# Random walks on networks
# Part 1

R. Lambiotte
renaud.lambiotte@maths.ox.ac.uk

*"I think the next century will be the century of complexity."*

Stephen Hawking

**com.plex**
*[adj., v. kuh m-pleks, kom-pleks; n. kom-pleks]*

1) composed of many interconnected parts; compound; composite: a complex highway system

2) characterized by a very complicated or involved arrangement of parts, units, etc.: complex machinery

3) so complicated or intricate as to be hard to understand or deal with: a complex problem

*Source: Dictionary.com*

**Box 1.1**

"Complex systems consist of a large number of interacting components. The interactions give rise to emergent hierarchical structures. The components of the system and properties at systems level typically change with time."

H.J. Jensen, in Encyclopedia of Complexity and Systems Science

# Importance of metaphors, analogies and common languages



## LATEST NEWS

COBBS LAB, ISC-CNR

**All together now.** Flocks of starlings fly over Rome's city center.

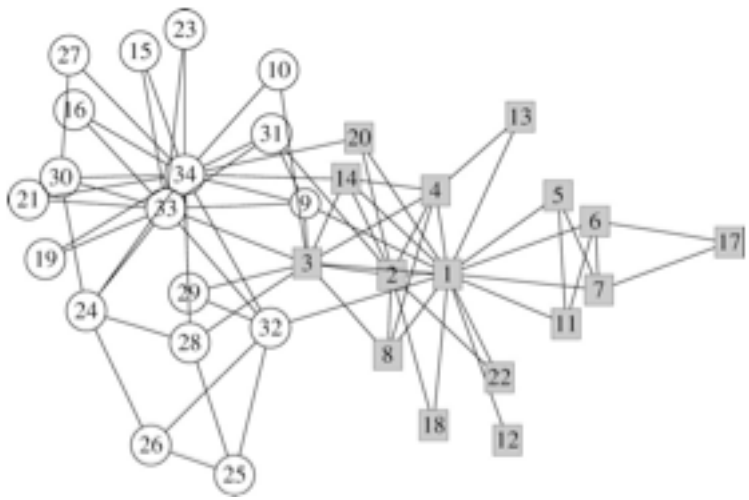
## How bird flocks are like liquid helium

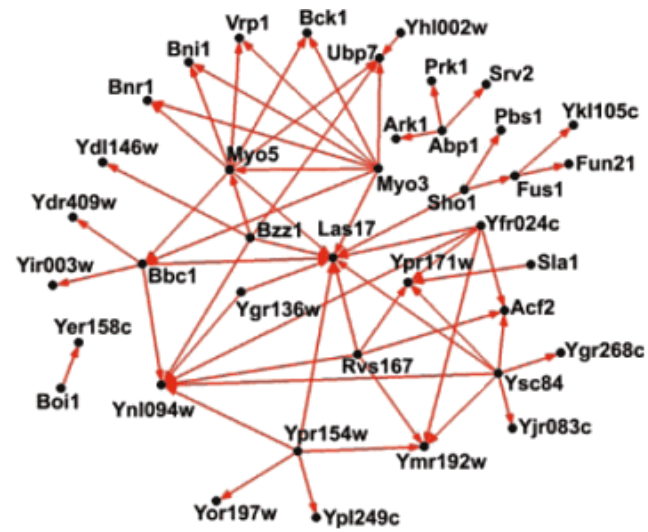
# Tool box

Agent-based and numerical simulations
(Non-linear) Dynamical systems
Stochastic processes
Data mining and optimisation
Networks

# Tool box

Agent-based and numerical simulations
(Non-linear) Dynamical systems
Stochastic processes
Data mining and optimisation
**Networks**

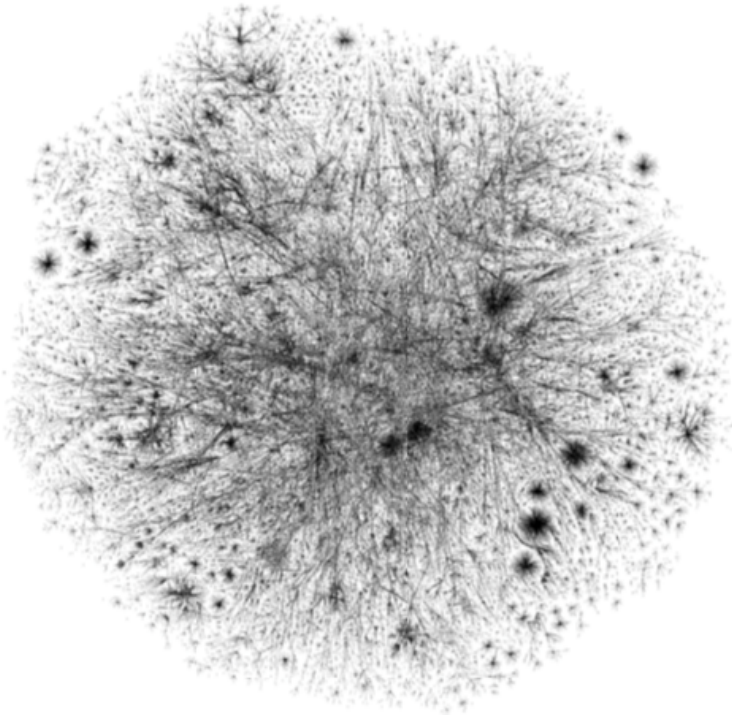


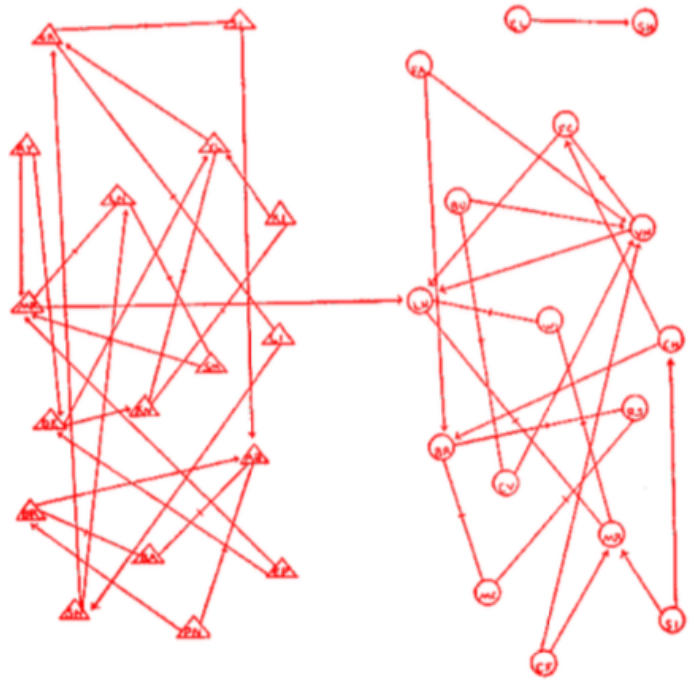



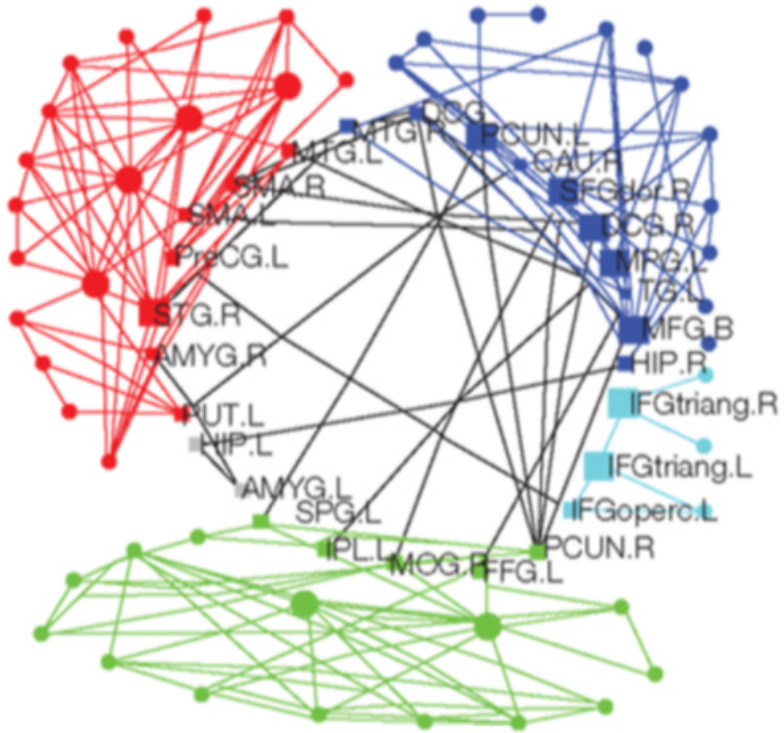

**Social networks**
- Collaboration networks
- Communication networks
- Online social networks

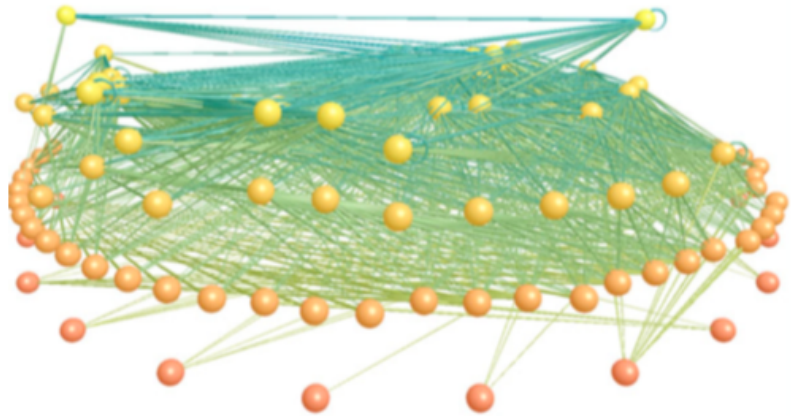
**Biological networks**
- Protein-protein interaction networks
- Neural networks
- Food webs


KONINGSBERGA






DCG
MTG.R
PCUN.L
MTG.L
CAU.R
SMA.R
SFGdor.R
SMA.L
DCG.R
PreCG.L
MFG.L
TG.L
STG.R
MFG.B
AMYG.R
HIP.R
PUT.L
IFGtriang.R
HIP.L
IFGtriang.L
AMYG.L
SPG.L
IFGoperc.L
IPL.L
PCUN.R
MOG.R
FFG.L



# Consensus

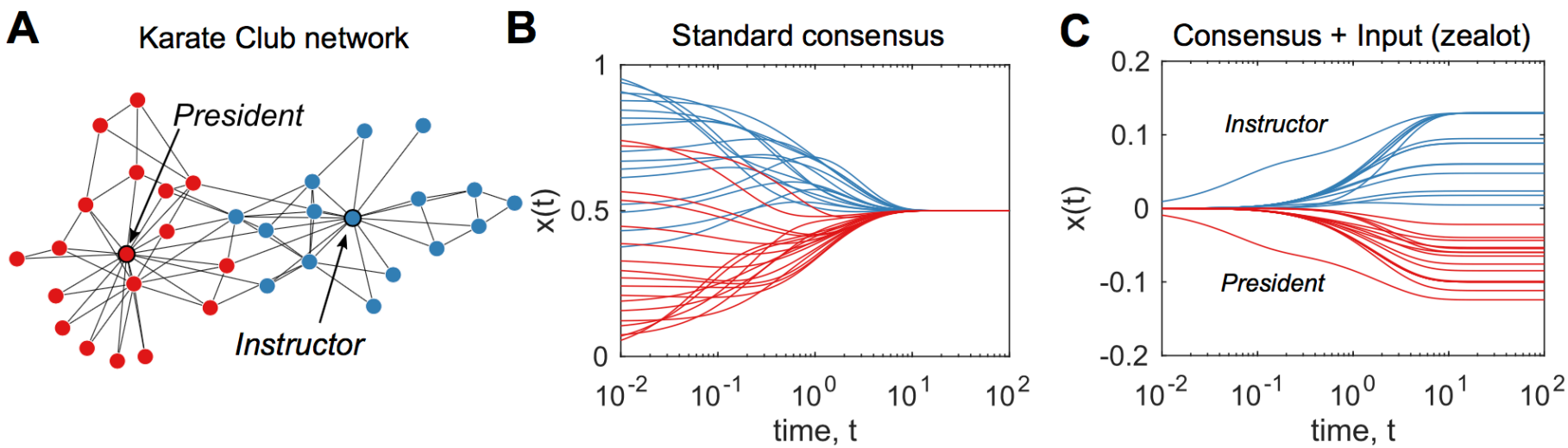



**Figure 13.1 Consensus dynamics on the Karate Club network.** **A** The Karate Club network originally analysed by Zachary [8] with nodes coloured according to the split that occurred in the real case. **B** Consensus dynamics on the Karate club network starting from a random initial condition. As time progresses, the states of the individual nodes become more aligned and eventually reach the consensus value equal to the arithmetic average of the initial condition. Note that above the time scale given by the eigenvalue $1/\lambda_2(\mathbf{L}) \approx 1/0.47$, the agents converge into two groups that reflect the observed split before converging to global consensus (see Section 13.3.1). **C** If an external input is applied to the system (see text), the opinion dynamics will in general not converge to a single value but lead to a dispersed set of final opinions, which still reflect the split observed in reality.

# Random walk

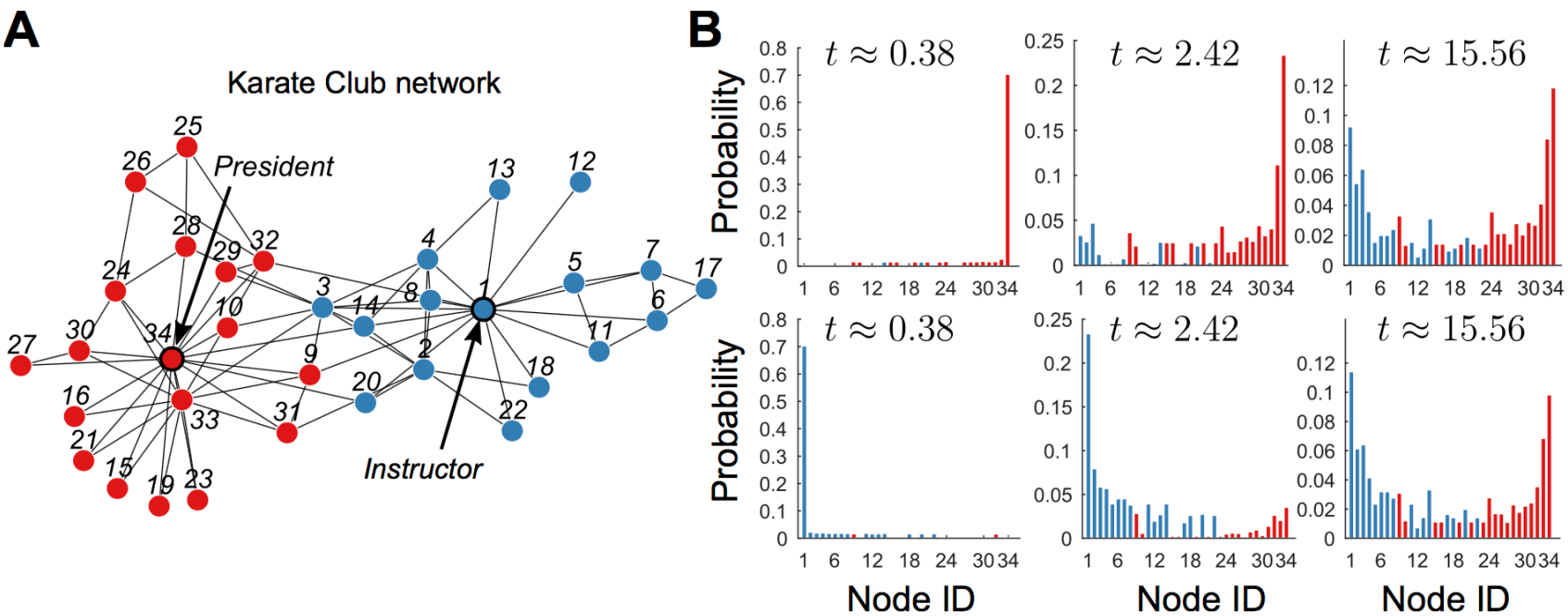



**Figure 13.2 Illustration of the evolution of a random walk dynamics on the Karate Club network.** **A** The Karate Club network with labelled nodes. The factions of the ultimate split observed in reality are indicated by color (grayscale). **B** The evolution of the probability distribution of the random walk over time exemplified at 3 time snapshots from two different initial conditions: the random walker starts at time $t = 0$ at the 'president' node 34 (upper three panels), or at the 'instructor' node 1 (lower three panels). As time evolves, the probability of the walker to be found on the other nodes becomes more spread out on the graph and eventually reaches the stationary distribution $\boldsymbol{\pi} = \mathbf{d}/2w$. Note that for short times, the probability is spread mostly within the corresponding 'factions' (i.e., president for the top panels; instructor for the bottom panels). Beyond the slowest time scale in this dynamics given by $1/\lambda(\mathbf{L}_{\text{RW}}) \approx 1/0.13$, the random walk becomes well mixed and the information spreads across factions (see Section 13.3.1 for more details).

# Time scale separation

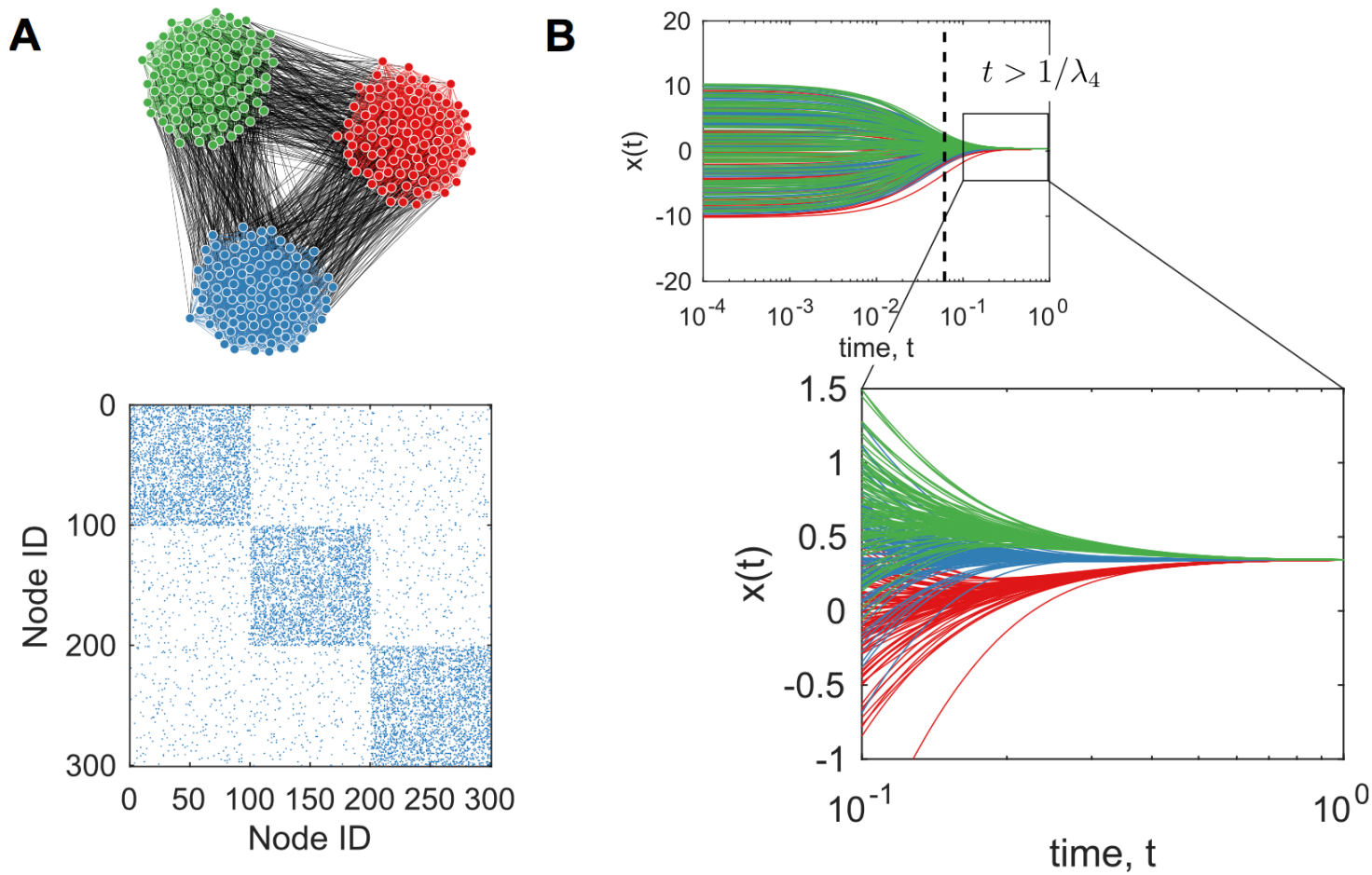



**Figure 13.3 Consensus dynamics on a structured network. A** Visualisation of the network and adjacency matrix of an unweighted structured network with 3 groups of the form (13.8). **B** The consensus dynamics on this network displays time scale separation: after $t \approx 1/\lambda_4 = 0.06$, approximate consensus is reached within each group (groups indicated by color/grayscale) followed by global consensus across the whole network. A similar effect can be observed in the consensus dynamics and random walk on the Karate club network in Figures 13.1 and 13.2

# Diffusion on a network

We can model this process as diffusion on a network where:

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

In general, the dynamics of node $i$ is given by

$$\frac{\mathrm{d}C_i}{\mathrm{d}t} = \sum_{j=1}^{n} pA_{ij}(C_j - C_i).$$

Note that the network may be directed too (e.g., a one-way membrane).

# Diffusion on a network

Examining the system of equations we find that:

$$\frac{\mathrm{d}C_i}{\mathrm{d}t} = \sum_{j=1}^{n} pA_{ij}(C_j - C_i) \;=\; \sum_{j=1}^{n} pA_{ij}C_j - \sum_{j=1}^{n} pA_{ij}C_i,$$

$$= \sum_{j=1}^{n} pA_{ij}C_j - pk_iC_i \;=\; \sum_{j=1}^{n} p(A_{ij} - \delta_{ij}k_i)C_j,$$

where $\delta_{ij}$ is the *Kronecker delta*: $\delta_{ij} = 1$ if $i = j$, and 0 otherwise.

In matrix form these ODEs are given by:

$$\dot{C} = p(A - D)C.$$

Here $D = \mathrm{diag}(k)$, where $k$ is the vector of node degrees.

# The graph Laplacian

The matrix $L = D - A$ is the *Laplacian* of the network.

Rearranging $\dot{C} = p(A - D)C$ we can see why:

$$\dot{C} + pLC = 0$$

In a simple network (no self-loops) the entry-wise definition of the Laplacian is

$$\begin{aligned} L_{ij} &= \begin{cases} -1, & i \neq j \,\&\, (i,j) \in \mathcal{E}, \\ k_i, & i = j, \\ 0, & \text{otherwise.} \end{cases} \\ &= \delta_{ij} k_i - A_{ij}. \end{aligned}$$

# The graph Laplacian

We know the solution to the linear system of equations $\dot{C} = -pLC$ is

$$C(t) = e^{-ptL}C_0.$$

Let $v_1, \ldots, v_n$ be the eigenvectors of $L$ with corresponding eigenvalues $\lambda_1, \ldots, \lambda_n$ such that

$$Lv_i = \lambda_i v_i.$$

We can write the solution $C(t)$ as linear combinations of $v_i$ with time-dependent coefficients:

$$C(t) = \sum_i a_i(t)v_i.$$

# The graph Laplacian

If $C(t) = \sum_i a_i(t) v_i$, the equation $\dot{C} + pLC = 0$ now becomes

$$\sum_i \left( \frac{\mathrm{d}a_i}{\mathrm{d}t} + p\lambda_i a_i \right) v_i = 0.$$

Given that the eigenvectors are orthogonal, we can dot-multiply the equation with any other eigenvector to get:

$$\frac{\mathrm{d}a_i}{\mathrm{d}t} + p\lambda_i a_i = 0,$$

with solution

$$a_i(t) = a_i(0) e^{-p\lambda_i t}.$$

# Properties of the Laplacian

The Laplacian has many interesting and useful properties.

For instance, let $\mathbb{1}$ be the $n \times 1$ vector of ones, then:

$$\begin{aligned} L\mathbb{1} &= (D - A)\mathbb{1}, \\ &= D\mathbb{1} - A\mathbb{1}, \\ &= k - A\mathbb{1}, \end{aligned}$$

where $k$ the vector of degrees, and the $i$-th component of the vector $A\mathbb{1}$ is $\sum_j A_{ij} = k_i$, so

$$L\mathbb{1} = 0.$$

The vector $\mathbb{1}$ is an eigenvector associated with the eigenvalue $\lambda = 0$.

In fact, all eigenvalues of the Laplacian are real and non-negative.

# Properties of the Laplacian

If the network has different connected components we can arrange the rows and columns of $L$ so that

$$L = \begin{bmatrix} \boxed{\phantom{00}} & 0 & \dots \\ 0 & \ddots & \\ \vdots & & \boxed{\phantom{00}} \end{bmatrix},$$

then vectors with the following structure

$$v^T = [\ \underbrace{1, 1, \dots, 1, 1}_{\text{nodes in component}}\ , 0, 0, \dots],$$

are also eigenvectors of $L$ with eigenvalue 0.

# Properties of the Laplacian

Then the total number of zero eigenvalues of $L$ is equal to the number of components.

When $\lambda_2 > 0$ then there is only one component, this eigenvalue is know as the *algebraic connectivity* of the network.

Remember that $a_i(t) = a_i(0)e^{-p\lambda_i t}$; given that $\lambda_i \geq 0$, what happens to $a(t)$ as $t \to \infty$?

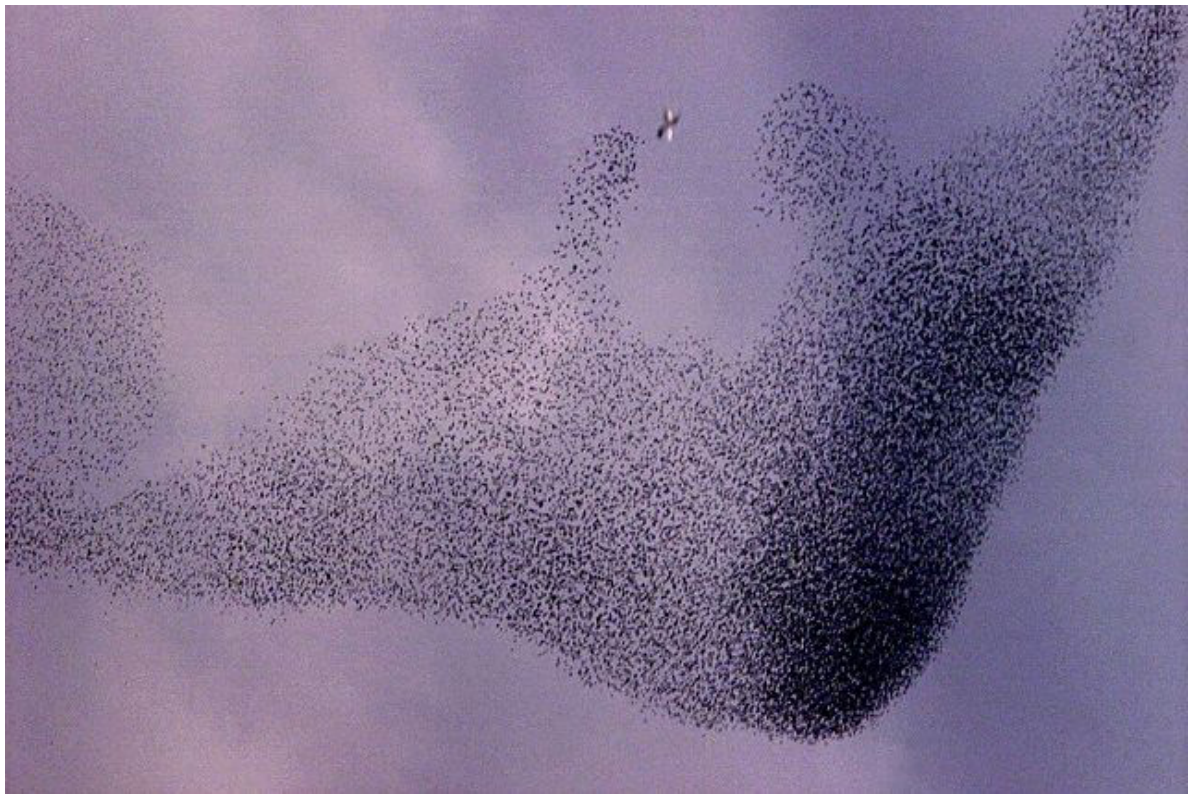
# Consensus dynamics on a network

Now we turn our attention to the problem of consensus.

*Consensus* is to reach an agreement regarding a certain quantity of interest in a system that depends on the state of all agents.

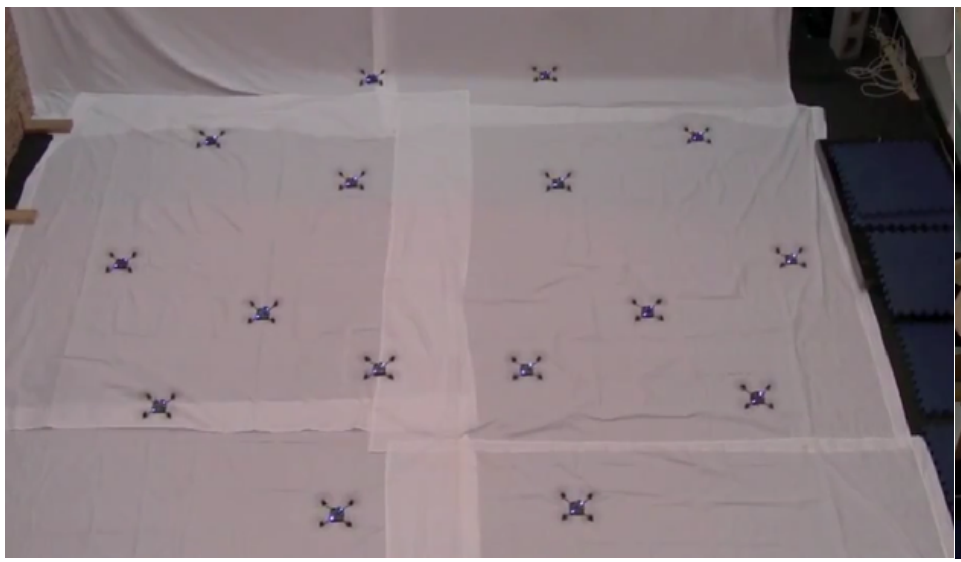
# Consensus dynamics on a network

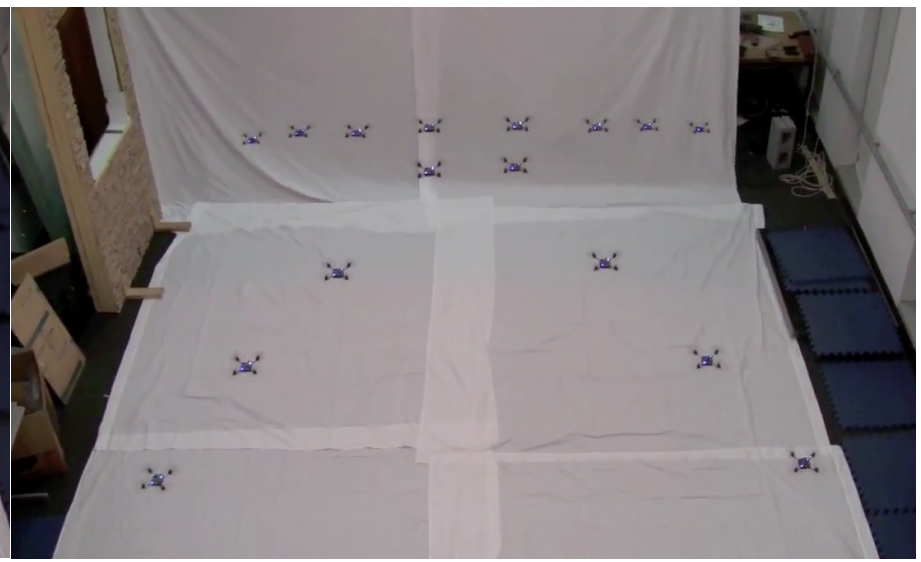
Consensus dynamics are an important part of understanding how flocks work.



# Consensus dynamics on a network

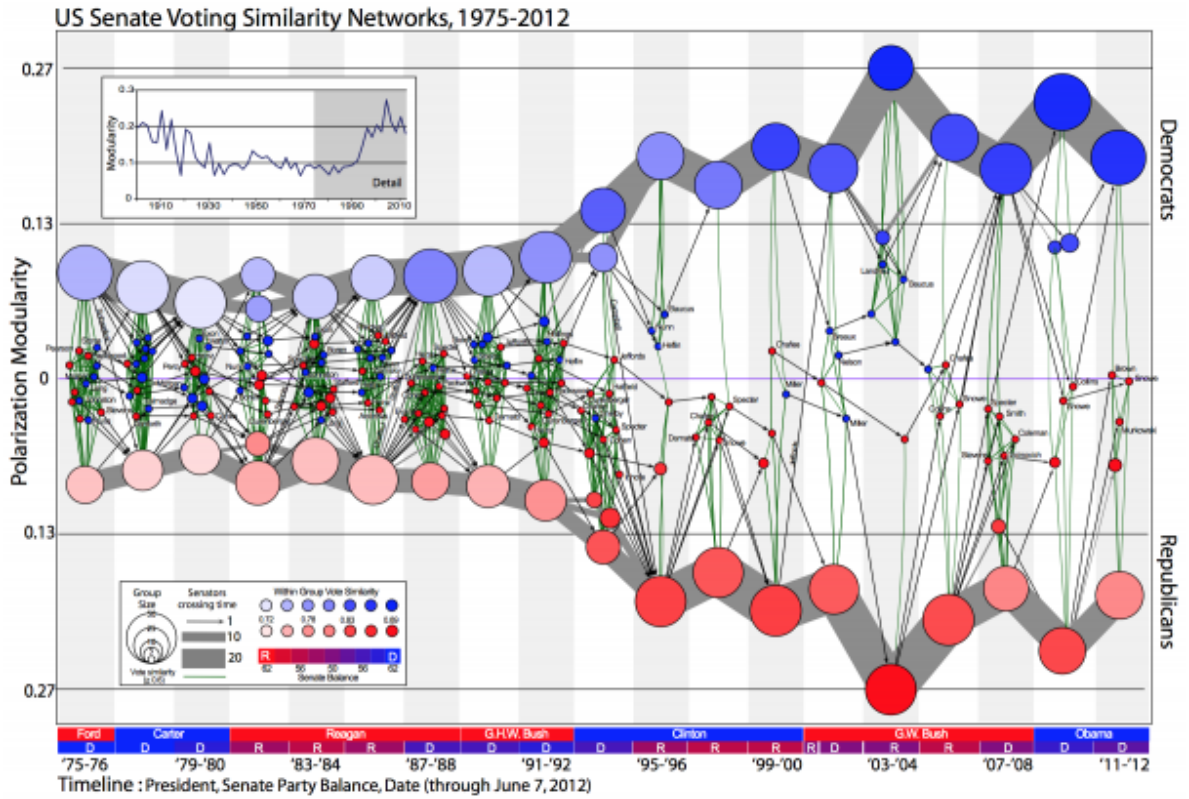
Consensus dynamics are an important part of the design of ensembles of autonomous agents.





# Consensus dynamics on a network

Consensus dynamics are an important to understand the evolution of political opinion.



# Applicability

Other applications include:

Synchronisation of oscillators.

Fast consensus in Small Worlds.

Rendezvous in space.

Distributed Sensor Fusion in Sensor Networks.

Distributed formation control.

etc.

# Mathematical formulation

Suppose we have a system with $N$ interacting agents, each of which has an *opinion* $x_i(t)$ that evolves in time. For example, we can have $x_i \in [0, 1]$.

If agents can share information then their opinion is governed by:

$$\dot{x}_i = \sum_j A_{ij} \left(x_j - x_i\right) + b_i(t),$$

where $b_i(t)$ can be any forcing function.

# Mathematical formulation

When $b_i(t) = 0$ the system becomes:

$$\begin{aligned}\dot{x}_i &= \sum_j A_{ij}\left(x_j - x_i\right), \\ &= \sum_j A_{ij}x_j - k_i x_i, \\ &= \sum_j A_{ij}\left(x_j - \delta_{ij}k_i x_i\right).\end{aligned}$$

In other words, the collective dynamics of the system are *Laplacian*:

$$\dot{x} = -Lx.$$

# Mathematical formulation

We say that consensus is reached whenever

$$x_i(t) = \alpha \ \forall\, i.$$

However, in many networks, this is not always achieved.

# Undirected consensus

A consensus is reached when $\dot{x}_i = 0$ for all $i$, which means that

$$0 = -Lx.$$

Remember that the vector $\mathbb{1}$ is an eigenvector of $L$ and $L\mathbb{1} = 0$.

In an undirected network this means that

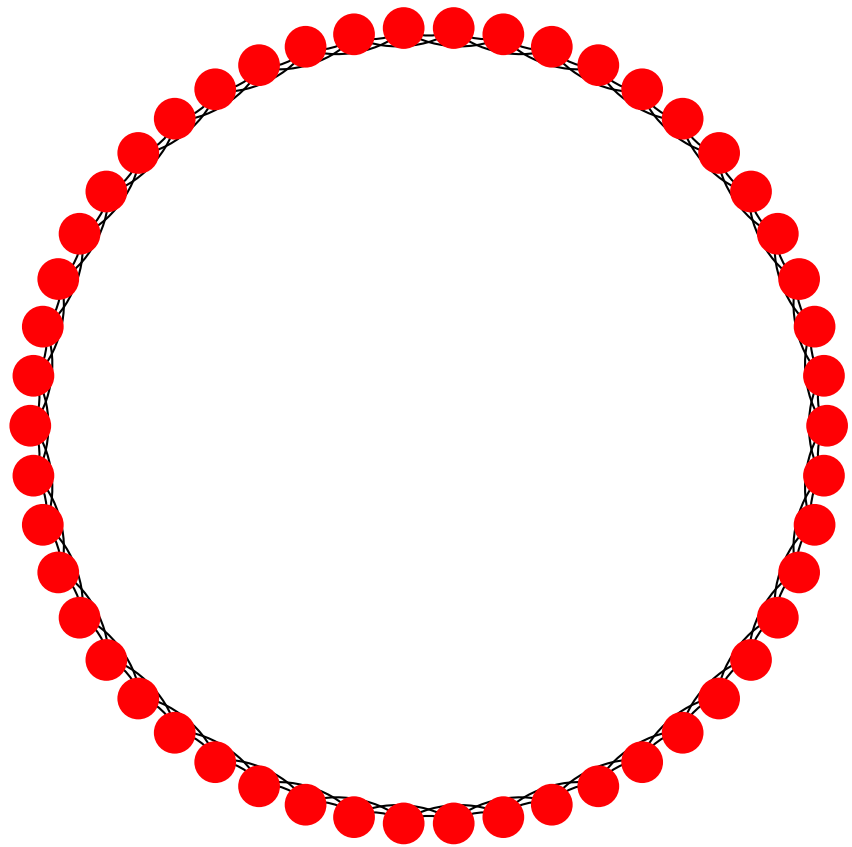
$$\lim_{t\to\infty} x(t) = \lim_{t\to\infty} e^{-tL}x(0) = \alpha\mathbb{1},$$

where

$$\alpha = \frac{1}{N}\sum_i x_i(0).$$

# Examples

Consider a ring lattice with 50 nodes:



All nodes have degree 4.

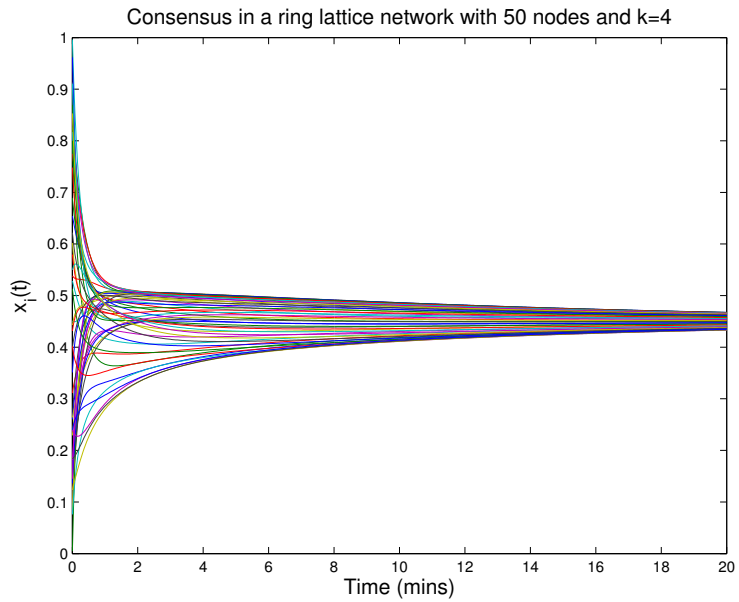
$\lambda_2 = 0.0786$.

# Examples

Consider a ring lattice with 50 nodes:

All nodes have degree 4.

$\lambda_2 = 0.0786$.

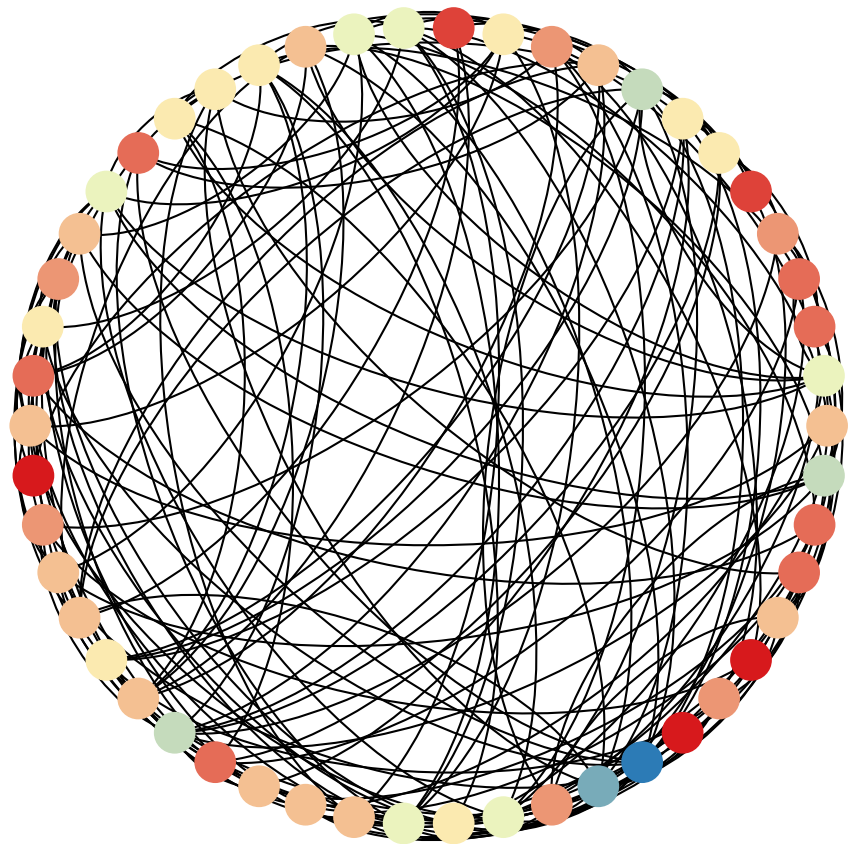


# Examples

An ER graph with 50 nodes :

All nodes have degree on average 4.

$\lambda_2 = 1.3$.

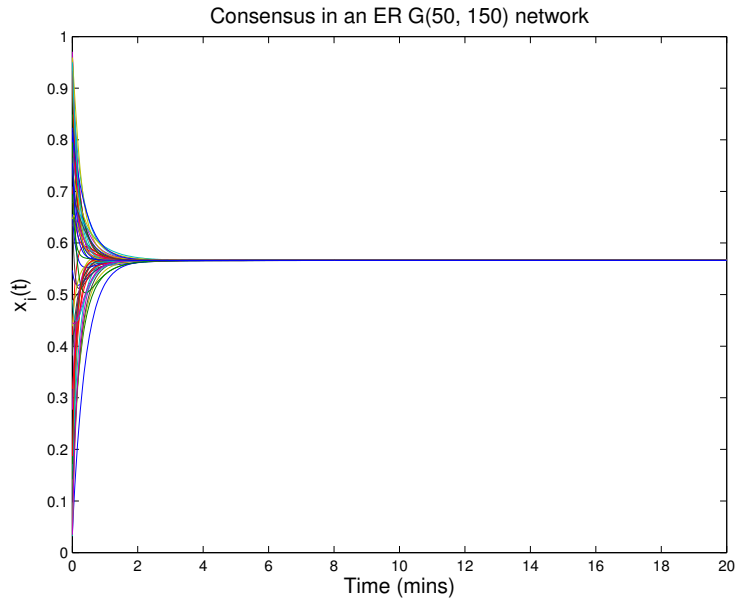


# Examples

An ER graph with 50 nodes :

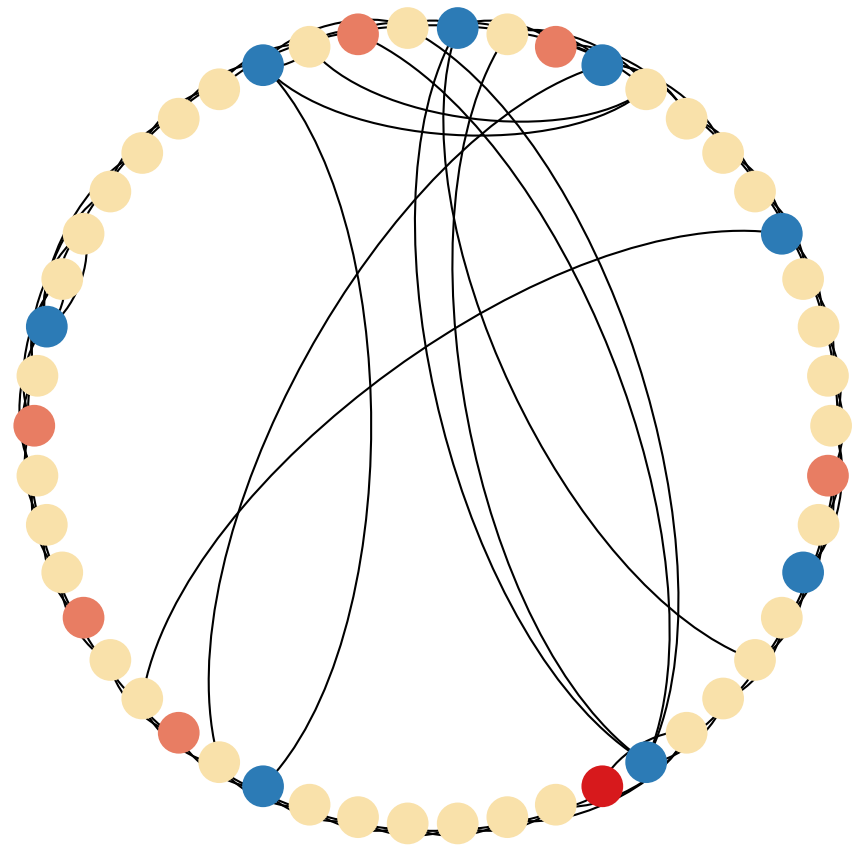
All nodes have degree on average 4.

$\lambda_2 = 1.3$.



# Examples

A SW graph with 50 nodes :



All nodes have degree 4.

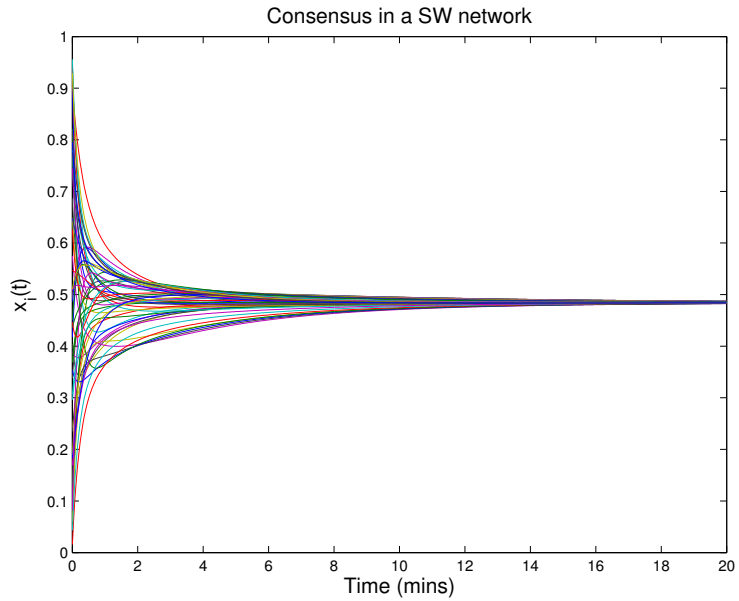
$\lambda_2 = 0.14$.

# Examples

A SW graph with 50 nodes :

All nodes have degree 4.

$\lambda_2 = 0.14$.

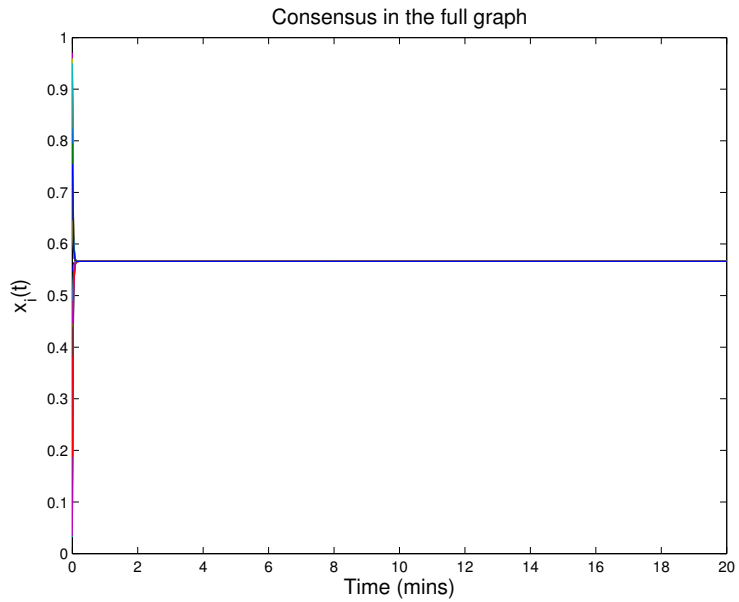


# Examples

A full graph with 50 nodes :

All nodes have degree 49.

$\lambda_2 = 50$.



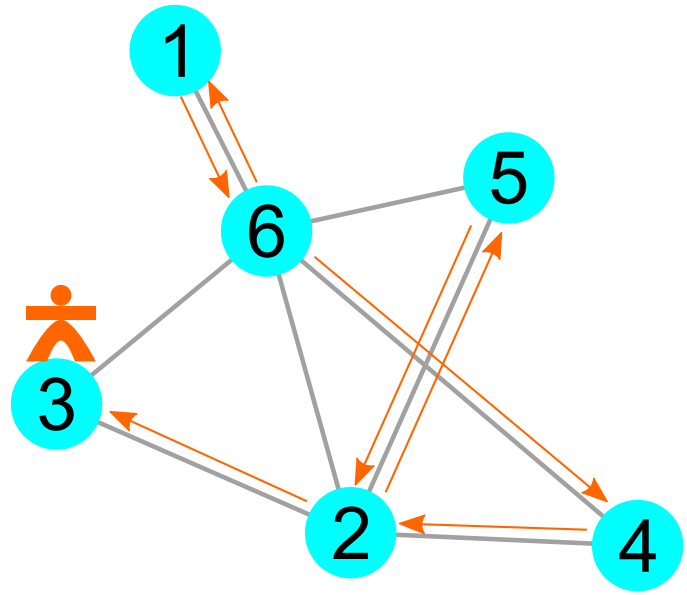
# Random walks

A *random walker* navigates the network, jumping from node to node.

At each node, it chooses its next destination by choosing one neighbour at random.

The way in which the walker chooses where to go next determines the type of random walk.

The simplest way is to choose uniformly at random.



# Random walks

Let $p_i(t)$ be the probability that the walker is at node $i$ at time $t$.

The probability that a node is visited at time $t+1$ is

$$p_i(t+1) = \sum_j \frac{A_{ji}}{k_j} p_j(t).$$

In matrix form this can be written as:

$$p(t+1) = AD^{-1}p(t).$$

If the network is directed then

$$p(t+1) = A^T D_{out}^{-1} p(t).$$

# Normalised Laplacian

If the network is strongly connected and acyclic then the probability converges to a steady state $p \to \pi$ as $t \to \infty$:

$$\pi = AD^{-1}\pi,$$

which means

$$\begin{aligned} 0 &= \pi - AD^{-1}\pi, \\ 0 &= (I - AD^{-1})\pi, \\ 0 &= (D - A)D^{-1}\pi, \\ 0 &= LD^{-1}\pi. \end{aligned}$$

The steady state $\pi$ is an eigenvector of $LD^{-1}$ with eigenvalue 0.

# Normalised Laplacian

The matrix $LD^{-1}$ is called the *normalised Laplacian* of the network.

The vector of ones $\mathbb{1} = [1, 1, \ldots, 1]^T$ is clearly an eigenvector of $LD^{-1}$.

In general, if $v$ is an eigenvector of $L$, then $Dv$ is an eigenvector of $LD^{-1}$.

# Steady state

Suppose the network is undirected and connected and $\pi = AD^{-1}\pi$.

Given that

$$L\mathbb{1} = 0,$$

we have that $\pi = aD\mathbb{1}$, where $a \in \mathbb{R}$ because

$$LD^{-1}\pi = aLD^{-1}D\mathbb{1} = aL\mathbb{1} = 0.$$

This means that $\pi_i \sim k_i$, the steady state probability (occupation) of a node is proportional to its degree.

Choosing $a$ appropriately $\pi_i = \frac{k_i}{2m}$, and $\sum \pi_i = 1$.

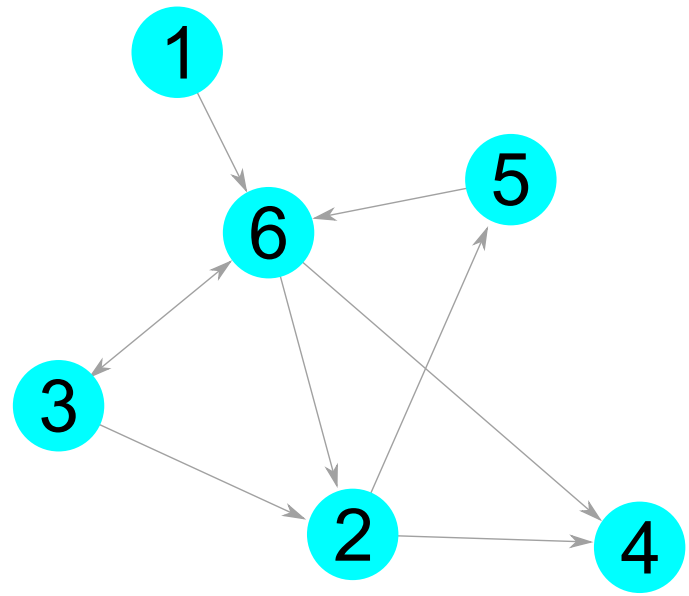
# Random walks on directed networks

On directed networks, there are other factors to consider.

A random walker can get stuck in a *sink*.

*Source* nodes may never be visited.

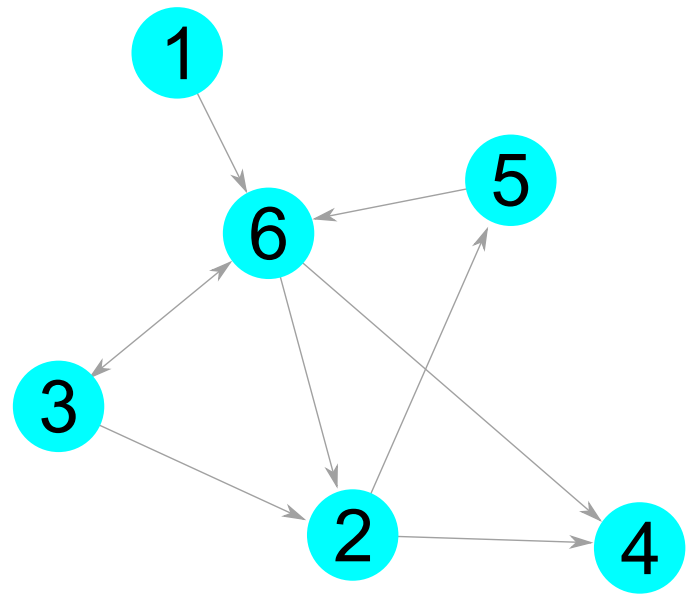
Strong connectivity is necessary for a unique stationary state.



# Teleporting random walk

A walker in a node chooses to follow an edge with probability $\alpha \in [0, 1]$ or *teleport* to any other node with probability $1 - \alpha$.

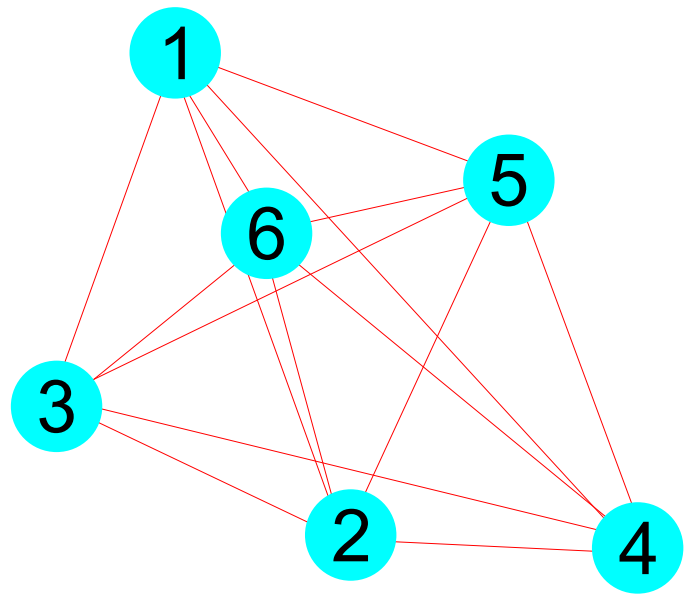
How the teleporting node is chosen can vary; uniformly at random is a common choice.



# Teleporting random walk

A walker in a node chooses to follow an edge with probability $\alpha \in [0, 1]$ or *teleport* to any other node with probability $1 - \alpha$.

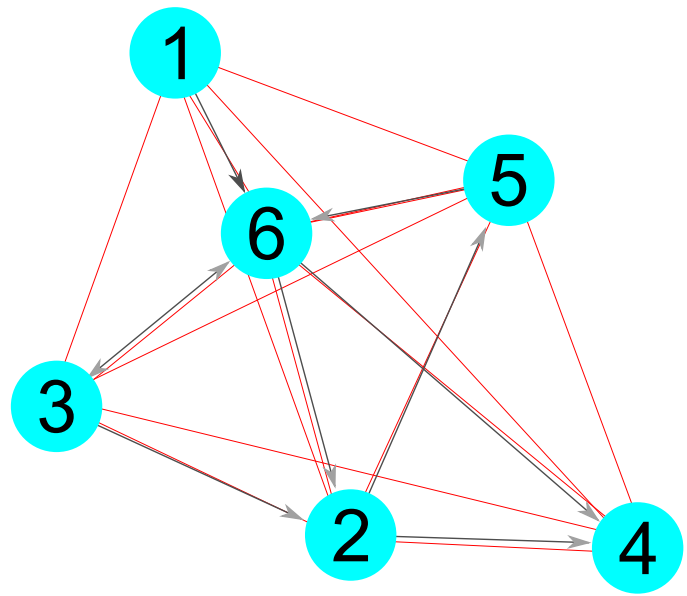
How the teleporting node is chosen can vary; uniformly at random is a common choice.



# Teleporting random walk

A walker in a node chooses to follow an edge with probability $\alpha \in [0, 1]$ or *teleport* to any other node with probability $1 - \alpha$.

How the teleporting node is chosen can vary; uniformly at random is a common choice.



# Teleporting random walk

If $A$ is the adj. matrix, and $k_{out} = A\mathbb{1}$ the vector of out-degrees let

$$B = \overbrace{\alpha D^{-1} A}^{\text{Random walk}} + \left[ \overbrace{(1-\alpha)\frac{1}{N} I}^{\text{Uniform teleportation}} + \overbrace{\alpha \frac{1}{N} \text{diag}(a)}^{\text{Dangling nodes}} \right] \mathbb{1}\mathbb{1}^T.$$

$D = \text{diag}(k_{out})$, if $k_{out}(i) = 0$ for any node we set it to 1.

$a$ is a $N \times N$ rank-1 matrix $a_{ij} = 1 \; \forall \, j$ if $k_{out}(i) = 0$.

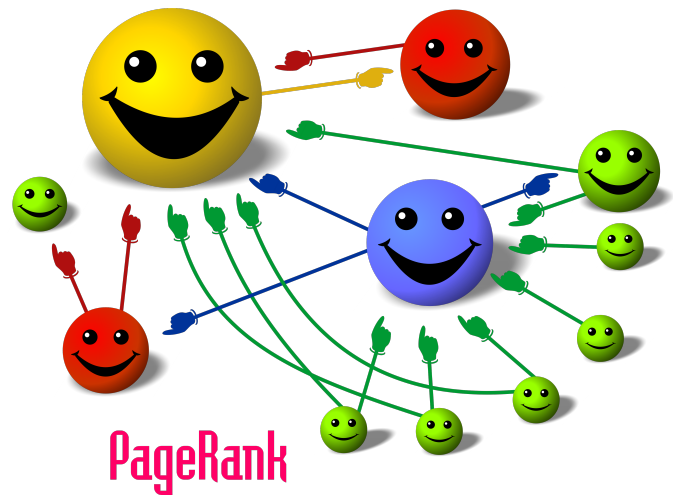
$B^T$ is a stochastic matrix that defines a random walk with teleportation on the network.

# Page Rank

*Page Rank* a popular centrality metric based on teleporting random walks.

Page Rank's premise is that a hyperlink from one page to another is a *vote* for the target page.

It is akin to a democracy where your vote's worth depends on the worth of the votes you receive.



# Page Rank

For initial condition $x(0)$ the vector of probabilities at the next step is

$$x(1) = B^T x(0).$$

The steady-state of the Markov chain is given by the leading left-eigenvector of $B$:

$$\pi = B^T \pi,$$

The leading eigenvalue $\lambda_1 = 1$ (guaranteed by Perron-Frobenius).

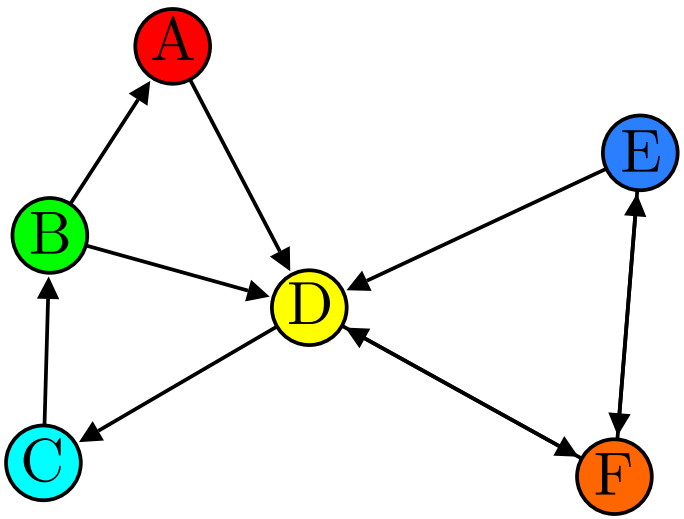
# Page Rank

Each entry $x_i$ is the percentage of time that a random walker would spend on node $i$ in the long-time limit.

If the network is weighted the same procedure still works using the out-strength instead of the out-degree.

# Page Rank

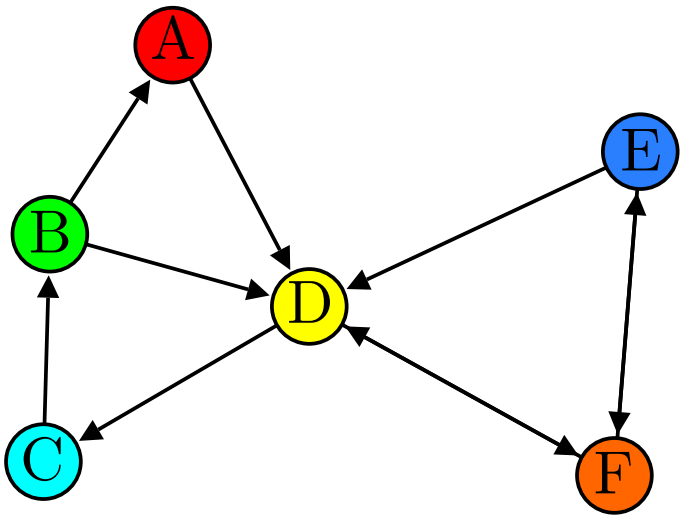
Consider the following example:




$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

# Page Rank

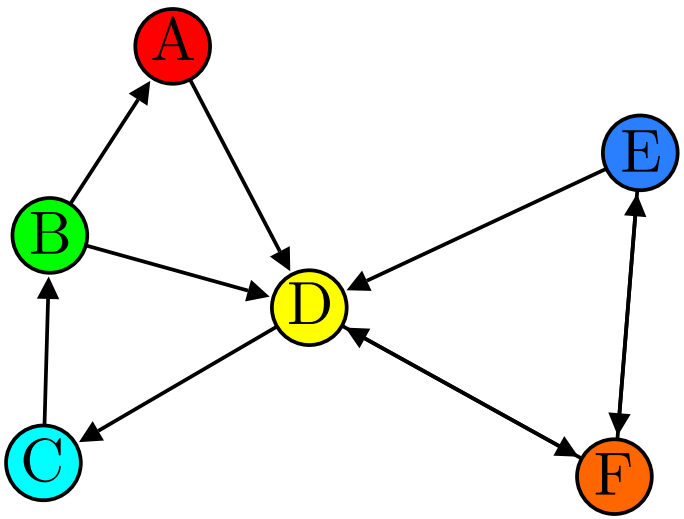
Normalise the adjacency matrix :




$$D^{-1}A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0.5 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0.5 \\ 0 & 0 & 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 0 & 0.5 & 0.5 & 0 \end{bmatrix}$$

# Page Rank

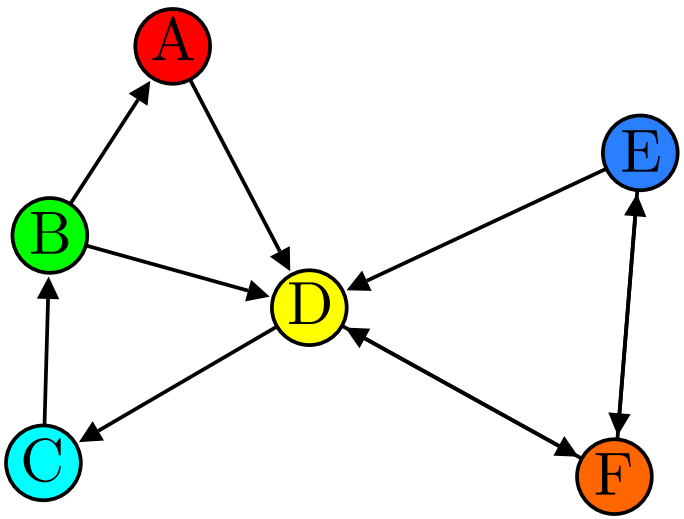
Add teleportation with $\alpha = 0.85$:




$$B = \begin{bmatrix} 0.025 & 0.025 & 0.025 & 0.875 & 0.025 & 0.025 \\ 0.450 & 0.025 & 0.025 & 0.450 & 0.025 & 0.025 \\ 0.025 & 0.875 & 0.025 & 0.025 & 0.025 & 0.025 \\ 0.025 & 0.025 & 0.450 & 0.025 & 0.025 & 0.450 \\ 0.025 & 0.025 & 0.025 & 0.450 & 0.025 & 0.450 \\ 0.025 & 0.025 & 0.025 & 0.450 & 0.450 & 0.025 \end{bmatrix}$$

# Page Rank

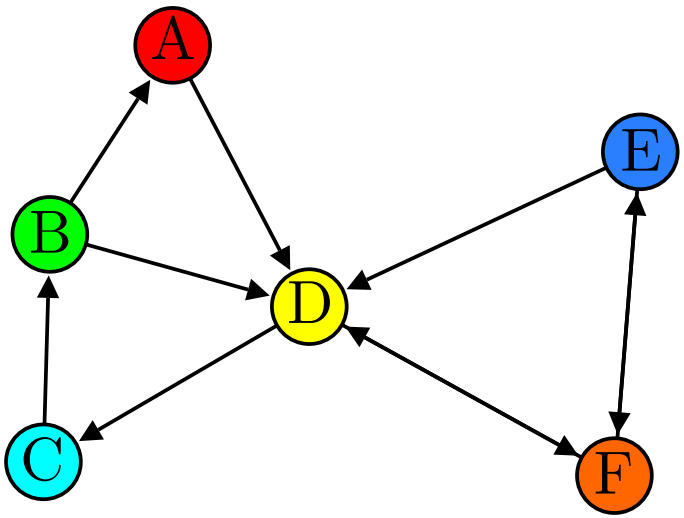
The lead eigenvector of $B^T$ is then:




$$x = \begin{array}{c} A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{bmatrix} 0.2049 \\ 0.3488 \\ 0.3436 \\ 0.6750 \\ 0.2474 \\ 0.4488 \end{bmatrix} \begin{array}{c} 6 \\ 4 \\ 3 \\ 1 \\ 5 \\ 2 \end{array}$$

# Page Rank

The lead eigenvector of $D^1 A$ (i.e., when $\alpha = 1$) is:




$$x = \begin{array}{c} A \\ B \\ C \\ D \\ E \\ F \end{array} \begin{bmatrix} 0.1718 \\ 0.3436 \\ 0.3436 \\ 0.6871 \\ 0.2290 \\ 0.4581 \end{bmatrix} \begin{array}{c} 6 \\ 3 \\ 3 \\ 1 \\ 5 \\ 2 \end{array}$$

Why is it different to when $\alpha = 0.85$?

# Graph partition and community detection

Graph partition and community detection are two different but related problems in network science.

Generally, both deal with the problem of finding groups (or clusters) of nodes such that the number of edges among the groups is minimal.

# Graph partition and community detection

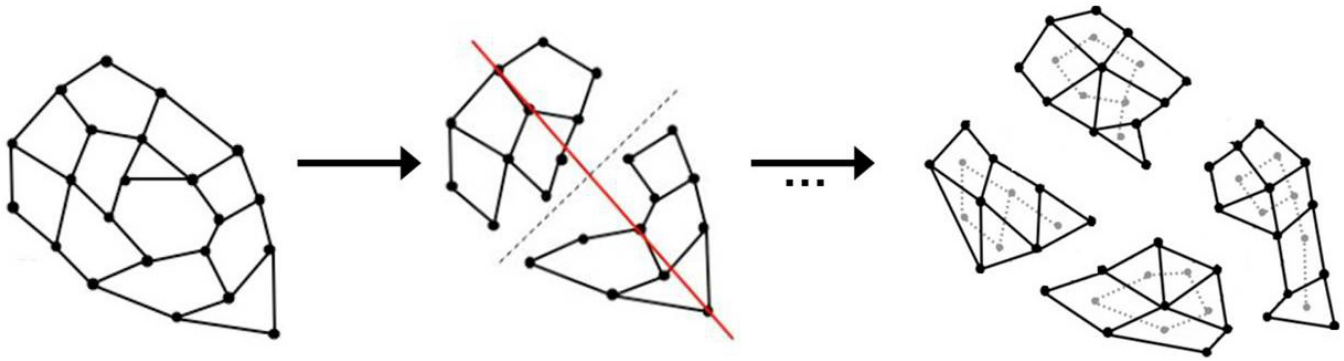
The difference resides in what they are used for:

Graph partition is usually needed when one needs to *do* something on the network.

Community detection is used when one wants to analyse and *understand* the properties of the network.

# Graph partition and community detection

For example, graph partitioning can help in the numerical solution of PDEs on a multicore architecture:

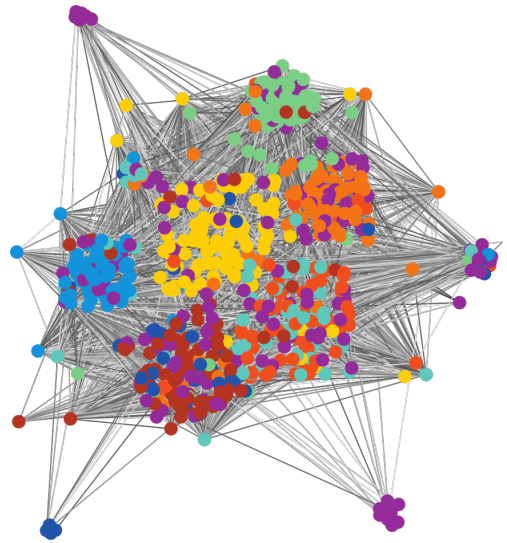


N Gourdain et al 2009 Comput. Sci. Disc. 2 015003.

Usually the number of groups will be external rather than intrinsic constraint, e.g., the number of cores in a computer.

# Graph partition and community detection

With community detection we can understand how students form tightly-knit groups in online networks:



A L Traud et al 2009 Chaos 19, 041104.

The number of communities in determined by the structure of the network (if there are communities, that is).