BMC Bioinformatics

SOFTWARE Open Access

Agalma: an automated phylogenomics workflow

Casey W Dunn[1*], Mark Howison[1,2] and Felipe Zapata[1]

Abstract

Background: In the past decade, transcriptome data have become an important component of many phylogenetic studies. They are a cost-effective source of protein-coding gene sequences, and have helped projects grow from a few genes to hundreds or thousands of genes. Phylogenetic studies now regularly include genes from newly sequenced transcriptomes, as well as publicly available transcriptomes and genomes. Implementing such a phylogenomic study, however, is computationally intensive, requires the coordinated use of many complex software tools, and includes multiple steps for which no published tools exist. Phylogenomic studies have therefore been manual or semiautomated. In addition to taking considerable user time, this makes phylogenomic analyses difficult to reproduce, compare, and extend. In addition, methodological improvements made in the context of one study often cannot be easily applied and evaluated in the context of other studies.

Results: We present Agalma, an automated tool that constructs matrices for phylogenomic analyses. The user provides raw Illumina transcriptome data, and Agalma produces annotated assemblies, aligned gene sequence matrices, a preliminary phylogeny, and detailed diagnostics that allow the investigator to make extensive assessments of intermediate analysis steps and the final results. Sequences from other sources, such as externally assembled genomes and transcriptomes, can also be incorporated in the analyses. Agalma is built on the BioLite bioinformatics framework, which tracks provenance, profiles processor and memory use, records diagnostics, manages metadata, installs dependencies, logs version numbers and calls to external programs, and enables rich HTML reports for all stages of the analysis. Agalma includes a small test data set and a built-in test analysis of these data. In addition to describing Agalma, we here present a sample analysis of a larger seven-taxon data set. Agalma is available for download at https://bitbucket.org/caseywdunn/agalma.

Conclusions: Agalma allows complex phylogenomic analyses to be implemented and described unambiguously as a series of high-level commands. This will enable phylogenomic studies to be readily reproduced, modified, and extended. Agalma also facilitates methods development by providing a complete modular workflow, bundled with test data, that will allow further optimization of each step in the context of a full phylogenomic analysis.

Keywords: Transcriptomes, Assembly, Phylogenetics, Homology, Workflow, Pipeline

Background

Transcriptome data are fast becoming an important and cost effective component of phylogenetic studies [1-5]. The rapid fall in sequencing prices has contributed to the growing number of phylogenetic studies that integrate data from genomes and transcriptomes, often referred to as "phylogenomic" analyses. There is wide recognition that adding data from a larger number of genes is necessary to address many open phylogenetic questions, though of course additional gene sequences alone will not be sufficient to resolve them all [6-8].

The primary impediments to wider adoption and further improvement of phylogenomic methods are the complexity of the analyses and the lack of integrated tools to conduct them. Each phylogenomic study requires many steps, the vast majority of which concern matrix construction rather than phylogenetic analysis itself. These steps include raw data filtering, assembly, identification

*Correspondence: casey_dunn@brown.edu
[1]Department of Ecology and Evolutionary Biology, Brown University, Providence, Rhode Island, USA
Full list of author information is available at the end of the article

of ribosomal RNA, selection of transcript splice variants, translation, identification of homologous sequences, identification of orthologous sequences, sequence alignment, phylogenetic analysis, and summary of results. Implementing a phylogenomic analysis is not just a matter of executing available tools for each of these steps. Among other challenges, results must be summarized across multiple steps, detailed records must be kept of all analysis steps, data files often need to be reformatted between analyses, and computational load must be balanced according to the available resources.

Because phylogenomic studies are complex and have been manual or semi-automated, they are difficult to implement and explicitly describe, and require extensive technical effort to reproduce. These problems can make it difficult to evaluate results, integrate data across studies, expand analyses, or test the impact of alternative analysis approaches. In addition, manual analyses often include many subjective decisions that may impact the final results.

Some higher-level pipelines have addressed subsets of phylogenomic analyses. These tools include PartiGene [9], a pipeline to aid in the assembly and annotation of Sanger transcriptome data collected across a diversity of species, and SCaFoS [10], a semi-automated tool for the curation of super matrices from previously assembled transcriptomes. No existing tool, however, can execute a full phylogenomic analysis of modern sequence data.

We addressed these needs by developing Agalma, an automated phylogenomics workflow. Using Agalma, an investigator can conduct complete phylogenomic analyses, from raw sequence reads to preliminary phylogenetic trees, with a small number of high-level commands. The results are accompanied by detailed reports that integrate diagnostic information across data sets and analysis steps. In a first pass with Agalma, the investigator conducts the analysis under default settings. The investigator then consults the reports to consider how best to optimize the analyses, and easily re-executes them with updated settings to produce new matrices and preliminary trees. The investigator can then analyze the optimized matrices with external phylogenetic inference tools not already included within Agalma to explore other factors, such as model sensitivity.

Implementation

We built Agalma with BioLite [11], a generic bioinformatics pipeline framework. BioLite profiles memory and CPU use, tracks provenance of all data and analyses, and centralizes diagnostic reporting. Agalma is a modular workflow comprised of helper scripts and a series of pipelines. Each pipeline is made up of stages that call internal analysis functions (many implemented with the help of Biopython [12]) and wrappers from the BioLite Python module. The wrappers invoke command-line tools, which include external bioinformatics tools such as the Bowtie2 aligner [13] and Trinity assembler [14], as well as several C++ tools from BioLite. All calls to these external programs are logged, as are the version numbers of the programs and a hash of the binary so that it is possible to detect differences in compiler settings. This information about calls to external programs is available to the user via the `agalma diagnostics` command. We provide automated installation tools for Agalma, BioLite, and all required third party software. As a result, complete installation of Agalma and all dependencies requires only a handful of commands on OS X and Ubuntu, and the scripts facilitate installation on many other UNIX compliant systems as well. Installation details are provided in the `README` and `INSTALL` files included with BioLite and Agalma.

All interactions with Agalma are with the command `agalma`. The command `agalma assemble`, for example, provides access to the `assemble` pipeline. These commands include built-in help via the `-h` option, which provides details on all arguments and the parameters for external packages that are exposed to the user through Agalma. The command `agalma -h` provides help for the top-level wrapper. The command `agalma assemble -h`, for example, provides help for the assemble command. Additional details regarding the interface can be found in the `README` file.

The first step for analyzing each data set, whether it consists of raw reads to be assembled or of previously assembled gene predictions, is to catalog the data. This creates a database entry that includes sample metadata and the paths to the data files. Agalma has built-in support for transcriptome assembly of pair-end Illumina data only in FASTQ format. When analyzing public data, raw reads and associated metadata can be imported directly from the NCBI Sequence Read Archives (SRA) using the command `sra import`. This command downloads the reads for a given SRA accession number (experiment, study, sample, or run), converts them into FASTQ format, and populates the catalog with the corresponding data paths and metadata. When previously assembled gene predictions are used, the assembly files in FASTA format can be directly catalogued.

There are several distinct tasks subsequent to cataloging the data–sequence assembly, loading the genes into the database, and phylogenetic analysis. These tasks are described in detail in the `README` and `TUTORIAL` files provided with Agalma, and are briefly summarized below.

Assembly

The pipeline `transcriptome` runs an assembly from read filtering through assembly and preliminary annotation. In a typical analysis, `transcriptome` would be run once for each species for which raw Illumina transcriptome data are available. The `transcriptome` pipeline executes the following sub-pipelines, which can also be run individually:

- `sanitize` filters raw paired-end Illumina data and randomizes the order of reads (maintaining read order between paired files) to facilitate subsequent subsetting. Reads are discarded if they do not meet a specified mean quality threshold, if they contain Illumina adapter sequences, or if the base composition is highly skewed (if any base represents either $< 5\%$ or $> 60\%$ of the sequence). This pipeline also generates FastQC [15] summaries of read quality.
- `insert_size` uses subassemblies and Bowtie2 mapping to estimate the mean and variance of the insert size (i.e., the length of the fragment between the sequencing adapters). This information provides important feedback on the success of sample preparation, and is also used in some downstream analysis steps.
- `remove_rrna` removes most reads that are derived from ribosomal RNA (rRNA). Removing rRNA in advance of assembly can reduce both the number of chimeric transcripts and the time required for assembly. This pipeline first assembles multiple subsets of reads. A range of subset sizes is used since the optimal number of reads for assembling a particular rRNA transcript depends upon multiple factors, including the fraction of reads that originate from rRNA and the uniformity of coverage across the rRNA transcripts (which can vary greatly, depending on how denatured the samples were prior to fragmentation). rRNA transcripts are then identified by blast comparison of these subassemblies to an included dataset of known rRNA sequences. The entire set of reads is then compared to the rRNA transcripts that are identified from the subassemblies, and any reads that Bowtie2 maps to them are removed. A plot of the distribution of reads across exemplar rRNA transcripts is shown to help evaluate rRNA assembly success. The top hit in the NCBI `nt` database is also provided as an independent check on sample identity and to help spot potential contaminants. The fraction of reads that derive from rRNA is also reported to aid in improving library preparations.
- `assemble` filters the reads at a higher quality threshold and assembles them. Assemblies can be conducted under multiple protocols (such as multiple assemblers, or the same assembler under different settings). This pipeline can also assemble multiple subsets of different numbers of reads, which provides perspective on how sequencing effort impacts assembly results. The default assembler is Trinity [16]. The wrapper we have included in Agalma for running Trinity makes two improvements over the wrapper script that comes with Trinity. First, we have added a filter in between the Chrysalis and Butterfly stages to remove components that are smaller than the minimum transcript length parameter passed to Butterfly, since running Butterfly on these components will not yield a transcript. For the five assemblies in our test data set, this reduces the number of Butterfly commands from roughly 100,000 to 60,000. Second, we have replaced the ParaFly utility that is used for concurrent execution of the Butterfly commands with the GNU `parallel` tool [17] because it has better parallel efficiency. ParaFly executes the commands concurrently, but in blocks, so that the time to execute a block is the runtime of the slowest individual command. The runtimes can vary greatly because of variance in transcript length and complexity. In contrast, `parallel` load balances the commands across the available processors.
- `postassemble` uses `blastn` against the rRNA reference sequences to identify rRNA transcripts (these could include low abundance transcripts, such as parasite contaminants, that were not removed as reads by `remove_rrna`) and screens against the NCBI UniVec database to identify vector contaminants (such as protein expression vector contaminants in the sample preparation enzymes, which we have encountered in multiple samples). The longest likely coding region per transcripts is identified and annotated using `blastp` against the NCBI SwissProt database. Reads are mapped to the assembly and all transcripts quantified using RSEM [18], and the splice variant with the highest expression is selected as the exemplar transcript for each gene for downstream analyses. It also produces a coverage map that helps assess the distribution of sequencing effort across genes. Finally, it uses `blastx` to compare all the transcripts against the NCBI SwissProt database to establish which are similar to previously known proteins.

Following these steps, the investigator can inspect the assembled data directly or load them into the Agalma database to prepare them for phylogenetic analysis.

Load genes into the local Agalma database

Subsequent phylogenetic analyses require that all gene sequences to be considered are loaded into the local Agalma database. The `load` command takes care of this process. In a typical analysis, `load` is executed once for each dataset that has been assembled by the `transcriptome` pipeline described above, and once for each set of gene predictions from external sources (e.g., externally assembled 454 transcriptome data or gene predictions from genome sequencing projects).

Phylogenetic analysis

Once assemblies for multiple species are loaded into the local Agalma database, the user carries out a phylogenomic analysis by consecutively executing the following pipelines:

- `homologize` allows the user to specify which datasets to include in a particular phylogenetic analysis. It then uses an all-by-all `tblastx` search to build a graph with edges representing hits above a stringent threshold, and breaks the graph into connected components corresponding to clusters of homologous sequences with the Markov Clustering Algorithm (MCL) tool [19].
- `multalign` applies sampling and length filters to each cluster of homologous sequences, and uses another all-by-all `tblastx` within each cluster to trim sequence ends that have no similarity to other sequences in the cluster (these could include, for example, chimeric regions). The sequences of each cluster are aligned using MACSE [20], a translation-aware multiple-sequence aligner that accounts for frameshifts and stop codons. Multiple sequence alignments are then cleaned with GBLOCKS [21]. Optionally, the alignments can be concatenated together to form a supermatrix.
- `genetree` uses RAxML [22] to build a maximum likelihood phylogenetic tree for each cluster of homologous sequences. Gene trees can be filtered according to mean bootstrap support, which eliminates genes that have little phylogenetic signal [23] and reduces overall computational burden. This filter can be applied prior to running `treeprune` (described below), which has the added advantage of restricting ortholog selection to only well-supported gene trees. All options available in RAxML can be passed as optional arguments. If the input is a supermatrix consisting of concatenated orthologs, it builds a species tree.
- `treeprune` identifies orthologs according to the topology of gene phylogenies, using a new implementation of the method introduced in a previous phylogenomic study [24]. It uses DendroPy [25] to prune each gene tree into maximally inclusive subtrees with no more than one sequence per taxon. Each of these subtrees is considered as a set of putative orthologs. The pruned subtrees are re-entered as clusters into Agalma's database.

After `treeprune`, the user can build a supermatrix and a preliminary maximum likelihood species tree with RAxML. These steps, which include rerunning `multalign` and `genetree` on the orthologs, are detailed in the Agalma `TUTORIAL` file. Agalma produces a partitions file that describes which regions of the supermatrix correspond to which genes. The user can then proceed with more extensive phylogenetic analyses of the supermatrix using external phylogenetic inference software of their choice (only RAxML is included with Agalma at this time). As the alignments for each gene are also provided, the investigator can also apply promising new approaches that simultaneously infer gene trees and species trees [26].

Test data and analyses

A small set of test data is provided with Agalma. It consists of 25,000 100bp Illumina read pairs for the siphonophore *Hippopodius hippopus*, a subset of 72-74 gene sequences assembled for each of five siphonophores, and a subset of 40 gene predictions from the *Nematostella vectensis* genome assembly. These data were chosen because they run relatively quickly and enable testing of most commonly used features.

These test data serve several purposes. They allow a user to validate that Agalma is working correctly, and users are strongly encouraged to run this test with the command `agalma test` right after installation. This test analysis takes about 20 minutes on a 2-core 2 GHz computer. The test data also serve as the foundation for the example analysis described in the `TUTORIAL` file. For the developer, the `agalma test` command serves as a regression test to check if changes break existing features. We routinely run this test in the course of adding and refactoring code. The test data also serve as a minimal case study for developing new features without needing to first download and curate data.

Results and discussion

In addition to the small test data sets included with Agalma, here we present an example analysis of larger data sets from seven species. Though most phylogenetic analyses would include more taxa than this simple example case, the size of the dataset for each species is typical for contemporary phylogenomic analyses. This seven-taxon data set consists of new raw Illumina reads for five species of siphonophores, *Abylopsis tetragona*, *Agalma elegans*, *Nanomia bijuga*. *Physalia physalis*, *Craseoa sp.*, and gene predictions for two outgroup taxa, *Nematostella vectensis* [27] and *Hydra magnipapillata* [28], produced by previous genome projects. mRNA for the five siphonophore samples was isolated with Dynabeads mRNA DIRECT (Life Technologies) and prepared for sequencing with the TruSeq RNA Sample Prep Kit (Illumina). The sample preparation was modified by including a size selection step (agarose gel cut) prior to PCR amplification. Analyses were conducted with Agalma at git commit `dc549d23` and BioLite at git commit `025fe65e` (versions 0.3.4 with patches).

We deposited the new data in a public archive (NCBI Sequence Read Archive, BioProject PRJNA205486) prior to running the final analysis. A git repository of the scripts we used to execute the example analyses is available at https://bitbucket.org/caseywdunn/dunnhowisonzapata2013. These scripts download the data from the public archives, execute the analyses, and generate the analysis reports. All of the figures presented here are taken from the analysis reports generated by Agalma. This illustrates how a fully reproducible and open phylogenomic analysis can be implemented and communicated with Agalma. These scripts can be used as they are to repeat the analyses. They could also be modified to try alternative analysis strategies on these same data, or they could be adapted to run similar analyses on different data.

Assembly

The tabular assembly report (`index.html` in Additional file 1) summarizes assembly statistics across samples, and links to more detailed assembly reports for each sample. For the example analysis, this summary indicates, among other things, that the fraction of rRNA in each library ranged from 0.4% to 27.2% and the insert sizes were on average 266 bp long. The detailed assembly reports have extensive diagnostics that pertain to sample quality and the success of library preparation. As an example, Figure 1 shows several of the plots from the detailed assembly report for *Agalma elegans*, the siphonophore after which our tool is named. The distribution of sequencing effort across transcripts (Figure 1a) and the size distribution of transcripts (Figure 1b) are typical for *de novo* Illumina transcriptome assemblies.

Phylogenetic analyses

The Agalma phylogeny report includes a plot of the number of sequences considered at each stage of the

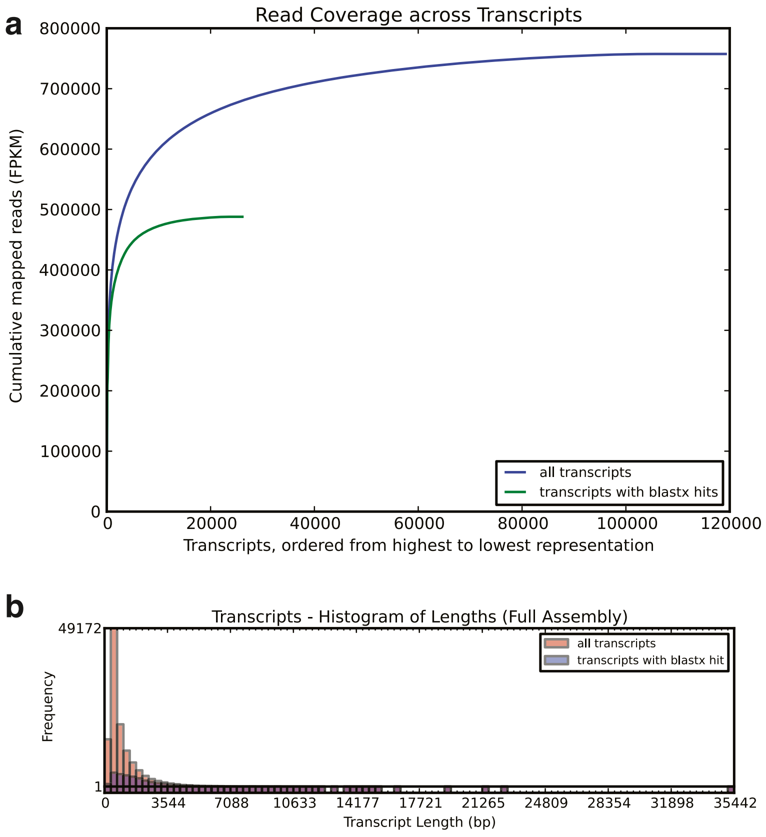


Figure 1 Transcriptome assembly statistics for *Agalma elegans*. (a) Cumulative read coverage, sorted from the most frequently sequenced exemplar transcripts to the least frequently sequenced transcripts. Most reads map to a small fraction of the genes, as is typical for a transcriptome sequencing project. **(b)** The size distribution of the assembled transcripts, for all transcripts and for only those transcripts that have a blastx hit to a protein in the Swissprot database. Most small transcripts do not have blastx hits and likely include some non-coding RNA and many assembly errors. These are removed prior to phylogenetic analysis. See Additional file 1 for further assembly diagnostics.

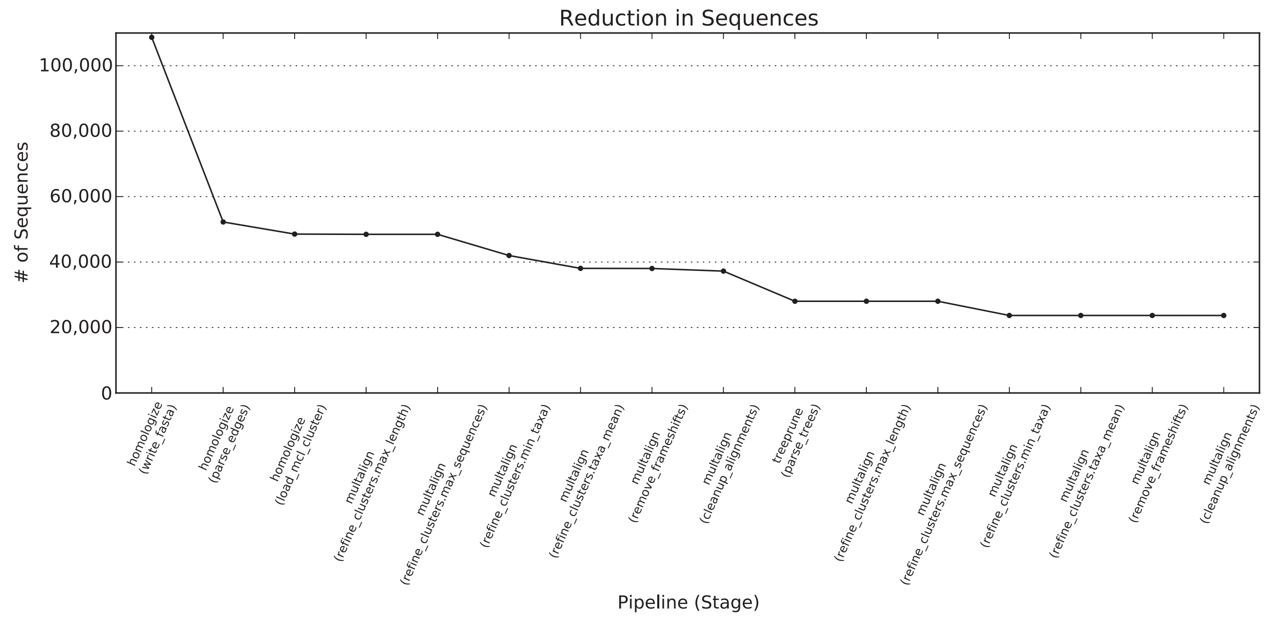


Figure 2 The number of gene sequences under consideration at each stage of super matrix construction. Most gene sequences are eliminated in the first step of homology evaluation (the `parse_edges` stage of the `homologize` pipeline), due to low similarity to other sequences. Of the remaining sequences, many are eliminated during multiple sequence alignment (the `multalign` pipeline) in clusters failing taxon sampling and cluster size criteria reflecting uncertainty in homology across species, and during orthology evaluation (the `treeprune` pipeline) mainly reflecting poor sampling of some ortholog groups. See Additional file 2 for further diagnostics regarding matrix construction.

analysis (Additional file 2). In the example analysis, the step that removed the most sequences was the first step of homology evaluation in the `homologize` pipeline (Figure 2). This reduction is due to low similarity to other sequences and clusters with poor taxon sampling. The next major reduction occurred in cluster refinement in `multalign`. This reduction is largely due to the elimination of clusters that failed the taxon sampling criteria, and reflects uncertainty regarding the homology of some sequences and sparse sampling of some homologs. The next major reduction in the number of genes occurred in `treeprune`. These reductions are due to both uncertainty regarding orthology and poor sampling of some ortholog groups. The preliminary species tree for the example analysis (Figure 3) is congruent with previous analyses of siphonophore relationships [29].

Resource utilization

Phylogenomic analyses are computationally intensive. Detailed information about resource utilization helps investigators plan resources for projects and balance computational load more efficiently. It is also critical for the optimization of the analyses, and can help guide design decisions. For each analysis, Agalma produces a resource utilization plot that displays the time and maximum memory used by external executables (Figure 4). The

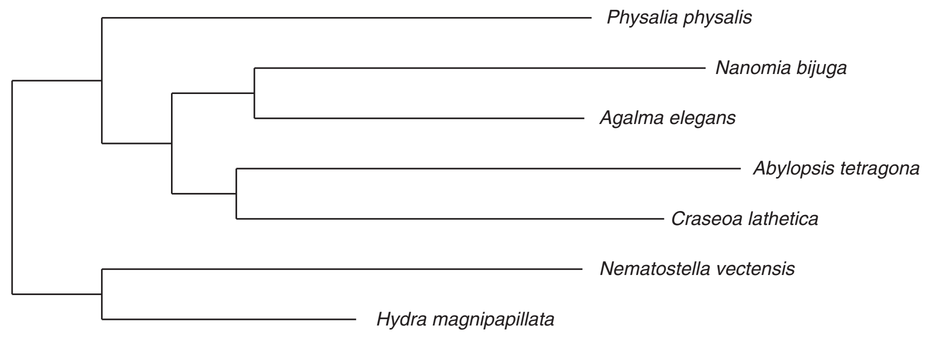
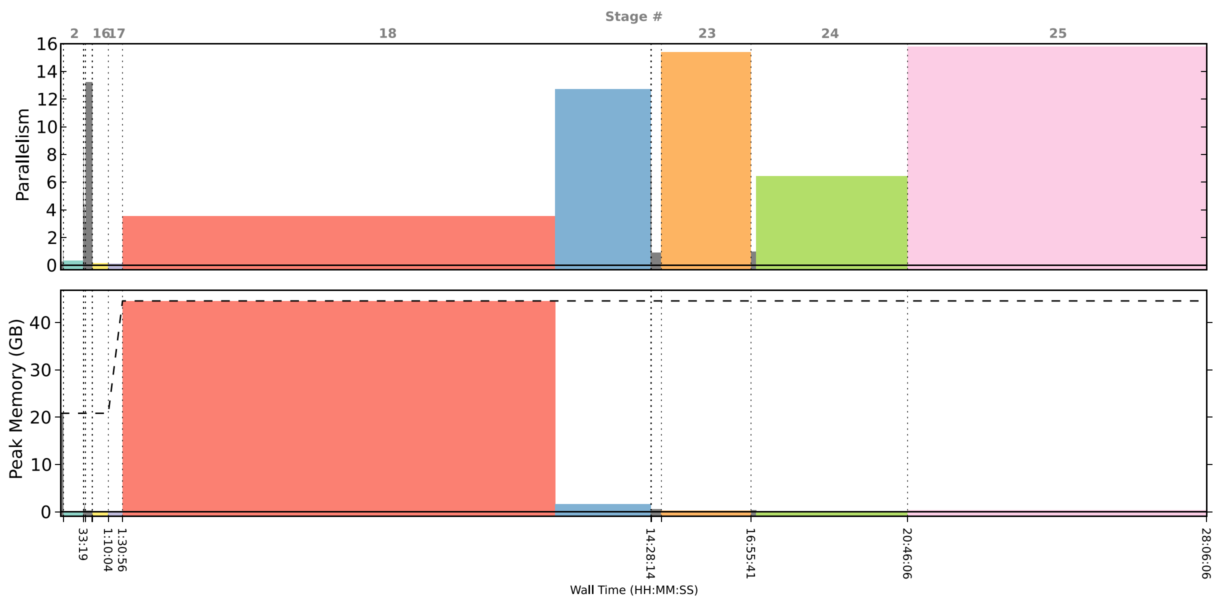


Figure 3 The preliminary maximum likelihood phylogeny resulting from the example analysis. This unrooted tree was inferred from the protein supermatrix under the $WAG + \Gamma$ model.

Resource Usage for *Agalma elegans* (Runs 3,12-13,18,32,38)



Calls longer than 1% of total runtime

#	Stage / Call	Runtime	User CPU%	System CPU%	Peak Memory
2	🟦 sanitize.sanitize.filter_illumina	29:16	32%	67%	1.3 MB
16	🟨 remove_rrna.exclude_ids.exclude	23:30	16%	83%	96.8 MB
17	🟪 assemble.quality_filter.filter_illumina	20:52	14%	84%	1.3 MB
18	🟥 assemble.assemble.trinity	10:36:37	354%	8%	44.6 GB
18	🟦 assemble.assemble.butterfly	2:20:40	1271%	55%	1.6 GB
23	🟧 postassemble.blastp.parallel	2:11:53	1540%	2%	262.6 MB
24	🟩 postassemble.coverage.rsem-calculate-expression	3:42:46	643%	11%	0 KB
25	🟪 postassemble.blastx.parallel	7:19:59	1580%	4%	394.0 MB

Figure 4 A profile of computational resource utilization for the `transcriptome` pipeline. This plot is from the report for the *Agalma elegans* assembly. Although annotation of transcripts (the calls to `blastx` and `blastp` during the `postasemble` stage) showed the highest processing time (top row), assembly (the call to Trinity during the `assemble` stage) displayed the highest memory use and it took the longest real time to finish (bottom row).

peak memory use, and the longest-running step, in the `transcriptome` pipeline was the call to Trinity during the `assemble` stage.

Conclusions

A distinction is sometimes drawn between manual approaches that enable close user inspection of data and results, and automated approaches that isolate the user from their results. This is a false dichotomy–automating analyses and examining the results closely are not mutually exclusive. Automated analyses with detailed diagnostics provide the best of both worlds. The user has a very detailed perspective on their analysis, and the added efficiencies of automation leave the investigator with far more time to assess these results. Automation also allow improvements made in the context of one study to be applied to other studies much more effectively.

For a study to be fully reproducible, both the data and the analysis must be described explicitly and unambiguously. The best description of an analysis is the code that was used to execute the analysis. By automating phylogenomic analyses from data download through matrix construction and preliminary phylogenetic trees, Agalma enables fully reproducible phylogenomic studies. This will allow reviewers and readers to reproduce an entire analysis exactly as it was run by the authors, without needing to re-curate the same dataset or rewrite the analysis code.

There are alternative approaches to many of the steps in a phylogenomic analysis presented here. There are, for example, multiple tools that identify orthologs according to different methods and criteria [30,31]. Agalma is a general framework and can be expanded to include these additional methods, and directly compare them in the context of a complete workflow that is consistent in all other regards.

Availability and requirements

Project name: Agalma
Project home page: https://bitbucket.org/caseywdunn/agalma
Operating system(s): Linux, Mac OS X
Programming language: Python
Other requirements: BioLite, and the dependencies listed in the BioLite `INSTALL` file
License: GNU

Additional files

Additional file 1: HTML report for assembly of the sample data sets. The HTML report for the assembly of the test data sets from raw reads. The tabular report (index.html) provides an overview across the five assemblies for the ingroup taxa, and includes links (in the Catalog ID column) to detailed reports for the assembly of each species. Fasta files for the annotated transcripts have been removed from the report to reduce file size.

Additional file 2: HTML report for phylogenetic analyses. The HTML report for the phylogenetic analysis of the sample data.

Competing interests
The authors declare that they have no competing interests.

Author's contributions
CWD and MH conceived of the original design of Agalma, and together with FZ made most design decisions. MH, CWD, and FZ implemented and tested the software and wrote the manuscript. MH did most of the coding. MH and FZ conducted the computational analyses. All authors read and approved the final manuscript.

Acknowledgements
Steve Haddock provided several of the specimens, and Freya Goetz prepared all material for sequencing. Thanks to Deb Goodwin and the students aboard *SSV Robert C. Seamans* for collecting the *Physalia physalis* specimens, and to Erik Zettler and the Sea Education Association for facilitating this opportunity. Thanks also to Christian Sardet for assisting with collecting the *Hippopodius hippopus*. Freya Goetz prepared the two siphonophore samples for sequencing. Stephen Smith and Nicholas Sinnott-Armstrong made important contributions to early planning of Agalma. Stefan Siebert assisted with curation of the test dataset and provided suggestions on features. Richard Nguyen provided HTML and CSS styling for the reports. Thanks to Steve Haddock, Warren Francis, Lingsheng Dong, Adrian Reich, Vanessa Gonzalez, and other early-adopters for feedback on previous versions of Agalma. Rebecca Helm provided helpful comments on an earlier version of this manuscript. This research was conducted using computational resources and services at the Center for Computation and Visualization, Brown University. Agalma was developed with the support of the US National Science Foundation (grants 0844596, 0844596, 1004057, and 1256695, and the Alan T. Waterman Award).

Author details
[1]Department of Ecology and Evolutionary Biology, Brown University, Providence, Rhode Island, USA. [2]Center for Computation and Visualization, Brown University, Providence, Rhode Island, USA.

Received: 3 July 2013 Accepted: 13 November 2013
Published: 19 November 2013

References
1. Dunn CW, Hejnol A, Matus DQ, Pang K, Browne WE, Smith SA, Seaver E, Rouse GW, Obst M, Edgecombe GD, Sørensen MV, Haddock SHD, Schmidt-Rhaesa A, Okusu A, Kristensen RM, Wheeler WC, Martindale MQ, Giribet G: **Broad phylogenomic sampling improves resolution of the animal tree of life.** *Nature* 2008, **452**(7188):745–749. [http://dx.doi.org/10.1038/nature06614]
2. Hejnol A, Obst M, Stamatakis A, Ott M, Rouse GW, Edgecombe GD, Martinez P, Baguñà J, Bailly X, Jondelius U, Wiens M, Müller WEG, Seaver E, Wheeler WC, Martindale MQ, Giribet G, Dunn CW: **Assessing the root of bilaterian animals with scalable phylogenomic methods.** *Proc R Soc B* 2009, **276**(1677):4261–4270.
3. Smith SA, Wilson NG, Goetz FE, Feehery C, Andrade SCS, Rouse GW, Giribet G, Dunn CW: **Resolving the evolutionary relationships of molluscs with phylogenomic tools.** *Nature* 2011, **480**(7377):364–367.
4. Philippe H, Derelle R, Lopez P, Pick K, Borchiellini C, Boury-Esnault N, Vacelet J, Renard E, Houliston E, QuEinnec E, Da Silva C, Wincker P, Le Guyader H, Leys S, Jackson DJ, Schreiber F, Erpenbeck D, Morgenstern B, WOrheide G, Manuel M: **Phylogenomics Revives Traditional Views on Deep Animal Relationships.** *Curr Biol : CB* 2009, **19**(8):706–712. [http://dx.doi.org/10.1016/j.cub.2009.02.052]
5. Delsuc F, Brinkmann H, Chourrout D, Philippe H: **Tunicates and not cephalochordates are the closest living relatives of vertebrates.** *Nature* 2006, **439**(7079):965–968. [http://www.nature.com/doifinder/10.1038/nature04336]
6. Sanderson MJ: **Phylogenetic Signal in the Eukaryotic Tree of Life.** *Sci (New York, N.Y.)* 2008, **321**(5885):121–123. [http://www.sciencemag.org/cgi/doi/10.1126/science.1154449]
7. Philippe H, Telford MJ: **Large-scale sequencing and the new animal phylogeny.** *Trends Ecol Evol* 2006, **21**(11):614–620. [http://dx.doi.org/10.1016/j.tree.2006.08.004]
8. Edgecombe GD, Giribet G, Dunn CW, Hejnol A, Kristensen RM, Neves RC, Rouse GW, Worsaae K, Sørensen MV: **Higher-level metazoan relationships: recent progress and remaining questions.** *Org Divers Evol* 2011, **11**(2):151–172. [http://www.springerlink.com/index/10.1007/s13127-011-0044-4]
9. Parkinson J, Anthony A, Wasmuth J, Schmid R, Hedley A, Blaxter M: **PartiGene–constructing partial genomes.** *Bioinformatics (Oxford, England)* 2004, **20**(9):1398–1404. [http://dx.doi.org/10.1093/bioinformatics/bth101]
10. Roure B, Rodriguez-Ezpeleta N, Philippe H: **SCaFoS: a tool for selection, concatenation and fusion of sequences for phylogenomics.** *BMC Evol Biol* 2007, **7**(Suppl 1):S2. [http://www.biomedcentral.com/1471-2148/7/S1/S2]
11. Howison M, Dunn CW, Sinnott-Armstrong NA: **BioLite, a lightweight bioinformatics framework with automated tracking of diagnostics and provenance.** In *Proceedings of the 4th USENIX Workshop on the Theory and Practice of Provenance (TaPP '12)*. Boston, MA, USA; 14–15 June 2012. https://www.usenix.org/system/files/conference/tapp12/tapp12-final5.pdf.
12. Cock PJA, Antao T, Chang JT, Chapman BA, Cox CJ, Dalke A, Friedberg I, Hamelryck T, Kauff F, Wilczynski B, Hoon MJLd: **Biopython: freely available Python tools for computational molecular biology and bioinformatics.** *Bioinformatics* 2009, **25**(11):1422–1423. [http://bioinformatics.oxfordjournals.org/content/25/11/1422] [PMID: 19304878]
13. Langmead B, Salzberg SL: **Fast gapped-read alignment with Bowtie 2.** *Nat Methods* 2012, **9**(4):357–359. [http://www.nature.com/doifinder/10.1038/nmeth.1923]
14. Grabherr MG, Haas BJ, Yassour M, Levin JZ, Thompson DA, Amit I, Adiconis X, Fan L, Raychowdhury R, Zeng Q, Chen Z, Mauceli E, Hacohen N, Gnirke A, Rhind N, di Palma F, Birren BW, Nusbaum C, Lindblad-Toh K, Friedman N, Regev A: **Full-length transcriptome assembly from RNA-Seq data without a reference genome.** *Nat Biotechnol* 2011, [http://www.nature.com/doifinder/10.1038/nbt.1883]
15. Andrews S: **FastQC: A quality control tool for high throughput sequence data** [http://www.bioinformatics.bbsrc.ac.uk/projects/fastqc/]
16. Grabherr MG, Haas BJ, Yassour M, Levin JZ, Thompson DA, Amit I, Adiconis X, Fan L, Raychowdhury R, Zeng Q, Chen Z, Mauceli E, Hacohen N, Gnirke A, Rhind N, di Palma F, Birren BW, Nusbaum C, Lindblad-Toh K, Friedman N, Regev A: **Full-length transcriptome assembly from RNA-Seq data without a reference genome.** *Nat Biotech* 2011, **29**(7):644–652.

17. Tange O: **GNU Parallel: the command-line power tool.** *Login* 2011, **36:**42–47. [https://www.usenix.org/publications/login/february-2011-volume-36-number-1/gnu-parallel-command-line-power-tool]
18. Li B, Dewey CN: **RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome.** *BMC Bioinformatics* 2011, **12:**323.
19. Enright AJ, Van Dongen S, Ouzounis CA: **An efficient algorithm for large-scale detection of protein families.** *Nucleic Acids Research* 2002, **30**(7):1575–1584. [http://dx.doi.org/10.1093/nar/30.7.1575].
20. Ranwez V, Harispe S, Delsuc F, Douzery EJP: **MACSE: multiple alignment of coding SEquences accounting for frameshifts and stop codons.** *PLoS ONE* 2011, **6**(9):e22594.
21. Talavera G, Castresana J: **Improvement of phylogenies after removing divergent and ambiguously aligned blocks from protein sequence alignments.** *Syst Biol* 2007, **56**(4):564–577.
22. Stamatakis A: **RAxML-VI-HPC: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models.** *Bioinformatics (Oxford, England)* 2006, **22**(21):2688–2690.
23. Salichos L, Rokas A: **Inferring ancient divergences requires genes with strong phylogenetic signals.** *Nature* 2013, **497**(7449):327–331. [http://dx.doi.org/10.1038/nature12130]
24. Hejnol A, Obst M, Stamatakis A, Ott M, Rouse GW, Edgecombe GD, Martinez P, Baguñà J, Bailly X, Jondelius U, Wiens M, Müller WEG, Seaver E, Wheeler WC, Martindale MQ, Giribet G, Dunn CW: **Assessing the root of bilaterian animals with scalable phylogenomic methods.** *Proc Biol Sci / R Soc* 2009, **276**(1677):4261–4270. [http://dx.doi.org/10.1098/rspb.2009.0896]
25. Sukumaran J, Holder MT: **DendroPy: a Python library for phylogenetic computing.** *Bioinformatics* 2010, **26**(12):1569–1571. [http://bioinformatics.oxfordjournals.org/content/26/12/1569]
26. Boussau B, Szollosi GJ, Duret L, Gouy M, Tannier E, Daubin V: **Genome-scale coestimation of species and gene trees.** *Genome Res* 2013, **23**(2):323–330. [http://genome.cshlp.org/cgi/doi/10.1101/gr.141978.112]
27. Putnam NH, Srivastava M, Hellsten U, Dirks B, Chapman J, Salamov A, Terry A, Shapiro H, Lindquist E, Kapitonov VV, Jurka J, Genikhovich G, Grigoriev IV, Lucas SM, Steele RE, FINNERTY JR, Technau U, Martindale MQ, Rokhsar DS: **Sea anemone genome reveals ancestral Eumetazoan gene repertoire and genomic organization.** *Sci (New York, N.Y.)* 2007, **317**(5834):86–94. [http://www.sciencemag.org/cgi/doi/10.1126/science.1139158]
28. Chapman JA, Kirkness EF, Simakov O, Hampson SE, Mitros T, Weinmaier T, Rattei T, Balasubramanian PG, Borman J, Busam D, Disbennett K, Pfannkoch C, Sumin N, Sutton GG, Viswanathan LD, Walenz B, Goodstein DM, Hellsten U, Kawashima T, Prochnik SE, Putnam NH, Shu S, Blumberg B, Dana CE, Gee L, Kibler DF, Law L, Lindgens D, Martinez DE, Peng J, et al.: **The dynamic genome of Hydra.** *Nature* 2010, **464**(7288):592–596. [http://dx.doi.org/10.1038/nature08830]
29. Dunn C, Pugh P, Haddock S: **Molecular Phylogenetics of the Siphonophora (Cnidaria), with implications for the evolution of functional specialization.** *Syst Biol* 2005, **54**(6):916–935. [http://sysbio.oxfordjournals.org/cgi/doi/10.1080/10635150500354837]
30. Chen F, Mackey AJ, Vermunt JK, Roos DS: **Assessing performance of orthology detection strategies applied to eukaryotic genomes.** *PLoS ONE* 2007, **2**(4):e383. [http://dx.plos.org/10.1371/journal.pone.0000383]
31. Altenhoff AM, Gil M, Gonnet GH, Dessimoz C: **Inferring hierarchical orthologous groups from orthologous gene pairs.** *PLoS ONE* 2013, **8:**e53786. [http://dx.doi.org/10.1371/journal.pone.0053786]

doi:10.1186/1471-2105-14-330
Cite this article as: Dunn *et al.*: Agalma: an automated phylogenomics workflow. *BMC Bioinformatics* 2013 **14**:330.