treesifter advanced

Introduction

treesiftr is an R package (R Core Team 2018) for visualizing the relationship between phylogenetic trees and phylogenetic data. Phylogenetic trees are crucial to the study of comparative biology, taxonomy, and evolution. However, understanding how to read a phylogenetic tree, and how a phylogenetic tree relates to underlying phylogenetic data, remains challenging.

In today's lab exercise, we will learn about the data in a phylogenetic matrix, and then use these data in the matrix to visualize a phylogenetic tree. We will use RStudio (RStudio Team 2015) to conduct our exercise.

> *Tip*
>
> Linked text goes to the glossary. If you see a term you don't recognize, remember you can refresh your memory at the bottom of this worksheet!

The fossil bear matrix

The data matrix included with treesiftr is a matrix of binary ("0" and "1") characters compiled to estimate a topology of living and extinct bear species (Abella et al. 2012). This matrix is fairly typical in size for a paleontological matrix, comprising 62 characters. It is, however, atypically complete, with only 18% missing data. In the following exercises, missing data will be represented by a thin black line. The "0" state will be represented in pale blue, and the "1" in brown.

treesiftr

treesiftr works by subsetting a phylogenetic matrix using the `start`, `stop` and `step` arguments. The `start` argument controls where in the matrix you would like to begin visualizing characters. For example, a start value of 1 would indicate to begin visualizing characters from the first character in the matrix. The `stop` value indicates what will be the first character in the last visualization. A stop value of 10 would indicate that the first character in the last visualization should be character 10. The `step` value indicates how many characters at once to visualize. A step value of three would indicate characters should be viewed in threes. For example, if `start` = 1, `stop` = 10, and step = 3, 10 visualizations will be produced. The first will visualize characters 1, 2, and 3. The final will be characters 10, 11, and 12.

A maximum parsimony tree is then estimated from each dataset. The tree is scored under both parsimony and Lewis' Mk model (Lewis 2001) for discrete character data. The data and tree are then visualized using ggtree, based upon the ggplot2 package. This application makes use of Shiny to provide a graphical interface, but in this tutorial, we will use the R Studio interface to visualize our data.

Installation

Currently, treesiftr can be installed via the devtools `install_github` function. treesiftr has a number of required packages. Install and load the below.

```
devtools::install_github("wrightaprilm/treesiftr")
library(treesiftr)
library(treesiftr)
library(phangorn)
library(alignfigR)
```

```
library(ggtree)
library(ggplot2)
```

Subsetting the phylogenetic matrix

The first step to making a treesiftr visualization is to select the subset of the phylogenetic matrix that we would like to visualize. This is performed via a function called `generate_sliding`. The below command will subset the

```
# Locate package data and read alignment
fdir <- system.file("extdata", package = "treesiftr")
aln_path <- file.path(fdir, "bears_fasta.fa")
bears <- read_alignment(aln_path)
# Read in a starting phylogenetic tree
tree <- read.tree(file.path(fdir, "starting_tree.tre"))

# Generate our list of dataframe subsets
sample_df <- generate_sliding(bears, start_char = 1, stop_char = 5, steps = 1)
```

The result of this is a dataframe, shown below:

```
sample_df
```

```
##   starting_val stop_val step_val
## 1            1        2        1
## 2            2        3        1
## 3            3        4        1
## 4            4        5        1
## 5            5        6        1
```

This dataframe dispays the start character (the first character that will be visualized) and stop character (the final character that will be visualized).

We can then build trees from each subset:

```
output_vector <- generate_tree_vis(sample_df = sample_df, alignment =
```

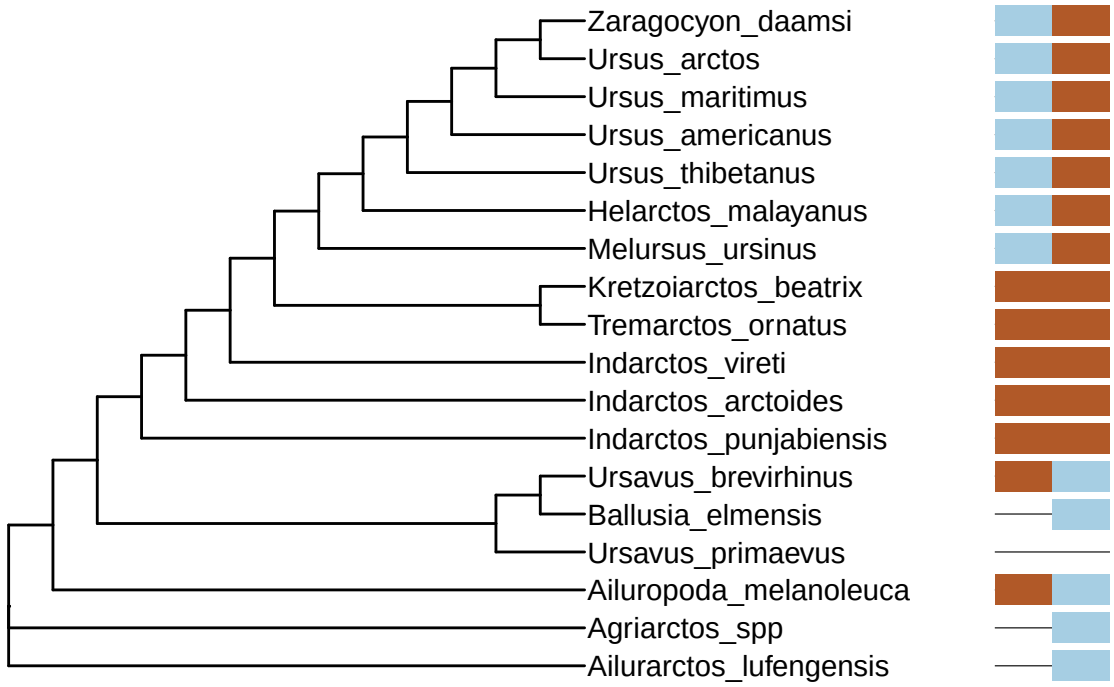
```
## Final p-score 2 after  0 nni operations
## Final p-score 2 after  0 nni operations
## Final p-score 2 after  0 nni operations
## Final p-score 2 after  1 nni operations
## Final p-score 2 after  1 nni operations
```

The above code saves the trees to a vector, but does not visualize them. They can be viauslized by naming the vector, like so:

```
output_vector
```

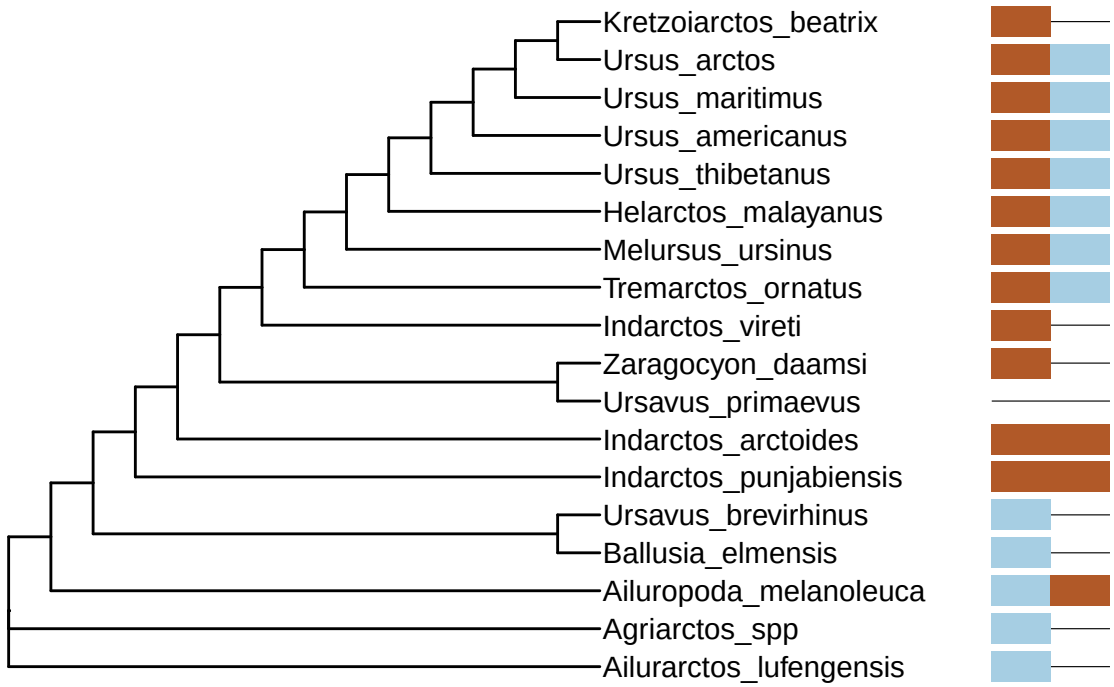
```
## [[1]]
```

1
2


Zaragocyon_daamsi
Ursus_arctos
Ursus_maritimus
Ursus_americanus
Ursus_thibetanus
Helarctos_malayanus
Melursus_ursinus
Kretzoiarctos_beatrix
Tremarctos_ornatus
Indarctos_vireti
Indarctos_arctoides
Indarctos_punjabiensis
Ursavus_brevirhinus
Ballusia_elmensis
Ursavus_primaevus
Ailuropoda_melanoleuca
Agriarctos_spp
Ailurarctos_lufengensis

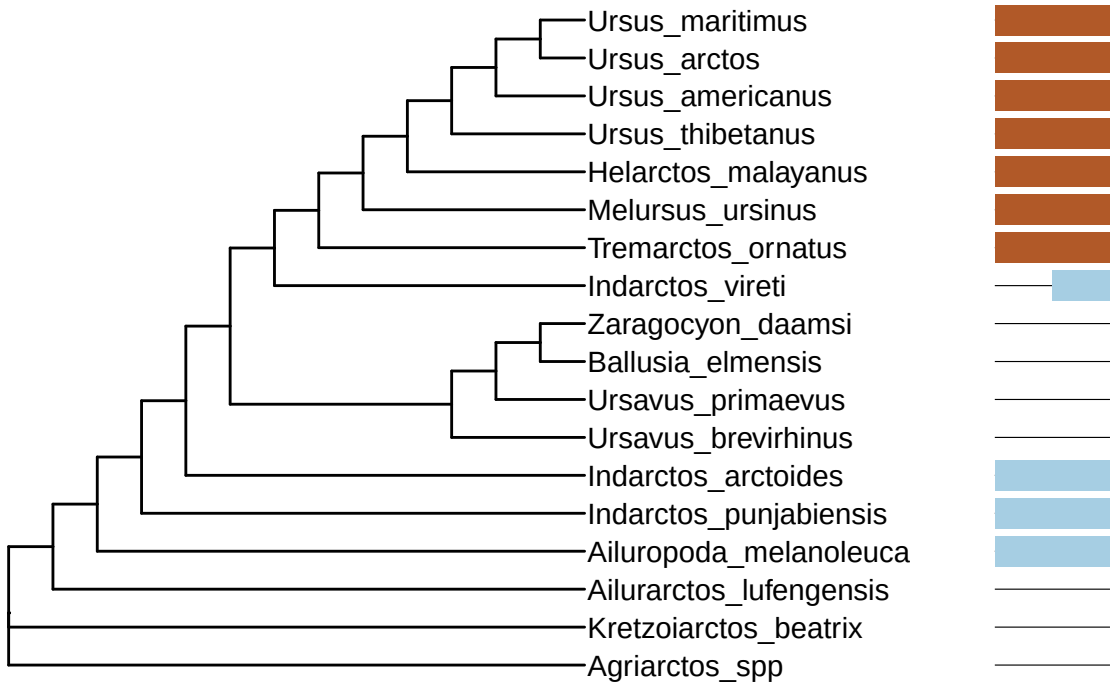
```
##
## [[2]]
```

2
3


Kretzoiarctos_beatrix
Ursus_arctos
Ursus_maritimus
Ursus_americanus
Ursus_thibetanus
Helarctos_malayanus
Melursus_ursinus
Tremarctos_ornatus
Indarctos_vireti
Zaragocyon_daamsi
Ursavus_primaevus
Indarctos_arctoides
Indarctos_punjabiensis
Ursavus_brevirhinus
Ballusia_elmensis
Ailuropoda_melanoleuca
Agriarctos_spp
Ailurarctos_lufengensis

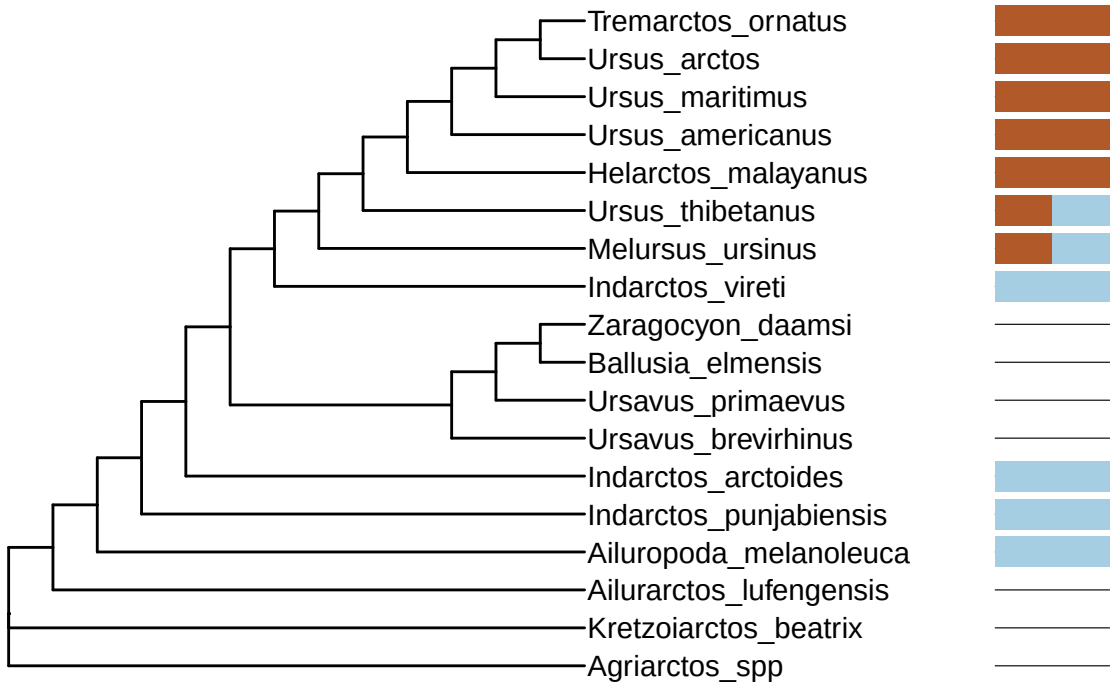
```
##
## [[3]]
```

3
4


Ursus_maritimus
Ursus_arctos
Ursus_americanus
Ursus_thibetanus
Helarctos_malayanus
Melursus_ursinus
Tremarctos_ornatus
Indarctos_vireti
Zaragocyon_daamsi
Ballusia_elmensis
Ursavus_primaevus
Ursavus_brevirhinus
Indarctos_arctoides
Indarctos_punjabiensis
Ailuropoda_melanoleuca
Ailurarctos_lufengensis
Kretzoiarctos_beatrix
Agriarctos_spp

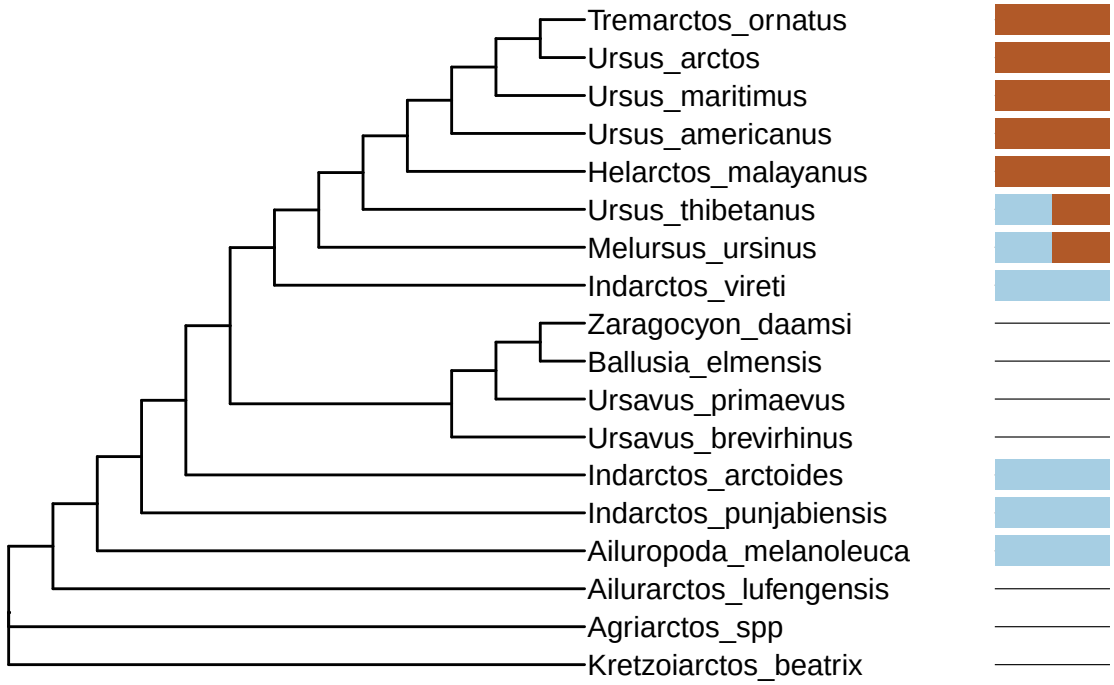
```
##
## [[4]]
```

4
5


Tremarctos_ornatus
Ursus_arctos
Ursus_maritimus
Ursus_americanus
Helarctos_malayanus
Ursus_thibetanus
Melursus_ursinus
Indarctos_vireti
Zaragocyon_daamsi
Ballusia_elmensis
Ursavus_primaevus
Ursavus_brevirhinus
Indarctos_arctoides
Indarctos_punjabiensis
Ailuropoda_melanoleuca
Ailurarctos_lufengensis
Kretzoiarctos_beatrix
Agriarctos_spp

```
##
## [[5]]
```

5
6


Tremarctos_ornatus
Ursus_arctos
Ursus_maritimus
Ursus_americanus
Helarctos_malayanus
Ursus_thibetanus
Melursus_ursinus
Indarctos_vireti
Zaragocyon_daamsi
Ballusia_elmensis
Ursavus_primaevus
Ursavus_brevirhinus
Indarctos_arctoides
Indarctos_punjabiensis
Ailuropoda_melanoleuca
Ailurarctos_lufengensis
Agriarctos_spp
Kretzoiarctos_beatrix

`Phangorn` (Schliep 2011, Schliep2017) requires a starting tree to estimate a parsimony tree. We specify the tree we read in earlier for this purpose. The trees, which were generated with `ggtree` (Yu et al. 2017) have been saved to a vector, which can be displayed in its entirety, or subsetted to look at specific trees.

Questions

1. Visualize characters 1 and 2. What is the parsimony score for this character set? Once you have answered this question,check your answer by adding the parismony score using the `pscore` argument:

```
output_vector <- generate_tree_vis(sample_df = sample_df, alignment =
                                   pscore = TRUE)
```

```
## Final p-score 2 after  0 nni operations
## Final p-score 2 after  0 nni operations
## Final p-score 2 after  0 nni operations
## Final p-score 2 after  1 nni operations
## Final p-score 2 after  1 nni operations
```

2. Visualize characters 2 and 3. What monophyletic group from the tree of characters 1 and 2 is no longer on this tree?
3. What is the parsimony score of the 31-34 character set?
4. Which character, 8, 9 or 10, represents a reversal from a derived state to ancestral?
5. What information would we need to decide if the "1" state possesed by *Zaragocyon_daamsi* in character 52 is an autapomorphy?
6. Do all characters with the same parsimony score have the same likelihood score? You can add the likelihood score to the visualization using the below code:

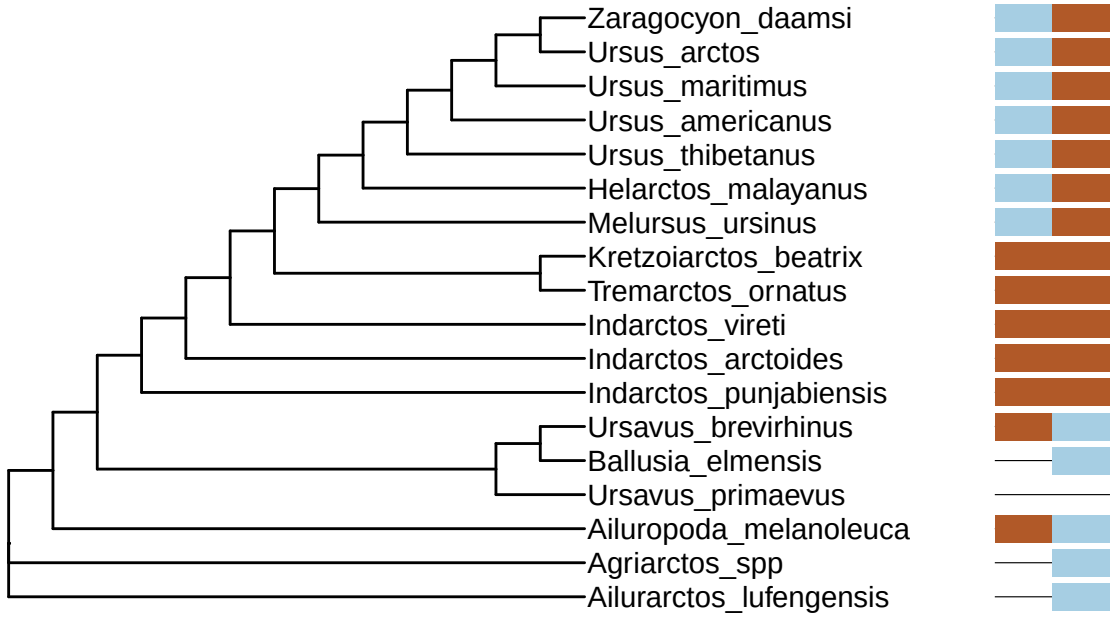
```
output_vector <- generate_tree_vis(sample_df = sample_df, alignment =
                                   pscore = TRUE, lscore = TRUE)
```

```
## Final p-score 2 after  0 nni operations
## Final p-score 2 after  0 nni operations
## Final p-score 2 after  0 nni operations
## Final p-score 2 after  1 nni operations
## Final p-score 2 after  1 nni operations
```

```
output_vector
```

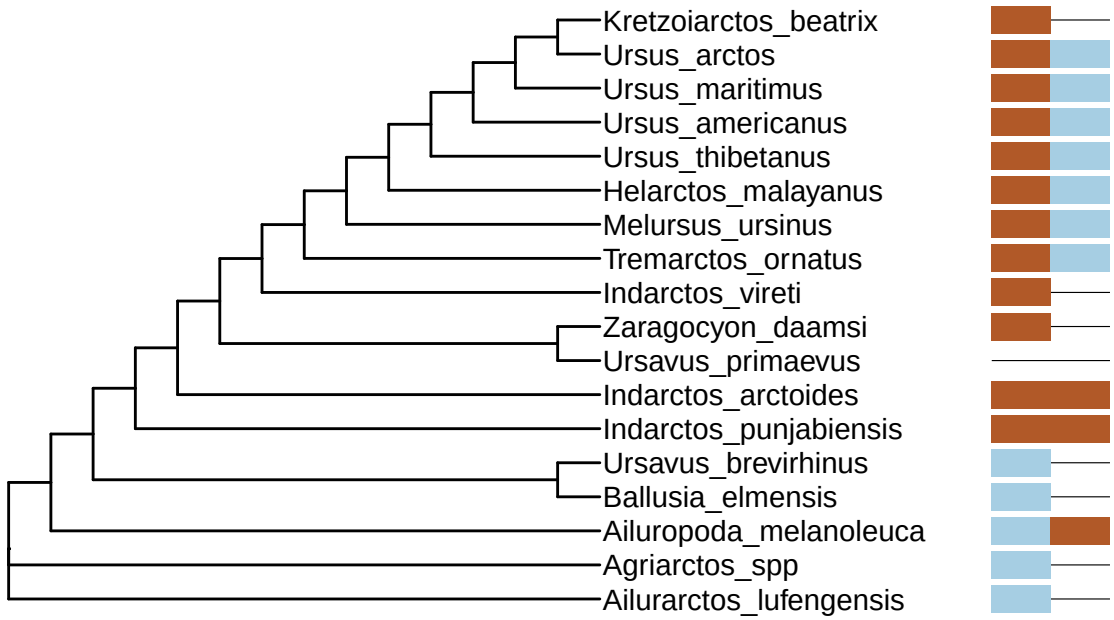
```
## [[1]]
```

1
2
LScore under Mk model −4.19585397701144
PScore 2



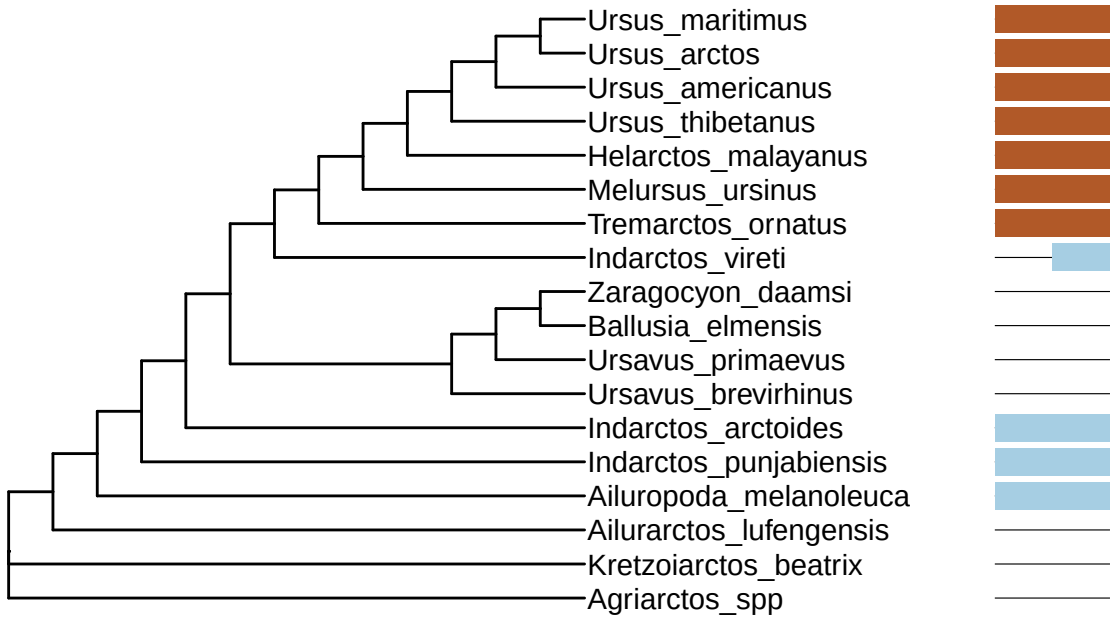

```
##
## [[2]]
```

2
3
LScore under Mk model −4.19585397701144
PScore 2



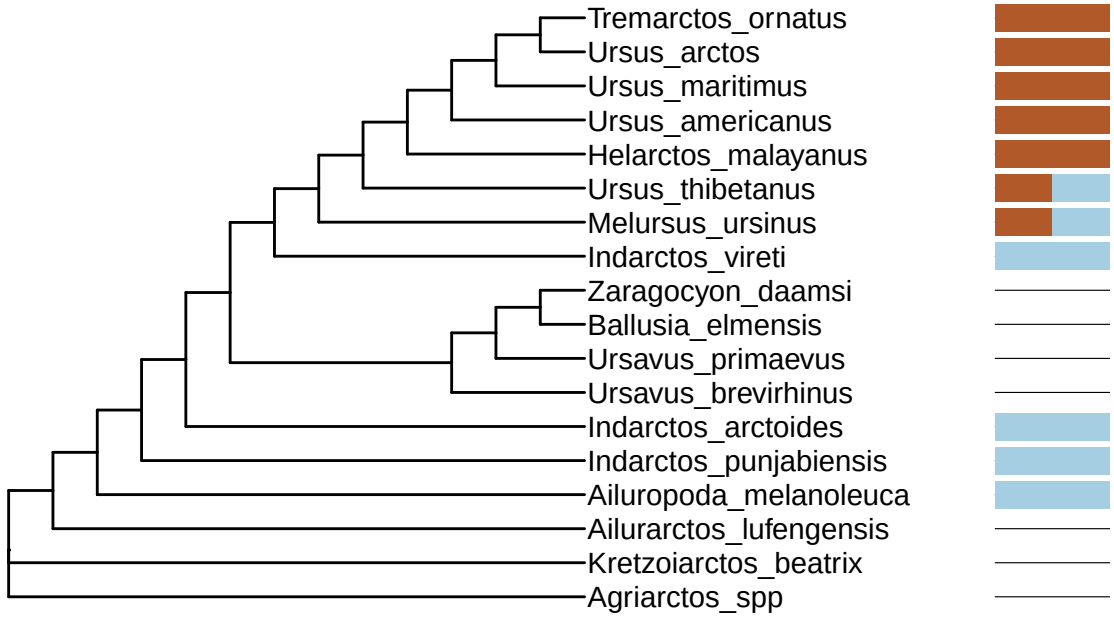
```
##
## [[3]]
```

3
4
LScore under Mk model −3.5027067964515
PScore 2



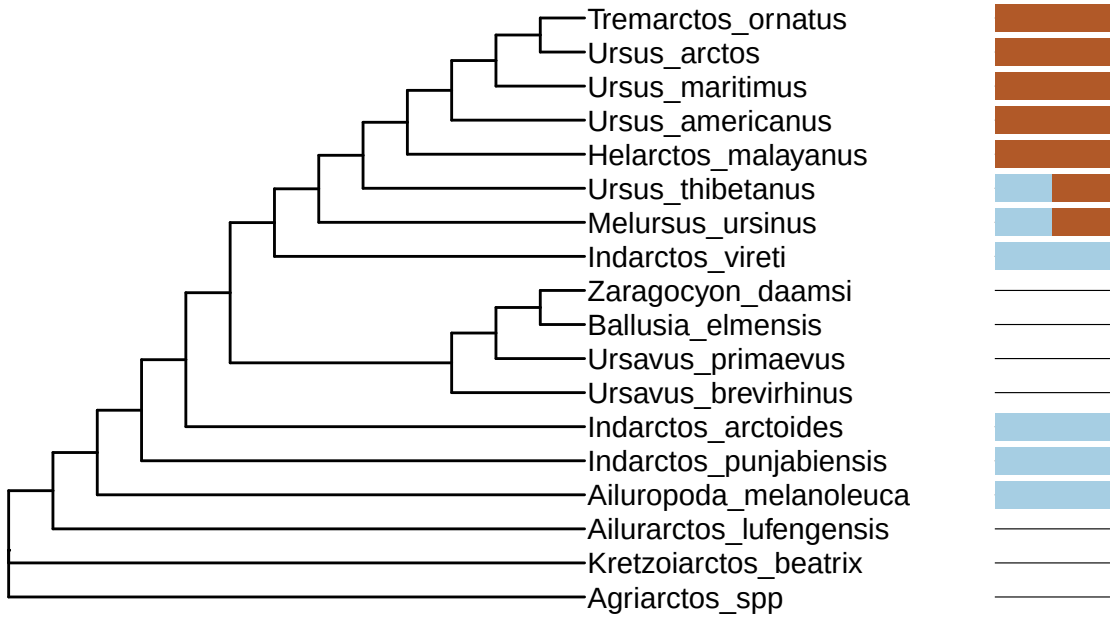

```
##
## [[4]]
```

4
5
LScore under Mk model −4.19585397701144
PScore 2




```
##
## [[5]]
```

5
6
LScore under Mk model −4.19585397701144
PScore 2



7. Compare characters 46-49 and 47-50. Why does set 47-50 have a better likelihood than 46-49?
8. What is the relationship between the likelihood score and increasing the number of characters visualized?
9. What is the minimum number adding a character can add to the parsimony score?
10. These trees are fully resolved. Based on your exploration of the data, does this degree of resolution make sense?

Glossary

Ancestral State: A character state possessed by the ancestor of a group

Autapomorphy: A character state that is unique to a specific taxon.

Derived State: A character state that is different from the ancestral state.

Likelihood Score: The likelihood of the observed data under a specific model.

Maximum likelihood: A phylogenetic optimatlity criterion under which phylogenetic data are modeled according to sets of assumptions. Under this criterion, the tree that has the best ("maximum") likelihood score under the assumed model is to be preferred.

Maximum parsimony: A phylogenetic optimality criterion. This criterion holds that the tree implying the fewest changes in the characters used to generate it should be preferred.

Monophyletic: A group on a phylogeny of an ancestor and all of its descendents.

Parsimony Score: The number of changes implied by a character on a tree.

Reversal: A change from the derived state back to the ancestral state.

References

Abella, Juan, David M. Alba, Josep M. Robles, Alberto Valenciano, Cheyenn Rotgers, Raül Carmona, Plinio Montoya, and Jorge Morales. 2012. "*Kretzoiarctos* Gen. Nov., the Oldest Member of the Giant Panda Clade." *PLoS One* 17: e48985.

Lewis, Paul O. 2001. "A Likelihood Approach to Estimating Phylogeny from Discrete Morphological Character Data." *Systematic Biology* 50 (6). Oxford University Press: 913–25.

R Core Team. 2018. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

RStudio Team. 2015. *RStudio: Integrated Development Environment for R*. Boston, MA: RStudio, Inc. http://www.rstudio.com/.

Schliep, K.P. 2011. "Phangorn: Phylogenetic Analysis in R." *Bioinformatics* 27 (4): 592–93. https://doi.org/10.1093/bioinformatics/btq706.

Yu, Guangchuang, David Smith, Huachen Zhu, Yi Guan, and Tommy Tsan-Yuk Lam. 2017. "Ggtree: An R Package for Visualization and Annotation of Phylogenetic Trees with Their Covariates and Other Associated Data." *Methods in Ecology and Evolution* 8 (1): 28–36. https://doi.org/10.1111/2041-210X.12628.