Received: 10 May 2018 | Accepted: 7 September 2018

DOI: 10.1111/2041-210X.13096

APPLICATION

Methods in Ecology and Evolution BRITISH ECOLOGICAL SOCIETY

# PyPHLAWD: A python tool for phylogenetic dataset construction

**Stephen A. Smith** | **Joseph F. Walker**

Department of Ecology and Evolutionary Biology, University of Michigan, Ann Arbor, Michigan

**Correspondence**
Stephen A. Smith
Email: eebsmith@umich.edu

**Funding information**
S.A.S. was supported by NSF 1458466 and 1207915. J.F.W. was supported by NSF 1354048.

Handling Editor: Natalie Cooper

## Abstract

1. Comprehensive phylogenetic trees are essential for many ecological and evolutionary studies. Researchers may use existing trees or construct their own. In order to infer new trees, researchers often rely on programmes that construct datasets from publicly available molecular data.
2. Here, we present PyPHLAWD, a phylogenetically guided tool written in PYTHON that creates molecular datasets for building trees. PyPHLAWD constructs clusters (putative orthologs) that may be used for downstream analyses and provides users with a set of easy to interpret results. PyPHLAWD can conduct both baited (analyses that require the identification of gene regions *a priori*) and clustering analyses (analyses that do not require *a priori* identification of gene regions).
3. PyPHLAWD is extensible, flexible, open source, and available at https://github.com/FePhyFoFum/PyPHLAWD, with a detailed website outlining instructions and functionality at https://fephyfofum.github.io/PyPHLAWD/.
4. The utility of PyPHLAWD is highlighted here with an example workflow for the plant clade Dipsacales and may be applied to any clade with publicly available data on GenBank.

**KEYWORDS**
GenBank, phylogenetics, Python, sequences, tree of life

## 1 | INTRODUCTION

Large phylogenies have become important for addressing outstanding questions in comparative ecological and evolutionary studies. While resources are available that offer broadly sampled taxonomies and phylogenies (e.g., Open Tree of Life), for many reasons, researchers may want to process the available molecular data for the construction of targeted phylogenies.

Several programmes have been developed to process publicly available GenBank data in order to create phylogenetic datasets including SUPERSMART (Antonelli et al., 2017), PHLAWD (Smith, Beaulieu, & Donoghue, 2009), Phylota (Sanderson, Boss, Chen, Cranston, & Wehe, 2008), Phylogenerator (Pearse & Purvis, 2013), among others. These programmes can serve as complements/verification of each other, along with providing utility that offers greater flexibility in analyses. While automation can simplify many steps, recently, it has been shown that human intervention improves the quality of the resulting phylogeny (Beaulieu & O'Meara, 2018; Smith & Brown, 2018). Here, we describe PyPHLAWD, a PYTHON package providing a diverse set of tools for constructing datasets using GenBank in a semi-automated fashion.

PyPHLAWD offers users flexibility throughout an analysis allowing for researchers to intervene at any step. The programme implements both baited (Smith et al., 2009) and a tip-to-root clustering method (Smith & Brown, 2018; Walker et al., 2018) for thorough and rapid orthology identification. PyPHLAWD is an extensible and flexible tool written in PYTHON that can be easily incorporated into different workflows.


© 2018 The Authors. Methods in Ecology and Evolution © 2018 British Ecological Society

## 2 | PYPHLAWD WORKFLOW

PyPHLAWD is an open source, freely available, set of scripts written in PYTHON. The source code and instructions are available from https://github.com/FePhyFoFum/PyPHLAWD. PyPHLAWD requires a few, easy to install, dependencies including several programmes from the PHYX (Brown, Walker, & Smith, 2017) package, NCBI-BLAST+ (Altschul et al., 1997), `mcl` (Van Dongen, 2000), `mafft` (Katoh & Standley, 2013) and several PYTHON packages (SQLITE, NETWORKX, CLINT). For increased speed, PyPHLAWD can optionally be compiled with cython. Other programmes, if installed, may be used such as `FastTree` (Price, Dehal, & Arkin, 2010). All of these are freely available, and many through software repositories. Installation and examples are available from the website https://fephyfofum.github.io/PyPHLAWD/.

PyPHLAWD can be used unsupervised or with human intervention. It relies on a preassembled database of sequences that may be created using `phlawd_db_maker` (https://github.com/blackrim/phlawd_db_maker) or downloaded from resources provided on the programme website. PyPHLAWD allows a user to conduct both clustering analyses, without specifying the genes of interest, as well as a baited analysis.

A baited analysis in PyPHLAWD is done by running `setup_clade_bait.py` and providing a directory of *bait* sequence files that can be used to filter sequences from the database. A bait sequence file for a single gene should contain full length sequence for several species (e.g., 10–20) across the clade of interest. This was the standard analysis mode for PHLAWD (Smith et al., 2009) and can be useful when there are a few well-known genes sampled for a particular clade. Unlike PHLAWD, however, PyPHLAWD will conduct an analysis on each file (i.e., gene) in the bait directory instead of the user having to conduct an analysis on each gene separately.

In contrast to a baited analysis, a clustering analysis does not require a set of identified bait sequences. Instead, all genes available for the user specified taxonomic group will be identified and summarized as a user-friendly html file. During the process all-by-all BLAST analyses along with markov clustering analyses, MCL, are conducted to identify clusters. In PyPHLAWD, a clustering analysis is conducted by running `setup_clade.py`.

Whether baited or clustered, once gene regions are identified, the remaining analyses are the same (Figure 1). A set of directories representing the taxonomic hierarchy of the taxon of interest are created. Within each of these directories, the sequences representing the taxon of the directory are placed in a file. By default, whole genomes are excluded from analyses but are placed in a file in the relevant directory. Sequence similarity analyses are conducted from the tip to the root of the folder hierarchy. Because the sequence similarity analyses themselves do not impose a taxonomic constraint, only the order of analyses, it is unlikely that bias will be introduced by a potentially incorrect taxonomy. However, if there are egregious errors in the taxonomy, they may be edited within the database.

There are several parameters that can be edited to customize analyses. For example, sequences associated with misidentified taxa

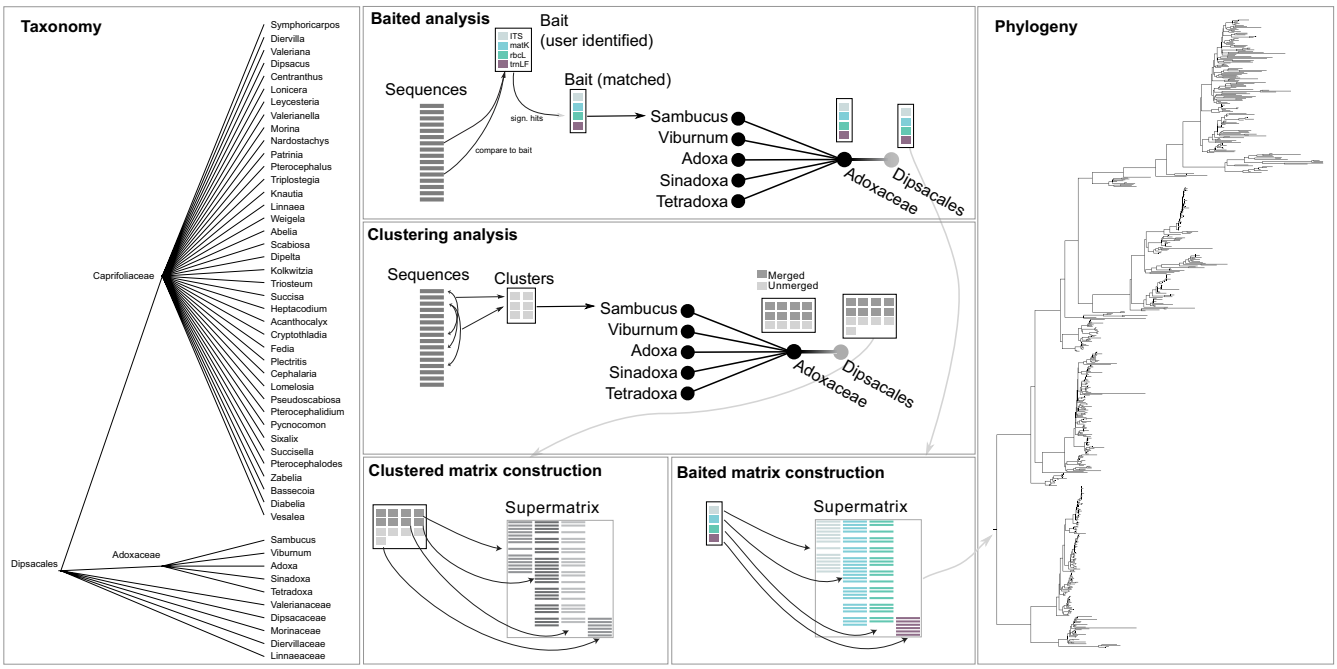


**FIGURE 1** PyPHLAWD procedure with an example from the plant clade Dipsacales. The taxonomy on the left illustrates the clustering order and the folder structure generated from PyPHLAWD. The baited analysis panel demonstrates that sequences will be compared to bait sequences. Clustering is conducted from tip-to-root as shown in Adoxaceae. Supermatrices may be constructed (though not required) from the resulting clusters or bait sequences. A phylogeny may be built from these supermatrices (right panel) or from individual gene regions

can be excluded by adding their taxonomic or sequence id to the files `bad_seqs.py` or `bad_taxa.py`. If the user wants to exclude sequences based on patterns in the sequence description (e.g., particular collections of sequences), these patterns can be added to the `exclude_patterns.py` or the `exclude_desc_patterns.py` files.

PyPHLAWD has a general configuration file, `conf.py`, that defines several parameters including user options like `smallest_size` meant to define the smallest sequences to exclude as well as several parameters related to the BLAST analyses (e.g., `length_limit`, `evalue_limit`, `evalue_limit_lc`, `perc_identity`). PyPHLAWD functions as a set of scripts instead of a PYTHON library installed at the system level to encourage users that may be less familiar with library development to add functionality or edit scripts. Therefore, the `conf.py` file also requires that the installation directory of PyPHLAWD be specified (as this allows PyPHLAWD to locate the other scripts in the package).

## 3 | DEMONSTRATION

We performed two demonstration analyses, a baited and a clustering analysis. Both analyses were performed on the plant clade Dipsacales. These runs were conducted on a 3.9Ghz quad-core laptop with 16 GB RAM and running Xubuntu Linux (kernel 4.16). For both analyses, we set `smallest_size = 400`, `length_limit = 0.55`, and `perc_identity = 20`. We used a copy of GenBank for the `pln` division built with `phlawd_db_maker` constructed on 04/11/18. For baited analyses, we used rbcL, trnL-trnF, ITS, and matK as bait (files available as part of the examples in the distributed source code). For clustering analyses, we conducted analyses with the above parameters.

## 4 | RUNS FOR SIMILAR METHODS

In order to compare the performance of PyPHLAWD, we also conducted analyses using PHLAWD, Phylota, SUPERSMART, and Phylogenerator. For PHLAWD, we used the parameters `mad = 0.05`, `coverage = 0.55`, and `identity = 0.2` using the same bait as the PyPHLAWD baited runs. We examined Phylota results as available on 05/01/18 using GenBank release 194. For SUPERSMART, we conducted runs, without divergence time estimation, on the plant clade Adoxaceae, within the Dipsacales, as runs conducted on the entire Dipsacales failed to successfully construct some elements of the tree. We, therefore, compared the Adoxaceae results from PyPHLAWD to those of SUPERSMART. Phylogenerator was tested using Dipsacales and searching for the genes "rbcL", "matK", "ITS" (alias: "internal transcribed spacer 1", fussy: false), and trnLF (alias: "trnL-trnF intergenic spacer", fussy: false). The Hawkeye sequence adjustment method was turned off and the minimum reference length was set to 400 bp.

## 5 | RESULTS OF VERIFICATION

For the Dipsacales, there are 33,178 sequences available representing 864 taxa (as of 04/11/18).

### 5.1 | Baited analysis

We constructed the bait sequences by manually downloading 20 representative sequences for each of the target genes on GenBank. The baited analysis took 52 s and PyPHLAWD found 299 taxa with rbcL, 333 with trnL-trnF, 553 with ITS, and 322 with matK. We constructed a phylogenetic dataset with these resulting gene regions that resulted in 641 taxa (173 taxa in Adoxaceae). We constructed a phylogeny using RAxML v. 8.2.11 (Stamatakis, 2014) with the `-f a` option generating 100 rapid bootstraps with GTR+G model of evolution and partitioned by gene region.

### 5.2 | Clustered analysis

The clustered analysis took 5 min and 16 s and resulted in 684 taxa represented in 617 clusters. The largest clusters included ITS (559 taxa), trnLF (334 taxa), matK (333 taxa), and rbcL (299 taxa) with others containing between 268 and fewer taxa (Figure 1). We constructed a phylogenetic dataset with the `find_good_clusters_for_concat.py` (with at least 20% taxon sampling and at least 20 taxa) script that identified 16 clusters representing 672 taxa (173 taxa in Adoxaceae). We constructed a phylogeny using RAxML v. 8.2.11 with the `-f a` option generating 100 rapid bootstraps with GTR+G model of evolution and partitioned by gene region. The `find_good_clusters_for_concat.py` script can construct a constraint tree. However, this was not used in the construction of the phylogeny presented here.

### 5.3 | Results from other methods

Phylota recovered 536 ITS, 330 matK, 354 trnLF, and 242 rbcL taxa for Dipsacales with a concatenated dataset of 581 taxa (155 in Adoxaceae). PHLAWD baited analyses recovered 659 ITS, 329 matK, 349 trnLF, and 295 rbcL sequences for Dipsacales with a concatenated dataset of 731 taxa (201 in Adoxaceae). Run-times were between 16 and 40 seconds. SUPERSMART recovered 147 taxa for Adoxaceae with a run-time of c.4 min to perform the taxize, align, orthologize and bbmerge steps. The total run time for Phylogenerator was between 32 min and 1.3 hr depending on whether alignment and reconstruction was done and resulted in a phylogeny consisting of 627 tips and 347 matK, 298 rbcL, 325 trnLF, and 566 ITS sequences found.

## 6 | DISCUSSION

PyPHLAWD is a package that can be used for constructing phylogenetic datasets with molecular data. In addition to conducting baited

analyses, as in PHLAWD, it also allows for clustering sequences without bait. PyPHLAWD constructs clusters rapidly by only conducting all-by-all comparisons for the most nested taxonomic units (e.g., genera) and then merging these clusters using BLAST. Additionally, PyPHLAWD is written as a set of scripts to allow for more flexibility for the user.

### 6.1 | Comparison to other methods

We compared the results of baited and clustering analyses to those from PHLAWD, Phylota, SUPERSMART, and Phylogenerator. PyPHLAWD baited analyses are different than PHLAWD analyses in that PyPHLAWD uses the BLAST algorithm instead of the Smith–Waterman algorithm for sequence homology analyses. As a result, the specific parameters used in each analysis are not comparable. Nonetheless, given that these parameter values are typical for many analyses, the *results* are comparable. PyPHLAWD and PHLAWD baited results were similar with differences reflecting the differences between BLAST and Smith–Waterman and the specific cutoffs used (e.g., sequence length, overlap). PHLAWD recovered slightly fewer sequences of rbcL and matK and more sequences of trnLF and ITS. In both cases, where more sequences were recovered, it was the result of PHLAWD allowing the retrieval of shorter sequences. Including short sequences in the analysis can increase error in alignment and so are excluded from PyPHLAWD by default. The phylogenies produced from the PyPHLAWD datasets had fewer long branches (i.e., from alignment error) than those from PHLAWD.

Instead of comparing all of the clusters found by Phylota, we compared the results for ITS, matK, rbcL, and trnLF from the PyPHLAWD clustering analyses to the equivalent clusters available through Phylota. There were several differences that were due to the differences in the GenBank releases used as Phylota covers release 194 while PyPHLAWD used release 224. PyPHLAWD also uses a more stringent sequence length cutoff. For example, PyPHLAWD recovered 559 taxa with ITS and Phylota, which includes much smaller sequences (e.g., 154 bp the case of ITS), recovered 536 taxa. Similar differences were found for other gene regions.

Phylogenerator and PyPHLAWD baited analyses differ in the method of orthologous sequence detection. PyPHLAWD recognizes sequences based on sequence similarity, whereas Phylogenerator uses sequence label matching. Phylogenerator requires the user to specify the alias of a sequence and relies on proper curation of deposited samples. It, however, does not require a local database and therefore downloads sequences at the time the user runs the programme. This difference in the time required for sequence retrieval helps explain the overall differences in runtime. Otherwise, the results were comparable with fewer taxa per gene in certain cases and more in others.

SUPERSMART provides a comprehensive package for the construction of trees from GenBank data. Despite some similarities, there are several differences between SUPERSMART and PyPHLAWD. First, SUPERSMART is run as a virtual machine, allowing for fewer steps in installing dependencies. However, this comes at the cost of flexibility. There are also methodological differences. For example, SUPERSMART relies on initial clusters constructed from Phylota that it then verifies with BLAST. PyPHLAWD uses a database created by the user that may be much more up to date. SUPERSMART conducts all analyses required for dataset construction, phylogenetic reconstruction, and divergence time estimation, presenting the user with the final results. Intermediate outputs are obfuscated to the user. PyPHLAWD expects human intervention at any stage. Each output can be used for other analyses. For example, PyPHLAWD provides the clustered genes, aligned and unaligned, in folders based on taxonomy. The detailed summary of the output given by PyPHLAWD gives users greater ease to conduct an array of downstream analyses on the clades.

We do not aim to criticize other programmes and procedures that may be useful and available to the community. Instead, we feel that these are complimentary and may serve different purposes for diverse goals and analyses.

### 6.2 | On baited vs. clustering analyses

As shown here, it is possible for baited and clustering analyses to produce different results with PyPHLAWD. There are several reasons for this. For example, the behaviour of the baited analysis may change depending on the quality, size, and taxonomic coverage of the user identified bait files. Alternatively, clustering runs will only be dependent on the available sequences in the source database. A user should choose the analysis that is the most reasonable given their goals. When common gene regions for a clade are unknown, clustering analyses are preferable.

#### ACKNOWLEDGMENTS

We are grateful to early adopters, testers, 2 anonymous reviewers, Javier Igea, and Nathanael Walker-Hale.

#### AUTHORS' CONTRIBUTIONS

J.F.W. and S.A.S. developed the procedure. S.A.S. wrote the code. J.F.W. and S.A.S. conducted the analyses and wrote the manuscript.

#### DATA ACCESSIBILITY

All code is open source and available from at Smith and Walker (2018), https://doi.org/10.5281/zenodo.1400789, and https://github.com/FePhyFoFum/PyPHLAWD. A web manual is also available at https://fephyfofum.github.io/PyPHLAWD/.

#### ORCID


*Stephen A. Smith* http://orcid.org/0000-0003-2035-9531

#### REFERENCES

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. https://doi.org/10.1093/nar/25.17.3389

Antonelli, A., Hettling, H., Condamine, F. L., Vos, K., Nilsson, R. H., Sanderson, M. J., … Bacon, C. D. (2017). Toward a self-updating platform for estimating rates of speciation and migration, ages, and relationships of taxa. *Systematic Biology*, *66*, 152–166.

Beaulieu, J. M., & O'Meara, B. C. (2018). Can we build it? Yes we can, but should we use it? Assessing the quality and value of a very large phylogeny of Campanulid angiosperms. *American Journal of Botany*, *105*, 417–432. https://doi.org/10.1002/ajb2.1020

Brown, J. W., Walker, J. F., & Smith, S. A. (2017). Phyx: Phylogenetic tools for unix. *Bioinformatics*, *33*, 1886–1888. https://doi.org/10.1093/bioinformatics/btx063

Katoh, K., & Standley, D. M. (2013). MAFFT multiple sequence alignment software version 7: Improvements in performance and usability. *Molecular Biology and Evolution*, *30*, 772–780. https://doi.org/10.1093/molbev/mst010

Pearse, W. D., & Purvis, A. (2013). PhyloGenerator: An automated phylogeny generation tool for ecologists. *Methods in Ecology and Evolution*, *4*, 692–698. https://doi.org/10.1111/2041-210X.12055

Price, M. N., Dehal, P. S., & Arkin, A. P. (2010). FastTree 2–approximately maximum-likelihood trees for large alignments. *PLoS ONE*, *5*, e9490. https://doi.org/10.1371/journal.pone.0009490

Sanderson, M. J., Boss, D., Chen, D., Cranston, K. A., & Wehe, A. (2008). The phylota browser: Processing genbank for molecular phylogenetics research. *Systematic Biology*, *57*, 335–346. https://doi.org/10.1080/10635150802158688

Smith, S. A., Beaulieu, J. M., & Donoghue, M. J. (2009). Mega-phylogeny approach for comparative biology: An alternative to supertree and supermatrix approaches. *BMC Evolutionary Biology*, *9*, 37. https://doi.org/10.1186/1471-2148-9-37

Smith, S. A., & Brown, J. W. (2018). Constructing a broadly inclusive seed plant phylogeny. *American Journal of Botany*, *105*, 302–314. https://doi.org/10.1002/ajb2.1019

Smith, S. A., & Walker, J. F. (2018). PyPHLAWD: Release V.1.0. https://doi.org/10.5281/zenodo.1400789

Stamatakis, A. (2014). RAxML version 8: A tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics*, *30*, 1312–1313. https://doi.org/10.1093/bioinformatics/btu033

Van Dongen, S. M. (2000). *Graph clustering by flow simulation*. PhD thesis.

Walker, J. F., Yang, Y., Feng, T., Timoneda, A., Mikenas, J., Hutchison, V., … Walker-Hale, N. (2018). From cacti to carnivores: Improved phylotranscriptomic sampling and hierarchical homology inference provide further insight into the evolution of caryophyllales. *American Journal of Botany*, *105*, 446–462. https://doi.org/10.1002/ajb2.1069

**How to cite this article:** Smith SA, Walker JF. PyPHLAWD: A python tool for phylogenetic dataset construction. *Methods Ecol Evol*. 2019;10:104–108. https://doi.org/10.1111/2041-210X.13096