International Cartographic Association
Association Cartographique Internationale

Flexible Algorithms for Visualizing Geophylogenies

Thomas C. van Dijk*, Bettina Speckmann, Edwin Steenkamer

TU Eindhoven - t.c.v.dijk@tue.nl, b.speckmann@tue.nl, e.j.steenkamer@student.tue.nl

* Corresponding Author

Abstract: *Geophylogenies* enrich the leaves (species) of a phylogenetic tree with geographic locations (sites) that are biologically meaningful, such as sightings, habitats, or fossil finds. To extract geographic patterns from this data, geophylogenies are customarily visualized using a map to indicate the locations, a tree drawing for the phylogenetic tree, and some sort of linking mechanism, such as labels or leaders, which matches leaves and locations. So far, such geophylogeny visualizations are mostly created without sophisticated algorithmic assistance. An exception is recent work by Klawitter et al. (2023) which focusses mainly on point sites as geographic locations, tree drawings with fixed leaf positions on a line, and linking via leaders. In this paper we significantly extend their work by adding additional flexibility to all components of the visualization pipeline: we support not only point, but also region sites, the locations of leaves are adaptive to the data and can lie on either a line or a circle, and the locations are visually linked to leaves without the need for explicit connections via leaders. We implemented our algorithms and evaluated them experimentally. Our results show that the added flexibility indeed results in visualizations of higher quality for datasets of up to medium size; for large datasets the leaves of the phylogenetic tree tend to be so crowded around the map that the difference between adaptive and fixed leaf positions becomes negligible. However, the possibility of placing leaves on a circle and the linking without leaders still improve the readability of our visualizations when compared Klawitter et al. (2023).

Keywords: geophylogeny, algorithm, map design, geovisualization

1. Introduction

Phylogenetics studies the evolutionary history of species and organisms to gain a better understanding of the evolution of life. This evolution of species is often visualized as a *phylogenetic tree*, also called a *phylogeny* (Baum et al., 2008). The leaves of this tree represent the most modern species and internal nodes correspond to speciation events where a single species splits into multiple distinct species.

Geography can play a role in such speciation events and Soininen et al. (2007) suggest that Tobler's first law of geography[1] also holds for biological systems. A *phylogeographic tree* (or: *geophylogeny*) enriches the leaves of a phylogenetic tree with geographic data (*sites*) that are biologically meaningful for the corresponding species, such as fossil finds, observations, or known habitats. Geophylogenies are often visualized using a map R which depicts the set $\mathscr{P}$ of (point or area) sites, a tree drawing of the corresponding phylogenetic tree T, and some form of linking between the leaves and their sites. See Figures 1-3 for examples from the biological literature; leaders as well as labels and colours are used to link leaves to sites.

There are different ways in which the tree T is drawn alongside R. Most commonly, the leaves of T are projected on a boundary of R (see Figures 1 and 2) and the tree is drawn orthogonally as a *rectangular cladogram*. But there are also examples of circular drawings, referred to as *inner-circular cladograms*, where the leaves of T are arranged on a circle around R; see Figure 3. The tree T is always drawn in a planar manner, without crossings. The leaves of T can hence be arranged only in those orders which are compatible with the structure of T.

The biggest challenge when visualizing geophylogenies lies in the linking of leaves to sites. Explicit linking via leaders is unambiguous but has the disadvantage that the leaders obscure parts of the map R and potentially also other sites. Furthermore, since the positions of the leaves must respect the tree structure, it often not possible to find an order of the leaves such that the leaders do not cross. Linking with leaders is hence not a viable option when there are more than a handful of leaves and sites.

Implicit linking commonly uses colours or labels to match sites to leaves. For the reader to make the correct connec-

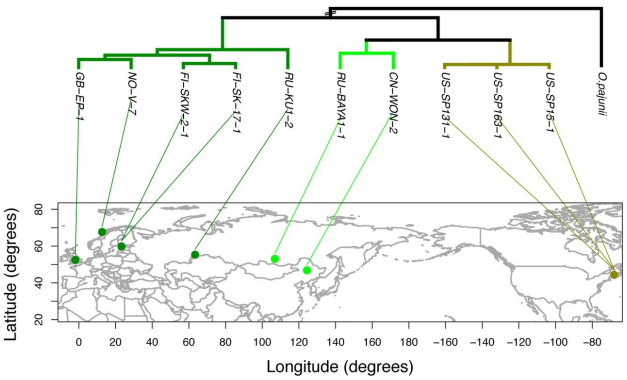


Figure 1. Geophylogeny with leaders (Angst et al., 2023).

[1] "Everything is related to everything else, but near things are more related than distant things." Tobler (1970)

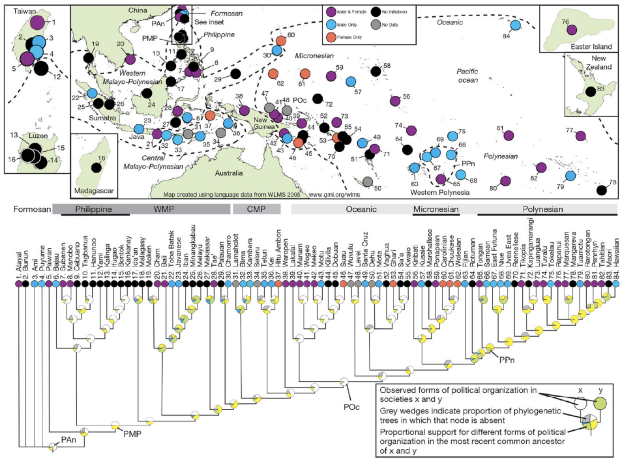


Figure 2. Implicit geophylogeny (Bentley et al., 2021).

tions, leaves need to be placed in close proximity to the sites. That is, the order of the leaves should match the spatial distribution of the sites well. See, for example, Figure 2: the left-to-right order of the leaves matches the order of the sites fairly closely and hence the association between leaves and sites is strong. In the remainder of this paper we focus on implicit linking with colours.

Definitions and Notation. We largely follow the definitions and notation introduced by Klawitter et al. (2023). A rooted binary tree (the phylogenetic tree) T has vertex set $V(T)$ and a set of n leaves $L(T) = \{l_1, \ldots, l_n\} \subset V(T)$. We denote by $T(v)$ the subtree rooted at vertex v and by $n(v) = |L(T(v))|$ the number of leaves in $T(v)$. An *embedding* of T is defined by the left-to-right order of the children of each internal vertex. T is always drawn planar, hence an embedding of T uniquely specifies a left-to-right or circular order of the leaves $L(T)$.

Let $\mathscr{P} = \{p_1 \ldots, p_n\}$ be a set of n geographic point or polygon sites. All sites are contained in a bounding area B, which can either be an axis-aligned rectangle or a disk. If B is a rectangle, then T is drawn as a *rectangular cladogram*. That is, all leaves are drawn on the upper *linear boundary* of B, at the same y-coordinate, and T is drawn *downward planar*: all edges of T point towards the leaves and no two edges cross. When B is a disk, then T is drawn as a planar *inner-circular cladogram*, that is, all edges of T point towards the center of B and no two edges cross; all leaves are placed on the *circular boundary* of the disk.

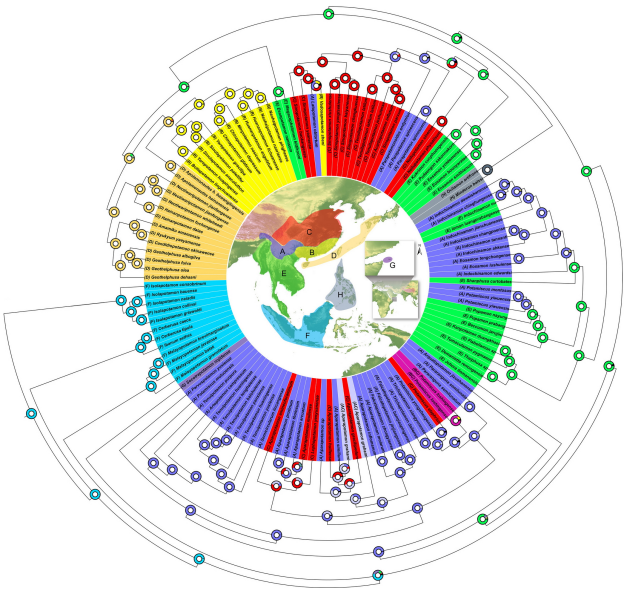


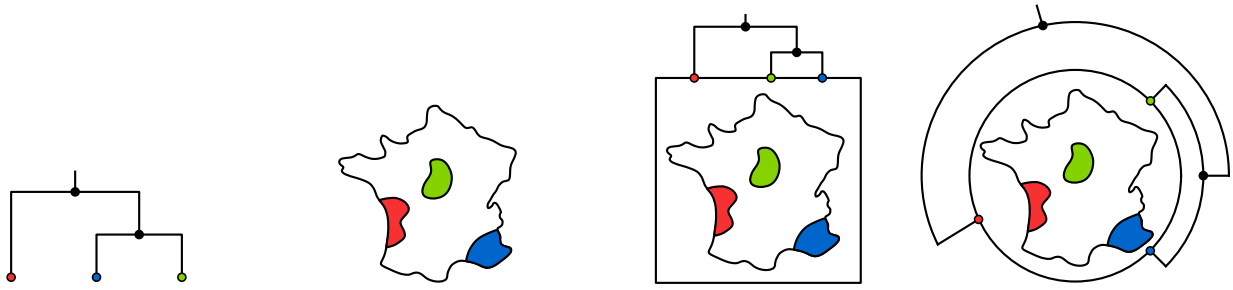
Figure 3. Inner-circular geophylogeny (Pan et al., 2022).

A *geophylogeny* G then consists of a phylogenetic tree $T(G)$, a boundary, a set of sites $\mathscr{P}(G)$, and a one-to-one mapping between $L(T(G))$ and $\mathscr{P}(G)$. The indices indicate this mapping, so $l_i \leftrightarrow p_i$, for all $i \in \{1, .., n\}$. For ease of notation, we will use T and $\mathscr{P}$ instead of $T(G)$ and $\mathscr{P}(G)$.

Problem Statement. Given a geophylogeny G our goal is to construct an optimal drawing Γ of G that visualizes T, R, $\mathscr{P}$, and the mapping between $L(T)$ and $\mathscr{P}$. Γ must satisfy the following requirements: (1) T is drawn either as a rectangular or an inner-circle cladogram. (2) The leaves of T are placed on a linear or circular boundary. (3) The mapping between $L(T)$ and $\mathscr{P}$ is indicated by colour. (4) The order and position of the leaves should match the spatial distribution of $\mathscr{P}$ in such a way that the mapping between $L(T)$ and $\mathscr{P}$ is visually as clear as possible. Figure 4 shows a simplified example of an input with two possible outputs.

Contribution and organization. In this paper we present three algorithms that solve the problem of visualizing geophylogenies. We introduce added flexibility with respect to previous work by Klawitter et al. (2023); this flexibility allows us to explore a greater range of possible solutions and to often find visualizations of higher quality.

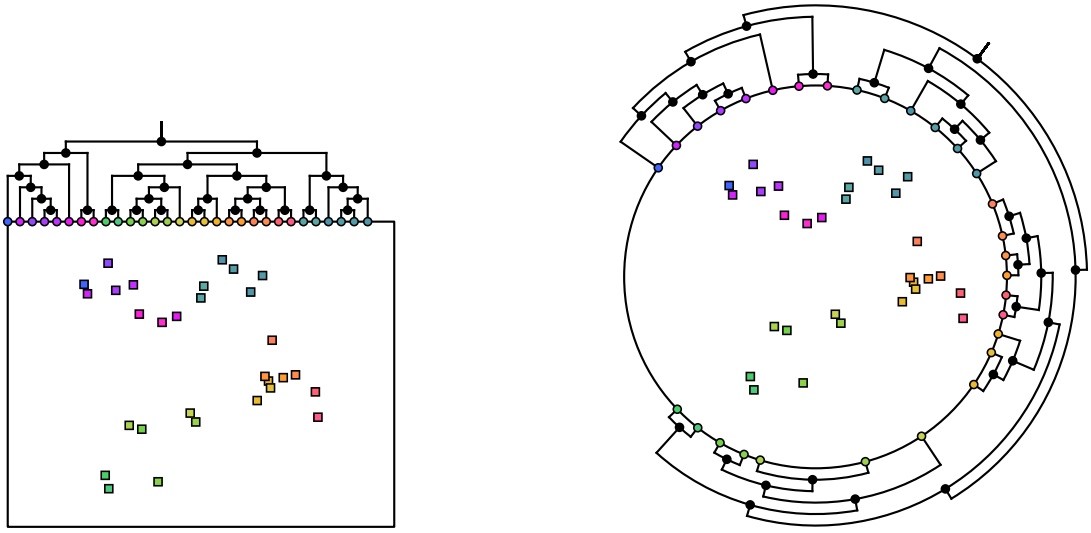
Specifically, in Section 3 we show how to compute inner-circular geophylogenies. Figure 5 illustrates that the added



(a) Input phylogeny (b) Input sites (c) Rectangular geophylogeny (d) Circular geophylogeny

Figure 4. Example input (a) and (b), and possible outputs (c) or (d).

Advances in Cartography and GIScience of the International Cartographic Association, 5, 34, 2025.
32nd International Cartographic Conference (ICC 2025), 17–22 August 2025, Vancouver, Canada. This contribution underwent double-blind peer review based on the full paper. https://doi.org/10.5194/ica-adv-5-34-2025 | © Author(s) 2025. CC BY 4.0 License



(a) Rectangular cladogram.

(b) Inner-circular cladogram.

Figure 5. Synthetic *Clustered* instance: an inner-circular cladogram has better association between leaves and sites.

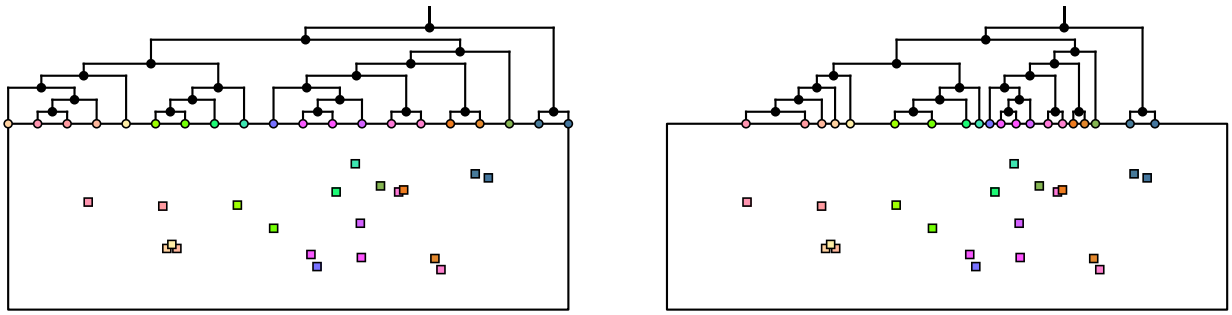
flexibility of approaching the map from all sides can greatly increase the quality of the drawing. In Section 4 we then show how to compute visualizations also for polygonal sites. Inspired by algorithms for necklace maps (Speckmann and Verbeek, 2010), we introduce adaptive leaf positions in Section 5. Figure 6 illustrates the improvements to the drawing given the added flexibility for the tree. In Section 6 we report on experimental results; generally we can conclude that our flexible algorithms compute visualizations with a better association between leaves and sites than previous work.

2. Related Work

The most well-known software for generating geophylogenies is the `phytools` R package by Revell (2024). It provides a convenient way to correctly draw geophylogenies in a variety of styles, but it does not provide proper optimization of the embedding of the tree: the package either chooses some random embeddings or leaves it to the user to provide one. Klawitter et al. (2023) performed the first systematic study of geophylogenies from an algorithmic perspective. They formalized the problem and mainly focussed on minimizing leader crossings. Tanglegrams are another tree embedding problem with applications in biology; optimizing one-sided tanglegrams (Fernau et al., 2010) is most closely related to our work.

See Figures 1–3 for several examples of geophylogenies from the biological literature. In general, both rectangular cladograms (Poczai et al., 2011, Jauss et al., 2021) and rectangular phylograms (Mehraban et al., 2020, Pham et al., 2017) can be found; the latter do not draw the leaves at the same height. Pan et al. (2022) use a circular cladogram; they also use feature areas instead of point sites. Another approach found in the literature is to overlay the tree directly onto the map, placing the leaves at the sites. Although this approach results in unambiguous linking, the underlying map is occluded and Page (2015) indicates that the tree itself becomes hard to read.

Most geophylogenies we have found in the literature use point sites. Arguably, polygonal sites can capture complex spatial distributions of species better and at the same time afford greater freedom to link visually. In fact, the literature on boundary labelling shows that the additional flexibility created by polygonal sites can lead to better maps (Bekos et al., 2010). Hence we hope that our results for polygonal sites will promote their use in the future.

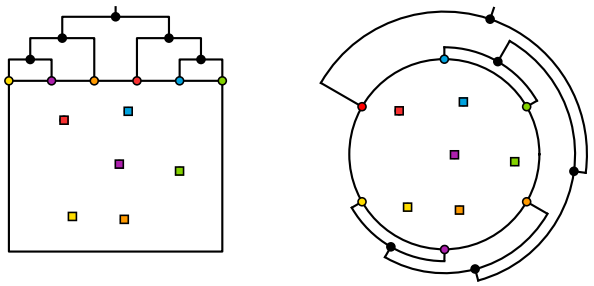


(a) Fixed leaf positions are spread out evenly and do not match the geography.

(b) Adaptive leaf positions reflect the geography of the sites more clearly.

Figure 6. Geophylogeny of European green lizards; data from Jauss et al. (2021).

Advances in Cartography and GIScience of the International Cartographic Association, 5, 34, 2025.
32nd International Cartographic Conference (ICC 2025), 17–22 August 2025, Vancouver, Canada. This contribution underwent double-blind peer review based on the full paper. https://doi.org/10.5194/ica-adv-5-34-2025 | © Author(s) 2025. CC BY 4.0 License



(a) Rectangular boundary. (b) Circular boundary.

Figure 7. Fixed uniformly spaced leaf positions.

3. Circular Boundary

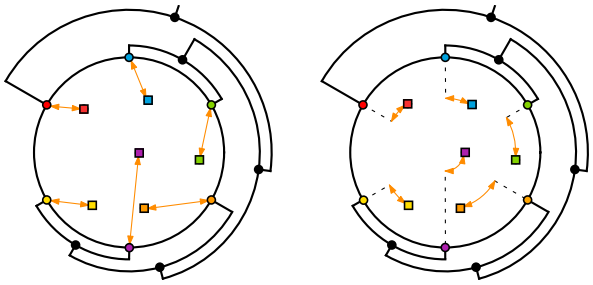
The algorithms of Klawitter et al. (2023) label geophylogenies by putting the leaves evenly spaced on one side of a rectangular boundary (Figure 7a). Without significant modification, their dynamic programming algorithm can already handle arbitrarily spaced candidate positions for the leaves, as long as those positions are fixed and totally ordered (in their case: left-to-right). We now extend the algorithm to handle circularly ordered candidate positions, which will allow us to draw inner-circular geophylogenies (Figure 7b).

3.1 Quality Measures

The overall goal when placing the leaves of the tree is to clearly communicate the correspondence to the sites on the map. Generally speaking, this means that we would want a site p_i and its corresponding leaf l_i to be close together. For linear boundaries, Klawitter et al. defined *Distance*, which sums the Euclidean distance of each pair; *XOffset*, which sums the horizontal distance of each pair, and *IndexOffset*, which sums how many steps away each pair is the left-to-right permutation. These quality measures are referred to as *leaf-additive* since they simply sum the individual quality of each leaf, where the quality of the leaf is determined only by its own position and the position of its site, and is independent of the other leaves. The algorithm uses a number of candidate positions that matches the number of leaves of T, so none are left empty and the quality of a drawing is determined solely by the embedding of T.

For circular drawings, the quality measure *Distance* can be adopted without change (Figure 8a). Rather than the horizontal distance of *XOffset*, we will use circular arc length and call it *ROffset*: $\theta(p_i, l_i) \cdot d(c, l_i)$, where c is the center and $\theta(p_i, l_i)$ denotes the radial angle between p_i and l_i with respect to c. This has the effect of weighing the contribution of the sites by their distance from the center, which is appropriate since a particular difference in angle is more pronounced near the boundary: see Figure 8b.

Adopting *IndexOffset* is conceptually reasonable, by having it consider how much the leaves permute the cyclic order of the sites with respect to c. However, this is not a leaf-additive quality measure: just the permutation of the leafs no longer completely determines the drawing, as it did for a linear boundary, since we can now shift all labels along the circle. Other leaves can influence which shift is best, so we cannot tell in advance what the quality is of putting a particular leaf at a particular position. Hence we do not consider *IndexOffset* for circular drawings.



(a) *Distance* (b) *ROffset*

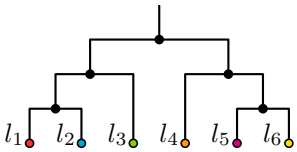
Figure 8. The orange segments indicate the distances being summed for the quality measures.

3.2 Algorithm

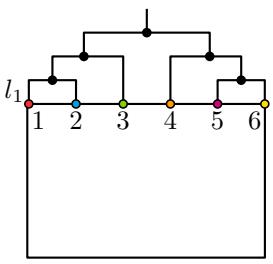
Here we briefly sketch Klawitter et al.'s algorithm for linear boundaries and observe that the circular solution can be found using the same techniques.

Note that the leaves of any subtree are placed at consecutive candidate positions, otherwise drawing the tree would require crossings. We can thus say that a subtree is placed at a particular position: put its leftmost leaf there. Since we use a leaf-additive objective function, a subtree at a particular position has a well-defined best embedding regardless of the rest of the tree. Let $F(v, i)$ be the minimum cost of embedding $T(v)$ with its leftmost leaf placed at position i. Klawitter et al. give a recurrence for F and get an optimal embedding for T by evaluating $F(\text{root}(T), 1)$: the best way to embed the entire tree starting at the leftmost position. Dynamic programming yields a runtime of $O(n^2)$.

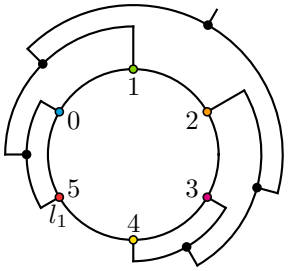
Their recurrence only considers placing subtrees that fit on the boundary. For a circular boundary, blocks of consecutive leaves can "wrap around" index space: see e.g. Fig-



(a) Embedding with l_1 as left-most leaf.



(b) l_1 is placed at position 1.



(c) l_1 is placed at position 5.

Figure 9. The embedding in (a) uniquely gives drawing (b) on a linear boundary. On a circular boundary, there are n cyclic shifts, such as the one in (c).

Advances in Cartography and GIScience of the International Cartographic Association, 5, 34, 2025.
32nd International Cartographic Conference (ICC 2025), 17–22 August 2025, Vancouver, Canada. This contribution underwent double-blind peer review based on the full paper. https://doi.org/10.5194/ica-adv-5-34-2025 | © Author(s) 2025. CC BY 4.0 License

ure 9, where l_1 and l_2 are consecutive at positions 5 and 0. (For notational convenience, we start counting circular positions at 0 instead of 1.)

Theorem 1. *Consider a geophylogeny with n leaves and let f be a leaf additive objective function. A drawing with circular boundary of that minimizes (or maximizes) f can be computed in $O(n^2)$ time.*

Proof. Adapting the original recurrence, where v is a vertex with children v_1 and v_2, n is the number of leaves in T, and $n(v)$ is the number of leaves in $T(v)$, we know the following: either $T(v_1)$ comes first in clockwise order, or $T(v_2)$ comes first, with the other following immediately.

$$F(v,i) = \min\{\, F(v_1,i) + F(v_2,(i+n(v_1)) \bmod n), \\ F(v_2,i) + F(v_1,(i+n(v_2)) \bmod n) \,\} \quad (1)$$

Computing F for all vertices and all positions can be done in $O(n^2)$ time by the same arguments as in the original paper. For a linear boundary, the optimum was given by $F(\text{root}(T),1)$, since the tree must start at the leftmost position; we must consider the minimum over all cyclic shifts by taking $\min_{i\in[0,n-1]} F(\text{root}(T),i)$, but this does not influence the asymptotic runtime. □

4. Polygonal Sites

Our second generalization to the drawing style is to consider instances where the geographic information associated with a leaf is a polygon instead of a single point. One way to handle this is to reduce the polygon to a representative point, such as the centroid or the geodesic center. However, this may lead to worse results, as the actual shape of a polygon can potentially provide useful freedom for good leaf placement.

As quality measure, we stick to *Distance* and interpret it as follows: Euclidean distance from the leaf to the closest point of the polygon; see Figure 10. We provide an efficient preprocessing procedure for the regions that allows *Distance* to be queried efficiently from the boundary and enables the algorithm from Section 3 to handle polygonal regions at no asymptotic cost (except when the regions are exceedingly detailed).

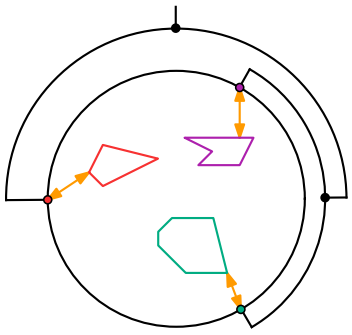


Figure 10. Distance of three uniformly spaced leaves to the corresponding polygonal site.

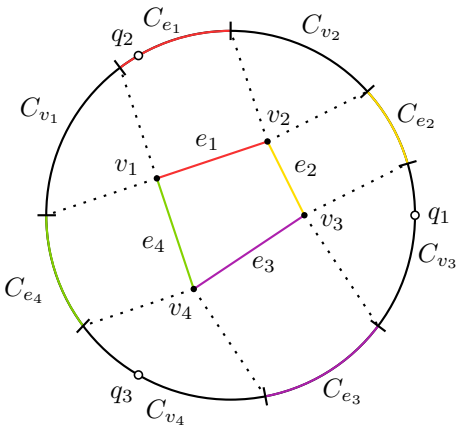


Figure 11. Circular boundary divided into intervals by the Voronoi diagram of a polygon. Point q_1 lies in the interval C_{v_3}, so the distance from q_1 to the polygon is realised between q_1 and v_3; between q_2 and e_1, and q_3 and v_4.

Theorem 2. *Consider a polygonal site P with k vertices, and a linear or a circular boundary. After $O(k\log k)$ preprocessing time, we can query: the distance of an arbitrary position on the boundary in $O(\log k)$ time; the distances of an ordered list Q of arbitrary positions on the boundary in $O(k+|Q|)$ time. The derivative of the distance at the position(s) can be reported in the same time.*

Proof. First, compute the Voronoi diagram of the vertices and edges of P. This can be done in $O(k\log k)$ time using the algorithm of Yap (1987) and subdivides the plane into $O(k)$ convex regions defined by being closest to a particular element of P (a vertex or an edge); see Figure 11. Then in $O(k)$ time we find the intersections between the edges of the Voronoi diagram and the boundary, in sorted order along the boundary.[2] This cuts the boundary into $O(k)$ intervals, which we store in a sorted list, along with which element of P that interval is closest to.

To answer a distance query for a point q on the boundary, find the interval that contains it in $O(\log k)$ time using binary search, then compute the distance to the appropriate element of P in constant time. The derivative along the boundary can also be computed analytically based on the shape of the boundary and the closest element.

To answer a distance query for a sorted list of such points, find the interval that contains the first point as before. Then traverse both the query list and the interval list simultaneously in the manner of a *merge*. This traverses the entire query list and, at most, the entire interval list, for a time bound of $O(k+|Q|)$. □

Corollary 3. *Geophylogenies with polygonal sites can be optimized in $O(n^2 + K\log K)$ time, where K is the total number of vertices in the polygons.*

[2] Let q be a point on the boundary. Use a point location data structure on the Voronoi diagram to find the cell that contains q. Test all edges of the cell against the boundary to find any intersection points; trace the boundary through such edges to the adjacent cell and continue. This process finds the intersections in sorted order and runs in $O(k)$ time since every edge of the Voronoi diagram is considered at most a constant number of times during this traversal.

Proof. For each polygon $P_i \in \mathscr{P}$, query the distance to all candidate leaf positions and construct a distance matrix: preprocess polygon P_i in $O(k_i \log k_i)$ time, where k_i is its number of vertices, then use a list query in $O(n+k_i)$ time. Doing this for all n polygons takes a total of $O(K \log K)$ for the preprocessing and $O(n^2+K)$ for the queries. (The k_i terms add up to K.) Then use the appropriate dynamic program depending on the boundary type; either takes $O(n^2)$ time using the distance matrix. □

5. Adaptive Leaf Positions

For our final generalization of the drawing style, we return to point sites but forgo preset candidate positions: rather than fixed uniform spacing, the leaves can be placed freely along the boundary. This flexibility can be used to better reflect the geography of the sites.

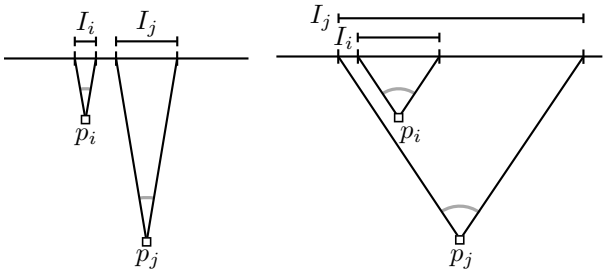
We give each leaf an *interval* on the boundary within which it can be placed. Conceptually, these intervals start as singleton points at the best possible position (that is, closest to the site) and grow until the tree can be drawn without crossings. We design the process by which the intervals grow so that sites close to the boundary have smaller intervals: it is visually most important that those leaves are near the site. For sites that are far from the boundary, we tolerate more discrepancy since the reader must make a significant visual jump in any case. See Figure 12.

5.1 Determining Placement Intervals

We formalize the case of a linear boundary; a circular boundary can be handled analogously. Let W_i be a wedge straight up from the site p_i; all wedges have the same angle α (value to be determined later). Let $I_i = [\ell_i, r_i]$ be the intersection of W_i with the boundary: this is where leaf l_i may be placed. Note that I_i is a function of α.

We say that $I_i < I_j$ if I_i is strictly to the left of I_j (that is: $r_i < \ell_j$). Then the leaf l_i must certainly be placed to the left of l_j when using this value of α. Let v be an internal vertex with children v_1 and v_2; let $l_a, l_b \in L(v_1)$ and $l_c, l_d \in L(v_2)$. If such leaves exist where $I_a < I_c$ and $I_d < I_b$, we say v has a *conflict* for this value of α.

To see why conflicts are bad, first note that the leaves under v_1 and the leaves under v_2 cannot be interleaved from left to right, or the tree would have crossings. However, a conflict



(a) Smaller value of α. (b) Larger value of α.

Figure 12. Site p_j is farther from the boundary than p_i, so its interval grows faster when increasing the wedge angle.

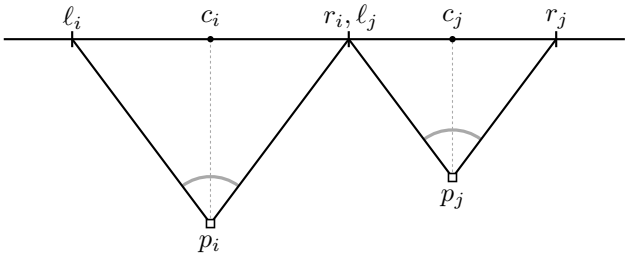


Figure 13. The angle $\alpha^*(i,j)$, which realizes $r_i = \ell_j$.

implies that the leaves *must* interleave, so there is no way to draw the tree that respects these placement intervals: α must be larger, so that at least $I_a \not< I_c$ or $I_d \not< I_b$.

Indeed, increasing α makes all intervals larger, which eventually resolves any conflict; call a value of α *feasible* if the tree has no conflicts. Still, in order to keep the leaves as close as possible to their ideal position, we will find the smallest feasible α.

Let $\alpha^*(i,j)$ be the minimum angle such that I_i and I_j touch; see Figure 13. This means that if $\alpha > \alpha^*(i,j)$, the pair l_i, l_j cannot contribute to a conflict.

Theorem 4. *Computing the minimum feasible α can be done in $O(n^2)$ time for a linear boundary and $O(n^3)$ for a circular boundary.*

Proof. Consider a pair of leaves (l_a, l_b) where p_a is left of p_b, and let v be their lowest common ancestor in T. Then $I_a < I_b$ if and only if $\alpha < \alpha^*(a,b)$, in which case these leaves could participate in a conflict at v (but not at any other vertex): it would force the subtree containing a to be the left subtree of v. Call this a *demand* on the embedding of v and note that v has a conflict if and only if it has at least one demand in each direction.

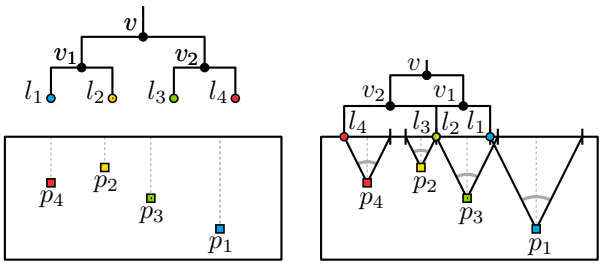
In $O(n^2)$ time, we mark each internal vertex with the demands on it, and which way around they want to embed the children of v; call these opposing sets A_v and B_v. For there to be no conflict at v, we must pick α such that at least one of A_v and B_v becomes empty. The smallest α that achieves this is

$$\alpha^*(v) = \min\{\max_{(i,j)\in A_v} \alpha^*(l_i,l_j), \max_{(i,j)\in B_v} \alpha^*(l_i,l_j)\}.$$

All conflicts are resolved if and only if $\alpha \geq \max_v \alpha^*(v)$, so that is in fact the minimum feasible α. The runtime of evaluating these expressions is $O(n^2)$ since every pair of leaves occurs in only one place.

Conflicts on a circular boundary are unwieldy, so for the algorithm we reduce to the linear-boundary case: cut the circle between two radially adjacent sites, trying all n options for where to cut. This adds an $O(n)$ factor to the runtime. □

Picking the minimum feasible α has the disadvantage that it leads to leaves placed on top of each other, since we stop at the exact moment the last conflict is resolved. In practice, it is advisable to pick α^* so that the intervals share at least a given amount of overlap; see Figure 15.



(a) Input tree and sites. (b) Minimum feasible α.

Figure 14. Internal node v has a conflict if $\alpha < \alpha^*(2,3)$.

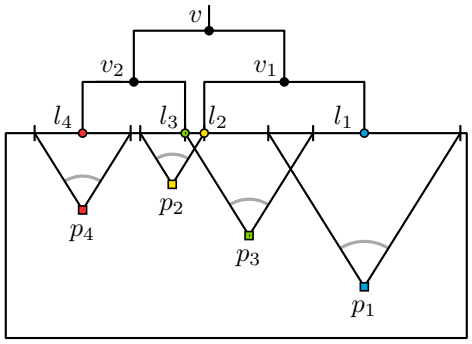


Figure 15. Drawing where α^* is modified to produce some minimum amount of overlap between intervals; notice l_2 and l_3, and compare to Figure 14b.

5.2 Placing the Leaves

Once a feasible α has been determined and a set of intervals without conflicts has thus been found, the sets A_v and B_v determine if v must be embedded one way or another (or is free to do either). The leaves can then be placed greedily in the order given by T, for example at their leftmost available position.

This places the leaves unnecessarily far to the left, so we then use the force-directed method of Speckmann and Verbeek (2010), attracting leaves to the middle of their interval while repelling each other, and making sure to keep the leaves in the same order and within their interval.

We note that the force-directed method can take any starting point of the leaves, including those that were computed with uniform leaf positions. Thus we can add adaptive leaves to algorithms that do not support it natively.

6. Experimental Evaluation

We have implemented the algorithms from Sections 3 and 5 and have run them on real-world and the synthetic instances. We have not implemented the algorithm with polygonal sites, since – although several biologists argued for this in personal communication – we found very few data sets that include polygonal sites and it is unclear what realistic instances would look like. Even though our code is not engineered for speed, all instances of reasonable size (up to a few hundred leaves) are solved in a fraction of a second.

We use the same process for generating random instances as Klawitter et al. (2023); see their paper for details. We focus on synthetic instances, rather than real instances, since

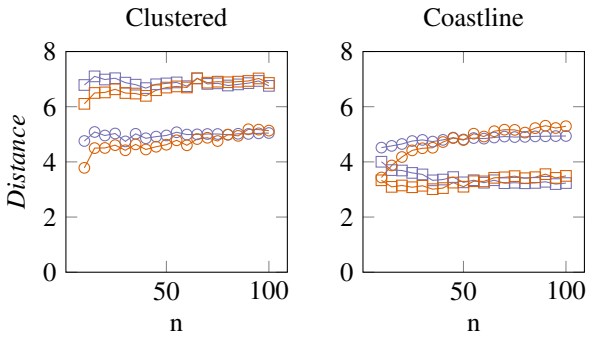


Figure 16. Average *Distance* of optimal solutions.

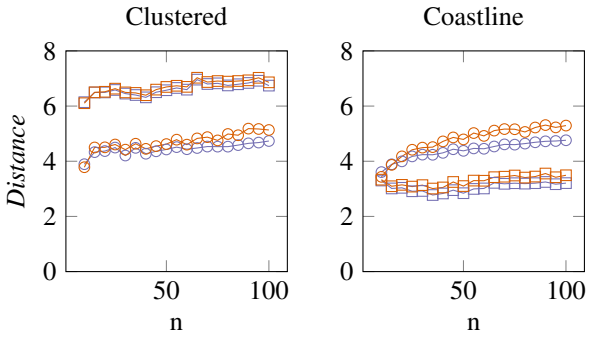


Figure 17. Average *Distance* of optimal solutions after force-directed postprocessing.

this allows us to make structural observations about what happens with an increasing the number of leaves. All plots use the following symbology, and averages are taken over 100 instances for each value of n.

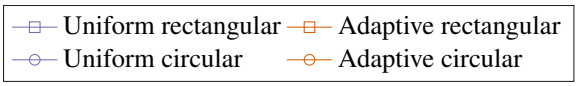


Figure 16 compares our various drawing styles by the average distance between leaves and the corresponding site. Note that a circular boundary significantly outperforms a linear boundary on Clustered instances (cf. Figure 5) and that adaptive leaves outperform uniform leaves for small n. The latter advantage disappears with increasing n, as the boundary becomes cluttered and the minimum spacing between the leaves pushes adaptive leaves to position them-

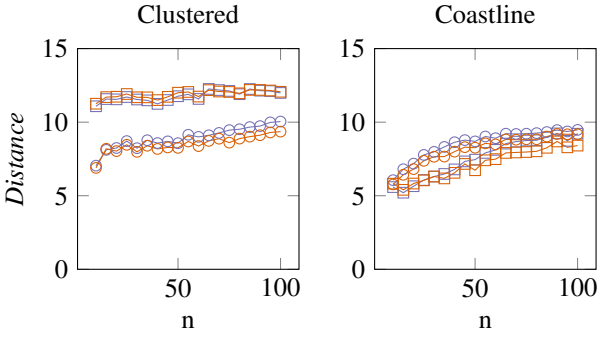


Figure 18. Maximum *Distance* of optimal solutions after force-directed postprocessing.

Advances in Cartography and GIScience of the International Cartographic Association, 5, 34, 2025.
32nd International Cartographic Conference (ICC 2025), 17–22 August 2025, Vancouver, Canada. This contribution underwent double-blind peer review based on the full paper. https://doi.org/10.5194/ica-adv-5-34-2025 | © Author(s) 2025. CC BY 4.0 License

selves uniformly. In contrast, we see that a linear boundary is preferred for Coastline instances, since this boundary shape better matches the geography; adaptive leaves on a circular boundary can cope when n is small, but end up having to take the entire circle for large n.

Figure 17 suggests that using force-directed postprocessing on a drawing computed with uniform leaf placement (Section 3) is just as good as directly optimizing adaptive leaves with the algorithm from Section 5; if anything, it works slightly better, possibly due to having found a more appropriate embedding of the tree. However, Figure 18 shows the *maximum* distance between a leaf and its site (instead of the average): here the advantage of specifically optimized adaptive leaves reappears, since the global optimization of the wedge angles ensures that every leaf receives a somewhat reasonable position.

7. Conclusion

In this paper we introduced three algorithms to more flexibly draw optimized geophylogenies: circular boundaries, polygonal sites, and adaptive leaf positions. Each in their own way increases the ability to reflect the geography of the sites and thus more clearly communicate the relation between the phylogenetic tree and the geography.

We experimentally studied the quality of the drawings produced by these algorithms and found that different styles and objective functions perform well on different kinds of geography, which illustrates the importance of flexible algorithms. We hope in particular that support for region sites may find use in practice.

References

Angst, P., Ebert, D. and Fields, P. D., 2023. Population genetic analysis of the microsporidium Ordospora colligata reveals the role of natural selection and phylogeography on its extremely compact and reduced genome. *G3 Genes|Genomes|Genetics* 13(3), pp. 1–11.

Baum, D. et al., 2008. Reading a phylogenetic tree: the meaning of monophyletic groups. *Nature Education* 1(1), pp. 190.

Bekos, M. A., Kaufmann, M., Potika, K. and Symvonis, A., 2010. Area-feature boundary labeling. *The Computer Journal* 53(6), pp. 827–841.

Bentley, R. A., Moritz, W. R., Ruck, D. J. and O'Brien, M. J., 2021. Evolution of initiation rites during the Austronesian dispersal. *Science Progress* 104(3), pp. 00368504211031364.

Fernau, H., Kaufmann, M. and Poths, M., 2010. Comparing trees via crossing minimization. *Journal of Computer and System Sciences* 76(7), pp. 593–608.

Jauss, R.-T., Solf, N., Kolora, S. R. R., Schaffer, S., Wolf, R., Henle, K., Fritz, U. and Schlegel, M., 2021. Mitogenome evolution in the Lacerta viridis complex (Lacertidae, Squamata) reveals phylogeny of diverging clades. *Systematics and Biodiversity* 19(7), pp. 682–692.

Klawitter, J., Klesen, F., Scholl, J. Y., van Dijk, T. C. and Zaft, A., 2023. Visualizing Geophylogenies - Internal and External Labeling with Phylogenetic Tree Constraints. In: *12th International Conference on Geographic Information Science (GIScience 2023)*, LIPIcs, Vol. 277, pp. 5:1–5:16.

Mehraban, H., Esmaeili, H. R., Zarei, F., Ebrahimi, M. and Gholamhosseini, A., 2020. Genetic diversification, population structure, and geophylogeny of the Scarface rockskipper Istiblennius pox (teleostei: Blenniidae) in the Persian Gulf and Oman Sea. *Marine Biodiversity* 50, pp. 1–12.

Page, R., 2015. Visualising geophylogenies in web maps using geojson. Public Library of Science, https://eprints.gla.ac.uk/109946/, pp. 1–5.

Pan, D., Shi, B., Du, S., Gu, T., Wang, R., Xing, Y., Zhang, Z., Chen, J., Cumberlidge, N. and Sun, H., 2022. Mitogenome phylogeny reveals indochina peninsula origin and spatiotemporal diversification of freshwater crabs (potamidae: Potamiscinae) in China. *Cladistics* 38(1), pp. 1–12.

Pham, K. K., Hipp, A. L., Manos, P. S. and Cronn, R. C., 2017. A time and a place for everything: phylogenetic history and geography as joint predictors of oak plastome phylogeny. *Genome* 60(9), pp. 720–732.

Poczai, P., Hyvönen, J. and Symon, D. E., 2011. Phylogeny of kangaroo apples (solanum subg. archaesolanum, solanaceae). *Molecular biology reports* 38, pp. 5243–5259.

Revell, L. J., 2024. phytools 2.0: an updated r ecosystem for phylogenetic comparative methods (and other things). *PeerJ* 12, pp. e16505.

Soininen, J., McDonald, R. and Hillebrand, H., 2007. The distance decay of similarity in ecological communities. *Ecography* 30(1), pp. 3–12.

Speckmann, B. and Verbeek, K., 2010. Necklace maps. *IEEE Transactions on Visualization and Computer Graphics* 16(6), pp. 881–889.

Tobler, W. R., 1970. A computer movie simulating urban growth in the detroit region. *Economic Geography* 46, pp. 234–240.

Yap, C. K., 1987. An $O(n \log n)$ algorithm for the Voronoi diagram of a set of simple curve segments. *Discrete & Computational Geometry* 2(4), pp. 365–393.