LinguaPhylo: a probabilistic model specification language for reproducible phylogenetic analyses

Alexei J. Drummond[1,2,3]*, Kylie Chen[1,2,3], Fábio K. Mendes[1,4], Dong Xie[1,2,3]

1 Centre for Computational Evolution, University of Auckland, Auckland, New Zealand
2 School of Biological Sciences, University of Auckland, Auckland, New Zealand
3 School of Computer Science, University of Auckland, Auckland, New Zealand
4 Department of Biology, Washington University in St. Louis, St. Louis, United States

* a.drummond@auckland.ac.nz

Abstract

Phylogenetic models have become increasingly complex and phylogenetic data sets larger and richer. Yet inference tools lack a model specification language that succinctly describes a full phylogenetic analysis independently of implementation details. We present a new lightweight and concise model specification language, called 'LPhy', that is both human and machine readable. 'LPhy' is accompanied by a graphical user interface for building models and simulating data using this new language, as well as for creating natural language narratives describing such models. These narratives can form the basis of manuscript method sections. We also introduce a command-line interface for converting LPhy-specified models into analysis specification files (in XML format) to be used alongside the BEAST2 software platform. Together, these tools will clarify the description and reporting of probabilistic models in phylogenetic studies, and improve result reproducibility.

Author summary

We describe a succinct domain-specific language to accurately specify the details of a phylogenetic model for the purposes of reproducibility or reuse. In addition, we have developed a graphical software package that can be used to construct and simulate data from models described in this new language, as well as create natural language narratives that can form the basis of a description of the model for the method section of a manuscript. Finally, we report on a command-line program that can be used to generate input files for the BEAST2 software package based on a model specified in this new language. These tools together should aid in the goal of reproducibility and reuse of probabilistic phylogenetic models.

1 Introduction

Transparency is a scientific ideal, and replicability and reproducibility lie at the heart of the scientific endeavor [1, 2]. Metaresearch efforts have uncovered the so-called "reproducibility crisis" [3] in many scientific domains [3]. In recent years, the growing number of computational biology software packages available has enabled greater choice in data analyses, but at the cost of increased complexity in the data-preparation and

analytical pipelines [4]. This increases the difficulty of accurately reporting and reproducing analyses. These barriers have been recognized by the wider genomics research community [4] as well as within evolutionary biology [5].

In evolutionary biology, phylogenetics has become a highly technical discipline [5]. The most general phylogenetic tools are Bayesian methods (e.g., BEAST, BEAST 2, MrBayes and RevBayes; [6–9]) that can simultaneously reconstruct phylogenetic tree topology and divergence times, as well as estimate the related micro-evolutionary and macro-evolutionary parameters. Phylogenetic analyses often combine multiple models within a complex pipeline to answer questions in evolutionary biology such as species evolution [10–12], ancestral bio-geographical ranges [13, 14], and epidemic dynamics [15, 16].

Reproducing, reusing and interpreting a phylogenetic model is not trivial, and requires an understanding of the input data, details of the model (i.e., its parameters and how they are are related and their priors), and inference methodology. The latter can include complex Markov chain Monte Carlo (MCMC) proposal distributions and sampling algorithms which are not part of the model. Currently, little research has been done on the readability, reproducibility and re-usability of phylogenetic analyses employing phylogenetic models. Our paper presents a tool that aims to: (i) facilitate concise communication of phylogenetic models, (ii) improve reproducibility, and (iii) increase re-usability of phylogenetic models and their variations on new datasets.

Previous attempts to address model specification and analysis setup include BEAST-style XMLs (eXtensible Markup Language) developed for the BEAST software [6, 7] and the Rev programming language for RevBayes [8]. The extensibility of XMLs provides flexibility to developers allowing them to create new descriptive tags for specifying new models. However, BEAST-style XMLs are hard to read due to their verbose syntax and the usual complexity of the models being specified. Unsurprisingly, translating BEAST or BEAST 2 analyses from XMLs into text descriptions is difficult and error prone. Our experience suggests that most users are unable to verify if the XML analysis file matches the description in their manuscript. The Rev language [8] is an alternative to XMLs, and it incorporates conventional notation from more general probabilistic programming languages. This feature makes Rev model specification more recognisable to statistically literate users and sometimes more flexible, but users must still contend with verbose and prosaic implementation details extraneous to the model (such as MCMC sampling settings, logging details and proposal distributions), as well as with the error-prone task of describing the REV model accurately in natural language when describing it in the methods section of a manuscript.

Here, we introduce LinguaPhylo (LPhy, pronounced ‘el-fee’), an open-source model specification language aimed at improving readability, reproducibility, and re-usability of phylogenetic models. The LPhy language has a simple syntax for succinct specification of complex models, and is implemented in a framework that generates precise text descriptions and graphical diagrams of phylogenetic models from user input.

2 Methods

The LPhy language is designed to enable the specification of phylogenetic models using a concise and readable syntax. The reference implementation is built on top of the Java programming language and provides features for: (i) concise formal specification of phylogenetic models on real or synthetic data, (ii) data simulation from phylogenetic models, (iii) integration with the BEAST 2 phylogenetic inference framework, and (iv) an extensibility mechanism for adding new functionality and data types to the LPhy language.

2.1 Language features

The LPhy language provides a simple syntax for specifying and simulating under different evolutionary models models. These include tree models for phylogenies and genealogies (e.g., birth-death, coalescent), substitution models for genomic sequences (e.g., GTR [17]), and parametric distributions for discrete and continuous parameters (e.g., Dirichlet, Normal).

In the context of simulation under a model specified with LPhy, "generative distributions" produce values for random variables, which in turn can be of different "datatypes": trees, continuous morphology, sequence alignments. These random variables constitute stochastic nodes in the probabilistic graphical model (PGM) that LPhy builds, but it is also possible to assign a fixed value to a node, in which case a constant node is created (see more on this below).

The specification of generators and named variables is done through commands the user can type on LPhy's command prompt or execute from a script file:

```
data {
  L = 200;
  taxa = taxa(names=1:10);
}
model {
  Θ ~ LogNormal(meanlog=3.0, sdlog=1.0);
  ψ ~ Coalescent(theta=Θ, taxa=taxa);
  D ~ PhyloCTMC(L=L, Q=jukesCantor(), tree=ψ);
}
```

Listing 1. An LPhy script defining a constant-size coalescent tree prior with log-normally distributed population sizes, a strict clock model, and a Jukes-Cantor model on 10 nucleotide sequences with 200 sites (base pairs).

Listing 1 specifies a complete phylogenetic model using only five lines of code inside two blocks. Constant nodes with fixed values are shown in magenta. The first block specifies "200" nucleotide sites in "L" and ten different taxa named '1' to '10' in "taxa". The second block declares the stochastic nodes or random variables in green, and their generative distributions in blue.

There are three classes of generators: (i) generative distrivutions which produce values for random variables, (ii) deterministic functions, and (iii) method calls. For deterministic functions and method calls, these generators produce deterministic nodes in the PGM. This is illustrated in Figure 1, which shows a graphical representation of the model specified in Listing 1. Deterministic nodes are shown as diamonds (e.g., the "Q" matrix of the Jukes-Cantor model). Stochastic nodes are represented by circles (e.g., Θ, the population size governing the coalescent times generated by the Coalescent process), and constant nodes are represented by squares (e.g., the mean of the log-normal generative distribution underlying Θ).

2.1.1 Variable vectorization

Named variables in LPhy can be scalars or vectors. Any generator can be vectorized to produce a vector of i.i.d. values by using the `replicates` keyword:

```
κ ~ LogNormal(meanlog=0.5, sdlog=1.0, replicates=3);
```

Vectorization can also be applied to a generative distribution that already produces vectors. In which case, the output will be a matrix as in the following example, where the major dimension has size 3, with each element of the π random variable being a vector of base frequencies.

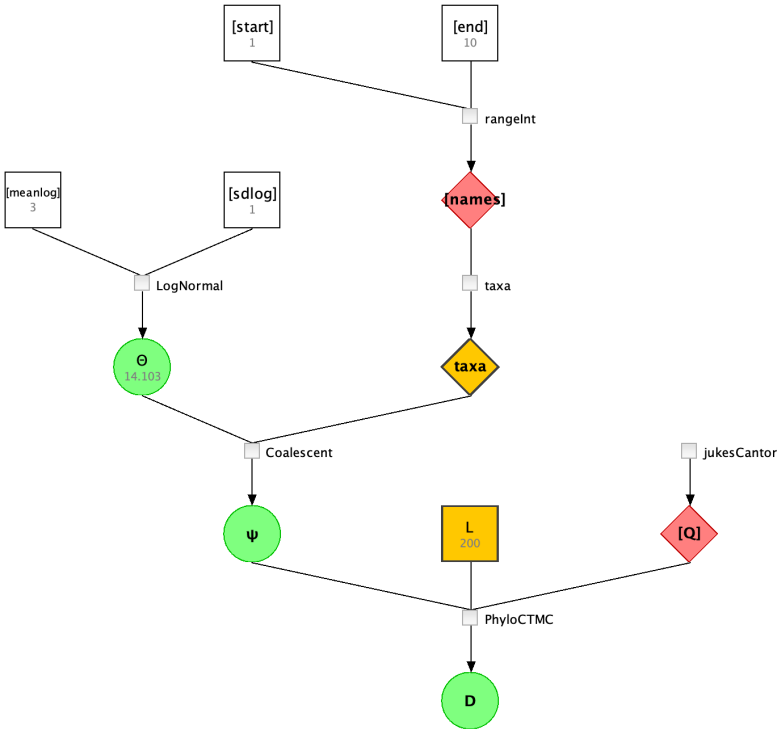

Fig 1. The graphical representation of the probabilistic model defined in Listing 1.

```
π ~ Dirichlet(conc=[2.0, 2.0, 2.0, 2.0], replicates=3);
```

In the example above, κ is a random vector of three log-normally distributed i.i.d. values.

Finally, vectorization can be coerced simply by passing a vector input instead of a scalar input to one or more of the inputs of a generator:

```
Q = hky(kappa=κ, freq=π);
```

Here, since both κ and π are random vectors with the same major dimension length (3), we can assign them as values of the `hky` deterministic function, which in turn outputs a vector of three instantaneous rate matrices, stored in Q.

2.1.2 Parametric distributions

LPhy implements a series of parametric distributions commonly used in evolutionary models, such as Uniform, Normal, Lognormal, Gamma, Exponential, and Dirichlet. Specifying parametric distributions as generative distributions for model parameters can be achieved by:

```
μ ~ LogNormal(meanlog=-5.0, sdlog=1.25);
```

Each parametric distribution is characterized by its own parameters. In the example above, the extinction rate parameter μ is drawn from a Lognormal distribution with mean -5 and standard deviation 1.25 in log space.

2.1.3 Tree models

Tree models are central components in phylogenetic simulation and analysis, and are used to generate phylogenetic trees. Below we briefly expand on some of the main tree models implemented in LPhy.

Coalescent models

Serially sampled coalescent
The simplest coalescent model LPhy implements is the constant-population size coalescent, which can be extended to generate serially sampled (heterochronous) data [18]:

```
ψ ~ Coalescent(theta=Θ, taxa=taxa(names=["a", "b", "c", "d"],
  ages=[0.0, 1.0, 2.0, 3.0]));
```

The script above specifies a serially sampled constant-population size coalescent (a generative distribution) for tree ψ with four taxa, – "a", "b", "c", and "d" – sampled at 0.0, 1.0, 2.0 and 3.0 time points, respectively. Here, sample time is defined as the age of a sample, with 0.0 meaning the present moment.

Structured coalescent
The structured coalescent [19,20] generalizes the constant-population size coalescent [21] by allowing multiple demes, each of which are characterized by a distinct population size (in the simplest case this population size does not change through time). Demes exchange individuals according to migration rates m specified in the off-diagonal elements of a migration matrix M, where the diagonal elements store the population sizes, θ, of each deme. For K demes, the population size parameter "theta" is a K-tuple, and m is a $(K^2 - K)$-tuple.

```
M = migrationMatrix(theta=[0.1, 0.1], m=[1.0, 1.0]);
g ~ StructuredCoalescent(M=M, n=[15, 15]);
```

In the above, `migrationMatrix` is a deterministic function and `StructuredCoalescent` is a generative distribution; both are generators. A stochastic node "g" stores a gene tree sampled from a two-deme structured coalescent process.

Skyline coalescent model
The skyline coalescent model [22] is a coalescent process that models changes in population sizes. This model is characterized having a constant population size for each coalescent interval, with instantaneous changes in population size at some coalescent events.

The following script specifies a Skyline coalescent model with 10 coalescent intervals (hence 11 taxa), governed by four distinct population sizes.

```
g ~ SkylineCoalescent(theta=[0.1, 0.2, 0.3, 0.4], groupSizes=[4,3,2,1]);
```

Here, "g" is a stochastic node in the PGM, with its value sampled from the `SkylineCoalescent` generative distribution. Ten coalescent intervals are defined through the "groupSizes" argument: the first four coalescent intervals will be drawn assuming a "theta" of 0.1, the next three intervals with "theta" equal to 0.2, and so on.

Birth-death models

Birth-death models are commonly used in macroevolution as sampling distributions for species trees. Models that parameterize the fossilization process can be especially

useful, as they allow users to leverage fossil ages as data; when fossil morphological characters have also been scored, total-evidence dating can be carried out [11]. One such tree model is the serially sampled birth-death process [23], whose parameters "psi" and "rho" (see below) represent the rate of sampling extinct and extant lineages, respectively:

```
ages = [0.0, 1.0, 2.0, 3.0, 4.0];
tree ~ BirthDeathSerialSampling(lambda=1, mu=0.5, rho=0.1,
psi=1, rootAge=5, ages=ages);
```

Other tree models include the birth-death [24] and fossilized birth-death processes [25], as well as the Yule process [26]. See the Supplementary Material for more examples.

2.1.4 Substitution models

Substitution models consist of continuous-time Markov chains (CTMC) used to model the evolution of discrete characters, such as nucleotides and amino acid residues. LPhy implements a general formulation of a phylogenetic CTMC, known as the GTR model [17], under which several nested models can be specified. The first line below constructs the instantaneous rate matrix (Q) for an HKY model [27], which is then used to in `PhyloCTMC`, the generative distribution over sequence alignment data (D):

```
Q = hky(kappa=2.0, freq=[0.2, 0.25, 0.3, 0.25]);
Θ ~ LogNormal(meanlog=3.0, sdlog=1.0);
ψ ~ Coalescent(theta=Θ, taxa=taxa);
D ~ PhyloCTMC(L=200, Q=Q, tree=ψ);
```

Other substitution models can be easily specified by assigning different instantaneous transition rate (Q) matrices to `PhyloCTMC`, e.g., the matrix of the Jukes-Cantor model [28]:

```
D ~ PhyloCTMC(L=200, Q=jukesCantor(), tree=ψ);
```

For forward simulation `PhyloCTMC` is used as a generative distribution for a multiple sequence alignment, which is here represented by stochastic node "D". When the model is employed for statistical inference, and data D is known, the `PhyloCTMC` represents the phylogenetic likelihood (see Data clamping below).

2.1.5 Evolutionary clock models

Evolutionary (molecular) clock models are used to model the rate of evolutionary change and whether/how it varies over time. The LPhy language supports strict [29, 30], local [31] and relaxed clock [32] models. Specifying a clock model is done by generating evolutionary rate values, one per phylogenetic tree branch, and then multiplying those rates by the length of the corresponding branch (measured in the chosen units of time) – effectively scaling the tree to units of expected substitutions per site.

The simplest clock model is the strict clock, under which the evolutionary rate remains constant over the entire tree. Specifying a strict molecular clock can be done by specifying the "mu" parameter in the `PhyloCTMC` distribution (default is 1.0):

```
λ ~ LogNormal(meanlog=3.0, sdlog=1.0);
ψ ~ Yule(lambda=λ, n=16);
D ~ PhyloCTMC(L=200, Q=jukesCantor(), tree=ψ, mu=0.5);
```

More realistic clock models like the uncorrelated relaxed clock model [32] assume the rate for each branch is drawn according to a parametric distribution. For example, a relaxed clock with rates drawn from a log-normal distribution can be constructed as follows:

```
λ ~ LogNormal(meanlog=3.0, sdlog=1.0);
ψ ~ Yule(lambda=λ, n=16);
branchRates ~ LogNormal(sdlog=0.5, meanlog=-0.25, replicates=ψ.branchCount());
D ~ PhyloCTMC(L=200, Q=jukesCantor(), branchRates=branchRates, tree=ψ);
```

Here, 30 rates are drawn independently from a log-normal distribution, and then each is assigned to one of the 30 branches of tree ψ.

2.1.6 Inference and data clamping

In addition to simulation, LPhy allows users to use a specified model for inference (at the moment, LPhy interfaces only with BEAST 2, see the "LPhy and BEAST 2" section below). The key step when setting up an inferential analysis with LPhy, after specifying the model, is to carry out "data clamping".

Data clamping should be familiar to users of the Rev language [8], and consists of assigning an observed value to a random variable in the probabilistic model (i.e., to a stochastic node in the PGM). Effectively, by clamping data to a node, a user tells the inference machinery that the value of a random variable is known and will be conditioned on for purposes of inference. In LPhy, data clamping can be achieved using the "data block", for example:

```
data {
   options = {ageDirection="forward", ageRegex="s(\d+)"};
   nexusFilePath = "tutorials/data/RSV2.nex";
   D = readNexus(file=nexusFilePath, options=options);
   codon = D.charset(["3-629\3", "1-629\3", "2-629\3"]);
   n = 3;
   L = [209, 210, 210];
   taxa = D.taxa();
}
model {
   π ~ Dirichlet(replicates=n, conc=[2.0, 2.0, 2.0, 2.0]);
   κ ~ LogNormal(sdlog=0.5, meanlog=1.0, replicates=n);
   r ~ WeightedDirichlet(conc=rep(element=1.0, times=n), weights=L);
   μ ~ LogNormal(meanlog=-5.0, sdlog=1.25);
   Θ ~ LogNormal(meanlog=3.0, sdlog=2.0);
   ψ ~ Coalescent(taxa=taxa, theta=Θ);
   Q = hky(kappa=κ, freq=π, meanRate=r);
   codon ~ PhyloCTMC(L=L, Q=Q, mu=μ, tree=ψ);
}
```

Listing 2. An LPhy script for phylodynamic analysis of a virus dataset containing Respiratory syncytial virus subgroup A (RSVA) genomic samples [33, 34].

In the example above, we used a Respiratory syncytial virus subgroup A (RSVA) dataset [33, 34] containing 129 molecular sequences coding for the G protein collected between years 1956 and 2002. We use three partitions corresponding to the codon position, an HKY substitution model [27], coalescent tree prior [21] and a strict molecular clock with a Lognormal prior on the mean clock rate. Within the `data` block we clamp the value of "codon", a stochastic node that appears below inside the `model`

block. This is achieved by specifying a data node of the same name (codon) in the data block. In this example the data is vectorized into three codon positions to allow different site models for the different codon positions.

2.2 LPhyStudio

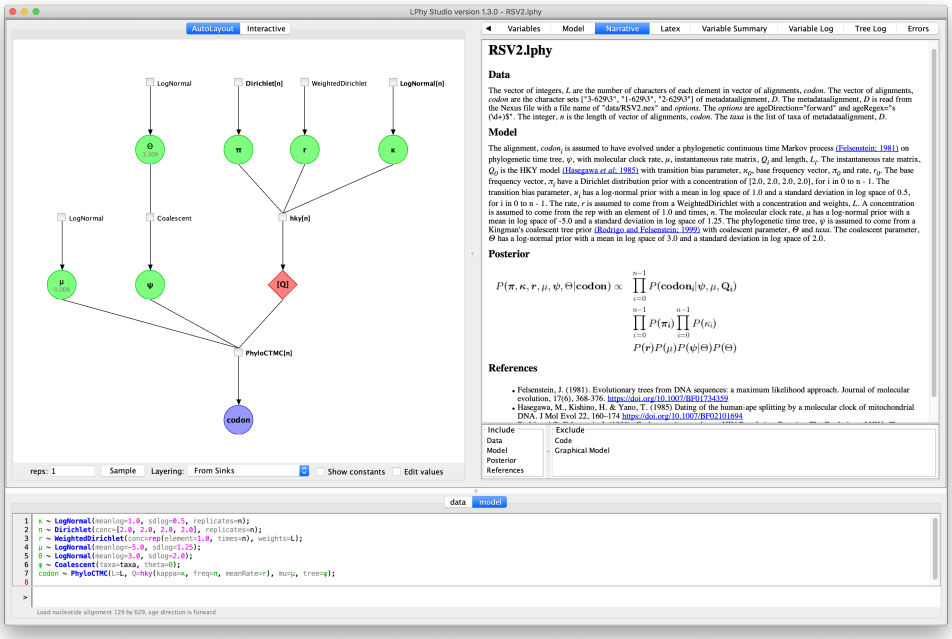

Fig 2. A screenshot of LPhy Studio showing the probabilistic graphical model on the left panel (constants hidden), and the auto-generated text description of the data and phylogenetic model on the right panel.

Along with the language definition, we introduce LPhy Studio, a GUI intended for (i) model specification, (ii) PGM graphical and textual display, and (iii) simulated data visualization. Figure 2 shows a screenshot of LPhyStudio after a simple phylogenetic model was specified. LPhyStudio's additional features include the option to specify models via loading LPhy scripts (rather than building the model line-by-line), and to export PGMs and their descriptions as LaTeX documents.

2.3 LPhy and BEAST2

To facilitate the application of specified models for evolutionary inference, the companion program "LPhyBEAST" was developed as an interface between LPhy and BEAST2. LPhyBEAST is a command-line tool that takes as input an LPhy script file specifying a model and clamping data, and produces a BEAST2 XML file as output.

2.4 A community resource

LPhy, LPhySTudio and LPhyBEAST were developed to support both end-users and model developers. As such, this suite of programs is accompanied by extensive documentation and a growing list of tutorials (available on `https://linguaphylo.github.io/tutorials/`) covering common use cases and

extension mechanisms. LPhy is designed in a modular fashion: researchers interested in implementing new models within the LPhy language can do so by releasing extension modules that can extend the LPhy application post-deployment.

3 Results

We present key features of the LPhy software using the GT16 substitution and error models [35]. Starting from an LPhy script, our software generates a text description of the model, a graphical representation of the model (i.e., a PGM), and multiple displays of the simulated data. Additionally, we showcase how LPhy can be used to validate the correctness of the BEAST 2 implementation of GT16 [36].

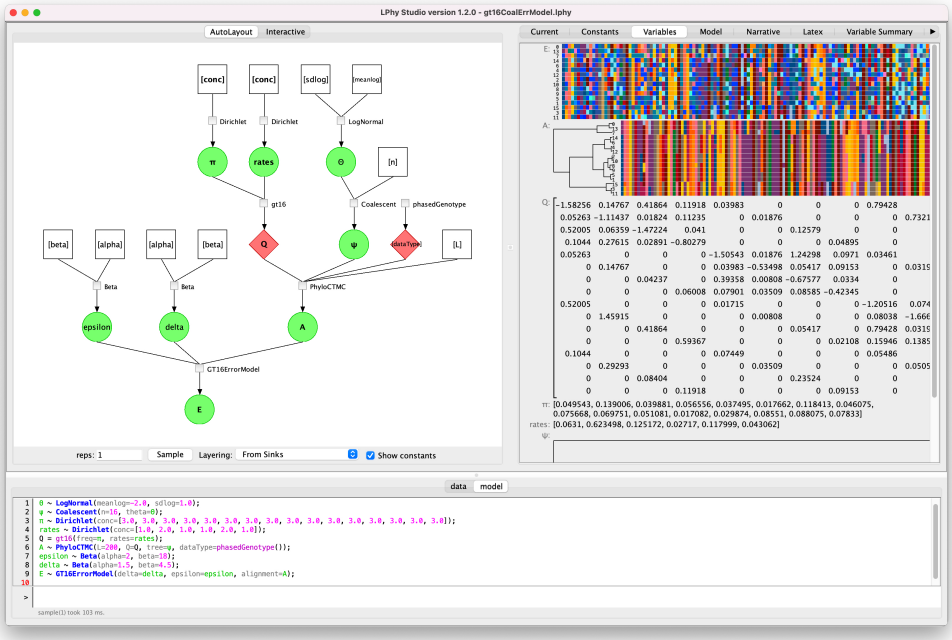

Fig 3. A screenshot of LPhyStudio showing the GT16 substitution and error model [35, 36]. The left panel shows the graphical model representation, the right panel shows the simulated tree and diploid nucleotide genotypes, and the bottom panel shows the LPhy script.

LPhy script

We start from an LPhy script below which specifies a GT16 substitution and error model [35] for single-cell diploid nucleotides with sequencing and allelic dropout error.

```
model {
  π  ~ Dirichlet(conc=[3.0, 3.0, 3.0, 3.0,
                       3.0, 3.0, 3.0, 3.0,
                       3.0, 3.0, 3.0, 3.0,
                       3.0, 3.0, 3.0, 3.0]);
  rates ~ Dirichlet(conc=[1.0, 2.0, 1.0, 1.0, 2.0, 1.0]);
  Q = gt16(rates=rates, freq=π);
  Θ ~ LogNormal(meanlog=-2.0, sdlog=1.0);
  ψ ~ Coalescent(n=16, theta=Θ);
```

```
  A ~ PhyloCTMC(L=200, Q=Q, dataType=phasedGenotype(), tree=ψ);
  delta ~ Beta(alpha=1.5, beta=4.5);
  epsilon ~ Beta(alpha=2, beta=18);
  E ~ GT16ErrorModel(alignment=A, delta=delta, epsilon=epsilon);
}
```

Listing 3. Example of an Lphy script for a GT16 substitution and error model for diploid single-cell nucleotide data.

Natural text description

LPhyStudio automatically generates a text description of the model above:

> The alignment, $\boldsymbol{E}$ is assumed to come from a GT16ErrorModel. The alignment, $\boldsymbol{A}$ is assumed to have evolved under a phylogenetic continuous time Markov process [37] on phylogenetic time tree, $\boldsymbol{\psi}$, with instantaneous rate matrix, $\boldsymbol{Q}$, a length of 200 and a dataType. The instantaneous rate matrix, $\boldsymbol{Q}$ is the general time-reversible rate matrix on phased genotypes [35] with relative rates, **rates** and base frequencies, $\boldsymbol{\pi}$. The base frequencies, $\boldsymbol{\pi}$ have a Dirichlet distribution prior with a concentration of [3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0]. The relative rates, **rates** have a Dirichlet distribution prior with a concentration of [1.0, 2.0, 1.0, 1.0, 2.0, 1.0]. The dataType is the phased genotype data type. The phylogenetic time tree, $\boldsymbol{\psi}$ is assumed to come from a Kingman's coalescent tree prior [21] with coalescent parameter, Θ and an n of 16. The coalescent parameter, Θ has a log-normal prior with a mean in log space of -2.0 and a standard deviation in log space of 1.0. The delta has a Beta distribution prior with an alpha of 1.5 and a beta of 4.5. The epsilon has a Beta distribution prior with an alpha of 2 and a beta of 18.

This natural text narrative can provide a precise starting point for the model description section in a research article.

Model validation

The LPhy framework can be used to verify implementation correctness of new models, in which case they are said to be well-calibrated. Bayesian model validation consists of a series of steps, the first of which is simulation of synthetic data (for recent examples within the BEAST 2 platform, see [36, 38]). By making it possible to simulate under complex models, LPhy greatly simplifies the validation procedure. Figure 4 presents the validation results for the model described above, when model specification and simulation were performed using LPhy and LPhyBEAST [36].

Discussion

Although there are many programming languages through which statistical models can be succintly described (e.g., Stan [39], JAGS [40], BUGS [41, 42]), these languages do not support the unique feature of phylogenetic models: the phylogenetic tree. Phylogenetic trees are complex high-dimensional objects, part discrete, part continuous. There is no bijection between tree space and Euclidean space, so these objects cannot be treated with standard statistical distributions [43]. Hence, specialist software is commonly employed to perform inference involving phylogenetic trees [44, 45].

LinguaPhylo differs from existing specialist software in the way it handles model specification. By using vectorization, LinguaPhylo obviates the need for for-loop control

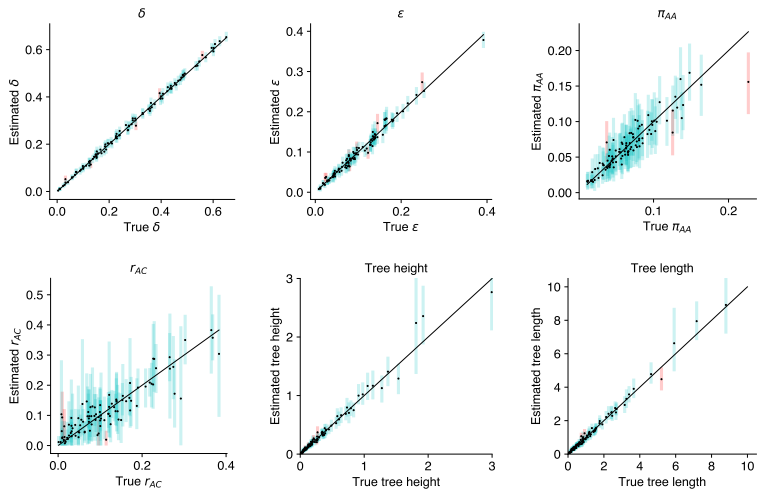

Fig 4. Model validation for the GT16 diploid nucleotide substitution model and GT16 error model. Each plot shows the 95% highest posterior density for model parameters: allelic dropout error δ, sequencing error ϵ, equilibrium frequency for π_{AA}, relative rate r_{AC}, tree height, and tree length.

flow to describe repetitive structural elements of a model. This feature lowers the risk of syntactic or programming logic mistakes when defining a model relative to a full programming language such as Rev [8]. In its declarative nature, LPhy's language resembles the XML specification adopted by BEAST 2 [46], but shares the central notion of probabilistic graphical models with the Rev language.

LinguaPhylo provides for a form of array programming (vectorization), so that any function or generative distribution can be called with its arguments in vectorized form. In such situations the function or generative distribution is "broadcast" over each element of the array (or indeed pairs or tuples of elements across multiple parallel arrays), which allows for very concise model descriptions.

Finally, future work on integration of LPhy with other popular Bayesian phylogenetic inference tools, such as BEAST [47], MrBayes [48], RevBayes [44] will increase the flexibility of the framework, and enable easy validation of and comparison between different Bayesian phylogenetic inference engines.

Acknowledgments

AJD was supported by a James Cook Fellowship. FKM was supported by Marsden grant 16-UOA-277 and by National Science Foundation grant DEB-2040347. We thank the New Zealand eScience Infrastructure (NeSI) for access to high-performance computation resources.

References

1. National Academies of Sciences, Engineering, and Medicine. Reproducibility and replicability in science. Washington, DC: The National Academies Press; 2019.

2. Munafò MR, Nosek BA, Bishop DV, Button KS, Chambers CD, Percie du Sert N, et al. A manifesto for reproducible science. Nature human behaviour. 2017;1(1):1–9.

3. Baker M. Is there a reproducibility crisis? Nature. 2016;533:452–454.

4. Eren AM, Kiefl E, Shaiber A, Veseli I, Miller SE, Schechter MS, et al. Community-led, integrated, reproducible multi-omics with anvi'o. Nature microbiology. 2021;6(1):3–6.

5. Oakley TH, Alexandrou MA, Ngo R, Pankey MS, Churchill CK, Chen W, et al. Osiris: accessible and reproducible phylogenetic and phylogenomic analyses within the Galaxy workflow management system. BMC bioinformatics. 2014;15(1):1–9.

6. Drummond AJ, Rambaut A. BEAST: Bayesian evolutionary analysis by sampling trees. BMC Evolutionary Biology. 2007;7:214.

7. Bouckaert R, Vaughan TG, Barido-Sottani J, Duchêne S, Fourment M, Gavryushkina A, et al. BEAST 2.5: An advanced software platform for Bayesian evolutionary analysis. PLoS computational biology. 2019;15:e1006650.

8. Höhna S, Landis MJ, Heath TA, Boussau B, Lartillot N, Moore BR, et al. RevBayes: Bayesian phylogenetic inference using graphical models and an interactive model-specification languages. Systematic biology. 2016;65:726–736.

9. Ronquist F, Teslenko M, van der Mark P, Ayres DL, Darling A, Höhna S, et al. MrBayes 3.2: efficient Bayesian phylogenetic inference and model selection across a large model space. Systematic biology. 2012;61:539–542.

10. Gavryushkina A, Heath TA, Ksepka DT, Stadler T, Welch D, Drummond AJ. Bayesian total-evidence dating reveals the recent crown radiation of penguins. Systematic biology. 2017;66.

11. Ogilvie HA, Mendes FK, Matzke NJ, Stadler T, Welch D, Drummond AJ. Novel integrative modeling of molecules and morphology across evolutionary timescales. Systematic Biology. 2022;71:208–220.

12. Zhang R, Drummond AJ, Mendes FK. Scalable Bayesian inference of phylogenies from molecular and continuous traits in a probabilistic total-evidence framework. bioRxiv. 2021;.

13. Lemey P, Rambaut A, Welch JJ, Suchard MA. Phylogeography takes a relaxed random walk in continuous space and time. Molecular biology and evolution. 2010;27:1877–1885.

14. Landis MJ, Freyman WA, Baldwin BG. Retracing the Hawaiian silversword radiation despite phylogenetic, biogeographic, and paleogeographic uncertainty. Evolution. 2018;72:2343–2359.

15. Faria NR, et al. Genomics and epidemiology of the P.1 SARS-CoV-2 lineage in Manaus, Brazil. Science. 2021;372:815–821.

16. Douglas J, Mendes FK, Bouckaert R, Xie D, Jiménez-Silva CL, Swanepoel C, et al. Phylodynamics reveals the role of human travel and contact tracing in controlling the first wave of COVID-19 in four island nations. Virus evolution. 2021;7(2):veab052.

17. Tavaré S, et al. Some probabilistic and statistical problems in the analysis of DNA sequences. Lectures on mathematics in the life sciences. 1986;17(2):57–86.

18. Rodrigo AG, Felsenstein J. Coalescent Approaches to HIV Population Genetics. In: K C, editor. The Evolution of HIV. Baltimore: Johns Hopkins Univ. Press; 1999.

19. Hudson R. Oxford Surveys in Evolutionary Biology 7, chapter Gene genealogies and the coalescent process. Oxford. 1990;.

20. Notohara M. The coalescent and the genealogical process in geographically structured population. Journal of mathematical biology. 1990;29(1):59–75.

21. Kingman JFC. The coalescent. Stochastic processes and their applications. 1982;13(3):235–248.

22. Drummond AJ, Rambaut A, Shapiro B, Pybus OG. Bayesian coalescent inference of past population dynamics from molecular sequences. Molecular biology and evolution. 2005;22(5):1185–1192.

23. Stadler T, Yang Z. Dating phylogenies with sequentially sampled tips. Syst Biol. 2013;62(5):674–88. doi:10.1093/sysbio/syt030.

24. Kendall DG. On the generalized" birth-and-death" process. The annals of mathematical statistics. 1948;19(1):1–15.

25. Heath TA, Huelsenbeck JP, Stadler T. The fossilized birth-death process for coherent calibration of divergence-time estimates. Proceedings of the National Academy of Sciences. 2014;111(29):E2957–E2966.

26. Yule GU. II.—A mathematical theory of evolution, based on the conclusions of Dr. JC Willis, FR S. Philosophical transactions of the Royal Society of London Series B, containing papers of a biological character. 1925;213(402-410):21–87.

27. Hasegawa M, Kishino H, Yano Ta. Dating of the human-ape splitting by a molecular clock of mitochondrial DNA. Journal of molecular evolution. 1985;22(2):160–174.

28. Jukes TH, Cantor CR, et al. Evolution of protein molecules. Mammalian protein metabolism. 1969;3:21–132.

29. Zuckerkandl E, Pauling L. Evolutionary divergence and convergence in proteins. In: Evolving genes and proteins. Elsevier; 1965. p. 97–166.

30. Zuckerkandl E, Pauling L. Molecules as documents of evolutionary history. Journal of theoretical biology. 1965;8(2):357–366.

31. Drummond AJ, Suchard MA. Bayesian random local clocks, or one rate to rule them all. BMC biology. 2010;8(1):1–12.

32. Drummond A, Ho S, Phillips M, Rambaut A. Relaxed phylogenetics and dating with confidence. PLOS Biology. 2006;4(e88):699–710. doi:10.1371/journal.pbio.0040088.

33. Zlateva KT, Lemey P, Vandamme AM, Van Ranst M. Molecular evolution and circulation patterns of human respiratory syncytial virus subgroup A: positively selected sites in the attachment G glycoprotein. Journal of virology. 2004;78(9):4675–4683.

34. Zlateva KT, Lemey P, Moës E, Vandamme AM, Van Ranst M. Genetic variability and molecular evolution of the human respiratory syncytial virus subgroup B attachment G protein. Journal of virology. 2005;79(14):9157–9167.

35. Kozlov A, Alves JM, Stamatakis A, Posada D. CellPhy: accurate and fast probabilistic inference of single-cell phylogenies from scDNA-seq data. Genome biology. 2022;23(1):1–30.

36. Chen K, Moravec JC, Gavryushkin A, Welch D, Drummond AJ. Accounting for errors in data improves divergence time estimates in single-cell cancer evolution. Molecular biology and evolution. 2022;in press. doi:10.1093/molbev/msac143.

37. Felsenstein J. Evolutionary trees from DNA sequences: a maximum likelihood approach. J Mol Evol. 1981;17(6):368–76. doi:10.1007/BF01734359.

38. Gaboriau T, Mendes FK, Joly S, Silvestro D, Salamin N. A multi-platform package for the analysis of intra- and interspecific trait evolution. Methods in Ecology and Evolution. 2020;11:1439–1447.

39. Carpenter B, Gelman A, Hoffman MD, Lee D, Goodrich B, Betancourt M, et al. Stan: A probabilistic programming language. Journal of statistical software. 2017;76(1).

40. Plummer M, et al. JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In: Proceedings of the 3rd international workshop on distributed statistical computing. vol. 124. Vienna, Austria; 2003. p. 1–10.

41. Lunn D, Spiegelhalter D, Thomas A, Best N. The BUGS project: Evolution, critique and future directions. Statistics in medicine. 2009;28(25):3049–3067.

42. Gilks WR, Thomas A, Spiegelhalter DJ. A language and program for complex Bayesian modelling. Journal of the Royal Statistical Society: Series D (The Statistician). 1994;43(1):169–177.

43. Gavryushkin A, Drummond AJ. The space of ultrametric phylogenetic trees. Journal of theoretical biology. 2016;403:197–208.

44. Höhna S, Landis MJ, Heath TA, Boussau B, Lartillot N, Moore BR, et al. RevBayes: Bayesian phylogenetic inference using graphical models and an interactive model-specification language. Systematic biology. 2016;65(4):726–736.

45. Bouckaert R, Vaughan TG, Barido-Sottani J, Duchêne S, Fourment M, Gavryushkina A, et al. BEAST 2.5: an advanced software platform for Bayesian evolutionary analysis. PLoS computational biology. 2019;15(4):e1006650. doi:10.1371/journal.pcbi.1006650.

46. Bouckaert R, Heled J, Kühnert D, Vaughan T, Wu CH, Xie D, et al. BEAST 2: a software platform for Bayesian evolutionary analysis. PLoS computational biology. 2014;10(4):e1003537.

47. Suchard MA, Lemey P, Baele G, Ayres DL, Drummond AJ, Rambaut A. Bayesian phylogenetic and phylodynamic data integration using BEAST 1.10. Virus Evol. 2018;4(1):vey016. doi:10.1093/ve/vey016.

48. Ronquist F, Teslenko M, Van Der Mark P, Ayres DL, Darling A, Höhna S, et al. MrBayes 3.2: efficient Bayesian phylogenetic inference and model choice across a large model space. Systematic biology. 2012;61(3):539–542.