PhyNetPy (MP-SUGAR) Tutorial

Botany 2023, Boise, Idaho, July 23rd 2023

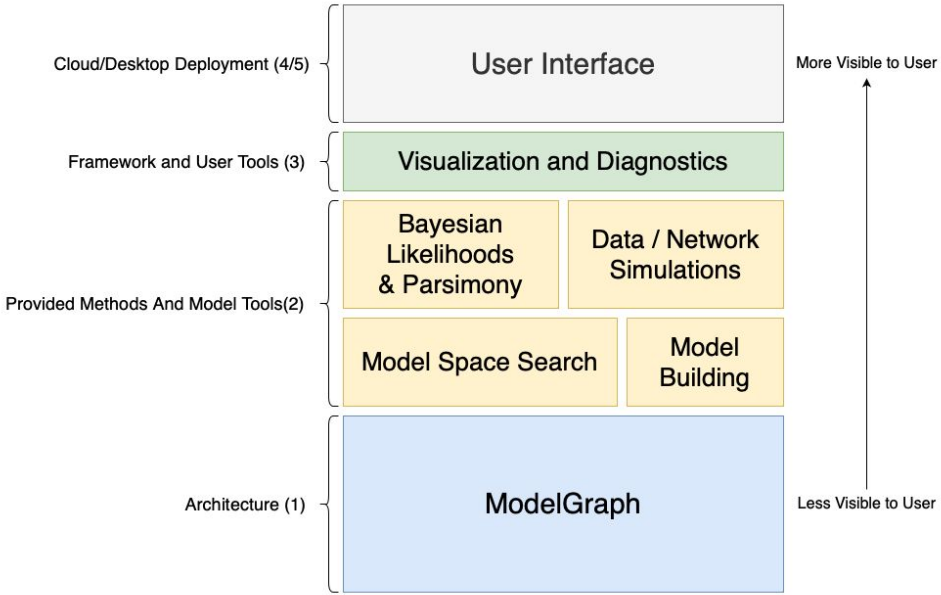
Introduction

- Huw Ogilvie – PI
  - huw@huw.id.au
- Mark Kessler – Software Engineer
  - mak17@rice.edu

- Find the tutorial and follow along at https://phylogenomics.rice.edu/html/phynetpy.html

PhyNetPy Overview

- Brand new, state-of-the-art framework for phylogenetics
- Native support for all kinds of gene tree discordance, including reticulation
- Python front end interface, C# backend (for efficiency, as needed)
- Maximum likelihood, Bayesian, simulation-based and maximum parsimony inference
- Centered around a graphical modeling approach that is highly modular



PhyNetPy architecture and plan

PhyNetPy Goals

- Efficiency
  - Tree inference is hard because the number of topologies grows hyper-exponentially with the number of taxa
  - Phylogenetic networks are significantly more complex than trees
  - Need efficient network operations, algorithms, and tricks to reduce runtime
  - Many methods also suffer from a scaling issue with large data sets and networks. Algorithms that scale effectively are also a must
- Functionality/User-Friendliness
  - Phylogenetic network inference and analysis software lags behind tree software
- Flexibility
  - Software that isn't flexible to use is software nobody uses :)
  - Developing methods that can handle as many biological systems as possible is important

The Problem of Allopolyploidy

- One type of hybrid speciation
- More common in plants
- **Two** copies are maintained of each chromosome from both parental species
- Multiple pathways exist that lead to allopolyploids

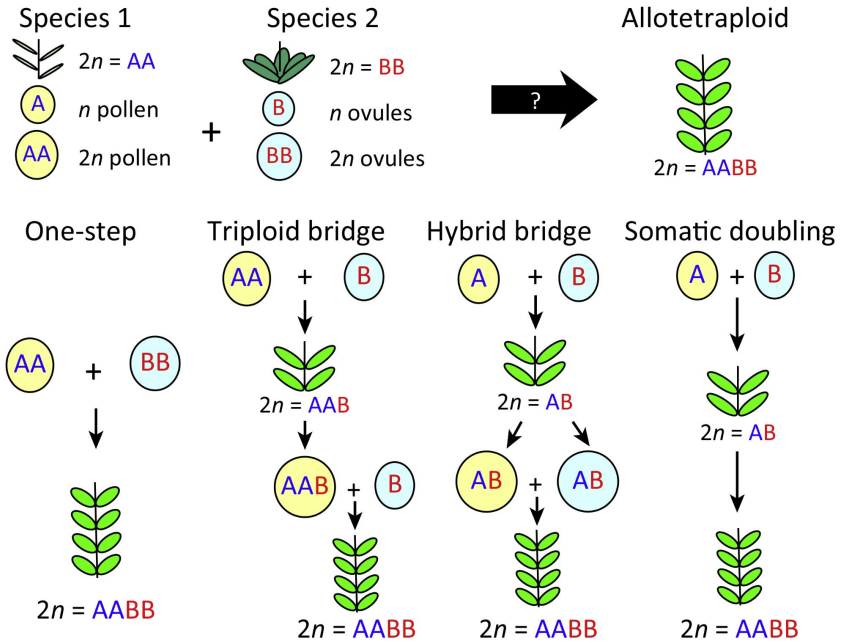


Figure from Mason & Pires (2016), "Unreduced gametes: meiotic mishap or evolutionary mechanism?" *Trends in Genetics*

Allopolyploidy and Recombination

- Meiotic recombination causes the evolutionary history of individual loci to differ from the species phylogeny (even **without** any hybridization!)
- In allopolyploids, this recombination may **not** occur between chromosomes derived from different parent species
- Sets of chromosomes from different parent species maintained as **subgenomes**

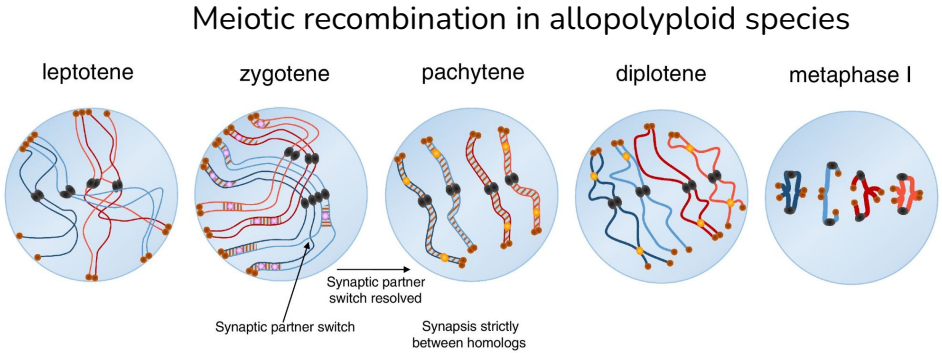
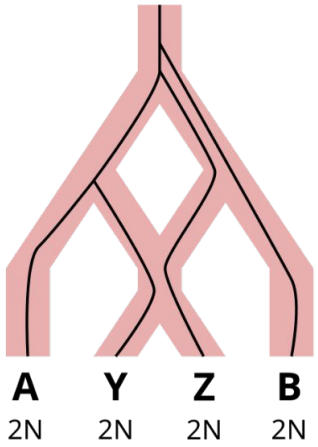


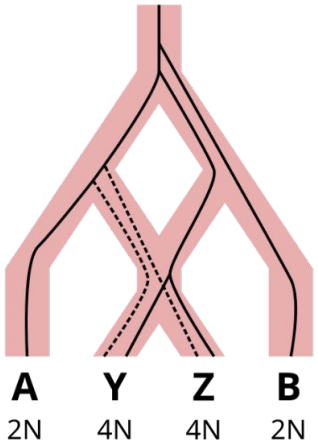
Figure adapted from Lloyd & Bomblies (2015), "Meiosis in autopolyploid and allopolyploid Arabidopsis" *Current Opinion in Plant Biology*

Inferring Allopolyploid Species Phylogenies

Introgression or homoploid hybridization

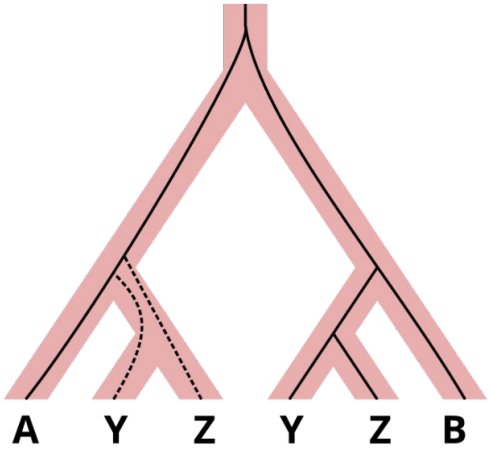


Phylogenetic network



Phylogenetic network

Allopolyploid hybridization



Multilabeled (MUL) tree

Minimizing Deep Coalescence (MDC)

- Developed by Maddison (1997)
- Infers **rooted** phylogenies, accounts for incomplete lineage sorting (ILS a.k.a. deep coalescence)
- For each gene tree, sum the number of excess lineages that fail to coalesce at each speciation event
- Find the species tree that minimizes that number summed over all gene trees
- Implemented for allopolyploidy by Yan et al. (2022) in PhyloNet and called MPAllopp
- More flexible and scalable than the Bayesian method AlloppNET (Jones, Sagitov & Oxelman 2013)

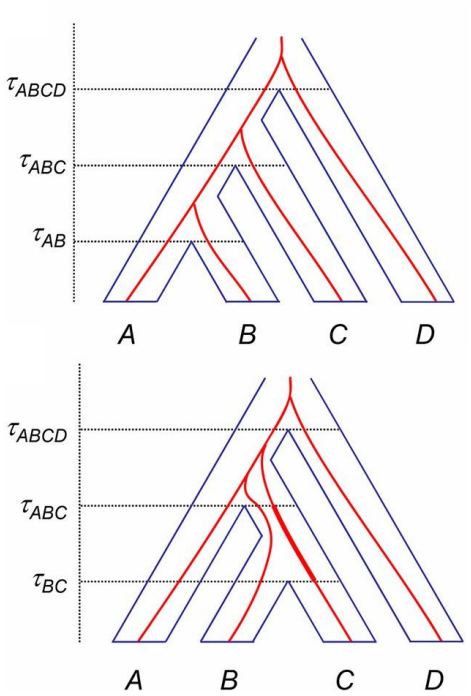


Figure adapted from Xu & Yang (2016), “Challenges in Species Tree Estimation Under the Multispecies Coalescent Model” *Genetics*

MP-SUGAR: improving on MPAllopp

The bolded lines denote changes that will be implemented for the next release (1.0.0), unbolded items are currently provided in the tutorial release (0.0.8)

- Speed
  - More efficient data structures, algorithms, and optimized call structure
  - **Time sensitive algorithms in C#**
  - Runtime improvement of ~100x compared to PhyloNet
    - 10 sec vs 30 min in some analyses
  - **Multicore functionality for running multiple chains**
- Flexibility
  - **Ability to handle multiple samples for each gene copy**
  - **Ability to create custom constraints for allele mappings**
  - **Ability to infer autopolyploidy / network bubbles**
  - **Choice of chain type (Simulated Annealing vs HC)**
- User friendly
  - A couple of simple Python lines and one function call
  - Network data structures that provide ease of I/O and post-processing

Taxon Mappings

- Multi-locus methods require mapping sequences to taxa
- Refer to the section Taxon Mappings at https://phylogenomics.rice.edu/html/mp_allop_tutorial.html for an additional example
- A gene/taxa may have multiple samples and multiple copies
  - Therefore, in our data sets, we have named taxa as follows:
    - Sample number (01, 02, 03, ...)
    - Gene Name (generally just single characters from {a-z}
    - Subgenome Location (from {A-Z}
  - The name 01aB means “sample 1 from gene A on subgenome B”
  - Currently only data sets with 1 sample for one gene is supported. This will change soon
- For the trimmed version of scenario J, with taxa labels from
  - {01fA, 01tB, 01vB, 01bA, 01tA, 01oA, 01vA, 01cA, 01aA, 01dA, 01uA, 01uB}
  - The taxon mapping is the map from Taxon Names to all the samples/gene copies
  - `{'U': ['01uA', '01uB'], 'T': ['01tA', '01tB'], 'B': ['01bA'], 'F': ['01fA'], 'V': ['01vB', '01vA'], 'C': ['01cA'], 'A': ['01aA'], 'D': ['01dA'], 'O': ['01oA']}`

Setup and Installation

- Go to https://phylogenomics.rice.edu/html/phynetpy.html to install
- Using pip:
  - pip install phynetpy

  OR

- Using a virtual conda environment:
  - Create new environment
  - conda activate name_of_new_env
  - pip install phynetpy
- If any other version of PhyNetPy other than 0.0.8 is downloaded, run
  - pip uninstall phynetpy
  - pip install phynetpy==0.0.8
- Download *"tutorial.py"* and open it in an IDE

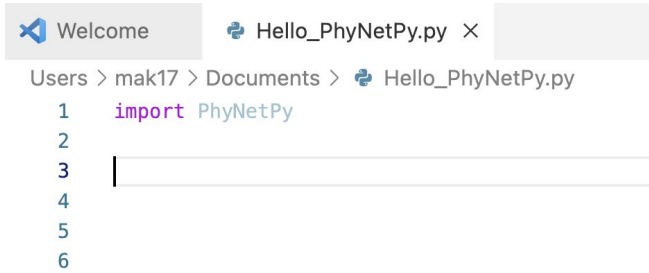
- https://pastebin.com/X22685wG

Using PhyNetPy in an IDE

- If using pip
  - Ensure python interpreter selected is the same as the python version used to install PhyNetPy with pip (if you have more than one python version installed on your computer)
- If using a virtual conda environment
  - Ensure Python interpreter selected has the same name as the environment name that was activated during installation

Relevant PhyNetPy Modules

- PhyNetPy.MPSugar
- PhyNetPy.Graph (DAG obj)
- PhyNetPy.GeneTrees(GeneTrees obj)
- PhyNetPy.NetworkParser (NetworkParser obj)



Tutorial Scenario 1

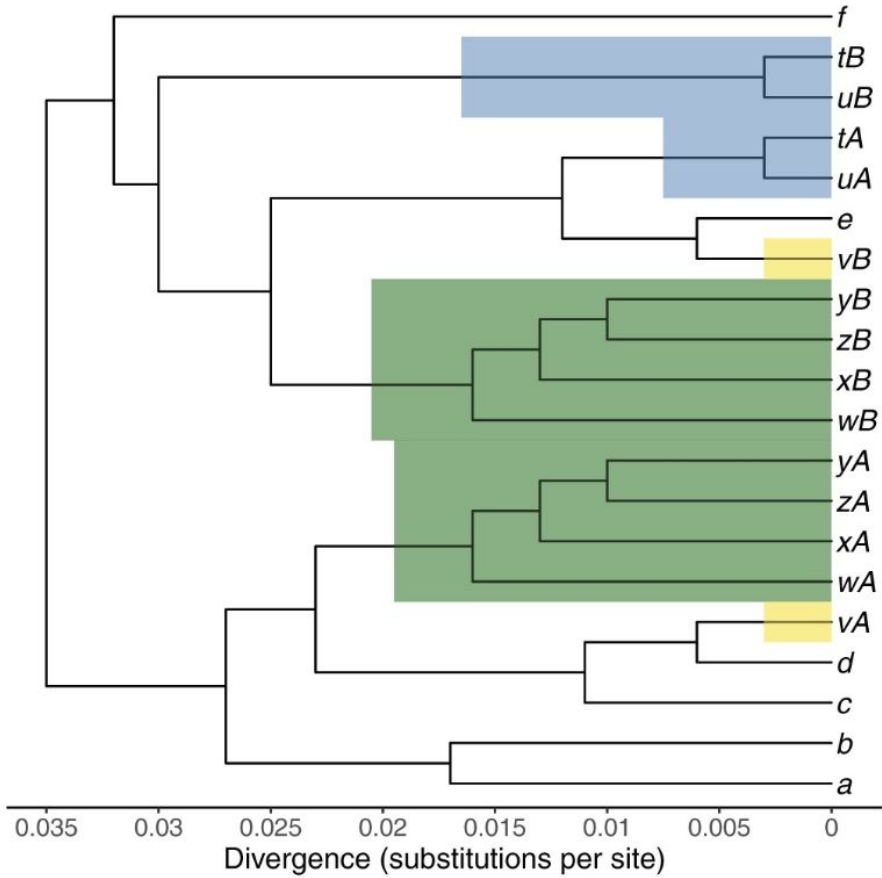
Source: Zhi Yan and others, Maximum Parsimony Inference of Phylogenetic Networks in the Presence of Polyploid Complexes, *Systematic Biology*, Volume 71, Issue 3, May 2022, Pages 706–720,

https://doi.org/10.1093/sysbio/syab081

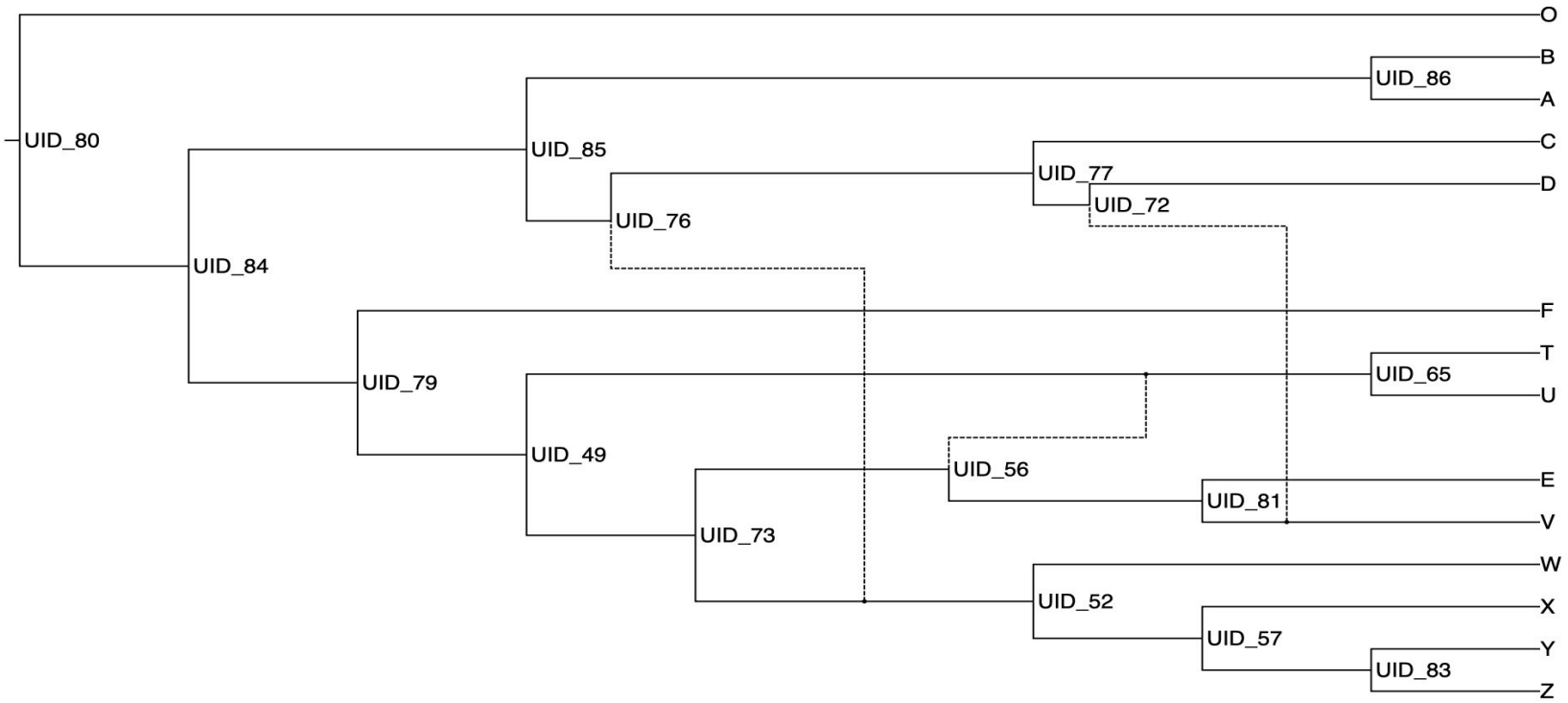
- Download the gene tree nexus file
  - *"J_untrimmed.nex"* on tutorial page

SIDENOTE: These files download as .txt files.

Simply rename the extension to .nex

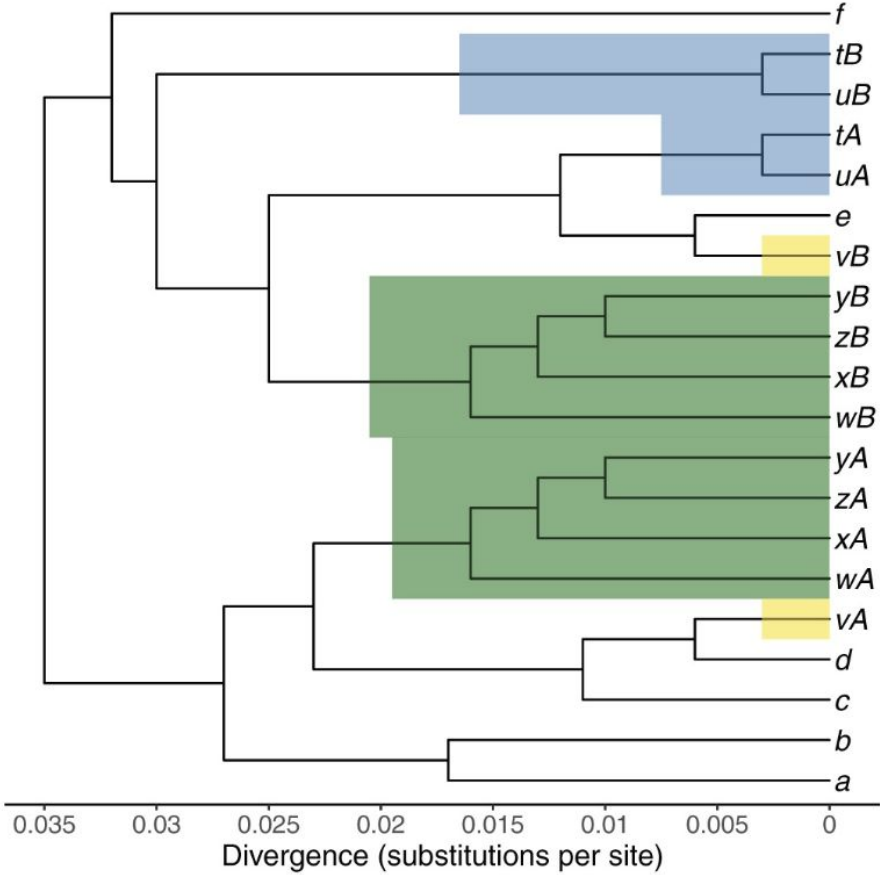


Untrimmed J Network

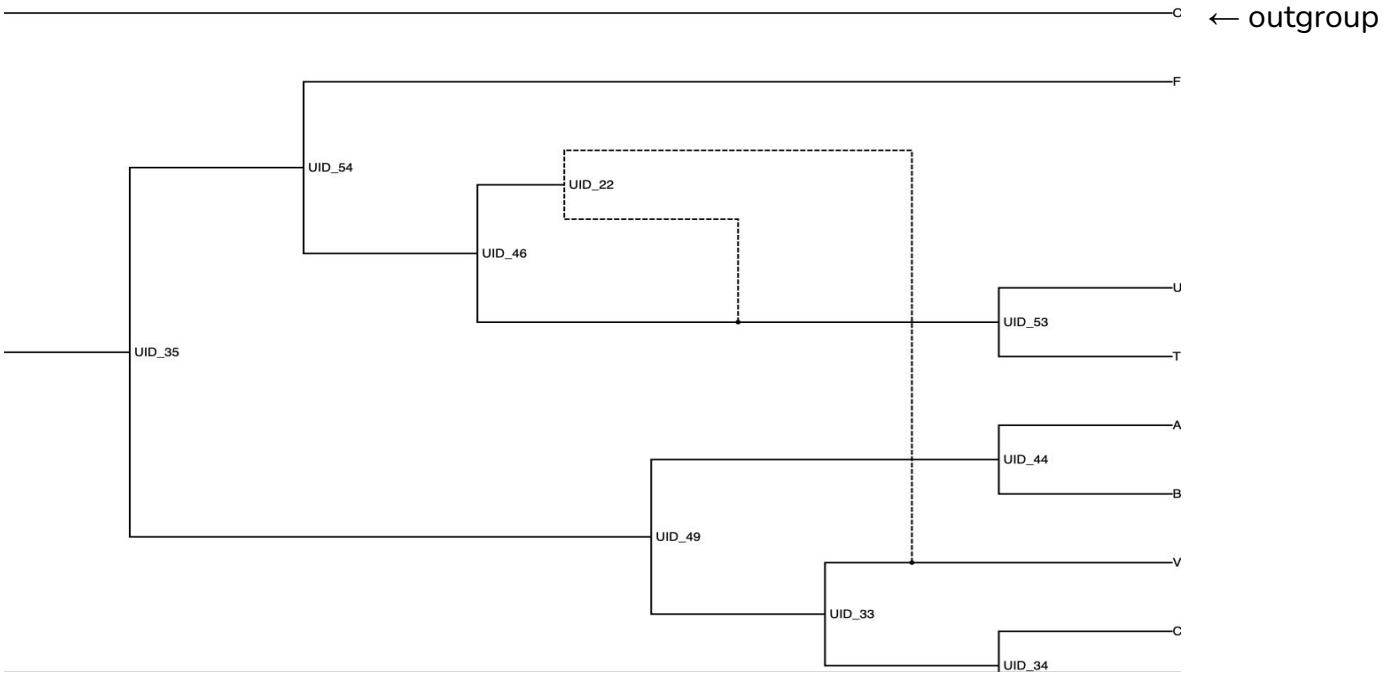


Tutorial Scenario 2

- Download the gene tree nexus file
  - *"J_trimmed.nex"* on tutorial page
- We have removed these taxa:
  - E, Y, Z, X, W
  - Leaving F, T, U, V, A, B, C, and D.
- The resulting network should contain a "bubble" above T and U, with V connecting in.



Trimmed J Network



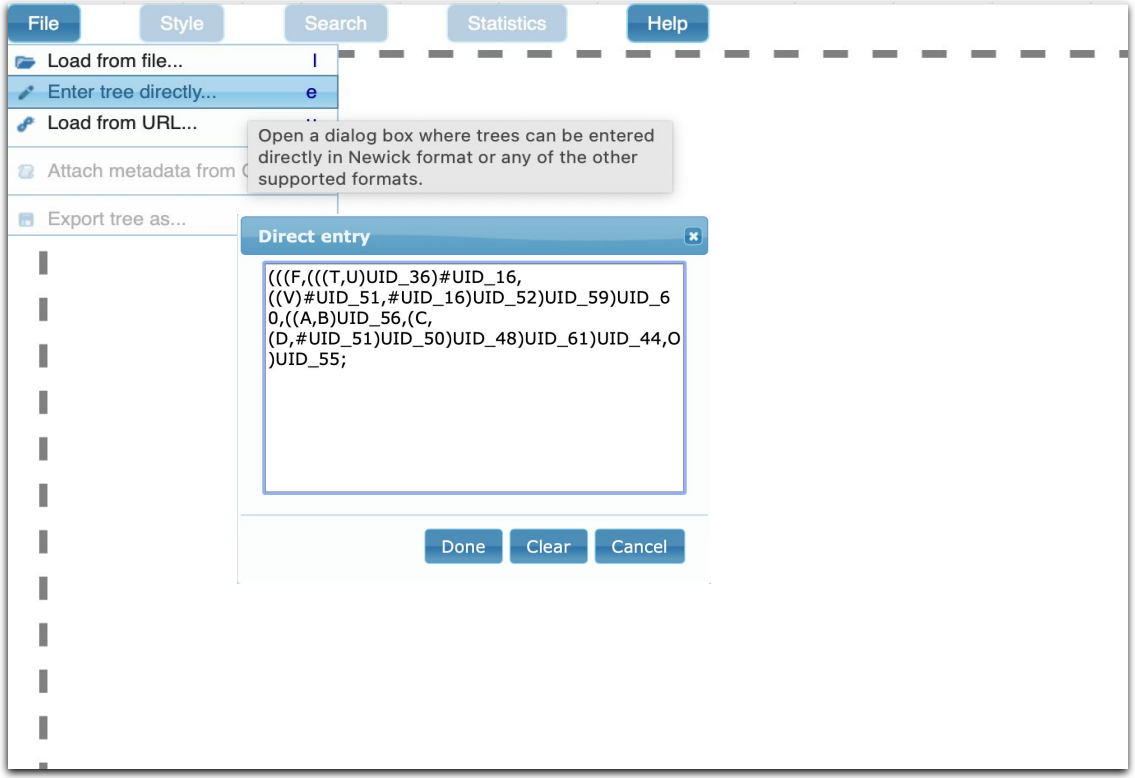
Python Script to Call MP-SUGAR

- Download “*tutorial.py*” if you haven’t already, and open it in your IDE.
- Be sure to replace all file names with the proper paths!
  - Use absolute paths if you don’t want to move the file from the download folder to a special folder for this tutorial
- We recommend running the trimmed version first, as that runs much faster and if you have problems it will be easier to deal with!
- To run the trimmed version of J, uncomment the relevant lines and change the file paths to the right nexus file

Important Information

- Input is **rooted** gene trees (outgroup required)
- Output is **rooted** network (outgroup, in theory, not required)
- Uses hill climbing so can get stuck in local optima
    - Need to run multiple chains to be confident best solution is found
- The trimmed version should score **-6**, the untrimmed should score **-17**
- The untrimmed version may need 4-5 (or more) chains to find the correct network, which should take about 10-15 minutes to run. The trimmed version should take 10-15 seconds per chain, and 1 or 2 chains should suffice to find the network of score -6. If the correct scores don't show up on the first run, simply rerun the script.
- **Copy the newick strings and paste into icytree.org to visualize the network!**

Icytree Visualization



PhyNetPy Notes

- Current version is 0.0.8
  - This is meant only as a tutorial release.
    - Data structures, namings of various classes, functions, and fields MAY change
    - Performance is not currently as good as it will be
- Next version will be 1.0.0
  - The first stable, production quality release that is fully documented. Will contain more methods such as MCMC-BiMarkers and MCMC-Seq, and a consistent/performative set of data structures for development purposes
  - Targeted for October 1. Check with the website for news!

Questions?

Acknowledgements

- Dr. Luay Nakhleh, co-PI
- Zhi Yang, PhD student & developer of MPAllopp

- Submit any questions, bugs, issues, or comments in the future to https://github.com/NakhlehLab/PhyNetPy/issues or contact Mark directly!