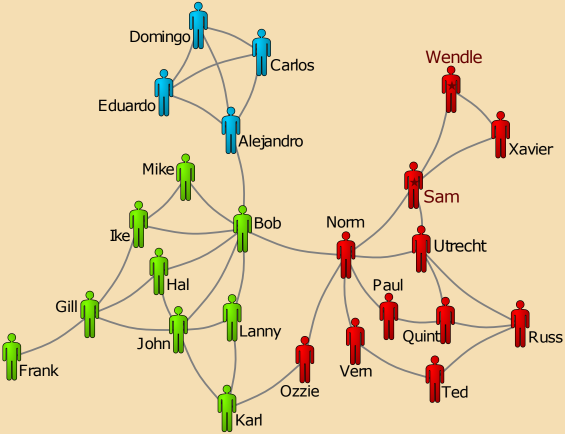
Additional methods for analyzing large networks



Andrej Mrvar

Clustering Coefficient...

Definition...

Clustering coefficient – CC is a ratio between number of links among neighbours of selected vertex and maximum number of links among them (all neighbours are linked among themselves).

CC tells us how much vertices in a network tend to cluster together. In social networks, people want to create tightly connected groups with a high density of links among them.

Let $\deg(v)$ denotes degree of vertex v, $|E(G_1(v))|$ number of links among vertices in 1-neighborhood of vertex v, and MaxDeg maximum degree of vertex in a network.

$$CC_1(v) = \frac{2|E(G_1(v))|}{\deg(v) \cdot (\deg(v) - 1)} \qquad CC_1'(v) = \frac{\deg(v)}{\mathrm{MaxDeg}} CC_1(v)$$

If $\deg(v) \leq 1$ all clustering coefficients for vertex v get missing value (999999998).

...Clustering Coefficient...

...Definition

$CC_1(v)$ is *Clustering Coefficient* as defined by Watts and Strogatz, 1998. In the following examples we will use only $CC_1(v)$.
This is a **local** clustering coefficent: it tells us for each vertex how close its neighbours are to being a clique.

Network / Create Vector / Clustering Coefficients / CC1

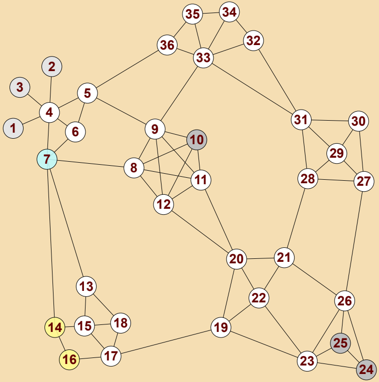
In Report window *Watts-Strogatz Clustering Coefficient* and *Network Clustering Coefficient (Transitivity)* are also displayed. The first is unweighted average and the second is weighted average of local clustering coefficients.

We will use the second one:
The *Network Clustering Coefficient or Transitivity* of a network is the proportion of all two-paths in the network that are closed.

This is a **global** CC - an indication of the clustering in the network.

...Clustering Coefficient...

Example and explanation



In the network above (shr1.net) vertices v_{10}, v_{24}, and v_{25} have the highest possible clustering coefficient, $CC_1 = 1$ (all their neighbours are connected among themselves); vertices v_{14}, and v_{16} have the lowest possible clustering coefficient, $CC_1 = 0$ (no lines among neighbours); vertex v_7 has the next smallest $CC_1 = 0.1$ (1 line among 5 vertices with 10 possible lines); for vertices v_1, v_2, and v_3 CC_1 cannot be computed.

...Clustering Coefficient

Transitivity revisited

We compute average (arithmetic mean) of all local clustering coefficients (without missing values) using **Vector / Info**. We get 0.451. That is unweighted or Watts-Strogatz Clustering Coefficient.

We have already talked about transitivity when we introduced *triadic census*. For the example network we get 30 transitive and 129 intransitive triads. Weighted clustering coefficient, also called Network Clustering Coefficient (Transitivity) is computed in the following way (complete triads must be counted three times):

$$\frac{30 * 3}{30 * 3 + 129} = 0.411$$

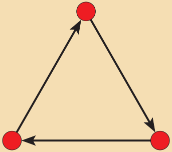
We will use Network Clustering Coefficient (Transitivity).

Short Cycles...

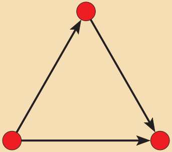
Three Rings

How many times each line belongs to predefined three ring. Ring counts are stored as line values.
Network / Create New Network / with Ring Counts stored as Line Values / 3-Rings

- **Undirected** – for undirected networks – count undirected 3-rings.
- **Directed** – for directed networks – count **cyclic**, **transitive**, or all 3-rings, or count how many times each line is a transitive shortcut.



cyclic



transitive

Examples: flow2.net (directed), shr1.net (undirected).

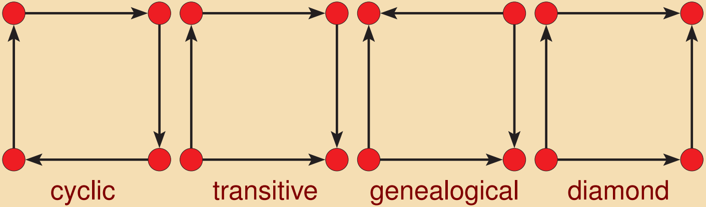
...Short Cycles

Four Rings

How many times each line belongs to predefined four ring. Ring counts are stored as line values.

Network / Create New Network / with Ring Counts stored as Line Values / 4-Rings

- **Undirected** – for undirected networks – count undirected 4-rings.
- **Directed** – for directed networks – count **cyc**lic, **dia**monds, **gen**ealogical, **tra**nsitive, or all 4-rings, or count how many times each line is a transitive shortcut.



Note: in 2-mode network 3-rings cannot exist, only 4-rings exist.

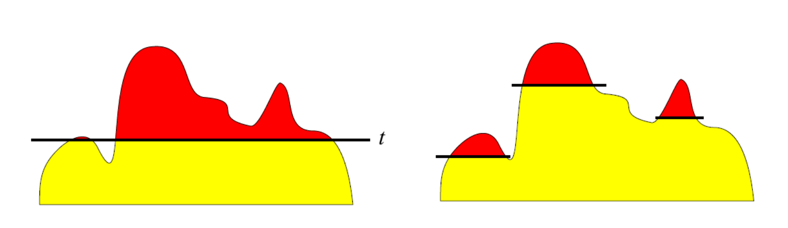
Examples: flow2 (directed), shr1 (undirected), Davis (2-mode).

Islands...

If we represent a given or computed value of vertices / lines as a height of vertices / lines and we sink the network into a water up to selected level we get islands. Varying the level we get different islands.

Usually we want to find islands which are not too large and not too small - we provide minimum and maximum islands size.

Searching for islands starts with complete network sunk into the water. Then we lower the level of water until we get islands of selected size...



...Islands

Line Islands:
Network / Create Partition / Islands / Line Weights

For line islands we need network with values on lines (weighted network). If the original network is unweighted we can obtain values on lines using counting 3 or 4 rings, or (in case of 2-mode network) we can transform a two mode network into two weighted one mode networks.
Example: shr1.net, Davis.net (2-mode)

Vertex Islands:
Operations / Network + Vector / Islands / Vertex Weights

We need a Vector describing some properties of vertices as additional input.

Community Detection...

Communities - dense clusters for which there are more lines inside clusters than among clusters (values of lines are taken into account too).

In Pajek two community detection methods are available: **Louvain method** and **VOS Clustering**

When applying **Louvain method** we search for partition into clusters with the highest value of **modularity**. Modularity is defined in the following way:

$$Q = \frac{1}{2m} \sum_{s} (e_s - r * \frac{K_s^2}{2m})$$

- m – total number of lines in network,
- s – cluster (community),
- $e_s = \sum_{ij \in s} A_{ij}$ – 2 times the number of lines in community s
- $K_s = \sum_{i \in s} k_i$ – sum of degrees in community s
- r – **resolution parameter**, default value 1 means modularity as originally defined

Similar method is **VOS Clustering**, where **VOS quality function** is taken into account instead of modularity.

...Community Detection

Pajek command: **Network / Create Partition / Communities**
Several parameters can be changed, but they are important only when analysing very large networks.

By changing *resolution parameter - r* we can get larger or smaller communities. By default resolution parameter is set to 1. Setting *r* larger than 1 means searching for *larger number* of *smaller communities*. Setting *r* smaller than 1 means searching for *smaller number* of *larger communities*.

Differences between *Community detection* and *Islands*:

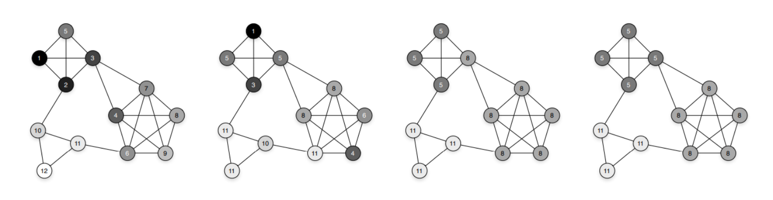
- Community detection methods (unlike Islands) work also on uweighted networks.
- As result of community detection each vertex is assigned to one cluster (community). It is partition or 'classification'.
- When searching for islands 'high enough' vertices are assigned to clusters (islands), other vertices are assigned to cluster 0 (not assigned to any island).

Example: shr1.net, Davis.net (2-mode)

Label Propagation

Label propagation algorithm assigns labels (clusters, communities) to previously unlabeled vertices. At the start of the algorithm, each vertex has its own label (is in its own cluster $C_i = i$). Then labels are propagated in random order over the network in several iterations. In each iteration of propagation, each vertex updates its label to the one that the maximum numbers of its neighbours belongs to. Algorithm is finished when iteration finishes without any updates of labels. This is the fastest algorithm.

Example (by Lovro Šubelj):



Pajek: **Network / Create Partition / Label Propagation / Fast**
Example: shr1.net

E-I Index

Simple measure of how well clusters divide a network into cohesive subgroups is the **E-I Index: External-Internal Index**.
The E-I Index subtracts the number of lines within clusters (I) from the number of lines between the clusters (E). The difference is divided by the total number of lines ($E + I$). Line values can be taken into account.

$$Index_{E-I} = \frac{E - I}{E + I}$$

Values of the E-I Index range from -1 to 1. If the E-I Index is -1, all lines are inside clusters whereas the value 1 means that all lines are between clusters. The value 0 indicates that the number of lines (or the sum of line values) between clusters equals the number of lines (or sum of line values) inside clusters.
If a partition classifies vertices into cohesive groups well, lines should be within clusters rather than between clusters, so the E-I Index should be negative and preferably close to -1.

Operations / Network + Partition / Info / E-I Index

Comparing partitions...

Cramer coefficient

Communities obtained by different methods are usually very similar. We can check this by
Partitions / Info
which computes **Cramer**, **Adjusted Rand** and **Rajski coefficients** for comparison of two partitions.

Several measures can be computed to compare two partitions which represent nominal properties (usually represented as a contingency table):

Partitions / Info / Cramer's V, Rajski, Adjusted Rand Index

Two nominal variables can be compared using Cramer coefficient (*Cramer's V*).

$$V = \sqrt{\frac{\chi^2}{n(k-1)}}$$

...Comparing partitions...

Rajski coefficients...

Rajski coefficient (1964) is constructed using *enthropy*:
Lets take two nominal variables X and Y. Variable X has n different values, variable Y has m different values.

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

$$H(Y) = -\sum_{i=1}^{m} p(y_i) \log_2 p(y_i)$$

and

$$H(XY) = -\sum_{i=1}^{n} \sum_{j=1}^{m} p(x_i, y_j) \log_2 p(x_i, y_j)$$

...Comparing partitions...

...Rajski coefficients...

Additional methods

Clustering Coefficient

Short Cycles

Islands

Communities

Label Propagation

E-I Index

Comparing partitions

Assignments

Information between variables X and Y is defined in the following way:

$$I(X, Y) = H(X) + H(Y) - H(XY)$$

Information $I(X, Y)$ gets value 0, when it holds for each pair x_i and y_j: $p(x_i, y_j) = p(x_i)p(y_j)$, what means, that the two variables are independant.

Information $I(X, Y)$ gets maximum value, when functional relationship exists between the two variables – in each row and each column of corresponding contingency table there is at most one non zero element. Then it holds:

$$H(X) = H(Y) = H(XY) = I(X, Y)$$

Information I(X,Y) is a measure of functional dependancy between X and Y.

...Comparing partitions...

...Rajski coefficients...

Rajski coefficients:

$$R(X \leftrightarrow Y) = \frac{I(X,Y)}{H(XY)}$$

$$R(X \rightarrow Y) = \frac{I(X,Y)}{H(Y)}$$

$$R(X \leftarrow Y) = \frac{I(X,Y)}{H(X)}$$

All three coefficients get values in range 0 to 1. Value 0 means that variables are undependant.
$R(X \rightarrow Y) = 1$, when Y is a function of X,
$R(X \leftarrow Y) = 1$, when X is a function of Y in
$R(X \leftrightarrow Y) = 1$, when variables determine each other (in both directions).

...Comparing partitions...

...Rajski coefficients...

Example 1:

	y_1	y_2	y_3	Sum
x_1	2	2	1	5
x_2	2	1	2	5
Sum	4	3	3	10

$p(x_i, y_j)$	y_1	y_2	y_3	$p(x_i)$
x_1	0.2	0.2	0.1	0.5
x_2	0.2	0.1	0.2	0.5
$p(y_j)$	0.4	0.3	0.3	1

$$R(X \leftrightarrow Y) = 0.0194$$

$$R(X \rightarrow Y) = 0.0312$$

$$R(X \leftarrow Y) = 0.0490$$

All three coefficients are low, we cannot predict the value of one variable if we know the value of the other.

...Comparing partitions...

...Rajski coefficients...

Example 2:

	y_1	y_2	y_3	Sum
x_1	0	3	0	3
x_2	4	0	3	7
Sum	4	3	3	10

$p(x_i, y_j)$	y_1	y_2	y_3	$p(x_i)$
x_1	0	0.3	0	0.3
x_2	0.4	0	0.3	0.7
$p(y_j)$	0.4	0.3	0.3	1

$$R(X \leftrightarrow Y) = 0.5610$$

$$R(X \rightarrow Y) = 0.5610$$

$$R(X \leftarrow Y) = 1$$

Variable X is a function of Y: if we know value of Y we can predict value of X.
However variable Y is not a function of X.

...Comparing partitions...

...Rajski coefficients...

Example 3:

	y_1	y_2	y_3	Sum
x_1	4	0	0	4
x_2	0	0	3	3
x_3	0	3	0	3
Sum	4	3	3	10

$$R(X \leftrightarrow Y) = 1$$

$$R(X \rightarrow Y) = 1$$

$$R(X \leftarrow Y) = 1$$

Variable X is a function of Y. Variable Y is a function of X.

...Comparing partitions...

...Rajski coefficients

Example 4:

	y_1	y_2	Sum
x_1	2	2	4
x_2	2	2	4
Sum	4	4	8

$$R(X \leftrightarrow Y) = 0$$

$$R(X \rightarrow Y) = 0$$

$$R(X \leftarrow Y) = 0$$

Variables X and Y are independant.

...Comparing partitions...

Adjusted Rand Index

Additional methods

Clustering Coefficient

Short Cycles

Islands

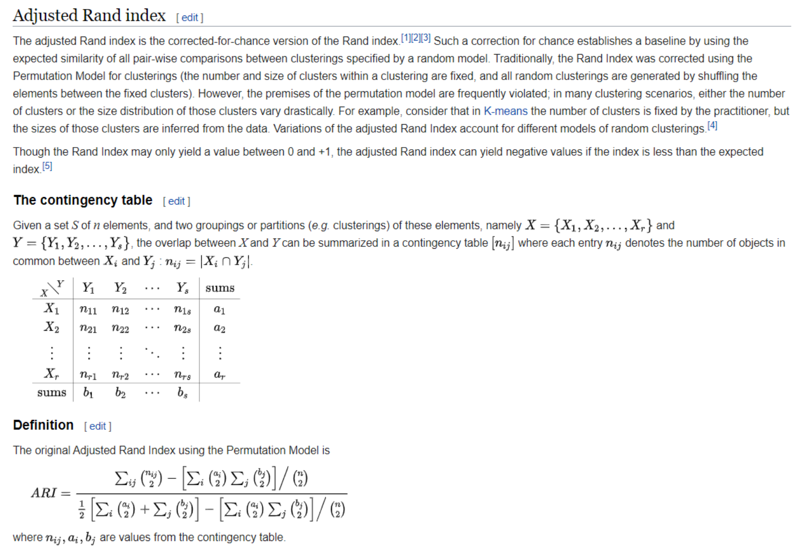
Communities

Label Propagation

E-I Index

Comparing partitions

Assignments



Adjusted Rand index [edit]

The adjusted Rand index is the corrected-for-chance version of the Rand index.[1][2][3] Such a correction for chance establishes a baseline by using the expected similarity of all pair-wise comparisons between clusterings specified by a random model. Traditionally, the Rand Index was corrected using the Permutation Model for clusterings (the number and size of clusters within a clustering are fixed, and all random clusterings are generated by shuffling the elements between the fixed clusters). However, the premises of the permutation model are frequently violated; in many clustering scenarios, either the number of clusters or the size distribution of those clusters vary drastically. For example, consider that in K-means the number of clusters is fixed by the practitioner, but the sizes of those clusters are inferred from the data. Variations of the adjusted Rand Index account for different models of random clusterings.[4]

Though the Rand Index may only yield a value between 0 and +1, the adjusted Rand index can yield negative values if the index is less than the expected index.[5]

The contingency table [edit]

Given a set S of n elements, and two groupings or partitions (*e.g.* clusterings) of these elements, namely $X = \{X_1, X_2, \ldots, X_r\}$ and $Y = \{Y_1, Y_2, \ldots, Y_s\}$, the overlap between X and Y can be summarized in a contingency table $[n_{ij}]$ where each entry n_{ij} denotes the number of objects in common between X_i and Y_j : $n_{ij} = |X_i \cap Y_j|$.

$X \backslash Y$	Y_1	Y_2	$\cdots$	Y_s	sums
X_1	n_{11}	n_{12}	$\cdots$	n_{1s}	a_1
X_2	n_{21}	n_{22}	$\cdots$	n_{2s}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
X_r	n_{r1}	n_{r2}	$\cdots$	n_{rs}	a_r
sums	b_1	b_2	$\cdots$	b_s	

Definition [edit]

The original Adjusted Rand Index using the Permutation Model is

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i\binom{a_i}{2}\sum_j\binom{b_j}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_i\binom{a_i}{2}+\sum_j\binom{b_j}{2}\right] - \left[\sum_i\binom{a_i}{2}\sum_j\binom{b_j}{2}\right]/\binom{n}{2}}$$

where n_{ij}, a_i, b_j are values from the contingency table.

https://en.wikipedia.org/wiki/Rand_index

...Comparing partitions

Example

If two partitions are exactly the same, all three measures (Cramer's V, Rajski, and Adjusted Rand Index) return the same result - the two partitions are equal (coefficients are equal to 1).

However if one partition is a subpartition of the other (one or more clusters are divided to smaller groups) Cramer's V coefficient will report value 1, while other indices will recognize the difference (one partition is refinement of the other).

Example:
Try for example **shr1.net**, compute Louvain communities with resolution parameter once 1 and once 1.5 and compute all three indices.

Assignments...

Load **usair97.net**.

Line values in this network represent geographical distances among airports which we do not need (even more: taking these values into account results would be wrong) so remove them before doing any analyses:
Network / **Create New Network** / **Transform** / **Line Values** / **Set All Line Values to 1**

1. **Communities**
   1. Search for *Louvain* and *VOS communities* with resolution parameter 1.
   2. Compute how well the obtained communities divide network into cohesive subgroups (E-I index).
   3. Compare partitions obtained by Louvain and VOS method with different indices (Cramer, Rajski, Adjusted Rand).
   4. Repeat assignment with resolution parameter once larger and once smaller than 1.

...Assignments

2. Run **Label Propagation** algorithm.
3. **Line Islands**
    1. Compute undirected *3-rings* and find Line Islands in the obtained network.
    2. Compute undirected *4-rings* and find Line Islands.
    3. Compare partitions on islands obtained on network with 3-rings and 4-rings (Cramer, Rajski, Adjusted Rand).

Load **USCounties.net** (network on neighbour counties in US) and repeat assignments 1..3.