## Phylogenetic analyses

Analyses have been performed on single copy gene trees using ASTRAL to account for among gene tree variation due to ILS. In addition, supermatrix analyses have been performed on concatenated gene alignments.  All scripts used to perform analyses are available online at <https://github.com/smirarab/1kpscripts>.

### Multiple sequence alignment and data filtering

Initial alignments: We build an MSA based on Amino Acid (FAA) sequences of each gene, and then impose these alignments on DNA sequences. To do so, we first divide sequences in each gene into two subsets, full-length and fragmentary/chimeric sequences, and then we use PASTA1 to align full-length sequences and use UPP2 to add fragmentary sequences to the full-length alignment. We designate as fragmentary/chimeric any sequence that is 66% shorter or longer than the median length of *genome* sequences in that gene. Once UPP alignments are obtained, we remove from them all unaligned (i.e., insertion) sites. We then back-translate FAA alignments to DNA (FAA2FNA) and filter the third codon position (accounting for removed insertion sites). To reduce the running time burden of dealing with very gappy sites that include minimal phylogenetic information in standard maximum likelihood analyses, we then masked all sites from the alignment that were made of at least 90% gaps. Finally, since the inclusion of fragmentary data in gene tree estimation can be problematic, we removed any sequence that had a gap for at least 67% of the sites in the site-filtered alignment. These produced our initial alignments.

Long branch filtering: In addition to filtering gappy sites and fragmentary sequences, we identified and removed sequences that would be placed on extremely long branches on their respective gene trees. To identify these, we used the initial alignments to build gene trees (see below). We then rooted each gene tree by finding the split that has the highest number of Red and Chromosita Algae exclusively on one of its two sides. If these algae were entirely missing from the gene, we used Glaucophyta, Prasinococcales, prasinophytes, Volvox carteri, Chlamydomonas reinhardtii, or Klebsormidium flaccidum, respectively. We then removed any taxa that had a root to tip distance that was 4 standard deviations longer than the median root to tip distance in each gene tree. Once these taxa on long branches were removed, alignments were re-estimated using the same approach described above, and new gene trees were estimated.

### Gene tree estimation:

To estimate gene trees, we used RAxML, version 8.1.17, starting from one starting tree for building initial trees used for long branch filtering and 10 different starting trees for final gene trees. We also performed 100 replicates of bootstrapping. For DNA analyses, the GTR substitution model and the GAMMA rates across the sites model were used. For Amino Acid genes, we used a perl script (available at ???) adopted from the RAxML website to search among 16 different substitution models on a fixed starting tree per gene, and chose the model with the highest likelihood (JTT, JTTF, or JTTDCMUT were selected for 349 out of 410 genes). For Amino Acid gene trees, we also used the GAMMA model of rate heterogeneity.

### Species tree estimation

We used ASTRAL-II3, version 4.7.8 to estimate the species tree, based on 384 genes that each at least included half of the species. We used multi-locus bootstrapping4,5 and ASTRAL’s built-in local posterior probabilities6 to estimate branch support, drawn on species trees estimated based on best ML gene trees found. We also used ASTRAL’s built-in functionality to compute the percentage of gene trees that agreed with each branch in the species tree, by finding the average number of gene tree quartets defined around the branch (choosing one taxon from each side) that were congruent with the species tree. Computation of local posterior probabilities and quartet support was done using version 4.10.3.

Binning: We also used weighted statistical binning7,8 with a threshold of 75% to see if the effects of gene tree estimation error could be reduced. Binning left the majority of genes in singleton bins and had minimal impact on the overall species tree.

### Supermatrix analyses:

All supermatrix analyses are based upon the filtered amino acid and 1st and 2nd codon alignments generated by the gene alignment and tree pipeline. The (1) **unfiltered** **supermatrices** use the gene alignments as is, the (2) **Eudicot supermatrices** retain only Eudicots species in the supermatrix, and the (3) **rogue-taxa-removed** supermatrices removed the eight rogue taxa (Dillenia indica, Tetrastigma obtectum,Tetrastigma voinierianum,Vitis vinifera, Cissus quadrangularis, Spirotaenia sp., Ceratophyllum demersum, Prasinococcus capsulatus) that tended to move in our initial analyses.

ML supermatrix analyses were performed using ExaML v. 3.0.149. Similar to the gene tree analyses, the GAMMA model of rate heterogeneity across sites were used for all the ML supermatrix analyses. To better handle the rate heterogeneity across genes, we divided the supermatrix into partitions. For the amino acid alignments, the protein model selected for each gene family in the gene tree estimation process was used to group genes into partitions, creating one partition per substitution model. In the nucleotide alignments, we estimated the GTR rate parameters and the alpha shape parameter for each codon position (1st and 2nd positions) of each codon alignment using RAxML10 v 8.1.21. We then projected the GTR rate and alpha parameters of each codon position of each gene into a 2-D plane using principle component analysis11 (PCA). We performed K-means clustering12 to group the codon positions into partitions, selecting the *k=?, which* accounts for 80% of the variation. The clustering analyses of the nucleotide data were performed using R13.

ExaML takes as inputs a reformatted version of the supermatrix and an initial starting tree. We reformatted the supermatrix alignment using parse-examl program provided by ExaML. In order to examine the impact of the starting tree on the ML score of the final tree, we performed preliminary analyses on an earlier version of our supermatrices. We generated nine different Maximum Parsimony (MP) starting trees using RAxML v 8.1.21 and one ML starting tree using FastTree-214 v 2.1.5. We then ran ExaML on each of the starting trees, noting the final ML score. We found that in all cases, the ExaML ML tree using the FastTree-2 ML starting tree had a better ML score than any of the ExaML ML trees using RAxML MP starting trees. Thus, for all of the supermatrix analyses, we used FastTree-2 to generate our initial starting tree. Support was inferred for the branches of the final tree from 100 bootstrap replicates.

**Commands Used:**

All scripts mentioned below are available at XXX.

**Alignment and filtering:**

To build the combined PASTA and UPP alignments, we used the following script:

runupp.sh <gene\_id> <num\_cpus> <genes directory>

We then used scripts mask-gt.sh to mask gappy sites and sequences from alignments and used remove-long-branch.sh to mask long branches. The last script uses Dendropy15 internally.

**Gene tree estimation:**

We used the following scripts to estimate gene trees:

runraxml-rapidboot.sh upp-masked.fasta.mask10sites.mask33taxa FAA 4 [gene id] main 10 100 mp slow

runraxml-rapidboot.sh upp-masked.fasta.mask10sites.mask33taxa FNA2AA 4 [gene id] main 10 100 mp slow

These scripts internally run the following RAxML commands:

raxmlHPC-PTHREADS -m <GTRGAMMA|PROTGAMMA…> -T 4 -n best -s <alignment\_file> -p <random number> -N 10

raxmlHPC-PTHREADS -m <GTRGAMMA|PROTGAMMA…> -T 4 -n ml -s <alignment\_file> -p <random number> -N 100 -b <random number>

**Species tree estimation:**

To estimate species trees, we use:

java -jar astral.4.7.8.jar -i <gene trees> -o <output> -m 589

To score a species tree to find quartet scores per branch, we use:

java -jar astral.4.10.3.jar -i <gene trees> -q <tree to score> -t 8 |tee <output file>

To score a species tree to find posterior probabilities per branch, we use:

java -jar astral.4.10.3.jar -i <gene trees> -q <tree to score> -t 4|tee <output file>

To test for a polytomy, we use

java –jar astral.4.10.3.jar -i <gene trees> -q <tree to score> -t 10 |tee <output file>

**concatenation trees**:

alltaxa

FAA-upp-masked.fasta.mask10sites.mask33taxa.bootstrap.tre

FNA2AA-upp-masked-c12.fasta.mask10sites.mask33taxa.bootstrap.tree

eudicots

eudicot.FAA-upp-masked.fasta.mask10sites.mask33taxa.bootstrap.tree

eudicot.FNA2AA-upp-masked-c12.fasta.mask10sites.mask33taxa.bootstrap.tree

remove-rogue

filtered\_FAA-upp-masked.fasta.mask10sites.mask33taxa.bootstrap.tree

filtered\_FNA2AA-upp-masked-c12.fasta.mask10sites.mask33taxa.bootstrap.tree

The commands used to reproduce the results are given below.

**Convert the supermatrix alignment to the ExaML input format**

parse-examl -s <input\_alignment> -n <output\_binary\_file> -m <DNA|PROT> -q <partition\_file>

**Compute the RAxML MP starting tree**

raxmlHPC-PTHREADS-SSE3 -T 12 -s <input\_alignment> -p <random\_seed> -m <GTRGAMMA|PROTGAMMAJTT> -n <name> –y -q <partition\_file>

**Compute the FastTree-2 ML starting tree**

FastTreeMP (if DNA: -nt -gtr) -nosupport -out <output\_tree> -nopr <input\_alignment>

**Compute GTR matrix and alpha parameter for codon position**

raxmlHPC-PTHREADS-SSE3 -T 12 -s <input\_codon\_alignment> -T 6 –m GTRGAMMA -n <name> -p 1111

**Generate bootstrap replicate alignments**

raxmlHPC-PTHREADS-SSE3 -T 12 -s <input\_alignment> -f j –m <GTRGAMMA|PROTGAMMAJTT> -n <name> -q <partition\_file> -b 1111 –N 100

**Compute the ExaML final/replicate tree**

examl-OMP -s <input\_binary\_file> -m GAMMA -n <name> -t <starting\_tree>

1. Mirarab, S. *et al.* PASTA: Ultra-Large Multiple Sequence Alignment for Nucleotide and Amino-Acid Sequences. *J. Comput. Biol.* **22,** 377–386 (2015).

2. Nguyen, N., Mirarab, S., Kumar, K. & Warnow, T. Ultra-large alignments using phylogeny-aware profiles. *Genome Biol.* **16,** 124 (2015).

3. Mirarab, S. & Warnow, T. ASTRAL-II: coalescent-based species tree estimation with many hundreds of taxa and thousands of genes. *Bioinformatics* **31,** i44–i52 (2015).

4. Seo, T.-K., Kishino, H. & Thorne, J. L. Incorporating gene-specific variation when inferring and evaluating optimal evolutionary tree topologies from multilocus sequence data. *Proc Natl Acad Sci USA* **102,** 4436–41 (2005).

5. Mirarab, S., Bayzid, M. S. & Warnow, T. Evaluating summary methods for multi-locus species tree estimation in the presence of incomplete lineage sorting. *Syst. Biol.* syu063 (2014). doi:doi: 10.1093/sysbio/syu063

6. Sayyari, E. & Mirarab, S. Fast coalescent-based computation of local branch support from quartet frequencies. *arXiv* **1601.07019,** (2016).

7. Mirarab, S., Bayzid, S. M., Boussau, B. & Warnow, T. Statistical binning enables an accurate coalescent-based estimation of the avian tree. *Science (80-. ).* **346,** 1250463 (2014).

8. Bayzid, M. S., Mirarab, S., Boussau, B. & Warnow, T. Weighted Statistical Binning: Enabling Statistically Consistent Genome-Scale Phylogenetic Analyses. *PLoS One* **10,** e0129183 (2015).

9. Kozlov, A. M., Aberer, A. J. & Stamatakis, A. ExaML version 3: A tool for phylogenomic analyses on supercomputers. *Bioinformatics* **31,** 2577–2579 (2015).

10. Stamatakis, A. RAxML version 8: A tool for phylogenetic analysis and post-analysis of large phylogenies. *Bioinformatics* **30,** 1312–1313 (2014).

11. Pearson, K. On lines and planes of closest fit to systems of points in space. *London, Edinburgh, Dublin Philos. Mag. J. Sci.* **2,** 559–572 (1901).

12. Hartigan, J. A. & Wong, M. A. A K-Means Clustering Algorithm. *Appl. Stat.* **28,** 100–108 (1979).

13. R Development Core Team. R: A Language and Environment for Statistical Computing. *R Found. Stat. Comput. Vienna Austria* **0,** {ISBN} 3–900051–07–0 (2010).

14. Price, M. N., Dehal, P. S. & Arkin, A. P. FastTree-2 – Approximately Maximum-Likelihood Trees for Large Alignments. *PLoS One* **5,** e9490 (2010).

15. Sukumaran, J. & Holder, M. T. DendroPy: a Python library for phylogenetic computing. *Bioinformatics* **26,** 1569–1571 (2010).