1 The Physcraper Framework

The general Physcraper framework consists of 4 general steps: **1) identifying and processing** a phylogenetic **tree to update and its underlying alignment**; **2) performing a constrained BLAST search** of sequences from the original alignment on the GenBank DNA database, and filtering of newly found sequences; **3) profile-aligning new sequences** that passed the filtering to the original alignment; **4) performing a phylogenetic analysis and comparing the updated tree** to previous phylogenetic estimates in the focus group. Next, we will describe each step with some technical detail.

1.1 The inputs: a tree and an alignment

In order to take advantage of the OpenTree tools, it is recommended that the input tree is either stored in the OpenTree Phylesystem, or submitted via OpenTree's curator application (McTavish *et al.* 2015). Currently, only trees connected to a published study can be stored in the Phylesystem. Users can choose from among the 2, 950 studies in OpenTree's Phylesystem that have alignments on TreeBASE. If the user is not ready to make the input tree public, tree tip labels can be standardized to the unified OpenTree taxonomy using OpenTree's bulk Taxonomic Name Resolution Service TNRS tool. This step is referred to as taxonomic name mapping and Phylesystem stored trees are processed in this way upon submission. Physcraper saves a summary "csv" file with results from the taxon name standardization for advantage of the user, including the mappings to unique identifiers in the OpenTree and NCBI taxonomies. If taxon names can't be mapped, their taxonomic information is not used in comparison analysis. These taxa will still be used in the sequence search and phylogenetic reconstruction steps. Mapping tip names to OpenTree's unified taxonomy saves a set of user defined characteristics that are essential for automatizing the phylogeny updating process. The most relevant of these is the standardized taxonomic names and the definition of ingroup and outgroup taxa, allowing to automatically set the root for the updated tree on the final steps of the pipeline.

The input alignment should be a single locus alignment that was used in part or in whole, to generate the tree. Alignments are often stored in a public repository such as TreeBase (Piel *et al.* 2009; Vos *et al.* 2012), DRYAD (www.datadryad.org), or a data repository associated with the journal where the tree was

originally published. If the alignment is stored in TreeBase, Physcraper can download it directly, either from the TreeBASE website (www.treebase.org) or through the TreeBASE GitHub repository (SuperTreeBASE; github.com/TreeBASE/supertreebase). If the alignment is on another repository, or constitutes personal data, a path to a local copy of the alignment has to be provided.

Single locus alignments sometimes have fewer taxa than the tree inferred from the full concatenated data, simply because a single molecular marker usually does not cover all the taxa sampled for the full phylogenetic analysis. Physcraper prunes the input tree to taxa found in the alignment, and verifies that all taxon names on the tips of the tree are in the DNA character matrix and vice versa. Technically, just one taxon name (and its corresponding sequence in the alignment) is needed to continue the algorithm. The standardized and pruned tree and alignment (checked tree and alignment from now on) are output as "newick" and "fasta" respectively in the "inputs" folder to be used in the following steps.

1.2 DNA sequence search and filtering

Physcraper uses the GenBank DNA database as source to search for new sequences. The DNA sequence search can be performed on the GenBank remote database or in a GenBank local database set up by the user, which can speed up the search process. Detailed instructions to setup a local database are provided on Physcraper's software documentation.

The next step is to identify a "search taxon" to constrain the sequence search on the GenBank database within that taxonomic group. The search taxon can be chosen by the user from the NCBI taxonomy. If none is provided, then the search taxon is automatically set using the taxa in the input tree labeled as the "ingroup" (Fig. **??**). The search taxon is The Most Recent Common Ancestor (MRCA) of the ingroup taxa in the OpenTree synthetic tree, that is also a named clade in the NCBI taxonomy. This is known in the OpenTree as the Most Recent Common Ancestral Taxon (MRCAT; also referred as the Least Inclusive Common Ancestral taxon - LICA). The MRCAT can be different from the phylogenetic MRCA when the latter is an unnamed clade in the synthetic tree. To identify the MRCAT of a group of taxon names, we use the OpenTree API (Rees & Cranston 2017).

Users can provide a search taxon that is either a more or a less inclusive clade relative to the ingroup of the original phylogeny. If the search taxon is more inclusive, the sequence search will be performed outside the MRCAT of the matched taxa, e.g., including all taxa within the family or the order that the ingroup belongs to. If the search taxon is a less inclusive clade, the users can focus on enriching a particular clade/region within the ingroup of the phylogeny.

The Basic Local Alignment Search Tool, BLAST (Altschul *et al.* 1990, 1997) is used to identify similarity between DNA sequences within the search taxon in a nucleotide database, and the sequences on the checked alignment. The `blastn` function from the BLAST command line tools (Camacho *et al.* 2009) is used for local database sequence searches. For remote database searches, we modified the BioPython (Cock *et al.* 2009) BLAST function from the NCBIWWW module to accept an alternative BLAST address (URL). This is useful when a user has no access to the computer capacity needed to setup a local database, and a local blast database can be set up on a remote machine to BLAST avoiding NCBI's required waiting times, which slow down the searches markedly. A constrained BLAST search is performed, in which each sequence in the alignment is BLASTed once against all database DNA sequences belonging to the search taxon. All results from each BLAST run are stored, and sequences with match scores better than the e-value cutoff (default to 0.00001) are saved along with their corresponding metadata, i.e., their GenBank accession number. The full sequence for each match is downloaded from NCBI into a dedicated library within the "physcraper" folder, allowing for secondary analyses to run significantly faster.

BLAST result sequences will be discarded if they fall outside the user set min and max length cutoffs, set as proportions of the average length without gaps of sequences in the input alignment (defaults values of 80% and 120%, respectively). This filtering guarantees the exclusion of whole genome sequences, which create problems in multiple sequence alignment. The GenBank accession numbers of sequences removed due to not meeting e-value or length cutoffs are stored in output files. All sequences accepted up to this point are assigned an internal identifier. New sequences that are either identical or a subset of any existing sequence in the input alignment are discarded, unless they represent a different taxon in the OTT taxonomy or the NCBI taxonomy, or they are longer than the sequence in the input alignment. Among the filtered sequences, there

are often several representatives per taxon. Although it can be useful to keep some of them, for example, to investigate monophyly within species, there can be hundreds of exemplar sequences per taxon for some markers. To control the number of sequences per taxon in downstream analyses, 5 sequences per taxon are chosen at random. This number is set by default but can be modified by the user.

All BLAST and filtering parameters can be customized by the user. Reverse, complement, and reverse-complement BLAST result sequences are identified and translated using BioPython internal functions (Cock *et al.* 2009). Iterative cycles of sequence similarity search can be performed, by blasting the newly found sequences until no new sequences are found. By default only one BLAST search cycle is performed in which only sequences in the input alignment are blasted. New sequences passing all filtering steps are added to the "csv" taxon summary file. A "fasta" file containing all new filtered and processed sequences resulting from the BLAST search is generated for the user, and is used as an input for alignment.

1.3 New DNA sequence alignment

By default, Physcraper uses the software MUSCLE (Edgar 2004) to perform DNA sequence alignments. Instructions on how to install all software dependencies used by Physcraper are provided in the documentation. The process to align new sequences consists of two steps. First, all new sequences are aligned using the default MUSCLE options.

Second, a MUSCLE profile alignment is performed, in which the original alignment is used as a template to align the new sequences. This ensures that the final alignment follows the homology criteria established by the original alignment. The final alignment is not further processed by Physcraper. It is recommended that the alignment is checked by the user, by eye followed by manual refinement, or using a tool for automatic alignment processing (e.g., GBlocks; Castresana 2000, 2002). While curating the alignment is a critical step, it is not a reproducible one. The main reason for its lack of reproducibility might be that it is hard to track changes made on the alignment. A form of version control, to register the differences between the alignment that was produced by the software and the manually curated alignment would ideal. Users may also use Physcraper to only gather new GenBank sequences, to then apply their own preferred alignment and

phylogenetic inference methods.

1.4 Tree reconstruction and comparison

A Maximum Likelihood (ML) gene tree is reconstructed for each alignment provided, using the software RAxML (Stamatakis 2014) with default settings, such as a GTRCAT model of molecular evolution and 100 bootstrap replicates with the default algorithm. Currently only the number of bootsrap replicates can be specified by the user. By default, the original tree is used as a starting tree for the ML searches. Alternatively, users can set the original tree as a full topological constraint, or ignore it completely for the searches. Bootstrap results are summarized with the SumTrees module of DendroPy (current version 4.4.0; Sukumaran & Holder 2010).

Physcraper's final result is an updated phylogenetic hypothesis for the locus provided in the input alignment. Tips on all trees generated by Physcraper are defined by a taxon "name space". The taxon metadata captures the NCBI accession information, as well as the taxon identifiers, allowing the user to perform comparisons and conflict analyses. Two ways to compare the updated tree with the original tree are implemented in Physcraper. First, Robinson Foulds weighted and unweighted metrics are estimated using Dendropy functions (Sukumaran & Holder 2010). Second, a conflict analysis is performed. This is a node by node comparison between the the synthetic OpenTree and the original and updated tree individually. This is performed with OpenTree's conflict Application Programming Interface (Redelings & Holder 2017). For the conflict analysis to be meaningful, the root of the tree needs to be accurately defined. A suggested default rooting based on OpenTree's taxonomy is implemented for now. This approach uses the taxon labels for all the tips in the updated tree, pulls an inferred subtree from OpenTree's taxonomy and then applies the same rooting to the inferred updated tree. However, if the updated tree changes expectations from taxonomy, the root may no longer be appropriate. Automatic identification of a phylogenetic tree root is indeed a difficult problem that has not been solved yet. The best way right now is for users to define outgroup directly on the updated tree, so trees are accurately rooted.

References

Altschul, S.F., Gish, W., Miller, W., Myers, E.W. & Lipman, D.J. (1990). Basic local alignment search tool. *Journal of molecular biology*, **215**, 403–410.

Altschul, S.F., Madden, T.L., Schäffer, A.A., Zhang, J., Zhang, Z., Miller, W. & Lipman, D.J. (1997). Gapped blast and psi-blast: A new generation of protein database search programs. *Nucleic acids research*, **25**, 3389–3402.

Camacho, C., George, C., Vahram, A., Ning, M., Jason, P., Kevin, B. & Thomas, L. (2009). BLAST+: Architecture and applications. *BMC bioinformatics*, **10**, 421.

Castresana, J. (2002). GBLOCKS: Selection of conserved blocks from multiple alignments for their use in phylogenetic analysis. *Version 0.91 b. Copyrighted by J. Castresana, EMBL.*

Castresana, J. (2000). Selection of conserved blocks from multiple alignments for their use in phylogenetic analysis. *Molecular biology and evolution*, **17**, 540–552.

Cock, P.J., Antao, T., Chang, J.T., Chapman, B.A., Cox, C.J., Dalke, A., Friedberg, I., Hamelryck, T., Kauff, F., Wilczynski, B. & others. (2009). Biopython: Freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*, **25**, 1422–1423.

Edgar, R.C. (2004). MUSCLE: Multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research*, **32**, 1792–1797.

McTavish, E.J., Hinchliff, C.E., Allman, J.F., Brown, J.W., Cranston, K.A., Holder, M.T., Rees, J.A. & Smith, S.A. (2015). Phylesystem: A git-based data store for community-curated phylogenetic estimates. *Bioinformatics*, **31**, 2794–2800.

Piel, W., Chan, L., Dominus, M., Ruan, J., Vos, R. & Tannen, V. (2009). Treebase v. 2: A database of phylogenetic knowledge. E-biosphere.

Redelings, B.D. & Holder, M.T. (2017). A supertree pipeline for summarizing phylogenetic and taxonomic information for millions of species. *PeerJ*, **5**, e3058.

Rees, J.A. & Cranston, K. (2017). Automated assembly of a reference taxonomy for phylogenetic data synthesis. *Biodiversity Data Journal*.

Stamatakis, A. (2014). RAxML version 8: A tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, **30**, 1312–1313.

Sukumaran, J. & Holder, M.T. (2010). DendroPy: A python library for phylogenetic computing. *Bioinformatics*, **26**, 1569–1571.

Vos, R.A., Balhoff, J.P., Caravas, J.A., Holder, M.T., Lapp, H., Maddison, W.P., Midford, P.E., Priyam, A., Sukumaran, J., Xia, X. & others. (2012). NeXML: Rich, extensible, and verifiable representation of comparative data and metadata. *Systematic biology*, **61**, 675–689.