# *Supplementary Material*

## 1 SUPPLEMENTARY DATA

Supplementary Material should be uploaded separately on submission. Please include any supplementary data, figures and/or tables. All supplementary files are deposited to FigShare for permanent storage and receive a DOI.

Supplementary material is not typeset so please ensure that all information is clearly presented, the appropriate caption is included in the file and not in the manuscript, and that the style conforms to the rest of the article. To avoid discrepancies between the published article and the supplementary material, please do not add the title, author list, affiliations or correspondence in the supplementary files.

### 1.1 Analytical Pipeline

All the analyses were run in the cluster of the Museu de Zoologia da Universidade de São Paulo, with different settings of number of nodes and threads as described for each part of the process.

A total of XXXX loci were recovered with the selected scheme. We used a custom modification of the TICR pipeline (REF) in order to generate gene tree distributions for each of the loci, so that topological uncertainty at the locus level is correctly incorporated into the network estimation down the pipeline.

#### 1.1.1 Bayesian infernce

We used the script `mb.pl` in order to automatically generate input files for MrBayes (REF) and then run them in parallel in:

```
# divide in three nodes manually, and then use 60 threads per node
./mb.pl -params
```

#### 1.1.2 Concordance Factors

We used a custom script for generating concordance factors and their relative frequencies for each quartet in the posterior distribution of each gene tree using `PhyloNetworks` (REF).

```
#!/usr/bin/julia
using Distributed
addprocs(15)
@everywhere using PhyloNetworks
#using PhyloNetworks
using PhyloPlots
using CSV
using DataFrames

gtfilename = ARGS[1]

println("Processing $gtfilename...")

# load input trees
trees = readMultiTopology(gtfilename)
```

```
# calculate CFs
q,t = countquartetsintrees(trees)
df = writeTableCF(q,t)
cfs = readTableCF(df)

# add dummy columns to df so that it is get-pop-tree.pl-compliant
zeros = repeat([0], nrow(df))

df_zeros = DataFrame(CF12_34_lo=zeros,
                     CF12_34_hi=zeros,
                     CF13_24_lo=zeros,
                     CF13_24_hi=zeros,
                     CF14_23_lo=zeros,
                     CF14_23_hi=zeros)

df_ticr = hcat(select(df, 1:5),
               select(df_zeros, 1:2),
               select(df, 6),
               select(df_zeros, 3:4),
               select(df, 7),
               select(df_zeros, 5:6))

CSV.write("CFs_$gtfilename.csv", df_ticr)
```

The concordance factor table `name_of_the_table.csv` that is then used as input for SNAq (REF).

### 1.1.3 Initial Tree

SNAq requires an initial tree for iteratively calculating the pseudolikelihood and refining the search for hybridisations in a dataset. This can be an arbitrary tree, but often an informed tree is useful for speeding up convergence. We used quartet max cut QMC (REF) in order to infer a tree from the concordance factor table. That tree was used as input for SNAq

```
# calculate the starting tree using part of the TICR pipeline
run(Cmd(["/home/balleng/programs/TICR/scripts/./get-pop-tree.pl", "CFs_$gtfil
```

### 1.1.4 Phylogenetic Network Inference

Once the initial tree and concordance factor table were estimated, we used SNAq (REF) for estimation of hybrid edges under three different sequential assumptions: `h = 0`, `h = 1`, `h = 2`, and `h = 3`. SNAq infers the best topologies wich include no, one, two and three hybridisations and score them using the logpseudolikelihood (CHECK THIS). The script below was used for estimating these networks using XX threads in a single core.

```
# read CF table around here

# read starting tree
start_tree = readTopology("CFs_$gtfilename.csv.QMC.tre")
```

```
# calculate a h=0 network
net0 = snaq!(start_tree, cfs, hmax=0, filename="snaq_h0_$gtfilename", seed=12

# calculate a h=1 network
net1 = snaq!(net0, cfs, hmax=1, filename="snaq_h1_$gtfilename", seed=1234, ru

# calculate a h=2 network
net2 = snaq!(net1, cfs, hmax=2, filename="snaq_h2_$gtfilename", seed=1234, ru

#exit julia
exit()
```

### 1.1.5 Postprocessing

Include here any further step such as the test of tree-likeness that we have never run before in the snaq project.

## 2 SUPPLEMENTARY TABLES AND FIGURES

For more information on Supplementary Material and for details on the different file types accepted, please see the Supplementary Material section of the Author Guidelines.

Figures, tables, and images will be published under a Creative Commons CC-BY licence and permission must be obtained for use of copyrighted material from other sources (including re-published/adapted/modified/partial figures and images from the internet). It is the responsibility of the authors to acquire the licenses, to follow any citation instructions requested by third-party rights holders, and cover any supplementary charges.

### 2.1 Figures


**Figure S1.** Enter the caption for your figure here. Repeat as necessary for each of your figures


**Figure S2.** This is a figure with sub figures, **(A)** is one logo, **(B)** is a different logo.