Chapter 20

Fungal Phylogenomics

Robert Riley and Laszlo Nagy

Abstract

Phylogenomics aims to infer the evolutionary relationships of organisms, and their genomes, genes, and proteins, from genomic data. Understanding the evolution of these components can provide clues about their biological functions. Here we describe minimal protocols for inferring families of genes (and the proteins they encode), and using them in phylogenomic analyses to infer species trees.

Key words Phylogenomics, Genomics, Fungi, Phylogeny inference, Phylogenetic tree, MCL clustering, Protein families, Orthologs

1 Introduction

A key insight emerging from early genomics studies [1, 2] was that proteins occur in families: groups of proteins with a common evolutionary origin, inferred by their similar sequence, structure, and biological function. Protein families are groups of homologous proteins, that comprise both orthologs (sets of sequences separated from each other by speciations) and paralogs (resulting from duplications after speciation). Orthologs generally retain the function of the ancestral gene, whereas paralogs can evolve new functions (through neofunctionalization), or the two descendent paralogs might partition the ancestral function between each other (subfunctionalization), an important consideration for choosing gene families for phylogenomics.

Despite floods of genome sequence data the bioinformatic detection of protein families, given a set of predicted genes, and proteins they encode, remains a significant challenge [3–6]. Identifying groups of orthologous proteins is important because they usually have similar functions, and similarity to proteins of known function is often a basis for assigning functional annotations to the proteins predicted from newly sequenced genomes. Moreover, for protein families of known general function, observing the copy number variations (expansions and contractions) can provide insights into the biology of organisms, as reported for the distribution of plant cell wall degrading enzyme genes (e.g., class II lignin peroxidases) in white rot versus brown rot fungi [7].

Markov clustering methods, e.g., OrthoMCL [5], are useful for assigning proteins into families given a matrix of all-vs-all pairwise sequence similarity in a set of proteins. Importantly for phylogenomic analyses, these protein sets can be from multiple organisms. OrthoMCL is an example of readily available software that applies the Markov Cluster algorithm [8], a clustering method for graphs, to the problem of assigning proteins to families. Other MCL-based methods, such as TRIBE-MCL [3], will produce comparable results for the analyses we describe here.

Phylogenetic tree inference [9–11] is generally based on multiple sequence alignments as input. A typical approach is to concatenate the sequences of a few conserved genes, resulting in a "super-sequence" from each organism. Genome sequencing, and the comprehensive sets of proteins they provide, enables us to use hundreds to thousands of orthologous genes and the combined historical evidence they make up for phylogenetic tree inference, thus taking full advantage of genome-wide data. Thus, the differences in the genealogies, evolutionary rates, potential biases and phylogenetic signal between individual gene families—if randomly distributed—are expected to average out and lead to robust support at multiple phylogenetic depths.

Genome-scale data provide a wealth of data for inferring phylogeny. These include ultraconserved genetic elements [12], conserved noncoding sequence regions, and whole transcriptome, genome, or proteome-based methods when complete genome sequences are available. Here, we will focus on the latter. We present a strategy for inferring a phylogenetic tree of a collection of organisms for which we have sequenced genomes and predicted protein-coding gene sets. The strategy is based on identifying single copy gene families (those families where each organism contributes one and only one gene) among MCL-inferred clusters, inferring multiple sequence alignments for the protein sequences, and performing phylogenetic analyses to infer species trees.

2 Materials

2.1 Data Download

We provide an example data based on the dataset from [7]. Download the "filtered" protein sets for the organisms and the downloads section of the given URLs in Table 1.

2.2 Software Download and Install

Download and install the software in Table 2 according to each program's documentation and your system's requirements.

Table 1
Organisms and URLs to access them

Organism	JGI URL	Abbreviation in Fig. 1
Aspergillus niger	http://genome.jgi.doe.gov/Aspergillusniger	asp
Auricularia Subglabra	http://genome.jgi.doe.gov/Auricularia	aur
Batrachochytrium dendrobatidis	http://genome.jgi.doe.gov/Batrachochytrium	bat
Coniophora puteana	http://genome.jgi.doe.gov/Coniophora	con
Coprinopsis cinerea	http://genome.jgi.doe.gov/Coprinopsis	cop
Cryptococcus neoformans	http://genome.jgi.doe.gov/Cryptococcus	cry
Cryphonectria parasitica	http://genome.jgi.doe.gov/Cparasitica	cryp
Dacryopinax primogenitus	http://genome.jgi.doe.gov/Dacryopinax	dac
Dichomitus squalens	http://genome.jgi.doe.gov/Dsqualens	dic
Fomitiporia mediterranea	http://genome.jgi.doe.gov/Fomitiporia	fom
Fomitopsis pinicola	http://genome.jgi.doe.gov/Fomitopsis	fomp
Gloeophyllum trabeum	http://genome.jgi.doe.gov/Gloeophyllum	glo
Heterobasidion annosum	http://genome.jgi.doe.gov/Heterobasidion	het
Laccaria bicolor	http://genome.jgi.doe.gov/Laccaria_bicolor	lac
Malassezia globosa	http://genome.jgi.doe.gov/Malassezia	mal
Melampsora laricis-populina	http://genome.jgi.doe.gov/MelampsoraLaricisPopulinaV2	mel
Phanerochaete chrysosporium	http://genome.jgi.doe.gov/Phanerochaete	phc
Phycomyces blakesleeanus	http://genome.jgi.doe.gov/Phycomyces	phy

(continued)

Table 1
(continued)

Organism	JGI URL	Abbreviation in Fig. 1
Pichia stipitis	http://genome.jgi.doe.gov/pichia	pic
Postia placenta	http://genome.jgi.doe.gov/PplacentaRSB12	pos
Punctularia strigosozonata	http://genome.jgi.doe.gov/Punctularia	pun
Schizophyllum commune	http://genome.jgi.doe.gov/Scommune3	sch
Serpula Lacrymans	http://genome.jgi.doe.gov/Serpula	ser
Sporobolomyces roseus	http://genome.jgi.doe.gov/Sroseus	spo
Stagonospora nodorum	http://genome.jgi.doe.gov/Stagonosporanodorum	sta
Stereum hirsutum	http://genome.jgi.doe.gov/Stereum	ste
Trametes versicolor	http://genome.jgi.doe.gov/Trametes	tra
Tremella mesenterica	http://genome.jgi.doe.gov/Tremella	tre
Trichoderma Reesei	http://genome.jgi.doe.gov/Treesei	tri
Ustilago maydis	http://genome.jgi.doe.gov/Ustilago	ust
Wolfiporia cocos	http://genome.jgi.doe.gov/Wolfiporia	Wol

Table 2
Required software

Program	Version	URL
OrthoMCL	2.0.9	http://orthomcl.org
Python	2.7.4	https://www.python.org
MAFFT	7.221	http://mafft.cbrc.jp/alignment/software/
Gblocks	0.91b	http://molevol.cmima.csic.es/castresana/Gblocks.html
ClustalW	2.1	http://www.clustal.org/clustal2/
RAxML	7.6.3	https://github.com/stamatak/standard-RAxML
FastTree	2.1.9	http://www.microbesonline.org/fasttree/#Install
ETE toolkit	3.0.0b36	http://etetoolkit.org

3 Methods

3.1 Assign Proteins to Families Using OrthoMCL

Run OrthoMCL on your protein set as in http://orthomcl.org/common/downloads/software/v2.0/UserGuide.txt.

Using OrthoMCL v2.0.9, we performed all-vs.-all BLAST with an E-value threshold of 10^{-5} and percent identity threshold of 50% on 383,910 protein sequences predicted from 31 fungal genomes used in [7] (note that several of the genome annotations were updated since 2012, so our numbers might not exactly match that study). The BLAST results were processed using OrthoMCL's scripts and clustering was performed with an inflation parameter of 2.0 (consult current version of software's documentation for details).

Running OrthoMCL on the above dataset resulted in some 32,372 nonsingleton clusters. OrthoMCL also identified some 2,149,920 pairs of orthologous genes (best-hit pairs of proteins that are across two species), 695,564 pairs of proteins whose best hit is within a species, and 634,134 co-orthologs (pairs of proteins across two species where the proteins are connected through both orthology and inparalogy).

3.2 Identify a Single-Copy Gene Set

Next, we identify the set of conserved single-copy clusters, in which each organism contributes only one protein (*see* **Note 1**). Such clusters could be extracted from the OrthoMCL results (e.g., called groups.txt file) using a Python script like the one in Text Box 1.

Text Box 1: Python Script to Extract Single-Copy Clusters from an OrthoMCL Run

```
# Extract single-copy clusters from OrthoMCL run, e.g. file 'groups.txt'
#
# File has one cluster per line, lines look like:

# cluster16469: asp|1176580 cryp|42756 sta|8442 tri|75434
# cluster16470: sch|2664171 dic|139591 fomp|1166292 glo|128976
# cluster16471: asp|1177110 cryp|357688 sta|3771 tri|22774

# Assume your proteins are named e.g. 'asp|1176580' where 'asp' is a species
# name and '1176580' is a unique identifier for that protein in asp, etc.

# Write out those clusters that have one and only one protein from each species

import sys
import re

TRUE = 1
FALSE = 0

try:
  f = open(sys.argv[1], 'r')
except:
  print "Supply a file, e.g. 'groups.txt' from an OrthoMCL run"
  sys.exit()

clusters = {}

# Read in the clusters
for line in f.readlines():
  m = re.match("^(\w+):", line)
  if m:
    cluster_id = m.group(1)
    clusters[cluster_id] = re.findall('(\w+)\|(\d+)', line)

# Determine what species are in the clusters
species = set([s for c in clusters.itervalues() for s, p in c])


# Find the cluster, where one and only one protein is contributed by each species
single_copy_clusters = {}
N = len(species)
for k, c in clusters.iteritems():
  if len(c) == N:
    species_in_cluster = set([s for s, p in c])  # Give the list of species in a cluster
    if len(species_in_cluster) == N:
      single_copy_clusters[k] = c

sys.stderr.write("%d single-copy clusters obtained\n" % len(single_copy_clusters))

for k, c in single_copy_clusters.iteritems():
  sys.stdout.write("%s:" % k)  # Write the cluster id
  for i in range(N):  # Write each protein in the cluster
    s, p = c[i]
    sys.stdout.write(" %s|%s" % (s, p))
  sys.stdout.write("\n")
```

Such a script could be run using the UNIX command line:

```
python orthomcl_single_copy.py groups.txt > sin-
gle_copy.txt
```

Using this script, we extracted 510 single-copy clusters from the OrthoMCL results file groups.txt.

3.3 Concatenate Protein Sequences from the Single-Copy Clusters

To produce a FASTA-format input file for multiple sequence alignment, we concatenate each organism's protein sequences from the single-copy clusters, using a Python script as in Text Box 2. *See* also **Note 2**. on concatenation after multiple sequence alignment.

Text Box 2: Python Script to Produce a FASTA File with Concatenated Sequences from the Single-Copy OrthoMCL Clusters

```
# Read in a list of single-copy clusters from some MCL-based clustering method
# (e.g. OrthoMCL), and concatenate the sequences for each organism.  For each
# organism, write out one FASTA sequence containing the concatenated sequences,
# in the same cluster order for each organism.

import sys
import re
from Bio import SeqIO
from Bio.Seq import Seq
from Bio.SeqRecord import SeqRecord
from Bio.Alphabet import IUPAC

all_fastas = {}
clusters = []

try:
  try:
    fasta_sequences = SeqIO.parse(open(sys.argv[1]),'fasta')
    for fasta in fasta_sequences:
      name, sequence = fasta.id, str(fasta.seq)
      all_fastas[name] = fasta  # fasta is a SeqRecord object.  See http://biopython.org
  except:
    sys.stderr.write("ERROR: supply a valid FASTA file\n")
    sys.exit()
  try:
    f = open(sys.argv[2], 'r')
    for line in f.readlines():
      c = re.findall('(\w+)\|(\d+)', line)
      if c:
        clusters.append(c)
  except:
    sys.stderr.write("ERROR: supply a cluster file\n")
    sys.exit()
  try:
    out_file = open(sys.argv[3], 'w')
  except:
    sys.stderr.write("ERROR opening %s\n" % sys.argv[3])
    sys.exit()
except:
  print "ARGUMENTS: <input fasta file> <single copy clusters file> <output fasta file name>"
  sys.exit()

all_species = set([s for c in clusters for s, p in c])

concat_fastas = []  # Initialize dictionary

for species in all_species:
  concat_seq = ''
  for c in clusters:
    for s, p in c:
      if s == species:
        concat_seq += str(all_fastas[s + '|' + p].seq)
  concat_fastas.append(SeqRecord(Seq(concat_seq, IUPAC.protein), id=species, description=""))

SeqIO.write(concat_fastas, out_file, "fasta")
```

We would run this script as, for example:

```
python concatenate_seq.py allprot.fasta single_
copy.txt concat.fasta
```

3.4 Multiple-Align Sequences with MAFFT

Use MAFFT [13], to align the concatenated sequences (*see* **Note 2**). We generally run MAFFT with automatically determined run options as follows:

```
mafft --auto concat.fasta > concat.mafft
```

The –auto option determines an optimal tradeoff between speed and accuracy—for larger datasets it will choose one of the speed-oriented built-in algorithms of the software. On our data set, MAFFT v7.182 was able to align the 31 concatenated sequences, totaling about 2.7 Mb of data, in a few hours on a fairly typical Linux machine (compute time will vary depending on hardware and other considerations). Note that accuracy can be prioritized using other options of MAFFT (e.g., L-INS-I, G-INS-I, *see* also notes below) unless run times become prohibitive.

3.5 Extracting Well-Aligned Regions

If we manually inspect the resulting MAFFT alignment, we see that it contains a significant number of regions of low alignment quality. This can be caused by multiple factors, including protein sequence divergence [14], insertions and deletions, potential gene fragments and inaccuracies in the alignment. Gapped and highly variable alignment regions may introduce noise into the analyses, which can result in low signal-to-noise ratios during phylogenetic inference. Therefore it is desirable to extract only the well-aligned portions of the alignment. Gblocks [15] is the most widely used method for this, although more recent tools, such as Trim-al [16] and Aliscore [17] also exist. Gblocks searches for contiguous stretches of well-aligned regions of a minimum length flanked by regions containing gaps or low overall alignment score. To extract the well-aligned regions from the MAFFT alignment, run Gblocks as follows:

```
Gblocks concat.mafft -t=p -e=-gb1 -b4=5
```

The Gblocks output indicates that the original alignment consisted of 517,466 positions, whereas in the reduced Gblocks alignment, consisting of some 4736 conserved blocks, there are now 121,960 positions (23%).

3.6 Compute Neighbor Joining Tree with ClustalW

Use ClustalW to compute a tree using the neighbor-joining algorithm [18]:

```
clustalw2 -tree -infile=concat.mafft-gb1
```

This tree should take seconds to compute on a typical Linux machine. The tree file produced is `concat.ph`.

3.7 Convert Gblocks Output to Phylip Format

To prepare to run RAxML, use ClustalW to convert the Gblocks output to Phylip format:

```
clustalw2 -convert -infile=concat.mafft-gb1 -
output=phylip \
-outfile=concat.mafft-gb1.phy
```

This step is necessary because, depending on version, RAxML may require that the alignment is input in Phylip format, not the FASTA format output by Gblocks.

3.8 Compute Maximum Likelihood Species Phylogeny Using RAxML

Run the multithreading-enabled version of RAxML (a multicore computer with 16 cores is assumed for this example—specified with the parameter "–T") as follows:

Variant 1. Maximum likelihood (ML) search without bootstrap:

```
raxmlHPC-PTHREADS-SSE3 -T 16 -s concat.mafft-gb1.
phy -n \
concat.mafft-gb1.phy.RAxML -o bat -p 12345 -m
PROTGAMMAWAG
```

Variant 2. ML tree search with 100 rapid bootstrap replicates:

```
raxmlHPC-PTHREADS-SSE3 -T 16 -s concat.mafft-gb1.
phy -n \
concat.mafft-gb1.phy.RAxML -o bat -f a -x 12345 -p
12345 \
-# 100 -m PROTGAMMAWAG
```

Variant 3. ML tree search with more thorough bootstrap analysis:

```
raxmlHPC-PTHREADS-SSE3 -T 16 -s concat.mafft-gb1.
phy -n \
concat.mafft-gb1.phy.500bootstrap.RAxML -b 12345 -p
12345 \
-# 500 -m PROTGAMMAWAG
```

If bootstrapping and ML tree inference are separated in time, bootstrap frequencies will need to be mapped on the ML tree (or any other tree of interest), which can be done using the SumTrees script of the Dendropy package [19]:

```
sumtrees.py --decimals=0 --percentages \
--output-tree-filepath=ML_tree_annotated500boot-
strap.tre \
--target=concat.mafft-gb1.phy.RAxML \
concat.mafft-gb1.phy.500bootstrap.RAxML
```

Partitioned models can provide much better fit to the data (see below) and thus their use is recommended for careful tree searches. A partition table for RAxML has the definition and model for each gene on a separate line:

```
WAG, Cluster5679 = 1 - 194
WAG, Cluster4143 = 195 - 732
WAG, Cluster5655 = 733 - 887
...
```

We can perform partitioned tree search and bootstrap analysis as follows:

```
raxmlHPC-PTHREADS-SSE3 -T 16 -s concat.mafft-gb1.
phy -n \
concat.mafft-gb1.phy.RAxML -o bat -p 12345 -q par-
tition.table \
-m PROTGAMMAWAG
raxmlHPC-PTHREADS-SSE3 -T 16 -s concat.mafft-gb1.
phy -n \
concat.mafft-gb1.phy.500bootstrap.RAxML -b 12345 -p
12345 \
-# 500 -q partition.table -m PROTGAMMAWAG
```

3.9 Compute Species Tree Using FastTree

Alternatively, we can use FastTree (which, as its name suggests, generally runs faster) to compute an approximately maximum likelihood tree as follows:

```
FastTreeMP -wag < concat.mafft-gb1.phy > concat.
mafft-gb1.phy.wag.ft
```

3.10 Compare the Trees Obtained with Different Methods

How do the trees generated with RAxML, FastTree, and ClustalW compare with the published tree from [7]? We compare the trees using the ETE Toolkit [20] as follows:

```
ete3 compare -t RAxML_bestTree.concat.mafft-gb1.
phy.RAxML \
concat.mafft-gb1.phy.wag.ft concat.ph -r floud-
as_2012.ph \
--unrooted
```

The `-r` option sets the published tree as a reference tree to compare the others to. We see in Table 3 that the trees generated with the more computationally expensive methods (maximum likelihood RAxML and approximately maximum likelihood FastTree), are somewhat more similar to the published tree than the NJ tree, as indicated by a shorter Robinson–Foulds distance [21] and greater percentage of shared edges. Although in this case the difference between the NJ tree and the ML trees are small, the this difference can be significant for larger and/or more challenging datasets, e.g., trees with short internal branches, long branches (long branch attraction) or in the presence of rate variation across genes or branches of the tree (*see* **Note 3**). Notice that while the tree topologies from the NJ and ML analyses are mostly identical, there are some differences (*see* **Note 4**).

Table 3
Comparison of trees using ETE Toolkit

Source target tree used	RAxML	FastTree	NJ
Effective tree size used for comparisons (after pruning not shared items)	31	31	31
Normalized Robinson–Foulds distance (RF/maxRF)	0.11	0.11	0.14
Robinson–Foulds symmetric distance	6	6	8
Maximum Robinson–Foulds value for this comparison	56	56	56
Frequency of edges in target tree found in the reference (1.00 = 100% of branches are found)	0.95	0.95	0.93
Frequency of edges in the reference tree found in target (1.00 = 100% of branches are found)	0.95	0.95	0.93

4 Notes

1. The quality of genome-scale datasets for phylogenomic inference largely determines the outcome of the analyses. The aim of dataset assembly is to maximize the amount of reliable phylogenetic information but minimize noise in the datasets. Some considerations for assembling maximally informative datasets follow.

   The number of single copy genes, and thus the amount of universally available information, naturally decreases as the number of species increases, due to rarely occurring gene duplications, even in housekeeping gene families. When analyzing a large number of species (>30) gene tree-based methods for identifying suitable marker genes may yield more genes that can be used for phylogenomic reconstruction (*see* methods in [22] for details). In this case, gene trees are used to distinguish deep paralogs (genes which were duplicated prior to the last speciation events) from inparalogs (paralogs that arise in terminal nodes of the species tree, i.e., species-specific). Note that deep paralogs interfere with species tree estimation, whereas inparalogs do not—the choice of which inparalog to retain for phylogenetic analysis can be arbitrary, or can be based on their distance from the root of the tree. Collections of gene trees can be screened for deep paralogs using the scripts published in [22].

Another factor that should be considered during the assembly of phylogenomic datasets is contamination by highly divergent genes (e.g., due to sequencing errors or pseudogenes). Excessively long branches in individual gene trees can are usually signs of such contamination: a general rule of thumb (but quite liberal cutoff) is to exclude genes whose branch length accounts for >60% of the sum of all branch lengths in the gene tree [23]. Lower cutoff values will result in less contamination by divergent genes, but the cutoff should depend on the number of species in the dataset too (e.g., in a 4-species tree, if all branches are of equal length then each branch accounts for 20% of the total tree length).

The identified single-copy clusters usually show a decreasing trend in taxon occupancy: some clusters contain sequences for all species, whereas from most clusters few or more species will be missing due to functional constraints or the incompleteness of genomic assemblies and annotations [24]. Incomplete clusters are still useful for phylogenomic inference, although nonrandomly distributed missing data can compromise results [25]. Some authors use only orthologous genes in which all the species are represented, while others apply taxon occupancy cut-off. Considering there is a tradeoff between taxon occupancy and combined alignment length and that concatenated phylogenetic analyses are generally robust to even high amounts (50–80%) of missing data, when the distribution of missing data is random, we recommend a taxon occupancy cutoff of 50%—only gene families that contain genes for >50% of the total number of species will be included in the analyses. Naturally, this cutoff can be adjusted to the specific phylogenetic exercise, dataset size and availability of genomic data.

2. As an alternative to a priori concatenation of sequences, single gene alignments are often inferred first, followed by the concatenation of quality-filtered alignments. This strategy preserves gene boundaries, allows for both concatenation and summary-based phylogenetic methods (which combine data from individual gene trees to infer a species tree) to be applied to the data, and may be substantially less computationally intensive. The implementation of this alternate approach is left as an exercise to the reader. Inferring alignments for each gene can be done by MAFFT (*see* above), or by alternative approaches such as the probabilistic method PRANK [26], which is among the most accurate multiple sequence alignment software available (which comes at a cost of longer run time). Note that PRANK produces more fragmented, but generally more accurate alignments.

3. In Maximum Likelihood or Bayesian methods, the evolutionary model used to model nucleotide or amino acid substitutions is an important parameter to consider. This is particularly true for phylogenomic analyses, where biases, such as long branch attraction, can become pronounced due to poor model fit in larger datasets [25] under some circumstances. Software like jModeltest [27] and Partitionfinder [28] can be used to identify best-fit models for each gene in the dataset, although run times often limit the use of these methods and constrain the analyses to be performed with ad hoc selected models. In these cases exploring the sensitivity of the results to alternative commonly used evolutionary models is recommended. Advanced models that can account for incongruence among gene genealogies (e.g., incomplete lineage sorting) or differences in the rate of evolution across sites or in time (heterotachy) are now available for the analysis of genome-scale data [25, 29]. One very straightforward way to improve the fit of the model to the data is the use of partitioned models. Partitioning is an important factor in accounting for data heterogeneity and different evolutionary rates. Most commonly, datasets are partitioned by gene, however, other partitioning schemes, e.g., binning genes by evolutionary rate are also commonly used. Finally, while concatenation-based methods can be sensitive to certain biases, summary-based methods that combine information from individual gene trees into a species tree hold promise to evade some of these caveats [30, 31].

   The promise of phylogenomics has been to eliminate uncertainty from the reconstructions of evolutionary relationships [32]. However, this turned out to be an optimistic expectation [33] as a number of case studies reporting spurious, but strongly supported, relationships came to light. It is therefore very important to assess the robustness of the inferred relationships under multiple parameter combinations. These can include phylogeny reconstruction under multiple partitioning schemes (e.g., partitioned vs. unpartitioned), methods (ML vs. Bayesian), evolutionary models or data selection strategies. Although a detailed review of these strategies is beyond the scope of this chapter, we have highlighted several possibilities for exploring the robustness of results attained using the outlined protocols.

4. The tree topologies from the various analyses are mostly in agreement, but as is to be expected with phylogenomic analyses, there are some differences (Fig. 1). The NJ and ML trees differ in their relative placement of the three clades shown in Fig. 1. In the NJ tree (Fig. 1, panel A) clade 1, containing the orders Polyporales (*Postia placenta*, *Wolfiporia cocos*, *Fomitopsis pinicola*, *Trametes versicolor*, *Dichomitus squalens*, and

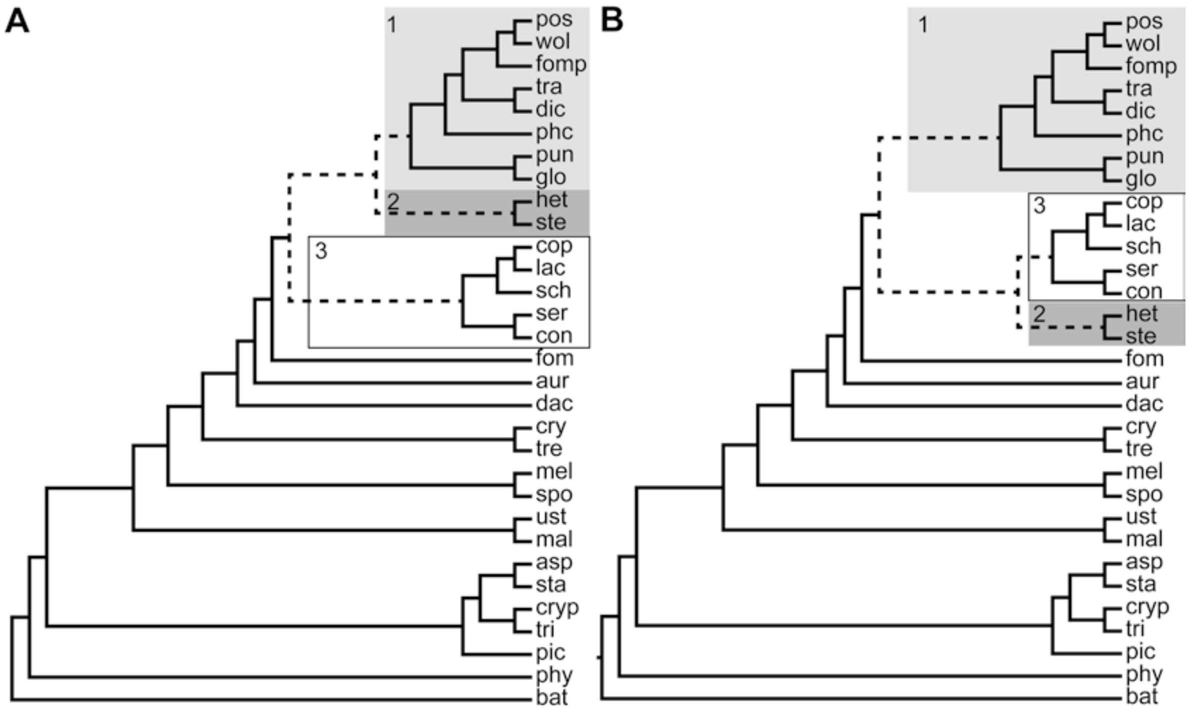

Fig. 1 Difference in tree topologies of ML and NJ trees. The branches that differ between the trees are indicated in dashed grey lines. The NJ tree (panel **a**) places clade 1 as a sister to clade 2, and clade 3 sister to them. The ML tree (panel **b**) places clade 2 as sister to clade 3, with clade 1 sister to them

Phanerochaete chrysosporium), Corticiales (*Punctularia strigosozonata*), and Gloeophyllales (*Gloeophyllum trabeum*), is a sister group (meaning that they have the same parent node) to clade 2, which consists of fungi of the order Russulales (*Heterobasidion irregulare* and *Stereum hirsutum*). Clade 3, containing the orders Agaricales (*Coprinopsis cinerea*, *Laccaria bicolor*, and *Schizophyllum commune*) and Boletales (*Serpula lacrymans* and *Coniophora puteana*) is a sister group to the clade made up of clades 1 and 2. However, in the ML analysis (Fig. 1, panel b), clades 2 and 3 are sister groups to each other, and clade 1 is sister to their combined clade. The published tree in [7], which used BEAST, a Bayesian method [34], is again slightly different, and places *P. strigosozonata* and *G. trabeum* in a separate clade sister to a clade containing the Agaricales, Boletales, Russulales, and Polyporales. The correct answer is an open research question, and the phylogeny of the fungi is continually being revised as new genomic data become available [7, 14, 22, 35, 36]. We thus see that varying the phylogenetic inference method used, data sets, and data selection strategies, can yield slightly different results. The wise researcher is advised to try them all.

Acknowledgments

The work conducted by the US Department of Energy Joint Genome Institute, a DOE Office of Science User Facility, is supported under Contract No. DE-AC02-05CH11231. LGN was supported by the "Momentum" Program LP 2014/12.

References

1. Goffeau A, Barrell BG, Bussey H et al (1996) Life with 6000 genes. Science 274. 546, 563-547
2. Lander ES, Linton LM, Birren B et al (2001) Initial sequencing and analysis of the human genome. Nature 409:860–921
3. Enright AJ, Van Dongen S, Ouzounis CA (2002) An efficient algorithm for large-scale detection of protein families. Nucleic Acids Res 30:1575–1584
4. Finn RD, Coggill P, Eberhardt RY et al (2016) The Pfam protein families database: towards a more sustainable future. Nucleic Acids Res 44:D279–D285
5. Li L, Stoeckert CJ Jr, Roos DS (2003) OrthoMCL: identification of ortholog groups for eukaryotic genomes. Genome Res 13:2178–2189
6. Tatusov RL, Fedorova ND, Jackson JD et al (2003) The COG database: an updated version includes eukaryotes. BMC Bioinformatics 4:41
7. Floudas D, Binder M, Riley R et al (2012) The Paleozoic origin of enzymatic lignin decomposition reconstructed from 31 fungal genomes. Science 336:1715–1719
8. Van Dongen S (2000) A cluster algorithm for graphs. Report-information systems:1–40
9. Thompson JD, Gibson T, Higgins DG (2002) Multiple sequence alignment using ClustalW and ClustalX. Curr Protoc Bioinformatics. https://doi.org/10.1002/0471250953.bi0203s00. 2.3. 1-2.3. 22
10. Price MN, Dehal PS, Arkin AP (2009) FastTree: computing large minimum evolution trees with profiles instead of a distance matrix. Mol Biol Evol 26:1641–1650
11. Stamatakis A (2014) RAxML version 8: a tool for phylogenetic analysis and post-analysis of large phylogenies. Bioinformatics 30: 1312–1313
12. Faircloth BC, McCormack JE, Crawford NG, Harvey MG, Brumfield RT, Glenn TC (2012) Ultraconserved elements anchor thousands of genetic markers spanning multiple evolutionary timescales. Syst Biol 61:717–726
13. Katoh K, Standley DM (2013) MAFFT multiple sequence alignment software version 7: improvements in performance and usability. Mol Biol Evol 30:772–780
14. James TY, Kauff F, Schoch CL et al (2006) Reconstructing the early evolution of fungi using a six-gene phylogeny. Nature 443:818–822
15. Castresana J (2000) Selection of conserved blocks from multiple alignments for their use in phylogenetic analysis. Mol Biol Evol 17:540–552
16. Capella-Gutierrez S, Silla-Martinez JM, Gabaldon T (2009) trimAl: a tool for automated alignment trimming in large-scale phylogenetic analyses. Bioinformatics 25:1972–1973
17. Misof B, Misof K (2009) A Monte Carlo approach successfully identifies randomness in multiple sequence alignments: a more objective means of data exclusion. Syst Biol 58:21–34
18. Saitou N, Nei M (1987) The neighbor-joining method: a new method for reconstructing phylogenetic trees. Mol Biol Evol 4:406–425
19. Sukumaran J, Holder MT (2010) DendroPy: a python library for phylogenetic computing. Bioinformatics 26:1569–1571
20. Huerta-Cepas J, Serra F, Bork P (2016) ETE 3: reconstruction, analysis, and visualization of phylogenomic data. Mol Biol Evol 33:1635–1638
21. Robinson DF, Foulds LR (1981) Comparison of phylogenetic trees. Math Biosci 53:131–147
22. Nagy LG, Riley R, Tritt A et al (2016) Comparative genomics of early-diverging mushroom-forming fungi provides insights into the origins of lignocellulose decay capabilities. Mol Biol Evol 33:959–970
23. dos Reis M, Inoue J, Hasegawa M, Asher RJ, Donoghue PC, Yang Z (2012) Phylogenomic datasets provide both precision and accuracy in estimating the timescale of placental mammal phylogeny. Proc Biol Sci 279:3491–3500
24. Parra G, Bradnam K, Korf I (2007) CEGMA: a pipeline to accurately annotate core genes in eukaryotic genomes. Bioinformatics 23:1061–1067

25. Philippe H, Brinkmann H, Lavrov DV et al (2011) Resolving difficult phylogenetic questions: why more sequences are not enough. PLoS Biol 9:e1000602
26. Loytynoja A (2014) Phylogeny-aware alignment with PRANK. Methods Mol Biol 1079:155–170
27. Posada D (2008) jModelTest: phylogenetic model averaging. Mol Biol Evol 25:1253–1256
28. Lanfear R, Calcott B, Ho SY, Guindon S (2012) PartitionFinder: combined selection of partitioning schemes and substitution models for phylogenetic analyses. Mol Biol Evol 29:1695–1701
29. Lartillot N, Philippe H (2004) A Bayesian mixture model for across-site heterogeneities in the amino-acid replacement process. Mol Biol Evol 21:1095–1109
30. Mirarab S, Reaz R, Bayzid MS, Zimmermann T, Swenson MS, Warnow T (2014) ASTRAL: genome-scale coalescent-based species tree estimation. Bioinformatics 30:i541–i548
31. Szollosi GJ, Tannier E, Daubin V, Boussau B (2015) The inference of gene trees with species trees. Syst Biol 64:e42–e62
32. Gee H (2003) Evolution: ending incongruence. Nature 425:782
33. Jeffroy O, Brinkmann H, Delsuc F, Philippe H (2006) Phylogenomics: the beginning of incongruence? Trends Genet 22:225–231
34. Drummond AJ, Suchard MA, Xie D, Rambaut A (2012) Bayesian phylogenetics with BEAUti and the BEAST 1.7. Mol Biol Evol 29:1969–1973
35. Binder M, Justo A, Riley R et al (2013) Phylogenetic and phylogenomic overview of the Polyporales. Mycologia 105:1350–1373
36. Hibbett DS, Binder M, Bischoff JF et al (2007) A higher-level phylogenetic classification of the fungi. Mycol Res 111:509–547