Topology and Intelligent Data Analysis*

V. Robins[1], J. Abernethy[2], N. Rooney[2], and E. Bradley[2]

[1] Department of Applied Mathematics
Research School of Physical Sciences and Engineering
The Australian National University
ACT 0200 Australia

[2] University of Colorado
Department of Computer Science
Boulder, CO 80309-0430

Abstract. A broad range of mathematical techniques, ranging from statistics to fuzzy logic, have been used to great advantage in intelligent data analysis. Topology—the fundamental mathematics of shape—has to date been conspicuously absent from this repertoire. This paper shows how topology, properly reformulated for a finite-precision world, can be useful in intelligent data analysis tasks.

1 Introduction

Topology is the fundamental descriptive machinery for shape. Putting its ideas into real-world practice, however, is somewhat problematic, as traditional topology is an infinite-precision notion, and real data are both limited in extent and quantized in space and time. The field of *computational topology* grew out of this challenge[5,9]. Among the formalisms in this field is the notion of *variable-resolution topology*, where one analyzes the properties of the data—*e.g.*, the number of components and holes, and their sizes—at a variety of different precisions, and then deduces the topology from the limiting behavior of those curves. This framework, which was developed by one of the authors of this paper (Robins)[20,21,22], turns out to be an ideal tool for intelligent data analysis.

Our approach to assessing connectedness and components in a data set has its roots in the work of Cantor. We define two points as *epsilon (ϵ) connected* if there is an ϵ-chain joining them; all points in an *ϵ-connected set* can be linked by an ϵ-chain. For the purposes of this work, we use several of the fundamental quantities introduced in [21]: the number $C(\epsilon)$ and maximum diameter $D(\epsilon)$ of the ϵ-connected components in a set, as well as the number $I(\epsilon)$ of ϵ-isolated points—that is, ϵ-components that consist of a single point. As demonstrated in [20,22], one can compute all three quantities for a *range* of ϵ values and deduce the topological properties of the underlying set from their limiting behavior. If the underlying set is connected, the behavior of C and D is easy to understand.

* Supported by NSF #ACI-0083004 and a Grant in Aid from the University of Colorado Council on Research and Creative Work.

M.R. Berthold et al. (Eds.): IDA 2003, LNCS 2810, pp. 111–122, 2003.

When ϵ is large, all points in the set are ϵ-connected and thus it has one ϵ-component ($C(\epsilon) = 1$) whose diameter $D(\epsilon)$ is the maximum diameter of the set. This situation persists until ϵ shrinks to the largest interpoint spacing, at which point $C(\epsilon)$ jumps to two and $D(\epsilon)$ shrinks to the larger of the diameters of the two subsets, and so on.

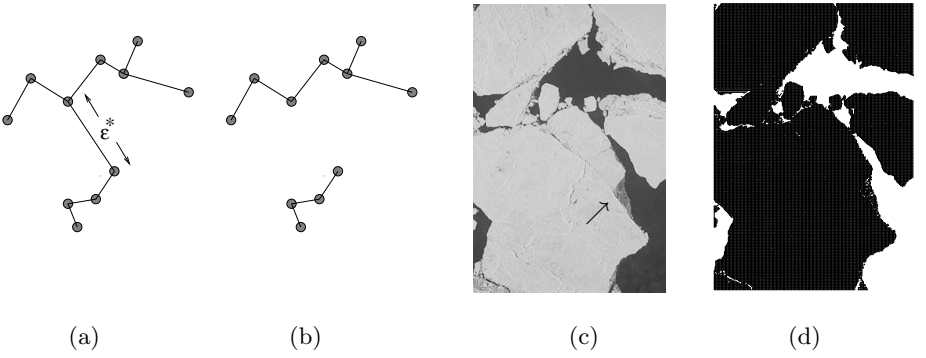


Fig. 1. Computing connectedness: (a) The *minimal spanning tree* (MST) whose edges connect nearest neighbors in a data set (b) This set contains two ϵ-connected components if ϵ is slightly less than ϵ^* (c) An aerial image of a patch of the arctic ocean; the arrow indicates a bay full of small ice floes (d) The MST of the ice pixels in (c)

When ϵ reaches the *smallest* interpoint spacing, every point is an ϵ-connected component, $C(\epsilon) = I(\epsilon)$ is the number of points in the data set, and $D(\epsilon)$ is zero. If the underlying set is a disconnected fractal, the behavior is similar, except that C and D exhibit a stair-step behavior with changing ϵ because of the scaling of the gaps in the data. When ϵ reaches the largest gap size in the middle-third Cantor set, for instance, $C(\epsilon)$ will double and $D(\epsilon)$ will shrink by 1/3; this scaling will repeat when ϵ reaches the next-smallest gap size, and so on. [21] derives, explains, and demonstrates these results in detail, and discusses the obvious link to fractal dimension. Pixellation can also cause stair-step effects because it quantizes inter-point distances; this can be confusing, as it mistakenly suggests that the underlying data set has a disconnected fractal structure.

Our computer implementation of these connectedness calculations relies on constructs from discrete geometry: the minimal spanning tree (MST) and the nearest neighbor graph (NNG). The former is the tree of minimum total branch length that spans the data; see Figure 1(a) for an example. To construct the MST, one starts with any point in the set and its nearest neighbor, adds the closest point, and repeats until all points are in the tree[1]. The NNG is a directed graph that has an edge from x_A to x_B if x_B is the nearest neighbor of x_A. To

[1] Our implementation actually employs Prim's algorithm[7], which is a bit more subtle. It begins with any vertex as the root and grows the MST in stages, adding at each stage an edge (x, y) and vertex y to the tree if (x, y) is minimal among all edges where x is in the tree and y is not.

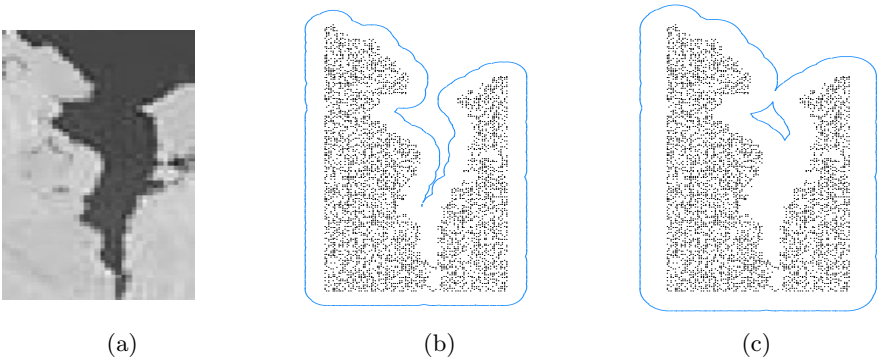
construct it, one starts with the MST and keeps the shortest edge emanating from each point. Both algorithms may be easily implemented in R^d; the computational complexity of the MST is $O(N^2)$ in general and $O(N \log N)$ in the plane, where N is the number of data points. To compute C and I from these graphs, one simply counts edges. $C(\epsilon)$, for example, is one more than the number of MST edges that are longer than ϵ, and $I(\epsilon)$ is the number of NNG edges that are longer than ϵ. Note that one must count NNG edges with multiplicity, since x_A being x_B's nearest neighbor does not imply that x_B is x_A's nearest neighbor (*i.e.*, if a third point x_C is even closer to x_A). Note, too, that the MST and NNG need only be constructed once; all of the C and I information for different ϵ values is captured in their edge lengths. To identify the individual ϵ-components, one simply removes edges that are longer than ϵ. Diameters $D(\epsilon)$ of ϵ-connected components are then found using standard computational geometry techniques.

These trees and the information encoded in their edges are extremely useful in intelligent data analysis. Vertical jumps in $C(\epsilon)$, for instance, take place at ϵ values that correspond to sizes of gaps in the dataset. $D(\epsilon)$ is of obvious utility in describing the sizes of objects, and $I(\epsilon)$ can be used to filter out noise. The MST itself can bring out structure. Figure 1(c), for example, shows a black-and-white of a patch of the arctic ocean. The small, right-facing bay just below and to the left of center (arrow), which faces into the prevailing wind, is full of small ice floes. The floes are so small that this region simply appears as grey in the raw image, but a MST of the white (ice) pixels brings out the small pockets of ice and water, as shown in part (d). All of these quantities mesh well with experimental reality: the MST of a sea-ice image, for instance, captures exactly how many floes will be resolved if the instrument has a precision of 10m, 5m, 1m, etc. The examples in Section 2 expand upon some of these ideas. MSTs can also be used to aggregate points into structures, and so they have been widely used in the kinds of clustering tasks that arise in pattern recognition[10] and computer vision[4]. Their branching structure can also be exploited—*e.g.*, to identify orbit types in dynamical systems[27] or to find discontinuities in bubble-chamber tracks[28]. Clustering and coherent-structure extraction are not the only applications of the MST; additive noise creates small transverse 'hairs' on these trees, and so filtering out those points is simply a matter of pruning the associated edges. Section 3 covers this idea in more depth.

Another fundamental topological property is the number, size, and shape of the *holes* in an object. One way to characterize this mathematically is via homology. This branch of topology describes structure using algebraic groups that reflect the connectivity of an object in each dimension. The rank of each homology group is called the *Betti number*, b_k. The zeroth-order Betti number, b_0, counts the number of connected components. Geometric interpretations of the other Betti numbers depend on the space the object lives in: in 2D, b_1 is the number of holes, while in 3D b_1 is (roughly speaking) the number of open-ended tunnels and b_2 is the number of enclosed voids. See our website[1] for some images that make these definitions more concrete. The definition of the homology groups requires a discrete geometric representation of the object, *e.g.*, a triangulation

of subsets of 2D, and simplicial complexes in three or more dimensions. See Munkres[16] for further details.

In order to put these hole-related ideas into computational practice, we use the α-shape algorithm developed by Edelsbrunner[12], which computes the Betti numbers from triangulations that capture the topology of the coarse-grained data at different resolutions. The full algorithm involves some fairly complex computational geometry techniques, but the basic idea is to "fatten" the data by forming its α-neighborhood: that is, to take the union of balls of radius α centered at each data point. When α is large (on the order of the diameter of the data) this α-neighborhood is a single connected blob. As α decreases, the α-neighborhood shrinks and more shape detail is resolved in the data. When α is just small enough that a ball of radius α can fit inside the data without enclosing any data points, a hole is created in the α-neighborhood. For very small α—less than half the minimum spacing of the data—the individual data points are resolved. The associated calculations are non-trivial, but fast algorithms have been developed for α-shapes of 2D and 3D data[8], and associated software is available on the world-wide web[2].



(a) (b) (c)

Fig. 2. Computing holes: α-neighborhoods can be used to detect whether the image in part (a) contains a bay, and, if so, how wide its mouth is. Images (b) and (c) show α-neighborhoods of the set of white (ice) pixels in (a) for a few interesting α values.

As in the connectedness discussion above, one can compute the number of holes in an α-neighborhood of a data set while varying α, and then use that information to deduce the topological properties of the underlying set[19,20]. There is one important difference, however. The *geometry* of the set can create holes in the α-neighborhoods, even if the set contains no holes. This effect is demonstrated in Figure 2. Mathematically, this problem can be resolved by incorporating information about how the set maps inside its α-neighborhood. This leads to the definition of a *persistent Betti number*, which was introduced in [19]: for $\epsilon < \alpha$, $b_k(\epsilon, \alpha)$ is the number of holes in the α-neighborhood for which there are corresponding holes in the ϵ-neighborhood (equivalently, the number of holes in the ϵ-neighborhood that do not get filled in by forming the fatter α-

neighborhood). These persistent Betti numbers are well defined for sequences of complexes that provide successively better approximations to a manifold[19] and are computable using linear algebra techniques. Recently, Edelsbrunner and collaborators have made a similar definition of persistent Betti number specifically for α-shapes, and devised an incremental algorithm for their calculation[11].

While the non-persistent holes effect makes it difficult to make a correct diagnosis of the underlying topology, it has important implications for intelligent data analysis because it gives us geometric information about the embedding of the set in the plane. This can be very useful in the context of coherent structure extraction. Consider a narrow bay in an icepack, as shown in Figure 2. In this case, $b_1(\alpha)$ computed for the set of white (ice) pixels would be zero for α smaller than half the width of the mouth of the bay, zero for α larger than the largest radius of its interior, and one in between—that is, where an α ball fits inside the bay, but not through its mouth. Note that the α value where the spurious hole first appears is exactly the half-width of the entrance to the bay. If there were a *hole* in the ice, rather than a bay, $b_1(\alpha)$ would be a step function: zero when α is greater than the largest radius of the hole and one when it is less. This technique for finding and characterizing coherent structures whose defining properties involve holes and gaps of various shapes—ponds, isthmuses, channels, tunnels, etc.—is potentially quite useful in intelligent data analysis. More examples and discussion follow in Section 2.

2 Coherent Structure Extraction

In this section, we present three examples that demonstrate how to use our variable-resolution topology techniques to find coherent structures in aerial images of sea ice[2]. Scientists look for several things in these kinds of images: open water or "leads," ice floes, and melt ponds. all of which are visible in Figure 3. Climatology studies that track the seasonal evolution of these coherent structures are a major current thrust of Arctic science, but they require an automated way of analyzing large numbers of images. While it is certainly possible for someone to sit down and outline the floes in an image with a mouse, modern scientific instruments can easily generate gigabytes of data in an hour[13], and any human analyst will rapidly be overwhelmed (and bored) by the task. Traditional image-processing and machine-learning tools can help with tasks like this, to a point—*e.g.*, edge-finding, contrast enhancement, etc.—but topology-based methods are an even better solution. The following paragraphs treat three examples in detail: finding a lead through an icepack, distinguishing regions of different ice/water concentration, and studying how the number and size of melt ponds are distributed in sea ice.

The size, shape, and position of floes and leads is of obvious interest to travelers in the polar regions. Finding a path for a ship through a complicated region of the ice pack, in particular, is a common practical problem, and the

[2] courtesy of D. Perovich from CRREL.

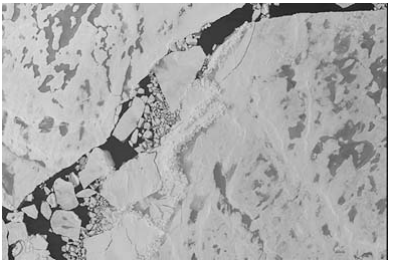
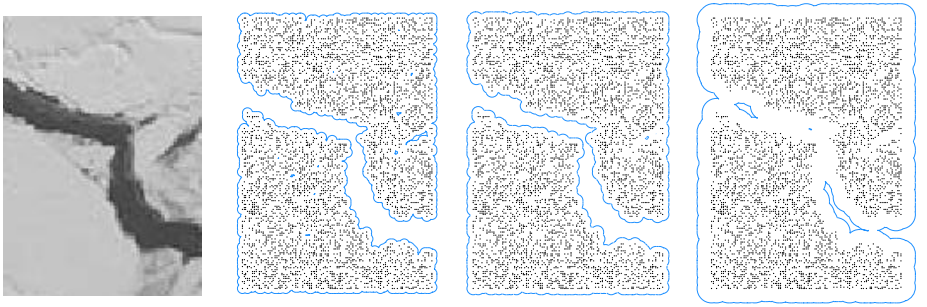


Fig. 3. The arctic ice pack is made up of open water, ice, and melt ponds, which image as black, white, and grey, respectively. Studying the climatology of this system—the seasonal evolution of the different kinds of *coherent structures*—is a major current thrust of Arctic science, and doing so requires an automated way of analyzing large numbers of images.

variable-resolution computational topology techniques described in Section 1 are quite useful in doing so. Consider an nm-wide ship that must traverse the patch of ocean depicted in Figure 4(a), from left to right. Solving this problem



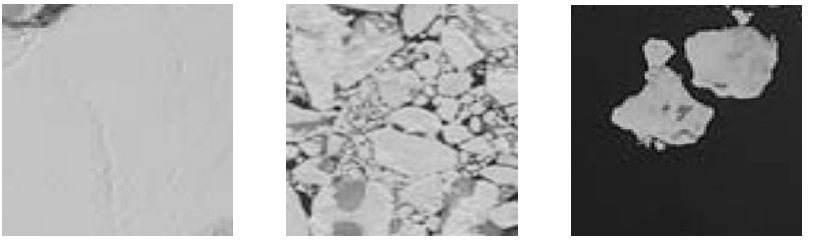
(a) (b) (c) (d)

Fig. 4. Using α-shapes to find a path through an icepack: (b), (c), and (d) show three different α-neighborhoods of the set of ice pixels in (a).

requires assessing the holes in the ice, so we threshold the data and discard the dark water pixels, then apply the α-shape algorithm with a radius $\alpha = n/2$. The results, shown in Figure 4(b), identify all holes that are at least nm wide. There are actually two holes in this α-shape, one corresponding to the kinked, diagonal channel and a much smaller one about halfway up the right-hand edge of Figure 4(b). The next step is to use simple computational geometry techniques on these holes to ascertain which ones touch both sides of the image

and then determine which is the shortest[3]. Parts (c) and (d) of the Figure show the effects of raising α beyond $n/2$m, demonstrating the relationship between the channel shape and the holes in the α-neighborhood—*e.g.*, successively resolving the various tight spots, or identifying regions of water that are at least α meters from ice.

More generally, α shapes can be used to assess the mix of water and ice in a region—not just the relative area fractions, but also the distribution of sizes of the regions involved—or to find and characterize regions of different ice concentration. Depending on the task at hand, one can analyze either the



(a) (b) (c)

Fig. 5. Using α-shapes to distinguish different ice/water distributions: (a) almost-solid ice (b) a lead filled with various-size floes (c) open water with a few big floes. An α-shapes analysis of these images yields a direct measure of the morphology of the ice and water.

water pixels (where α-holes are α-size ice floes) or the ice pixels, where holes are leads. When the ice is close to solid, as in Figure 5(a), the narrow water channel is resolved as a small α-hole in the set of ice pixels, similar to the lead in the previous paragraph. When the image contains large areas of open water and a few floes, as in part (c), an α-shape analysis of the water pixels resolves a few large holes over a wide range of α values—after a band of smaller αs where the α-neighborhoods are filling in the various gaps and bumps in the floes and creating spurious holes, as discussed in conjunction with Figures 2 and 4. The middle image is more interesting; the wide distribution of floe sizes translates to a wide α range where a small change in that parameter resolves a few new holes in the set of water pixels (*i.e.*, floes). All of these results accurately reflect important properties of the data.

The temporal evolution of the albedo of Arctic sea ice is of great importance to climate modelers because of its role in heat transfer. A key factor governing the sea-ice albedo in summer is the size distribution of the melt ponds, which appears to take the form of a power law[18]. We can easily automate these distribution calculations using α-shapes. The first step is to threshold the data

[3] Note that this development assumes a circular ship; for oblong ships, one should use an α value that is the largest chord of the hull.

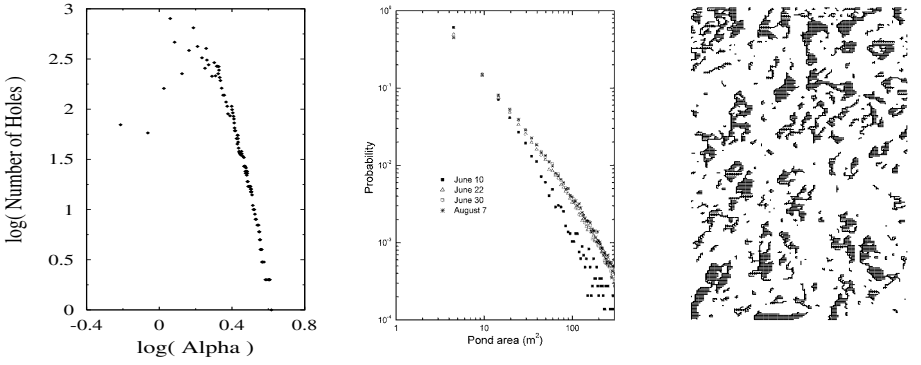


Fig. 6. Computational topology and melt ponds: (a) An α-shapes analysis of the ponds in the lower-right quadrant of Figure 3 (b) Corresponding figure from [18] (c) Connected components in pond data

so that the 'nonpond' points (both the ice and the water) are black; the second is to look for holes in that data. The results, shown in Figure 6(a), corroborate the power-law distribution hypothesis quite nicely; after an initial region below about $\log \alpha = 0.3$, where changes in this parameter are resolving promontories and bays in the melt ponds, the log-log plot is fairly linear. Part (b) reproduces the corresponding figure from [18] for comparison. The power-law slopes in (a) and (b) are different because the former plot has pond *radius* on the horizontal axis, while the latter has pond *area*. If the extra power of two is taken into account, the slopes of (a) and (b) actually match to within 8%[4]. We can obtain similar curves using the connectedness tools described in Section 1: *e.g.*, computing a minimal spanning tree of the 'pond' points in the image, as in Figure 6(c), and then extracting the connected components and computing their areas and/or diameters.

Our next step will be to combine these notions of multiple-resolution topology with some ideas from spatial statistics and artificial intelligence in order to handle the ill-defined nature of real coherent structures and to recognize the spatial and temporal patterns in which those structures move, evolve, and interact. Spatial statistics are useful in this application because computational topology produces aggregate measures over a whole dataset. Adding a sliding window to the process, and manipulating its geometry intelligently—an approach that we term *topospatial analysis*—allows one to distinguish sub-regions where those measures are different. Artificial intelligence (AI) techniques are useful because coherent structures are not crisp. The Gulf Stream, for instance, is a hot-water jet extending from the eastern seaboard of the US into the Atlantic. It does not have a natural, well-defined boundary. Rather, the temperature at its edges falls off smoothly, and so any threshold-based definition is necessarily somewhat

[4] The residual discrepancy is due to the fact that not all ponds are perfectly circular.

arbitrary. Moreover, coherent structures are much easier to recognize than to describe, let alone define in any formal way. The AI community has developed a variety of representations and techniques for problems like these, and these solutions mesh well with topospatial analysis.

3 Filtering

Real-world data invariably contain noise of one form or another. Traditional linear filters are useful in removing some kinds of noise, but not all. Chaotic behavior, for instance, is both broad band and sensitively dependent on system state, so linear or Fourier-based filters—which simply remove all signal in some band of the power spectrum—can alter important features of the dynamics, and in a significant manner[26]. In cases like this, topology-based filtering can be a much better solution. As mentioned in Section 1, noise often manifests as isolated points (*e.g.*, from an intermittent glitch in a sensor), and variable-resolution topology is an effective way to identify these points. Figure 7 demonstrates the basic ideas. The spanning tree clearly brings out the displacement of the noisy

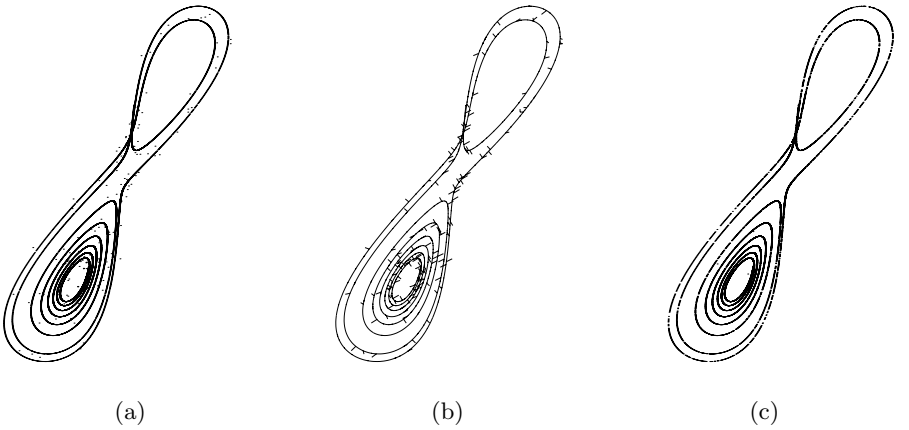


Fig. 7. The effects of noise upon data topology. The dataset in part (a) is a canonical example in dynamical systems—the Lorenz attractor—each point of which was perturbed, with probability 0.01, by a small amount of additive noise. The MST in part (b) clearly shows these noisy points. Part (c) shows the same data set after processing with a topology-based filter that prunes the 'hairs' off the MST.

points from the rest of the orbit, both in direction and in edge length. Specifically, if the noise magnitude is large compared to the inter-point spacing, the edges joining the noisy points to the rest of the tree are longer than the original edges. (Of course, if the magnitude of the noise is *small* compared to that spacing, the associated MST edges will not be unusually long, and the noisy points are not

so obvious.) This kind of *separation of scale* often arises when two processes are at work in the data—*e.g.*, signal and noise. Because variable-resolution topology is good at identifying scale separation, we can use it to identify and remove the noisy points. The first step is to look for a second peak in the edge-length distribution of the MST (equivalently, a shoulder in the plot $I(\epsilon)$ of the number of isolated points as a function of ϵ). The maximum edge length of the MST of the *non*-noisy data occurs in the interval between the two peaks (or at the location of the $I(\epsilon)$ breakpoint). At that value, most of the noisy points—and few of the regular points—are ϵ-isolated. The next step is to prune all MST edges that exceed that length, removing the points at their termini[5]. Part (c) of the Figure shows the results; in this case, the topology-based filter removed 534 of the 545 noisy points and 150 of the 7856 non-noisy points. This translates to 98.0% success with a 1.9% false positive rate. These are promising results—better than any linear filter, and comparable to or better than the noise-reduction techniques used by the dynamical systems community[3]. Increasing the pruning length, as one would predict, decreases the false-positive rate; somewhat less intuitively, though, larger pruning lengths do *not* appear to significantly affect the success rate—until they become comparable to the length scales of the noise. As described in [23], these rates vary slightly for different types and amounts of noise, but remain close to 100% and 0%; even better, the topology-based filter does not disturb the dynamical invariants like the Lyapunov exponent.

While topology-based filtering is very effective, it does come with some caveats. For example, the method simply *removes* noisy points; it does not deduce where each one 'should be' and move it in that direction. There are several obvious solutions to this, all of which add a bit more geometry into the recipe—*e.g.*, averaging the two points on either side of the base of the edge that connects an isolated point to the rest of the trajectory. Rasterized images bring up a different set of problems, as noise does not move points around in these images, but rather reshades individual pixels. The computer vision community distinguishes these two kinds of noise as "distortion" and "salt and pepper," respectively. Because the metric used in the MST captures distances between points, it is more effective at detecting the former than the latter. Space limitations preclude a more-thorough treatment of this issue here; for more details, see[6].

4 Conclusion

The mathematical framework of variable-resolution topology has tremendous potential for intelligent data analysis. It is, among other things, an efficient way

[5] The exact details of the pruning algorithm are somewhat more subtle than is implied above because not all noisy points are terminal nodes of the spanning tree. Thus, an algorithm that simply deletes *all* points whose connections to the rest of the tree are longer than the pruning length can sever connections to other points, or clusters of points. This is an issue if one noisy point creates a 'bridge' to another noisy point and only one of the associated MST edges is longer than the pruning length.

to automate the process of finding and characterizing coherent structures. Figure 6(b), for instance, was constructed by hand-processing roughly 2000 images, which required roughly 50 working days[17]; Figure 6(a) required only a few minutes of CPU time. Because it reveals separation of scale, variable-resolution topology can also be useful in noise removal. In a series of numerical and laboratory experiments, a topology-based filter identified close to 100% of the noisy points from dynamical system data sets, with a false-positive rate of only a few percent. Separation of scale is fundamental to many other processes whose results one might be interested in untangling, so this method is by no means limited to filtering applications—or to dynamical systems.

There have been a few other topology-based approaches to filtering. Mischaikow *et al.*[15], for example, use algebraic topology to construct a coarse-grained representation of the data. This effectively finesses the noise issue, and thus constitutes a form of filtering. Rather than use algebraic topology to construct a useful coarse-grained representation of the dynamics, our approach uses *geometric* topology to remove noisy points while working in the original space, which allows us to obtain much finer-grained results.

A tremendous volume of papers on techniques for reasoning about the structure of objects has appeared in various subfields of the computer science literature. Very few of these papers focus on distilling out the topology of the coherent structures in the data. Those that do either work at the pixel level (*e.g.*, the work of Rosenfeld and collaborators [24]) or are hand-crafted for a particular application (*e.g.*, using wavelets to find vortices in images of sea surface temperature[14]), and all are constrained to two or three dimensions. Our framework is algorithmically much more general, and it works in arbitrary dimension. Many geographic information systems[25] and computer vision[4] tools incorporate simple topological operations like adjacency or connectedness; these, too, generally only handle 2D gridded data, and none take varying resolution into account—let alone exploit it.

References

1. `http://www.cs.colorado.edu/~lizb/topology.html`.
2. `http://www.alphashapes.org`.
3. H. Abarbanel. *Analysis of Observed Chaotic Data*. Springer, 1995.
4. D. Ballard and C. Brown. *Computer Vision*. Prentice-Hall, 1982.
5. M. Bern and D. Eppstein. Emerging challenges in computational topology, 1999.
6. E. Bradley, V. Robins, and N. Rooney. Topology and pattern recognition. Preprint; see `www.cs.colorado.edu/~lizb/publications.html`.
7. T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 2001. pp 570–573.
8. C. Delfinado and H. Edelsbrunner. An incremental algorithm for Betti numbers of simplicial complexes on the 3-sphere. *Computer Aided Geometric Design*, 12:771–784, 1995.
9. T. Dey, H. Edelsbrunner, and S. Guha. Computational topology. In B. Chazelle, J. Goodman, and R. Pollack, editors, *Advances in Discrete and Computational Geometry*. American Math. Society, Princeton, NJ, 1999.

10. R. Duda and P. Hart. *Pattern Classification.* Wiley, New York, 1973.
11. H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. In *IEEE Symposium on Foundations of Computer Science*, pages 454–463, 2000.
12. H. Edelsbrunner and E. Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics*, 13:43–72, 1994.
13. U. Fayyad, D. Haussler, and P. Stolorz. KDD for science data analysis: Issues and examples. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 1996.
14. H. Li. Identification of coherent structure in turbulent shear flow with wavelet correlation analysis. *Transactions of the ASME*, 120:778–785, 1998.
15. K. Mischaikow, M. Mrozek, J. Reiss, and A. Szymczak. Construction of symbolic dynamics from experimental time series. *Physical Review Letters*, 82:1144–1147, 1999.
16. J. Munkres. *Elements of Algebraic Topology.* Benjamin Cummings, 1984.
17. D. Perovich. Personal communication.
18. D. Perovich, W. Tucker, and K. Ligett. Aerial observations of the evolution of ice surface conditions during summer. *Journal of Geophysical Research: Oceans*, 10.1029/2000JC000449, 2002.
19. V. Robins. Towards computing homology from finite approximations. *Topology Proceedings*, 24, 1999.
20. V. Robins. *Computational Topology at Multiple Resolutions.* PhD thesis, University of Colorado, June 2000.
21. V. Robins, J. Meiss, and E. Bradley. Computing connectedness: An exercise in computational topology. *Nonlinearity*, 11:913–922, 1998.
22. V. Robins, J. Meiss, and E. Bradley. Computing connectedness: Disconnectedness and discreteness. *Physica D*, 139:276–300, 2000.
23. V. Robins, N. Rooney, and E. Bradley. Topology-based signal separation. *Chaos.* In review; see `www.cs.colorado.edu/~lizb/publications.html`.
24. P. Saha and A. Rosenfeld. The digital topology of sets of convex voxels. *Graphical Models*, 62:343–352, 2000.
25. S. Shekhar, M. Coyle, B. Goyal, D. Liu, and S. Sarkar. Data models in geographic information systems. *Communications of the ACM*, 40:103–111, 1997.
26. J. Theiler and S. Eubank. Don't bleach chaotic data. *Chaos*, 3:771–782, 1993.
27. K. Yip. *KAM: A System for Intelligently Guiding Numerical Experimentation by Computer.* Artificial Intelligence Series. MIT Press, 1991.
28. C. Zahn. Graph-theoretical methods for detecting and describing Gestalt clusters. *IEEE Transactions on Computers*, C-20:68–86, 1971.