phylo-node: A Molecular Phylogenetic Toolkit using Node.js

Damien M. O'Halloran[1,2*]

1. Department of Biological Sciences, The George Washington University, Science and Engineering Hall, Rm 6000, 800 22nd Street N.W., Washington DC 20052, USA.

2. Institute for Neuroscience, The George Washington University, 636 Ross Hall, 2300 I St. N.W. Washington DC 20052, USA.

***Corresponding author address:**

Damien O'Halloran, The George Washington University, 636 Ross Hall, 2300 I St. N.W. Washington DC 20052, USA.

Tel: 202-994-8955

Fax: 202-994-6100

Email: damienoh@gwu.edu

ABSTRACT

Background: Node.js is an open-source and cross-platform environment that provides a JavaScript codebase for back-end server-side applications. JavaScript has been used to develop very fast and user-friendly front-end tools for bioinformatic and phylogenetic analyses. However, no such toolkits are available using Node.js to conduct comprehensive molecular phylogenetic analysis.

Results: To address this problem, I have developed, *phylo-node*, which was developed using Node.js and provides a stable and scalable toolkit that allows the user to perform diverse molecular and phylogenetic tasks. phylo-node can execute the analysis and process the resulting outputs from a suite of software options that provides tools for read processing and genome alignment, sequence retrieval, multiple sequence alignment, primer design, evolutionary modeling, and phylogeny reconstruction. Furthermore, phylo-node enables the user to deploy server dependent applications, and also provides simple integration and interoperation with other Node modules and languages using Node inheritance patterns, and a customized piping module to support the production of diverse pipelines.

Conclusions: phylo-node is open-source and freely available to all users without sign-up or login requirements. All source code and user guidelines are openly available at the GitHub repository: https://github.com/dohalloran/phylo-node

Keywords: Node.js, JavaScript, phylogenetics

BACKGROUND

The cost of whole genome sequencing has plummeted over the last decade and as a consequence, the demand for genome sequencing technology has risen significantly [1]. This demand has meant that producing large complex datasets of DNA and RNA sequence information is common in small research labs, and in terms of human health this boom in sequence information and precipitous drop in sequencing costs has had a direct impact in the area of personalized medicine [2-5]. However, once the sequence information becomes available, perhaps the greater challenge is then processing, analyzing, and interpreting the data. To keep pace with this challenge, the development of new, fast, and scalable software solutions is required to visualize and interpret this information.

JavaScript is a lightweight programming language that uses a web browser as its host environment. JavaScript is cross-platform and supported by all modern browsers. Because JavaScript is client-side, it is very fast as it doesn't have to communicate with a server and wait for a response in order to run some code. Web browsers are ubiquitous and require no dependencies to deploy and operate, and so JavaScript represents an obvious solution for visualizing sequence information. Front-end developments using JavaScript have proven to be extremely efficient in providing fast, easy-to-use, and embeddable solutions for data analysis [6-14]. A very active community of developers at BioJS (http://www.biojs.io/) provides diverse components for parsing sequence data types, data visualization, and bioinformatics analysis in JavaScript [6,7,15-19].

Node.js provides server-side back-end JavaScript. Node.js is written in C, C++, and JavaScript and uses the Google Chrome V8 engine to offer a very fast cross-platform environment for developing server side Web applications. Node is a

single-threaded environment, which means that only one line of code will be executed at any given time; however, Node employs non-blocking techniques for I/O tasks to provide an asynchronous ability, by using *callback* functions to permit the parallel running of code. Node holds much potential for the bioinformatic analysis of molecular data. A community of Node developers provides modules for bioinformatic sequence workflows (http://www.bionode.io/) which in time will likely parallel the BioJS community for the number of modules versus components. However, as of now there are no robust tools for phylogenetic analysis pipelines currently available using the Node.js codebase. To fill this void I have developed, *phylo-node*, which provides a Node.js toolkit that provides sequence retrieval, primer design, alignment, phylogeny reconstruction and as well as much more, all from a single toolkit. *phylo-node* is fast, easy to use, and offers simple customization and portability options through various inheritance patterns. The Node package manager, *npm* (https://www.npmjs.com/), provides a very easy and efficient way to manage dependencies for any Node application. phylo-node is available at GitHub (https://github.com/dohalloran/phylo-node), npm (https://www.npmjs.com/package/phylo-node), and also BioJS (http://www.biojs.io/d/phylo-node).

IMPLEMENTATION

phylo-node was developed using the Node.js codebase. The phylo-node core contains a base wrapper object that is used to prepare the arguments and directory prior to program execution. The base wrapper module is contained within the *Wrapper_core* directory (Figure 1). An individual software tool can be easily accessed and executed by importing the module for that tool so as to get access to

the method properties on that object. These method properties are available to the user by using the *module.exports* reference object. Inside a driver script file, the user can import the main module object properties and variables by using the *require* keyword which is used to import a module in Node.js. The *require* keyword is actually a global variable, and a script has access to its context because it is wrapped prior to execution inside the *runInThisContext* function (for more details, refer to the Node.js source code: https://github.com/nodejs). Once imported, the return value is assigned to a variable which is used to access the various method properties on that object. For example: a method property on the *phyml* object is *phyml.getphyml()*, which invokes the *getphyml* method on the *phyml* object to download and decompress the PhyML executable. For a complete list of all methods, refer to the *README.md* file at the GitHub repository (https://github.com/dohalloran/phylo-node/blob/master/README.md). In order to correctly wrap and run each executable, new shells must be spawned so as to execute specific command formats for each executable. This was achieved by using *child.process.exec*, which will launch an external shell and execute the command inside that shell while buffering any output by the process. Binary files and executables were downloaded and executed in this manner and the appropriate file and syntax selected by determining the user's operating system. phylo-node was validated on Microsoft Windows 7 Enterprise ver.6.1, MacOSX El Capitan ver.10.11.5, and Linux Ubuntu 64-bit ver.14.04 LTS.

RESULTS AND DISCUSSION

phylo-node is a toolkit to interface with key applications necessary in building a phylogenetic pipeline (Table 1). Firstly, phylo-node allows the user to remotely

download sequences by building a unique URL and passing this string to the NCBI e-utilities API (http://www.ncbi.nlm.nih.gov/books/NBK25501/). Any number of genes can be supplied as command-line arguments to phylo-node by accessing the *fetch_seqs.fasta* method on the *fetch_seqs* object in order to retrieve sequence information in FASTA format. The module for remote sequence retrieval is contained within the *Sequence* directory. phylo-node also provides methods on specific objects to download various executable files using the *download* module (Figure 1). Any binary can be downloaded using the base module *get_executable* contained within the *Download* directory, however objects pertaining to specific tools such as PhyML also contain methods for downloading and unpacking binaries (see README.md file for details). phylo-node also enables the user to create a web server to deploy embeddable applications such as JBrowse [11] which provides genome visualization from within a web browser. To facilitate interoperation between various applications and components, the phylo-node package also contains a module called *phylo-node_pipes* inside the *Pipes* directory. The *phylo-node_pipes* module allows the user to easily pipe data between different applications by requiring the *child_process* module which provides the ability to spawn child processes. Through *phylo-node_pipes*, the user can chain commands together that will be executed in sequence to build consistent, and extensive pipelines. The *Pipes* directory contains sample driver scripts for using the *phylo-node_pipes* module.

phylo-node is highly scalable and new modules for diverse applications can easily be plugged in. The modules required to wrap and execute applications are all contained within the *Run* directory. The following tools can be implemented using phylo-node from within the *./Tool/Run* directory: Trimmomatic [20] to process reads prior to read alignment; Bowtie2 [21] for read alignment to a reference sequence;

Primer3 [22-24] to facilitate primer design; Clustal Omega [25], K-align [26], and MUSCLE [27,28] for multiple sequence alignments; Codeml [29], PAL2NAL [30], and Slr [31] for selection analysis; jModelTest2 [32] and ProtTest3 [33] to determine the best-fit model of evolution; and PhyML [34,35] for phylogeny reconstruction. The PhyML executable is also employed by jModelTest2 and ProtTest3. These specific tools were selected because they are some of the most popular choices and applications in many bioinformatics pipelines: for example, Primer3 is the most popular software (over 15,000 citations) for primer design [36]; Clustal Omega, K-align, and MUSCLE are very fast and accurate multiple sequence alignment tools that are commonly used to build robust DNA, RNA, or protein alignments [37]; Codeml is part of the PAML suite [29], and alongside PAL2NAL [30] and Slr [31] are commonly used to determine rates of selection [38,39]. ProtTest3 and jModelTest2 are widely used to determine best-fit models of amino-acid replacements and nucleotide substitution [40-42]; PhyML is also a popular program (over 12,000 citations) for building phylogenies using maximum likelihood [34]; for next generation sequencing data, Trimmomatic and Bowtie2 are commonly implemented in read processing and mapping pipelines [43]. Sample input files for all applications deployed by phylo-node can be found in the *Input_examples* directory and sub-directories. Taken together, phylo-node provides a diverse toolkit that allows the user to develop robust pipelines and instances using Node.

phylo-node is highly scalable and customizable, and was inspired by projects such as BioPerl [44] which provides very diverse tools that include Perl modules for many bioinformatic tasks and also parsers and wrappers for diverse sequence formats and applications. BioPerl's open source structure and architecture allows users to plug new modules into BioPerl pipelines to design new applications. Node.js

implements prototypal inheritance as per JavaScript but also provides access to the *module.exports* object which permits easy portability between the phylo-node toolkit and any other modules, and also interoperation between different languages by using the *child.process.exec* process. Therefore, *phylo-node* can be integrated with existing Node.js bioinformatics tools [45,46] or software written in other languages. For example, jModelTest2, ProtTest3 and Trimmomatic require a Java runtime environment (http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html), and by using *require* to import each module, the user can execute the analysis of these tools.

CONCLUSIONS

In conclusion, phylo-node is a novel package that leverages the speed of Node.js to provide a robust and efficient toolkit for researchers conducting molecular phylogenetics. phylo-node can be easily employed to develop complex but consistent workflows, and integrated with existing bioinformatics tools using the Node.js codebase.

AVAILABILITY AND REQUIREMENTS

- Project name: phylo-node
- Project home page: https://github.com/dohalloran/phylo-node
- Operating system(s): Platform independent
- Programming language: Node.js
- Other requirements: none
- License: MIT
- Any restrictions to use by non-academics: no restrictions or login requirements

ACKNOWLEDGMENTS

I thank members of the O'Halloran lab for critical reading of the manuscript, and would like to thank The George Washington University (GWU) Columbian College of Arts and Sciences, GWU Office of the Vice-President for Research, and the GWU Department of Biological Sciences for Funding.

AUTHOR CONTRIBUTIONS

D.O'H. conceived the idea for *phylo-node*, wrote and tested the code, and wrote the manuscript.

COMPETING INTERESTS

The author declares no competing interests.

References

1. Shaffer C. Next-generation sequencing outpaces expectations. Nat Biotechnol. 2007;25: 149. doi: nbt0207-149 [pii].

2. Wade N. The quest for the $1,000 human genome: DNA sequencing in the doctor's office? At birth? It may be coming closer. N Y Times Web. 2006: F1, F3.

3. Mardis ER. Anticipating the 1,000 dollar genome. Genome Biol. 2006;7: 112. doi: gb-2006-7-7-112 [pii].

4. Service RF. Gene sequencing. The race for the $1000 genome. Science. 2006;311: 1544-1546. doi: 311/5767/1544 [pii].

5. Hayden EC. The $1,000 genome. Nature. 2014;507: 294-295.

6. Yachdav G, Goldberg T, Wilzbach S, Dao D, Shih I, Choudhary S, et al. Anatomy of BioJS, an open source community for the life sciences. Elife. 2015;4: 10.7554/eLife.07009. doi: 10.7554/eLife.07009 [doi].

7. Gomez J, Garcia LJ, Salazar GA, Villaveces J, Gore S, Garcia A, et al. BioJS: an open source JavaScript framework for biological data visualization. Bioinformatics. 2013;29: 1103-1104. doi: 10.1093/bioinformatics/btt100 [doi].

8. Salazar GA, Meintjes A, Mulder N. PPI layouts: BioJS components for the display of Protein-Protein Interactions. F1000Res. 2014;3: 50-50.v1. eCollection 2014. doi: 10.12688/f1000research.3-50.v1 [doi].

9. Gomez J, Jimenez R. Sequence, a BioJS component for visualising sequences. F1000Res. 2014;3: 52-52.v1. eCollection 2014. doi: 10.12688/f1000research.3-52.v1 [doi].

10. Cui Y, Chen X, Luo H, Fan Z, Luo J, He S, et al. BioCircos.js: an interactive Circos JavaScript library for biological data visualization on web applications. Bioinformatics. 2016;32: 1740-1742. doi: 10.1093/bioinformatics/btw041 [doi].

11. Buels R, Yao E, Diesh CM, Hayes RD, Munoz-Torres M, Helt G, et al. JBrowse: a dynamic web platform for genome visualization and analysis. Genome Biol. 2016;17: 66-016-0924-1. doi: 10.1186/s13059-016-0924-1 [doi].

12. Salavert F, Garcia-Alonso L, Sanchez R, Alonso R, Bleda M, Medina I, et al. Web-based network analysis and visualization using CellMaps. Bioinformatics. 2016. doi: btw332 [pii].

13. Franz M, Lopes CT, Huck G, Dong Y, Sumer O, Bader GD. Cytoscape.js: a graph theory library for visualisation and analysis. Bioinformatics. 2016;32: 309-311. doi: 10.1093/bioinformatics/btv557 [doi].

14. Vanderkam D, Aksoy BA, Hodes I, Perrone J, Hammerbacher J. pileup.js: a JavaScript library for interactive and in-browser visualization of genomic data. Bioinformatics. 2016. doi: btw167 [pii].

15. Garcia L, Yachdav G, Martin MJ. FeatureViewer, a BioJS component for visualization of position-based annotations in protein sequences. F1000Res. 2014;3: 47-47.v2. eCollection 2014. doi: 10.12688/f1000research.3-47.v2 [doi].

16. Kalderimis A, Stepan R, Sullivan J, Lyne R, Lyne M, Micklem G. BioJS DAGViewer: A reusable JavaScript component for displaying directed graphs. F1000Res. 2014;3: 51-51.v1. eCollection 2014. doi: 10.12688/f1000research.3-51.v1 [doi].

17. Villaveces JM, Jimenez RC, Habermann BH. KEGGViewer, a BioJS component to visualize KEGG Pathways. F1000Res. 2014;3: 43-43.v1. eCollection 2014. doi: 10.12688/f1000research.3-43.v1 [doi].

18. Villaveces JM, Jimenez RC, Habermann BH. PsicquicGraph, a BioJS component to visualize molecular interactions from PSICQUIC servers. F1000Res. 2014;3: 44-44.v1. eCollection 2014. doi: 10.12688/f1000research.3-44.v1 [doi].

19. Yachdav G, Hecht M, Pasmanik-Chor M, Yeheskel A, Rost B. HeatMapViewer: interactive display of 2D data in biology. F1000Res. 2014;3: 48-48.v1. eCollection 2014. doi: 10.12688/f1000research.3-48.v1 [doi].

20. Bolger AM, Lohse M, Usadel B. Trimmomatic: a flexible trimmer for Illumina sequence data. Bioinformatics. 2014;30: 2114-2120. doi: 10.1093/bioinformatics/btu170 [doi].

21. Langmead B, Salzberg SL. Fast gapped-read alignment with Bowtie 2. Nat Methods. 2012;9: 357-359. doi: 10.1038/nmeth.1923 [doi].

22. You FM, Huo N, Gu YQ, Luo MC, Ma Y, Hane D, et al. BatchPrimer3: a high throughput web application for PCR and sequencing primer design. BMC Bioinformatics. 2008;9: 253-2105-9-253. doi: 10.1186/1471-2105-9-253 [doi].

23. Untergasser A, Cutcutache I, Koressaar T, Ye J, Faircloth BC, Remm M, et al. Primer3--new capabilities and interfaces. Nucleic Acids Res. 2012;40: e115. doi: gks596 [pii].

24. Untergasser A, Nijveen H, Rao X, Bisseling T, Geurts R, Leunissen JA. Primer3Plus, an enhanced web interface to Primer3. Nucleic Acids Res. 2007;35: W71-4. doi: gkm306 [pii].

25. Sievers F, Wilm A, Dineen D, Gibson TJ, Karplus K, Li W, et al. Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega. Mol Syst Biol. 2011;7: 539. doi: 10.1038/msb.2011.75 [doi].

26. Lassmann T, Sonnhammer EL. Kalign--an accurate and fast multiple sequence alignment algorithm. BMC Bioinformatics. 2005;6: 298. doi: 1471-2105-6-298 [pii].

27. Edgar RC. MUSCLE: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Res. 2004;32: 1792-1797. doi: 10.1093/nar/gkh340.

28. Edgar RC. MUSCLE: a multiple sequence alignment method with reduced time and space complexity. BMC Bioinformatics. 2004;5: 113. doi: 10.1186/1471-2105-5-113 [doi].

29. Yang Z. PAML: a program package for phylogenetic analysis by maximum likelihood. Comput Appl Biosci. 1997;13: 555-556.

30. Suyama M, Torrents D, Bork P. PAL2NAL: robust conversion of protein sequence alignments into the corresponding codon alignments. Nucleic Acids Res. 2006;34: W609-12. doi: 34/suppl_2/W609 [pii].

31. Massingham T, Goldman N. Detecting amino acid sites under positive selection and purifying selection. Genetics. 2005;169: 1753-1762. doi: genetics.104.032144 [pii].

32. Darriba D, Taboada GL, Doallo R, Posada D. jModelTest 2: more models, new heuristics and parallel computing. Nat Methods. 2012;9: 772. doi: 10.1038/nmeth.2109 [doi].

33. Darriba D, Taboada GL, Doallo R, Posada D. ProtTest 3: fast selection of best-fit models of protein evolution. Bioinformatics. 2011;27: 1164-1165. doi: 10.1093/bioinformatics/btr088; 10.1093/bioinformatics/btr088.

34. Guindon S, Gascuel O. A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. Syst Biol. 2003;52: 696-704.

35. Guindon S, Dufayard JF, Lefort V, Anisimova M, Hordijk W, Gascuel O. New algorithms and methods to estimate maximum-likelihood phylogenies: assessing the performance of PhyML 3.0. Syst Biol. 2010;59: 307-321. doi: 10.1093/sysbio/syq010 [doi].

36. Rozen S, Skaletsky H. Primer3 on the WWW for general users and for biologist programmers. Methods Mol Biol. 2000;132: 365-386.

37. Thompson JD, Linard B, Lecompte O, Poch O. A comprehensive benchmark study of multiple sequence alignment methods: current challenges and future perspectives. PLoS One. 2011;6: e18093. doi: 10.1371/journal.pone.0018093 [doi].

38. Zhang C, Wang J, Long M, Fan C. gKaKs: the pipeline for genome-level Ka/Ks calculation. Bioinformatics. 2013;29: 645-646. doi: 10.1093/bioinformatics/btt009 [doi].

39. Huerta-Cepas J, Serra F, Bork P. ETE 3: Reconstruction, Analysis, and Visualization of Phylogenomic Data. Mol Biol Evol. 2016;33: 1635-1638. doi: 10.1093/molbev/msw046 [doi].

40. Guesmi-Mzoughi I, Archidona-Yuste A, Cantalapiedra-Navarrete C, Regaieg H, Horrigue-Raouani N, Palomares-Rius JE, et al. First Report of the Spiral Nematode Rotylenchus incultus (Nematoda: Hoplolaimidae) from Cultivated Olive in Tunisia, with Additional Molecular Data on Rotylenchus eximius. J Nematol. 2016;48: 136-138.

41. Mazza G, Menchetti M, Sluys R, Sola E, Riutort M, Tricarico E, et al. First report of the land planarian Diversibipalium multilineatum (Makino & Shirasawa, 1983) (Platyhelminthes, Tricladida, Continenticola) in Europe. Zootaxa. 2016;4067: 577-580. doi: 10.11646/zootaxa.4067.5.4 [doi].

42. Matassi G. Horizontal gene transfer drives the evolution of Rh50 permeases in prokaryotes. BMC Evol Biol. 2017;17: 2-016-0850-6. doi: 10.1186/s12862-016-0850-6 [doi].

43. Conesa A, Madrigal P, Tarazona S, Gomez-Cabrero D, Cervera A, McPherson A, et al. A survey of best practices for RNA-seq data analysis. Genome Biol. 2016;17: 13-016-0881-8. doi: 10.1186/s13059-016-0881-8 [doi].

44. Stajich JE, Block D, Boulez K, Brenner SE, Chervitz SA, Dagdigian C, et al. The Bioperl toolkit: Perl modules for the life sciences. Genome Res. 2002;12: 1611-1618. doi: 10.1101/gr.361602 [doi].

45. Kim J, Levy E, Ferbrache A, Stepanowsky P, Farcas C, Wang S, et al. MAGI: a Node.js web service for fast microRNA-Seq analysis in a GPU infrastructure. Bioinformatics. 2014;30: 2826-2827. doi: 10.1093/bioinformatics/btu377 [doi].

46. Page M, MacLean D, Schudoma C. blastjs: a BLAST+ wrapper for Node.js. BMC Res Notes. 2016;9: 130-016-1938-1. doi: 10.1186/s13104-016-1938-1 [doi].

47. Skinner ME, Uzilov AV, Stein LD, Mungall CJ, Holmes IH. JBrowse: a next-generation genome browser. Genome Res. 2009;19: 1630-1638. doi: 10.1101/gr.094607.109 [doi].

48. Yang Z. PAML 4: phylogenetic analysis by maximum likelihood. Mol Biol Evol. 2007;24: 1586-1591. doi: msm088 [pii].

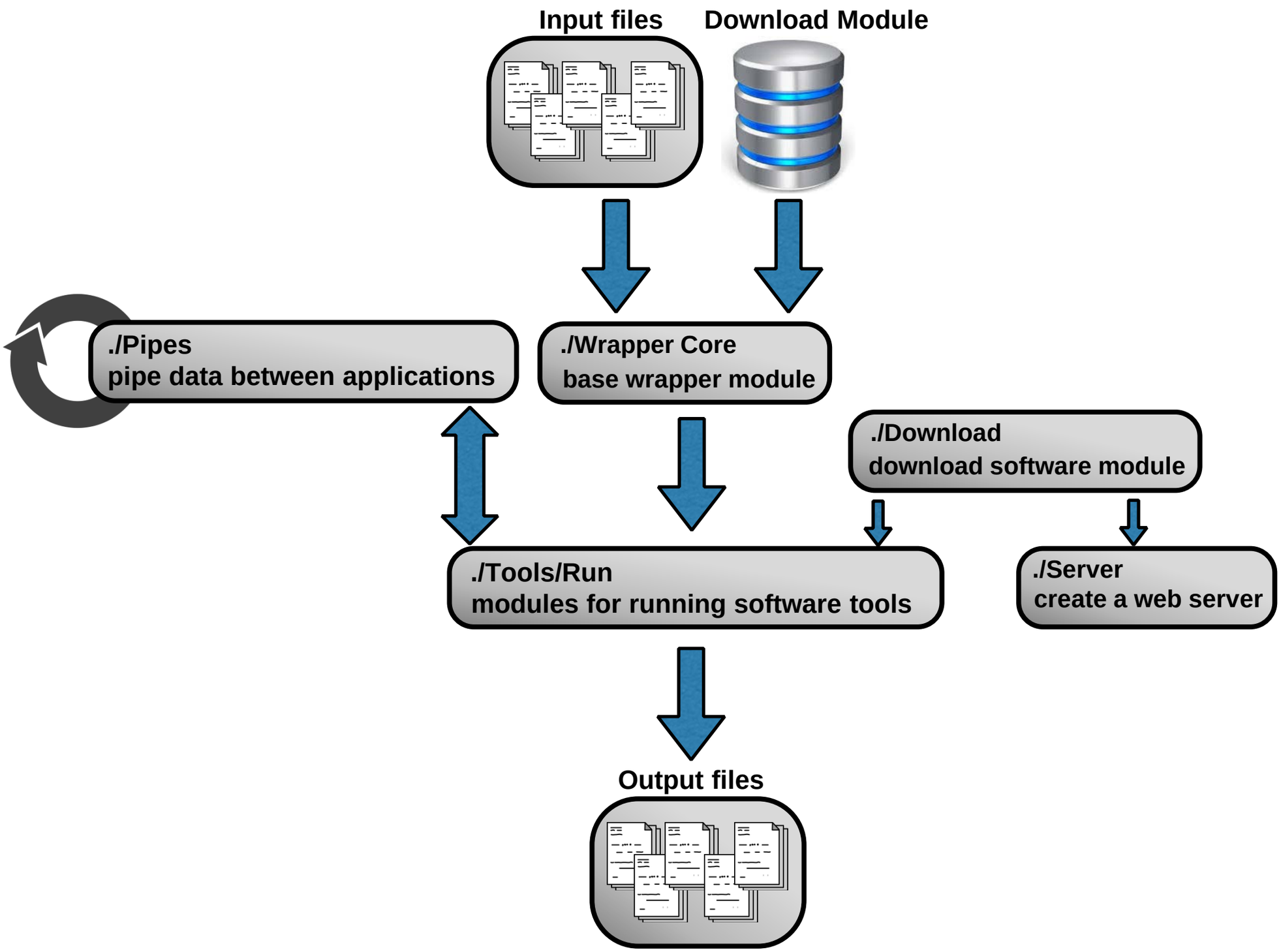
FIGURE LEGENDS

Figure 1. Workflow for *phylo-node*.

phylo-node is organized into a workflow of connected modules and application scripts. In order to interface with a software tool, the base wrapper module is invoked to process command-line requests that are then passed into the software specific module. The input for the specific software can be passed into the base wrapper from a folder specified by the user or by using the sequence retrieval module which is contained within the *Sequence* directory. The *Pipes* directory contains a module for easy piping of data between applications while binaries and executables can be downloaded using the *get_executable* module from within the *Download* folder to deploy software specific modules within the *Run* directory or to provide applications to a web server from within the *Server* directory.

392 **Table 1**. Summary of *phylo-node* applications.

Module	Description	Source	Application	Citation
fetch_seqs	Remotely retrieve sequence data	https://www.ncbi.nlm.nih.gov/books/NBK25501/	ASN.1 and FASTA sequences	-
get_executable	Download binaries and executables	https://www.npmjs.com/package/download	batch, exe, jar etc..	-
http_server	Creates a web server	https://nodejs.org/api/http.html and http://jbrowse.org/jbrowse-1-12-1/	local version of JBrowse	Skinner et al. 2009 [47]
bowtie2	Sequence aligner	http://bowtie-bio.sourceforge.net/bowtie2/index.shtml	align fastq reads onto reference genome	Langmead and Salzberg (2012) [21]
trimmomatic	Read Trimming	http://www.usadellab.org/cms/?page=trimmomatic	preprocessing for reference alignment	Bolger et al. (2014) [20]
phyml	Maximum-Likelihood Phylogenies	http://www.atgc-montpellier.fr/phyml/binaries.php	model testing (ProtTest3 and jModelTest2) and tree building	Guindon et al. (2010) [35]
primer3	Primer design	https://sourceforge.net/projects/primer3/	PCR and sequencing	Untergasser et al. (2012) [23]
muscle	MSA	http://www.drive5.com/muscle/downloads.htm	sequence alignment	Edgar (2004a); Edgar (2004b) [27,28]
clustal_Omega	MSA	http://www.clustal.org/omega/#Download	sequence alignment	Sievers et al. (2011) [25]
kalign	MSA	http://msa.sbc.su.se/cgi-bin/msa.cgi	sequence alignment	Lassmann and Sonnhammer (2005) [26]
pal2nal	Generate codon alignments	http://www.bork.embl.de/pal2nal/	processing of alignments for selection analysis	Suyama et al. (2006) [30]
Slr	Selection analysis	http://www.ebi.ac.uk/goldman-srv/SLR/#download	detect rates of selection in coding DNA	Massingham and Goldman (2005) [31]
codeml	Selection analysis	http://abacus.gene.ucl.ac.uk/software/paml.html	ML analysis of coding DNA using codon substitution models	Yang (1997); Yang (2007) [29,48]
prottest	Model selection	https://github.com/ddarriba/prottest3	best-fit model determination of amino-acid replacement	Darriba et al. (2011) [33]
jmodeltest2	Model selection	https://github.com/ddarriba/jmodeltest2	best-fit model determination of nucleotide substitutions	Darriba et al. (2012) [32]
base_wrap	Base module for program execution	https://nodejs.org/api/child_process.html	handles arguments and spawns child processes	-
phylo-node_pipes	Module for chaining commands	-	used to pipe data between applications	-


Input files
Download Module
./Pipes
pipe data between applications
./Wrapper Core
base wrapper module
./Download
download software module
./Tools/Run
modules for running software tools
./Server
create a web server
Output files