

A short course on

# Social Network Analysis

Lecture 0

Antonio Calcagni

`<antonio.calcagni@unipd.it>`

Master in Data Science per il Welfare  
University of Salento

AY 2023/24

# Outline

- 1 Data structures
- 2 Notable examples of network data
- 3 Background on graph theory
- 4 Typical network features

# Outline

- 1 Data structures
- 2 Notable examples of network data
- 3 Background on graph theory
- 4 Typical network features

# Data structures

Statistics is the art of collecting, modeling, analyzing, and interpreting data. To pursue these goals, several data structures can be used to represent data (e.g., units, variables) and the relationships among them. The main available structures are the following:

- Arrays
- Dictionaries
- Trees
- Graphs

# Data structures

## Arrays

An array  $\mathbf{X}_{I \times J \times \dots \times K}$  is a collection of  $IJK$  data items of the **same type**, where each item  $x_{ij\dots k}$  is referred to as an element. The elements in an array can be of any valid data type, such as character, integer, or real/double.

The elements of array share the same variable name but each one carries a different index number known as subscript. The array can be 1-dimensional (row/column vectors), 2-dimensional (matrices) or multidimensional.

# Data structures

## Arrays

$$\begin{bmatrix} x_{11} & \cdots & x_{1j} & \cdots & x_{1J} \end{bmatrix}$$

$$\begin{bmatrix} x_{11} \\ \vdots \\ x_{i1} \\ \vdots \\ x_{I1} \end{bmatrix}$$

Example of a 1-dimensional matrix  $\mathbf{X}_{1 \times J}$  (row vector) or  $\mathbf{X}_{I \times 1}$  (column vector)

# Data structures

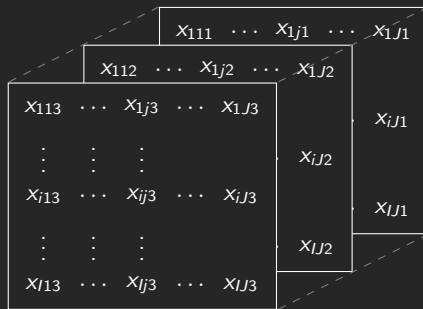
## Arrays

$x_{11}$	$\dots$	$x_{1j}$	$\dots$	$x_{1J}$
$\vdots$		$\vdots$		$\vdots$
$x_{i1}$	$\dots$	$x_{ij}$	$\dots$	$x_{iJ}$
$\vdots$		$\vdots$		$\vdots$
$x_{I1}$	$\dots$	$x_{Ij}$	$\dots$	$x_{IJ}$

Example of a 2-dimensional matrix  $\mathbf{X}_{I \times J}$

# Data structures

## Arrays



Example of a multidimensional matrix  $\mathbf{X}_{I \times J \times 3}$



# Data structures

## Arrays

### Examples:

- Data series (1-d matrix)
- Units  $\times$  Variables matrices (2-d matrix)
- Corr/Cov matrices (2-d matrix symmetric and positive-definite)
- Image data, Tensor data (M-d matrix like those used in Neural Networks)

# Data structures

## Dictionaries

A dictionary is a data structure that stores data in **key-value pairs**  $\langle x, a \rangle$ . Each **unique key**  $a_i$  is associated with a specific value  $x_i$ , and the key can be used to retrieve the corresponding value efficiently. Dictionaries are also known as associative arrays or hash maps.

# Data structures

## Dictionaries

$a_1$	$x_1$
$\vdots$	$\vdots$
$a_i$	$x_i$
$\vdots$	$\vdots$
$a_l$	$x_l$

Example of a dictionary with  $l$  elements. The key values can be represented using characters (labels).

# Data structures

## Dictionaries

### Examples:

- Frequency distributions with  $a$  representing the units and  $x$  the associated frequency or counts.

# Data structures

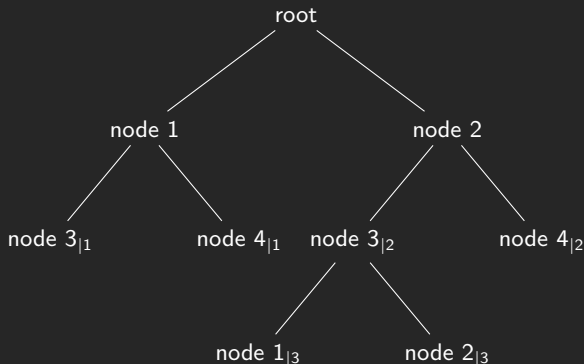
## Trees

A tree data structure is a **hierarchical** data structure that consists of nodes connected by edges. Each node can have multiple child nodes, but only one parent node. It has no cycles: there is exactly one path between any two nodes.

The topmost node in the tree is called the root node, which has no parent. The terminal node with no children is called leaf.

# Data structures

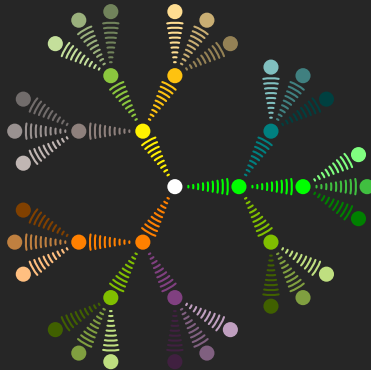
## Trees



Example of a standard tree with height equal to 3.

# Data structures

## Trees



Example of a phylogenetic tree with height equal to 3.

# Data structures

## Trees

### Examples:

- Dendrograms to represent hierarchical clustering
- Classification and Regression trees



# Data structures

## Graphs

A graph consists of a set of **nodes** (also called vertices) and a set of **edges** (also called arcs) that connect pairs of nodes. Graphs are versatile and powerful for representing complex relationships between objects or units.

# Data structures

## Graphs

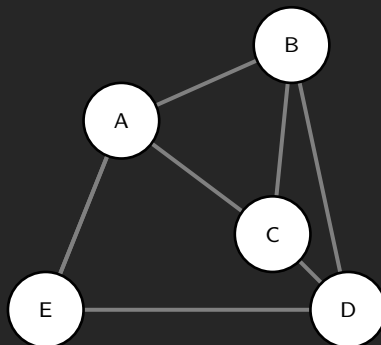
A graph consists of a set of **nodes** (also called vertices) and a set of **edges** (also called arcs) that connect pairs of nodes. Graphs are versatile and powerful for representing complex relationships between objects or units.

Unlike trees, graphs have a more general structure without a strict hierarchy and they

- can contain cycles (i.e., multiple paths between nodes)
- do not have a single root node
- can be either connected (there is a path between every pair of nodes) or disconnected (some nodes are not reachable from others)
- can have directed edges (one-way connections) or undirected edges (two-way connections)

# Data structures

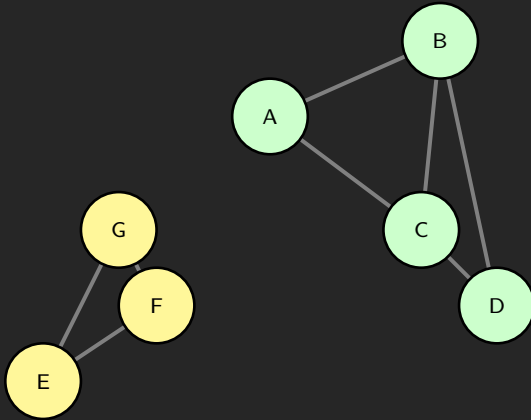
## Graphs



Example of undirected and connected network structure with 5 nodes.

# Data structures

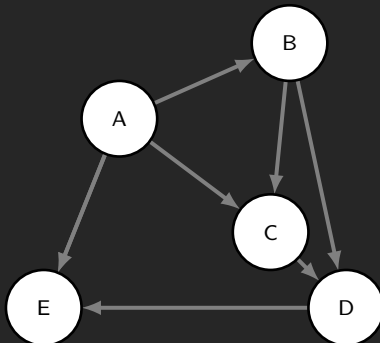
## Graphs



Example of undirected and disconnected network structure with 7 nodes.

# Data structures

## Graphs



Example of directed and connected network structure with 5 nodes.

# Data structures

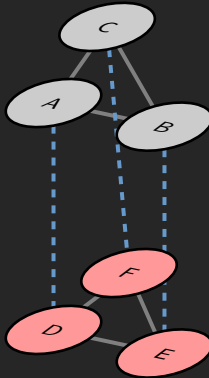
## Graphs

### Examples:

- Properly network-based data  
(e.g., social, economic, biological, transportation, communication)
- Markov-based models  
(e.g., dynamic, with hidden states, queues, supply chains)

# Data structures

## Graphs



Example of multiplex network structure with 2 layers.

# Data structures

## Graphs

Multiplex networks are powerful and versatile in representing complex and structured phenomena where basic network structures can vary as a function of other variables (e.g., covariates).

### **Examples:**

A typical example is that of social network data where relationships among units/individuals vary as a function of the type of relationship (e.g., friendship, professional, familiar).

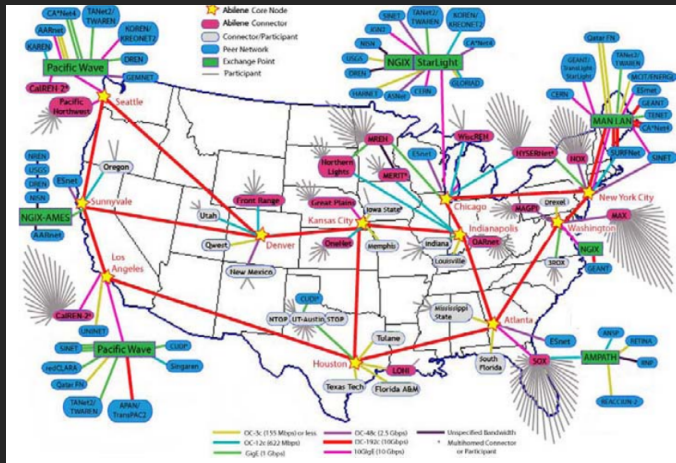


# Outline

- 1 Data structures
- 2 Notable examples of network data
- 3 Background on graph theory
- 4 Typical network features

# Notable examples of network data

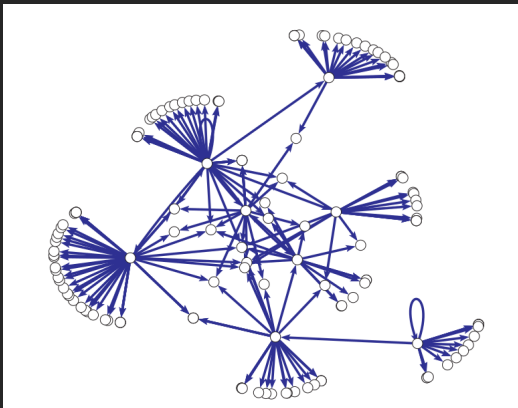
## Abilene Internet Network



Source: Kolaczyk E. (2009). *Statistical analysis of network data*. Springer

# Notable examples of network data

## Pattern citation among Internet blogs

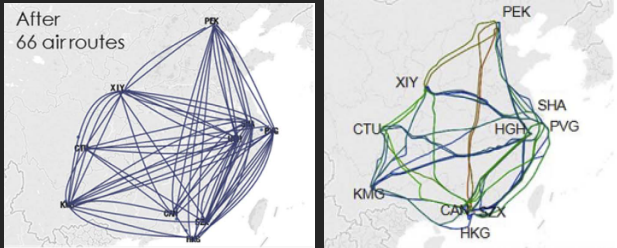


Source: Kolaczyk E. (2009). *Statistical analysis of network data*. Springer

Note: Directed edges indicate direct webpage links from a webpage to another.

# Notable examples of network data

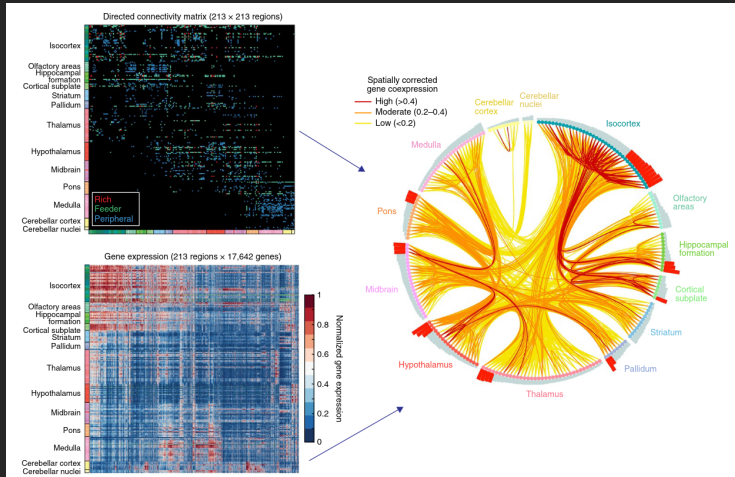
## Air traffic networks



Source: Ren, P., & Li, L. (2018). Characterizing air traffic networks via large-scale aircraft tracking data. *Journal of Air Transport Management*, 67, 181-196.

# Notable examples of network data

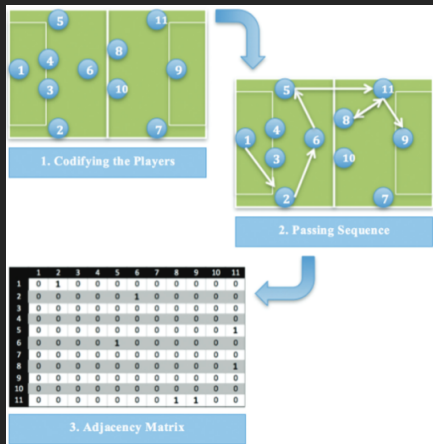
## Anatomical connectivity and brain network



Source: Bassett, D. S., & Sporns, O. (2017). Network neuroscience. *Nature Neuroscience*, 20(3), 353-364.

# Notable examples of network data

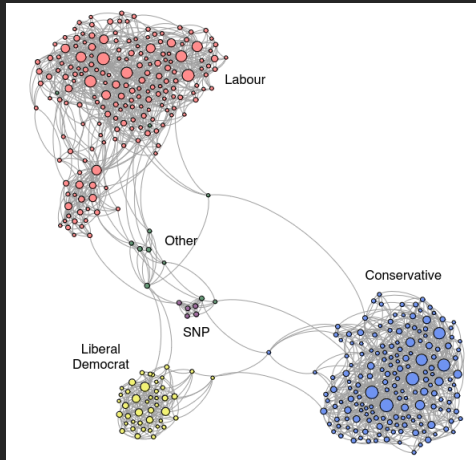
## Network analysis on team sports



Source: Clemente, F. M., Silva, F., Martins, F. M. L., Kalamaras, D., & Mendes, R. S. (2016). Performance Analysis Tool for network analysis on team sports: A case study of FIFA Soccer World Cup 2014. *Proceedings of the Institution of Mechanical Engineers, Part P: Journal of Sports Engineering and Technology*, 230(3), 158-170.

# Notable examples of network data

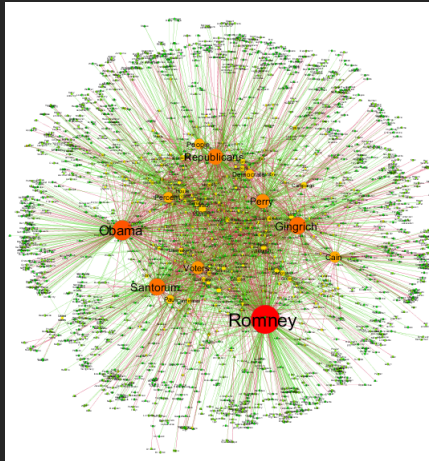
## Network analysis of political groups



Source: Greene, D., & Cunningham, P. (2013). Producing a unified graph representation from multiple social network views. *In Proceedings of the 5th annual ACM web science conference.*

# Notable examples of network data

## Network analysis of narratives in US elections 2012



Source: Sudhahar, S., Veltri, G., Cristianini N. (2015). Automated analysis of the US presidential elections using Big Data and network analysis. *Big Data & Society*, 2(1).



# Outline

- 1 Data structures
- 2 Notable examples of network data
- 3 Background on graph theory
- 4 Typical network features

# Background on graph theory

## Introduction

So far we have treated a **network** structure qualitatively, as a collection of nodes and edges without any formal representation. The graph theory is usually adopted to provide a mathematical representation of network structures.

# Background on graph theory

## Basic definitions

A **graph**  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  consists of

- a set  $\mathcal{V} \subset \mathbb{N}$  of vertices (*nodes*)
- a set  $\mathcal{E} = \{(u, v) \in \mathcal{V} \times \mathcal{V} \mid u \neq v\}$  of edges (*links*) with  $(u, v)$  unordered pair of distinct vertices.

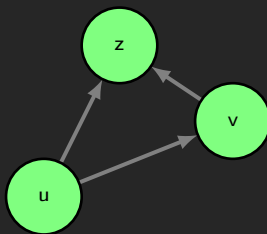
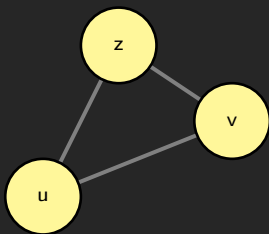
The number of vertices  $N_v = |\mathcal{V}|$  is the **order** of the graph  $\mathcal{G}$  whereas the number of edges  $N_e = |\mathcal{E}|$  is the **size** of  $\mathcal{G}$ .

A graph  $\mathcal{H}$  is a **subgraph** of  $\mathcal{G}$  if  $\mathcal{V}_{\mathcal{H}} \subseteq \mathcal{V}_{\mathcal{G}}$  and  $\mathcal{E}_{\mathcal{H}} \subseteq \mathcal{E}_{\mathcal{G}}$ .

# Background on graph theory

## Basic definitions

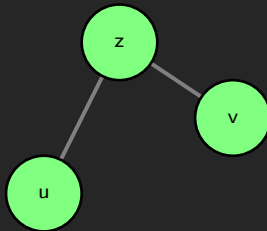
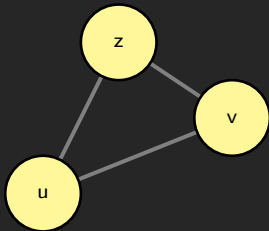
A graph  $\mathcal{G}$  for which the pair  $\{u, v\}$  is distinct from  $\{v, u\}$  is called **direct graph**. Otherwise is an **undirect graph**.



# Background on graph theory

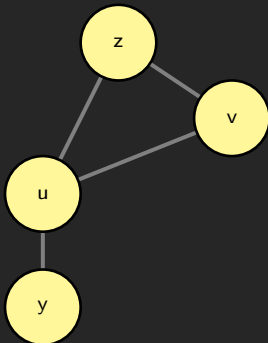
## Basic definitions

Two vertices  $u \in \mathcal{V}$  and  $v \in \mathcal{V}$  are said to be **adjacent** if there is an edge  $e \in \mathcal{E}$  between them. In this case, they are also **connected**. However, three vertices  $u, v, z$  are said to be **connected** if there is a path that can be traversed to go from one vertex to the other. Adjacency implies connectivity but the opposite is not always true.



# Background on graph theory

## Basic definitions



A **path** connecting  $z$  and  $y$  is a sequence of intermediate vertices connecting them. In this case, we have several possibilities, e.g.  $p_1 = \{z, u, y\}$ ,  $p_2 = \{z, v, u, y\}$ . The quantity  $L(p_i) = |p_i| - 1$  is the **path length**.

The **geodesic distance** between  $z$  and  $y$  is the shortest path connecting them.

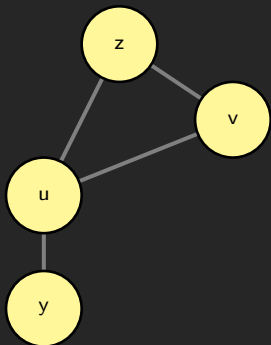
The **diameter** of a graph is the maximum geodesic distance between any pair of vertices.

Shortest path:  $L(\{z, u, y\}) = 2$

Diameter:  $D(\mathcal{G}) = 3$

# Background on graph theory

## Basic definitions



The **density** is a measure that indicates how dense an undirect graph is relative to its size:

$$D(\mathcal{G}) = \frac{2N_e}{N_v(N_v - 1)}$$

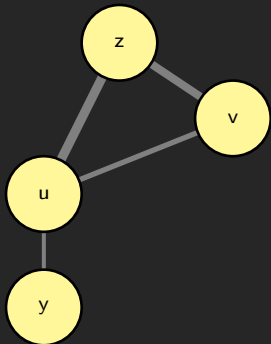
When  $\mathcal{G}$  is directed then the numerator of  $D(\mathcal{G})$  is simply  $N_e$ .

$D(\mathcal{G}) \in (0, 1)$  indicates partial connectivity, with  $D(\mathcal{G}) = 1$  indicating that  $\mathcal{G}$  is a dense graph.

Density:  $D(\mathcal{G}) = 0.33$

# Background on graph theory

## Basic definitions

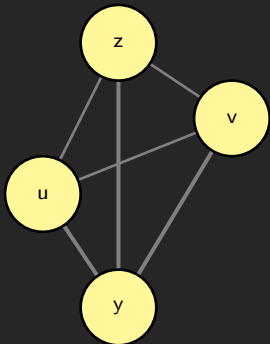


In simple network data, edges of a graph  $\mathcal{G}$  are unweighted. However, there are several important situations where edges should be weighted according to their importance. In this case,  $\mathcal{G}$  is said to be **weighted graph** and edges are usually depicted by varying their size.



# Background on graph theory

## Relevant types of graph



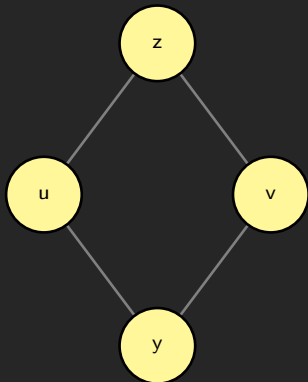
A graph  $\mathcal{G}$  is said to be **complete** if every vertex  $v_i \in \mathcal{V}$  is joined to every other vertex  $v_j \in \mathcal{V}$ .

It contains  $N_v$  vertices and it has  $\frac{N_v(N_v-1)}{2}$  edges.

It is the densest graph with maximum connectivity. The case  $\mathcal{G}_3$  depicts a triangle.

# Background on graph theory

## Relevant types of graph



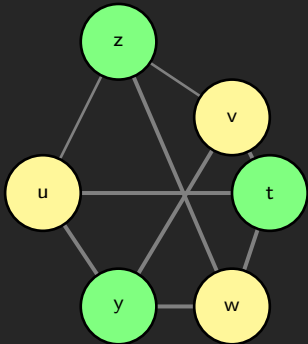
Example of 2-regular graph

A graph  $\mathcal{G}_d$  is said to be  **$d$ -regular** if every vertex in  $\mathcal{G}$  has exactly  $d$  edges connected to it.

They are usually used in network design (to design networks with uniform connectivity), in coding theory (they are applied in error-correcting codes), or combinatorial optimization.

# Background on graph theory

## Relevant types of graph

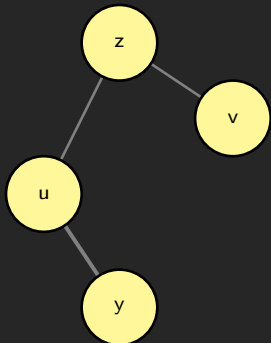


A graph  $\mathcal{G}_d$  is said to be **bipartite** if there exists  $\mathcal{V}_0 \subseteq \mathcal{V}$  and  $\mathcal{V}_1 = \mathcal{V} \setminus \mathcal{V}_0$  s.t. every edge of  $\mathcal{G}$  has one vertex in  $\mathcal{V}_0$  and one in  $\mathcal{V}_1$ .

This type of graphs are typically used to represent membership networks, with *members* denoted by vertices in  $\mathcal{V}_1$  and *organizations* by vertices in  $\mathcal{V}_0$

# Background on graph theory

## Relevant types of graph

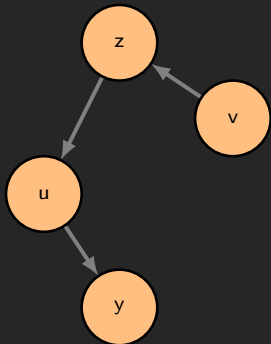


A graph  $\mathcal{G}$  is said to be **undirected acyclic** if there are no loops/circuits (i.e., a disconnected graph).

This type of graphs are typically represented as trees (connected acyclic graphs) or forests (collection of disjoint trees).

# Background on graph theory

## Relevant types of graph

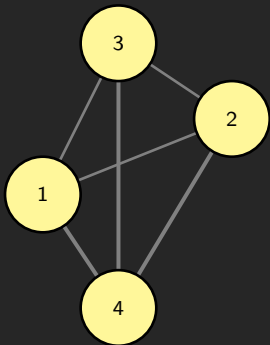


A graph  $\mathcal{G}$  is said to be **directed acyclic** (DAG) if there are no directed loops/cycles (i.e., no way to start at a vertex  $v$ , follow the directed edges, and return to  $v$  again).

This type of graphs have a topological ordering and they are commonly used to represent dependencies (e.g., causal models). In Epidemiology, causal DAGs are systematic representation of causal relationships.

# Background on graph theory

## Matrix representation of a graph



The connectivity of a graph  $\mathcal{G}$  can be represented in terms of a  $N_v \times N_v$  binary **adjacency matrix**  $\mathbf{A}$  with entries:

$$a_{ij} = \begin{cases} 1, & \text{if } \{i, j\} \in \mathcal{E} \\ 0, & \text{otherwise} \end{cases}$$

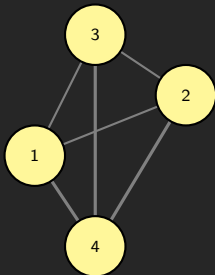
The matrix  $\mathbf{A}$  is non-zero for those entries whose row-column indices correspond to vertices joined by an edge.

$$\mathbf{A}_{4 \times 4} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

Note: for undirected graphs,  $\mathbf{A}$  is symmetric whereas for directed graphs  $\mathbf{A}$  is not necessarily symmetric.

# Background on graph theory

## Matrix representation of a graph



$$\mathbf{A}_{4 \times 4} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

### Vertex degree:

$$d(i)^{\text{out}} = A_{i+} = \sum_{j=1}^{N_v} a_{ij}$$

$$d(j)^{\text{in}} = A_{+j} = \sum_{i=1}^{N_v} a_{ij}$$

### Number of walks:

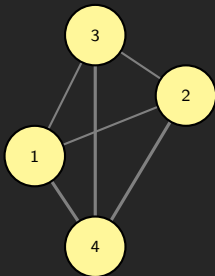
If  $\mathbf{A}^r$  then  $a_{ij}^r$  is the number of walks of length  $r$  between  $i$  and  $j$  on  $\mathcal{G}$ .

### Regular graphs:

$\mathcal{G}$  is *regular* if the maximum degree  $d_{\max}$  of  $\mathcal{G}$  is an eigenvalue of  $\mathbf{A}$ .

# Background on graph theory

## Matrix representation of a graph



From  $\mathbf{A}_{N_v \times N_v}$  one can construct the so-called **incidence matrix**  $\mathbf{B}_{N_v \times N_e}$ , which relates vertices to edges as follows

$$b_{ij} = \begin{cases} 1, & \text{if vertex } i \text{ is incident to edge } j \\ 0, & \text{otherwise} \end{cases}$$

In the example, the collection of edges is

$$e_1 = \{1, 2\}, e_2 = \{1, 3\}, e_3 = \{1, 4\}$$

$$e_4 = \{2, 3\}, e_5 = \{2, 4\}, e_6 = \{3, 4\}$$

and the incidence matrix is the following

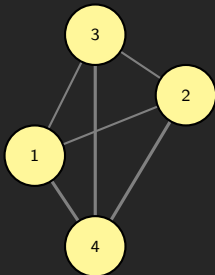
$$\mathbf{A}_{4 \times 4} = \begin{bmatrix} 0 & & & \\ 1 & 0 & & \\ 1 & 1 & 0 & \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

$$\mathbf{B}_{4 \times 6} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$



# Background on graph theory

## Matrix representation of a graph



$$\mathbf{A}_{4 \times 4} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

The **Laplacian matrix**  $\mathbf{L}_{N_v \times N_v}$  encapsulates the connectivity and structure of the graph  $\mathcal{G}$  and it is computed as

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

where  $\mathbf{D} = \text{diag}(\mathbf{d})$  is the diagonal matrix containing the  $N_v \times 1$  vector of degrees  $\mathbf{d}$  of  $\mathcal{G}$ .

Studying the matrix  $\mathbf{L}$  provides valuable insights into the structure of  $\mathcal{G}$ . For instance, since the first eigenvalue  $\lambda_1 = 0$  of  $\mathbf{L}$ , then the larger  $\lambda_2$  is, the more *connected*  $\mathcal{G}$  is. Consequently, the more difficult it is to separate  $\mathcal{G}$  into disconnected subgraphs.

# Outline

- 1 Data structures
- 2 Notable examples of network data
- 3 Background on graph theory
- 4 Typical network features

# Typical network features

**Small world:** nodes are highly clustered locally yet connected by short paths globally, promoting efficient communication and network navigation.

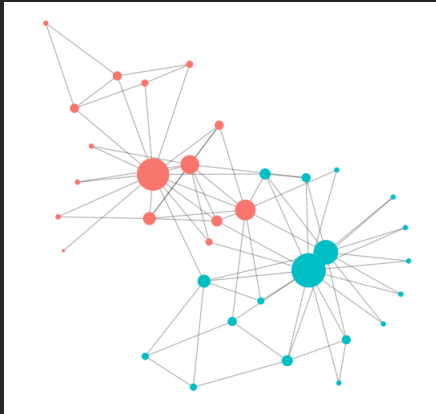
**Hub:** a few nodes with a higher number of connections (edges) compared to other nodes in the network. Hubs play a crucial role in network structure and function, often serving as pivotal points of connectivity or influence.

**Scale-free:** The *degree distribution* follows a power-law, meaning that there are a few highly connected nodes (hubs) and many nodes with relatively few connections. This property contrasts with random networks, where node degrees follow a Poisson distribution.

**Community structure:** groups of nodes that are densely connected internally but less connected to nodes in other groups, revealing natural clusters or communities within the network.

**Homophily:** tendency of nodes to connect preferentially with other nodes that share similar attributes or characteristics.

# Typical network features



A typical social network with two communities, three hubs, and a certain degree of homophily in the communities. The small-world property can be also appreciated.