An introduction to blockmodeling

Carl Nordlund

Carl.nordlund@liu.se

www.carlnordlund.net

LINKÖPING UNIVERSITY

Institute for Analytical Sociology

Roadmap to lecture

- Role-analysis: theoretical considerations and sociological origins
- Conceptual examples
- Learn about positions, roles, blocks
- Equivalences: core concept
  - Focus on structural equivalence
- Blockmodeling heuristics
  - Focus on indirect approaches
- Run through examples

Followed by a short interactive session
Python notebook on blockmodeling

Terminology and network formats

- Nodes: “actors” (classical SNA notion) – here, interchangeably
- Data formats:

Blockmodeling: exclusively in sociomatrix format

Edgelist

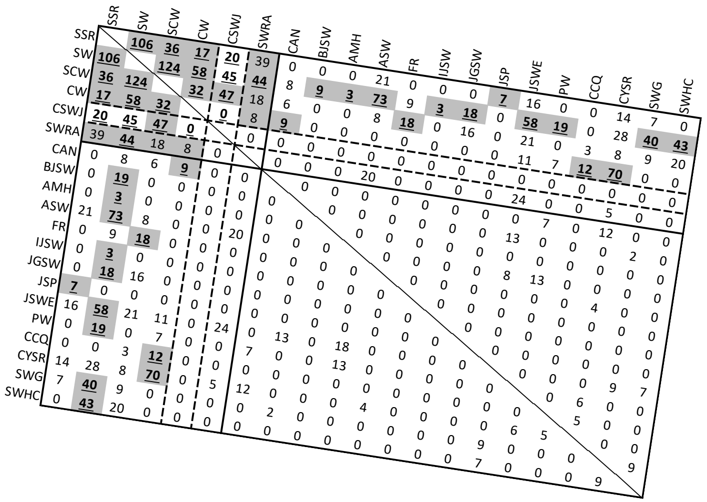
```
c1 c2 c4 p11 p12
c2 c1 c3 p21 p22
c3 c2 c4 p31 p32 p33
c4 c1 c3 p41 p42
p11 c1
p12 c1
p21 c2
p22 c2
p31 c3
p32 c3
p33 c3
p41 c4
p42 c4
```

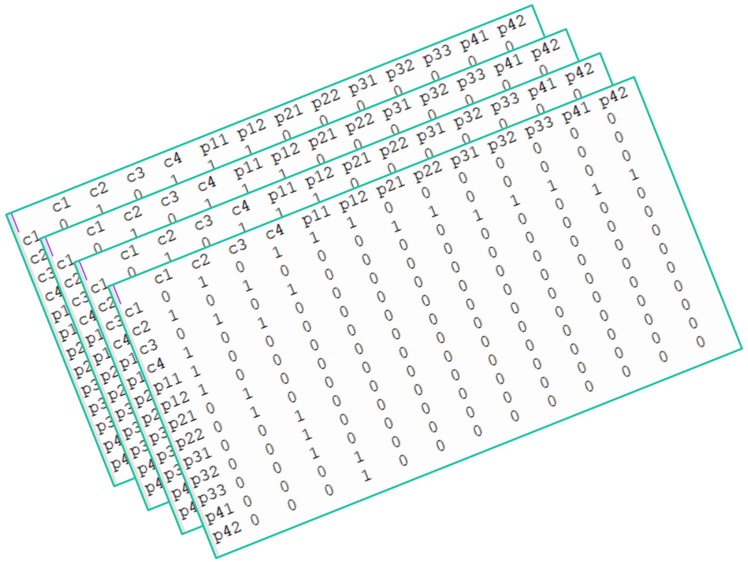
Sociomatrix

	c1	c2	c3	c4	p11	p12	p21	p22	p31	p32	p33	p41	p42
c1	0	1	0	1	1	1	0	0	0	0	0	0	0
c2	1	0	1	0	0	0	1	1	0	0	0	0	0
c3	0	1	0	1	0	0	0	0	1	1	1	0	0
c4	1	0	1	0	0	0	0	0	0	0	0	1	1
p11	1	0	0	0	0	0	0	0	0	0	0	0	0
p12	1	0	0	0	0	0	0	0	0	0	0	0	0
p21	0	1	0	0	0	0	0	0	0	0	0	0	0
p22	0	1	0	0	0	0	0	0	0	0	0	0	0
p31	0	0	1	0	0	0	0	0	0	0	0	0	0
p32	0	0	1	0	0	0	0	0	0	0	0	0	0
p33	0	0	1	0	0	0	0	0	0	0	0	0	0
p41	0	0	0	1	0	0	0	0	0	0	0	0	0
p42	0	0	0	1	0	0	0	0	0	0	0	0	0

Limitations in this introduction

- Single-layer (no multilayer/plex)
- 1-mode/unipartite (no 2-mode/bipartite)
- Binary (not valued, not signed)





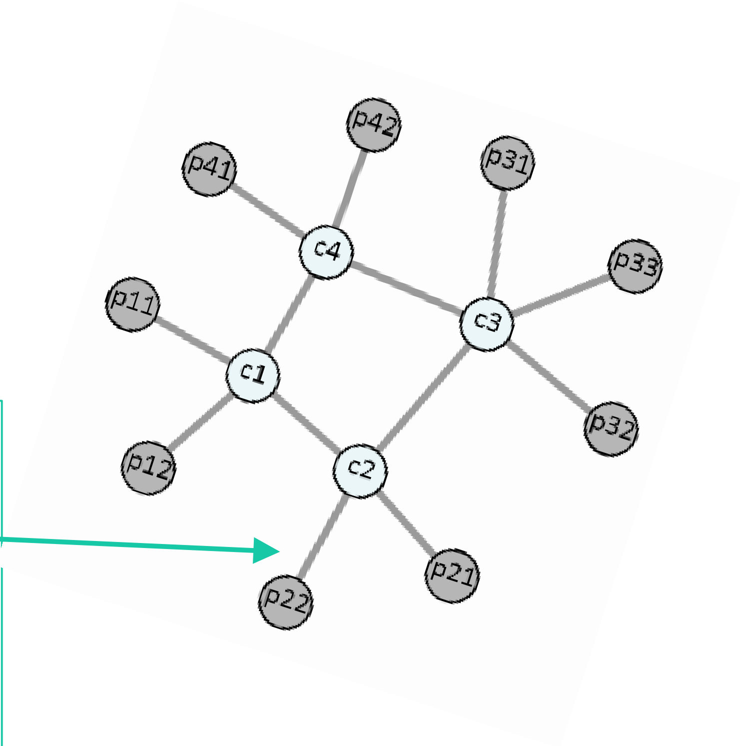
Aim: provide fundamentals, opening up for future studies!

Reading a sociomatrix

“to”

“from”

	c1	c2	c3	c4	p11	p12	p21	p22	p31	p32	p33	p41	p42
c1	0	1	0	1	1	1	0	0	0	0	0	0	0
c2	1	0	1	0	0	0	1	1	0	0	0	0	0
c3	0	1	0	1	0	0	0	0	1	1	1	0	0
c4	1	0	1	0	0	0	0	0	0	0	0	1	1
p11	1	0	0	0	0	0	0	0	0	0	0	0	0
p12	1	0	0	0	0	0	0	0	0	0	0	0	0
p21	0	1	0	0	0	0	0	0	0	0	0	0	0
p22	0	1	0	0	0	0	0	0	0	0	0	0	0
p31	0	0	1	0	0	0	0	0	0	0	0	0	0
p32	0	0	1	0	0	0	0	0	0	0	0	0	0
p33	0	0	1	0	0	0	0	0	0	0	0	0	0
p41	0	0	0	1	0	0	0	0	0	0	0	0	0
p42	0	0	0	1	0	0	0	0	0	0	0	0	0



Role-analysis?

"3rd pillar" of Social Network Analysis

See what I did there? I used the word "actor" instead of "node"!

1st: Node-centric properties and metrics (micro-level)
Measures and metrics that say something about individual actors, e.g. centrality measures

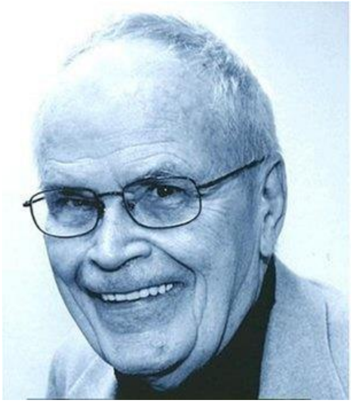
2nd: Cohesive subgroups, triadic analysis etc (meso-level)
Measures and metrics relating to sets of actors (or more often: the sets themselves)

3rd: Macro-level: measures and metrics that say something about the whole network
Density! Average betweenness centrality! Etc
But more overarching and relevant: role analysis and blockmodeling

Indeed dealing with individual actors and subsets of actors, but the analysis and approach is about the network as a whole

Historical roots

Harrison White



With colleagues, asked whether sociologists can define the “role concept” in a purely relational form?

Society IS social structure: a multiplex network of inter-personal relations (friendships, kinships, dialogue, interaction, exchange, affiliation networks etc)

Roles and identities in society are “reciprocally defined”: only emerges in the interactions. How we relate to each other.

Robinson Crusoe didn’t have a role!

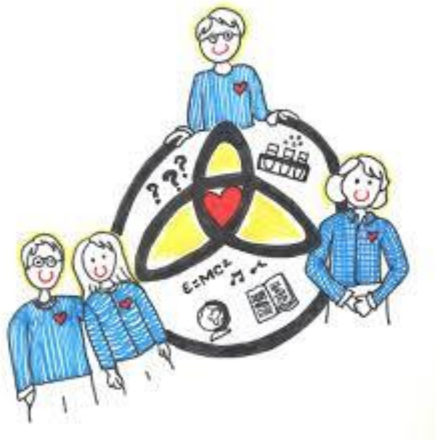
Notions of role and position

At a hospital!

- We have patients, nurses, doctors, and admin personnel
  - Different attributes: work descriptions, salaries, tasks, educations
- But also different roles: different types of relations with each other:
  - Doctors relate to each other, and to patients
  - Nurses relate to doctors, patients and to admin personnel
  - Admin personnel relate to patients
  - Patients relate to all, except other patients

Parents, kids, school teachers

- The parents in a family relate with each other, some of other parents
- Parents relate to their kids, not other kids
- Teachers relate to all kids, some parents, some teachers



Notions of role and position

Identifying and operationalizing positions

Somehow we want to partition the actors (nodes) in a network into subsets (positions/clusters) based on some notion of equivalence

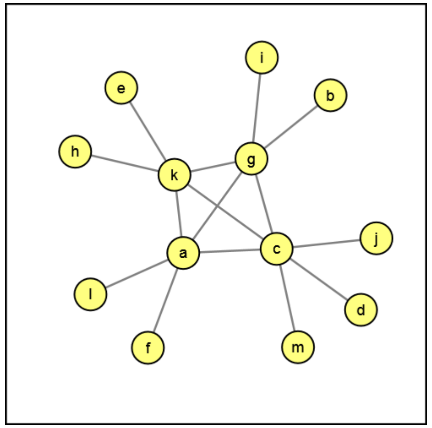
Having similar patterns of relations somehow…

Mapping role-structure

Mapping overall relationships of actors within and between such positions

Quintessential technique for doing the above:
BLOCKMODELING

Quick'n'dirty example: Galtung's feudal interaction structure



(Galtung 1971)

	a	b	c	d	e	f	g	h	i	j	k	l	m
a		0	1	0	0	1	1	0	0	0	1	1	0
b	0		0	0	0	0	1	0	0	0	0	0	0
c	1	0		1	0	0	1	0	0	1	1	0	1
d	0	0	1		0	0	0	0	0	0	0	0	0
e	0	0	0	0		0	0	0	0	0	1	0	0
f	1	0	0	0	0		0	0	0	0	0	0	0
g	1	1	1	0	0	0		0	1	0	1	0	0
h	0	0	0	0	0	0	0		0	0	1	0	0
i	0	0	0	0	0	0	1	0		0	0	0	0
j	0	0	1	0	0	0	0	0	0		0	0	0
k	1	0	1	0	1	0	1	1	0	0		0	0
l	1	0	0	0	0	0	0	0	0	0	0		0
m	0	0	1	0	0	0	0	0	0	0	0	0	

Sociomatrix representation

Intuitive understanding of this network

A set of 4 "central" and connected actors in the middle. Remaining actors hanging on to these, otherwise poorly connected.

So let's hypothesize that these two represent two different positions:

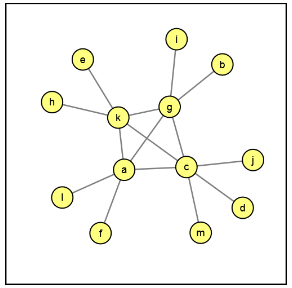
P1={a,c,g,k}
P2={b,d,e,f,h,i,j,l,m}

We sort the original matrix according to these positions

		P1				P2								
		a	g	k	c	l	f	i	b	e	h	m	j	d
P1	a		1	1	1	1	1	0	0	0	0	0	0	0
	g	1		1	1	0	0	1	1	0	0	0	0	0
	k	1	1		1	0	0	0	0	1	1	0	0	0
	c	1	1	1		0	0	0	0	0	0	1	1	1
P2	l	1	0	0	0		0	0	0	0	0	0	0	0
	f	1	0	0	0	0		0	0	0	0	0	0	0
	i	0	1	0	0	0	0		0	0	0	0	0	0
	b	0	1	0	0	0	0	0		0	0	0	0	0
	e	0	0	1	0	0	0	0	0		0	0	0	0
	h	0	0	1	0	0	0	0	0	0		0	0	0
	m	0	0	0	1	0	0	0	0	0	0		0	0
	j	0	0	0	1	0	0	0	0	0	0	0		0
	d	0	0	0	1	0	0	0	0	0	0	0	0	

Galtung’s feudal interaction structure

A hypothetical blockmodel of the Galtung network



		P1				P2								
		a	g	k	c	l	f	l	b	e	h	m	j	d
P1	a		1	1	1	1	1	0	0	0	0	0	0	0
	g	1		1	1	0	0	1	1	0	0	0	0	0
	k	1	1		1	0	0	0	0	1	1	0	0	0
	c	1	1	1		0	0	0	0	0	0	1	1	1
P2	l	1	0	0	0		0	0	0	0	0	0	0	0
	f	1	0	0	0	0		0	0	0	0	0	0	0
	l	0	1	0	0	0	0		0	0	0	0	0	0
	b	0	1	0	0	0	0	0		0	0	0	0	0
	e	0	0	1	0	0	0	0	0		0	0	0	0
	h	0	0	1	0	0	0	0	0	0		0	0	0
	m	0	0	0	1	0	0	0	0	0	0		0	0
	j	0	0	0	1	0	0	0	0	0	0	0		0
	d	0	0	0	1	0	0	0	0	0	0	0	0	

I.e. a specific partition of the actors (with the sociomatrix ordered according this this partition)

Two (2) positions, four (2x2) blocks

Having sorted the original sociomatrix according to our 2 positions (P1 and P2), a total of four (2x2) blocks can be delineated, capturing the ties within and between actors of the positions.

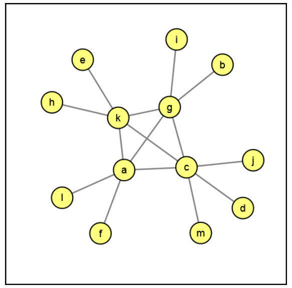
Blocks: P1-P1, P1-P2, P2-P1, P2-P2

In this state, we observe certain patterns in these blocks!

P1-P1: completely filled with ties (except for the diagonal)
P2-P2: totally missing ties
P1-P2, P2-p1: not empty, not filled, something in-between (with ties on each row and each column of the blocks)

Galtung's feudal interaction structure

A hypothetical blockmodel of the Galtung network



		P1				P2								
		a	g	k	c	l	f	I	b	e	h	m	j	d
P1	a		1	1	1	1	1	0	0	0	0	0	0	0
	g	1		1	1	0	0	1	1	0	0	0	0	0
	k	1	1		1	0	0	0	0	1	1	0	0	0
	c	1	1	1		0	0	0	0	0	0	1	1	1
P2	l	1	0	0	0		0	0	0	0	0	0	0	0
	f	1	0	0	0	0		0	0	0	0	0	0	0
	I	0	1	0	0	0	0		0	0	0	0	0	0
	b	0	1	0	0	0	0	0		0	0	0	0	0
	e	0	0	1	0	0	0	0	0		0	0	0	0
	h	0	0	1	0	0	0	0	0	0		0	0	0
	m	0	0	0	1	0	0	0	0	0	0		0	0
	j	0	0	0	1	0	0	0	0	0	0	0		0
	d	0	0	0	1	0	0	0	0	0	0	0	0	

Interpreting the blocks

Turns out that the blocks we identify in this hypothetical blockmodel represent three ideal blocks in blockmodeling:

Structural equivalence blockmodeling

Regular equivalence blockmodeling

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

	1	1	1
1		1	1
1	1		1
1	1	1	

The complete block (com)

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	

The null block (nul)

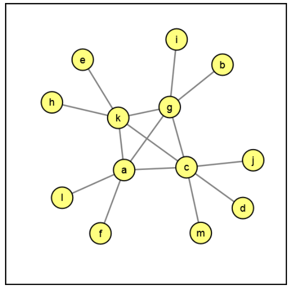
1	0	1	0	0
0	0	0	1	0
0	1	0	0	1
0	0	1	0	0

	1	1	0
1		1	0
0	0		1
0	1	0	

The regular block (reg)

Galtung's feudal interaction structure

A hypothetical blockmodel of the Galtung network



		P1				P2								
		a	g	k	c	l	f	l	b	e	h	m	j	d
P1	a		1	1	1	1	1	0	0	0	0	0	0	0
	g	1		1	1	0	0	1	1	0	0	0	0	0
	k	1	1		1	0	0	0	0	1	1	0	0	0
	c	1	1	1		0	0	0	0	0	0	1	1	1
P2	l	1	0	0	0		0	0	0	0	0	0	0	0
	f	1	0	0	0	0		0	0	0	0	0	0	0
	l	0	1	0	0	0	0		0	0	0	0	0	0
	b	0	1	0	0	0	0	0		0	0	0	0	0
	e	0	0	1	0	0	0	0	0		0	0	0	0
	h	0	0	1	0	0	0	0	0	0		0	0	0
	m	0	0	0	1	0	0	0	0	0	0		0	0
	j	0	0	0	1	0	0	0	0	0	0	0		0
	d	0	0	0	1	0	0	0	0	0	0	0	0	

Collapsing the blockmodel

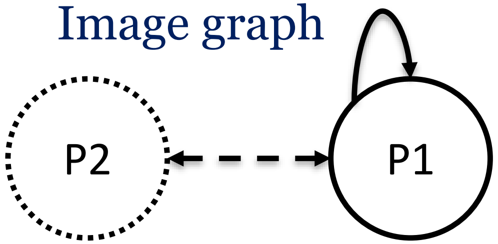
Having modeled each empirical block in terms of an ideal block, we can now "collapse" the blockmodel into a block image, or an image graph.

Here, nodes represent positions (subsets of equivalent actors)

Block image

	P1	P2
P1	com	reg
P2	reg	nul

Image graph



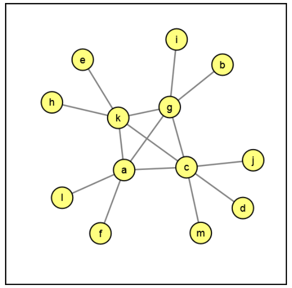
From this: roles

Role of P1 is to be connected other in P1, and having sporadic ties to and from P2

Role of P2 is to have sporadic ties with P1, but lack ties to other in P2

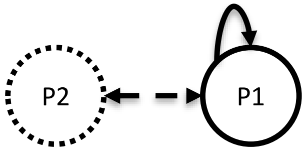
Galtung’s feudal interaction structure

But why do we do blockmodeling?



		P1				P2								
		a	g	k	c	l	f	I	b	e	h	m	j	d
P1	a		1	1	1	1	1	0	0	0	0	0	0	0
	g	1		1	1	0	0	1	1	0	0	0	0	0
	k	1	1		1	0	0	0	0	1	1	0	0	0
	c	1	1	1		0	0	0	0	0	0	1	1	1
P2	l	1	0	0	0		0	0	0	0	0	0	0	0
	f	1	0	0	0	0		0	0	0	0	0	0	0
	I	0	1	0	0	0	0		0	0	0	0	0	0
	b	0	1	0	0	0	0	0		0	0	0	0	0
	e	0	0	1	0	0	0	0	0		0	0	0	0
	h	0	0	1	0	0	0	0	0	0		0	0	0
	m	0	0	0	1	0	0	0	0	0	0		0	0
	j	0	0	0	1	0	0	0	0	0	0	0		0
	d	0	0	0	1	0	0	0	0	0	0	0	0	

	P1	P2
P1	com	reg
P2	reg	nul



Reduction of complexity

A blockmodel reduces a complex network into its fundamental structure. We tease out the “functional anatomy”.

In the Galtung network, we find a blockimage that corresponds to a classical core-periphery structure.

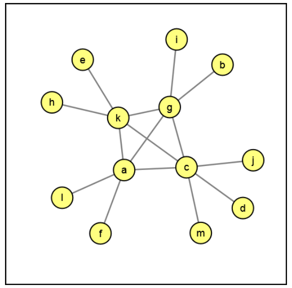
For a network twice (or 100 times) the size, the same block image could appear, with different relative sizes of P1 and P2. Still the same underlying functional anatomy.

Allows for comparing complex networks: finding similarities in what could appear as dissimilar

Galtung’s feudal interaction structure

But why do we do blockmodeling?

		P1				P2								
		a	g	k	c	l	f	l	b	e	h	m	j	d
P1	a		1	1	1	1	1	0	0	0	0	0	0	0
	g	1		1	1	0	0	1	1	0	0	0	0	0
	k	1	1		1	0	0	0	0	1	1	0	0	0
	c	1	1	1		0	0	0	0	0	0	1	1	1
P2	l	1	0	0	0		0	0	0	0	0	0	0	0
	f	1	0	0	0	0		0	0	0	0	0	0	0
	l	0	1	0	0	0	0		0	0	0	0	0	0
	b	0	1	0	0	0	0	0		0	0	0	0	0
	e	0	0	1	0	0	0	0	0		0	0	0	0
	h	0	0	1	0	0	0	0	0	0		0	0	0
	m	0	0	0	1	0	0	0	0	0	0		0	0
	j	0	0	0	1	0	0	0	0	0	0	0		0
	d	0	0	0	1	0	0	0	0	0	0	0	0	



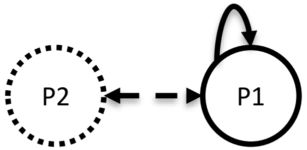
Comparisons with structural types

The Galtung network turned out to be a core-periphery structure

…but there are a multitude of other possibilities: hierarchies, transitive structures, cohesive subgroups.

Community structures: a special case (one possible outcome) of blockmodeling!

	P1	P2
P1	com	reg
P2	reg	nul



	P1	P2	P3
P1	com	nul	nul
P2	nul	com	nul
P3	nul	nul	com

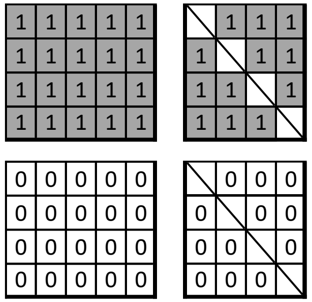
Types of equivalences

What is meant by equivalence in a network? Different notions and usages exist.

Structural equivalence

Two actors are structurally equivalent if they have identical sets of ties to the other actors.

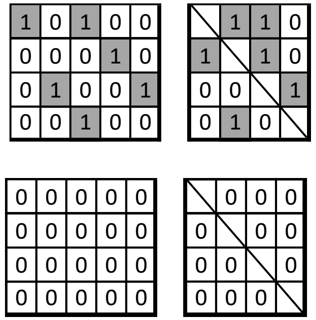
Under this notion of equivalence, blockmodels would ideally consist of complete and null blocks



Regular equivalence

Two actors are regular equivalent if they have ties to actors that in turn are also regular equivalent (!?)

Under this notion of equivalence, blockmodels would ideally consist of null blocks and regular blocks (at least one tie in each row, and at least one tie in each column)

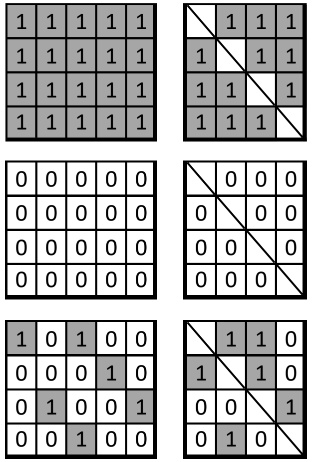


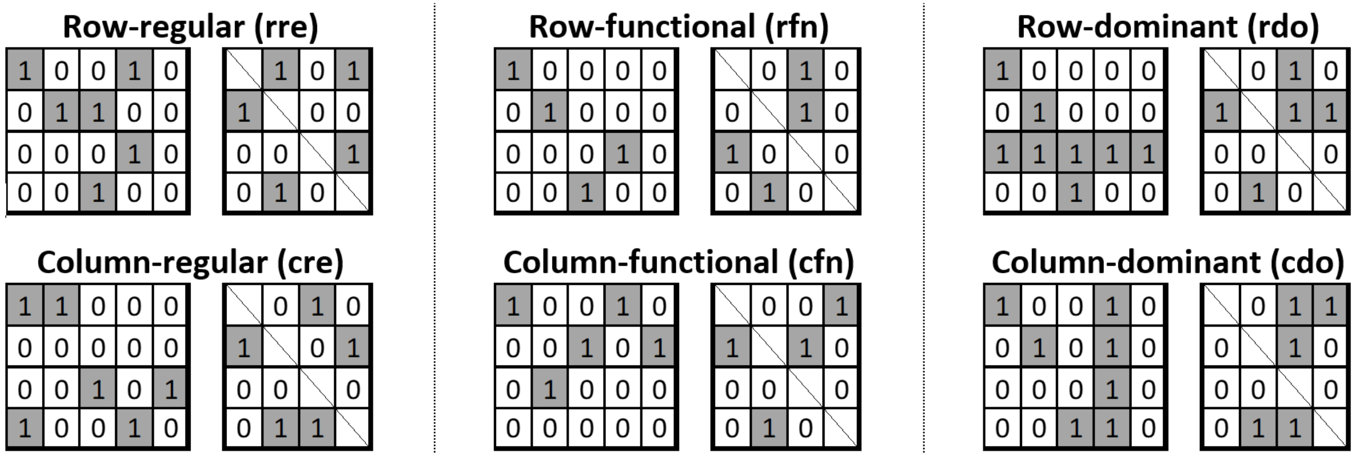
Types of equivalences

What is meant by equivalence in a network? Different notions and usages exist.

Generalized equivalence

Anything goes! In generalized blockmodeling, ideal blocks consists of null, complete and regular blocks – and a host of other block types:





Types of equivalences

What is meant by equivalence in a network? Different notions and usages exist.

Structural, regular, generalized equivalence

Depending on the notion of equivalence that is used, different outcomes will likely emerge. Most common: structural equivalence, then regular equivalence. Sometimes special blockimages need specific ideal blocks (e.g. power-relational core-periphery structures have complete, null and regular blocks)

Different notions of equivalence => Different outcomes and interpretations

Ideal vs. Real world

In the Galtung network, we managed to perfectly fit the network to ideal blocks (and on first attempt as well!). This is typically rare! Real-world networks are more messy!

Thus: ambition is to find the best possible fit (but we are likely to find inconsistencies)

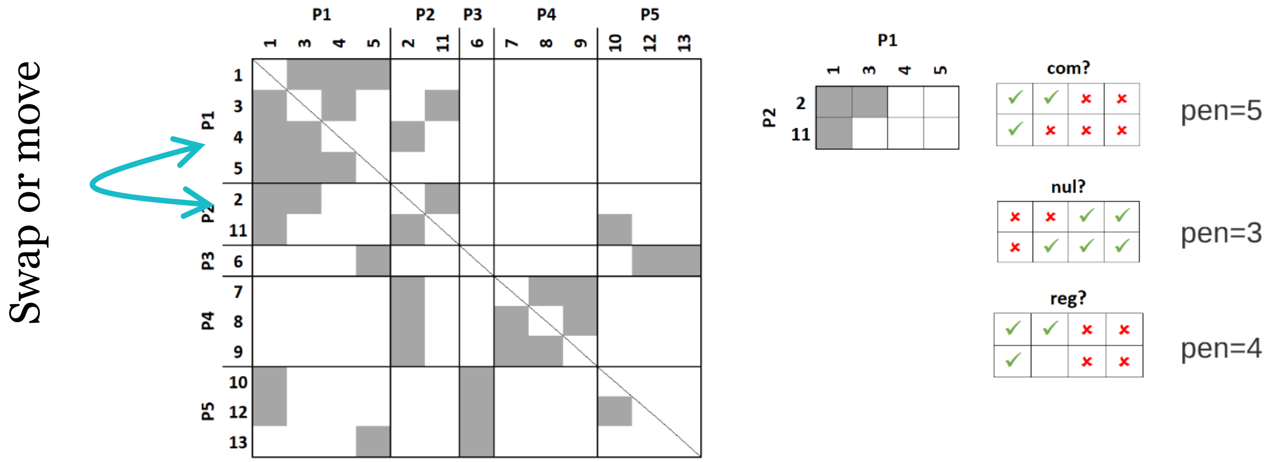
Finding positions

There are two broad approaches for finding equivalent partitions in a network: direct and indirect methods.

Actually three ways: hypothesis testing as well

Direct blockmodeling (semi-supervised)

Given a random starting partition, given number of positions, and a set of ideal blocks, the direct approach uses genetic algorithms (local optimization) to move/swap actors between positions. A criteria function is used (e.g. number of inconsistencies), testing among the various types of ideal blocks.



Advantage:
Deemed better
Allows for capturing complicated ideal blocks (generalized blockmodeling)

Disadvantage:
Computationally heavy
Potentially finding local minima

Finding positions

There are two broad approaches for finding equivalent partitions in a network: direct and indirect methods.

Indirect blockmodeling

First, calculate/determine a suitable indirect measure of equivalence. How these are calculated depend on the notion of equivalence we are interested in.

This yields a new matrix (distances or correlations)

Apply hierarchical clustering to this matrix,
Choose suitable number of positions
Create blockmodel

In what follows:
Indirect structural equivalence!

Advantage:
Works for large networks
Analytical, not computational

Disadvantage:
Restricted to certain notions of equivalence
Generalized blockmodeling: nope!
Lots of choices in the analysis

Indirect structural equivalence

Little League Baseball (Transatlantic Industries team)

Reported friendships between kids playing baseball (Fine 1987)

	Ron_1	Tom_2	Frank_3	Boyd_4	Tim_5	John_6	Jeff_7	Jay_8	Sandy_9	Jerry_10	Darrin_11	Ben_12	Arnie_13
Ron_1		0	1	1	1	0	0	0	0	0	0	0	0
Tom_2	1		1	0	0	0	0	0	0	0	1	0	0
Frank_3	1	0		1	0	0	0	0	0	0	1	0	0
Boyd_4	1	1	1		0	0	0	0	0	0	0	0	0
Tim_5	1	0	1	1		0	0	0	0	0	0	0	0
John_6	0	0	0	0	1		0	0	0	0	0	1	1
Jeff_7	0	1	0	0	0	0		1	1	0	0	0	0
Jay_8	0	1	0	0	0	0	1		1	0	0	0	0
Sandy_9	0	1	0	0	0	0	1	1		0	0	0	0
Jerry_10	1	0	0	0	0	1	0	0	0		0	0	0
Darrin_11	1	1	0	0	0	0	0	0	0	1		0	0
Ben_12	1	0	0	0	0	1	0	0	0	1	0		0
Arnie_13	0	0	0	0	1	1	0	0	0	0	0	0	

Self-ties: non-existent:
instead, check for reciprocity

What does structural equivalence mean?
That two actors have the same patterns of ties to the other actors!

Let's pick two actors (e.g. Tom_2 and Frank_3) and look at their respective rows (outbound ties):

Tom_2	1		1	0	0	0	0	0	0	0	1	0	0
Frank_3	1	0		1	0	0	0	0	0	0	1	0	0

Inconsistencies: 7 (+2)
Similarities: 14

...and columns (inbound ties):

Tom_2	0		0	1	0	0	1	1	1	0	1	0	0
Frank_3	1	1		1	1	0	0	0	0	0	0	0	0

Repeat for all pairs of actors!

Indirect structural equivalence

Indirect measures of structural equivalence

Tom_2	1		1	0	0	0	0	0	0	0	1	0	0
Frank_3	1	0		1	0	0	0	0	0	0	1	0	0

Tom_2	0		0	1	0	0	1	1	1	0	1	0	0
Frank_3	1	1		1	1	0	0	0	0	0	0	0	0

Different ways to compare these vectors

Hamming: number of inconsistencies
Euclidean distances
Correlations
(etc.)

Additional choices

Just rows, just columns or both
What to do with reciprocal ties (exclude, include once, include twice)?
Valued networks: correlations or Euclidean distances?
(Dichotomization?)

The provided Python code checks both rows and columns, including the recipcrocal once, and either using the Hamming distance or the correlation.

Indirect structural equivalence

Little League Baseball (Transatlantic Industries team)

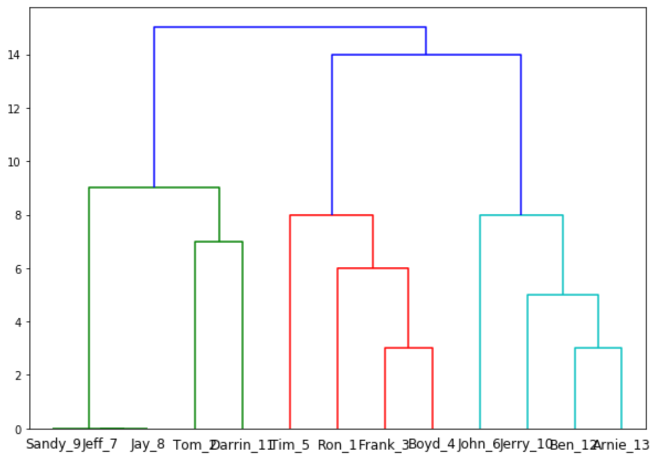
Indirect structural equivalence (Hamming distances)

	Ron_1	Frank_3	Boyd_4	Tim_5	Tom_2	Darrin_11	John_6	Ben_12	Arnie_13	Jeff_7	Jay_8	Sandy_9	Jerry_10
Ron_1		1	1	1	0	0	0	0	0	0	0	0	0
Frank_3	1		1	0	0	1	0	0	0	0	0	0	0
Boyd_4	1	1		0	1	0	0	0	0	0	0	0	0
Tim_5	1	1	1		0	0	0	0	0	0	0	0	0
Tom_2	1	1	0	0		1	0	0	0	0	0	0	0
Darrin_11	1	0	0	0	1		0	0	0	0	0	0	1
John_6	0	0	0	1	0	0		1	1	0	0	0	0
Ben_12	1	0	0	0	0	0	1		0	0	0	0	1
Arnie_13	0	0	0	1	0	0	1	0		0	0	0	0
Jeff_7	0	0	0	0	1	0	0	0	0		1	1	0
Jay_8	0	0	0	0	1	0	0	0	0	1		1	0
Sandy_9	0	0	0	0	1	0	0	0	0	1	1		0
Jerry_10	1	0	0	0	0	0	1	0	0	0	0	0	

Original sociomatrix

dist	Ron_1	Frank_3	Boyd_4	Tim_5	Tom_2	Darrin_11	John_6	Ben_12	Arnie_13	Jeff_7	Jay_8	Sandy_9	Jerry_10
Ron_1	0	5	6	8	12	11	10	14	11	15	15	15	10
Frank_3	5	0	3	7	9	8	13	9	10	12	12	12	9
Boyd_4	6	3	0	6	10	5	12	8	9	9	9	9	8
Tim_5	8	7	6	0	10	9	10	6	7	11	11	11	8
Tom_2	12	9	10	10	0	7	14	10	11	9	9	9	8
Darrin_11	11	8	5	9	7	0	11	5	8	8	8	8	7
John_6	10	13	12	10	14	11	0	6	3	11	11	11	8
Ben_12	14	9	8	6	10	5	6	0	3	9	9	9	4
Arnie_13	11	10	9	7	11	8	3	3	0	8	8	8	5
Jeff_7	15	12	9	11	9	8	11	9	8	0	0	0	9
Jay_8	15	12	9	11	9	8	11	9	8	0	0	0	9
Sandy_9	15	12	9	11	9	8	11	9	8	0	0	0	9
Jerry_10	10	9	8	8	8	7	8	4	5	9	9	9	0

Distance matrix (Hamming)



Hierarchical clustering (complete-link)

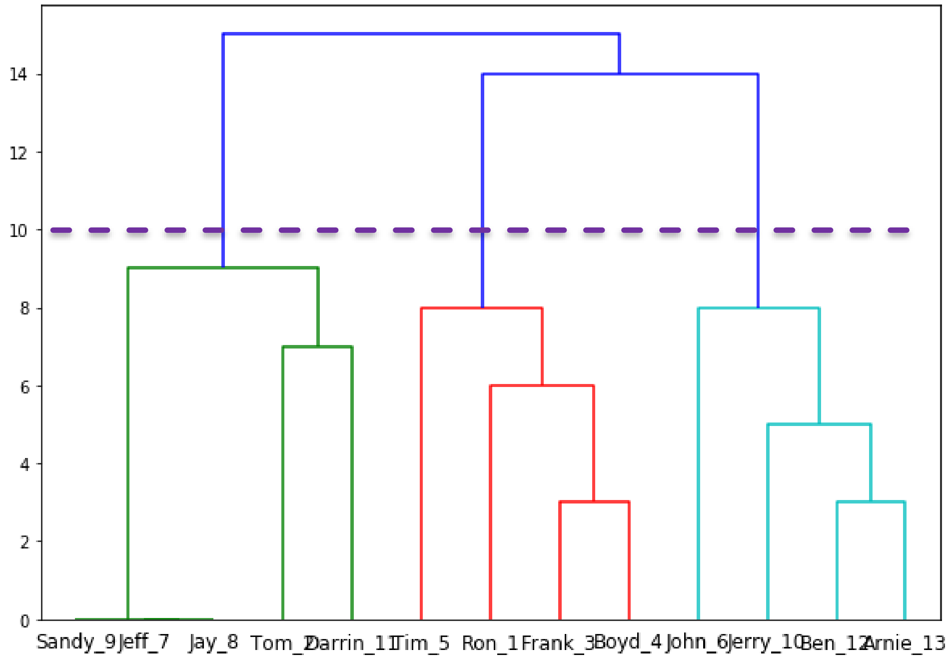
bm.indirectSE(mat,method='hamming')

Note how Jeff_7, Jay_8 and Sandy_9 are all perfectly structurally equivalent!

Indirect structural equivalence

Little League Baseball (Transatlantic Industries team)

Indirect structural equivalence (Hamming distances); cutting dendrogram



Cutting the dendrogram: deciding on the number of positions!

Further down (strict criteria for equivalence): many positions
Further up (loose criteria for equivalence): few positions

Ways to decide

Theoretically : Perhaps you know how many to expect (e.g. professions at a hospital)
Test metrics (similar to modularity in Newman-Girvan)
Or just try several, evaluate resulting blockmodel

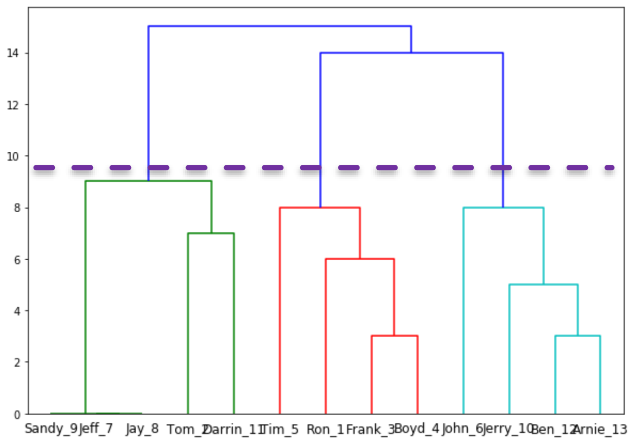
Cutting at 7: 6 positions
Cutting at 10: 3 positions (We try this here)

Indirect structural equivalence

Little League Baseball (Transatlantic Industries team)

Indirect structural equivalence (Hamming distances)

	Ron_1	Frank_3	Boyd_4	Tim_5	Tom_2	Darrin_11	John_6	Ben_12	Arnie_13	Jeff_7	Jay_8	Sandy_9	Jerry_10
Ron_1		0	1	1	1	0	0	0	0	0	0	0	0
Frank_3	1		1	0	0	0	0	0	0	0	1	0	0
Boyd_4	1	0		1	0	0	0	0	0	0	1	0	0
Tim_5	1	1	1		0	0	0	0	0	0	0	0	0
Tom_2	1	0	1	1		0	0	0	0	0	0	0	0
Darrin_11	0	0	0	0	1		0	0	0	0	0	1	1
John_6	0	1	0	0	0	0		1	1	0	0	0	0
Ben_12	0	1	0	0	0	0	1		1	0	0	0	0
Arnie_13	0	1	0	0	0	0	1	1		0	0	0	0
Jeff_7	1	0	0	0	0	1	0	0	0		0	0	0
Jay_8	1	1	0	0	0	0	0	0	0	1		0	0
Sandy_9	1	0	0	0	0	1	0	0	0	1	0		0
Jerry_10	0	0	0	0	1	1	0	0	0	0	0	0	



		P1					P2				P3			
		Tom_2	Darrin_11	Jeff_7	Jay_8	Sandy_9	Ron_1	Frank_3	Boyd_4	Tim_5	John_6	Ben_12	Arnie_13	Jerry_10
P1	Tom_2		1	0	0	0	1	1	0	0	0	0	0	0
	Darrin_11	1		0	0	0	1	0	0	0	0	0	0	1
	Jeff_7	1	0		1	1	0	0	0	0	0	0	0	0
	Jay_8	1	0	1		1	0	0	0	0	0	0	0	0
	Sandy_9	1	0	1	1		0	0	0	0	0	0	0	0
P2	Ron_1	0	0	0	0	0		1	1	1	0	0	0	0
	Frank_3	0	1	0	0	0	1		1	0	0	0	0	0
	Boyd_4	1	0	0	0	0	1	1		0	0	0	0	0
	Tim_5	0	0	0	0	0	1	1	1		0	0	0	0
P3	John_6	0	0	0	0	0	0	0	0	1		1	1	0
	Ben_12	0	0	0	0	0	1	0	0	0	1		0	1
	Arnie_13	0	0	0	0	0	0	0	0	1	1	0		0
	Jerry_10	0	0	0	0	0	1	0	0	0	1	0	0	

P1={Tom_2, Darrin_11, Jeff_7, Jay_8, Sandy_9}
P2={Ron_1, Frank_3, Boyd_4, Tim_5}
P3={John_6, Ben_12, Arnie_13, Jerry_10}

Indirect structural equivalence

Little League Baseball (Transatlantic Industries team)

Indirect structural equivalence (Hamming distances)

bm.calcDensityBlockimage(mat,blockdict)

		P1					P2				P3			
		Tom_2	Darrin_11	Jeff_7	Jay_8	Sandy_9	Ron_1	Frank_3	Boyd_4	Tim_5	John_6	Ben_12	Arnie_13	Jerry_10
P1	Tom_2	\	1	0	0	0	1	1	0	0	0	0	0	0
	Darrin_11	1	\	0	0	0	1	0	0	0	0	0	0	1
	Jeff_7	1	0	\	1	1	0	0	0	0	0	0	0	0
	Jay_8	1	0	1	\	1	0	0	0	0	0	0	0	0
	Sandy_9	1	0	1	1	\	0	0	0	0	0	0	0	0
P2	Ron_1	0	0	0	0	0	\	1	1	1	0	0	0	0
	Frank_3	0	1	0	0	0	1	\	1	0	0	0	0	0
	Boyd_4	1	0	0	0	0	1	1	\	0	0	0	0	0
	Tim_5	0	0	0	0	0	1	1	1	\	0	0	0	0
P3	John_6	0	0	0	0	0	0	0	0	1	\	1	1	0
	Ben_12	0	0	0	0	0	1	0	0	0	1	\	0	1
	Arnie_13	0	0	0	0	0	0	0	0	1	1	0	\	0
	Jerry_10	0	0	0	0	0	1	0	0	0	1	0	0	\

To assist in interpreting the blockmodel, we can calculate block densities:

	P1	P2	P3
P1	0.55	0.15	0.05
P2	0.10	0.83	0.00
P3	0.00	0.25	0.50

Although not very “crisp”, we could interpret it like:

	P1	P2	P3
P1	com	nul	nul
P2	nul	com	nul
P3	nul	nul	com

Which is a cohesive subgroup!

Indirect structural equivalence

What to do with these results?

- **Interpret block image**
  What is the functional anatomy of the network at large?
  Does it follow a given structural template (e.g. cohesive groups, core-periphery etc) or is it best explained with a "custom" solution?
- **Compare position/role with other nodal attributes**
  In the early blockmodeling studies of international trade networks, identified roles of countries were connected to country developmental attributes (GDP/cap, gini coefficients etc)
- **Test goodness-of-fit measures for solution**
  Test with different number of positions (k=2,3,4…)
  Test with other notions of equivalence (but ok, in this course only structural equivalence)
  Make different choices along the way (clustering approach, indirect measures etc)
  …all to examine the "stability" of your finding

Python exercises

Python resources for blockmodeling and structural equivalence

Virtually non-existent (there is some functions for collapsing blockmodels)
R is the go-to environment for role-analysis (sna, blockmodeling etc)

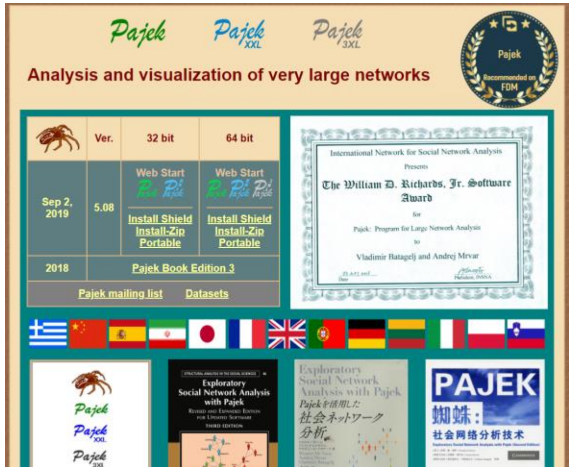
For this course, I have however coded together functionality for doing indirect structural equivalence (although I'm not an active Python user)

Notebook:
Lab_Blockmodeling.ipynb

Module:
blockmodeling.py

/bm_examples : folder with data

Other software resources for Blockmodeling



Pajek

Developed by Ljubljana scholars: fast, complicated to use, direct blockmodeling
http://mrvar.fdv.uni-lj.si/pajek/

R

Also Ljubljana: Ziberna's valued blockmodeling approach but also other related direct/indirect approaches:
https://cran.r-project.org/web/packages/blockmodeling/index.html

Ucinet

Borgatti/Everett/Freeman – classic piece of software (Fortran…). Contains many indirect approaches:
https://sites.google.com/site/ucinetsoftware/home

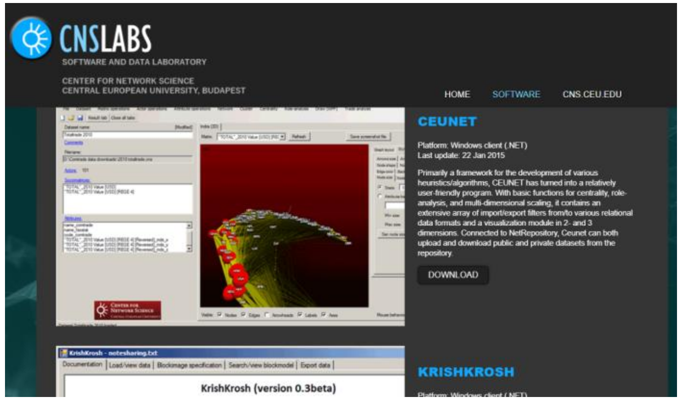
Other software resources for Blockmodeling

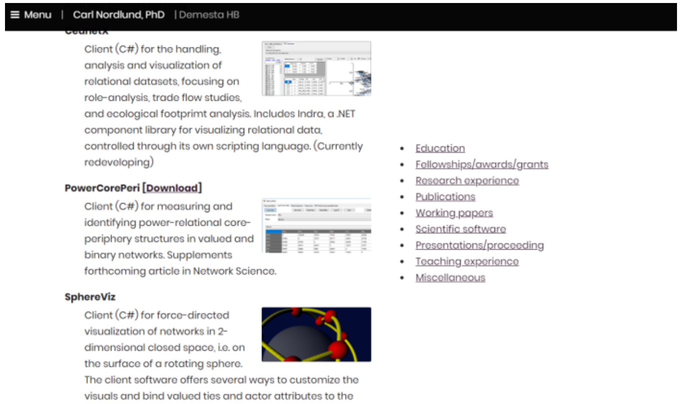
KrishKrosh, BEblocker, Ceunet

Developed by myself (C#) – fast, not very stable, direct and indirect methods. Ceunet has indirect methods, hierarchical clustering functionality; KrishKrosh/BEblocker has direct methods.

http://cnslabs.ceu.edu/software.php

https://demesta.com/page1.php?page=academia#software





Direct blockmodeling needs more computational power, thus preferably more low-level languages (C/C#)

Thank you!

And for future blockmodeling questions (software, projects, ideas, questions etc):

carl.nordlund@liu.se

or

carl.nordlund71@gmail.com

Linköping University

Institute for Analytical Sociology

analytical.sociology@liu.se

www.liu.se/ias

www.liu.se

LINKÖPING UNIVERSITY