WOMEN INSTITUTE IN TECHNOLOGY UGANDA

DIPLOMA IN COMPUTER SCIENCE

COURSE UNIT: MATHEMATICAL COMPUTING

COURSE CODE: CSD115

LECTURER: TUGUME BRENDA

ASSIGNMENT ONE

GROUP 4

S/N	NAME	REGISTRATION NUMBER
1) 1	BRIDGET ATUHAIRE	2023/DCSE/0019/SS
2)	KWAGALA JOANNA VICTOR	2023/DCSE/0032/SS
3)	NAKAGGA SHANITAH	2023/DCSE/0055/SS
4)	MILAN PIFUA	2023/DCSE/0095/SS
5)	MBABAZI HAWA BINT EDRISA	2023/DCSE/0041/SS
6)	WANJIKU HANANNY	2023/DCSE/00106/SS
7)	NAMBUYA SHAMIRA AHMED	2023/DCSE/0074/SS
8)	ASAABA SHALLOT MASIKA	2023/DCSE/0014/SS

LINEAR ALGEBRA

Definition.

Linear algebra is a branch of mathematics that focuses on the study of vector spaces and linear mappings between these spaces. It provides a framework for representing and solving a wide range of problems in various fields, including mathematics, physics, engineering, computer science, and many others.

Key concepts in linear algebra

Vectors: Vectors are objects that have both magnitude and direction and are typically represented as arrows in space. They can be added together (vector addition) and multiplied by scalars (scalar multiplication).

Vector Spaces: A vector space is a collection of vectors that satisfies certain properties, including closure under addition and scalar multiplication. Common examples of vector spaces include the space of real numbers, the space of n-dimensional Euclidean vectors, and spaces of matrices.

Matrices: Matrices are rectangular arrays of numbers organized in rows and columns and they can be used to represent linear transformations and systems between vector spaces. Matrices can be added, multiplied, and manipulated to solve systems of linear equations and perform various transformations.

Linear Transformations: A linear transformation is a function that maps vectors from one vector space to another while preserving the basic structure of vector spaces for example rotations, scalings and projections. Linear transformations are often represented by matrices.

Eigenvalues and Eigenvectors: Eigenvalues and eigenvectors are important concepts in linear algebra. They are used to analyze and understand the behavior of linear transformations and have applications in various fields, including quantum mechanics, computer graphics, and data analysis.

Determinants: The determinant of a square matrix is a scalar value that provides information about the matrix's invertibility and the scaling factor of linear transformations represented by the matrix.

Solving Systems of Linear Equations: Linear algebra provides methods for solving systems of linear equations, both numerically and symbolically. These methods are fundamental in engineering, physics, and many other scientific disciplines.

Inner Product Spaces: An inner product space is a vector space equipped with an inner product (or dot product) that measures the angle between vectors and their lengths. Euclidean space is an example of an inner product space.

Orthogonal vectors: Vectors are said to be orthogonal if their inner product is zero. Orthogonal vectors are important in many applications, including orthogonal bases for vector spaces and orthogonal projections.

Note: basically in python numpy library is mostly used for working with matrices and vectors

Why do we study Linear Algebra?

Linear algebra is a foundational mathematical tool that provides computer scientists with the necessary framework to solve a wide range of problems, particularly those involving data manipulation, graphics, machine learning, and scientific computing. It is a crucial part of a computer scientist's toolkit and is used extensively in both theoretical and practical applications in the field.

- **Graphics and Computer Vision:** Understanding linear algebra is crucial for creating realistic computer-generated images and for developing applications like augmented reality and facial recognition.
- **Machine Learning and Data Science:** Concepts like vector spaces, matrices, and eigenvalues are used in feature engineering, dimensionality reduction, and solving optimization problems in machine learning.
- **Data Representation:** Linear algebra provides the tools to manipulate and analyze data efficiently. For example, you can perform matrix operations, compute dot products, and use matrix factorization techniques for data analysis.
- **Game Development:** In computer game development, linear algebra is essential for tasks like rendering graphics, physics simulations, and controlling the movement of objects.

Game engines rely heavily on linear algebra to achieve realistic and immersive gaming experiences.

- **Computer Networks**: Linear algebra can be used to analyze and optimize data flow in computer networks. Concepts like graph theory and matrix manipulation play a role in designing efficient routing algorithms and network protocols.
- **Artificial Intelligence**: Neural networks and deep learning models are often described using matrices and vectors. Understanding linear transformations, weight matrices, and activation functions is crucial for developing and training AI models.
- **Data Compression and Encryption**: Techniques like Singular Value Decomposition (SVD) and various cryptographic algorithms use linear algebra concepts. These are essential for data compression, encryption, and security in computer systems.
- **Simulation and Modeling**: Linear algebra is used in simulations and modeling to represent physical systems as sets of equations that can be solved numerically. This is essential in fields like physics, engineering, and computer-aided design.
- **Computer Programming**: Many programming libraries and frameworks used in computer science, such as NumPy (for Python) and MATLAB, provide extensive support for linear algebra operations. Knowledge of linear algebra helps programmers leverage these libraries effectively.

VECTORS

Vectors are mathematical objects that have both magnitude and direction and are often represented as arrows in space.

When working with vectors, you can perform two fundamental operations: combining vectors and scaling vectors.

1) **Combining vectors;**
   a) **Vector Addition:** You can combine two vectors by adding them together. Mathematically, if you have two vectors A and B, their sum (A + B) is found by adding their corresponding components.
      For example, $A = (A_x, A_y)$, $B = (B_x, B_y)$
      $A + B = (A_x + B_x, A_y + B_y)$
   b) **Vector Subtraction:** Vector subtraction is similar to vector addition, but you subtract one vector from another. $A - B = (A_x - B_x, A_y - B_y)$.
   c) **Scalar Multiplication:** You can also scale a vector by a scalar (a real number). This operation involves multiplying each component of the vector by the scalar value. Scalar multiplication changes the magnitude of the vector but not its direction.
      $c * A = (c * A_x, c * A_y)$
2) **Scaling Vectors:**
   a) **Magnitude:** The magnitude (or length) of a vector is a scalar value that represents the vector's size. For a vector $A = (A_x, A_y)$, its magnitude (often denoted as $||A||$ or $|A|$) is calculated using the Pythagorean Theorem: $||A|| = \text{sqrt}(A_x^2 + A_y^2)$.
   b) **Normalization:** Normalizing a vector means scaling it to have a magnitude of 1 while preserving its direction. To normalize a vector A, you divide each component by its magnitude: for example, $A_{normalized} = (A_x / ||A||, A_y / ||A||)$.
   c) **Scalar Projection:** The scalar projection of one vector onto another measures the component of one vector in the direction of another vector. Given vectors A and B, the scalar projection of A onto B is calculated as: Scalar Projection of A onto B = $||A|| * \cos(\theta)$; where θ is the angle between vectors A and B.
   d) **Vector Projection:** The vector projection of one vector onto another is the vector component of one vector in the direction of another vector. It is calculated as the scalar

projection of A onto B multiplied by the unit vector of B: Vector Projection of A onto B = (Scalar Projection of A onto B) * (Unit Vector of B).

TRANSFORMING VECTORS AND MATRICES

Transforming vectors and matrices involves applying linear transformations to change their positions, orientations, or values. Linear transformations are mathematical operations that map vectors or matrices from one space to another while preserving certain properties.

Types of vector transformations;

a) **Translation:** This shifts a vector by a fixed amount in a specified direction.
b) **Rotation:** This changes the orientation of a vector about an axis or a point.
c) **Scaling:** This changes the size of a vector by multiplying its components by scaling factors.

Types of vector transformations;

a) **Matrix Multiplication:** Matrices can be transformed by multiplying them with other matrices, which represents a composition of linear transformations. If you have a matrix A and a matrix B, their product AB results in a new matrix that combines the transformations represented by A and B.
b) **Matrix Transposition:** Transposing a matrix involves flipping it along its main diagonal, which swaps rows and columns. For a matrix A with dimensions m x n, its transpose A^T has dimensions n x m, and its elements are rearranged as A^T[i][j] = A[j][i].
c) **Matrix Inversion:** Not all matrices are invertible, but when a matrix is invertible (non-singular), you can find its inverse. Given a matrix A, its inverse A^ (-1) satisfies the equation A * A^ (-1) = I, where I is the identity matrix. Inverting a matrix is useful for solving linear systems and undoing transformations.
d) **Eigenvalue and Eigenvector Transformation:** Eigenvalues and eigenvectors of a matrix represent transformations that stretch or compress space along specific directions. Transformations using eigenvalues and eigenvectors are used in various applications, such as principal component analysis (PCA) and spectral decomposition.
e) **Matrix Decomposition:** Matrices can be decomposed into simpler matrices to understand their structure better. Common matrix decompositions include LU decomposition, QR

decomposition, and singular value decomposition (SVD). These decompositions are used for solving linear systems, finding orthogonal bases, and dimensionality reduction.

SYSTEMS OF LINEAR EQUATIONS AND INVERSE MATRICES

A system of linear equations consists of multiple linear equations involving the same set of variables.

These equations are typically written in the form;

$a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = b_1$

$a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = b_2$

...

$a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n = b_m$

Where; **m** is the number of equations, **n** is the number of variables, $\mathbf{a_{ij}}$ represents the coefficients of the variables, $\mathbf{x_1, x_2, ..., x_n}$ are the variables to be solved for, $\mathbf{b_1, b_2, ..., b_m}$ are the constants on the right-hand side of each equation.

The system of a matrix is represented in the form as **Ax = b**, where:

A is the coefficient matrix, with dimensions **m x n**.

x is the column vector of variables, with dimensions **n x 1**.

b is the column vector of constants, with dimensions **m x 1**.

In this matrix form, the system becomes: A * x = b

An inverse matrix, denoted as **A^ (-1)**, is a matrix that, when multiplied by the original matrix **A**, results in the identity matrix **I**

DOT PRODUCT

Definition

Dot product is also **scalar product** and it's a measure of how closely two **vectors** align in terms of direction.

Dot product formula

This can be denoted either algebraically or geometrically

Geometrically; The geometric meaning of dot product is that the dot product between two given vectors a and b is denoted by: $a.b = |a||b|cos\theta$

Here, $|a|$ and $|b|$ are called the magnitudes of vectors a and b and θ is the angle between the vectors a and b. If the two vectors are orthogonal, that is, the angle between them is 90, then $a.b = 0$ since cos 90 = 0. If the two vectors are parallel to each other, then : $a.b = |a||b|$since $cos0 = 1$

Algebraically; The dot product algebra says that the dot product of the given two products

a = (a_1, a_2, a_3) and b= (b_1, b_2, b_3) is given by: a.b= ($a_1b_1 + a_2b_2 + a_3b_3$)

Properties of the dot product

- Commutative Property; For any two vectors a and b, a.b= b.a
- Scalar Multiplication Property; For any two vectors a and b and any real number z, (za)·b = a·(zb) = z(a·b) or xy(a.b)=(xa.yb)
- Distributive Property; For any 3 vectors a, b and c, a·(b+c) = a·b + a·c
- Orthogonal Property; Two vectors are orthogonal only when a.b = 0

Numpy dot (Numerical python)

The numpy.dot function is used to return the dot product of given arrays/matrix. It can handle 2-D arrays but it considers them as matrix and it will perform matrix multiplication.

Finding the magnitude of a vector in numpy

We use the **norm()** method in **linalg** (linalg stands for linear algebra) module of numpy. The **numpy.linalg.norm()** method is used to get the magnitude of a vector.

MATRIX DECOMPOSITION

Matrix decompositions are methods that reduce a matrix into constituent parts that make it easier to calculate more complex matrix operations. Matrix decomposition methods, also called matrix factorization methods, are a foundation of linear algebra in computers, even for basic operations such as solving systems of linear equations, calculating the inverse, and calculating the determinant of a matrix.

It is an approach that can simplify more complex matrix operations that can be performed on the decomposed matrix rather than on the original matrix itself.

A common similarity for matrix decomposition is the factorization of numbers, such as the factorizing of 10 into 2 x 5. For this reason, matrix decomposition is also called **matrix factorization.** Like factorizing real values, there are many ways to decompose a matrix, hence there are a range of different matrix decomposition techniques. The mainly used techniques are LU matrix decomposition and the QR matrix decomposition.

The LU matrix decomposition

The LU decomposition is for square matrices and decomposes a matrix into L and U components.

A = L.U Or, without the dot notation A =LU where A is the square matrix that we wish to decompose, L is the lower triangle matrix and U is the upper triangle matrix.

The LU decomposition can be implemented in **NumPy using the lu() function** after import it
from numpy.linalg import lu

The QR matrix decomposition

The QR decomposition is for m, x, n matrices (not limited to square matrices) and decomposes a matrix into Q and R components.

A=Q.R Or, without the dot notation A=QR where A is the matrix that we wish to decompose, Q a matrix with the size m x m, and R is an upper triangle matrix with the size m x n.

The QR decomposition can be implemented in **NumPy using the qr() function** after importing it **from numpy.linalg import qr.** By default, the function returns the Q and R matrices with smaller or 'reduced' dimensions that is more economical. We can change this to return the expected sizes of m x m for Q and m x n for R by specifying the mode argument as 'complete', although this is not required for most applications.

Cholesky decomposition

The Cholesky decomposition is for square symmetric matrices where all eigenvalues are greater than zero, so-called positive definite matrices.

For our interests in machine learning, we will focus on the Cholesky decomposition for real-valued matrices and ignore the cases when working with complex numbers.

The decomposition is defined as follows:

A=L.L^T Or without the dot notation A=LL^T where A is the matrix being decomposed, L is the lower triangular matrix and L^T is the transpose of L. The decompose can also be written as the product of the upper triangular matrix, for example:

A=U^T.U Where U is the upper triangular matrix.

The Cholesky decomposition is used for solving linear least squares for linear regression, as well as simulation and optimization methods.

When decomposing symmetric matrices, the Cholesky decomposition is nearly twice as efficient as the LU decomposition and should be preferred in these cases.

REFERENCES

Mathworld: Dot Product

Dot product definitions and examples at Texas A&M

Cormen, “Introduction to Algorithms”, Chapter 15.2, p. 370-378

https://en.wikipedia.org/wiki/Matrix_chain_multiplication

Page 97, Introduction to Linear Algebra, Fifth Edition, 2016.

Page 100, Numerical Recipes: The Art of Scientific Computing, Third Edition, 2007.

Matrix decomposition on Wikipedia