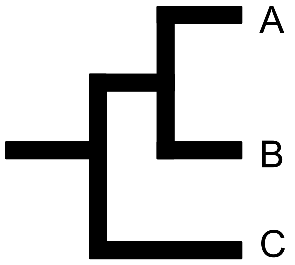
CloudForest Workshop

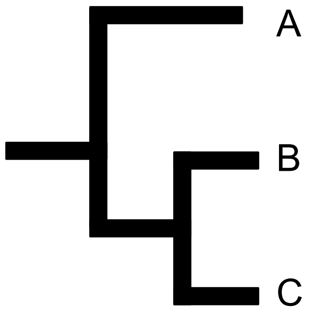
Tree-to-Tree Distances

Comparing phylogenetic trees

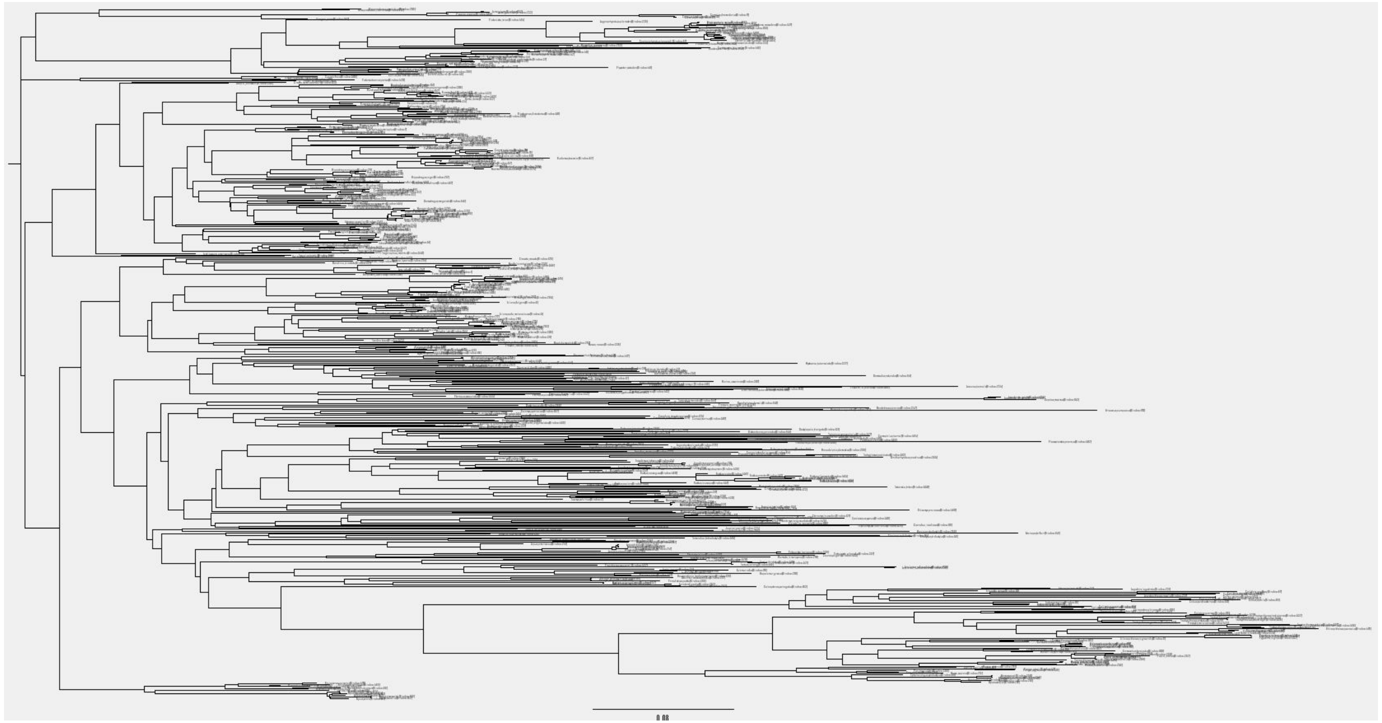
- Phylogenetic trees contain a vast amount of information
  - Topology, branch lengths, etc

- Easily compared at low numbers of taxa





However, at high numbers of taxa….



Comparison becomes a lot more difficult!

Comparing trees numerically

- We need to be able to compare trees at large scales, so having a numeric way to do this becomes very useful!

- Tree-to-tree distances give us a pairwise way to quantify differences between trees

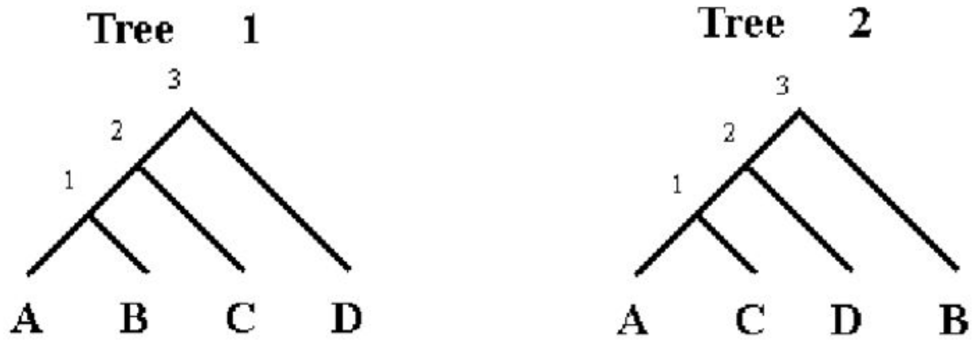
Comparing trees numerically

- Several different metrics for measuring the “distance” between two trees exist

- Robinson-Foulds (RF), Subtree Prune-Regraft (SPR), Matching Splits, Geodesic, etc.
    - At their base level, all operate by quantifying the similarity of two trees

The RF distance

- The most widely used tree-to-tree distance metric
  - Can be modified to accommodate both weighted and unweighted trees (trees containing branch lengths vs no branch lengths)
- Defined as *A* + *B*
  - A = # of partitions implied by Tree 1, but not Tree 2
  - B = # of partitions implied by Tree 2, but not Tree 1



Trees 1 and 2 have 2 clades that do not exist in the other one (excluding the node 3).
Hence the Robinson-Foulds distance is 2 + 2 = 4 in this example.

The RF distance

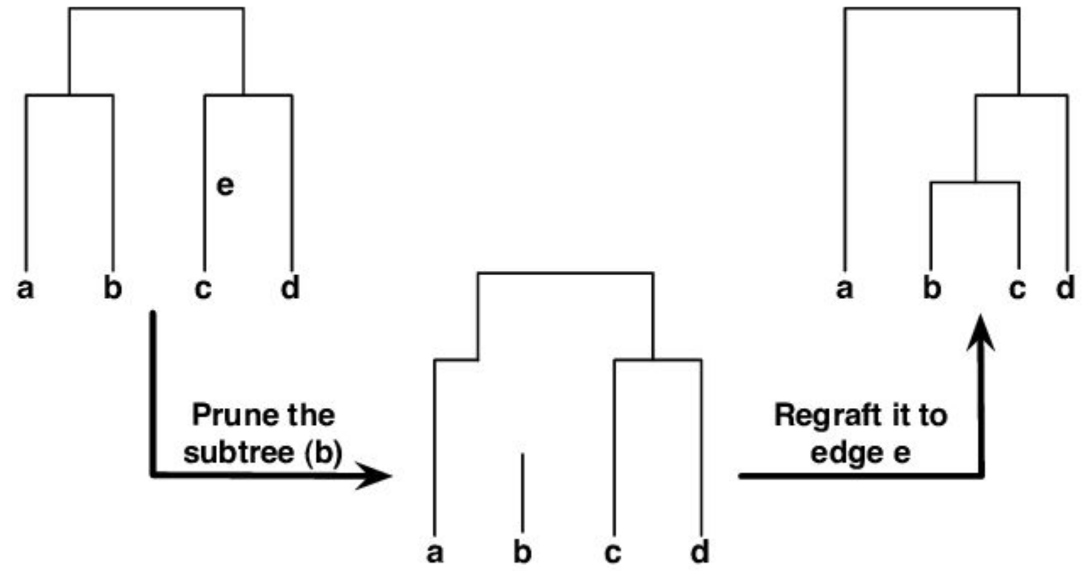
- Simple and effective way to quantify distance
- Number of splits that differ between trees is an intuitive way to quantify their similarity

- Potential shortcomings:
  - Small differences between trees can result in large distances
  - Can occasionally be counterintuitive (i.e. moving one branch can result in a larger distance than moving two

- Still, remains widely used to effectively compare trees due to its simplicity and wide implementation

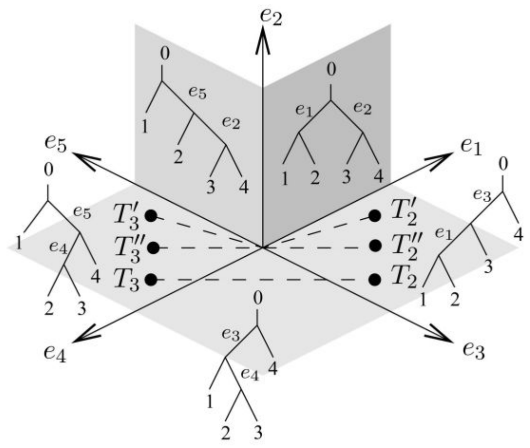
Other distance metrics

- SPR distance
  - SPR moves “prune” a subtree from one area of the tree and “regrafting” it on to another area
  - SPR distance defined as minimum number of moves required to convert one tree into another
  - Slow to compute with large trees



Other distance metrics

- Matching split distance
  - Similar to RF, quantifies similar splits between two trees
  - However, more sensitive than RF (resistant to displacement of a small number of tips)
- Geodesic distance
  - Defined as the shortest "path" between two trees in continuous treespace (highly complex!)
  - The most "natural" distance between trees, but the least well implemented and tested



Discussion

Form small groups or talk with your neighbors!

- How familiar are you with tree-to-tree distances in general?
- If you have used distances in the past, which metric have you used most often?
- Do you think distances are useful for phylogenomic studies?
- What are some interesting ways that you've used distances in your own research?

Calculating distances in CloudForest

- All of the distance metrics described here are available for use in CloudForest, with the exception of the Geodesic distance
    - Coming soon!

- Different metrics perform differently, so trying each out and comparing results can be informative to an analysis!

Calculating distances in CloudForest

- Tree-to-tree distances are a very useful way to begin exploring variation within a treeset, especially in phylogenomic data!

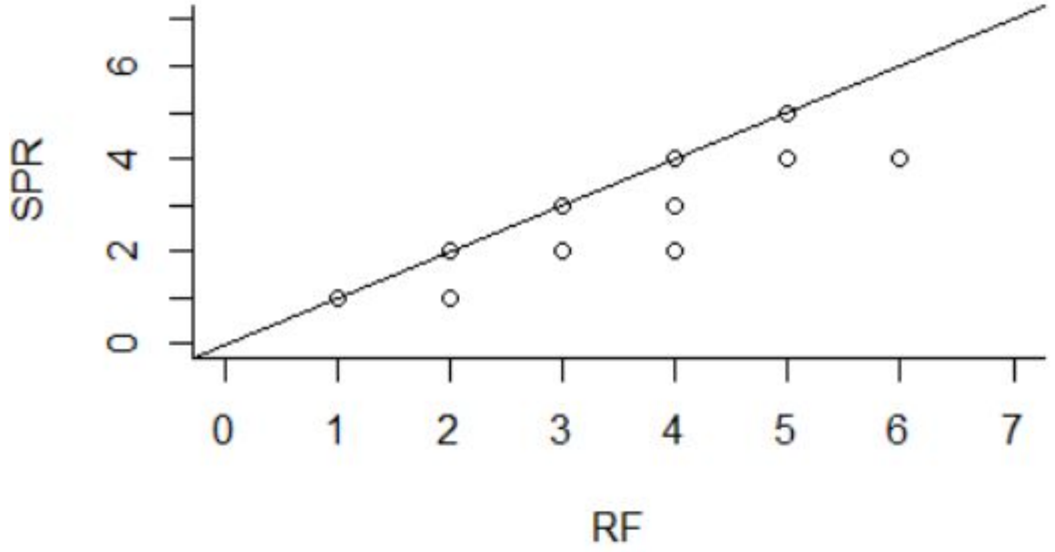
- Distances also give us a base measurement of variation within CloudForest that can be used as the starting point for other analyses

Activity: Calculating Tree-to-Tree Distances

- Find the “SSBWorkshop_2023” repository at https://github.com/TreeScaper
- Download the “crocs_ATP6.tre” and “turtle_mitochondrial_genes.tre” files and input them into CloudForest
- Navigate to the “Visualizing Treespace Using NLDR” tutorial at https://treescaper.github.io
- Work through the tutorial through steps 1 and 2 for both datasets
- Calculate a URF distance matrix for the turtle dataset
- Calculate a URF and SPR distance matrix for the crocodilian dataset
    - View the contents of the output files, and compare them by eye. Do they look the same or different?

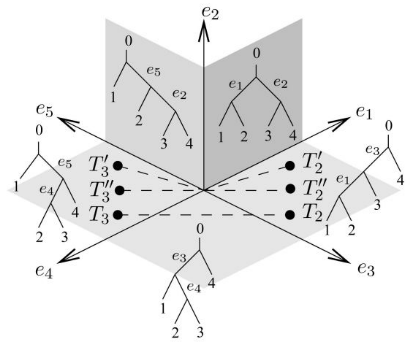
Different distances can tell different stories!

- Since each distance calculates their values differently, you can expect to see variation in their estimates
- Useful to explore your data with multiple options to see which best suits it!



Non-Linear Dimensionality Reduction

The Dimensionality of Treespace



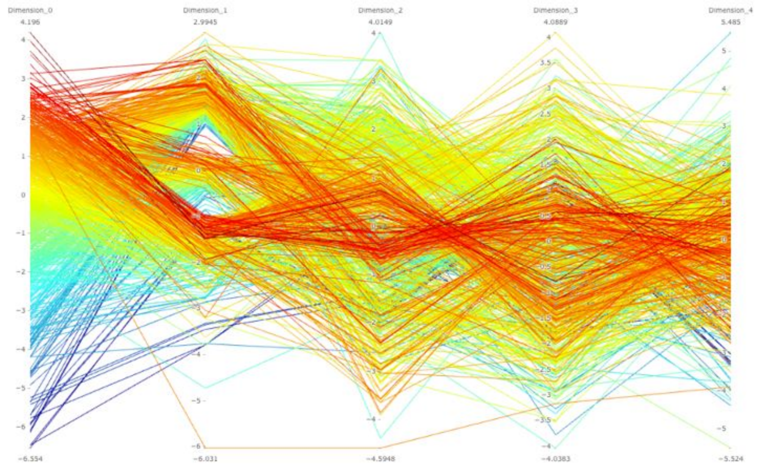
- Imagine we want to display treespace on a coordinate plane
    - Each coordinate point represents a tree
    - Coordinate of each point determined by its **distance** to other trees
- Smaller trees with less complexity are easier to place
- Larger, more complex trees require much higher dimensionality in order to accurately plot

Treespace of larger treesets

- Using our tree-to-tree distance as the measurements that define the tree’s coordinates, we can try to plot our trees
  - The natural structure of our treeset may require high dimensionality to plot
  - In some cases, 20+ dimensions!

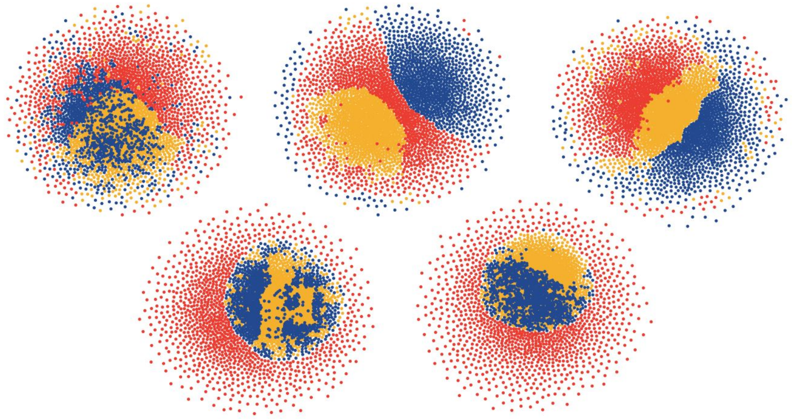
- Hiplots, like the one shown here, can be used to visualize these high-dimension results

- Can be difficult to interpret, but we can use a method to attempt to reduce the dimensionality of our treespace



Non-Linear Dimensionality Reduction (NLDR)

- The general aim of NLDR is to project high-dimensional data into lower-dimensional spaces
- Can be used to project our highly dimensional distance-based tree coordinates into lower dimensional spaces!
    - Easier interpretation, but at a cost! (More on that later!)

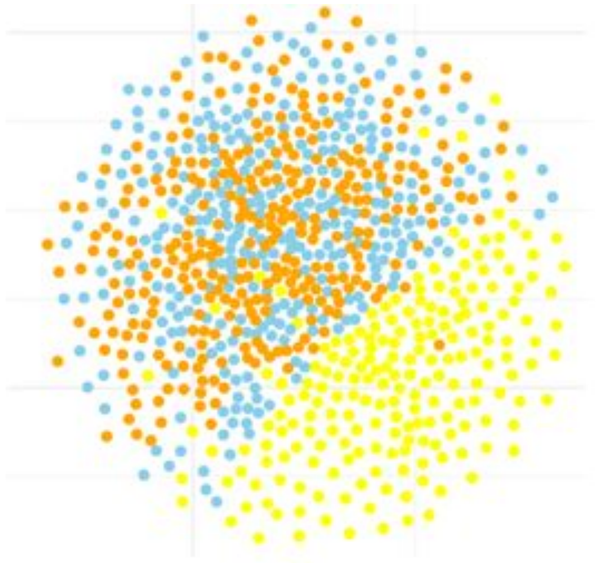


Visualizing Treespace with NLDR

- NLDR gives us a great jumping-off point to begin visualizing variation within treesets

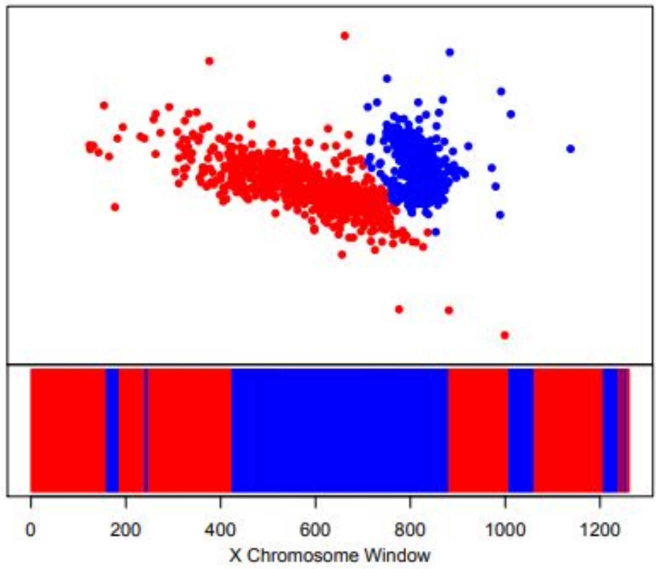
- In some examples, NLDR alone is enough to pinpoint the sources of some variation!

Visualizing Treespace with NLDR



- Here, trees are generated using the same dataset, but trees generated by different sequence evolution models are colored differently!

Visualizing Treespace with NLDR



- Here, trees are colored according to which 1,000kb window they fall in
  - Different signal in the centromere!

Dimensionality Reduction Algorithms

- Different algorithms and cost functions with which to perform the dimensionality reduction exist, and can yield slightly different results

- Multiple options for both are available in CloudForest, and we encourage you to try multiple to determine how sensitive to these changes your results are!

Discussion

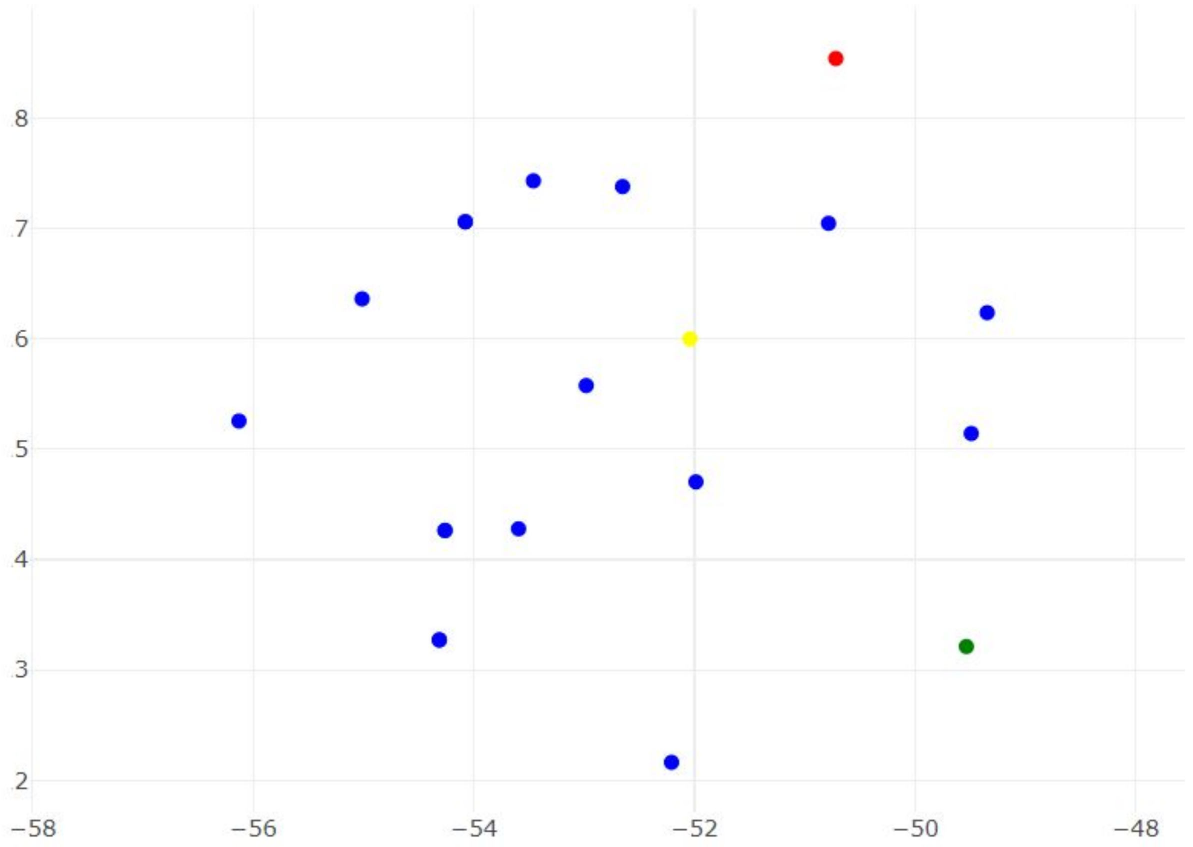
Form small groups or talk with your neighbors!

- Before today, had you heard of NLDR?
- If you have heard of it, have you used it before?
- If you have used it before, what sort of question were you using it to address?
- What are some potential uses you could think of for NLDR?

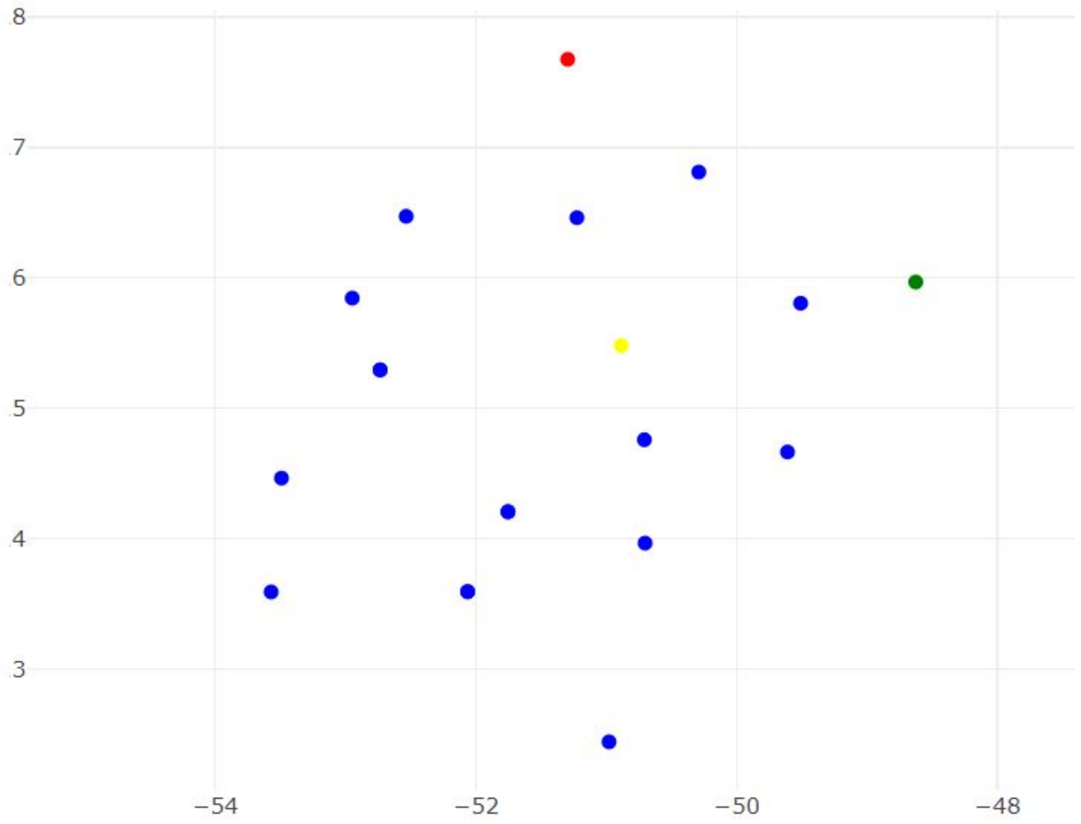
Activity: Visualizing Trees with NLDR

- In the “Visualizing TreeSpace with NLDR” tutorial, continue through steps 3-5 with your previously computed distance matrices
- Compute and visualize 2D NLDRs using the Crocodilian URF and SPR distances
    - Do they look identical?
- Using the turtle URF distances, calculate a 2D NLDR using the “STOCHASTIC” algorithm option, as well as the “GAUSS_SEIDEL” algorithm option
    - Compare the two!
- Calculate two 3D NLDRs using the turtle dataset, once with an unweighted distance matrix and once with a weighted distance matrix

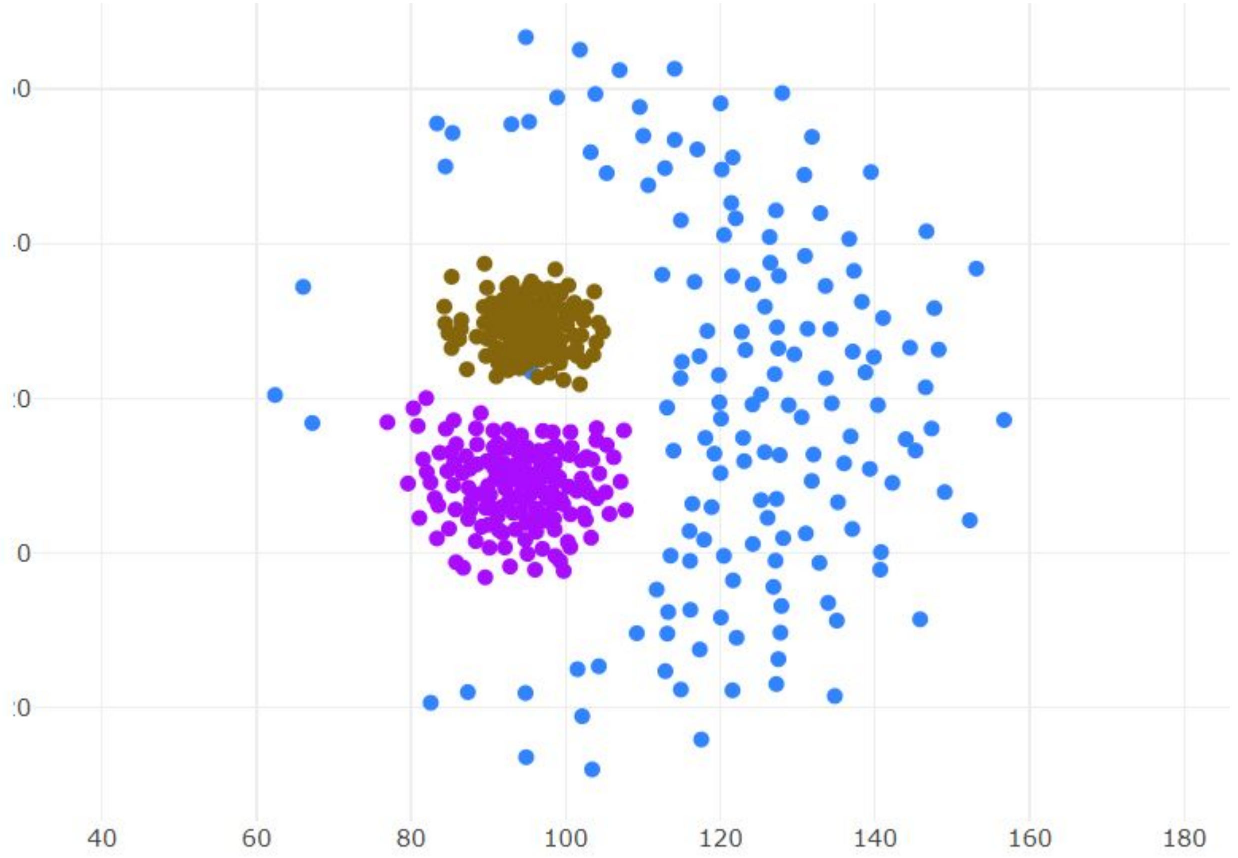
Crocodilian URF NLDR



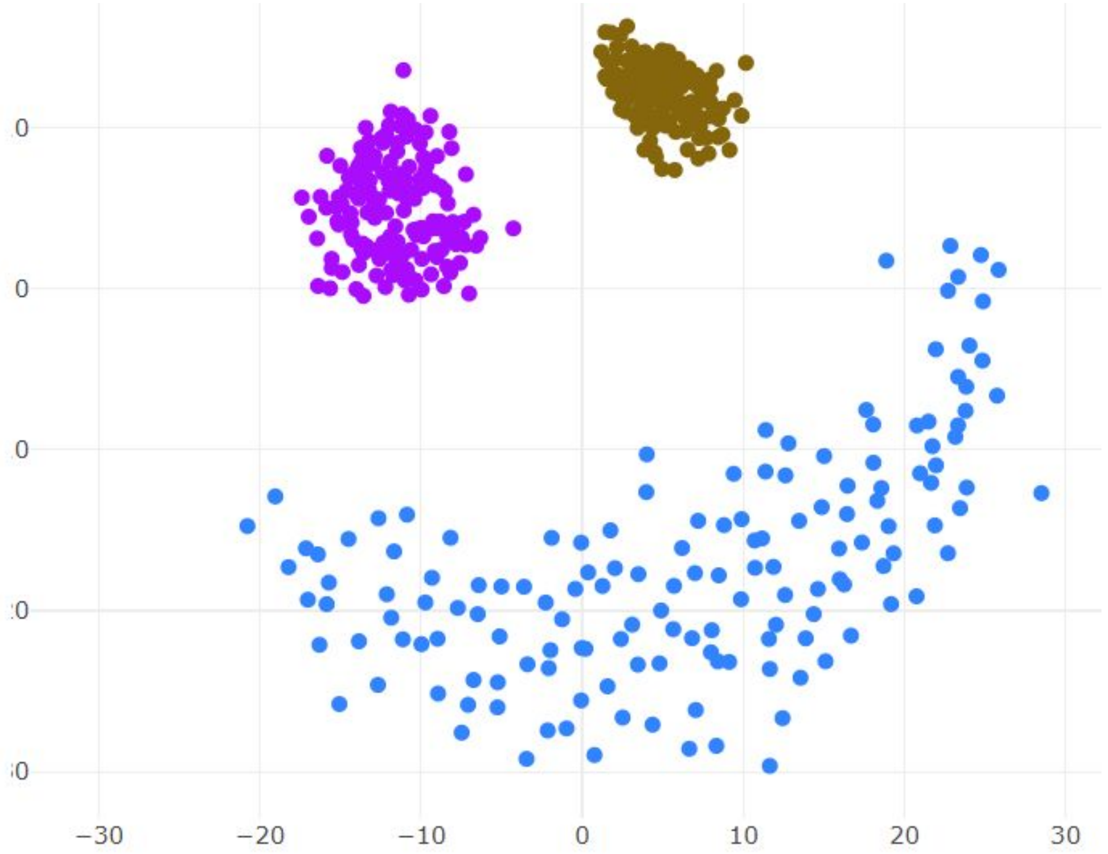
Crocodilian SPR NLDR



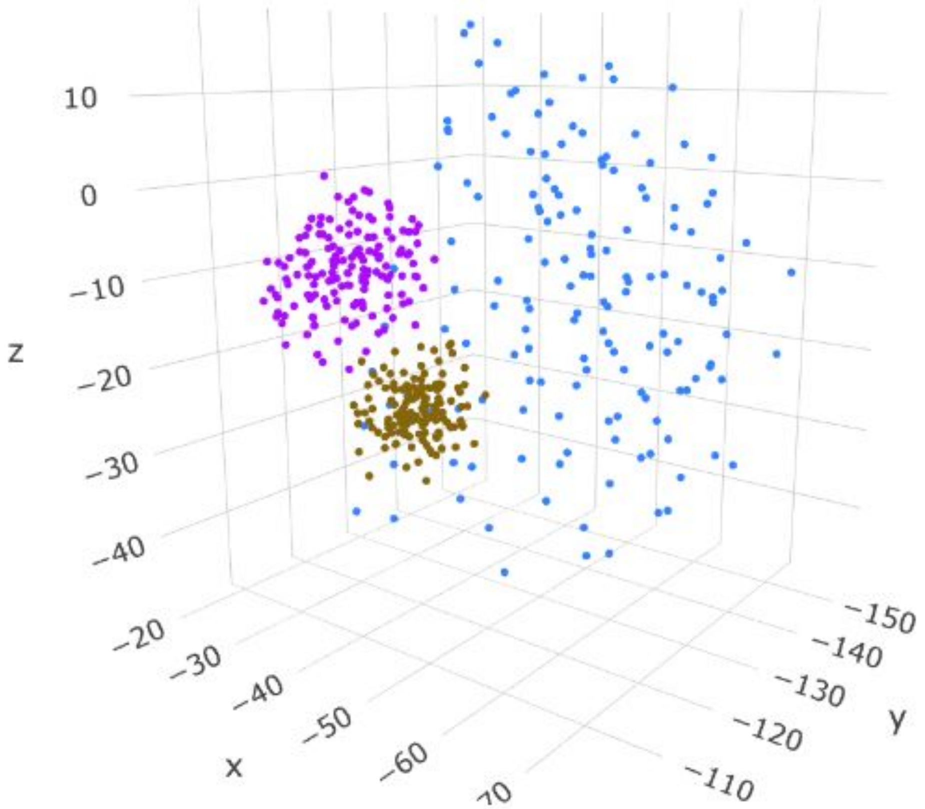
Turtle NLDR, Stochastic Algorithm



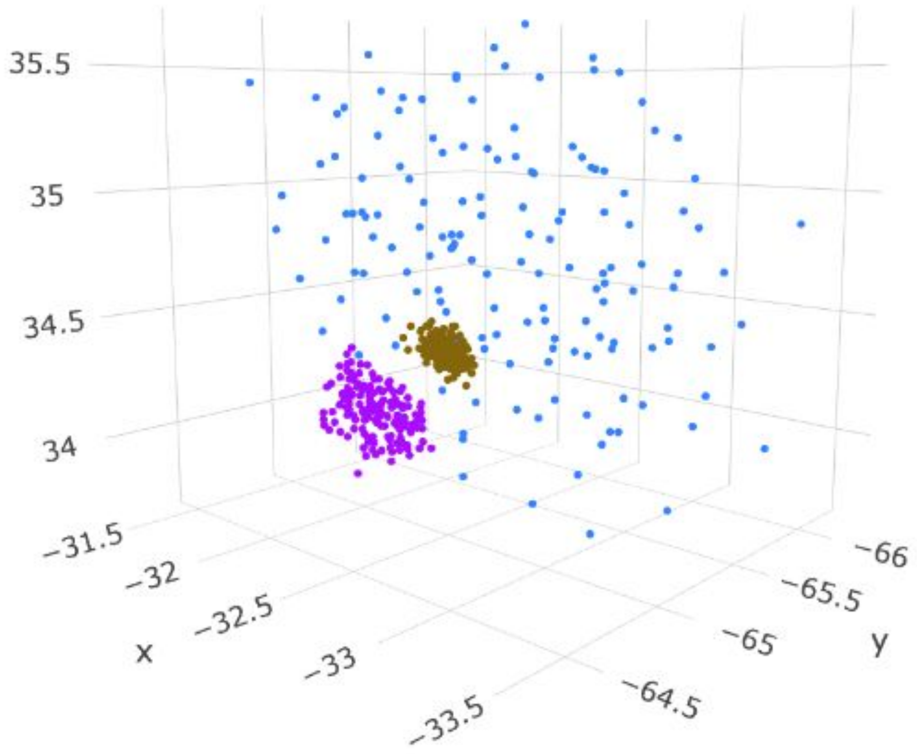
Turtle NLDR, Gauss-Seidel Algorithm



3D Turtle NLDR, Unweighted



3D Turtle NLDR, Weighted



Community Detection

Visualizing Treespace

- When we visualize treespace using NLDR, we do so using pairwise Tree-to-Tree distances in the treeset
- When reducing the dimensionality of treespace, sometimes NLDR algorithms have to project the data in a way that can mask the true relationships of the trees in higher dimensions

- How can we avoid this?

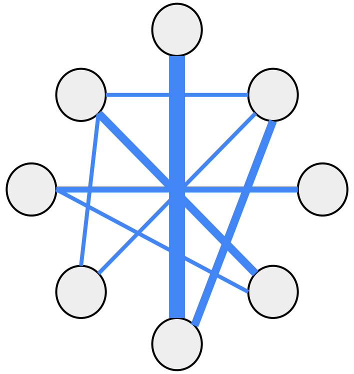
Converting Distances to Affinities

- Instead of thinking of distances, where high distances represent more dissimilarity, lets think in terms of affinities!
    - Higher affinity, more similar
- To do this, we take the inverse of our tree-to-tree distances

$$\frac{1}{\text{Distance}} = \text{Affinity}$$

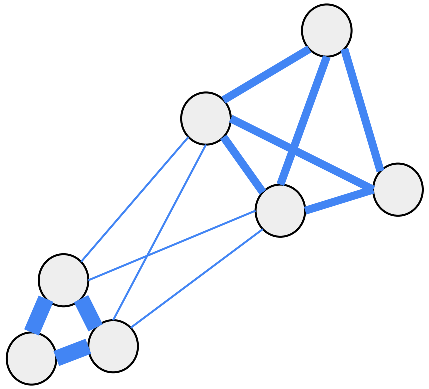
Affinity Networks

- Now, using these affinities, we can construct a network!
- In this network…
  - Nodes = Trees
  - Edges = Affinities
- Trees with higher affinities will have strong connections in the network, and vice versa!



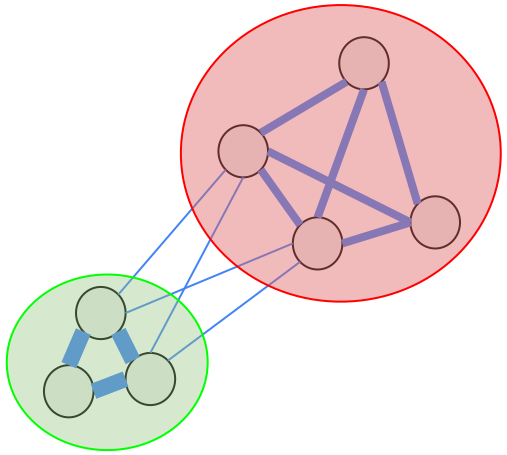
Communities of Trees

- Within these affinity networks, natural grouping of similar trees tend to appear
  - These groupings are usually visible within NLDRs, but sometimes they can be more cryptic and not appear due to projection distortion



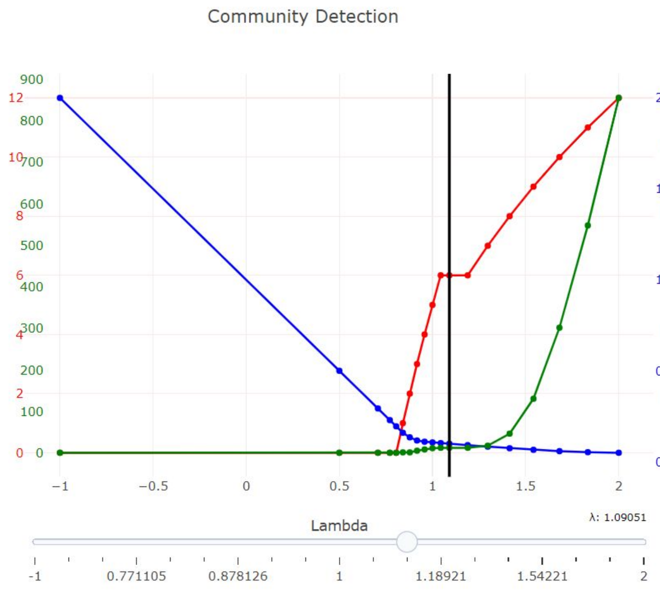
Detecting Tree Communities

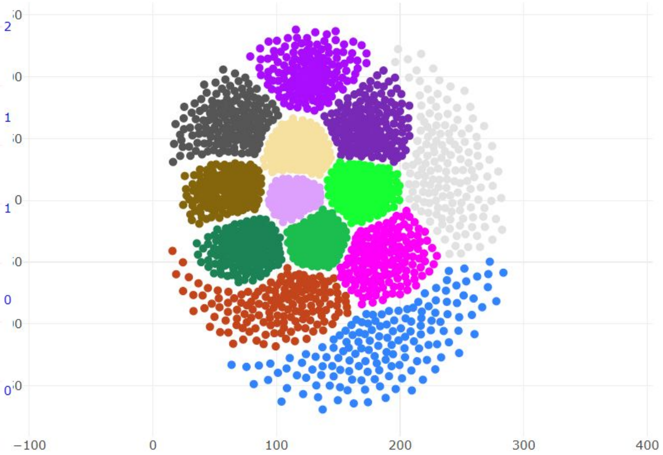
- In order to detect these groupings, we can leverage community detection algorithms!
- These algorithms act on the underlying affinity networks to find communities of similar trees, thus bypassing any potential projection issues in NLDR!



Combining Community Detection and NLDR Results

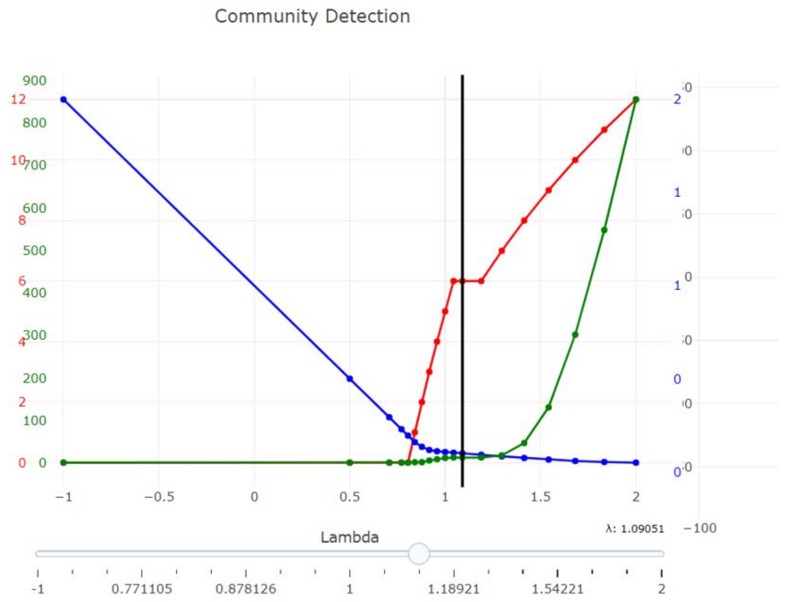
- Once an NLDR and Community Detection have both been performed in CloudForest, we can overlay the results of CD on to our NLDR to explore the natural groupings of trees!





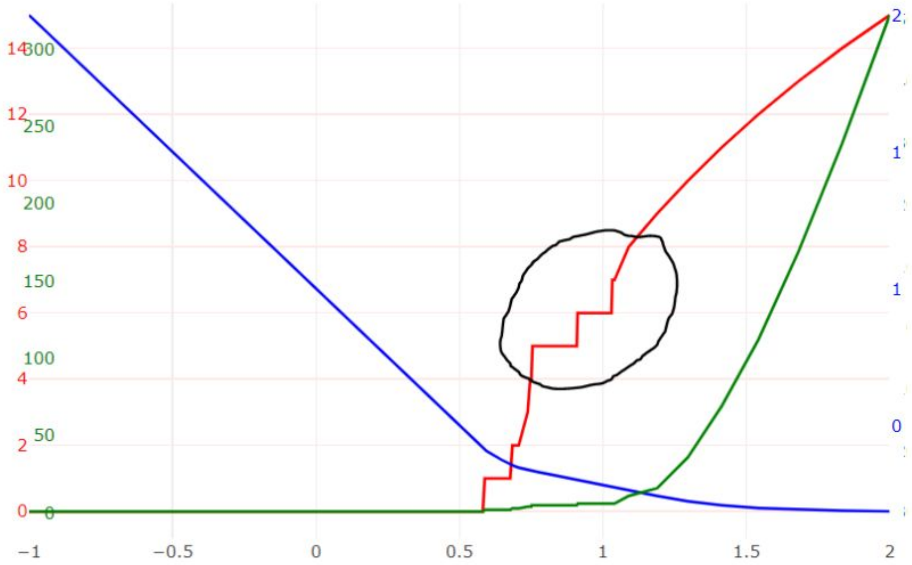
Analyzing CD Results

- In our CD Results, the X-axis represents a range of Lambda values
- Each Lambda value changes how the CD algorithm behaves, and how sensitive it is to tree groupings of different sizes



Looking for Lambda Plateaus

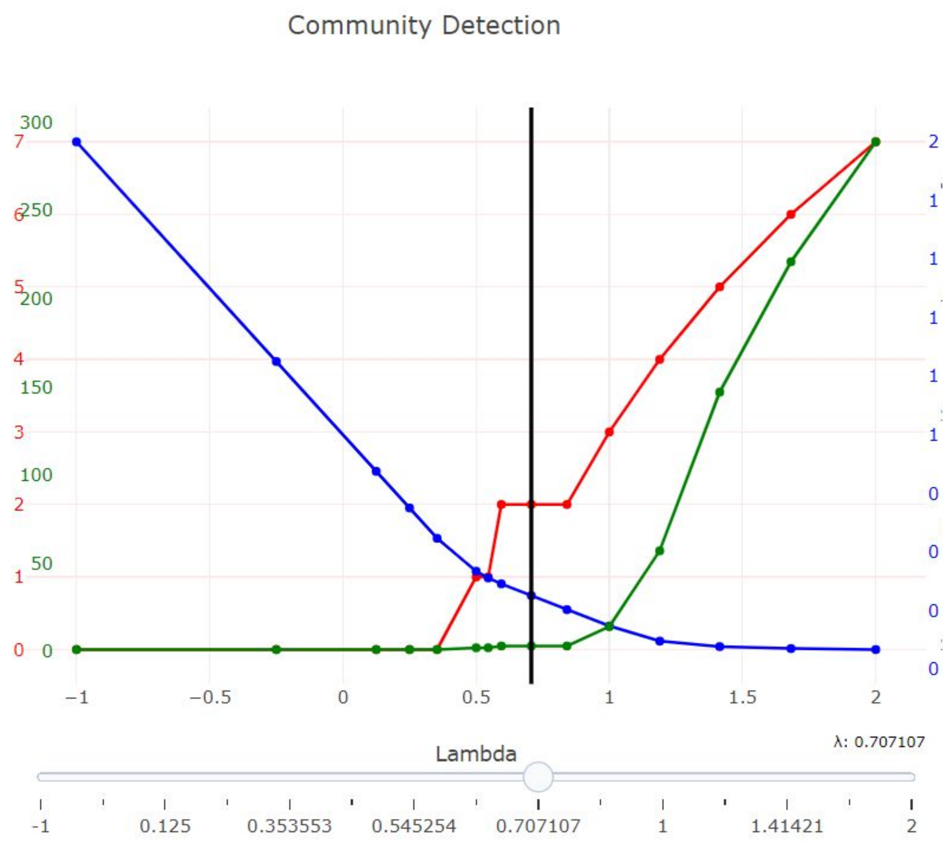
- When we look at these results, we're looking for ranges of lambda values, or "plateaus", where all values detect the same communities of trees
  - These plateaus represent the most natural groupings of trees!

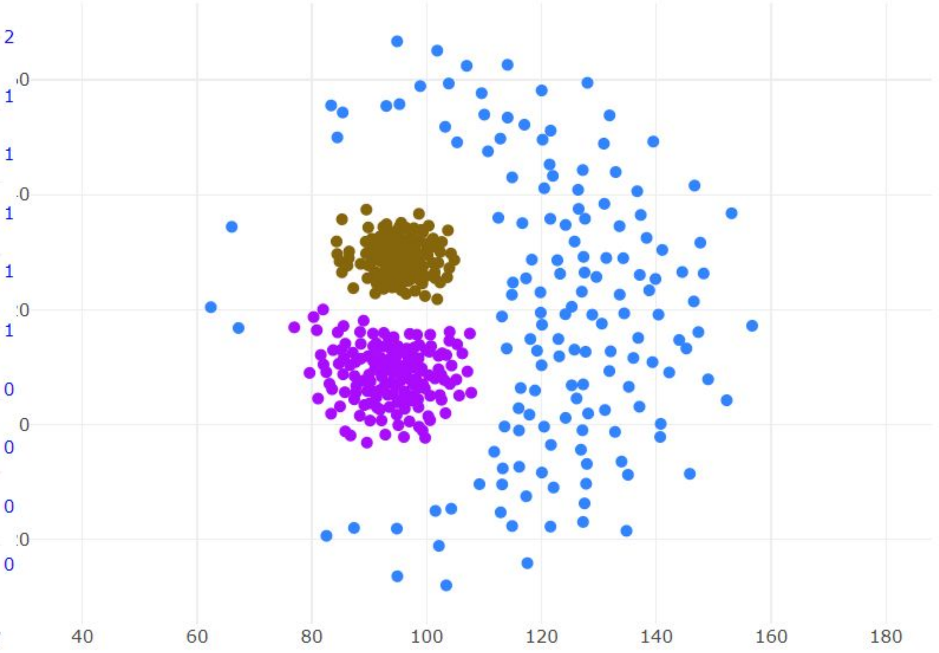


Activity: Performing Community Detection

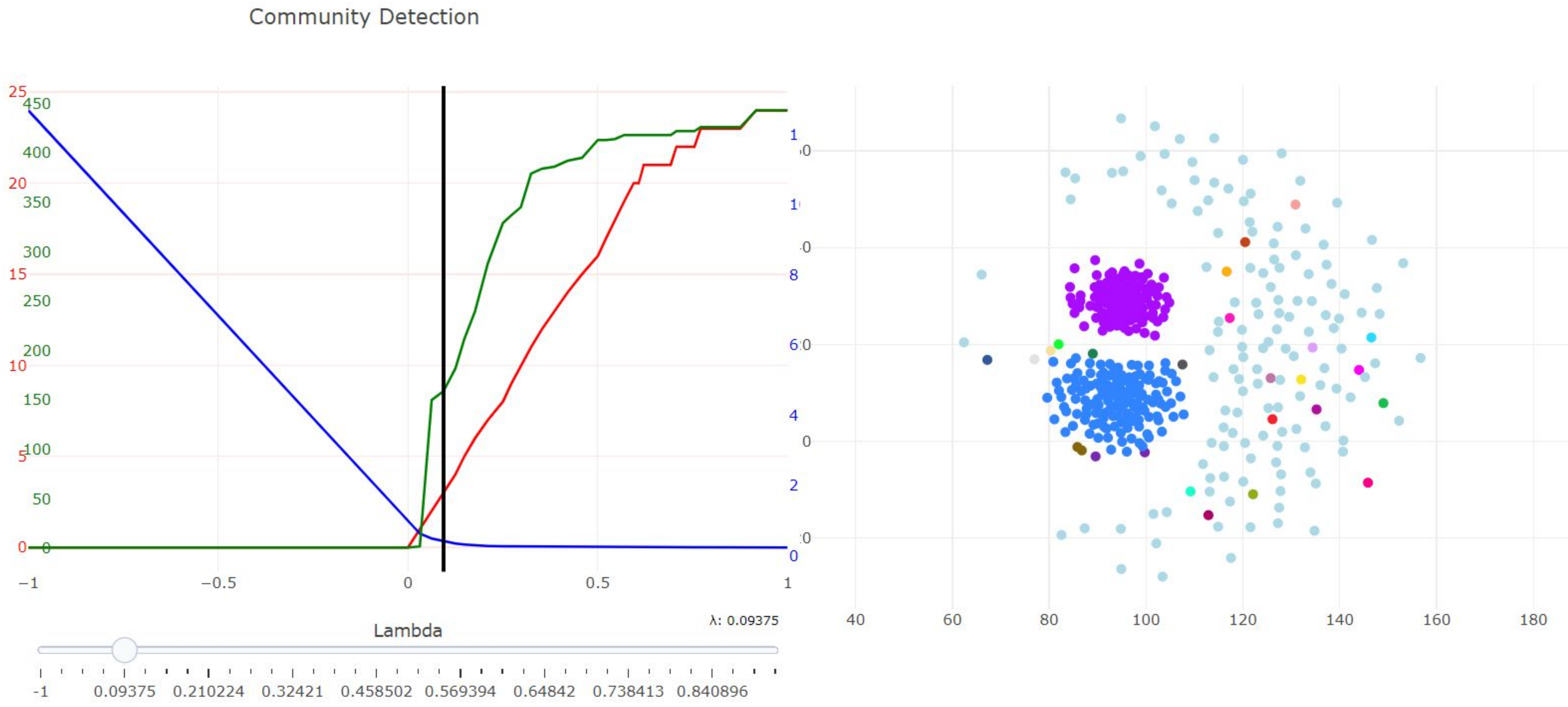
- Using the “Finding Structured Patterns in TreeSpace using Community Detection” tutorial at https://treescaper.github.io, perform an automatic community detection on the turtle mitochondrial dataset with both the **Configuration Null Model** and the **Constant Potts Model.** Visualize both with their associated NLDR, and using the slider compare the ways the two algorithms go about grouping trees
- Navigate to the SSBWorkshop_2023 repo at https://github.com/TreeScaper and download the Cat X-chromosome dataset
- Using the previous tutorials, compute a **weighted** distance matrix and affinity matrix for the cat dataset, then perform a 2D NLDR and an automatic community detection analysis and visualize the results of both
- Using your automatic community detection results as a starting point, perform a manual community detection focusing in on a specific range of lambda values to get a better look at any plateaus

Turtle Dataset, Configuration Null Model



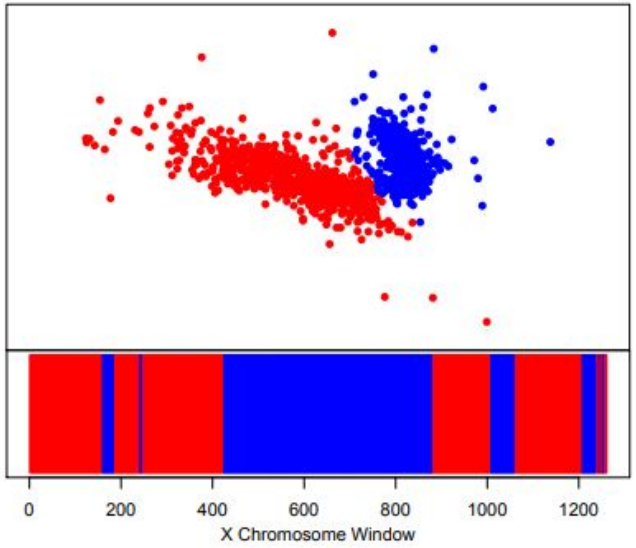


Turtle Dataset, Constant Potts Model



CD Informing NLDR

- These community detection plateaus often detect more cryptic structuring in treesets, such as in these results where community detection helped to identify the differing phylogenetic signal in low-recombination regions of the X chromosome!



Wagner et al. 2021, “Investigating the Genomic Distribution of Phylogenetic Signal with CloudForest”