Wasserstein Distances, Geodesics and Barycenters of Merge Trees
Mathieu Pont, Jules Vidal, Julie Delon, Julien Tierny

To cite this version:
Mathieu Pont, Jules Vidal, Julie Delon, Julien Tierny. Wasserstein Distances, Geodesics and Barycenters of Merge Trees. IEEE Transactions on Visualization and Computer Graphics, 2021, 28 (1), pp.291-301. 10.1109/TVCG.2021.3114839. hal-03288210

HAL Id: hal-03288210
https://hal.science/hal-03288210
Submitted on 16 Jul 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Wasserstein Distances, Geodesics and Barycenters of Merge Trees

Mathieu Pont, Jules Vidal, Julie Delon and Julien Tierny

Fig. 1. The merge trees of three members (a-c) of the Isabel ensemble (wind velocity) concisely and visually encode the number and salience of the features of interest found in the data (eyewall and region of high speed wind, blue and cyan). They also describe how these features are globally connected in the data. In these trees, branches with a low persistence (less than 20% of the function range) are shown with small white arcs. The pointwise mean for the three members (d) exhibits 5 salient maxima (due to distinct eyewall locations, blue, cyan and black) and its merge tree is not representative of the input trees (containing at most 3 large features). In contrast, the Wasserstein barycenter (e) is representative of the input trees, with a number and persistence of large branches that better match the input trees (a-c). Our framework for distances, geodesics and barycenters enables a variety of merge tree based applications, including (f) feature tracking, (g) temporal reduction – key frames are automatically identified (white insets) and deleted merge trees (blue insets) are accurately reconstructed with geodesics – and (h) ensemble clustering and summarization – the clusters and centroids automatically computed by our approach provide a visual summary of the main trends of features found in the ensemble.

Abstract—This paper presents a unified computational framework for the estimation of distances, geodesics and barycenters of merge trees. We extend recent work on the edit distance [106] and introduce a new metric, called the Wasserstein distance between merge trees, which is purposely designed to enable efficient computations of geodesics and barycenters. Specifically, our new distance is strictly equivalent to the $L^1$-Wasserstein distance between extremum persistence diagrams, but it is restricted to a smaller solution space, namely, the space of rooted partial isomorphisms between branch decomposition trees. This enables a simple extension of existing optimization frameworks [112] for geodesics and barycenters from persistence diagrams to merge trees. We introduce a task-based algorithm which can be generically applied to distance, geodesic, barycenter or cluster computation. The task-based nature of our approach enables further accelerations with shared-memory parallelism. Extensive experiments on public ensembles and SciVis contest benchmarks demonstrate the efficiency of our approach – with barycenter computations in the orders of minutes for the largest examples – as well as its qualitative ability to generate representative barycenter merge trees, visually summarizing the features of interest found in the ensemble. We show the utility of our contributions with dedicated visualization applications: feature tracking, temporal reduction and ensemble clustering. We provide a lightweight C++ implementation that can be used to reproduce our results.

Index Terms—Topological data analysis, merge trees, scalar data, ensemble data

1 INTRODUCTION

Modern datasets, acquired or simulated, are continuously gaining in geometrical complexity, thanks to the ever-increasing accuracy of acquisition devices or computing power of high performance systems. This geometrical complexity makes interactive exploration and analysis difficult, which challenges the interpretation of the data by the end users. This motivates the definition of expressive data abstractions, capable of capturing the main features of the data into concise representations, which visually convey the most important information to the users.

In that context, Topological Data Analysis (TDA) [32] forms a family of generic, robust, and efficient techniques whose utility has been demonstrated in a number of visualization tasks [55] for revealing the implicit structural patterns present in complex datasets. Examples of popular application fields include turbulent combustion [23, 51, 65], material sciences [40, 53, 54], nuclear energy [71], fluid dynamics [61], bioimaging [4, 20, 26], quantum chemistry [16, 47, 76] or astrophysics [103, 105]. Among the data abstractions developed in TDA (see Sec. 1.1), the merge tree [25], which describes the global structure of the connected components of the sub-level sets of scalar datasets (Fig. 2), is a prominent example in the visualization literature [20, 23, 26].

In practice, in addition to the increasing geometrical complexity of datasets, users are also confronted to the emergence of ensemble datasets, where a given phenomenon is not described with only one dataset, but with a collection of datasets, called ensemble members. Regarding topological features, a topological data abstraction such as the merge tree can be computed for each ensemble member (possibly in situ [9, 11]). Then, a major challenge for end users is the interpretation of the resulting ensemble of merge trees. To address this, a statistical analysis framework for merge trees is needed, requiring several key building blocks, such as: distances (to compare merge trees), geodesics (to visualize optimum transitions between them), and barycenters (to visualize one merge tree representative of a set). These building blocks have been well studied for persistence diagrams [64, 112, 113]. However,
perspective diagrams suffer from a lack of specificity (Fig. 3), which can prevent the identification of distinct feature trends within the ensemble.

This paper addresses this problem by introducing a unified computational framework for the automatic computation of distances, geodesics, barycenters and clusters of merge trees. In particular, we extend recent work on the edit distance [106] and introduce a new metric, called the Wasserstein distance between merge trees, which is purposely designed to enable efficient computations of geodesics (i.e. length minimizing morphings) and barycenters. In that regard, our work can be interpreted as an extension of previous work on the edit distance [106], to adapt it to the optimization strategy previously developed for the computation of barycenters of persistence diagrams [112]. We present efficient, task-based algorithms using shared-memory parallelism, resulting in the computation of distances, geodesics and barycenters in practical times for real-life datasets. We illustrate the utility of each of our contributions in dedicated visualization tasks. First, we show that our distance computation algorithm can be used for a merge-tree based tracking of features through time. Second, we show that our framework for computing geodesics between merge trees can be used for the reliable sub-sampling of temporal sequences of merge trees. Third, we illustrate the utility of our barycenters for clustering ensemble members based on their merge trees, while providing cluster centroids which visually summarize the main features of interest present in each cluster.

1.1 Related work

The literature related to our work can be classified into three main groups, reviewed in the following: (i) uncertainty visualization, (ii) ensemble visualization, and (iii) topological methods for ensembles.

(i) Uncertainty visualization: Variability in data can be modeled and encoded in several ways. In particular, uncertain datasets capture variability by modeling each point of the domain as a random variable, whose variability is explicitly modeled by an estimator of an a priori probability density function (PDF). The analysis of uncertain data is a notoriously challenging problem in visualization, described in several surveys [1, 21, 59, 70, 80, 96]. Early techniques focused on estimating the entropy of the random variables [94], their correlations [88] or their gradient variations [86]. The positional uncertainty of level sets has been studied for several interpolation schemes and PDF models [5–7, 87, 90–93, 100]. Similarly, the positional uncertainty of critical points has been studied for Gaussian [66, 78, 79, 85] or uniform distributions [17, 50, 107]. A general limitation of existing methods for uncertain data is their dependence on the specific PDF model for which they have been designed. This reduces their usability for ensemble data, where the PDF estimated from the ensemble members can follow an arbitrary, unknown model. Also, most existing techniques for uncertain data do not consider multi-modal PDF models, which is however necessary when several, distinct trends are present in the ensemble data.

(ii) Ensemble visualization: Another way to model and encode variability in data consists in considering ensemble datasets. In this setting, the variability is directly encoded by an ensemble of empirical observations (i.e. the members of the ensemble). Current approaches to ensemble visualization typically compute some geometrical objects describing the features of interest (level sets, streamlines, etc), for each member of the ensemble. Then, an aggregation phase estimates a representative object for the resulting ensemble of geometrical objects. For instance, spaghetti plots [30] are a typical example for studying level-set variability, especially for weather data [95, 99]. More specifically, box-plots [115] describe the variability of contours and curves [72]. For flow ensembles, Hummel et al. [58] introduce a Lagrangian framework for classification purposes. Clustering techniques have been investigated, to identify the main trends, and their variability, in ensembles of streamlines [42] and isosurfaces [43]. However, only few approaches have applied this overall aggregation strategy to topological objects. Favelier et al. [39] and Athawale et al. [8] introduced approaches for analyzing the variability of critical points and gradient separatrices respectively. Several techniques attempted to generate an aggregated contour tree from an ensemble based on overlap-driven heuristics [63, 117]. Recently, Lohfink et al. [68] introduced an approach for the consistent layout of multiple contour trees, to support effective visual comparisons between the contour trees of the distinct members of an ensemble.

(iii) Topological methods: Concepts and algorithms from computational topology [32] have been investigated, adapted and extended by the visualization community for more than twenty years [55, 118]. Popular topological representations include the persistence diagram [32, 36] (Sec. 2.2), which represents the population of features of interest in function of their salience, and which can be computed via matrix reduction [13, 32]. The Reeb graph [18], which describes the connectivity evolution of level sets, has also been widely studied and several efficient algorithms have been documented [31, 81, 83, 111], including parallel algorithms [49]. Efficient algorithms have also been documented for its variants, the merge and contour trees [25, 108] (Sec. 2.3), and parallel algorithms have also been described [2, 27, 48, 69]. The Morse-Smale complex [22, 33, 34], which depicts the global behaviour of integral lines, is another popular topological data abstraction in visualization [29]. Robust and efficient algorithms have been introduced for its computation [52, 97, 102] based on Discrete Morse Theory [44].

Distance metrics, which are necessary ingredients for the computation of barycenters, have been studied for most of the above objects. Inspired by the literature in optimal transport [60, 73], the Wasserstein distance between persistence diagrams [32] (Sec. 2.2) and its variant the Bottleneck distance [36] have been extensively studied. They are based on a bipartite assignment problem, for which exact [75] and approximate [15, 62] implementations are publicly available [110]. Several similarity measures have been introduced for Reeb graphs [57] and their variants [98]. However, since these measures are not distance metrics (the preservation of the triangle inequality is not specifically enforced), they do not seem conducive to barycenter computation. Stable distance metrics between Reeb graphs [12] and merge trees [74] have been studied from a theoretical point of view but their computation, following an exponential time complexity, is not tractable for practical datasets in general, except if reliable correspondence labels between the nodes of the trees are provided on the input [46], which is not practical either for large ensembles. Distances with polynomial time computation algorithms have also been investigated. Similarly to our overall strategy, Beketayev et al. [14] focus on a dual representation,
1.2 Contributions

This paper makes the following new contributions:

1. A practical distance metric between merge trees: We extend recent work on the edit distance [106] and introduce a new distance between merge trees, which, in contrast to previous work, is purposely designed to enable efficient computations of geodesics and barycenters. It can be computed efficiently, it has acceptable practical stability and it has a strong connection to established metrics, which eases its interpretation. Specifically, it can be interpreted as a variant of the $L^2$-Wasserstein distance for persistence diagrams, for which we constrain the underlying search space to account for the additional structural information provided by the merge tree.

2. A simple approach for computing geodesics between merge trees: Given our new metric, we present a simple approach for computing geodesics between merge trees. It uses a simple linear interpolation of the assignments resulting from our new metric, enabling the exact computation of geodesics in linear time. This follows from previous work on persistence diagram geodesics [112] and it is made possible thanks to a new, local normalization procedure, guaranteeing the topological consistency of the interpolated trees.

3. An approach for computing barycenters between merge trees: Our method for geodesics between merge trees enables a straightforward adaptation of previous optimization strategies for persistence diagram barycenters [112], resulting, to our knowledge, in the first approach for the computation of barycenters of merge trees.

4. Unified computational framework: We present a unified computational framework for the estimation of distances, geodesics, barycenters, and clusters of merge trees. In particular, we introduce an efficient, task-based algorithm adapted from previous work on edit distances [106, 121], which is generally applicable to any of the above tasks. Our algorithm supports shared-memory parallelism, allowing for further accelerations in practice.

Applications: We illustrate the utility of each of our contributions with dedicated visualization tasks, including feature tracking, temporal reduction and ensemble clustering and summarization.

Implementation: We provide a lightweight C++ implementation of our algorithms that can be used for reproduction purposes.

2 Preliminaries

This section presents the theoretical background of our work. It contains definitions adapted from the Topology ToolKit [110]. We refer the reader to textbooks [32] for an introduction to computational topology.

2.1 Input data

The input data is an ensemble of $N$ piecewise linear (PL) scalar fields $f_i: \mathcal{M} \to \mathbb{R}$, with $i \in \{1, \ldots, N\}$, defined on a PL $d$-manifold $\mathcal{M}$, with $d \leq 3$ in our applications. The sub-level set of $f_i$, noted $f_i^{-1}(w) = \{ p \in \mathcal{M} \mid f_i(p) \leq w \}$, is defined as the pre-image of $(-\infty, w]$ by $f_i$. The super-level set of $f_i$ is defined symmetrically: $f_i^{+1}(w) = \{ p \in \mathcal{M} \mid f_i(p) > w \}$. As $w$ continuously increases, the topology of $f_i^{-1}(w)$ changes at specific vertices of $\mathcal{M}$, called the critical points of $f_i$ [10]. In practice, $f_i$ is enforced to contain only isolated, non-degenerate critical points [35, 37]. Critical points are classified by their index $\delta_i$: 0 for minima, 1 for 1-saddles, $d-1$ for $(d-1)$-saddles and $d$ for maxima.

2.2 Persistence diagrams

The persistence diagram is a visual summary of the topological features (i.e. connected components, independent cycles, voids) of $f_i^{-1}(w)$. Specifically, each topological feature of $f_i^{-1}(w)$ can be associated with a unique pair of critical points $(c, c')$, corresponding to its birth and death. The Elder rule [32] states that critical points can be arranged according to this observation in a set of pairs, such that each critical point appears in only one pair $(c, c')$, with $f_i(c) < f_i(c')$ and $\mathcal{F}(c) = \mathcal{F}(c') - 1$. For instance, if two connected components of $f_i^{-1}(w)$ meet at a critical point $c'$, the younger component (created last, in $c$ dies, in favor of the older one (created first). Then the persistence diagram, noted $\mathcal{D}(f_i)$, embeds each pair to a single point in 2D at coordinates $(f_i(c), f_i(c'))$. The persistence of a pair is given by its height $f_i(c') - f_i(c)$. Then, the persistence diagram provides a visual overview of the features of interest of a dataset (Fig. 2), where salient features stand out from the diagonal while pairs corresponding to noise are located in the vicinity of the diagonal. Note that, in addition to its interest as a visual summary, the persistence diagram captures all the information about the persistent homology groups of the data [32].

Given two diagrams $\mathcal{D}(f_i)$ and $\mathcal{D}(f_j)$, a pointwise distance, noted $d_q$ (with $q > 0$), can be introduced in the 2D birth/death space between two points $p_i = (x_i, y_i) \in \mathcal{D}(f_i)$ and $p_j = (x_j, y_j) \in \mathcal{D}(f_j)$:

$$d_q(p_i, p_j) = \left( |x_j - x_i|^q + |y_j - y_i|^q \right)^{1/q} = \|p_i - p_j\|_q.$$

(1)
By convention, $d_q(p_i, p_j)$ is set to zero if both $p_i$ and $p_j$ exactly lie on the diagonal ($x_i = y_i$ and $x_j = y_j$). Let $P_i$ be a subset of the off-diagonal points of $D(f_i)$ and $\overline{P}_i$ its complement (i.e. the other off-diagonal points of $D(f_i)$ not in $P_i$). Let $(\Phi, \overline{P}_i, \overline{P}_j)$ be a partial assignment between $D(f_i)$ and $D(f_j)$, i.e. a bijective map between $P_i$ and a subset of off-diagonal points $P_j$ of $D(f_j)$, with complement $\overline{P}_j$ (Fig. 5a). Then, the $L^q$-Wasserstein distance, noted $W^q_{\overline{P}_i}(D(f_i), D(f_j))$ can be introduced as:

$$W^q_{\overline{P}_i}(D(f_i), D(f_j)) = \min_{(\Phi, \overline{P}_i, \overline{P}_j) \in \Phi} \left( \sum_{p \in P_i} d_q(p_i, \Phi(p_i))^q \right)^{1/q}$$

This augmentation (Fig. 5a) preserves the distance, while making the assignment problem balanced, and thus easily solvable with traditional algorithms [15, 75] (with $P_i = D(f_i) \setminus \{p_i\}$, see (2)), or to its diagonal projection, $\Delta(p) = \frac{x_i + y_i}{2}$, denoting the removal of the corresponding feature from $D(f_i)$ or $D(f_j)$ (lines 3 and 4). Intuitively, the Wasserstein metric optimizes a matching between the two diagrams, and evaluates their distance given the resulting mismatch. In practice, $D(f_i)$ and $D(f_j)$ are augmented into $D(f_i)$ and $D(f_j)$ [62], by injecting the diagonal projections of one diagram into the other (Fig. 5a):

$$D(f_i) = D(f_i) \cup \{\Delta(p) \mid p \in P_i\}$$
$$D(f_j) = D(f_j) \cup \{\Delta(p) \mid p \in P_i\}$$

This section introduces our new distance metric between merge trees, which is specifically designed for the subsequent computation of geodesics (Sec. 4) and barycenters (Sec. 5). For this, we bridge the gap between the edit distance between merge trees [106] and existing work addressing the computation of geodesies and barycenters for persistence diagrams according to the $L^2$-Wasserstein distance [112].

### 3.1 Overview

The end goal of our work is the computation of barycenters of merge trees. For this, we extend the edit distance $D_E$ [106] (formalized in Appendix 1, additional material), to make it fit the optimization strategy used for barycenters of persistence diagrams [112]. Our key idea consists in transforming $D_E$ such that it becomes strictly equivalent to the $L^2$-Wasserstein distance of persistence diagrams, thus given a restricted space of possible assignments, constrained by the structure of the input trees $T(f_i)$ and $T(f_j)$, hence its name Wasserstein distance between merge trees. Then, thanks to this compatibility with the $L^2$-Wasserstein distance, the assignments resulting from our metric can be directly used for interpolation-based geodesic and barycenter computations (Secs. 4 and 5). Overall, our strategy involves four major modifications to the edit distance $D_E$ [106], detailed in the remainder of this section:

1. To consider assignments between persistence pairs instead of merge tree nodes, we consider an edit distance between the BDTs $T(f_i)$ and $T(f_j)$ (Sec. 2.3) instead of the input merge trees $T(f_i)$ and $T(f_j)$ (as done with $D_E$). This is described in Sec. 3.2.

2. We constrain the assignment search space to the space of rooted partial isomorphisms. Specifically, similarly to $D_E$, we enforce the assignment of disjoint subtrees of $T(f_i)$ to disjoint subtrees of $T(f_j)$. Moreover, in contrast to $D_E$, we additionally extend this constraint by enforcing the destruction of entire subtrees upon the destruction of their root. These two constraints together enforce assignments describing isomorphisms between rooted subtrees of $T(f_i)$ and $T(f_j)$. Such isomorphisms pave the way for interpolation-based geodesics. This is described in Secs. 3.2, 3.3 and 4.1.

3. We introduce a cost model based on the Euclidean distance $d_2$ to enable geodesic computation by linear interpolation of the assignments in the 2D birth/death space. This is described in Secs. 3.2 and 4.1.

4. We finally extend our metric with a local normalization term, which enforces nested birth-death values, along the interpolation of the assignments, for nested branches. This is described in Sec. 4.2.
3.2 Definition and properties

Given two input merge trees, \( \mathcal{T}(f_i) \) and \( \mathcal{T}(f_j) \), we first consider their BDTs \( \mathcal{B}(f_i) \) and \( \mathcal{B}(f_j) \) (Sec. 2.3). Let \( B_i \) be a subset of the nodes of \( \mathcal{B}(f_i) \) and \( \overline{B_i} \) its complement. Note that each node in \( B_i \) corresponds to a persistence pair of \( \mathcal{B}(f_i) \). Let \( (\phi', \overline{B}, \overline{B}_f) \) be a partial assignment between \( B_i \) and a subset \( B_j \) of the nodes of \( \mathcal{B}(f_j) \) (with complement \( \overline{B}_f \)). Then we introduce the \( L^2 \)-Wasserstein distance \( W_2^2(\mathcal{B}(f_i), \mathcal{B}(f_j)) \) between the BDTs \( \mathcal{B}(f_i) \) and \( \mathcal{B}(f_j) \) of the merge trees \( \mathcal{T}(f_i) \) and \( \mathcal{T}(f_j) \) as:

\[
W_2^2(\mathcal{B}(f_i), \mathcal{B}(f_j)) = \min_{(\phi', \overline{B}, \overline{B}_f) \in \Phi'} \left( \sum_{b_i \in B_i} \gamma(b_i \rightarrow \phi'(b_i))^2 \right) \tag{6}
\]

where \( \Phi' \) is the space of constrained partial assignments mapping disjoints subtrees of \( \mathcal{B}(f_i) \) to disjoints subtrees of \( \mathcal{B}(f_j) \) and mapping entire subtrees to the empty tree \( \emptyset \) if their root is itself mapped to \( \emptyset \). Then, given the \( k^{th} \) direct child of \( b_i \), noted \( b_i^k \), it follows that \( b_i^k \) either maps through \( \phi' \) to a direct child of \( \phi'(b_i) \in \mathcal{B}(f_j) \) (then \( b_i, b_i^k \in B_i \)) or to the empty tree \( \emptyset \) (then the subtree rooted in \( b_i^k \), noted \( \mathcal{B}(b_i^k) \), belongs to \( \overline{B}_f \)). This further implies that the rooted subtrees \( B_i \subseteq \mathcal{B}(f_i) \) and \( B_j = \phi'(B_i) \subseteq \mathcal{B}(f_j) \) are isomorphic and we call \( (\phi', \overline{B}, \overline{B}_f) \) a rooted partial isomorphism.

Unlike \( \Phi \) (see Appendix 1) but similarly to \( W_2^2 \) (Eq. 2), the cost of each operation (mapping, line 18, destruction, line 18, and creation, line 19) is squared, and the square root of the sum of the squared costs is considered as the overall distance.

Next, we define the edit costs as follows (we recall that each branch \( b_i \in \mathcal{B}(f_i) \) exactly coincides with a persistence pair \( p_i \in \mathcal{T}(f_i) \)):

\[
\begin{align*}
\gamma(b_i \rightarrow \phi'(b_i)) &= d_2(b_i, \phi'(b_i)) &\text{for } \phi'(b_i) \neq \emptyset \\
\gamma(b_i \rightarrow \emptyset) &= d_2(b_i, \Delta(b_i)) &\text{for } \phi'(b_i) = \emptyset \\
\gamma(\emptyset \rightarrow b_j) &= d_2(\Delta(b_j), b_j) \tag{9}
\end{align*}
\]

3.3 Computation

This section describes our algorithm for the recursive exploration of the search space \( \Phi \) (Eq. 6). It is based on the same recursive traversal as Zhang’s algorithm [121], which we simplify as our search space is significantly more constrained. Specifically, as detailed in Appendix 3, our distance evaluation between subtrees (Eq. 11) involves fewer solutions and it is restricted to subtrees rooted at identical depth only.

Given the subtree \( \mathcal{B}(f_i, b) \) of \( \mathcal{B}(f_i) \) (rooted in \( b \)) and \( b \) the \( n^{th} \) direct child of \( b \) in \( \mathcal{B}(f_i, b) \), the distance between the subtree \( \mathcal{B}(f_j, b) \) and the empty tree \( \emptyset \) is then obtained recursively by:

\[
W_2^2(\mathcal{B}(f_i, b), \emptyset) = \left( \gamma(b \rightarrow \emptyset)^2 + \sum_k W_2^2(\mathcal{B}(f_i, b^k), \emptyset) \right)^{1/2}. \tag{10}
\]
4 WASSERSTEIN GEOEDESICS BETWEEN MERGE TREES

This section introduces our approach for the efficient computation of geodesics between merge trees, according to the metric $W_2^p$ (Sec. 2). For this, we leverage the rooted partial isomorphism resulting from the distance computation, as well as linear interpolations of the matchings, as introduced for persistence diagrams [112].

4.1 Definition and properties

Given two input merge trees $T(f_i)$ and $T(f_j)$, our approach to geodesic computation (Fig. 7) simply consists in linearly interpolating the rooted partial isomorphism $(\phi', \overline{B}_f, \overline{T}_f)$ resulting from the optimization involved in the computation of $W_2^p(\overline{B}_f, \overline{T}_f)$ (Eq. 6). In particular, given the two BDTs $\overline{B}_f$ and $\overline{T}_f$, the interpolated BDT, noted $\overline{B}_f(\alpha \rightarrow f_i)$ with $\alpha \in [0,1]$ such that $\overline{B}_f(\alpha \rightarrow f_i) = \overline{B}_f(f_i)$ and $\overline{B}_f(\alpha \rightarrow f_j) = \overline{B}_f(f_j)$, is obtained by considering the union of:

1. the linear interpolation $B_A \subseteq \overline{B}_f(\alpha \rightarrow f_i)$, between the nodes $B_i \subseteq \overline{B}_f(f_i)$ and these of $B_j \subseteq \overline{B}_f(f_j)$, given the isomorphism $\phi'$ (the trees $B_i$, $B_j$ and $B_A$ are then isomorphic, Fig. 7):
   \[ b(\alpha) = (1-\alpha) b + \alpha \phi'(b) \quad \forall b \in B_i \]  

2. the linear interpolation of the destruction of the subtrees $B_B$ noted $\overline{B}_B \subseteq \overline{B}_f(\alpha \rightarrow f_i)$ (\overline{T}_f$ and $\overline{B}_B$ are also isomorphic):
   \[ b(\alpha) = (1-\alpha) b + \alpha \Delta(b) \quad \forall b \in B_B \]  

3. the linear interpolation of the creation of the subtrees $B_C$, noted $\overline{B}_C \subseteq \overline{B}_f(\alpha \rightarrow f_i)$ (\overline{T}_f$ and $\overline{B}_C$ are also isomorphic):
   \[ b(\alpha) = (1-\alpha) \Delta(b) + \alpha b \quad \forall b \in B_C \]  

Similarly to the distance $W_2^p$ between persistence diagrams, since the edit costs involved in the distance $W_2^p$ are Euclidean distances in the birth/death space (Eq. 9), the interpolated branches $b(\alpha)$ of $\overline{B}_f(\alpha \rightarrow f_i)$ can be efficiently computed with the simple linear interpolations described above. As detailed in Appendix 5, the resulting interpolated tree $\overline{B}_f(\alpha \rightarrow f_i)$ is indeed a geodesic given $W_2^p$.

4.2 From branch decomposition trees to merge trees

The previous section described the computation of geodesics between BDTs, given $W_2^p$. In this section, given an interpolated BDT $\overline{B}_f(\alpha \rightarrow f_i)$, we describe how to retrieve the corresponding merge tree $\overline{T}_f(\alpha \rightarrow f_i)$ (i.e. a merge tree whose BDT is indeed equal to $\overline{B}_f(\alpha \rightarrow f_i)$).

A requirement for an arbitrary BDT $\overline{T}_f$ to be the valid BDT of a merge tree $T$ is that subtrees of $\overline{T}_f$ need to respect a nesting condition on their birth/death (i.e. $x, y$) values (to respect the Elder rule, Sec. 2.2).

In particular, given a direct child $b_A^k$ of a branch $b_A \subseteq \overline{B}_f(\alpha \rightarrow f_i)$, we need to guarantee that $[y_{b_A}, y_{b_A}^k] \subseteq [y_{b_A}, y_{b_A}]$. While this is guaranteed by construction for the subset $B_A \subseteq \overline{B}_f(\alpha \rightarrow f_i)$ ($B_A$ is isomorphic to $B_i$ and $B_j$), this is not necessarily the case for the subsets of $\overline{B}_f(\alpha \rightarrow f_i)$ involved in subtree creation or destruction ($\overline{B}_B$ and $\overline{B}_C$, Sec. 4.1). In particular, since the branches involved in destructions map independently to the diagonal (Eq. 15), it is possible that the above nesting condition is not respected along their interpolation. This is shown in Fig. 8c (red interpolation), where the resulting merge tree, $\overline{T}_f(\alpha \rightarrow f_i)$, becomes disconnected and hence invalid (i.e. $\overline{T}_f(\alpha \rightarrow f_i)$, Fig. 8d, is connected and not equal to the BDT of $\overline{T}_f(\alpha \rightarrow f_i)$ from Fig. 8c).

In the following, we introduce a pre-processing step for the trees $\overline{B}_f$ and $\overline{T}_f$ (together with its inverse post-processing step), which we call local normalization, which addresses this issue and guarantees the above nesting condition, even in case of destruction/creation.

Given a direct child $b_A^k$ of a branch $b_A \in \overline{T}_f(f_i)$, we consider the following local birth/death normalization $N(\alpha) = (\alpha_i(\alpha), \alpha_j(\alpha))$:

\[ \alpha_i(\alpha) = (x_{b_A} - x_{b_A})/(y_{b_A} - y_{b_A}) \]

\[ \alpha_j(\alpha) = (y_{b_A} - y_{b_A})/(y_{b_A} - y_{b_A}) \]

Once this pre-process is recursively completed, the Wasserstein distance $W_2^p$ between the locally normalized BDTs, noted $N(\overline{B}_f(f_i))$ and $N(\overline{T}_f(f_i))$ is computed as described in Sec. 3.3. Then, the interpolation of the locally normalized BDTs, noted $N(\overline{B}_f(\alpha \rightarrow f_i))$ is evaluated as described in Sec. 4.1. Next, the local normalization is recursively reverted to turn $N(\overline{B}_f(\alpha \rightarrow f_i))$ back into $\overline{B}_f(\alpha \rightarrow f_i)$, by explicitly evaluating $N^{-1}(\alpha_i(\alpha))$ for each branch $b_A^k \in N(\overline{B}_f(\alpha \rightarrow f_i))$. Now, even in case of branch destruction, by construction, the birth/death interval of each interleaved branch $N(b_a)$, noted $[N_i(b_a), N_j(b_a)]$, is included in $[0,1]$ (since $\Delta(N(b_a)) \subseteq [N_i(b_a), N_j(b_a)] \subseteq [0,1]$). Therefore, after reverting the local normalization, we have the guarantee that $[x_{b_A}, y_{b_A}] \subseteq [x_{b_A}, y_{b_A}]$ for all the branches $b_A$ of $\overline{B}_f(\alpha \rightarrow f_i)$.

At this stage, $\overline{B}_f(\alpha \rightarrow f_i)$ indeed respects the nesting condition on the birth/death values of all its subtrees. Then, given the dual relation between merge trees and BDTs, the merge tree $\overline{T}_f(\alpha \rightarrow f_i)$ can be simply obtained by creating a vertical branch for each node $b_A$ of $\overline{B}_f(\alpha \rightarrow f_i)$ and connecting them according the arcs of $\overline{B}_f(\alpha \rightarrow f_i)$, as illustrated in Fig. 8 (right). The distance $W_2^p$ between $N(\overline{T}_f(f_i))$ and $N(\overline{B}_f(f_i))$ then still describes a metric between $\overline{T}_f(f_i)$ and $\overline{B}_f(f_i)$, such that $\overline{B}_f(\alpha \rightarrow f_i)$ is indeed on a geodesic (see Appendix 6).

Note that the local normalization shrinks all the input branches to the interval $[0,1]$, irrespective of their original persistence. To mitigate this effect, we introduce a pre-processing step on the input BDTs, which are then isomorphic (Fig. 7): the interpolated merge trees (black) in (c) is disconnected, unlike the interpolated BDT (d). Our local normalization (Sec. 4.2) addresses this issue by enforcing nested birth/death values for nested branches. This results in a valid interpolated merge tree (e) whose BDT is indeed equal to the interpolated BDT (d).

Fig. 7. Our geodesic computation extends interpolation-based geodesics from persistence diagrams (a) to merge trees (b). The interpolated BDT $\overline{B}_f(\alpha \rightarrow f_i)$ is obtained by linear interpolation (with local normalization) of the partial isomorphism $\phi'$ in the birth/death space. In the data, the feature matching (dashed lines) induced by $\phi'$ with $W_2^p(b)$ better preserves the global structure of the data than $\phi$ with $W_2^p$ (a, red crossing).

Fig. 8. Given two scalar fields $f_i$ (a) and $f_j$ (b), a simple interpolation of the birth/death values of the branches of their BDTs may result in inconsistencies upon branch destruction (red): the interpolated merge tree (black) in (c) is disconnected, unlike the interpolated BDT (d). Our local normalization (Sec. 4.2) addresses this issue by enforcing nested birth/death values for nested branches. This results in a valid interpolated merge tree (e) whose BDT is indeed equal to the interpolated BDT (d).

Fig. 9. Visual comparison between our barycenter (Sec. 5) and the 1-center of Yan et al. [119,120]. Left: an ensemble is created with an outlier member $f_j$ (red, 7 persistent branches) and 10 noisy versions of a field $f_i$ (4 persistent branches). Right: planar view of the ensemble computed by multi-dimensional scaling of $W_2^p$. The barycenter computed with our approach (cyan) is more similar to the merge trees of $f_j$ (same number and persistence of large branches) and hence better captures the overall trend of the ensemble, despite the presence of the outlier $f_j$ (red sphere).
Our distance $W_2^\mathcal{B}$ (Sec. 2) is identical to $W_2^\mathcal{B}$, but with a smaller search space, restricted to rooted partial isomorphisms. This enabled an extension of interpolation-based geodesics from persistence diagrams to merge trees (Sec. 4). Given these two components, the strategy presented by Turner et al. [112] for minimizing the Fréchet energy over the space of persistence diagrams can be directly extended to our framework. For this, we consider an algorithm that resembles a Lloyd relaxation [67], and which alternates an (i) assignment and an (ii) update procedure. First, the candidate $\mathcal{B}$ is initialized at an arbitrary tree of $\mathcal{J}_\mathcal{B}$. Then the assignment step (i) computes an optimal assignment $(\phi^*_i, B_i, B_\mathcal{F})$ between $\mathcal{B}$ and each tree $\mathcal{B}(f_i) \in \mathcal{J}_\mathcal{B}$. Next, the update step (ii) updates the candidate $\mathcal{B}$ to a position in $\mathcal{B}$ which minimizes $F(\mathcal{B})$ under the current set of assignments $(\phi^*_i, B_i, B_\mathcal{F})_{i=1,...,N}$. This is achieved by moving each branch $b \in \mathcal{B}$ (in the birth/death space) to the arithmetic mean of the assignments (by generalizing the interpolation defined in Eqs. 12, 13, and 14, to more than two trees):

$$b \leftarrow \frac{1}{N} \sum_{i=1,...,N} \begin{cases} \phi^*_i(b) & \text{if } b \in B_i \\ \Delta(b) & \text{if } b \in \overline{B_i} \\ b & \text{if } b \in \overline{B_\mathcal{F}}. \end{cases}$$

This overall assignment/update sequence is then iterated (as discussed in Appendix 7, each iteration of this sequence decreases the Fréchet energy constructively). In our implementation, the algorithm stops and returns the barycenter estimation $\mathcal{B}^\ast$ when the Fréchet energy decreased by less than 1% between two consecutive iterations. Given $\mathcal{B}^\ast$, we obtain its dual merge tree $\mathcal{F}^\ast$ as described in Sec. 4.2. Fig. 9 illustrates a barycenter computed with this strategy for a toy example.

### 5.3 Parallelism

The $N$ assignment problems (between the candidate $\mathcal{B}$ and the trees of the set $\mathcal{J}_\mathcal{B}$, Sec. 5.2) are independent and can be computed in parallel. However, this naive strategy is subject to load imbalance, as the input trees can have different sizes. Hence, each iteration would be bounded by the sequential execution of the largest of the $N$ assignment problems.

We address this issue by leveraging the task-based parallelization of our distance computation algorithm (Sec. 3.4). In particular, we use a single task pool for all of the $N$ assignment problems. Then, the task environment picks up at runtime the tasks to compute irrespective of their tree of origin, and place them on different threads. This fine scheduling granularity has the beneficial effect of triggering the execution of the tasks of a new assignment problem while a first problem is reaching completion (and thus exploiting less threads, Sec. 3.4). This improves thread load imbalance and thus increases the overall parallel efficiency.

### 6 Applications

The section illustrates the utility of our contributions (distances, geodesics, and barycenters) in concrete visualization tasks (Fig. 1).

#### 6.1 Branch matching for feature tracking

Our distance (Sec. 3.3) relies on the optimization of a partial isomorphism between the input BDTs. Then, the resulting matchings can be used to track features in time-varying data, as studied for persistence diagrams [104]. Fig. 11 illustrates this on a temporal sequence (SciVis contest 2008 [109]). Since $W_2^\mathcal{B}$ considers persistence pairs individually, it can generate inconsistent matchings with a typical incorrect crossing in the feature tracking (already visible on synthetic data, Fig. 7). Our distance $W_2^\mathcal{B}$ improves this aspect by better preserving the global structure of the data, thanks to our more constrained, merge-tree driven,
Appendix 8). This enables the reliable visualization of time-varying features (i.e. which have been matched to the same centroid branch, Fig. 13).

Figs 12 and 13 present clustering examples obtained with this strategy on an acquired ensemble [41] and cosmology ensemble [56]. Our approach correctly assigns the members to each cluster (distinct colors in the bottom planar view, generated in a post-process by multi-dimensional scaling of $W^p_2$). Our centroids (larger spheres in the planar view) provide a visual summary of the features of interest (matching colors) for each cluster.

To understand the main trends within an ensemble, in terms of features of interest, it may be desirable to cluster the ensemble by grouping members with a similar topological profile. For this, we adapt the k-means algorithm [28, 38] to the problem of clustering merge trees. In particular, this can be easily achieved by using our merge tree barycenter computation algorithm (Sec. 5) as the centroid estimation routine of k-means, and by using $W^p_2$ (Sec. 2) to measure the distance between merge trees. Note that in practice, our entire computational framework is implemented in this single clustering algorithm (with a unique task pool), as the above clustering generalizes the barycenter problem ($k = 1$) as well as the geodesic and distance problems ($N = 2$).

Our experiments were performed on a variety of simulated and acquired 2D and 3D ensembles used in previous work [39] (vorticity and sea surface height) or extracted from past SciVis contests: 2004 (wind velocity magnitude [114]), 2006 (wavefront velocity magnitude [77]), 2008 (ion concentration [109]), 2014 (sulfur dioxide concentration [41]), 2015 (dark matter density [56]), 2016 (salt concentration [45]), 2017 (pressure [116]), 2018 (matter density [84]). A detailed specification of these ensembles is provided in Appendix 9.

### 7.1 Time performance

The time complexity of our algorithm for exploring the search space of $W^p_2$ (Sec. 3.3) is similar to that of the edit distance [106, 121]. It takes $O(|A|^2)$ steps in practice, with $|A|$ the number of nodes in the input BDTS (in our implementation, each local forest assignment problem is solved with the efficient Auction approximation [15] with default parameters). Once $W^p_2(A(f_i), A(f_j))$ is computed, the computation of a point on the geodesic (Sec. 4) between $A(f_i)$ and $A(f_j)$ is obtained in $O(|A|)$ steps. Regarding our barycenter computation algorithm (Sec. 5), each of its iterations takes $O(|N||A|^2)$ steps. Tab. 1 evaluates the practical time performance of our computational framework for the barycenter computation (which includes itself distance and geodesic computations). In sequential mode, we observe that the running time is indeed a function of the number of ensemble members ($N$) and the average size of the trees ($|A|$). It is slightly slower for $W^p_2$ than for $W^F_2$, but runtimes remain comparable overall. In parallel, speedups are the most important for the largest examples. However, the iterative nature of our barycenter optimization algorithm seems to limit parallel efficiency globally (the end of each iteration still constitutes a strong synchronization). For the smaller examples, the cost of the task runtime seems to become non-negligible in comparison to the actual computation, resulting in moderate speedups. Still, our parallelization significantly reduces runtimes overall, with less than 3 minutes of computation on average and at most 15 minutes for the largest examples.

### 7.2 Framework quality

$W^p_2$ is indeed a distance metric (Appendix 2). It is more discriminative than $W^F_2$ (i.e. $W^p_2 \geq W^F_2$, Sec. 2, Fig. 3). Fig. 14 evaluates empirically its stability. For this, given a scalar field $f_i$, a noisy version $f_j$ is created such that $\|f_i - f_j\|_\infty \leq \varepsilon$, for increasing values of $\varepsilon$. Then, we observe the evolution of $W^p_2(A(f_i), A(f_j))$, as a function of $\varepsilon$ (Fig. 14, right), to estimate how $W^p_2$ varies under input perturbations. For $\varepsilon_1 = 1$, we have $W^p_2 = W^F_2$ (Sec. 2) and the curve evolves nearly linearly ($W^p_2$ is stable [112]). For other $\varepsilon_1$ values, the curves indicate clear transition points (colored dots) before which $W^p_2$ evolves nearly linearly too. This indicates that for reasonable noise levels (smaller than the $\varepsilon$ value of each transition point, vertical lines), $W^p_2$ is also stable and that only mild increases of $\varepsilon_1$ result in fast shifts of these transition points (to an accepted noise level of 64% at $\varepsilon_1 = 0.15$). This illustrates overall...
Table 1. Running times (in seconds, 10 run average) of our approach for the barycenter computation, with respect to $W^2_F$ ($\varepsilon = 1$, Sec. 3.3, sequential) and to our new metric $W^2_D$ (sequential, then with 20 cores).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$W^2_F$</th>
<th>$W^2_D$ (c)</th>
<th>$W^2_D$ (a)</th>
<th>$W^2_D$ (b)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advection Instability [149]</td>
<td>1,269</td>
<td>718.51</td>
<td>818.71</td>
<td>818.71</td>
<td>4.89</td>
</tr>
<tr>
<td>Cloud processes [118] (2D)</td>
<td>2,089</td>
<td>64.09</td>
<td>124.99</td>
<td>124.99</td>
<td>3.55</td>
</tr>
<tr>
<td>Viscous Inflow [85] (1D)</td>
<td>52</td>
<td>4.38</td>
<td>5.12</td>
<td>5.12</td>
<td>1.34</td>
</tr>
<tr>
<td>Dark matter [56] (3D)</td>
<td>40</td>
<td>76.77</td>
<td>104.02</td>
<td>104.02</td>
<td>2.56</td>
</tr>
<tr>
<td>Volcanic eruption [41] (2D)</td>
<td>89</td>
<td>171.13</td>
<td>48.52</td>
<td>48.52</td>
<td>3.59</td>
</tr>
<tr>
<td>Isolation forest [109] (1D)</td>
<td>15</td>
<td>41.84</td>
<td>12.10</td>
<td>12.10</td>
<td>3.44</td>
</tr>
<tr>
<td>Isolation forest [109] (1D)</td>
<td>12</td>
<td>122.52</td>
<td>219.61</td>
<td>219.61</td>
<td>1.80</td>
</tr>
<tr>
<td>Earthquake [77] (1D)</td>
<td>12</td>
<td>14.68</td>
<td>44.01</td>
<td>44.01</td>
<td>3.00</td>
</tr>
<tr>
<td>Isabel [114] (3D)</td>
<td>12</td>
<td>168.16</td>
<td>300.88</td>
<td>300.88</td>
<td>1.80</td>
</tr>
<tr>
<td>Starting Vortex [39] (2D)</td>
<td>64</td>
<td>41.58</td>
<td>41.58</td>
<td>41.58</td>
<td>1.54</td>
</tr>
<tr>
<td>Sea Surface Height [39] (2D)</td>
<td>48</td>
<td>1,787.12</td>
<td>1,787.12</td>
<td>1,787.12</td>
<td>1.00</td>
</tr>
<tr>
<td>Vortex Street [39] (2D)</td>
<td>45</td>
<td>22.11</td>
<td>1.71</td>
<td>1.71</td>
<td>1.33</td>
</tr>
</tbody>
</table>

that the stability of $W^2_D$ can indeed be controlled with $\varepsilon_1$ and that small values already lead to stable results for reasonable noise levels. A detailed empirical analysis of the other two parameters of our approach ($\varepsilon_2$, $\varepsilon_3$, Sec. 4.2) is provided in Appendix 10 (supplemental material).

Next, we study the practical relevance of $W^2_D$ by evaluating our clustering performance. For this, each ensemble of Tab. 1 is associated with a ground truth classification (distinct phases of a time-varying phenomenon), distinct input parameters, etc), by following the companion specifications [41, 45, 56, 77, 84, 109, 114, 116]. Clustering performance is evaluated with accepted scores, namely the normalized mutual information and adjusted rand index ($NMI$, $ARI$). When using our barycenters (Sec. 5), our clustering approach (Sec. 6.3) achieves a perfect classification for all ensembles ($NMI = ARI = 1$). These scores decrease to $NMI = 0.78$ and $ARI = 0.69$ on average when using, within $k$-means, a barycenter of persistence diagrams [112] ($\varepsilon_1 = 1$), and to $NMI = 0.73$ and $ARI = 0.56$ when using the 1-center of Yan et al. [120] (obtained with the authors' implementation [119], using leaf labels generated by our distance computation, Sec. 2). This simply confirms experimentally that $k$-centers in general are not suited for clustering tasks. A standard clustering approach (multi-dimensional scaling to $kD$ followed by $k$-means) using the distance $D_k$ [106] achieves lower average scores than our approach, with $NMI = 0.89$ and $ARI = 0.85$ on average. Overall, this confirms that $D_k$ induces more discriminative classifiers than $W^2_D$, and that our metric $W^2_D$ further improves that.

Fig. 15 shows the evolution of the Fréchet energy for our barycenter algorithm (Sec. 5) for various $\varepsilon_1$ values. In practice, the algorithm stops when the Fréchet energy decreases by less than 1% between consecutive iterations, which occurs early in the process.

7.3 Limitations

The search space associated with our metric $W^2_D$ is constrained to rooted partial isomorphisms. Then, if a matching exists between two BDTs (i.e. if they are not both degraded when optimizing $W^2_D$), it has to match their roots together. In other words, $W^2_D$ nearly always matches the most persistent branch of the two trees together, which might be too restrictive (in particular for feature tracking applications). Note however, that $W^2_D$ behaves equivalently: the most persistent branch of $\mathcal{B}(f_1)$ corresponds to the component of $f_1^{-1}(w)$ created in the global minimum of $f_1$, which in principle has infinite persistence and which is typically treated separately when evaluating $W^2_D$. Similarly to Sridharamurthy et al. [106], saddle swap instabilities are handled in our approach by a pre-processing step which merges adjacent saddles (controlled by $\varepsilon_1$). An alternative would consist in exploring the space of all possible branch decompositions (not necessarily persistence-driven), as studied by Beketayev et al. [14]. However, the search space would then become significantly larger. Moreover, the nesting of birth/death values within the BDTs would no longer be guaranteed, which is however a key property which we exploit in our framework (Sec. 4). When computing barycenters of persistence diagrams, Vidal et al. [113] showed that the optimization could be drastically accelerated by introducing persistence pairs progressively along the iterations, while implicitly maintaining previous assignments at each initialization. We leave the study of such a progressive strategy for future work, although the fact that $W^2_D$ handles many small assignment problems (unlike $W^2_F$) indicates that such a strategy may result in only modest gains for merge trees. Fig. 14 provides an empirical evaluation of the stability of $W^2_D$. Similarly to Sridharamurthy et al. [106], we believe that the theoretical investigation of the stability of $W^2_D$ goes beyond the scope of this paper and we leave it for future work.

8 Conclusion

In this paper, we presented a computational framework for the estimation of distances, geodesics and barycenters of merge trees, with applications to feature tracking, temporal reduction and ensemble clustering and summarization. Our approach filled the gap between the edit distance [106] and existing optimization frameworks for persistence diagrams [112]. Our work enables faithful interpolations of merge trees (Fig. 10) and the generation of merge trees representative of a set (Figs. 1, 12, and 13). Moreover, our task-based algorithm enables automatic barycenter computations within minutes for real-life ensembles.

A natural direction for future work is the extension of our framework to other topological data representations, such as Reeb graphs or Morse-Smale complexes. However, the question of defining relevant and computable metrics for these objects is still an active research debate. Moreover, as illustrated by this paper, extending existing metrics to make them conducive to efficient geodesic computation further requires additional efforts. We believe our work is an important practical step towards the definition of a larger statistical framework on the space of merge trees. In the future, based on our framework, we will study the definition of more sophisticated statistical indexes (for instance by investigating a notion of covariance matrix for merge trees), to support even more advanced analyses of large-scale ensemble data.

ACKNOWLEDGMENTS

This work is partially supported by the European Commission grant ERC 2019-COG “TORI” (ref. 863464, https://erc-tori.github.io/).
References


APPENDIX

1 THE EDIT DISTANCE BETWEEN MERGE TREES [106]

This section formalizes the edit distance introduced by Sridharanmurti et al. [106] (Sec. 1.1) and discusses some of its technical aspects which make it not conducive to interpolation-based geodesics (Sec. 1.2).

1.1 Definition

The edit distance between two merge trees $\mathcal{T}(f)$ and $\mathcal{T}(f)$, noted $D_k(\mathcal{T}(f), \mathcal{T}(f))$, is defined as follows. Let $N_i$ be a subset of the nodes of $\mathcal{T}(f)$ and $N_i$ its complement. Let $\Phi$ be a partial assignment between $N_i$ and a subset $N_j$ of the nodes of $\mathcal{T}(f)$ (with complement $N_j$). Then $D_k(\mathcal{T}(f), \mathcal{T}(f))$ is given by:

$$D_k(\mathcal{T}(f), \mathcal{T}(f)) = \min_{\Phi^0, \Phi^0, \Phi^0} \left( \sum_{n \in N_i} \gamma(n \rightarrow \Phi^0(n)) (15) \right) + \sum_{n \in N_i} \gamma(n \rightarrow \emptyset) (16) + \sum_{n \in N_j} \gamma(\emptyset \rightarrow n) (17)$$

where $\Phi^0$ is the space of constrained partial assignments (i.e. $\Phi^0$ maps disjoint subtrees of $\mathcal{T}(f)$ to disjoint subtrees of $\mathcal{T}(f)$) and where $\gamma$ refers to the cost for: (i) mapping a node $n \in N_i$ to a node $\Phi^0(n) \in N_j$ (line 15), (ii) deleting a node $n \in N_i$ (line 16), and (iii) creating a node $n \in N_i$ (line 17), $\emptyset$ being the empty tree.

Zhang [121] introduced a polynomial time algorithm for computing a constrained sequence of edit operations with minimal edit distance. (Eq. 15), and showed that the resulting distance is indeed a metric if each cost $\gamma$ for the above three edit operations is itself a metric (non-negativity, identity, symmetry, triangle inequality). Sridharanmurti et al. [106] exploited this property to introduce their metric, by defining the following distance-based cost model, where $p_i$ and $p_j$ stand for the persistence pairs containing the nodes $n \in N_i$ and $n \in N_j$.

$$\gamma(n \rightarrow n) = \min(d_e(p_i, p_j), \gamma(n \rightarrow \emptyset) + \gamma(\emptyset \rightarrow n))$$

$$\gamma(n \rightarrow \emptyset) = d_e(p_i, p_j)$$

$$\gamma(\emptyset \rightarrow n) = d_e(\Delta(p_i), p_j)$$

In our work, we introduce an alternative edit distance which further adheres to the $L^2$-Wasserstein distance between persistence diagrams.

1.2 Interpolation

As shown in the main manuscript (Fig. 4), the linear interpolation of $D_k$’s matchings does not describe a shortest path (i.e. it generates inaccurate midpoints). A key technical reason for this is that $D_k$ involves assignments between nodes of (input merge trees) and not persistence pairs. This has several consequences. First, given two input trees $\mathcal{T}(f)$ and $\mathcal{T}(f)$, $D_k$’s matchings may assign a saddle node in $\mathcal{T}(f)$ to an extremum node in $\mathcal{T}(f)$, resulting in inconsistent interpolations in the data (from a valley to a peak). Second, $D_k$’s matchings can possibly assign two nodes in $\mathcal{T}(f)$ belonging to a single persistence pair of $f_i$ to nodes in $\mathcal{T}(f)$ belonging to distinct persistence pairs in $f_i$. This second phenomenon further challenges interpolation-based geodesics.

2 $W_2^2$ IS A METRIC

As further described in the main manuscript, given two merge trees $\mathcal{T}(f)$ and $\mathcal{T}(f)$ and their branch decomposition trees (BDTs) $\mathcal{B}(f)$ and $\mathcal{B}(f)$, the dissimilarity measure $W_2^2(\mathcal{B}(f), \mathcal{B}(f))$ is given by:

$$W_2^2(\mathcal{B}(f), \mathcal{B}(f)) = \min_{\Psi^0, \Psi^0, \Psi^0} \left( \sum_{b \in B} \gamma(b \rightarrow \Psi^0(b))^2 (18) \right) + \sum_{b \in B} \gamma(b \rightarrow \emptyset)^2 + \sum_{b \in B} \gamma(\emptyset \rightarrow b)^2 (19)$$

where $\Psi^0$ is an isomorphism between $B_i \subseteq \mathcal{B}(f)$ and $B_i \subseteq \mathcal{B}(f)$.

In this section, we argue that $W_2^2$ is a metric. $W_2^2(\mathcal{B}(f), \mathcal{B}(f))$ is always non-negative (the costs $\gamma$ are squared). $W_2^2(\mathcal{B}(f), \mathcal{B}(f))$ is symmetric (creation and creation costs are symmetric, lines 18 and 19). $W_2^2(\mathcal{B}(f), \mathcal{B}(f)) = 0$ if and only if all costs $\gamma = 0$, which only happens if $\mathcal{B}(f) = \mathcal{B}(f)$ (the identity is included in $\Psi^0$).

We now argue that $W_2^2$ preserves the triangle inequality, given three trees $\mathcal{B}(f)$, $\mathcal{B}(f)$ and $\mathcal{B}(f)$. For this, we follow a classical approach which we detail here for the sake of completeness. First, we argue that a composition of (optional) partial rooted isomorphisms (from $\mathcal{B}(f)$ to $\mathcal{B}(f)$, then from $\mathcal{B}(f)$ to $\mathcal{B}(f)$) is itself a valid partial rooted isomorphism (and hence belong to our solution space $\Psi^0$) and that its associated cost consequently bounds by above $W_2^2(\mathcal{B}(f), \mathcal{B}(f))$ (Eq. 20). Second, we argue that this associated cost is itself bounded by above by $W_2^2(\mathcal{B}(f), \mathcal{B}(f)) + W_2^2(\mathcal{B}(f), \mathcal{B}(f))$.

Let $(\Psi^0, \Psi^0, \Psi^0)$ be the optimal solution of the partial assignment problem between $\mathcal{B}(f)$ and $\mathcal{B}(f)$, $\Psi^0$ is a rooted isomorphism (i.e. an isomorphism between rooted subtrees) between a subtree $B_i$ of $\mathcal{B}(f)$ and a subtree $B_j$ of $\mathcal{B}(f)$ (blue, Fig. 16). Equivalently, $\Psi^0$ can also be interpreted as a bijection between the arcs of $B_i$ and those of $B_j$.

Let $(\Psi^0, \Psi^0, \Psi^0)$ be the optimal solution of the partial assignment problem between $\mathcal{B}(f)$ and $\mathcal{B}(f)$, $\Psi^0$ is a rooted isomorphism between a subtree $B_i'$ of $\mathcal{B}(f)$ and a subtree $B_i'$ of $\mathcal{B}(f)$ (green, Fig. 16).

$$W_2^2(\mathcal{B}(f), \mathcal{B}(f)) \leq \left( \sum_{b \in B_i} \gamma(b \rightarrow \Psi^0(b))^2 (18) \right) + \sum_{b \in B_i} \gamma(b \rightarrow \emptyset)^2 + \sum_{b \in B_j} \gamma(\emptyset \rightarrow b)^2 (19)$$

Now, let $U$, $V$, $W$ be scalar functions on the nodes of the set $B_a = B_i \cup B_j \cup (B_i \setminus B_j) \cup (B_j \setminus B_i)$ (cyan subset, Fig. 16) such that:

$$U(b) = \begin{cases} \gamma(b \rightarrow \Psi^0(b)) & \text{for } b \in B_i' \setminus B_j' \\ \gamma(b \rightarrow \emptyset) & \text{for } b \in B_i \setminus B_i' \\ \gamma(\emptyset \rightarrow b) & \text{for } b \in B_j \setminus B_j' \\ \gamma(\emptyset \rightarrow \emptyset) & \text{for } b \in B_a \setminus B_i' \setminus B_j' \\ \psi(b) & \text{for } b \in B_i \setminus B_i' \\ \psi^{-1}(b) & \text{for } b \in B_j \setminus B_j' \\ \psi^{-1}(b) & \text{for } b \in B_a \setminus B_i' \setminus B_j' \end{cases} (21)$$

$$V(b) = \begin{cases} \gamma(b \rightarrow \Psi^0(b)) & \text{for } b \in B_i' \setminus B_j' \\ \gamma(b \rightarrow \emptyset) & \text{for } b \in B_i \setminus B_i' \\ \gamma(b \rightarrow \emptyset) & \text{for } b \in B_j \setminus B_j' \\ \gamma(\emptyset \rightarrow \emptyset) & \text{for } b \in B_a \setminus B_i' \setminus B_j' \end{cases} (23)$$

$U$ describes all the possible individual costs involved in the composition $\Psi^0 \circ \Psi^0$. In particular, we can re-write Eq. 20 as:

$$W_2^2(\mathcal{B}(f), \mathcal{B}(f)) \leq \|U\|_2 = \left( \sum_{b \in B_i} U(b)^2 \right)^{1/2} (22)$$
This can be verified by comparing the cost of each child task to determine which child task is the last one to complete, which significantly more constrained.

Similarly to \( V \), \( W \) describes a subset of the individual costs involved in the optimal rooted partial isomorphism \( \psi \). In particular, only the costs involving \( B_j \) (red square, Fig. 16, middle) are excluded. Thus, we have:

\[
W^2 (\psi(f_j), \psi(f_j)) \geq |V|_2^2 = (\sum_{b \in B_k} V(b))^2 / 2.
\]  

\[ (44) \]

Now, since \( y \) is defined by the Euclidean distance (Equation 9 of the main manuscript), we have for each node \( b \in B_k \):

\[
0 \leq U(b) \leq V(b) + W(b).
\]

This can be verified by comparing the \( i^{th} \) line of Eq. 21 to the sum of the \( j^{th} \) lines of Eq. 23 and Eq. 25. Then, we have:

\[
|U|_2 \leq |V + W|_2.
\]

Now, since the \( L^2 \) norm between vectors respects itself the triangle inequality, we have the following inequality:

\[
|V + W|_2 \leq |V|_2 + |W|_2.
\]

Then, by combining equations 22, 27, 28, 24, and 26, it follows that:

\[
W^2 (\psi(f_j), \psi(f_j)) \leq |U|_2^2 \leq |V + W|_2 \leq |V|_2^2 + |W|_2^2 \leq W^2 (\psi(f_j), \psi(f_j)) + W^2 (\psi(f_j), \psi(f_j))
\]

which concludes the proof.

### 3 COMPARISON TO THE EDIT DISTANCE ALGORITHM [121]

In addition to considering squared costs in our edit distance (equations of the section 3.3.3 of the main manuscript), our algorithm for the exploration of the search space indeed simplifies the approach by Zhang [121] (used by Sridharamurthy et al. [106]), as our search space is significantly more constrained.

First, since our solution space only considers partial isomorphisms between rooted subtrees, this implies that the destruction of a node (a branch) \( b \) \( \in \psi(f_j) \) necessarily implies the destruction of its subtrees, i.e. of its forest \( F_j(b) \). Thus, the admissible solutions in [106, 121] consisting in deleting \( f_j \) and mapping a subtree \( \psi(f_j, b) \) to one of the subtrees of \( f_j \) in the forest \( F_j(b) \) are no longer admissible given our overall solution space \( \Phi \). The removal of such solutions drastically simplifies the evaluation of the distance between subtrees (being the minimum of three solutions in [106], Eq. 12) to the Equation 11 of our main manuscript (containing only one expression to evaluate).

Second, our solution space (rooted partial isomorphisms) also implies that the nodes of \( \psi(f_j) \) can only be assigned to nodes with the same depth in \( \psi(f_j) \). This further implies that the distance between subtrees (Equation 11 of the main manuscript) only needs to be evaluated for subtrees rooted at nodes of identical depth (see Fig. 6 of the main manuscript).

Together, these two simplifications (i) simpler subtree distance and (ii) distance evaluation restricted to subtrees of identical depth from the root) are the key adaptations of Zhang’s algorithm [121] that are required for the exploration of our (more constrained) solution space.

### 4 PARALLEL COMPUTATION OF \( W^2 \)

In our work, we express the computation of \( W^2 \) in terms of tasks, to leverage task-based shared memory parallelism. First, the Equation 10 of the main manuscript is evaluated. For this, we initiate a task at each leaf \( f_j \) of \( \psi(f_j) \). If a task is the last one to compute among all the direct children of a node \( b \) \( \in \psi(f_j) \), it is then authorized to continue and estimate Equation 10 in \( b \). Atomic counters in \( b \) are implemented (and atomically incremented by the task of each child) to determine which child task is the last one to complete, which enables an efficient lightweight synchronization (Fig. 6 of the main manuscript). Overall, Equation 10 is completely estimated with this strategy in a bottom-up fashion. Second, Equation 11 (main manuscript) is evaluated similarly, by initiating a task at each leaf \( b \) of \( \psi(f_j) \). In particular, this task will evaluate Equation 11 given \( b \) against all subtrees of \( \psi(f_j) \) of identical depth (again using independent tasks initiated at the leaves of \( \psi(f_j) \), see Fig. 6). Similarly to Equation 10, we employ the same lightweight synchronization mechanism based on atomic counters to continue a task over to its parent only when it is the last child task reaching it. Thus, in both cases (Eqs. 10 and 11), the number of parallel tasks is initially bounded by the number of leaves in \( \psi(f_j) \) and \( \psi(f_j) \) (which is typically much larger than the number of cores) and progressively decreases during the computation.

### 5 \( \mathcal{D} = (f_j \rightarrow f_j) \) IS A GEODESIC FOR \( 0 \leq \alpha \leq 1 \)

We now argue that \( W^2 \) defines a geodesic space. For this, for any two BDTs \( \psi(f_j) \) and \( \psi(f_j) \), we describe the existence of a path between whose length is equal to \( W^2 (\psi(f_j), \psi(f_j)) \) (and thus minimal).

Let \( \mathcal{P} = \{ \psi(f_j) \} \) be a path of BDTs parameterized by \( t \).

We recall that the length \( L(P) \) of \( P \) is given by:

\[
L(P) = \sup_{u,v \in \mathcal{P}} \sum_{i=0}^{n} W^2 (\psi(u), \psi(v)).
\]

Now, let \( P_{0} \) be the path corresponding to the interpolation between \( \psi(f_j) \) and \( \psi(f_j) \), as defined in section 4.1 of the main manuscript. We now argue that \( L(P_{0}) \) is \( W^2 (\psi(f_j), \psi(f_j)) \).

Let \( (\psi', \mathcal{F}', \mathcal{R}') \) be the optimal rooted partial isomorphism between \( \psi(f_j) \) and \( \psi(f_j) \). Moreover, let \( \psi(f_j \rightarrow f_j) \) and \( \psi(f_j \rightarrow f_j) \) be two interpolated trees obtained respectively with \( \alpha = c \) and \( \alpha = t \), given \( 0 \leq c \leq t \leq 1 \). We will note \( \psi' \) the application on \( \mathcal{R} \cup \mathcal{F} \cup \mathcal{R}' \) defined by interpolation (section 4.1 of the main manuscript):

\[
\psi'(b) = \begin{cases}
(1-t)b + t\psi'(b) & \text{for } b \in B_t \\
(1-t)b + tb & \text{for } b \in B_c \\
xb + (1-s)h(x) & \text{for } b \in B_c.
\end{cases}
\]

\( \psi' \) is defined similarly for \( t \). Then, as discussed in Sec. 2 of this appendix, since the composition of partial rooted isomorphisms is itself a partial rooted isomorphism, the composition \( \psi \circ \psi' \) (which goes from \( \psi(f_j \rightarrow f_j) \) to \( \psi(f_j) \) and then from \( \psi(f_j) \) to \( \psi(f_j \rightarrow f_j) \)) does define a valid partial rooted isomorphism between \( \psi(f_j \rightarrow f_j) \) and \( \psi(f_j \rightarrow f_j) \) and we have:

\[
W^2 (\psi(f_j \rightarrow f_j), \psi(f_j \rightarrow f_j)) \leq \left( \sum_{b \in (\mathcal{F}, \mathcal{R}) \setminus (\mathcal{F}', \mathcal{R}')} \gamma(\psi'(b) \rightarrow \psi(b))^2 \right)^{1/2}
\]

and we also have by definition of \( \psi' \) and \( \psi' \) (Eq. 29):

\[
\left( \sum_{b \in (\mathcal{F}, \mathcal{R}) \setminus (\mathcal{F}', \mathcal{R}')} \gamma(\psi'(b) \rightarrow \psi'(b))^2 \right)^{1/2} = (t-s)W^2 (\psi(f_j), \psi(f_j)).
\]

Now, given the triangle inequality on the path \( P_{0} \), we have:

\[
W^2 (\psi(f_j), \psi(f_j)) \leq W^2 (\psi(f_j \rightarrow f_j), \psi(f_j \rightarrow f_j)) + W^2 (\psi(f_j \rightarrow f_j), \psi(f_j \rightarrow f_j)) + W^2 (\psi(f_j \rightarrow f_j), \psi(f_j \rightarrow f_j)) \leq (t-s) W^2 (\psi(f_j), \psi(f_j)).
\]

It follows that the above inequalities are in fact equalities and we have:

\[
W^2 (\psi(f_j \rightarrow f_j), \psi(f_j \rightarrow f_j)) = (t-s) W^2 (\psi(f_j), \psi(f_j)).
\]

Then, for any subdivision \( 0 = t_0 \leq t_1 \leq \cdots \leq t_n = 1 \) of \( P_{0} \), we have:

\[
\sum_{i=1}^{n} W^2 (\psi(f_j \rightarrow f_j), \psi(f_j \rightarrow f_j)) = W^2 (\psi(f_j), \psi(f_j)).
\]

Thus \( L(P_{0}) = W^2 (\psi(f_j), \psi(f_j)) \).

Hence the space of merge trees equipped with \( W^2 \) is a geodesic space, and \( \psi(f_j \rightarrow f_j) \) constructs paths of minimal length on it.
6 W_2^F WITH NORMALIZED COSTS DEFINES A GEODESIC SPACE

Let W_2^F (\mathcal{B}(f_i), \mathcal{B}(f'_i)) be a similarity measure between \mathcal{B}(f_i) and \mathcal{B}(f'_i), defined as:

W_2^F (\mathcal{B}(f_i), \mathcal{B}(f'_i)) = W_2^F \left( \mathcal{N}(\mathcal{B}(f_i)), \mathcal{N}(\mathcal{B}(f'_i)) \right)

where \mathcal{N} is the local normalization described in Section 4.2 of the main manuscript. Since \mathcal{N} is invertible, W_2^F inherits all the properties of W_2 and is also a distance metric. The normalized interpolation \mathcal{B}(f_i \rightarrow f'_j), x \in [0,1] between \mathcal{B}(f_i) and \mathcal{B}(f'_j) is defined as the image by \mathcal{N} of the interpolation between the normalized trees \mathcal{N}(\mathcal{B}(f_i)) and \mathcal{N}(\mathcal{B}(f'_j)). Then, given s \leq t, it follows that:

W_2^F (\mathcal{B}(f_s \rightarrow f_j), \mathcal{B}(f_s \rightarrow f'_j)) = (t-s)W_2^F (\mathcal{N}(\mathcal{B}(f_s)), \mathcal{N}(\mathcal{B}(f'_j)))

which proves that the space of merge trees equipped with W_2^F is a geodesic space, and that the above normalized interpolation constructs paths of minimal length on it.

7 MINIMIZING THE FRÉCHET ENERGY

The optimization algorithm described in Section 5.2 of the main manuscript constructively decreases the Fréchet energy at each iteration. In particular, once a local minimizer of the Fréchet energy is obtained for a fixed assignment with the update step (ii), the subsequent assignment step (i) does further improve the assignments hence iteratively decreasing the Fréchet energy constructively.

Let F be a function of an arbitrary BDT \mathcal{B} and of an arbitrary (i.e. not necessarily optimal) set of N rooted partial isomorphisms \{f_i \cup f_j, i, j \leq N\} between \mathcal{B} and the N BDTs of \mathcal{F}_\mathcal{B}:

F' (\mathcal{B}(f_i \cup f_j), \mathcal{B}(f'_i \cup f'_j)) := \sum_{\mathcal{A} \in \mathcal{F}_\mathcal{B}} \left( \sum_{b \in \mathcal{A}} \gamma(b_i \rightarrow f_b(b_j)) \right)^2

+ \sum_{b \in \mathcal{A}} \gamma(b_i \rightarrow 0)^2

+ \sum_{b \in \mathcal{A}} \gamma(0 \rightarrow b_j)^2)^{1/2}.

Now, let \mathcal{B}_k be the candidate barycenter at the iteration k of the algorithm and let \{f'_i \cup f'_j\} be the optimal rooted partial isomorphism between \mathcal{B}_k and \mathcal{B}(f_i \cup f_j), computed by the assignment step of the iteration. Then, we have:

F' (\mathcal{B}_k \cup f'_i \cup f'_j, \mathcal{B}(f_i \cup f_j)) \leq \sum_{\mathcal{A} \in \mathcal{F}_\mathcal{B}} W_2^F (\mathcal{B}_k \cup \mathcal{B}(f_i \cup f_j)).

Next, the update step of the iteration k consists in moving \mathcal{B}_k to \mathcal{B} by placing in the 2D birthdeath space each branch b \in \mathcal{B}_k at the arithmetic mean of the assignments. Since the arithmetic mean generally minimizes sums of Euclidean distances, we have:

F' (\mathcal{B}_k \cup f'_i \cup f'_j, \mathcal{B}(f_i \cup f_j)) \leq \sum_{\mathcal{A} \in \mathcal{F}_\mathcal{B}} W_2^F (\mathcal{B}_k \cup \mathcal{B}(f_i \cup f_j)).

Now, observe that since the previous rooted partial isomorphisms are not optimal anymore for \mathcal{B}_k, we also have:

\sum_{\mathcal{A} \in \mathcal{F}_\mathcal{B}} W_2^F (\mathcal{B}_k \cup \mathcal{B}(f_i \cup f_j)) \leq F' (\mathcal{B}_k \cup f'_i \cup f'_j, \mathcal{B}(f_i \cup f_j)).

Once \mathcal{B}_k is fixed, all the rooted partial isomorphisms are then optimized again with the assignment step of the iteration k+1 to attain:

F' (\mathcal{B}_k \cup f'_i \cup f'_j, \mathcal{B}(f_i \cup f_j)) \leq \sum_{\mathcal{A} \in \mathcal{F}_\mathcal{B}} W_2^F (\mathcal{B}_k \cup \mathcal{B}(f_i \cup f_j)).

The result of these two steps is that:

\sum_{\mathcal{A} \in \mathcal{F}_\mathcal{B}} W_2^F (\mathcal{B}_k \cup f'_i \cup f'_j, \mathcal{B}(f_i \cup f_j)) \leq \sum_{\mathcal{A} \in \mathcal{F}_\mathcal{B}} W_2^F (\mathcal{B}_k \cup f'_i \cup f'_j, \mathcal{B}(f_i \cup f_j))

Then, each iteration of our algorithm indeed decreases the Fréchet energy. Since there is a finite number of combinations of rooted partial isomorphisms between the barycenter and the N input trees \mathcal{B}(f_i), it follows that the algorithm converges, in a finite number of steps, to a local minimum of \mathcal{B} of the Fréchet energy (if multiple, equally valued, optimal sets of assignments exist between \mathcal{B} and \mathcal{F}_\mathcal{B}, each one needs to be explored with the update step of our algorithm). In practice, as described in the manuscript, we stop our algorithm when the Fréchet energy has decreased by less than 1% between consecutive iterations.

8 TEMPORAL REDUCTION ALGORITHM

Let \mathcal{F} = \{\mathcal{B}(f_1), \mathcal{B}(f_2), \ldots, \mathcal{B}(f_6)\} be the input temporal sequence of BDTs (we assume a regular temporal sampling). Let \mathcal{F} = \{f_i\} be a set of key frames. Let \mathcal{F}' = \{\mathcal{B}(f_i), \mathcal{B}(f'_i), \ldots, \mathcal{B}(f_{6-i})\} be a reduced temporal sequence, where:

\mathcal{B}(f_i) = (1 - \alpha_i)\mathcal{B}(f_i) + \alpha_i \mathcal{B}(f'_i)

where \mathcal{B}(f_i) and \mathcal{B}(f'_i) are two consecutive trees in \mathcal{F}, such that 1 \leq i \leq k and \alpha_i = (i - j)/(k - j). \mathcal{B}(f'_i) is then on a geodesic between \mathcal{B}(f_i) and \mathcal{B}(f_{6-i}). We introduce the following distance between the temporal sequences \mathcal{F} and \mathcal{F}':

d_k (\mathcal{F}, \mathcal{F}') = \left( \sum_{i = 1}^{k} W_2^F (\mathcal{B}(f_i), \mathcal{B}(f'_i)) \right)^{1/2}.

Our algorithm for temporal reduction consists in initializing \mathcal{F} = \{X\} and then removing greedily, at each iteration, the tree \mathcal{B}' from \mathcal{F} = \mathcal{F} \setminus \{X\} which minimizes \sum_{i = 1}^{k} W_2^F (\mathcal{B}(f_i), \mathcal{B}(f'_i)) and which, hence, better preserves the input sequence, until the size of \mathcal{F} reaches a target size.

Fig. 17 shows the temporal reduction performed by this algorithm on the Asternon impact sequence (see Section 9.1). This figure illustrates key frames, which correspond to time steps for which \mathcal{B}(f_i) = \mathcal{B}(f'_i); these are the time steps which have not been removed from the sequence through the reduction. In particular, this figure compares the usage of two metrics in the reduction algorithms: W_2^F (blue, top) and W_2^F (red, bottom, obtained with \epsilon = 1). By construction, since our reduction algorithm is based on interpolation only, the first and last BDTs in the sequence \mathcal{F} are always kept in the reduced sequence \mathcal{F}'. In other words, the first (leftmost, Fig. 17) and last (rightmost, Fig. 17) time steps are always identified as key frames, irrespective of the employed metric. For this specific example, the second key frame (second from left, Fig. 17) also happens to be identical for both metrics. In contrast to the sequence extremities, the common identification of this time step as a key frame by W_2^F and W_2^F is not obtained by construction: the reduction algorithm did not select this key frame in both configurations. Then, only the third key frame (third from left, Fig. 17) is different in this example. In particular, when using W_2^F, the reduction algorithm identifies one key frame per key phase of the simulation (see Section 9.1, each key phase is represented in Fig. 17 with a frame of distinct color). In contrast, when using W_2^F, the third key frame belongs to the same key phase as the last key frame ("Aftermath", light blue frame). Then the reduction driven by W_2^F fails at identifying a key frame for the third key phase ("Impact", dark blue frame). This is confirmed visually in Fig. 17, as the third key frame identified with W_2^F (in blue) seems to represent an intermediate step in the simulation between the second and fourth key frames. In contrast, the third key frame identified by the reduction with W_2^F (in red) is more visually similar to the fourth key frame, and hence possibly more redundant.

9 DATA SPECIFICATION

This section provides a complete specification of the ensemble datasets used in the paper. In particular, we document the data provenance, its representation, its pre-processing when applicable, and we specify the associated ground-truth classification. All of these ensemble datasets were extracted from public repositories. We additionally provide a set of scripts which automatically download all of these datasets (at the exception of Asternon impact and Cloud processes, for which the dataset providers need to be contacted personally), pre-process them with TTK and output them in VTK file format, with the ground-truth classification attached to the files as metadata (i.e. "Field Data" in the VTK terminology). For convenience, we also provide an archive containing the entire curated ensemble datasets (in VTK file format). All of this new material (scripts and curated data) is located at the following address: https://github.com/MaxPont/WassersteinMergeTreesData.

Moreover, we also provide in additional material all the ensembles of merge trees computed from these datasets (in the code archive containing the implementation of our method).
9.1 Asteroid impact

This ensemble is composed of 7 members, given as 3D regular grids (sampled at 300 × 300 × 300, implicitly triangulated by TTK). It has been made available in the context of the SciVis contest 2018 [84]. Each member corresponds to the last time step of the simulation of the impact of an asteroid with the sea at the surface of the Earth, for two configurations of asteroid diameter. The considered scalar field is the matter density, which is one of the variables of the simulation which discriminates well the asteroid from the water and the ambient air. This ensemble corresponds to a parameter study (in this case, studying the effect of the asteroid’s diameter on the resulting wave), which is a typical task in numerical simulation. In this application, salient maxima capture well the asteroid and large water splashes. Thus, each member is represented by the split tree (capturing maxima). The associated ground-truth classification assigns members computed with similar asteroid diameters to the same class. Thus, the corresponding classification task consists in identifying, for a given member, its correct asteroid diameter class. The ground-truth classification is as follows:

- Class 1 (3 members): yA11, yB11, yC11.

Another selection of the original data has been used for the evaluation of our temporal reduction framework (Fig. 10 of the main manuscript). For this experiment, we used the asteroid diameter configuration “yA11” and considered the following time steps, organized in 4 phases (according to the SciVis contest companion documentation [84]):

- **Phase 1, initial state** (5 time steps): 01141, 03429, 05700, 07920, 09782.
- **Phase 2, approach** (5 time steps): 13306, 16317, 18124, 19599, 21255.
- **Phase 3, impact** (5 time steps): 28649, 31737, 34654, 37273, 39476.
- **Phase 4, aftermath** (5 time steps): 44229, 45793, 47190, 48557, 49978.

9.2 Cloud processes

This ensemble is composed of 12 members, given as 2D regular grids (sampled at 1430 × 1557, implicitly triangulated by TTK). Each member corresponds to a time step of the simulation of cloud formations [116]. For this application, large clouds are well captured by the maxima of the pressure variable (pre-processed with 10 iterations of smoothing). Thus, split trees (capturing maxima) are considered for this ensemble. The associated ground-truth classification assigns each time step to one of the three key phases of the simulation. The corresponding classification task therefore consists in identifying, for each time step, to which phase it belongs. The ground-truth classification is as follows:

- **Class 1** (4 members): 0, 5, 10, 15.
- **Class 2** (4 members): 500, 505, 510, 515.
- **Class 3** (4 members): 1000, 1005, 1010, 1015.

9.3 Viscous fingering

This ensemble is composed of 15 members, given as 3-dimensional point clouds (representing a particle-based flow simulation). Each point cloud is turned into a Eulerian representation of the variables by using the “Gaussian Resampling” filter of ParaView, effectively transforming, via interpolation [101], each ensemble member into a 3D regular grid (sampled at 50 × 50 × 50, implicitly triangulated by TTK). The original data has been made available in the context of the SciVis contest 2016 [45]. Each member corresponds to the last time step of the simulation of a viscous fingering phenomenon, occurring when dissolving salt in water. The considered scalar field is the salt concentration, whose salient maxima capture well the most prominent fingers. Thus, each member is represented by the split tree (capturing maxima). Given the studied physical phenomenon, the simulation approach is not deterministic, resulting in distinct outputs for identical initial configurations. In this application, three distinct solver resolutions have been considered, corresponding to three distinct numbers of particles (resolution code 20: 194k particles, resolution code 30: 544k particles, resolution code 44: 1.7M particles). Thus, this ensemble corresponds to a parameter study (in this case, studying the effect of the input resolution on the output fingering), which is a typical task in numerical simulation. The associated ground-truth classification assigns members with the same input resolution to the same class. Thus, the corresponding classification task consists in identifying, for a given ensemble member, its corresponding particle count. The ground-truth classification is as follows:

- **Class 1, resolution 20** (5 members): 20run1, 20run3, 20run4, 20run5, 20run6.
- **Class 2, resolution 30** (5 members): 30run1, 30run2, 30run3, 30run4, 30run5.
- **Class 3, resolution 44** (5 members): 30run1, 30run2, 30run3, 30run4, 30run5.

9.4 Dark matter

This ensemble is composed of 40 members, given as 3-dimensional point clouds (representing a particle-based simulation). Each point cloud is turned into a Eulerian representation of the variables by using the “Gaussian Resampling” filter of ParaView, effectively transforming, via interpolation [101], each ensemble member into a 3D regular grid (sampled at 100 × 100 × 100, implicitly triangulated by TTK). The original data has been made available in the context of the SciVis contest 2015 [56]. Each member corresponds to a time step of a simulation of the universe formation, where regions of high concentration of dark matter form a filament structure known as the cosmic web. The considered scalar field is therefore dark matter density, whose salient maxima capture well large clusters of galaxies. Thus, each member is represented by the split tree (capturing maxima). The associated ground-truth classification assigns each time step to one of the four key phases of the simulation. The corresponding classification task therefore consists in identifying, for each time step, to which phase it belongs. The ground-truth classification is as follows:

- **Class 1** (10 members): 0.0020, 0.0300, 0.0400, 0.0500, 0.0600, 0.0700, 0.0800, 0.0900, 0.1000, 0.1100.
- **Class 2** (10 members): 0.2700, 0.2800, 0.2900, 0.3000, 0.3100, 0.3200, 0.3300, 0.3400, 0.3500, 0.3600.
- **Class 3** (10 members): 0.5900, 0.6000, 0.6100, 0.6200, 0.6300, 0.6400, 0.6500, 0.6600, 0.6700, 0.6800.
- **Class 4** (10 members): 0.9100, 0.9200, 0.9300, 0.9400, 0.9500, 0.9600, 0.9700, 0.9800, 0.9900, 1.0000.

9.5 Volcanic eruptions

This ensemble is composed of 12 members, given as 2D regular grids (sampled at 500 × 500, implicitly triangulated by TTK). Each member corresponds to an observation of a volcanic eruption, obtained by satellite imaging (as this data exhibits a bit of noise, it has been pre-simplified by removing all saddle-maxima pairs with a persistence lower than 0.5% of the data range). The original data has been made available in the context of the SciVis contest 2014 [41]. The considered scalar field is the sulfur dioxide concentration, for which salient maxima correspond to volcanic eruptions. Thus, each observation is represented by the split tree (capturing maxima). Each member corresponds to a specific acquisition period, itself corresponding to the eruption of one particular volcano at the surface of the Earth. The associated ground-truth classification assigns observations acquired in the same period of time to the same class. The corresponding classification task therefore consists in identifying, for each observation (taken at a specified date), the erupting volcano it corresponds to. The ground-truth classification is as follows:

- **Class 1** (4 members): 150_1m, 150_pm, 151_1m, 151_pm.
- **Class 2** (4 members): 156_1m, 156_pm, 157_1m, 157_pm.
- **Class 3** (4 members): 164_1m, 164_pm, 165_1m, 165_pm.

9.6 Ionization front (3D)

This ensemble is composed of 16 members, given as 3D regular grids (sampled at 300 × 124 × 124, implicitly triangulated by TTK). Each member corresponds to a time step of a simulation of ionization front propagation [109]. For this application, large ionization flares are well captured by salient maxima of the ion concentration. Thus, split trees (capturing maxima) are considered for this ensemble. The associated ground-truth classification assigns each time step to one of the four key phases of the simulation. The corresponding classification task therefore consists in identifying, for each time step, to which phase it belongs. The ground-truth classification is as follows:

- **Class 1** (4 members): 0.0025, 0.0026, 0.0027, 0.0028.
- **Class 2** (4 members): 0.0075, 0.0076, 0.0077, 0.0078.
- **Class 3** (4 members): 0.0125, 0.0126, 0.0127, 0.0128.
- **Class 4** (4 members): 0.0175, 0.0176, 0.0177, 0.0178.

9.7 Ionization front (2D)

This ensemble is a 2D version of the above ensemble, where the dataset providers have selected a 2D slice in the center of the volume (sampled at 600 × 248). The associated classification task is therefore identical.

9.8 Earthquake

This ensemble is composed of 12 members, given as 3D regular grids (sampled at 375 × 188 × 50, implicitly triangulated by TTK). Each member corresponds to a time step of the simulation of an earthquake at the San Andreas fault [77]. For this application, the shock wave can be tracked with the local maxima of the wave front velocity magnitude (this scalar field is pre-processed to pre-simplify all saddle-maxima pairs with a persistence smaller than 0.05% of the data range). Thus, split trees (capturing maxima) are considered for this ensemble. The associated ground-truth classification assigns each time step to one of the three key phases of the simulation. The corresponding classification task therefore consists in identifying, for each time step, to which phase it belongs. The ground-truth classification is as follows:
9.12 Vortex street

This ensemble is composed of 45 members, given as 2D regular grids (sampled at 300 × 100, implicitly triangulated by TTK). It has been generated with the Gerris flow solver [89] and was provided in previous work [39, 113]. It is available at the following address: https://github.com/julesvidal/wasserstein-pd-barycenter. The data models flow turbulence behind an obstacle. The considered scalar field is the orthogonal component of the curl of the flow velocity, for 5 fluids of different viscosity. This ensemble corresponds to a parameter study (in this case, studying the effect of viscosity on turbulence), which is a typical task in numerical simulation. In this application, salient extrema are typically considered as reliable estimations of the center of vortices. Thus, each run is represented by two merge trees (the join tree – capturing minima, and the split tree, capturing maxima), which are processed independently by our algorithms. The associated ground-truth classification assigns members computed with similar viscosities to the same class. The corresponding classification task therefore consists in identifying, for a given ensemble member, its correct wing configuration class. The ground-truth classification is as follows:

- Class 1 (9 members): Viscosity=100.0, Viscosity=100.1, Viscosity=100.2, Viscosity=100.3, Viscosity=100.4, Viscosity=100.5, Viscosity=100.6, Viscosity=100.7, Viscosity=100.9
- Class 4 (9 members): Viscosity=50.0, Viscosity=50.1, Viscosity=50.2, Viscosity=50.3, Viscosity=50.5, Viscosity=50.6, Viscosity=50.7, Viscosity=50.8, Viscosity=50.9
- Class 5 (9 members): Viscosity=60.0, Viscosity=60.2, Viscosity=60.3, Viscosity=60.4, Viscosity=60.5, Viscosity=60.6, Viscosity=60.7, Viscosity=60.8, Viscosity=60.9

10 PARAMETER ANALYSIS

In this section, we study the practical effect of the parameters of our approach. In particular, we extend the empirical stability evaluation of our metric with regard to all the parameters of our approach and we illustrate their effect on geodesic computation.

10.1 Interpretation

The first parameter of our approach is \( \varepsilon_1 \in [0, 1] \). It dictates the merge of saddles in the input trees, to mitigate saddle swap instabilities, as previously documented by Sridharamurthy et al. [106]. Adjacent saddles in the input trees are merged if their relative difference in scalar value (relative to the largest function difference between adjacent saddles) is smaller than \( \varepsilon_1 \). For \( \varepsilon_1 = 0 \), no saddle merge is performed whereas for \( \varepsilon_1 = 1 \), all saddles are merged and \( W^2 \) becomes equivalent to the \( L^2 \) Wasserstein distance between persistence diagrams, noted \( W^2 \).

The local normalization step of our framework (Section 4.2 of the main manuscript) guarantees the topological consistency of the interpolated merge trees (Fig. 8 of the main manuscript). However, this normalization shrinks the birth/death values of all the input branches to the interval \([0, 1]\), irrespective of their original persistence. To mitigate this effect, the input BDTs are pre-processed, so that branches with small initial persistence (i.e. small branches) are not given too much importance in the metric. In particular, small branches are moved up the input BDT if their persistence relative to their parent is larger than \( \varepsilon_1 \in [0, 1] \). When \( \varepsilon_2 = 0 \), all branches are moved up to the root of the BDT and again, \( W^2 \) becomes equivalent to \( W^2 \). When \( \varepsilon_2 = 1 \), no branch is moved up the BDT and \( \varepsilon_2 \) has no effect on the outcome (i.e. the input BDT is left unchanged). In practice, we recommend the default value \( \varepsilon_2 = 0.05 \); if a branch \( b \) has a nearly identical persistence to that of its parent \( b' \), it is moved higher in the BDT, so that its normalized persistence becomes nearly identical to that of its parent \( b' \) (instead of being artificially larger due to the local normalization).

The parameter \( \varepsilon_1 \in [0, 1] \) further restricts the application of the above BDT pre-processing, by only considering (for displacement up the BDT) the branches with a relative persistence (with respect to the overall data range) smaller than \( \varepsilon_1 \). When \( \varepsilon_1 = 1 \), all branches are subject to the above pre-processing and \( \varepsilon_2 \) fully dictates the BDT pre-processing. When \( \varepsilon_1 = 0 \), no branch is moved up the BDT and the two parameters \( \varepsilon_2 \) and \( \varepsilon_3 \) have no effect on the outcome. In practice, we recommend the default value \( \varepsilon_3 = 0.9 \), which prevents the most persistent branches from moving up the BDT.

Overall, when the parameters \((\varepsilon_1, \varepsilon_2, \varepsilon_3)\) are set to the values \((0, 1, 1)\), the input trees are not pre-processed by the above procedures (i.e. they are left unchanged) and their structure has a strong impact on \( W^2 \). When \( \varepsilon_1 = 0 \) and when \( \varepsilon_1 \) or \( \varepsilon_2 \) or \( \varepsilon_3 \) becomes equivalent to \( W^2 \) and the structure of the input trees has no impact anymore on the metric. In-between values balance the importance of the structure of the trees on the metric. We recommend the default values \((0.05, 0.95, 0.9)\), which provides an acceptable stability with regard to saddle swaps (mitigated by \( \varepsilon_1 \)) and which gives a reasonable importance to small branches in the metric (controlled by \( \varepsilon_2 \) and \( \varepsilon_3 \), which are dependent parameters).
10.2 Metric stability

Figure 14 of the main manuscript provides an empirical stability evaluation of our new metric $W^2_{\varepsilon}$, as a function of an input perturbation, modeled by a random noise of amplitude $\varepsilon$. In particular, this experiment is achieved for several values of $\varepsilon$. The conclusion of this experiment is that $W^2_{\varepsilon}$ is not stable when $\varepsilon = 0$ (sudden increase in $W^2_{\varepsilon}$ for small values of $\varepsilon$) and that it is stable when $\varepsilon = 1$ (as anticipated [112]). For in-between values, $W^2_{\varepsilon}$ is stable until a transition point (colored dots in Fig. 14 of the main manuscript), located at increasing noise levels ($\varepsilon$) for increasing values of $\varepsilon$. In particular, for the recommended default value $\varepsilon = 0.05$, $W^2_{\varepsilon}$ is stable up to a perturbation noise of amplitude 16% (of the overall data range).

In the following, we perform the same study for the other parameters of our approach, $\varepsilon_1$ and $\varepsilon_2$. Fig. 18 studies the practical stability of $W^2_{\varepsilon}$, for several values of $\varepsilon$. For this experiment, $\varepsilon_3$ has been set to 1 (then, only $\varepsilon$ has an impact on the BDT pre-processing described in the previous section). Moreover, $\varepsilon$ has been set to its recommended value, 0.05. Several curves are reported, one per $\varepsilon$ values. For $\varepsilon = 0$, all branches are moved up the BDT (irrespective of $\varepsilon$) and $W^2_{\varepsilon}$ becomes equivalent to $W^2_{\varepsilon}$ and the corresponding curve (red) exactly coincides with the light blue curve of the Figure 14 of the main manuscript ($\varepsilon_1 = 1$). For $\varepsilon = 1$, the input BDT is not pre-processed at all and the corresponding curve (pink) exactly coincides with the cyan curve of Figure 14 of the main manuscript (obtained for the default value $\varepsilon_1 = 0.05$). In-between values of $\varepsilon$ result in continuous transitions between these two extreme cases (blue, green and cyan curves).

Fig. 19 studies the practical stability of $W^2_{\varepsilon}$, for several values of $\varepsilon$. For this experiment, we set $\varepsilon_1 = 0$ and $\varepsilon_2 = 0$, to better isolate the effect of $\varepsilon$. When $\varepsilon_1 = 1$, all the branches of the input BDTs are subject to the BDT pre-processing. Since $\varepsilon_2 = 0$, all branches are moved up to the root and $W^2_{\varepsilon}$ becomes equivalent to $W^2_{\varepsilon}$ and the corresponding curve (red) exactly coincides with the light blue curve of the Figure 14 of the main manuscript ($\varepsilon_1 = 1$). When $\varepsilon_1 = 0$, no branch is moved up in the input BDTs and the corresponding curve (pink) exactly coincides with the grey curve of the Figure 14 of the main manuscript ($\varepsilon_1 = 0$).

In-between values of $\varepsilon_1$ result in transitions between these two extreme cases (blue, green and cyan curves), with transition points (similar to the Figure 14 of the main manuscript), before which $W^2_{\varepsilon}$ is stable. Note however, that since it is dependent on $\varepsilon_1$ (default value: 0.95), $\varepsilon_1$ has only a very mild practical impact on the metric.

10.3 Geodesic analysis

Figures 20, 21 and 22 respectively illustrate the effect of the parameters $\varepsilon_1$, $\varepsilon_2$ and $\varepsilon_3$ on the geodesics between merge trees. In particular, each figure shows, on the left, the geodesic obtained with a disabling value of the parameter (no effect on the computation). In contrast, the right side of each figure shows the geodesic obtained with the recommended default value of the parameter, to clearly visualize its impact.

Overall, as discussed in the detailed captions, these three parameters have the effect of moving branches up the input BDT, hence reducing the structural impact of the trees on the metric, but also improving its stability (as discussed in Section 10.2). In the data, moving a branch up the BDT corresponds to only slight modifications, which consist in reconnecting maxima to distinct saddles. For each parameter, the resulting pre-processing addresses cases where nearby saddles have very close function values, which impacts the stability of the metric. Similarly to Sridharamurthy et al. [106], we mitigate this effect with $\varepsilon_1$, but we also introduce $\varepsilon_2$ and $\varepsilon_3$ to specifically limit the importance in the metric of branches with persistence close to that of their parent.
Fig. 20. Impact of the parameter $\epsilon_1$ on geodesic computation (left: $\epsilon_1 = 0$, right: $\epsilon_1 = 0.05$). In this example (left), the white branch in $\mathcal{T}(f_i)$ is not matched to the white branch in $\mathcal{T}(f_j)$ as they have distinct depths in the corresponding BDTs (2 versus 1). However, these features are visually similar in the data (Gaussians with the white maximum in $f_i$ and $f_j$, bottom left corner of the domain). With $\epsilon_1 = 0.05$ (right), the saddle of the white branch in $\mathcal{T}(f_j)$ gets merged with its ancestor saddle (whose $f_j$ value was less than $\epsilon_1$ away). Consequently, the white branch gets moved up the BDT (the white branch is attached to the main light blue branch in $\mathcal{T}(f_j)$, right). Since they now have identical depths in the corresponding BDTs, the white branches of $\mathcal{T}(f_i)$ and $\mathcal{T}(f_j)$ can now be matched together (right), which results in an overall matching (and geodesic) between these two trees which better conveys the resemblance between the two scalar fields $f_i$ and $f_j$. Equivalently, one can interpret this procedure of saddle merge in the input trees as a modification of the input scalar field, turning $f_i$ into $f_j$. In particular, this field modification disconnects the Gaussian with the white maximum from the Gaussian with the dark blue maximum ($f_i$) and reconnects it to the Gaussian with the light blue maximum ($f_j$).

Fig. 21. Impact of the parameter $\epsilon_2$ on geodesic computation (left: $\epsilon_2 = 1$, right: $\epsilon_2 = 0.95$). In this example (left), the white branch in $\mathcal{T}(f_i)$ is not matched to the white branch in $\mathcal{T}(f_j)$ as they have distinct depths in the corresponding BDTs (2 versus 1). Moreover, given the function difference between the white branch’s saddle and its ancestor, that branch cannot be moved up the BDT under the effect of the $\epsilon_1$ procedure (above). The white branch in $\mathcal{T}(f_i)$ has a persistence nearly identical to its parent (cyan). Thus, after local normalization (necessary to guarantee the topological consistency of the interpolated trees), its normalized persistence would become artificially high, which can have an undesirable effect on the metric. The BDT pre-processing addresses this issue and moves up the BDT branches with a relative persistence to their parent larger than $\epsilon_2$ (recommended default value: 0.95). In this example (right), the white branch in $\mathcal{T}(f_i)$ moves up the BDT and becomes adjacent to the main light blue branch in $\mathcal{T}(f_j)$. Since they now have identical depths in the corresponding BDTs, the white branches of $\mathcal{T}(f_i)$ and $\mathcal{T}(f_j)$ can now be matched together (right), which better conveys the resemblance between the two scalar fields $f_i$ and $f_j$. Equivalently, one can interpret this procedure of BDT pre-processing as a modification of the input scalar field, turning $f_i$ into $f_j$. In particular, this field modification disconnects the Gaussian with the white maximum from the Gaussian with the cyan maximum ($f_i$) and reconnects it to the Gaussian with the light blue maximum ($f_j$).

Fig. 22. Effect of the parameter $\epsilon_3$ on geodesic computation (left: $\epsilon_3 = 0$, right: $\epsilon_3 = 0.9$). In this example (left), the white branch in $\mathcal{T}(f_i)$ is not matched to the white branch in $\mathcal{T}(f_j)$ as they have distinct depths in the corresponding BDTs (3 versus 1). Applying the above BDT pre-processing ($\epsilon_2$) to all branches would move the cyan branch in $\mathcal{T}(f_j)$ up the BDT, which would prevent it to match to the cyan branch in $\mathcal{T}(f_j)$. The parameter $\epsilon_3$ restricts the application of the above BDT pre-processing and prevents the movement of the most persistent branches (relative persistence larger than $\epsilon_3$, default: 0.9). In this example (right), the white branch in $\mathcal{T}(f_i)$ moves up the BDT and becomes adjacent to the main light blue branch in $\mathcal{T}(f_j)$. Since they now have identical depths in the corresponding BDTs, the white branches of $\mathcal{T}(f_i)$ and $\mathcal{T}(f_j)$ can now be matched together (right), which results in an overall matching (and geodesic) between these two trees which better conveys the resemblance between the two scalar fields $f_i$ and $f_j$. Equivalently, one can interpret this procedure on the BDTs as a modification of the input scalar field, turning $f_i$ into $f_j$. In particular, this field modification disconnects the Gaussian with the white maximum from the Gaussian with the dark green maximum ($f_i$) and reconnects it to the Gaussian with the light blue maximum ($f_j$).