Detection and skeletonization of single neurons and tracer injections using topological methods

Dingkang Wang[1], Lucas Magee[1], Bing-Xing Huo[2], Samik Banerjee[2], Xu Li[2], Jaikishan Jayakumar[3], Meng Kuan Lin[2], Keerthi Ram[3], Suyi Wang[1], Yusu Wang[1], Partha P. Mitra[2]

Neuroscientific data analysis has traditionally relied on linear algebra and stochastic process theory. However, the tree-like shapes of neurons cannot be described easily as points in a vector space (the subtraction of two neuronal shapes is not a meaningful operation), and methods from computational topology are better suited to their analysis. Here we introduce methods from Discrete Morse (DM) Theory to extract the tree-skeletons of individual neurons from volumetric brain image data, and to summarize collections of neurons labelled by tracer injections. Since individual neurons are topologically trees, it is sensible to summarize the collection of neurons using a consensus tree-shape that provides a richer information summary than the traditional regional 'connectivity matrix' approach. The conceptually elegant DM approach lacks hand-tuned parameters and captures global properties of the data as opposed to previous approaches which are inherently local. For individual skeletonization of sparsely labelled neurons we obtain substantial performance gains over state-of-the-art non-topological methods (over 10% improvements in precision and faster proofreading). The consensus-tree summary of tracer injections incorporates the regional connectivity matrix information, but in addition captures the collective collateral branching patterns of the set of neurons connected to the injection site, and provides a bridge between single-neuron morphology and tracer-injection data.

Summary

Neuroscientific data analysis has traditionally involved methods for statistical signal and image processing, drawing on linear algebra and stochastic process theory. However, digitized neuroanatomical datasets containing labelled neurons, either individually or in groups labelled by tracer injections, do not fit into this classical framework. The tree-like shapes of neurons cannot be adequately described as points in a vector space (e.g. the subtraction of two neuronal shapes is not a meaningful operation). There is therefore a need for new approaches, which has become more urgent given the growth in whole-brain datasets with sparsely labelled neurons or tracer injections.

Methods from computational topology and geometry are naturally suited to the analysis of neuronal shapes. In this paper we introduce methods from Discrete Morse Theory to extract tree-skeletons of individual neurons from volumetric brain image data, and to summarize collections of neurons labelled by anterograde tracer injections. Since individual neurons are topologically trees, it is sensible to summarize the collection of neurons using a consensus tree-shape. This consensus tree provides a richer information summary than the regional or voxel-based "connectivity matrix" approach that has previously been used in the literature.

The algorithmic procedure for single neuron skeletonization includes an initial neurite-detection step to extract a density field from the raw volumetric image data, followed by Discrete Morse theoretic skeleton extraction from the density field using the 1-(un)stable manifold of the density field. We apply the method to high-resolution 3D volumes of sparsely-labelled neurons to quantitatively extract the single neuron topology, and achieve better performance than state-of-the-art algorithms using non-topological methods (over 10% improvement in precision, and significant speedups at the proofreading stage).

In image volumes of brains with neuronal tracer injections, a summarized *consensus tree* of the collective neuronal projection patterns is extracted to characterize brain-wide neuronal connectivity. The procedure for building the tree skeleton assigns the weight associated with detected tracer projections locally to the nearest point on the skeleton. This ensures that the resulting weighted skeleton also provides a summary of the tracer projection data: integrating the weighted skeleton over a brain region, helps recover the regional "connectivity matrix". Thus the consensus-tree summary of tracer injections incorporates the traditional regional connectivity matrix information, but in addition captures the collective *collateral branching patterns* of the set of neurons connected to the injection site. We propose that this summary can provide a future means of characterizing tracer injection data, and also provide a bridge to a growing body of single-neuron morphology data.

We find that the DM method is able to trace a tree branch through regions of low intensity that pose challenges to baseline methods. This is a particular strength of the DM approach as it utilizes the global topological structure present in the data, whereas relevant literature methods are inherently spatially local. Additionally the DM approach is theoretically well principled and conceptually clean without multiple ad-hoc hand-engineered steps. There is a significant computational overhead to the topological approaches but we are able to mitigate the speed issues using parallelized implementations of the core algorithms.

---

[1] *Computer Science and Engineering Department, The Ohio State University, Columbus, OH, USA 43210.*

[2] *Cold Spring Harbor Laboratory, NY, USA 11724.*

[3] *Indian Institute of Technology, Chennai, Tamil Nadu, India 600036.*

Background

Topological Data Analysis and Discrete Morse Theory: Within the last decade or so, topological data analysis (TDA) has grown in influence. Several new TDA methodologies have been proposed for analyzing complex high-dimensional datasets. These approaches use a variety of topological concepts to characterize essential structure behind the data and algorithmic implementations have been developed [1–5]. Some of the ideas (in particular persistent homology [6]) have been applied to multiple application domains [7–11], including neuroscience [12–14].

The area of TDA that relates to the present work is the persistence-guided Discrete Morse theory based framework for reconstructing hidden graphs from observed data. Discrete Morse theory has been utilized to capture hidden structure in 2D or 3D volumetric data [15–17]. Extraction of hidden graphs was formulated in [18], and the framework was simplified and theoretical guarantees provided in [19]. Morse theory exploits the global (instead of local) topology of a scalar field. Thus the hidden graph skeleton of the scalar field can be extracted reliably through regions of weak signals. Classical Morse theory applies to continuous functions on manifolds. Discrete Morse theory [20] provides a discretized computational framework suitable for algorithmic implementation on digitized data. Persistent homology is used to separate signal from noise and remove potentially noise-related structure from the graph. Such a framework has been applied previously to reconstructing hidden road networks from noisy GPS trajectories and satellite images [21,22]. Here we adapt and extend this approach to develop a computational methodology suitable for computational neuroanatomy and to address neuroscientific problems.

In this manuscript we introduce a data analysis framework entitled DM-skeleton that uses TDA and the Discrete Morse approach to skeletonize individual neurons as well as groups of neurons labelled by tracer injections. For single neuron reconstruction, we show that DM-skeleton significantly outperforms the best available automated approaches[23,24] (over 10% improvements in F1 scores) and reduces proofreading times. For tracer injection skeletonization, DM-skeleton provides a conceptually new route to the analysis of mesoscale projection data, and shows robust performance.

Single Neuron Reconstruction: Many methods have been previously proposed for single-neuron reconstruction from high resolution image stacks [23–50] (see also surveys and books [51–54] for more detailed discussions.) Most of the existing methods sequentially expand a tree from a collection of seed points that are often selected based on density information. A popular line of sequential tracing algorithms use shortest-path based approaches, such as APP2 [23] and SmartTracing[49]. More recently, methods using a principal-curve to sequentially include more nodes in the reconstruction (e.g., NeuroGPS-Tree [50] and GTree [24]) have shown high performance on single neuron reconstruction in mouse brain datasets. Our proposed DM-skeleton approach shows robust performance gains over APP2 and GTree, and shortens proofreading time. Importantly, it also opens up a conceptually new direction in single neuron tracing by using global topological information in a manner that we feel will generalize beyond this specific work.

Tracer Injection Skeletonization: Brain-wide tracer injection datasets are another area where we apply the DM skeletonization approach. In tracer injection data sets, thousands of neurons are collectively labelled and do not have the topological simplicity of individual trees. Conventionally, brain-wide connectivity information is summarized in the form of regional connectivity matrices [55]. Such a representation ignores the fundamental tree-like morphology of neurons and contains no information relating to collateral branching patterns. In this manuscript we introduce a new approach to the analysis of tracer injection data, by *skeletonizing* the tracer injections using a *summary tree*. To the best of our knowledge, this is a new approach to conceptualizing tracer-injection data, and could provide a biologically better-grounded approach to the study of mesoscale connectivity mapping using tracer injections. One important advantage of the tree-skeletonization approach is that it connects more directly with single-neuron reconstructions.

It is possible to utilize methods such as APP2 or GTree to perform tracer-injection skeletonization (and we use these as baseline comparison techniques). However, these methods do not automatically interpolate through regions of weak label, a problem that is exacerbated for tracer-injection data. In contrast, the DM based approach utilizes global information and is more robust.
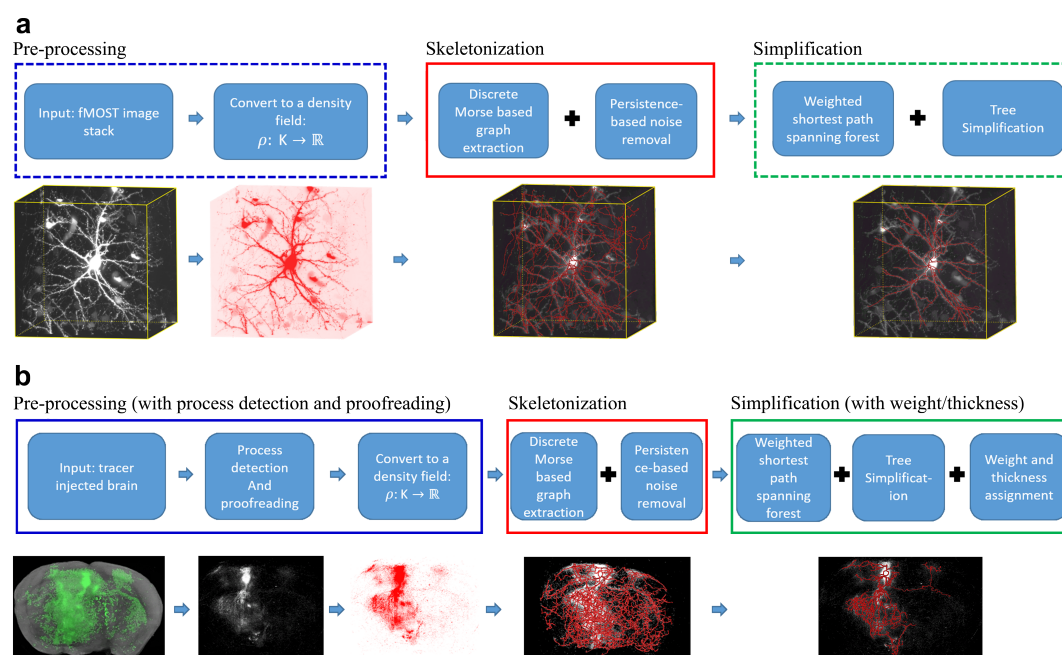
The DM pipeline for summarizing multiple-neuron tracer injection datasets has multiple stages. First, the raw image stack is preprocessed to detect or highlight neuronal processes [56]. Then a variant of the Discrete-Morse module [19,21] is employed to produce a graph skeleton containing all potential neuronal trajectories. This graph skeleton may contain false positives, so the next stage of our algorithm performs a persistent homology based simplification step which removes superfluous branches in low-density regions and branches that misalign with estimated "flow vectors". This simplified graph is skeletonized into a minimum spanning tree, which is further simplified using a persistence threshold. During all simplification steps, weights assigned to points on the tree (summarizing the neighboring projection density) is re-assigned so as to preserve the total weight. The resulting weighted tree-summary provides a new way of characterizing tracer injection data, simultaneously capturing regional connectivity information as well as collateral branching patterns.

Results

Summary method overview The high-level workflow of our new Discrete Morse-based (DM-Skeleton) pipeline is shown in Fig. 1. The detailed flows for high-resolution single neuron data (e.g, fluorescent micro-optical sectioning tomograph (fMOST)) and for whole-brain 3D densely labelled tracer injection data (e.g, serial two-photon tomography (STP)) are slightly different, as the types of input and goals are different. Nevertheless, both workflows have three main components, namely preprocessing, skeletonization, and simplification. (For details, see Method section.)

DM-Skeleton takes 3D image stacks as input. In Step 1, the input image is converted to a density field defined on a 3D grid $\rho : K \to \mathbb{R}$. For fMOST data, raw image intensity was treated as the density value subjected to Morse skeletonization. For the STP data, a process-detection step [56] was first applied to segment labelled axon fragments in the high-resolution 2D images. A 3D volume was created to summarize the density of axon fragments

Fig. 1 | Workflow and sample outputs for the DM-Skeleton pipeline. **a,** Workflow for fMOST datasets (from sparse neuronal label to individual skeleton extraction) and sample outputs for each step. **b,** Workflow for tracer injection datasets (from dense neuronal label to injection summarization) and sample outputs for each step.

within each voxel at lower resolution. This summary volume was the density field subjected to Morse analysis. Given the starting density field ρ, the goal of the DM-skeleton algorithm is to capture center-lines passing through (relatively) high density regions. By viewing this density map as a terrain (see Fig. 2 for a 2D example), this step corresponds to extracting the 'mountain ridges' of this terrain. This extraction was achieved using 1-stable manifolds from discrete Morse theory.
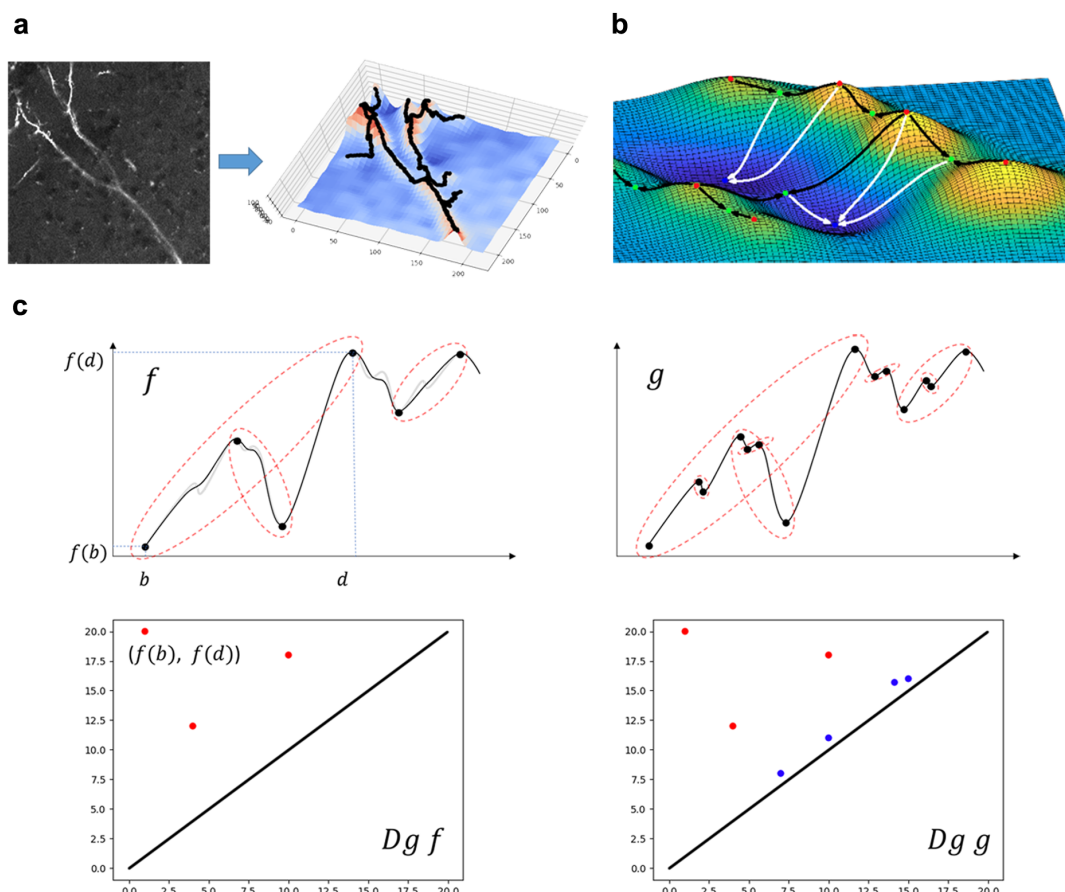
In Step 2, a variant of the persistence-guided discrete Morse-based framework of [19,21] was applied to the 3D density field ρ. We call the output the Morse 'graph skeleton' G. Note that this approach takes into account the global topology of the density field: the 1-stable manifold connects through low-density regions (e.g., gaps and weak signals around the Y-junction in Fig. 2a) reliably. Finally, in Step 3, we extracted either a forest (e.g, each tree representing a single neuron in fMOST data) or a tree summary from the Morse graph skeleton G. The forest output may have false positives, thus we developed a simplification module to further remove such branches (see Fig. 3; more details in Method).

fMOST sparse labeling dataset. Three semi-manually reconstructed neurons from fMOST dataset were taken as ground truth. A $224 \times 224 \times 251 \mu m$ neighborhood around the soma was taken as the test dataset. More details about the test dataset and the ground truth are provided in Methods. The Morse graph skeleton obtained from the persistence-guided discrete Morse-based framework (Step 2) was fed to the simplification module (Step 3). In Step 3, the positions of somata were taken as given and used as roots to extract a forest such that one tree is rooted at each soma. Note that there exist many automatic methods for reliably detecting soma positions [57,58]. Automatic methods can be applied to reduce human labor when a region has many somata. The tree rooted at the soma of the target neuron was then simpli-
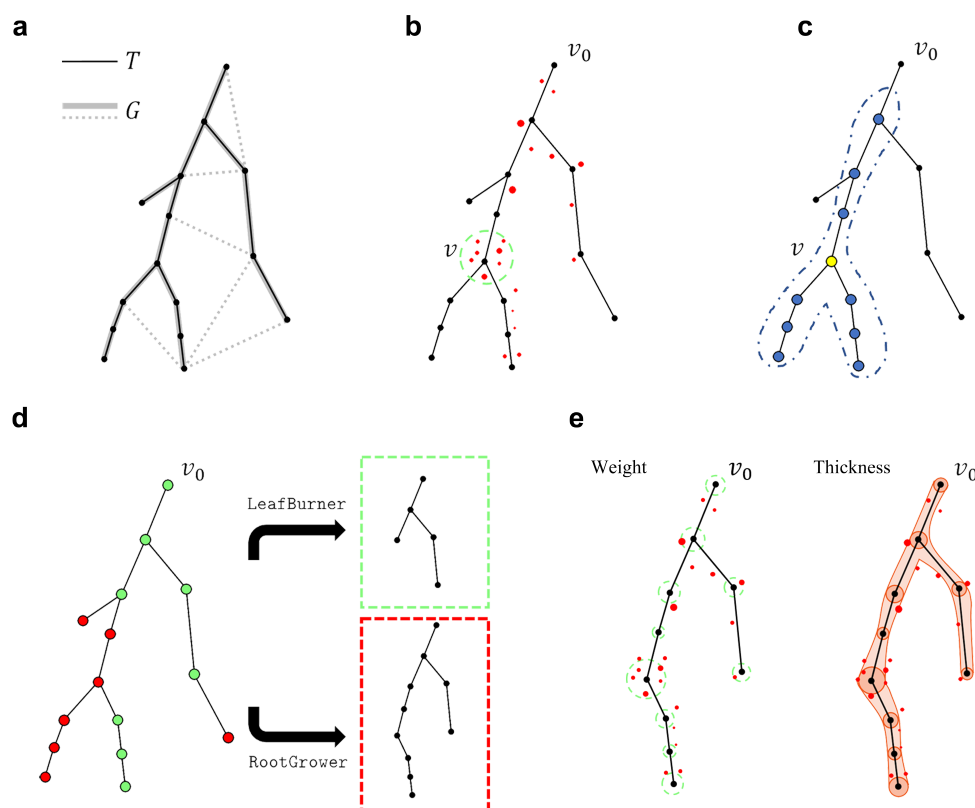
fied. In this step, each tree node was assigned a score based on density (see Supplementary Methods). Next, the `RootGrower` strategy was applied (see Method section) to grow a simplified neuron tree. This strategy (instead of `LeafBurner`) is more suitable for the fMOST dataset because there is observable autofluorescence scattered in the extracted regions. A bottom-up process (`LeafBurner`) would stall in such regions when the autofluorescence has high-intensity values similar to the somata.

We evaluated the performance of the DM-Skeleton pipeline on three fMOST single-neuron regions. We compared our output with the state-of-the-art approaches GTree [24] and APP2 [23] methods. For GTree and APP2, we tested different parameters and used the outputs having the fewest gross errors for comparison. We compared the precision, recall and F1-score of the outputs of these three methods over ground-truth reconstructions (Fig. 4c). In all three regions, we note that both our DM-Skeleton and the GTree method performed better than the APP2 method. Furthermore, on average, our DM-Skeleton method outperformed GTree by around 14% in precision and 3% in recall. This suggests that our DM-Skeleton has significantly fewer false positives than the output of GTree.

To further illustrate the potential use of our output, we proofread the outputs from GTree and DM-Skeleton using 3D virtual finger tool [59] on fMOST regions. Proofreading was performed on both the GTree outputs and the DM-Skeleton outputs by two independent annotators without domain-specific knowledge and with no prior knowledge of the ground truth. The annotators also constructed the skeleton from scratch on the region of neuron1 using the same proofreading tool with an average time spent of roughly one hour. Supplementary Table 1 shows that the automatic methods reduce the human time of annotation by at least two thirds. Precision, recall, and F1-score were calculated to compare proofread results with the ground truth
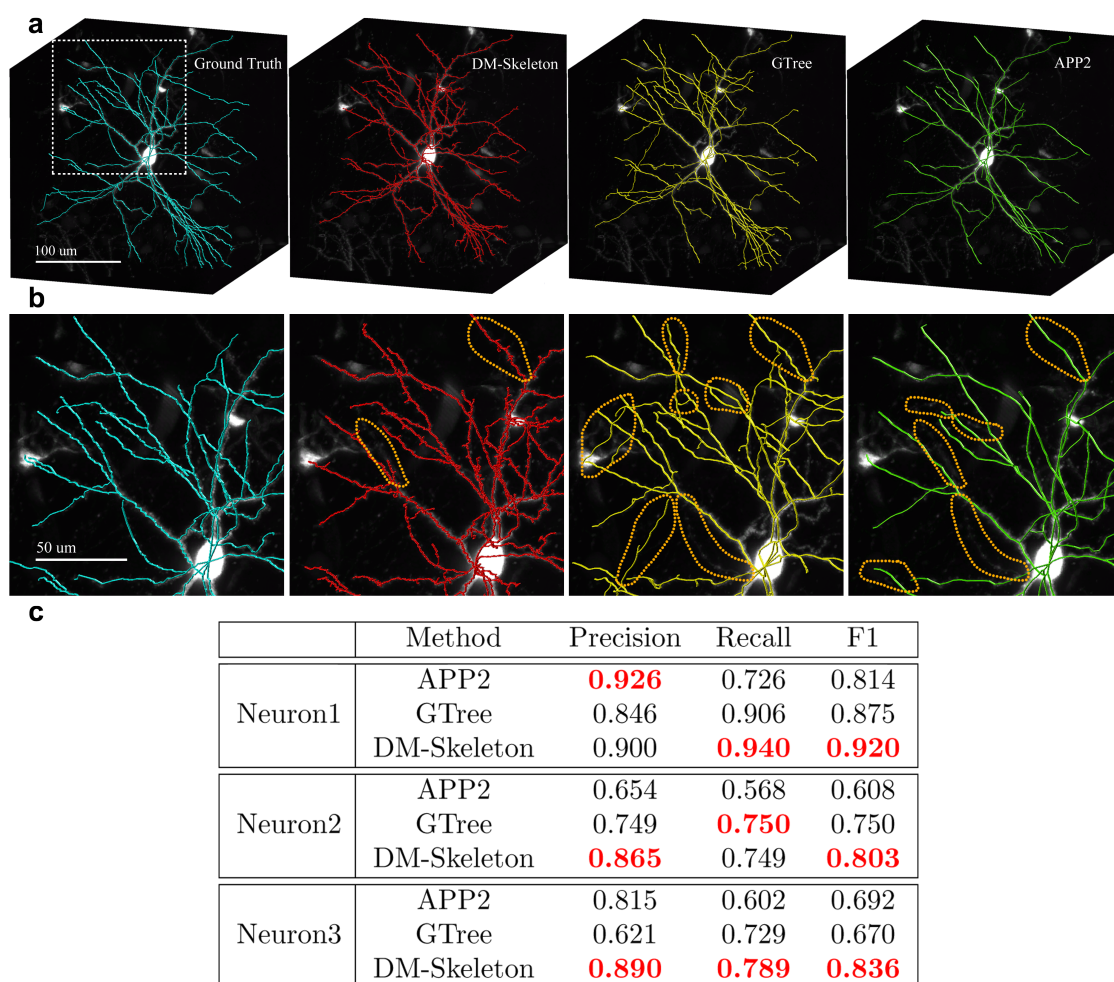
Fig. 2 | Illustration and basic concepts for Discrete Morse theory. **a,** An example of a 2D image (left) converted to a density function with the graph (terrain) of this function (right). The weak y-shape connection is captured by the mountain ridges (shown in black) on this terrain. **b,** An example of 2D terrain (graph of a 2D function): red points are local maxima, green points are saddles, while blue points are local minima. The white paths are some examples of the integral paths following the local gradients, ending in minima. The black curves are a collection of 1-stable manifolds (integral paths between maxima and saddles). **c,** An example of persistence pairs on 1D functions. For a function $f : \mathbb{R} \to \mathbb{R}$, topological features (connected components in this simple case) are 'born' at local minima, and 'die' at local maxima. Each persistence pair (b, d) indicates the birth and death of some feature, born at $f(b)$ and killed at $f(d)$. This gives rise to a point $(f(b), f(d))$ in the so-called persistence diagram $Dg f$ w.r.t. f (lower pictures). The persistence of this feature is the difference in function values $|f(d) - f(b)|$ which can be considered as a measure of importance as it gives how "long" the feature persists. In the figures, persistence pairings are marked by red dotted curves. The function g (shown on the right) can be viewed as a noisy perturbation of function f (shown on the left). The function f has 3 prominent features (persistence pairs), while the perturbed version also has additional "smaller" features with lower persistence (importance). As shown in the persistence diagrams (lower row): In addition to the three red persistence points, diagram $Dg g$ also has blue points close to the diagonal, indicating features whose persistence (i.e., deathtime−birthtime) is small. These small features ("noise") can be detected using their persistence values and removed in our algorithm.

Fig. 3 | Summarization and simplification algorithms. a, An example of spanning forest algorithm. Only one of the trees is shown for simplicity. Assume G is the graph output from Discrete Morse, and T is its weighted shortest path spanning tree. The edges are weighted by the average intensity of both endpoints - larger weights mean smaller distances. Dotted lines represent those edges that are removed from G. **b,** Score assignment for tree nodes. Each voxel is associated with its nearest neighbor in the set of tree nodes. The score of a tree node v is the sum of the density of voxels associated with v. An upper bound β_s is set to avoid distant associations. **c,** Score smoothing: To reduce noise, we smooth the scores of each tree node v by averaging those of its neighbors within k-hops (i.e., connected via at most k edges to v). We restrict only to neighbors that are ancestors or descendants of v. An example with $k = 3$ is shown in the figure, where blue nodes are neighbors of node v. **d,** Simplification process: We utilize two strategies. Strategy 1 (LeafBurner) starts from the leaves and iteratively removes tree nodes with scores $< \tau$. Strategy 2 (RootGrower) grows the tree from the root v_0 by keeping tree nodes with scores $> \tau$. In the tree on the left, green nodes (red nodes) correspond to nodes with score higher (lower) than the threshold τ. Deploying different strategies will result in different outputs. **e,** To achieve a weight-preserving summarization, each tree node has a weight (represented by the size of the dotted green circle) equal to the sum of the weights of associated voxels. We also assign a thickness value to each tree node (represented by the radius of the orange circle) for better visualization. This thickness value is proportional to the square root of the number of associated voxels.

	Method	Precision	Recall	F1
	APP2	**0.926**	0.726	0.814
Neuron1	GTree	0.846	0.906	0.875
	DM-Skeleton	0.900	**0.940**	**0.920**
	APP2	0.654	0.568	0.608
Neuron2	GTree	0.749	**0.750**	0.750
	DM-Skeleton	**0.865**	0.749	**0.803**
	APP2	0.815	0.602	0.692
Neuron3	GTree	0.621	0.729	0.670
	DM-Skeleton	**0.890**	**0.789**	**0.836**

Fig. 4 | Results of fMOST single-neuron skeletonization. For the purpose of visualization, all thickness values were suppressed. **a,** 3D volumes of ground-truth (cyan), DM-Skeleton method (red), GTree (yellow) and APP2 (green) outputs. All the methods are further compared in the zoomed-in region (the region in the white square shown on the ground truth) in (**b**). **b,** Ground-truth (cyan on the left), and comparisons of zoomed-in regions of DM-Skeleton method and GTree method. APP2 output (green) clearly misses many branches and is considered having the worst performance. False positives and false negatives are highlighted in orange circles. GTree method has many false positives and APP method misses many branches. Our DM-Skeleton method has a visible advantage over GTree and APP2 in these examples. **c,** Evaluation on fMOST dataset, best scores achieved are highlighted in red. DM-Skeleton used persistence threshold = 256 and simplification threshold = 0.2 for all the cases. The APP2 method does not have a good performance in terms of F1-score. On average, when compared with the GTree method, DM-Skeleton has around 14% advantage on precision and slightly better recall; it achieves the best F1-scores ($\sim 5\%$ to $\sim 16\%$ better) in all the regions.

(Supplementary Table 1). All of these metrics demonstrate that proofread results even by naive annotators were consistent and of high-quality. Importantly, the time spent proofreading the DM-Skeleton outputs was much shorter than the time spent proofreading GTree outputs. The reason is that DM-Skeleton outputs only have minor early-termination issues, which can be corrected by extending corresponding traces. By contrast, GTree outputs usually have a considerable amount of false positives. They also occasionally miss important mainstream branches, which causes difficulties and prolongs the proofreading process.

To utilize our framework, one labeled dataset was used to tune the parameters (for persistence simplification and tree simplification). These parameters were then applied to other datasets with similar contrast and SNR. In the results reported above, Neuron-1 from the fMOST dataset was used to tune the parameters. We chose multiple persistence and tree simplification thresholds. For each such pair of choices, we generated the final tree and computed its F1-score compared with ground-truth tree (Supplementary Table 2). We then selected the thresholds that give the best F1-score and apply the same thresholds to all other fMOST neurons. From Supplementary Table 2, we note that results were reasonably stable to perturbations of thresholds.

STP tracer injection dataset. When applying the DM-Skeleton pipeline on the tracer injection dataset, in Step 1, the original image data (see Section Data Collection) was passed through a hybrid deep CNN with topological priors[56] to detect axon fragments. The output of this automated detection process was then manually proofread and corrected. The proofread images were summarized into lower resolution to construct a manageable 3D volume of the entire brain. The remaining process is similar to the single neuron skeletonization algorithm above. However, there are some notable differences. On one hand, the pre-processed STP data was clean and did not have as much noise as the autofluorescence and bright background in fMOST data, making the `LeafBurner` simplification strategy preferred over the strategy `RootGrower` for STP dataset (see Fig. 3, and also Method section). On the other hand, the tracer injection data has more discontinuities and gaps. Therefore, a Gaussian filter (kernel radius 2) was applied to the image stack (after preprocessing) before further processing. The persistence threshold was adjusted to account for bit-depth differences. The simplification threshold for the tracer data was taken to be the same as for the single neuron data set even though we changed the strategy for simplification. This parameter was robust to input image types because an average score of all tree nodes was used as the reference (Supplementary Methods).

The Morse graph skeleton obtained in Step 2 may have some redundant straight segments connecting signals to the boundary of the domain (Supplementary Fig. 2). This boundary effect is caused by the zero-value background and degenerated gradient on those pixels in the cleaned STP data. Such redundant segments can be easily removed based on distance to the nearest non-zero voxel.

We can in principle further streamline the output using flow vectors estimated with weighted principal component analysis[60]. Paths not well aligned with estimated flow vectors can be removed as long as doing so does not increase the number of connected components that make up the output. In the results shown below, we did not apply this flow vector simplification, but results with or without this flow vector simplification step are compared in Supplementary Fig. 4. The output with flow vector simplification is visually cleaner and still has good coverage rates.
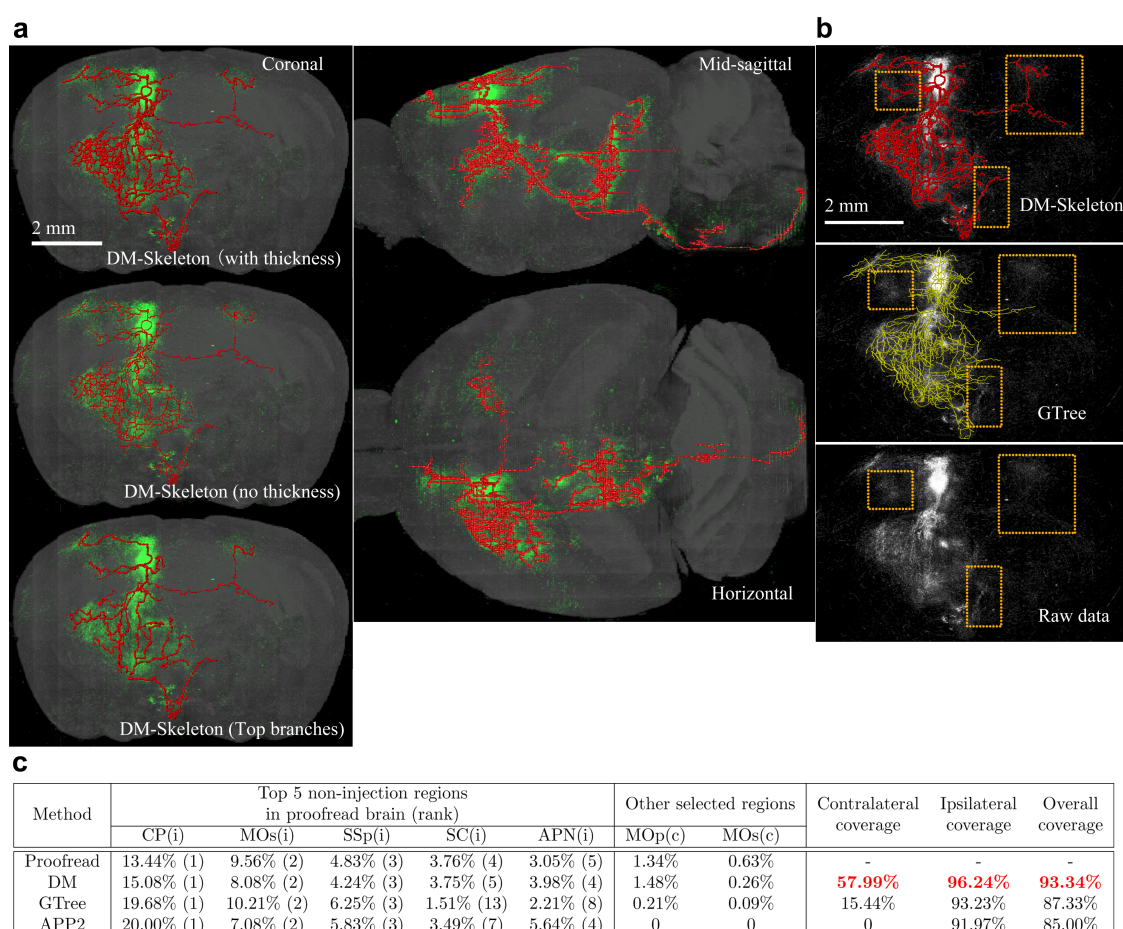
In Step 3, the root position was manually selected within the projection site. Then, a shortest path spanning tree was extracted and simplified. In contrast with the single-neuron skeletonization, the tracer injection skeleton has a *weight* assigned to each tree node after simplification. The weight of a given tree node v represents the total weight of voxels whose nearest neighbor in the tree node set is v (see Fig. 3, and also Methods section). This is a weight-preserving process where the weights of the tree nodes of the output preserve the total tracer density of points assigned to that node.

In addition, for visualization purposes we assigned a thickness value for each tree node. The thickness value uses the number of voxels instead of their weights to avoid excessively large tree nodes around the injection site (see Fig. 5a and Methods section). Keeping only top branches (based on length) can provide a more concise summary or skeleton. See Fig. 5a.

Visually, the output by APP2 method is over-simplified and misses many obvious signals (Supplementary Fig. 3). GTree and DM-Skeleton method have more plausible results (Fig. 5b). In Fig. 5b, we further compared GTree and DM-Skeleton outputs in several regions. From the raw data, we can see that these regions, corresponding to motor cortex and striatum in the contralateral hemisphere, contain visible signals. GTree, however, does not have enough branches covering these regions. This may be because GTree is strongly dependent on brightness and contrast, and is not robust enough to handle regions of images with relatively low brightness. This necessitates fine-tuning of the pixel-value histogram prior to running GTree. In contrast, the Discrete Morse-based method is able to overcome these issues because only the relative order of pixel-values matters and effectively there is a built-in histogram localization. Thus the method is automatically adaptive to different types of input images, as well as to different local pixel intensity distributions, with minimal image pre-processing requirements. We also note that even the top few branches from our DM-Skeleton output (the image at the left bottom corner in Fig. 5a) still capture key branches and present a good coverage of different brain regions (including those within orange boxes shown in Fig. 5b, which are missed in the outputs by GTree and APP2 methods).

To verify the biological interpretability of the skeletonization results, we first mapped the 3D volume of the simplified tree to the Allen mouse brain atlas[61] for regional projection strength analysis (Supplementary Methods). The intensity of projected pixels reflected the weights preserved by tree nodes. The projected segments were categorized into the corresponding brain regions based on the atlas map. We applied the same procedure to the proofread volume. A good summarization should cover most of the signals in the input data. Thus, the regions covered by the skeleton-based summary and the conventional connectivity matrix based summary should coincide. Moreover, the distributions of weights over different regions should be similar. In Fig. 5c, we computed the percentages of (projected) weights within certain regions over the total weights for the proofread data and all three summarizations. Note that the output from GTree is merely a skeletonization without thickness.

We selected the top 5 regions with the highest weights based on the conventional connectivity matrix summary and found the ranks of these regions in the summarization outputs. DM-Skeleton output has the identical top 5 regions as those calculated

Method	Top 5 non-injection regions in proofread brain (rank)					Other selected regions		Contralateral coverage	Ipsilateral coverage	Overall coverage
	CP(i)	MOs(i)	SSp(i)	SC(i)	APN(i)	MOp(c)	MOs(c)			
Proofread	13.44% (1)	9.56% (2)	4.83% (3)	3.76% (4)	3.05% (5)	1.34%	0.63%	-	-	-
DM	15.08% (1)	8.08% (2)	4.24% (3)	3.75% (5)	3.98% (4)	1.48%	0.26%	57.99%	96.24%	93.34%
GTree	19.68% (1)	10.21% (2)	6.25% (3)	1.51% (13)	2.21% (8)	0.21%	0.09%	15.44%	93.23%	87.33%
APP2	20.00% (1)	7.08% (2)	5.83% (3)	3.49% (7)	5.64% (4)	0	0	0	91.97%	85.00%

Fig. 5 | Skeletonization result on STP data with tracer injection. **a,** On the left, we show the results in a coronal plane. The result with thickness assigned (top) have tree nodes with greater thickness in regions with higher process density. In the output before thickness assignment (middle), the thickness of every tree node is set to a constant. Top 20 branches are also shown (bottom). Selecting top branches based on total length can provide a concise summarization. The DM-Skeleton result after thickness assignment are presented from other angles on the right. The images and summaries were rescaled along the rostral-caudal direction for isotropic visualization. **b,** The APP2 outputs (shown in Supplementary Fig. 3) do not have comparable quality to the other two, so the visual comparison here is between GTree and our method. The raw data (bottom) has several challenging regions marked by orange rectangles. DM-Skeleton method successfully captures these regions, while the GTree method misses these regions. In addition, the GTree method has many redundant edges in the high density region, which causes difficulties in observing the main skeleton. **c,** We computed the weight distribution over regions for all summarization methods and the proofread brain with an injection in the primary motor cortex (MOp). The same brain regions in either hemisphere were considered separately. The 5 projection regions with top weights are analyzed for the summarization methods. The injection site is on the ipsilateral hemisphere. The DM-Skeleton method recovers the correct top 5 regions, while the GTree method has relatively low ranks for SC and APN regions. In other regions, e.g., the MOp and MOs in the contralateral hemisphere, the GTree method does not capture the signals very well and the APP2 method completely misses those regions. Our DM-Skeleton has the most accurate summary for all cases. The abbreviated region names are Caudoputamen (CP), Secondary motor area (MOs), Primary somatosensory area (SSp), Superior colliculus (SC) and Anterior pretectal nucleus (APN). The suffixes (i) or (c) correspond to the ipsilateral or contralateral hemisphere. DM-Skeleton is abbreviated as DM in the table for better presentation.

from the proofread data, while GTree or APP2 have difficulties in capturing the signals in SC and APN. The DM-Skeleton output captured the projection into the contralateral areas, while the GTree output did not cover many of these regions. The APP2 output completely missed the contralateral hemisphere. There are other regions where our weighted tree qualitatively matches the conventional summary, while GTree and APP2 have weight percentages close to zero. The coverages for each hemisphere as well as the entire brain were calculated individually. Our DM-Skeleton had the best coverage rates in all cases. In the contralateral hemisphere, it outperformed the GTree method by more than 40%. There is a small discrepancy in the weight distribution over regions between the conventional process density summary and the DM-Skeleton based summary. The reason is that DM-Skeleton did not capture some poorly labelled regions, and the voxels may be assigned to a tree node in a neighboring region.

In addition, we compared our DM-Skeleton results with single neuron data from the Mouselight project [62]. Eleven neurons which have soma positions in the injection site of the STP dataset were selected and quantitatively summarized. Visually, the DM-Skeleton output showed good correspondence with this collection of neurons, which were expected to cover similar regions as the STP volume (Supplementary Fig. 5).

Discussion

In this paper we introduced a new conceptual framework of data analysis utilizing Discrete Morse Theory for extracting underlying tree structures and applied it to two neuroscience problems of current importance: the automatic extraction of single neuron skeletons, and biologically meaningful analysis of mesoscale connectivity mapping data. Apart from being conceptually and mathematically well grounded, this approach significantly outperforms state-of-the-art approaches relevant for these problems.

Current approaches to analyzing brain-wide tracer-injection data all start from regional connectivity matrices, which lose the biologically important tree-like structure of neurons composing those projections. Apart from respecting the collateral branching patterns of the underlying set of neurons and providing a natural bridge to single neuron data, the DM-Skeleton method utilizes the global structure of the signal and is robust to noise and missing data, and naturally adaptive to signal intensity variations. We demonstrated the approach using a tracer-injected whole-brain data set. The resulting tree-skeleton faithfully captured the regional "connectivity matrix" information, and significantly outperformed baseline methods. In addition it captured the collateral branching patterns, consistent with corresponding single neuron skeletons with somata in the injection region. We expect that this approach will be useful in the future for mesoscale brain connectivity mapping using tracer injections and form a bridge to the single neuron data.

We also demonstrated significance performance improvements for single neuron reconstruction over existing best in class methods. Automated single neuron skeletonization is a well-studied problem with a large associated literature and hundreds of algorithms. Our methodology breaks new ground by taking a different conceptual approach that leads to a simple and theoretically transparent algorithmic framework simultaneously with performance improvements. As in the case of tracer-injection

skeletonization, the method takes into account the global structure of the data, is robust to noise and missing data, and is data-adaptive as construction of the Morse skeleton depends on the order relations between neighboring pixel intensities rather than their absolute values.

While our method achieves strong performance, it is computationally expensive compared with other methods. The computational bottleneck comes from the persistence-guided discrete Morse-based framework. Computing persistence pairings takes $O(n^3)$ running time in the worst case (although usually significantly faster in practice), where n is the number of cells that make up the input cell complex. To address this bottleneck, we utilized DIPHA [63] to compute persistence pairings. This algorithm is distributed and allows for a significant speedup compared to non-distributed persistence algorithms. Further code optimizations over our current implementation are possible and run-time can be further reduced in future work. Despite the higher computational complexity, the conceptual elegance and theoretical transparency, performance improvements including significant reduction in human proof-reading times, and incorporation of prior biological structure are strong arguments in favor of the approaches proposed here.

References

1. Edelsbrunner, H. & Harer, J. *Computational Topology : an Introduction* (American Mathematical Society, 2010).

2. Carlsson, G. Topology and data. *Bull. Amer. Math. Soc.* **46**, 255–308 (2009).

3. Chazal, F. & Michel, B. An introduction to Topological Data Analysis: fundamental and practical aspects for data scientists. *CORR* (2017).

4. Lum, P. Y. *et al.* Extracting insights from the shape of complex data using topology. *Scientific Reports* **3** (2013).

5. Tierny, J. *Topological Data Analysis for Scientific Visualization* (Springer, 2018).

6. Edelsbrunner, Letscher & Zomorodian. Topological persistence and simplification. *Discrete & Computational Geometry* **28**, 511–533 (2002).

7. Buchet, M., Hiraoka, Y. & Obayashi, I. *Persistent Homology and Materials Informatics*, 75–95 (Springer Singapore, Singapore, 2018).

8. Singh, G. *et al.* Topological analysis of population activity in visual cortex. *Journal of vision* **8**, 11 (2008).

9. Platt, D. E., Basu, S., Zalloua, P. A. & Parida, L. Characterizing redescriptions using persistent homology to isolate genetic pathways contributing to pathogenesis. *BMC Systems Biology* **10**, S10 (2016).

10. Lamar-León, J., García-Reyes, E. B. & Gonzalez-Diaz, R. Human gait identification using persistent homology. In Alvarez, L., Mejail, M., Gomez, L. & Jacobo, J. (eds.) *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, 244–251 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2012).

11. Lee, Y. *et al.* Quantifying similarity of pore-geometry in nanoporous materials. *Nature Communication* 15396 (2017).

12. Li, Y., Wang, D., Ascoli, G. A., Mitra, P. & Wang, Y. Metrics for comparing neuronal tree shapes based on persistent homology. *PloS one* **12** (2017).

13. Chaudhuri, R., Gerçek, B., Pandey, B., Peyrache, A. & Fiete, I. The intrinsic attractor manifold and population dynamics of a canonical cognitive circuit across waking and sleep. *Nature Neuroscience* **22**, 1512–1520 (2019).

14. Kanari, L. *et al.* A topological representation of branching neuronal morphologies. *Neuroinformatics* **16**, 3–13 (2018).

15. Delgado-Friedrichs, O., Robins, V. & Sheppard, A. Skeletonization and partitioning of digital images using discrete morse theory. *IEEE Trans. Pattern Anal. Machine Intelligence* **37**, 654–666 (2015).

16. Gyulassy, A. *et al.* Topologically clean distance fields. *IEEE Trans. Visualization Computer Graphics* **13**, 1432–1439 (2007).

17. Robins, V., Wood, P. J. & Sheppard, A. P. Theory and algorithms for constructing discrete morse complexes from grayscale digital images. *IEEE Trans. Pattern Anal. Machine Intelligence* **33**, 1646–1658 (2011).

18. Sousbie, T. The persistent cosmic web and its filamentary structure – I. Theory and implementation. *Monthly Notices of the Royal Astronomical Society* **414**, 350–383 (2011).

19. Dey, T. K., Wang, J. & Wang, Y. Graph reconstruction by discrete morse theory. In *Proc. 34th Intl. Sympos. Comput. Geom. (SoCG)*, 31:1–31:15 (2018).

20. Forman, R. Morse theory for cell complexes. *Advances in Mathematics* **134**, 90 – 145 (1998).

21. Wang, S., Wang, Y. & Li, Y. Efficient map reconstruction and augmentation via topological methods. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '15, 25:1–25:10 (ACM, New York, NY, USA, 2015).

22. Dey, T. K., Wang, J. & Wang, Y. Road network reconstruction from satellite images with machine learning supported by topological methods. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM SIGSPATIAL GIS)* (2019). To appear.

23. Xiao, H. & Peng, H. App2: automatic tracing of 3d neuron morphology based on hierarchical pruning of a gray-weighted image distance-tree. *Bioinformatics* **29**, 1448 (2013).

24. Hang, Z. *et al.* Dense reconstruction of brain-wide neuronal population close to the ground truth. *bioRxiv* (2018).

25. Al-Kofahi, K. A. *et al.* Rapid automated three-dimensional tracing of neurons from confocal image stacks. *Trans. Info. Tech. Biomed.* **6**, 171–187 (2002).

26. Bas, E. & Erdogmus, D. Principal curves as skeletons of tubular objects. *Neuroinformatics* **9**, 181–191 (2011).

27. Basu, S., Ooi, W. T. & Racoceanu, D. Improved marked point process priors for single neurite tracing. In *2014 International Workshop on Pattern Recognition in Neuroimaging*, 1–4 (2014).

28. Boykov, Y., Veksler, O. & Zabih, R. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **23**, 1222–1239 (2001).

29. Choromanska, A., Chang, S.-F. & Yuste, R. Automatic reconstruction of neural morphologies with multi-scale tracking. *Frontiers in Neural Circuits* **6**, 25 (2012).

30. Chothani, P., Mehta, V. & Stepanyants, A. Automated tracing of neurites from light microscopy stacks of images. *Neuroinformatics* **9**, 263–278 (2011).

31. Gala, R., Chapeton, J., Jitesh, J., Bhavsar, C. & Stepanyants, A. Active learning of neuron morphology for accurate automated tracing of neurites. *Frontiers in Neuroanatomy* **8**, 37 (2014).

32. Lee, P.-C., Ching, Y.-T., Chang, H. M. & Chiang, A.-S. A semi-automatic method for neuron centerline extraction in confocal microscopic image stack. In *2008 5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, 959–962 (2008).

33. Lee, P.-C., Chuang, C.-C., Chiang, A.-S. & Ching, Y.-T. High-throughput computer method for 3d neuronal structure reconstruction from the image stack of the drosophila brain and its applications. *PLOS Computational Biology* **8**, 1–12 (2012).

34. Myatt, D., Hadlington, T., Ascoli, G. & Nasuto, S. Neuromantic – from semi-manual to semi-automatic reconstruction of neuron morphology. *Frontiers in Neuroinformatics* **6**, 4 (2012).

35. Oh, S. W. *et al.* A mesoscale connectome of the mouse brain. *Nature* **508**, 207–214 (2014).

36. Sironi, A., Lepetit, V. & Fua, P. Projection onto the manifold of elongated structures for accurate extraction. In *2015 IEEE International Conference on Computer Vision (ICCV)*, 316–324 (2015).

37. Schmitt, S., Evers, J. F., Duch, C., Scholz, M. & Obermayer, K. New methods for the computer-assisted 3-d reconstruction of neurons from confocal image stacks. *NeuroImage* **23**, 1283 – 1298 (2004).

38. Sui, D., Wang, K., Chae, J., Zhang, Y. & Zhang, H. A pipeline for neuron reconstruction based on spatial sliding volume filter seeding. *Computational and Mathematical Methods in Medicine* **2014**, 1–8 (2014).

39. Srinivasan, R. *et al.* Automated axon tracking of 3d confocal laser scanning microscopy images using guided probabilistic region merging. *Neuroinformatics* **5**, 189–203 (2007).

40. Türetken, E., Benmansour, F., Andres, B., Pfister, H. & Fua, P. Reconstructing loopy curvilinear structures using integer programming. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 1822–1829 (2013).

41. Türetken, E., González, G., Blum, C. & Fua, P. Automated reconstruction of dendritic and axonal trees by global optimization with geometric priors. *Neuroinformatics* **9**, 279–302 (2011).

42. Vasilkoski, Z. & Stepanyants, A. Detection of the optimal neuron traces in confocal microscopy images. *Journal of Neuroscience Methods* **178**, 197 – 204 (2009).

43. Yang, J., Gonzalez-Bellido, P. T. & Peng, H. A distance-field based automatic neuron tracing method. *BMC Bioinformatics* **14**, 93 (2013).

44. Zhang, Y. *et al.* A novel tracing algorithm for high throughput imaging: Screening of neuron-based assays. *Journal of Neuroscience Methods* **160**, 149 – 162 (2007).

45. Zhao, T. *et al.* Automated reconstruction of neuronal morphology based on local geometrical and global structural models. *Neuroinformatics* **9**, 247–261 (2011).

46. Zhou, Y. & Toga, A. W. Efficient skeletonization of volumetric objects. *IEEE Transactions on Visualization and Computer Graphics* **5**, 196–209 (1999).

47. Zhou, Z., Sorensen, S. A. & Peng, H. Neuron crawler: An automatic tracing algorithm for very large neuron images. In *2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI)*, 870–874 (2015).

48. Zhou, Z., Liu, X., Long, B. & Peng, H. Tremap: Automatic 3d neuron reconstruction based on tracing, reverse mapping and assembling of 2d projections. *Neuroinformatics* **14**, 41–50 (2016).

49. Chen, H., Xiao, H., Liu, T. & Peng, H. Smarttracing: self-learning-based neuron reconstruction. *Brain Informatics* **2**, 135–144 (2015).

50. Quan, T. *et al.* Neurogps-tree: automatic reconstruction of large-scale neuronal populations with dense neurites. *Nature Methods* **13**, 51–54 (2016).

51. Acciai, L., Soda, P. & Iannello, G. Automated neuron tracing methods: An updated account. *Neuroinformatics* **14**, 353–367 (2016).

52. Donohue, D. E. & Ascoli, G. A. Automated reconstruction of neuronal morphology: An overview. *Brain Research Reviews* **67**, 94 – 102 (2011).

53. Meijering, E. Neuron tracing in perspective. *Cytometry Part A* **77A**, 693–704 (2010).

54. Arenkiel, B. *Neural Tracing Methods: Tracing Neurons and Their Connections*. Neuromethods (Springer New York, 2014).

55. Chapter 3 - connectivity matrices and brain graphs. In Fornito, A., Zalesky, A. & Bullmore, E. T. (eds.) *Fundamentals of Brain Network Analysis*, 89 – 113 (Academic Press, San Diego, 2016).

56. Banerjee, S. *et al.* Semantic segmentation of microscopic neuroanatomical data by combining topological priors with encoder-decoder deep networks. *bioRxiv* (2020).

57. Yan, C. *et al.* Automated and accurate detection of soma location and surface morphology in large-scale 3d neuron images. *PLOS ONE* **8**, 1–12 (2013).

58. Cheng, S., Quan, T., Liu, X. & Zeng, S. Large-scale localization of touching somas from 3d images using density-peak clustering. *BMC Bioinformatics* **17**, 375 (2016).

59. Peng, H. *et al.* Virtual finger boosts three-dimensional imaging and microsurgery as well as terabyte volume image visualization and analysis. *Nature Communications* **5**, 4342 (2014).

60. Delchambre, L. Weighted principal component analysis: a weighted covariance eigendecomposition approach. *Monthly Notices of the Royal Astronomical Society* **446**, 3545–3555 (2014).

61. Allen mouse brain atlas. http://download.alleninstitute.org/informatics-archive/current-release/mouse_ccf/.

62. Mouselight neuronbrowser. https://www.janelia.org/project-team/mouselight/neuronbrowser.

63. Bauer, U., Kerber, M. & Reininghaus, J. *Distributed Computation of Persistent Homology*, 31–38 (2014).

64. Matho, K. *et al.* Genetic dissection of glutamatergic neuron subpopulations and developmental trajectories in the cerebral cortex. In prep.

65. Josh Huang, Z. & Zeng, H. Genetic approaches to neural circuits in the mouse. *Annual Review of Neuroscience* **36**, 183–215 (2013).

66. Ragan, T. *et al.* Serial two-photon tomography for automated ex vivo mouse brain imaging. *Nature Methods* **9**, 255–258 (2012).

67. Gong, H. *et al.* High-throughput dual-colour precision imaging for brain-wide connectome with cytoarchitectonic landmarks at the cellular level. *Nature Communications* **7**, 12142 (2016).

68. Wang, Y. *et al.* Teravr empowers precise reconstruction of complete 3-d neuronal morphology in the whole brain. *Nature Communications* **10**, 3474 (2019).

69. Dey, T. K., Wang, J. & Wang, Y. Improved road network reconstruction using discrete morse theory. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (ACM SIGSPATIAL GIS)*, 58–66 (2017).

70. Schneider, C. A., Rasband, W. S. & Eliceiri, K. W. Nih image to imagej: 25 years of image analysis. *Nature methods* **9**, 671–675 (2012).

Data and code availability

The fMOST and STP data were collected as a part of Brain Initiative Cell Census Network and shared online. The raw images of the STP dataset are available from: `ftp://download.brainimagelibrary.org:8811/biccn/huang/connectivity/anterograde/180830_JH_WG_Fezf2LSLflp_CFA_female_processed/`. The raw images of the fMOST dataset are available from: `ftp://download.brainimagelibrary.org:8811/biccn/zeng/luo/fMOST/732664811/`. Single neuron reconstruction ground truth are available from `ftp://download.brainimagelibrary.org:8811/biccn/zeng/luo/fMOST/cells/732664811/`. Processed projection summary of STP data, subregions of fMOST data, the code and documentation are available on Github at `https://github.com/wangdingkang/DiscreteMorse`.

Acknowledgements

Author contributions

D. W., L. M., S. W., Y. W. and P. M designed the pipeline. D.W. and L.M. implemented the pipeline. B. H., S. B., J. J., K. R., X. L. and M. L. provided the test datasets. D. W. and L. M. collected results on all test datasets, and together with Y. W. and P. M. analyzed the results on fMOST dataset. D. W., B. H., X. L., Y. W. and P. M. analyzed the results on STP dataset. D. W., L. M., B. H., S. B., J. J., X. L., Y. W. and P. M. wrote the manuscript.

Competing interests

The authors declare no competing interests.

Methods

Data Collection. The Serial Two-Photon (STP) dataset presented in this paper was collected as a part of Brain Initiative Cell Census Network [64]. Specific Cre-dependent transgenic mouse lines were crossed with IslFlp reporter lines. Flp-dependent AAV tracers were utilized to reveal cell type-specific axon connection [65]. Each brain was prepared and imaged using STP tomography [66] with $1\mu m \times 1\mu m$ in-plane resolution, and sectioned coronally every 50 μm. Two channels of 16-bit data were collected, where Channel 1 collected the autofluorescence and Channel 2 collected the fluorescent tracer information. Only Channel 2 data were used in the subsequent analysis. One STP dataset was involved in the development and demonstration of methods in this paper (available from: `ftp://download.brainimagelibrary.org:8811/biccn/huang/connectivity/anterograde/180830_JH_WG_Fezf2LSLflp_CFA_female_processed/`).

The dual-color fluorescent micro-optical sectioning tomography (fMOST) data presented in this paper , both raw images and single neuron reconstruction data ("ground truth"), were collected as a part of Brain Initiative Cell Census Network and downloaded from Brain Image Library (available from `http://www.brainimagelibrary.org/`). Specific Cre-dependent transgenic mouse lines with reporter transgenic mouse lines. Sparse labeling of the entire neuron was achieved by injecting small amount of viral tracer. The brain was fixed and embedded in resin before mounted for fMOST imaging at $0.32\mu m \times 0.32\mu m$ in-plane resolution and sectioned serially at $1\mu m$. Two channels of data were collected, with 16-bit data in each channel [67]. Only green channel contained the neuron tracing information and therefore used in the current study. One fMOST dataset was involved in the development and demonstration of methods in this paper (available from: `ftp://download.brainimagelibrary.org:8811/biccn/zeng/luo/fMOST/732664811/`). Single neuron reconstruction was performed using a combination of automatic tracing and manual annotation [68]. Information of nodes and edges of individual neuron was stored in separate files in swc format (available from: `ftp://download.brainimagelibrary.org:8811/biccn/zeng/luo/fMOST/cells/732664811/`).

Data pre-processing. For fMOST data, neighborhoods of 3 reconstructed neurons were taken as the test datasets. Each neighborhood volume was saved as one VTK file (simple legacy format). The associated single neuron reconstruction data, saved in SWC format, were taken as ground truth for the subsequent analysis. The VTK and SWC files are available on `https://github.com/wangdingkang/DiscreteMorse`.

The STP dataset was first processed with a topologically motivated convolutional neural network [56] for the detection of the tracers. The network, termed as DM++, takes in whole STP sections and divides them into 512×512 pixel tiles. These tiles are passed through a topological algorithm based on Discrete Morse [69] and a CNN counterpart for determining the topological and neuronal priors, respectively. The topological priors capture the faint connectivity which is used to boost the performance of the CNN in a supervised Siamese setting for the dual priors incorporated into the DM++ framework. The final output likelihood map is converted into a binary mask for the neuronal processes using an optimal empirically determined threshold. This captures most of the processes in the tiles, which are then stitched back together to form a mask for an entire reference section of the brain.

The preliminary outputs of process detection were manually verified for the entire brain by an experienced neuroanatomist using Fiji [70]. Briefly, the preliminary outputs consisting of detected signal were masked with the original brain section image and error corrected using the pixel painting tool. The filled processes were identified as those having a brighter intensity compared to the background.

The proofread brain from the previous step was annotated in the format of binary images. The images were further downsampled to the desired resolution using the sum method, which preserved all the signal information. Volumes of neuronal axons were summarized by counting the number of pixels ($1\mu m \times 1\mu m$ resolution) containing fluorescent signal into $10\mu m \times 10\mu m$ in coronal sections while keeping the inter-section spacing the same ($50\mu m$). The resulting images are 8-bit, and the value for each pixel indicates the signal strength for its corresponding voxel. The image stack was saved in VTK simple legacy format and is available on `https://github.com/wangdingkang/DiscreteMorse`.

DM-Skeleton This section provides intuition and descriptions for the DM-Skeleton pipeline; recall the workflow in Figure 1.

Step 1: Pre-processing Each image volume in VTK format was loaded as an image stack, and converted into a density field $\rho : K \rightarrow \mathbb{R}$ defined on the cubical complex (3D grid) K, where each vertex corresponds to a voxel in the input image and has a density value. If the input is fMOST data, then the density value at each voxel is simply the pixel value in the input raw image stack. For whole-brain tracer injection STP data, a significant portion of the raw images is background (see the example in Figure 1). Hence we first apply the learning-based process-detection module [56] to remove the background and segment the foreground. We further apply a Gaussian filter to smooth the values across the domain.

Step 2: Skeletonization. The Discrete Morse graph reconstruction algorithm [19,21] takes a density field of any dimension as input, and outputs a graph skeleton capturing center-lines passing through relatively high density regions. In our case, the input density field $\rho : K \rightarrow \mathbb{R}$ is defined at vertices of a cubical complex K of the domain. In all subsequent operations, only the 2-skeleton of this cubical complex K is needed, that is, we assume K consists of vertices, edges, and squares. Cubic cells are not needed in the algorithm.

To explain the main idea, consider first the smooth case where we have a smooth function $\rho : \Omega \rightarrow \mathbb{R}$ over the domain Ω. Consider the terrain of the density function values plotted over the domain (Fig. 2) where the terrain of a function defined on $\mathbb{R}^2$ is given. The underlying graph skeleton of ρ can be captured by

the mountain ridges of this terrain (Fig. 2a). Locally, the density along these mountain ridges is higher than the density off of them. These ridges form the so-called *1-stable manifold* of the function in Morse theory, and are defined by the integral lines "connecting" local maxima to saddle points (Fig. 2b). (An integral line is a curve in the domain where at any point on it, its tangent vector coincides with the gradient of the density field. Integral lines are thus intuitively flow lines, following the steepest descending direction of the density fields.)

Inside the algorithm, roughly speaking, ridges are associated with certain persistence values, as captured by the so-called persistent homology [6], which can be interpreted as an importance score. This makes it easy to filter out ridges of low importance, which are typically associated with noise, from the final output by providing the algorithm with a persistence threshold. An example of simplification for a 1D function is shown in Fig. 2c.

For the input to our algorithm, we have a density field $\rho : \mathsf{K} \to \mathbb{R}$ defined at vertices of a cubical complex K of the domain Ω (a 3D region). Following [19], discrete Morse theory [20] is used to capture the mountain ridges mentioned above, combined with the persistence algorithm to measure importance. See Supplementary Methods for a description of the algorithm. To improve the efficiency of the algorithm, we modified the algorithm of [19] so that it works directly with cubical complexes and also uses DIPHA [63] to compute persistence pairs in a distributed manner.

These mountain ridges cover the neural branches as locally, points along these neural branches tend to have relatively higher density (signal strength) than off the branches. The global nature of the 1-stable manifolds makes the output skeleton robust to small gaps in signal, and effective at capturing junctions; see e.g., Fig. 2a, where the global nature of 1-manifolds connects through low-density region around the Y-junction.

In ideal circumstances, we would find a persistence threshold that would remove all of the noise and only keep the ridges that make up the true neuron tree. However, because of the noisy nature of biological data and also the Discrete Morse graph reconstruction algorithm will not necessarily output a tree, we cannot take the algorithm's output as a final output. Instead, we first run the algorithm with a low persistence threshold such that we do not remove any ridges that would be part of an ideal output. Then we simplify the Morse graph skeleton in the next step.

Step 3: Simplification. The output of the above persistence-guided Morse-based framework is a geometric graph G, also referred to as the Morse graph skeleton. Next, G is converted into a spanning forest F. Prior to this, both boundary and estimated flow vector simplification can be applied to G to limit the number of unnecessary branches in the spanning forest. (The estimation of flow vector at each graph node can be found Initial vector-based Morse graph simplification in Supplementary Methods.) Further simplification strategy to remove false positives and an option to control the level of details presented in the final summarization.

Extraction of a forest. The Morse graph skeleton G extracted in Step 2 already serves as a good initial skeleton. Each arc $e \in E$ in graph $G = (V, E)$ is realized by a polygonal path, consisting of edges from the input grid K. Then, a forest (a collection of rooted trees) $F = \{T_0, ..., T_m\}$ from G (See Fig. 3a) is extracted, which is in fact a spanning forest of G (that is, F contains all vertices from V, and edges of F are from G). Here we assume we are given the set of tree roots $R = \{v_0, ..., v_m\}$. In our experiments, the positions of roots correspond to soma locations and injection sites for fMOST and tracer injection data, respectively. The injection site and soma positions can be provided by human annotators or automatically detected [57,58].

With G and roots $R = \{v_0, ..., v_m\} \subset V$, a weighted shortest path spanning forest algorithm is applied to produce the spanning forest F. Intuitively, the weight of an edge depends on the density values of its endpoints, and edges in the regions of higher density values should have larger weights (i.e., smaller distances). The details of the weighted shortest path spanning forest algorithm is introduced in Supplementary Methods. We remark that it may also be reasonable to use the minimum spanning forest of G. However, the shortest path spanning forest mimics the natural process of tracers spreading from the injection site, and has a better empirical performance than minimum spanning forest.

Tree simplification. Now consider any tree $T = (V_T, E_T)$ from the initial spanning forest constructed above.

The simplification based on persistence in Step 2 can remove false-positive branches to some extent, however, false positives with small density surrounded by background can still remain in each tree T. We further develop the following simplification strategy to remove these false positives.

First, scores $\overline{s}(v)$ are assigned to each tree node v in tree T; the details of calculating node scores are explained in Supplementary Methods. Next, one of the following two strategies is chosen to simplify T based on the $\overline{s}$ scores. The first strategy, called LeafBurner, starts from the leaves of T and iteratively removes leaves with scores at most τ to obtain the final simplified tree $T_s = (V_s, E_s)$. An originally internal tree node $v \in T$ will become a leaf if all of its children are already removed by LeafBurner in previous iterations. The second strategy, called RootGrower, instead expands a simplified tree T_s from the given root v_0 and gradually includes tree nodes from T with scores at least τ. At any moment, only children of those nodes already included in the simplified tree from earlier iterations will be considered. Each strategy has its own merits and disadvantages. For RootGrower, gaps (weak connections) might cause breaks, while LeafBurner can be potentially stuck at branches in remote noise with high scores and fail to remove them (See Fig. 3d).

Weight-preserving summarization In this process, our goal is to summarize voxels in the input proofread brain while conserving weight information. The weights of voxels in a certain region should be assigned to a tree node of the DM-Skeleton output in the same region. We use a similar idea when assigning scores in

the simplification step, but now only use the voxel weights. The summarization weight of a tree node v is total weights of voxels whose nearest neighbor in the tree node set is v. Additionally, there is an upper bound β_w for the distance allowed. More specifically, $summary(v) = \sum_{x \in I, d(x,v)=d(x,V_s), d(x,v)<\beta_w} \rho(x)$. The upper bound β_w is intentionally large to avoid missing weights. For the STP dataset, we used $\beta_w = 300\mu\text{m}$.

Additional features. For visualization purpose, a "thickness value" can be assigned to each tree node in accordance with the SWC format (Fig. 3e). When calculating the density score for each tree node in the simplified tree T_s (Supplementary Methods), voxel information is leveraged in the surrounding area of that tree node. Here a similar idea is applied, we associate each voxel in the segmented foreground (in the case that the input is not preprocessed in Step 1, such as the fMOST data, then we use a low threshold to remove background) to its nearest neighbor in V_s within distance β_t which is specified later. The thickness value of a node v is proportional to the square root of the number of associated voxels. More specifically, $r(v) = c \cdot \sqrt{N(v)}$, where c is a certain constant, $r(v)$ is the radius (thickness) assigned to node v, and $N(v)$ is the number of associated voxels. Any tree node with no voxel associated is set to have the minimum non-zero raidus, i.e., c. Empirically, the default value of c is 1. In the experiment on STP dataset, we set $\beta_t = 20\mu\text{m}$ for the best visualization (Fig. 5a).

An option for users to control the desired level of detail is provided. In particular, given the simplified tree $T_s = (V_s, E_s)$ rooted at v_0, an "importance" value is assigned to each branch, and the top k (a user provided threshold) branches are then greedily selected. Here, a branch is a unique path from the root node v_0 to a degree-1 leaf node l. It turns out that branch length works well as the importance value. An example on STP data is shown in Fig. 5a.