Analyzing Collective Motion with Machine Learning and Topology

Dhananjay Bhaskar,[1] Angelika Manhart,[2] Jesse Milzman,[3] John T. Nardini,[4] Kathleen Storey,[5] Chad M. Topaz,[6] and Lori Ziegelmeier[7, a)]

[1)] *Center for Biomedical Engineering, Brown University, Providence, RI, USA*

[2)] *Department of Mathematics, University College London, London, UK*

[3)] *Department of Mathematics, University of Maryland, College Park, MD, USA*

[4)] *The Statistical and Applied Mathematical Sciences Institute (SAMSI), Durham, NC, USA*

[5)] *Department of Mathematics, University of Michigan, Ann Arbor, MI, USA*

[6)] *Department of Mathematics and Statistics, Williams College, Williamstown, MA, USA*

[7)] *Department of Mathematics, Statistics, and Computer Science, Macalester College, Saint Paul, MN, USA*

(Dated: 27 August 2019)

We use topological data analysis and machine learning to study a seminal model of collective motion in biology [D'Orsogna *et al.*, Phys. Rev. Lett. 96 (2006)]. This model describes agents interacting nonlinearly via attractive-repulsive social forces and gives rise to collective behaviors such as flocking and milling. To classify the emergent collective motion in a large library of numerical simulations and to recover model parameters from the simulation data, we apply machine learning techniques to two different types of input. First, we input time series of order parameters traditionally used in studies of collective motion. Second, we input measures based in topology that summarize the time-varying persistent homology of simulation data over multiple scales. This topological approach does not require prior knowledge of the expected patterns. For both unsupervised and supervised machine learning methods, the topological approach outperforms the traditional one.

PACS numbers: 02.40.Re, 05.45.-a, 87.23.Cc
Keywords: collective motion; swarming; dynamics; topological data analysis; machine learning

A fundamental goal in the study of complex, nonlinear systems is to understand the link between local rules and collective behaviors. For instance, what influence does coupling between oscillators have on the ability of a network to synchronize? How does policing affect hotspots of crime in urban areas? Why do the movement decisions that fish make lead to schooling structures? We examine such links by bringing together tools from applied topology and machine learning to study a seminal model of collective motion that replicates behavior observed in biological swarming, flocking, and so forth. Studies of collective motion often focus on the so-called forward problem: given a particular mathematical model, what dynamics are observed for different parameters input to the model? In contrast, we study an inverse problem: given observed data, what model parameters could have produced it? Also, given observed data, what paradigmatic dynamics are being exhibited? To answer these questions, we use machine learning techniques and find that they achieve higher accuracy when applied to topological summaries of the data – called crockers – as compared to more traditional summaries of the data that are commonly used in biology and physics to characterize collective motion. Ours is the first study to use crockers for nonlinear dynamics classification and parameter recovery.

I. INTRODUCTION

Fundamental to many nonlinear systems is the link between local rules and global behaviors. For instance, what influence does coupling between oscillators have on the ability of a network to synchronize? How does policing affect hotspots of crime in urban areas? Why do the movement decisions that fish make lead to schooling structures? In this paper, we consider the relationship between local rules and global behavior in service of two tasks of interest in nonlinear science, namely classification of dynamics and parameter recovery. We conduct this study in the context of one of the aforementioned applications: collective motion in biology.

Animal groups such as flocks, herds, schools, and swarms are ubiquitous in nature[1,2] and inspire numer-

[a)] Electronic mail: lziegel1@macalester.edu

ous mathematical models[3–6] and biomimetic approaches to engineering and computer science.[7,8] In these groups, two levels of animal behavior come into play. First, at the individual level, organisms make decisions about how to move through space. It is well-established in the biological literature that social interactions between organisms play a key role. Second, at the group level, collective motion may occur, where animals coordinate and produce emergent patterns. Mathematical models can help link these two levels of behavior. This type of linkage is fundamental not only to collective motion but to innumerable nonlinear systems displaying collective behavior.

While there is a vast literature on mathematical models of collective motion, we focus on the influential model of D'Orsogna *et al.*[9] This model describes agents whose movement is determined by self-propulsion, drag, and social attraction-repulsion, forces frequently used in collective motion studies.[10,11] The model produces paradigmatic behaviors such as rotating rings, vortices, disorganized swarms, and traveling groups; these mimic fish schools, swarms of midges, bird flocks, and more.[12–14]

Studies of collective motion models often focus on a *forward problem*: given a particular model, what dynamics are observed for different parameters? In our present work, however, we study an *inverse problem*: given observed data, what parameters could have produced it? Also, given observed data, what paradigmatic dynamics are being exhibited?[15,16] Our primary contribution is to answer these questions by using tools from topological data analysis and machine learning. Though we focus on collective motion in the D'Orsogna model, our protocol is applicable to a wide range of settings.

Machine learning and mathematical modeling have traditionally been viewed as separate ways of understanding data. On one hand, machine learning can extract predictions of complex relations within large data sets. On the other hand, modeling can be used to hypothesize how mechanisms lead to an observed behavior. There is a growing understanding, however, that modeling and machine learning can be used in synergy.[17]

Our study hinges on the use of topological data analysis (TDA), a set of tools that "help the data analyst summarize and visualize complex datasets."[18] TDA has played a pivotal role in studies of breast cancer, spinal cord injury, contagion, and other biological systems.[19–21] A primary tool in the TDA toolbox is *persistent homology*. *Homology* has to do with calculating certain topological characteristics, while *persistence* refers to examining which of these are maintained across multiple scales in the data. In its fundamental form, persistent homology provides a framework for describing the topology of a static data set. However, ideas from persistent homology have been extended to time-varying data. One approach is the *Contour Realization Of Computed k-dimensional hole Evolution in the Rips complex*, known more simply as a *crocker*.[22] A crocker shows contours of quantities called *Betti numbers* as a function of time and of persistence scale, providing a topological summary of time-varying point clouds of data. Recent work has shown how crockers can be used for exploratory data analysis of collective motion and for judging the fitness of potential mathematical models of experimental data.[22,23]

Our present work brings together mathematical modeling, machine learning, and TDA to study collective motion in the D'Orsogna model. More specifically, we construct crockers from numerical simulations of the model and use them as inputs to machine learning clustering and classification algorithms in order to identify different paradigmatic patterns and the model parameters that produce them. We compare this approach to a more traditional one, which uses order parameters commonly used in studies of collective motion. While the traditional order parameters are typically chosen using prior knowledge of the system, the TDA tools can be used with no prior knowledge and are problem-independent.

The rest of this paper is organized as follows. Section II describes our methods, namely, numerical simulation of the D'Orsogna model, computation of traditional order parameters and crockers, and machine learning techniques. Section III presents our results, including our primary finding: the topological approach outperforms the traditional one. We conclude in Section IV.

II. METHODS

A. D'Orsogna Model and Numerical Simulations

The D'Orsogna model[9,24] describes the motion of N interacting agents of mass m. Each agent is characterized by its position and velocity $\boldsymbol{x}_i, \boldsymbol{v}_i \in \mathbb{R}^d$, $i = 1, \ldots, N$. We focus on the two-dimensional case, $d = 2$, throughout this study, though the case for $d = 3$ has been considered in Ref. 25. Position and velocity obey the coupled, nonlinear ordinary differential equations:

$$\dot{\boldsymbol{x}}_i = \boldsymbol{v}_i, \tag{1a}$$

$$m\dot{\boldsymbol{v}}_i = (\alpha - \beta|\mathbf{v}_i|^2)\mathbf{v}_i - \vec{\nabla}_i U(\boldsymbol{x}_i), \tag{1b}$$

$$U(\boldsymbol{x}_i) = \sum_{j\neq i}^{N} \left[C_r e^{-|\boldsymbol{x}_i - \boldsymbol{x}_j|/\ell_r} - C_a e^{-|\boldsymbol{x}_i - \boldsymbol{x}_j|/\ell_a} \right], \tag{1c}$$

where $\vec{\nabla}_i$ is the gradient with respect to $\boldsymbol{x}_i$. Eq. (1a) states that the time derivative of position is velocity. Eq. (1b) is Newton's law, with the right hand side describing three forces acting on each agent: self-propulsion with strength α, nonlinear drag with strength β, and social interactions. These social interactions are attractive-repulsive, as specified by the Morse-type potential in (1c). The first term inside the brackets describes social repulsion of overall strength $C_r > 0$. Repulsion decays exponentially in space, with characteristic length scale of decay $\ell_r > 0$. The exponential decay reflects that organisms' ability to sense each other through sight, sound, or smell decays over distance. Restated, an organism will be more heavily influenced by near individuals than far

ones. The second term is signed oppositely, and hence describes social attraction. The summation in (1c) means that a given organism interacts with all other organisms, albeit with influence decaying exponentially in space.

Biological modeling studies typically assume that repulsion is strong but operates over a short length scale, while attraction is weaker but operates over a longer scale. For (1), this would mean $C_r > C_a$ and $\ell_r < \ell_a$. In this case, the Morse potential U has a well-defined minimum, representing the distance at which attraction and repulsion balance for two organisms in isolation. However, this distance is not the separation observed between individuals in a group because of the nonlinear all-to-all coupling. Regardless, as in Ref. 9, we do not enforce the restriction $C_r > C_a$ and $\ell_r < \ell_a$.

For our study, we set $N = 200$. After nondimensionalizing (1) we are left with four dimensionless parameters: α, β, $C = C_r/C_a$ and $l = l_r/l_a$. We set $\alpha = 1.5$, and $\beta = 0.5$, corresponding to a base case of parameters from Ref. 9, and explore the remaining parameter space by varying the ratios $C = C_r/C_a$ and $\ell = \ell_r/\ell_a$. Both C and ℓ will take on values in $\{0.1, 0.5, 0.9, 2.0, 3.0\}$, resulting in 25 different possible parameter combinations. For each combination, we perform 100 simulations using MATLAB's `ode45` function, seeded with initial values of $\boldsymbol{x}_i$ and $\boldsymbol{v}_i$ drawn from a uniform distribution on $[-1, 1]^d$. We simulate until $t = 100$, allowing the swarm to attain a dynamic equilibrium state. From the computed trajectories, we sample the positions and velocities every 0.05 time units. Thus, the final output is $\{(\boldsymbol{x}_i(t_j), \boldsymbol{v}_i(t_j))\}_{i=1,..,N}^{j=1,...,M}$ for $t_j = (j-1)\Delta t, M = 2001, \Delta t = 0.05$ for each of our 2500 simulations.

These simulations produce paradigmatic collective motion including single mills, double mills, double rings (referred to simply as rings in Ref. 9), collective swarms, and group escape, which we split into three distinct classes.[9] For the remainder of this paper, we refer to paradigmatic collective behaviors as *phenotypes*. The first column of Fig. 1 shows a representative snapshot of each major phenotype, and Fig. 2 shows the three distinct escape types. Table I lists the values of (C, ℓ) that produce each phenotype in our library of simulations. We describe these phenotypes in more detail at the end of Section II C.

TABLE I. Collective motion phenotypes and corresponding social interaction potential parameters in our library of simulations of the D'Orsogna model (1). Here, $C = C_r/C_a$, $\ell = \ell_r/\ell_a$, and $N = 200$. We fix the remaining parameters $\alpha = 1.5$ and $\beta = 0.5$ as in Ref. 9.

Phenotype	**Parameters** $(\mathbf{C}, \ell)$
Single mill	(0.5,0.1),(0.9,0.1),(2,0.1),(2,0.5), (3,0.1)
Double mill	(0.9,0.5)
Double ring	(0.1,0.1),(0.5,0.5),(0.9,0.9)
Collective swarm	(0.1,0.5),(0.1,0.9),(0.1,2),(0.1,3), (0.5,0.9),(0.5,2),(0.5,3),(0.9,2),(0.9,3)
Escape (symmetric)	(2,0.9),(3,0.9)
Escape (unsymmetric)	(2,2),(3,2),(3,3)
Escape (collective)	(2,3),(3,0.5)

B. Order Parameters

Investigators often use *order parameters* to summarize the output of collective motion experiments or simulations.[9,11,22] Summaries are necessary because it is impractical or impossible to manually inspect large amounts of raw output data. The order parameters are intended to suggest if and when certain types of group behavior emerge in a population, for instance, when many individuals move in the same direction or rotate with the same orientation. Typical order parameters used for (1) include polarization, angular momentum, absolute angular momentum, and the mean distance to the nearest neighbor[22]. We use these four in our present study, plotted for each phenotype in the second column of Fig. 1.

Group polarization $P(t)$ measures the degree of alignment between agents and is given by

$$P(t) = \left| \frac{\sum_{i=1}^{N} \mathbf{v}_i(t)}{\sum_{i=1}^{N} |\mathbf{v}_i(t)|} \right| \in [0, 1], \tag{2}$$

with $P = 1$ signifying that all agents have the same direction of motion. All phenotypes in Fig. 1 exhibit low $P(t)$, suggesting no translational flocking. Angular momentum $M_{ang}(t)$ can detect rotational motion, and is given by

$$M_{ang}(t) = \left| \frac{\sum_{i=1}^{N} \mathbf{r}_i(t) \times \mathbf{v}_i(t)}{\sum_{i=1}^{N} |\mathbf{r}_i(t)||\mathbf{v}_i(t)|} \right| \in [0, 1], \tag{3}$$

where $\mathbf{r}_i(t) = \mathbf{x}_i(t) - \mathbf{x}_{cm}(t)$, and $\mathbf{x}_{cm}(t)$ refers to the center of mass of the agents. A group with $M_{ang} = 1$ would have individuals sharing perfectly rotational motion. Following Ref. 24, we also consider the absolute angular momentum $M_{abs}(t)$, given by

$$M_{abs}(t) = \left| \frac{\sum_{i=1}^{N} |\mathbf{r}_i(t) \times \mathbf{v}_i(t)|}{\sum_{i=1}^{N} |\mathbf{r}_i(t)||\mathbf{v}_i(t)|} \right| \in [0, 1]. \tag{4}$$

Discrepancies between angular momentum and absolute angular momentum can distinguish a single mill from a double mill, in which counter-rotating agents cancel out each others' angular momentum. For example, we observe in Fig. 1 that $M_{\text{abs}}(t) \approx M_{\text{ang}}(t)$ for the single mill, whereas $M_{\text{abs}}(t) > M_{\text{ang}}(t)$ for the double mill. Finally, we consider the mean distance to nearest neighbor, $D_{NN}(t)$, which may distinguish group escape behavior from other phenotypes.[26] $D_{NN}(t)$ is given by

$$D_{NN}(t) = \frac{1}{N} \sum_{i=1}^{N} \min_{j \neq i} |\mathbf{x}_i(t) - \mathbf{x}_j(t)| \in \mathbb{R}_{\geq 0}.$$

$D_{NN}(t)$ becomes very large for the escape phenotype over time in Fig. 1 as the particles act repulsively.

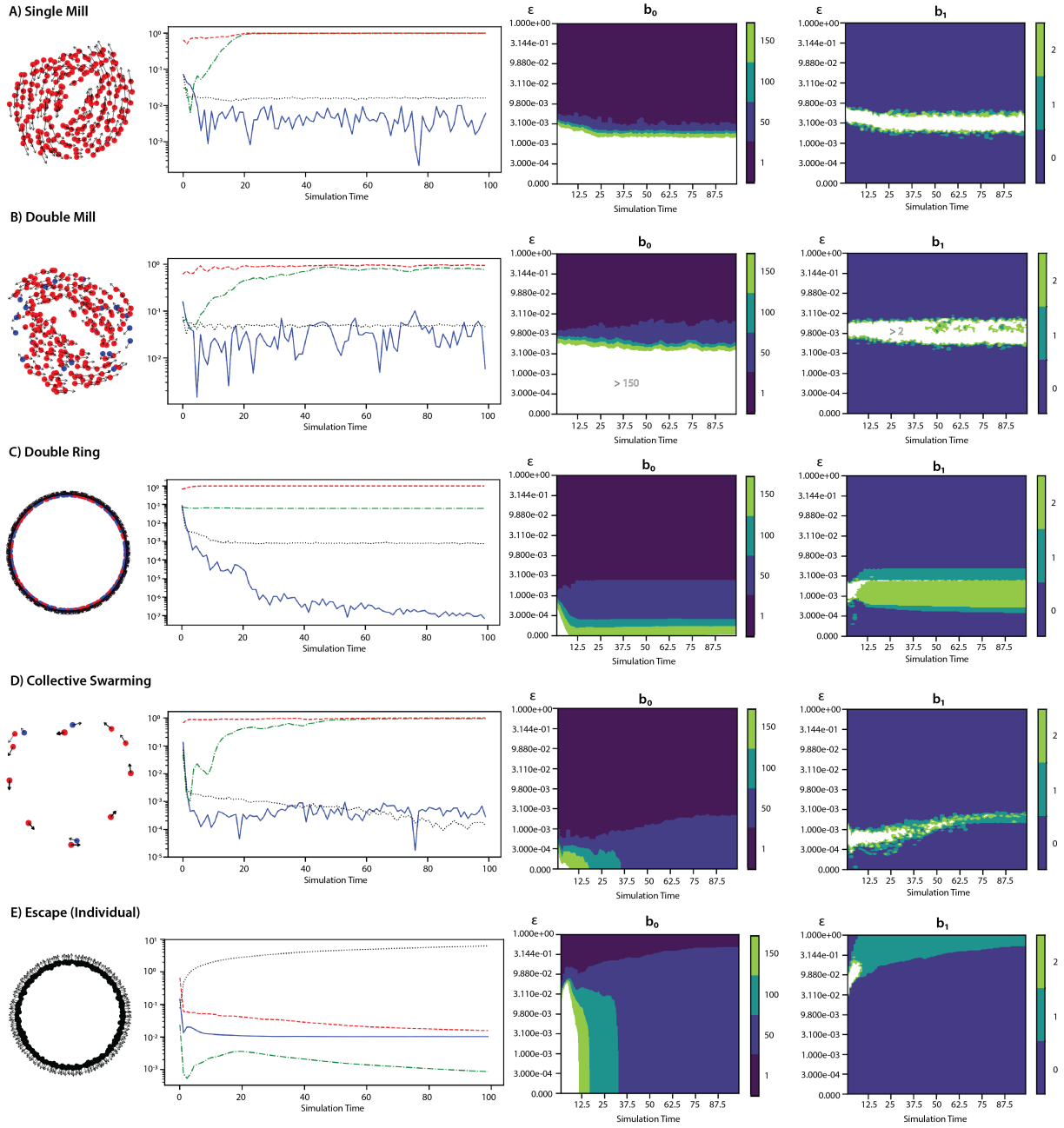

FIG. 1. Collective motion phenotypes generated by the D'Orsogna model (1) with $N = 200$ agents, $\alpha = 1.5$, and $\beta = 0.5$. First column (not to scale): Snapshots of agents' positions (dots) and velocities (arrows) at $t = 100$ with clockwise motion in blue and counter-clockwise motion in red (where applicable). Second column: Order parameter time series, namely polarization $P(t)$ (blue), angular momentum $M_{ang}(t)$ (green), absolute angular momentum $M_{abs}(t)$ (red), and average distance to nearest neighbor $D_{NN}(t)$ (black). Third and fourth columns: crockers showing, respectively, Betti numbers b_0 and b_1 as a function of time t and persistence parameter ϵ (log scale). We include only a small subset of contours for visual clarity. White regions correspond to larger values of Betti numbers, not shown. (A) Single mill, $C = 0.5$, $\ell = 0.1$. (B) Double mill, $C = 0.9$, $\ell = 0.5$. (C) Double ring, $C = 0.1$, $\ell = 0.1$. (D) Collective swarming, $C = 0.1$, $\ell = 0.5$. (E) Escape (individual), $C = 2.0$, $\ell = 0.9$.

In calculating time series of these order parameters, we downsample by a factor of 23 (chosen since it is a divisor of 2001, the original number of simulation frames), resulting in $M = 87$ time points. This downsampling makes subsequent computations feasible, while containing sufficient information about the underlying dynamics.

C. Persistent Homology and Crockers

While order parameters can be useful summaries of collective motion data, they are typically designed in a problem-specific manner and with some knowledge of

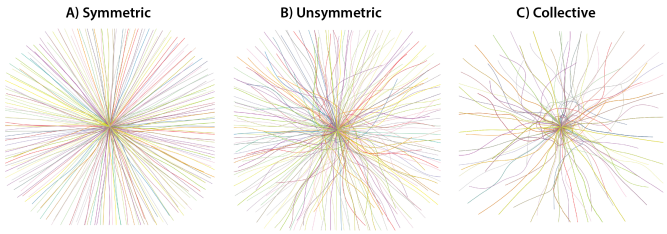

FIG. 2. Three sub-classes of the escape phenotype observed in the D'Orsogna model (1). N, α, and β are as in Fig. 1. Each agent's trajectory is shown in a randomly chosen color for times $t \in [0, 40]$. (A) $(C, \ell) = (2, 0.9)$. Particles escape to infinity individually, and the overall pattern is radially symmetric. (B) $(C, \ell) = (2, 2)$. Agents escape individually, but form gaps in angular distribution. (C) $(C, \ell) = (2, 3)$. Agents superpose in small groups and move outwards with large gaps in angular distribution.

the expected dynamics. We compare an order parameter approach to one that measures the topology of the data. This approach is arguably more agnostic and less application-specific. We now review relevant ideas from topology. To make this review broadly accessible, we keep it conceptual. For some technical details, see, *e.g.*, Refs. 22, 27, and 28.

We begin our explanation of persistent homology and crockers by focusing on agents' positions during one time step of a simulation (or experiment). This data constitutes a *point cloud*, made up of N points in $\mathbb{R}^d$. To study the topology of a point cloud, we transform it into an object called a *simplicial complex*. While there are many ways to construct a simplicial complex, we use the *Vietoris-Rips* (VR) complex, a common choice in TDA because it is efficient to compute.

To build a VR complex, we select a distance ϵ and draw a ball of diameter ϵ around each point. If two balls intersect, we connect them with an edge. If three balls all pairwise intersect, we connect all three edges and fill in the resulting triangle. If four balls all pairwise intersect, we connect all six edges, fill in each of the four triangles bounded by those edges, and fill in the solid tetrahedron bounded by the four triangles. Points, edges, filled triangles, and solid tetrahedra are called *0-, 1-, 2-*, and *3-simplices*, and more generally, k-simplices for any $k+1$ points with ϵ-balls that pairwise intersect.

With a simplicial complex built from our point cloud, we now measure its topology by calculating its *Betti numbers*. Betti numbers b_k are topological invariants, meaning that they are unchanged under continuous deformations of the object such as stretching, compressing, warping, and bending. Thus, they measure something fundamental about the shape of the object. More specifically, Betti numbers enumerate the number of distinct holes in the complex that have a k-dimensional boundary, that is, a hole surrounded by k-simplices. For instance, b_0 counts the number of connected components in the simplicial complex. Similarly, b_1 counts the number of topological loops that bound a 2-D void. Betti number b_2 counts the number of trapped 3-D volumes, and so on as dimension increases. Algebraic topology tells us how to encode the calculation of Betti numbers as a linear algebra problem; see standard topology texts or Ref. 29 for a tutorial. The calculation is a *homology* computation because b_k is the rank of an algebraic object called a *homology group*.

In the discussion above, no value of ϵ was specified. *Persistent homology* constructs simplicial complexes and calculates Betti numbers for a range of ϵ values. There exist powerful software packages that automate this process.[28] We use the `Ripser` package.[30] The outputs of these computations are *birth* and *death* values of ϵ, that is, the values of ϵ for which the various features enumerated by b_k appear and disappear. The word *persistence* refers to the ranges of ϵ over which features persist. For example, features that persist over large ranges of ϵ might be interpreted as signals rather than topological noise. There exist many ways to organize the birth and death information, with the most common being objects called barcodes and persistence diagrams.[28] Additionally, for a given value of k, one could construct a vector in which each entry gives the value of b_k for a specific value of ϵ (say, on a grid). This information, $b_k(\epsilon)$, is a *Betti curve*.

Thus far, we have discussed topological analysis of static point clouds. If we allow our agents' positions to evolve dynamically in time, t, then we can construct a Betti curve for each frame of the simulation or experiment, and concatenate these into a matrix. We let time t vary along columns and ϵ vary along rows. Each entry specifies the Betti number b_k for a specific pair (t, ϵ). The matrix is a topological signature of the time-varying data of a simulation and once vectorized, can serve as input to machine learning algorithms. Equivalently, for visualiation, one could take this matrix and construct a contour plot of $b_k(t, \epsilon)$. Such a plot is the *Contour Realization Of Computed k-dimensional hole Evolution in the Rips complex*, or *crocker*, defined in Ref. 22.

As mentioned in Section II A, our data consist of numerical solutions of (1). While the D'Orsogna model tracks agents' positions and velocities, we restrict ourselves to using position data in our topological analysis. This approach has two advantages. First, it renders our techniques applicable to experimental data, where position is the most easily observed quantity. Second, it circumvents a potential scaling disparity between numerical values of position and velocity when performing TDA on the data, as exhibited in Figure 3. In contrast, three out of four order parameters described above require knowledge of the velocity.

To regain some of the information lost by excluding velocity, we incorporate time-delayed position information into some of our analyses. We demonstrate this time-delay approach for a double ring phenotype in Fig. 3, finding $b_1 = 2$ for a range of ϵ values, as we would expect based on Ref. 22. Our time-delayed point cloud consists

of points in $\mathbb{R}^4$ of the form $(\boldsymbol{x}_i(t_j), \boldsymbol{x}_i(t_j - 5\Delta t))$, where j ranges from 23 to 2001 with a spacing of 23 to result in 86 time samples. Fig. 5 and accompanying text in Appendix A describe the 4-D data for single mills, double mills, and double rings. We will also consider position-only crockers (computed on a point cloud consisting of $(\boldsymbol{x_i(t_j)})$, where the sampling of j is the same as that discussed for the order parameters in Section II B).

Still, challenges remain with the normalization of our topological data. With escape phenotypes, inter-agent distances can approach infinity, whereas they remain bounded for other phenotypes. To circumvent this problem, we take any agent whose distance from the origin crosses the threshold $\|\boldsymbol{x}_i\|_\infty = 10$ and hold it at this fixed position for the remainder of the simulation.

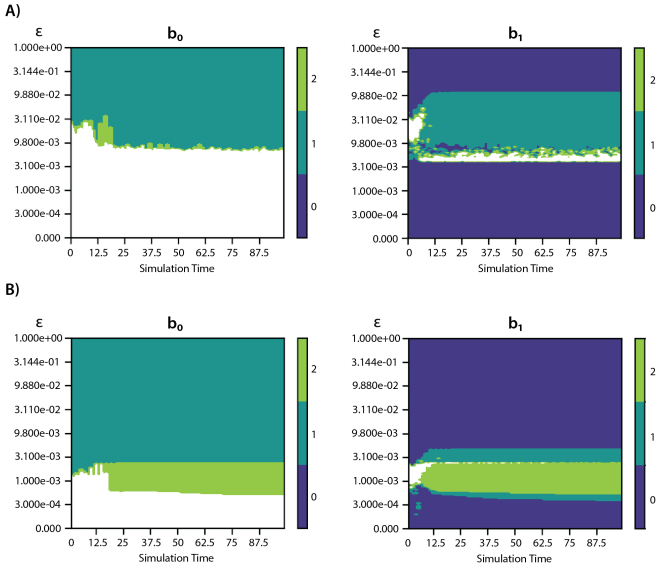

FIG. 3. Crockers for b_0 (left) and b_1 (right) computed on a simulation of the D'Orsogna model (1) with $(C, \ell) = (0.1, 0.1)$. The simulation produces a double ring phenotype consisting of subsets of agents moving clockwise and counterclockwise around a circle. In the crockers, we interpret contour levels greater than two as noise and exclude them. (A) Crockers obtained from position and velocity data. We normalize position and velocity by using their respective maximum and minimum magnitudes across all simulations. This approach identifies only a single topological loop. (B) Crockers computed from 4-D data that incorporates time-delayed position. This approach avoids a disparity between the scales of position and velocity, and correctly detects two topological loops (and two connected components).

Even after enforcing this bound, phenotypes occur on a range of scales. While agent coordinate positions are capped at 10 for group escape, they are as small as 10^{-3} for collective swarms. We normalize position data across all simulations with a global normalization constant to ensure $-1 \leq \|\boldsymbol{x}_i(t)\|_\infty \leq 1$. With this scaling, the smallest normalized phenotypes have typical distances of 10^{-4}. Thus, we compute persistent homology with ϵ varying logarithmically between 10^{-4} and 1 with 200 grid points such that $\epsilon_q = 10^{-4+q\Delta\epsilon}, \Delta\epsilon = 4/200, q = 1, ..., 200$.

The third and fourth columns of Fig. 1 show crockers (b_0 and b_1) for five example simulations. The crockers for single and double mills differ markedly. In (B), the b_1 crocker for the double mill contains small islands corresponding to two loops ($b_1 = 2$) within a large area of topological noise ($b_1 > 2$). In (A), the b_1 crocker for the single mill lacks this signature. In (C), a very strong signature of two loops ($b_1 = 2$) appears for the double ring simulation. Appendix A provides an explanation for the presence of two loops for double mills and rings. In (D), multiple agents form tight clumps in the collective swarm simulation, with each clump sufficiently dense that it appears as a single dot in the figure. The time-scale at which clumps form manifests as the disappearance of high-valued regions in the b_0 contour. On a macroscopic scale, we notice that each clump eventually travels with rotational motion, consistent with $b_1 = 1$ over a range of scales at later times. In (E), agents escape to infinity in a radially expanding circular arrangement. The strong signature of $b_1 = 1$ occurs at larger scales as time increases, consistent with an expanding circle.

D. Unsupervised Learning

We use the k-medoids algorithm to cluster numerical simulations. Each simulation is characterized by a *feature vector* constructed either from traditional order parameters or from topology. The order parameter feature vectors consist of time series $P(t)$, $M_{ang}(t)$, $M_{abs}(t)$, $D_{NN}(t)$, or the concatenation of all four. The topological feature vectors consist of vectorized crocker matrices for b_0, b_1, or the concatenation of the two, calculated from agent position (in some cases, augmented with time-delayed position as described in Section II C).

The k-medoids algorithm divides the ensemble of simulations into k clusters, each of which is defined by one member of the ensemble that serves as the *medoid*. The algorithm chooses medoids to minimize the sum of pairwise distances within each cluster, and each simulation is assigned to the cluster containing its closest medoid. We use the R software function `pam` to cluster our simulations into $k = 25$ groups, since there are 25 distinct parameter choices (C,ℓ). This is an *unsupervised* approach, as it does not require labeled training data.

E. Supervised Learning

As an alternative approach, we use a multiclass linear support vector machine (SVM) to infer parameters. Our use of SVMs is *supervised* because we train them on a subset of our simulations, each labeled with its true (C, ℓ) values. A linear SVM takes this training data and finds hyperplanes that maximally divide the simulations according to parameter values. To classify a simulation not included in the training set, one identifies the intra-

hyperplane region in which it falls and reads off the appropriate label, *i.e.*, the parameter values.

We use Matlab's `fitcecoc` function to build and train our SVMs using a one-versus-one approach and 5-fold cross validation. That is, for each round of cross-validation, we withhold 20% of the data (20 simulations from each parameter combination) for the testing data set. We then train the linear SVM on the remaining data and compute the out-of-sample accuracy for simulations in the testing set.

Order parameter and topological feature vectors are not of the same dimension. The time series of each order parameter is 87-dimensional, and the time series of all four concatenated is $4 \times 87 = 348$-dimensional. On the other hand, each position crocker is $200 \times 87 = 17400$-dimensional, and the concatenation of b_0 and b_1 is double that. To make a fair comparison between the order parameter and topological approaches, we use principal component analysis[31] (PCA) to reduce the dimensionality of our input feature vectors. In one case, we reduce crockers and the concatenated order paramers to 87 dimensions in order to compare them directly to the individual order parameter time series. In a second case, we reduce all feature vectors to three dimensions to investigate performance at low dimensionality.

III. RESULTS

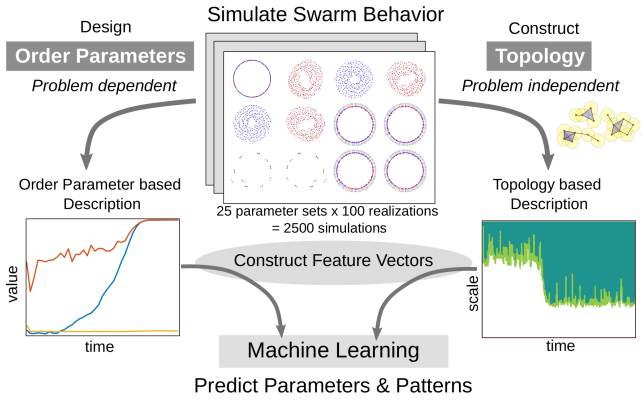

FIG. 4. Our analysis pipeline. We summarize the dynamics of the D'Orsogna model (1) using problem-specific order parameters (left) and a problem-independent description based on topology (right). We construct feature vectors from each summary and input them into machine learning algorithms to identify parameters and phenotype for each model simulation.

Fig. 4 recapitulates the entire analysis pipeline for our study. We summarize the dynamics of (1) using two approaches. The order parameter-based approach uses problem-specific quantities designed to distinguish between observed dynamics. The topology-based approach does not require this *a priori* knowledge and is instead based purely on the shape of the data. We then construct feature vectors from each type of summary and input them into machine learning algorithms to identify parameters and a phenotypic pattern for each model simulation. In the following subsections, we compare how accurately these different approaches can recover simulation parameters and identify dynamic phenotypes.

TABLE II. Unsupervised classification accuracy for parameter values using k-medoid clustering with various input feature vectors and $k = 25$. The third column displays the accuracy when simulations are classified by parameter vector (C, ℓ), and the fourth column displays the accuracy when simulations are classified by phenotype.

Summary	Feature	Parameter	Phenotype
Order Parameters	$P(t)$	20.6%	62.9%
	$M_{ang}(t)$	19.3%	60.7%
	$M_{abs}(t)$	47.3%	63.9%
	$D_{NN}(t)$	48%	80.1%
	All	49.9%	97.4%
TDA (position)	b_0	76.6%	83.6%
	b_1	71.9%	79.8%
	b_0 & b_1	76.6%	99.8%
TDA (time-delayed position)	b_0	71.1%	82.6%
	b_1	73.7%	84.4%
	b_0 & b_1	71.3%	99.0%

A. Unsupervised Learning Results

Table II summarizes results for k-medoids clustering with $k = 25$. Columns 1 and 2 specify the feature vector used, while columns 3 and 4 give the accuracies obtained for parameter recovery and phenotype identification. For the parameter recovery task, using the concatenation of all four order parameters yields 49.9% accuracy. For the concatenation of b_0 & b_1 based only on position data, we obtain 76.6% accuracy. Finally, for b_0 & b_1 based on time-delayed position, we have 71.3% accuracy. In Appendix B, Fig. 6 displays confusion matrices. These reveal that regardless of feature vector type, the collective swarming and single mills are the most frequently misclassified phenotypes.

For all feature vector types, phenotype identification is significantly more accurate than parameter recovery. The confusion matrices reveal that most cases of parameter recovery failure nonetheless place a simulation in its appropriate phenotypic regime. Using topological feature vectors based on position, we classify nearly every simulation correctly by phenotype, and this approach slightly outperforms concatenation of all order parameters.

B. Supervised Learning Results

Table III summarizes supervised classification results for linear SVMs. For topological feature vectors based on position, an SVM for b_0 does best with 97.0% accuracy. Similarly, for delayed positions, b_0 also does best, with

99.6% accuracy. Finally, for order parameter feature vectors, $D_{NN}(t)$ does best with 91.1% accuracy, while concatenating all four order parameters yields an accuracy of 89.2%.

For any feature vectors with dimension greater than 87, we also include results obtained after reducing the dimensionality to 87 via a principal component analysis, allowing for a more fair comparison. After dimension reduction, the time-delayed topological information for b_1 achieves the highest classification accuracy at 99.9%, followed by the position-only topological information for b_0 at 96.2%. The classification accuracy for all four concatenated order parameters drops to 69.6%. Thus, a four-fold reduction in the dimension of the concatenated order parameters results in a 19.6% loss of accuracy, whereas a 200-fold reduction for time-delayed topological information leads merely to a 0.8% drop in accuracy for position-only information and a 0.1% increase for time-delayed topological information. These results suggest that even with dimensionality reduction, the topological feature vectors still carry more discriminative information.

To examine the limit of low-dimensional data, we also calculate accuracies obtained after reducing all feature vectors to three dimensions. In this case, for topological feature vectors based on position data, the concatenation of b_0 and b_1 does best, with an accuracy of 93.1%. For delayed position data, b_0 does best, achieving 87.7% accuracy, and for order parameters, $D_{NN}(t)$ does best, yielding 81.5% accuracy.

In Appendix B, Fig. 7 visualizes the classification results. These results suggest that, similarly to the unsupervised case, collective swarms are the most difficult phenotype to classify. Still, overall, using topological data rather than order parameter data can significantly improve parameter recovery.

Summary	Feature	Dimension	Accuracy
Order Parameters	$P(t)$	87	57.7%
	$M_{ang}(t)$	87	34.4%
	$M_{abs}(t)$	87	68.0%
	$D_{NN}(t)$	87	91.1%
	All	4×87	89.2%
	All (PCA)	87	69.6%
	$P(t)$ (PCA)	3	46.7%
	$M_{ang}(t)$ (PCA)	3	30.0%
	$M_{abs}(t)$ (PCA)	3	58.8%
	$D_{NN}(t)$ (PCA)	3	81.5%
	All (PCA)	3	68.6%
TDA (position)	b_0	200×87	97.0%
	b_1	200×87	93.7%
	b_0 and b_1	2×200×87	96.4%
	b_0 (PCA)	87	96.2%
	b_1 (PCA)	87	95.2%
	b_0 & b_1 (PCA)	87	96.2%
	b_0 (PCA)	3	93.0%
	b_1 (PCA)	3	79.4%
	b_0 & b_1 (PCA)	3	93.1%
TDA (time-delayed position)	b_0	200×86	99.6%
	b_1	200×86	99.3%
	b_0 & b_1	2×200×86	99.1%
	b_0 (PCA)	87	99.7%
	b_1 (PCA)	87	99.9%
	b_0 & b_1 (PCA)	87	99.7%
	b_0 (PCA)	3	89.7%
	b_1 (PCA)	3	82.8%
	b_0 & b_1 (PCA)	3	89.6%

TABLE III. Supervised classification accuracy for parameter recovery using a trained linear SVM with various input feature vectors. In some cases, the dimensionality of feature vectors has been reduced using PCA.

IV. CONCLUSIONS AND DISCUSSION

We have combined mathematical modeling, topological data analysis, and machine learning to study nonlinear dynamics and parameter inference in the D'Orsogna model of collective motion. More specifically, we simulated (1), summarized the data using traditional and topological descriptors, and input these summaries into unsupervised and supervised machine learning algorithms in order to recover model parameters and classify pattern phenotypes.

Our machine learning classifiers achieved higher accuracy when using topological feature vectors (namely, crockers) than when using feature vectors based on traditional order parameters. Since the crocker feature vectors have higher dimensionality than the order parameter ones, we sought a fair comparison by reducing them via PCA. In this case, the crocker approach still achieved better classification accuracy using a supervised approach. In fact, crockers generated from time-delayed position data produced a nearly perfect classification.

One limitation of the topological approach is the computational cost required to produce crockers. While an order parameter is scalar-valued at each time, a crocker is vector-valued. However, recent software improvements, including the development of the `Ripser` package, have led to a significant reduction in cost.

A major advantage of using topological data summaries is that they do not require prior knowledge about the patterns resulting from model simulation. Order parameters, on the other hand, are typically developed to capture specific features of previously observed model behavior. We found that for the D'Orsogna model, topological approaches to phenotype classification and parameter recovery achieved higher accuracy than order parameters even though they do not incorporate knowledge of the model or its dynamics.

In future work, we would like to apply this approach to data from biological experiments or field observations. There is a scarcity of publicly available data describing real biological aggregation dynamics, so for the present, we have demonstrated our method on simulation data. Furthermore, we would like to extend our work to more complex settings, *e.g.*, to the D'Orsogna model posed in three dimensions, in which dynamical transitions occur

between distinct phenotypic regimes[25]. Finally, it would be useful to augment the model with noise and assess its effect by using our topological methods.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant Number DMS 1641020. We wish to acknowledge the American Mathematical Society's Mathematical Research Community which brought our collaboration together and supported our work. LZ is partially supported by NSF grant CDS&E-MSS-1854703. DB is funded by the National Cancer Institute IMAT Program (R21CA212932). JN is partially supported by NSF grant DMS-1638521 to SAMSI. CT is supported by NSF grant DMS-1813752.

Appendix A: Time-Delayed Crockers

In Fig. 5, we examine data used to compute crockers in order to understand differences in b_1 for the single mill, double mill, and double ring phenotypes. Since the data is 4-D, as described in Section II C, we show various 2-D projections. Panel (A) shows a single mill. We see a single loop-like structure that arises from the arrangement of particles in an annulus all traveling with the same orientation. This structure produces $b_1 = 1$ in Fig. 1(A). Panel (B) shows a double mill. Especially in the last two columns, we see a signal of two loops. The two loops correspond to the two counter-rotating swarms of the double mill. However, the signal is quite noisy. This noisy signal manifests as the transient islands of $b_1 = 2$ in Fig. 1(B). Panel (C) shows a double ring. Agents occupy a well-defined circle, with some rotating clockwise and others counterclockwise. This phenotype gives rise to two loops in the 4-D space of our data (see last two columns) and produces a clear signal of $b_1 = 2$ in Fig. 1(C).

Appendix B: Visualization of Machine Learning Results

Fig. 6 displays confusion matrices arising from the unsupervised classification (k-medoids clustering) performed in Section III A. In panel (A), we cluster on the concatenation of all four order parameters. Most of the simulations with misidentified model parameters are those exhibiting single mill behavior or collective swarm behavior. However, for these incorrect cases, most were clustered with cases sharing the same phenotype. In panel (B), we use the concatenation of b_0 and b_1 crockers derived from 2-D position data. In this case, parameter recovery is more accurate for single mill simulations than in panel (A), but collective swarms remain challenging. Panel (C) is similar to (B), but is based on 4-D data incorporating position and time-delayed position. This approach yields results similar to those in (B), with a slight increase in misclassification among the three escape phenotypes.

Fig. 7 depicts the out-of-sample parameter classification results from linear SVMs, as described in Section III B. Note that in this figure, we are depicting the classification of each individual simulation and not binning these classifications together as we do in the confusion matrices of Fig. 6. Because of the high supervised classification accuracies, depicting the individual classifications is more informative than the summary confusion matrix. In panel (A), the linear SVM is trained on feature vectors comprised of $D_{NN}(t)$ without any dimensionality reduction. We observe a high misclassification from parameters $(C, \ell) = (0.1, 0.5)$ and $(0.5, 3.0)$ as each other, as well as simulations from $(C, \ell) = (0.1, 0.9), (0.1, 2.0), (0.1, 3.0)$. All of these parameter choices produce the collective swarm phenotype (see Table I), suggesting that this is the most difficult phenotype for parameter recovery. In panel (B), the linear SVM is trained on feature vectors comprised of the concatenation of b_0 and b_1 crockers derived from 2-D position data with dimensionality reduction down to 87 (to match the $D_{NN}(t)$ dimensionality). Here, we observe a marked reduction in the misclassification of the collective swarm parameter values as compared to Panel (A). In panel (C), the linear SVM is trained on feature vectors comprised of b_0 and b_1 crockers derived from 4-D time-delayed data with dimensionality reduction down to 87. Here, we observe very accurate classifications with only seven simulations being misclassified out of 2500.

[1] S. Camazine, J. L. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau, *Self-Organization in Biological Systems*, Princeton Studies in Complexity (Princeton University Press, Princeton, NJ, 2001).
[2] D. Sumpter, *Collective Animal Behavior* (Princeton University Press, Princeton, NJ, 2010).
[3] I. Giardina, "Collective behavior in animal groups: theoretical models and empirical studies," HFSP journal **2**, 205–219 (2008).
[4] T. Vicsek and A. Zafeiris, "Collective motion," Physics reports **517**, 71–140 (2012).
[5] P. Degond, A. Frouvelle, S. Merino-Aceituno, and A. Trescases, "Quaternions in collective dynamics," Multiscale Modeling & Simulation **16**, 28–77 (2018).
[6] P. Degond, A. Manhart, and H. Yu, "An age-structured continuum model for myxobacteria," Mathematical Models and Methods in Applied Sciences **28**, 1737–1770 (2018).
[7] J. F. Vincent, O. A. Bogatyreva, N. R. Bogatyrev, A. Bowyer, and A.-K. Pahl, "Biomimetics: Its practice and theory," J. Roy. Soc. Interface **3**, 471–482 (2006).
[8] B. Bhushan, "Biomimetics: Lessons from nature – an overview," Phil. Trans. R. Soc. Lond. A **367**, 1445–1486 (2009).
[9] M. R. D'Orsogna, Y. L. Chuang, A. L. Bertozzi, and L. S. Chayes, "Self-propelled particles with soft-core interactions: Patterns, stability, and collapse," Phys. Rev. Lett. **96**, 104302 (2006).
[10] H. Levine, W. J. Rappel, and I. Cohen, "Self-organization in systems of self-propelled particles," Phys. Rev. E **63**, 017101 (2001).
[11] I. D. Couzin, J. Krause, R. James, G. D. Ruxton, and N. R. Franks, "Collective memory and spatial sorting in animal groups," J. Theor. Biol. **218**, 1–11 (2002).
[12] J. K. Parrish and L. Edelstein-Keshet, "Complexity, pattern, and evolutionary trade-offs in animal aggregation," Science **284**, 99–101 (1999).

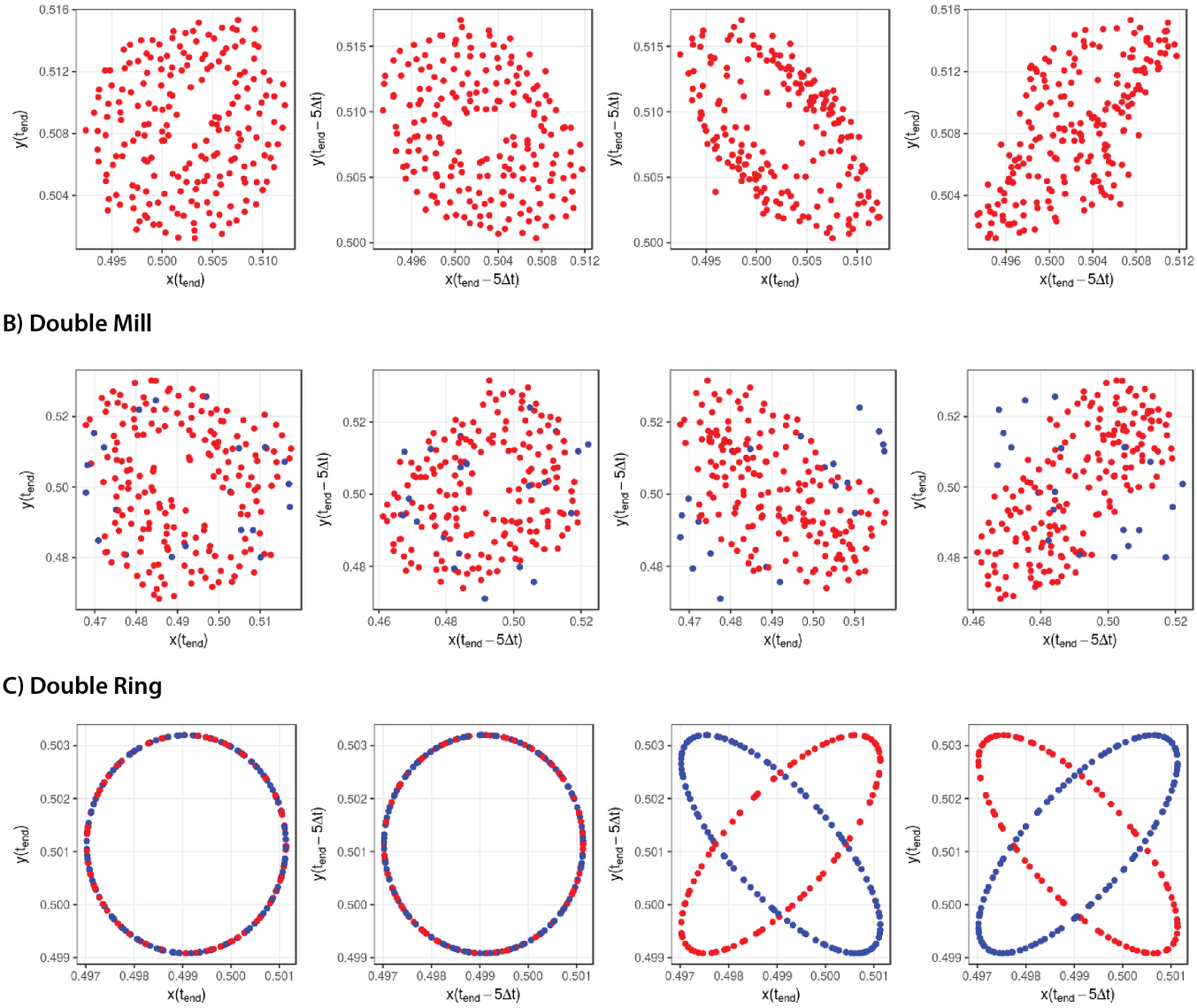

FIG. 5. Snapshots ($t = 100$) of data simulated from the D'Orsogna model (1), used to compute Betti numbers for the phenotypes in Fig. 1(A-C). Because the data is 4-D, as described in Section II C, we show various 2-D projections. Agents rotating clockwise around the group's center of mass are colored in blue, and agents moving counterclockwise are red. (A) Single mill. (B) Double mill. (C) Double ring.

[13]F. Heppner, "Three-dimensional structure and dynamics of bird flocks," in *Animal Groups in Three Dimensions*, edited by J. K. Parrish and W. M. Hamner (Cambridge University Press, Cambridge, UK, 1997) pp. 68–89.
[14]R. Ni and N. Ouellette, "Velocity correlations in laboratory insect swarms," Euro. Phys. J. ST **224**, 3271–3277 (2015).
[15]R. Lukeman, Y.-X. Li, and L. Edelstein-Keshet, "Inferring individual rules from collective behavior," Proceedings of the National Academy of Sciences **107**, 12576–12580 (2010).
[16]A. Manhart, S. Windner, M. Baylies, and A. Mogilner, "Mechanical positioning of multiple nuclei in muscle cells," PLoS computational biology **14**, e1006208 (2018).
[17]R. E. Baker, J.-M. Peña, J. Jayamohan, and A. Jérusalem, "Mechanistic models versus machine learning: A fight worth fighting for the biological community?" Biol. Lett. **14** (2018), http://doi.org/10.1098/rsbl.2017.0660.
[18]L. Wasserman, "Topological data analysis," Ann. Rev. Stat. Appl. **5**, 501–532 (2018).
[19]M. Nicolau, A. J. Levine, and G. Carlsson, "Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival," Proc. Natl. Acad. Sci. , 7265–7270 (2011).
[20]J. L. Nielson, J. Paquette, A. W. Liu, C. F. Guandique, C. A. Tovar, T. Inoue, K.-A. Irvine, J. C. Gensel, J. Kloke, T. C. Petrossian, P. Y. Lum, G. E. Carlsson, G. T. Manley, W. Young, M. S. Beattie, J. C. Bresnahan, and A. R. Ferguson, "Topological data analysis for discovery in preclinical spinal cord injury and traumatic brain injury," Nat. Comm. **6**.
[21]D. Taylor, F. Klimm, H. A. Harrington, M. Kramr, K. Mischaikow, M. A. Porter, and P. J. Mucha, "Topological data

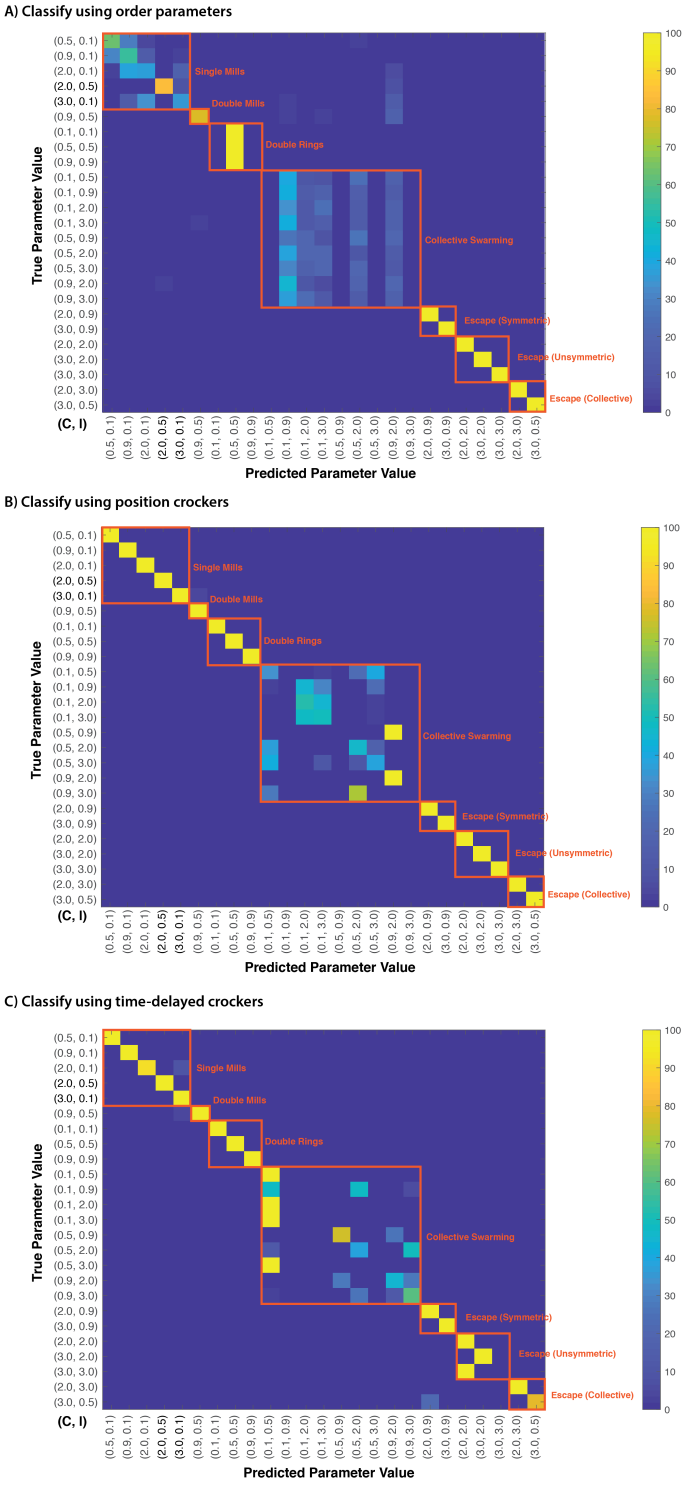

FIG. 6. Confusion matrices for the unsupervised classification of Section III A. Using the k-medoids algorithm, we cluster simulations into $k = 25$ groups. (A) Classification based on the concatenation of four order parameters. (B) Classification using the concatenation of b_0 and b_1 crockers computed from 2-D position data. (C) Like (B), but based on 4-D position data incorporating time delay. Rows correspond to actual parameter values (C,ℓ) and columns correspond to those assigned by the classifier. There are 100 simulations for each true parameter bin. Color indicates the number of simulations for each true/predicted combination.

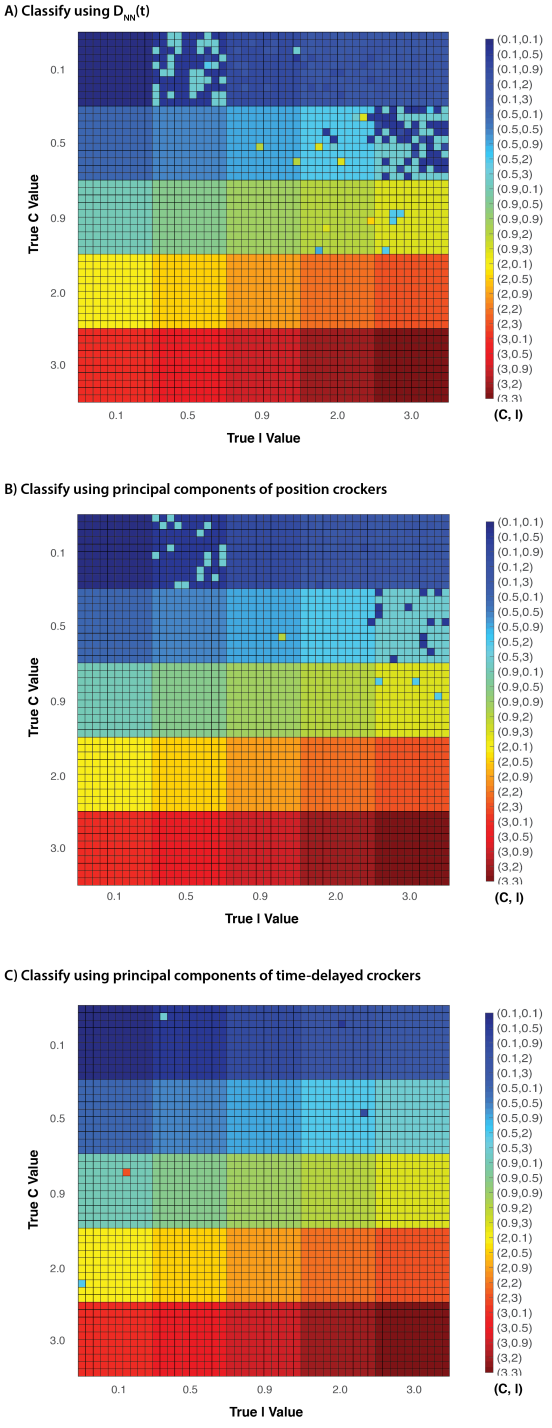

FIG. 7. Results from the supervised classification of Section III B. In each plot, the location of a small square denotes the true underlying parameter vector (C,ℓ) that generates one simulation. For instance, the 10×10 bin in the top left corner represents 100 simulations for $(C, \ell) = (0.1, 0.1)$. The color of each square denotes a linear SVM's out-of-sample classification. For example, a dark blue square designates a classification of $(C, \ell) = (0.1, 0.1)$. (A) Classification based on average distance to nearest neighbor, $D_{NN}(t)$. (B) Classification using the concatenation of b_0 and b_1 crockers computed from 2-D position data and reduced down to 87 dimensions via PCA for fair comparison with (A). (C) Like (B), but based on 4-D position data incorporating time delay.

analysis of contagion maps for examining spreading processes on networks," Nat. Comm. **6**.
[22]C. M. Topaz, L. Ziegelmeier, and T. Halverson, "Topological data analysis of biological aggregation models," PLoS One **10**, e0126383 (2015).
[23]M. Ulmer, L. Ziegelmeier, and C. M. Topaz, "A topological approach to selecting models of biological experiments," PLoS One **14**, 1–18 (2019).
[24]Y. L. Chuang, M. R. D'Orsogna, D. Marthaler, A. L. Bertozzi, and L. S. Chayes, "State transitions and the continuum limit for a 2D interacting, self-propelled particle system," Physica D **232**, 33–47 (2007).
[25]Y.-L. Chuang, T. Chou, and M. R. D'Orsogna, "Swarming in viscous fluids: Three-dimensional patterns in swimmer- and force-induced flows," Phys. Rev. E **93**, 043112 (2016).
[26]C. Huepe and M. Aldana, "New tools for characterizing swarming systems: A comparison of minimal models," Physica A **387**, 2809–2822 (2008).
[27]A. Hatcher, *Algebraic Topology* (Cambridge University Press, 2002).
[28]N. Otter, M. A. Porter, U. Tillmann, P. Grindrod, and H. A. Harrington, "A roadmap for the computation of persistent homology," Euro. Phys. J. Data Sci. **6**, 17 (2017).
[29]C. M. Topaz, "Self-help homology tutorial for the simple(x)-minded," (2015), available at `http://www.chadtopaz.com/publications`.
[30]C. Tralie, N. Saul, and R. Bar-On, "Ripser.py: A lean persistent homology library for python," J. Open Source Softw. **3**, 925 (2018).
[31]I. Jolliffe, *Principal Component Analysis* (Springer Verlag, 1986).
[32]C. Giusti, E. Pastalkova, C. Curto, and V. Itskov, "Clique topology reveals intrinsic geometric structure in neural correlations," **112**, 13455–13460 (2015).