A cynical primer to network analysis

Jonathan St-Onge

2023-10-22

Table of contents

Preface

> cyn · i · cal (/ sinək(ə)l/): doubtful as to whether something will happen or whether it is worthwhile.

There are many guides to network analysis out there. Those are often *enthusiastic* about the power of networks; they start by (i) noting how networks are all around us, and (ii) how network science emerged as the natural set of (mathematical) tools to study them. As such, they provide a sense of coherence into the field of network science. In truth, when using networks as practical tools, the process can be quite messy. The ambition here is to put in one place the wisdom that results from embracing our doubts about network analysis. What doubts?

- Do networks even exist; we always make choice about how to represent them. There is no end to this branching process, underminig the confidence in our analysis.
- What happens when the communities we discover clash with the metadata of our networks?
- Things can get messy fast, what if we need *hierarchical higher-order temporal multilayerd degree corrected stochastic block models* (HHOTMDCSBM) to solve our problems?
- Many tools look nice when applied with (small-scale) social networks. How do they generalize with protein-protein interactions/brain/twitter networks?

The takeaway is the same than any other model-based approach at the moment; to make your way through the mess, you need to be clear about what you are doing, your assumptions and why you are doing it.

The second benefit of stepping away from the usual narrative is the freedom to focus on deepening our comprehension of network analysis, prioritizing depth over breadth. As Grant Sanderson from 3Blue1Brown puts it, I want you to feel that you could have discovered central ideas to network analysis. As such, we are gonna use all the power of the front-end dev tools and interactive data analysis to make the ideas come alive.

Finally, we won't shy away from making connection to the rest of maths. The big advantage here is that by doing so, we can more easily prepare students to integrate other tools from probability theory and linear algebra to think about how to best integrate model in our toolkit.

My notes are similar in spirit to:

- Revisiting the foundations of network analysis
- Statistics Done Wrong: The woefully complete guide

Do networks even exist?

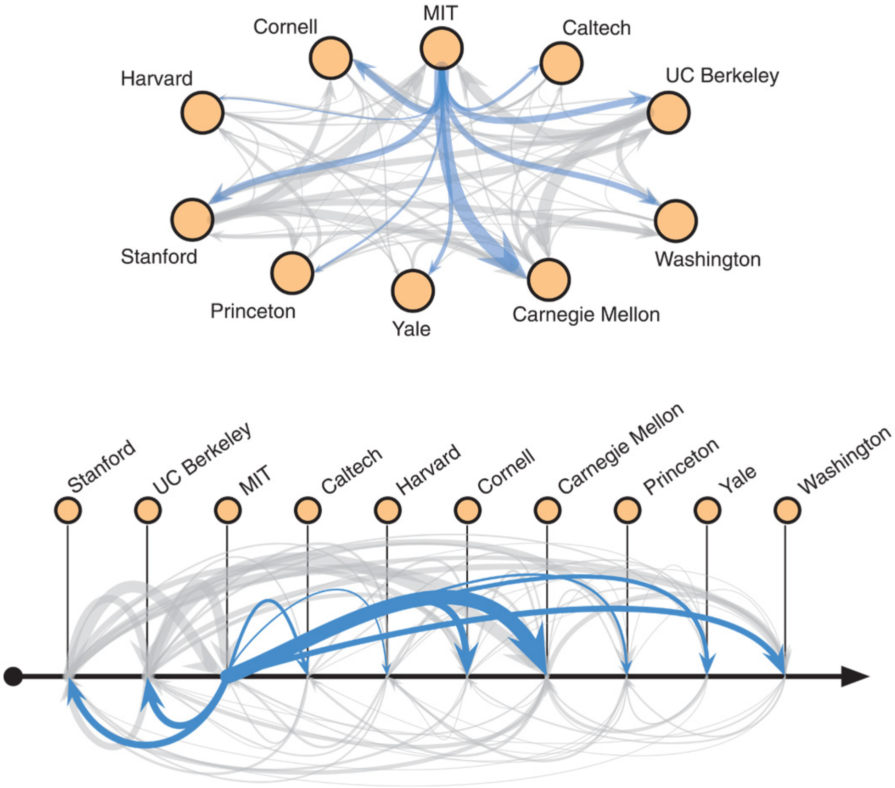


Figure 1: Clauset et al. 2015 Fig. 1

The type of your graph is always an assumption of your making. In the figure above, Clauset and colleagues drawn from theories in sociology to justify the directed edges, aka when institution A hires a PhD from institution B, this is a signal of endorsment from A to B.

We are talking about institutions here, not people. Thus, the authors are willing to imbue “institution” with intentionality, endorsing each other. Could we deconstruct institutions at department level, looking to explain the whole in part by hiring committes?!

Or perhaps at individual-level, with the set of endorsments on a hiring committe sums up in a decision of endorsing a candidate?

Uncynical guides:

These books/notes are uncynical in that they start by (i) noting how networks are all around us, and (ii) how network science emerge as the natural set of (mathematical) tools to study them.

- Menczer, Fortunato, and Davis (2020) (github): Great book with accompanying code to do the basics of network analysis in NetworkX. It feels refreshing to have examples drawn
- Newman (2018): Mandatory reference (no books get 19K citations without becoming some kind of obligatory reference)
- Network Analysis and Modeling CSCI 5352, Fall 2022: Aaron Clauset's notes on how to do networks right.
- Kolaczyk (2009)

Networks are cool, actually

- Creativity:
  - Curious Minds: The Power of Connection
- Neurolinguistics
  - Using network science to map what Montréal bilinguals talk about across languages and communicative context
- Faculty hiring market:
  - Systematic inequality and hierarchy in faculty hiring networks
- Hierarchy:
  - Quantifying hierarchy and dynamics in US faculty hiring and retention
- Survey:
  - Latent Network Models to Account for Noisy, Multiply-Reported Social Network Data.
- Network talks:
  - Larremore's 2023 Erdos-Renyi Prize Lecture
- Citation analysis:
  - Choosing to grow a graph: Modeling network formation as discrete choice

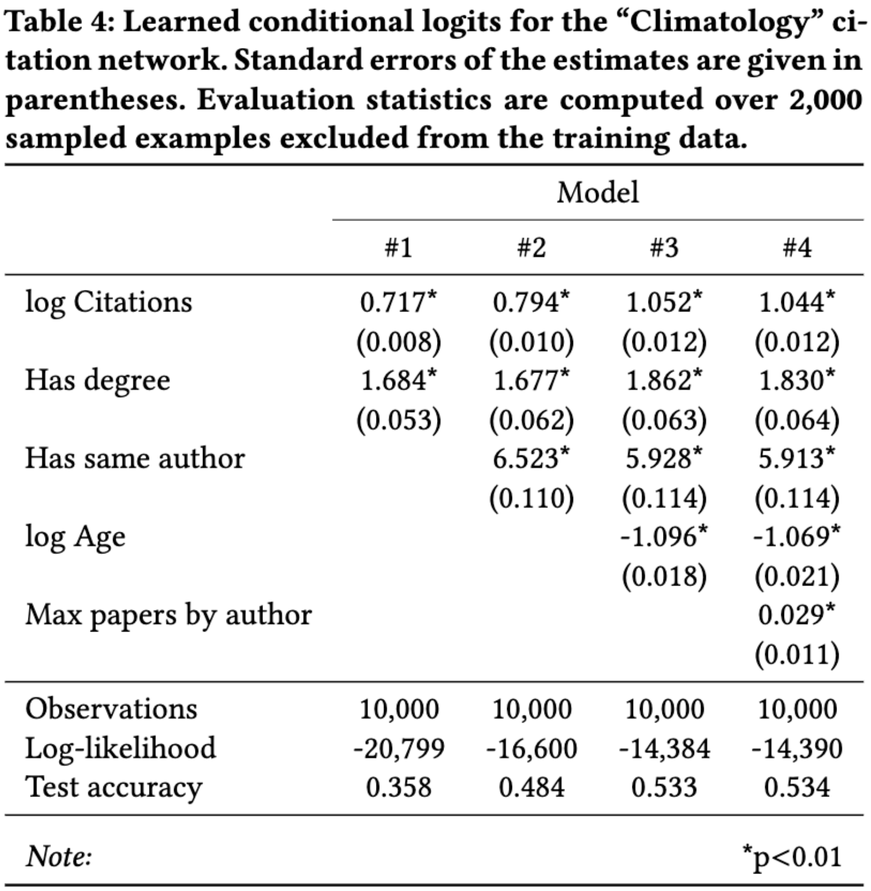


Table 4: Learned conditional logits for the “Climatology” citation network. Standard errors of the estimates are given in parentheses. Evaluation statistics are computed over 2,000 sampled examples excluded from the training data.

	Model			
	#1	#2	#3	#4
log Citations	0.717*	0.794*	1.052*	1.044*
	(0.008)	(0.010)	(0.012)	(0.012)
Has degree	1.684*	1.677*	1.862*	1.830*
	(0.053)	(0.062)	(0.063)	(0.064)
Has same author		6.523*	5.928*	5.913*
		(0.110)	(0.114)	(0.114)
log Age			-1.096*	-1.069*
			(0.018)	(0.021)
Max papers by author				0.029*
				(0.011)
Observations	10,000	10,000	10,000	10,000
Log-likelihood	-20,799	-16,600	-14,384	-14,390
Test accuracy	0.358	0.484	0.533	0.534
Note:				*p<0.01

Figure 2: **?(caption)**

1 I/O standards

> All course data are alike; each dataset of our own is different in its own way.

Reading data that we care about can be frustrating because it is never in the tidy format we want it to be. At the same time, this make sense. If it were, it would probably already be analyzed by someone.

1.1 Data Format

1.2 NetworkX

A good starting point is networkx documentation on reading and writing graphs.

1.3 Igraph

A good starting point is igraph documentation on reading and writing graphs.

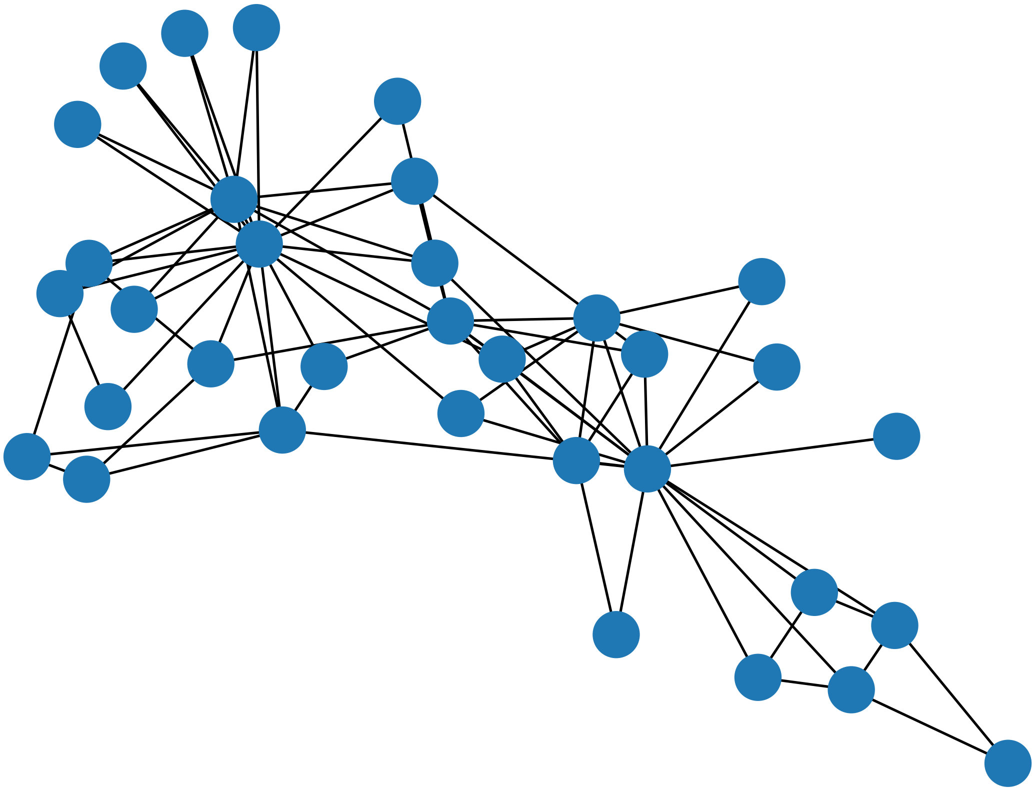
But really, let's just try it out. We will look at all the different data formats of the classical `Zachary karate club` dataset that we can find out there.

```
from pathlib import Path
import networkx as nx
import numpy as np
import json

root_dir = Path()
data_dir = root_dir / 'data'
```

1.3.1 Format 1

ℹ Format 1

```
# see https://networkx.org/documentation/stable/_modules/networkx/readwrite/gml.html#read_
nx.draw(nx.read_gml(data_dir / 'zachary' / 'karate1.txt', label='id'))
```



1.3.2 Format 2

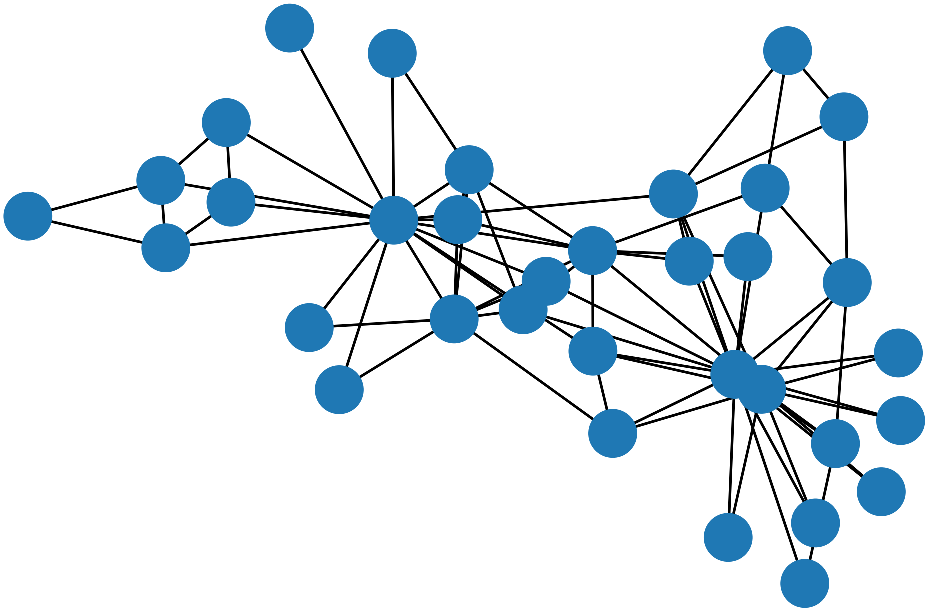
ℹ Format 2

```
DL
N=34 NM=2
FORMAT = FULLMATRIX DIAGONAL PRESENT
LEVEL LABELS:
ZACHE
ZACHC
DATA:
 0 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 0
 1 0 1 1 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0 0 0 0 1 0 0 0
 1 1 0 1 0 0 0 1 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 1 0
 1 1 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1
 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 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 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1
 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1
 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 0 1 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1
 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1
 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 1
 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1
 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 1 1
 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0 1 0 1 0 1 1 0 0 0 0 0 1 1 1 0 1
 0 0 0 0 0 0 0 0 1 1 0 0 0 1 1 1 0 0 1 1 1 0 1 1 0 0 1 1 1 1 1 1 1 0
 0 4 5 3 3 3 3 2 2 0 2 3 1 3 0 0 0 2 0 2 0 2 0 0 0 0 0 0 0 0 0 2 0 0
 4 0 6 3 0 0 0 4 0 0 0 0 0 5 0 0 0 1 0 2 0 2 0 0 0 0 0 0 0 0 2 0 0 0
 5 6 0 3 0 0 0 4 5 1 0 0 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 0 0 0 2 0
 3 3 3 0 0 0 0 3 0 0 0 0 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 3 0 0 0 0 0 2 0 0 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 3 0 0 0 0 0 5 0 0 0 3 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 3 0 0 0 2 5 0 0 0 0 0 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 2 4 4 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 2 0 5 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 0 3 4
 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2
 2 0 0 0 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

1.4 python

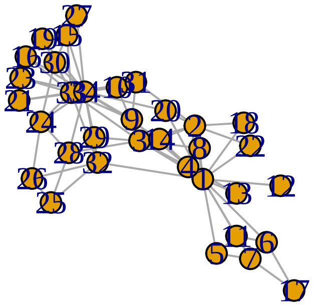
```
def read_ucinet():
  """
    DATASET ZACHARY

    DESCRIPTION Two 34×34 matrices.

        ZACHE symmetric, binary.
        ZACHC symmetric, valued.

    ref: http://vlado.fmf.uni-lj.si/pub/networks/data/ucinet/ucidata.htm#zachary
  """
  out = []
  with open(data_dir / 'zachary' / "karate2.txt", "r") as f:
    for i, line in enumerate(f.readlines()):
      # index starts at zero
      if i >= 7 and i <= 40:
        out.append([int(x) for x in line.strip().split()])
  return nx.from_numpy_array(np.array(out))
```

```
nx.draw(read_ucinet())
```



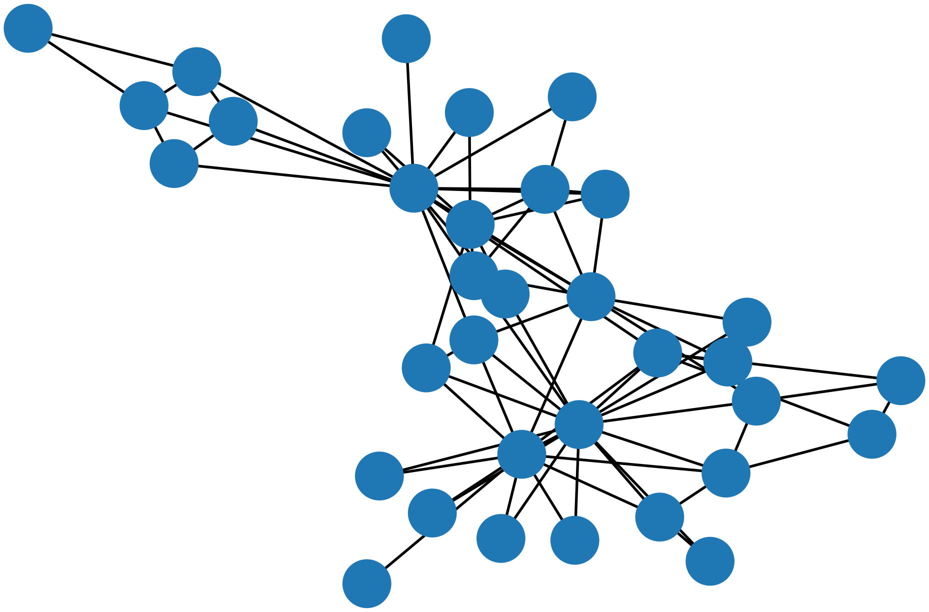
1.5 R

```
# see https://gephi.org/users/supported-graph-formats/ucinet-dl-format/
adjm <- multiplex::read.dl(here::here('data', 'zachary', 'karate2.txt'))
plot(igraph::graph_from_adjacency_matrix(adjm[,,"ZACHE"], mode="undirected"))
```



1.5.1 Format 3

```
nx.draw(nx.read_edgelist(data_dir / 'zachary' / 'karate4.txt'))
```



Tip

Sometimes, you might find yourself in a situation where `nx.read_*` doesn't work out. You'll need to write some code (you're allowed to use `gpt4`, I didn't know about `xml.etree.ElementTree` python module before this week. This is great.)

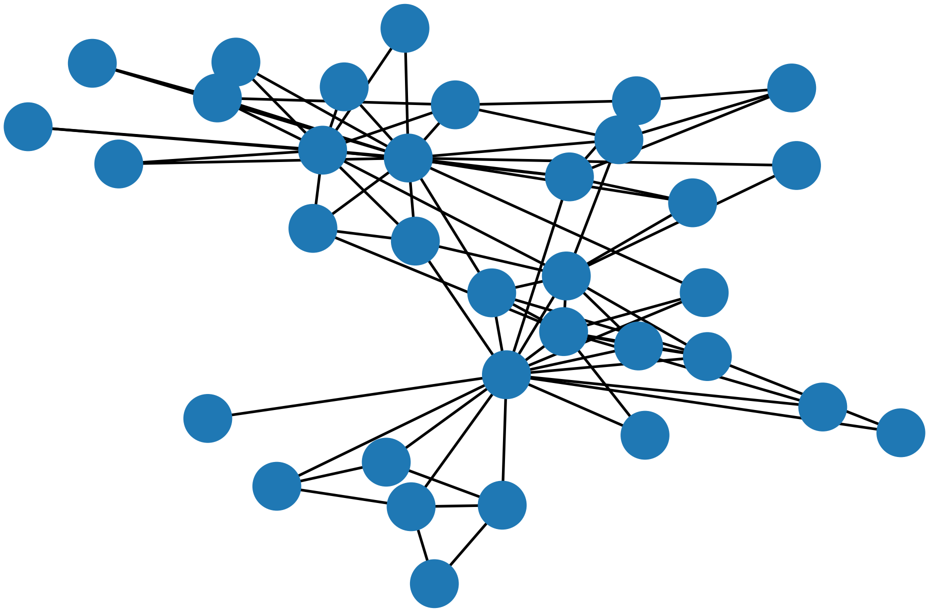
1.5.2 Format 4

ℹ Format 4

1.6 Python

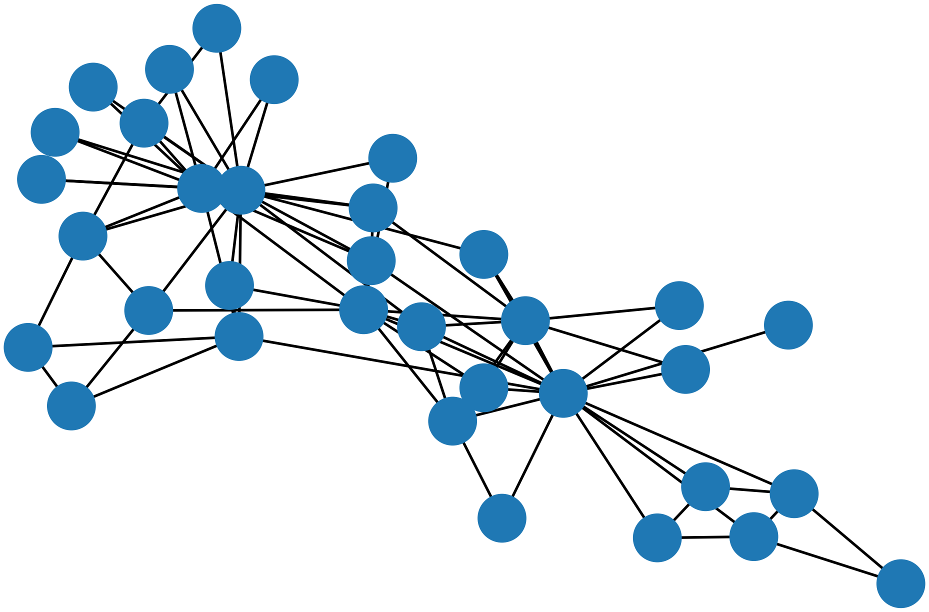
```
import xml.etree.ElementTree as ET

def read_xml():
  with open(data_dir / 'zachary' / "karate3A.txt", "r") as file:
      xml_data = file.read()

  root = ET.fromstring(xml_data)

  G = nx.Graph()

  for node in root.findall(".//node"):
      G.add_node(node.attrib['id'])

  for link in root.findall(".//link"):
      source = link.attrib['source']
      target = link.attrib['target']
      value = float(link.attrib['value'])  # assuming you want to use this as a weight
      G.add_edge(source, target, weight=value)

  return G
```

```
nx.draw(read_xml())
```



Or does it? If `graphml` is well formatted, that should just work:

```
nx.draw(nx.read_graphml(data_dir / 'zachary' / 'karate3B.txt'))
```



1.7 R

1.7.1 Format 5

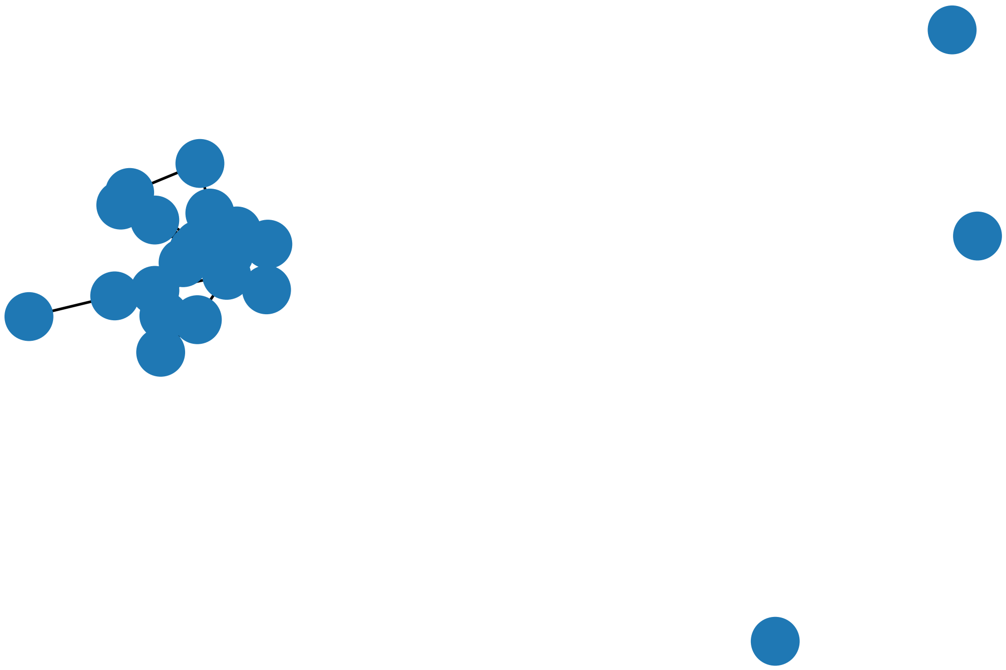
💡 Tip

In my dataviz work, i end up coming back and forth alot between `javascript` and `python`. This means you need to get used to the `.json` format, as it is the *defacto* format for the web (APIs mostly return `.json`, `d3.js` also make heavy used of `.json`). Here is an example from my work.

ℹ format 5

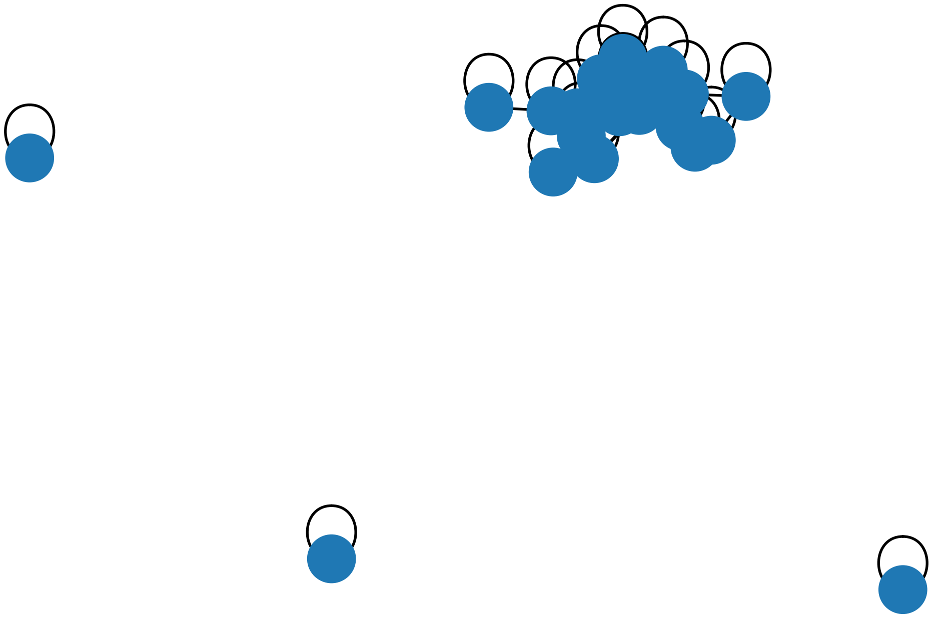
Then you need to write some code. Here is one way to do it

```
with open(data_dir / "csys_collab" / "csys_collab.json") as f:
  dat=json.loads(f.read())

g = nx.Graph() # undirected by default

# https://networkx.org/documentation/stable/reference/classes/generated/networkx.Graph.add
nodelist = [(n['id'], dict(group=n['group'], label=n['label'])) for n in dat['nodes']]
g.add_nodes_from(nodelist)

# https://networkx.org/documentation/stable/reference/classes/generated/networkx.Graph.add
edgelist = [(e['source'], e['target'], e['value']) for e in dat['links']
            if e['source'] != e['target']]

g.add_weighted_edges_from(edgelist)

nx.draw(g)
```



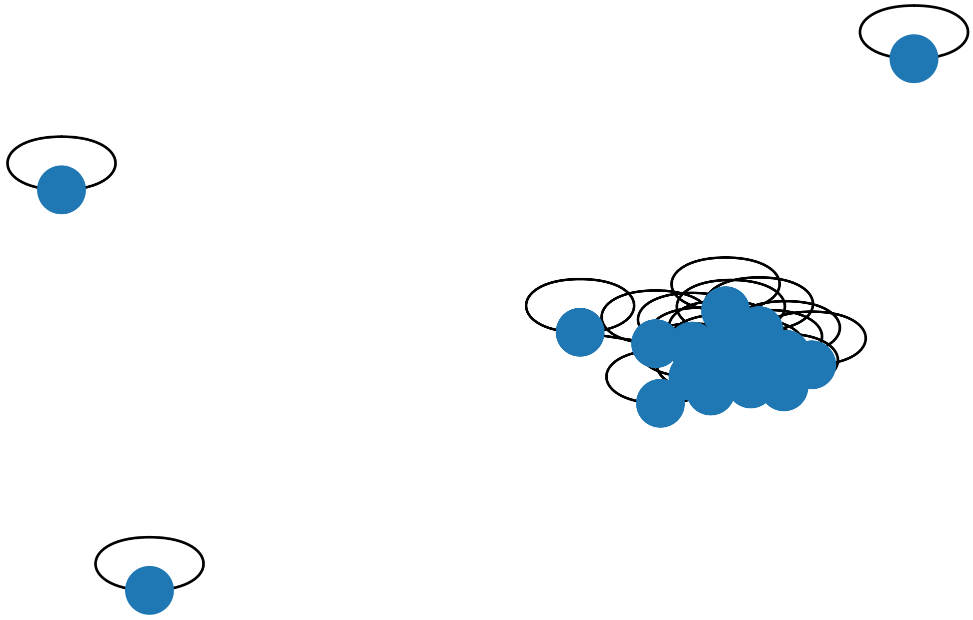
`Networkx` is very friendly to `numpy`, use it to your advantage:

```
nx.draw(nx.Graph(np.loadtxt(data_dir / 'csys_collab' / 'csys_collab.txt')))
```



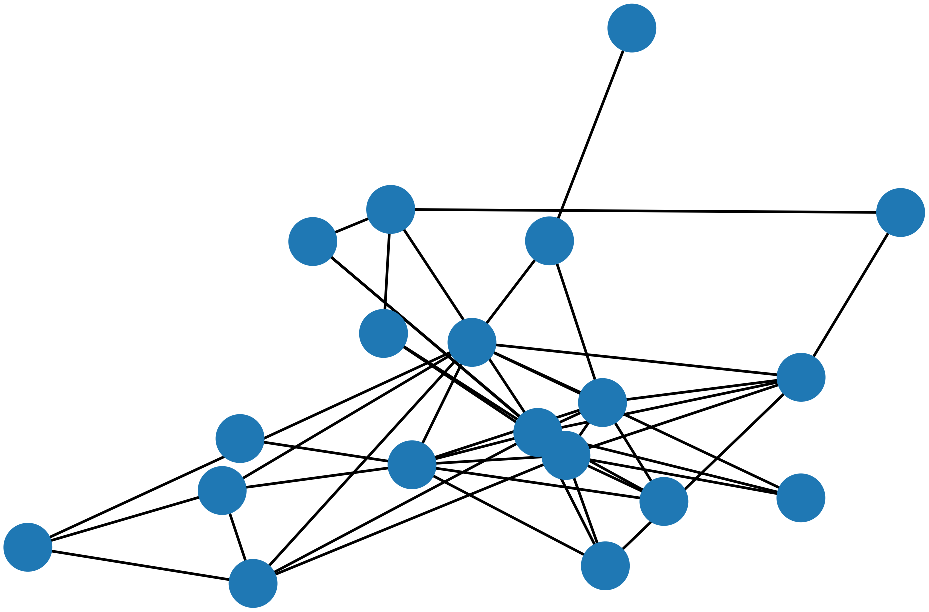
You can read directly from edges, and nodes are gonna be created on the fly:

```
nx.draw(nx.read_weighted_edgelist(data_dir / 'csys_collab' / 'csys_collab.edgelist'))
```



But be mindful that if some nodes that you care about are disconnected, they won't show up in the graph if you just read the edgelist:

```
nx.draw(nx.read_weighted_edgelist(data_dir / 'csys_collab' / 'csys_collab_noselfloop.edgel
```



1.8 Format 6

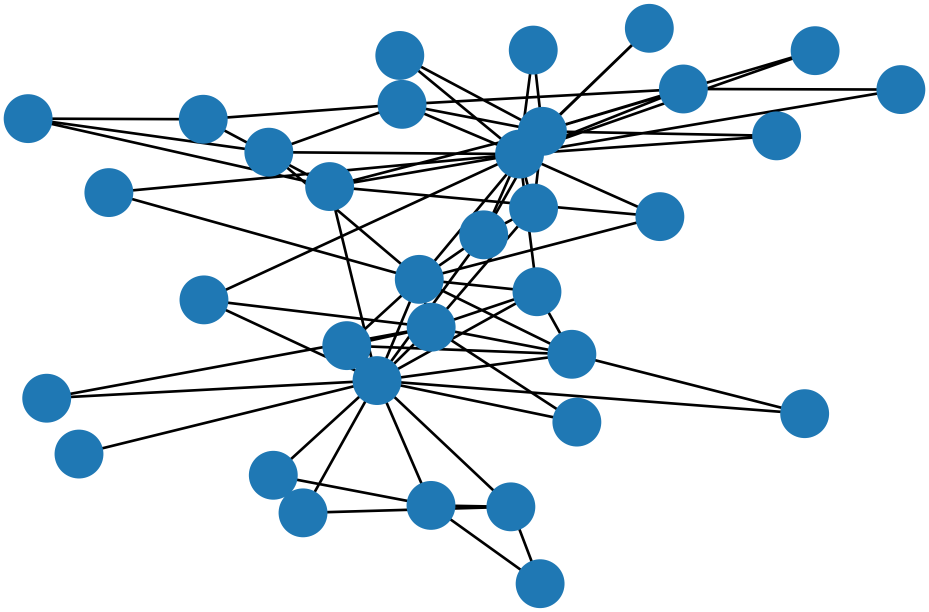
> Tip
>
> Sometimes, you won’t be able to inspect the data. It happens

1.8.0.1 Bonus

Zachary’s dataset comes out of the box in many standard network packages.

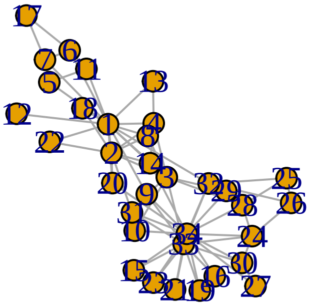
1.9 python

```
nx.draw(nx.karate_club_graph())
```



1.10 R

```
plot(igraph::make_graph('Zachary'))
```



1.11 But, why?

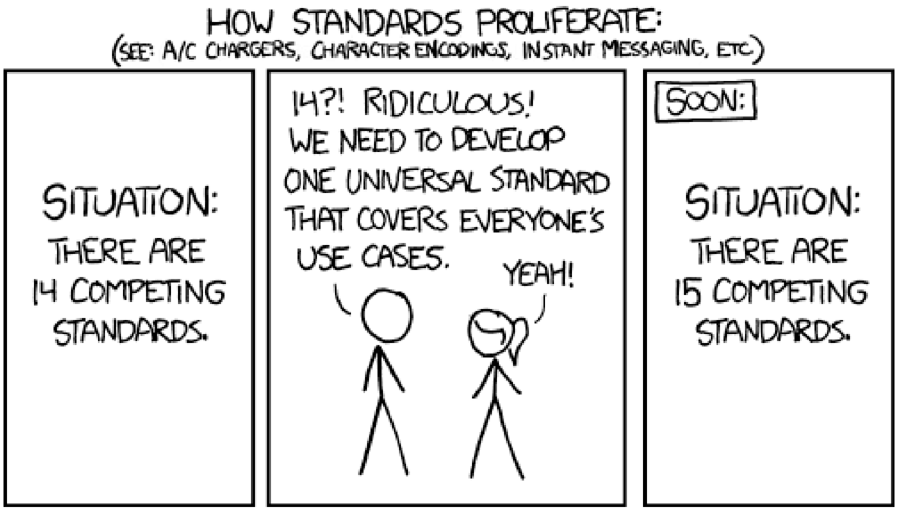

HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC)
SITUATION: THERE ARE 14 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.

Figure 1.1: I/O standards (xkcd - Randall Munroe)

Three reasons come to my mind for all these standards:

1. Scientific culture

2. Inertia
3. Trade-offs.

By scientific culture, I rally mean different set of expectations about what scientists should know when it comes to programming. For instance, the `dl-format` was written for the `Ucinet` GUI by sociologists when they weren't execting sociologists to be both coders and scientists at the same time. Like `SAS` and `STATA`, the idea is that software should be done by professionals who are paid through licensing to develop the product, not a bunch of amateurs.

When you take time to understand the trade-offs, i find understanding I/O to be surprisingly insightful in why some people store data the way they do. Think about storing

1.12 Exercises

1.12.0.1

read `karate5` data.

1.12.1 Multilayer network of Zachary Karate Club Club (ZKCC)

1.12.1.1 Temporal network

- Process on network
- Process of network

1.12.2 Higher-order networks

What if we have edges beyond pairwise interactions?

1.12.3 Graphtool

Installation is the main challenge.

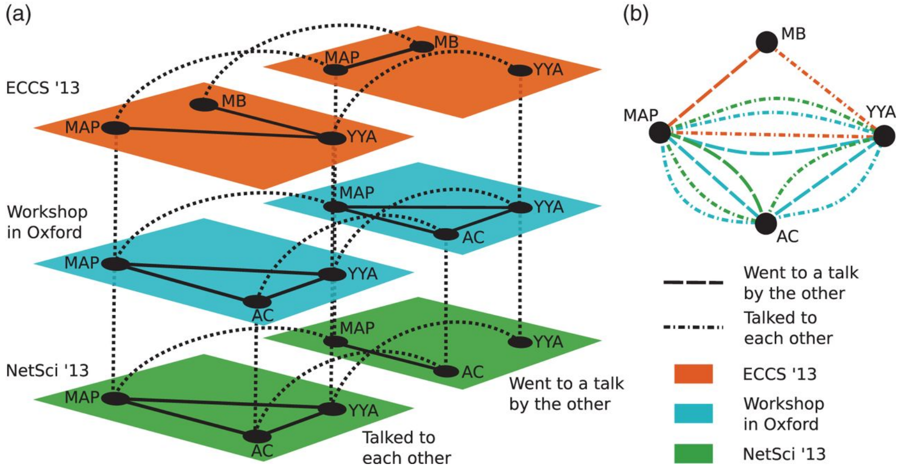


Figure 1.2: Visualization of the Zachary Karate Club Club (ZKCC) network as a multilayer network. Nodes (i.e. elements of V) in the network are the four network scientists who have held the coveted karate trophy for a period of time and have been awarded the associated membership in the ZKCC [142]. The current members of the ZKCC are Cris Moore (CM), Mason A. Porter (MAP), Yong-Yeol Ahn (YYA) and Marián Boguñá (MB). In the figure, Aaron Clauset (AC) is standing in for CM, as the former awarded the karate trophy to MAP at NetSci'13 on behalf of the latter (who did not attend any of the conferences in the figure).

2 Describing networks, then what?

How do we describe networks in meaningful ways?

```
Q: Give me your best centrality measure?
A:
Q: There is no better centrality measure??
A:
```

Seriously, where do we start when we want to understand our networks of interests?

In many ways, network analysis is similar to statistical analysis (sorry). It is statistics.

Usually, in a typical class, we first introduce the language of networks (nodes, edges, directed, weighted, etc.), then network properties. The problem is that we need to avoid what i call the "pitfall of itemizing" (ok, this is a GPT4's suggestion). Lets try different paths

2.1 itemizing

Here this is a list, as presented by Menczer, Fortunato, and Davis (2020) (I indicate with the properties that are used to measure centrality):

1. density/sparsity
2. degree
   i) in/out-degree
   ii) weighted; in/out-strength
3. assortativity/homophily
   i) degree assortativity
   ii) disassortative/core-periphery
4. paths
   i) shortest path A. breadth-first search A. depth-first search
   ii) diameter
   iii) average path length

5. components
    i) connected components
    ii) giant component
    iii) weakly/strongly connected components
6. clustering coefficient/triadic closures
7. closeness
8. k-core decomposition

2.2 catego

scales

- local: about a single item (node, edge)
- meso: about a group of items
- global: about the whole

types

- connectivity: related to (directly or not) to the number of contacts of "things"
- position: related to position (relative, absolute)
- motif: countrs or frequency of patterns of connection

Misc - hubs

p.s. we can turn local into global by using summaries of the distribution of local properties.

2.3 algebraic

- density

$$d = L/L_{\text{max}} = \frac{L}{\frac{N(N-1)}{2}} = \frac{2L}{N(N-1)}$$

where $L_{\text{max}} = \binom{N}{2} = N(N-1)/2$ for undirected networks. A easy way to remember that is by seeing that $2L$ is really counting the number of edges

> ℹ Note
>
> What is a dense network? Why do we care? It often help to think in the extremes, like if $d = 1$ it means that everybody is connected to everybody. If i tell you that coauthorship as much as high-school friendship networks, what does this tells you.

2.4 $\binom{N}{2}$

A little extravangaza about $\binom{N}{2}$, for those who never took discrete math. Get used to $\binom{N}{2}$, but don't reify it. $\binom{N}{2}$ said N *choose* 2 pops u eveyrtime we need to count the total number of ways *unordered* sets can happen...
$\binom{N}{2}$ works well because we are in a pairwise land (a bit like flatland? Am i right).

- degree
- paths
  - shortest path (betweenness)

$$b_i = \sum_{h \neq j \neq i} \frac{\sigma_{hj}(i)}{\sigma_{hj}}$$

where σ_{hj} is the total number of messages from h to j and while $\sigma_{hj}(i)$ are messages from h to j that go through node i.

Tip

In `networkX`, they give the possibility to include the endpoints in the shortest path counts. Why?
Another important calculation when it comes to this kind of approaches is to know how it'll behave when you scale up your system. What is the maximum number of ways shortest paths could go through relevant nodes i. In the book, they write: $\binom{(N-1)}{2} = \frac{(N-1)(N-2)}{2}$. Do you get why?
Final easter egg, it is worth pointing out that betweenness is *old* (Freeman (1977)). It was understood in terms of information flow in sociology. In this context, what if we wanted to generalize betweenness to higher-order networks. For instance, do messages going through a clique of best friends is any different than going through one of the bffs? We don't need to assume that everything is pairwise, as in the original formulation.

- clustering coefficient

$$C(i) = \frac{2\tau(i)}{k_i(k_i - 1)}$$

where τ_i is the number of triangles involving i.

⚠ Warning

Do you think that clustering coefficient has a different flavor than closeness and betweenness? Think about it for a second.

- closeness:

$$g_i = \frac{1}{\sum_{j \neq i} \ell_{ij}}$$

ℹ Note

In `networkX`, they write it such that:

$$C(u) = \frac{n-1}{\sum_{v=1}^{n-1} d(v,u)},$$

Can you think how and why you would prefer that over the other one?

2.5 stats

Focusing on metrics as distributions (have you heard of our lord and savior, the log-log plot), turning local into global:

A good first step is to remember your good ol' summary statistics. Given a vector of single things $\vec{x} = [x_1, x_2, ..., x_n]$, we can summarize with moments:

$$\langle x \rangle = \frac{1}{n} \sum_{u=1}^{n} x_i$$

$$\langle x^2 \rangle = \frac{1}{n} \sum_{u=1}^{n} x_i^2$$

then, we can find the variance doing $\langle x^2 \rangle - \langle x \rangle$. For instance, going back to degrees, we can find the average of the squares of the degrees degree by taking $\frac{k_1^2 + k_2^2 + \ldots k_{N-1}^2 + k_N^2}{N}$. Doing this kind of math is interesting because we can then construct quantities of interests of a network's degree distribution, such as:

$$\kappa = \frac{\langle k^2 \rangle}{\langle k \rangle^2}$$

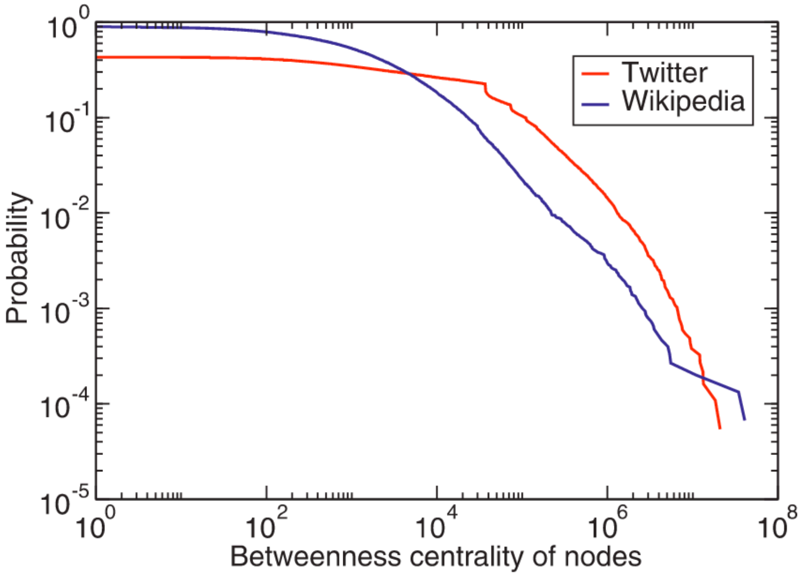


Figure 2.1: Menczer et al. 2020 Figure 3.14

When the average squared degree than the sqaure of the average degree, meaning that $\kappa \gg 1$, you know that you have a heavy-tailed distribution hiding somewhere.

Because i find stats confusing, here is another example of how to use averages to say something about networks. The average path length is given by:

$$\langle \ell \rangle = \frac{\sum_{i,j} \ell_{ij}}{\binom{N}{2}} = \frac{2 \sum_{i,j} \ell_{ij}}{N(N-1)}$$

where ℓ_{ij} is shortest-path counts between nodes i and j, and N is the number of nodes.

2.6 computational

What if we wanted to write a library in, say, `Javascript` to implement these properties? How would you got about it? We can start by looking at `networkX` ways of doing things. Here's degree centrality:

2.6.0.1 Degree centrality

easy-peasy code, if you have a `Graph` class object with `degree()` as method.

But what is this `Graph` object, lets find out in the doc:

2.7 what if..

Understanding through counterfactuals...

2.8 functional

Some properties are relevant to us because they predict network functionality, e.g.

- connect components x robustness

3 Searching for communities, should they be there

Alternative titles from `GPT4`:

```
"Detecting patterns in a sea of noise"
"Unearthing communities amidst overwhelming distractions"
"Isolating clusters in a cacophony of data"
"Sifting through the noise to identify meaningful groups"
"Discerning communities in the din of information"
```

I mean, do communities even exist? Its all about seeing a signal where there is none.

The Boogeyman of modularity:

- https://social.skewed.de/(**tiago/110303348191572767?**)
- https://skewed.de/tiago/blog/modularity-harmful
- https://reticular.hypotheses.org/1924

Ok how do people do it anyway?

3.0.1 Modularity (descriptive)

3.0.2 SBM (inferential)

3.1 Uncynical view of community detection

- Larremore's Large-scale structures in networks: Hidden communities and latent hierarchies

4 Visualizing networks, as long as it doesn't mislead you.

5 Art gallery

5.1 Sex networks

Figure 5.1: https://www.nature.com/articles/35082140/figures/1

6 The Old Gods and the New

Before Newman and Barabási, there were social scientists (Borgatti, Everett, and Johnson (2018)). You have `NetSci` on the one hand, and the `Sunbelt` on the other one. This is a fight about the old (social) gods and the New(man). The main difference, it seems to me, is about whether you think about networks verbally or mathematically. Given that, you get the following (heated) argument; social scientists getting pissy at physicists when they say that they started network science, while physicists think they are right because, before the 1990s, networks were merely described in words.

Anyway, here are artefacts of the old world that I still think are awesome.

6.1 the Seven Bridges of Königsberg

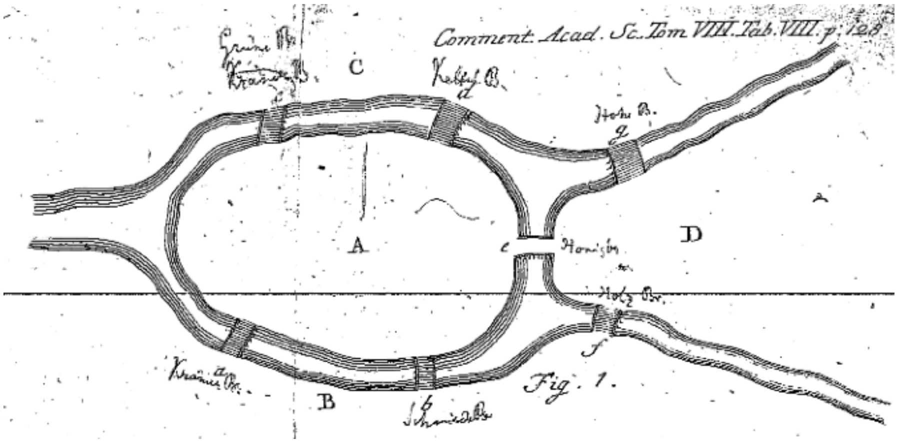
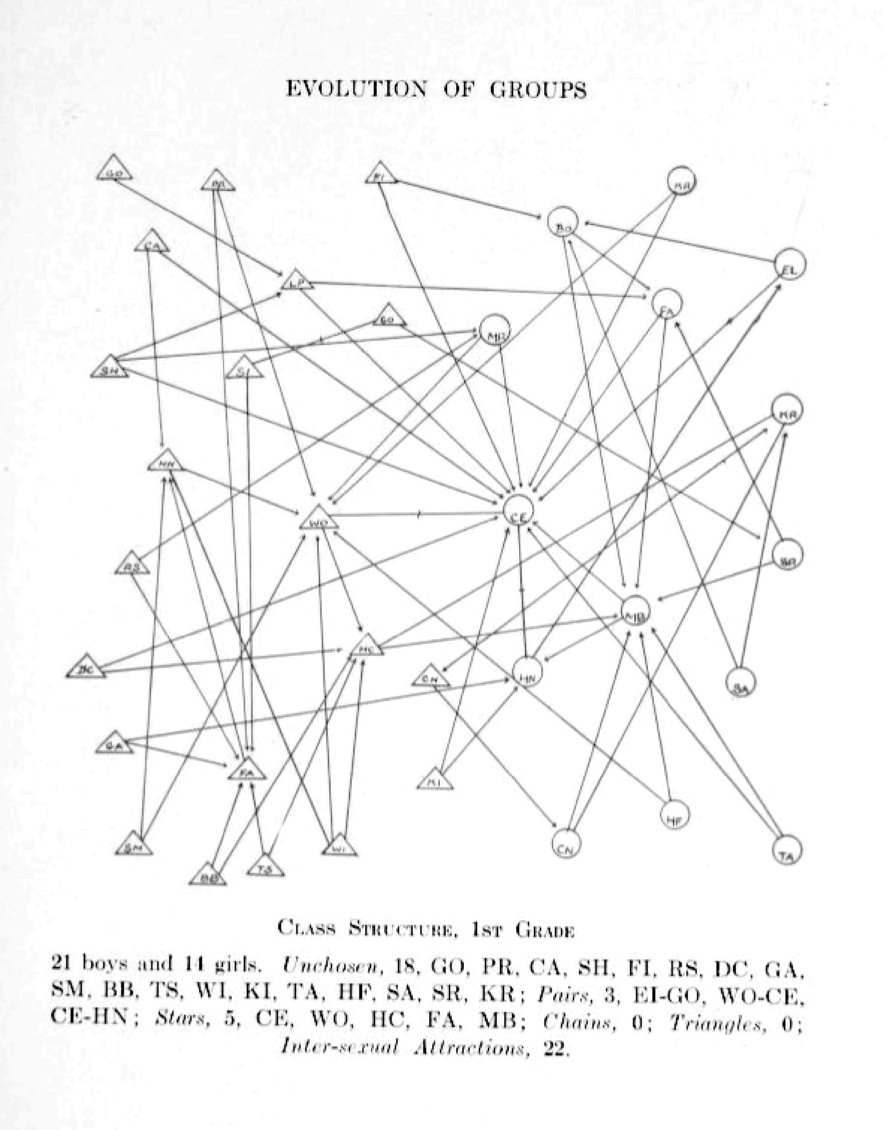


Figure 6.1: https://www.nature.com/articles/35082140/figures/1

6.2 first graders



EVOLUTION OF GROUPS

CLASS STRUCTURE, 1ST GRADE

21 boys and 14 girls. *Unchosen,* 18, GO, PR, CA, SH, FI, RS, DC, GA, SM, BB, TS, WI, KI, TA, HF, SA, SR, KR; *Pairs,* 3, EI-GO, WO-CE, CE-HN; *Stars,* 5, CE, WO, HC, FA, MB; *Chains,* 0; *Triangles,* 0; *Inter-sexual Attractions,* 22.

Figure 6.2: First Graders

References

Borgatti, Stephen P., Martin G. Everett, and Jeffrey C. Johnson. 2018. *Analyzing Social Networks*. SAGE Publications.

Freeman, Linton C. 1977. “A Set of Measures of Centrality Based on Betweenness.” In *Sociometry*, 40:35. https://doi.org/10.2307/3033543.

Kolaczyk, Eric D. 2009. *Statistical Analysis of Network Data: Methods and Models*. Springer Science & Business Media.

Menczer, Filippo, Santo Fortunato, and Clayton A. Davis. 2020. *A First Course in Network Science*. Cambridge University Press.

Newman, Mark. 2018. *Networks*. Oxford University Press.

Data sources

Network catalogues:

- http://konect.cc/
- https://icon.colorado.edu/#!/networks
- https://networks.skewed.de/
- http://www-personal.umich.edu/~mejn/netdata/
- http://vlado.fmf.uni-lj.si/pub/networks/data/Ucinet/UciData.htm
- https://www.cs.cornell.edu/~arb/data/

Books:

- https://github.com/CambridgeUniversityPress/FirstCourseNetworkScience/tree/master/datasets
- http://networksciencebook.com/translations/en/resources/data.html
- Network analysis made simple: I remember starting my network analysis journey with Eric Ma's online tutorials. He put them into this simple book, being very careful about how he introduces the code. This is a great resource for newcomers.

Kaggle:

- https://www.kaggle.com/datasets/mylesoneill/game-of-thrones

Guide to other guides:

- https://guides.library.jhu.edu/datavisualization/network