Chaos
An Interdisciplinary Journal of Nonlinear Science

RESEARCH ARTICLE | MARCH 08 2021

Reservoir computing with swarms

Thomas Lymburn ✉; Shannon D. Algar; Michael Small; ... et. al

Check for updates

Chaos 31, 033121 (2021)
https://doi.org/10.1063/5.0039745

View Online Export Citation CrossMark

Articles You May Be Interested In

Boid Based Timetable Generation

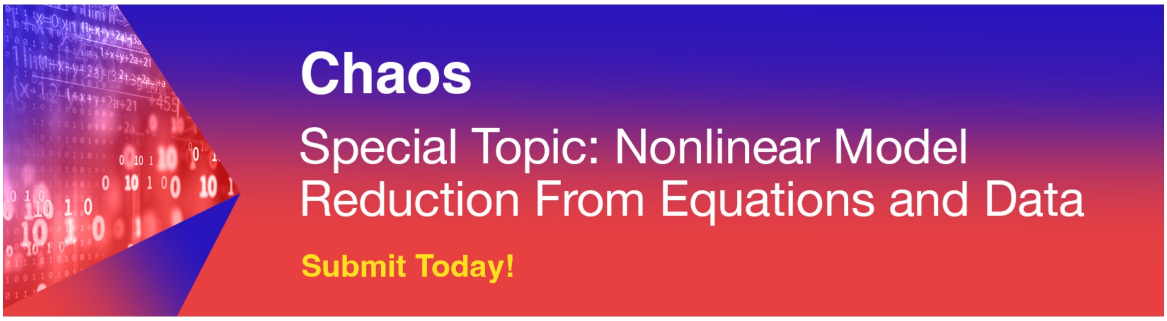
AIP Conference Proceedings (September 2008)

Virtual ethology of aquatic animal heterogeneous behaviours

AIP Conference Proceedings (August 2016)

Learned emergence in selfish collective motion

Chaos (December 2019)



AIP Publishing

Reservoir computing with swarms

Cite as: Chaos **31**, 033121 (2021); doi: 10.1063/5.0039745
Submitted: 6 December 2020 · Accepted: 19 February 2021 ·
Published Online: 8 March 2021

Thomas Lymburn,[1,a)] Shannon D. Algar,[1] Michael Small,[1,2,3] and Thomas Jüngling[1]

AFFILIATIONS

[1] Complex Systems Group, Department of Mathematics and Statistics, The University of Western Australia, Crawley, Western Australia 6009, Australia
[2] ARC Centre for Transforming Maintenance Through Data Science, The University of Western Australia, Crawley, Western Australia 6009, Australia
[3] Mineral Resources, CSIRO, Kensington, Western Australia 6151, Australia

[a)] **Author to whom correspondence should be addressed:** thomas.lymburn@research.uwa.edu.au

ABSTRACT

We study swarms as dynamical systems for reservoir computing (RC). By example of a modified Reynolds boids model, the specific symmetries and dynamical properties of a swarm are explored with respect to a nonlinear time-series prediction task. Specifically, we seek to extract meaningful information about a predator-like driving signal from the swarm's response to that signal. We find that the naïve implementation of a swarm for computation is very inefficient, as permutation symmetry of the individual agents reduces the computational capacity. To circumvent this, we distinguish between the computational substrate of the swarm and a separate observation layer, in which the swarm's response is measured for use in the task. We demonstrate the implementation of a radial basis-localized observation layer for this task. The behavior of the swarm is characterized by order parameters and measures of consistency and related to the performance of the swarm as a reservoir. The relationship between RC performance and swarm behavior demonstrates that optimal computational properties are obtained near a phase transition regime.

Reservoir computing (RC) is the art of utilizing one of a wide range of excitable systems for machine learning. Tasks like time series prediction and spoken-digit recognition have been demonstrated with novel electronic, photonic, or mechanic systems, as well as with numerical RC models, among others. We add biologically inspired computational swarms to this list of reservoir substrates. The common foundation of the diverse field of RC is given by nonlinear dynamical systems that are employed for information processing in an input–output scheme. We consider here a numerical model of a swarm subject to external forcing by a predator that moves on a chaotic trajectory. The swarm is a dynamical system that is generally suitable for RC. We find that the permutation symmetry with regard to identical particles of the swarm prevents the reservoir from functioning well in a simple setup. An additional observation layer enhances functionality when measuring the shape of the swarm rather than individual particle trajectories. We characterize the swarm dynamics with order parameters, as well as with a consistency measure that captures the proportion of signal and noise, and relate these measures to RC performance. We demonstrate that a swarm is capable of performing meaningful prediction of its continuous stimulus—that is, simple computational swarms can act as productive information processing systems. Moreover, we propose the RC paradigm as a means of analysis of swarming systems.

I. INTRODUCTION

In responding to perturbations by an external signal, dynamical systems naturally process the information carried by that signal. Reservoir Computing (RC) is a growing field of study that seeks to utilize this information processing capacity for various forms of functional approximation, time series analysis, and machine learning.[1] With roots in neuroscience and computer science, neural networks are often used to implement the RC framework, as with liquid state machines and echo state networks.[2,3] However, to operate successfully as a substrate for RC a dynamical system need to satisfy only some basic properties, such as the separation property (different inputs result in separable outputs), the approximation property (similar inputs lead to similar outputs), the fading memory property (information about recent inputs is retained), and the consistency property (outputs are a function of the input). This effectively translates to a requirement that the dynamical system must generate a broad range of nonlinear functions that are relevant for a given task.[4] Hence, other classes of high-dimensional systems, such as delay systems,[5,6] cellular automata,[7] and self-organizing systems,[8,9] have also been used as reservoirs. There are also many physical implementations of the RC principle, with potential for fast and efficient computation.[1]

In this work, we demonstrate for the first time how a swarm can act as a reservoir computer. Swarms are complex dynamical systems comprised of many simple agents or active particles, with each individual limited in their ability. The power of the swarm lies in the interaction of these agents as a source of information propagation, which engenders emergent group behavior and swarm intelligence. The latter has previously been used for computation in a variety of contexts, such as optimization and routing.[10] For example, ant colony optimization has been used to "solve" the traveling salesman problem with local incentives and communication via pheromones leading to motion along a path that optimizes the objective function.[11] The RC approach proposed in this work is distinct from previous optimization methods, as instead of searching for a location or path in the space the agents occupy, we measure the response of the swarm to perturbation. We then use the functions generated by that response to approximate a target function.

In these biological and bio-inspired systems, appropriate rules governing the local interactions of the swarm are central to its behavior. The individual will often have autonomy over its movement and the complexity of movement is too great to be described by differential equations. Agent-based models are, therefore, an increasingly utilized framework for implementing the movement of these many-particle systems. Rules that govern the evolution of an individual's state can be built from the bottom up by incorporating non-physical interactions and local knowledge into the movement decisions. The motion of each individual can then be numerically simulated and the resulting group dynamics can be observed. Bird-like objects (boids) that steer their own trajectory in this way provide an efficient method for the generation of realistic group behaviors and greater global controllability. Consequently, such models are ubiquitous in studies of collective behavior, collective motion, and, more recently, collective cognition.

An agent-based model (ABM) is a high-dimensional dynamical system with optional external forcing and thus is a candidate for RC. The concept of ABM embraces a large number of different equations of motion, mainly distinguished by whether the agents (particles) are active or passive. Physical (pseudo-)particles in various settings may also satisfy the requirements for RC and thus present interesting novel RC designs. Nevertheless, in this work, we focus on bio-inspired ABMs. This approach has two complementary aims. First, we propose swarms as candidates for RC by studying this combination on the fundamental level of dynamical systems. Second, the RC framework may serve as a novel way of characterizing and measuring the information flow and signal transformation in swarms. The ABM and RC modeling paradigms have similar and complementary approaches to processing information. Both were integrated for the first time in Ref. 12, where trajectories generated from the ABM provided training data to the RC and the trained output from the RC was then used to update the agent's state in a separate ABM, ultimately demonstrating that the proposed agent-based movement rules were learnable. Here, we take advantage of the high level of transparency in ABM, which allows for a detailed and quantitative study of the relationship between the dynamics of the swarm and its function as a reservoir.

The remainder of this paper is organized into the following sections: an outline of the ABM used for the swarm layer (Sec. II A); how the swarm is employed as a reservoir by introducing an additional observation layer in the RC (Sec. II B); exploration of the swarm's behavior and subsequent influence on its capability as a RC (Sec. III); followed by our concluding remarks.

II. BUILDING A RESERVOIR

A. The swarm model

We use an agent-based model derived from the Reynolds boids model.[13] In this model, each agent (boid) experiences a set of forces corresponding to simple principles of separation, alignment, and cohesion. The actual implementation of the forces can take many forms. In this work, we define the separation and alignment objectives as the result of interaction within the flock via social forces of repulsion and Vicsek-like alignment that are activated within defined distances.[14] Global attraction to a home ensures cohesion without the need for more involved boundary considerations. The repulsion (r), alignment (a), and homing (h) forces on the ith particle with position $\mathbf{x}_i$ and velocity $\mathbf{v}_i$ are

$$F_{ri} = \sum_{j=1}^{N_r} \frac{\mathbf{x}_i - \mathbf{x}_j}{\|\mathbf{x}_i - \mathbf{x}_j\|^2}, \tag{1}$$

$$F_{ai} = \sum_{j=1}^{N_a} \mathbf{v}_j - \mathbf{v}_i, \tag{2}$$

$$F_{hi} = \mathbf{x}_{h_i} - \mathbf{x}_i, \tag{3}$$

where N_r and N_a are the number of neighbors, j, within the repulsion and alignment radii r_r and r_a, respectively, and $||\cdot||$ denotes the Euclidean norm. The home location for each agent is $\mathbf{x}_h$ and we position this at the origin. We also include a friction-like force that aims to achieve an agent speed of $s = 10$ and ensures boundedness and stationarity,

$$F_{f_i} = -\mathbf{v}_i \frac{(\|\mathbf{v}_i\| - s)}{s}. \tag{4}$$

The total force experienced is the sum of all forces,

$$\mathbf{F}_i(t) = K_a F_{ai} + K_r F_{ri} + K_f F_{f_i} + K_h F_{hi}, \tag{5}$$

where the K coefficients determine the strength of each force. This force is then processed by a sigmoidal wrapper, which can be interpreted as the disconnect between an agent's desired motion and their physical capabilities,

$$\mathbf{F}_i(t) \mapsto \alpha \tanh(\beta \mathbf{F}_i(t)). \tag{6}$$

We heuristically choose $\alpha = 200$ and $\beta = 0.1$ to produce reasonable swarm behavior. The qualitative behavior of the swarm does not sensitively depend on these parameter choices but is robust within an order of magnitude. This also has the effect of limiting the extreme responses that cause undesirable behavior.

Assuming a constant mass for each agent, the system evolves according to

$$\mathbf{x}_i(t) = \mathbf{F}_i(t). \tag{7}$$

To implement the evolution, we use an Euler integration to update each point-like particle's position and velocity every Δt,

$$\mathbf{v}_i(t+\Delta t) = \mathbf{v}_i(t) + \mathbf{F}_i(t)\Delta t, \tag{8}$$

$$\mathbf{x}_i(t+\Delta t) = \mathbf{x}_i(t) + \mathbf{v}_i(t)\Delta t. \tag{9}$$

The time increment $\Delta t = 0.02$ was chosen to be sufficiently small to ensure smooth dynamics.

B. The swarm as a reservoir

We seek to use the swarm as the reservoir of an RC scheme. This entails measuring the response of the agents to a signal carrying perturbation and using the measured response to approximate the desired function of that signal. To do so, we need to answer two questions: First, how do we inject the signal into the reservoir? Second, how do we measure the swarm's response to that signal for use in a simple readout mechanism as typically employed in RC?

In other RC structures, the choice of how to input information into the system is often obvious or predetermined. For example, in echo state networks, an input node with its state fixed as the input signal is used. This node connects to the nodes of the reservoir according to the weights on its outgoing connections in a way that is indistinguishable from other nodes. For a photonic reservoir, electrical or optical modulation of the dynamical element, e.g., a semiconductor laser, is determined by physical constraints like bandwidth limitation and accessibility of relevant degrees of freedom.[15–17] In particular, the delay-based architecture, where a number of virtual nodes are derived from a single dynamical element, relies on temporal multiplexing for signal injection.[5] However, when using the swarm as a reservoir, there are a range of methods that may be used. In a physical setting, in which actual particles of some type form the "swarm," one may think of external fields or other mechanical forces acting on these particles. Perhaps, the most natural solution in a bio-inspired setting, and the one which we will use, is to take inspiration from the original biological context of these models and input the signal via a predator which the agents will try to avoid. To input information into the reservoir, the predator's position is fixed by the signal on which we wish to perform computation. As the agents of the swarm flee the predator, their motion will be determined by the location and recent history of the predator, and thus by the signal. We, therefore, include an additional force into Eq. (5) as

$$F_{p_i} = H(r_p - \|\mathbf{x}_i - \mathbf{x}_p\|)\frac{\mathbf{x}_i - \mathbf{x}_p}{\|\mathbf{x}_i - \mathbf{x}_p\|^2}, \tag{10}$$

$$\mathbf{F}_i(t) = K_a F_{ai} + K_r F_{ri} + K_f F_{f_i} + K_h F_{hi} + K_p F_{p_i}, \tag{11}$$

where $\mathbf{x}_p$ is the position of a predator and $H(\cdot)$ is the Heaviside function, which is activated when the predator is within r_p of an agent.

This is not the only biologically plausible way to input information into the swarm. Alternatively, we could flip the scenario and have the agents attracted to the predator, in effect making the agents the predators chasing a single desirable target. Another possible method takes inspiration from soft control of swarms, where one or more "shill" agents do not respond to the other agents, and instead move on a predetermined path.[18] The other agents do not recognize the special nature of the shill agents and respond to them as they would any other agent. Although this method does not involve any special rules for the agents, it is capable of controlling the motion of a swarm, and so may be a suitable method to inject information into the reservoir. Finally, one could add an arbitrary forcing term corresponding to the input signal into the equations of motion of the agents. We leave further discussion of these methods to later research and focus on the predator driving the swarm.

To demonstrate this method, we will fix the path of the predator, $\mathbf{x}_p$, to the x- and y-coordinates of a realization of the Lorenz system with the standard parameter values in the chaotic regime,

$$\begin{aligned} \dot{x}_L &= \sigma(y_L - x_L), \\ \dot{y}_L &= x_L(\rho - z_L) - y_L, \\ \dot{z}_L &= x_L y_L - \beta z_L, \end{aligned} \tag{12}$$

with $\sigma = 10$, $\rho = 28$, and $\beta = \frac{8}{3}$. We rescale the state of the Lorenz system for the predator's position so that the predator's path and the swarm are of similar size. The rescaled coordinates are labeled $\bar{x}_L$, $\bar{y}_L$, and $\bar{z}_L$ and have a standard deviation of 2 and a mean of 0. The predator samples the Lorenz system with a sampling time of 0.02 units. This value was chosen to ensure that the speed of the predator is similar to the speed of the agents in the swarm. If the predator is significantly faster than the agents in the swarm, then the agents do not have enough time to respond to the motion of the predator and instead the predator acts as a source of noise to their autonomous motion. Alternatively, if the predator is significantly slower than the agents, they are able to respond instantaneously to the predator's movement. The result of this is that the drive becomes approximately adiabatic. Neither of these scenarios allow the swarm to respond in a way that is dependent on the current and recent positions of the predator and so are unlikely to lead to good reservoir dynamics. The choices of the scaling factor and sampling time were made heuristically and with little optimization. An example of the swarm's response to this drive is shown in Fig. 1.

The second step of the standard RC approach is to measure the multivariate response of the reservoir to the perturbation by the input. Once the state is observed, a projection is found by linear regression in order to best estimate a target time series, presumably a function of the input time series. In order to do so, one needs to consider how to observe the state of the system for use in linear regression. The distinction between the true and observed reservoir state is subtle, but potentially important in physical RC applications.[19] Here, we will show the importance of examining this distinction by first considering the naïve approach of taking the two position coordinates of the N agents in the swarm and forming a $2N$ node reservoir. We wish to investigate how the swarm's response dynamics can be used for RC and so do not include the predator's position in the reservoir. The state of this reservoir is labeled $\mathbf{r}_p(t)$ and the training is performed using ridge regression with a training, validation, and a testing set.

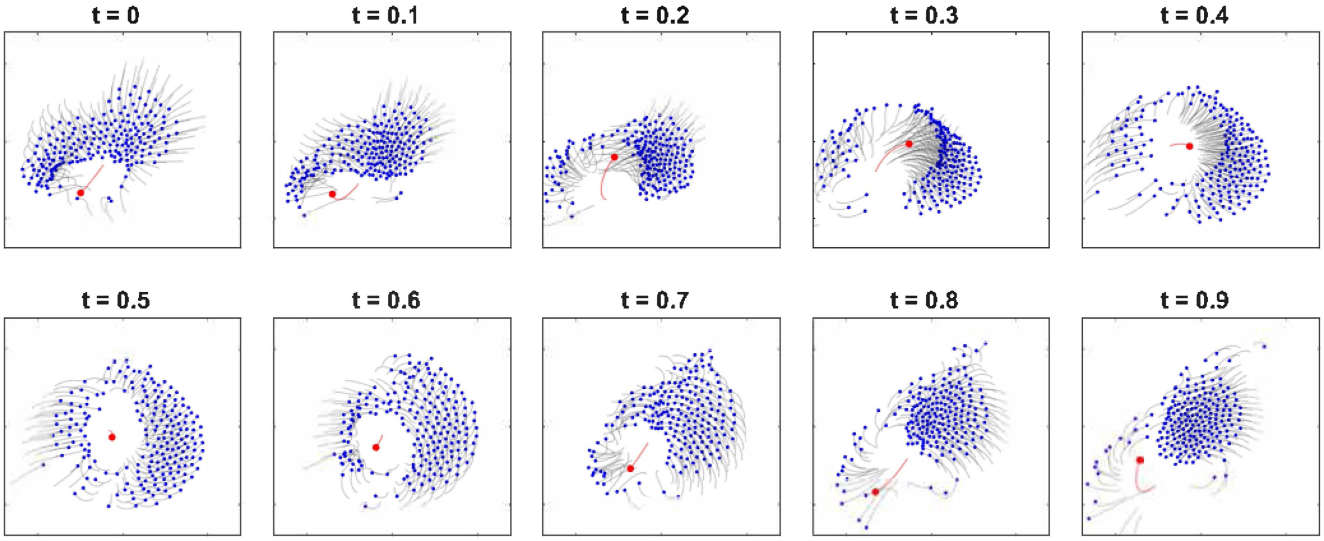


FIG. 1. Typical response of a swarm (blue dots) of $N = 200$ agents to the predator (red dot) with trails showing their previous positions. The parameters of the swarm model are chosen to be $K_r = 1,\ K_a = 0.1,\ K_h = 2,\ K_p = 100$, and $K_f = 20$, with radii $r_r = r_a = 1$ and $r_p = 2$.

To test our reservoir, we will task it with predicting $\bar{x}_L$ with a prediction horizon of 0.5 units, which is approximately half a Lyapunov time. We drive a swarm of $N = 200$ agents, each with identical equations of motion. The weights for the respective forces are as in Fig. 1 ($K_r = 1,\ K_a = 0.1,\ K_h = 2,\ K_p = 100$, and $K_f = 20$, with radii $r_r = r_a = 1$ and $r_p = 2$). Other parameter choices were experimented with and lead to comparable results. A detailed sweep of K_r and K_a is discussed in Sec. III. All other parameters are fixed for the remainder of this work. The swarm's prediction is shown in Fig. 2. Performance is measured by a correlation coefficient between the reservoir's trained output $y(t) = \mathbf{W}_o\mathbf{r}_p(t)$ and the target $z(t) = \bar{x}_L(t + 0.5)$ in the testing set

$$R = \frac{\langle z(t)y(t)\rangle}{\sqrt{\langle (z(t))^2\rangle\langle (y(t))^2\rangle}}, \tag{13}$$

where $\langle\cdot\rangle$ denotes averaging over time. This configuration of the reservoir achieved a correlation coefficient of only 0.19.

This poor performance stems from the choice of the agent's positions as the reservoir coordinates. Since each of the agents has identical equations of motion, there is a symmetry to any permutation of the agents. The effect of symmetry on reservoir computers has previously been investigated in echo state networks.[20] Symmetries in the network were found to decrease the covariance rank of the reservoir's state and thus be detrimental to performance. In this example, we have an extremely symmetrical system, leading to a lack of performance.

The effect of this symmetry in the swarm can be explained as follows. In most parameter regimes studied here, the agents are free to move throughout the swarm and may interchange positions with one another if given enough time. If then a recurrence of the drive happens, i.e., the chaotic trajectory comes arbitrarily close to a state visited earlier, the same drive pattern acts on a swarm with different—including exchanged—particle positions. The two similar stimuli will then lead to significantly different responses, meaning that the swarm trajectory will not be useful for linear readouts. This variability in response to repeated inputs is captured by the concept of *consistency*, which is a measure of the degree of functional dependence of a dynamical system to a driving signal.[21–23] The fluctuations of particle positions in response to the input of the swarm then correspond to a lack of consistency.

In the context of RC, consistency may be thought of as an extension of the echo state property (ESP), which refers to the state of the reservoir being a function of the input signal only. The ESP corresponds to complete consistency, whereas the general case is given by some level of inconsistency which may be understood as noise in the reservoir. The consistency correlation γ^2 measures the *degree* to which the reservoir state is a function of the input signal.[22,24] This is conceptually similar to a signal-to-noise ratio, although with a different measure that is based on pairwise correlation in a replica experiment. The value of the consistency correlation is bounded by $0 \leq \gamma^2 \leq 1$, apart from finite-size effects and given ergodicity, with the upper bound corresponding to the ESP. Consistency of the reservoir's readout determines a fundamental limit on its performance,[22,23] with the correlation between a reservoir's output and the target bounded by γ. A consistency spectrum (or consistency profile) describes the multivariate response of the reservoir by measuring the directional distribution of signal and noise among all degrees of freedom. It consists of a set of correlations $\{\gamma_k^2\}_{k=1}^N$, which measure the consistency along a corresponding set of orthogonal projections of the multivariate state. For details, see the Appendix or Ref. 23.

Figure 2(b) shows the consistency profile of the swarm. The profile is calculated with a method that utilizes the permutation

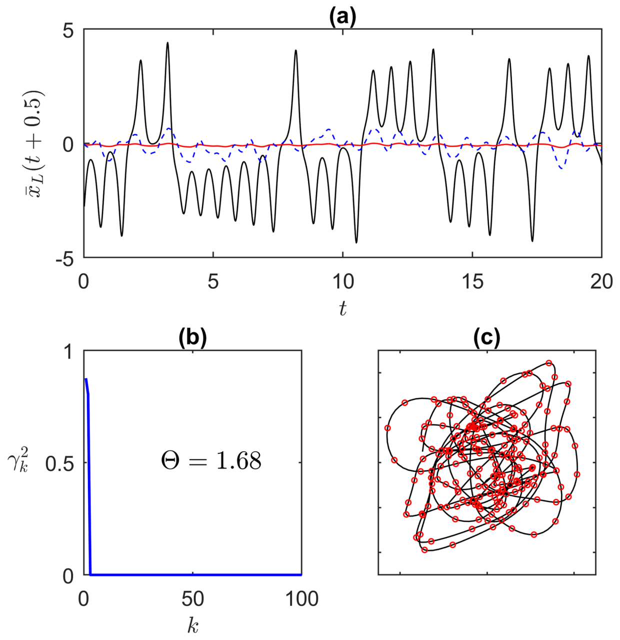


FIG. 2. (a) The *x*-coordinate of the Lorenz system 0.5 units in the future (black). The prediction by the particle position based reservoir is shown in red. The prediction by the center of mass reservoir is shown in blue. Both predictions are extremely poor, and the center of mass reservoir is only slightly less bad. (b) Consistency profile of the reservoir truncated to the largest 100 components. (c) Center of mass of the swarm (black) and the state of the swarm projected onto the two highly consistent components from panel (b) (red dots).

symmetry (see the Appendix, Subsection 2) to reduce finite-size effects. There are two components with a high degree of consistency, whereas the rest of the spectrum is equal to zero. The two non-zero components correspond to the center of mass (CoM) of the swarm, which is the only signal-carrying mode because each particle has an identical equation of motion. In contrast, the deviation of each particle from the CoM trajectory does not have any mutual information with the drive. Figure 2(c) shows the CoM trajectory together with the equivalent projection on the two non-zero consistency components.

The total number of effectively responsive degrees of freedom is given by the consistency capacity,[25]

$$\Theta = \sum_k \gamma_k^2 . \tag{14}$$

In other words, the capacity measures the number of linearly independent functions produced by the reservoir that have a good degree of consistency. The capacity is bounded by the state space dimension (in this case, $\Theta \le 4N$) and may be thought of as the effective size of a reservoir or a soft covariance rank (see the Appendix, Subsection 1). The capacity of this swarm reservoir is $\Theta = 1.68$, meaning that the entire swarm is approximately equivalent to a reservoir of size $N = 2$. In order to support the argument that the swarm reservoir reduces to its CoM, we compare the performance of the full swarm reservoir and one made out of the two CoM coordinates only. The CoM reservoir achieves a correlation of $R = 0.20$, while the particle position reservoir has a correlation of $R = 0.19$.

C. Observation layer

The choice of particle positions as the reservoir coordinates led to symmetry issues due to the identical equations of motion of each agent. These issues are reflected in the consistency profile of the reservoir and significantly restrict the capacity, leading to poor reservoir performance. One way to address this problem is to modify the equations of motion to break the symmetry. For example, we have experimented on a swarm with separate home locations for each agent. This design forces the agents to remain in different locations within the swarm and thus gives rise to a level of diversity that succeeds in enhancing consistency capacity and performance. We have also experimented with breaking the symmetry by assigning each agent a distinct value of K_f. This did not result in a significant increase in performance. Theoretically, any heterogeneity introduced to the equations of motion will remove the limit on consistency capacity. However, this is only a necessary but not sufficient criterion for performance,[25] meaning that the introduction of heterogeneity may affect other properties of the reservoir in an unfavorable way simultaneously to the increase of capacity.

An alternative method to bypass the symmetry collapse without intervening with the swarm is to add an observation layer. The method of observation has previously been used to enhance the nonlinear transformation provided by the reservoir without changing its dynamics.[19] Here, we will aim to bypass the swarm's symmetries by careful choice of how to observe the reservoir's internal state. While the above consistency analysis identified only two responsive components in the swarm, intuitively there appears to be more information in the motion of the swarm as a single entity than what is given by the CoM only. This apparent contradiction is resolved by recalling that consistency measures only indicate the first moment of the time-dependent state distribution in response to the drive.[22,26] The unlabeled particles form a realization of this distribution. We note that the distribution in the spatial domain is a marginal of the full state-space distribution, and we work with the approximation that multiple particles of a single realization are in the spatial domain equivalent to multiple realizations. This would be correct if the particles were not interacting, however, we find that the approximation holds well also for interacting particles.

In order to obtain a better estimate of the distribution, we overlay several realizations of the swarm, i.e., solutions from different initial conditions all receiving the same driving signal, see Fig. 3. Mutual information with the drive is contained in the entire shape of the distribution, offering significantly more components than just the CoM. Consistency does not play a role on this level of consideration, because the leading eigenvalue determining the evolution of the PDF is negative.[26]

We aim to characterize the distribution, and thus extract information about its shape, at individual points in time. One way to do this is by calculating statistical moments (i.e., mean, standard deviation, skewness, curtosis, and higher orders). These moments are tensors with time-dependent elements that reflect the temporal evolution of the distribution. Another way is given by the

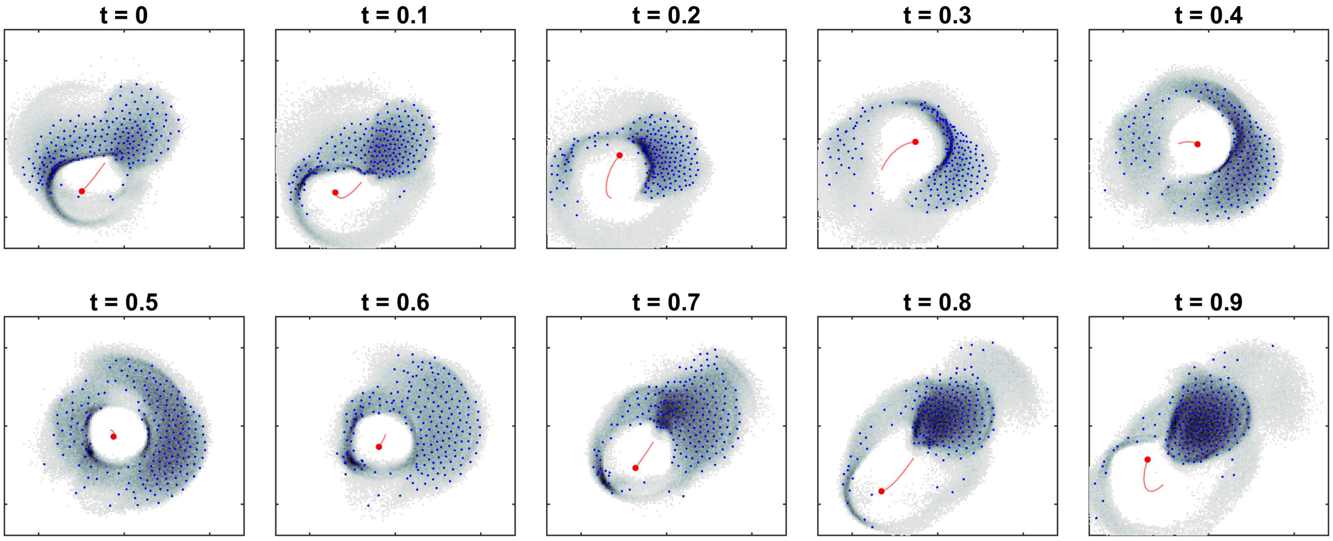


FIG. 3. Underlying time-dependent probability distribution, estimated by overlaying 1000 replicas. The time steps align with those in Fig. 1. The blue dots show one of the replica, which were used to estimate the distribution.

statistical modeling paradigm. If a model is built for each time step from a fixed dictionary of candidate functions, for example, $f(x, y, t) \approx \sum_i \alpha_i(t)\varphi_i(x, y)$, then the time-dependent model coefficients will reflect the shape of the distribution. We will demonstrate the statistical modeling approach to describe the underlying distribution, though we note that experiments with statistical moments were also performed. In general, those reservoirs performed worse than the version we will discuss here, possibly due to higher order moments being needed to accurately describe the shape of the distribution.

From the modeling perspective, we wish to extract information about the underlying distribution based on a single realization of the swarm. For example, one may estimate discretized values of the time-dependent PDF and define these as the observables. Estimates based on binning can lead to discontinuities as agents enter and leave bins. Instead, we choose to build an observation layer based on kernels. Inspired by the radial basis function modeling technique,[27] we place a set of Gaussian observation kernels on the space of the swarm. The kernel functions are defined as

$$\psi_m(\mathbf{x}) = e^{-\frac{(\mathbf{x}-\mathbf{c}_m)^2}{2w_m}}, \tag{15}$$

where $m = 1, 2, \ldots, M$ enumerates the functions and $\mathbf{c}_m$ and w_m determine the location and width of the kernels. Using these kernels, we measure the velocities and positions of the agents in order to obtain the observations

$$r_{1,m}(t) = \sum_{i=1}^{N} \psi_m(x_i(t)), \tag{16}$$

$$r_{2,m}(t) = \sum_{i=1}^{N} \psi_m(x_i(t))\, v_{xi}(t), \tag{17}$$

$$r_{3,m}(t) = \sum_{i=1}^{N} \psi_m(x_i(t))\, v_{yi}(t). \tag{18}$$

The first of these functions effectively counts the number of agents in the receptive field of a kernel, with the soft boundary removing discontinuities. The next two functions return the average velocity of the agents near the center of the kernel. Together, they describe the motion of the swarm at different locations in space. The $3M$ observations are concatenated to obtain the reservoir state, which is forwarded to the linear readout. We label this reservoir state $\mathbf{r}_g(t)$. The entire setup of our reservoir from input to prediction is summarized in Fig. 4.

In order to determine the location and width of an observation function, a random agent and a random point in time are chosen. The location of the agent at that time sets the location of the center of the observation kernel $\mathbf{c}_m$. This choice ensures that there are no observation functions located away from the swarm and that the distribution of observation functions reflects the distribution of the swarm integrated over time. The width of the kernel is set to the distance to the 5th neighbor of the agent used to determine the location of the kernel at that time. This ensures greater resolution where the swarm is denser and gives less importance to outlying agents. We note that the described design of the observation layer contains several choices that do not necessarily optimize reservoir performance. Many other distributions, including regular and purely random distributions, were also experimented with and lead to similar results. An example of our standard kernel design is shown in Fig. 5(c). There is some overlap of the kernel functions which leads to redundancy, in the sense that adding a kernel function that overlaps with the existing ones will not increase reservoir capacity significantly. Decorrelation of the kernel functions through an unsupervised pre-training step may lead to maximally effective observation

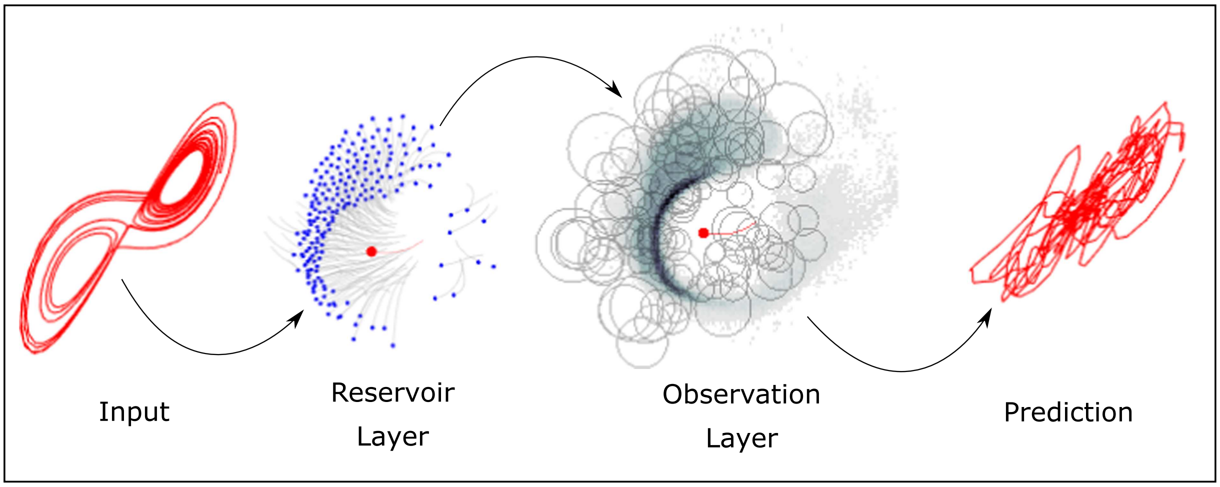


FIG. 4. Schematic of the swarm reservoir. The input signal is injected into the swarm via the trajectory of the predator. In fleeing the predator, the swarm provides a nonlinear transformation of the input signal. Information about this transformation is carried in the shape of the swarm, which is captured by a layer of observation kernels. The measurements on these kernels are combined by a linear readout to synthesize the target, in this case a prediction of the Lorenz system.

layers, however, such technique is out of the scope of this study.

We build a reservoir with $M = 200$ observation kernels from the same swarm trajectory that was used in Fig. 2. The time series prediction task is shown in Fig. 5, achieving a correlation of $R = 0.74$.

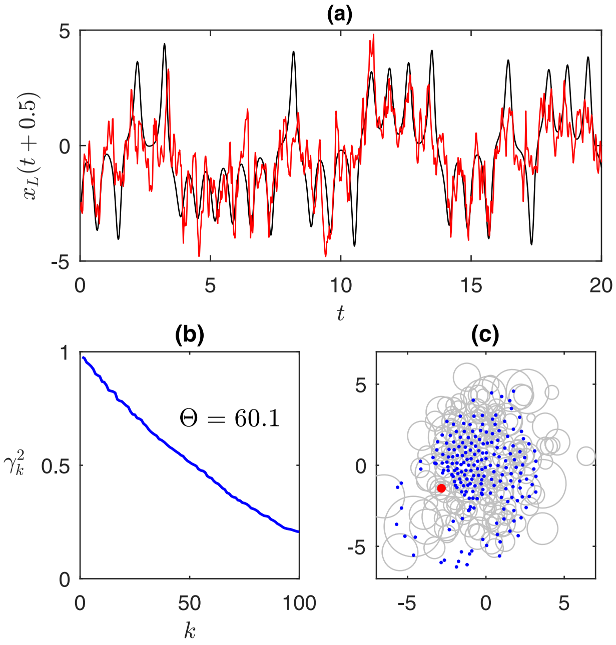


FIG. 5. (a) Prediction of the Lorenz x-coordinate 0.5 units in the future with a reservoir formed by Gaussian kernel observation functions. (b) Consistency profile of the Gaussian kernel reservoir truncated to 100 components. (c) The positions and widths of the observation kernels.

The observation layer was introduced to measure the shape and motion of the swarm as a single entity. As the observation functions are invariant under permutations of the agents, the symmetry issue which previously led to decreased capacity is bypassed. Figure 5(b) shows that the addition of the observation layer on top of the swarm layer has lead to a substantial increase in capacity. While this capacity of $\Theta = 60.1$ may appear relatively small compared to the total number of kernels, this means—recalling that there are only $N = 200$ agents—that on average about three agents create a consistent observable. This is a massive gain with respect to the direct observation in which the response of the swarm effectively collapses to the CoM. In terms of capacity, the swarm with the observation layer is equivalent to an ESN with about 60 nodes and the ESP. Though it is common practice to simply use larger reservoirs, such an ESN is capable of performing simple computation tasks, and we observe that this is also true for the swarm with regard to predicting the Lorenz system [Fig. 5(a)].

III. INFLUENCE OF SWARM BEHAVIOR ON RC CAPABILITY

Basic boid models have been shown to produce a spectrum of behaviors from unorganized swarming to ordered milling and directed motion. The purpose of the following exploration is to understand which types of group behaviors, if any, are most capable of "learning" the predator's movement and therefore performing well as a reservoir.

The group's behavior can be partially categorized into collective modes of motion by analyzing the standard order metrics for polarization and rotation,[28]

$$\Phi_P = \frac{1}{N}\left|\sum_i \frac{\mathbf{v}_i}{|\mathbf{v}_i|}\right|, \tag{19}$$

$\Phi_P \in [0,1]$ with $\Phi_P = 1$ for a perfectly aligned group.

$$\Phi_R = \frac{1}{N}\left|\sum_i \frac{\mathbf{x}_{C_i} \times \mathbf{v}_i}{|\mathbf{x}_{C_i} \times \mathbf{v}_i|}\right|, \tag{20}$$

where $\mathbf{x}_{C_i}$ are centered positions $\mathbf{x}_{C_i} = \mathbf{x}_i(t) - \mathbf{x}_{CoM}$ and hence the numerator is the angular momentum of each particle *i*. We take the absolute value of the normalized angular momentum so that $\Phi_R \in [0,1]$, with $\Phi_R = 1$ indicating a perfectly coherent rotation and $\Phi_R \sim 0$ for non-coherent group rotation as positive and negative contributions (anticlockwise and clockwise rotation, respectively) cancel.

For a simulation of 5×10^4 time steps, we calculate the time average $\langle\Phi\rangle$ after an initial period of 1000 time steps. These metrics alone do not perfectly capture the complexity of the swarm dynamics as they are global metrics for a single, specific type of behavior that is averaged over time, however, for the purpose of this exploration they are sufficient indicators.

A pseudo-logarithmic parameter sweep over the repulsion and alignment strengths, K_r and K_a, was done as these forces were found to be the most influential with regards to both the swarm behavior and the RC performance. Specifically, we test values of $K_r, K_a \in [0.001, 100]$. The swarm is presented with two distinct scenarios: a safe world, free from the predator with $K_p = 0$ and; a "risky" environment where behavior is largely driven by fear of the predator with $K_p = 100$. A range of predator strengths were also investigated, indicating a gradual change between these two extremes (not shown). Other parameters are kept fixed as in Sec. II B. We record both the behavior of the swarm and performance of the RC in each of these scenarios over the entire grid of interest and display this in Figs. 6 and 8.

Without the predator, the swarm demonstrates a rich variety of dynamics as is expected of boid-like models (Fig. 6). When the predator is included, the behavioral state of the swarm naturally adapts. Sharp transitions with minor adjustments to force strengths are replaced by more gradual changes in behavior (Fig. 6). The behavior of the swarm demonstrates a transition from a highly condensed and ordered flock (A), through a dynamically rich "critical" regime (B) into a "disordered" regime (C) as the repulsion force is increased. Typical behavior in these regimes is shown in Fig. 7. Full simulations can be found online (Multimedia view). In connection to physical systems going through a phase change, we view this as a single swarm droplet evaporating into a gas-like state, with the critical behavior occurring at the boiling point. In this analogy, the repulsive force parameter K_r effectively plays the role of temperature in systems of physical particles.

Figure 8 shows the RC performance R across parameter space. We note a strong dependence on K_a and K_r. In particular, reasonable performance is found in a region where $K_a < K_r < K_p$. A qualitative comparison of order and performance in Figs. 6 and 8 suggests that small amounts of order may substantially improve the capability of the reservoir. Conversely, a reduction in performance is likely with

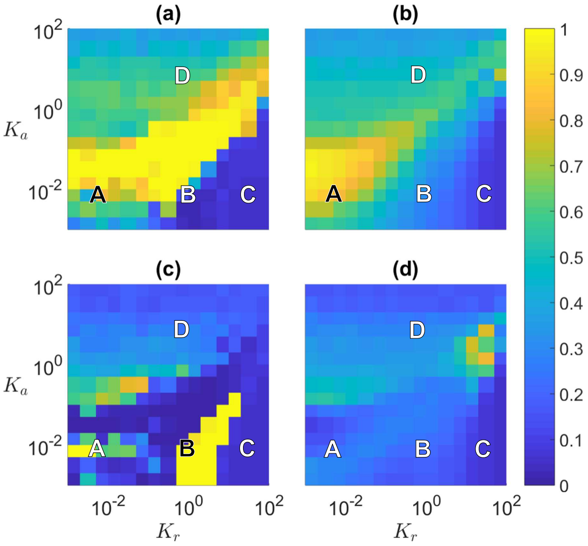


FIG. 6. The time averaged magnitude of the polarization (row 1) and rotation (row 2) order parameters for the case of no predator $K_p = 0$ (column 1) and Lorenz predator with $K_p = 100$ (column 2) are indicated by color as a function of varying repulsive and alignment force strengths plotted on a log scale. Each color point is a single simulation. Labels A, B, C, and D correspond to the behaviors shown in Fig. 7.

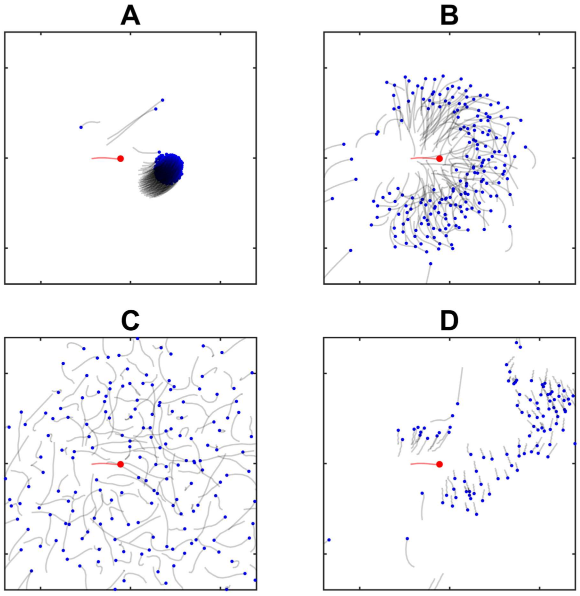


FIG. 7. Typical behaviors of the swarm for different parameters. (a) $K_a = 0.01, K_r = 0.01$, (b) $K_a = 0.01, K_r = 2$, (c) $K_a = 0.01, K_r = 50$, and (d) $K_a = 10, K_r = 1$. Full simulations available online. Multimedia views: https://doi.org/10.1063/5.0039745.1; https://doi.org/10.1063/5.0039745.2; https://doi.org/10.1063/5.0039745.3; https://doi.org/10.1063/5.0039745.4.

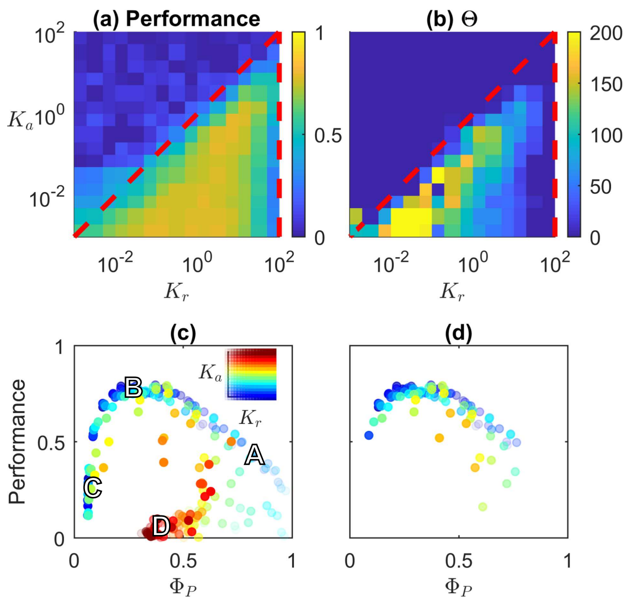


FIG. 8. Points in the (repulsion, alignment) parameter space are mapped to their (a) polarization order and (b) capacity for $K_p = 100$. The red dashed lines indicate $K_r = K_a$ and $K_r = K_p$. (c) Performance against polarization order. Colors indicate the parameters K_a and K_r (see the inset). Labels A, B, C, and D correspond to characteristic behaviors in Fig. 7. (d) Performance against polarization order thresholded such that only reservoirs with $\Theta > 3$ are shown.

order that is so large that groups are rigid and unable to respond diversely.

To better understand the relationship between order and performance, we plot these two metrics directly against each other in Figs. 8(c) and 8(d). We focus only on the polarization order. For sufficient drive strength, K_p, an arc-like structure emerges. This is also visible for other order measures but is not shown here. All subsequent analysis is for polarization only.

The arc is generated by the range of swarm configurations, which is roughly spanned by the three points: (A) high polarization and low performance; (B) medium polarization and high performance; and (C) low polarization and low performance. (A), (B), and (C) here correspond to those used previously describing the condensed (droplet-like), critical ("boiling"), and disordered (gas-like) regimes, respectively. Taking slices through the parameter space for a constant alignment strength, we traverse the arc from right to left as the repulsion strength is increased. The arc structure disappears for unreasonably large alignment force (D), where the K_r dependence collapses to a cluster of intermediate order with low performance.

Regarding the suitability of the swarm as a reservoir in each of these regimes: (A) has poor performance due to a lack of diversity; (B) has good performance due to the swarm's ability to respond to the predator while maintaining cohesive and at least locally ordered motion with "interesting" oscillations and pulsations of the swarm; and (C) again has poor performance, but due to particles moving in a seemingly random fashion with little impact by the modulation from the predator.

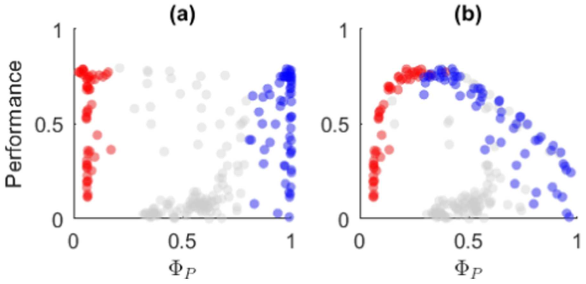


FIG. 9. Polarization order for the case of (a) $K_p = 0$ and (b) Lorenz predator with $K_p = 100$. Performance is determined for both plots by the swarm when the predator is turned on with $K_p = 100$, i.e., the *y*-axis values are the same in both plots. Thresholds of $\langle \Phi_P \rangle < 0.2$ and $\langle \Phi_P \rangle > 0.8$ are shown in red and blue, respectively. The thresholded points in (a) are mapped to their corresponding points in (b).

These observations are further supported by the consistency capacity measure, Eq. (14), see Fig. 9(b). There is good agreement between the areas of high performance and high capacity. In particular, we note that these areas correspond to the aforementioned triangular region $K_a < K_r < K_p$. An intuitive explanation of this domain is given by the competing roles of repulsion and alignment, which tend to decorrelate and correlate the particles, respectively. The principle of RC relies on the reservoir producing a diverse range of functions, so it is important for the decorrelating repulsion force to dominate the alignment. This holds for sections (A) and (B). However, if K_r is increased to the level of K_p [sections (B) and (C)] and beyond, then the repulsion force between particles exceeds the predator force, resulting in decreasing impact by the predator signal. This leads to reduced functional dependency of the swarm on the input signal, and thus lower capacity and poorer performance.

The origin of the arc can be further understood by observing the behavior of the swarms in the absence of a predator. In Fig. 9(a), we plot the polarization order of the swarms without the predator, as well as the eventual performance when the predator is introduced. Interestingly, high performance can come from a swarm that has either high or low polarization originally. We threshold based on the polarization order in the absence of a predator ($\langle \Phi_P \rangle < 0.2$ and $\langle \Phi_P \rangle > 0.8$) and map those points to the corresponding ones in Fig. 9(b), which shows the performance and order with the predator. The high and low polarization regions merge at the top of the arc. This implies that a good reservoir does not necessarily result from a swarm that is intrinsically in a moderate order regime. Rather, it is the swarms that readily adapt their behavior in response to a perturbing signal, which have the best characteristics for reservoir computing.

IV. DISCUSSION AND CONCLUSION

We have presented an analytical and computational study of information processing in complex dynamical systems. Processing here is understood as the nonlinear transformation of signals in a drive response scheme, where a dynamical system is subject to an information-carrying excitation. We have studied this by the example of active-particle swarms within the reservoir computing paradigm. By merging these two concepts, we have laid the foundation for a broad range of developments in biology, machine learning, and other forms of unconventional computation.[29] On the one hand, our design may be extended toward novel forms of physical reservoir computing like using ion traps subject to laser excitation as a medium for ultra-fast signal throughput. This might ultimately lead to hybrids of reservoir and quantum computing.[30] On the other hand, the reservoir computing framework can be interpreted as a form of nonlinear analysis that offers a novel way of understanding the information flow in biological systems. The prediction capacity of the swarm as a reservoir thus may be read as an inherent potential of the whole system to augment the responsiveness of the individual agents to predator attacks. Future studies may aim at the correlation between the RC ability of the swarm and its fitness as measured by other means. Finally, the design of robotic swarms[31] may benefit from the nonlinear dynamics theory of how information propagates and is being transformed in the interaction of active agents.

Within the fundamental mathematical context of dynamical systems as considered here, swarming systems are suitable candidates to operate as a reservoir computer due to their diverse and high-dimensional response to a driving signal. We have implemented a chaotic time series prediction task such that an agent-based model receives the driving signal via the movement of a predator. For indistinguishable particles, the naïve approach of identifying the individual particle coordinates with reservoir variables is ineffective, because the permutation symmetry reduces the responsive degrees of freedom to the center-of-mass coordinate only. An additional observation layer solves this problem by capturing information from the time-dependent distribution of particles. We used randomly placed Gaussian kernels as observation functions to generate the reservoir variables, resulting in a significant improvement in performance. In addition to bypassing the symmetry issues, the observation layer provides robustness to the loss or gain of agents, as the distribution of the swarm remains largely unaffected. This may be an advantage of this method in potential future applications. Were this system be deployed for operational time series prediction tasks, we expect that this kernel structure would need to be optimized. Proceeding with a naïve random placement of kernels, we employed a consistency measure to indicate the computational capacity of the reservoir, showing an effective number of 60 responsive degrees of freedom extracted from a 200-particle swarm.

We studied reservoir computing performance over a range of interaction parameters, as well as the subsequent dynamical regimes of the swarm as characterized by common order parameters. We found that swarms operating in a critical regime of intermediate polarization order were most responsive to the predator and performed best as a reservoir. The observed transition in behavior from fluid to a gas-like state of the swarm corresponds to a transition from low to high and back to low performance. This supports the idea of information processing capability being optimized at criticality.[32] Extensions and more detailed explorations of this work will likely lead to a better understanding of what a swarm is capable of learning and how information propagates within it, ultimately extending our understanding of swarm intelligence in terms of computation, as well as information processing capacity of a dynamical system in general.

ACKNOWLEDGMENTS

We would like to thank Ben Giangiulio for his assistance in performing simulations. T.L. is supported by the Australian Government Research Training Program at The University of Western Australia. S.D.A. is supported by the Australian Research Council Discovery through Grant No. DP200102961. M.S. is partially supported by the Australian Research Council through the Centre for Transforming Maintenance Through Data Science (Grant No. IC180100030) funded by the Australian Government.

APPENDIX: CONSISTENCY ANALYSIS

The analysis of the consistent response of a multivariate system starts with the replica test, an extension of the Abarbanel test for generalized synchronization.[33] Two copies of the system are driven with an identical drive signal but from different initial conditions. These are chosen to be close in order to make sure the two trajectories are on the same attractor, in case there is more than one. The different responses may be labeled $\mathbf{x}(t)$ and $\mathbf{x}'(t)$ and are assumed to have zero mean. We assume sufficiently long time series to neglect any finite-size effects. Each replica may be decomposed into a signal and a noise component $\mathbf{x}(t) = \mathbf{s}(t) + \mathbf{n}(t)$ and $\mathbf{x}'(t) = \mathbf{s}(t) + \mathbf{n}'(t)$. The signal component $\mathbf{s}(t)$ is defined as the ensemble average of an infinite number of replicas, and the noise component is defined as the remainder for each replica. The degree of consistency is then found by comparing the ratio of the signal component to the full response. For the univariate case, this can be found by the ratio of the variances

$$\gamma^2 = \frac{\langle s^2 \rangle}{\langle (s+n)^2 \rangle}, \tag{A1}$$

where γ^2 is termed the consistency correlation.

In the multivariate case, a method analog to principal component analysis (PCA) is performed to decompose the response into a set of characteristic modes with the consistency spectrum $\{\gamma_k^2\}$ as follows. First, the covariance matrix is calculated as

$$[C_{xx}]_{ij} = \langle x_i(t) x_j(t) \rangle. \tag{A2}$$

Eigendecomposition of this positive semi-definite matrix reads $C_{xx} = Q\Sigma^2 Q^\top$. The reservoir states are normalized with the transformation $T_\circ = Q\Sigma^{-1}Q^\top$, which yields coordinates in which the covariance is an identity matrix. To ensure numerical stability, we add a small regularization term $10^{-9} \times \mathbf{I}$ to the covariance matrix prior to calculating $T_\circ$.

In the new coordinates $\mathbf{x}_\circ(t) = T_\circ \mathbf{x}(t)$, we calculate the covariance matrix of the signal component, which in the long-time limit is identical to the cross-covariance matrix of the two replicas

$$[C_{ss}]_{ij} = \langle s_{\circ,i}(t) s_{\circ,j}(t) \rangle = \langle x_{\circ,i}(t) x'_{\circ,j}(t) \rangle. \tag{A3}$$

The eigendecomposition of this positive semi-definite matrix reads $C_{ss} = Q_{ss}\Sigma_{ss}^2 Q_{ss}^\top$. The diagonal entries of Σ_{ss}^2 are the consistency correlations γ_k^2. This spectrum (or profile) tells us the distribution of useful information in a reservoir computer across its degrees of freedom and can be used to explain its performance in the presence of noise or chaos.

1. Capacity

Capacity in general refers to the ability of a driven dynamical system to produce a range of independent functions.[34] The consistency profiles can be used to derive a special capacity measure, which counts the linearly independent functions in the presence of noise or chaos. This measure can be thought of as the effective size of such a reservoir and is predictive of performance.[25]

It can be shown that for a reservoir with a consistency profile given by $\{\gamma_k^2\}_{k=1}^N$ and the correlation between the signal components and some target signal given by a_k, the performance on that task measured by the correlation coefficient is given by $\sum_{k=1}^N \gamma_k^2 a_k^2$. Since the correlations a_k are measures of a reservoir's performance on a specific task, while γ_k^2 are independent of that task, we can define a predictor of performance in general as

$$\Theta = \sum_{k=1}^{N} \gamma_k^2 = \mathrm{Tr}(C_{ss}). \tag{A4}$$

We call Θ the consistent capacity. This measure can be thought of as a modified version of the covariance rank, to which it is identical in the absence of noise or chaos.

2. Consistency with symmetry

Underlying consistency theory is an assumption of infinitely long time series, but of course in reality this is not the case. Finite-size effects can cause errors in calculating consistency, and to account for this one typically uses as long a time series as feasible. However, in high dimensional and highly inconsistent systems, extremely long time series may be required to accurately measure a consistency profile. If possible, it is desirable to use the known structure of the system to improve the efficiency of calculating its consistency profile. An example of a structure that may improve efficiency is when a system has symmetry.

Consider a system like the one described in Sec. II A, where each agent has an identical equation of motion. We will also first consider agents moving in one dimension, then extend to higher dimensions later. We further assume a single global attractor, which means the response of the swarm is ergodic, such that any permutation of the agents results in an equivalent solution with asymptotically identical statistics. A consequence of this permutation symmetry for consistency analysis is that the covariance matrices for each permutation of a given solution must be equal to all others. The effect of swapping two agents on the covariance matrix is to swap the rows and columns corresponding to those agents. Any permutation can be represented as a sequence of such swaps, leading to the swapping of any two off-diagonal elements or any two diagonal elements of the covariance matrices. Therefore, in order for the covariance matrices to be identical before and after the permutation, we require that all off-diagonal elements are equal to each other and the same for the diagonal elements. While this is only true in the limit of infinite trajectories, we can enforce the structure by averaging on the diagonal and off-diagonal elements and thus better approximate the asymptotic behavior.

Analog to the covariance matrix, the symmetry also implies that the cross-covariance matrix C_{ss} has a simplified structure asymptotically. Each element is given by $[C_{ss}]_{ij} = \langle s_{\circ,i}(t) s_{\circ,j}(t)\rangle$, see Eq. (A3). Since each agent has an identical equation of motion, and we assume a single attractor, the functional part of each agent's response must also be the same, meaning that $s_{\circ,i}(t) = s_{\circ,j}(t) = s_\circ(t)$. Thus, all matrix elements are the same, namely, $C_{ss} = \langle (s_\circ(t))^2\rangle \times \mathbf{1}_N$, where $\mathbf{1}_N$ is a $N \times N$ matrix of ones. This structure means that there is a single non-zero eigenvalue of $\lambda = N\langle (s_\circ(t))^2\rangle$, with a corresponding eigenvector composed of all ones. This implies that there is a single consistent component, which aligns with the center of mass. As before, we can enforce the long-time structure from a finite time estimate by averaging the elements of the matrix, this time mixing the diagonal and off-diagonal elements together.

For simplicity, we have focused on one-dimensional systems. The extension to higher-dimensional systems results in a restriction on the allowed permutations only within the corresponding physical dimensions, which enforces a block structure in the covariance matrices. A system of N identical agents in a D-dimensional space will result in a covariance matrix of size ND, consisting of D^2 $N \times N$ blocks. The permutation of agents allows the swapping of elements within each block, but not between. Again, for C_{xx}, the diagonal elements within each block may only be swapped with other diagonal elements within that same block, while for C_{ss} they may mix. For a two-dimensional example, this results in the following structure:

$$\begin{aligned} C_{xx} &= A \otimes \mathbf{I}_N + B \otimes \mathbf{1}_N \\ &= \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \otimes \mathbf{I}_N + \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} \otimes \mathbf{1}_N, \\ C_{ss} &= H \otimes \mathbf{1}_N \\ &= \begin{pmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{pmatrix} \otimes \mathbf{1}_N. \end{aligned} \tag{A5}$$

DATA AVAILABILITY

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

REFERENCES

[1] G. Tanaka, T. Yamane, J. B. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose, "Recent advances in physical reservoir computing: A review," Neural Netw. **115**, 100 (2019).

[2] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," Neural Comput. **14**, 2531 (2002).

[3] H. Jaeger, "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note," GMD Technical Report 148, German National Research Center for Information Technology, 2001.

[4] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," Comput. Sci. Rev. **3**, 127 (2009).

[5] L. Appeltant, M. Soriano, G. Van der Sande, J. Danckaert, S. Massar, J. Dambre, B. Schrauwen, C. Mirasso, and I. Fischer, "Information processing using a single dynamical node as complex system," Nat. Commun. **2**, 468 (2011).

[6] J. Schumacher, H. Toutounji, and G. Pipa, "An introduction to delay-coupled reservoir computing," in *Artificial Neural Networks*, edited by P. Koprinkova-Hristova, V. Mladenov, and N. K. Kasabov (Springer, 2015), pp. 63–90.

[7] Ö. Yılmaz, "Machine learning using cellular automata based feature expansion and reservoir computing," J. Cell. Autom. **10**, 435 (2015).

[8] M. Dale, J. F. Miller, S. Stepney, and M. A. Trefzer, "Evolving carbon nanotube reservoir computers," in *Unconventional Computation and Natural Computation*, edited by M. Amos and A. Condon (Springer, 2016), pp. 49–61.

[9] A. Z. Stieg, A. V. Avizienis, H. O. Sillin, C. Martin-Olmos, M. Aono, and J. K. Gimzewski, "Emergent criticality in complex turing b-type atomic switch networks," Adv. Mater. **24**, 286 (2012).

[10] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-International Conference on Neural Networks* (IEEE, 1995), Vol. 4, pp. 1942–1948.

[11] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," IEEE Trans. Evol. Comput. **1**, 53 (1997).

[12] S. D. Algar, T. Lymburn, T. Stemler, M. Small, and T. Jüngling, "Learned emergence in selfish collective motion," Chaos **29**, 123101 (2019).

[13] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques* (ACM SIGGRAPH, 1987), pp. 25–34.

[14] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet, "Novel type of phase transition in a system of self-driven particles," Phys. Rev. Lett. **75**, 1226 (1995).

[15] L. Larger, M. Soriano, D. Brunner, L. Appeltant, J. Gutierrez, L. Pesquera, C. Mirasso, and I. Fischer, "Photonic information processing beyond turing: An optoelectronic implementation of reservoir computing," Opt. Express **20**, 3241 (2012).

[16] D. Brunner, M. C. Soriano, C. R. Mirasso, and I. Fischer, "Parallel photonic information processing at gigabyte per second data rates using transient states," Nat. Commun. **4**, 1364 (2013).

[17] L. Larger, A. Baylón-Fuentes, R. Martinenghi, V. S. Udaltsov, Y. K. Chembo, and M. Jacquot, "High-speed photonic reservoir computing using a time-delay-based architecture: Million words per second classification," Phys. Rev. X **7**, 011015 (2017).

[18] J. Han, M. Li, and L. Guo, "Soft control on collective behavior of a group of autonomous agents by a shill agent," J. Syst. Sci. Complex. **19**, 54 (2006).

[19] K. Vandoorne, P. Mechet, T. Van Vaerenbergh, M. Fiers, G. Morthier, D. Verstraeten, B. Schrauwen, J. Dambre, and P. Bienstman, "Experimental demonstration of reservoir computing on a silicon photonics chip," Nat. Commun. **5**, 1 (2014).

[20] T. L. Carroll and L. M. Pecora, "Network structure effects in reservoir computers," Chaos **29**, 083130 (2019).

[21] A. Uchida, R. McAllister, and R. Roy, "Consistency of nonlinear system response to complex drive signals," Phys. Rev. Lett. **93**, 244102 (2004).

[22] T. Jüngling, M. C. Soriano, N. Oliver, X. Porte, and I. Fischer, "Consistency properties of chaotic systems driven by time-delayed feedback," Phys. Rev. E **97**, 042202 (2018).

[23] T. Lymburn, A. Khor, T. Stemler, D. C. Corrêa, M. Small, and T. Jüngling, "Consistency in echo-state networks," Chaos **29**, 023118 (2019).

[24] N. Oliver, T. Jüngling, and I. Fischer, "Consistency properties of a chaotic semiconductor laser driven by optical feedback," Phys. Rev. Lett. **114**, 123902 (2015).

[25] T. Lymburn, T. Jüngling, and M. Small, "Quantifying robustness and capacity of reservoir computers with consistency profiles," in *International Conference on Artificial Neural Networks* (Springer, 2020), pp. 447–458.

[26] G. Giacomelli, S. Barland, M. Giudici, and A. Politi, "Characterizing the response of chaotic systems," Phys. Rev. Lett. **104**, 194101 (2010).

[27] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," Complex Syst. **2**, 321 (1988).

[28] A. Attanasi, A. Cavagna, L. Del Castello, I. Giardina, S. Melillo, L. Parisi, O. Pohl, B. Rossaro, E. Shen, E. Silvestri, and M. Viale, "Collective behaviour without collective order in wild swarms of midges," PLoS Comput. Biol. **10**, 1 (2014).

[29] *Unconventional Computing*, Encyclopedia of Complexity and Systems Science, 2nd ed., edited by A. Adamatzky (Springer, 2018).

[30] K. Fujii and K. Nakajima, "Harnessing disordered-ensemble quantum dynamics for machine learning," Phys. Rev. Appl. **8**, 024030 (2017).

[31] H. Hamann, *Swarm Robotics: A Formal Approach* (Springer, 2018).

[32] C. G. Langton, "Computation at the edge of chaos: Phase transitions and emergent computation," Physica D **42**, 12 (1990).

[33] H. D. I. Abarbanel, N. F. Rulkov, and M. M. Sushchik, "Generalized synchronization of chaos: The auxiliary system approach," Phys. Rev. E **53**, 4528 (1996).

[34] J. Dambre, D. Verstraeten, B. Schrauwen, and S. Massar, "Information processing capacity of dynamical systems," Sci. Rep. **2**, 514 (2012).