



Hochschule  
Bonn-Rhein-Sieg  
University of Applied Sciences



# Linear Algebra

## A (re-)introduction

March 24, 2024

Shinas Shaji

# 1. Introduction

## 2. Basic Concepts

### 2.1 Vectors

### 2.2 Matrices

## 3. Systems of Linear Equations

### 3.1 Matrix Representations

## 4. Basis Vectors and Vector Spaces

### 4.1 Prof. Plöger's notes

### 4.2 Change of Basis

## 5. Linear Transformations

### 5.1 Rotations

### 5.2 Scaling

## 6. Homogeneous Transforms

## 7. Eigenvalues and Eigenvectors

### 7.1 Singular Value Decomposition



# Linear Algebra

*What is it?*

- Deals with linear equations of the form:

$$a_1x_1 + \dots + a_nx_n = b$$

- Represented as systems of equations, matrices and vectors
- **Python Coding for Linear Algebra** in a separate session

# Linear Algebra

*Why is it important?*

- A powerful way of representing and solving problems in:
  - physics and engineering
  - computer vision and graphics
  - robotics
  - economics
  - and more
- Fundamental in understanding:
  - geometry
  - optimization
  - machine learning
  - and more

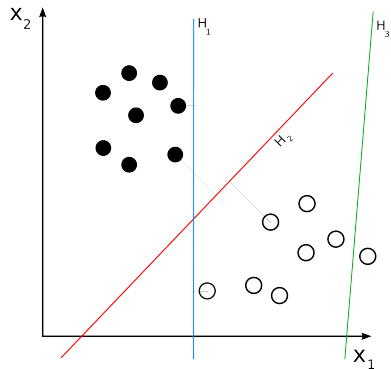


Figure 1: Linear Classifiers - Image from Wikipedia, by Cyclic; Public Domain

# In Machine Learning

## Example in Single-Layer Perceptron

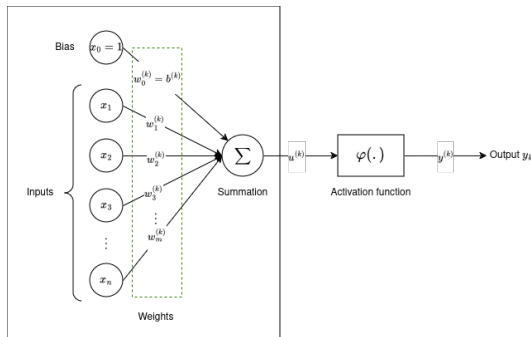


Figure 2: Single-Layer Perceptron - Image by Harley Lara

The highlighted part is of the form:  $x_1 w_1 + \dots + x_n w_n + x_0$

# In Computer Graphics

*Example from Blender*

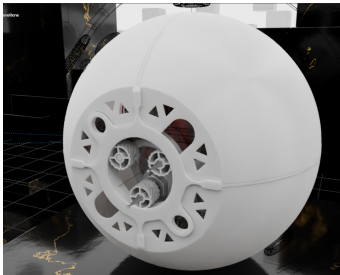


Figure 3: Rendered view

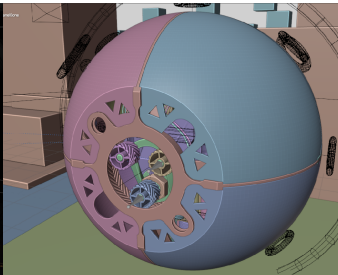


Figure 4: Solid view

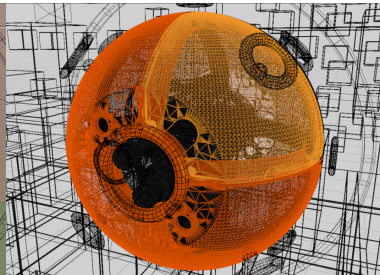


Figure 5: Geometry / wireframe

**It's all lines** (mostly)

## 1. Introduction

## 2. Basic Concepts

### 2.1 Vectors

### 2.2 Matrices

## 3. Systems of Linear Equations

### 3.1 Matrix Representations

## 4. Basis Vectors and Vector Spaces

### 4.1 Prof. Plöger's notes

### 4.2 Change of Basis

## 5. Linear Transformations

### 5.1 Rotations

### 5.2 Scaling

## 6. Homogeneous Transforms

## 7. Eigenvalues and Eigenvectors

### 7.1 Singular Value Decomposition



# Vectors

- Quantities with magnitude and direction
- Represent positions, velocities, accelerations in space
- Consider vector  $P$  from  $(0, 0)$  to  $(5, 4)$ 
  - $\|P\| = 6.4031$  - distance from origin
  - $\angle P = 38.659^\circ$  - angle from origin
- How was this calculated?

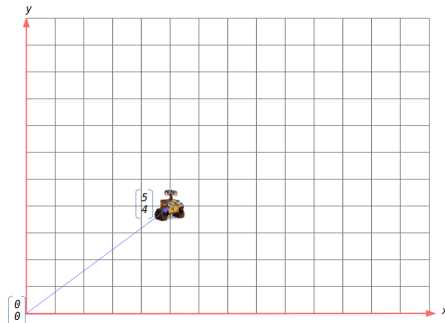


Figure 6: Position vector - Image from a previous session by Divin and Santosh



# Vector Norms

- Represent the magnitude of a vector
- Mapping from vector space to non-negative real numbers
  - L1 norm:
$$\|x\|_1 = \sum_{i=1}^n |x_i|$$
  - L2 norm (Euclidean distance):
$$\|x\|_2 = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$$
  - p-norm:
$$\|x\|_p = (\sum_{i=1}^n |x_i|^p)^{\frac{1}{p}}$$
- Use `numpy.linalg.norm()`

# Vector Angles

- Represent direction of vector
- Given by  $\tan^{-1}\left(\frac{y}{x}\right)$
- Also given by  $\tan^{-1}\left(\frac{\text{opposite}}{\text{adjacent}}\right)$
- Use `numpy.arctan2()` and `numpy.arctan()`
- `numpy.arctan2()` considers quadrants
- Use `numpy.rad2deg()` or `numpy.deg2rad()` for conversions

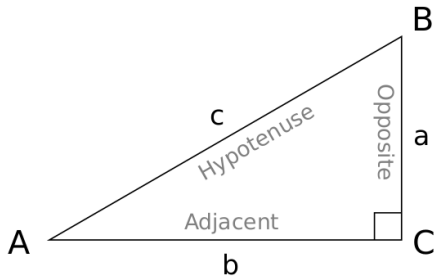


Figure 7: Triangle sides for  $\angle A$  From Wikipedia; By TheOtherJesse - Own work, Public Domain

# Matrices

- Multi-dimensional arrays of elements
- Dimensions given by number of rows, columns, depth, and so on
- Can be added, subtracted, multiplied (according to rules)
- Elegant way to represent
  - vectors, rotations, translations (and composite transformations)
  - systems of equations
  - masks, kernels, images, and so on
- `numpy.array()`, `numpy.zeros()`, `numpy.ones()`, `numpy.identity()`, `numpy.eye()` useful for matrix construction

# Matrix Operations

## *Notable rules*

- Addition (**translation**) possible with matrices of same dimensions  
e.g.  $(m \times n)$  and  $(m \times n)$ , or  $(n)$  and  $(n)$
- Scalar multiplication with a scalar
- Matrix multiplication possible with arrays of equal adjacent dimension  
e.g.  $(m \times n)$  and  $(n \times o)$  giving  $(m \times o)$
- Important to note when performing operations, debugging code

# 1. Introduction

## 2. Basic Concepts

### 2.1 Vectors

### 2.2 Matrices

## 3. Systems of Linear Equations

### 3.1 Matrix Representations

## 4. Basis Vectors and Vector Spaces

### 4.1 Prof. Plöger's notes

### 4.2 Change of Basis

## 5. Linear Transformations

### 5.1 Rotations

### 5.2 Scaling

## 6. Homogeneous Transforms

## 7. Eigenvalues and Eigenvectors

### 7.1 Singular Value Decomposition



# Systems of Linear Equations

*and their representations*

- Two or more equations of the same set of variables, e.g.:

$$2x + 3y = 7$$

$$x - y = 1$$

- Various methods exist to solve such systems:
  - substitution
  - elimination
  - graphical methods
  - matrix methods

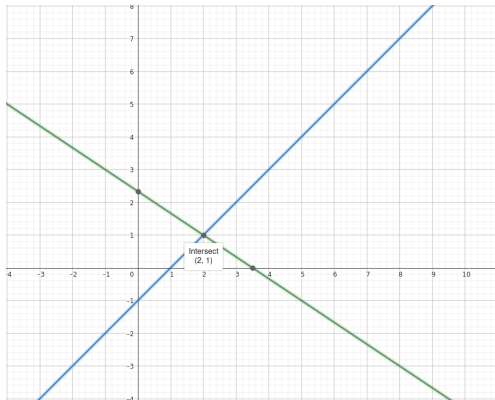


Figure 8: Graphical representation plotted in [GeoGebra](#)

# Systems of Linear Equations

## Matrix Representations

- Represented in matrix form as:

$$\begin{bmatrix} 2 & 3 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 7 \\ 1 \end{bmatrix}$$

- or:

$$\begin{bmatrix} 2 & 3 & -7 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- Can then be solved with
  - Gaussian elimination
  - Least-squares (`numpy.linalg.lstsq()`)
  - Singular Value Decomposition (`numpy.linalg.svd()`) (this may seem magical)
- More on these methods in **Mathematics for Robotics and Control**

- 1. Introduction
- 2. Basic Concepts
  - 2.1 Vectors
  - 2.2 Matrices
- 3. Systems of Linear Equations
  - 3.1 Matrix Representations
- 4. Basis Vectors and Vector Spaces
  - 4.1 Prof. Plöger's notes
  - 4.2 Change of Basis
- 5. Linear Transformations
  - 5.1 Rotations
  - 5.2 Scaling
- 6. Homogeneous Transforms
- 7. Eigenvalues and Eigenvectors
  - 7.1 Singular Value Decomposition



# Basis Vectors and Vector Spaces

*What are they?*

- Set of linearly independent vectors  $B$  that span a vector space  $V$
- **Unique** linear combination of basis vectors  $B$  can represent any vector in space  $V$
- Minimal number of vectors, but maximal span
- $B$  can be ordered
- Number of basis vectors – **dimension**
- e.g. 2D Cartesian space in  $\mathbf{R}^2$  spanned by  $[1, 0]$  and  $[0, 1]$

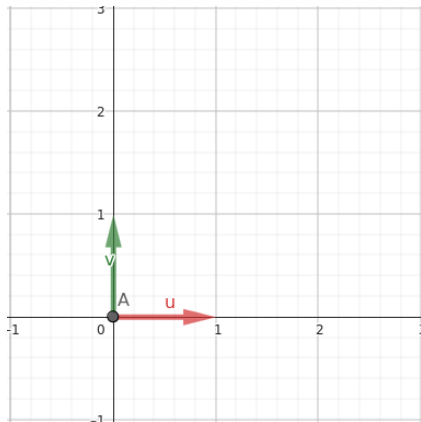


Figure 9: 2D Cartesian Basis Vectors, plotted in [GeoGebra](#)

# Basis Vectors and Vector Spaces

*How are they used?*

- Unit vectors of desired basis directions

e.g.  $\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ ,  $\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ ,  $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

- Compose as column vectors –

we get  $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

- Dot product to project vectors into this space
- Useful in **Principal Component Analysis** (PCA)

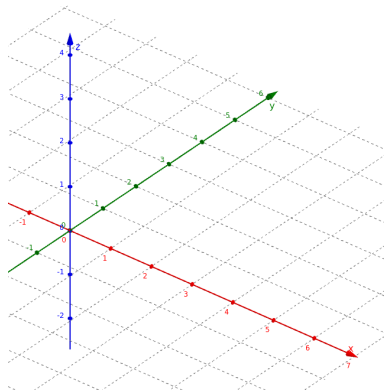


Figure 10: 3D Cartesian Space, shown in GeoGebra

# Basis Vectors

Prof. Plöger's notes

- “Any set of vectors which fulfills:
  - number == N (dimension of domain), and
  - are linearly independent”
- Consider  $\mathbf{R}^2$  basis  $B_{my}$  with vectors  $b_1 = [1, 1]$  and  $b_2 = [-1, 1]$

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} = 0.5 \begin{bmatrix} 1 \\ -1 \end{bmatrix} + 0.5 \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix} = -0.5 \begin{bmatrix} 1 \\ -1 \end{bmatrix} + 0.5 \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$



Figure 11: An alternative basis for  $\mathbf{R}^2$ – Prof. Paul G. Plöger – plotted in [GeoGebra](#)

# Basis Vectors

Prof. Plöger's notes

- Compared to regular unit vectors, vectors in  $B_{my}$ :
  - are **not** unit length
  - are **not** axis aligned
- However, still perpendicular
- Need not be the case for a general basis  $B$



Figure 12: An alternative basis for  $\mathbf{R}^2$  – Prof. Paul G. Plöger – plotted in [GeoGebra](#)

# Change of Basis

*What is it?*

- Expresses vector coordinates in one basis relative to another basis
- In matrix notation, written as:

$$\mathbf{x}_1 = A\mathbf{x}_2$$

where

- $\mathbf{x}_1$  and  $\mathbf{x}_2$  are vector coordinates in each basis
  - $A$  is the **change of basis matrix**
- A projection into a vector space
- Projection into spaces useful in **Principal Component Analysis** (PCA) and **dimensionality reduction**

- 1. Introduction
- 2. Basic Concepts
  - 2.1 Vectors
  - 2.2 Matrices
- 3. Systems of Linear Equations
  - 3.1 Matrix Representations
- 4. Basis Vectors and Vector Spaces
  - 4.1 Prof. Plöger's notes
  - 4.2 Change of Basis
- 5. Linear Transformations
  - 5.1 Rotations
  - 5.2 Scaling
- 6. Homogeneous Transforms
- 7. Eigenvalues and Eigenvectors
  - 7.1 Singular Value Decomposition

# Linear Transformations

## *Definitions and Properties*

- Transforms that **preserve** vector addition, scalar multiplication
  - rotation
  - scaling
  - reflection
- Defined as a function  $T : V \rightarrow W$ , where  $V, W$  are vector spaces with  $\mathbf{u}, \mathbf{v} \in V$  such that:
  1.  $T(\mathbf{u} + \mathbf{v}) = T(\mathbf{u}) + T(\mathbf{v})$
  2.  $T(k\mathbf{v}) = kT(\mathbf{v})$
- **Matrices** can represent arbitrary linear transformations
- Transform vectors with a **transformation matrix**
- Translation is **not** a linear transform
  - Does not preserve above properties
  - Shifts vector / space without changing orientation, shape

# Rotations

## As Linear Transforms

- Represented by **rotation matrices**
- Consider **counterclockwise** rotation by  $\theta$  in 2D

$$x' = x \cos \theta - y \sin \theta$$

$$y' = x \sin \theta + y \cos \theta$$

- Can be represented in matrix form as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

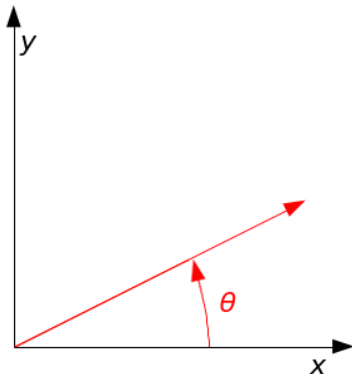


Figure 13: Vector rotation – From Wikipedia; By MathsPoetry - Own work, CC BY-SA 3.0



# Rotation

*As change in basis*

- Rotation using rotation matrix  $R$  also **change in basis**
  - Sufficiently many column vectors (since square and  $R^{-1} = R^T$ )
  - Linearly independent column vectors (since  $\det(R) = 1$ )

- Written as

$$\mathbf{x}' = R\mathbf{x}$$

- $R$  is a **change of basis** matrix
- More on rotations in **Mathematics for Robotics and Control**
  - Axis angle and Euler angle representations
  - 3D composite rotations and rotation matrices

# Scaling

## *As a Linear Transform*

- Changes length or magnitude of vector
- Uniform 2D scaling:

$$k\mathbf{x} = k\mathbf{I}\mathbf{x} = \begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix} \mathbf{x}$$

- 2D scaling along an axis:

$$\begin{bmatrix} k & 0 \\ 0 & 1 \end{bmatrix} \mathbf{x}$$

- 1. Introduction
- 2. Basic Concepts
  - 2.1 Vectors
  - 2.2 Matrices
- 3. Systems of Linear Equations
  - 3.1 Matrix Representations
- 4. Basis Vectors and Vector Spaces
  - 4.1 Prof. Plöger's notes
  - 4.2 Change of Basis
- 5. Linear Transformations
  - 5.1 Rotations
  - 5.2 Scaling
- 6. Homogeneous Transforms
- 7. Eigenvalues and Eigenvectors
  - 7.1 Singular Value Decomposition

# Homogeneous Transforms

## Affine Transforms

- Represent rotations, translations, scaling, and combinations
- Square matrix of  $N + 1$ 
  - $N$  columns represent rotation, last column represents translation

$$T = \begin{bmatrix} R & & t \\ & \ddots & \vdots \\ 0 & \dots & 1 \end{bmatrix}$$

- Compose transformations by multiplication

$$T = T_a \cdot T_b$$

- Inverse transform represented by  $T^{-1}$
- Important to note transformation **frames**, order of composition
  - More on transforms and frames in **Mathematics for Robotics and Control**

1. Introduction
2. Basic Concepts
  - 2.1 Vectors
  - 2.2 Matrices
3. Systems of Linear Equations
  - 3.1 Matrix Representations
4. Basis Vectors and Vector Spaces
  - 4.1 Prof. Plöger's notes
  - 4.2 Change of Basis
5. Linear Transformations
  - 5.1 Rotations
  - 5.2 Scaling
6. Homogeneous Transforms
7. Eigenvalues and Eigenvectors
  - 7.1 Singular Value Decomposition

# Eigenvalues and Eigenvectors

*What are they?*

- Vectors  $\mathbf{v}$  with unchanged direction under a linear transform  $T$  – **eigenvectors**
- Only scaled by constant factor  $\lambda$  under linear transform  $T$  – **eigenvalues**

$$T(\mathbf{v}) = \lambda \mathbf{v}$$

- For a transformation matrix  $A$

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

- Compute by solving **characteristic equation** of  $A$

$$\det(A - \lambda I) = 0$$

# Eigenvalues and Eigenvectors

## Visual Example

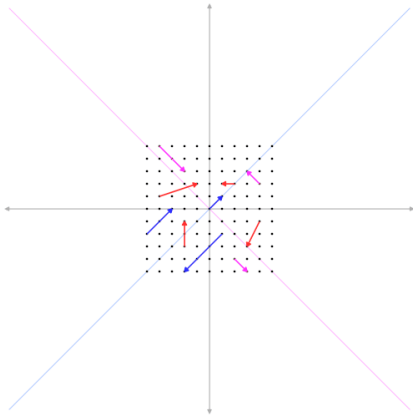


Figure 14: Vectors on a grid – From Wikipedia; By Lucas Vieira - Own work, Public Domain

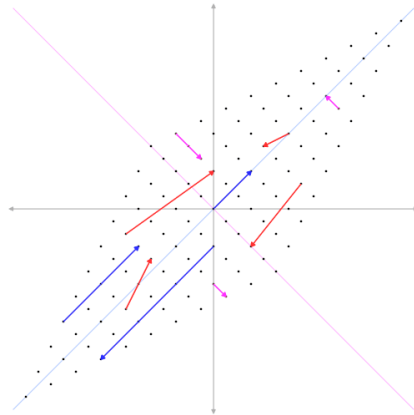


Figure 15: Scaling diagonally – note vectors parallel to drawn diagonals – From Wikipedia; By Lucas Vieira - Own work, Public Domain

# Eigenvalues and Eigenvectors

## *From Visual Example*

- Shown was the transformation of

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

- Eigenvectors and eigenvalues of this transform are

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix}, 1 \quad \text{and} \quad \begin{bmatrix} 1 \\ 1 \end{bmatrix}, 3$$

- Vectors along  $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$  are scaled by 3; vectors along  $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$  are scaled by 1
- Help understand the basic effects of a transformation, among others



# Eigenvalues and Eigenvectors

*How to find them*

- Use `numpy.linalg.eig()` to get eigenvalues and eigenvectors
- Can use `numpy.linalg.eigh()` for symmetric matrices – sorted output
- **Singular Value Decomposition** (SVD) returns a decomposition of data matrix  $A$  such that (for real valued matrices):

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

where

- Columns of  $U$  are eigenvectors of  $A^T A$ ; left-singular vectors of  $A$
- Elements of  $\Sigma$  represent eigenvalues of  $A^T A$  or  $AA^T$ ; squared singular values of  $A$
- Rows of  $V^T$  are eigenvectors of  $AA^T$ ; right-singular vectors of  $A$
- $A^T A$  or  $AA^T$  is symmetric, positive semi-definite (non-negative eigenvalues, orthogonal eigenvectors)
- Use `numpy.linalg.svd()` for computing SVD

# Singular Value Decomposition

*and its uses*

- Singular vectors capture directions where data in  $A$  is spread or aligned
- Closely related to **principal components** – directions in which data points vary the most – capture maximum variance
- SVD useful for
  - **Principal Component Analysis** (PCA)
  - **dimensionality reduction**
  - **data compression** and **reconstruction**
  - **pseudoinverse** ( $A^+$ )
  - solution of homogeneous linear equations
  - least squares solution
  - und so weiter. . .
- More on SVD and its applications in **Mathematics for Robotics and Control**

# Singular Value Decomposition

## *Principal Component Analysis*

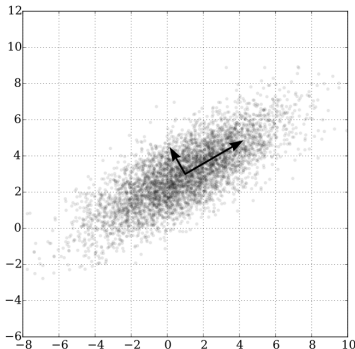


Figure 16: PCA of a multivariate Gaussian distribution – note principal vectors along directions of maximum variance – From Wikipedia; By Nicoguardo - Own work, CC BY 4.0