Geodesics to characterize the phylogenetic landscape

Marzieh Khodaei[1,*], Megan Owen[2], and Peter Beerli[1]

[1] *Department of Scientific Computing, Florida State University, Tallahassee, FL 32306, USA*
[2] *Department of Mathematics, Lehman College and Graduate Center, CUNY*

**Corresponding author: Marzieh Khodaei, email: mk16e@fsu.edu*

Abstract

Phylogenetic trees are fundamental for understanding the evolutionary history of a set of species. The local neighborhoods of a phylogenetic tree provide important information, but since trees are high-dimensional objects, characterizing these neighborhoods is difficult. Based on the Billera-Holmes-Vogtmann (BHV) distance between pairs of trees, we describe a method to generate intermediate trees on the shortest path between two arbitrary trees, called pathtrees. These pathtrees give a structured way to investigate intermediate neighborhoods between trees of interest in the BHV treespace and can also be used to find high likelihood trees independently of traditional heuristic search mechanisms. We implemented our algorithm in the Python package Pathtrees, which enables the construction of the continuous tree landscape interior of the convex hull of starting trees, low-dimensional visualization of the generated tree landscape, identifying clusters of trees with the same topologies, and searching for the best tree.

Key words: Geodesic distance; phylogenetic treespace; likelihood; convex hull; pathtrees; optimal tree; multidimensional scaling; interpolation

Evolutionary trees, or phylogenetic trees, have been used extensively throughout systematic biology and other fields to represent the evolutionary history of species. Researchers were mostly concerned about recovering the best tree and how to characterize the uncertainty of estimates of the branch lengths and the topology. Today, extracting DNA sequences from individuals from different species is trivial once a tissue sample is acquired. However, generating large phylogenies is still a problem because the number of potential trees to evaluate is astronomical. As a result, heuristic methods are commonly used to find the best tree to fit the observed data. To date, none of the non-exhaustive methods can deliver with certainty the globally best tree, but sophisticated methods explore locally best trees in the hope that the best local tree found is equivalent to the global best tree. Initially, many optimality criteria to compare trees were developed, the most prominent being parsimony (Fitch 1971), distance methods (Fitch & Margoliash 1967, Saitou & Nei 1987), and maximum likelihood (Felsenstein 1981). Computational power has increased considerably since then, so computationally intense probabilistic methods such as maximum likelihood and Bayesian inference have supplanted faster but less accurate methods. Researchers now commonly use maximum likelihood programs , such as RAxML (Stamatakis et al. 2005), PhyML (Guindon & Gascuel 2003), and Paup* (Swofford 2003), or Bayesian inference programs, such as MrBayes (Huelsenbeck & Ronquist 2001), RevBayes (Höhna et al. 2016), and BEAST (Drummond & Rambaut 2007).

Maximum likelihood methods usually produce one best tree, whereas Bayesian inference methods produce a set of trees, of which often the most probable *a posteriori* tree is picked as the best tree. This Bayesian set of trees can be used to construct a tree landscape that focuses on areas of high posterior probability or high tree likelihood but may not allow discussing the ridges and valleys of the surface because the Markov chain Monte Carlo approach favors more probable trees during the run. The common tool for establishing uncertainty of maximum likelihood analyses is the Bootstrap (Felsenstein

1985); it changes the data and therefore cannot be used to investigate the likelihood tree surface because likelihood comparisons depend on having fixed data.

Both heuristic methods use random changes in the tree topology. Even when we record all visited trees in a maximum likelihood search or look at all collected trees in a Bayesian Markov chain Monte Carlo run, we can recognize regions in treespace that are more concentrated and other regions that were not sampled. We may worry that by not reporting about some regions in treespace we may fail to describe potentially good candidates for the best tree, or then do not describe the potential neighborhood of the best tree appropriately.

It is essential to extract the differences and similarities among a set of trees. One approach is to measure distances between pairs of trees, with numerous such distances having been proposed over the years. Some distances only depend on the topology of the trees, such as the popular Robinson-Foulds distance (Robinson & Foulds 1981), the quartet distance (Estabrook et al. 1985), and some rearrangement distances (for example, Nearest-Neighbor Interchange (NNI) distance (Robinson 1971, DasGupta et al. 1997), subtree prune and regraft (SPR) distance (Hein 1990, 1993)). Other distances depend on both the tree topology and branch lengths, such as the weighted Robinson-Foulds distance (Robinson & Foulds 1979), the branch score distance (Kuhner & Felsenstein 1994), and the distance of Kendall & Colijn (2016), which uses a tuning parameter to mix the effect of topology and branch lengths. Finally, some tree distances are the lengths of the shortest paths, or geodesics, between trees in some underlying geometric space, called a treespace. Billera et al. (2001) introduced the concept of a continuous treespace, now called the Billera-Holmes-Vogtmann (BHV) treespace, and corresponding BHV distance, which incorporates both the branch lengths and tree topology. Owen & Provan (2011) presented the first polynomial-time algorithm and the software GTP for computing BHV distances between phylogenetic trees in the BHV treespace.

Sets of trees and their log likelihoods can be visualized as a 3D phylogenetic

"landscape": the first two coordinates give a projection of the trees onto the plane so that the pairwise distances between trees are approximated by the Euclidean distances between points on the plane, and the third coordinate gives the log likelihood (see Fig. 3). Sometimes only the distances between trees are visualized, often using multidimensional scaling (MDS) or another dimensionality reduction technique. MDS approximates the pairwise distances between points by projecting or otherwise arranging them in a lower-dimensional Euclidean space (Kruskal 1964), and was first applied to sets of phylogenetic trees by Amenta & Klingner (2002) and popularized by Hillis et al. (2005*a*). Two more recent packages for visualizing sets of trees are TREESCAPER (Huang et al. 2016), which offers different cost functions for dimensionality reduction and community detection, and TREESPACE (Jombart et al. 2017), which offers a wide variety of tree metrics and methods for clustering trees. Smith (2022) evaluated the performance of multiple aspects of low-dimensional representations of sets of trees. Note that there are other ways to visualize sets of related trees beyond dimensionality reduction, such as super-imposing the trees on each other, as in DensiTree (Bouckaert 2010), or sophisticated tree comparison visualizers, like ADView (Liu et al. 2019).

However, to our best knowledge, there are no current software packages for visualizing tree landscapes - that is, for visualizing a 3D surface showing both the arrangement of the trees in space and their log likelihoods. Log likelihoods were visualized for an MCMC run in Hillis et al. (2005*a*), but only the log likelihoods of the trees encountered on the run, without any interpolation to intermediate trees. Whidden & Matsen IV (2015) include the log likelihoods in their representations of treespaces based on the SPR distance, but this discrete distance and its treespace do not include branch lengths, and we use continuous ones that do. Being able to visualize tree landscapes may shed light on the well-known problem of terraces in treespace, or contiguous regions or treespace containing trees with the same or similar log-likelihood scores, that pose problems for tree search algorithms (Sanderson et al. 2011, 2015).

We remedy the lack of up-to-date tree landscape visualization by extending the method of Owen & Provan (2011) to create trees along the geodesic between arbitrary pairs of trees. This method allows investigating the treespace between a set of arbitrarily picked trees and makes it possible to visualize the surface using the likelihood of the evaluated trees even in areas where the standard heuristic methods fail to visit trees. We distribute our approach in the Python package Pathtrees. This package has two major objectives: to visualize treespace and find the best tree independent of heuristic searches. We explore our method with two examples and compare our results with heuristic searches executed in Paup*, RAxML, and the maximum a posteriori (MAP) phylogeny in RevBayes.

Materials and Methods

We have developed a Python package Pathtrees that allows the generation of an arbitrary number of trees along the shortest path between pairs of bifurcating phylogenetic trees through the Billera-Holmes-Vogtmann treespace (Billera et al. 2001). These trees may help to improve visualization of the treespace and may find the best tree given a genetic dataset without a heuristic search. The method uses the Java package GTP (Owen & Provan 2011) to generate the geodesic, the program Paup* (Swofford 2003) for likelihood optimization, the Python modules Dendropy (Sukumaran & Holder 2010) for the Robinson-Foulds metric, and several other standard Python modules, such as Scipy and Numpy (Virtanen et al. 2020, Harris et al. 2020).

Finding the Shortest Path through Treespace

The Geodesic Treepath Problem (GTP) is a polynomial algorithm with runtime $O(n^4)$, where n is the number of leaves, proposed by Owen & Provan (2011) to compute the shortest distance between phylogenetic trees in the continuous treespace introduced by Billera et al. (2001). This Bilera-Holmes-Vogtman (BHV) treespace models all phylogenetic trees with a fixed set of leaves. It is formed from a set of euclidean regions,

called orthants, such that every orthant consists of all topologically identical trees. Two orthants with corresponding tree topologies that share splits are adjacent in the treespace, and their shared boundary orthant contains all trees containing exactly the shared splits. All orthants contain the origin, which corresponds to the star tree, so the space is connected. The length of a path between two trees is computed by measuring the Euclidean length of the path in each orthant it passes through, and summing those lengths. There is a unique shortest path, or geodesic, connecting two phylogenetic trees T_1 and T_2 in BHV treespace (Billera et al. 2001).

If trees T_1 and T_2 have no common splits, then the GTP algorithm to compute the geodesic starts with a simple initial path, called cone path, which is obtained by connecting tree T_1 to the origin (a star tree) by a straight line and then connecting T_2 to the origin by another straight line. Then, the cone path is transformed into a series of successively shorter paths until the geodesic is obtained. At each step, the algorithm identifies one new orthant that the current path can be modified to pass through to yield a shorter path. When trees T_1 and T_2 have splits in common, the algorithm first subdivides T_1 and T_2 along the common splits, and runs the GTP algorithm described above on each pair of subtrees. The shortest paths between the subtrees are then combined into the overall geodesic between T_1 and T_2.

Fig. 1 demonstrates an example of a geodesic between two trees, and the geometric representation of treespace. The top row shows a starting tree (T_1), an ending tree (T_2), and the two trees where the geodesic between T_1 and T_2 crosses orthant boundaries. The bottom row shows the geodesic, the cone path, parts of the three orthants that the geodesic passes through, and an example tree of each orthant.

Creating New Trees along the Shortest Path through Treespace

Recall that each orthant of BHV treespace contains only trees with the same topology. Each tree topology consists of a unique set of splits (Buneman 1971), and each of

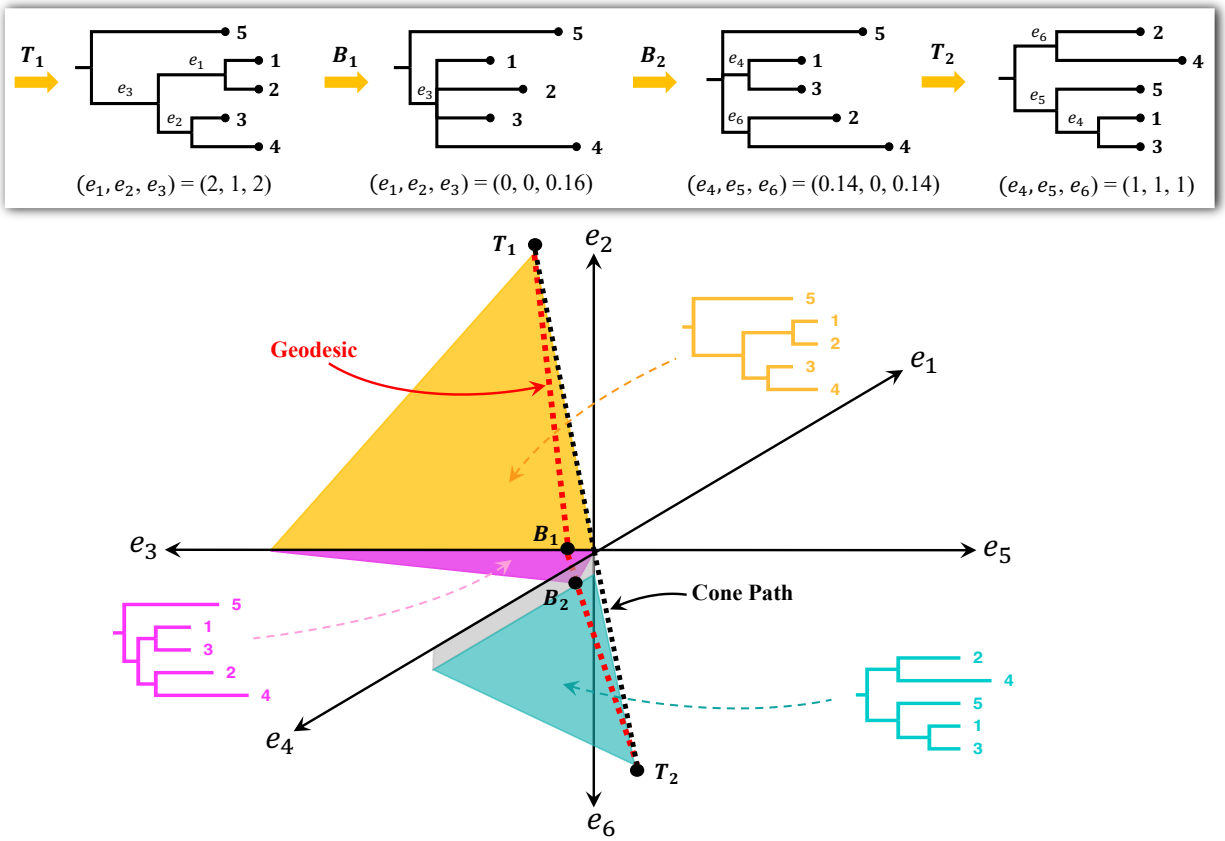


Figure 1. Shortest path between two trees in the BHV treespace. First row: a start tree T_1 and an end tree T_2 with two intermediate trees B_1 and B_2 on the boundary of orthants. Bottom row: Black dots mark the trees in the top row; the cone path is the black dotted line; the geodesic is the red dotted line; each colored region is part of a different orthant containing the geodesic; and an example tree with arbitrary branch lengths is shown for each orthant.

these splits is assigned to one of the dimensions of the orthant. Each branch length in a tree becomes the coordinate in the orthant in the dimension of the branch's corresponding split. Moving along the geodesic from start tree T_1 to end tree T_2, branches shrink to zero length at orthant boundaries, and new branches begin to grow, as tree T_1 morphs into tree T_2.

Specifically, when the program GTP computes the geodesic between trees T_1 and T_2, it generates the support (A, B), with $A = (A_1, A_2, ..., A_k)$ where A_i is the set of splits whose branches shrink to zero length at the i-th orthant boundary along the geodesic, and with $B = (B_1, B_2, ..., B_k)$ where B_i is the set of splits whose branches begin growing from zero length at the i-orthant boundary. Let C be the set of splits common to T_1 and T_2, and parametrize the geodesic between T_1 and T_2 by $0 \leqslant \lambda \leqslant 1$. For split e in tree T, denote its

branch length in T by $|e|_T$ and for the set of splits S in tree T, let $||S|| = \sqrt{\sum_{e \in S} |e|_T^2}$. Then by Theorem 2.4 of Owen & Provan (2011) and Theorem 1.2 of Miller et al. (2015), a tree T_i on this geodesic at position λ in the i-th orthant contains exactly the splits

$$\mathcal{S} = C \cup B_1 \cup ... \cup B_i \cup A_{i+1} \cup ... \cup A_k, \tag{0.1}$$

such that split $e \in \mathcal{S}$ has branch length

$$|e|_{T_i} = \begin{cases} \frac{(1-\lambda)||A_j|| - \lambda||B_j||}{||A_j||}|e|_{T_1} & e \in A_j \\ \frac{\lambda||B_j|| - (1-\lambda)||A_j||}{||B_j||}|e|_{T_2} & e \in B_j \\ (1-\lambda)|e|_{T_1} + \lambda|e|_{T_2} & e \in C \end{cases} \tag{0.2}$$

Thus, the trees along the geodesic between T_1 and T_2 can only contain splits from T_1 and T_2, and any split common to T_1 and T_2 appears in all trees along the geodesic between them. Each tree T_i generated along the path is then stored in the standard Newick format. Some trees on the geodesic are not bifurcating trees. To deal with that, we transform each internal non-bifurcating edge into multiple bifurcating edges with zero length.

Visualization of Treespace

To visualize the pathtrees and input trees, we generate a distance matrix of all n trees and then use this distance matrix as input into a multidimensional scaling (MDS) algorithm (Cox & Cox 2008) to compress the higher dimensional treespace into two dimensions; every tree is represented by an x, y coordinate and we can establish the neighborhood of each tree (Hillis et al. 2005*b*). Pathtrees has two options for generating the distance matrix: the BHV distance and the weighted Robinson-Foulds (RF) distance (Robinson & Foulds 1979). The weighted RF distance is faster to compute than the BHV distance but is a $\sqrt{2}$ approximation of it (Amenta et al. 2007).

Since we are interested not only in the relationship among the trees but also in how well they fit the data, we calculate the log likelihood for each tree. The Pathtrees

package currently contains a simple likelihood calculator to compute the log-likelihood score using a fixed topology, fixed branch lengths, and the Jukes-Cantor mutation model (Jukes & Cantor 1969).

For a smooth representation of the likelihood surface, we interpolate the log-likelihood values between the n trees using an interpolation method. In PATHTREES, there are two options to interpolate: the cubic spline interpolation method (De Boor 1978) and the radial basis function (RBF) thin-plate spline interpolation (Buhmann 2003).

The pathtrees extracted along the shortest path between two trees do not have optimized branch lengths, and some may share the same topology. To find the best tree and also find potential local likelihood maxima, we need to find pathtrees with the same topology and then optimize the branch lengths for that topology. We apply the unweighted Robinson-Foulds distance (Robinson & Foulds 1981) to detect the different topologies; then we use PAUP* to optimize the branch lengths of a tree in each topology cluster.

Finding the Starting Trees for PATHTREES

Our package can generate random starting trees, but with a larger number of taxa, these random trees span a very large section of treespace. Given preliminary experiments, it doesn't seem easy to achieve good improvements to find the best likelihood tree. Instead, we use REVBAYES to generate a large sample of trees using Markov chain Monte Carlo. Then, we discard a portion of the trees as burn-in and project the remaining trees using multidimensional scaling. The trees that are on the vertices of the convex hull, the smallest convex polygon enclosing all of the trees on the projected plane, are then extracted and used as the starting trees for PATHTREES to generate the tree landscape interior of the convex hull. For example, we generated 50,000 trees using the program RevBayes and the primate dataset (outlined in more detail in the section Application to Real Data) and then extracted 1000 trees from the last 1/10 of the MCMC chain. Fig. 2 demonstrates the space of 1000 trees and the trees on the vertices of the corresponding convex hull.

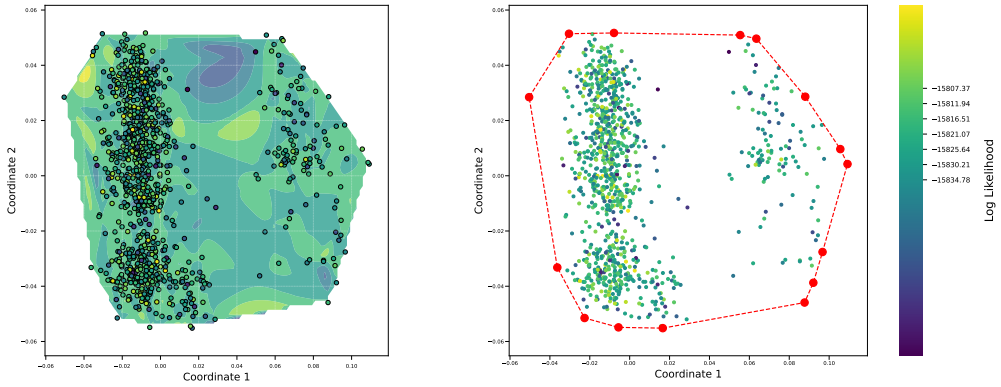


Figure 2. An example of sample treespace and its convex hull. Left: A log-likelihood contour plot of the first two multidimensional scaling (MDS) coordinates of the sampled trees. The log-likelihood contour is a cubic spline interpolation of the log likelihoods of all trees in the MDS plane; the MDS coordinates are computed from the BHV distances between trees. Each dot is a tree; the lighter the dot, the higher the likelihood of the tree. Right: the convex hull of the set of trees. The red dots are the vertices of the convex hull, displaying the sample trees on the boundary of the treespace.

Generating and Visualizing Pathtrees

The distribution of the sampled trees shown in Fig. 2 highlights that some areas of treespace were sampled less well than others. In contrast, choosing trees along geodesics allow us to visualize trees that are evenly spaced between two arbitrary end-point trees. We demonstrate these pathtrees in Fig. 3 where we selected three trees (colored triangles) from the 1000 sample trees visualized in Fig. 2 and generated 20 trees on the shortest path between each pair of them and then visualized the contour and the surface of all 1060 trees. The selected three trees are provided in the Appendix (Fig. 8). The pathtrees bridge the gaps between the sampled trees. We do not control the orientation of the MDS plot, and comparing Fig. 2 with Fig. 3 shows that the orientation changed.

The pathtrees lay on the shortest path between a start and end tree (anchor trees). This path is constructed by refinement of the cone path as discussed in section *Finding the Shortest Path through Treespace*, so the resulting pathtrees will more commonly have short branches. To use our pathtrees to find maxima in treespace, we optimize the branch lengths without changing the topology. Fig. 4 shows the locations of the pathtrees and their corresponding optimized trees. If the anchor trees have different topologies, they are

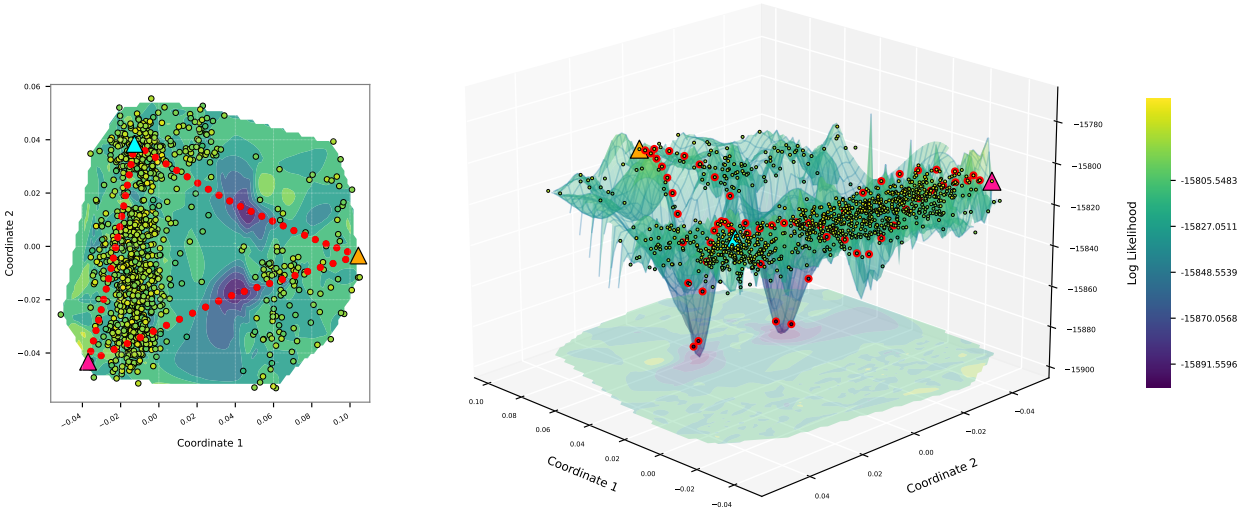


Figure 3. An example of pathtrees between three arbitrary trees in the treespace. Cubic spline interpolation of the log likelihood was used for the contour color (left) and the surface height (right) of the space inside the convex hull. 20 pathtrees (red dots) were generated on the shortest path between each pair of 3 trees (triangles).

located in different orthants. The pathtrees can have different intermediate topologies that, when optimized, will be located away from the shortest path. Unoptimized trees that start in low likelihood areas of treespace may move particularly far away, whereas trees that start on a ridge may not move far.

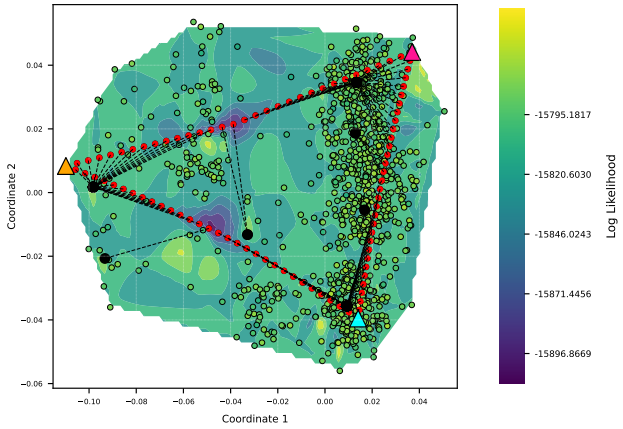


Figure 4. An example of pathtrees and their corresponding optimized trees. Each pathtree is connected to its corresponding optimized tree (black circles).

Treespace Visualization and Best Tree Search Algorithm

We summarize the tree searching strategy of our package PATHTREES in the algorithm:

Input: N sample trees that are rooted and non-ultrametric, and the sequence data used to generate trees. The trees need to be in plain Newick format, and the sequence data needs to be in PHYLIP format.

Output: Trees on the shortest path through treespace between all pairs of starting trees; optimized branch lengths for different encountered topologies in the pathtrees; MDS visualization of treespace inside the convex hall of input trees.

Algorithm:

1. Compute all pairwise distances between N sample trees, and compute their MDS coordinates.
2. Extract the trees on the vertices of the convex hull of projected N sample trees and consider them as starting trees.
3. Generate m equally spaced trees on the geodesic between each pair of starting trees. Put all starting trees and generated pathtrees in the set current sample trees.
4. Calculate the likelihood of all current sample trees.
5. Select n trees with the highest likelihood among current sample trees and classify them with respect to their topologies. Let there be t topologies among them.
6. For each topology cluster, optimize the branch lengths for that topology and add these t optimized trees to the current sample trees.
7. Visualize the contour and surface of the current sample trees by creating a distance matrix using either the weighted Robinson-Foulds or the BHV distance metric, interpolating log-likelihood values, and recomputing the MDS coordinates.
8. Return the current sample trees, the pathtrees, and the visualization; or continue with step 3 to zoom in the area of optimized trees of the current iteration.

Application to Real Data

We evaluate our approach PATHTREES using two datasets that were previously published: D_1 is a dataset of primates used in the tutorial for the program REVBAYES (Höhna et al. 2021). The dataset consists of 1141 base pairs of the mitochondrial cytochrome b gene of 23 primate taxa. D_2 is a larger mitochondrial cytochrome b dataset of 182 milksnakes (1117 bp) (Ruane et al. 2014, Chambers & Hillis 2019). We chose the two different datasets because they represent very different situations. D_1 is a relatively small dataset but still too large to consider an exhaustive tree search. The species in the dataset are well defined, and the dataset contains enough variability to establish a phylogeny with branch lengths that are neither zero nor huge. In contrast, D_2 has about $8\times$ more individuals than D_1; these individuals are only from a few species or subspecies, and many individuals share the same DNA sequence with others in the dataset. We deliberately did not attempt to consolidate the dataset into unique site patterns, anticipating that the many zero branch lengths would be a stress test for our method.

For the first dataset, we generated 50,000 trees using the program REVBAYES and dataset D_1. We used the instructions from the tutorial of REVBAYES (Höhna et al. 2021) which are shown in the Appendix. We selected every 38th tree from the last 3/4 of the total 50,000 sample trees (around 1000 sample trees) and then extracted the trees on the vertices of the convex hull of these sample trees (14 trees) as starting trees for PATHTREES. For the second dataset, we generated 10,000 trees using the program REVBAYES and dataset D_2. After removing the first 300 trees as burn-in, we selected every 20 trees and extracted 500 trees to be considered in convex hull analysis. The 14 trees on the vertices of the convex hull were used as starting trees in PATHTREES.

Comparison of PATHTREES *with Heuristic Tree Searches*

We compared our results with those generated by the maximum likelihood software PAUP* 4.0a (build 168) (Swofford 2003), RAXML 8.2.12 (Stamatakis et al. 2005), and the

Bayesian inference software RevBayes 1.1.1 (Höhna et al. 2016). These programs perform heuristic searches. Paup* and RAxML search will swap on new tree topologies until a local maximum has been reached and no new tree topologies need further evaluations. These heuristics do not guarantee to recover the global maximum likelihood tree but usually deliver good results. In contrast, RevBayes uses Markov chain Monte Carlo to evaluate the posterior probability of a tree while collecting trees along a Markov chain. The run time is user-determined and needs to be long enough to sample good candidate trees. These trees are then used to estimate the maximum a posteriori tree.

We conducted several experiments to evaluate whether the number of intermediate pathtrees for each pair of anchor trees affects the accuracy of the MDS reconstruction of the tree-likelihood surface and how well we can recover the best tree. We use the Jukes-Cantor mutation model for the likelihood calculation throughout all analyses. Using such a simple model reduces potential difficulties introduced by parameter fitting. Both Paup* and RAxML were run without improving parameters that tune the heuristic search.

Running the heuristic programs was problematic in a few cases: RAxML insists on using a site-rate variation, which is often a good suggestion, but we decided to use a simpler model to compare trees. As a result, we compared Pathtrees with Paup* and RevBayes for dataset D_1, and Pathtrees with Paup* and RAxML for dataset D_2.

Results

We illustrate the behavior of Pathtrees by examining the two different datasets D_1 and D_2. For the first dataset, we experimented with using 1 or 15 pathtrees between each pair of anchor trees. For both datasets, we focus on the description of the treespace and similarities of trees found with heuristic methods.

Fig. 5 illustrates the contour and surface plots of the generated treespace of Pathtrees, using MDS and thin-plate spline interpolation of the treespace defined by the

BHV distance matrix for D_1. We generated one tree on the geodesic between each pair of 14 starting trees (the triangles), creating 91 pathtrees, excluding starting and ending trees. In the first experiment (the first row), we selected 15 trees with the highest likelihood from the $91 + 14$ trees and then extracted the trees with different topologies among them (4 different topologies). The branch lengths in each topology cluster were optimized (medium-size circles). The purple spectrum color bar shows the relationship between the topology of each optimized tree and its likelihood value; the lighter the color of the topology (medium-sized dots), the lower the likelihood value of the corresponding optimized tree in that topology. Among locally optimal trees associated with different topologies, we found the best tree with the log likelihood of -15795.1817 (red square). We compared our results with the best tree found by Paup* and with the MAP tree of RevBayes. Our Pathtrees finds the best tree of Paup* (big white circle) as the optimal tree and the MAP tree (big black circle) as a locally optimal tree. In the second experiment (the second row in Fig. 5), we selected all 91 pathtrees and extracted the trees with different topologies among them (30 different topologies in total) . Among all locally optimized trees (30 local optima corresponding to different topologies) , we found again the best tree with the log likelihood of -15795.1817.

In Fig. 6, we applied the weighted Robinson-Foulds distance metric to calculate the distance matrix to be used in the MDS projection for the first dataset. We generated 15 pathtrees for each pair to show a detailed treespace with a higher number of pathtrees (in total, 1365 pathtrees). Among all 43 different topologies, Pathtrees found the same best tree as Paup*, like in the other trials, with the log likelihood of -15795.1817. With a higher number of pathtrees, we have a surface where trees are evenly spread out, and the gaps are filled. The best trees detected by Pathtrees, Paup*, and RevBayes are added to the Appendix (Fig. 9). We have discussed in Appendix, in the section Effect of Interpolation Method on Visualization, the differences between two methods of the cubic spline interpolation and the thin-plate spline interpolation (Fig. 11 and Fig. 12).

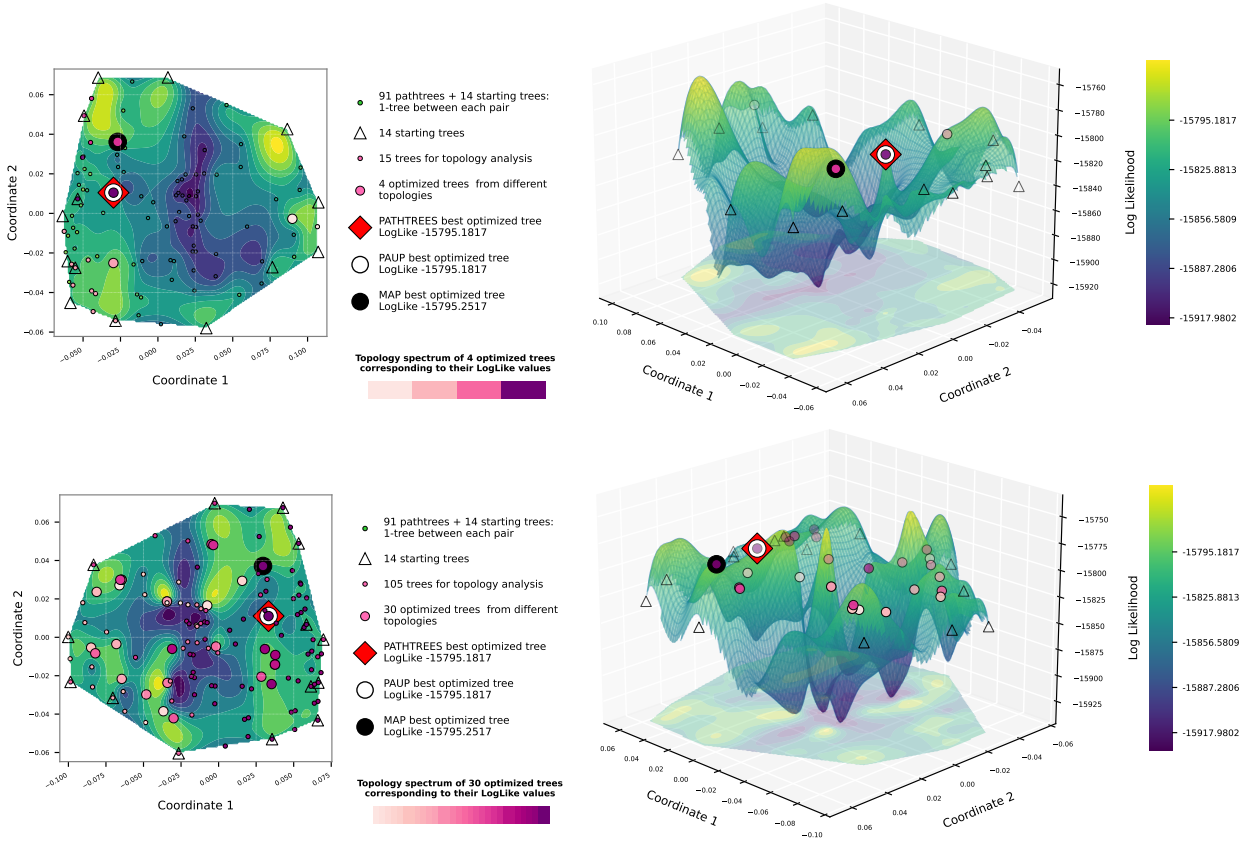


Figure 5. Contour and surface plots of PATHTREES for dataset D_1 by generating one pathtree per anchor tree pair, and using MDS and thin-plate spline interpolation defined by the BHV distance metric. The vertices of the convex hull of the selected 1000 sample trees are the starting trees (14 triangles). One tree was generated on the geodesic between each pair of starting trees (91 pathtrees). First row: 15 trees with the highest likelihood were selected from the 91+14 trees and classified based on their topologies (4 topologies with different colors of the purple spectrum). Each medium-size circle with a purple color shows the optimized tree with the corresponding topology (locally optimal trees). The red square shows the best likelihood tree in the treespace found by PATHTREES. The large white disk shows the best tree of PAUP*, which is identical to PATHTREES' optimal tree. One of PATHTREES' locally optimal trees matches the MAP tree (big black circle). Second row: all trees were selected and classified based on their topologies (30 topologies). Medium-size disks on a purple spectrum showed the 30 locally optimal trees. The big red square shows the best likelihood tree of PATHTREES.

Fig. 7 shows the contour and surface plots of the generated treespace by PATHTREES for the second dataset. We applied thin-plate spline interpolation and the MDS projection defined by the weighted Robinson-Foulds distance metric. We applied two iterations of PATHTREES to zoom in the treespace. In the first iteration (first row), we generated 4 pathtrees on the geodesic between each pair of 14 starting trees and then selected the 100 trees with the highest likelihood and classified them with respect to their topologies. All selected trees have different topologies. We optimized a tree in each

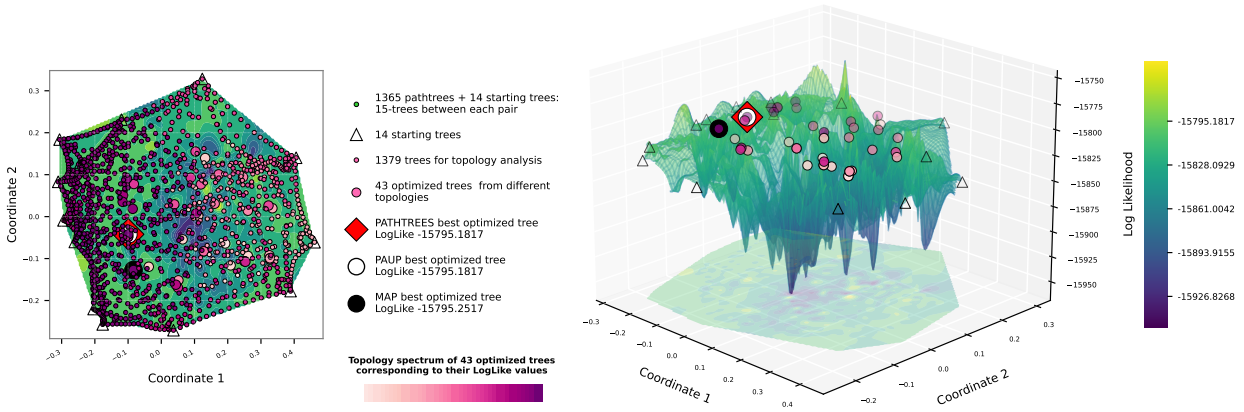


Figure 6. Contour and surface plots of PATHTREES for dataset D_1 by generating 15 pathtrees per anchor tree pair, and using MDS and thin-plate spline interpolation defined by the weighted Robinson-Foulds distance metric. 15 trees were generated on the geodesic of each pair of starting trees (1365 pathtrees). Among 43 optimized trees with the highest likelihood in topology clusters, PATHTREES found the best tree (red square) with the log likelihood -15795.1817, which is the same as the optimal tree of PAUP* (white circle).

topology cluster. PATHTREES found a similar tree to the best tree from PAUP*, both with the log likelihood -5225.5856. In the second iteration, we zoomed in. The starting trees for this second iteration are the trees on the vertices of the convex hull of the 100 optimized trees of the first iteration. After generating 5 pathtrees between each pair and classifying the generated pathtrees based on topology, we found 154 different topologies. Among optimized trees in each topology cluster, we found the same optimal trees as before.

A comparison of the three optimal trees, namely ours, the best PAUP* tree, and the best RAxML tree, revealed that all trees are different from each other. Table 1 shows the weighted and unweighted Robinson-Foulds distances between the trees and their log likelihoods. Our tree and the PAUP* tree are close and only differ by four splits, whereas the RAxML tree differs from both our tree and the PAUP* tree by nine splits. The weighted RF distances between these three trees show similar relationships. The RAxML tree is different because its topology was found by applying a JC69 model with site rate variation (RAxML always uses site rate variation), but we then used PAUP* to find the optimal branch lengths for that topology, and the plain JC69 model to compute the likelihoods. The log likelihoods for all these 'best' trees are very similar, and comparing their location on the surface in Fig. 7 also shows that the best PATHTREES and PAUP*

trees are close, whereas the RAxML tree seems to be on a different local maximum on the surface. The Appendix Fig. 10 contains all three trees showing their topology differences. Trees with 182 leaves that only differ by nine splits are certainly very similar looking.

Table 1. Comparison of the best tree found by Pathtrees, Paup*, and RAxML using the data D_2. Above the diagonal is unweighted Robinson-Foulds (uRF) for all pairs and below the diagonal is the weighted Robinson-Foulds distance (wRF). The last column is the log likelihood for each tree.

Tree	uRF / wRF			ln L
	Pathtrees	Paup*	RAxML	
Pathtrees	-	4	9	-5225.5856
Paup*	0.0054	-	9	-5225.5856
RAxML	0.0343	0.0325	-	-5225.8638

Discussion

Our approach describes the relationship among phylogenetic trees using the Billera-Holmes-Vogtmann treespace framework. We focused on two aspects: the visualization of treespace and whether we can improve searching for the best trees, including by investigating global and local maxima using maximum likelihood. As far as we know, Pathtrees is the only recent tool for visualizing phylogenetic landscapes. It can be used at different scales (e.g., see Fig. 7), which is very helpful for understanding the relationship between the highest likelihood trees and tree neighborhoods. For example, in Fig. 7, the initial landscape shows the highest likelihood trees are close together within the MCMC searched area of treespace; zooming in allows us to see that these trees form two high likelihood ridges, with a lower-likelihood region between them. Thus Pathtrees could potentially be used to understand treespace terraces better. The treespace is enormous. For our two example datasets, we need to consider a space that has 5.6×10^{26} orthants and 4.7×10^{384} orthants, respectively. Our current approach will need trees to start that are good candidates, and we decided to use a sample of trees from a Bayesian phylogenetic inference program; we picked RevBayes. In principle, any set of reasonably

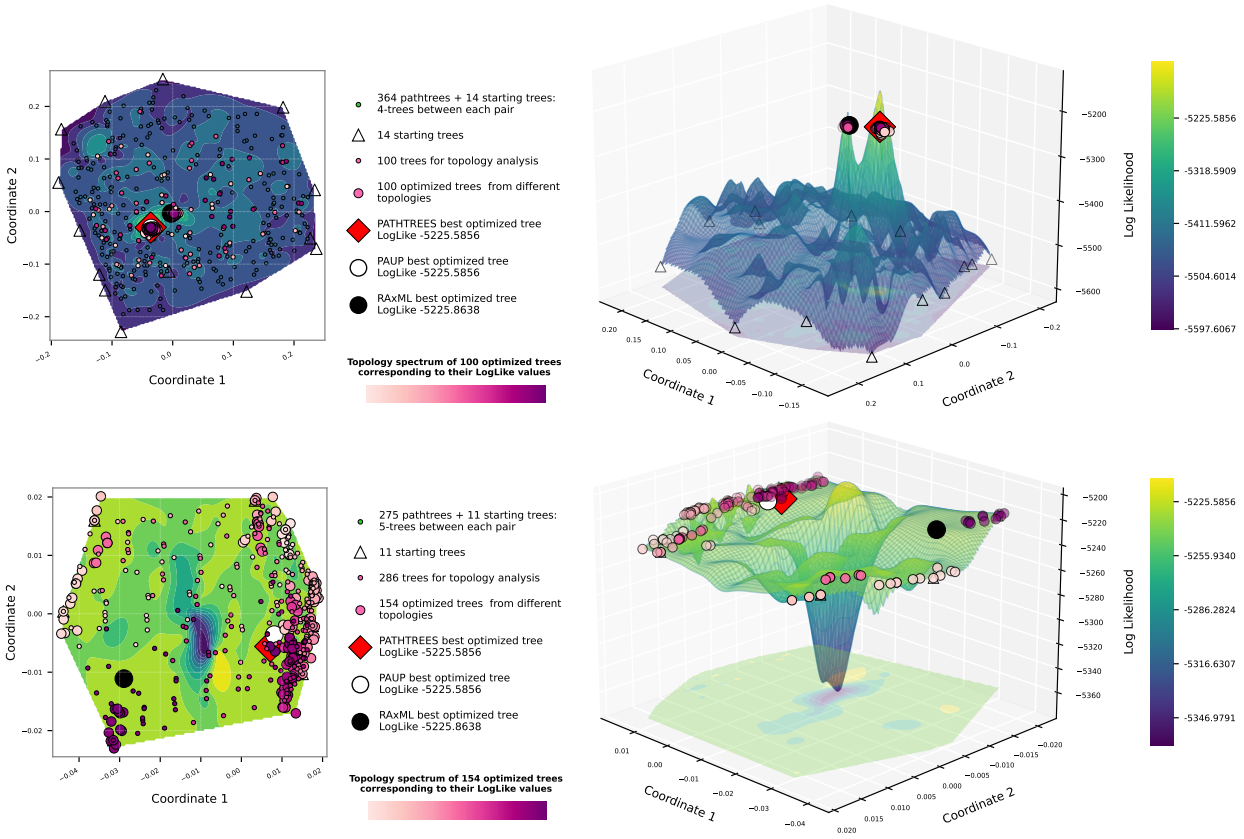


Figure 7. Contour and surface plots of PATHTREES for dataset D_2 , using MDS and thin-plate spline interpolation defined by the weighted Robinson-Foulds distance metric. First row: the vertices of the convex hull of the selected 500 sample trees are the starting trees (14 triangles). Four trees were generated on the geodesic of each pair of starting trees (364 pathtrees). Among them, 100 trees with the highest likelihood were selected to be classified based on topology. All 100 selected trees have different topologies (small circles with colors from the purple spectrum). Each medium-size circle with a color from the purple spectrum shows the optimized tree with the corresponding topology. Among optimized trees, the red square shows the best likelihood tree in the treespace recognized by PATHTREES, which is close but not identical to the best tree of PAUP* (white circle). The RAXML tree is different. Second row: Displays the treespace after zooming in the optimized trees. The vertices of the convex hull of 100 optimized trees from the first iteration are the starting trees (11 triangles). Five trees were generated between each pair (275 trees) and then all of them were selected to be classified according to their topology category.

close trees to the best tree may work as a starting point for our explorations. Using the convex hull in MDS space to start our approach helped reduce the geodesic distance matrix size used to create the visualizations. In a way, we treat the convex hull in MDS space as an approximation of the convex hull in treespace (Lubiw et al. 2020, Lin et al. 2017). However, we believe that the hull defined by MDS allows us to investigate good and best trees within its boundary. In our examples, the best tree, found by other procedures and ours, is within this MDS hull.

A Bayesian inference method evaluates trees according to the posterior probability, which is dominated by the likelihood of the tree when we assume vague prior distributions. It is fair to say that even a long inference run will not explore all possible topologies. Even our small dataset of 23 primate species has too many different topologies to explore all in a Bayesian context in a reasonable time. Of course, most of these topologies will have an inferior fit to the sequence data, but even those trees that fit the data relatively well are many. In contrast to Bayesian inference and heuristic search methods, our method does not depend on an optimality criterium to pick trees that lay on the shortest path between two arbitrarily chosen trees. This allows exploring topologies that were never visited with a good Bayesian run or any other heuristic search.

While Pathtrees does not depend on an optimality criterion to generate new tree topologies, tree topologies along geodesics can only contain splits that are in one of the two endpoints trees and contain all splits that are in both of the endpoint trees. This constraint is not onerous - there can still be exponentially many topologies fitting this description - but means that the intermediate tree topologies explored are relevant.

Trees on the shortest path between two start trees are most likely not in the set of best trees because this shortest path defines their branch lengths and topology. Our method takes these pathtrees as starting points and optimizes branch lengths but does not change the topology. This procedure allows us to describe the tree landscape along the path and describes the maxima for specific topologies. We believe that since a maximum for a particular topology is not necessarily at an orthant boundary of the BHV treespace, they are local maxima. Using many pathtrees that were optimized or not optimized will give a precise picture of the tree landscape (for example, Fig. 5, Fig. 6, and Fig. 7). Pathtrees provides a simple framework for exploring landscapes of phylogenetic trees and visualizing their relationship in a continuous and low-dimensional projection facilitated by multidimensional scaling and an interpolation method, cubic spline or thin-plate spline interpolation, to reveal potential tree islands. The visualization gives a

good impression of the likelihood landscape: general patterns can be shown with few trees, but details may need many more trees to create a more smooth surface. However, even with many trees, the visualization may contain artifacts in areas where there are no trees, for example, the spikes in Fig. 7.

We picked the two datasets because they represent very different situations. The primate dataset D_1 is relatively small. However, it is still too big to be solved exhaustively. It provides many mutational differences allowing good resolution of branch lengths and branching patterns. The second dataset D_2 has $8\times$ more individuals that are closely related. Many sequences are identical, leading to many multifurcations.

Heuristic searches for D_1 and D_2 are fast, and even an MCMC run with RevBayes does not need a long time for D_1. However, we had difficulties estimating a MAP tree for D_2. Pathtrees generates independent trees that help visualize the space and independently find optimal trees and allow describing the neighborhood of these optimal trees. Pathtrees optimizes its pathtrees and finds local maxima for the evaluated topologies; these are the same as those found by Paup* and RevBayes for D_1. Interestingly, the MAP tree and the Paup* tree differ by two splits but when the branch lengths are optimized deliver log likelihoods that are very similar. Even 10x longer runs in RevBayes deliver the same MAP tree. So while it may seem difficult for RevBayes to explore that particular topology with the highest likelihood, the difference is only 0.07 log units. We certainly would not exclude the MAP tree in a likelihood ratio test. The second dataset D_2 reveals that many trees will be good candidates for the best likelihood tree. The Paup*, RAxML, and Pathtrees best trees have all different topologies but very similar log likelihoods. These trees are also very similar, with only 4 or 9 different splits between them. Pathtrees helps give insights about the local log-likelihood surface, such that it is rather flat and therefore will have many potential trees with similar likelihoods.

We believe that Pathtrees complements heuristic search phylogenetic analyses and allows visualization of the treespace and finding alternative trees with log likelihoods

that are potentially better than those of heuristic searchers. The visualization of the likelihood surface also allows the discussion of local log-likelihood maxima, which we hope will lay the groundwork for improving search algorithms.

Acknowledgements

PB was partially supported by the US National Science Foundation grant DBI 1564822 and DBI 2019989. MO was partially supported by the US National Science Foundation grant DMS 1847271.

Software Availability

The software Pathtrees is available as opensource code in the repository: `https://github.com/pbeerli/pathtrees_project.git`

The primate data is available from the Tutorial website of RevBayes: `https://revbayes.github.io/tutorials/ctmc/#example-character-evolution-under-the-jukes-cantor-substitution-model` (the dataset url is `https://revbayes.github.io/tutorials/ctmc/data/primates\_and\_galeopterus\_cytb.nex`).

The snake dataset is available from the Dryad Digital Repository: `http://dx.doi.org/10.5061/dryad.7hs34mj`.

References

Amenta, N., Godwin, M., Postarnakevich, N. & John, K. S. (2007), 'Approximating geodesic tree distance', *Inf. Process. Lett.* **103**(2), 61–65.

Amenta, N. & Klingner, J. (2002), Case study: Visualizing sets of evolutionary trees, *in* 'IEEE Symposium on Information Visualization, 2002. INFOVIS 2002.', IEEE, pp. 71–74.

Billera, L. J., Holmes, S. P. & Vogtmann, K. (2001), 'Geometry of the space of phylogenetic trees', *Adv. Appl. Math.* **27**, 733–767.

Bouckaert, R. R. (2010), 'Densitree: making sense of sets of phylogenetic trees', *Bioinformatics.* **26**(10), 1372–1373.

Buhmann, M. D. (2003), Chapter 2 - summary of methods and applications, *in* 'Radial basis functions: theory and implementations', Vol. 12, Cambridge university press.

Buneman, P. (1971), The recovery of trees from measures of dissimilarity, *in* 'Mathematics the the Archeological and Historical Sciences: Proceedings of the Anglo-Romanian Conference, Mamaia, 1970', Edinburgh University Press, pp. 387–395.

Chambers, E. A. & Hillis, D. M. (2019), 'The Multispecies Coalescent Over-Splits Species in the Case of Geographically Widespread Taxa', *Syst. Biol.* **69**(1), 184–193.

Cox, M. A. & Cox, T. F. (2008), Multidimensional scaling., *in* 'Handbook of data visualization', Springer, Heidelberg, Berlin, pp. 315–347.

DasGupta, B., He, X., Jiang, T., Li, M., Tromp, J. & Zhang, L. (1997), 'On distances between phylogenetic trees', *SODA.* **97**, 427–436.

De Boor, C. (1978), Piecewise cubic interpolation., *in* 'A practical guide to splines', Springer-Verlag, New York, pp. 40–47.

Drummond, A. J. & Rambaut, A. (2007), 'Beast: Bayesian evolutionary analysis by sampling trees', *BMC Evol. Biol.* **7**, 1–8.

Estabrook, G. F., McMorris, F. R. & Meacham, C. A. (1985), 'Comparison of undirected phylogenetic trees based on subtrees of four evolutionary units', *Syst. Zool.* **34**, 193–200.

Felsenstein, J. (1981), 'Evolutionary trees from DNA sequences: a maximum likelihood approach', *J. Mol. Evol.* **17**(6), 368–376.

Felsenstein, J. (1985), 'Confidence limits on phylogenies: An approach using the bootstrap', *Evolution.* **39**(4), 783–791.

Fitch, W. M. (1971), 'Toward defining the course of evolution: minimum change for a specified tree topology', *Syst. Zool.* **20**, 406–416.

Fitch, W. M. & Margoliash, E. (1967), 'Construction of phylogenetic trees', *Science.* **155**(3760), 279–284.

Guindon, S. & Gascuel, O. (2003), 'A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood', *Syst. Biol.* **52**(5), 696–704.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., Fernández del Río, J., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C. & Oliphant, T. E. (2020), 'Array programming with NumPy', *Nature.* **585**, 357–362.

Hein, J. (1990), 'Reconstructing evolution of sequences subject to recombination using parsimony', *Math. Biosci.* **98**, 185–200.

Hein, J. (1993), 'A heuristic method to reconstruct the history of sequences subject to recombination', *J. Mol. Evol.* **36**, 396–405.

Hillis, D. M., Heath, T. A. & John, K. S. (2005*a*), 'Analysis and visualization of tree space', *Syst. Biol.* **54**(3), 471–482.

Hillis, D. M., Heath, T. A. & John, K. S. (2005*b*), 'Analysis and visualization of tree space', *Syst. Biol.* **54**, 471–482.

Höhna, S., Landis, M. J., Heath, T. A., Boussau, B., Lartillot, N., Moore, B. R., Huelsenbeck, J. P. & Ronquist, F. (2016), 'Revbayes: Bayesian phylogenetic inference using graphical models and an interactive model-specification language', *Syst. Biol.* **65**, 726–736.

Höhna, S., Landis, M., Moore, B. & Heath, T. (2021), 'Nucleotide substitution models: Phylogenetic inference of nucleotide data using revbayes'.
URL: *https://revbayes.github.io/tutorials/ctmc/#example-character-evolution-under-the-jukes-cantor-substitution-model*

Huang, W., Zhou, G., Marchand, M., Ash, J. R., Morris, D., Van Dooren, P., Brown, J. M., Gallivan, K. A. & Wilgenbusch, J. C. (2016), 'Treescaper: visualizing and extracting phylogenetic signal from sets of trees', *Mol. Biol. Evol.* **33**(12), 3314–3316.

Huelsenbeck, J. P. & Ronquist, F. (2001), 'Mrbayes: Bayesian inference of phylogenetic trees', *Bioinformatics.* **17**, 754–755.

Jombart, T., Kendall, M., Almagro, A., Garcia, J. & Colijn, C. (2017), 'treespace: Statistical exploration of landscapes of phylogenetic trees', *Mol. Ecol.* **17**, 1385–1392.

Jukes, T. H. & Cantor, C. R. (1969), Chapter 24 - evolution of protein molecules, *in* H. MUNRO, ed., 'Mammalian Protein Metabolism', Academic Press, pp. 21–132.

Kendall, M. & Colijn, C. (2016), 'Mapping phylogenetic trees to reveal distinct patterns of evolution', *Mol. Biol.* **33**, 2735–2743.

Kruskal, J. B. (1964), 'Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis', *Psychometrika.* **29**(1), 1–27.

Kuhner, M. K. & Felsenstein, J. (1994), 'A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates', *Mol. Biol.* **11**, 459–468.

Lin, B., Sturmfels, B., Tang, X. & Yoshida, R. (2017), 'Convexity in tree spaces', *SIAM J. Discrete Math.* **31**(3), 2015–2038.

Liu, Z., Zhan, S. H. & Munzner, T. (2019), 'Aggregated dendrograms for visual comparison between many phylogenetic trees', *IEEE Trans. Visual Comput. Graphics.* **26**(9), 2732–2747.

Lubiw, A., Maftuleac, D. & Owen, M. (2020), 'Shortest paths and convex hulls in 2d complexes with non-positive curvature', *Comput. Geom.* **89**, 101626.

Miller, E., Owen, M. & Provan, J. S. (2015), 'Polyhedral computational geometry for averaging metric phylogenetic trees', *Adv. Appl. Math.* **68**, 51–91.

Owen, M. & Provan, J. S. (2011), 'A fast algorithm for computing geodesic distances in tree space', *IEEE/ACM Trans. Comput. Biol.* **8**, 2–13.

Robinson, D. F. (1971), 'Comparison of labeled trees with valency three', *J. Comb.* **11**, 105–119.

Robinson, D. F. & Foulds, L. R. (1979), 'Comparison of weighted labelled trees', *Lect. Notes Math.* **748**, 119–126.

Robinson, D. F. & Foulds, L. R. (1981), 'Comparison of phylogenetic trees', *Math. Biosci.* **7**, 1–8.

Ruane, S., Bryson, R. W. J., Pyron, R. A. & Burbrink, F. T. (2014), 'Coalescent species delimitation in milksnakes (genus lampropeltis) and impacts on phylogenetic comparative analyses.', *Syst. Biol.* **63**(2), 231–250.

Saitou, N. & Nei, M. (1987), 'The neighbor-joining method: a new method for reconstructing phylogenetic trees.', *Mol. Biol. Evol.* **4**(4), 406–425.

Sanderson, M. J., McMahon, M. M., Stamatakis, A., Zwickl, D. J. & Steel, M. (2015), 'Impacts of terraces on phylogenetic inference', *Syst. Biol.* **64**(5), 709–726.

Sanderson, M. J., McMahon, M. M. & Steel, M. (2011), 'Terraces in phylogenetic tree space', *Science.* **333**(6041), 448–450.

Smith, M. R. (2022), 'Robust analysis of phylogenetic tree space', *Syst. Biol* .

Stamatakis, A., Ludwig, T. & Meier, H. (2005), 'Raxml-iii: a fast program for maximum likelihood-based inference of large phylogenetic trees', *Bioinformatics.* **21**(4), 456–463.

Sukumaran, J. & Holder, M. T. (2010), 'Dendropy: a python library for phylogenetic computing.', *Bioinformatics.* **26**(12), 1569–1571.

Swofford, D. (2003), 'PAUP*. phylogenetic analysis using parsimony (*and other methods). version 4.'.

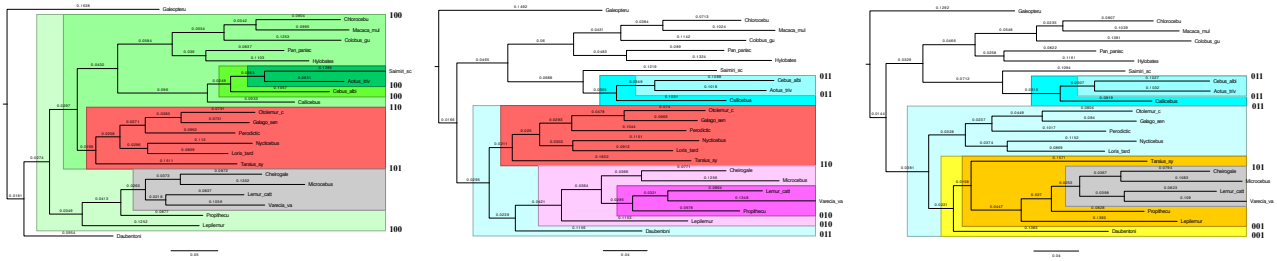
Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P. & SciPy 1.0 Contributors (2020), 'SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python', *Nat. Methods.* **17**, 261–272.

Whidden, C. & Matsen IV, F. A. (2015), 'Quantifying mcmc exploration of phylogenetic tree space', *Syst. Biol.* **64**(3), 472–491.

APPENDIX

Trees for the Primate Dataset D_1

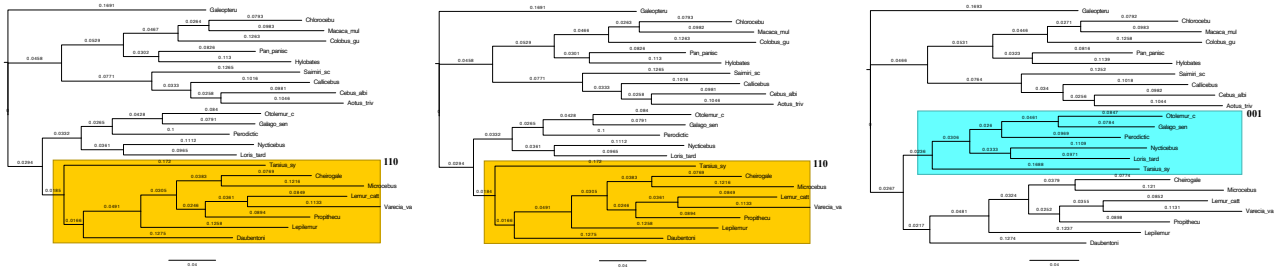
For Fig. 3 we used the three trees shown in Fig. 8.



(a) tree ▲ (b) tree ▲ (c) tree ▲

Figure 8. The three trees used in Fig. 3. Blue spectrum (011) shows subtrees in trees (b) and (c) and not in tree (a). Red color (110) shows the subtree in trees (a) and (b) and not in the tree (c). Grey color (101) shows the subtree in trees (a) and (c) and not in the tree (b). Green (100), pink (010), and yellow (001) spectrums show the subtrees just in tree (a), tree (b), and tree (c), respectively.

For the first dataset, we used RevBayes to generate a set of starting trees and then searched for the best tree using Pathtrees. We compared our best tree with the maximum posterior tree (MAP) of RevBayes and the best tree found by Paup* using a heuristic search. As shown in Fig. 9, the best Pathtrees tree and the Paup* tree are the same, whereas the MAP tree differs from both the Pathtrees tree and the Paup* tree by two splits. These trees are shown on the likelihood landscape in Fig. 5 and Fig. 6.

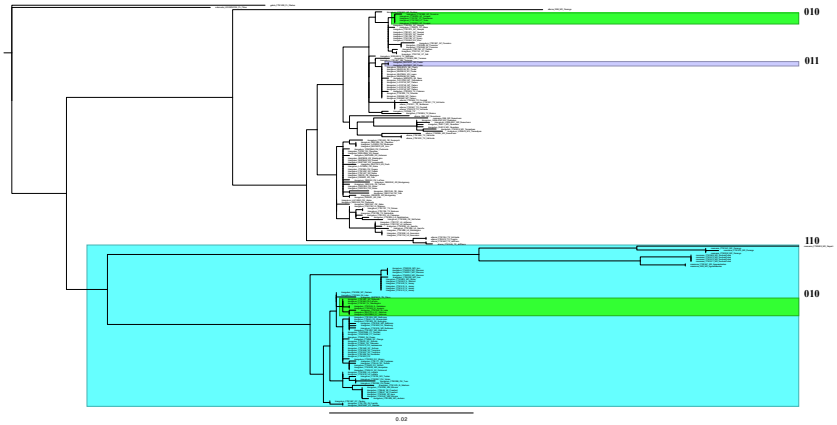


(a) Best tree by Pathtrees (b) Best tree by Paup* (c) Best tree by RevBayes

Figure 9. Best trees observed by Pathtrees (a), Paup* (b), and RevBayes (c) for the first dataset. Yellow color (110) shows subtrees in the best trees found by Pathtrees and Paup* and not in the best tree of RevBayes. Blue color (001) shows the subtrees just in the best tree of RevBayes.

572 *Trees for the Snake Dataset D_2*

(a) Best tree by PATHTREES



(b) Best tree by PAUP*

(c) Best tree by RAxML

Figure 10. Best trees observed by PATHTREES (a), PAUP* (b), and RAxML (c) for the second dataset. Blue color (110) shows the subtree in the best trees found by PATHTREES and PAUP* and not in the best tree of RAxML. Purple color (011) shows the subtree in the best trees found by PAUP* and RAxML and not in the best tree of PATHTREES. Pink color (100), green color (010), and yellow spectrum (001) show the subtrees just in the best tree of PATHTREES, PAUP*, and RAxML, respectively.

Effect of Interpolation Method on Visualization

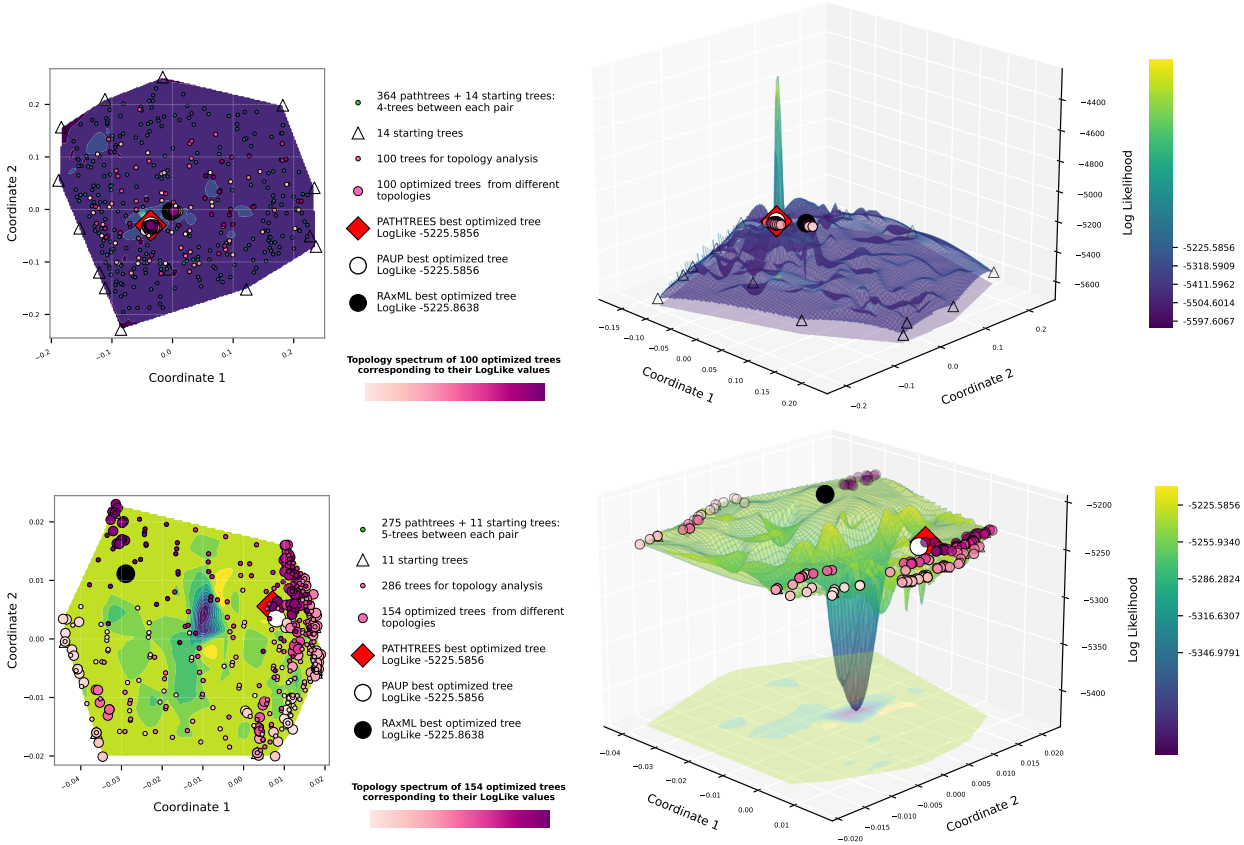


Figure 11. Contour and surface plots of PATHTREES for the dataset D_2 , using MDS and cubic spline interpolation defined by the weighted Robinson-Foulds distance metric. First row: the first iteration of PATHTREES using 14 starting trees. Second row: the second iteration of PATHTREES displaying the treespace after zooming inside the convex hull of 100 optimized trees from the first iteration.

Our package PATHTREES uses two different interpolation methods for the likelihood contour and surface. The default interpolation method is the RBF thin-plate spline with smoothness parameter of $s = 1e - 10$. The cubic spline interpolation can be used by setting the option `-interpolate cubic`, and the RBF thin-plate spline with any value of smoothness s can be used by setting the option `-interpolate rbf,s`. The thin-plate spline delivers surfaces that are less noisy. For example, comparing Fig. 7 (thin-plate spline) and Fig. 11 (cubic spline), the overall impression of the contour and surface features are similar but the cubic spline interpolation on the top row shows more extreme peaks than the thin-plate spline interpolation, and the distribution of the range of

observed log-likelihood values on the color bar shows this clearly. We have a good evidence that the optimal trees of PAUP* and PATHTREES are on likelihood peaks.

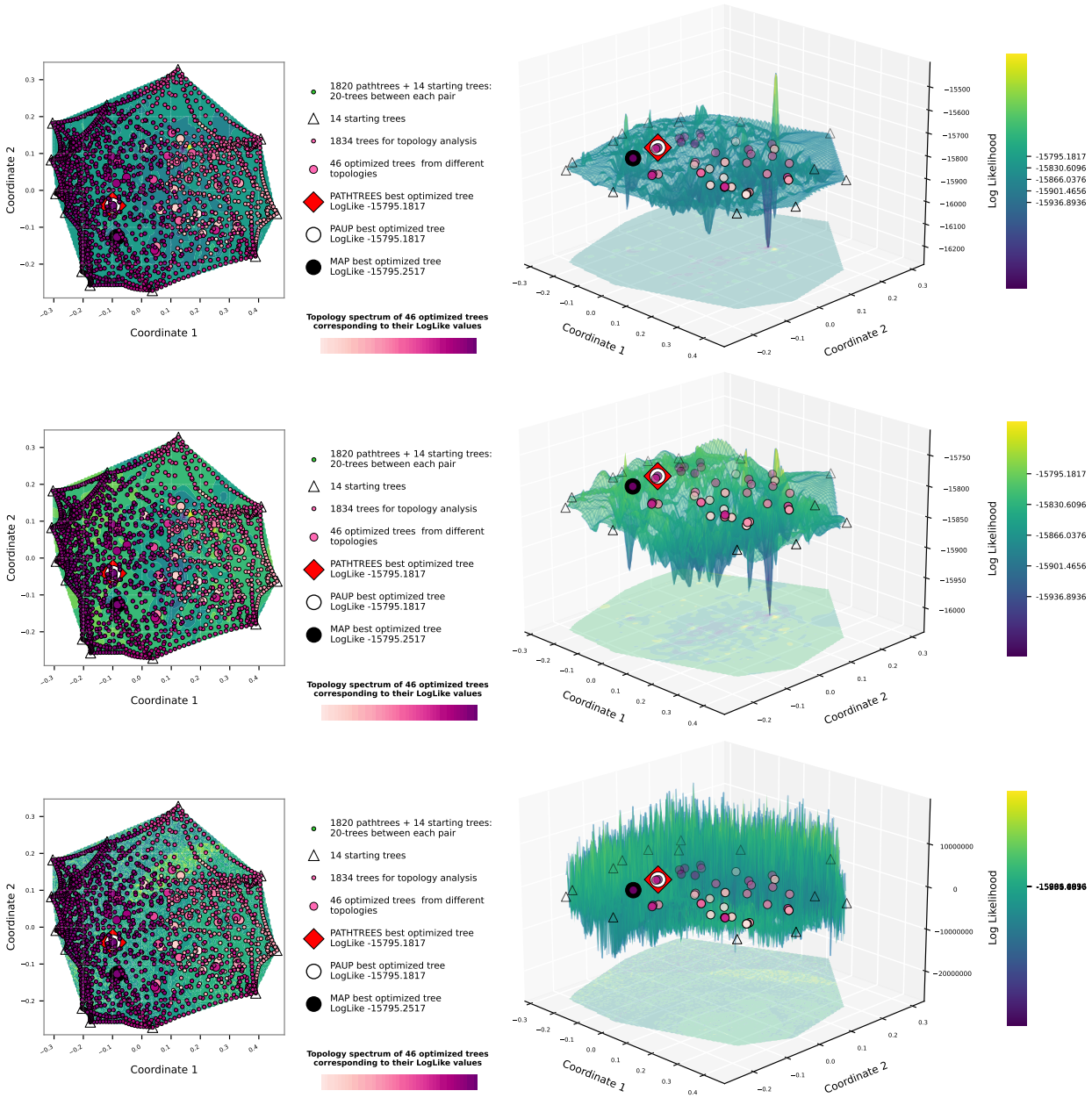


Figure 12. Contour and surface plots of PATHTREES for dataset D_1 by generating 20 pathtrees per anchor tree pair, using MDS, and (top row) cubic spline interpolation, (middle row) thin-plate spline with smoothness parameter of $s = 1e - 10$, (bottom row) thin-plate spline with smoothness parameter of $s = 0.0$, defined by the weighted Robinson-Foulds distance metric.

Therefore, the RBF thin-plate spline interpolation gives a better impression of the

true likelihood surface. Interpolation can fail if many trees are mapped close together in the MDS plot but have sufficiently different likelihoods. RBF will fail in such situations when the surface is forced through the observed data points or will give a singular matrix error (Fig. 12 bottom row). As a solution, increasing the smoothness parameter of the RBF interpolation from zero to a small value, such as 10^{-10} can lead to an acceptable visualization (Fig. 12 middle row) compared to the cubic spline interpolation ((Fig. 12 top row). The cubic spline interpolation exaggerates the surface compared to the RBF thin-plate spline interpolation.

We use MDS to place all trees onto a 2-D plane and then use the likelihood of the trees to interpolate the likelihood tree landscape. This visualization of the MDS plane may differ dependent on the trees used and once in a while may deliver plots that are difficult to interpret; Fig. 13 shows such a plot where the viewpoint of the plane shifted to be perpendicular to our original convex hull: we look at the convex hull sideways. In such cases, changing the number of optimized trees or the number of pathtrees between the anchor trees may help to remedy the view; for example, Fig. 7 (top row) shows an adequate visualization by using fewer number of optimized trees compared to Fig. 13.

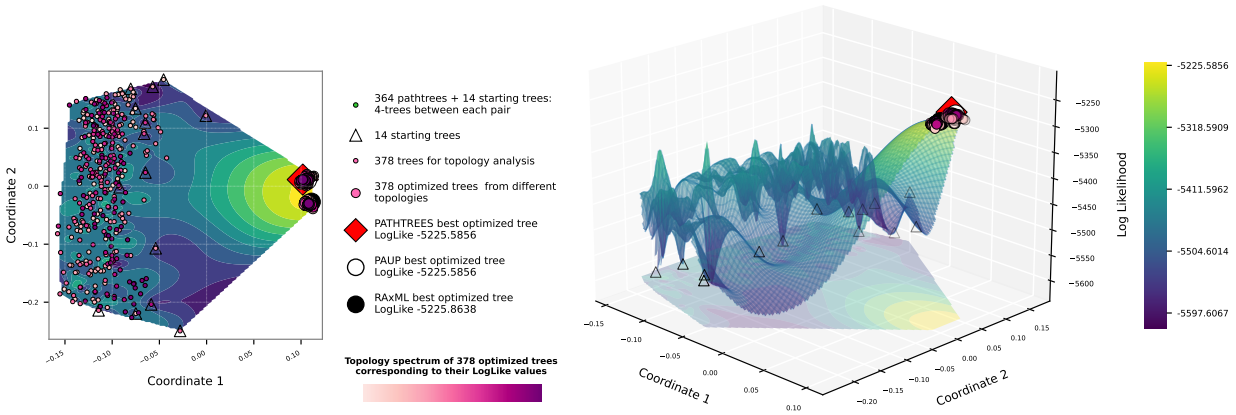


Figure 13. Contour and surface plots of PATHTREES for dataset D_2 , using MDS and thin-plate spline interpolation defined by the weighted Robinson-Foulds distance metric. Four trees were generated on the geodesic of each pair of 14 starting trees (364 pathtrees). All 364 + 14 trees were selected to be optimized. All trees have different topologies.

RevBayes *Script to Generate Start Trees*

This script is a shortened version of the tutorial page (Höhna et al. 2021)

```
################################################################################
#
# RevBayes Example: Bayesian inference of phylogeny using a Jukes-Cantor
#             substitution model on a single gene.
#
# authors: Sebastian Hoehna, Michael Landis, and Tracy A. Heath
# modified by Peter Beerli 2021
#
################################################################################


### Read in sequence data for both genes
data = readDiscreteCharacterData("data/primates_and_galeopterus_cytb.nex")

# Get some useful variables from the data. We need these later on.
n_species <- data.ntaxa()
n_branches <- 2 * n_species - 3
taxa <- data.taxa()


mvi = 1
mni = 1


######################
# Substitution Model #
######################

# create a constant variable for the rate matrix
Q <- fnJC(4)


##############
# Tree model #
##############

out_group = clade("Galeopterus_variegatus")
# Prior distribution on the tree topology
topology ~ dnUniformTopology(taxa, outgroup=out_group)
moves[mvi++] = mvNNI(topology, weight=5.0)
moves[mvi++] = mvSPR(topology, weight=1.0)

# Branch length prior
for (i in 1:n_branches) {
    bl[i] ~ dnExponential(10.0)
    moves[mvi++] = mvScale(bl[i])
}

TL := sum(bl)

psi := treeAssembly(topology, bl)


```

```
####################
# PhyloCTMC Model #
####################

# the sequence evolution model
seq ~ dnPhyloCTMC(tree=psi, Q=Q, type="DNA")

# attach the data
seq.clamp(data)


############
# Analysis #
############

mymodel = model(psi)

# add monitors
monitors[mni++] = mnScreen(TL, printgen=1000)
monitors[mni++] = mnFile(psi, filename="output/primates_cytb_JC.trees", printgen=10)
monitors[mni++] = mnModel(filename="output/primates_cytb_JC.log", printgen=10)

# run the analysis
mymcmc = mcmc(mymodel, moves, monitors)
#mymcmc.burnin(10000,200)
mymcmc.run(generations=500000)



####################
# Post processing #
####################

# Now, we will analyze the tree output.
# Let us start by reading in the tree trace
treetrace = readTreeTrace("output/primates_cytb_JC.trees", treetype="non-clock")
# and then get the MAP tree
map_tree = mapTree(treetrace,"output/primates_cytb_JC_MAP.tree")


# you may want to quit RevBayes now
q()
```