Article

From Trees to Barcodes and Back Again: Theoretical and Statistical Perspectives

Lida Kanari [1],*[ID]**, Adélie Garin** [2][ID] **and Kathryn Hess** [2]

[1] Blue Brain Project, École Polytechnique Fédérale de Lausanne (EPFL), Campus Biotech,
    1202 Geneva, Switzerland
[2] Laboratory for Topology and Neuroscience, Brain Mind Institute,
    École Polytechnique Fédérale de Lausanne (EPFL), 1015 Lausanne, Switzerland; adelie.garin@epfl.ch
    (A.G.); kathryn.hess@epfl.ch (K.H.)
*   Correspondence: lida.kanari@gmail.com

check for
updates

Abstract: Methods of topological data analysis have been successfully applied in a wide range of fields to provide useful summaries of the structure of complex data sets in terms of topological descriptors, such as persistence diagrams. While there are many powerful techniques for computing topological descriptors, the inverse problem, i.e., recovering the input data from topological descriptors, has proved to be challenging. In this article, we study in detail the Topological Morphology Descriptor (TMD), which assigns a persistence diagram to any tree embedded in Euclidean space, and a sort of stochastic inverse to the TMD, the Topological Neuron Synthesis (TNS) algorithm, gaining both theoretical and computational insights into the relation between the two. We propose a new approach to classify barcodes using symmetric groups, which provides a concrete language to formulate our results. We investigate to what extent the TNS recovers a geometric tree from its TMD and describe the effect of different types of noise on the process of tree generation from persistence diagrams. We prove moreover that the TNS algorithm is stable with respect to specific types of noise.

Keywords: tree; topological data analysis; persistence barcode; symmetric group; inverse methods

1. Introduction

Although geometric approaches to analyzing data have been extensively used for many years, the first topological methods for data analysis were developed only recently, e.g., [1–6]. Topological Data Analysis (TDA) is a fairly new field at the intersection of data science and algebraic topology, the aim of which is to provide robust mathematical, statistical, and algorithmic methods to infer and analyze the topological and geometric structures underlying complex data. These data are often represented as point clouds in Euclidean or metric spaces, though TDA methods have also been generalized to geometric objects and graphs. TDA has proved its utility in a wide range of applications in biology [7–10], material science [11], and climate science [12], among other fields. Although it is still rapidly evolving, TDA now provides a set of powerful and efficient tools that can be used in combination with or as complements to other data science tools.

One of the most promising applications of TDA is to the study of the brain, where it has served to analyze neuronal morphologies [13], brain networks [13–15], and brain functionality [16]. Motivated by the desire to objectively classify neuronal morphologies, in a previous publication (Kanari and Hess in [13]), we designed a topological signature for trees, the Topological Morphology Descriptor (TMD), that assigns a *barcode* (i.e., a multi-set of open intervals—called *bars*—in the real line) to any *geometric tree* (i.e., any finite binary tree embedded in $\mathbb{R}^3$). We showed that the TMD algorithm effectively determines the reliability of the clustering of random and neuronal trees. Moreover, using the TMD

algorithm, we performed an objective, stable classification of pyramidal cells in the rat neocortex [17], based only on the shape of their dendrites.

A frequent topic of discussion in the context of TDA is how to define an inverse to the process of associating a particular topological descriptor to a dataset, i.e., how to design a practical algorithm to recover the input data from a topological descriptor, such as a barcode. Oudot and Solomon [18] and Curry et al. [19] have proposed partial solutions to this problem. The main obstacle that renders this endeavor particularly challenging has proven to be the computational complexity of the space of inputs considered. To avoid this obstacle, it is reasonable to constrain the input space and search only for an inverse transformation that is relevant in a specific context, for instance to look for solutions only in the space of embedded graphs, as in [20].

In the context of geometric trees, we have designed an algorithm to reverse-engineer the TMD [21], in order to digitally generate artificial neurons, to compensate for the dearth of available biological reconstructions. This algorithm, called Topological Neuron Synthesis (TNS), stochastically generates a geometric tree from a barcode, in a biologically grounded manner. As shown in [21], the synthesized neurons are statistically indistinguishable from the corresponding reconstructed neurons in terms of both their morphological characteristics and the networks they form.

In this article, we further study the properties of this generative algorithm, from mathematical and statistical perspectives. We perform a theoretical and computational analysis of the TMD and TNS algorithms and their mathematical properties, in which symmetric groups play a key role. In particular, we investigate in detail the extent to which the TNS provides an inverse to the TMD.

First, we carefully define our objects of study—geometric trees, barcodes, and persistence diagrams—and then recall the TMD and TNS algorithms. We also introduce two distinct classifications of geometric trees: into combinatorial types and into TMD-types. The symmetric groups play an important role in our classification of trees into TMD-types. These complementary descriptions provide us with a language in which to formulate our results on the relationship between the TMD and the TNS.

In the next section, we introduce tools to describe the set of geometric trees that realize a specific barcode, i.e., whose TMD is equal to that barcode. In particular, we establish an explicit formula for the cardinality of this set, which we use to describe how the cardinality changes when a new bar is added to a barcode or two bars of a barcode permuted. Cayley graphs of symmetric groups provide a useful visualization of these effects.

We then study the composite of the TNS and TMD algorithms from a theoretical perspective, to quantify the extent to which the TNS acts as an inverse to the TMD. For a given barcode B, we show that, for a reasonable choice of parameter in the TNS, the probability that the bottleneck distance between the barcodes B and TMD $\circ$ TNS(B) is greater than ε decreases with ε, thus establishing a form of stability for the TNS. We prove, moreover, that the probability that two bars of a barcode B will be permuted by applying TMD $\circ$ TNS decreases exponentially with the distance between the terminations of the two bars, which is another form of stability. Together, these stability results imply that the TNS is an excellent approximation to a (right) inverse to the TMD.

In the final section, we present computational results that illustrate the complex relationship between a barcode and its possible tree-realizations. In particular, we study the distinguishing characteristics of "biological" geometric trees, i.e., those that arise from digital reconstructions of neurons, as opposed to arbitrary geometric trees. We also show that both the combinatorial type and the TMD-type of a geometric tree can change significantly when applying the composite TNS $\circ$ TMD, from which it follows that the TNS is not a left inverse to the TMD (Figure 1).
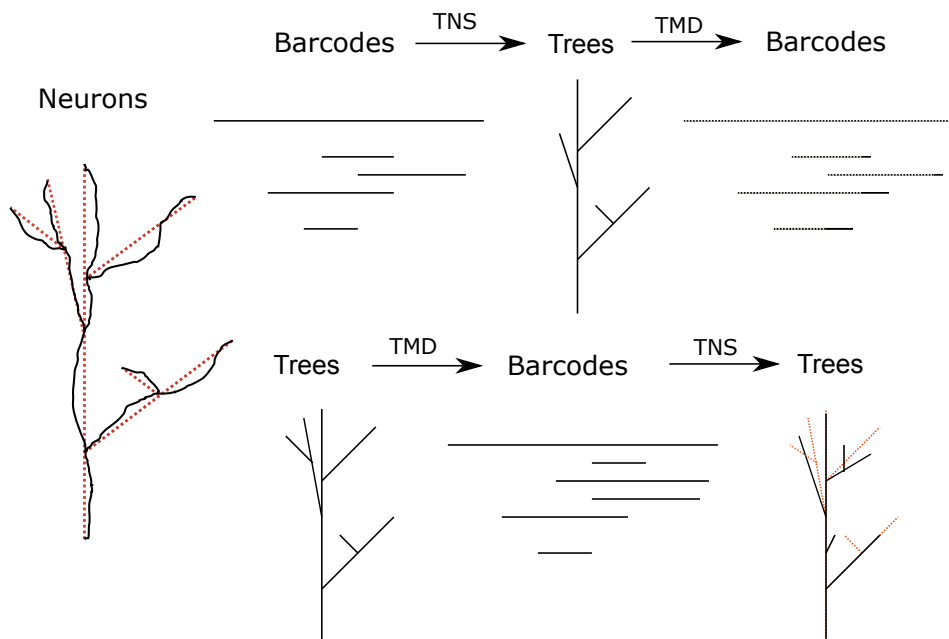
Figure 1. The two composites of TMD and TNS. (Left) An illustration of how a neuron (black) is modeled as a tree (dashed red lines). We describe this process and how to extract a barcode from a tree in Section 2.3. (Top) The composite TMD ∘ TNS applied to a barcode *B*. The new barcode $B' = \text{TMD} \circ \text{TNS}(B)$ is indicated in dashed lines on top of the barcode *B* on the right. We show in Section 4 that the barcodes *B* and *B'* will almost certainly be very similar and quantify this similarity. (Bottom) The composite TNS ∘ TMD applied to a tree *T*. The tree *T* that we start with is indicated in dashed red lines under the new tree $T' = \text{TNS} \circ \text{TMD}(T)$. The trees *T* and *T'* can be quite different combinatorially, as seen on the right.

2. Mathematical Background

Precisely what a mathematician means by the terms "tree" and "barcode" can vary depending on context. First, we specify what these terms mean in this article. We then recall biologically motivated algorithms for generating barcodes from trees [13] and trees from barcodes [21], the relation between which will be made clear in the following sections.

2.1. Trees

A *finite rooted tree T* is an acyclic, finite, directed graph such that each vertex is of degree at most 3, with a distinguished vertex *r* of degree 1, called the *root*. A vertex *v* of *T* is a *parent* of a vertex *w* if there is a directed edge from *w* to *v*; the vertex *w* is then a *child* of *v*. Each vertex of *T* has a single parent, except for the root *r*, which has no parent, and at most two children. The non-root vertices of degree 1 are called the *leaves* of *T*, and the vertices of degree 3 the *branch points* of *T*. A finite tree *T* is fully specified by its set of vertices, equipped with the partial order "is a parent of".

Our main objects of study in this article are *geometric trees*, i.e., embeddings of finite rooted trees in $\mathbb{R}^3$, which are often used to model neurons. We assume, moreover, that, if a vertex *v* is the parent of a vertex *w*, then the distance from the root to *v* is less than that from the root to *w* (Much of the framework that we develop in this article could be extended to geometric trees equipped with a different distance function that does not necessarily satisfy this condition, at the price of a more involved combinatorial representation. Since our geometric trees of interest—digitally reconstructed neurons equipped with the path-distance from the root—do satisfy our extra assumption, we prefer to leave this extension to the interested reader.). Let $\mathcal{T}$ denote the set of geometric trees. We say that two geometric trees *T* and *T'* are *combinatorially equivalent*, denoted $T \underset{\text{comb}}{\sim} T'$, if they are embeddings of the same finite rooted tree. In other words, the combinatorial type of a geometric tree is independent of its embedding in $\mathbb{R}^3$.

2.2. Barcodes

A persistence *barcode* is a finite multi-set $B = \{(b_i, d_i)\}_{i=0,\dots,n}$ of open intervals, called *bars*, in the real line, $\mathbb{R}$. We call b_i the *birth time* and d_i the *death time* of the ith interval. For barcodes of geometric trees, generated with the TMD algorithm (cf. Section 2.3), the birth time b_i is the distance from the first bifurcation of branch i to the root, while the death time d_i is the distance from the branch termination to the root. Let $\mathcal{B}$ denote the set of all barcodes.

A persistence barcode can equivalently be represented as a multi-set of points in $\mathbb{R}^2$, called a *persistence diagram*, where a bar (b_i, d_i) corresponds to a point in $\mathbb{R}^2$ with x-coordinate d_i and y-coordinate b_i (The inversion of the coordinates that we apply here is motivated by the TMD algorithm (cf. Section 2.3), which decomposes a geometric tree into bars, starting at the leaves and progressing down to the root.). If B is a barcode, we let $\text{PD}(B)$ denote the associated persistence diagram. Note that, under this correspondence, the points of $\text{PD}(B)$ lie below the diagonal, since b_i is less than d_i for every i.

We say that a barcode $B = \{(b_i, d_i)\}_{i=0,\dots,n}$ is *strict* if the first bar (b_0, d_0) properly contains all the others, i.e., $b_0 < b_i$ and $d_i < d_0$ for all i, and no bars are born or die at the same time, i.e., $b_i \neq b_j$ and $d_i \neq d_j$ for all $i \neq j$. The birth times of a strict barcode admit a total ordering. Without loss of generality, we assume that the bars are ordered by birth value that is $b_0 < b_1 < \cdots < b_n$. Let $\mathcal{B}^{\text{st}}$ denote the set of strict barcodes, and let $\mathcal{B}_n^{\text{st}}$ denote the subset of those strict barcodes with $n + 1$ bars.

We say that two strict barcodes $B = \{(b_i, d_i)\}_{i=0,\dots,n}$ and $B' = \{(b_i', d_i')\}_{i=0,\dots,n}$ with the same number of bars *are equivalent*, denoted $B \underset{\text{bar}}{\sim} B'$, if their deaths occur in the same order ($d_i > d_j \Leftrightarrow d_i' > d_j'$), which clearly defines an equivalence relation on $\mathcal{B}_n^{\text{st}}$. There is a bijection from the set of equivalence classes of strict barcodes with $n + 1$ bars to the symmetric group $\mathfrak{S}_n$ on n letters, taking the equivalence class of a barcode $B = \{(b_i, d_i)\}_{i=0,\dots,n}$ to the permutation $\sigma_B : \{1, \dots, n\} \to \{1, \dots, n\}$, where $\sigma_B(i) = \#\{j \mid d_j \leq d_i\}$.

We denote the equivalence class containing a strict barcode $B = \{(b_i, d_i)\}_{i=0,\dots,n}$ by $(i_1 \dots i_n)$, where $d_{i_k} > d_{i_{k+1}}$ for all $1 \leq k < n$. For example, (2134) corresponds to the barcode with five bars shown in Figure 2.
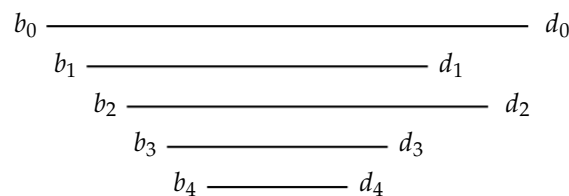
$$
\begin{array}{ll}
b_0 \, \rule{5cm}{0.4pt} \, d_0 \\
\quad b_1 \, \rule{4cm}{0.4pt} \, d_1 \\
\quad\quad b_2 \, \rule{4cm}{0.4pt} \, d_2 \\
\quad\quad\quad b_3 \, \rule{2.5cm}{0.4pt} \, d_3 \\
\quad\quad\quad\quad b_4 \, \rule{1.5cm}{0.4pt} \, d_4
\end{array}
$$

Figure 2. A strict barcode belonging to the equivalence class (2134). One bar (b_0, d_0) contains all the others. The remaining bars are ordered by their birth times ($b_1 < b_2 < b_3 < b_4$). Similarly, the deaths are ordered $d_2 > d_1 > d_3 > d_4$, leading to the notation (2134).

2.3. The TMD: From Trees to Barcodes

The TMD (Topological Morphology Descriptor) is a many-to-one function from the set of geometric trees to the set of barcodes,

$$\text{TMD} : \mathcal{T} \to \mathcal{B},$$

that encodes the overall shape of the tree, both the topology of the branching structure of a tree and its embedding in $\mathbb{R}^3$ [13]. It is defined recursively as follows.

Let T be a rooted tree with root r and set N of vertices, with subset L of leaves. Let $\delta : N \to \mathbb{R}_{\geq 0}$ be the function that assigns to each vertex its Euclidean distance to the root r.

Intuitively, the output of the TMD algorithm is a barcode (For those used to thinking in terms of persistent homology, the TMD computes the 0-dimensional barcode, or persistence diagram, of the distance function δ. Each bar (b, d) corresponds to a connected component in the sublevel sets $\delta^{-1}([0, t))$, that is, a branch of the tree. Note that the birth and death roles are reversed in the TMD algorithm compared to "usual" terminology in persistent homology: the birth corresponds to the bifurcation and the death to the termination of a branch.), where each bar represents a branch of the

tree. The endpoints of a bar correspond to the distances to the root from the tip of the branch and from the point where the branch bifurcates from another, longer branch, see Figure 3. Table 1 summarizes the terminology used for the TMD algorithm.
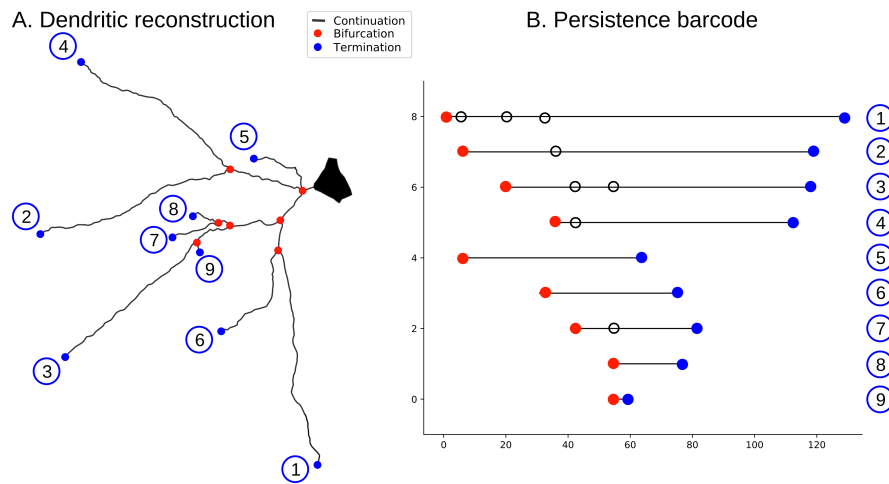

Figure 3. The algorithm to encode a tree structure as a persistence barcode. (**A**) neuronal tree; (**B**) persistence barcode generated with TMD. Each branch in the tree (**A**) corresponds to a bar in the barcode (**B**); the circled numbers encode the correspondence between branches and bars. Terminations are shown in blue, bifurcations in red, and branches in between in black.

Table 1. Summary and terminology of the TMD and TNS algorithms. The TMD computes the barcode of a tree from the tips of branches towards the root, whereas the TNS grows the tree in the opposite direction, from the root to the leaves.

	TMD	TNS
Goal	Compute the barcode of a tree based on a distance function	Grow a new tree from a barcode
Directionality	From leaves to root	From root to leaves
Domains	{geometric trees} $\longrightarrow$ {barcodes}	{barcodes} $\longrightarrow$ {geometric trees}

For each $v \in N \smallsetminus L$, let L_v denote the set of leaves of the subtree of T with root at the branch point v. Let $\mu : N \to \mathbb{R}$ be the function defined by

$$\mu(v) = \begin{cases} \max\{\delta(l) \,|\, l \in L_v\} & : v \in N \smallsetminus L, \\ \delta(v) & : v \in L. \end{cases}$$

We order the children of any vertex of T by their μ-value: if $v_1, v_2 \in N$ are siblings, then v_1 is younger than v_2 if $\mu(v_1) < \mu(v_2)$.

The algorithm that extracts the TMD of a geometric tree T proceeds as follows (Figure 3). Start by creating a set A of *active vertices*, originally set equal to L, and an empty barcode. For each leaf l, the algorithm proceeds recursively along its unique path to the root r. At each branch point b, one applies the standard *Elder Rule* from topological data analysis [22], removing from A all of the children of b, and adding b to A. One bar is added to the barcode for each child of b except (any one of) the longest. Each child removed from A corresponds to a path from some leaf l to b, which is recorded in the barcode as a bar $(\delta(b), \delta(l))$. These operations are applied iteratively to all the vertices until the root r is reached, at which point A contains only r and a leaf l for which μ is maximal among all leaves, which is recorded in the barcode as a bar $(0, \delta(l))$.

If T is a digital reconstruction of a neuron, and the function δ is the path distance from the soma, then TMD(T) is actually a strict barcode. Indeed, the probability for two branch points or leaves to be

exactly the same distance from the soma is almost zero, and TMD(T) always has a longest bar that contains all the others. This observation justifies our interest in the subset of strict barcodes.

The TMD gives rise to an equivalence relation on $\mathcal{T}$: two geometric trees T and T' are *TMD-equivalent*, denoted $T \underset{\text{tmd}}{\sim} T'$, if TMD($T$) $\underset{\text{bar}}{\sim}$ TMD(T'). We provide below an in-depth analysis of the TMD-equivalence classes of geometric trees. Geometric trees can be combinatorially equivalent without being TMD-equivalent and vice versa, cf. Figure 5.

2.4. The TNS: From Barcodes to Trees

The topological neuron synthesis (TNS) algorithm [21] stochastically generates synthetic neurons, in particular for use in digital reconstructions of brain circuitry [23]. In this paper, we focus on the sub-process of the TNS that stochastically generates a geometric tree from a strict barcode, in such a way that, if a tree T is generated from a barcode B, then TMD(T) is "close to" to B, with respect to an appropriate metric on the set of barcodes, up to some stochastic noise, cf. Section 4. Henceforth, when we refer to the TNS, we mean this sub-process.

To grow geometric trees, the TNS algorithm first initiates growth, then loops through steps of *elongation* and *branching/termination*. Each branch of the tree is elongated as a directed random walk [24] with memory. At each step, a growing tip is assigned probabilities to bifurcate, to terminate, or to continue that depend on the path distance from the root and on a chosen bar of the selected barcode. Once a bar has been used, it is removed from the barcode. The growth of a tree terminates when no bars remain to be used. We now provide further details of the two steps in this process.

2.4.1. Bifurcation/Termination

The branching process in the TNS algorithm is based on the concept of a *Galton–Watson tree* [25], which is a finite rooted tree recursively generated as follows. At each step, a number of offspring is independently sampled from a distribution. Since a geometric tree consists only of bifurcations, terminations, and continuations, the accepted values for the number of offspring are: zero (termination), one (continuation), and two (bifurcation). The Galton–Watson algorithm generates only a combinatorial tree, with no embedding in space, so we modify the traditional process to introduce a dependency of the tree growth on the embedding, so that the bifurcation/termination probabilities depend on the path distance of the growing tip from the root.

The bifurcation/termination step of the growth process of a geometric tree with associated barcode B proceeds as follows. Each growing tip of the tree is assigned a bar (b_i, d_i) sampled from the barcode B and a bifurcation angle a_i. The growing tip first checks the probability to bifurcate, then the probability to terminate. If the growing tip does not bifurcate or terminate, then the branch continues to elongate. The probability to bifurcate depends on b_i: as the distance from the root to the growing tip approaches b_i, the probability to bifurcate increases exponentially until it attains a maximum of 1 at b_i. Similarly, the probability to terminate depends exponentially on d_i.

The probabilities to bifurcate and terminate are sampled from an exponential distribution $e^{-\lambda x}$, whose free parameter λ should be wisely chosen. A very steep exponential distribution (high value of λ) reduces the variance of the population of geometric trees synthesized based on the same barcode. On the other hand, a very low value of λ results in trees that are almost random, since the dependence on the input persistence barcode is decreased significantly. If we assume that growth takes place in discrete steps of size L, the value of the parameter λ should be of the order of the step size L, to ensure biologically appropriate variance [21]. Assuming $L = 1$ in some appropriate units, we usually select $\lambda \approx 1$, so that the bifurcation and termination points are stochastically chosen but still strongly correlated with the input persistence barcodes.

Contrary to other neuron synthesis algorithms [26] that sample the branching and termination probabilities from independent distributions, in the TNS, the correlation of these probabilities is captured in the structure of the barcode. When the growing tip bifurcates, the corresponding bar is removed from the input barcode to exclude re-sampling of the same conditional probability,

thus recording the tree's growth history, which is essential for reproducing the branching structure. In the event of a termination, the growing tip is deactivated, and the bar that corresponds to this termination point is removed from the reference barcode.

At a bifurcation, the directions of the two daughter branches created depend on the bifurcation angle a_i. In this study, we focus primarily on the combinatorial type and the TMD type of the generated geometric tree, so we do not investigate the effect of bifurcation angles on the growth.

2.4.2. Elongation

We now describe how the synthesized trees are embedded in $\mathbb{R}^3$. A *segment* of a growing tree is the portion of the tree between a pair of consecutive vertices (parent and child). Each synthesized tree is grown segment by segment. The *direction* of a segment, i.e., the vector $\vec{d}$ from its starting point to its end point, is a weighted sum of three unit vectors: the cumulative *memory* $\vec{m}$ of the directions of previous segments within a branch, a *target vector* $\vec{t}$, and a random vector $\vec{r}$ [26]. The memory term is a weighted sum of the previous directions of the branch, with the weights decreasing with distance from the tip. As long as the memory function decreases faster than linearly with the distance from the growing tip, the exact choice of function is not important [21]. The target vector is chosen at the beginning of each branch and depends on the bifurcation angles. The random component is a unit vector sampled uniformly from $\mathbb{R}^3$ at each step. The direction of the segment

$$\vec{d} = \rho\vec{r} + \tau\vec{t} + \mu\vec{m},$$

then depends on three weight parameters ρ, τ, and μ, where $\rho + \tau + \mu = 1$.

An increase of the randomness weight ρ results in a highly tortuous branch, approaching the limit of a simple random walk when $\rho = 1$. If the targeting weight $\tau = 1$, the branch will be a straight line in the target direction. Different combinations of the three parameters (τ, ρ, μ) generate more or less meandering branches and thus reproduce a large diversity of geometric trees.

2.4.3. The Elder Rule and TNS

The TNS provides a sort of right inverse to the TMD. To recreate a tree that is close to TMD-equivalent to the original, the branch corresponding to a particular bar (b_i, d_i) in the barcode can be attached only to branches corresponding to bars (b_j, d_j) such that $d_i < d_j$ and $b_i > b_j$. This rule ensures that the Elder rule (at a bifurcation, the longer component survives) holds in the TMD transformation. As a result, only a subset of trees with n branches can be generated by the TNS from a given strict barcode with n bars.

3. Tree-Realizations of Barcodes

In this section, we provide an in-depth analysis of the set of geometric trees that realize a specific strict barcode B, i.e., each of which has TMD equal to B.

3.1. Realizing Barcodes as Trees

A geometric tree T is a *tree-realization* of a barcode B if $\text{TMD}(T) = B$, i.e., $T \in \text{TMD}^{-1}(B)$. Examples of tree-realizations are provided in Figure 4B, while Figure 5 shows all the possible combinatorial types of tree-realizations of a strict barcode with $n = 4$.

In Figures 4 and 5, we encode the combinatorial structure of the tree, i.e., how the branches may be attached to each other, in an adjacency matrix in which the (i, j) coefficient is non-zero if the Elder Rule allows bar i to be connected to bar j. For example, in Figure 4A, bars 1–3 may all be connected to the black bar 0, thus the coefficients $(0, 1), (0, 2), (0, 3)$ are all non-zero in the corresponding adjacency matrix. Note that in each realization only a subset of these possible attachments is actually made (Figure 4B), since each branch can be attached to only one other branch.

The connectivity diagram (bottom of Figure 4A) provides another representation of the pairs of branches that may be connected, in agreement with the Elder Rule. The arrow on an edge in the diagram indicates the direction of the connection. In this example, there are arrows from 0 towards 1, 2, and 3, from 1 to 2 and 3, and from 2 to 3.
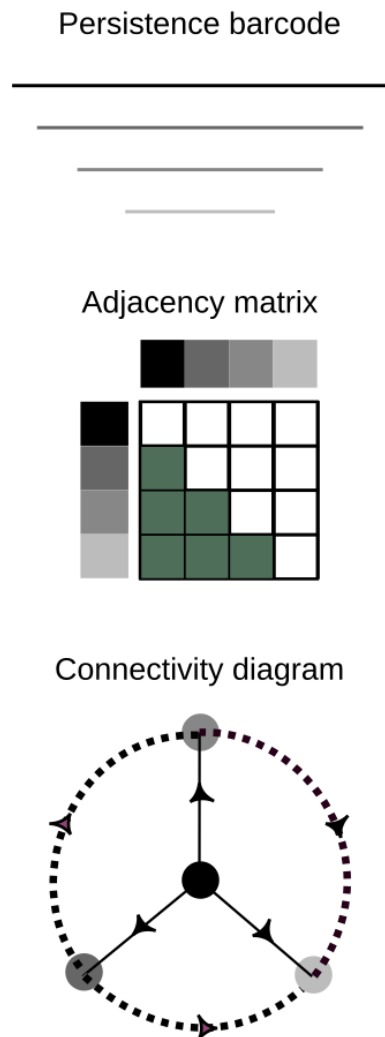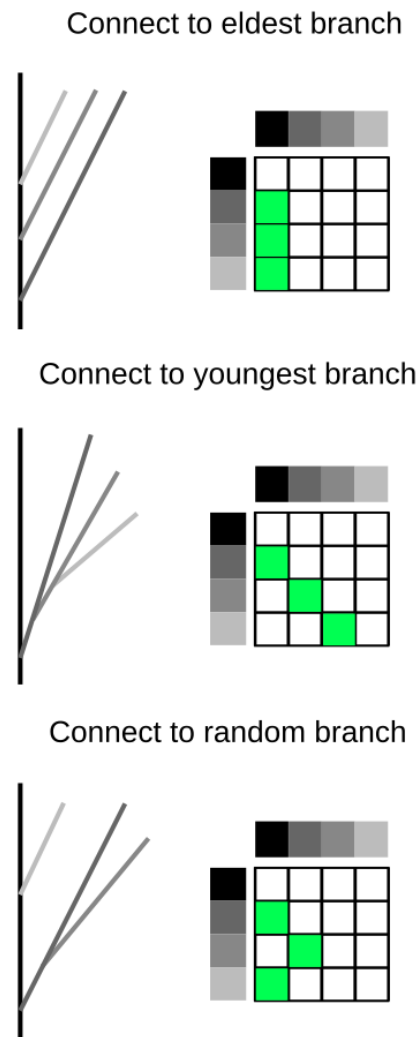

Figure 4. A strict barcode, whose bars are ordered according to birth times (greyscale), defines a unique ordering of death times. This ordering and the Elder Rule constrain the possible combinatorial types of trees that can be realized from this barcode. (**A**) the notation that will be used in this paper from a barcode that corresponds to an adjacency matrix of possible connectivities. Equivalently, the possible connectivities are presented in the connectivity diagram; (**B**) examples of possible tree realizations from branches that connect to the longest one (top) to random (bottom).
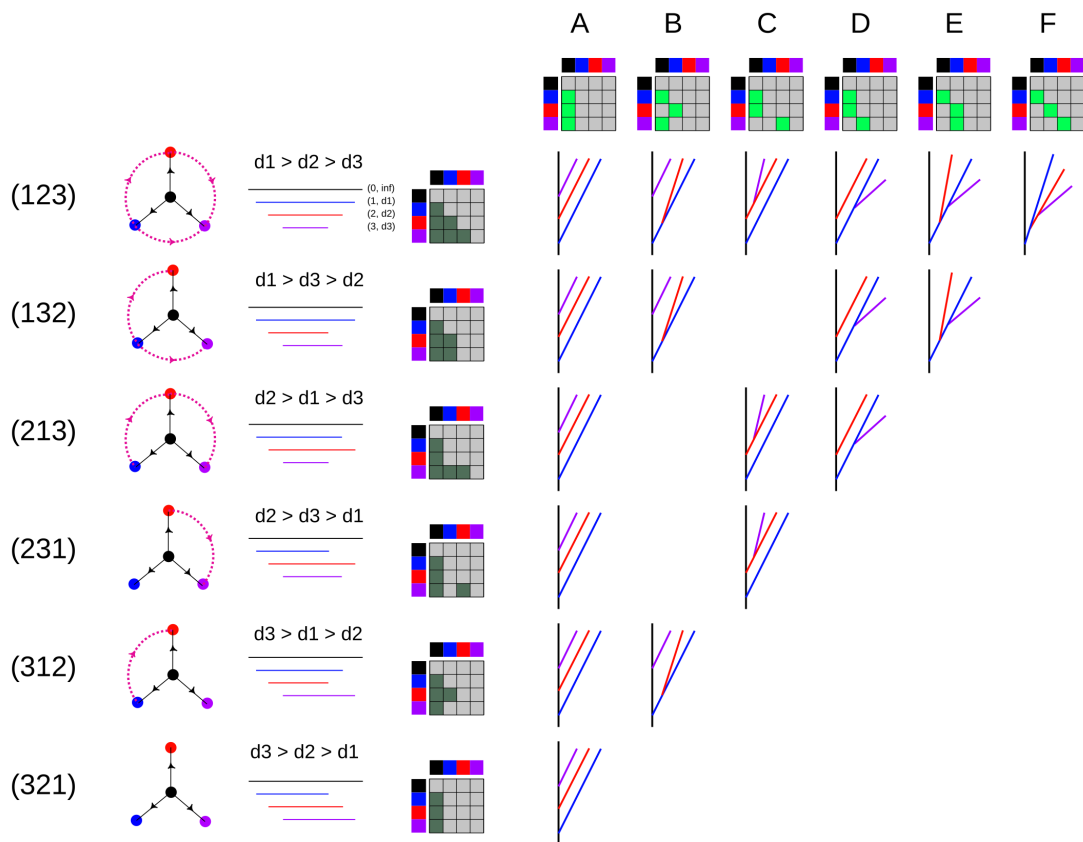
Figure 5. Tree-realizations of all possible strict barcodes B with four bars. Each row (left) represents a possible permutation of death times for the corresponding order of bars in B, i.e., a possible TMD-type. Each barcode can be realized by a subset of all the combinatorial tree types, each represented by a column, with a corresponding adjacency matrix.

For any strict barcode B, let $\mathcal{T}(B)$ denote the set of combinatorial equivalence classes of tree-realizations of B, i.e.,

$$\mathcal{T}(B) = \mathrm{TMD}^{-1}(B)/\underset{\mathrm{comb}}{\sim}.$$

We can characterize the equivalence relation on strict barcodes in terms of $\mathcal{T}(B)$.

Lemma 1. *If B and B' are two strict barcodes with the same number of bars, then*

$$B \underset{\mathrm{bar}}{\sim} B' \iff \mathcal{T}(B) = \mathcal{T}(B').$$

Proof. The order of the deaths in a strict barcode B completely determines the set of combinatorial equivalence classes of its possible tree realizations.

Indeed, the two pairs of bars in Figure 6B lead to the same adjacency possibilities for their respective branches. Only move (A) in Figure 6, corresponding to switching the order of the deaths of the two bars, modifies the permutation equivalence class of the barcode, hence also the set of trees that return the given barcode.
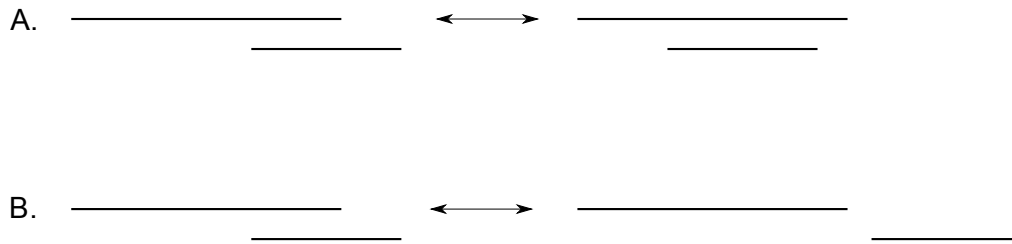
A.

B.

Figure 6. The two possible moves that respect the condition of a realizable barcode. Move (**A**) modifies the barcode's ordering, whereas move (**B**) does not change the order of the deaths.

□

3.2. The Combinatorics of Tree-Realization

Let $B = \{(b_i, d_i)\}_{i=0,\ldots,n}$ be a strict barcode. In this section, we analyze the *tree-realization number* of B,

$$\text{TRN}(B) = \#\mathcal{T}(B),$$

i.e., the number of combinatorial equivalence classes of tree-realizations of B. We provide a formula for $\text{TRN}(B)$ in terms of the *index* of each bar (b_i, d_i), i.e., the number of bars that include (b_i, d_i) strictly:

$$\text{index}_i(B) = \#\{j \mid b_j < b_i < d_i < d_j\} = \#\{j < i \mid d_i < d_j\}.$$

A version of this formula was established by Curry in [22].

Lemma 2. *The tree-realization number of a strict barcode $B = \{(b_i, d_i)\}_{i=0,\ldots,n}$ is equal to the product of the indices of its bars, i.e.,*

$$\text{TRN}(B) = \prod_{1 \leq i \leq n} \text{index}_i(B).$$

Proof. Because of the Elder Rule applied in the TMD, one branch can be attached to another only if its corresponding bar is included in the other bar. This simple observation enables us to prove the lemma by a straightforward recursion on the number of bars.   □

In particular, the maximum tree-realization number for a strict barcode with $n + 1$ bars is $n!$, in the specific case where $d_n < \ldots < d_1 < d_0$. We call this case a strictly ordered barcode.

Note that the tree-realization number does **not** satisfy

$$\text{TRN}(B) = \text{TRN}(B') \implies B \sim B'$$

in general, i.e., the tree-realization number is not a complete invariant of the barcode equivalence relation. For instance, barcodes (231) and (312) in Figure 7 both have $\text{TRN}(B) = 2$ but have different permutation types. The inverse clearly does hold, however:

$$\text{TRN}(B) \neq \text{TRN}(B') \implies B \nsim B',$$

enabling us to detect non-equivalence of barcodes. For the pairs of barcodes studied in Sections 4 and 5 of this paper, though, it is usually true that if they have same TRN, then they are equivalent, so that we can use the TRN to detect equivalence of barcodes in these special cases. In particular, if a new bar is added to a barcode or two deaths are transposed and no other changes take place, then the tree-realization number does change.
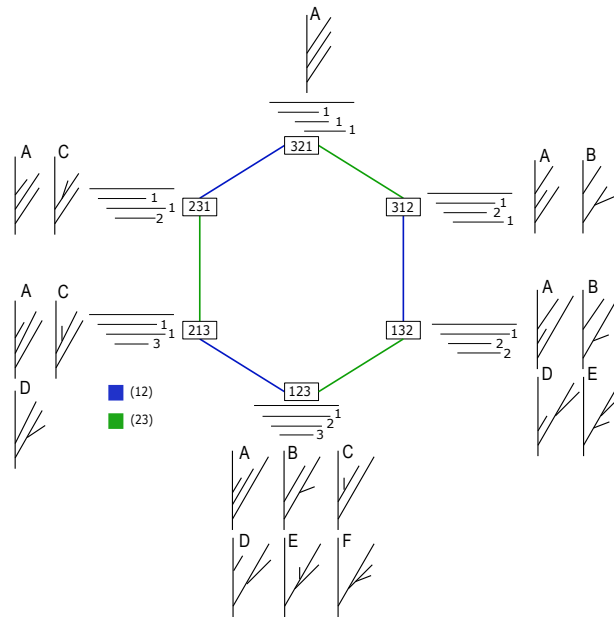
Figure 7. A representation of the space of barcodes with four bars up to permutation equivalence class as the Cayley graph of $\mathfrak{S}_3$ generated by the transpositions (12) and (23), respectively. Each vertex is an element of the group. The edges represent the transposition to convert one end point into the other, colored by generator. The number to the right of each bar is its index. All trees T such that $\text{TMD}(T) = B$ are indicated next to a barcode B with the corresponding tree type of Figure 5. The number of such trees can be computed using Lemma 2.

Lemma 2 enables us to quantify how adding a new bar changes tree-realization number.

Lemma 3. *Let* $B = \{(b_i, d_i)\}_{i=0,\ldots n}$ *and* $B' = B \cup \{(b_{n+1}, d_{n+1})\}$, *where* $b_{n+1} > b_i$ *for all* $0 \le i \le n$, *be strict barcodes. If* $d_{i_1} > \ldots d_{i_{k-1}} > d_{n+1} > d_{i_k} > \ldots d_{i_n}$, *then*

$$\text{TRN}(B') = \text{TRN}(B) \cdot k.$$

Proof. The condition on d_{n+1} implies that the new bar (b_{n+1}, d_{n+1}) is included in exactly k other bars, so its index is k. □

Example 1. *Let* B *be a barcode with four bars such that* $b_0 < b_1 < b_2 < b_3$ *and* $d_0 > d_2 > d_1 > d_3$, *i.e., its equivalence class is* (213). *It is easy to see that* $\text{TRN}(B) = 3$ *(see Figure 8). If we add a new bar* (b_4, d_4) *such that* $d_1 > d_4 > d_3$, *the equivalence class of the new barcode* B' *is* (2143), *and bar* (b_4, d_4) *is included in* (b_0, d_0), (b_2, d_2), *and* (b_1, d_1), *but not in* (b_3, d_3) *because* $d_4 > d_3$. *Therefore, its index is 3, whence* $\text{TRN}(B') = 3 \times 3 = 9$.
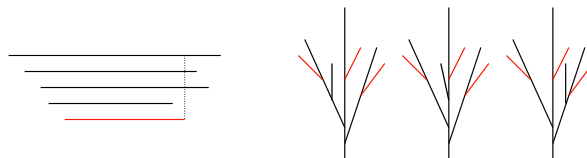

Figure 8. A barcode B in equivalence class (213) is shown in black. There are three possible combinatorial equivalence classes of trees whose TMD barcode is B, also represented in black. After adding the extra bar in red, we obtain a new barcode B', in the equivalence class (2143). In a tree-realization of B', the branch corresponding to the red bar can be attached to any of the branches corresponding to the 0th, 1st, and 2nd bars, represented on the trees by the red branches. This leads to nine possible combinatorial equivalence classes of trees for the barcode (2134).

We can also apply Lemma 2 to determining how switching the order of two consecutive deaths in the barcodes affects the tree realization number.

Proposition 1. *Let* $B = \{(b_i, d_i)\}_{i=0,...n}$ *be a strict barcode in the equivalence class* $(i_1...i_n)$. *Let* $B' = \{(b'_i, d'_i)\}_{i=0,...n}$ *be a new barcode obtained by permuting the deaths* d_{i_k} *and* $d_{i_{k+1}}$, *i.e.,* $b_i = b'_i$ *for all* i *and* $d_i = d'_i$ *for all* $i \neq i_k, i_{k+1}$, *while* $d_{i_k} = d'_{i_{k+1}}$ *and* $d_{i_{k+1}} = d'_{i_k}$:

1. *If* $i_k < i_{k+1}$, *then* $\text{index}_{i_{k+1}}(B') = \text{index}_{i_{k+1}}(B) - 1$, *and*

$$\text{TRN}(B') = \frac{\text{TRN}(B)(\text{index}_{i_{k+1}}(B) - 1)}{\text{index}_{i_{k+1}}(B)}.$$

2. *If* $i_k > i_{k+1}$, *then* $\text{index}_{i_{k+1}}(B') = \text{index}_{i_{k+1}}(B) + 1$, *and*

$$\text{TRN}(B') = \frac{\text{TRN}(B)(\text{index}_{i_{k+1}}(B) + 1)}{\text{index}_{i_{k+1}}(B)}.$$

Proof. It is enough to prove (1), since (2) then follows by switching the roles of B and B'.

If $i_k < i_{k+1}$, then $b_{i_k} < b_{i_{k+1}}$. Since B is in the equivalence class $(i_1...i_n)$, $d_{i_{k+1}} < d_{i_k}$ as well, whence $(b_{i_{k+1}}, d_{i_{k+1}}) \subset (b_{i_k}, d_{i_k})$. On the other hand, $(b'_{i_{k+1}}, d'_{i_{k+1}}) \not\subset (b'_{i_k}, d'_{i_k})$, but otherwise respects all of the same inclusion relations as $(b_{i_{k+1}}, d_{i_{k+1}})$, so that

$$\text{index}_{i_{k+1}}(B') = \text{index}_{i_{k+1}}(B) - 1,$$

as desired. Moreover, $(b'_{i_k}, d'_{i_k}) \not\subset (b'_{i_{k+1}}, d'_{i_{k+1}})$, so (b'_{i_k}, d'_{i_k}) respects exactly the same inclusion relations as (b_{i_k}, d_{i_k}), i.e.,

$$\text{index}_{i_k}(B') = \text{index}_{i_k}(B).$$

Because no other bars are affected when passing from B to B', we can conclude. □

Example 2. *In Figure 9, we show all the possible death-transpositions in a strict barcode with five bars. As an example, take B in the equivalence class* (2134), *so the barcode satisfies* $d_2 > d_1 > d_3 > d_4$. *The index of* (b_4, d_4) *is 4 because it is included in all the other bars. Permuting d_3 and d_4 leads to a barcode in the equivalence class* (2143). *The index of the last bar is now 3 because it is no longer included in the third bar.*
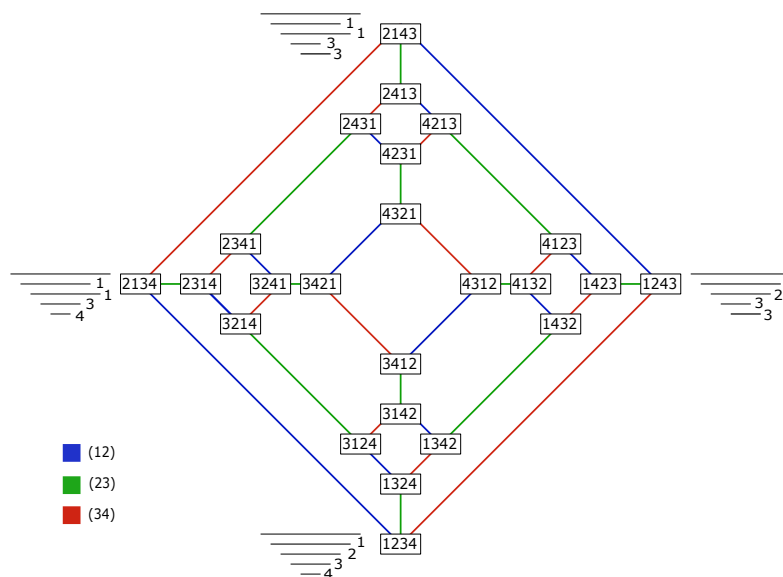

Figure 9. The Cayley graph representing $\mathfrak{S}_4$ generated by (12), (23), (34). For the four outside elements, we provide the barcode associated with the permutation. The number next to each bar is its index.

Recall the bijection $\sigma : \mathcal{B}_n^{\mathrm{st}} \to \mathfrak{S}_n$ from Section 2.2. Permuting the order of the deaths d_i and d_{i+1} corresponds to a transposition $(i, i+1)$ in $\mathfrak{S}_n$ (and to move (A) in Figure 6). Studying the allowed moves and their effects on the barcode is equivalent to studying the symmetric group seen as generated by transpositions of type $(i, i+1)$, enabling us to create the following revealing visualization of the effect of switching the order of deaths or of adding a new bar by using Cayley graphs of $\mathfrak{S}_n$.

We illustrate the results of Lemma 3 and Proposition 1 on the two Cayley graphs of $\mathfrak{S}_3$ and $\mathfrak{S}_4$ in Figures 7 and 9. Figure 7 shows the Cayley graph of $\mathfrak{S}_3$ generated by the permutations (12) and (23) and the corresponding equivalence classes of barcodes. The vertices of the graph correspond to the permutations in the symmetric group and their corresponding barcode types, and the edges between them to the transpositions transforming one permutation into another. The number next to each bar is its index. The trees that return a given barcode are sketched next to each vertex of the Cayley graph of $\mathfrak{S}_3$.

Figure 9 shows the Cayley graph of $\mathfrak{S}_4$ generated by the transpositions (12), (23), and (34), illustrating the effect on tree-realization number both of switching the order of two deaths and, in comparison with the previous figure, of adding an extra bar.

4. Stability of the TNS

In this section, we investigate the effect of the composition of the TNS and TMD algorithms from a theoretical perspective. Given a strict barcode $B = \{(b_i, d_i)\}_{i=0,\dots,n}$, we apply the TNS to B, for a fixed choice of the parameter λ (cf. Section 2.4), obtaining a tree T_B, and then compute the barcode of T_B, $\mathrm{TMD}(T_B)$, which we denote by $B' = \{(b_i', d_i')\}_{i=0,\dots,n}$. To quantify to what extent the TNS acts as an inverse to the TMD, we are interested in determining how similar B and B' are.

Expressing the similarity between B and B' in terms of the bottleneck distance enables us to establish one form of stability for the TNS in the first part of this section. We establish another type of stability for the TNS in the second part, when we show that the probability that the order of two specific bars will be altered upon applying $\mathrm{TMD} \circ \mathrm{TNS}$ decreases exponentially with the distance between the death times of the two bars. Together, these stability results imply that the TNS is an excellent approximation to a (right) inverse to the TMD.

4.1. Bottleneck Stability

Henceforth, we call the endpoints of the bars of the barcode B *targets*, as the TNS algorithm either creates a new branch or terminates a branch when the distance from the root approaches a birth or death point, respectively.

By definition of the TNS algorithm (cf. Section 2.4), when approaching a target, there is an exponential probability to bifurcate (create a new branch) or terminate, depending on λ. It follows that, for any bar (b_i, d_i) of a given barcode B, the distance between b_i and b_i' (the bifurcation distance and the target bifurcation distance of the ith branch) and the distance between d_i and d_i' (the termination distance and the target termination distance of the ith branch) should follow an exponential distribution of parameter λ,

$$|b_i - b_i'| \sim \mathrm{Exp}(\lambda) \text{ and } |d_i - d_i'| \sim \mathrm{Exp}(\lambda).$$

The notion of similarity between barcodes that we consider here is the standard *bottleneck distance*. Given two strict barcodes B and B' with $n+1$ bars, the bottleneck distance between them is

$$d(B, B') = \inf_{\gamma \in \mathfrak{S}_n} \sup_i |b_i - b_{\gamma(i)}'| + |d_i - d_{\gamma(i)}'|.$$

Lemma 4. *Let B be a strict barcode with n bars, and let $B' = \mathrm{TMD} \circ \mathrm{TNS}(B)$. If $B \sim B'$, then*

$$\mathbb{P}\big(d(B, B') > \varepsilon\big) \leq 1 - (1 - \exp(-\lambda\varepsilon)(\lambda\varepsilon + 1))^n. \tag{1}$$

Proof. Considering the case where γ is the identity, we see that

$$d(B, B') \leq \sup_i |b_i - b_i'| + |d_i - d_i'|.$$

If $B \sim B'$, the differences between the new and original values of the births and deaths all follow an exponential distribution,

$$|b_i - b_i'| \sim \text{Exp}(\lambda) \text{ and } |d_i - d_i'| \sim \text{Exp}(\lambda).$$

The cumulative probability distribution function of $|b_i - b_i'| + |d_i - d_i'|$ is thus given by an Erlang$(2, \lambda)$ distribution

$$\mathbb{P}\left(|b_i - b_i'| + |d_i - d_i'| \leq \varepsilon\right) = 1 - (1 + \lambda\varepsilon)\exp(-\lambda\varepsilon).$$

Because we consider the supremum over i of the sum $|b_i - b_i'| + |d_i - d_i'|$, and all of the $|b_i - b_i'| + |d_i - d_i'|$ are *i.i.d*, it follows from the theory of order statistics that

$$\mathbb{P}\left(d(B, B') \leq \varepsilon\right) \geq \mathbb{P}(\sup_i |b_i - b_i'| + |d_i - d_i'| \leq \varepsilon) = (1 - \exp(-\lambda\varepsilon)(\lambda\varepsilon + 1))^n. \tag{2}$$

Considering the probability of the complement leads to the result in Equation (1). □

Lemma 4 implies that the TNS is stable with respect to the bottleneck distance, in a manner dependent on the parameter λ. To illustrate this dependence, we plot the function of Equation (1) for different values of λ in Figure 10. The curve obtained for $\lambda = 1$ (blue in Figure 10) makes it clear that setting $\lambda = 1$ ensures that the TNS gives rise to a diverse family of new trees that are nonetheless topologically not significantly far from the original ones, which is the desired goal from a biological perspective. Making an appropriate choice of the parameter λ is thus essential.
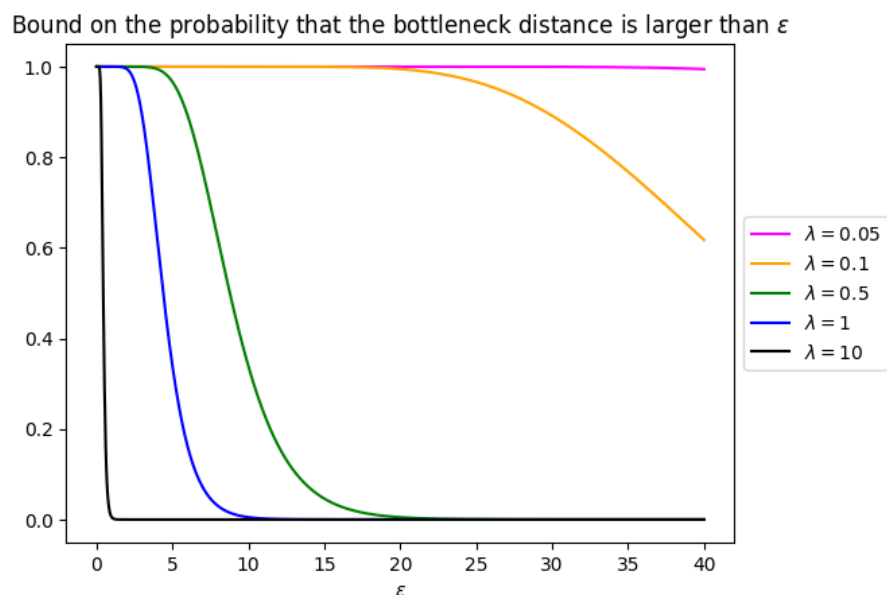

Figure 10. Upper bound on the probability that the bottleneck distance between B and $\text{TNS} \circ \text{TMD}(B)$ is larger than ε (Equation (1)) for various values of λ and for $n = 10$.

If $B \sim B'$, the bound by $\gamma = I$ in the formula for the bottleneck distance is computed between pairs of points that follow the same exponential law, as the order of bars is preserved. If $B \nsim B'$, for example when a switch of bars occurs, then we cannot assume that the distances between matched

pairs of points in the computed bottleneck distance follow the same law. Change of permutation type between B and B' is more frequent for small λ (Figure 14). Therefore, the previous lemma is usually not applicable for small values of λ, for which it is any case not particularly useful, as shown in Figure 10. In Figure 11, we summarize graphically the discussion above. The transposition of bars is studied in detail in the next section.
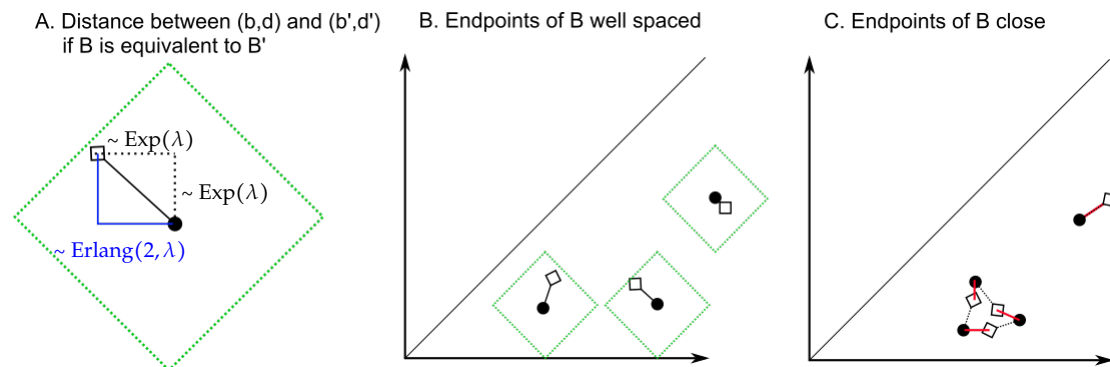

Figure 11. (**A**) The ℓ_1-distance between the black bullet and the diamond follows an Erlang$(2, \lambda)$ distribution. The interior of the green square defines a bound for the ℓ_1-distance from the black bullet that depends on the value of the parameter λ. (**B**) If the endpoints of the bars of B are sufficiently far away from each other and $B \sim \mathrm{TMD} \circ \mathrm{TNS}(B)$, then, with high probability, taking $\gamma = I$ will minimize the ℓ_1-distance between pairs of endpoints of bars. (**C**) If the endpoints of B are instead close to each other, then it is more likely that $B \nsim \mathrm{TMD} \circ \mathrm{TNS}(B)$, so that the optimal choice of γ (represented by red segments) is not the identity. The red distances do not necessarily follow exponential distributions, so the proof of Lemma 4 does not apply.

We perform two experiments to illustrate our theoretical results computationally. First, we compute the bottleneck distance between input barcodes B and output barcodes B', for increasing values of lambda λ from 0.01 to 2 (see Figure 12A1-3). The computational results (average bottleneck distances in red, Figure 12A2) fit the curve of the expected mean of the probability density function (The PDF can be deduced from Equation (2) in the proof of Lemma 4 by taking the derivative of $(1 - \exp(-\lambda\varepsilon)(\lambda\varepsilon + 1))^n$.) well (blue curve).

We also compute the cumulative density function for $0 \leq \epsilon \leq 200$ and $0 \leq \lambda \leq 2$, which we compare to the computational results (red points, Figure 12A3), showing that they match the theoretical prediction (blue colormap) very closely for a wide range of sufficiently large λ (zoom-in, Figure 12A3). However, for very small values of λ, the condition $B \sim B'$ is not always satisfied, leading to the observation that, for $\lambda < 0.2$, the computationally computed bottleneck distances are larger than the theoretically expected values.

Second, we compute the bottleneck distance between input and output barcodes for various fixed values of λ, where the input barcodes arise by gradually decreasing the death time of one bar of an initial barcode B and thus increasing the distance to the next death time in the sequence (see Figure 12B1-2). All other bars of the initial barcode B remain the same. We observe that the bottleneck distance depends only on the value of λ and not on the distance between the bars of the input barcode.
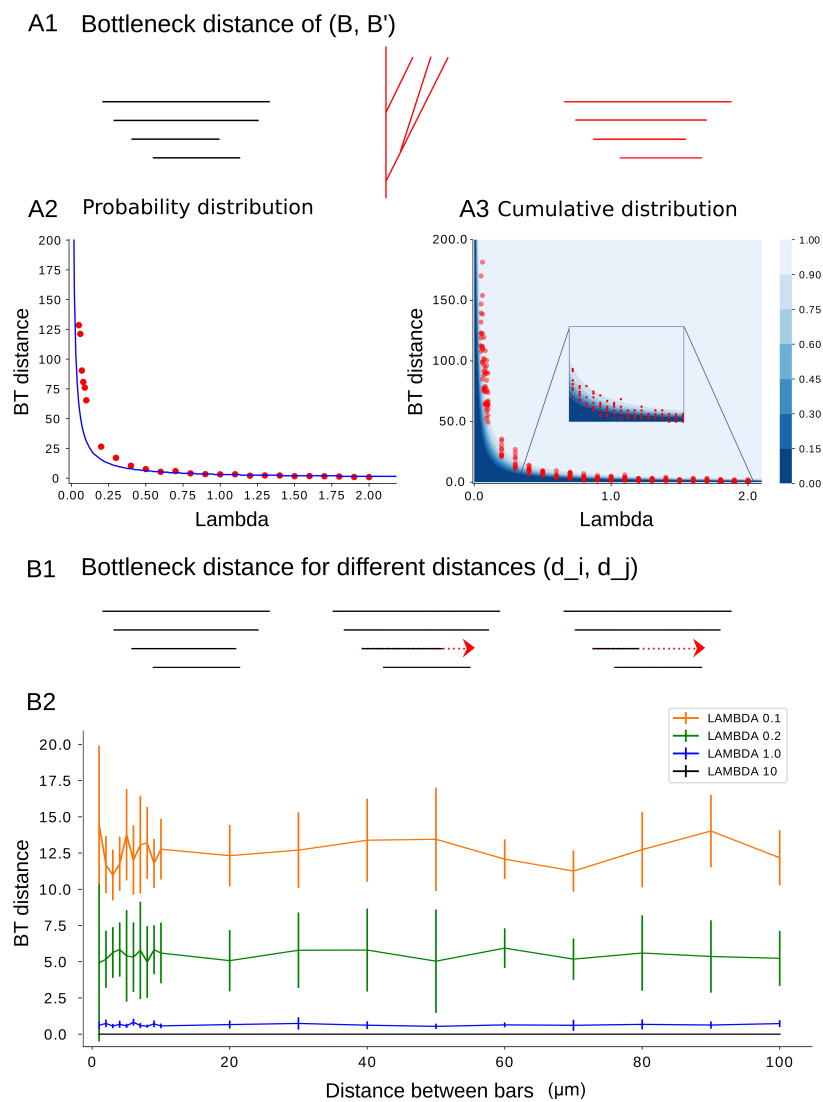
A1 Bottleneck distance of (B, B')

A2 Probability distribution

A3 Cumulative distribution

B1 Bottleneck distance for different distances (d_i, d_j)

B2

Figure 12. (**A1**–**A3**) Bottleneck distance as a function of λ. We compute the bottleneck distance between an input barcode B and an output barcode B' for $\lambda = 0.01 - 2$. (**A1**) From barcode B (in black), a tree (in red) is generated using the TNS which results in a new barcode $B' = \mathrm{TMD} \circ \mathrm{TNS}(B)$ (in red). (**A2**) The average bottleneck distance (red points) is compared to the expected mean of the probability distribution function found in Lemma 4 (blue curve). (**A3**) The bottleneck distances (red) are compared to the cumulative distribution probability for $0 < \epsilon < 200$ and $0 < \lambda < 2$ (blue). (**B1**–**B2**) bottleneck distance between B and B' as a function of distances between bars in B. (**B1**). We consider barcodes of the same permutation type for different distances between two bars (b_i, d_i) and (b_j, d_j) of the initial barcode B that are consecutive in the order of deaths. (**B2**) For each input barcode with increasing d_i, the distance between death times presented in x-axis, 100 synthesized barcodes are generated and the bottleneck distance between the input and output barcodes is computed (y-axis), which depends only on the value of λ and not on the distance between the bars.

4.2. Transposition Stability

As the TNS algorithm is a stochastic process, the image of any strict barcode $B = \{(b_i, d_i\}$ under the composite $\mathrm{TMD} \circ \mathrm{TNS}$ essentially always differs at least slightly from B. Here, we determine the probability that the orders of the death times of two specific bars of B and $\mathrm{TMD} \circ \mathrm{TNS}(B) = B' = \{(b_i', d_i')\}$ are different, so that B and $\mathrm{TMD} \circ \mathrm{TNS}(B)$ are not combinatorially equivalent, i.e., the associated permutations are different, as long as the birth times are not also transposed (Figure 13).
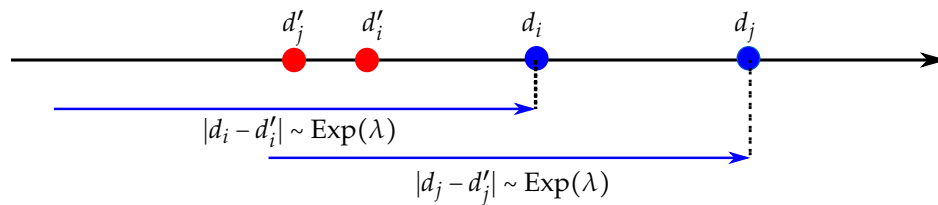
Figure 13. We are interested in the case where $d'_j < d'_i$ when we start from $d_i < d_j$. The distances $|d_i - d'_i|$ and $|d_j - d'_j|$ both follow an exponential law of parameter λ. The probability to terminate increases exponentially when approaching d_i and d_j, as represented by the blue arrows.

Lemma 5. *Let B be a strict barcode, and let* $(b_i, d_i), (b_j, d_j)$ *be bars of B such that* $d_i < d_j$. *Let* (b'_i, d'_i) *and* (b'_j, d'_j) *denote the corresponding bars in* $B' = \text{TMD} \circ \text{TNS}(B)$. *The probability that* $d'_j < d'_i$ *is*

$$\mathbb{P}(d'_j < d'_i) = \frac{1}{2}\exp(-\lambda(d_j - d_i)).$$

The TNS thus exhibits a sort of "transposition stability": the probability that the death times of two bars will be transposed decreases exponentially with the distance between those death times.

Proof. We compute $\mathbb{P}(d'_j < d'_i) = \mathbb{P}(d'_j < d'_i | d_i < d_j)$, the probability that $d'_j < d'_i$ given that $d_i < d_j$. Observe first that

$$\mathbb{P}(d'_j < d'_i) = \mathbb{P}(d_j + (d_i - d'_i) < d_i + (d_j - d'_j))$$
$$= \mathbb{P}(d_j + X_i < d_i + X_j) = \mathbb{P}(X_j - X_i > d_j - d_i).$$

Let $Y = X_j - X_i$. As X_j and X_i both follow an exponential law, the density function of their difference, Y, is given by $f_Y(t) = \frac{\lambda}{2}\exp(-\lambda t)$ when $t \geq 0$. Therefore,

$$\mathbb{P}(d'_j < d'_i) = \mathbb{P}(X_j - X_i > d_j - d_i) = \int_{d_j - d_i}^{\infty} f_Y(t)dt = \frac{1}{2}\exp(-\lambda(d_j - d_i)).$$

□

Remark 1. *Since the TNS is based on a stochastic process, multiple transpositions can occur when generating a new tree from a barcode. This makes it challenging to determine the overall probability of changing equivalence classes when computing the composite* TMD ∘ TNS. *Note that the TNS might also affect the birth order, but we will not discuss this possible effect in this paper. For the following experiments, the selected examples do not experience birth-switches, as the cells from which we computed the barcodes were chosen with sufficient gaps between birth values to avoid such switches.*

We perform the following computational experiment to evaluate the transposition stability results. We systematically vary the distance between two bars by changing the death time of a bar in the input barcode and compute the percentage of order changes that occur for different values of lambda (see Figure 14). We compare the theoretical results (solid lines) to the computational experiment (scatter points) for five different values of lambda. Note that, for this experiment, the birth times are chosen to be sufficiently distinct, and only the number of switches due to permutations that correspond to death changes are counted. The experimental results match the theoretical prediction with high accuracy, where we compute the error as the average distance of the computational points from the theoretical curve ($\lambda = 10, error = 0\%$, $\lambda = 5, error = 0.02\%$, $\lambda = 1, error = 0.5\%$, $\lambda = 0.5, error = 0.9\%$, $\lambda = 0.1, error = 3\%$, $\lambda = 0.05, error = 5\%$). Note that the error increases for smaller values of λ, due to the computational artifacts introduced when λ is small.
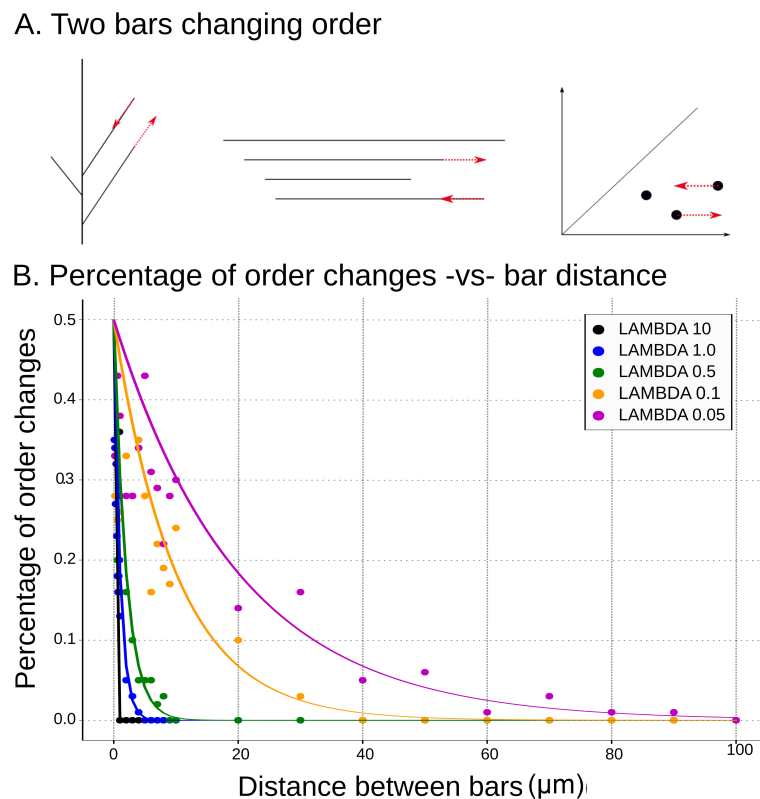
Figure 14. (**A**) Example of two bars changing order, which results in switching of classes. We show here a tree, its barcode, and the corresponding persistence diagram when two consecutive deaths switch their order. The impact of the change is illustrated by the red arrows; (**B**) percentage of order changes per 100 repetitions for varied distance between death times of two consecutive bars of the input barcode. Comparison of theoretical results (solid lines) to simulations (scatter plot) for different values of lambda.

5. Computational Exploration of Tree-Realization

In this section, we present computational results that illustrate the complex relationship between the equivalence class of a barcode and its possible tree-realizations.

We first present four results concerning all geometric trees: a computation of the distribution of tree-realization numbers across the set of equivalence classes of strict barcodes for various numbers of bars, a computation of the empirical distribution of combinatorial types of geometric trees in a synthesized population as a function of the equivalence class of the input barcode, a measurement of the diversity of TMD-equivalence classes among the realizations of a fixed barcode, and simulations of the fluctuations in tree-realization number that can occur as two bars gradually switch the order of their deaths.

We conclude by reporting on an experiment that sheds light on the distinguishing characteristics of "biological" geometric trees, i.e., those that arise from digital reconstructions of neurons.

5.1. The Distribution of Tree-Realization Numbers

We illustrate here how the number of tree-realizations of strict barcodes with $n + 1$ bars depends on n. In Figure 15, we present the distribution of tree-realization numbers across equivalence classes of barcodes with $n + 1$ bars, for $1 \leq n \leq 10$. As mentioned in Section 3.2, the tree-realization number is maximal for a fixed number of bars if and only if the barcode is strictly ordered. We observe an exponential-like behavior in the distribution of tree-realizations with the increase of the number of bars. We expect to link this behavior with intrinsic properties in the space of barcodes in a future work.
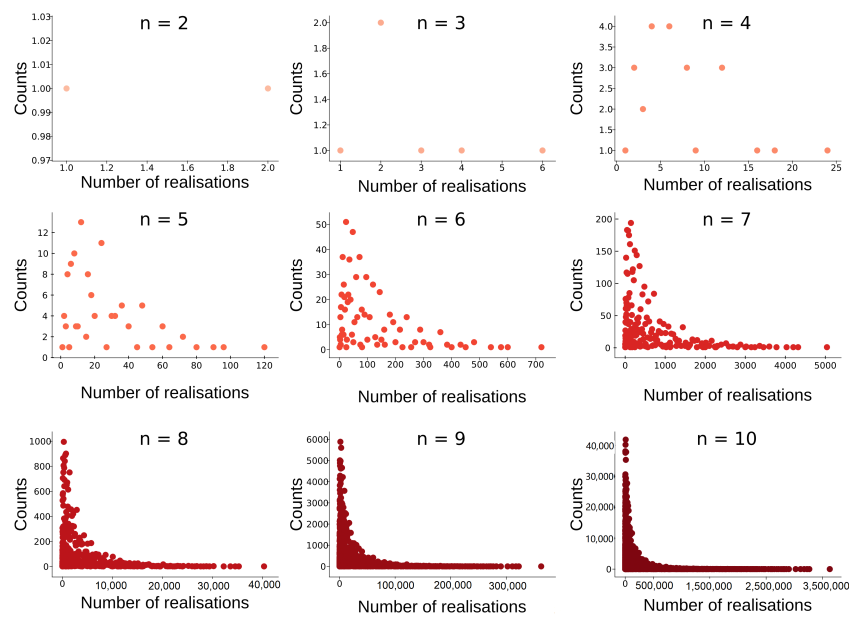
Figure 15. Histogram of tree-realization numbers for equivalence classes of barcodes with $n + 1$ bars ($1 \le n \le 10$). The maximal tree-realization number for a fixed number of bars can be achieved with exactly one equivalence class that of the strictly ordered barcode.

5.2. Empirical Distributions of Combinatorial Types of Trees

In this section, we explore computationally the probability to generate different combinatorial tree types (see Figure 5B (A–F)) with the TNS. We observe that this probability depends on the choice of the parameter λ (cf. Section 2.4). When $\lambda > 2$, the TNS is more likely to generate trees with all branches connected to the longest branch, due to the design of the algorithm. On the other hand, for smaller values of λ, the probability to generate different types of trees is approximately uniform.

Focusing on our preferred value of λ, we generated 1000 trees for $\lambda = 1$ and computed the percentage of each combinatorial tree type that is realized for each equivalence class of barcodes with four bars (Figure 16). There are six possible equivalence classes of strict barcodes with four bars and six combinatorial equivalence classes of geometric trees with four branches.
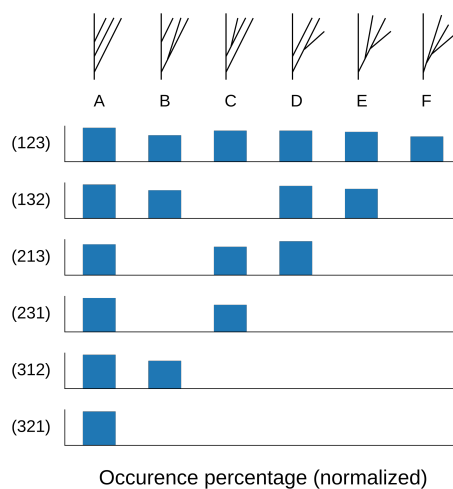

Figure 16. Empirical distribution (percentage of 1000 trees) of synthesized geometric trees with four branches by combinatorial tree type (A–F) for a given input barcode equivalence class (rows), when $\lambda = 1$. We observe that the distribution is approximately uniform.

5.3. Diversity of Realized TMD-Equivalence Classes

We now explore the diversity of TMD-equivalence classes of geometric trees that can be synthesized from a fixed barcode, in the particular case of the TMD of a biologically meaningful tree. For a fixed geometric tree with eight branches arising from a digital reconstruction of a layer 4 pyramidal cell, we computed its TMD barcode, to which we applied the TNS with $\lambda = 1$ to generate a set of 100 geometric trees. We computed the barcode-type and the persistence diagrams of the synthesized trees (Figure 17).

In agreement with the results presented in Figure 12, the persistence diagrams of the synthesized trees (Figure 17B, in blue) are essentially indistinguishable from the persistence diagram of the original barcode (Figure 17B, in red). On the other hand, the TMD-equivalence class of a synthesized tree is not necessarily equal to that of the original tree (Figure 17A). Here, we represent the TMD-equivalence class of a tree in terms of the permutation σ_B corresponding to the equivalence class of its TMD barcode B. In this graphical representation, we plot birth (or bifurcation) index k (on the x-axis) versus death (or termination) index $\sigma_B(k)$ (on the y-axis). A strictly ordered barcode would thus correspond to the set of points along the diagonal in this representation.
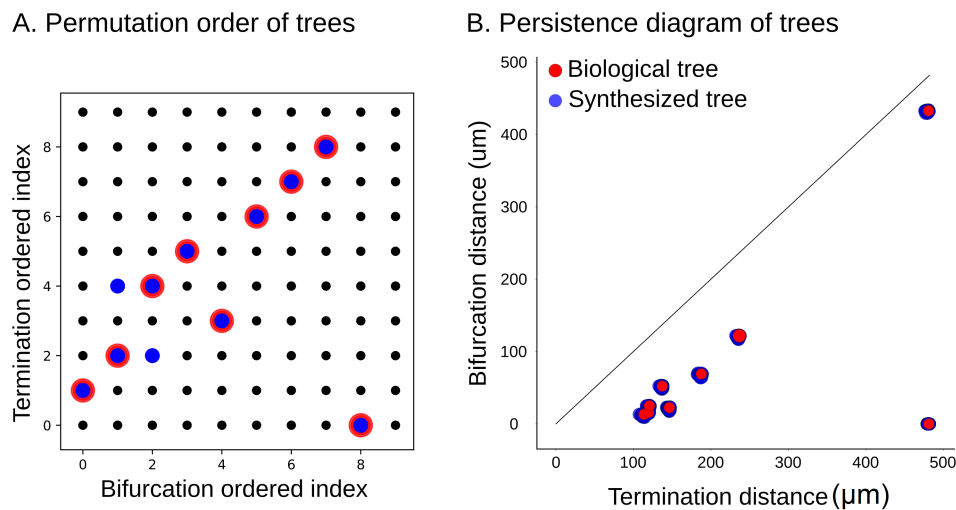

Figure 17. Barcode-equivalence class, represented by the corresponding permutation, (**A**) and persistence diagram (**B**) of 100 synthesized cells based on a geometric tree with eight branches, extracted from a layer 4 pyramidal cell. The barcode-equivalence classes of the synthesized trees (represented by blue dots) can differ from that of the original tree due to the stochastic nature of synthesis algorithm. The persistence diagrams of the synthesized trees ((**B**), blue) are essentially indistinguishable from those of the original tree ((**B**), red).

5.4. Statistics of Changing Classes

Motivated by the theoretical results on the probability to change classes in Section 4.2, we analyze here several simulations of gradual switching of death order of two bars and the resulting effect on tree realizations and their associated barcodes.

Let B be a strict barcode, and let $(b_i, d_i), (b_j, d_j)$ be bars of B such that $d_i < d_j$. By Lemma 5, for a fixed choice of the parameter λ (cf. Section 2.4), the probability that the order of these two bars is reversed in TMD $\circ$ TNS(B) depends exponentially on the distance between d_i and d_j:

$$\mathbb{P}(d'_j < d'_i) = \frac{1}{2} \exp(-\lambda(d_j - d_i)).$$

Thus, when the distance between d_i and d_j decreases, the probability that the order of bars changes increases. When there is no k such that $d_i < d_k < d_j$, Proposition 1 provides a formula for the

tree-realization number of the new barcode obtained when such a switch happens, as long as the order of the birth times is not also reversed.

We start with a geometric tree T extracted from a digital reconstruction of a neuron and compute its associated barcode $B = \text{TMD}(T)$. We choose two bars (b_i, d_i) and (b_j, d_j) of B that are consecutive in the order of deaths and divide the interval (d_i, d_j) into 50 equally sized subintervals. For $0 \le k \le 50$, let B_k be a barcode that is identical to B, except that its ith bar is $(b_i, d_i + k(d_j - d_i)/50)$ and its jth bar is $(b_j, d_j - k(d_j - d_i)/50)$. An interesting way to visualize this change is to think of B_k as migrating along the edge between B and the barcode with d_i and d_j permuted in the corresponding Cayley graph as k increases. The middle point of the edge corresponds to the non-strict barcode for which the two deaths are equal.

Let $B_k' = \text{TMD} \circ \text{TNS}(B_k)$ for all k. Because of the stochastic nature of the TNS algorithm, the permutation equivalence class of B_k' may be different from that of B_k. Figure 18 provides an example of this construction, where the barcodes are represented as persistence diagrams for visualization purposes (cf. Section 2.2).
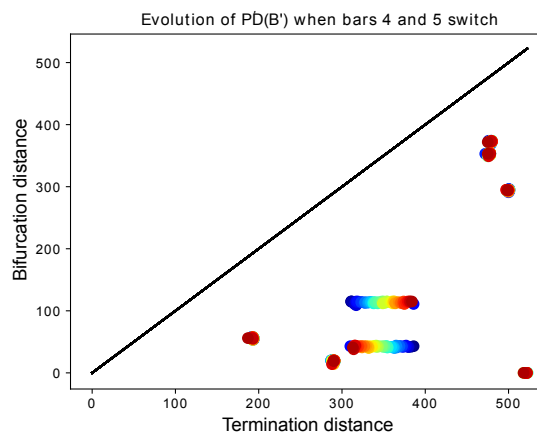

Figure 18. We begin with a barcode with eight bars. The death times d_{i_4} and d_{i_5} (i.e., the 4th and 5th largest death times) are slowly switching as k increases, represented by red-shifting of the color of the points in the persistence diagram. When $k = 0$ (in red), we have the original barcode B, and when $k = 50$ (in blue) we obtain a barcode identical to the original, except that (b_{i_4}, d_{i_4}) is replaced by (b_{i_4}, d_{i_5}) and (b_{i_5}, d_{i_5}) by (b_{i_5}, d_{i_4}).

To test whether the barcode B_k' is equivalent to the original barcode B, we compute its tree-realization number: if $\text{TRN}(B_k') \ne \text{TRN}(B)$, then B and B_k' are not equivalent. Note that, for the specific process that gives rise to B_k, it is likely that only the studied death-switch could lead to a difference between the tree-realization numbers of the input and output barcodes, unless two other deaths are too close to each other in the input barcode, as in the last row of Figure 19, which we explain further below. Therefore, the tree-realization number provides a very good indication of whether the switch of deaths took place, i.e., if $B \sim B_k'$ or not. Indeed, two barcodes that are the same except for two deaths that switched have different tree-realization numbers, cf. Proposition 1. Figure 19 shows several examples of the endpoint-switching process described above and the corresponding evolution of the tree-realization number as k increases. The corresponding permutation type and tree-realization number of each initial barcode, and the bars that are switched are listed in Table 2.
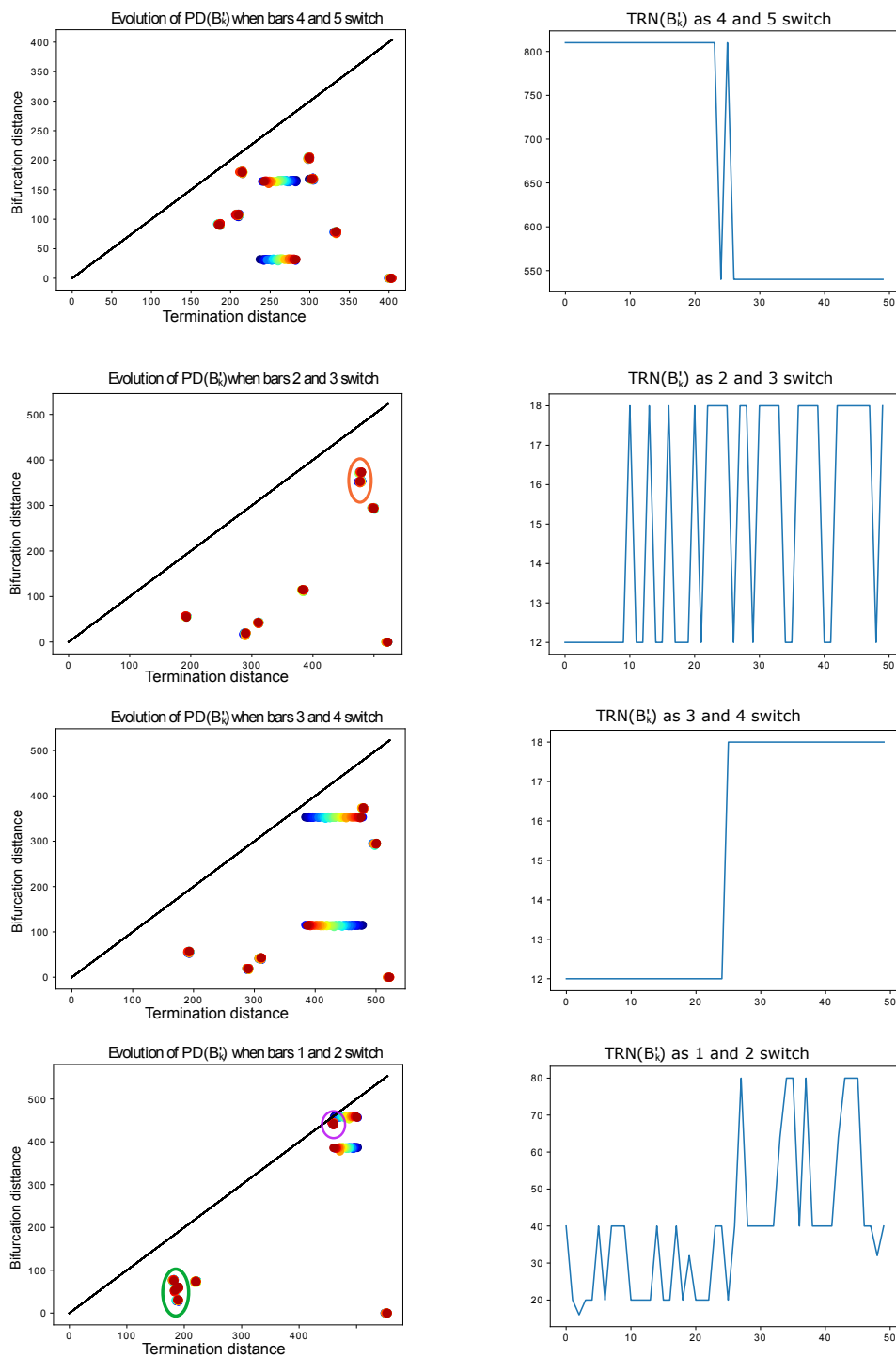
Figure 19. On the left, evolution of $\mathrm{PD}(B_k')$ as k increases (represented by red-shifting of the point color, from red $k = 0$ to blue $k = 50$), for various pairs of bars. When not clear, we circle in orange the two points that switch. On the right, the corresponding evolution of the tree realization number $\mathrm{TRN}(B_k')$ as k increases. For instance, as indicated in Table 2, the tree-realization number of B^1 is 810 and that of $\hat{B}^1 = B_{50}^1$ is 540. The barcodes B_k' exhibit the behavior described in Lemma 5, except for the last row, in which death times that are too close to each other (circled in purple and green) interfere with the process. Without this interference, the tree-realization numbers should oscillate between 20 and 40. When k gets close to 50 (blue), the death time d_{i_1} (largest death time) starts interfering with the third one d_{i_3} (circled in purple) in the tree synthesis process.

Table 2. For each example displayed in Figure 19, we list the permutation type and the tree-realization number of the original barcode B and of $\hat{B} = B_{50}$, and the indices of the bars that are switched. The superscript i in B^i indicates the corresponding row of Figure 19. For example, the largest death time of barcode B^1 is the second bar (in order of birth times), and its shortest death is the third one. When we switch the 4th and 5th (from largest to smallest) death times in B^1 and $\hat{B}^1$, the TRN changes from 810 to 540.

	Permutation	TRN	Bars That Switch
B^1	$(2,6,8,1,5,7,4,3)$	810	4 and 5
$\hat{B}^1$	$(2,6,8,5,1,7,4,3)$	540	4 and 5
B^2	$(5,7,6,4,2,1,3)$	12	2 and 3
$\hat{B}^2$	$(5,6,7,4,2,1,3)$	18	2 and 3
B^3	$(5,7,6,4,2,1,3)$	12	3 and 4
$\hat{B}^3$	$(5,7,4,6,2,1,3)$	18	3 and 4
B^4	$(8,6,7,4,3,1,2,5)$	20	1 and 2
$\hat{B}^4$	$(6,8,7,4,3,1,2,5)$	40	1 and 2

The top row of Figure 19 illustrates very well the exponential behavior of changing classes. When the distance between the death times of the two bars is very small (they are the closest when $k = 25$), the tree-realization number oscillates between its values for two different classes and otherwise stays constant.

The two middle rows come from the same biological tree and hence have the same starting barcode. The difference is that, in the second row, the death times of the two bars are already very close, leading to more frequent changes of equivalence class than in the third row.

The bottom row illustrates Remark 1 well. Since several bars are close to each other (represented here by several points in the persistence diagram that are close to each other), applying the TNS algorithm leads to frequent changes in equivalence classes, leading to the oscillatory behavior of the tree-realization number curve.

5.5. Tree-Realizations of Biological Barcodes

Since the original objective in developing the TMD was to classify digital reconstructions of neurons, it is natural to ask whether those barcodes that arise biologically exhibit any special characteristics compared to those arising from other sets of geometric trees. In Figure 20, we employ the graphical representation of permutations introduced in Section 5.3 to display as red dots all possible permutations corresponding to TMD-barcodes of biological trees with at most 30 branches arising from a population of digital reconstructions of neurons. Clearly, only a small fraction of the set of all possible permutations can be realized as the barcode-equivalence classes of geometric trees extracted from digital reconstructions of neurons, as every black dot in this plot can arise as a pair $(k, \sigma(k))$ for some permutation σ. In future work, we intend to study the biological relevance of this restriction.

To provide further insight into the subset of TMD-equivalence classes of biological geometric trees within the set of all possible TMD-equivalence classes, we computed the tree-realization number as a function of the number of bars, for a population of barcodes obtained by applying the TMD to geometric trees extracted from a population of digitally reconstructed neurons. We compared the values obtained to the maximum tree-realization number and to the tree-realization numbers of randomly chosen barcodes with the same number of bars (Figure 21). Interestingly, the barcodes that correspond to apical dendrites (relatively complex neural trees that perform significant processing tasks) exhibit a more narrow range of possible tree-realization numbers than random barcodes of the same size. On the other hand, barcodes of basal dendrites (less complex neuronal trees) exhibit tree-realization numbers similar to those of the randomly generated barcodes.
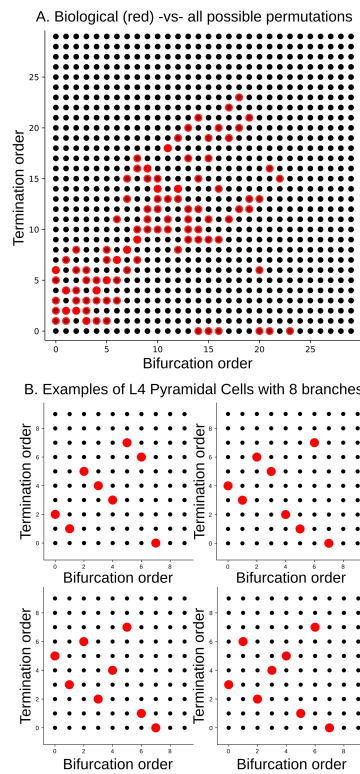
Figure 20. (**A**) TMD-equivalence classes of a population of biological geometric trees with at most 30 bars (red dots), represented by their associated permutations; (**B**) examples of TMD-equivalence classes of individual biological geometric trees with eight branches, extracted from layer 4 pyramidal cells (red dots).
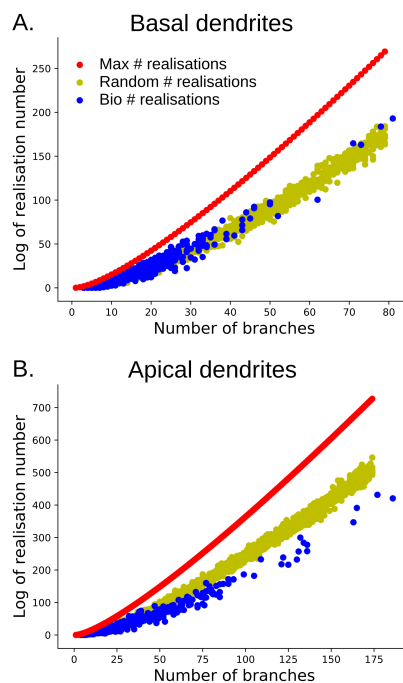

Figure 21. The log of the tree-realization number for barcodes with varying numbers of bars. (**A**) the log of tree-realization number for barcodes of basal dendrites (in blue) in comparison with random barcodes (in yellow) and the maximum tree-realization number ($n!$ for $n + 1$ bars) (in red); (**B**) the log of the tree-realization number for barcodes of apical dendrites (in blue) in comparison with random barcodes (in yellow) and the maximum maximum tree-realization number (in red).

6. Discussion

In this paper, we presented and analyzed two algorithms that are relevant in topological data analysis: the TMD, which encodes the structure of a geometric tree in a barcode, and the TNS, which generates a geometric tree from a barcode. We proved that, for a good choice of parameter, the TNS is robust with respect to small perturbations of barcodes; an analogous stability result for the TMD was established in [13].

We observed that ordering the bars in the persistence barcode according to birth times results in the natural association of a permutation to the barcode, based on death times, giving rise to a meaningful equivalence relation on the set of barcodes. We also introduced a natural, combinatorial equivalence relation on geometric trees. For any barcode, we analyzed the set of combinatorial equivalence classes of those geometric trees whose TMD corresponds to that barcode, providing a simple, explicit formula for its cardinality in terms of the permutation associated with the barcode. Cayley graphs of symmetric groups provide a useful visualization of how this cardinality varies as bars in the barcode are transposed.

We illustrated our theoretical results computationally. In addition, we computed the probability for the TNS to generate different combinatorial tree types from a fixed barcode and found it to be a function of the parameter λ on which the TNS depends, a result which can be explained only by the stochastic nature of the TNS algorithm. The stochastic nature of the TNS algorithm also leads to variation in the equivalence classes of barcodes associated by the TMD to the trees generated from a fixed barcode by the TNS. In particular, when starting with the TMD of a "biological" tree (i.e., arising from a digital neuron reconstruction) including bars with similar birth or death times, we observed an oscillatory behavior between two (or more) different classes states, increasing the variance of the generated trees.

We also initiated an analysis of the distinctive features of biological trees compared to random trees. We discovered that the barcodes associated by the TMD to trees representing neuronal morphologies represent a small fraction of possible equivalence classes of barcodes. It follows that the set of combinatorial types of geometric trees that are biologically realized is also constrained, indicating a biological preference for specific tree structures. There is much still to discover about which geometric or combinatorial features distinguish biological trees among all geometric trees and why.

In future work, we intend to further investigate the effect of different types of noise on the TNS algorithm. For instance, we have considered only the effect of transposing two bars, but other types of changes are certainly also relevant, such as investigating the effects of switching both births and deaths, as mentioned in Remark 1. On a more neuroscientific note, we intend to continue exploring the distinguishing characteristics of biological trees, with the goal of explaining the structural and functional reasons for the observed geometric and combinatorial constraints.

On the mathematical side, we are currently analyzing the structure on the space of barcodes revealed by the symmetric groups and determining what information can be extracted from the induced stratification of this space. This structure on the space of barcodes should also provide significant insights into the still somewhat mysterious space of geometric trees, which is of considerable interest to a wide range of mathematicians.

References

1. Frosini, P.; Landi, C. Size Functions and Morphological Transformations. *Acta Appl. Math.* **1997**, *49*, 85–104. [CrossRef]
2. Edelsbrunner, H.; Letscher, D.; Zomorodian, A. Topological Persistence and Simplification. *Discret. Comput. Geom.* **2002**, *28*, 511–533. [CrossRef]
3. Robins, V. Computational Topology for Point Data: Betti Numbers of α-Shapes. In *Morphology of Condensed Matter*; Springer: Berlin/Heidelberg, Germany, 2002.
4. Zomorodian, A.; Carlsson, G. Computing persistent homology. *Discret. Comput. Geom.* **2005**, *33*, 249–274 [CrossRef]
5. Verri, A.; Uras, C.; Frosini, P.; Ferri, M. On the use of size functions for shape analysis. *Biol. Cybern.* **2004**, *70*, 99–107. [CrossRef]
6. Carlsson, G. Topology and data. *Bull. Am. Math. Soc.* **2009**, *46*, 255–308. [CrossRef]
7. Harrington, H.; Feliu, E.; Wiuf, C.; Stumpf, M.P. Cellular compartments cause multistability in biochemical reaction networks and allow cells to process more information. *arXiv* **2012**, arXiv:1210.2993v1.
8. Byrne, H.; Harrington, H.; Muschel, R.; Reinert, G.; Stolz, B.J.; Tillmann, U. Topological Methods for Characterising Spatial Networks: A Case Study in Tumour Vasculature. *arXiv* **2019**, arXiv:1907.08711.
9. Martino, A.; Rizzi, A.; Mascioli, F.M.F. Supervised Approaches for Protein Function Prediction by Topological Data Analysis. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.
10. Gameiro, M.; Hiraoka, Y.; Izumi, S.; Kramár, M.; Mischaikow, K.; Nanda, V. A topological measurement of protein compressibility. *Jpn. J. Ind. Appl. Math.* **2015**, *32*, 1–17. [CrossRef]
11. Lee, Y.; Barthel, S.; Dłotko, P.; Moosavi, S.M.; Hess, K.; Smit, B. High-Throughput Screening Approach for Nanoporous Materials Genome Using Topological Data Analysis: Application to Zeolites. *J. Chem. Theory Comput.* **2018**, *14*, 4427–4437. [CrossRef]
12. Muszynski, G.; Kashinath, K.; Kurlin, V.; Wehner, M.F.; Prabhat, M. Topological Data Analysis and Machine Learning for Recognizing Atmospheric River Patterns in Large Climate Datasets. *Geosci. Model Dev.* **2019**, *12*, 613–628. [CrossRef]
13. Kanari, L.; Dłotko, P.; Scolamiero, M.; Levi, R.; Shillcock, J.; Hess, K.; Markram, H. A Topological Representation of Branching Neuronal Morphologies. *Neuroinformatics* **2018**, *16*, 3–13. [CrossRef] [PubMed]
14. Reimann, M.W.; Nolte, M.; Scolamiero, M.; Turner, K.; Perin, R.; Chindemi, G.; Dlotko, P.; Levi, R.; Hess, K.; Markram, H. Cliques of Neurons Bound into Cavities Provide a Missing Link between Structure and Function. *Front. Comput. Neurosci.* **2017**, *11*, 48. [CrossRef] [PubMed]
15. Sizemore, A.; Giusti, C.; Kahn, A.E.; Vettel, J.; Betzel, R.; Bassett, D. Cliques and cavities in the human connectome. *J. Comput. Neurosci.* **2017**, *44*, 115–145. [CrossRef] [PubMed]
16. Stolz, B.J.; Harrington, H.; Porter, M. Persistent homology of time-dependent functional networks constructed from coupled time series. *Chaos* **2017**, *27*, 047410. [CrossRef]
17. Kanari, L.; Ramaswamy, S.; Shi, Y.; Morand, S.; Meystre, J.; Perin, R.; Abdellah, M.; Wang, Y.; Hess, K.; Markram, H. Objective Morphological Classification of Neocortical Pyramidal Cells. *Cereb. Cortex* **2019**, *29*, 1719–1735. [CrossRef]
18. Oudot, S.; Solomon, E. Inverse Problems in Topological Persistence. *arXiv* **2018**, arXiv:1810.10813.
19. Curry, J.; Mukherjee, S.; Turner, K. How Many Directions Determine a Shape and other Sufficiency Results for Two Topological Transforms. *arXiv* **2018**, arXiv:1805.09782.
20. Belton, R.L.; Fasy, B.T.; Mertz, R.; Micka, S.; Millman, D.L.; Salinas, D.; Schenfisch, A.; Schupbach, J.; Williams, L. Reconstructing Embedded Graphs from Persistence Diagrams. *arXiv* **2020**, arXiv:1912.08913.
21. Kanari, L.; Dictus, H.; Chalimourda, A.; Van Geit, W.; Coste, B.; Shillcock, J.; Hess, K.; Markram, H. Computational Synthesis of Cortical Dendritic Morphologies. *Cell* **2020**. [CrossRef]
22. Curry, J. The Fiber of the Persistence Map for Functions on the Interval. *arXiv* **2019**, arXiv:1706.06059.
23. Markram, H.; Muller, E.; Ramaswamy, S.; Reimann, M.W.; Abdellah, M.; Sanchez, C.A.; Ailamaki, A.; Alonso-Nanclares, L.; Antille, N.; Arsever, S.; et al. Reconstruction and Simulation of Neocortical Microcircuitry. *Cell* **2015**, *163*, 456–492. [CrossRef] [PubMed]
24. Aslangul, C.; Pottier, N.; Chvosta, P.; Saint-James, D. Directed random walk with spatially correlated random transfer rates. *Phys. Rev. E* **1993**, *47 3*, 1610–1617. [CrossRef]

25. Galton, F.; Watson, H.W. On the Probability of the Extinction of Families. *J. Anthropol. Inst. Great Br. Irel.* **1875**, *4*, 138–144.

26. Koene, R.; Tijms, B.; van Hees, P.; Postma, F.; Ridder, A.; Ramakers, G.; Pelt, J.; Ooyen, A.V. NETMORPH: A Framework for the Stochastic Generation of Large Scale Neuronal Networks with Realistic Neuron Morphologies. *Neuroinformatics* **2009**, *7*, 195–210. [CrossRef] [PubMed]