Article type: software

PhyloPipe: a graphical pipelining software for phylogenomic analysis

Yixuan Guo[1], Haoyang Wu[2], Yang Liu[2], Rebecca J. Stones[1], Gang Wang[1,*], Xiaoguang Liu[3,*], Qiang Xie[2,*] and Mingming Ren[3].

[1]College of Computer and Control Engineering, Nankai University, Tianjin, 300071, China.

[2]College of Life Sciences, Nankai University, Tianjin, 300071, China.

[3]College of Software, Nankai University, Tianjin, 300071, China.

*corresponding authors E-mail address: wgzwp@nbjl.nankai.edu.cn (W. Gang), liuxg@nbjl.nankai.edu.cn (X. Liu), qiangxie@nankai.edu.cn (Q. Xie).

Keywords: Phylogenomics, graphical user interface, software.

Abstract

Background: Due to the rapid spread of high-throughput sequencing technologies, molecular sequence data has increased dramatically, raising a challenge for current phyologenetics programs in reconstructing evolutionary histories. Currently, a widely adopted protocol for sequence data analysis requires user intervention at multiple stages. While there is existing software for each step, the process usually requires command line scripting for manipulating the intermediate data between steps.

Results: We present PhyloPipe, a user-friendly graphical front-end software for phylogenomic analysis. PhyloPipe combines a collection of widely used software in phylogenomic research, HaMStR, MAFFT, MUSCLE, Aliscore, and RaxmlGUI, together with intermediate procedures into an integrated pipeline. It provides a graphical user interface for manipulating the complied input files of annotated transcriptomic data into a RAxML-compatible format, which can then be used to generate a phylogenetic tree.

Conclusions: PhyloPipe can guide researchers through the various stages of processing molecular data in phylogenetic analysis, freeing them from running multiple programs from the command line and writing scripts to manipulate intermediate data.

Background

During the past thirty years or so, after the foundation of international databases of molecular sequences, a large amount of data has been contributed from phylogenetic studies. With the rapid spread of high-throughput sequencing technologies, the exponential increase of sequence data has raised a challenge for current bioinformatics programs in reconstructing evolutionary histories. Recently, a largely universal protocol, which incorporates advanced programs that can process genome-scale sequence data, has been widely used in various phylogenomic studies about birds and insects, which were based on transcriptomic or genomic data [1][2][3]. A series of programs, such as HaMStR [5], MAFFT [5], MUSCLE [6], Aliscore [7], ALICUT [9], RAxML [7], etc., have been regularly included in this process. For example, Dell'ampio et al. [3] studied the phylogenetic relationships of primarily wingless insects, adopting HaMStR for the orthology prediction, MAFFT for the alignment, Aliscore for identification of the randomly similarly aligned sections, ALICUT v.2.0 for masking according to the Aliscore results, and FASconCAT v.1.0 [10] for concatenation of masked alignments. The final step, maximum likelihood tree reconstruction, mainly uses RaxML. Additionally, extra tree searches and bootstrap analyses were performed. Most of these programs have command-line interfaces, and thus make a series of tedious obstacles for researchers. This motivates us to develop a graphical pipeline software, which we call PhyloPipe, which offers the users an interface to perform these steps without the need for writing multiple in-between scripts.

PhyloPipe is a graphical user interface (GUI) which helps the users in managing the multiple functions required for phylogemonic analyses, including ortholog prediction, data filtering, sequence alignment, matrix masking, concatenation, and phylogenetic reconstruction. Further, PhyloPipe contains additional procedures for the management of data between stages, which is innovatively written by us, and thus provides a seamless pipeline which performs both correctly and efficiently.

PhyloPipe was written in C++ and developed using the QT Creator platform; it runs on Unix/Linux systems.

Implementation

PhyloPipe is easy to operate with a friendly user interface. It pipelines six consecutive stages of phylogenetic analysis: (a) orthology prediction [HaMStR], (b) data filtering, (c) alignment [MAFFT, MUSCLE], (d) matrix masking [Aliscore], (e) concatenation, and (f) phylogenetic reconstruction [raxmlGUI [11] (Silvestro and Michalak, 2012)], each of the stages utilize either state-of-the-art, well established external programs or procedures developed for PhyloPipe. A flow chart of PhyloPipe is given in Fig.1.

Fig.1 Flowchart of PhyloPipe. (A) Outline of PhyloPipe's analysis pipeline. (B) Methods used for each stage of PhyloPipe. Stages which utilize available software are highlighted in green; other components are newly developed and are highlighted in red.

The installation details of PhyloPipe are given in the user manual available as supplementary material. In PhyloPipe, each stage is controlled from an individual dialog box, where parameter settings may be changed. If desired, users can pick and choose which stages to use.

To maneuver through the pipeline, the user selects each stage and runs the respective function. The output from one step is used as the input for the next step, which allows users to intervene between any of the stages.

In the following sections, we describe the individual stages in PhyloPipe.

Orthology Prediction

Expressed sequence tags (ESTs) are widely used in evolutionary studies and particularly in molecular systematics studies; they are often the only source for biological sequence data from taxa outside mainstream interest.

We incorporated HaMStR, a wildly adopted efficient profile hidden Markov model (HMM) based search for mining EST data for the presence of orthologs and giving a curated set of genes. Usually, HaMStR would translate the input nucleotide sequences into the corresponding amino acid sequences. In PhyloPipe, we implement a graphical interface for the original command-line based HaMStR program. Flexible settings are available in the HaMStR dialog, including refspec, the reference species stored in the HaMStR program directory, hmmset, the available core ortholog sets in the chosen reference species library, and hmm, where only a single HMM is used for ortholog prediction. Specifically, PhyloPipe would search for the hmmset once the refspec is chosen by user, and provide check boxes for the hmmset core orthologs, whereby users can choose their desired orthologs.

Data Filtering

Data Filtering is applied to combine each HaMStR output file (the homologous matrix data of the taxa) we obtain from the previous stage. The mechanism of Data Filtering is illustrated in Fig. 2 (A).

Data Filtering serves as a bridge between HaMStR and sequence alignment. For each taxon, HaMStR generates a group of amino acid sequences; we retain a selection of those sequences. Specifically, we have two options: The “combine factor” option uses a user-determined threshold to decide whether or not ortholog data should be retained (if the data occurs for a number of taxa exceeding the threshold, it is retained). The “original data selection” option instead allows the user to manually select which core orthologs they would like to retain from the list provided by HaMStR’s orthologs library.

Fig. 2 Illustrating the Data Filtering and Random Similarity Exclusion components. (A) Two directories of input files are merged into a single directory through Data Filtering. The input directories are the output data from HaMStR program; they contain the amino acid sequences files in FASTA format, and are named after the corresponding amino acid sequences in the chosen ortholog library while running HaMStR. In this example, both files contain the amino acid sequences, orig_orth1, orig_orth2, and orig_orth3 from the ortholog library generated by HaMStR from taxa input data: tax1 or tax2. The Data Filtering algorithm retains the amino acid sequence files that exist in 100% of the input directories, orig_orth1 and orig_orth2, in this case. (B) The input directory consists of a group of amino acid sequence files and their Aliscore-based random similarity results which contains the loci whose scores are below zero. For each amino acid sequence file, those loci are deleted from the sequences.

Alignment

This step aligns orthologous sequences to give the homologous matrix files needed for the Data Filtering step. It involves two available programs, both of which serve the function of multiple sequence alignment. (a) MAFFT, a method for rapid multiple sequence alignment based on fast Fourier transforms (FFT). MAFFT is known for drastically reducing the CPU time while increasing the accuracy of alignments owing to the FFT method and the simplified scoring system presented in it. We include this program because of its high efficiency. MAFFT also adjusts the alignment option to best suit the size of input sequence file. In addition, MAFFT provides a flexible choice between high speed and high accuracy while running the sequence alignment. (b) MUSCLE, a multiple sequence alignment with highest accuracy and high throughput compared with other current alignment programs. We include this program in PhyloPipe as a recommended refinement of the alignment results of MAFFT. In this way, we can obtain a multiple sequence alignment with high accuracy in a relatively short period of time.

Ordinarily, both MAFFT and MUSCLE can process one file at a time. In order to process a batch of files, we implement SysExec, a C++ program which serves as a loop to successively process the files in input directory. With SysExec, users can call MAFFT and MUSCLE to process all the files in the input directory, instead of processing files one by one.

Matrix Masking

Random similarity of sequences or sequence sections can not only impede phylogenetic analyses, but also negatively interfere with the estimation of substitution model parameters. Therefore, we used an Aliscore-based exclusion method to analyze and mask the ambiguously aligned or highly diverged regions of the alignment results from the previous step. This step contains two successive sub-steps, detailed below, operated via a dialog box.

Aliscore

We applied Aliscore, a program based on a Monte Carlo resampling with a sliding window method which identifies random similarity in multiple sequence alignments. Similar to MAFFT and MUSCLE, Aliscore uses the SysExec script to process a batch of files at one time.

Random Similarity Exclusion

We implement a random similarity exclusion method which utilizes the Aliscore results; it is motivated by ALICUT and both delete the loci with a negative randomness score. ALICUT is a relatively widely used program which provides the function of random similarity exclusion, but it mixes the output file with the input file and does nothing to the files which have no randomly similar loci. PhyloPipe's Random Similarity Exclusion function generates separate output files, and includes those without randomly similar loci. The process of Random Similarity Exclusion is illustrated in Fig. 2 (B).

Concatenation

After the Matrix Masking step, we performed concatenation on the sub-datasets to form a supermatrix for subsequent phylogenetic analysis, together with a file containing the partition information of the supermatrix. In this stage, we adapt the Perl script named concatenator in the bioinformatics pipeline provided by Peters et al. [12], which can be used to concatenate different kinds of sequence data into one supermatrix file. We implement a GUI for the concatenator script. The GUI provides two parameters: matrix name and charset list.

Phylogenetic Reconstruction

This function aims at building a phylogenetic tree from the concatenated supermatrix. PhyloPipe uses raxmlGUI for phylogenetic reconstruction, which is a graphical user interface of RAxML. RAxML uses a maximum likelihood method for phylogenetics, and is widely used due to its high computational performance and accuracy. Users can interact with raxmlGUI via PhyloPipe's phylogenetic reconstruction menu.

Discussion

We have performed software tests on PhyloPipe described in the supplementary material available online. The intermediate results of each stage have been carefully checked and show correctness.

There are three major benefits of PhyloPipe:

Complete solution with ease of use. The phylogenomic analysis pipeline offered by PhyloPipe is seamless and requires no additional data processing. The six stages and intermediate data adjustment are incorporated into this single pipeline. It utilizes a protocol that has been widely adopted, but ordinarily researchers will need to use separate programs and write independent scripts. Without PhyloPipe, as many as eight individual scripts or programs could be needed to process the pipeline from compiled transcriptomic data to a final phylogenetic tree, which is a burden for researchers. With the assistance of PhyloPipe, a researcher is able to operate the pipeline without the need for scripting.

High performance and flexibility. High performance in PhyloPipe is inherited from its subprograms, which are all state-of-the-art. We provide appropriate default parameter settings for users, although these can be modified as required. Users can also choose to use the whole pipeline or just some specific stages within it.

Free and open source. PhyloPipe and its source code are freely available for download and modification. Detailed information and instructions are provided in the manual for PhyloPipe.v1.1, available at http://sf.net/projects/phylopipe/.

Conclusion

PhyloPipe is a freely available software which facilitates the use of widely used programs in phylogenomic analysis. It gives a pipeline with a user-friendly graphical interface for bioinformatics for conducting phylogenomic analyses. This paper describes the six consecutive stages in PhyloPipe, illustrating how PhyloPipe can benefit bioinformaticians in their phylogenomic analyses.

Availability and requirements

- **Project name:** PhyloPipe
- **Project home page:** documentation, installation instructions, source code: http://sf.net/projects/phylopipe/
- **Operating system(s):** Linux
- **Programming language:** C++
- **Other requirements:** None
- **License:** None
- **Any restrictions to use by non-academics:** None

Additional file 1: PhyloPipe software test documentation
Additional file 2: Pseudo Code for data filtering and random similarity exclusion algorithm
Additional file 3: PhyloPipe.v1.1 manual

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

QX conceived the idea of PhyloPipe. YG, YL and HW designed the software and performed testing. YG did the coding for the implementation of PhyloPipe software. YG wrote the manuscript, RJS edited the manuscript. YG, GW, XL, QX, RJS and MR supervised the work and critically revised the manuscript. All authors read and approved the final manuscript.

Acknowledgements

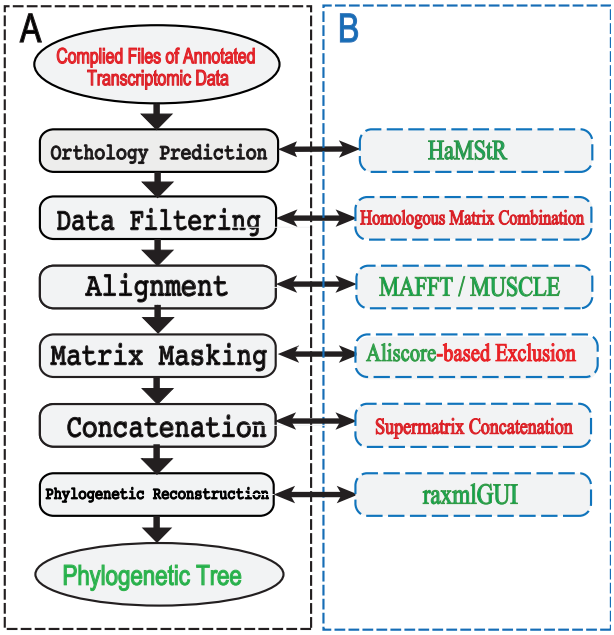
This work was supported by the National Science Foundation of China (Grant numbers: 61373018, 11301288, 31222051, J1210005), Program for New Century Excellent Talents in University (Grant number: NCET130301) and the Fundamental Research Funds for the Central Universities (Grant number: 65141021). Stones was supported by her NSF China Research Fellowship for International Young Scientists (grant number: 11450110409).

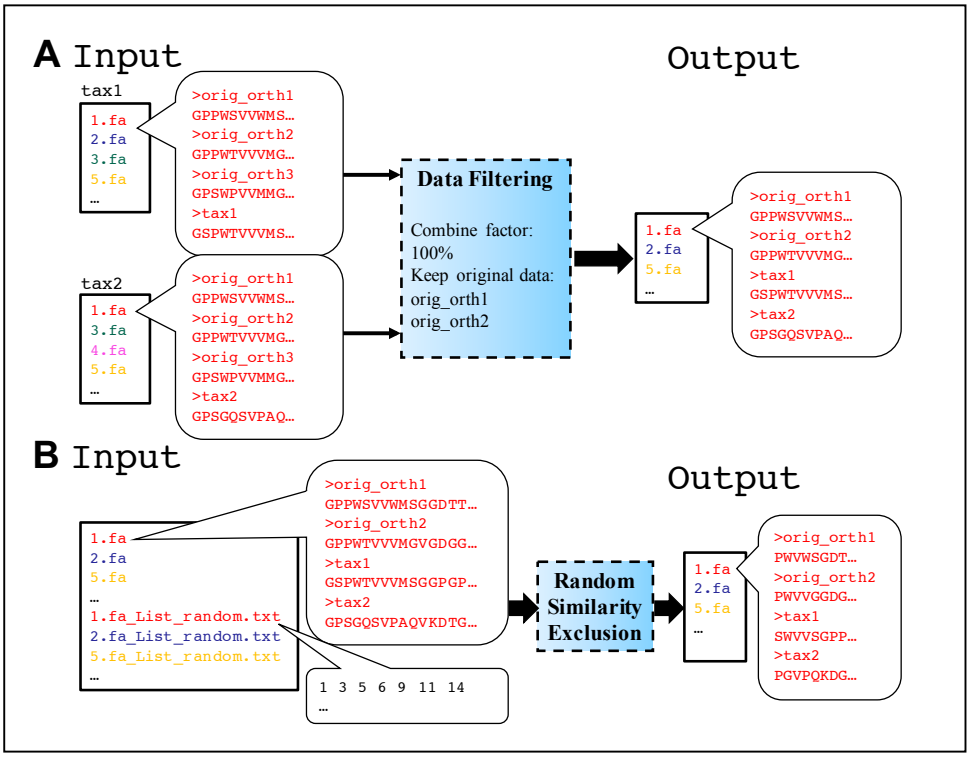
Author Details

[1]College of Computer and Control Engineering, Nankai University, Tianjin, 300071, China.
[2]College of Life Sciences, Nankai University, Tianjin, 300071, China.
[3]College of Software, Nankai University, Tianjin, 300071, China.

References

1. Jarvis ED et al. Whole-genome analyses resolve early branches in the tree of life of modern birds. Science. 2014;346(6215): 1320-1331.
2. Misof B, et al. Phylogenomics resolves the timing and pattern of insect evolution. Science. 2014; 346(6210): 763-767.
3. Dell'ampio E. et al. Decisive data sets in phylogenomics: lessons from studies on the phylogenetic relationships of primarily wingless insects. Mol. Biol. Evol. 2014;31(1): 239-249.
4. Ebersberger I, Strauss S, Haeseler AV. HaMStR: Profile hidden Markov model based search for orthologs in ESTs. BMC Evol. Biol. 2009; 9(157).
5. Katoh K, Misawa K, Kuma K, Miyata T. MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform. Nucleic Acids Res. 2002;30(14): 3059-3066.
6. Edgar RC. MUSCLE: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Res. 2004; 32(5): 1792-1797.
7. Misof B, Misof K. A Monte Carlo approach successfully identifies randomness in multiple sequence alignments: a more objective means of data exclusion. Syst. Biol. 2009; 58(1): 21-34.
8. Kück P. ALICUT: a PerlScript which cuts ALISCORE identified RSS Department of Bioinformatics, Zoologisches Forschungsmuseum A. 2009; Koenig (ZFMK), Bonn, Germany, version 2.0.
9. Stamatakis A. RaxML-VI-HPC: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. Bioinformatics. 2006; 22(21): 2688-2690.
10. Kück P, Meusemann K. FASconCAT: Convenient handling of data matrices. Mol. Phylogenet. Evol. 2010; 56: 1115-1118.
11. Silvestro D, Michalak I. raxmlGUI: a graphical front-end for RAxML. Org. Divers. Evol. 2012; 12(4): 335-337.
12. Peters RS, Meyer B, Krogmann L, Borner J, Meusemann K, Schütte K, Niehuis O, Misof B. The taming of an impossible child: a standardized all-in approach to the phylogeny of Hymenoptera using public database sequences. BMC Biology. 2011; 9:55.


A
B
Complied Files of Annotated Transcriptomic Data
Orthology Prediction
Data Filtering
Alignment
Matrix Masking
Concatenation
Phylogenetic Reconstruction
Phylogenetic Tree
HaMStR
Homologous Matrix Combination
MAFFT / MUSCLE
Aliscore-based Exclusion
Supermatrix Concatenation
raxmlGUI


A Input
Output
tax1
1.fa
2.fa
3.fa
5.fa
…
>orig_orth1
GPPWSVVWMS…
>orig_orth2
GPPWTVVVMG…
>orig_orth3
GPSWPVVMMG…
>tax1
GSPWTVVVMS…
tax2
1.fa
3.fa
4.fa
5.fa
…
>orig_orth1
GPPWSVVWMS…
>orig_orth2
GPPWTVVVMG…
>orig_orth3
GPSWPVVMMG…
>tax2
GPSGQSVPAQ…
Data Filtering
Combine factor: 100%
Keep original data:
orig_orth1
orig_orth2
1.fa
2.fa
5.fa
…
>orig_orth1
GPPWSVVWMS…
>orig_orth2
GPPWTVVVMG…
>tax1
GSPWTVVVMS…
>tax2
GPSGQSVPAQ…
B Input
Output
1.fa
2.fa
5.fa
…
1.fa_List_random.txt
2.fa_List_random.txt
5.fa_List_random.txt
…
>orig_orth1
GPPWSVVWMSGGDTT…
>orig_orth2
GPPWTVVVMGVGDGG…
>tax1
GSPWTVVVMSGGPGP…
>tax2
GPSGQSVPAQVKDTG…
1 3 5 6 9 11 14
…
Random Similarity Exclusion
1.fa
2.fa
5.fa
…
>orig_orth1
PWVWSGDT…
>orig_orth2
PWVVGGDG…
>tax1
SWVVSGPP…
>tax2
PGVPQKDG…