OPEN ACCESS

ARTICLE

INNOVATION PRESS

Journal homepage: www.the-innovation.org/ai-plus

AI Plus

AI Plus 1 (2026):100006

Contents lists available at INNOVATION PRESS

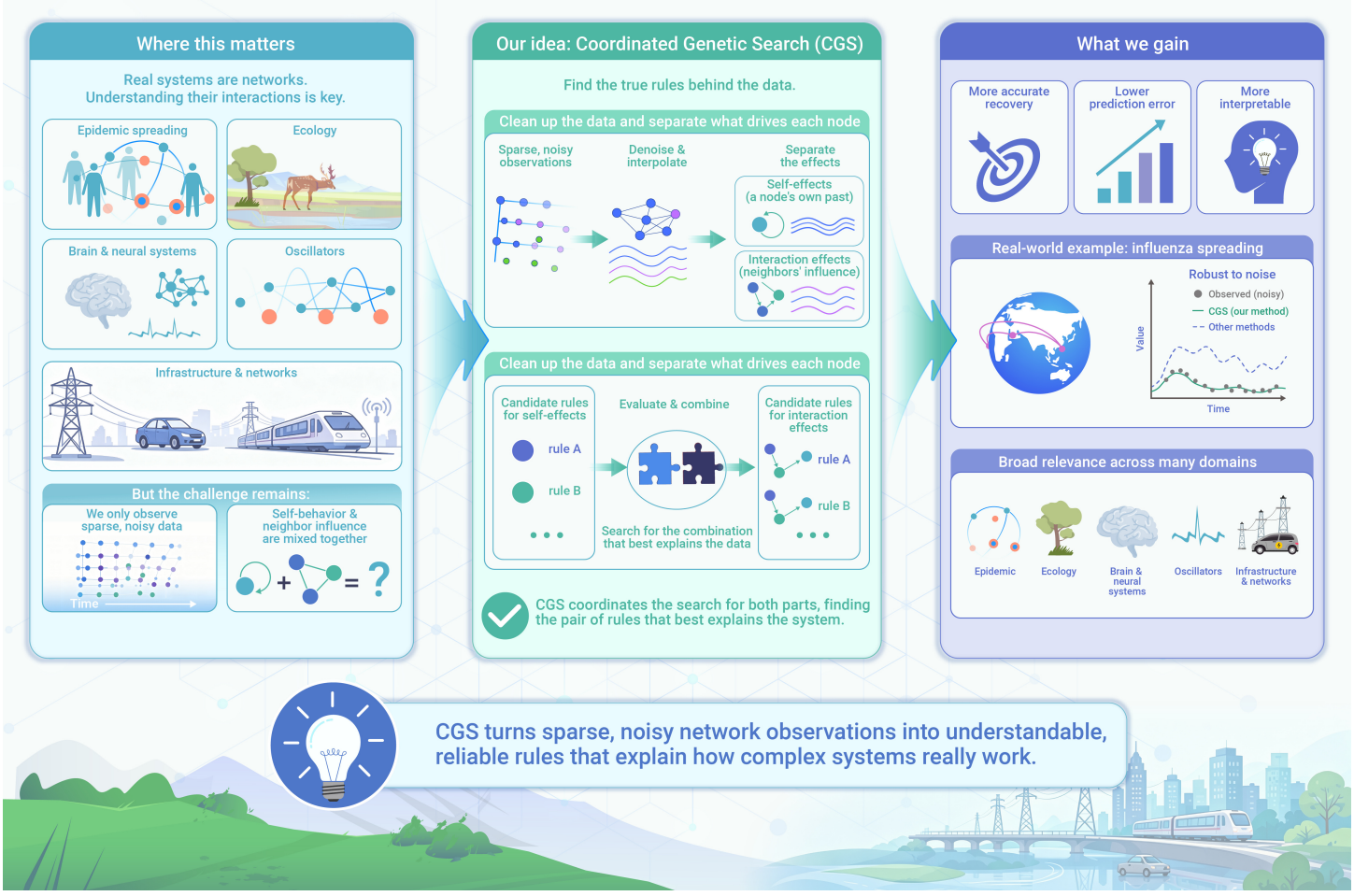
Coordinated discovery of mechanistic dynamics in complex networks

Haiquan Qiu,[1,3] Shuzhi Liu,[2,3] You Wu,[1] Yong Li,[1] and Quanming Yao[1,*]

*Correspondence: qyaoaa@tsinghua.edu.cn

Received: May 11, 2026; Accepted: August 15, 2026; Published Online: August 18, 2026; https://doi.org/10.59717/ipj.aiplus.2026.100006

GRAPHICAL ABSTRACT



PUBLIC SUMMARY

- CGS discovers symbolic laws for network dynamics from sparse noisy data.
- Neural ODE proxies denoise trajectories and separate node and edge effects.
- Coordinated search reduces compensatory overfitting between dynamic components.
- CGS shows strong synthetic recovery and provides an influenza case study.

ARTICLE

OPEN ACCESS

INNOVATION PRESS

Journal homepage: www.the-innovation.org/ai-plus

AI Plus

AI Plus 1 (2026):100006

Contents lists available at INNOVATION PRESS

Coordinated discovery of mechanistic dynamics in complex networks

Haiquan Qiu,[1,3] Shuzhi Liu,[2,3] You Wu,[1] Yong Li,[1] and Quanming Yao[1,*]

[1]Department of Electronic Engineering, Tsinghua University, Beijing 100084, China

[2]School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798, Singapore

[3]These authors contributed equally

*Correspondence: qyaoaa@tsinghua.edu.cn

Received: May 11, 2026; Accepted: August 15, 2026; Published Online: August 18, 2026; https://doi.org/10.59717/ipj.aiplus.2026.100006

Citation: Qiu H., Liu S., Wu Y., et al. (2026). Coordinated discovery of mechanistic dynamics in complex networks. *AI Plus* **1**:100006.

Symbolic regression offers a route to mechanistic understanding of complex network dynamics, but existing methods often infer node and edge equations independently, allowing errors in one component to be compensated by the other. We present Coordinated Genetic Search (CGS), a framework for discovering governing equations from sparse and noisy network observations. CGS first trains a graph neural ordinary differential equation proxy with network inductive bias to disentangle node and edge effects and reconstruct denoised trajectories. It then co-evolves symbolic populations for node and edge dynamics, adaptively prioritizing the component that deviates most from its neural reference. This coordination is designed to reduce compensatory overfitting and improve identifiability. Across synthetic epidemic, ecological, neural, and oscillator systems on multiple graph topologies, CGS provides strong evidence of improved symbolic recovery and trajectory prediction relative to the tested baselines. On a real influenza A spreading network, where no ground-truth governing equation is available, CGS provides a proof-of-concept case study with physically plausible expressions and a modest reduction in normalized prediction error.

INTRODUCTION

Complex networks [1–4] are the fabric of our interconnected world, from the intricate web of social interactions [5] and the pathways of global pandemics [6,7] to the complex wiring of the human brain [8,9]. Understanding how these networks evolve is a central challenge in modern science [10–12]. The ultimate goal is not just to observe their dynamics, but to distill them into concise, symbolic formulas [13–20], i.e., the fundamental laws that govern their behavior. Like elegant physical equations, these symbolic expressions offer a clear window into the underlying mechanisms of a system, enabling us to predict its future and understand its core principles [21–25].

A promising approach is neural symbolic regression [19,23,24], which leverages deep learning models with inductive biases to discover symbolic laws. In networked systems, recent work has combined neural fitting or pretrained symbolic regressors with physically motivated decompositions of self dynamics and interactions, enabling interpretable equation discovery from high-dimensional observations [26,27]. Such neural symbolic methods are robust to noise and irregular sampling, making them more suitable for real-world data than traditional genetic programming [16,17] or sparse regression techniques [12,18]. However, a critical challenge remains. With this inductive bias, the dynamics of a node in a complex network are modeled as a composite of two distinct processes: neural node dynamics (how a node acts on its own) and neural edge dynamics (how it interacts with its neighbors) [28]. Existing methods often search for these components' formulas independently [23,24], which can lead to a critical form of overfitting: errors in one component are compensated for by an inaccurate expression for the other. This results in models that are neither interpretable nor generalizable.

To address this, we introduce **Coordinated Genetic Search (CGS)** (Fig. 1), a novel search algorithm that co-evolves symbolic expressions for both node and edge dynamics. CGS operates on the principle that these components should be discovered cooperatively to reduce compensatory overfitting, where errors in one component are offset by an overly complex expression for the other. The algorithm first trains a disentangled neural proxy model to generate references for each neural dynamics component and to provide denoised, interpolated trajectories for fitness evaluation. It then maintains two distinct populations of symbolic expressions—one for node dynamics and one for edge dynamics—and coordinates their evolution by prioritizing the population that deviates most from its neural reference. This coordinated process, guided by neural references and evaluated against the interpolated trajectories, helps steer the search toward simpler and more faithful symbolic models. We evaluate CGS on synthetic dynamics and on a real-world disease-spreading dataset. The synthetic experiments provide strong evidence for improved formula recovery and prediction accuracy under controlled ground-truth settings, while the real-data experiment should be interpreted cautiously as a plausibility case study because real-world systems lack known governing equations.

Notations Matrix, vector, and scalar are denoted as bold capital letters $\mathbf{X}$, bold lowercase letters $\mathbf{x}$, and lowercase letters x, respectively. The element in the i-th row and j-th column of matrix $\mathbf{X}$ is denoted as $\mathbf{X}_{ij}$. The v-th row of matrix $\mathbf{X}$ is denoted as $\mathbf{x}_v$. A complex network is denoted as $\mathbf{G} = (\mathsf{G}, \mathbf{X}(t), t \in \mathcal{T})$. $\mathsf{G} = (V, E)$ denotes a network with node set V and edge set E. $\mathbf{X}(t) = [\mathbf{x}_1(t)^\top, \cdots, \mathbf{x}_N(t)^\top]^\top \in \mathbb{R}^{N\times d}$ is a d-dimensional node state matrix of N nodes at timestamp t, and $\mathcal{T} = \{t_0, t_1, \cdots, t_{K_\mathcal{T}-1}\}$ is the set of $K_\mathcal{T}$ observed timestamps. $\dot{\mathbf{x}}(t)$ represents the time derivatives of $\mathbf{x}(t)$.

MATERIALS AND METHODS

This section details our Coordinated Genetic Search (CGS) algorithm and is organized as follows. Neural proxy model introduces a neural proxy model that provides reliable references for node and edge dynamics, along with denoised and interpolated trajectories. Coordination via neural references explains how these neural references are used to coordinate the search, and Fitness evaluation and evolution describes the fitness evaluation and evolution process. Finally, Comparison compares CGS with existing methods.

Neural proxy model

Inductive Bias of Complex Network Dynamics. The complex network dynamics is defined by the following ordinary differential equation:

$$\dot{\mathbf{x}}_v(t) = F(\mathbf{x}_v(t)) + \textstyle\sum_{u\in\mathcal{N}_v} a_{vu} G(\mathbf{x}_v(t), \mathbf{x}_u(t)), \tag{1}$$

In (1), $\dot{\mathbf{x}}_v(t)$ denotes the time derivative of $\mathbf{x}_v(t)$, $F(\mathbf{x}_v(t))$ denotes the node dynamics term of node v, which includes processes like influx, degradation, or reproduction. $G(\mathbf{x}_v(t), \mathbf{x}_u(t))$ is the edge dynamics describing the interactions between node v and node u, accounting for processes such as spreading and competition. G is shared across all edges in the network because of the universality in network dynamics [2,29]. a_{vu} denotes the adjacency entry for the edge from source node u to target node v, and $\mathcal{N}_v$ is the set of neighbors of node v. In the synthetic ER/BA experiments, we use binary adjacency, so $a_{vu} = 1$ for existing edges and $a_{vu} = 0$ otherwise; the summation over $\mathcal{N}_v$ therefore implicitly includes only nonzero adjacency entries. In the influenza aviation network, edges are directed and weighted. For a directed passenger-

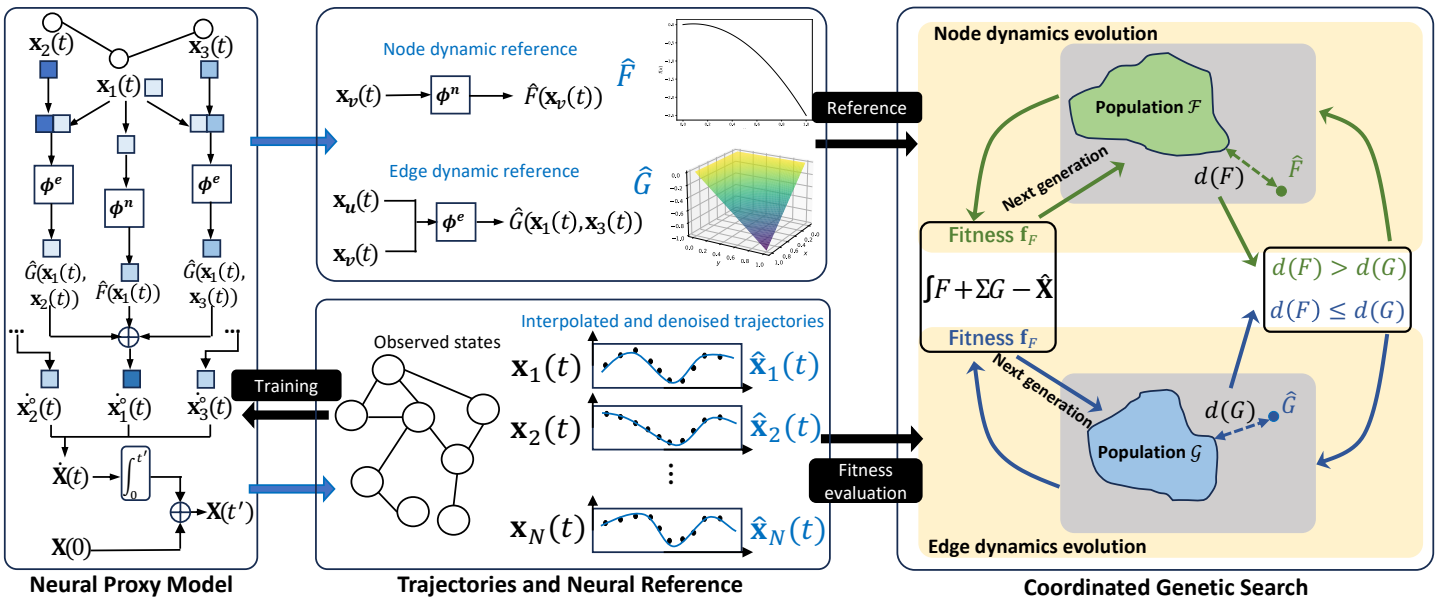


Figure 1. Overview of Coordinated Genetic Search (CGS). A neural proxy model provides references ($\hat{F}, \hat{G}$) and trajectories ($\hat{X}$) to guide the co-evolution of symbolic populations for node and edge dynamics. The search prioritizes the population deviating most from its reference to find accurate symbolic expressions (F^*, G^*).

flow edge $j \to i$, where i is the target region and j is the source region, we use the normalized weighted adjacency $\hat{A}_{ij}$, obtained by normalizing passenger volume with population information and total daily passenger volume. This corresponds to $v = i$ and $u = j$ in (1), so the influenza edge contribution is $\sum_j \hat{A}_{ij} G(\mathbf{x}_i(t), \mathbf{x}_j(t))$. Given observations of the node states $\{\mathbf{X}(t) | t \in \mathcal{T}\}$, the symbolic regression of complex network dynamics [12] aims to find the symbolic expressions of functions F and G in (1).

Neural Proxy Model with Inductive Bias. We follow the inductive bias from (1) to design a neural proxy model and train the model based on the observed trajectory. In the designed neural proxy, a graph neural network calculates the time derivative of the node state $\dot{\mathbf{x}}_v^\circ(t)$. An ODESolver then integrates this derivative to generate the full trajectory. Based on DNND [28], the time derivative $\dot{\mathbf{x}}_v^\circ(t)$ for node v is designed as an encoder-decoder free architecture:

$$\dot{\mathbf{x}}_v^\circ(t) = \hat{F}(\mathbf{x}_v(t)) + \sum_{u \in \mathcal{N}_v} \hat{G}(\mathbf{x}_v(t), \mathbf{x}_u(t)) \tag{2}$$

$$\text{where } \hat{F}(\mathbf{x}_v(t)) = \phi^n(\mathbf{x}_v(t), t), \hat{G}(\mathbf{x}_v(t), \mathbf{x}_u(t)) = \phi^e(\mathbf{x}_v(t), \mathbf{x}_u(t), t). \tag{3}$$

where ϕ^n and ϕ^e are two MLPs aligning with the node dynamics and edge dynamics in (1), respectively. For the weighted influenza network, the neural proxy uses the normalized weighted adjacency,

$$\dot{\mathbf{x}}_i^\circ(t) = \hat{F}(\mathbf{x}_i(t)) + \sum_j \hat{A}_{ij} \hat{G}(\mathbf{x}_i(t), \mathbf{x}_j(t)).$$

For binary synthetic graphs, the proxy uses the unweighted neighbor summation in (2). In (2), the neural node dynamics ϕ^n captures the evolution of nodes influenced by their properties, and the neural edge dynamics ϕ^e captures the interactions between two end nodes of an edge. Therefore, the proxy model of node v is written as

$$f_\theta(\mathbf{G}, \mathbf{X}(t_0), t)_v = \text{ODESolver}(\dot{\mathbf{x}}_v^\circ(t), \mathbf{X}(t_0), t_0, t). \tag{4}$$

The alignment between neural dynamics (4) and dynamics formulation (1) enables better learning of complex network dynamics. To train the neural dynamics, we minimize the error between $f_\theta(\mathbf{G}, \mathbf{X}(t_0), t)_v, \forall v \in V$ and the observed trajectories $\{\mathbf{X}(t) | t \in \mathcal{T}\}$:

$$\min_\theta \sum_{v \in V, t \in \mathcal{T}} \| f_\theta(\mathbf{G}, \mathbf{X}(t_0), t)_v - \mathbf{x}_v(t) \|_1. \tag{5}$$

The optimization is performed with standard deep learning techniques. After the training, we use the estimated node dynamics and edge dynamics as references for the coordinated search and use the interpolated trajectory $\hat{\mathbf{X}}(t)$ from $f_\theta(\mathbf{G}, \mathbf{X}(t_0), t)_v, \forall v \in V$ as the signal for fitness evaluation. Let $\mathcal{T}_{\text{int}}$ denote the interpolated timestamps used during symbolic search.

Coordination via neural references

Although the neural proxy model provides disentangled estimates for node dynamics ($\hat{F}$) and edge dynamics ($\hat{G}$), these components are not perfectly accurate. During training, errors in one component can be compensated for by the other, resulting in a model that fits the overall trajectory but misrepresents the individual dynamics. Therefore, using these neural components for direct supervision in separate searches for symbolic F and G would risk replicating this overfitting. Instead, CGS uses them as references to coordinate the evolution of two symbolic populations, ensuring a balanced search.

CGS maintains two populations of symbolic expressions: $\mathcal{F}$ for node dynamics and $\mathcal{G}$ for edge dynamics. To prevent one population from overfitting to compensate for inaccuracies in the other, their evolution is coordinated. At each step, CGS measures the deviation of each population from its respective neural reference. The evaluation points are sampled from the proxy-interpolated training trajectories:

$$\mathcal{S}_F = \{\hat{\mathbf{x}}_v(t) : v \in V,\ t \in \mathcal{T}_{\text{int}}\}, \tag{6}$$

$$\mathcal{S}_G = \{(\hat{\mathbf{x}}_v(t), \hat{\mathbf{x}}_u(t)) : v \in V,\ u \in \mathcal{N}_v,\ t \in \mathcal{T}_{\text{int}}\}. \tag{7}$$

Node distances are therefore evaluated on single-node states, while edge distances are evaluated on ordered node-neighbor state pairs. For a candidate expression, we define

$$\Delta_F(F) = \frac{1}{|\mathcal{S}_F|} \sum_{\mathbf{z} \in \mathcal{S}_F} \left\| F(\mathbf{z}) - \hat{F}(\mathbf{z}) \right\|_1, \tag{8}$$

$$\Delta_G(G) = \frac{1}{|\mathcal{S}_G|} \sum_{(\mathbf{z}, \mathbf{z}') \in \mathcal{S}_G} \left\| G(\mathbf{z}, \mathbf{z}') - \hat{G}(\mathbf{z}, \mathbf{z}') \right\|_1. \tag{9}$$

The population-level coordination distances are

$$d(\mathcal{F}) = \frac{1}{|\mathcal{F}|} \sum_{F \in \mathcal{F}} \Delta_F(F)^2, \quad d(\mathcal{G}) = \frac{1}{|\mathcal{G}|} \sum_{G \in \mathcal{G}} \Delta_G(G)^2. \tag{10}$$

In the current implementation, both distances are computed on samples drawn from the same training trajectory distribution and no additional scale normalization is applied beyond averaging over the sampled node or edge inputs. We found this practical for the tested systems. A more systematic normalization across components is an important direction for future work. The algorithm then prioritizes the evolution of the population with the larger distance to its reference. For instance, if $d(\mathcal{F}) > d(\mathcal{G})$, only the node dynam-

ics population $\mathcal{F}$ is evolved. This strategy encourages a balanced search and reduces the risk that one component compensates for errors in the other.

Fitness evaluation and evolution

While neural dynamics provide references for coordination, they are not precise enough for direct fitness calculation. Instead, CGS uses the denoised and interpolated trajectories from the proxy model as the target signal for fitness evaluation.

To evolve the selected population, we assess the fitness of each candidate expression. The fitness error of a symbolic node dynamics $F \in \mathcal{F}$ or a symbolic edge dynamics $G \in \mathcal{G}$ is calculated by pairing it with expressions from the other population and measuring the trajectory error against the interpolated trajectory:

$$f_{F,G}^{v,t} = \mathbf{x}_v(t_0) + \int_{t_0}^{t} \left(F(\mathbf{x}_v(\tau)) + \sum_{u \in \mathcal{N}_v} G(\mathbf{x}_v(\tau), \mathbf{x}_u(\tau))\right) d\tau, \tag{11}$$

$$f_{\theta}^{v,t} = f_{\theta}\left(\mathbf{G}, \mathbf{X}(t_0), t\right)_v, \tag{12}$$

$$\mathcal{E}(F,G) = \sum_{v \in V,\ t \in \mathcal{T}_{\text{int}}} e(f_{F,G}^{v,t}, f_{\theta}^{v,t}), \tag{13}$$

$$\mathrm{f}_F(F) = \mathbf{Mean}\left(\mathbf{BestK}_{K_{\text{top}}}\{\mathcal{E}(F,G) \mid G \in \mathcal{G}\}\right), \tag{14}$$

$$\mathrm{f}_G(G) = \mathbf{Mean}\left(\mathbf{BestK}_{K_{\text{top}}}\{\mathcal{E}(F,G) \mid F \in \mathcal{F}\}\right). \tag{15}$$

For the influenza weighted network, the same expression is evaluated by replacing the edge sum in $f_{F,G}^{v,t}$ with

$$\sum_j \hat{A}_{ij} G(\mathbf{x}_i(\tau), \mathbf{x}_j(\tau)).$$

For binary synthetic graphs, the corresponding edge sum uses binary adjacency entries. Here, $e(\cdot,\cdot)$ is the error function, $\mathbf{BestK}_{K_{\text{top}}}$ selects the K_{top} lowest-error pairings, and $\mathbf{Mean}$ averages their errors. A lower fitness error indicates a better symbolic expression. The expressions in the selected population then undergo selection, crossover, and mutation to generate the next generation.

Computational complexity Let P_F and P_G denote the node- and edge-population sizes, M the maximum number of generations, $|\mathcal{T}_{\text{int}}|$ the number of interpolated timestamps, and $|E|$ the number of edges. A naive pairwise evaluation of candidate node and edge expressions has complexity $O(MP_FP_G|\mathcal{T}_{\text{int}}|(|V|+|E|))$, excluding the expression-evaluation cost. In practice, the top-pairing strategy with K_{top} reduces the number of effective pair evaluations used to score each candidate, but CGS remains more computationally expensive than independent symbolic searches because candidate node and edge expressions are evaluated cooperatively.

The complete CGS algorithm is detailed in Fig. 2(a). The process begins by initializing populations $\mathcal{F}^{(0)}$ and $\mathcal{G}^{(0)}$. In each iteration, the algorithm decides which population to evolve based on (10). It then calculates fitness, checks for convergence, and applies genetic operators. The search terminates when a satisfactory solution is found or the maximum number of iterations is reached, returning the best-fit symbolic expressions F^* and G^*.

Comparison

We compare SymDL [23], NASSymDL [24], D-CODE [19], TP-SINDy [12] and CGS in 3(b). These methods cater to different problem settings, utilizing distinct forms of input and output. SymDL and NASSymDL perform general symbolic regression, finding a function $y = f(x)$ from input-output pairs $(x_i, y_i)_{i=1}^N$. D-CODE focuses on *symbolic regression of dynamics*, taking a single trajectory $\{x(t)|t \in \mathcal{T}\}$ to output the governing ODE $\dot{x} = dx/dt$. TP-SINDy and CGS target *symbolic regression of complex network dynamics*, using multiple trajectories $\{\mathbf{X}(t)|t \in \mathcal{T}\}$ to output symbolic network dynamics F and G.

We compare these algorithms in two aspects: proxy models and formula regression. Proxy models are trained to fit data and serve as a basis for deriving symbolic expressions. Formula regression directly extracts symbolic expressions from raw data or proxy models. For proxy model, SymDL uses graph networks (GN) with inductive bias, and NASSymDL employs neural architecture search (NAS) for skeleton search. D-CODE can incorporate any suitable regressor. CGS fits multiple trajectories using proxy, a graph neural ODE aligned with network dynamics for better generalization. SymDL and NASSymDL rely on potentially noisy estimated derivatives for dynamics fitting, whereas D-CODE and CGS train directly on raw observations for improved accuracy. TP-SINDy needs no proxy model.

In formula regression, methods using genetic search employ elementary operations, including addition, subtraction, multiplication, division, sine, and exponential, to represent formulas. This offers more flexibility and requires less prior knowledge than TP-SINDy's linear combination of functions in a predefined function library. SymDL and NASSymDL use the internal functions [23] from the proxy models as supervision to compute fitness in genetic search. D-CODE uses the interpolated trajectories as supervision. TP-SINDy is based on sparse regression and uses estimated derivatives for symbolic regression, which can be noisy and inaccurate over large time intervals. CGS uses (3) from the proxy model as references for search coordination and interpolated trajectories for fitness evaluation.

While our method co-evolves separate populations, it differs from general-purpose techniques like the Cooperative Co-evolutionary Genetic Algorithm (CCGA) [30]. CCGA is a generic optimization framework that decomposes a problem and evolves sub-populations in a fixed, round-robin schedule, agnostic to the individual performance of each component. CGS is designed for symbolic discovery in network dynamics, where compensatory overfitting between node and edge terms is a central concern. Its adaptive, reference-guided coordination strategy prioritizes the dynamic component (node or edge) that deviates most from its neural reference. This targeted approach is intended to reduce component-wise compensation when discovering physically meaningful governing equations.

Theoretical analysis

Our theoretical analysis (see full proof in Section S3 of the Supplemental Information) clarifies why CGS can be less sensitive than a Separate Search (SS) baseline such as SymDL to component-wise proxy errors. We show that for the SS bound to be small, the neural proxy must have small component-wise errors for both node and edge dynamics. In contrast, the CGS bound depends on the overall trajectory error of the proxy. This distinction aligns with the central motivation of our paper: CGS is designed for settings where errors in the proxy's components can compensate for each other even when the overall trajectory fit is accurate. This is formalized in the following theorem.

Theorem 1 (Error Bounds for CGS vs. SS). *Let $X_{gt}(t)$ be the ground-truth trajectory. Under the bounded neural proxy error, Lipschitz continuity, and bounded search error assumptions stated in Section S3 of the Supplemental Information, the ground-truth fitting errors for CGS (E_{CGS}) and Separate Search (E_{SS}) are bounded as follows:*

$$E_{\text{CGS}} = \|X_{F_{\text{CGS}},G_{\text{CGS}}}(t) - X_{gt}(t)\| \le \delta_{\text{CGS}} + \varepsilon_{\text{proxy}} \tag{16}$$

$$E_{\text{SS}} = \|X_{F_{\text{SS}},G_{\text{SS}}}(t) - X_{gt}(t)\| \le L_F(\delta_F + \varepsilon_F) + L_G(\delta_G + \varepsilon_G) \tag{17}$$

where $\varepsilon_{\text{proxy}}$ is the neural proxy's overall trajectory error, while ε_F and ε_G are the errors of its individual node and edge dynamic components. The δ terms represent search algorithm errors, and L_F, L_G are Lipschitz constants that reflect the sensitivity of the system to change in F and G.

Implication of the Theorem. The theorem highlights a key difference: for SS to achieve a small error, it requires both ε_F and ε_G to be small—a much stronger condition than simply requiring a small $\varepsilon_{\text{proxy}}$ as in CGS. In practice, overfitting often leads to large component-wise errors that could cancel out in the overall trajectory, so the SS bound can be much looser than the CGS bound. Thus, CGS is more robust to this type of overfitting, which is central to the motivation of our approach. Details of the assumptions, theorem proof, and implications are provided in Section S3 of the Supplemental Information.

RESULTS

Effective symbolic regression with Coordinated Genetic Search

Extracting governing equations from real-world complex networks presents fundamental challenges for symbolic regression. Observations are often sparse, noisy, and irregularly sampled, rendering traditional numerical

(a)

Require: Neural dynamics f_θ, node dynamics reference $\hat{F}$, edge dynamics reference $\hat{G}$, top-pair parameter K_{top}, maximum iteration M, threshold ϵ.

Initialize the node dynamics population $\mathcal{F}^{(0)}$ and edge dynamics population $\mathcal{G}^{(0)}$ with random symbolic expressions;
for $i = 1, 2, \cdots, M$ **do**
  Compute $d(\mathcal{F}^{(i-1)})$ and $d(\mathcal{G}^{(i-1)})$ using the coordination-distance definitions;
  if $d(\mathcal{F}^{(i-1)}) > d(\mathcal{G}^{(i-1)})$ **then**
    Calculate the fitness error f_F of each expression F in $\mathcal{F}^{(i-1)}$ using the node-fitness objective;
    if $\exists F \in \mathcal{F}^{(i-1)}, \mathrm{f}_F \leq \epsilon$, **break**;
    Select, cross, and mutate the expressions in $\mathcal{F}^{(i-1)}$ to generate the next population $\mathcal{F}^{(i)}$;
    $\mathcal{G}^{(i)} = \mathcal{G}^{(i-1)}$;
  else
    Calculate the fitness error f_G of each expression G in $\mathcal{G}^{(i-1)}$ using the edge-fitness objective;
    if $\exists G \in \mathcal{G}^{(i-1)}, \mathrm{f}_G \leq \epsilon$, **break**;
    Select, cross, and mutate the expressions in $\mathcal{G}^{(i-1)}$ to generate the next population $\mathcal{G}^{(i)}$;
    $\mathcal{F}^{(i)} = \mathcal{F}^{(i-1)}$;
  end if
end for
$F^* = \arg\min_{F \in \mathcal{F}^{(i)}} \mathrm{f}_F$;
$G^* = \arg\min_{G \in \mathcal{G}^{(i)}} \mathrm{f}_G$;
return F^*, G^*.

(b)

Category	Design	SymDL	NASSymDL	D-CODE	TP-SINDy	CGS
Input		input-output pairs	input-output pairs	single trajectory	multiple trajectories	multiple trajectories
Proxy model	Model design	GN w/ inductive bias	NAS	any regressor	–	GN w/ inductive bias
	Dynamics fitting data	estimated derivatives	estimated derivatives	raw observations	–	raw observations
Formula regression	Prior knowledge	elementary operation	elementary operation	elementary operation	function library	elementary operation
	Method	GS	GS	GS	sparse regression	coordinated GS
	Supervision	internal functions	internal functions	interpolated trajectory	estimated derivatives	network ref & interp. trajectories
Output		input-output mapping	input-output mapping	ODE	Graph ODE	Graph ODE

Figure 2. (a) Algorithm for Coordinated Genetic Search for symbolic regression. (b) Comparison with different methods for symbolic regression. (GN: graph network, NAS: neural architecture search, GS: genetic search)

differentiation unreliable. More critically, the dynamics of nodes and edges are intrinsically coupled: a node's state evolution is an aggregate of its self-dynamics and interactions with neighbors. Without a mechanism to disentangle these components, search algorithms face an identifiability crisis, often discovering complex edge formulas that erroneously compensate for misspecified node dynamics (or vice versa). Such models may fit the training data but fail to capture the underlying physical laws.

To resolve these issues of data quality and dynamic coupling, we propose Coordinated Genetic Search (CGS). As illustrated in Fig. 1, CGS proceeds in two integrated phases: neural proxy modeling and coordinated symbolic search. First, we train a neural proxy parameterized by graph neural ordinary differential equations (GN-ODEs). By incorporating the structural inductive bias of network dynamics, this model effectively disentangles the learned behavior into neural node functions ($\hat{F}$) and edge functions ($\hat{G}$). Crucially, the proxy acts as a robust interpolator, reconstructing dense, denoised trajectories from sparse observations to enable accurate derivative estimation.

Subsequently, we introduce a coordinated mechanism to discover the symbolic expressions F^* and G^*. Unlike traditional approaches that treat these components independently or sequentially, CGS co-evolves two populations of symbolic expressions. To reduce the emergence of compensatory but physically incorrect formulas, the algorithm leverages the neural references $\hat{F}$ and $\hat{G}$ to guide the search. Specifically, CGS dynamically prioritizes the population that deviates most from its neural reference. This coordination constrains the search space and helps guide the search toward generalizable, physically meaningful laws rather than formulas that only fit trajectory data.

Accurate recovery of synthetic network dynamics

To validate the efficacy of Coordinated Genetic Search (CGS), we benchmarked its performance against three established symbolic regression meth-

ods: SymDL [23], SINDy [18], and Two-Phase SINDy (TP-SINDy) [12]. We evaluated these approaches across four canonical network dynamics: Susceptible-Infected-Susceptible (SIS) epidemics [13], Lotka-Volterra (LV) population dynamics [14], Wilson-Cowan (WC) neural firing [8,9], and Kuramoto (KUR) oscillators [15]. Simulations were conducted on complex Erdős-Rényi (ER) [31] and Barabási-Albert (BA) [32] graph topologies consisting of 200 nodes. We assessed performance based on the recovery probability of the correct symbolic skeleton and the Mean Squared Error (MSE) between observed trajectories and those simulated from the recovered equations. We excluded NASSymDL [24] as our use of known inductive biases negates the need for neural architecture search, and D-CODE [19] due to its incompatibility with multi-trajectory regression. Detailed experimental settings, including dataset statistics, network training, and genetic-search hyperparameters, are provided in Section S1 of the Supplemental Information.

As shown in Fig. 3, CGS consistently achieved the highest recovery probability across the tested systems. While TP-SINDy improves upon standard SINDy by incorporating a fine-tuning phase to prune terms with small coefficients, it exhibited instability in recovering the correct skeletons for SIS and LV dynamics. This instability likely stems from the reliance on numerical derivative estimation (via five-point approximation [33]) and an inability to effectively constrain the model space; data normalization in these baselines can further exacerbate the issue by causing overfitting to candidate functions.

The limitations of sparse regression were particularly evident in the WC dynamics experiments. Here, TP-SINDy consistently failed to regress the correct skeleton because the edge dynamics evolve via a parametric function that cannot be represented as a linear combination of the predefined elementary functions used in the library. Conversely, for the KUR dynamics, both TP-SINDy and CGS successfully recovered the underlying laws with a probability of 1. In comparative terms, SymDL relies solely on a proxy model to search for node and edge dynamics separately. By contrast, CGS utilizes a coordinated search process involving neural references and interpolated trajectories, allowing it to achieve stronger recovery and prediction results in these controlled synthetic settings. These synthetic results provide strong evidence that CGS can recover the physical laws governing the tested network dynamics when the ground-truth equations are known. Additional results, including runtime comparisons, an overfitting case study under interpolated and extrapolated settings, and results on multi-dimensional FitzHugh–Nagumo dynamics, are provided in Section S2 of the Supplemental Information.

Case study on real influenza A dataset

Dataset We evaluate CGS on a real epidemic network using the same influenza A (H1N1) spreading dataset as [12]. In this dataset, each node represents a country or region, with the daily counts of newly reported cases as node states. The directed aviation edge $j \to i$ represents passenger flow from source region j to target region i. Its raw weight is the corresponding air-passenger volume. Following the preprocessing of [12], passenger volumes are normalized using population information and total daily passenger volume, yielding the normalized weighted adjacency $\hat{A}_{ij}$ used in the influenza model. Because this real-world dataset has no ground-truth symbolic governing equation, the experiment cannot evaluate symbolic recovery probability as in the synthetic systems. We instead treat it as a proof-of-concept case study and evaluate the inferred expressions by trajectory consistency with observed case counts, normalized prediction error, and physical plausibility. For a fair comparison, we employed the same data preprocessing procedures as [12], such as constructing the adjacency matrix and data cleaning. The symbolic regression is performed on population-normalized case trajectories. Specifically, daily case counts for all countries/regions are divided by the maximum population among the nodes in the influenza network before fitting. The reported MSE is computed on this normalized scale so that the fitted equations and error metric are defined consistently. For visualization and interpretation in Fig. 4(a), simulated trajectories are transformed back to the raw case-count scale. Therefore, the numerical coefficients in the recovered expressions should be interpreted with respect to population-normalized state variables rather than raw daily case counts.

Results We use CGS, SymDL, and TP-SINDy to regress candidate symbolic expressions for influenza A spread dynamics, and the regressed expressions are summarized in 5(b). The expression recovered by CGS in this case study is

$$\dot{\mathbf{x}}_i(t) = a\mathbf{x}_i(t) + \sum_j \hat{A}_{ij} \frac{b}{1+\exp-(m\mathbf{x}_j(t)+c)} \mathbf{x}_j(t), \tag{18}$$

where $j \to i$ denotes passenger flow from source region j to target region i, and $a = 0.0740$, $b = 0.0015$, $m = -0.0041$ and $c = 9.9643$. The node dynamics in (18) is a linear function, which is consistent with early exponential growth in epidemic models. The weighted edge contribution in (18) is proportional both to the source region's population-normalized case state and to the normalized passenger-flow weight, matching the intuition that mobility-driven spread increases with passenger flow from source to target and with infection level in the source region. The other factor of edge dynamics consists of a composition of a linear transformation followed by a sigmoid activation. This suggests that the rate of new infections from neighbors may be modulated by the local infection level, possibly due to factors such as population saturation, changes in mobility, or implemented control measures. This interpretation should be treated cautiously because the real dataset does not provide ground-truth mechanisms.

The symbolic expressions regressed by TP-SINDy and SymDL are

$$\dot{\mathbf{x}}_i(t) = a'\mathbf{x}_i(t) + \sum_j \hat{A}_{ij} \frac{b'}{1+\exp-\left(\mathbf{x}_i(t)-\mathbf{x}_j(t)\right)}, \tag{19}$$

$$\dot{\mathbf{x}}_i(t) = a''\mathbf{x}_i(t)^2 + \sum_j \hat{A}_{ij} \frac{b''}{1+\exp-\left(\mathbf{x}_i(t)-\mathbf{x}_j(t)\right)} \mathbf{x}_j(t) \tag{20}$$

where $a' = 0.074, b' = 7.130, a'' = 0.0742, b'' = 0.0012$. (19) is less physically plausible for epidemic spreading because it predicts a nonzero interaction contribution even when infected cases are absent. In contrast, (18) yields a zero interaction contribution when neighboring infected cases are zero, aligning with the expectation that epidemics cannot spread without infected individuals. Compared with (20) from SymDL, (18) has a simpler node dynamics and edge dynamics in this case study.

We compare the trajectories of symbolic expressions in (18), (19) and (20) with the real infected cases. Fig. 4(a) visualizes the simulation results of inferred dynamics in two regions, i.e., "Finland" and "Saint Pierre and Miquelon", after transforming the simulated normalized trajectories back to the raw case-count scale. The trajectories inferred by CGS are visually more consistent with the observed case trajectories in these two displayed regions, while the trajectories inferred by TP-SINDy and SymDL deviate more visibly. Because only observed case counts are available, these comparisons assess predictive consistency rather than correctness of a recovered symbolic law.

We evaluate the normalized prediction error of the regressed dynamics. Using this normalized error scale, CGS obtains a lower normalized MSE than TP-SINDy in this case study (0.8261 versus 0.9028), but the improvement is modest; therefore, we interpret the real-data result mainly as evidence of physical plausibility and predictive consistency rather than broad empirical dominance.

Robustness of Coordinated Genetic Search

We evaluate the robustness of CGS and TP-SINDy with the KUR dynamics, highlighting the advantages of the neural-symbolic approach over methods based on numerical derivatives (as stated in the introduction). We focus on performance under noisy observations and with large time intervals.

We add Gaussian noise to node states to assess performance under noise, with noise magnitude measured by the signal-to-noise ratio (SNR). As shown in Fig. 5(a), our method maintains a 100% recovery rate even as the SNR decreases to 25 dB, while TP-SINDy's recovery rate falls to 0% at 30 dB. In Fig. 5(b), CGS consistently produces more accurate symbolic expressions that have lower MSE. This occurs because TP-SINDy relies on numerically estimating time derivatives that are noisy and inaccurate, whereas our method uses neural dynamics to denoise and interpolate observations directly. Deep neural networks excel at handling noisy data by learning meaningful patterns from large amounts of data, even when the data contains significant noise. Using the accurately denoised observations, CGS predicts the constants in the formula more accurately and produces a more accurate trajectory when noise is present.

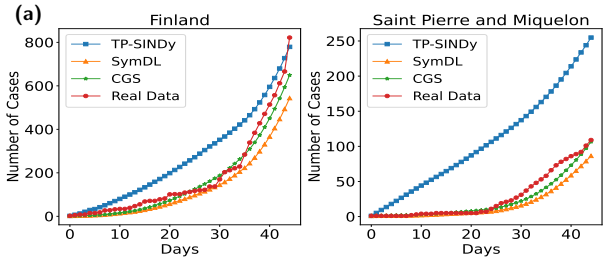
(a)

Graphs	Dynamics	Rec. Prob.↑				MSE↓ (10^{-2})			
		SymDL	SINDy	TP	PI	SymDL	SINDy	TP	PI
BA	SIS	0.35	0.11	0.15	1.00	0.979±0.173	0.484±0.056	0.434±0.052	0.312±0.012
	LV	0.16	0.12	0.20	1.00	2.075±0.303	1.170±0.049	0.875±0.057	0.136±0.008
	KUR	0.80	0.87	1.00	1.00	0.064±0.018	0.175±0.016	0.040±0.003	0.007±0.001
	WC	0.56	0.00	0.00	1.00	0.362±0.057	NA	NA	0.092±0.004
ER	SIS	0.31	0.11	0.17	1.00	1.173±0.095	0.468±0.059	0.386±0.051	0.119±0.025
	LV	0.15	0.09	0.19	1.00	1.784±0.236	0.941±0.041	0.763±0.077	0.251±0.007
	KUR	0.87	0.78	1.00	1.00	0.071±0.018	0.087±0.022	0.069±0.019	0.017±0.001
	WC	0.40	0.00	0.00	1.00	0.266±0.047	NA	NA	0.044±0.003

(b)

Dynamics	Models	Node dynamics	Edge dynamics
SIS	GT	$-0.5x_i(t)$	$(1-x_i(t))x_j(t)$
	CGS	$-0.48540x_i(t)$	$(1-x_i(t))x_j(t)$
	TP-SINDy (Rec.)	$-0.46640x_i(t)$	$(0.99119-1.09637x_i(t))x_j(t)$
	TP-SINDy (Fail)	$-0.47256x_i^2(t)-0.12596$	$0.10860\text{sigmoid}(x_j(t)-x_i(t))+$ $0.18835(x_j(t)-x_i^2(t))$ $0.19917(x_j(t)-x_i(t))+0.35416\sin(x_j(t))$
	SymDL (Rec.)	$-0.54827x_i(t)$	$(1.03945-0.92538x_i(t))x_j(t)$
	SymDL (Fail)	$-0.6017x_i(t)\sin(\cos(x_i(t)-0.5178))$	$(0.9966-1.1798x_i(t)+0.1984\sin(x_i(t)))x_j(t)$
LV	GT	$x_i(t)(0.75-0.5x_i(t))$	$-x_i(t)x_j(t)$
	CGS	$x_i(t)(0.75034-0.48812x_i(t))$	$-0.99428x_i(t)x_j(t)$
	TP-SINDy (Rec.)	$x_i(t)(0.69882-0.41853x_i(t))$	$-0.91701x_i(t)x_j(t)$
	TP-SINDy (Fail)	$0.03984+0.36330\ \sin(x_i(t))$	$-0.945810x_i(t)x_j(t)-0.11895x_i(t)x_j^2(t)$
	SymDL (Rec.)	$x_i(t)(0.703971-0.54885x_i(t))$	$-1.11962x_i(t)x_j(t)$
	SymDL (Fail)	$(x_i(t)-0.027459)\ \exp(-1.2791x_i(t))$	$-1.006(x_i(t)-0.0031034)x_j(t)$
KUR	GT	0.75	$\sin(x_i(t)-x_j(t))$
	CGS	0.75002	$\sin(1.0001x_i(t)-x_j(t))$
	TP-SINDy (Rec.)	0.75014	$0.99899\sin(x_i(t)-x_j(t))$
	TP-SINDy (Fail)	NA	NA
	SymDL (Rec.)	0.74777	$0.99037\sin(x_i(t)-x_j(t))$
	SymDL (Fail)	$x_i(t)\ \ 0.00815+0.74624$	$0.99725\sin(x_i(t)-1.003x_j(t)+0.0013507)$
WC	GT	$-x_i(t)$	$\text{sigmoid}(-0.75(x_j(t)-0.5))$
	CGS	$-x_i(t)$	$\text{sigmoid}(-0.74503(x_j(t)-0.49128))$
	TP-SINDy (Rec.)	NA	NA
	TP-SINDy (Fail)	$-0.82267x_i(t)$	$0.08513\text{sigmoid}(x_j(t)-x_i(t))$ $+0.68484\text{sigmoid}(x_j(t))$
	SymDL (Rec.)	$-1.0132x_i(t)$	$\text{sigmoid}(-0.78165(x_j(t)-0.47162))$
	SymDL (Fail)	$-1.0953x_i(t)-0.00437$	$\text{sigmoid}(\text{sigmoid}((x_i(t)+0.7432)\ \ x_i(t))-0.046806$

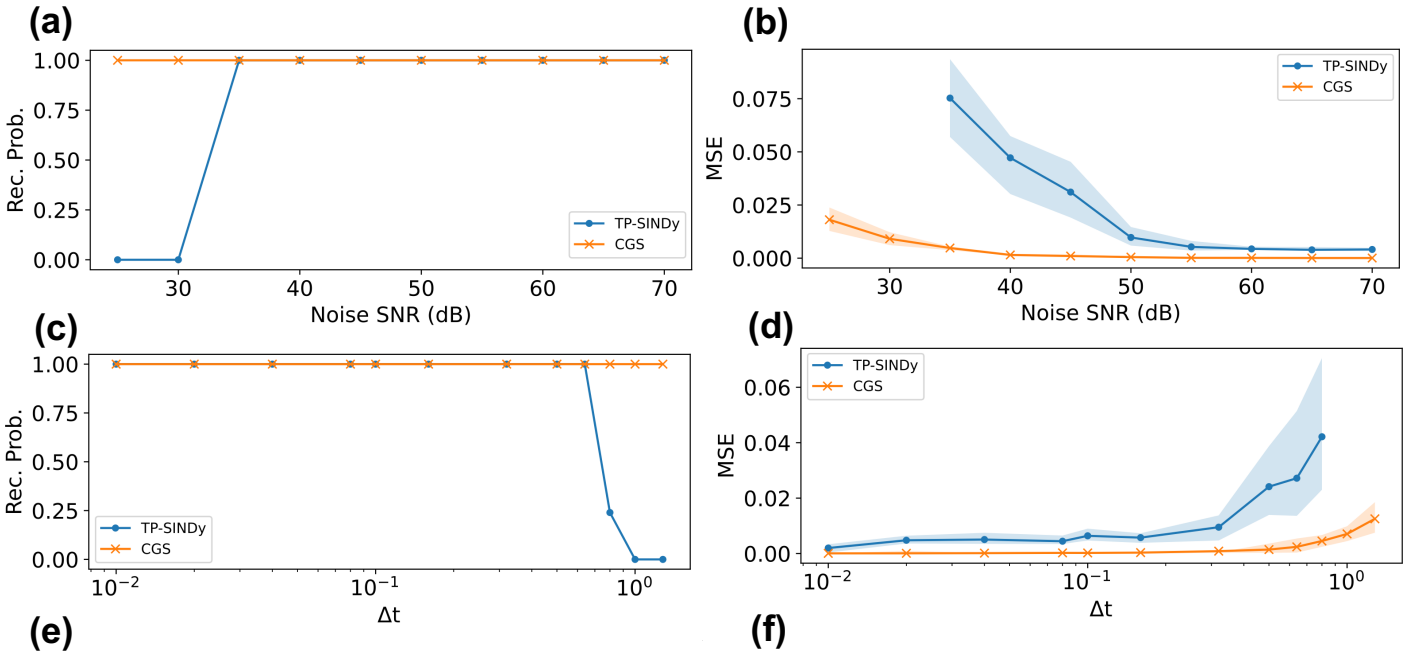
Figure 3. (a) Performance comparison on synthetic datasets. MSE values are scaled by 10^{-2} and should be multiplied by 10^{-2} to obtain the actual values. (TP: TP-SINDy, PI: CGS, NA: MSE is not applicable because of failure of the correct skeleton recovery.) (b) Symbolic regressions of CGS, TP-SINDy (Rec.), TP-SINDy (Fail), SymDL (Rec.), and SymDL (Fail) on SIS, LV, KUR, and WC dynamics in the BA graph. The ground-truth dynamics follow standard epidemic, ecological, oscillator, and neural dynamics formulations [8,9,13,14,15].



(b)

Models	Node dynamics	Weighted edge contribution
CGS	$0.0740x_i(t)$	$\hat{A}_{ij}\frac{0.0015}{1+\exp(-(-0.0041x_i(t)+9.9643))}x_j(t)$
TP-SINDy	$0.074x_i(t)$	$\hat{A}_{ij}\frac{7.130}{1+\exp(-(x_i(t)-x_j(t)))}$
SymDL	$0.0742x_i(t)^2$	$\hat{A}_{ij}\frac{0.0012}{1+\exp(-(x_i(t)-x_j(t)))}x_j(t)$

Figure 4. (a) Visualizing the predicted number of newly reported cases in two regions after transforming simulated normalized trajectories back to the raw case-count scale. (b) The symbolic expressions regressed by CGS, TP-SINDy, and SymDL on the influenza A dataset following the setup of [12]. The expressions are defined on population-normalized case states, and edge summations use the normalized weighted aviation adjacency $\hat{A}_{ij}$.



(e)

Model	SIS		LV	
	Rec. Prob.↑	MSE↓ (10^{-2})	Rec. Prob.↑	MSE↓ (10^{-2})
CGS	1	0.312±0.012	1	0.136±0.008
CGS w/o Interp	0.81	0.408±0.027	0.86	0.588±0.028

(f)

Models	Noise (SNR=35dB)		Time interval ($\Delta t = 1.28$)	
	Rec. Pro.↑	MSE↓ (10^{-2})	Rec. Pro.↑	MSE↓ (10^{-2})
CGS	1	0.478±0.103	1	0.454±0.213
CGS w/o Interp	0.84	6.970±1.870	0.78	2.645±1.138

Figure 5. Evaluation of robustness and the effect of denoised/interpolated trajectories. Shaded areas correspond to 95% confidence interval. (a) and (b) show the recovery probability and MSE when adding noise to observations. (c) and (d) show the recovery probability and MSE when increasing the time interval between observations. (e) Results for removing interpolated and denoised trajectories on SIS and LV dynamics in the BA graph. (f) Ablation variants compared with the full method on KUR dynamics in the BA graph.

TP-SINDy relies on the equal time interval to estimate time derivatives. Therefore, we increase the interval size to compare the performances of TP-SINDy and CGS. Sampling timestamps from [0,100] with different intervals Δt, Fig. 5(c) shows that our model maintains a 100% recovery rate, while TP-SINDy fails with larger intervals. Fig. 5(d) shows that CGS produces lower MSE across the tested intervals when both methods recover the correct skeleton of dynamics. This is because the interpolated observations in CGS are better suited when the time interval is large compared to the estimated time derivatives used by TP-SINDy. Visualization of interpolated trajectories and estimated time derivatives are shown in Figure S1 of the Supplemental Information.

Effect of denoised and interpolated trajectories

We conduct this ablation to isolate the effect of denoised and interpolated trajectories in CGS. Therefore, we test a variant of CGS that uses the original observations instead of the interpolated and denoised trajectory when calculating the fitness (CGS without Interp.).

Fig. 5(e) shows the results of SIS and LV dynamics in the BA graph. Without the interpolated and denoised observations, both the recovery probability and the accuracy of CGS drop. This indicates that the interpolated and denoised trajectories can provide high-quality fitness evaluation for symbolic regression.

We also evaluate the robustness of the ablation variants. Fig. 5(f) shows the results for KUR dynamics in the BA graph when the observations are noisy or the time interval is large. Unlike the results in Fig. 5(e), the success probability drops substantially when the interpolation component is removed. This supports the role of neural dynamics in denoising and augmenting trajectories.

This ablation isolates the trajectory-denoising and interpolation component rather than fully isolating adaptive coordination. A direct independent fitting strategy that fits F and G separately to neural references can inherit component-wise proxy errors: even when the proxy trajectory is accurate, its learned node and edge components may compensate for one another. CGS uses the neural references only to coordinate search priority, while the fitness

of candidate symbolic pairs is evaluated against denoised trajectories. The theoretical comparison with Separate Search in Theoretical analysis formalizes this distinction. A fully controlled empirical ablation isolating adaptive coordination from independent search, fixed alternation, and random-priority co-evolution remains an important direction for future work.

DISCUSSION

Complex network dynamics pose a central challenge for mechanistic discovery: accurate prediction does not necessarily imply correct mechanism. Because a node's observed evolution conflates intrinsic behavior with interactions, independently regressing node and edge equations can produce compensatory formulas that fit trajectories but obscure the underlying laws. CGS addresses this ambiguity by making coordination, rather than independent fitting, the core of symbolic discovery.

By combining a graph neural ODE proxy with coordinated genetic search, CGS uses neural models not as black-box endpoints but as scaffolds for interpretable equation discovery. The neural proxy denoises sparse observations and provides separate references for node and edge effects, while the symbolic search adaptively improves the component that deviates most from its reference. This strategy reduces compensatory overfitting and guides the search toward compact laws that are both predictive and physically meaningful.

Across synthetic systems, CGS improves formula recovery, prediction accuracy, and robustness to noise and coarse sampling under controlled settings with known ground-truth equations. The influenza A experiment provides a real-world plausibility case study: the recovered expression is consistent with basic epidemic-spreading intuition and obtains a modestly lower normalized prediction error than TP-SINDy, but it does not establish broad real-world superiority. These results suggest that coordinated symbolic discovery has potential to move network science beyond forecasting toward mechanism-level understanding. More broadly, this principle may extend to systems with hidden variables, adaptive connectivity, or higher-order interactions, where discovering the structure and the governing equations must ultimately become a joint problem.

Limitations of the study

Our method has several limitations:

- The real-data evaluation is limited to one influenza A case study. Because real-world systems lack ground-truth governing equations, symbolic correctness cannot be verified directly; the real-data result should therefore be interpreted as evidence of physical plausibility and predictive consistency rather than proof of the recovered mechanism.
- The current empirical ablation isolates the denoised/interpolated trajectory component, but it does not fully isolate adaptive coordination against independent search, fixed alternation, or random-priority co-evolution. A controlled empirical study of these variants remains important future work.
- CGS cannot successfully recover symbolic expressions when the formulations are highly complex or the dimension of node states is high. Highly complex formulations imply a large genetic-search space, often requiring larger populations and more generations. Since the fitness of our method is calculated based on pairwise combinations of node and edge dynamics, the fitness evaluation is computationally expensive and memory-consuming.
- CGS depends on the proxy model producing sufficiently accurate denoised trajectories. If the graph neural ODE proxy fits the observed trajectory poorly, the symbolic search may be guided by low-quality trajectory supervision.
- CGS remains challenging when variables are missing in the observations. In some complex systems, it is difficult to observe all variables at the same time. In this case, the prediction accuracy of neural dynamics (4) may not be high enough to provide high-quality supervision data for symbolic regression.

REFERENCES

1. Gerstner W., Kistler W. M., Naud R., et al. (2014). Neuronal dynamics: From single neurons to networks and models of cognition. (Cambridge University Press).
2. Gao J., Barzel B. and Barabási A. L.. (2016). Universal resilience patterns in complex networks. *Nature* **530**:307–12. DOI:10.1038/nature16948
3. Bashan A., Gibson T. E., Friedman J., et al. (2016). Universality of human microbial dynamics. *Nature* **534**:259–62. DOI:10.1038/nature18301
4. Newman M., Barabási A. L. and Watts D. J. (2011). The structure and dynamics of networks. (Princeton University Press).
5. Kitsak M., Gallos L. K., Havlin S., et al. (2010). Identification of influential spreaders in complex networks. *Nature Physics* **6**:888–93. DOI:10.1038/nphys1746
6. Pastor-Satorras R. and Vespignani A.. (2001). Epidemic spreading in scale-free networks. *Physical Review Letters* **86**:3200. DOI:10.1103/PhysRevLett.86.3200
7. Luo C., Lambiotte R. and Ji P.. (2026). Temporal heterogeneity shapes diffusion dynamics in complex networks. *Nature Communications* **17**:5601. DOI:10.1038/s41467-026-72161-w
8. Laurence E., Doyon N., Dubé L. J., et al. (2019). Spectral dimension reduction of complex dynamical networks. *Physical Review X* **9**:011042.
9. Wilson H. R. and Cowan J. D.. (1972). Excitatory and inhibitory interactions in localized populations of model neurons. *Biophysical Journal* **12**:1–24. DOI:10.1016/s0006-3495(72)86068-5
10. Zang C. and Wang F. (2020). Neural dynamics on complex networks. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (ACM): 892–902.
11. Murphy C., Laurence E. and Allard A.. (2021). Deep learning of contagion dynamics on complex networks. *Nature Communications* **12**:4720. DOI:10.1038/s41467-021-24732-2
12. Gao T. T. and Yan G.. (2022). Autonomous inference of complex network dynamics from incomplete and noisy data. *Nature Computational Science* **2**:160–8. DOI:10.1038/s43588-022-00217-0
13. Pastor-Satorras R., Castellano C., Van Mieghem P., et al. (2015). Epidemic processes in complex networks. *Reviews of Modern Physics* **87**:925. DOI:10.1103/RevModPhys.87.925
14. MacArthur R.. (1970). Species packing and competitive equilibrium for many species. *Theoretical Population Biology* **1**:1–11. DOI:10.1016/0040-5809(70)90039-0
15. Kuramoto Y. (1984). Chemical Oscillations, Waves, and Turbulence. (Springer).
16. Gaucel S., Keijzer M., Lutton E., et al. (2014). Learning dynamical systems using standard symbolic regression. *Genetic Programming: 17th European Conference, EuroGP 2014, Granada, Spain, April 23-25, 2014, Revised Selected Papers 17* (Springer): 25–36.
17. Kronberger G., Kammerer L. and Kommenda M. (2020). Identification of dynamical systems using symbolic regression. *Computer Aided Systems Theory – EUROCAST 2019: 17th International Conference, Las Palmas de Gran Canaria, Spain, February 17–22, 2019, Revised Selected Papers, Part I 17* (Springer): 370–7.
18. Brunton S. L., Proctor J. L. and Kutz J. N.. (2016). Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences* **113**:3932–7. DOI:10.1073/pnas.1517384113
19. Qian Z., Kacprzyk K. and van der Schaar M. (2022). D-Code: Discovering closed-form ODEs from observed trajectories. *International Conference on Learning Representations*.
20. d'Ascoli S., Becker S., Schwaller P., et al. (2024). ODEFormer: Symbolic regression of dynamical systems with transformers. *The Twelfth International Conference on Learning Representations*.
21. Schmidt M. and Lipson H.. (2009). Distilling free-form natural laws from experimental data. *Science* **324**:81–5. DOI:10.1126/science.1165893
22. Petersen B. K., Landajuela M., Mundhenk T. N., et al. (2019). Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. *arXiv preprint arXiv: 1912.04871*.
23. Cranmer M., Sanchez Gonzalez A., Battaglia P., et al. (2020). Discovering symbolic models from deep learning with inductive biases. *Advances in Neural Information Processing Systems* **33**:17429–42.
24. Shi H., Ding J., Cao Y., et al. (2023). Learning symbolic models for graph-structured physical mechanism. *The Eleventh International Conference on Learning Representations*.
25. Jing F., Zhang Z. K., Zhang Q., et al. (2026). Predictability of complex networks. *Proceedings of the National Academy of Sciences* **123**:e2535161123. DOI:10.1073/pnas.2535161123
26. Hu J., Cui J. and Yang B.. (2025). Learning interpretable network dynamics via universal neural symbolic regression. *Nature Communications* **16**:6226. DOI:10.1038/s41467-025-61575-7
27. Yu Z., Ding J. and Li Y.. (2026). Discovering network dynamics with neural symbolic regression. *Nature Computational Science* **6**:156–68. DOI:10.1038/s43588-025-00893-8
28. Liu B., Luo W., Li G., et al. (2023). Do we need an encoder-decoder to model dynamical systems on networks? *arXiv preprint arXiv: 2305.12185*.
29. Barzel B. and Barabási A. L.. (2013). Universality in network dynamics. *Nature Physics* **9**:673–81. DOI:10.1038/nphys2741
30. Potter M. A. and De Jong K. A. (1994). A cooperative coevolutionary approach to function optimization. *International Conference on Parallel Problem Solving from Nature* (Springer): 249–57.
31. Erdős P. and Rényi A.. (1959). On random graphs I. *Publicationes Mathematicae Debrecen* **6**:290–7.
32. Barabási A. L. and Albert R.. (1999). Emergence of scaling in random networks.

Science **286**:509–12. DOI:10.1126/science.286.5439.509
33. Sauer T. (2011). Numerical solution of stochastic differential equations in finance. *Handbook of Computational Finance* (Springer), pp: 529–50.

FUNDING AND ACKNOWLEDGMENTS

This work is supported by National Key Research and Development Program of China (under Grant No. 2023YFB2903904). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

AUTHOR CONTRIBUTIONS

H.Q. conceived the study, designed the algorithm, performed the theoretical analysis, and wrote the original draft. S.L. implemented the code and conducted the experiments. Y.W. contributed to the experiments and data analysis. Y.L. contributed to the interpretation of the results and revised the manuscript. Q.Y. supervised the project, contributed to the study design, and revised the manuscript. All authors contributed to the manuscript and approved the final version.

DECLARATION OF INTERESTS

The authors declare no competing interests.

ETHICAL STATEMENT AND PATIENT CONSENT

Not applicable.

DATA AND CODE AVAILABILITY

Data and code are available from the corresponding author upon reasonable request.

SUPPLEMENTAL INFORMATION

It can be found online at https://doi.org/10.59717/ipj.aiplus.2026.100006

LEAD CONTACT WEBSITE

Not applicable.