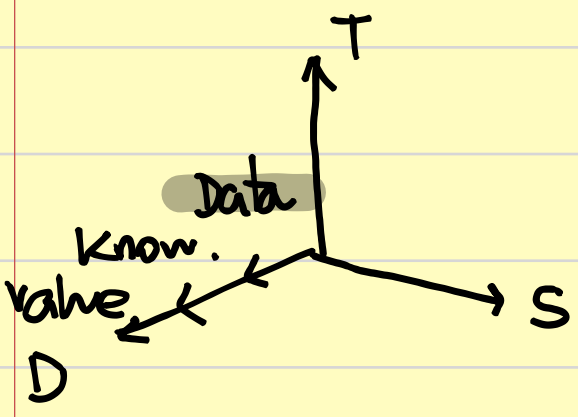
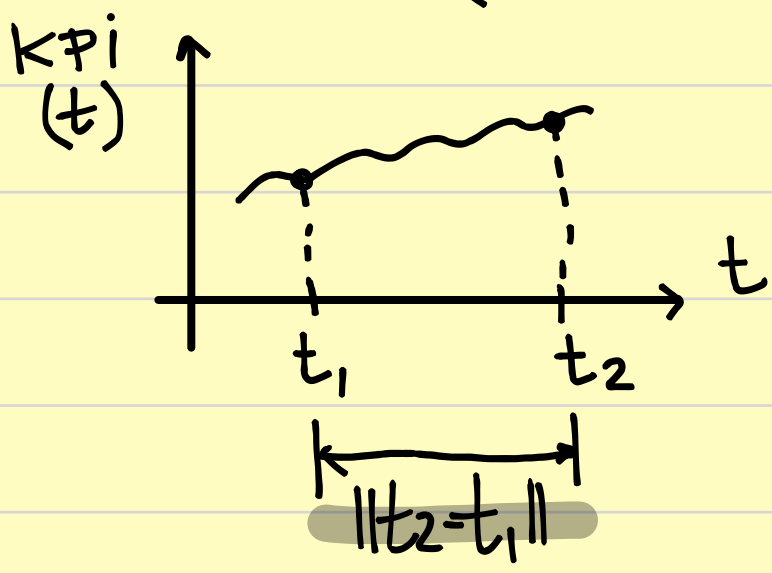


Datasets can be classified in 2 groups:

- Euclidean. if a metric (distance) can be defined to describe relationships btw. them.
- Non-Euclidean. otherwise.

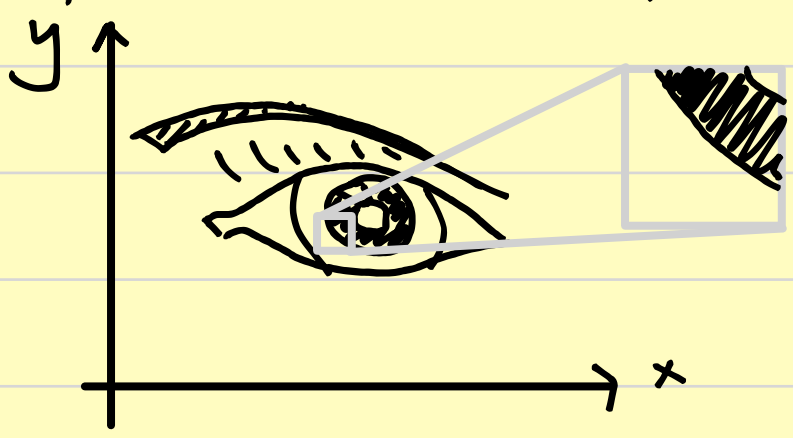
Examples of EUCLIDEAN Datasets:

- TIMESERIES. (1D)



$KPI = f(t)$

- Greyscale IMAGE. (2D)

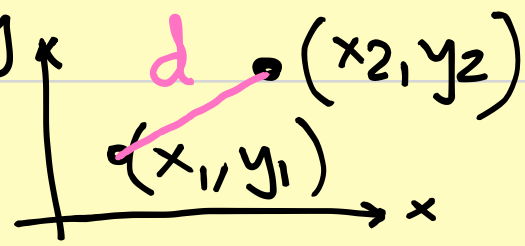


The reason why we "recognize" an EYE in this image is because the pixels are organized in space (2D).

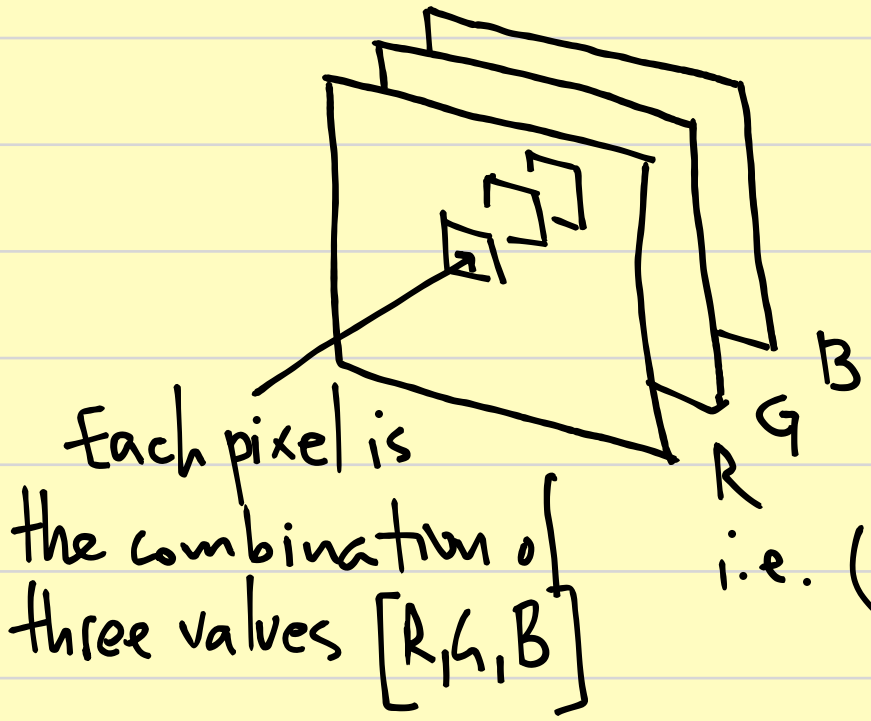
Pixel: □ $[0,255]$ 8 bits: 1 byte: ! ! . . . ! ! ! ! ! ! ! !

$2^8 \equiv 256$

$$d = \sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$



- COLOR IMAGE (3D)



$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Each pixel is the combination of three values [R,G,B] $\in [0,255]$

i.e. (37, 89, 210)
R G B

- VIDEO COLOR + SOUND

  3D + time + sound → 5D

- OLFATORY

  10D - 12D

- HAPTIC

  6D

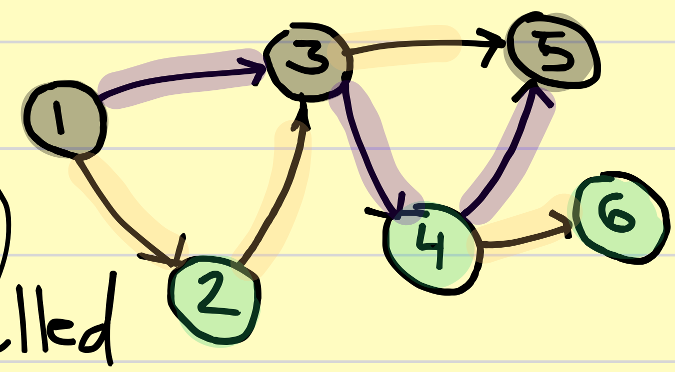
. The hypothesis underlying almost ALL machine learning algorythms is that the structure of the dataset allows for the calculation of a distance. This implies there is a way to "measure" between points in space.

Examples for NON-EUCLIDEAN Datasets: NETWORK

We learn today to transform NON-EUCLIDEAN DATASETS (Networks) into EUCLIDEAN datasets

because 99% of the data generated by businesse is NON-EUCLIDEAN.

What is a network? Network is a list of nodes and connections (edges). Both nodes and edges can have attributes.



A network is defined (mathematically) by a set (group) of nodes and edges. This set is called GRAPH.

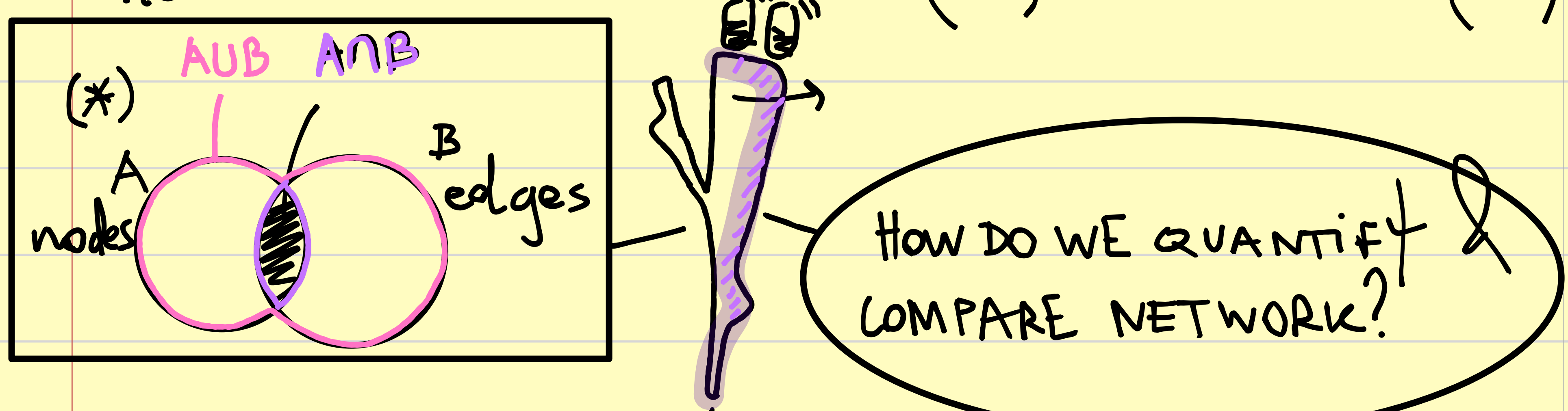
$$\text{GRAPH} = G(N \cup E) \quad [N: \text{nodes}, E: \text{edges}]$$

union (*)

example:

$N: \{1, 2, 3, 4, 5, 6\}$; $E: \{(1\to 2), (1\to 3), (2\to 3), (3\to 4), (3\to 5), (4\to 5), (4\to 6)\}$

Networks can be directed (→) or undirected (—).



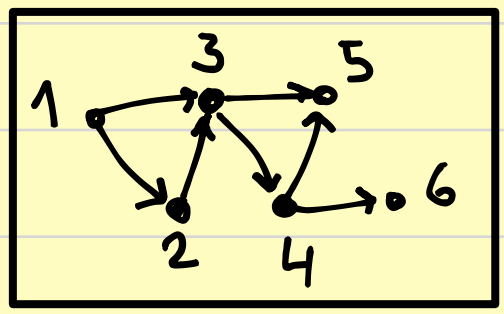
If you can compare & measure, you can manage.

We use 4 KPIs to answer this question.

1. AVERAGE PATH LENGTH (APL). 2. CLUSTERING COEFFICIENT (CC)
3. DEGREE DISTRIBUTION (DD). 4. LAPLACIAN MATRIX ($\mathcal{L}$)

1. AVERAGE PATH LENGTH (APL)

Average distance (steps) between the nodes along the graph.



$$APL = \frac{1}{N(N-1)} \underbrace{\sum_{i=1}^{N}}_{(1)} \underbrace{\sum_{j=1}^{N}}_{(2)} d_{ij}$$

(1) The maximum number of relationships in a graph of N nodes is $N \cdot (N-1)$.

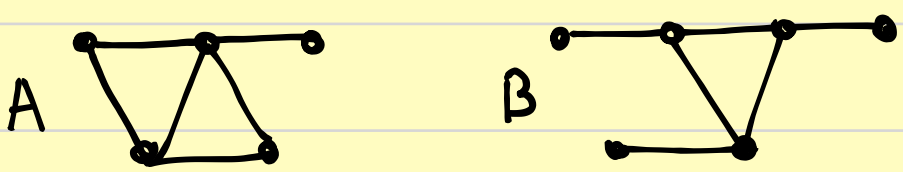
(2) The sum of all paths between the nodes.

$$APL = \frac{1}{6 \cdot (6-1)} \cdot \Big[\overset{1}{\big[} d_{12}\ d_{13}\ d_{14}\ d_{15}\ d_{16} \big] + \overset{2}{\big[} d_{21}\ d_{23}\ d_{24}\ d_{25}\ d_{26} \big] + \overset{3}{\big[} d_{31}\ d_{32}\ d_{34}\ d_{35}\ d_{36} \big] + \overset{4}{\big[} d_{41}\ d_{42}\ d_{43}\ d_{45}\ d_{46} \big] + \overset{5}{\big[} d_{51}\ d_{52}\ d_{53}\ d_{54}\ d_{56} \big] + \overset{6}{\big[} d_{61}\ d_{62}\ d_{63}\ d_{64}\ d_{65} \big] \Big] =$$

$$= \Big[\big[1+1+2+2+3 \big] + \big[1+1+2+2+3 \big] + \big[1+1+1+1+2 \big] + \big[2+2+1+1+1 \big] + \big[2+2+1+1+2 \big] + \big[3+3+2+1+2 \big] \Big] =$$

$$= \frac{50}{30} = 1'67$$

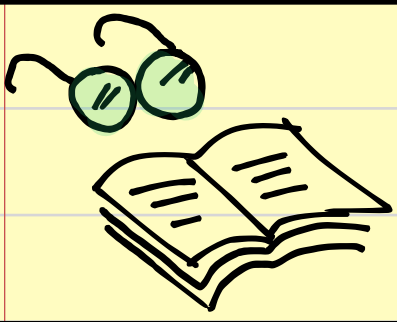
Example:



$APL_A < APL_B$

This means that you get information/material/... faster from one node to the other in network A than in network B. $APL_A < APL_B$.

. When we design business networks, we usually seek for the smallest possible APL.



NETWORK SCIENCE (Barabasi, 2016)

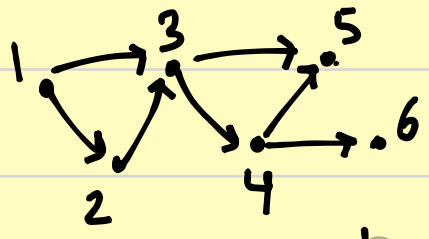
2. CLUSTERING COEFFICIENT (CC)

Describes how good groups are created in the network.

$$CC = \frac{1}{N} \sum_{i=1}^{N} \frac{2L_i}{k_i(k_i-1)}$$

$L_i \equiv$ Number of relationships between the neighbours of node "i". (FRIENDS of "i" WHO ARE FRIENDS to each other)

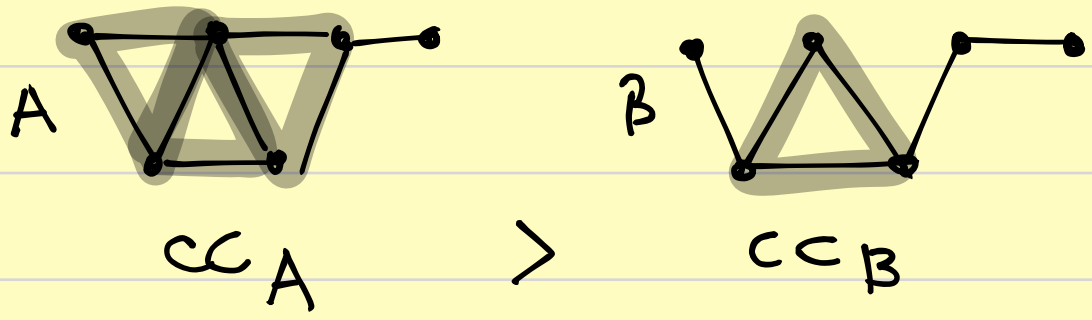
$k_i \equiv$ Number of neighbours of node "i".



$$CC = \frac{1}{6}\left[1\left[\frac{2 \cdot 1}{2\cdot(2-1)}\right]_{k_1}^{L_1} + 2\left[\frac{2 \cdot 1}{2\cdot(2-1)}\right]_{k_2}^{L_2} + 3\left[\frac{2 \cdot 2}{4\cdot(4-1)}\right]_{k_3}^{L_3} + 4\left[\frac{2 \cdot 1}{3(3-1)}\right]_{k_4}^{L_4} + 5\left[\frac{2 \cdot 1}{2(2-1)}\right]_{k_5}^{L_5} + 6\left[\frac{\emptyset}{\emptyset}\right]\right] = \frac{11}{18} = 0'611$$

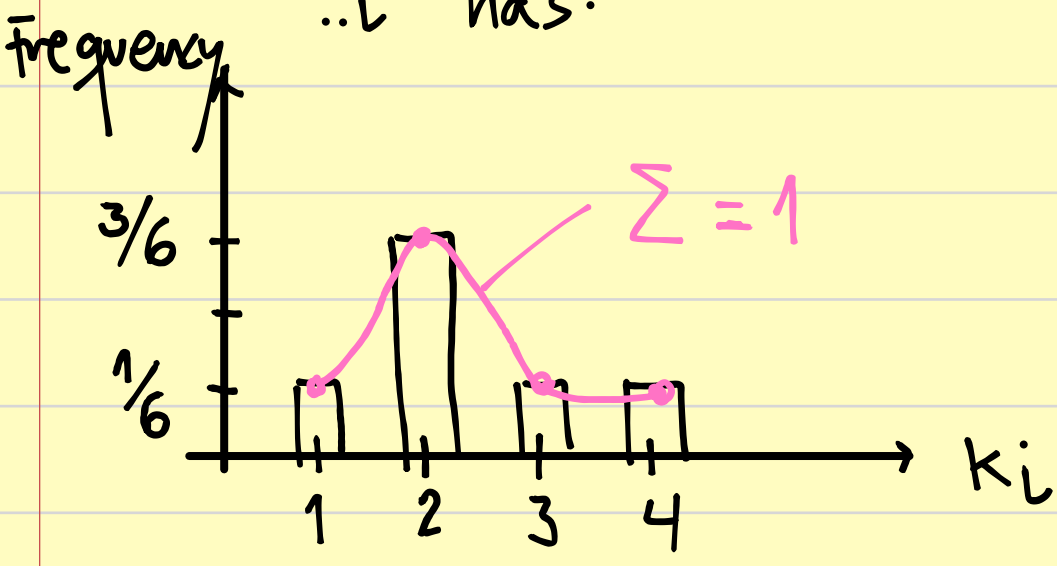
The cc measures how tight clusters (groups) are created in the network. Therefore, the higher, the better usually. We seek for maximization of the CC.

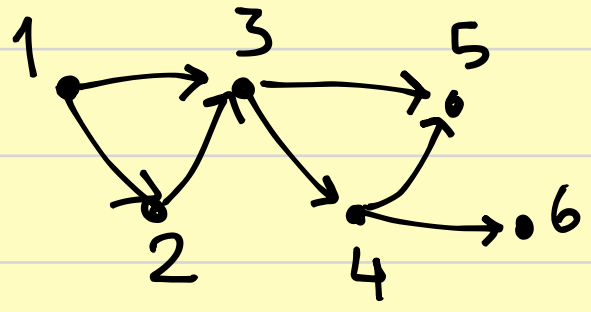
Example.



3. DEGREE DISTRIBUTION (DD)

$k_i \equiv$ degree of a node: number of neighbours a node "i" has.





Depending on the structure of the DD, we have different types of networks.

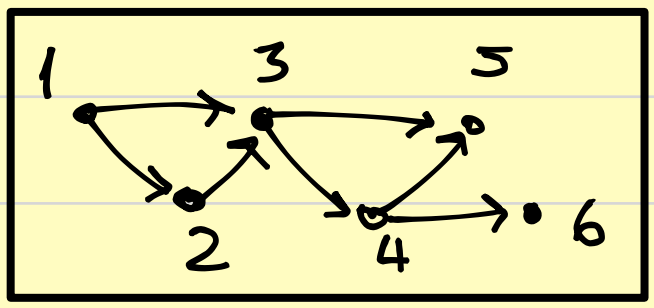
4. LAPLACIAN MATRIX ($\mathcal{L}$)

The laplacian matrix contains all relevant network information and is EUCLIDEAN (!), whereas the graph is NOT EUCLIDEAN.

Is defined as $\mathcal{L} = D - A$

D = Degree matrix
A = Adjacency matrix

$$D = \begin{cases} k_i & i = j \\ 0 & i \neq j \end{cases} \quad i \quad A = \begin{cases} 1 & \text{if connection btw } i,j \\ 0 & \text{otherwise.} \end{cases}$$



$G(N,E)$

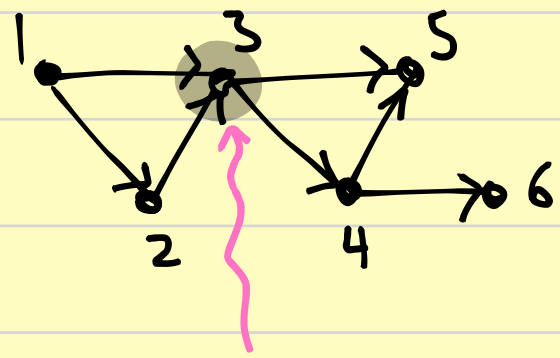
$$\mathcal{L} = \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix}\begin{bmatrix} 2 & & & & & \\ & 2 & & \phi & & \\ & & 4 & & & \\ & & & 3 & & \\ & \phi & & & 2 & \\ & & & & & 1 \end{bmatrix} - \begin{bmatrix} 0&1&1&0&0&0 \\ 0&0&1&0&0&0 \\ 0&0&0&1&1&0 \\ 0&0&0&0&1&1 \\ 0&0&0&0&0&0 \\ 0&0&0&0&0&0 \end{bmatrix} = \begin{matrix} 1&2&3&4&5&6 \end{matrix}$$

D $\qquad$ A

$$\mathcal{L} = \begin{bmatrix} 2&-1&-1&0&0&0 \\ 0&2&-1&0&0&0 \\ 0&0&4&-1&-1&0 \\ 0&0&0&3&-1&-1 \\ 0&0&0&0&2&0 \\ 0&0&0&0&0&0 \end{bmatrix}$$

✓ This laplacian matrix has the same information as the graph.

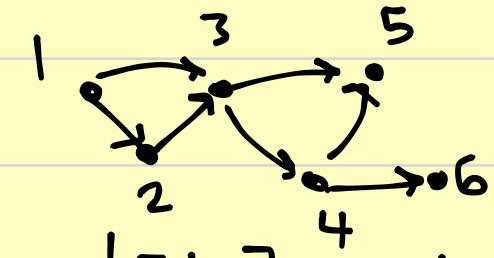
- we can discover bottlenecks in the network analytically.



The eigenvectors of the laplacian matrix contain information about its partitions.

$$A\vec{v} = \lambda\vec{v} \rightarrow \det(A - \lambda I) = 0 \rightarrow \lambda \rightarrow \vec{v}$$

. The second eigenvector of the $\mathcal{L}$ is called FIEDLER vector and has a symbol structure that helps find the bottleneck.



$$\vec{v}_2 = \begin{bmatrix} + \\ + \\ 0 \\ - \\ - \\ - \end{bmatrix} \leftarrow \text{Bottleneck}$$

$$\vec{v}_2 = \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix}\begin{bmatrix} + \\ + \\ 0 \\ - \\ - \\ - \end{bmatrix} \leftarrow \text{Bottleneck}$$

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

$$\det[A - \lambda I] = 0 \rightarrow \det\begin{bmatrix} 1-\lambda & 2 \\ 2 & 3-\lambda \end{bmatrix} = 0 \rightarrow$$

$$\rightarrow (1-\lambda)(3-\lambda) - 4 = 0 \rightarrow \lambda^2 - 4\lambda + 3 - 4 = 0 \rightarrow$$

$$\rightarrow \lambda^2 - 4\lambda - 1 = 0 \rightarrow \lambda = \frac{4 \pm \sqrt{16+4}}{2} = \frac{4 \pm \sqrt{20}}{2} = \begin{cases} \lambda_1 = 4'24 \\ \lambda_2 = -0'24 \end{cases}$$

Eigenvalues

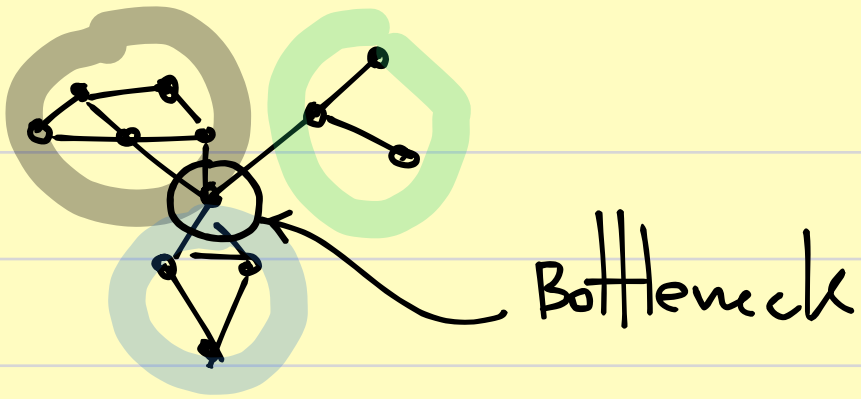
λ_1

$$\bullet \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix}\begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix} = 4'24\begin{bmatrix} v_{11} \\ v_{12} \end{bmatrix}$$

$$\rightarrow \begin{matrix} v_{11} + 2v_{12} = 4'24\, v_{11} \\ 2v_{11} + 3v_{12} = 4'24\, v_{12} \end{matrix} \rightarrow \begin{matrix} 2v_{12} = 3'24\, v_{11} \\ 2v_{11} = 1'24\, v_{12} \end{matrix} \rightarrow \begin{matrix} v_{11} = \dots \\ v_{12} = \dots \end{matrix}$$

λ_2

$$\bullet \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix}\begin{bmatrix} v_{21} \\ v_{22} \end{bmatrix} = -0'24\begin{bmatrix} v_{21} \\ v_{22} \end{bmatrix} \rightarrow \dots \rightarrow \begin{matrix} v_{21} = \dots \\ v_{22} = \dots \end{matrix}$$



$\vec{v}_2 = \begin{bmatrix} + \\ + \\ 0 \\ - \\ - \end{bmatrix}$ ← Bottleneck