SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

November 2026, Vol. 69, Iss. 11, 210205:1–210205:19
https://doi.org/10.1007/s11432-026-5236-2

Special Topic: Intelligent Characterization and Modeling for Complex Systems

Physics-informed learning for collective behavior prediction

Zifan SHI[1], Qizi HUANGPENG[1], Jingjie LIANG[1], Mingze QI[1,2*], Xiaojun DUAN[1] & Gang YAN[2]

[1]*College of Science, National University of Defense Technology, Changsha 410073, China*
[2]*School of Physical Science and Engineering, and Center for Frontier Interdisciplinary Research in Fundamental Sciences, Tongji University, Shanghai 200092, China*

Received 22 April 2026/Revised 9 October 2026/Accepted 23 October 2026/Published online 9 November 2026

Abstract Collective behaviors, such as those observed in bird flocks and fish schools, exhibit highly coordinated, self-organized motion. Predicting their trajectories is crucial for understanding collective intelligence and guiding unmanned systems. Existing methods face limitations, as traditional mechanism models rely on oversimplified assumptions and generalize poorly, while purely data-driven models lack interpretability and struggle to adapt to unseen behaviors. Here, we propose a physics-informed decomposition linear neural network (PI-DLinear). By treating the physical parameters in the classic Cucker-Smale model as learnable variables, our framework implements a dynamic physics-constraining module that adapts to underlying interaction mechanisms across diverse collective behaviors. Evaluated on multiple real-world biological datasets (including jackdaw and fish), PI-DLinear significantly improves prediction accuracy within each collective behavior and consistently enhances zero-shot cross-behavior transfer relative to the architecture-matched DLinear baseline, while also demonstrating robustness in small-sample and noisy conditions. By integrating interpretable physical equations with neural networks, our work establishes a paradigm for trajectory prediction that combines interpretability with data-driven flexibility, with promising applications in biological behavior analysis and multi-agent cooperative control.

Keywords collective behavior, swarm, trajectory prediction, decomposition neural network, physics-informed neural network, time-series forecasting

Citation Shi Z F, Huangpeng Q Z, Liang J J, et al. Physics-informed learning for collective behavior prediction. Sci China Inf Sci, 2026, 69(11): 210205, https://doi.org/10.1007/s11432-026-5236-2

1 Introduction

Collective behavior, the highly coordinated movements exhibited by organisms ranging from microbial colonies to flocks of birds and fish, is one of the most compelling self-organizing phenomena in nature [1]. Such behavior not only demonstrates the efficiency and robustness of biological systems in information transmission and group decision-making [2], but also provides key biomimetic design principles for artificial swarm systems, such as multi-robot coordination and drone swarm control [3]. Among various biological swarms, bird and fish flocks [4] stand out as typical examples for studying self-organizing movements due to their highly structured collective behavior patterns. In recent years, with the development of high-precision trajectory tracking technology, researchers have been able to quantitatively reveal the fine spatiotemporal dynamics in swarm movements. Studies on starlings [5,6], pigeons [7], jackdaws [8,9], and chimney swifts [10] have proposed topological, leader-based, or plastic interaction models in swarm models to explain different movement patterns in collective behavior. Studies on Danio rerio and Hemigrammus rhodostomus [11] have revealed time-coupled, stress-induced collective state transitions. Different types of collective behavior data have facilitated researchers' understanding of flocking systems, discovery of collective emergence mechanisms, and modeling based on first principles.

Inspired by real bird flock data, researchers have proposed many mechanism models depicting flocking behavior and statistical physics analysis methods have been proposed to explain the underlying causes of flocking behavior [12–14]. The classic Reynolds-Boids model proposes three basic rules: separation, alignment, and cohesion [15]. The Couzin-Boid model simulates group dynamics by dividing the three-layer interaction zones around individuals and combining different rules [16]. The Vicsek model reveals phase transitions in group movement by simplifying alignment behavior [17]. The swarmalator model establishes a bidirectional coupling between spatial position and

* Corresponding author (email: qimingze17@nudt.edu.cn)

© Science China Press 2026

phase to describe chemotactic oscillators [18]. The Chiral active particle model (CAP) is used to describe the behavior of oscillators with chiral properties [19]. Among numerous swarm dynamics models, the Cucker-Smale model (abbreviated as C-S model) has garnered significant attention due to its rigorous mathematical convergence analysis. This model guarantees asymptotic velocity alignment through a distance-dependent communication weight function [20]. Recent studies have increasingly incorporated real-world complexities, such as improved C-S models with time delays and heterogeneity [21]. The improved C-S model, featuring anti-collision and goal formation driving forces, can precisely control the final spatial distribution of the flock, achieving preset geometric patterns [22]. Such models are capable of simulating typical macroscopic phenomena such as flock cohesion, migration, and turning, and possess a certain degree of interpretability. However, most of them are built upon strong simplifying assumptions, such as fixed perception radii, linear force forms, or ideal communication conditions. These assumptions make it difficult to capture the dynamically adjusting interaction strategies employed by biological swarms in complex environments.

In recent years, the development of artificial intelligence methods has provided tools for directly analyzing swarm data [3, 23, 24], enabling the prediction and control of swarm behavior [25]. Specifically, deep learning methods are utilized to learn the dynamics of swarm movement from a large amount of observational data. Collaborative graph neural networks predict the future trajectories of multiple individuals in a swarm by dynamically learning the interactive graph structure [26]. The recurrent neural network (RNN)-based ReactNet method can predict information transmission and reactive behavior in animal swarms [27]. The feedback spatio-temporal graph neural network (FST-GNN) model can predict the formation behavior of networked multi-agent systems under noise interference [28]. Transformer, through its self-attention mechanism, captures long-range temporal dependencies and is used to predict phase transitions and collective motion patterns in active substances [29]. Beyond biological swarms, deep learning frameworks that account for individual driving styles and vehicle interactions have also substantially improved trajectory prediction accuracy in autonomous driving scenarios [30]. These data-driven methods provide effective data analysis tools, but such black box methods lack interpretability in their predictive performance, and there is no direct evidence to indicate that they have captured the underlying behavioral patterns of clusters [11]. Although symbolic regression approaches, such as the sparse identification of nonlinear dynamical systems (SINDy) for discovering control equations from data [31], the two-stage autonomous inference method for complex network dynamics [32], and the LaGNA method based on Langevin graph neural networks [33], have demonstrated promise in uncovering control laws from measurements, they primarily concentrate on inferring general dynamic equations rather than integrating existing mechanistic ensemble models for prediction. Consequently, the question of how to effectively leverage mechanistic models and modern artificial intelligence methods for the data-driven modeling and prediction of swarm behavior remains a key challenge that urgently needs to be addressed.

Recently, physics-informed neural networks (PINNs) [34–36], as a method that integrates mechanisms and data, have garnered widespread attention in engineering and scientific computing [33]. By embedding governing equations as soft constraints in the loss function, PINNs enforce physically consistent outputs during training [37, 38]. This method provides an effective approach for integrating the aforementioned swarm mechanism model with neural network algorithms. In this paper, we embed the swarm mechanism model into the neural network framework to achieve higher accuracy in collective behavior prediction, while also explaining the underlying interaction mechanisms. The main contributions of this work are as follows. (1) We propose a physics-informed decomposition linear neural network method (PI-DLinear), designed with differentiable physics modules that embed the classical Cucker-Smale model with learnable parameters, transforming physical constraints into learnable dynamic physical kernels to automatically identify and adjust the underlying physical rules behind different behaviors from data. (2) We verify the effectiveness and robustness of the proposed method. Through comprehensive ablation studies and noise injection experiments, we demonstrate that the embedded physical prior knowledge enhances the model's generalization ability under small sample conditions and its robustness to measurement uncertainty. Beyond within-dataset prediction, we directly evaluate zero-shot transfer across three real collective behaviors and provide controlled tests on unseen swarm sizes and agent-index permutations. (3) We apply this method to natural swarm phenomena, validating the proposed method on three distinct types of real-world behavior data, including anti-predation and transit behaviors of bird flocks, as well as circling behaviors of fish schools. Experiments show that the PI-DLinear model achieves superior prediction accuracy.

2 Problem description and modeling

The collective motion prediction problem could be formalized as a multivariate time-series forecasting task. We consider a swarm of N agents moving in d-dimensional space ($d = 3$ for the jackdaw datasets and $d = 2$ for

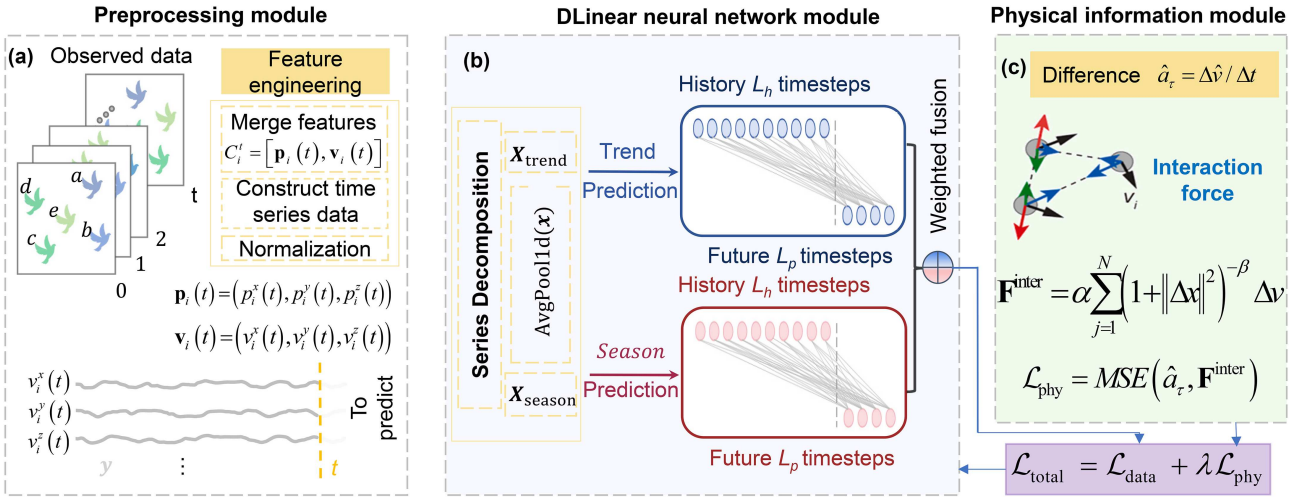

Figure 1 (Color online) Overview of the proposed physics informed decomposition linear neural network framework (PI-DLinear). It includes the input (a) preprocessing module, (b) DLinear neural network module, and (c) physical information module.

the fish dataset). The state of each agent i at time t is described by its position $\mathbf{p}_i(t)$ and velocity $\mathbf{v}_i(t)$. The historical trajectory of the swarm over a fixed-length time window is defined as $X^{(j)}$. The goal is to predict the future state $Y^{(j)}$ over a prediction horizon of L_p steps. To achieve this, we propose a PI-DLinear framework by integrating the physical rules of cluster convergence from the Cucker-Smale model into an effective time series prediction algorithm, DLinear. This module serves as a soft constraint, guiding the model to produce predictions that are not only accurate but also consistent with the underlying collective motion dynamics.

As illustrated in Figure 1, this framework integrates neural networks with physical information, offering good interpretability, prediction accuracy, and noise robustness. The architecture consists of three core components: preprocessing module, DLinear neural network module, and physical information module.

First, the input preprocessing module (Figure 1(a)) is responsible for receiving and processing raw observational data. The motion state of each agent in the swarm is described by its d-dimensional position and d-dimensional velocity, together forming a $2d$-dimensional state vector. The system organizes state sequences from multiple consecutive time steps into a sliding-window format, resulting in a multivariate time-series tensor. Standardization is then applied to eliminate scale differences, and the data are divided into training, validation, and test sets to ensure objective model evaluation.

The DLinear neural network module (Figure 1(b)), inspired by the decomposition linear neural network architecture [39], employs a seasonal-trend decomposition mechanism to separately process each agent's time series. The trend component captures long-term motion patterns (e.g., sustained acceleration), while the seasonal component extracts periodic and short-term fluctuations (e.g., oscillatory interactions). These components are independently predicted via linear projection networks and then summed to produce a purely data-driven baseline prediction. This decoupled design enhances the model's expressivity for complex motion patterns.

The physical information module (Figure 1(c)) is the core innovation of this framework. Based on the classic physical model of swarm intelligence, namely the Cucker-Smale model, this module calculates a physics-informed loss function

$$\mathcal{L}_{\text{phy}} = \frac{1}{L_p}\sum_{\tau=1}^{L_p}\left\|\hat{\mathbf{a}}_\tau - \mathbf{F}^{\text{inter}}\right\|^2. \tag{1}$$

All physical parameters, along with network weights, are jointly optimized during training, enabling the model to adaptively adjust interaction forces. By combining the data prediction results with dynamical constraints from the physical differential equations, a hybrid loss function is constructed,

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{data}} + \lambda\mathcal{L}_{\text{phy}}, \tag{2}$$

where $\mathcal{L}_{\text{data}}$ measures the mean squared error between predicted and ground-truth trajectories; $\mathcal{L}_{\text{phy}}$ computes the difference between predicted acceleration and acceleration derived from the physical model; and λ is an adaptive weight, computed at each training iteration by gradient-norm balancing rather than by gradient descent (its closed form is given in Subsection 2.3), balancing data fitting and physical consistency. Through backpropagation, both neural network weights and physical parameters are optimized simultaneously, allowing the model to gradually

Table 1 Dimensional correspondence of the tensors in PI-DLinear.

Symbol	Shape	Meaning
N	Scalar	Number of agents (dataset-dependent)
d	Scalar	Spatial dimension ($d=3$ jackdaw; $d=2$ fish)
L_h/L_p	Scalar	Historical window/prediction horizon (24/8)
B	Scalar	Mini-batch size (64)
$X^{(j)}$	$\mathbb{R}^{B\times L_h\times 2dN}$	Standardized input window (block form $[\mathbf{P}^{(j)}\|\mathbf{V}^{(j)}]$)
$\hat{Y}^{(j)}$	$\mathbb{R}^{B\times L_p\times 2dN}$	Predicted future window (same channel structure)

integrate patterns hidden in the data with known physical laws. The final result is an inference system that achieves both prediction accuracy and physical interpretability. During testing, the model takes an observation sequence and outputs multi-step predictions without requiring explicit numerical integration. Experiments demonstrate that this framework maintains stable prediction performance even under noise and partial observation conditions, significantly outperforming purely data-driven baseline models. It is suitable for real-world applications such as biological swarm motion analysis and multi-robot cooperative control.

2.1 Data preprocessing module

The data used in this study include both synthetic trajectories generated from two collective motion models and three real-world biological datasets (jackdaw mobbing, jackdaw transit, and Hemigrammus rhodostomus circling). All data are preprocessed into a unified state-space representation. Each agent i contributes a $2d$-dimensional state $[\mathbf{p}_i(t), \mathbf{v}_i(t)]$ (d position + d velocity components; $d=3$ for the jackdaw datasets and $d=2$ for the fish dataset). The implemented feature layout is the block form

$$X^{(j)} = \left[\mathbf{P}^{(j)} \,\middle|\, \mathbf{V}^{(j)}\right] \in \mathbb{R}^{B\times L_h\times 2dN}, \quad \mathbf{P}^{(j)} \in \mathbb{R}^{B\times L_h\times dN}, \quad \mathbf{V}^{(j)} \in \mathbb{R}^{B\times L_h\times dN},$$

i.e., the first dN channels carry the positions of all N agents and the last dN channels their velocities; accordingly, Eq. (3) is written in this block form so that the notation matches the released code exactly. The network output has the same channel structure with the time axis shortened to L_p: $\hat{Y}^{(j)} \in \mathbb{R}^{B\times L_p\times 2dN}$. In all experiments, $B=64$, $L_h=24$, $L_p=8$; N is dataset-dependent ($N\in\{20,40,60,80,100\}$ for the simulated swarms, $N=53$ for jackdaw mobbing, $N=8$ for jackdaw transit, and $N=10$ for the fish school). Table 1 states the dimensional correspondence once and for all.

Sequential states are organized into input-target pairs using a sliding window, then standardized using the Z-score method based solely on training set statistics.

First, the state observation data of the swarm at consecutive time steps, including position and velocity data, are collected and preprocessed. The position and velocity data are fused to obtain timestamped state data. The specific expression is

$$C^t = [\mathbf{p}_1(t), \mathbf{p}_2(t), \ldots, \mathbf{p}_N(t), \mathbf{v}_1(t), \mathbf{v}_2(t), \ldots, \mathbf{v}_N(t)] \in \mathbb{R}^{2dN}, \tag{3}$$

where C^t represents the state vector at the t-th time step; N denotes the total number of agents in the swarm system; d denotes the spatial dimension of the motion space ($d=3$ for the three-dimensional jackdaw datasets, $d=2$ for the two-dimensional fish dataset; accordingly, the flattened state dimension is $6N$ for 3D data and $4N$ for 2D data); $\mathbf{p}_i(t) = (p_i^x(t), p_i^y(t), p_i^z(t))$ represents the three dimensional position of the i-th agent at the t-th time step; $\mathbf{v}_i(t) = (v_i^x(t), v_i^y(t), v_i^z(t))$ represents the velocity of the i-th agent at the t-th time step (for $d=2$, the z-components are absent).

Secondly, time-series data pairs are constructed using the sliding window method. The specific expressions are

$$\begin{aligned} X^{(j)} &= \left[C^t, C^{t+1}, \ldots, C^{t+L_h-1}\right]^{\mathsf{T}} \in \mathbb{R}^{L_h\times 2dN}; \\ Y^{(j)} &= \left[C^{t+L_h}, C^{t+L_h+1}, \ldots, C^{t+L_h+L_p-1}\right]^{\mathsf{T}} \in \mathbb{R}^{L_p\times 2dN}, \end{aligned} \tag{4}$$

where L_h represents the length of the historical sequence (past h time steps); L_p represents the length of the prediction sequence (future p time steps); the superscript T denotes the transpose; $X^{(j)}$ denotes the j-th time slice of the historical sequence; $Y^{(j)}$ denotes the j-th time slice of the prediction sequence; C^{t+L_h-1} represents the state vector at the $t+L_h-1$-th time step; and $C^{t+L_h+L_p-1}$ represents the state vector at the $t+L_h+L_p-1$-th time

step. Thus, the sequence data of the past h time steps can be used to predict the sequence data of the future p time steps.

Next, the state observation data are standardized using the Z-score method. The specific expression is

$$Z = \frac{X - \mu}{\sigma}, \tag{5}$$

where Z represents the standardized state vector; μ denotes the mean of each feature in the training set; and σ denotes the standard deviation of each feature in the training set. In this implementation, to avoid data leakage, the same standardization is applied to all data based solely on the training set information.

To prevent temporal leakage, each experiment sequence is first divided chronologically into disjoint training, validation, and test segments, with borders at $T_1 = \lfloor rT \rfloor$ and $T_2 = T_1 + \lfloor 0.2T \rfloor$, where T is the sequence length and r is the training ratio; the remainder is the test segment. Sliding windows are constructed only afterwards. With window length L_h and prediction horizon L_p, training windows lie entirely inside $[0, T_1)$; validation windows start at $T_1 + 1 - L_h$, so that their inputs may look back into the training segment, but every predicted target step falls inside the validation segment; test windows likewise start at $T_2 + 1 - L_h$ with all targets inside the test segment. Hence no window mixes target steps from two segments, and no future information can leak into training or validation; the normalization statistics are computed from the training segment only. This is the standard border protocol used by recent forecasting benchmarks. The standardized state observation data are then divided into datasets in chronological order, specifically

$$\begin{cases} \text{Training set:} & t = 0 \sim M_{\text{train}}, \\ \text{Validation set:} & t = T_1 - L_h + 1 \sim T_1 - L_h + 1 + M_{\text{val}}, \\ \text{Test set:} & t = T_2 - L_h + 1 \sim T_2 - L_h + 1 + M_{\text{test}}, \end{cases} \tag{6}$$

where the total number of time slices $M = M_{\text{train}} + M_{\text{val}} + M_{\text{test}}$ is determined by the original sequence length T, L_h, L_p, and the segment borders, with $M_{\text{train}} = T_1 - (L_h + L_p) + 1$, $M_{\text{val}} = (T_2 - T_1) - L_p + 1$, and $M_{\text{test}} = (T - T_2) - L_p + 1$; r denotes the training ratio (the fraction of time steps used for training) and is chosen per experiment (0.3–0.7 for the real datasets; 0.1–0.7 for the training-ratio ablation; 0.7 for the simulated robustness experiments and for the transfer experiments of Subsection 3.6).

Models are trained separately when the number of agents N differs across datasets: one independent model is trained from scratch for each dataset, and no parameters are shared or aligned across datasets. Within one model, the two linear layers of DLinear are shared across all $2dN$ channels (the channel-independent design with a single $\mathbb{R}^{L_h \times L_p}$ map per branch), so the backbone has exactly $2(L_h L_p + L_p)$ weights regardless of N; changing N only changes the input width $2dN$, not the parameterization. Consequently, although a separate model is trained per dataset (because the physics parameters and normalization statistics are dataset-specific), no N-specific architecture or N-specific temporal weights are required, and a trained model can be applied to a different number of agents by looping over channels; we verify this empirically in the swarm-size robustness experiments in Subsection 3.6. Moreover, for any agent permutation π, $f(X_\pi) = f(X)_\pi$: because both the series decomposition and the channel-shared linear mappings operate identically on every coordinate channel, permuting the input agent blocks merely permutes the output blocks, provided that the normalization columns are permuted consistently (permutation equivariance by construction).

2.2 DLinear neural network module

We choose the DLinear backbone for four reasons. (1) Inductive bias: strongly ordered collective motion is dominated by a low-dimensional, smooth collective drift plus quasi-periodic fluctuations around it, which is exactly the trend/seasonal structure that series decomposition exposes. (2) Agent-agnostic weight sharing: the channel-independent linear map treats every agent and every coordinate identically, giving permutation equivariance by construction and a parameter count independent of N, which is essential for the small-sample regimes studied in this paper and for scaling across swarm sizes. (3) Stability: linear recurrence-free maps are robust to input noise and do not suffer from the error accumulation or training instabilities of recurrent/attention mechanisms on short windows ($L_h = 24$), which is consistent with our noise-injection results. (4) Fair attribution: DLinear is a strong, widely adopted forecasting baseline [39], so improvements over it can be attributed to the physics module rather than to a stronger backbone.

The complete time series of each agent after standardization is taken as the original sequence and independently subjected to boundary padding. The specific expression is

$$\begin{cases} \text{front} = x_0 \otimes \mathbf{1}_{(k-1)/2}, \\ \text{end} = x_T \otimes \mathbf{1}_{(k-1)/2}, \\ x_{\text{padded}} = [\text{front}, x, \text{end}], \end{cases} \tag{7}$$

where "front" represents the padded starting values; "end" represents the padded ending values; x_0 denotes the first value of the original sequence x; x_T denotes the last value of the original sequence x; $\mathbf{1}_{(k-1)/2} \in \mathbb{R}^{(k-1)/2}$ denotes an all-ones vector; the first (resp. last) value of the sequence is therefore replicated $(k-1)/2$ times at the front (resp. end); the kernel size k is odd; and x_{padded} denotes the padded sequence.

The sequence data are decomposed into a trend component X_{trend} and a seasonal component X_{season} using the moving average decomposition method. The specific expressions are

$$\begin{cases} X_{\text{trend}} = \text{MA}(X; k), \\ X_{\text{season}} = X - X_{\text{trend}}, \end{cases} \tag{8}$$

where $MA(X; k)$ denotes the moving average operation with kernel size k, with the specific expression

$$\text{MA}(X; k)_t = \frac{1}{k} \sum_{m=-(k-1)/2}^{(k-1)/2} x_{t+m}, \tag{9}$$

which matches the implementation (AvgPool1d over the padded sequence) with k odd; x_{t+m} denotes the state vector at the $(t+m)$-th time step, and m denotes the offset index.

After decomposition, two independent linear networks are applied to predict the trend component and the seasonal component, respectively. The specific expressions are

$$\begin{cases} \hat{X}_{\text{trend}} = W_t X_{\text{trend}}^{\mathsf{T}} + b_t, \\ \hat{X}_{\text{season}} = W_s X_{\text{season}}^{\mathsf{T}} + b_s, \end{cases} \tag{10}$$

where $\hat{X}_{\text{trend}}$ denotes the predicted value of the trend component; $\hat{X}_{\text{season}}$ denotes the predicted value of the seasonal component; W_t denotes the weight matrix for X_{trend}; W_s denotes the weight matrix for X_{season}; b_t denotes the bias for X_{trend}; and b_s denotes the bias for X_{season}.

The prediction results of the trend component and the seasonal component are summed to obtain the final prediction result. The specific expression is

$$\hat{y} = (\hat{X}_{\text{trend}} + \hat{X}_{\text{season}})^{\mathsf{T}}, \tag{11}$$

where $\hat{y}$ denotes the final prediction result; $\hat{X}_{\text{trend}}$ denotes the prediction result of the trend component; and $\hat{X}_{\text{season}}$ denotes the prediction result of the seasonal component.

2.3 Physical information module

This paper selects the classic Cucker-Smale model as the basis for the learnable physics kernel, mainly based on two considerations. First, theoretical relevance. The core of the C-S model lies in the distance-decaying velocity coupling (alignment) mechanism between individuals, which is widely regarded as the most critical driver for the emergence of global ordered motion (such as synchronous flight and turning) from local interactions in biological swarms [20]. Our goal is precisely to predict such coordinated swarm trajectories, so this model provides appropriate physical priors. Second, model simplicity and plasticity.

Before defining the physics-constrained loss, we clarify the input sources of the physics module (this workflow is also reflected in Algorithm 1). During training, the physics residual is evaluated on the concatenated trajectory $[X^{(j)}; \hat{Y}^{(j)}]$, i.e., the observed history followed by the network's own multi-step prediction, mapped back to physical units by the inverse Z-score transform. Concretely, at each predicted step $\tau \in \{1, \ldots, L_p\}$ the acceleration $\hat{\mathbf{a}}_\tau = (\hat{\mathbf{v}}_{\tau+1} - \hat{\mathbf{v}}_\tau)/\Delta t$ is computed from the predicted velocities, and the interaction force $\mathbf{F}_\tau^{\text{inter}}$ is computed from the predicted positions and velocities at step τ over all pairs $j \neq i$; the ground-truth future $Y^{(j)}$ is never fed into the physics loss. For the first predicted step, the velocity difference spans the boundary between the observed history and the prediction, which couples the two segments.

Algorithm 1 Training procedure of PI-DLinear.

Require: standardized windows $\{X^{(j)}, Y^{(j)}\}$ with $X^{(j)} \in \mathbb{R}^{B\times L_h\times 2dN}$, $Y^{(j)} \in \mathbb{R}^{B\times L_p\times 2dN}$, where d denotes the spatial dimension ($d = 3$ for the jackdaw datasets; $d = 2$ for the fish dataset); initialization values now distinguish simulated (0.5, 0.4) and real-world (0.2, 0.5) datasets; sampling interval Δt; training-set statistics $(\boldsymbol{\mu}, \boldsymbol{\sigma})$.

for epoch = 1 to 200 **do**
  Sample a mini-batch (X_b, Y_b);
  $\hat{Y}_b \leftarrow \mathrm{DLinear}(X_b)$: series decomposition (Eqs. (7)–(9)) + shared linear maps ((10) and (11));
  Form the concatenated trajectory $[X_b; \hat{Y}_b]$ and map it back to physical units: $T \leftarrow [X_b; \hat{Y}_b] \odot \boldsymbol{\sigma} + \boldsymbol{\mu}$;
  Compute accelerations by forward difference on T: $\hat{\mathbf{a}}_\tau = (\mathbf{v}_{\tau+1} - \mathbf{v}_\tau)/\Delta t$;
  Compute the interaction force at each predicted step (kernel φ as in (16)): $\mathbf{F}^{\mathrm{inter}}_{\tau,i} = \alpha \sum_{j\neq i} \left(1 + \|\mathbf{p}_{\tau,i} - \mathbf{p}_{\tau,j}\|^2\right)^{-\beta} (\mathbf{v}_{\tau,j} - \mathbf{v}_{\tau,i})$;
  $\mathcal{L}_{\mathrm{phy}} \leftarrow \frac{1}{L_p} \sum_{\tau=1}^{L_p} \|\hat{\mathbf{a}}_\tau - \mathbf{F}^{\mathrm{inter}}_\tau\|^2$;
  $\mathcal{L}_{\mathrm{data}} \leftarrow \mathrm{MSE}(\hat{Y}_b, Y_b)$;
  Adaptive balancing weight $\lambda \leftarrow \|\nabla_\theta \mathcal{L}_{\mathrm{data}}\|_2 / \left(\|\nabla_\theta \mathcal{L}_{\mathrm{phy}}\|_2 + 10^{-8}\right)$;
  $\theta \leftarrow \theta - \mathrm{AdamW}\left(\nabla_\theta \left[\mathcal{L}_{\mathrm{data}} + \lambda \mathcal{L}_{\mathrm{phy}}\right]\right)$ with cosine-annealed learning rate, gradient norm clipped to 1.0;
end for
Keep the checkpoint with the lowest validation loss.

The physics-constrained loss function is defined as follows, which transforms the self-organizing characteristics of the swarm into computable physical constraints.

$$\mathcal{L}_{\mathrm{phy}} = \frac{1}{L_p} \sum_{\tau=1}^{L_p} \left\| \hat{\mathbf{a}}_\tau \ - \mathbf{F}^{\mathrm{inter}} \right\|^2, \tag{12}$$

where $\mathcal{L}_{\mathrm{phy}}$ denotes the value of the physics-constrained loss function; $\hat{\mathbf{a}}_\tau$ denotes the predicted acceleration value at the τ-th time step, which can be calculated from the velocity components in $\hat{y}$.

We combine the data fitting loss and the physics constraint loss to construct the total loss function $\mathcal{L}_{\mathrm{total}}$. The specific expression is

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{data}} + \lambda \mathcal{L}_{\mathrm{phy}}, \tag{13}$$

where λ is the adaptive data-physics balancing weight of the total loss (see the closed form below); it is not gradient-learned but recomputed at each iteration by gradient-norm balancing, with a fixed fallback value used if the computation fails; $\mathbf{y}$ represents the real data; and $\mathcal{L}_{\mathrm{data}}$ denotes the data fitting loss.

Based on the sequence data, train and optimize the spatio-temporal decomposition linear neural network. During backpropagation, parameter gradients are calculated using the chain rule. The specific expression is

$$\frac{\partial \mathcal{L}_{\mathrm{total}}}{\partial \theta} = \frac{\partial \mathcal{L}_{\mathrm{data}}}{\partial \theta} + \lambda \frac{\partial \mathcal{L}_{\mathrm{phy}}}{\partial \theta}, \tag{14}$$

where θ denotes the set of learnable parameters over which Eq. (14) differentiates, including the model and physics parameters only, i.e., $\theta = \{W_t, b_t, W_s, b_s, \alpha, \beta\}$; the balancing weight λ is not updated by gradient descent, so no gradient term $\partial \mathcal{L}_{\mathrm{total}}/\partial \lambda$ is required. At every training iteration t, λ is computed adaptively by gradient-norm balancing

$$\lambda^{(t)} = \frac{\left\| \nabla_\theta \mathcal{L}^{(t)}_{\mathrm{data}} \right\|_2}{\left\| \nabla_\theta \mathcal{L}^{(t)}_{\mathrm{phy}} \right\|_2 + 10^{-8}}, \qquad \mathcal{L}^{(t)}_{\mathrm{total}} = \mathcal{L}^{(t)}_{\mathrm{data}} + \lambda^{(t)} \mathcal{L}^{(t)}_{\mathrm{phy}}$$

with the fixed fallback $\lambda = 0.1$ if the computation fails; hence $\lambda \in (0, \infty)$ adapts automatically to the relative gradient scales of the two losses.

Remark. The objective above is equivalent, up to a positive rescaling by $1/(1-\lambda')$, to the convex combination $(1-\lambda')\mathcal{L}_{\mathrm{data}} + \lambda' \mathcal{L}_{\mathrm{phy}}$ with $\lambda' = \lambda/(1+\lambda) \in (0, 1)$; the two parameterizations therefore yield exactly the same optimization trajectory.

The admissible ranges and parameterizations of the physics parameters are as follows. (1) Coupling strength α: a learnable scalar (initialized to 0.5 for the simulated datasets and 0.2 for the real-world datasets), updated by backpropagation together with the network weights; physically admissible values satisfy $\alpha \geqslant 0$ ($\alpha = 0$ disables alignment), and in practice α remains positive for all fitted models. (2) Decay coefficient β: a learnable scalar (initialized to 0.4 for the simulated datasets and 0.5 for the real-world datasets); the admissible range is $\beta > 0$, which guarantees that the kernel $(1 + x^2)^{-\beta}$ decays with distance, and larger β corresponds to a shorter effective interaction range. (3) Balancing weight λ: not a gradient-learned parameter, but computed in closed form by the gradient-norm balancing rule above. The ablation study in Subsection 3.3 confirms that this parameterization (learnable α, β; adaptively balanced λ) is the best-performing variant.

The C-S model has a concise mathematical form, and its key parameters (such as coupling strength α, decay coefficient β) have clear physical meanings, making it very suitable as a learnable module embedded in a neural network, facilitating optimization and interpretation. The model describes a swarm system containing N individuals: For any individual i, its motion state is described by the position vector $\mathbf{p}_i(t) \in \mathbb{R}^d$ and the velocity vector $\mathbf{v}_i(t) \in \mathbb{R}^d$ in d-dimensional space (d is a positive integer, corresponding to the spatial dimension of individual motion). The evolution of individual i's motion state over time is characterized by the following system of differential equations:

$$\begin{aligned} \frac{d}{dt}\mathbf{p}_i(t) &= \mathbf{v}_i(t), \\ \frac{d}{dt}\mathbf{v}_i(t) &= \sum_{j\neq i}^{N} \varphi\left(\|\mathbf{p}_j(t)-\mathbf{p}_i(t)\|\right)\left(\mathbf{v}_j(t)-\mathbf{v}_i(t)\right), \end{aligned} \tag{15}$$

where $\mathbf{v}_j(t)-\mathbf{v}_i(t)$ is the relative velocity between two individuals; $\varphi(x)$ is the interaction function between individuals, and its value is determined by the spatial distance $\|\mathbf{p}_j(t)-\mathbf{p}_i(t)\|$ between the two individuals. The specific form is

$$\varphi(x) = \frac{\alpha}{(1+x^2)^{\beta}}, \tag{16}$$

where α is the parameter regulating the interaction strength between individuals (also known as the coupling strength), and β is the decay coefficient, a learnable positive parameter.

The learnable physics module uses a fully connected, unweighted interaction graph: the neighbor set of agent i is $\mathcal{N}_i = \{j : j \neq i\}$ (adjacency matrix $\mathbf{A} = \mathbf{1}\mathbf{1}^{\mathsf{T}} - \mathbf{I}$). There is no hard distance cutoff; spatial selectivity enters softly through the distance-dependent kernel $(1+\|\mathbf{p}_i-\mathbf{p}_j\|^2)^{-\beta}$, whose effective range is learned via β. We note that the generative simulators use their own interaction rules (an Erdős-Rényi random graph in the second-order Vicsek model; the modified Cucker-Smale force terms), i.e., the prior is deliberately not exactly matched to the generators, and the experiments show the learned kernel generalizes across these mismatches. For each predicted step $\tau = 1, \ldots, L_p$, the force acting on agent i is computed from the network's predicted states at that step as

$$\mathbf{F}^{\text{inter}}_{\tau,i} = \alpha \sum_{j\neq i} \left(1+\|\hat{\mathbf{p}}_{\tau,i}-\hat{\mathbf{p}}_{\tau,j}\|^2\right)^{-\beta} \left(\hat{\mathbf{v}}_{\tau,j}-\hat{\mathbf{v}}_{\tau,i}\right),$$

where $\hat{\mathbf{p}}_\tau, \hat{\mathbf{v}}_\tau$ are the inverse-standardized predicted positions/velocities; the physics residual $\|\hat{\mathbf{a}}_\tau - \mathbf{F}^{\text{inter}}_\tau\|^2$ is then averaged over the L_p predicted steps. Ground-truth future states are never used in this computation. The time interval is $\Delta t = 0.1$ s for the simulated trajectories (sampled uniformly) and $\Delta t = 1/60 \approx 0.0167$ s for the real-world datasets (recorded at 60 frames per second); accelerations are computed by the forward difference $\hat{\mathbf{a}}(t) = (\mathbf{v}(t+\Delta t)-\mathbf{v}(t))/\Delta t$ on the concatenated observed-predicted trajectory. Because a standardized residual would be dimensionless and physically meaningless, all states entering the physics loss are mapped back to the original physical units with the inverse transform $Z \odot \sigma + \mu$.

The learnable parameterization also lets the kernel adaptively degenerate into weaker or more local interaction modes, which explains its applicability even in low-polarization systems: (1) the coupling strength α scales the magnitude of the alignment force, so $\alpha \to 0$ continuously suppresses velocity coupling, which is the regime expected in low-polarization systems such as the circling fish school (polarization < 0.3); (2) the decay exponent β controls the effective interaction range—increasing β shortens the range over which $(1+\|\mathbf{p}_i-\mathbf{p}_j\|^2)^{-\beta}$ remains non-negligible, recovering a local-interaction mode, whereas small β approaches global (all-to-all) coupling; (3) because the physics term enters as a soft constraint whose weight λ is set adaptively by gradient balancing, the data-fitting term automatically dominates whenever the prior mismatches the data, so a single fixed embedding cannot force incorrect dynamics on any scenario. The ablation study in Subsection 3.3 further supports this interpretation: prediction is best precisely when α and β are learnable, i.e., when the kernel can adjust its strength and range behavior-by-behavior.

Scope and limitations of the physics prior. The current physics prior is an alignment-dominant, first-order (velocity-coupling) model with homogeneous agents; behaviors whose dominant mechanism is not velocity alignment lie outside the scope of the homogeneous C-S prior and are left as future work (learnable multi-term or species-specific kernels).

DLinear processes each channel independently and therefore cannot represent inter-individual spatial coupling by itself. The physics module compensates exactly for this deficiency during training: through the residual $\|\hat{\mathbf{a}}_\tau - \mathbf{F}^{\text{inter}}_\tau\|^2$, the gradient with respect to any agent's predicted position/velocity depends on all other agents' predicted states via the pairwise kernel $\varphi(\|\mathbf{p}_i-\mathbf{p}_j\|)$, so interaction structure is injected into the otherwise channel-decoupled

backbone in a differentiable, physically interpretable way. Moreover, because α and β are learnable, the coupling strength and range are identified from data rather than hand-specified. In this sense, PI-DLinear follows the PINN philosophy: a simple, data-efficient surrogate is corrected by a soft physical law rather than building interaction structure into a black-box architecture whose inductive bias cannot be inspected.

2.4 Evaluation metrics

We quantify prediction performance using root mean square error (RMSE). Given a swarm of N agents, a prediction horizon of L_p steps, and a batch of test samples, these metrics are defined as

$$\mathrm{RMSE} = \sqrt{\frac{1}{NL_p}\sum_{i=1}^{N}\sum_{\tau=1}^{L_p}\|\hat{\mathbf{c}}_i(t+\tau)-\mathbf{c}_i(t+\tau)\|_2^2}, \tag{17}$$

where $\hat{\mathbf{c}}_i(t+\tau)$ and $\mathbf{c}_i(t+\tau)$ denote the predicted and true states of agent i at future step τ, respectively. The ℓ_1-norm and ℓ_2-norm are computed over the d spatial dimensions ($d = 3$ for the jackdaw datasets; $d = 2$ for the fish dataset). Lower values indicate better prediction accuracy. These metrics are reported for all experiments, both on simulated and real-world datasets, to enable comprehensive performance comparison.

To systematically assess model robustness, we design two complementary evaluation protocols: noise injection and small-sample training. For *noise injection*, we corrupt input sequences with additive Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma_n^2\mathbf{I})$, where the noise level $\sigma_n \in [0, 0.1]$ is defined relative to the per-feature standard deviation estimated on the training set (velocity noise scaled by a factor of 0.5), and the noise is applied to the test inputs only. Models are trained on clean data but evaluated on noisy inputs, testing their tolerance to measurement uncertainty. For *small-sample training*, we limit the training set to a fraction (e.g., 10%) of the full data while keeping validation and test sets unchanged, evaluating the model's ability to learn from limited demonstrations. Both protocols are applied consistently across all datasets and compared against baseline models under identical conditions, providing a rigorous assessment of generalization capability under realistic constraints.

2.5 Baselines

We compare the proposed PI-DLinear with several state-of-the-art pure data-driven prediction models that have been widely applied in time series prediction and population trajectory prediction. (1) DLinear (decomposition linear neural network) [39]: a recently proposed linear architecture that decomposes time series into trend and seasonal components using a moving average kernel, and applies two independent linear layers for prediction. It is the main baseline for us to isolate the impact of physical information modules. (2) Transformer [40]: the initial architecture based on self-attention mechanism has been adjusted for sequence-to-sequence prediction tasks. We adopted a standard encoder-decoder configuration with positional encoding. (3) TCN (time convolutional network) [41]: a convolutional model with dilated causal convolution that can capture long-range dependencies while maintaining temporal order. (4) SRU (simple recurrent unit) [42]: a recurrent architecture designed to accelerate training speed and achieve parallelization while retaining the sequence modeling capability of RNNs. (5) Autoformer [43]: an advanced Transformer variant that combines autocorrelation mechanisms and decomposition architecture to handle complex seasonal patterns. (6) iTransformer [44]: a recent forecaster (ICLR 2024) that embeds each variate as a token and applies attention across variates—a representative recent model for multivariate trajectory forecasting. (7) LaGNA [33]: a gray-box approach (Nature Communications 2024) that learns a message-passing GNN to disentangle cohesion, alignment, and self-propulsion kernels, distills them into explicit Langevin equations via sparse symbolic regression, and predicts by integrating the discovered equations. Following the equation-discovery protocol of the original paper, prediction integrates the explicit equations from a single initial state at the end of the validation segment across the whole test segment (drift term only, since the conditional mean is the RMSE-optimal point forecast), and the integrated trajectory is then sliced into the same $24 \to 8$ windows as every other method.

All data-driven models share one training protocol: input length 24, horizon 8, batch size 64, AdamW (learning rate 10^{-3}, cosine annealing over the full run), gradient clipping at 1.0, up to 200 epochs with checkpoint selection by validation loss, and ten independent runs per configuration. The baseline architectures use the configurations of their official implementations with hidden dimensions chosen so that their capacities bracket PI-DLinear from above: every learned baseline has two to four orders of magnitude more trainable parameters than PI-DLinear (402 parameters), so the performance advantage of PI-DLinear cannot be attributed to larger capacity. LaGNA, as an equation-discovery method, has no iterative epoch budget comparable to gradient-based forecasters; we followed its

official two-stage protocol and hyperparameters. Complete settings are documented in Appendix D (Tables D1 and D2).

For fair comparison, all methods share the input sequence length $L_h = 24$ and the prediction horizon $L_p = 8$. We apply the same AdamW optimizer, cosine annealing scheduler, gradient clipping, and early-stopping strategy, and optimize the hyperparameters (learning rate, batch size, number of layers, etc.) separately on the validation set of each dataset.

These baselines cover multiple architecture types (linear, cyclic, convolutional, and self-attention based), which can comprehensively evaluate the proposed physical information decomposition framework in terms of prediction accuracy, generalization ability for different collective behaviors, and robustness under small sample and noisy conditions.

3 Experiments and results analysis

In this section, we evaluate the proposed PI-DLinear model on simulated data and three real-world datasets. The simulated data include the second-order Vicsek model, as well as data generated from a modified Cucker-Smale model incorporating collision-avoidance and shape-preserving terms. The real-world data comprise jackdaw mobbing behavior, jackdaw transit behavior, and Hemigrammus rhodostomus circling behavior. The proposed gray-box model PI-DLinear, built on a linear architecture, is compared with a variety of state-of-the-art purely data-driven models based on RNN, CNN, and Transformer architectures. To ensure a fair comparison, all models are evaluated on the same training dataset. By contrasting PI-DLinear with corresponding purely data-driven models, we investigate its generalization capability and robustness. Furthermore, ablation experiments are conducted to study the generalization performance of PI-DLinear versus DLinear under small-sample conditions, as well as the potential influence of related parameters, thereby validating the effectiveness of physics-informed embedding. All experiments were conducted under the leakage-free split-then-window protocol described in Subsection 2.1.

3.1 Training strategy and hyperparameter setting

This study adopts an end-to-end joint training strategy, synchronously optimizing the data-driven loss and the physics-constrained loss to achieve bidirectional learning of model parameters and physical mechanisms. The specific training settings are as follows: we use the AdamW optimizer (learning rate = 1×10^{-3}), coupled with a cosine annealing learning rate scheduler (cycle length = 200) to enhance convergence stability.

We adjust the learning rate and update the parameter gradients using the cosine annealing method. The specific expression is

$$\eta_t = \eta_{\min} + \frac{1}{2}\left(\eta_{\max} - \eta_{\min}\right)\left(1 + \cos\left(\frac{t}{T_{\max}}\pi\right)\right), \tag{18}$$

where η_t denotes the learning rate at the current iteration step; $\eta_{\min}$ denotes the minimum learning rate; $\eta_{\max}$ denotes the maximum learning rate; t denotes the current iteration step; and $T_{\max}$ denotes the length of a complete decay cycle.

During training, gradient clipping (norm threshold = 1.0) is applied to prevent gradient explosion. The specific expression is

$$\text{grad} \leftarrow \frac{\text{grad} \cdot \max_{\text{norm}}}{\max(\|\text{grad}\|, \max_{\text{norm}})}, \tag{19}$$

where grad denotes the parameter gradient vector, and $\max_{\text{norm}}$ denotes the clipping threshold.

The batch size is set to 64, the model is trained for up to 200 epochs, and the checkpoint with the lowest validation loss is retained for testing. For statistical reliability, every configuration is run ten times with different random seeds (five seeds for the transfer experiments of Subsection 3.6), and all reported numbers are averaged over these runs. The physics parameters are initialized as follows: α to 0.5 (simulated datasets)/0.2 (real-world datasets), β to 0.4 (simulated)/0.5 (real-world), with the admissible ranges $\alpha \geqslant 0$ and $\beta > 0$; the balancing weight λ follows the gradient-norm balancing rule of (14) with the fixed fallback $\lambda = 0.1$. The simulated trajectories are sampled uniformly with interval $\Delta t = 0.1$ s; the real-world datasets are recorded at 60 frames per second, corresponding to $\Delta t = 1/60 \approx 0.0167$ s. Accelerations are computed by the forward difference $\hat{\mathbf{a}}(t) = (\mathbf{v}(t + \Delta t) - \mathbf{v}(t))/\Delta t$ on the concatenated observed-predicted trajectory.

All experiments were conducted on a high-performance computing platform equipped with an NVIDIA GeForce RTX 5090 GPU (24 GB memory). The software environment is based on Python 3.13, PyTorch 2.7.1, and CUDA 12.8. All dependency library versions are frozen in the requirements.txt file to ensure reproducibility. Additionally,

complete training code, pre-trained models, dataset processing scripts, and hyperparameter configuration documents are provided.

3.2 Simulated data generation

To validate the proposed PI-DLinear method, we consider simulating three dimensional collective motion using the second-order Vicsek model with topological communication [33] and a modified Cucker-Smale model incorporating collision-avoidance and shape-preserving terms [22]. Specifically, position and velocity data are generated for swarms containing [20, 40, 60, 80, 100] agents. The simulated trajectories are sampled with a uniform time interval of $\Delta t = 0.1$ s. We employ the second-order Vicsek model to simulate bird-like collective behavior. A swarm of N individuals evolves in three-dimensional space, with the position and velocity of each agent i governed by the following coupled stochastic differential equations:

$$\begin{aligned} \frac{d\mathbf{r}_i}{dt} &= \mathbf{v}_i, \\ \frac{d\mathbf{v}_i}{dt} &= \gamma(v_0^2 - \|\mathbf{v}_i\|^2)\mathbf{v}_i + \sum_{j\neq i} A_{ij}\left(\mathbf{F}_{ij}^{\text{rep/att}} + \mathbf{F}_{ij}^{\text{align}}\right) + \boldsymbol{\eta}_i(t), \end{aligned} \tag{20}$$

where γ is the velocity relaxation coefficient and v_0 is the desired cruising speed. The short-range repulsion and long-range attraction between individuals i and j are described by a modified Morse potential

$$\mathbf{F}_{ij}^{\text{rep/att}} = \epsilon_0 \cdot \phi\left(\frac{\|\mathbf{r}_{ij}\|}{r_0}\right) \cdot \hat{\mathbf{r}}_{ij}, \tag{21}$$

where $\mathbf{r}_{ij} = \mathbf{r}_j - \mathbf{r}_i$, $\hat{\mathbf{r}}_{ij}$ is the normalized direction vector, ϵ_0 is the strength coefficient, and r_0 is the characteristic interaction distance. The potential function $\phi(x)$ is defined as

$$\phi(x) = \frac{1 - x^3}{1 + x^6}. \tag{22}$$

This function exhibits repulsion for $x < 1$ and attraction for $x > 1$, with a smooth transition at $x = 1$. The velocity alignment interaction decays exponentially with distance

$$\mathbf{F}_{ij}^{\text{align}} = \epsilon_1 \cdot \exp\left(-\frac{\|\mathbf{r}_{ij}\|}{r_1}\right) \cdot (\mathbf{v}_j - \mathbf{v}_i), \tag{23}$$

where ϵ_1 is the alignment strength and r_1 is the characteristic alignment length. The interaction network is defined by an Erdős-Rényi random graph $G(N, p)$. The adjacency matrix elements A_{ij} satisfy

$$A_{ij} = \begin{cases} 1, & \text{with probability } p \text{ (a connection exists)}, \\ 0, & \text{otherwise}, \end{cases} \quad \text{and} \quad A_{ii} = 0. \tag{24}$$

A mechanical interaction exists between individuals i and j only if $A_{ij} = 1$, simulating the limited perception or communication range of biological agents. The model includes a multiplicative noise term $\boldsymbol{\eta}_i(t) = \sigma \cdot \mathbf{v}_i \circ d\mathbf{W}_i(t)$ to simulate environmental uncertainty, where $d\mathbf{W}_i(t)$ is the increment of a Wiener process and σ controls the noise intensity.

Eq. (15) introduces the classic Cucker-Smale model proposed in 2007. Since then, a large amount of research has focused on the formation and convergence analysis of formations based on this model by adding anti-collision and shape terms [45–47]. A modified Cucker-Smale model [22] is used to simulate collective behavior with specific formations. The basic dynamic equation of the modified model is

$$\frac{d\mathbf{v}_i}{dt} = \alpha\mathbf{F}_i^{\text{inter}} - \eta\mathbf{F}_i^{\text{col}} - \gamma\mathbf{F}_i^{\text{shape}}, \tag{25}$$

where $\mathbf{F}_i^{\text{inter}}$ represents the interaction force, $\mathbf{F}_i^{\text{col}}$ the collision avoidance force, $\mathbf{F}_i^{\text{shape}}$ the shape-preserving force, and α, η, γ are the corresponding strength parameters. $\mathbf{a}_i(t)$ is the acceleration of agent i at time t, calculated from the velocity difference. $\mathbf{F}_i^{\text{inter}}$ represents the interaction force on the i-th agent for velocity alignment with its neighbors. Its specific formula is

$$\mathbf{F}_i^{\text{inter}} = \sum_{j\in\mathcal{N}_i} c_{ij} \cdot \left(1 + \|\mathbf{p}_i - \mathbf{p}_j\|^2\right)^{-0.4} \cdot (\mathbf{v}_i - \mathbf{v}_j), \tag{26}$$

where $\mathcal{N}_i$ denotes the set of neighboring agents for agent i; c_{ij} is an element of the adjacency matrix representing the communication network ($c_{ij} = 1$ if interaction exists between i and j, otherwise $c_{ij} = 0$); $\mathbf{p}_i$ is the position of agent i. The collision avoidance force is calculated as

$$\mathbf{F}_i^{\text{col}} = \sum_{j \neq i} \frac{r}{2} \sin\left(\frac{\pi}{N}\right) \cdot \left(\frac{\mathbf{p}_{ij}}{\|\mathbf{p}_{ij}\|} - \mathbf{p}_{ij}\right) \cdot \mathbb{I}_{\{\|\mathbf{p}_{ij}\| < 2t_c\}}, \tag{27}$$

where N is the number of agents in the swarm, $\mathbb{I}(\cdot)$ is the indicator function (activated when the distance is less than $2t_c$), the individual radius for each agent is set to r, the collision avoidance threshold is $t_c = 2r\sin(\pi/N)$, and $\mathbf{p}_{ij} = \mathbf{p}_i - \mathbf{p}_j$ is the position difference between agents i and j. The shape-preserving force is calculated as

$$\mathbf{F}_i^{\text{shape}} = (\|\mathbf{d}_i\| - R_0) \cdot \frac{\mathbf{d}_i}{\|\mathbf{d}_i\|}, \tag{28}$$

where $\mathbf{d}_i = \mathbf{p}_i - \frac{1}{N}\sum_{k=1}^{N} \mathbf{p}_k$ represents the distance between the i-th agent and the swarm's center of mass. The target radius for the self-organized circular formation is set to $R_0 = 10$ m. The three-dimensional simulations using the modified Cucker-Smale model with collision-avoidance and shape-preserving terms not only reproduce macroscopic coordination in collective motion (such as the aggregating trend of overall trajectories) but also reveal heterogeneity at the microscopic level (complexity of individual trajectories, fluctuations in velocity synchronization). These characteristics constitute the core challenge of trajectory prediction: how to capture the nonlinear coupled dynamic laws in high-dimensional space. This establishes the data foundation and problem framework for subsequent predictive studies.

3.3 Results on simulated data

By comparing with corresponding purely data-driven models (Transformer [40], SRU [42], TCN [41], Autoformer [43], DLinear [39], iTransformer [44], and LaGNA [33]), we investigate the generalization capability and robustness of the PI-DLinear method on multi-agent simulated data of varying scales (see the methods for details). The experiments cover agent counts of [20, 40, 60, 80, 100], revealing the relationship between model performance and system scale. The prediction accuracy is evaluated using the RMSE between the predicted trajectories and the ground truth.

We first evaluate the models on two representative synthetic swarms: the Vicsek model, which enforces local velocity alignment over a fixed topological (Erdős-Rényi) random graph (Figure 2(a)), and the modified Cucker-Smale model, which adopts distance-dependent velocity coupling (Figure 2(d)). Under the Vicsek dynamics, agents become spatially dispersed while their velocities converge—a pattern consistent with "velocity synchronization dominating collective behavior". In contrast, the Cucker-Smale model produces spatially expanding clusters with dispersed endpoints, reflecting the heterogeneity of distance-based coupling and an alternating divergence-convergence pattern amplified by three-dimensional space.

Noise robustness is examined by injecting additive Gaussian noise into the input sequences (Figures 2(b) and (e)). On the C-S swarm (Figure 2(e)), PI-DLinear attains the lowest RMSE at every noise level: its error grows only modestly from 2.25 (clean) to 2.78 (strongest noise), remaining clearly below DLinear (3.52–4.05) and Autoformer (5.62–6.74) throughout, with a reduction of 31.3% relative to DLinear at the highest noise level. On the Vicsek swarm (Figure 2(b)), PI-DLinear (0.347–0.349) performs on par with, though marginally above, DLinear (0.344–0.347), while both remain clearly below Autoformer (0.366–0.367); all three curves are nearly flat, indicating that the linear backbones are intrinsically robust to input noise. The near-parity between PI-DLinear and DLinear on Vicsek is expected and is discussed transparently here and in Section 4: the Vicsek generator interacts over a topological random graph rather than through the distance-decaying alignment kernel embedded in PI-DLinear, so the physics prior does not exactly match the underlying mechanism, and with 70% of the data used for training, the data-fitting term dominates, and the physics term contributes little.

We also assess scalability by varying the number of agents from 20 to 100 (Figures 2(c) and (f)). For the Vicsek model (Figure 2(c)), PI-DLinear and DLinear remain clearly below all other baselines at every swarm size (both stay around 0.30–0.35, versus 0.34–0.38 for Autoformer, 0.50–0.64 for TCN, 0.58–0.63 for SRU, 0.57–0.71 for iTransformer, 0.85–0.97 for Transformer, and 2.18–2.58 for LaGNA), and the two curves are nearly identical, with DLinear marginally lower at every size; LaGNA, as an equation-discovery method, degrades strongly in this regime. For the C-S model (Figure 2(f)), PI-DLinear achieves the lowest RMSE for 20–80 agents (e.g., 3.57 versus 17.03 for DLinear at 60 agents) and remains within the leading group at 100 agents (26.33, versus 17.42 for DLinear and 18.27 for Transformer), where all methods degrade sharply because of the stronger collective coupling at large swarm sizes;

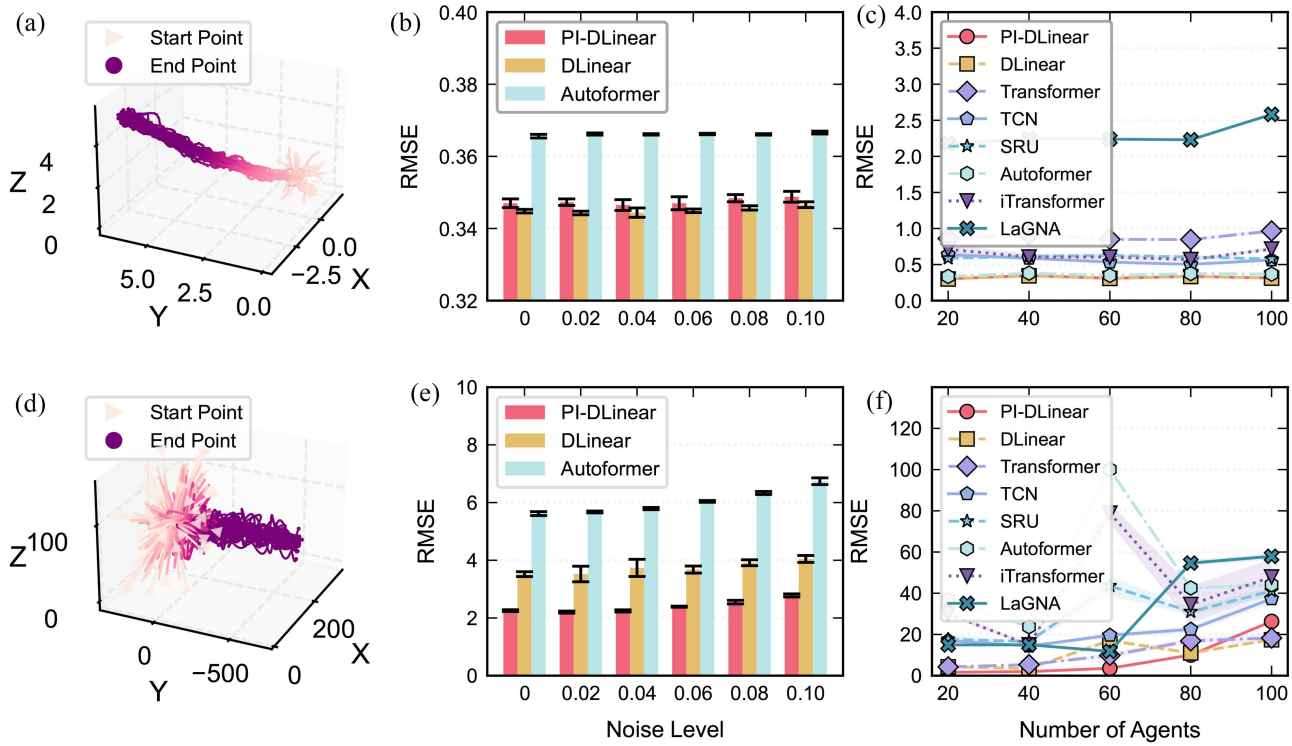

Figure 2 (Color online) Simulated trajectory data and comparative experimental results. (a) 3D trajectories of 40 agents simulated by the Vicsek model. (b) Robustness analysis under different noise levels for the Vicsek model. (c) Comparative results for different agent counts under the Vicsek model. (d) Three dimensional trajectories of 40 agents simulated by the C-S model. (e) Robustness analysis under different noise levels for the C-S model. (f) Comparative results for different agent counts under the C-S model. All models, including the newly added iTransformer and LaGNA baselines, are shown in (c) and (f).

TCN, SRU, Autoformer, iTransformer, and LaGNA deteriorate far more severely (up to 100.08 for Autoformer at 60 agents and 57.90 for LaGNA at 100 agents). We report these scale-induced degradations transparently and analyze them further in Appendix B. These results demonstrate that physics priors effectively regularize the model against scale expansion, a critical advantage for real-world swarms of varying sizes.

To further investigate the robustness and adaptability of PI-DLinear, we conducted a series of ablation experiments and hyperparameter sensitivity experiments (Figures 3(a)–(f)). Figures 3(a) and (d) show the results when the interaction strength coefficient α and decay coefficient β in the physical information calculation formula (16) are set as learnable parameters or fixed hyperparameters, respectively. Prediction is best precisely when both α and β are learnable (0.492 on Vicsek, 14.96 on C-S); fixing $\alpha = 0.4, \beta = 0.4$ gives nearly the same C-S error (14.96) but a slightly higher Vicsek error (0.497); fixing $\alpha = 0.2$ while keeping β learnable degrades performance (0.530 on Vicsek, 15.46 on C-S); and fixing both at $\alpha = 0.2, \beta = 0.4$ gives the worst physics configuration (0.661, 15.90). All physics variants outperform the physics-free DLinear on C-S (16.74), whereas on Vicsek only the learnable configuration and the 0.4/0.4 configuration do (0.492 and 0.497 versus 0.526). In addition, the interaction strength coefficient α is more important than the decay coefficient β and has a more significant impact on the prediction performance, so that the embedded dynamic physical kernel can automatically adjust the physical constraint strength behind different behaviors based on the data.

Figures 3(b) and (e) illustrate the impact of different training set ratios (0.1–0.7) on model performance. As the proportion of training data increases, the prediction errors of both PI-DLinear and the purely data-driven DLinear show a decreasing trend. On the C-S model (Figure 3(e)), PI-DLinear outperforms DLinear at every ratio, with the gap widening from 10.9% (14.65 versus 16.44 at ratio 0.1) to 38.3% (2.23 versus 3.61 at ratio 0.7), confirming that the matched physics prior acts as an effective regularizer. On the Vicsek model (Figure 3(b)), the advantage of the physics term concentrates in the low-data regime: PI-DLinear is clearly better at ratio 0.1 (0.505 versus 0.526) and comparable at ratios 0.2–0.5, while at ratios 0.6–0.7 DLinear is marginally lower (0.316/0.344 versus 0.324/0.347 for PI-DLinear). This pattern is consistent with the prior-mechanism mismatch on Vicsek discussed in Subsection 3.3 together with the abundance of training data: when data are plentiful, and the prior does not match the generative mechanism, the adaptively balanced physics term neither helps nor hurts much. Figures 3(c) and (f) illustrate the influence of the decomposition kernel size (9–29). Increasing the kernel size enhances the ability to capture temporal

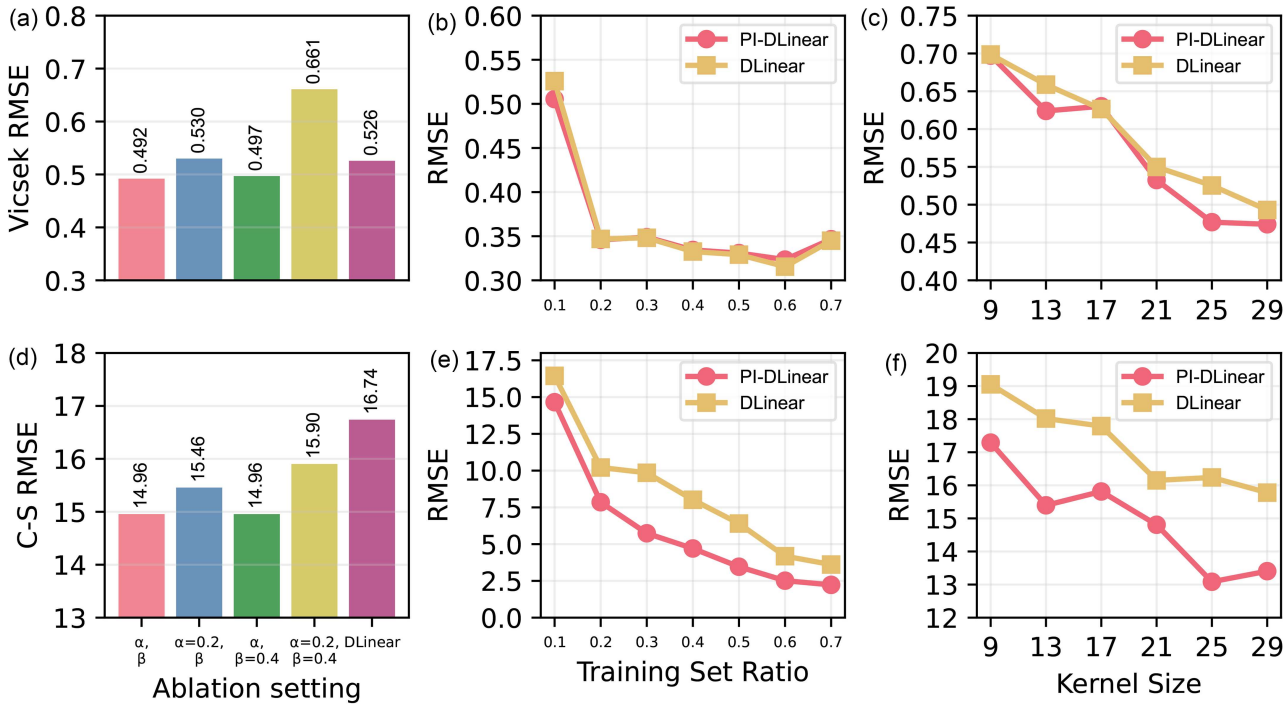

Figure 3 (Color online) Ablation experiments and parameter sensitivity experiments. (a) Ablation experiment on Vicsek model; (b) effect of different training set ratios on Vicsek model performance; (c) effect of different decomposition kernel sizes on Vicsek model performance; (d) ablation experiment on C-S model; (e) effect of different training set ratios on C-S model performance; (f) effect of different decomposition kernel sizes on C-S model performance.

patterns, and the errors of both models decrease as the kernel grows from 9 to 33. PI-DLinear maintains lower errors than DLinear at all kernel sizes on C-S (e.g., 13.74 versus 15.06 at $k = 33$) and at nearly all sizes on Vicsek (0.443 versus 0.479 at $k = 33$, where the two curves nearly coincide), indicating that the combination of physical mechanisms with the moving-average decomposition can more effectively extract dynamic features.

3.4 Real-world datasets

This study employs three high-resolution datasets of collective biological movements (jackdaw mobbing [9], jackdaw transit [9], and Hemigrammus rhodostomus circling [11]), covering collective behavior patterns across different ecological scenarios.

The jackdaw mobbing dataset documents high-resolution motion trajectories of jackdaws during collective anti-predator events. This behavior is characterized by individuals rapidly gathering and performing coordinated maneuvers to drive away predators. Its key movement features lie in high dynamism and mobility, with trajectories showing notably high curvature and frequent sharp turns. This reflects the fast, coordinated collective response of the group when facing immediate threats, serving as a typical model for studying group decision-making and dynamic self-organization in emergency situations.

The jackdaw transit dataset records the collective motion of jackdaws during long-distance migration. In this scenario, the group exhibits extremely high movement order (group order >0.8) and stability. Their trajectories are smooth and continuous, with very low curvature and stable speeds, showing no obvious sharp turns. This highly polarized and fluid movement pattern reflects the group's strategy for optimizing energy efficiency during long-distance navigation, providing ideal data for studying collective alignment in large-scale spatial navigation.

The Hemigrammus rhodostomus circling dataset records the circling collective state displayed by rummy-nose tetras under stress or specific environmental conditions (such as low light). In this mode, the school rotates as a whole around the center of the tank, with key quantitative features being high rotation (>0.7) and a strongly fluctuating polarization order. This dynamic equilibrium state of high cohesion with fluctuating directional alignment is considered a marker of the fish being in a critical or alert-ready state, serving as an important transitional model in theoretical studies that connects ordered schooling with disordered diffusive behavior.

To quantify the collective coordination degree of different biological swarms, we employ the polarization order

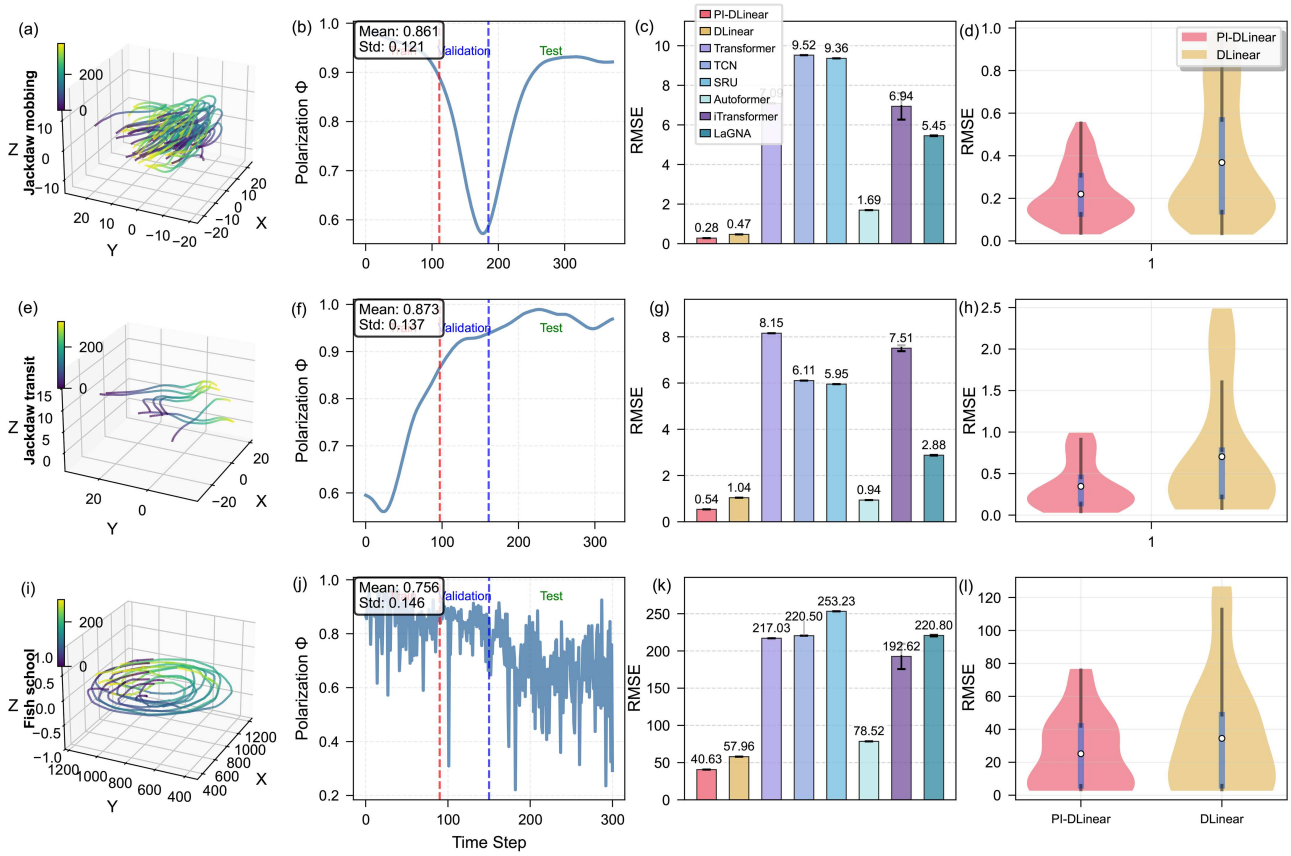

Figure 4 (Color online) Trajectories and prediction results for three real-world collective behaviors under small-sample conditions. (a) Trajectories of jackdaws engaged in mobbing behavior; (b) order parameter of the mobbing flock; (c) prediction results on mobbing; (d) error violin distribution of features for mobbing. (e) Trajectories of jackdaws transit; (f) order parameter of the transit flock; (g) prediction results on transit; (h) error violin distribution for transit. (i) Circling trajectories of the Hemigrammus rhodostomus school; (j) order parameter of the circling school; (k) prediction results on fish circling ; (l) error violin distribution for fish circling.

parameter Φ, defined as the normalized magnitude of the average velocity vector

$$\Phi = \frac{1}{N}\left\|\sum_{i=1}^{N}\hat{\mathbf{v}}_i\right\|, \quad \hat{\mathbf{v}}_i = \frac{\mathbf{v}_i}{\|\mathbf{v}_i\|}, \tag{29}$$

where $\hat{\mathbf{v}}_i$ denotes the unit velocity vector of agent i. This parameter ranges from 0 (random, uncorrelated motion) to 1 (perfect directional alignment). Figures 4(b), (f), and (j) present the temporal evolution of Φ for the three datasets, revealing distinct collective coordination patterns corresponding to different behavioral modes.

3.5 Results on three real datasets

By comparing with corresponding purely data-driven models, the generalization capability and robustness of the PI-DLinear method were investigated on three real-world datasets exhibiting different collective behavioral characteristics.

Figure 4(a) shows the trajectories of 53 jackdaws engaged in anti-predator (mobbing) behavior, characterized by high-frequency sharp turns and dynamic clustering. The trajectories exhibit high curvature and large variation in inter-individual distances, representing a "siege mode" driven by threats—requiring rapid velocity coordination and collision avoidance mechanisms. This corresponds to physical properties of strongly nonlinear interactions. Figure 4(e) illustrates the trajectories of 8 jackdaws migrating (transit) to a roost. The trajectories are highly ordered, with low curvature and stable velocities, and uniform inter-individual spacing. Attraction forces dominate, reflecting the need for global coordination in group directional flight. Figure 4(i) shows that the circling trajectories of the Hemigrammus rhodostomus school exhibit periodic circular motion, with individuals moving along nearly concentric paths. This reflects the interaction rule of "long-range attraction, short-range repulsion", a typical manifestation of dynamic equilibrium.

Table 2 RMSE results of different methods under different training ratios (re-run under the leakage-free protocol; mean over ten runs). The best result in each row is displayed in bold, and the second best result is shown with an underline.

Dataset	Ratio	Transformer	TCN	SRU	LaGNA	iTransformer	Autoformer	DLinear	PI-DLinear
		RMSE							
Mobbing	0.3	7.088	9.523	9.368	5.452	6.936	1.692	0.469	**0.281**
	0.4	6.785	8.096	7.069	3.796	6.514	1.660	0.364	**0.294**
	0.5	5.882	5.258	5.287	2.154	4.767	1.610	0.250	**0.207**
	0.6	5.011	4.698	4.801	1.044	3.529	1.478	0.280	**0.231**
	0.7	4.290	3.991	4.280	0.489	2.116	1.237	0.260	**0.218**
Transit	0.3	8.157	6.105	5.958	2.878	7.506	0.939	1.039	**0.536**
	0.4	7.466	5.033	5.668	2.504	5.362	0.842	0.859	**0.360**
	0.5	6.245	4.376	4.904	1.460	4.532	0.767	0.567	**0.303**
	0.6	5.796	3.766	4.226	0.852	3.521	0.708	0.509	**0.267**
	0.7	4.156	3.485	3.947	0.385	1.095	0.647	0.352	**0.264**
Fish	0.3	217.020	220.496	253.236	220.797	192.619	78.515	57.954	**40.627**
	0.4	188.908	177.546	213.853	184.318	164.931	81.972	30.001	**18.006**
	0.5	150.548	158.147	159.657	123.943	88.311	80.272	28.241	**16.787**
	0.6	140.406	143.910	138.103	84.782	45.529	88.707	25.264	**17.782**
	0.7	129.403	131.858	121.721	41.804	59.457	59.500	27.126	**22.602**

Under small-sample conditions, PI-DLinear consistently outperforms all baseline architectures across the three distinct behavioral regimes, as shown in Figures 4(c), (g), and (k). We use RMSE to quantify the difference in prediction accuracy between PI-DLinear and baseline models (DLinear, Transformer, TCN, SRU, Autoformer, iTransformer, and LaGNA) under small-sample conditions (the training set sizes of the three datasets are 20%, 30%, and 20%, respectively). The results validate the significant advantage of PI-DLinear across different behavioral modes, with performance improvements highly aligned with swarm dynamical characteristics. Figure 4(c) shows the prediction results for the jackdaw mobbing behavior data. The RMSE of PI-DLinear is only 0.28, which is 40.1% lower than that of DLinear (0.47), 83.4% lower than Autoformer (1.69), and clearly lower than the two newly added baselines iTransformer (6.94) and LaGNA (5.45), as well as an order of magnitude lower than Transformer (7.09), TCN (9.52), and SRU (9.36). The embedded physical priors, such as velocity coupling and repulsion constraints, effectively address the overfitting problem of purely data-driven models in highly nonlinear interaction scenarios. Figure 4(g) displays the prediction results for the jackdaw transit behavior. The RMSE of PI-DLinear is 0.54, which is 48.4% lower than DLinear (1.04) and 42.8% lower than Autoformer (0.94), and again lower than LaGNA (2.88), SRU (5.95), TCN (6.11), iTransformer (7.51), and Transformer (8.15). The model's physical mechanism of velocity alignment aligns with the global coordination needs of migratory behavior, reinforcing the consistency constraints of collective motion. Figure 4(k) shows the prediction results for the fish circling behavior. The RMSE of PI-DLinear is 40.63, which is 29.9% lower than DLinear (57.96), 48.3% lower than Autoformer (78.52), and lower than iTransformer (192.62), Transformer (217.03), LaGNA (220.80), TCN (220.50), and SRU (253.23). Its angular velocity coordination and distance-dependent interaction rules accurately match the dynamical essence of fish circling.

Figures 4(d), (h), and (l) use violin plots to display the RMSE distribution of PI-DLinear (red) versus the purely data-driven baseline model DLinear (yellow) for each feature. The error distribution of PI-DLinear is consistently more concentrated, with significantly lower mean values. This indicates that PI-DLinear not only reduces the overall prediction error but also decreases prediction variability across individuals and features—validating that physical priors guide the model to capture the inherent unified rules of swarm dynamics, enhancing prediction stability and reliability.

Table 2 presents the RMSE of each method on three real datasets. All experiments were conducted under the leakage-free split-then-window protocol described in Subsection 2.1. PI-DLinear attains the lowest RMSE in all 15 dataset × training-ratio settings, outperforming the strongest baseline in each setting by 16.2%–57.2%, with the largest gains in the low-data regime (e.g., 40.1% lower RMSE than the best baseline on mobbing and 42.9% on transit at ratio 0.3), supporting the claim that the physics priors act as an effective inductive bias when data are scarce. This indicates that by incorporating physical information, the model restricts the solution space to a subspace that satisfies physical feasibility, thereby reducing the demand for data samples. As shown in Table 2, iTransformer performs comparably to the strongest classical baselines but remains clearly behind PI-DLinear in all settings, and LaGNA, while the strongest gray-box baseline at high data ratios, is outperformed by PI-DLinear by 31.4%–55.4% at training ratio 0.7 and by larger margins in the low-data regime, where the discovered equations are

Table 3 Zero-shot cross-behavior transfer on real-world collective behavior data (RMSE, lower is better). Each entry reports the error when a model trained on the source behavior is directly evaluated on the target behavior without target-behavior training. The transfer protocol uses type-matched source-training normalization and reports predictions in the original physical scale (positions and velocities). Bold values indicate the lower RMSE between PI-DLinear and DLinear for each source-target pair. The diagonal is a within-behavior control under the same transfer protocol; the six off-diagonal entries are cross-behavior transfers. Results are averaged over five seeds.

Model	Source\target	Mobbing	Transit	Fish
PI-DLinear	Mobbing	**0.208**	**0.241**	**13.838**
	Transit	**0.192**	**0.260**	**14.683**
	Fish	**25.551**	**25.291**	**21.305**
DLinear	Mobbing	0.266	0.249	20.343
	Transit	0.220	0.339	19.751
	Fish	56.009	55.728	24.061

less stable. These results indicate that embedding interpretable interaction priors directly into a lightweight linear architecture is more effective—and far more parameter-efficient—than both recent general-purpose forecasters and equation-discovery approaches for collective behavior prediction.

In summary, by embedding physical priors of swarm dynamics, PI-DLinear achieves accurate prediction of different biological flocking behaviors. Its advantages stem from the deep alignment between the physical mechanisms and the characteristics of swarm dynamics, providing a paradigm for biological collective motion prediction that combines both accuracy and interpretability.

3.6 Generalization across behaviors, swarm sizes, and agent orderings

To directly evaluate cross-behavior generalization, we conduct a zero-shot transfer experiment in which a model trained on one real-world behavior is evaluated directly on another behavior without any target-behavior training. Because the three datasets differ in agent number, spatial dimension (3D for the jackdaw behaviors, 2D for fish), and physical scale, the transfer protocol uses type-matched source-training normalization: for each feature type (position or velocity in each coordinate dimension), the mean and standard deviation computed on the source training windows are aggregated across agents, and the target test inputs are normalized by matching feature types (for the 2D $\rightarrow$ 3D direction, the missing z-dimension statistics are filled by averaging the same-kind dimensions); predictions are then de-normalized with the same statistics, so no target-behavior statistics are ever used. All RMSE values are computed over the full state (positions and velocities) in the original physical scale of the target behavior. The source models are those of the main experiments (training ratio 0.7), so the transfer evaluation inherits the leakage-free windowing protocol in Subsection 2.1, and all numbers are averaged over five independent training seeds. The key comparison is against the architecture-matched DLinear baseline—the exact physics-free counterpart of PI-DLinear (identical architecture, capacity, and training pipeline, with only the physics-informed operator removed)—so that the comparison isolates the contribution of the interaction priors.

As shown in Table 3, PI-DLinear consistently outperforms the architecture-matched DLinear baseline on all six off-diagonal source-target pairs, reducing RMSE by 3.4%–54.6%, with an average reduction of 30.5%. For the two jackdaw behaviors, the zero-shot transfers in both directions are particularly effective: mobbing-to-transit RMSE is 0.2409 and transit-to-mobbing RMSE is 0.1916, both lower than the corresponding within-behavior DLinear results of 0.266 and 0.339—that is, PI-DLinear trained on a different behavior of the same species can surpass DLinear trained on the target behavior itself. For fish-school targets, PI-DLinear achieves 13.838 and 14.683 when transferred from mobbing and transit, respectively, compared with 20.343 and 19.751 for DLinear. Fish-to-jackdaw transfer remains more difficult because of the two-dimensional-to-three-dimensional and cross-species distribution shift; nevertheless, PI-DLinear reduces RMSE from 56.009 and 55.728 to 25.551 and 25.291, respectively. Controlled synthetic experiments (Appendix A) provide additional evidence and an honest boundary of the claim: PI-DLinear substantially improves the Cucker-Smale $\rightarrow$ Vicsek transfer (RMSE 6.231 $\rightarrow$ 0.666), while in the Vicsek $\rightarrow$ Cucker-Smale direction it is comparable to DLinear (7.612 versus 7.501). This indicates that the transfer benefit depends on the compatibility between the source and target dynamics and should not be interpreted as universal zero-shot superiority.

We next evaluate whether the same trained model can be applied to different swarm sizes. Figure 5 reports the real-data results when models trained on the complete swarm are evaluated on random subsets containing k agents. PI-DLinear remains stable as k decreases from 53, 8, and 10 to 1 for mobbing, transit, and fish, respectively, while maintaining lower RMSE than DLinear across the tested sizes. A stronger held-out-agent protocol further trains models using only 8, 4, and 5 agents for mobbing, transit, and fish, respectively, and evaluates them on disjoint agents, including test groups larger than the training group. As summarized in Supplementary Table B1,

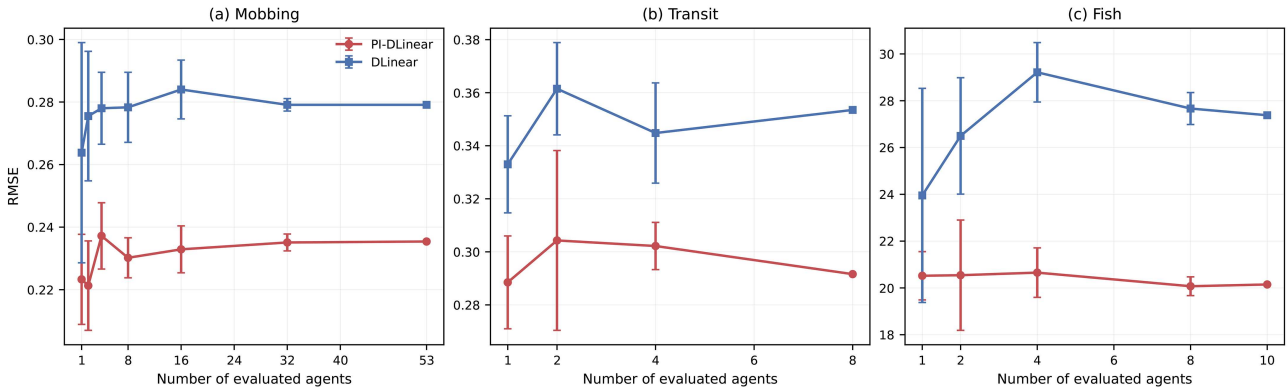

Figure 5 (Color online) Swarm-size robustness on real-world collective behavior data. Models are trained using the complete swarm and evaluated on random subsets containing k agents; error bars denote standard deviations over five random subsets.

PI-DLinear reduces RMSE by 17.6%–18.5% on mobbing, 22.3%–23.5% on transit, and 24.4%–26.3% on fish relative to DLinear. Finally, we assess sensitivity to agent indexing. Applying ten random agent permutations to a trained model changes the test RMSE by no more than 9×10^{-8} on the real datasets. Retraining from scratch under five random agent orders produces no systematic degradation, as reported in Table B2. These observations are consistent with the channel-shared architecture: permuting the input agent blocks permutes the output blocks without changing the prediction error.

4 Conclusion

We present PI-DLinear, a physics-informed decomposition linear neural network that integrates a learnable Cucker-Smale interaction kernel into a forecasting architecture. By treating physical parameters as trainable variables and embedding them as soft constraints, our framework overcomes the oversimplified assumptions of traditional mechanistic models and the lack of interpretability of purely data-driven approaches. The resulting model dynamically adapts to the underlying rules of some collective behaviors, such as mobbing, transit, and circling, without relying on fixed heuristics.

Evaluated on both synthetic swarms and three real-world biological datasets, PI-DLinear consistently outperforms state-of-the-art baselines in prediction accuracy, especially under small-sample and noisy conditions. The embedded physical priors not only reduce overfitting and improve robustness but also provide interpretable insights into how interaction forces are adaptively balanced across different behavioral modes. This demonstrates that combining interpretable physical equations with flexible neural networks establishes a promising paradigm for collective trajectory prediction. Our work provides a reference for verifying the existence of artificial physical rules in natural clusters. The added experiments directly demonstrate improved zero-shot transfer across three real collective behaviors, stable inference under variable swarm sizes, and invariance to agent-index permutations. Together with the controlled synthetic results, these findings support the value of embedding learnable Cucker-Smale alignment priors, while also indicating that cross-behavior transfer depends on the compatibility between the source and target dynamics.

The current physics prior is an alignment-dominant, first-order (velocity-coupling) model with homogeneous agents, and its benefits rely on the quality and relevance of the chosen prior: when the target behavior is not dominated by velocity alignment, the physics term degenerates gracefully but contributes little, as observed on the Vicsek swarms (Subsections 2.3 and 3.3). Despite these advances, several limitations remain. The current formulation assumes agent homogeneity and relies on the quality of the chosen physical prior. The dynamic models behind different species and their behaviors still need to be discovered. Moreover, emerging collective phenomena—such as swarmalators coupling phase synchronization with spatial clustering, or chimney swifts exhibiting opposite circling directions without full convergence—lie beyond the scope of homogeneous alignment models. Extending the framework to learnable multi-term or species-specific kernels is a promising direction for future work.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. U2230208, T2225022, U72401290), Major Scientific and Technological Innovation Platform Project of Hunan Province (Grant No. 2024JC1003), and Hunan Science and Technology Plan Project (Grant No. 2025JJ50417).

Supporting information Appendixes A–D. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains

entirely with the authors.

References

1 Zheng Z, Tao Y, Xiang Y, et al. Body orientation change of neighbors leads to scale-free correlation in collective motion. Nat Commun, 2024, 15: 8968
2 Cavagna A, Culla A, Feng X, et al. Marginal speed confinement resolves the conflict between correlation and control in collective behaviour. Nat Commun, 2022, 13: 2315
3 Zhou X, Wen X, Wang Z, et al. Swarm of micro flying robots in the wild. Sci Robot, 2022, 7: eabm5954
4 Amichay G, Li L, Nagy M, et al. Revealing the mechanism and function underlying pairwise temporal coupling in collective motion. Nat Commun, 2024, 15: 4356
5 Cavagna A, Cimarelli A, Giardina I, et al. Scale-free correlations in starling flocks. Proc Natl Acad Sci USA, 2010, 107: 11865–11870
6 Ballerini M, Cabibbo N, Candelier R, et al. Interaction ruling animal collective behavior depends on topological rather than metric distance: evidence from a field study. Proc Natl Acad Sci USA, 2008, 105: 1232–1237
7 Nagy M, Akos Z, Biro D, et al. Hierarchical group dynamics in pigeon flocks. Nature, 2010, 464: 890–893
8 Ling H, Mclvor G E, van der Vaart K, et al. Costs and benefits of social relationships in the collective motion of bird flocks. Nat Ecol Evol, 2019, 3: 943–948
9 Ling H, Mclvor G E, Westley J, et al. Behavioural plasticity and the transition to order in jackdaw flocks. Nat Commun, 2019, 10: 5174
10 Evangelista D J, Ray D D, Raja S K, et al. Three-dimensional trajectories and network analyses of group behaviour within chimney swift flocks during approaches to the roost. Proc R Soc B, 2017, 284: 20162602
11 Lin G, Escobedo R, Li X, et al. Experimental evidence of stress-induced critical state in schooling fish. PRX Life, 2025, 3: 033018
12 Peng H, Razi A, Afghah F, et al. A unified framework for joint mobility prediction and object profiling of drones in UAV networks. J Commun Netw, 2018, 20: 434–442
13 Cavagna A, Di Carlo L, Giardina I, et al. Natural swarms in 3.99 dimensions. Nat Phys, 2023, 19: 1043–1049
14 Múgica J, Torrents J, Cristín J, et al. Scale-free behavioral cascades and effective leadership in schooling fish. Sci Rep, 2022, 12: 10783
15 Reynolds C W. Flocks, herds and schools: a distributed behavioral model. SIGGRAPH Comput Graph, 1987, 21: 25–34
16 Couzin I D, Krause J, James R, et al. Collective memory and spatial sorting in animal groups. J Theor Biol, 2002, 218: 1–11
17 Vicsek T, Czirók A, Ben-Jacob E, et al. Novel type of phase transition in a system of self-driven particles. Phys Rev Lett, 1995, 75: 1226–1229
18 O'Keeffe K P, Hong H, Strogatz S H. Oscillators that sync and swarm. Nat Commun, 2017, 8: 1504
19 Liebchen B, Levis D. Collective behavior of chiral active matter: pattern formation and enhanced flocking. Phys Rev Lett, 2017, 119: 058002
20 Cucker F, Smale S. Emergent behavior in flocks. IEEE Trans Automat Contr, 2007, 52: 852–862
21 Cheng Q, Du L, Sang L. Two sparse control strategies for the delayed heterogeneous Cucker-Smale model. SIAM J Control Optim, 2025, 63: 1184–1212
22 Li X, Liu Y, Wu J. Flocking and pattern motion in a modified Cucker-Smale model. Bull Korean Math Soc, 2016, 53: 1327–1339
23 Jin B, Liang Y, Han Z, et al. Generating collective foraging behavior for robotic swarm using deep reinforcement learning. Artif Life Robotics, 2020, 25: 588–595
24 Costa T, Laan A, Heras F J H, et al. Automated discovery of local rules for desired collective-level behavior through reinforcement learning. Front Phys, 2020, 8: 200
25 Duan H, Huo M, Fan Y. From animal collective behaviors to swarm robotic cooperation. Natl Sci Rev, 2023, 10: 82–99
26 Li M, Chen S, Shen Y, et al. Online multi-agent forecasting with interpretable collaborative graph neural networks. IEEE Trans Neural Netw Learn Syst, 2022, 33: 5518–5531
27 Romero-Ferrero F, Heras F J H, Rance D, et al. A study of transfer of information in animal collectives using deep learning tools. Philos Trans R Soc B, 2023, 378: 20220073
28 Sun T, Fan L H, Yu X M, et al. FST-GNN: feedback space-time graph neural network model for networked multi-agent formation prediction with noise interference. IEEE Trans Netw Sci Eng, 2024, 11: 5983–5994
29 Casert C, Tamblyn I, Whitelam S. Learning stochastic dynamics and predicting emergent behavior using transformers. Nat Commun, 2024, 15: 1875
30 Hu J Q, Wang C N, Rakheja S, et al. Advanced trajectory prediction framework integrating diverse driving styles for autonomous vehicles. Sci China Inf Sci, 2026, 69: 132202
31 Brunton S L, Proctor J L, Kutz J N. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proc Natl Acad Sci USA, 2016, 113: 3932–3937
32 Gao T T, Yan G. Autonomous inference of complex network dynamics from incomplete and noisy data. Nat Comput Sci, 2022, 2: 160–168
33 Gao T T, Barzel B, Yan G. Learning interpretable dynamics of stochastic complex systems from experimental data. Nat Commun, 2024, 15: 6029
34 Raissi M, Perdikaris P, Karniadakis G E. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys, 2019, 378: 686–707
35 Xu P F, Han C B, Cheng H X, et al. A physics-informed neural network for the prediction of unmanned surface vehicle dynamics. JMSE, 2022, 10: 148
36 Long K, Sheng Z, Shi H, et al. Physical enhanced residual learning (PERL) framework for vehicle trajectory prediction. Commun Transp Res, 2025, 5: 100166
37 Liu Z, Liu Y, Yan X, et al. Automatic network structure discovery of physics informed neural networks via knowledge distillation. Nat Commun, 2025, 16: 9558
38 Kim D, Lee J, Kim H Y. Commanding emergent behavior with neural networks. Cell Rep Phys Sci, 2025, 6: 102857
39 Zeng A, Chen M, Zhang L, et al. Are Transformers effective for time series forecasting? In: Proceedings of the AAAI Conference on Artificial Intelligence, 2023. 11121–11128
40 Mohammadi F R, Pazouki E. A transformer self-attention model for time series forecasting. J Electr Comput Eng Innov, 2021, 9: 1-10
41 Jia P, Chen Z, Mao G, et al. Gas concentration prediction based on temporal attention mechanism in temporal convolutional networks. Sens Actuat B-Chem, 2025, 433: 137562
42 Lei T, Zhang Y, Wang S I W, et al. Simple recurrent units for highly parallelizable recurrence. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 2018. 4470–4481
43 Wu H, Xu J, Wang J, et al. Autoformer: decomposition transformers with auto-correlation for long-term series forecasting. In: Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS), 2021. 22419–22430
44 Liu Y, Hu T, Zhang H, et al. iTransformer: inverted transformers are effective for time series forecasting. In: Proceedings of the International Conference on Learning Representations, 2024
45 Cucker F, Dong J G. A conditional, collision-avoiding, model for swarming. Discrete Cont Dyn, 2014, 34: 1009–1020
46 Erban R, Haskovec J, Sun Y. A Cucker-Smale model with noise and delay. SIAM J Appl Math, 2016, 76: 1535–1557
47 Cheng J, Ru L, Wang X, et al. Collision-avoidance, aggregation and velocity-matching in a Cucker-Smale-type model. Appl Math Lett, 2022, 123: 107611