Bioinformatics, 34(6), 2018, 1053–1055
doi: 10.1093/bioinformatics/btx701
Advance Access Publication Date: 30 October 2017
Applications Note

Phylogenetics

Efficient comparative phylogenetics on large trees

Stilianos Louca[1,2,*] and Michael Doebeli[1,2,3]

[1]Biodiversity Research Centre, University of British Columbia, Vancouver, BC, V6T1Z4, Canada, [2]Department of Zoology, University of British Columbia, Vancouver, BC, V6T1Z4, Canada and [3]Department of Mathematics, University of British Columbia, Vancouver, BC, V6T1Z4, Canada

*To whom correspondence should be addressed.
Associate Editor: Alfonso Valencia

Received on June 8, 2017; revised on September 27, 2017; editorial decision on October 26, 2017; accepted on October 28, 2017

Abstract

Motivation: Biodiversity databases now comprise hundreds of thousands of sequences and trait records. For example, the Open Tree of Life includes over 1 491 000 metazoan and over 300 000 bacterial taxa. These data provide unique opportunities for analysis of phylogenetic trait distribution and reconstruction of ancestral biodiversity. However, existing tools for comparative phylogenetics scale poorly to such large trees, to the point of being almost unusable.
Results: Here we present a new R package, named 'castor', for comparative phylogenetics on large trees comprising millions of tips. On large trees castor is often 100–1000 times faster than existing tools.
Availability and implementation: The castor source code, compiled binaries, documentation and usage examples are freely available at the Comprehensive R Archive Network (CRAN).
Contact: louca.research@gmail.com
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

The advance of high-throughput sequencing generates whole genome sequences and marker gene-based phylogenies at rapidly increasing rates. For example, the SILVA 16S ribosomal RNA reference tree currently contains ~570 000 bacterial and archaeal tips (Quast *et al.*, 2013). Further, machine learning algorithms enable the automated inference of metabolic phenotypes for thousands of microbial genomes (Karp *et al.*, 2010). These phylogenetic and phenotypic data provide unprecedented opportunities for large-scale evolutionary analysis, such as reconstruction of past metabolic diversity and phenotype predictions for poorly characterized extant clades. However, these vast data also present a serious challenge for existing phylogenetics tools. Most existing phylogenetic packages (e.g. FitzJohn, 2012; Paradis *et al.*, 2004; Revell, 2012) have been designed for much smaller trees containing at most a few thousand tips, and thus scale poorly to large datasets. For example, a simple task such as pruning tips from the SILVA tree can take several hours on a modern laptop using the popular software package `ape` (Paradis *et al.*, 2004). Similarly, ancestral state reconstruction (ASR) for a binary trait with standard continuous-time Markov models (Mk models) takes several hours on the SILVA tree using `ape`. A simple power-law analysis reveals that these functions exhibit a time complexity that scales roughly quadratically with tree size. Re-designed efficient algorithms for large-scale phylogenetic analysis are thus urgently needed. This need is intensified when reconstructions are nested into cross-validation or bootstrapping algorithms. As we explain below, super-linear time complexities can be avoided using redesigned algorithms optimized for large trees.

Here we present a new package for the R statistics environment that enables phylogenetic analysis using algorithms optimized for large trees. We named this package '`castor`', after the animal able to fell large trees. `castor` emerged as part of our work on large microbial phylogenies (including hundreds of thousands of strains), which necessitate more efficient implementations of common functions than currently available.

2 Materials and methods

2.1 `castor`: a collection of highly optimized algorithms

`castor` provides efficient implementations of common phylogenetic functions, focusing on analysis of trait evolution on fixed trees.

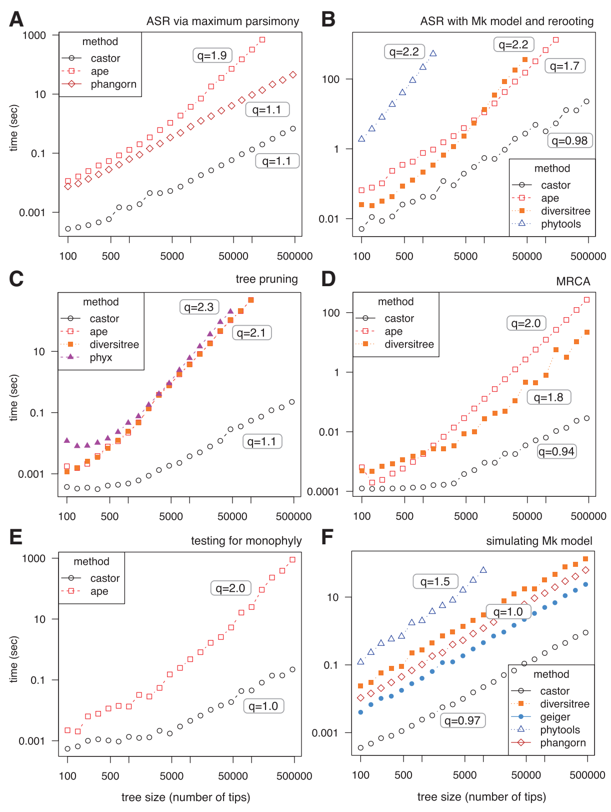

Fig. 1. Comparison of computation time needed for various tree operations in castor and other packages (time T over tree size S). (**A**) ASR of a discrete trait with 5 states, using maximum-parsimony. (**B**) ASR of a discrete trait with 5 states, using an 'equal rates' Mk model with rerooting. (**C**) Pruning trees by removing half of the tips. (**D**) Determining the most recent common ancestor (MRCA) of two random tips. (**E**) Testing whether a subset of tips is monophyletic. (**F**) Simulating an Mk model (5 states) for discrete trait evolution. Note the logarithmic axes in all figures. Package names are listed in the legends. Fitted power-law exponents ($T \propto S^q$) are shown next to every curve. Compared packages include phyx (Brown *et al.*, 2017), ape (Paradis *et al.*, 2004), diversitree (FitzJohn, 2012), phytools (Revell, 2012), phangorn (Schliep, 2011) and geiger (Harmon *et al.*, 2008). Detailed functions and options used are explained in Supplementary Material S2. For additional benchmarks of other functions see Supplementary Figure S1

Notably, castor provides functions for ASR of discrete traits, for example using Mk models (Yang *et al.*, 1995) or maximum-parsimony methods, as well as ASR of continuous traits, for example using squared-change parsimony (Maddison, 1991). Further, castor includes functions for hidden state prediction, i.e. for estimating a priori unknown trait values (states) in tips based on a subset of tips with known states (Zaneveld and Thurber, 2014). castor also enables statistical analysis of trait distribution, such as calculating phylogenetic autocorrelation, for simulating or fitting models of trait evolution, as well as for common operations such as tree pruning, inference of most recent common ancestors or calculating distances between tips. castor fully supports monofurcating and multifurcating trees, in contrast to the majority of existing tools that generally require bifurcating trees.

2.2 Comparison to existing software

Most of castor's functions exhibit a time complexity that scales linearly with tree size, in many cases achieving a 100- to 1000-fold efficiency when compared to existing packages (Fig. 1 and Supplementary Fig. S1). For example, removing 50% of the tips from the SILVA tree takes less than 1 s on a modern laptop using castor and over 4 h using the package ape (Fig. 1C). Similarly, maximum-likelihood ASR of a discrete trait on the SILVA tree using an Mk model (5 possible states, all rates equal) takes about 30 s using castor and over 4 h using ape (Fig. 1B). castor's high efficiency is achieved in multiple ways. First, dynamic programming algorithms are used wherever possible. Second, most algorithms benefit from auxiliary data structures that are temporarily created on demand. For example, calculation of most recent common ancestors is achieved in linear time (Fig. 1D) by using a lookup table that maps each node to its parent node, instead of repeatedly searching for each node's parent amongst all possible nodes. Third, in certain ASR algorithms involving rerooting (e.g. maximum-likelihood Mk models) redundant calculations are avoided by storing previously computed intermediate quantities (see Supplementary Material S1). Fourth, ASR of discrete traits using Mk models, which requires repeated exponentiation of the Markov transition matrix along each edge, was accelerated through an ad-hoc exponentiation algorithm that becomes highly efficient when the same matrix is exponentiated several times. Fifth, castor is almost entirely implemented in C++, a programming language optimized for high-performance computations.

3 Conclusion

castor is a collection of highly efficient algorithms for phylogenetic analysis on large trees, easily scaling to millions of tips. Although castor focuses on analyzing trait distributions and reconstructing trait evolution, it also includes several other common functions for working with phylogenies. On large trees, castor performs many of these functions orders of magnitude faster than other comparable packages, thereby enabling large-scale phylogenetics using substantially reduced computational resources.

Acknowledgements

We thank Laura W. Parfrey and Matthew Pennell for helpful discussions and for providing comments on our manuscript.

Funding

S.L. and M.D. were supported by the Natural Sciences and Engineering Research Council of Canada (NSERC). SL was also funded by the Biodiversity Research Centre, UBC. The authors declare no conflict of interest.

Conflict of Interest: none declared.

References

Brown,J.W. *et al.* (2017) Phyx: phylogenetic tools for unix. *Bioinformatics*, **33**, 1886–1888.
FitzJohn,R.G. (2012) Diversitree: comparative phylogenetic analyses of diversification in r. *Methods Ecol. Evol.*, **3**, 1084–1092.
Harmon,L.J. *et al.* (2008) GEIGER: investigating evolutionary radiations. *Bioinformatics*, **24**, 129.
Karp,P.D. *et al.* (2010) Pathway tools version 13.0: integrated software for pathway/genome informatics and systems biology. *Brief. Bioinform.*, **11**, 40–79.
Maddison,W.P. (1991) Squared-change parsimony reconstructions of ancestral states for continuous-valued characters on a phylogenetic tree. *Syst. Biol.*, **40**, 304–314.
Paradis,E. *et al.* (2004) APE: analyses of phylogenetics and evolution in R language. *Bioinformatics*, **20**, 289–290.

Quast,C. *et al*. (2013) The SILVA ribosomal RNA gene database project: improved data processing and web-based tools. *Nucleic Acids Res*., **41**, D590–D596.

Revell,L.J. (2012) phytools: an r package for phylogenetic comparative biology (and other things). *Methods Ecol. Evol*., **3**, 217–223.

Schliep,K.P. (2011) phangorn: phylogenetic analysis in r. *Bioinformatics*, **27**, 592–593.

Yang,Z. *et al*. (1995) A new method of inference of ancestral nucleotide and amino acid sequences. *Genetics*, **141**, 1641–1650.

Zaneveld,J.R.R. and Thurber,R.L.V. (2014) Hidden state prediction: a modification of classic ancestral state reconstruction algorithms helps unravel complex symbioses. *Front. Microbiol*., **5**, 431.

Efficient comparative phylogenetics on large trees - Supplementary Information -

Stilianos Louca[1,2] & Michael Doebeli[1,2,3]

[1]*Biodiversity Research Centre, University of British Columbia, Canada*
[2]*Department of Zoology, University of British Columbia, Canada*
[3]*Department of Mathematics, University of British Columbia, Canada*

S.1 A simple modification of rerooting

A large class of ASR algorithms are based on a postorder tree traversal (from tips to root) that successively calculates local quantities of each node based on its children, eventually yielding some sort of state estimate or state probabilities for the tree's root. For example, the squared-changes parsimony reconstruction algorithm by Maddison (1991) recursively calculates locally parsimonious states for each node based on its descending subtree, eventually yielding a globally parsimonious state estimate for the tree's root. Similarly, ML reconstructions of discrete traits using Mk models commonly deploy a postorder traversal algorithm that calculates conditional likelihoods for each node based on the conditional likelihoods of its children, eventually yielding the marginal ancestral state probabilities for the root (Yang *et al.*, 1995). For all other nodes but the root, these local quantities only account for information in their descending subtrees and are thus unsuitable as ancestral state estimates by common expectations of parsimony or maximum likelihood (Garland and Ives, 2000, Maddison, 1991). However, by rerooting the tree at any particular node of interest and repeating the postorder computation, one can obtain a global reconstruction for that node. Because of their versatility and intuitive nature, such rerooting methods are widely used for ancestral state reconstruction (Felsenstein, 2004, Garland *et al.*, 1999, Garland and Ives, 2000, Maddison, 1991, Swofford and Maddison, 1987, Yang *et al.*, 1995).

Common implementations of rerooting algorithms in R packages repeat the full postorder traversal with each rerooting, resulting in computational complexities that are at least quadratic in the size of the tree (Revell, 2012). While this practice may have been adequate for typical phylogenies in the past, it becomes extremely expensive for current larger (e.g., bacterial) phylogenies. Here we have deployed a modified approach to rerooting that is mathematically equivalent to conventional rerooting, but avoids redundant computations. Our approach is based on the fact that whenever the root changes, the postorder computation only differs for a subset of edges and nodes, namely the ones connecting the old and the new root. Hence, if all intermediate results of the first postorder traversal are stored for each node, then at each rerooting these results need only be updated for the affected nodes. Rerooting itself merely involves changing the direction *in situ* of those edges connecting the old and the new root. The number of edges and nodes affected by each rerooting is minimized by swapping roots in a depth-first-search manner, such that the root only moves between adjacent or nearly adjacent nodes each time. This depth-first-search rerooting approach is intuitive and straightforward to implement for any postorder traversal algorithm originally nested into a rerooting loop; one simply needs to apply the same calculations as in the postorder portion to a few nodes after rerooting.

The algorithm's time complexity is asymptotically linear in the number of nodes, since each edge leading to a node is traversed at most twice during depth-first-search. The algorithm is thus similarly efficient to postorder-preorder traversal algorithms previously suggested as alternatives to rerooting (Goolsby, 2017, Maddison, 1991). For example, our implementation of weighted-squared-changes parsimony (Maddison, 1991) is up to 50% more efficient than the recently published (mathematically equivalent) Brownian-motion-based algorithm by Goolsby (2017), the current record holder in speed (Supplementary Fig. S1B).

S.2 Benchmarks of computation time

The following software packages were included in the benchmarks: `phyx` v0.99 (Brown *et al.*, 2017), `ape` v4.1 (Paradis *et al.*, 2004), `diversitree` v0.9.10 (FitzJohn, 2012), `phytools` v0.6.0 (Revell, 2012), `Rphylopars` v0.2.9 (Goolsby, 2017, Goolsby *et al.*, 2016), `phangorn` v2.2.0 (Schliep, 2011), `geiger` v2.0.6 (Harmon *et al.*, 2008) and `mvMORPH` v1.0.8 (Clavel *et al.*, 2015). We considered trees of various sizes (100 to $\sim$500,000 tips), generated by removing random subsets of tips from the SILVA 16S reference tree. Because some of `diversitree`'s functions only work for ultrametric trees, the original SILVA tree was first made ultrametric using the program `pathd8` (Britton *et al.*, 2007). To meet additional requirements by some packages, trees were further modified as follows: To reduce numerical rounding errors in some packages, edge lengths in each tree were rescaled uniformly such that the average edge length became equal to 1. Next, because some packages failed in the presence of very small edge lengths, edge lengths below 10^{-6} were replaced with 10^{-6}. Finally, terminal edges were extended as necessary to ensure the resulting trees were still ultrametric.

For each tree, each benchmarked task was repeated 3 times, and the required time was averaged across trials. All benchmarks, described in detail below, were performed on a MacBook Pro laptop (2.9 GHz Intel Core i5) using a single core. The full R script for running the benchmarks is included as Supplementary Material.

To benchmark ASR of discrete traits using either Mk models or maximum-parsimony, the evolution of a discrete trait with 5 possible states was simulated beforehand on each tree using the `castor` function `simulate_mk_model`, based on a randomly generated Markov transition matrix (equal off-diagonal rates). Maximum-likelihood ASR with an equal-rates ("ER") Mk model was then performed using the `castor` function `asr_mk_model`, the `ape` function `ace`, the `phytools` function `rerootingMethod`, and the `diversitree` functions `make.mkn`, `find.mle` and `ancestral.pml`. The popular tool `geiger` (Harmon *et al.*, 2008) was not included in the comparison, because it only fits Mk models but does not perform ancestral state reconstruction. We note that ASR using Mk models involve iterative fitting of the transition rate matrix via optimization of the likelihood function, and the accuracy and duration of this optimization depends on the particular optimization algorithm and control options (Eliason, 1993). For example, the risk of reaching a local non-global optimum can be strongly reduced if fitting is repeated multiple times, each time starting the optimization from a different random choice of rates (option `Ntrials` in `castor::asr_mk_model`). Additional optimization control parameters are explained in the `castor` manual. For all tested packages we used the package's default optimization options.

Maximum-parsimony ASR (Sankoff, 1975) was performed using the `castor` function `asr_maximum_parsimony`, the `ape` function `MPR`, and the `phangorn` function `pace`.

To benchmark ASR of continuous traits, the evolution of a continuous trait was simulated beforehand on each tree using the `castor` function `simulate_bm_model`, based on a Brownian motion model with a diffusion coefficient of 0.1. Maximum-likelihood ASR under a Brownian motion model (Goolsby, 2017) was then performed using the `Rphylopars` function `anc.recon`. ASR via weighted-squared-change parsimony was performed using the `castor` function `asr_squared_change_parsimony`. ASR via phylogenetically independent contrasts (Felsenstein, 1985) was performed using the `castor` function asr_independent_contrasts and the ape function ace.

To benchmark tree pruning, a random subset of tips (half of total) was chosen for removal, and then pruned using the castor function get_subtree_with_tips, the ape function drop.tip, the diversitree function drop.tip.fixed and the phyx program pxrmt. To benchmark the calculation of most recent common ancestors (MRCAs), two tips were chosen randomly from each tree, and their MRCA determined using the castor function get_mrca_of_set, the ape function getMRCA and the diversitree function mrca.tipset.

To benchmark the simulation of an Mk model, an all-rates-different 5×5 transition matrix was created with transition rates chosen randomly between 0 and 10. The associated model was then simulated using the castor function simulate_mk_model, the diversitree function sim.character, the geiger function sim.char, the phytools function sim.history and the phangorn function simSeq. Note that because sim.character simulates individual transition events along edges (instead of just transitions between nodes), its efficiency depends strongly on the transition rates and the edge lengths, with higher transition rates and longer edges leading to decreased efficiency. In contrast, castor only models transitions between adjacent nodes according to the exponentiated transition matrix, hence retaining a similar efficiency regardless of the specific edge lengths and transition rates. Also note that phytools::sim.history simulates complete character histories on the tree, not just the states at tips and nodes, and this inevitably increases computation time that is not accounted for in our comparisons.

To benchmark the simulation of a scalar Brownian motion model (one continuous trait, with diffusion coefficient of 0.1), we used the castor function simulate_bm_model, the diversitree function sim.character, the phytools function fastBM and the geiger function sim.char.

To benchmark the simulation of a multivariate Brownian motion model (3 continuous traits), we created a random diffusivity matrix drawn from a Wishart distribution (3 free parameters, scale parameter $V = 0.01$). We then simulated the Brownian motion (starting from a fixed root state) using the castor function simulate_bm_model, the phytools function sim.corrs and the mvMORPH function mvSIM.

To benchmark the fitting of a multivariate Brownian motion model to trait data (3 continuous traits), we used the same approach as described above to simulate artificial Brownian-motion-distributed correlated trait data for all tips on the tree. We then fitted a 3-dimensional Brownian motion model using the castor function fit_bm_model, the phytools function evol.vcv and the mvMORPH function mvBM (model “BM1”).

To benchmark monophyly checking, we first picked a random node in the tree and then picked a random subset of tips descending from that node. The size of that subset was chosen randomly and uniformly among all possible sizes greater than 1. We then checked whether the picked tip subset was monophyletic using the castor function is_monophyletic and the ape function is.monophyletic.

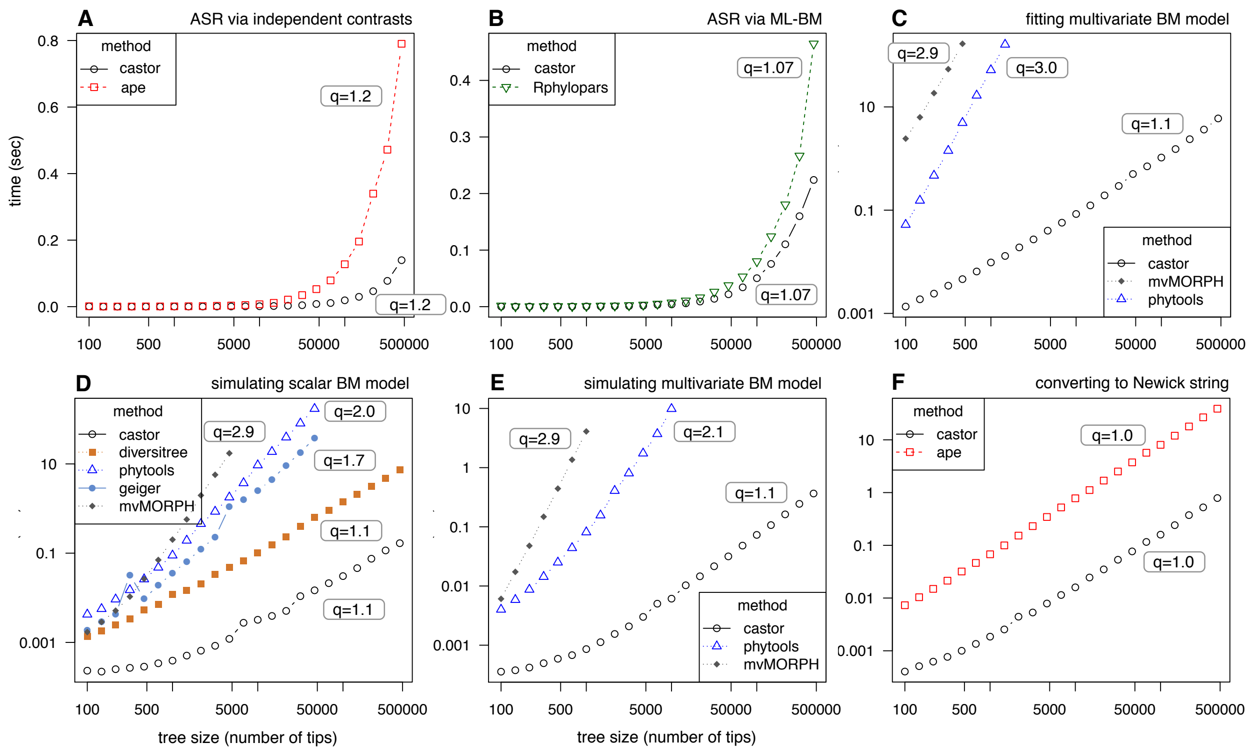

Figure S1: **Additional benchmarks of computation times.** (A,B): Comparison of computation time needed for ancestral state reconstruction (time T over tree size S) using (A) phylogenetic independent contrasts (Felsenstein, 2004) an (B) maximum-likelihood under a Brownian motion model of trait evolution (Goolsby, 2017), in `castor` and other phylogenetic packages. In (B), `castor`'s weighted-squared-changes parsimony reconstruction was used to obtain the same estimates as the ML-BM implementation in `Rphylopars`. (C) Fitting a multivariate Brownian motion model for the co-evolution of 3 traits. (D) Simulating a Brownian motion model for scalar trait evolution on a tree. (E) Simulating a multivariate Brownian motion model for the co-evolution of 3 traits. (F) Converting a tree to a string in Newick format. Package names are listed in the legends; detailed functions and options used are explained in the Methods. Fitted power-law exponents ($T \propto S^q$) are shown next to every curve.

References

Britton, T., Anderson, C. L., Jacquet, D., Lundqvist, S., and Bremer, K. (2007). Estimating divergence times in large phylogenetic trees. *Syst. Biol.*, **56**, 741–752.

Brown, J. W., Walker, J. F., and Smith, S. A. (2017). Phyx: phylogenetic tools for unix. *Bioinformatics*, pages 1–3.

Clavel, J., Escarguel, G., and Merceron, G. (2015). mvmorph: an r package for fitting multivariate evolutionary models to morphometric data. *Methods Ecol. Evol.*, **6**(11), 1311–1319.

Eliason, S. R. (1993). *Maximum Likelihood Estimation: Logic and Practice*. SAGE Publications, Newbury Park, CA.

Felsenstein, J. (1985). Phylogenies and the comparative method. *Am. Nat.*, **125**(1), 1–15.

Felsenstein, J. (2004). *Inferring Phylogenies*. Sinauer Associates.

FitzJohn, R. G. (2012). Diversitree: comparative phylogenetic analyses of diversification in r. *Methods Ecol. Evol.*, **3**(6), 1084–1092.

Garland, T., Midford, P. E., and Ives, A. R. (1999). An introduction to phylogenetically based statistical methods, with a new method for confidence intervals on ancestral values. *Am. Zool.*, **39**(2), 374–388.

Garland, T. J. and Ives, A. R. (2000). Using the past to predict the present: Confidence intervals for regression equations in phylogenetic comparative methods. *Am. Nat.*, **155**(3), 346–364.

Goolsby, E. W. (2017). Rapid maximum likelihood ancestral state reconstruction of continuous characters: A rerooting-free algorithm. *Ecology and Evolution*, **7**, 2791–2797.

Goolsby, E. W., Bruggeman, J., and Ané, C. (2016). Rphylopars: fast multivariate phylogenetic comparative methods for missing data and within-species variation. *Methods Ecol. Evol.*

Harmon, L. J., Weir, J. T., Brock, C. D., Glor, R. E., and Challenger, W. (2008). GEIGER: investigating evolutionary radiations. *Bioinformatics*, **24**(1), 129.

Maddison, W. P. (1991). Squared-change parsimony reconstructions of ancestral states for continuous-valued characters on a phylogenetic tree. *Syst. Biol.*, **40**(3), 304–314.

Paradis, E., Claude, J., and Strimmer, K. (2004). APE: analyses of phylogenetics and evolution in R language. *Bioinformatics*, **20**(2), 289–290.

Revell, L. J. (2012). phytools: an r package for phylogenetic comparative biology (and other things). *Methods Ecol. Evol.*, **3**(2), 217–223.

Sankoff, D. (1975). Minimal mutation trees of sequences. *SIAM J. Appl. Math.*, **28**(1), 35–42.

Schliep, K. P. (2011). phangorn: phylogenetic analysis in r. *Bioinformatics*, **27**(4), 592–593.

Swofford, D. L. and Maddison, W. P. (1987). Reconstructing ancestral character states under wagner parsimony. *Math. Biosci.*, **87**(2), 199–229.

Yang, Z., Kumar, S., and Nei, M. (1995). A new method of inference of ancestral nucleotide and amino acid sequences. *Genetics*, **141**(4), 1641–1650.