Community Detection Models

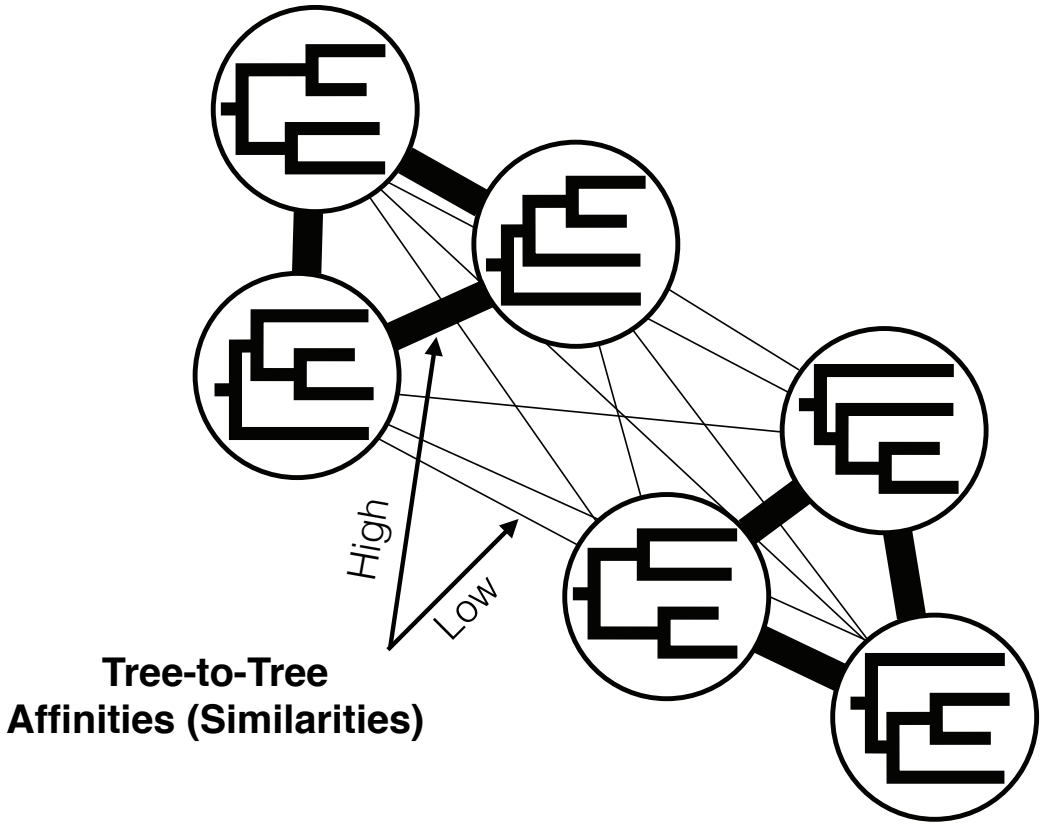
CloudForest Workshop, Mexico City, January 2023

What is Community Detection?

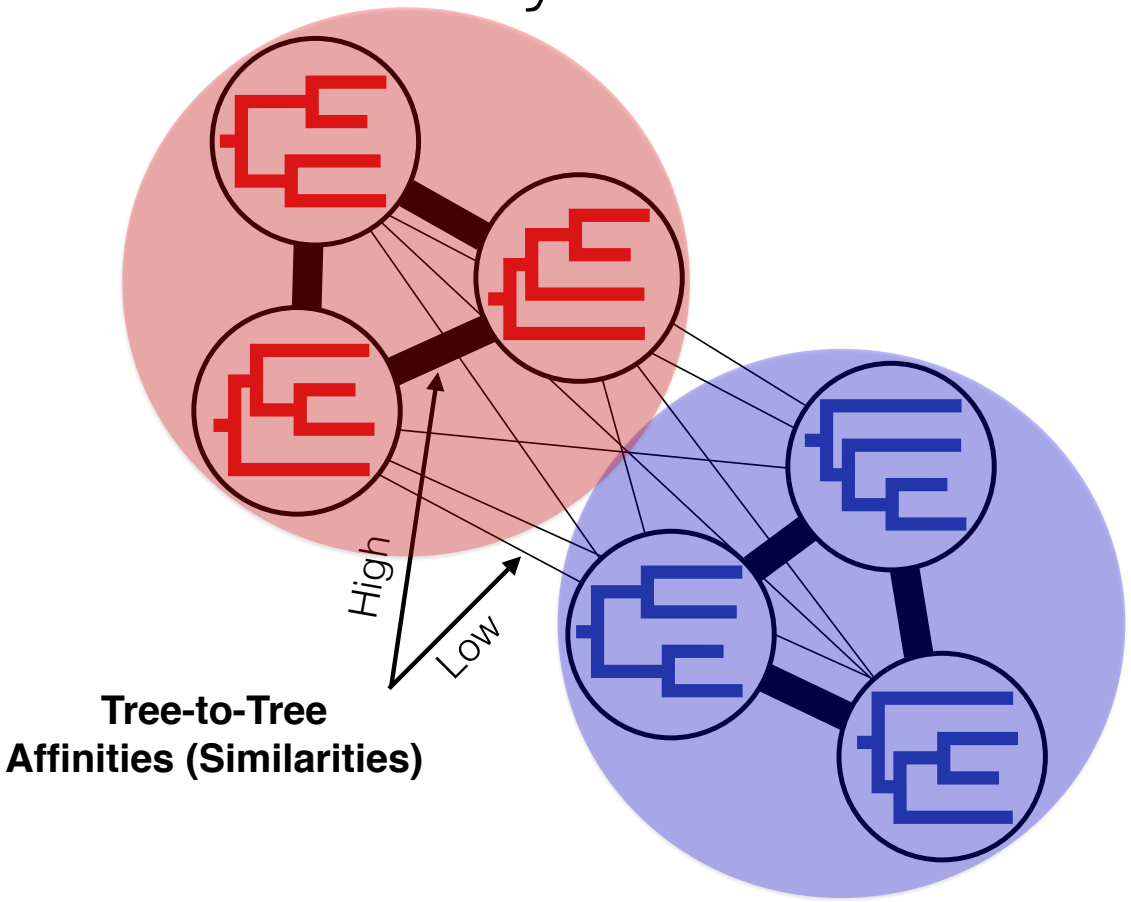
Community detection attempts to **identify groups of nodes**, known as communities, that are more **tightly connected** (by edges with large, positive weights) to each other than they are to the rest of the nodes in the network, with the goal of detecting the **natural structure** of a network.

The number and size of these communities does not need to be specified *a priori*.

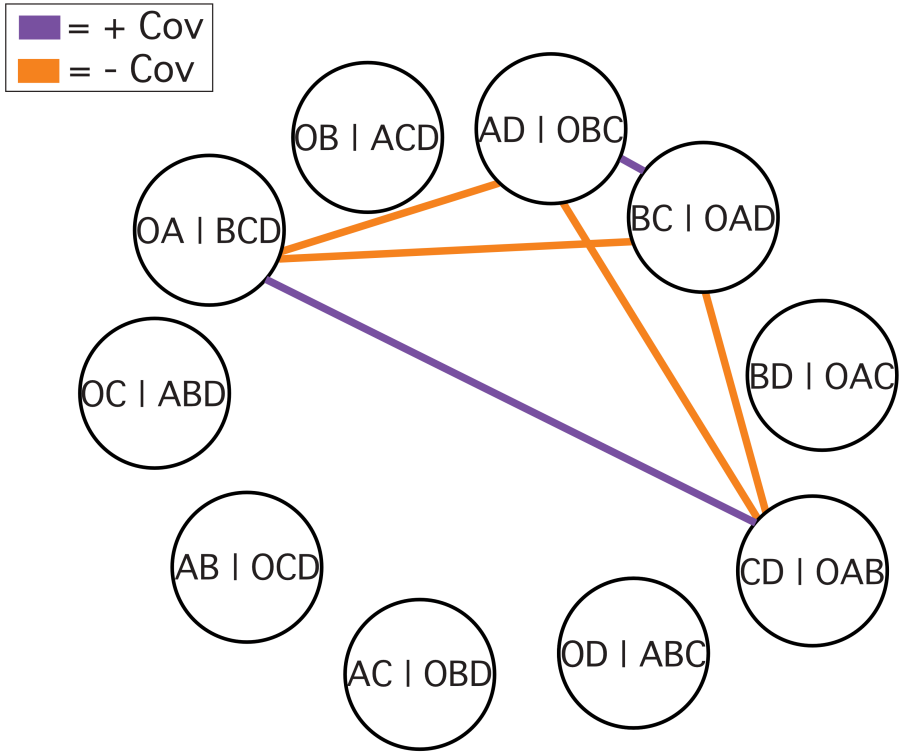
Community Detection in Tree-Based Affinity Networks



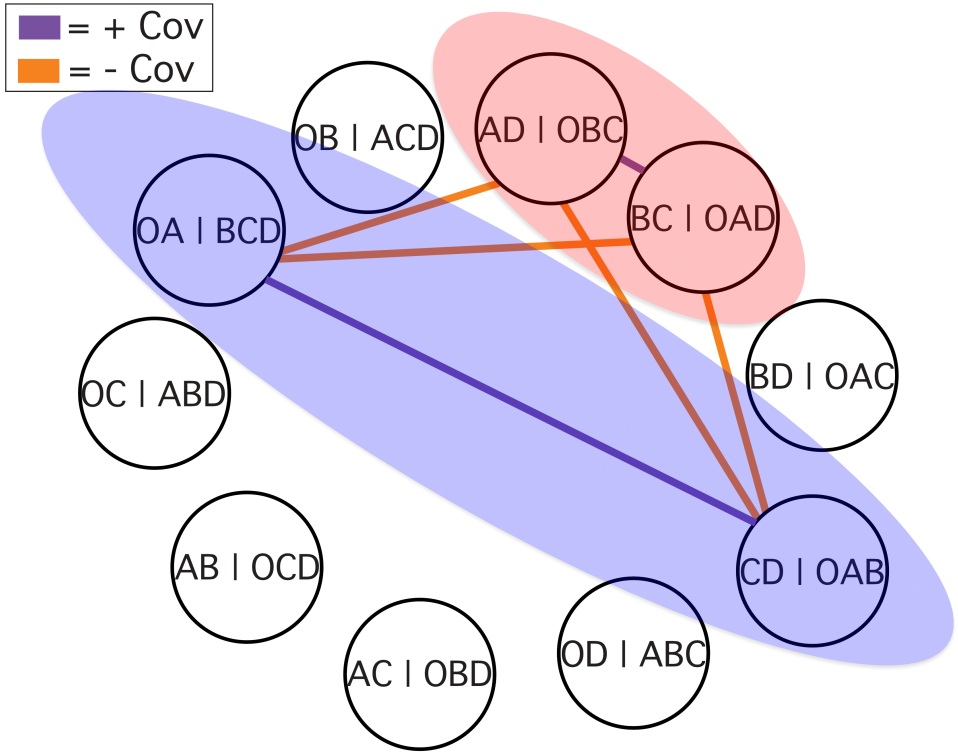
Community Detection in Tree-Based Affinity Networks



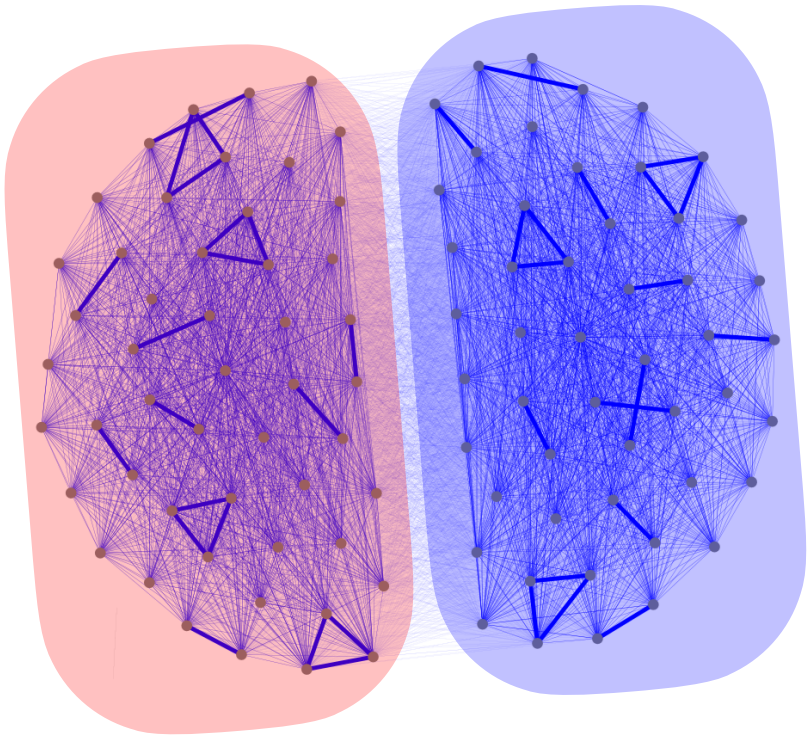
Community Detection in Bipartition Covariance Networks



Community Detection in Bipartition Covariance Networks



Community Detection Performs Well with Simulations



Set of 25-taxon trees simulated to have two distinct signals, but with varying levels of variation and divergence.

Modularity

Community detection methods rely on minimizing a function called the “modularity”.

Essentially, this is a cost function.

The community structure with the lowest cost is preferred.

Community Detection Models 1

No Null Model (NNM)

$$Q(\{\sigma\}) = \sum_{i,j} A_{i,j}\delta(\sigma_i, \sigma_j)$$

Community Detection Models 1

No Null Model (NNM)

$$Q(\{\sigma\}) = \sum_{i,j} A_{i,j}\delta(\sigma_i, \sigma_j)$$

$$Q(\{\sigma\}) = Modularity$$

Community Detection Models 1

No Null Model (NNM)

$$Q(\{\sigma\}) = \sum_{i,j} A_{i,j}\delta(\sigma_i, \sigma_j)$$

$A_{i,j} =$ "Adjacency" (i.e., edge weight) between *i* and *j*

Community Detection Models 1

No Null Model (NNM)

$$Q(\{\sigma\}) = \sum_{i,j} A_{i,j}\delta(\sigma_i, \sigma_j)$$

Q is “rewarded” for putting large edge weights in the same community.

Community Detection Models 1

No Null Model (NNM)

$$Q(\{\sigma\}) = \sum_{i,j} A_{i,j}\delta(\sigma_i, \sigma_j)$$

Q is “rewarded” for putting large edge weights in the same community.

However, NNM tends to put **all nodes together into one big community**!

Community Detection Models 1

No Null Model (NNM)

$$Q(\{\sigma\}) = \sum_{i,j} A_{i,j}\delta(\sigma_i, \sigma_j)$$

NNM less useful in practice.

Community Detection Models 2

Erdos-Rényi Null Model (ERNM)

$$Q(\{\sigma\}) = \sum_{i,j} [A_{i,j} - (p_{i,j}^{+}\lambda^{+} - p_{i,j}^{-}\lambda^{-})]\delta(\sigma_i, \sigma_j)$$

Community Detection Models 2

Erdos-Rényi Null Model (ERNM)

$$Q(\{\sigma\}) = \sum_{i,j} [A_{i,j} - (p_{i,j}^{+}\lambda^{+} - p_{i,j}^{-}\lambda^{-})]\delta(\sigma_i, \sigma_j)$$

Relies on the notion of a “random” network to compare against.

Community Detection Models 2

Erdos-Rényi Null Model (ERNM)

$$Q(\{\sigma\}) = \sum_{i,j} [A_{i,j} - (p^{+}_{i,j}\lambda^{+} - p^{-}_{i,j}\lambda^{-})]\delta(\sigma_i, \sigma_j)$$

$p^{+}_{i,j} =$ Prob. connecting *i* and *j* with a + edge

$p^{-}_{i,j} =$ Prob. connecting *i* and *j* with a - edge

Community Detection Models 2

Erdos-Rényi Null Model (ERNM)

$$Q(\{\sigma\}) = \sum_{i,j} [A_{i,j} - (p_{i,j}^{+}\lambda^{+} - p_{i,j}^{-}\lambda^{-})]\delta(\sigma_i, \sigma_j)$$

$\lambda^{+} =$ Tuning parameter for weighting of + edges

$\lambda^{-} =$ Tuning parameter for weighting of - edges

Community Detection Models 2

Erdos-Rényi Null Model (ERNM)

$$Q(\{\sigma\}) = \sum_{i,j} [A_{i,j} - (p_{i,j}^{+}\lambda^{+} - p_{i,j}^{-}\lambda^{-})]\delta(\sigma_i, \sigma_j)$$

Performance of the ERNM depends on how well the degree distribution of nodes (sets of edge weights) is approximated by the random null model. May not work well if approximation is poor.

Community Detection Models 2

Erdos-Rényi Null Model (ERNM)

$$Q(\{\sigma\}) = \sum_{i,j} [A_{i,j} - (p_{i,j}^{+}\lambda^{+} - p_{i,j}^{-}\lambda^{-})]\delta(\sigma_i, \sigma_j)$$

Tuning parameters don't necessarily have obvious defaults and difficult to estimate both simultaneously. Therefore, we fix one value and estimate the other using a heuristic strategy (we'll cover this in a minute).

Community Detection Models 3

Configuration Null Model (CNM)

$$Q(\{\sigma\}) = \sum_{i,j} [A_{i,j} - (\frac{k_i^+ k_j^+}{2m^+}\lambda^+ - \frac{k_i^- k_j^-}{2m^-}\lambda^-)]\delta(\sigma_i, \sigma_j)$$

Like the ERNM, the CNM uses a null model, but this one is based on the observed degree distribution of the network.

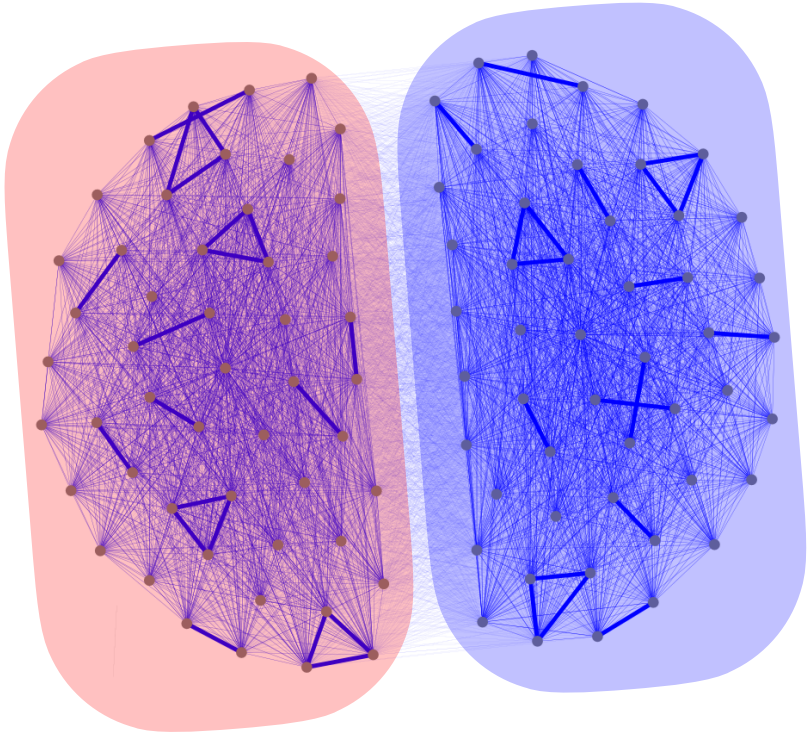
In other words, the null model is more sophisticated.

Community Detection Models 3

Configuration Null Model (CNM)

$$Q(\{\sigma\}) = \sum_{i,j} [A_{i,j} - (\frac{k_i^+ k_j^+}{2m^+}\lambda^+ - \frac{k_i^- k_j^-}{2m^-}\lambda^-)]\delta(\sigma_i, \sigma_j)$$

We have found that the CNM tends to work very well in practice for detecting large-scale structure in networks.



We have found that the CNM tends to work very well in practice for detecting large-scale structure in networks.

Community Detection Models 4

Constant Potts Model (CPM)

$$Q(\{\sigma\}) = \sum_{i,j} [A_{i,j} - (\lambda^{+} - \lambda^{-})]\delta(\sigma_i, \sigma_j)$$

Constant Potts model is fairly simple in form, but inspired by a model of atomic spins in physics.

Community Detection Models 4

Constant Potts Model (CPM)

$$Q(\{\sigma\}) = \sum_{i,j} [A_{i,j} - (\lambda^{+} - \lambda^{-})]\delta(\sigma_i, \sigma_j)$$

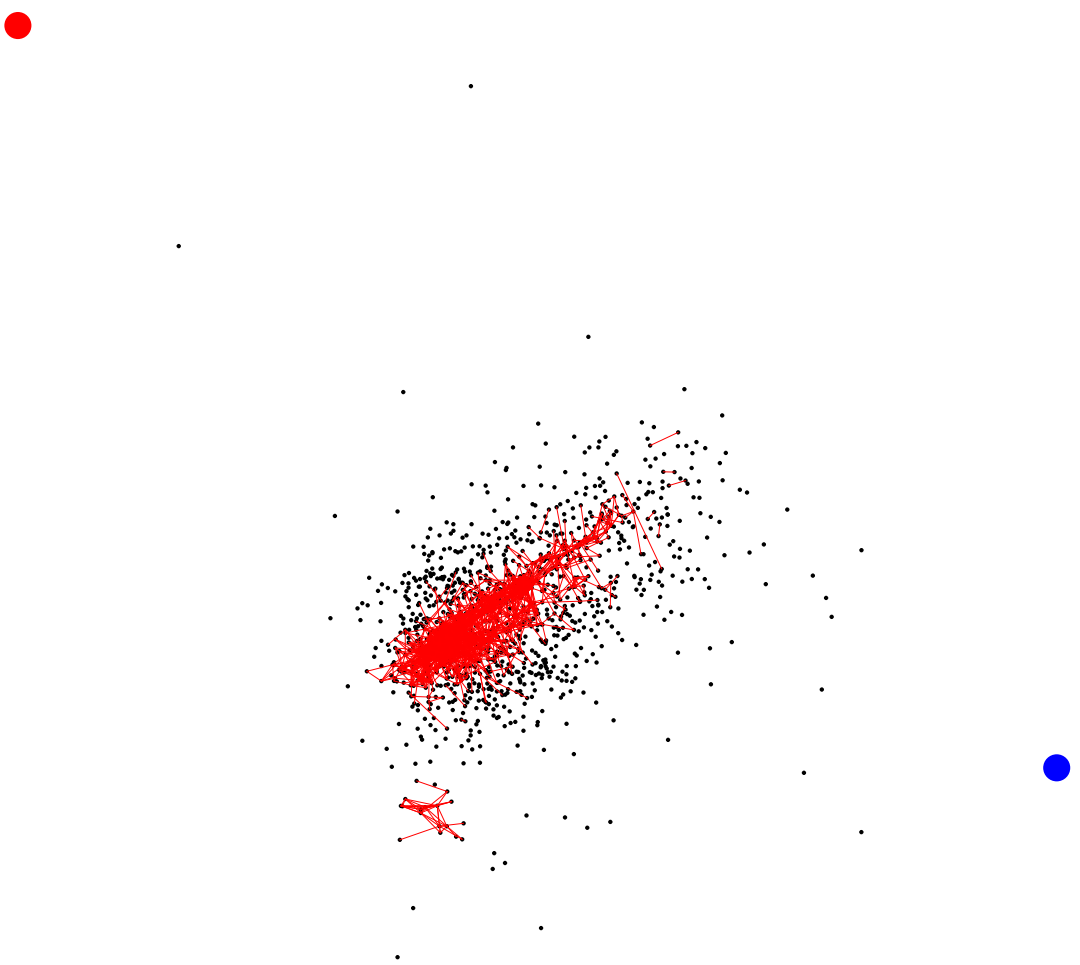
Is a “resolution limit free” model, which means it can simultaneously detect communities that differ greatly in size.

Community Detection Models 4

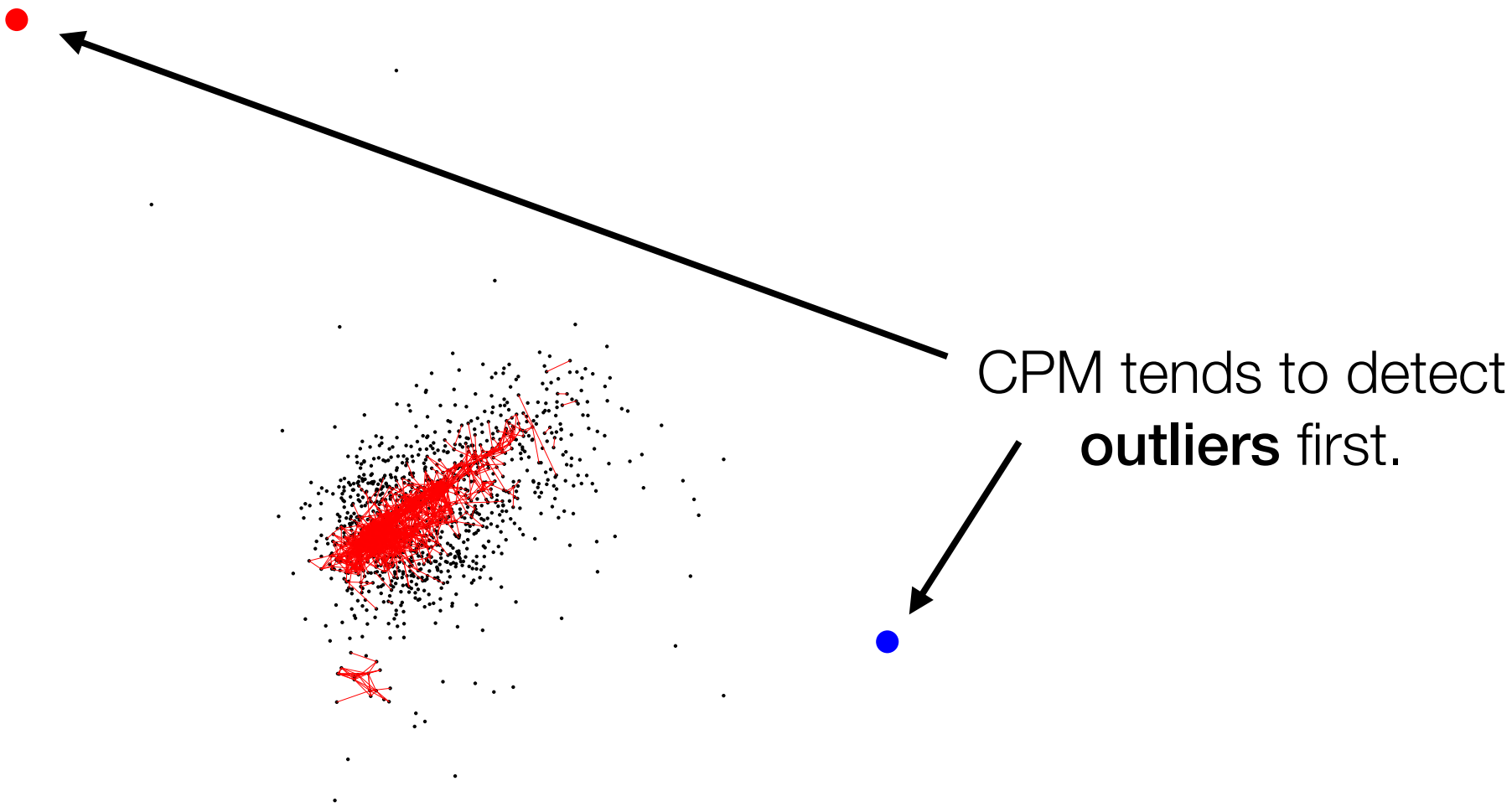
Constant Potts Model (CPM)

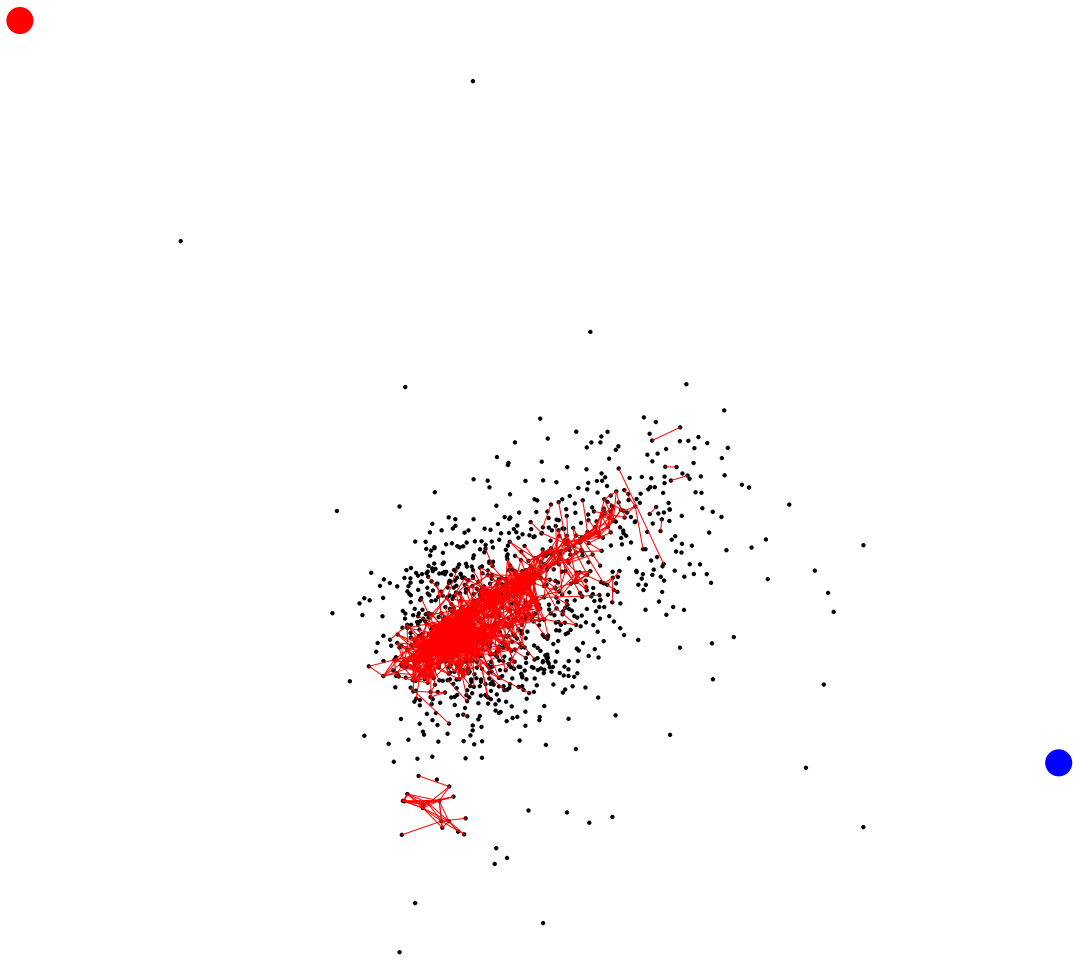
$$Q(\{\sigma\}) = \sum_{i,j} [A_{i,j} - (\lambda^{+} - \lambda^{-})]\delta(\sigma_i, \sigma_j)$$

In practice, we have found this model to be very useful, but primarily for **detecting "outlier" trees**.



CPM tends to detect **outliers** first.


CPM tends to detect
outliers first.

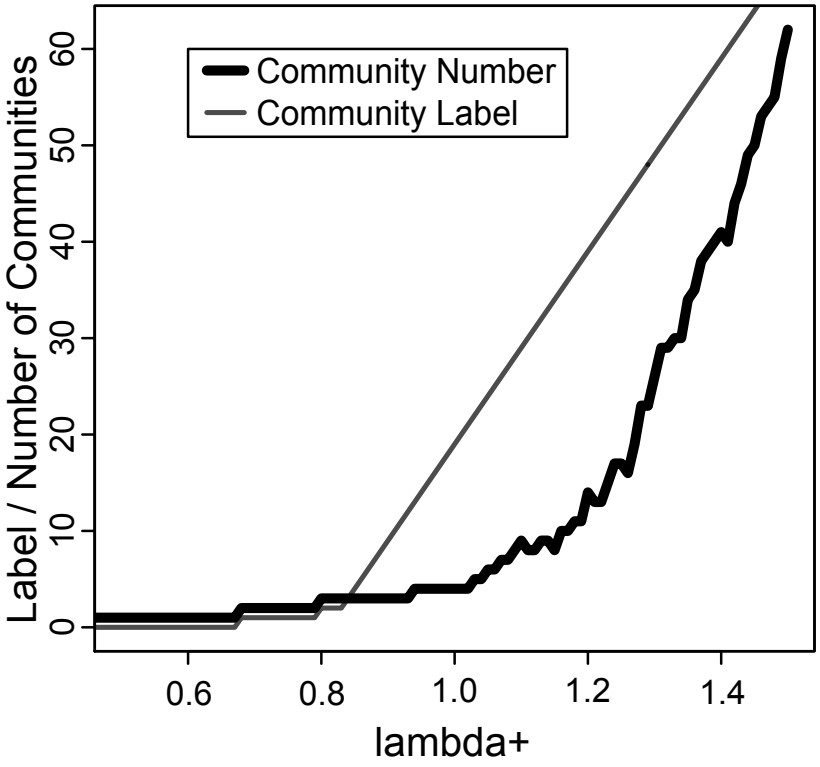


By the way, this is an NLDR plot, but with network edges connecting trees.

Not native to CloudForest, but fairly easy to code in R with CloudForest output.

Tuning Parameter Choice

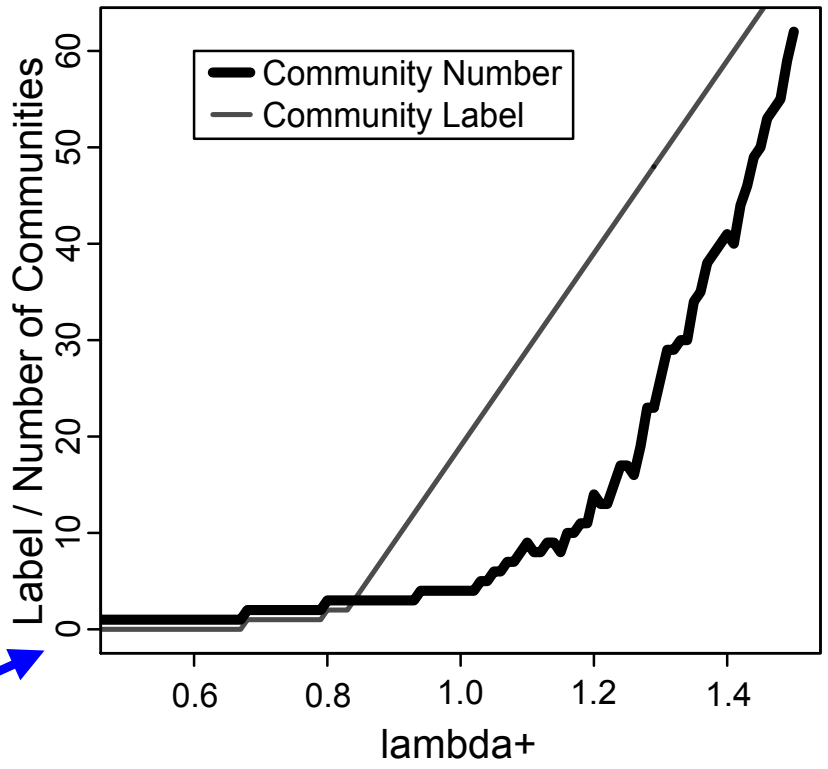
Plateau Analysis



Tuning Parameter Choice

Plateau Analysis

All nodes in different communities

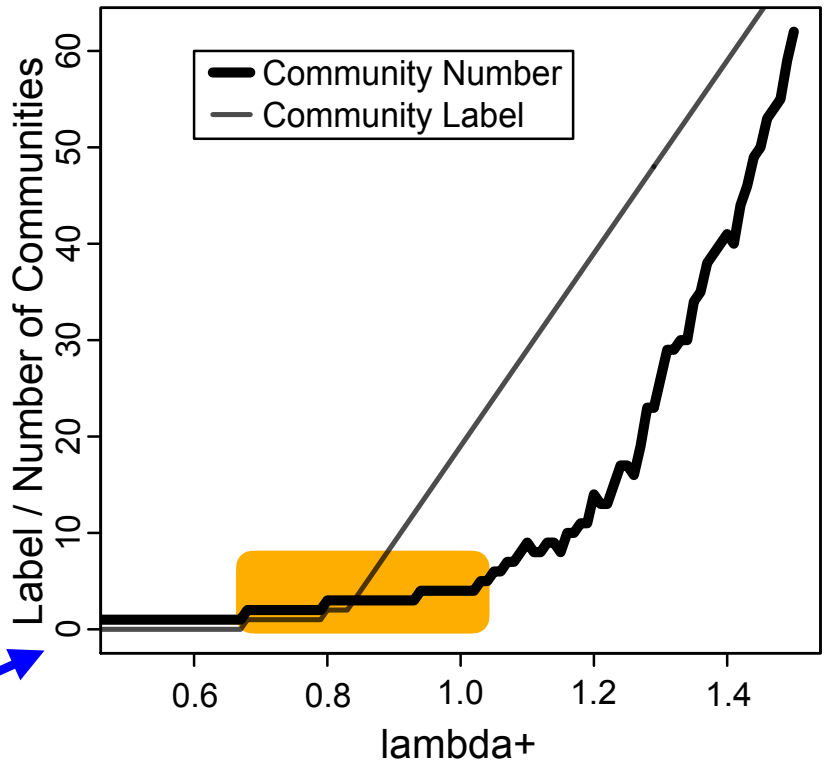


All nodes in the same community

Tuning Parameter Choice

Plateau Analysis

All nodes in different communities

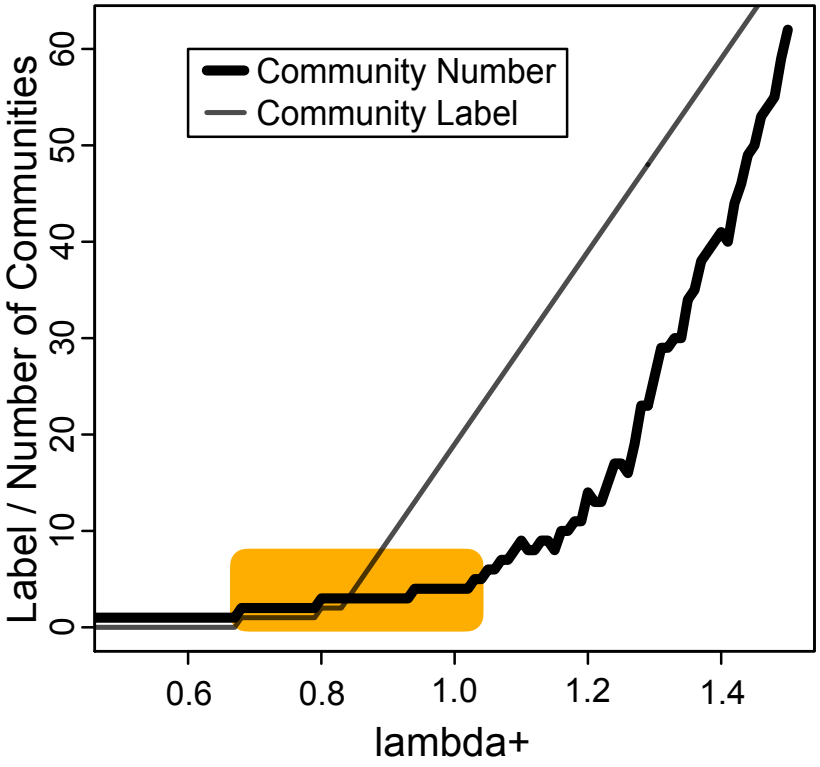


All nodes in the same community

Tuning Parameter Choice

Plateau Analysis

Plateaus indicate ranges of tuning parameter values where community number or structure stays the same.



There may be several “natural” community structures.

CloudForest can display community structure for trees on an NLDR plot, even though the structure was detected with the full network and not the projected (i.e., distorted) distances.

A slider can be used to move across different tuning parameters and see how community structure changes. Explore your data!

End Community Detection