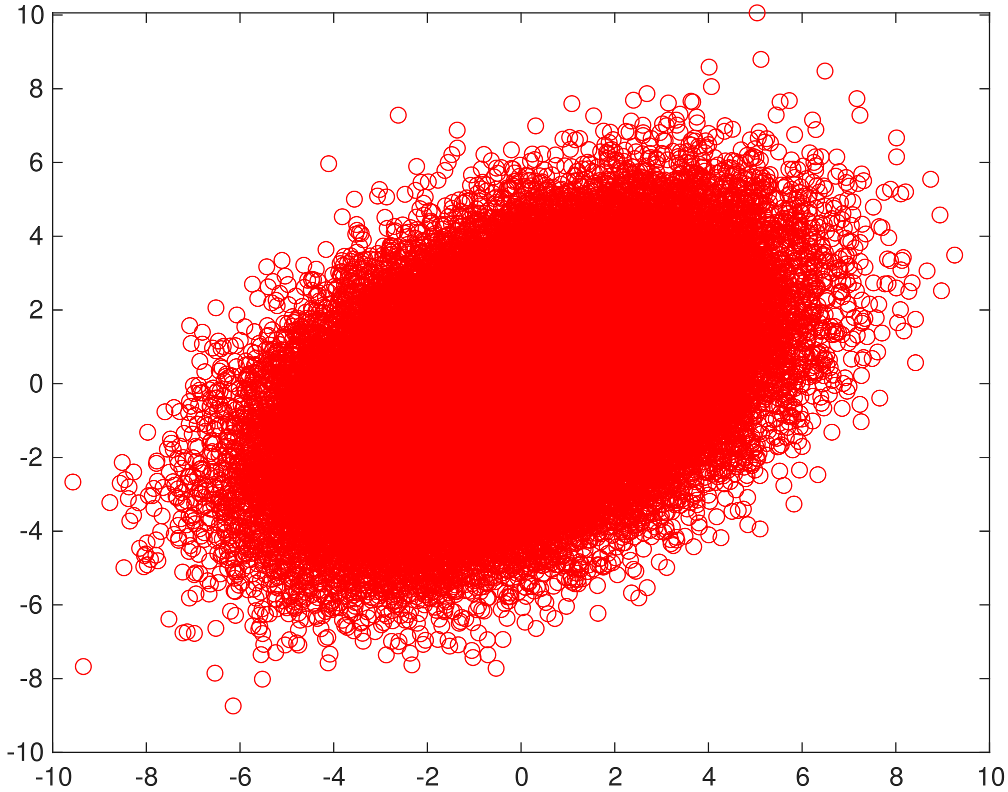
```
covy=(1/N).*(Y*Y')
```

```
covy = 3x3
    4.9850    1.9927    0.9703
    1.9927    3.9896    2.9980
    0.9703    2.9980    8.0073
```

```
figure
plot(x1,x2,'*');
```




```
figure
plot(Y(1,:),Y(2,:),'ro');
```



Where is this useful? One of its use is in the theory of 'Kalman Filters'. It is a family of classical control and information fusion algorithms. We will use this in Autonomous cars and UAV control applications.

RESULTS

1. Square root of a matrix is created from its spectral decomposition.
2. One application of square root of a square matrix is discussed