Linear Algebra

# Vector Part 1:
# Vector and Basic Operation of Vector

Automotive Intelligence Lab.

HANYANG UNIVERSITY

# Contents

# Generating and visualizing vectors with matlab

# Vector

## Vector

- Representations of numbers or symbols in a **one-dimensional** array.

## Notation for vectors

- **vectors** are typically denoted by bold lowercase Roman letters, such as $\mathbf{v}$.
- other expression : italicized ($v$) / with an arrow above ($\vec{\mathbf{v}}$).

## Characteristics of vectors

- Dimensionality: the number of elements a vector contains.
  - Represented as $\mathbb{R}^N$
    - $\mathbb{R}$ : Real Number
    - N : Dimension
- Orientation: indicates whether the vector is in column or row orientation.

# Column and Row Vector

## Column vector (or vector)

- A matrix with only one column.
- Each element of the vector is expressed as a **vertical** array.
- Column vectors are often represented as $\boldsymbol{v}$.
- Vectors are in column orientation unless otherwise specified.

## Row vector

- A matrix with only one row.
- Each element of the vector is expressed as a **horizontal** array.
- Row vectors are often represented as $\boldsymbol{w}^{\boldsymbol{T}}$.
- T represents the transpose operation.

$$\boldsymbol{x} = \begin{bmatrix} 1 \\ 4 \\ 5 \\ 6 \end{bmatrix}, \boldsymbol{y} = \begin{bmatrix} .3 \\ -7 \end{bmatrix}, \boldsymbol{z} = \begin{bmatrix} 1 & 4 & 5 & 6 \end{bmatrix}$$

Example of Column Vector and Row Vector

$\boldsymbol{x}$ is a $\boxed{\phantom{0}}$ dimensional $\boxed{\phantom{column}}$ vector
$\boldsymbol{y}$ is a $\boxed{\phantom{0}}$ dimensional $\boxed{\phantom{column}}$ vector
$\boldsymbol{z}$ is a $\boxed{\phantom{0}}$ dimensional $\boxed{\phantom{row}}$ vector

$\boldsymbol{x} \in \mathbb{R}^4$ can also be written.

# Transpose



- **Convert row vector to column vector or vice versa, effectively flipping its orientation.**
  - Transpose of a row vector = ______ vector.
  - Transpose of a column vector = ______ vector.

- **Notation**
  - Transpose of $\boldsymbol{v}$ = $\boldsymbol{v}^{\boldsymbol{T}}$.

- If we transposing **vector** twice, it returns the vector to its ______ **orientation.**
  - So, $\boldsymbol{v}^{TT} = \boldsymbol{v}$.

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}$$

Transpose of column vector

$$\begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Transpose of row vector

# Does Vector Orientation Matter?

- **It depends on how you use vectors!**

- **In case of using vectors to store data**
  - Orientation of vector usually doesn't matter.
  - The difference is simply whether to stack information

- **In case of using vectors to perform operations**
  - Orientation of vector does matter.
    - We will study properties of vector operations which the orientation of vector is important.
  - Operation results vary depending on the orientation of vector.

# Generating and Visualizing Vectors with Matlab

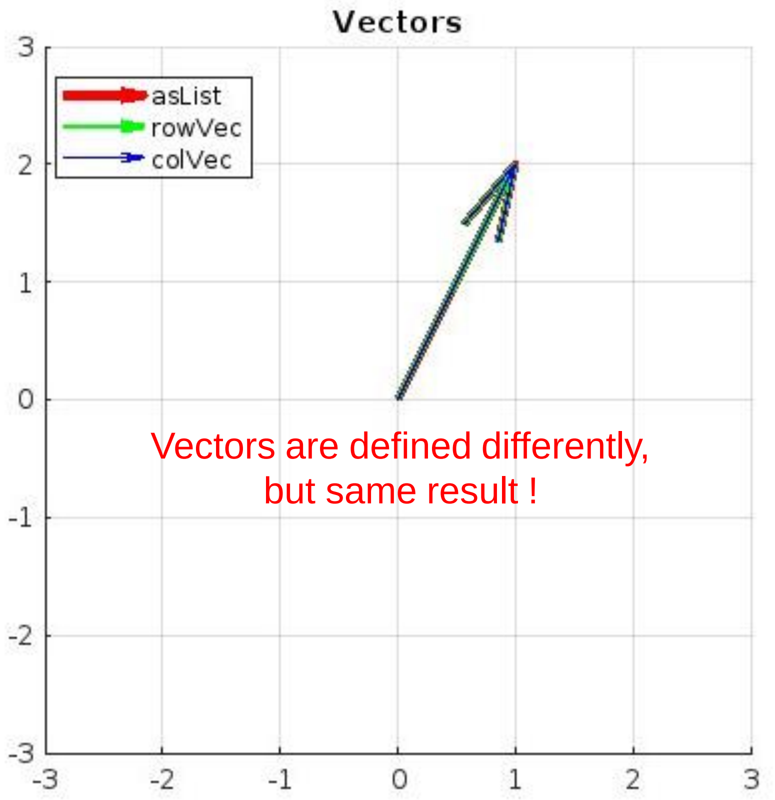


## Code Exercise (02_01)

- Three methods for creating vectors.

```
% Creating a vector as a MATLAB list
asList = [1, 2];

% Creating a row vector
rowVec = [1, 2]; % row

% Creating a column vector
colVec = [1; 2;]; % column

% Plotting the vectors using quiver
figure;
hold on;

% To prevent overlap, there is a 0.1 offset in the starting points of the vectors.
quiver(0, 0, asList(1), asList(2), 'r', 'LineWidth', 3, 'AutoScale', 'off', 'MaxHeadSize', 1);
quiver(0, 0, rowVec(1), rowVec(2), 'g', 'LineWidth', 2, 'AutoScale', 'off', 'MaxHeadSize', 1);
quiver(0, 0, colVec(1), colVec(2), 'b', 'LineWidth', 1, 'AutoScale', 'off', 'MaxHeadSize', 1);

% Set axes properties
axis equal;
xlim([-3, 3]);
ylim([-3, 3]);

% Show grid
grid on;

% Title for the visualization
title('Vectors');

% Legend for vectors
legend('asList', 'rowVec', 'colVec');
```

Source code




Source code result

# Equivalence of Vectors



- **If and only if their corresponding entries are equal.**
  - If the corresponding components of vectors $\boldsymbol{u} = (u_1, u_2, \dots, u_n)$ and $\boldsymbol{v} = (v_1, v_2, \dots, v_n)$ are equal, that is, $u_i = v_i$ for all i, then the two vectors are said to be $\boxed{\phantom{equivalent}}$ denoted by $\boldsymbol{u} = \boldsymbol{v}$.
  - $\boldsymbol{u} = \boldsymbol{v}$ iff $u_1 = v_1$ and $u_2 = v_2$ in vectors in $\mathbb{R}^2$.

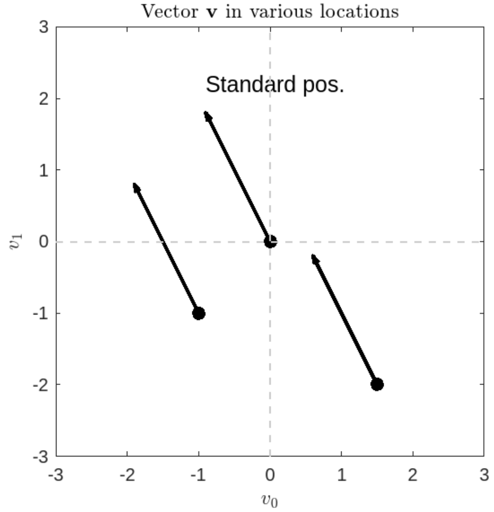
$$\boldsymbol{u} = (4,5,7,2), \boldsymbol{v} = (4,5,7,2), \boldsymbol{w} = (4,5,7,2,6)$$
$$\boldsymbol{u} = \boldsymbol{v}\,, \boldsymbol{u} \neq \boldsymbol{w}$$

Concept of equivalence between vectors

# Mathematical Interpretation of Vectors

- **Algebraic interpretation of vectors**
  - A list of numbers arranged in order. ➔ useful in data science
- **Geometric interpretation of vectors**
  - A line with a specific [ ] and [ ] (or angle: measured counterclockwise from the positive x-axis). ➔ useful in physics and engineering
  - A vector representing a physical quantity with both direction and magnitude.
  - Displacement, velocity, acceleration, force, electric field, etc.

- **Standard position in Geometric interpretation.**
  - Vectors and coordinates are different!
  - All arrows represent different [ ] but the same [ ]
  - If the vector equals the coordinate, it is a standard position.
    - A vector at the standard position has its **tail** at the **origin** and its **head points** to the **geometric coordinates**.




Various same vector $\boldsymbol{v}$

# Code Exercise of Generating Different Reference Vectors using Matlab

## Code Exercise (02_02)

- Generate vectors with different reference points.

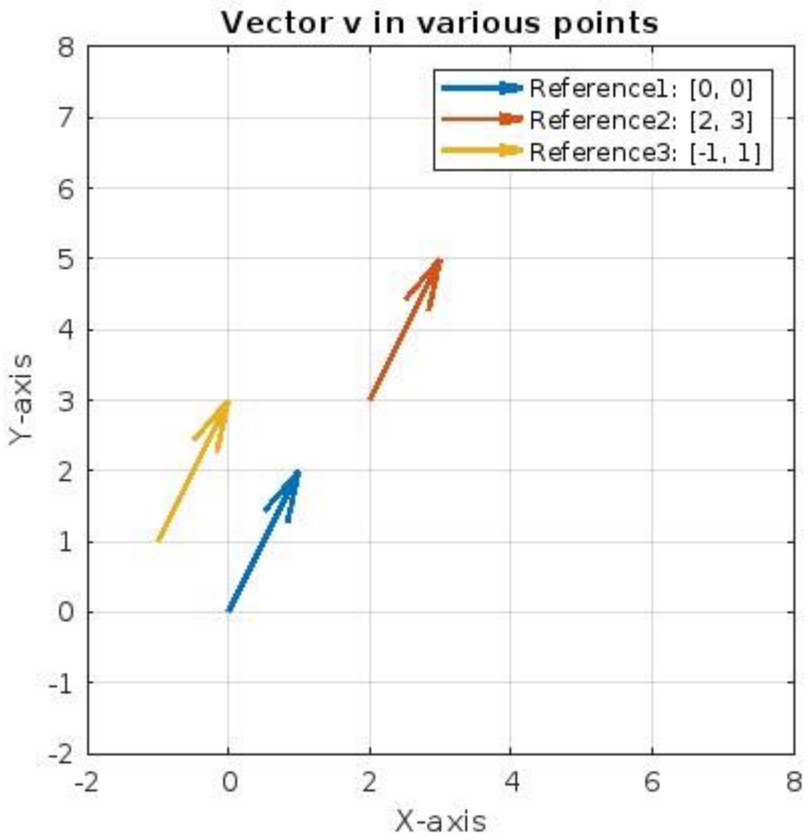
```
% Define the vector
v = [1, 2];

% Define three different reference points
reference_points = [0, 0; 2, 3; -1, 1];

% Create a figure
figure;

% Plot the vector with each reference point
for i = 1:size(reference_points, 1)
    quiver(reference_points(i, 1), reference_points(i, 2), v(1),
v(2),'LineWidth', 2, 'AutoScale', 'off', 'MaxHeadSize', 2);
    hold on;
end

% Set axes properties
axis equal;
xlim([-2, 8]);
ylim([-2, 8]);

% Show grid
grid on;

% Title for the visualization
title('Vector v in various points');

% Axes labels
xlabel('X-axis');
ylabel('Y-axis');

% Legend for vectors with different reference points
legend('Reference1: [0, 0]', 'Reference2: [2, 3]', 'Reference3: [-1,
1]');
```

Source code

# Visualization Result of Generating Vector using Matlab

## Code Exercise

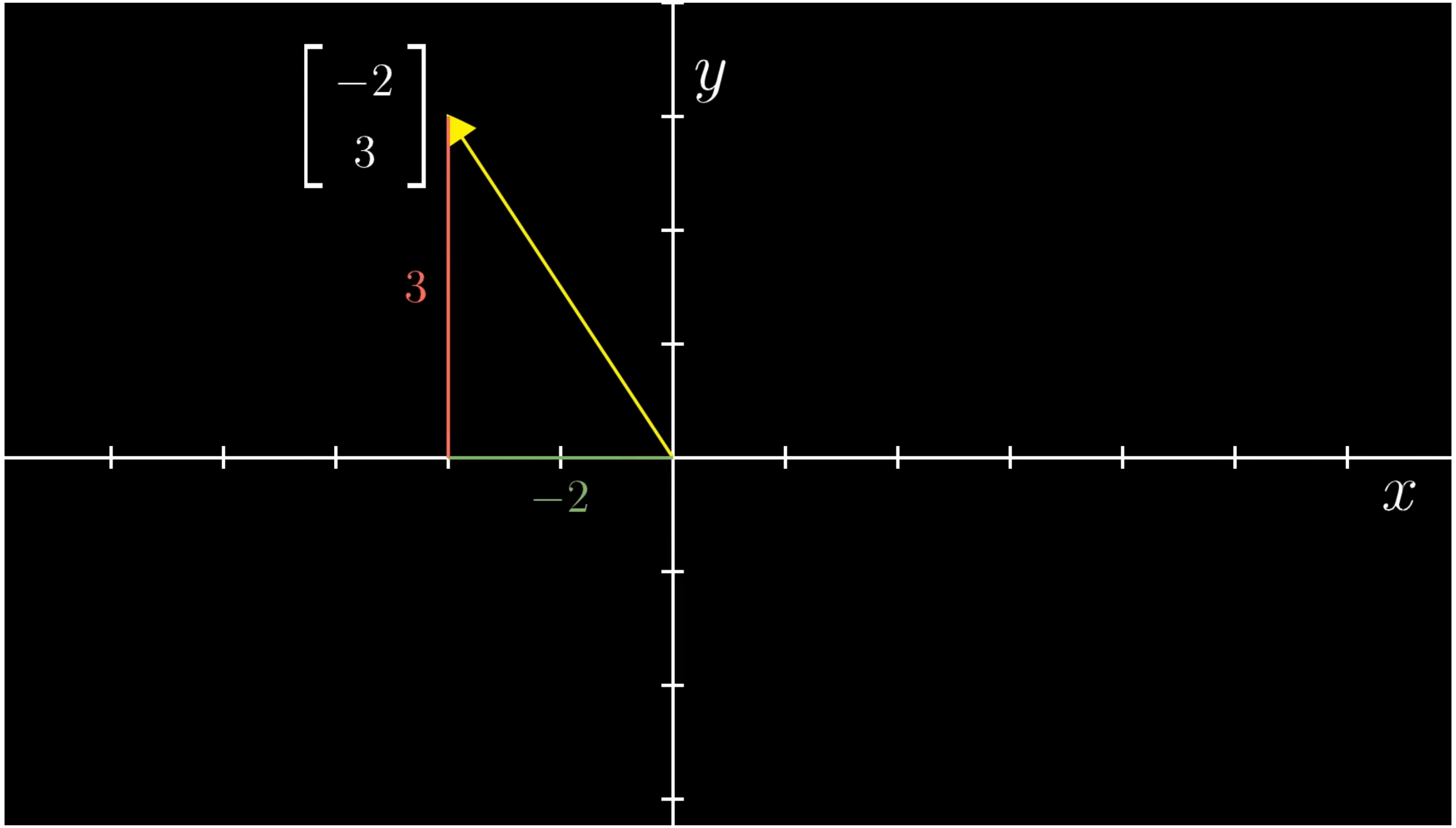
- Visualizing vectors with different reference points.




Source code result

# Geometric representation of vector

- Coordinate system (0:15 ~ 4:35)
- https://youtu.be/fNk_zzaMoSs?si=HvUOkaNK1-_BCLWL&t=15




Visualization of vector

# Vector operations

# Vector-Vector Addition and Subtraction

## Addition and subtraction of two vectors

- Vector addition, subtraction is only possible between vectors of the

$$\begin{bmatrix}4\\5\\6\end{bmatrix}+\begin{bmatrix}10\\20\\30\end{bmatrix}=\begin{bmatrix}14\\25\\36\end{bmatrix}$$

Addition between two vector

$$\begin{bmatrix}4\\5\\6\end{bmatrix}-\begin{bmatrix}10\\20\\30\end{bmatrix}=\begin{bmatrix}-6\\-15\\-24\end{bmatrix}$$

Subtraction between two vector

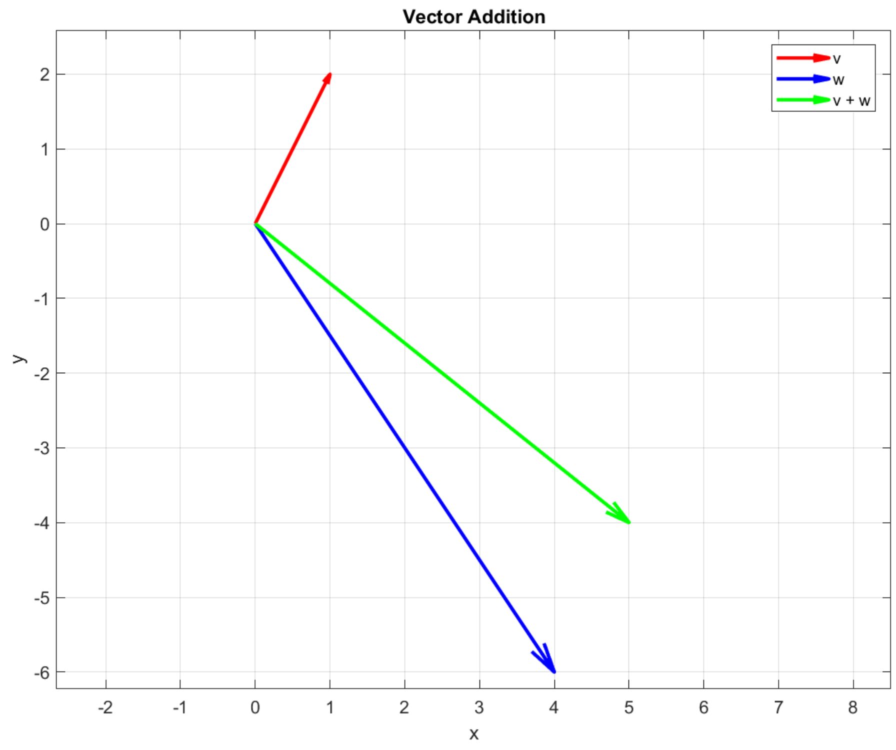
# Code Exercise of Vector Addition and Subtraction using Matlab

## Code Exercise (02_03)

- Addition between two vector.

```
%% Adding vectors

% Using 2D vectors here instead of 3D vectors in the book to
facilitate visualization
v = [1, 2];
w = [4, -6];
vPlusW = v + w;

% print out all three vectors
disp('v:');
disp(v);
disp('w:');
disp(w);
disp('vPlusW:');
disp(vPlusW);

% Plot vectors
quiver(0, 0, v(1), v(2), 0, 'r', 'LineWidth', 2);
hold on;
quiver(0, 0, w(1), w(2), 0, 'b', 'LineWidth', 2);
quiver(0, 0, vPlusW(1), vPlusW(2), 0, 'g', 'LineWidth', 2);
hold off;
axis equal;
xlabel('x');
ylabel('y');
title('Vector Addition');
legend('v', 'w', 'v + w');
grid on;
```

Source code




Source code result

# Vector Addition and Subtraction using Broadcasting

- **Addition and subtraction of two vectors using Broadcasting**
  - Broadcasting: Mechanism that automatically aligns the sizes of arrays when performing element-wise operations.
  - In MATLAB, broadcasting is possible when the dimensions of two vectors differ.

$$\begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} + \begin{bmatrix} 10 & 20 & 30 \end{bmatrix} = \begin{bmatrix} ? \end{bmatrix}$$

Is it possible?

# Code Exercise of Broadcasting using Matlab

## Code Exercise (02_04)

- Broadcasting – see diagonal element.

```
% column vector and row vector
column_vector = [1; 2; 3];
row_vector = [4 5 6];

% Using 2D vectors here instead of 3D vectors in the book to
facilitate visualization
sum_result = column_vector + row_vector;
difference_result = column_vector - row_vector;

% print out all three vectors
disp('addition:');
disp(sum_result);
disp('subtraction:');
disp(difference_result);
```

Source code

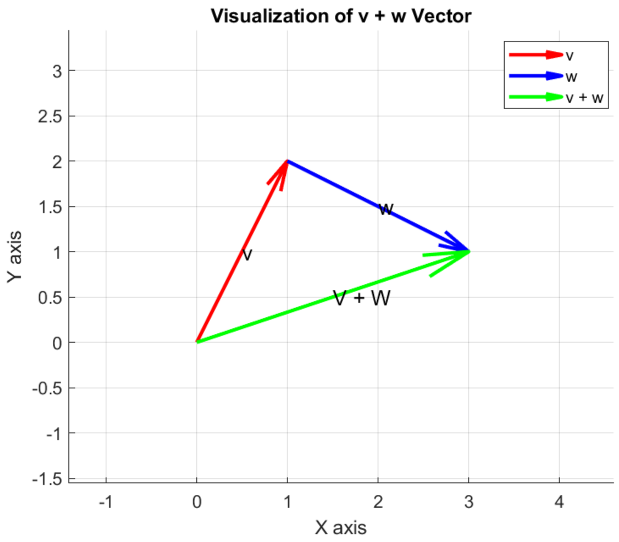
# Geometric Structure of Vector Addition and Subtraction

## Vector addition

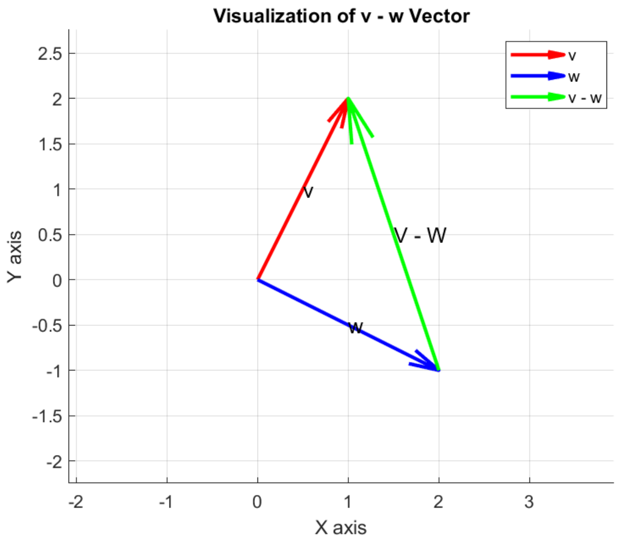
- Connecting the tail of one vector to the head of another vector.

## Vector subtraction

- Positioning the tails of two vectors at the same coordinate.
- The resulting vector from subtraction is directed from the head of the second vector to the head of the first vector.




Addition between two vector




Subtraction between two vector

# Scalar-Vector Multiplication

## Scalar-vector multiplication

- Scalar: A quantity that is not associated with any vector or matrix, but represents
  - Scalars are typically denoted by Greek lowercase letters such as α or λ.
  - example : scalar-vector multiplication can be represented as λ**w**.
    - λ : Scala
    - **w** : Vector

$$\lambda = 4, \boldsymbol{w} = \begin{bmatrix} 9 \\ 4 \\ 1 \end{bmatrix}, \lambda\boldsymbol{w} = \begin{bmatrix} 36 \\ 16 \\ 4 \end{bmatrix}$$

scalar-vector multiplication

# Code Exercise of Scalar-Vector Multiplication using Matlab

## Code Exercise (02_05)

- multiplication between scalar-vector.

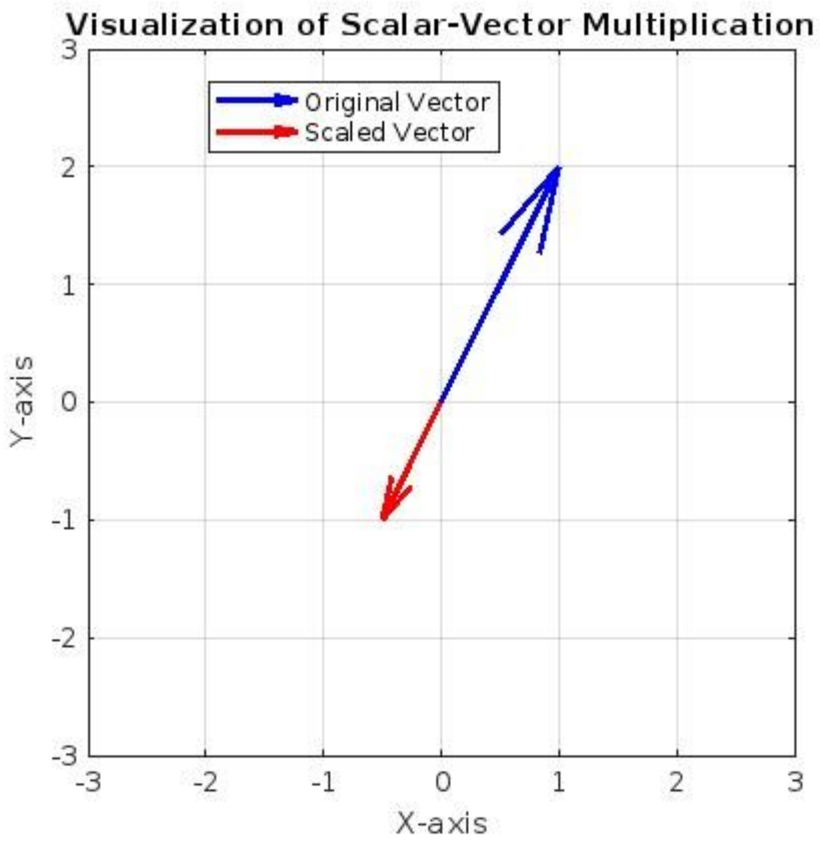
```matlab
% Define the vector
v = [1, 2];

% Define the scalar
s = -1/2;

% Compute the scaled vector
scaled_v = s * v;

% Create a figure
figure;

% Plot the original vector
quiver(0, 0, v(1), v(2), 'b', 'LineWidth', 3, 'AutoScale',
'off','MaxHeadSize', 2);
hold on;

% Plot the scaled vector
quiver(0, 0, scaled_v(1), scaled_v(2), 'r', 'LineWidth', 2,
'AutoScale', 'off', 'MaxHeadSize', 2);

% Set axes properties
axis equal;
xlim([-3, 3]);
ylim([-3, 3]);

% Show grid
grid on;

% Title for the visualization
title('Scalar-Vector Multiplication');

% Axes labels
xlabel('X-axis');
ylabel('Y-axis');

% Legend for vectors
legend('Original Vector', 'Scaled Vector');
```

Source code

# Visualization Result of Scalar-Vector Multiplication using Matlab

## Code Exercise

- multiplication between scalar-vector.




Source code result

# Scalar-Vector Addition and Subtraction

- **Scalar-vector addition**
  - In linear algebra : vectors and scalars are distinct mathematical objects and **cannot be combined**.
  - In Matlab: scalars to vectors can added or subtracted. How is it possible?

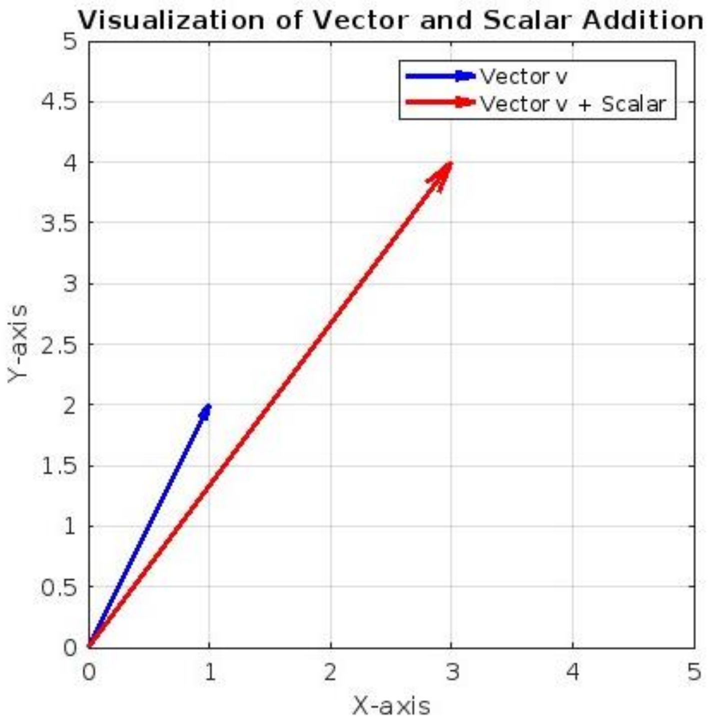
# Code Exercise of Scalar-Vector Addition using Matlab

## Code Exercise (02_06)

▶ Scalar - vector addition.

```
% Define vector
v = [1, 2];

% Define scalar
s = 2;

% Add scalar to vector
v_plus_s = v + s;

% Create figure
figure;

% Display vector v from the origin
quiver(0, 0, v(1), v(2), 'b', 'LineWidth', 2, 'AutoScale', 'off');
hold on;

% Display vector v + scalar from the origin
quiver(0, 0, v_plus_s(1), v_plus_s(2), 'r', 'LineWidth', 2, 'AutoScale', 'off');

% Set axes
axis equal;
xlim([0, 5]);
ylim([0, 5]);

% Show grid
grid on;

% Title for visualization of vector and scalar addition
title('Visualization of Vector and Scalar Addition');

% Axes labels
xlabel('X-axis');
ylabel('Y-axis');

% Legend for vectors and scalar
legend('Vector v', 'Vector v + Scalar');
```

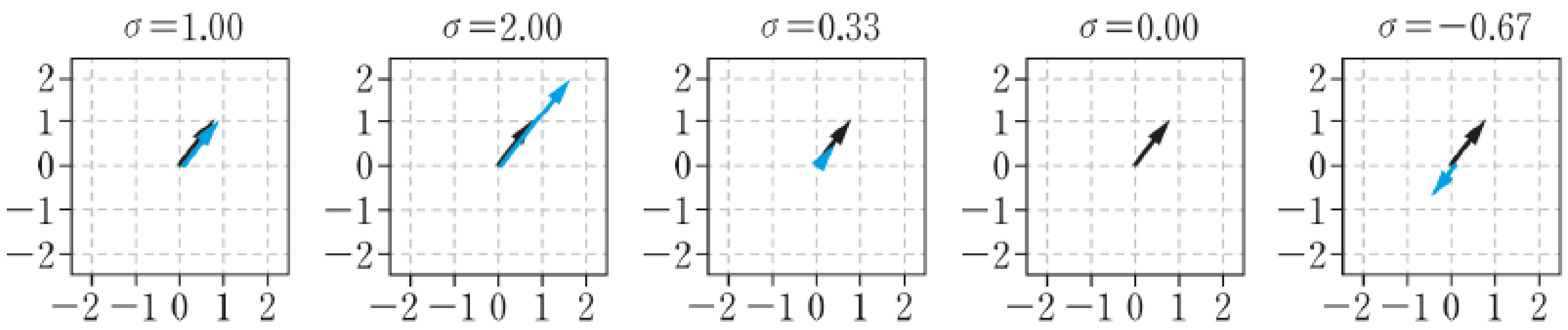
Source code




Source code result

## Geometric Understanding of Scalar-Vector Multiplication

- **Geometric** understanding in scalar-vector multiplication
  - Scalars **only scale the magnitude of vectors** without changing their ______

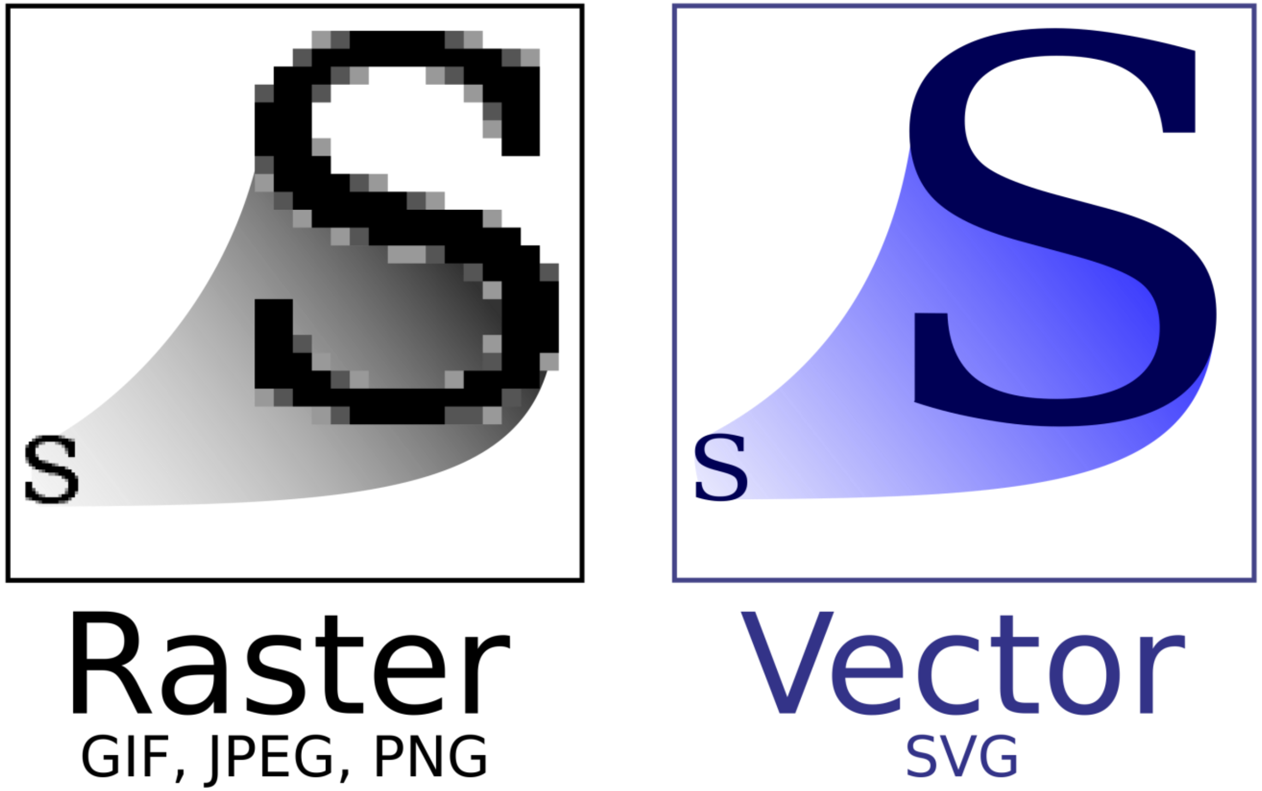



Various scalar-vector multiplication

  - In a diagram, when the scalar is negative, the vector direction is reversed (i.e., rotated 180 degrees).
  - The ______ vector still points along the **same infinite line**, so the **negative scalar hasn't changed its direction**.
- **Vector average**
  - Using **vector addition** and **scalar-vector multiplication**.
  - To find the average of N vectors, ______ them **all together and** ______ **by the scalar 1/N**.

# Example – Vector Graphics

- Vector graphics are a form of computer graphics in which visual images are created directly from geometric shapes defined on a Cartesian plane, such as points, lines, curves and polygons.



# Definition of Zero Vector

## Zero vector

- The zero vector (or ______) is a vector where all components are zero.
- Indicated using a boldfaced zero, **0**.

- In fact, using the zeros vector to solve a problem is often called the trivial solution and is excluded.
  - In linear algebra is full of statements like
    - Find a nonzeros vector that can solve…
    - Find a nontrivial solution to…

$$\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}0\\0\\0\end{bmatrix}, \begin{bmatrix}0 & 0 & 0 & 0 & 0\end{bmatrix}, \begin{pmatrix}0 & 0 & 0 & 0\end{pmatrix}$$

Example of Zero vector

# Properties of Vector Operations

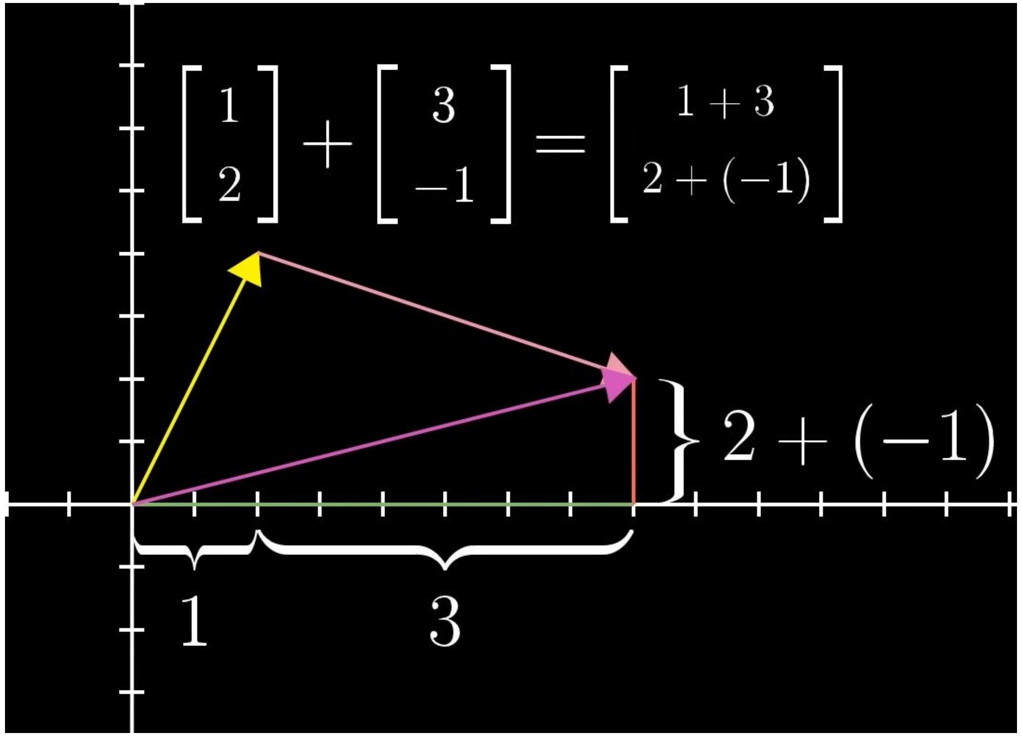
## Properties of vector operations

- Where $\alpha$,$\beta$ are scalar, $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$ are n-dimensional real vectors, 0 represents the zero vector.

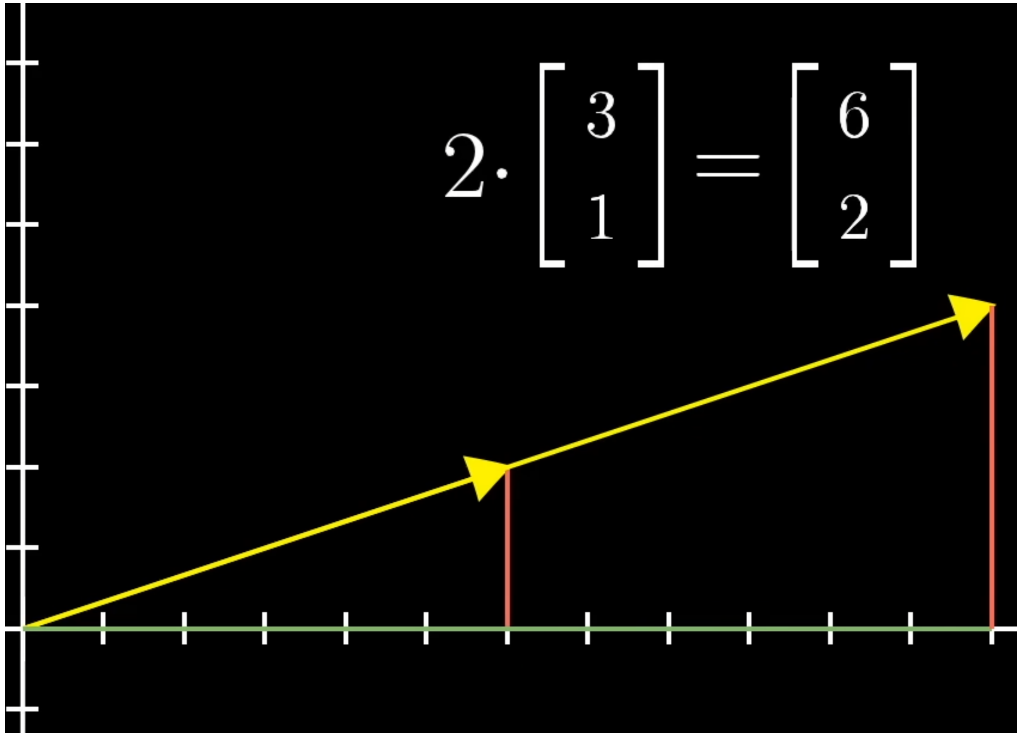
- $\boldsymbol{u} + \boldsymbol{v} = \boxed{\quad}$
- $\boldsymbol{u} + (\boldsymbol{v} + \boldsymbol{w}) = (\boldsymbol{v} + \boldsymbol{u}) + \boldsymbol{w}$
- $\boldsymbol{u} + \boldsymbol{0} = \boldsymbol{0} + \boldsymbol{u} = \boldsymbol{u}$
- $\boldsymbol{u} + (-\boldsymbol{u}) = (-\boldsymbol{u}) + \boldsymbol{u} = \boldsymbol{0}$
- $\alpha(\boldsymbol{u} + \boldsymbol{v}) = \alpha\boldsymbol{u} + \alpha\boldsymbol{v}$
- $(\alpha + \beta)\boldsymbol{u} = \boxed{\quad}$
- $\alpha(\beta\boldsymbol{u}) = \boxed{\quad}$
- $1\boldsymbol{u} = \boldsymbol{u}$

# Visual Materials

- **Geometric representation of vector operation**
  - Vector addition (4:36 ~ 6:53)
    - https://youtu.be/fNk_zzaMoSs?t=276&si=iIkRwYfl8Hl1Wyo3
  - Vector multiplication (6:53 ~ 8:07)
    - https://youtu.be/fNk_zzaMoSs?t=414&si=heZf3HVg9BpCFo4c




Visualization of vector addition operation




Visualization of vector multiplication operation

# Vector magnitude and unit vectors

# Vector Magnitude and Unit Vector

## Norm

- Function that calculates the $\boxed{\phantom{xxxxxxxxxx}}$
- Vector u's norm is presented as $\boxed{\phantom{xxx}}$ and norm satisfies the following properties.
  - $\boldsymbol{u}, \boldsymbol{v}$ is vector, and $\alpha$ is scala.

1. $\| \boldsymbol{u} \| \geq 0$
2. $\| \alpha \boldsymbol{u} \| = |\alpha| \, \| \boldsymbol{u} \|$
3. $\| \boldsymbol{u} + \boldsymbol{v} \| \leq \| \boldsymbol{u} \| + \| \boldsymbol{v} \|$
4. $\| \boldsymbol{u} \| = 0, only\ when\ \boldsymbol{u} = 0$

$$\|v\|_p = \left[ \sum_{k=1}^{N} |v_k|^p \right]^{1/p}$$

# Manhattan Norm (L1 norm)

- **For a vector $v = x1, x2, \ldots . xn$, the Manhattan norm is defined as follow.**

$$\|\boldsymbol{v}\|_1 = \sum_i^n |x_i| = |x_1| + |x_2| + \cdots + |x_n|$$

- **Manhattan norm is also called $\boxed{\phantom{xxxxxx}}$ and is used to define distance.**
- **Designed to express actual moving distance rather than simple straight-line distance.**

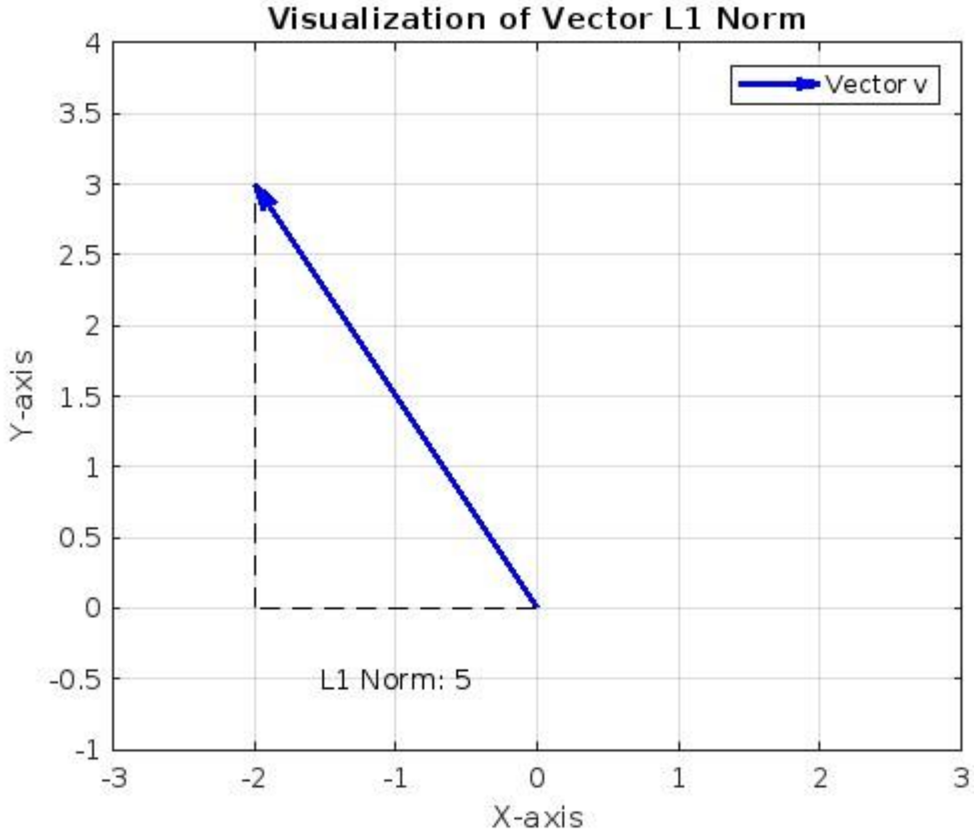
# Code Exercise of Manhattan Norm Norm using Matlab

## Code Exercise (02_07)

- L1 norm(Manhattan norm)

```
% Define vector
v = [-2, 3];

% Calculate L1 norm
l1_norm = norm(v, 1);

% Display vector from the origin
quiver(0, 0, v(1), v(2), 'b', 'LineWidth', 2, 'AutoScale', 'off');
hold on;

% Add lines representing movement along each axis to visualize Manhattan distance
plot([0, v(1)], [0, 0], '--k', 'LineWidth', 1); % Movement along x-axis
plot([v(1), v(1)], [0, v(2)], '--k', 'LineWidth', 1); % Movement along y-axis

% Display the value of L1 norm
text(v(1)/2, -0.5, ['L1 Norm: ', num2str(l1_norm)], 'HorizontalAlignment', 'center');

% Set axes properties
axis equal;
xlim([-3, 3]);
ylim([-1, 4]);

% Show grid
grid on;

% Title for visualization of vector L1 norm
title('Visualization of Vector L1 Norm');

% Axes labels
xlabel('X-axis');
ylabel('Y-axis');

% Legend for vectors and movement along axes
legend('Vector v');
```

Source code




Source code result

# Euclidean Norm (L2 norm)

- **For a vector $v = x1, x2, \ldots. xn$, the Euclidean norm is defined as follow.**

$$\|\boldsymbol{v}\|_2 = \sqrt{\sum_{i}^{n} x_i^2} = \sqrt{x_1^2 + x_2^2 + x_3^2 + x_4^2 + \cdot\cdot + x_i^2}$$

- **Euclidean norm is also called $\boxed{\phantom{xxxxxx}}$ and is used to define distance and magnitude.**
- **Regardless of dimension, $\boxed{\phantom{xxxxxx}}$ is obtained as the square root of the sum of the squares of the absolute values.**

- **When we refer to the norm of a vector, we usually mean the Euclidean norm.**

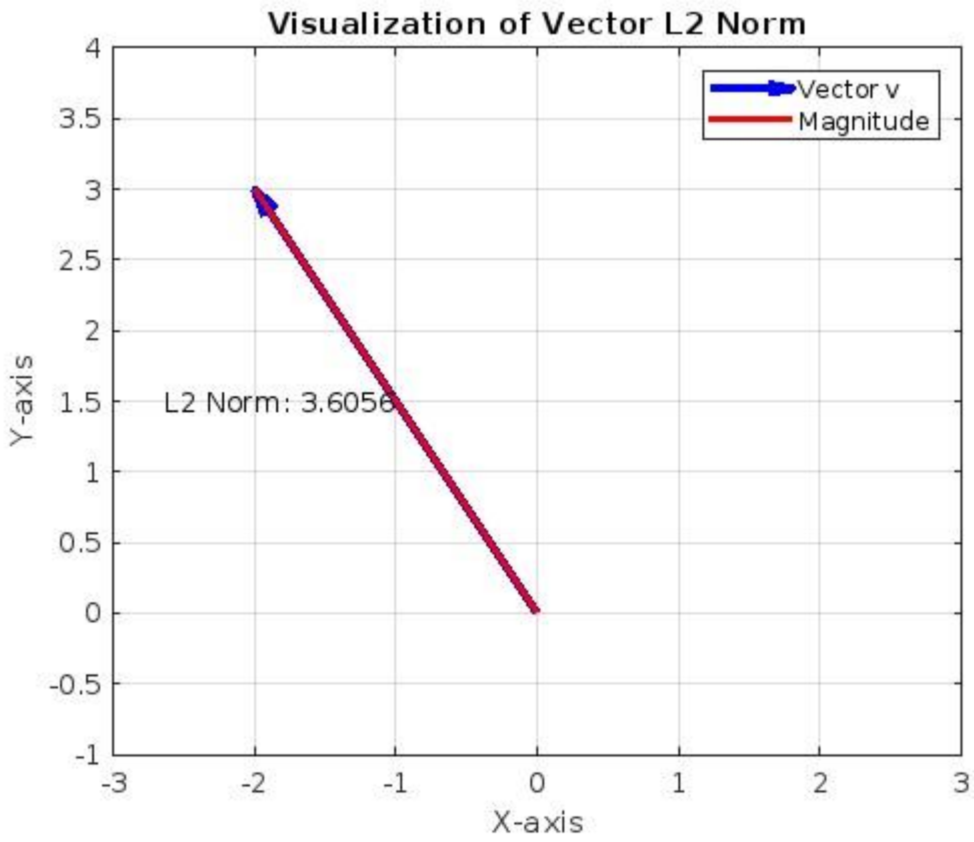
# Code Exercise of Euclidean Norm Norm using Matlab

## Code Exercise (02_08)

- L2 norm(Euclidean norm)

```
% Define vector
v = [-2, 3];

% Calculate L2 norm
l2_norm = norm(v, 2);

% Display vector from the origin
quiver(0, 0, v(1), v(2), 'b', 'LineWidth', 2, 'AutoScale', 'off');
hold on;

% Add line representing the vector to illustrate its magnitude
plot([0, v(1)], [0, v(2)], 'r', 'LineWidth', 2);

% Display the value of L2 norm
text(v(1)/2, v(2)/2, ['L2 Norm: ', num2str(l2_norm)], 'HorizontalAlignment', 'right');

% Set axes properties
axis equal;
xlim([-3, 3]);
ylim([-1, 4]);

% Show grid
grid on;

% Title for visualization of vector L2 norm
title('Visualization of Vector L2 Norm');

% Axes labels
xlabel('X-axis');
ylabel('Y-axis');

% Legend for vectors
legend('Vector v', 'Magnitude');
```

Source code




Source code result

# Meaning of Magnitude of Vector and Code Exercise

- **Magnitude of a vector (geometric length or norm):** the distance from tail to head of a vector
  - Calculate using the standard **Euclidean distance formula** (see equation below).
  - The magnitude of a vector is indicated by double vertical bars on either side ($\|\boldsymbol{v}\|$).
    - In some cases, the squared magnitude ($\|\boldsymbol{v}\|_2$) is used, in which case the square root term on the right-hand side is removed.

- **Code Exercise (02_09)**
  - Vector norm & length

```
%% Vector Norm
v = [-2, 3];

% Norm of vector
v_L1_norm = norm(v, 1);
v_L2_norm = norm(v, 2);

% Display norm of vector

disp(['Vector L1 norm: ', num2str(v_L1_norm)]);
disp(['Vector L2 norm: ', num2str(v_L2_norm)]);
```

Source code

$$\|\boldsymbol{v}\| = \sqrt{\sum_{i=1}^{n} \boldsymbol{v}_i^2}$$

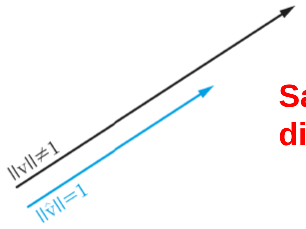
Euclidean distance formula

# Unit Vector

- **A vector with a geometric length of 1.**
  - Examples) Orthogonal matrices and rotation matrices, eigenvectors, singular vectors, etc.
  - The unit vector is defined as
- **How to create the associated unit vector?**
  - By scalar multiplication of the reciprocal of the vector norm.

$$\hat{\boldsymbol{v}} = \frac{1}{\|\boldsymbol{v}\|}\boldsymbol{v}$$

- **The general convention to denote a unit vector($\hat{v}$) in the same direction as the parent vector ($v$).**




Same angle with
different magnitude

# Vector dot product

# Definition of Vector Dot Product

- **The dot product (also known as the ______ product or ______ product) is one of the most important operations in the entirety of linear algebra.**
  - It forms the basis of many operations and algorithms such as convolution, correlation, Fourier transform, matrix multiplication, linear feature extraction, signal filtering, etc.
  - The ways to denote the dot product between two vectors include:
    - The general notation $\boldsymbol{a}^T\boldsymbol{b}$.
    - $\boldsymbol{a} \cdot \boldsymbol{b}$ or $< \boldsymbol{a}, \boldsymbol{b} >$
  - To calculate the dot product:
    - Multiply corresponding elements from the two vectors and then sum all the results.
    - The dot product is only defined between two vectors of the same ______

$$\delta = \sum_{i=1}^{n} a_i b_i$$

Dot product formula

# Calculation of Vector Dot Product

## Dot product is defined by following equation

- Let $\boldsymbol{u}$, and $\boldsymbol{v}$ vectors such that :

$$\boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \qquad \boldsymbol{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}$$

- Dot product of $\boldsymbol{u}$ and $\boldsymbol{v}$ is defined as $\boxed{\phantom{xx}}$, and represented as $< \boldsymbol{u}, \boldsymbol{v} >$ or $\boldsymbol{u} \cdot \boldsymbol{v}$.

$$\boldsymbol{u} \cdot \boldsymbol{v} = \boldsymbol{u}^T \boldsymbol{v} = \begin{bmatrix} u_1 & u_2 & \cdots & u_n \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} = u_1 v_1 + u_2 v_2 + \cdots + u_n v_n$$

$$\begin{aligned} [1\ 2\ 3\ 4] \cdot [5\ 6\ 7\ 8] &= 1 \times 5 + 2 \times 6 + 3 \times 7 + 4 \times 8 \\ &= 5 + 12 + 21 + 32 \\ &= 70 \end{aligned}$$

Example of dot product calculation

# Properties of Vector Dot Product

## Scala multiplication

- When positive scalar is multiplied by a vector, its dot product $\boxed{\phantom{xxxxxx}}$ by that factor.
  - $(\alpha\boldsymbol{u})^T\boldsymbol{v} = \alpha(\boldsymbol{u}^T\boldsymbol{v})$
  - If dot product of $\boldsymbol{v}$ and $\boldsymbol{w}$ is 70, and value of scala $s$ is 10, then the dot product of $s\boldsymbol{v}$ and $\boldsymbol{w}$ will be $\boxed{\phantom{xxx}}$
- If you try multiplying negative scalar the magnitude of the dot product remains the same, but the sign is opposite.
- Scala of value 0
  - If s = 0, then the dot product is also $\boxed{\phantom{x}}$

## The dot product is a measure of $\boxed{\phantom{xxxxxx}}$ or $\boxed{\phantom{xxxxxx}}$ between two vectors.

- Pearson correlation coefficient: the normalized dot product between two variables.

# Code Exercise of Vector Dot Product using Matlab

## Code Exercise (02_10)

- dot() function

```
%% Dot product
v = [0, 1, 2];
u = [13, 21, 34];

s = 10;

% scala multiplcate dot product
dot_product = dot(v, u);
scala_multiplicated = dot(s*v, u);

% show the result
disp('Dot Product:');
disp(dot_product);
disp('Scala multiplicated:');
disp(scala_multiplicated);
```

Source code

# Property and Code Exercise of Dot Product Distributive Law

## Distributive law of the dot product

- The dot product of the sum of vectors is equal to

$$a^T(b + c) = a^T b + a^T c$$

Distributive law of dot product

## Code Exercise (02_11)

- Distributive law of dot product.

```
%% The dot product is distributive

% some random vectors
v = [0, 1, 2];
w = [3, 5, 8];
u = [13, 21, 34];

% two ways to compute
res1 = dot(v, w + u);
res2 = dot(v, w) + dot(v, u);

% show that they are equivalent
disp('res1:');
disp(res1);
disp('res2:');
disp(res2);
```

The two results, res1 and res2, are the same (the answer is 110). This indicates that the distributive property of the dot product holds.

Source code

# Geometric Definition of Dot Product

## Geometric interpretation of dot product

- Multiplication the magnitudes of two vectors and increasing the size by the ______ of the angle between the two vectors.
- Eq 1. and Eq 2. are mathematically equivalent but expressed differently.

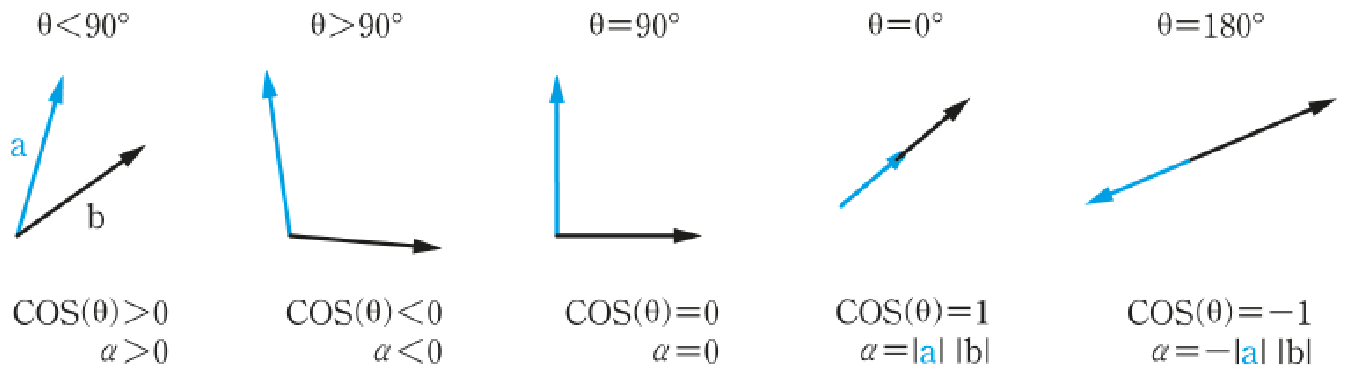
$$\delta = \sum_{i=1}^{n} a_i b_i$$

Eq 1. Dot product formula

$$\alpha = \cos\left(\theta_{\mathrm{v,w}}\right) \|\mathbf{v}\| \, \|\mathbf{w}\|$$

Eq 2. Geometric definition of vector dot product

- Five cases of dot product sign depending on the angle between two vectors.




Dot product sign of two vectors present geometric relationship between vectors

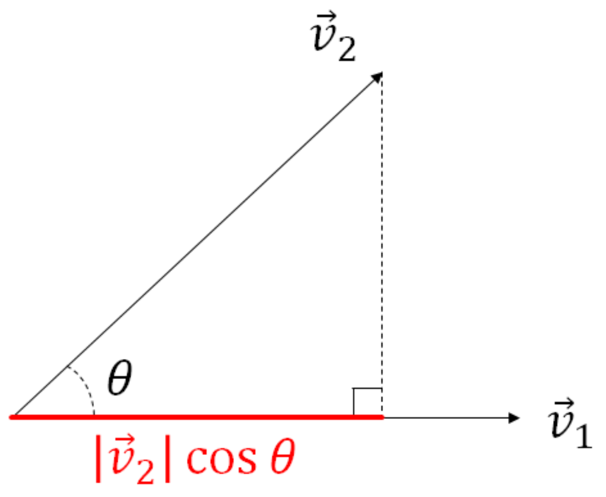
# Reference Materials of Vector Dot Product

- **Geometric meaning of vector dot product**
  - https://angeloyeo.github.io/2020/09/09/row_vector_and_inner_product.html#%ED%96%89%EB%B2%A1%ED%84%B0%EC%9D%98-%EC%8B%9C%EA%B0%81%ED%99%94

$$\vec{v}_1 \cdot \vec{v}_2 = [a \quad b]\begin{bmatrix} x \\ y \end{bmatrix} = ax + by$$

why?

=




$$\vec{v}_1 \cdot \vec{v}_2 = |\vec{v}_1||\vec{v}_2| \cos\theta$$

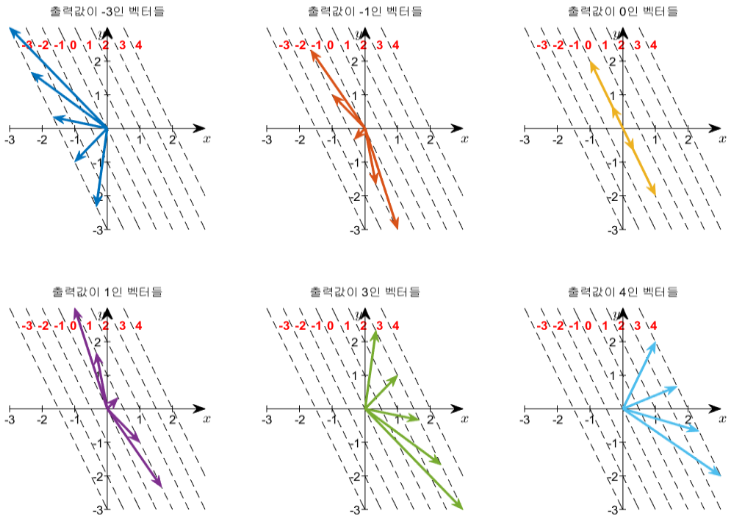
Geometrical proof of vector dot product

# Geometric Proofs of Vector Dot Product

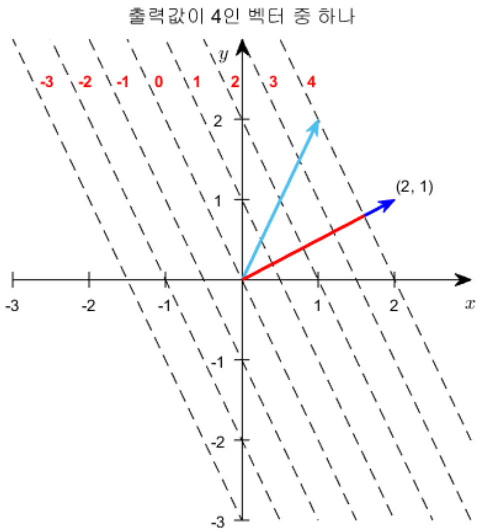
## Geometric meaning of vector dot product

1. Represent a map where locations of equal height are connected by a single line.
2. Consider the case where the output scalar value is 4.
3. Since the dashed lines corresponding to 2x+y=4 are all perpendicular to the row vector [2,1],
   - $4\times 2 = d \times \sqrt{20}$ , $d = \frac{4}{\sqrt{5}}$
   - Length of row vector [2,1] is $\sqrt{5}$, and multiplication of d and row vector is, $d \times \sqrt{5} = \frac{4}{\sqrt{5}} \times \sqrt{5} = 4$

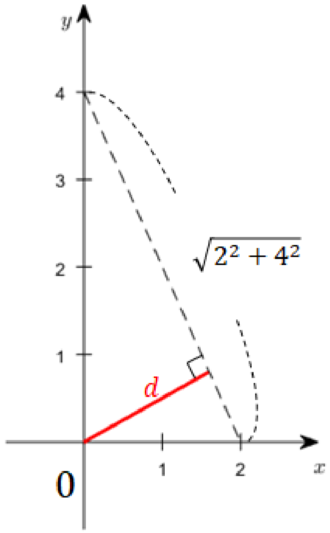
▶ So, product of the projection length of a column vector and the $\boxed{\quad\quad\quad\quad}$ = dot product value.




1) Visualization of different output of 2x+y (-3 ~ 4)



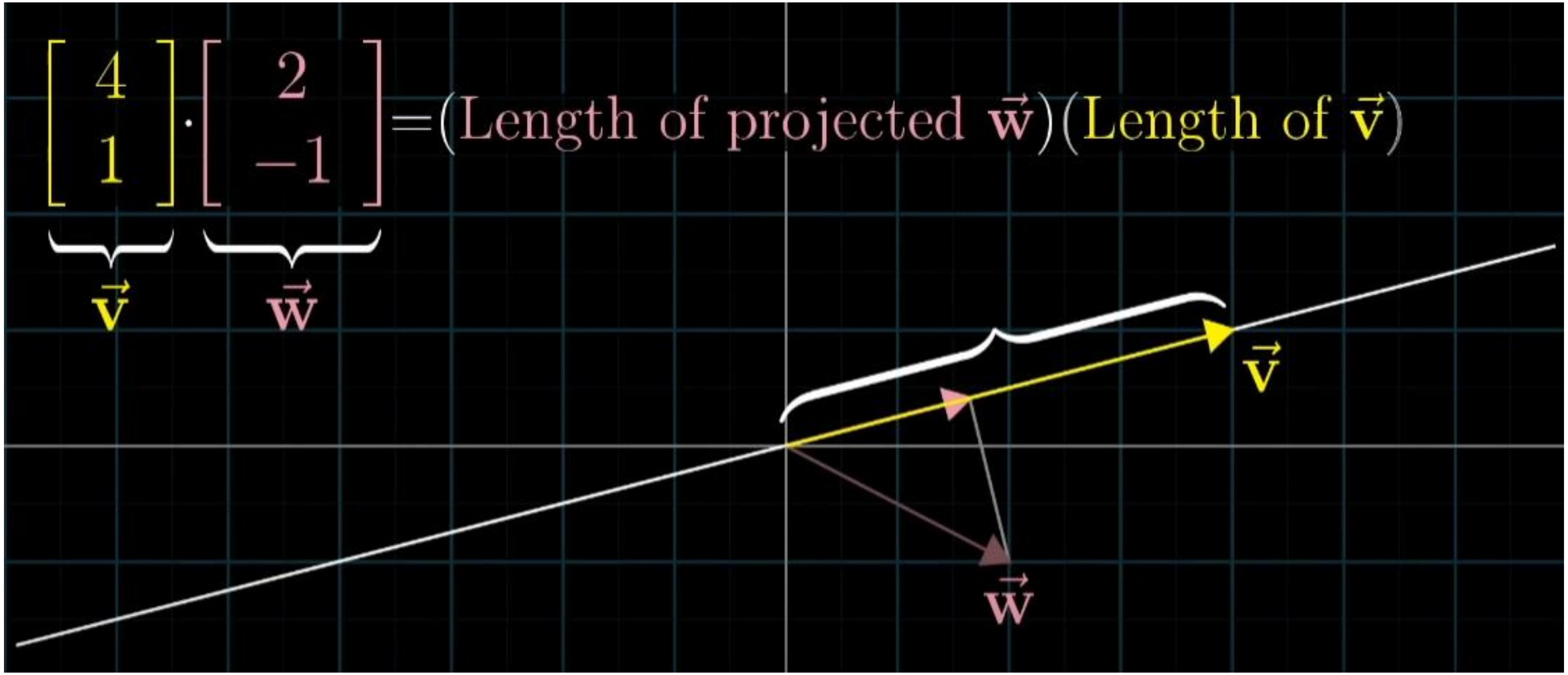

2) 2x + y = 4




3) Distance d

# Visual Materials (1)

- **Geometric representation of vector dot product with different angles**
  - Dot products, geometric interpretation (0:51 ~ 3:55)
  - https://youtu.be/LyGKycYT2v0?si=kSluHVZr478QXAkE&t=51

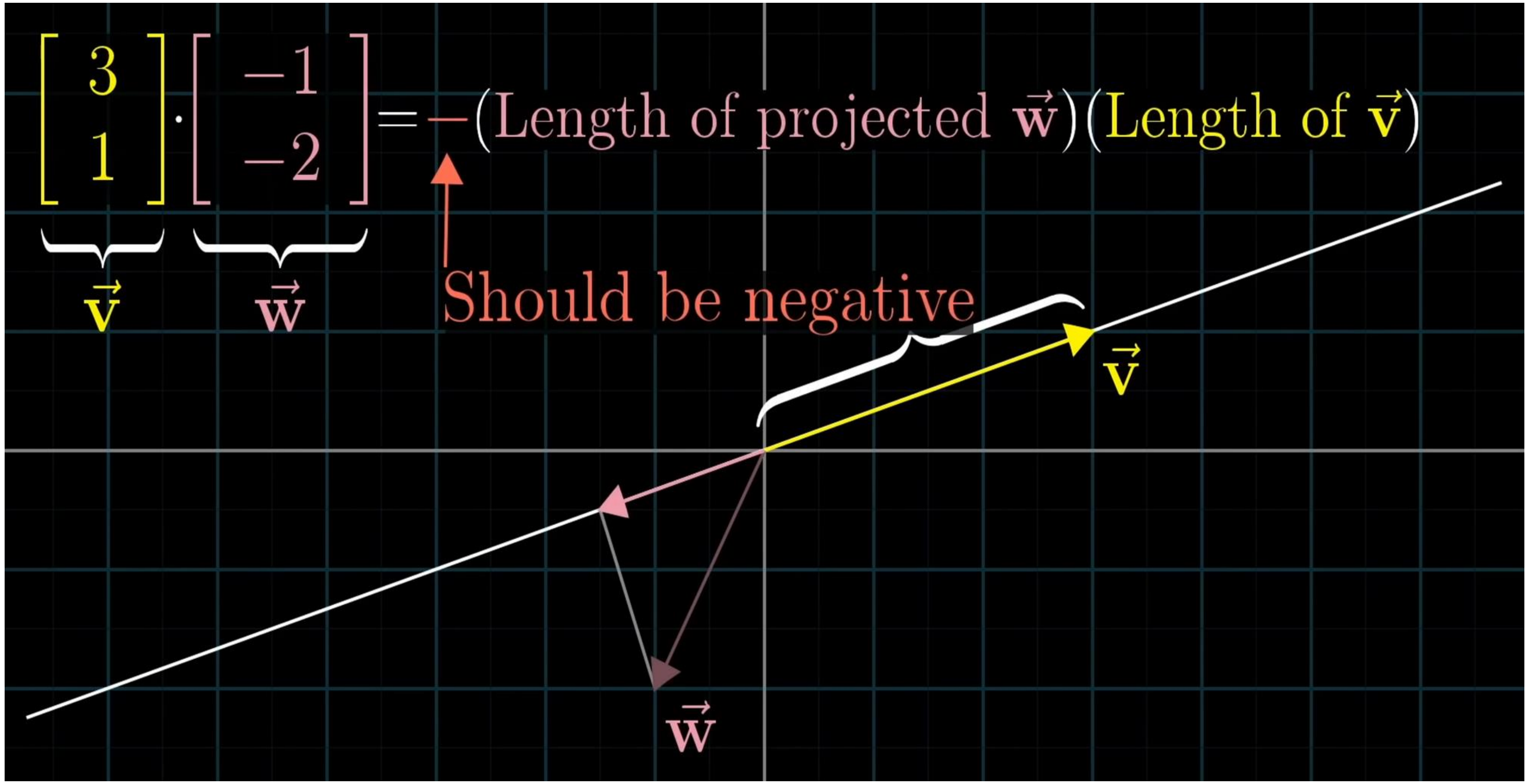



Visualization of dot production when angle between v,w is acute

# Visual Materials (2)



- **Geometric representation of vector dot product with different angles**
  - Dot products, geometric interpretation (0:51 ~ 3:55)
  - https://youtu.be/LyGKycYT2v0?si=kSluHVZr478QXAkE&t=51




Visualization of dot production when angle between v,w is acute

# Other vector multiplications

# Definition and Properties of Vector Cross Product

## Cross product

- The cross product($\boldsymbol{x} \times \boldsymbol{y}$) of vectors $\boldsymbol{x} = (x_1, x_2, x_3)$ and $\mathbf{y} = (y_1, y_2, y_3)$ in the $\mathbb{R}^3$ space is defined as follows.
- $\boldsymbol{x} \times \boldsymbol{y}$ is called ' $\boldsymbol{x}$ cross $\boldsymbol{y}$ '.

$$\boldsymbol{x} \times \boldsymbol{y} = (x_2y_3 - x_3y_2, x_3y_1 - x_1y_3, x_1y_2 - x_2y_1)$$

## Characteristics of cross product

- Characteristics of cross product of $\mathbb{R}^3$ space vector.
- The following properties hold for vector $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ in $\mathbb{R}^3$ space and scalar c.

$(1)\ \boldsymbol{x} \times \boldsymbol{y} = -\boldsymbol{y} \times \boldsymbol{x}$

$(2)\ \boldsymbol{x} \times (\boldsymbol{y} + \boldsymbol{z}) = (\boldsymbol{x} \times \boldsymbol{y}) + (\boldsymbol{x} \times \boldsymbol{z})$

$(3)\ (\boldsymbol{x} + \boldsymbol{y}) \times \boldsymbol{z} = (\boldsymbol{x} \times \boldsymbol{z}) + (\boldsymbol{y} \times \boldsymbol{z})$

$(4)\ c(\boldsymbol{x} \times \boldsymbol{y}) = (c\boldsymbol{x}) \times \boldsymbol{y} = \boldsymbol{x} \times (c\boldsymbol{y})$

$(5)\ \boldsymbol{x} \times 0 = 0 \times \boldsymbol{x} = 0$

$(6)\ \boldsymbol{x} \times \boldsymbol{x} = 0$

# Geometric Definition of Vector Cross Product

- **Geometric definition of vector cross product**
  - Normal vector of a plane and cross product.
    - Normal vector of a plane can be calculated through the cross product of the vectors corresponding to line segments forming the plane.

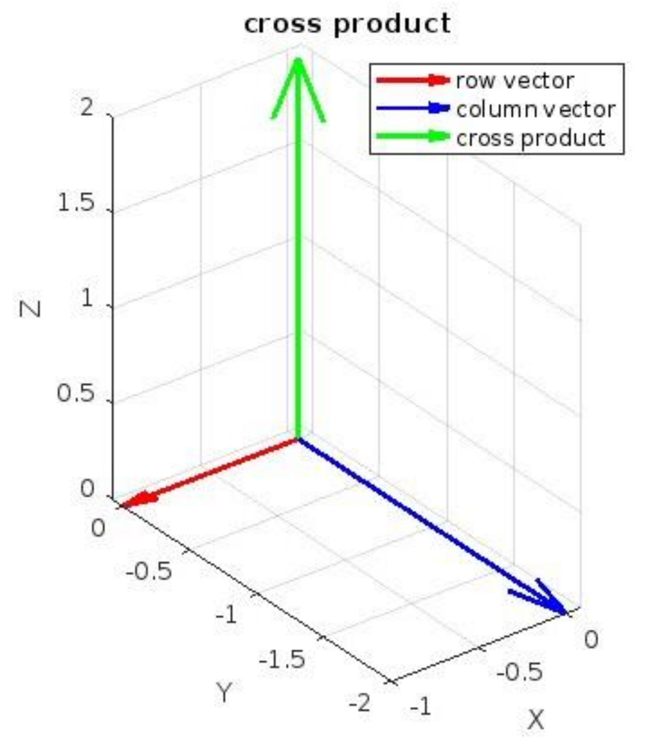
# Code Exercise of Vector Cross Product

## Code Exercise (02_12)

- Operation cross product between two vectors, one along the column direction and the other along the row direction.

```
% two vectors
row_vector = [-1 0 0];
column_vector = [0; -2; 0];

% cross product
cross_product = cross(row_vector, column_vector);

% result
disp('Cross Product:');
disp(cross_product);

% visualization
figure;
quiver3(0, 0, 0, row_vector(1), row_vector(2), row_vector(3), 'r', 'LineWidth', 2, 'AutoScale', 'off', 
'MaxHeadSize', 0.5);
hold on;
quiver3(0, 0, 0, column_vector(1), column_vector(2), column_vector(3), 'b', 'LineWidth', 2, 'AutoScale', 'off', 
'MaxHeadSize', 0.5);
quiver3(0, 0, 0, cross_product(1), cross_product(2), cross_product(3), 'g', 'LineWidth', 2, 'AutoScale', 'off', 
'MaxHeadSize', 0.5);
legend('row vector', 'column vector', 'cross product');
xlabel('X');
ylabel('Y');
zlabel('Z');
title('cross product');
axis equal;
grid on;
```

Source code

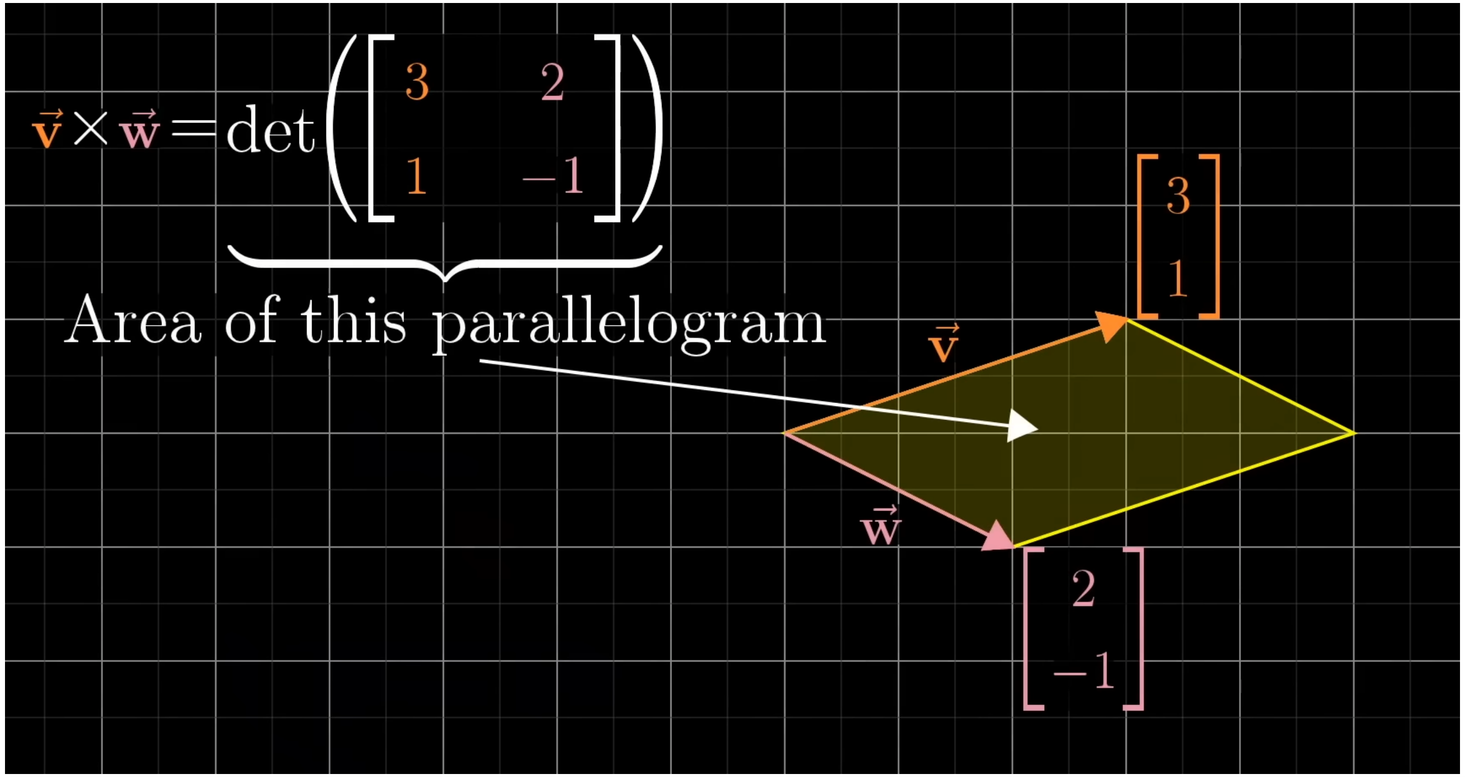



Source code result

# Visual materials

## Geometric representation of vector cross product

- Cross product (0:40 ~)
- https://youtu.be/eu6i7WJeinw?si=POJURAxWpOe_oQNa&t=40




Visualization of vector cross product in 2-dimensions

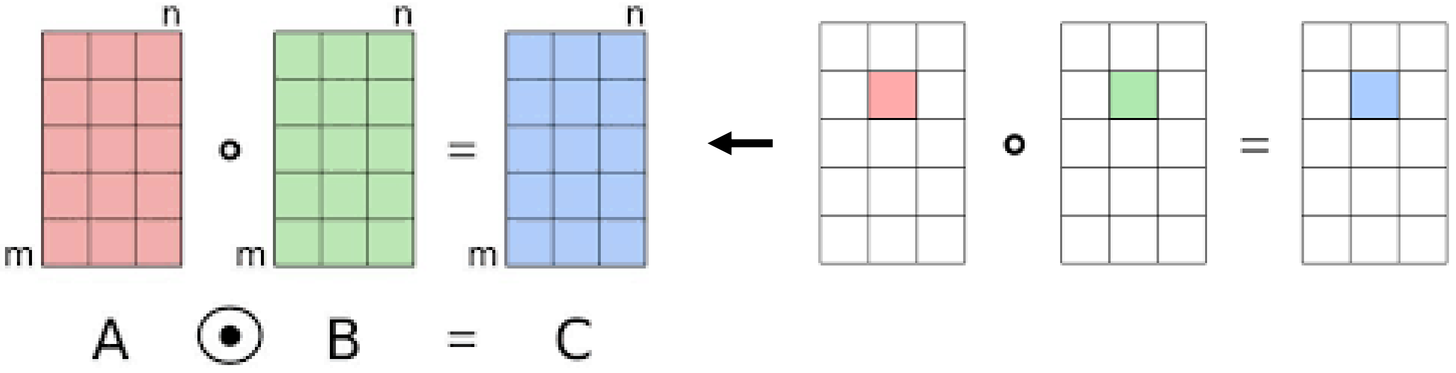
# Definition of Hadamard Product

## Hadamard product

- Implementation of Hadamard product
  - Operation that multiplies corresponding elements of two vectors of the same size.
    - The result of multiplication is vector of $\boxed{\qquad\qquad}$ with two vectors.
  - The symbol used to denote the Hadamard product is $\odot$.

$$\begin{bmatrix}5\\4\\8\\2\end{bmatrix} \odot \begin{bmatrix}1\\0\\.5\\-1\end{bmatrix} = \begin{bmatrix}5\\0\\4\\-2\end{bmatrix}$$

Representation of the Hadamard product (Vector)




Representation of the Hadamard product (Matrix)

# Code Exercise of Hadamard Product using Matlab

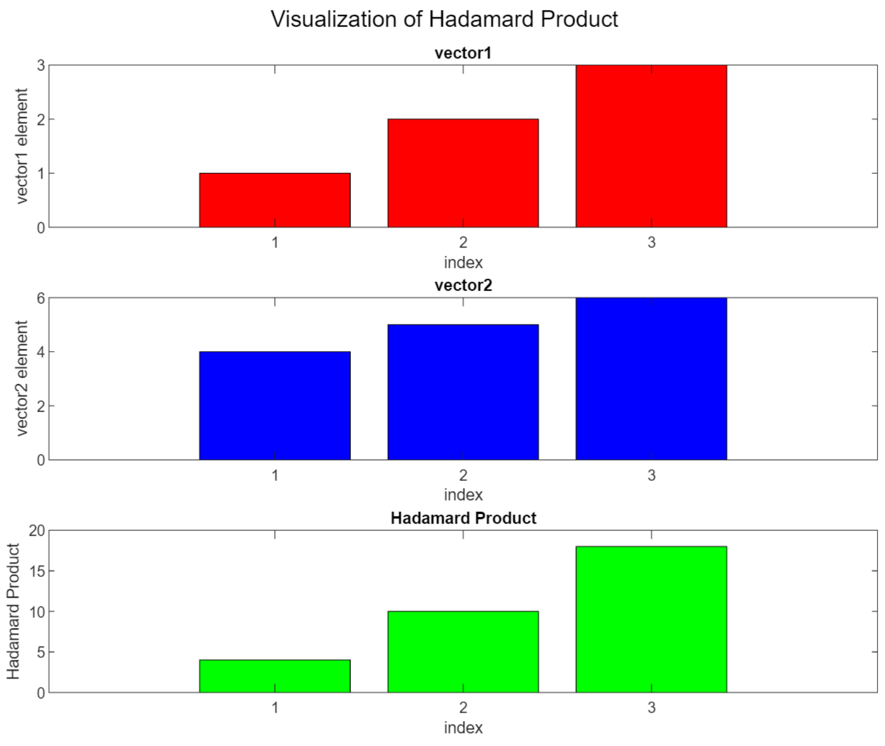


## Code Exercise (02_13)

- Multiplication between two vectors or matrices.

```
% two vectors
vector1 = [1 2 3];
vector2 = [4 5 6];

% Hadamard product - operator: .*
hadamard_product = vector1 .* vector2;

% plot
subplot(3, 1, 1);
bar(vector1, 'r');
xlabel('index');
ylabel('vector1 element');
title('vector1');

subplot(3, 1, 2);
bar(vector2, 'b');
xlabel('index');
ylabel('vector2 element');
title('vector2');

subplot(3, 1, 3);
bar(hadamard_product, 'g');
xlabel('index');
ylabel('Hardamard Product');
title('Hardamard Product');

sgtitle('visualization of Hardamard Product');
```

Source code




Source code result

# Orthogonal vector decomposition

# Definition of Orthogonality and Decomposition

## Concept of orthogonality

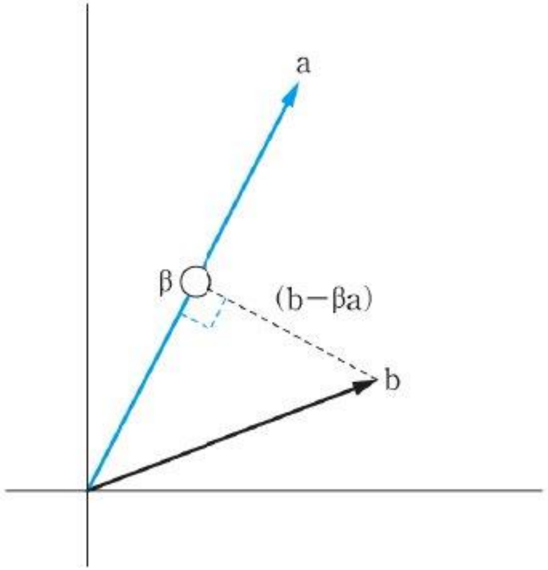
- In mathematics, orthogonality is the generalization of the geometric notion of **perpendicularity**.
- If dot product of two vector is $\boxed{\phantom{0}}$, they are Orthogonal.

## Concept of decomposition

- Scalar decomposition
  - The number 42.01 = 42 + 0.01
  - Prime factorization : decompose the number 42 into the product of the prime number 2, 3 and 7.
- Vector decomposition
  - To decompose a single vector into two vectors, one orthogonal to the reference vector and the other parallel to the reference vector.
    - The orthogonal vector decomposition has direct relevance to statistics in the Gram-Schmidt process and QR decomposition.

# Example of Vector Decomposition

- **Two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ exist in the standard position.**
- **Search the nearest point from $\boldsymbol{a}$ to the head of $\boldsymbol{b}$.**
  - It can be expressed as an optimization problem, where vector $\boldsymbol{b}$ is projected onto vector a such that the projection distance is $\boxed{\phantom{xxxxxx}}$
  - The point is β$\boldsymbol{a}$ that $\boxed{\phantom{xxxx}}$ the magnitude of $\boldsymbol{a}$.
  - Find Scalar β.




Vector decomposition visualization

# Definition of Orthogonal Projection

## Orthogonal projection

- It can be inferred that $\boldsymbol{b} - \beta\boldsymbol{a}$ is orthogonal to $\beta\boldsymbol{a}$.
  - Hence, these vectors are vertical. Therefore, dot product between two vectors should be

$$\boldsymbol{a}^T(\boldsymbol{b} - \beta\boldsymbol{a}) = 0$$

  - Finding $\beta$.

$$\boldsymbol{a}^T\boldsymbol{b} - \beta\boldsymbol{a}^T\boldsymbol{a} = 0$$

$$\beta\boldsymbol{a}^T\boldsymbol{a} = \boldsymbol{a}^T\boldsymbol{b}$$

$$\beta = \frac{\boldsymbol{a}^T\boldsymbol{b}}{\boldsymbol{a}^T\boldsymbol{a}}$$

Orthogonal projection

# Decompose Target Vector and Terminology

## 'Target vector' and 'Reference vector'

- The goal is to decompose the target vector into two different vectors.
  - Sum of the two vector is the target vector.
  - One **orthogonal** to the reference vector but the other **parallel** to the reference vector.

## Terminology clarification

- Target vector is $\boldsymbol{t}$, reference vector is $\boldsymbol{r}$.
- $\boldsymbol{t}_{\perp \boldsymbol{r}}$ is $\boxed{\phantom{xxxxxxxxxxxx}}$ created from target vector, $\boldsymbol{t}_{\parallel \boldsymbol{r}}$ is $\boxed{\phantom{xxxxxxxxxxxx}}$ created from target vector.

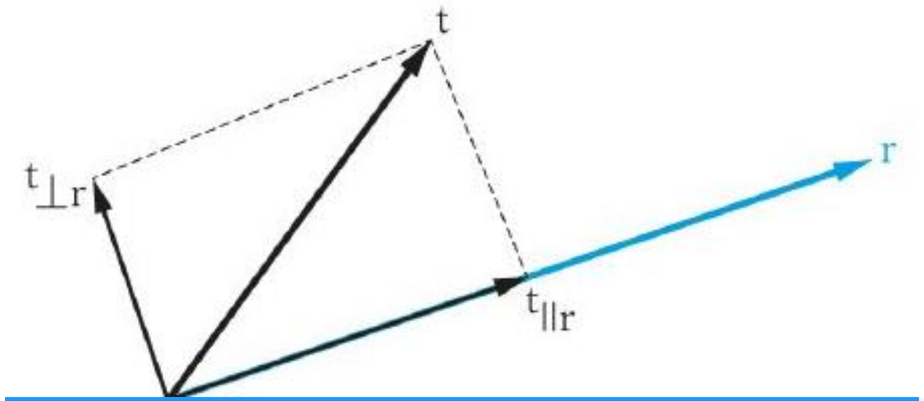



Figure of orthogonal vector decomposition

# Parallel Component Generated from Target Vector

## Parallel component

- Vector that resizing the size of r is [ ] to r.
- In Eq 1., only scalar β is calculated. Here, the resized vector β is calculated.
- [ ] of the two vector components is the target vector.

$$\beta = \frac{\boldsymbol{a}^T \boldsymbol{b}}{\boldsymbol{a}^T \boldsymbol{a}}$$

Eq 1. Orthogonal projection

$$\boldsymbol{t} = \boldsymbol{t}_{\perp \boldsymbol{r}} + \boldsymbol{t}_{\parallel \boldsymbol{r}}$$
$$\boldsymbol{t}_{\perp \boldsymbol{r}} = \boldsymbol{t} - \boldsymbol{t}_{\parallel \boldsymbol{r}}$$

Eq 2. Parallel component of target vector

# Vertical Component Generated from Target Vector

## Vertical component

- Is vertical component really orthogonal to the reference vector?

- Calculate if the dot product between [ ] and the [ ] is 0.
  - Prove it!

$$(\boldsymbol{t}_{\perp \boldsymbol{r}})^T \boldsymbol{r} = \boldsymbol{0}$$

$$\left(\boldsymbol{t} - \boldsymbol{r}\frac{\boldsymbol{t}^T\boldsymbol{r}}{\boldsymbol{r}^T\boldsymbol{r}}\right)^T \boldsymbol{r} = \boldsymbol{0}$$

**dot product of perpendicular component and reference vector**

# Summary

# Summary

- **Vector is a list of numbers arranged in a ______ or ______**
  - The number of elements in a vector is called its ______, and vector can be represented as a single line in a geometric space with the same number of axes as its dimension.

- **Vector arithmetic operations such as addition, minus and Hadamard product are calculated ______.**

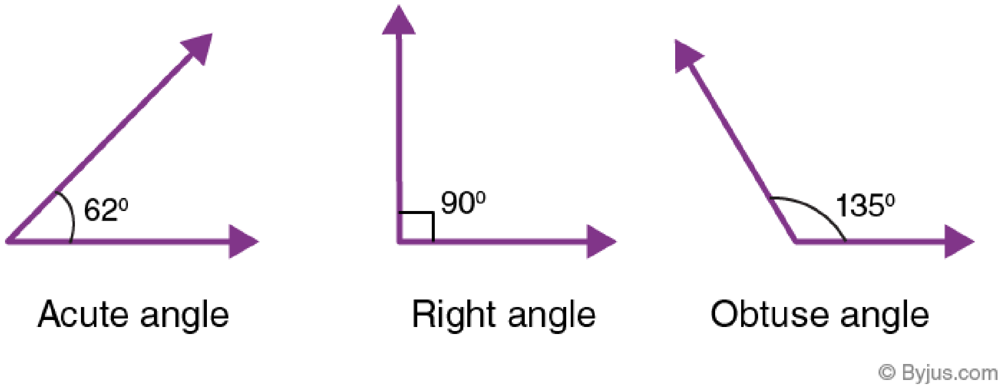
- **The dot product is calculated by multiplying corresponding elements of two vectors of the same ______ and summing them up, resulting in a single number encoding the relationship between the two vectors.**

# Summary

- If the two vectors [          ], the result of dot product is 0 and that means geometrically that the vectors meet at [          ]

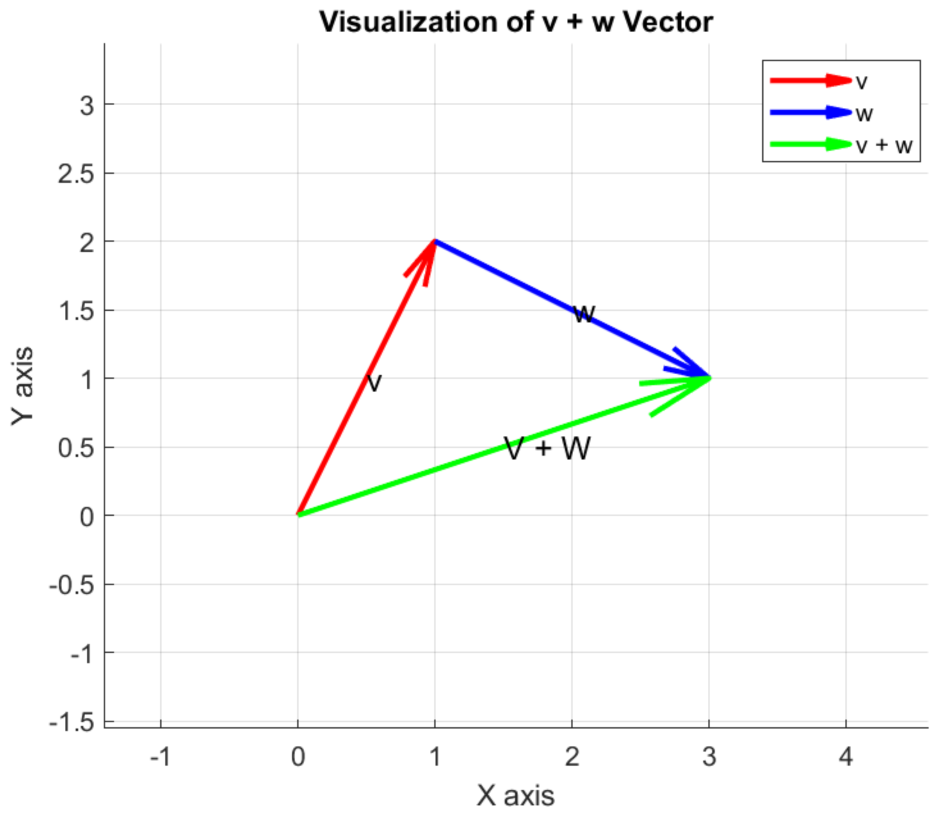
- Orthogonal vector decomposition is dividing one vector to reference vector, [          ] vector and [          ] vector.

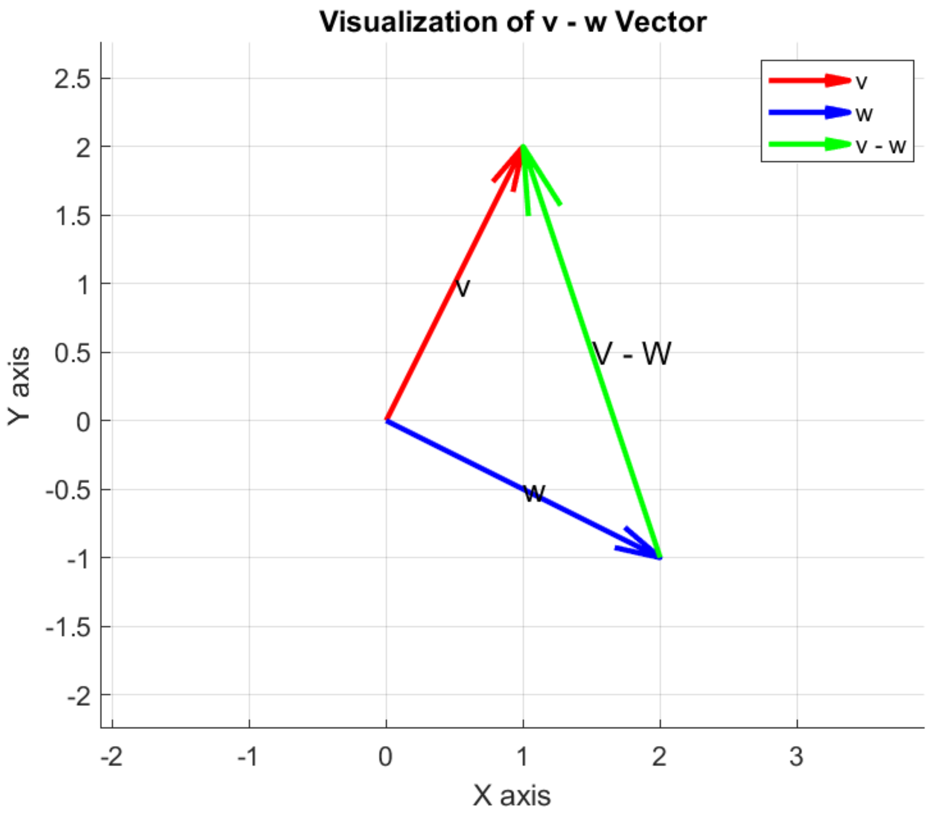
- Decomposition equation can be derived geometrically, but one must remember the phrase '[          ]', a concept implied by the equation.



# Exercise

1. **Write the code that creates figure.**



# Exercise

2. Implement a function that takes a vector as input and outputs a unit vector in the same direction.

# Exercise

3. Write the for loop that transposes row vector to column vector without using built-in functions (e.g., A.T).

# THANK YOU
# FOR YOUR ATTENTION