Interpretable Relational Inference with LLM-Guided Symbolic Dynamics Modeling

Xiaoxiao Liang* **Juyuan Zhang*** **Liming Pan†** **Linyuan Lü†**
School of Cyber Science and Technology
University of Science and Technology of China
Hefei, Anhui, China
pan_liming@ustc.edu.cn, linyuan.lv@ustc.edu.cn

Abstract

Inferring latent interaction structures from observed dynamics is a fundamental inverse problem in many-body interacting systems. Most neural approaches rely on black-box surrogates over trainable graphs, achieving accuracy at the expense of mechanistic interpretability. Symbolic regression offers explicit dynamical equations and stronger inductive biases, but typically assumes known topology and a fixed function library. We propose **COSINE** (**C**o-**O**ptimization of **S**ymbolic **I**nteractions and **N**etwork **E**dges), a differentiable framework that jointly discovers interaction graphs and sparse symbolic dynamics. To overcome the limitations of fixed symbolic libraries, COSINE further incorporates an outer-loop large language model that adaptively prunes and expands the hypothesis space using feedback from the inner optimization loop. Experiments on synthetic systems and large-scale real-world epidemic data demonstrate robust structural recovery and compact, mechanism-aligned dynamical expressions.[3]

1 Introduction

Interacting structures among entities are fundamental to understanding dynamical systems across physics, biology, neuroscience, epidemiology, finance, and sociology [2, 52, 22, 36, 34, 8]. In many settings, graphs are unobserved due to experimental, ethical, or economic constraints, while time-resolved dynamics are measurable, motivating inference of latent interactions from trajectories.

As illustrated in Figure 1, relational inference aims to estimate pairwise dependencies from dynamical observations. Classical methods, including Granger causality [17], mutual information [49], and transfer entropy [40], primarily characterize statistical dependencies, but typically lack an explicit model of the underlying dynamics. Neural approaches, such as Neural Relational Inference (NRI) and its extensions [25, 16, 35, 30, 9], jointly infer latent interaction structures and learn dynamics surrogates. These methods usually rely on the joint training of a graph generator and a dynamics surrogate; despite their predictive accuracy, the learned surrogates often remain mechanistically opaque. Attention-based models [31, 48] improve computational efficiency by replacing explicit pairwise inference with attention mechanisms, yet their limited mechanistic interpretability remains largely unresolved. More critically, such black-box neural methods may exploit over-parameterization to fit spurious edges, thereby accommodating observed trajectories at the expense of recovering the true governing mechanisms.

Symbolic regression (SR) [39] targets explicit governing equations from dynamics [43, 11, 10, 47, 23]. Concise expressions impose strong inductive bias and improve mechanistic readability;

*Equal contribution.
†Corresponding authors.
[3]Code: https://anonymous.4open.science/r/COSINE-6D43.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

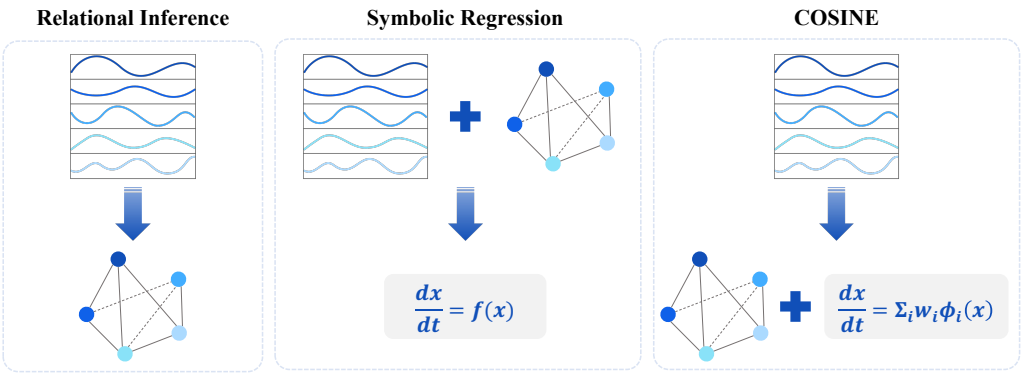

Figure 1: Conceptual comparison of relational inference, symbolic regression, and COSINE. Relational inference recovers latent interaction graphs from dynamical data; symbolic regression discovers governing equations of dynamics given known structures; COSINE integrates sparse symbolic modeling with graph learning to jointly infer both structure and dynamics.

expression-level structure is further supported by recent compositional models, including KAN and related studies [29, 46, 32]. This compactness acts as a functional constraint: incorrect edges are harder to absorb through over-parameterization.

Despite these advantages, symbolic regression on network dynamics remains challenging when networks are unknown. As shown in Figure 1, most existing SR methods assume known network [18, 50, 38, 42]. With latent structure and dynamics, discovery becomes a coupled search; SINDy reduces equation discovery to sparse regression over fixed libraries [5, 3, 21, 41, 51, 15, 27], but closed-world libraries require domain knowledge and can degrade robustness.

To address these challenges, we propose **COSINE** (**Co**-**O**ptimization of **S**ymbolic **I**nteractions and **N**etwork **E**dges), a differentiable framework for joint structure and mechanism discovery (Figure 1). COSINE decomposes dynamics into reusable *message* and *update* components as sparse combinations over shared symbolic libraries, enabling expression reuse and a search space independent of network size. To relax fixed-library bias, COSINE uses an outer-loop large language model (LLM) as a *symbolic hypothesis proposal mechanism* to prune and augment terms from inner-loop feedback, without system-specific templates. Our contributions are summarized as follows:

- **COSINE Framework:** We propose an end-to-end differentiable pipeline that jointly optimizes latent graph structures and sparse dynamical equations, enabling the discovery of interpretable dynamical models under unknown networks.
- **Sparse Symbolic Message Passing:** We cast the joint discovery problem as sparse regression over message/update libraries. This symbolic formulation enables dynamical mechanism discovery alongside structure inference.
- **LLM-Guided Library Evolution:** We utilize a feedback-driven updating strategy to adaptively adjust the basis library without hand-crafted, system-specific templates. This approach decouples LLM-driven *hypothesis generation* from optimization-based *hypothesis selection*, ensuring the discovered mechanisms remain numerically grounded.
- **Extensive Experiments:** Evaluations across diverse synthetic dynamical systems and real-world datasets demonstrate that COSINE consistently achieves state-of-the-art performance in relational inference while successfully recovering parsimonious and physically consistent symbolic governing laws.

2 Related Work

2.1 Relational Inference

Inferring latent interaction structures from multivariate dynamical data has been widely studied. Classical techniques characterize interaction structures in observational data using statistical mea-

sures, including Granger causality [17], correlation and partial correlation [37], mutual information [49], and transfer entropy [40]. These approaches focus on statistical dependencies, neglect explicit dynamical modeling, and fail under complex nonlinear dynamics.

Neural Relational Inference (NRI) [25] introduced a unified framework that jointly infers latent graphs and system dynamics via end-to-end learning. An encoder encodes graph from observed trajectories, while a decoder predicts future states conditioned on the inferred graph. Subsequent extensions address time-varying graphs [16], heterogeneous interactions [45, 20], directed graph inference [44], and sparsity/degree priors [28, 9]. Related work also integrates attention-based architectures to capture fine-grained, potentially asymmetric dependencies [19, 20, 31, 48]. However, despite their expressive power, these methods largely treat dynamics as a black box, limiting their interpretability and making them susceptible to spurious associations with over-parameterization.

2.2 Symbolic and Interpretable Dynamical Modeling

Sparse Symbolic Regression on Dynamics. Sparse Identification of Nonlinear Dynamics (SINDy) [5] identifies governing equations from dynamical systems by selecting a parsimonious subset of basis functions from a predefined library via sparse optimization. Numerous extensions improve its robustness under noise and higher-order integration schemes [33, 6, 15, 51]. While network-aware variants like SINDyG [3, 13] exist, they typically **assume a fixed or known graph structure**, making them difficult to apply when the network is hidden and no domain knowledge is available.

Symbolic Discovery on Network Dynamics. To explore broader symbolic spaces, neural-guided symbolic regression has emerged, utilizing evolutionary search [39, 10], neural approximations [43], or Transformer-based generation [23, 4]. For networked systems, recent methods like ND2 [50] and PSE [38] decompose dynamics into node-, interaction-, and aggregation-level operators, while LaSR [18] leverages LLM-assisted reasoning to refine hypotheses, introducing an LLM into its genetic algorithm to improve performance.

Neural relational inference methods often lack interpretability and can overfit spurious interactions. Symbolic regression offers stronger inductive biases for dynamical systems, but becomes difficult when the graph is unknown, since the graph structure and equation form are tightly coupled and the search space explodes. SINDy alleviates symbolic search via sparsity, yet relies on a pre-defined operator library and domain priors. **COSINE** addresses this gap by jointly optimizing a latent interaction graph and sparse symbolic dynamics, enabling accurate structure inference together with interpretable mechanism discovery.

3 Method

3.1 Overall Framework

COSINE (**C**o-**O**ptimization of **S**ymbolic **I**nteractions and **N**etwork **E**dges) aims to **jointly discover** the latent interaction graph and governing mechanisms from observed trajectories $\mathbf{X} = (\mathbf{x}^0, \ldots, \mathbf{x}^{T-1})$. We consider N interacting variables, where the state of node i at discretized time t is $x_i^t \in \mathbb{R}^D$. Let $\mathbf{x}^t = \{x_1^t, \ldots, x_N^t\}$ be the global state and $\mathbf{x}_i = (x_i^0, \ldots, x_i^{T-1})$ be the trajectory of node i. We write $x_i^{t,(d)}$ for the d-th component (omitting t in tables as $x_i^{(d)}$).

As shown in Figure 2, COSINE operates as a two-level closed-loop system. The **outer loop** employs an LLM as a symbolic supervisor to iteratively refine the basis function library $\Theta(\cdot)$ based on training feedback. Within the **inner loop**, we adopt a non-amortized setting where the graph A (parameterized by $\mathbf{\Psi}$) and dynamical coefficients $\mathbf{W}$ are **co-optimized** via gradient descent to minimize the prediction error. This design enables the discovery of parsimonious, interpretable governing mechanisms without requiring system-specific prior knowledge.

Unless otherwise specified, we train COSINE using one-step supervision (predicting $\mathbf{x}^{t+1}$ from $\mathbf{x}^t$) under teacher forcing, which provides a stable and comparable objective across continuous- and discrete-time systems. At evaluation time, the learned mechanisms can also be rolled out for multiple steps to inspect qualitative trajectory behavior, while the inferred graph is assessed directly by edge-level metrics.

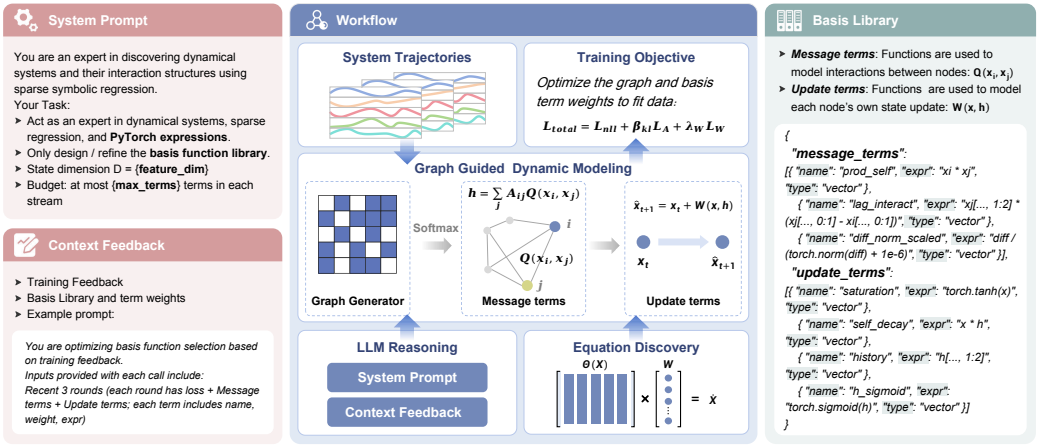

Figure 2: The COSINE architecture. (Left) LLM-based Reasoning refines the basis library $\Theta(\cdot)$ via performance feedback. (Middle) Graph-Guided Modeling co-optimizes latent edges A_{ij} and coefficients $\mathbf{W}$ through differentiable symbolic message-passing. (Right) Symbolic Basis Library bridges high-level reasoning with numerical discovery of governing mechanisms.

3.2 Graph Generation Module

The graph generator encodes unknown interaction structures into a set of trainable latent variables and maps them to an adjacency matrix. For binary graphs, $a = 0$ denotes a non-edge and $a = 1$ an edge. For each ordered pair (i, j), we maintain two real-valued latent variables $\Psi_{ij}^0, \Psi_{ij}^1 \in \mathbb{R}$. To obtain differentiable edge sampling and soft weights, we employ the temperature-controlled Gumbel–Softmax to map latent variables to edge-type probabilities:

$$A_{ij}^a = \text{softmax}\Big(\frac{1}{\tau}\left[\Psi_{ij}^0 + g_{ij}^0,\ \Psi_{ij}^1 + g_{ij}^1\right]\Big)_a, \tag{1}$$

where $a \in \{0, 1\}$ denotes the edge type, $\tau > 0$ is the temperature parameter controlling the sharpness of the distribution, and g_{ij}^a is independent Gumbel noise added to each edge logit to explore alternative edge assignments. As τ decreases, the edge distribution becomes increasingly peaked and converges to a deterministic graph in the limit. This construction naturally generalizes to multi-relation settings by assigning multi-dimensional latent scores to each edge and applying a Softmax to obtain a categorical distribution over relation types.

We use $A_{ij}^{\text{soft}} = A_{ij}^1$, the differentiable probability of edge existence, to weight message aggregation during training. The final binary graph is $A_{ij}^{\text{hard}} = \mathbb{I}(A_{ij}^{\text{soft}} \geq 0.5)$.

3.3 Sparse Regression Modeling

Complex networked dynamics can often be written as a combination of *self-dynamics* and *interaction dynamics* [13]:

$$\frac{d}{dt}x_i = f_i(\mathbf{x}, \mathbf{A}) = W(x_i) + \sum_{j=1}^{N} A_{ij}\, Q(x_i, x_j), \tag{2}$$

where $W(\cdot)$ describes the intrinsic state evolution of each node, $Q(\cdot, \cdot)$ captures interactions with neighboring nodes, and $A_{ij} \in \{0, 1\}$ indicates the network. In classical neural approaches, a surrogate message-passing GNN is leveraged to learn the unknown dynamics from data. A $v \to e$ transformation generates edge features, which are then aggregated back to nodes to update node states ($e \to v$):

$$v \to e: \quad \mathbf{z}_{(i,j)}^l = f_e^l\big([\mathbf{z}_i^l, \mathbf{z}_j^l, \mathbf{x}_{(i,j)}]\big), \tag{3}$$

$$e \to v: \quad \mathbf{z}_j^{l+1} = f_v^l\big([\sum_{i \in \mathcal{N}_j} \mathbf{z}_{(i,j)}^l, \mathbf{x}_j]\big), \tag{4}$$

where $\mathbf{z}_i^l$ denotes the embedding of node v_i at layer l, $\mathbf{z}_{(i,j)}^l$ denotes the embedding of edge $e_{(i,j)}$, and $\mathbf{x}_i$ and $\mathbf{x}_{(i,j)}$ represent the initial node and edge features, respectively. $\mathcal{N}_j$ denotes the set of neighbors with incoming edges to node j, and $[\cdot,\cdot]$ indicates vector concatenation. The functions f_v and f_e are node- and edge-level neural networks, respectively. The final node embeddings $\mathbf{z}^L$ are then used to predict the next-step state of the dynamical system. The recursion starts from $\mathbf{z}_i^0 = \mathbf{x}_i$; the first $v \to e \to v$ pass computes $\mathbf{z}_i^1$, and subsequent layers repeat these transformations.

Inspired by the message-passing decomposition in Equation (3), we design a sparse regression module that employs *two basis-function libraries* to approximate the two stages of GNN modeling. Specifically, we decompose the fitting process into a **message flow** and an **update flow**. The message flow generates interaction messages from neighbors, while the update flow updates node states based on aggregated messages. By modeling each component using a sparse linear combination of basis functions, we can explicitly capture key mechanisms and interpret the equation structure through coefficient sparsity:

$$\mathbf{e}_{ij} = \text{MESSAGE}(x_i, x_j), \tag{5}$$

$$x_i^{t+1} = \text{UPDATE}\big(x_i^t, \text{AGG}_{j\in\mathcal{N}_{\text{in}}(i)}\mathbf{e}_{ij}^t\big). \tag{6}$$

Here $\mathcal{N}_{\text{in}}(i)$ denotes the incoming neighbors of node i under the adjacency convention above (i.e., indices j such that $A_{ij} = 1$).

Message term. Given a message basis function set $\Theta_{\text{msg}} = \{\phi_1, \ldots, \phi_M\}$, for each directed pair (i, j) we compute

$$\phi_m^{\text{msg}}(x_i^t, x_j^t) \in \mathbb{R}^D, \quad m = 1, \ldots, M, \tag{7}$$

where D is the node feature dimension. A linear combination with learnable weights yields the edge message vector

$$\mathbf{e}_{ij}^t = \sum_{m=1}^{M} \mathbf{W}_{\text{msg}}[m] \odot \phi_m^{\text{msg}}(x_i^t, x_j^t) \in \mathbb{R}^D, \tag{8}$$

where $\odot$ denotes element-wise multiplication and $\mathbf{W}_{\text{msg}}[m] \in \mathbb{R}^D$ is the coefficient vector for the m-th basis. Given the soft structure $\mathbf{A}^{\text{soft}}$, messages are aggregated for node i as

$$\mathbf{h}_i^t = \sum_{j=1}^{N} A_{ij}^{\text{soft}}\, \mathbf{e}_{ij}^t. \tag{9}$$

Update term. Given an update basis set $\Theta_{\text{upd}} = \{\phi_1^{\text{upd}}, \ldots, \phi_U^{\text{upd}}\}$, we construct update features for node i using the current state x_i^t, the aggregated message $\mathbf{h}_i^t$, and the node degree $k_i = \sum_j A_{ij}^{\text{soft}}$:

$$\phi_n^{\text{upd}}(x_i^t, \mathbf{h}_i^t, k_i) \in \mathbb{R}^D, \quad n = 1, \ldots, U, \tag{10}$$

and obtain the final increment

$$\Delta\mathbf{x}_i^t = \sum_{n=1}^{U} \mathbf{W}_{\text{upd}}[n] \odot \phi_n^{\text{upd}}(x_i^t, \mathbf{h}_i^t, k_i). \tag{11}$$

Thus, the one-step prediction is

$$\hat{\mathbf{x}}_i^{t+1} = \mathbf{x}_i^t + \Delta\mathbf{x}_i^t. \tag{12}$$

For continuous-time systems, this can be viewed as an explicit Euler discretization where $\Delta\mathbf{x}_i^t$ approximates $\Delta t\, \dot{\mathbf{x}}_i^t$ (with Δt absorbed into the learned coefficients). For discrete-time systems, $\Delta\mathbf{x}_i^t$ acts as a residual update that directly models the transition $\mathbf{x}_i^t \mapsto \mathbf{x}_i^{t+1}$.

Joint optimization manifests at two levels: (i) the structural layer, where A_{ij}^{soft} represents the probability of edge existence; and (ii) the equation layer, where the **sparsity** of $\mathbf{W}_{\text{msg}}$ and $\mathbf{W}_{\text{upd}}$ determines which basis functions truly participate in generating the dynamics. Coefficient magnitudes rank terms within the chosen basis and normalization; each term's contribution also depends on the scale of its basis output.

3.4 Training Objective and Loss

Given the one-step prediction $\hat{\mathbf{x}}^{t+1}$ and the ground-truth observation $\mathbf{x}^{t+1}$, we employ the Gaussian negative log-likelihood (NLL) as the primary objective, which is equivalent to minimizing the scaled squared error under a fixed variance σ^2. The prediction loss is formulated as:

$$\mathcal{L}_{\text{nll}} = \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left\| \hat{\mathbf{x}}_i^{t+1} - \mathbf{x}_i^{t+1} \right\|_2^2 . \tag{13}$$

To stabilize structure learning and impose a prior over edge-type distributions, we include a KL regularization term (computed over all ordered node pairs):

$$\mathcal{L}_A = \mathrm{KL}\big(q(\mathbf{A}) \,\|\, \mathrm{r}\big), \tag{14}$$

where $q(\mathbf{A})$ denotes the edge distribution induced by the graph generator, and r is a simple factorized prior over edge types (e.g., a Bernoulli prior favoring sparsity when appropriate; if no prior knowledge is available, we default to a uniform prior). To encourage sparsity in symbolic regression, we apply ℓ_1 regularization to both sets of coefficients:

$$\mathcal{L}_W = \|\mathbf{W}_{\text{msg}}\|_1 + \|\mathbf{W}_{\text{upd}}\|_1. \tag{15}$$

The ℓ_1 penalty selects symbolic terms through $\mathbf{W}_{\text{msg}}$ and $\mathbf{W}_{\text{upd}}$. The total loss is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{nll}} + \beta_{\text{KL}} \mathcal{L}_A + \lambda_W \mathcal{L}_W, \tag{16}$$

where β_{KL} and λ_W are hyperparameters. The full training process is shown in Section A.1. After each inner-loop optimization, we summarize the current library Θ performance together with the learned coefficient weights and feed this information to the LLM to guide an *evolution* of the basis library for the next inner-loop.

3.5 LLM-Guided Library Evolution

The outer loop performs a lightweight search over symbolic space by *editing the basis library*. Each round uses inner-loop feedback (loss, sparsity, and basis libraries) to prompt the LLM to prune or augment terms, producing a new candidate library. To avoid expensive multi-branch search while keeping evolution controlled, we use a **best-so-far** strategy that maintains a single optimal library and only accepts candidates that strictly improve the validation loss.

For each library, $\mathcal{L}_{\text{val}}$ is evaluated on a held-out validation set after inner-loop training and used for library selection. Formally, we track the best library across rounds:

$$\Theta^\star = \arg \min_{\Theta \in \{\Theta^{(0)}, \ldots, \Theta^{(r)}\}} \mathcal{L}_{\text{val}}(\Theta), \tag{17}$$

and update $\Theta^\star \leftarrow \widetilde{\Theta}$ only if $\mathcal{L}_{\text{val}}(\widetilde{\Theta}) < \mathcal{L}_{\text{val}}(\Theta^\star)$. This strategy keeps computation predictable and reduces the risk of degradation from noisy edits, while leveraging strong numerical feedback to steer the symbolic space toward parsimonious, physically consistent mechanisms. In practice, the validation gate decouples symbolic exploration from numerical selection: the LLM only proposes candidates, while the validation loss determines acceptance. If no candidate improves $\mathcal{L}_{\text{val}}$ over successive rounds, the search naturally terminates without accumulating regressions. Further method details are provided in Section A; details of the library evolution and LLM prompts/models are presented in Sections A.2 and A.3.

4 Experiments

4.1 Results on Synthetic Data

Networks. We conduct experiments on three standard graph families: Erds–Rényi (ER), Barabási–Albert (BA), and Watts–Strogatz (WS). For ER graphs, the edge probability is set to $p = 0.1$; for BA graphs, each new node attaches to $m = 2$ existing nodes; for WS graphs, we start from a ring lattice with $k = 2$ nearest neighbors per node and rewire edges with probability $p = 0.1$.

Table 1: Relational inference performance (AUC %) on different graphs with $N = 50$ nodes across various dynamical systems. In the Volume column, $a \times b$ denotes trajectories $\times$ sampled steps. Boldface marks the highest accuracy.

DYN.	GRAPH	VOLUME	GC	MI	TE	NRI	GDP	RIVA	COSINE
MM	ER-50	50×10	60.24	77.84	53.66	96.25±2.22	98.31±1.41	52.17±1.15	**99.63±0.12**
	BA-50	50×10	62.79	88.04	63.96	82.67±1.81	93.02±3.94	52.83±1.22	**98.07±0.45**
	WS-50	50×10	80.33	97.77	57.56	**99.83±0.15**	56.77±2.55	53.65±1.40	99.10±0.25
DIFF	ER-50	50×10	63.84	56.00	57.63	91.87±7.80	93.44±4.87	53.02±1.35	**99.36±0.10**
	BA-50	50×10	75.54	72.06	61.71	94.16±11.62	94.41±3.23	52.89±1.18	**96.55±1.05**
	WS-50	50×10	73.63	76.39	90.04	99.31±0.55	56.68±3.12	53.72±1.25	**100.00±0.00**
SPR	ER-50	15×10	50.61	72.24	76.05	99.84±0.47	99.99±0.02	52.82±1.50	**100.00±0.00**
	BA-50	15×10	54.53	91.16	84.67	98.17±5.40	99.88±0.36	54.16±1.62	**100.00±0.00**
	WS-50	15×10	55.44	50.40	51.83	96.77±1.25	99.94±0.05	55.20±1.45	**100.00±0.00**
KURA	ER-50	30×30	54.94	64.69	64.76	82.09±19.14	94.93±12.94	75.98±2.34	**99.99±0.01**
	BA-50	30×30	51.50	55.46	61.87	69.70±18.16	90.13±12.38	76.06±2.15	**99.85±0.14**
	WS-50	30×30	59.27	**100.00**	96.63	50.17±1.05	**100.00±0.00**	98.19±0.55	**100.00±0.00**
FJ	ER-50	20×10	93.00	53.66	83.64	97.67±1.06	99.82±0.47	52.55±1.10	**100.00±0.00**
	BA-50	20×10	92.87	52.32	86.88	91.62±4.67	92.63±13.46	52.64±1.32	**99.98±0.02**
	WS-50	20×10	99.81	66.14	95.37	68.76±5.25	89.51±3.44	52.89±1.45	**100.00±0.00**
CMN	ER-50	20×10	52.89	87.39	64.35	89.76±2.59	97.58±3.38	50.84±0.95	**99.75±0.15**
	BA-50	20×10	53.44	87.84	71.51	83.35±2.30	88.83±6.19	53.87±1.67	**93.23±2.45**
	WS-50	20×10	61.08	98.94	86.70	69.21±4.12	51.51±1.10	55.96±1.88	**99.05±0.55**

Dynamical systems. We evaluate COSINE on six representative systems covering biochemical reactions, diffusion, mechanical coupling, synchronization, opinion dynamics, and chaotic maps: (1) *Michaelis-Menten kinetics* (MM) [24]; (2) *Diffusion* (Diff); (3) *Network of springs* (Spr); (4) *Kuramoto model* (Kura) [26]; (5) *Friedkin-Johnsen dynamics* (FJ) [12, 1]; and (6) *Coupled map network* (CMN) [14]. These systems, generated on ER, BA, and WS networks, span continuous and discrete time with diverse nonlinear interactions.

Evaluation metrics. We report AUC as our primary metric for relational inference, computed by comparing the predicted edge scores A_{ij}^{soft} against the binary ground-truth adjacency. In addition, for mechanism discovery, we compute *term accuracy* as primitive coverage: for each module (Message/Update), we take the top $K = 3$ terms by absolute coefficient magnitude and measure the fraction of required ground-truth primitives recovered among them (using simple equivalence rules, e.g., matching sinusoidal coupling on $x_j - x_i$ up to sign/constant scaling).

Baselines. We compare COSINE with various baselines from three categories: classical statistical methods (GC [17], MI [7], and TE [40]), graph neural network based neural relational inference models (NRI [25] and GDP [35]), and attention-based models (RIVA [48]). We adapt RIVA to the unweighted setting and tune hyperparameters for all baselines using their public implementations to ensure fair comparison in experiments.

Implementation details. For COSINE, the initial basis library is generated by the LLM without system-specific templates. We set the Gumbel–Softmax temperature to $\tau = 0.3$, $\beta_{\text{KL}} = 0.1$, and $\lambda_W = 0.1$, and optimize with Adam (lr 0.1 for structure and 0.005 for coefficients). Each outer round runs 1000 inner epochs for up to 10 rounds, using validation loss and coefficient statistics to refine the library under a controlled edit budget and a maximum library size. We use GPT-OSS (20B) as the default LLM. Seeds vary in the repeated runs (Tables 1, 8 and 14). More details of synthetic experiments are presented in Section B.

Relational Inference. Table 1 shows that COSINE achieves state-of-the-art or highly competitive performance, outperforming statistical methods (GC, MI, TE) in most settings and matching or surpassing neural baselines (NRI, GDP, RIVA) across many regimes. We also report AUPR results in Table 2 in the appendix. While statistical approaches struggle with high-order nonlinearities, COSINE's joint optimization of structure and symbolic dynamics supports robust reconstruction across diverse systems. Notably, in complex scenarios such as the Kuramoto system on BA graphs, COSINE maintains a near-perfect AUC (99.85), exceeding NRI and GDP. Although performance margins vary across graphs, the results indicate that COSINE is effective at disentangling interactions within heterogeneous networks. On weighted synthetic graphs, the learned edge scores correlate with continuous weights across all six dynamics (Table 3).

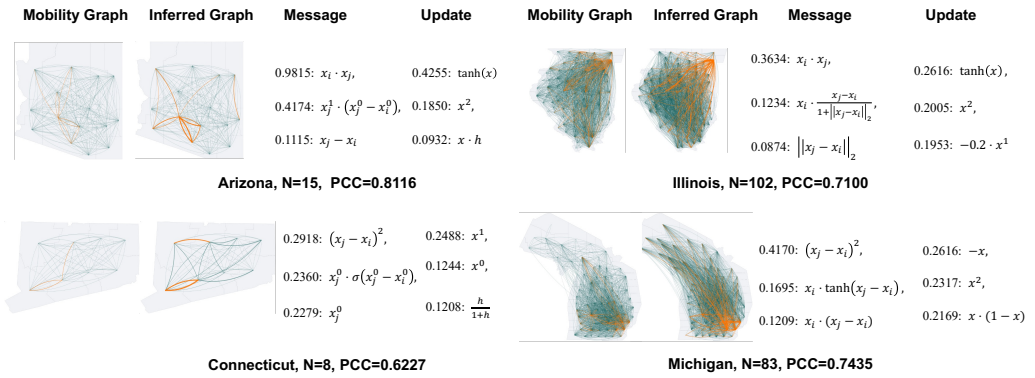

Figure 3: Results on COVID-19 epidemic data across four U.S. states. For each state, we compare the mobility graph with the inferred graph and report learned weights of dominant terms in the Message and Update modules. The inferred graph displays A_{ij}^{soft}, the edge-existence score. PCC is its Pearson correlation with the mobility graph, an weighted graph for inter-county interactions.

Mechanism Discovery. COSINE also recovers sparse, mechanism-aligned primitives rather than only edge scores. Across synthetic systems, the dominant terms identify key interaction forms such as $x_j - x_i$, sinusoidal coupling, and degree-normalized aggregation in the Update module; full discovered-term details are reported in Table 4 in the appendix. We additionally report MSE in Table 6 to evaluate the predictive accuracy of the discovered mechanisms. This MSE uses the full differentiable model without thresholding the graph or symbolic coefficients. Additional data-coverage, thresholding, and known-graph comparisons are reported in Tables 5 and 7 and Section B.8.

4.2 Results on Real-World Epidemic Data

We apply COSINE to COVID-19 data from four U.S. states (Section C) to jointly infer latent transmission structures and governing equations. For relational inference, COSINE also outperforms existing models in terms of PCC with the mobility graph (Table 16). The inferred graphs (Figure 3a) reveal pronounced regional heterogeneity: Arizona and Connecticut exhibit dense connectivity consistent with high mobility and metropolitan proximity, while Illinois and Michigan are sparser, potentially reflecting urban-rural hierarchies and localized clustering. Dynamically, the Message modules across Arizona and Illinois identify multiplicative terms ($x_i \cdot x_j$), which align with the classical mass-action effect of contact-based transmission. This suggests that cross-regional spreading intensity is closely related to the coupling between local and neighboring infection levels.

The Update modules further distinguish distinct regional transmission regimes (Figure 3b). In Arizona, the discovered terms frequently involve external pressure expressions such as $x \cdot h$, suggesting that local dynamics may be sensitive to incoming infections. Conversely, the dynamics in Illinois and Michigan are dominated by local nonlinearities—such as logistic growth $x(1-x)$ and negative feedback terms—which are consistent with endogenous spreading patterns and localized intervention effects. These findings demonstrate COSINE's capacity to extract mechanistic, physics-informed hypotheses for domain review in complex real-world systems.

4.3 Ablation Study

Ablation of LLM-Guided Evolution. As illustrated in Figure 4a, we evaluate the necessity of the LLM-guided evolution mechanism by comparing COSINE against static libraries (*w/o update*) and threshold-based pruning strategies (λ). While COSINE consistently achieves near-perfect performance (AUC ≈ 1.0) across all systems, baselines with fixed or passively filtered libraries exhibit significant performance drops and inconsistency, particularly in complex regimes like *MM* and *CMN*. This disparity underscores the "closed-world" bottleneck of static modeling, where structural inference is fundamentally capped by the initial quality of the basis set. Unlike numerical heuristics that provide only passive weight-based filtering, the LLM-guided module treats the library as a malleable hypothesis space, performing mechanistic reasoning to actively prune redundant terms and augment

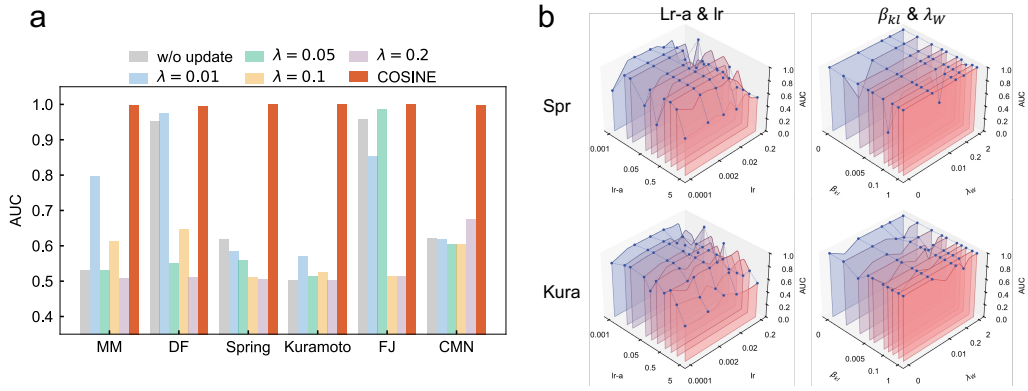

Figure 4: Ablation and sensitivity analyses. (a) LLM-guided evolution compared with fixed libraries (*w/o update*) and threshold-based pruning (λ). (b) Hyperparameter sensitivity for Spring (top row) and Kuramoto (bottom row): learning rates lr_a and lr in the left column, and regularization weights β_{KL} and λ_W in the right column. Performance is relational inference AUC.

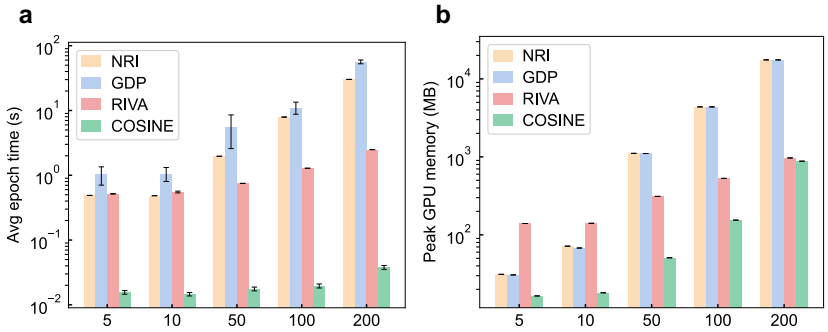

Figure 5: Computational efficiency and memory scaling comparison among COSINE, NRI, GDP, and RIVA (batch size 64, hidden dimension 256) across different network sizes $N \in \{5, 10, 50, 100, 200\}$. (a) Average inner-loop epoch time, (b) peak GPU memory during inner-loop training. Note the logarithmic scale on the vertical axes.

promising nonlinearities based on inner-loop feedback. We further ablate each module of COSINE, with results reported in Table 12. Random evolution and an oracle library provide additional controls in Table 13: COSINE remains close to the oracle across the four dynamics, while random evolution varies substantially across systems.

Impact of LLM Scale on Performance. We evaluate the effect of LLM scale across representative dynamics, with 10-run mean and standard deviation reported in Table 8 in the appendix. COSINE is weakly sensitive to model scale on simpler dynamics such as Spring and Kuramoto, while larger models generally help on more compositional nonlinear systems such as MM and CMN.

Hyperparameter Sensitivity Analysis. We test COSINE across learning rates (lr_a, lr) and sparsity weights ($\beta_{\mathrm{KL}}, \lambda_W$). As shown in Figure 4b, performance is stable across broad regions for both Spring and Kuramoto, with clear degradation mainly under unstable learning rates; the regularization weights have a milder effect once the dominant symbolic terms are identified.

Efficiency Analysis. Figure 5 compares inner-loop epoch time and peak GPU memory across graph sizes. End-to-end timings, including inner-loop training, validation, and outer-loop LLM calls, are reported separately in Table 9. In the measured local GPT-OSS 20B configuration, COSINE completes the three tested dynamics faster than NRI and GDP.

5 Limitations

Despite strong performance, COSINE has limitations. Complex dynamics can place stronger demands on the outer-loop LLM, and different LLMs may propose libraries of varying quality. Our sparse regression over a learned library improves tractability, but its expressivity is bounded and may require richer compositional primitives for systems with hidden states, delays, or higher-order interactions. The current message–update factorization also assumes pairwise graph-mediated coupling, which may not fully capture higher-order mechanisms. Under finite noisy data, symbolic equation discovery may be non-identifiable, with near-equivalent terms blurring mechanistic conclusions. For real-world epidemic data, the inferred structures should be interpreted cautiously because aggregate observations can mix true transmission effects with reporting bias, policy changes, and unobserved interventions. The graph learner and message passing represent node-pair interactions, so computation and memory grow at least as $O(N^2)$. GPU memory is the main bottleneck, as in other relational-inference models. If GPU memory allows, the implementation can run on larger graphs, although cost may be prohibitive.

6 Conclusion

In this paper, we presented COSINE, a differentiable framework for interpretable relational inference that jointly learns latent interaction graphs and sparse dynamical equations. COSINE models dynamics as sparse combinations over message/update basis libraries and employs an LLM to iteratively refine these libraries based on optimization feedback. Beyond physics-inspired systems, the message–update factorization provides a general template for *networked dynamics*, where interactions may encode biochemical regulation, social influence, mobility-driven transmission, or other mechanisms. Experiments on synthetic and real-world systems show that COSINE achieves strong structural recovery while yielding mechanism-aligned terms and favorable efficiency and scalability.

Impact Statement

This work develops methods for interpretable structure and equation discovery in networked dynamical systems. In our real-world experiment, we use public, aggregate COVID-19 data rather than individual-level health or mobility traces. Still, epidemic modeling is a sensitive domain: inferred structures may reflect reporting bias, testing policy changes, population heterogeneity, or unobserved interventions. COSINE should therefore be used as an exploratory tool for generating mechanistic hypotheses, not as standalone evidence for public-health decisions or regional comparisons without domain review and external validation.

Acknowledgments and Disclosure of Funding

We thank the anonymous reviewers for their helpful comments.

The authors acknowledge the support from National Natural Science Foundation of China (Grant No. T2293771), the STI 2030Major Projects (Grant No. 2022ZD0211400), the National Natural Science Foundation of China (Grant No. 42476191), the National Natural Science Foundation of China (Grant No. 42230406), and the New Cornerstone Science Foundation through the XPLORER PRIZE.

The authors declare no competing interests.

References

[1] Rediet Abebe, Jon Kleinberg, David Parkes, and Charalampos E Tsourakakis. Opinion dynamics with varying susceptibility to persuasion. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1089–1098, 2018.

[2] Alex Arenas, Albert Díaz-Guilera, Jurgen Kurths, Yamir Moreno, and Changsong Zhou. Synchronization in complex networks. *Physics reports*, 469(3):93–153, 2008.

[3] Mohammad Amin Basiri and Sina Khanmohammadi. Sindyg: sparse identification of nonlinear dynamical systems from graph-structured data, with applications to stuart–landau oscillator networks. *Journal of Complex Networks*, 13(5):cnaf029, 2025.

[4] Luca Biggio, Tommaso Bendinelli, Alexander Neitz, Aurelien Lucchi, and Giambattista Parascandolo. Neural symbolic regression that scales. In *International Conference on Machine Learning*, pages 936–945. Pmlr, 2021.

[5] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

[6] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Sparse identification of nonlinear dynamics with control (sindyc). *IFAC-PapersOnLine*, 49(18):710–715, 2016.

[7] Atul J Butte and Isaac S Kohane. Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. In *Biocomputing 2000*, pages 418–429. World Scientific, 1999.

[8] Claudio Castellano, Santo Fortunato, and Vittorio Loreto. Statistical physics of social dynamics. *Reviews of Modern Physics*, 81(2):591, 2009.

[9] Siyuan Chen, Jiahai Wang, and Guoqing Li. Neural relational inference with efficient message passing mechanisms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7055–7063, 2021.

[10] Miles Cranmer, Alvaro Sanchez Gonzalez, Peter Battaglia, Rui Xu, Kyle Cranmer, David Spergel, and Shirley Ho. Discovering symbolic models from deep learning with inductive biases. *Advances in neural information processing systems*, 33:17429–17442, 2020.

[11] Miles D Cranmer, Rui Xu, Peter Battaglia, and Shirley Ho. Learning symbolic physics with graph networks. *arXiv preprint arXiv:1909.05862*, 2019.

[12] Noah E Friedkin and Eugene C Johnsen. Social influence and opinions. *Journal of Mathematical Sociology*, 15(3-4):193–206, 1990.

[13] Ting-Ting Gao and Gang Yan. Autonomous inference of complex network dynamics from incomplete and noisy data. *Nature Computational Science*, 2(3):160–168, 2022.

[14] P Garcia, A Parravano, MG Cosenza, J Jiménez, and A Marcano. Coupled map networks as communication schemes. *Physical Review E*, 65(4):045201, 2002.

[15] Pawan Goyal and Peter Benner. Discovery of nonlinear dynamical systems using a runge–kutta inspired dictionary-based sparse regression approach. *Proceedings of the Royal Society A*, 478 (2262):20210883, 2022.

[16] Colin Graber and Alexander Schwing. Dynamic neural relational inference for forecasting trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 1018–1019, 2020.

[17] Clive WJ Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica: journal of the Econometric Society*, pages 424–438, 1969.

[18] Arya Grayeli, Atharva Sehgal, Omar Costilla Reyes, Miles Cranmer, and Swarat Chaudhuri. Symbolic regression with a learned concept library. *Advances in Neural Information Processing Systems*, 37:44678–44709, 2024.

[19] Seungwoong Ha and Hawoong Jeong. Towards automated statistical physics: Data-driven modeling of complex systems with deep learning. *arXiv preprint arXiv:2001.02539*, 2020.

[20] Seungwoong Ha and Hawoong Jeong. Learning heterogeneous interaction strengths by trajectory prediction with graph neural network. In *International Conference on Learning Representations*, 2023.

[21] Moritz Hoffmann, Christoph Fröhner, and Frank Noé. Reactive sindy: Discovering governing reactions from concentration data. *The Journal of chemical physics*, 150(2), 2019.

[22] Eugene M Izhikevich. *Dynamical systems in neuroscience*. MIT press, 2007.

[23] Pierre-Alexandre Kamienny, Stéphane d'Ascoli, Guillaume Lample, and François Charton. End-to-end symbolic regression with transformers. *Advances in Neural Information Processing Systems*, 35:10269–10281, 2022.

[24] Guy Karlebach and Ron Shamir. Modelling and analysis of gene regulatory networks. *Nature Reviews Molecular Cell Biology*, 9(10):770–780, 2008.

[25] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In *International Conference on Machine Learning*, pages 2688–2697. PMLR, 2018.

[26] Yoshiki Kuramoto. Self-entrainment of a population of coupled non-linear oscillators. In *International Symposium on Mathematical Problems in Theoretical Physics: January 23–29, 1975, Kyoto University, Kyoto/Japan*, pages 420–422. Springer, 1975.

[27] Ying-Cheng Lai. Finding nonlinear system equations and complex network structures from data: A sparse optimization approach. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 31(8), 2021.

[28] Yaguang Li, Chuizheng Meng, Cyrus Shahabi, and Yan Liu. Structure-informed graph auto-encoder for relational inference and simulation. In *ICML Workshop on Learning and Reasoning with Graph-Structured Data*, volume 8, page 2, 2019.

[29] Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks. *arXiv preprint arXiv:2404.19756*, 2024.

[30] Sindy Löwe, David Madras, Richard Zemel, and Max Welling. Amortized causal discovery: Learning to infer causal graphs from time-series data. In *Conference on Causal Learning and Reasoning*, pages 509–525. PMLR, 2022.

[31] Ziyu Lu, Anika Tabassum, Shruti Kulkarni, Lu Mi, J Nathan Kutz, Eric Shea-Brown, and Seung-Hwan Lim. Attention for causal relationship discovery from biological neural dynamics. *arXiv preprint arXiv:2311.06928*, 2023.

[32] Nour Makke and Sanjay Chawla. Interpretable scientific discovery with symbolic regression: a review. *Artificial Intelligence Review*, 57(1):2, 2024.

[33] Niall M Mangan, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Inferring biological networks by sparse identification of nonlinear dynamics. *IEEE Transactions on Molecular, Biological, and Multi-Scale Communications*, 2(1):52–63, 2017.

[34] Raha Moraffah, Paras Sheth, Mansooreh Karami, Anchit Bhattacharya, Qianru Wang, Anique Tahir, Adrienne Raglin, and Huan Liu. Causal inference for time series analysis: Problems, methods and evaluation. *Knowledge and Information Systems*, 63(12):3041–3085, 2021.

[35] Liming Pan, Cheng Shi, and Ivan Dokmanic. A graph dynamics prior for relational inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 14508–14516, 2024.

[36] Romualdo Pastor-Satorras, Claudio Castellano, Piet Van Mieghem, and Alessandro Vespignani. Epidemic processes in complex networks. *Reviews of Modern Physics*, 87(3):925, 2015.

[37] Jie Peng, Pei Wang, Nengfeng Zhou, and Ji Zhu. Partial correlation estimation by joint sparse regression models. *Journal of the American Statistical Association*, 104(486):735–746, 2009.

[38] Kai Ruan, Yilong Xu, Ze-Feng Gao, Yang Liu, Yike Guo, Ji-Rong Wen, and Hao Sun. Discovering physical laws with parallel symbolic enumeration. *Nature Computational Science*, pages 1–14, 2025.

[39] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85, 2009.

[40] Thomas Schreiber. Measuring information transfer. *Physical Review Letters*, 85(2):461, 2000.

[41] Alex M Stanković, Aleksandar A Sarić, Andrija T Sarić, and Mark K Transtrum. Data-driven symbolic regression for identification of nonlinear dynamics in power systems. In *2020 IEEE Power & Energy Society General Meeting (PESGM)*, pages 1–5. IEEE, 2020.

[42] Ho Fung Tsoi, Dylan Rankin, Cecile Caillol, Miles Cranmer, Sridhara Dasu, Javier Duarte, Philip Harris, Elliot Lipeles, and Vladimir Loncar. Symbolfit: Automatic parametric modeling with symbolic regression. *Computing and Software for Big Science*, 9(1):12, 2025.

[43] Silviu-Marian Udrescu and Max Tegmark. Ai feynman: A physics-inspired method for symbolic regression. *Science advances*, 6(16):eaay2631, 2020.

[44] Aoran Wang and Jun Pang. Iterative structural inference of directed graphs. *Advances in Neural Information Processing Systems*, 35:8717–8730, 2022.

[45] Ezra Webb, Ben Day, Helena Andres-Terre, and Pietro Lió. Factorised neural relational inference for multi-interaction systems. *arXiv preprint arXiv:1905.08721*, 2019.

[46] Sebastian Johann Wetzel, Seungwoong Ha, Raban Iten, Miriam Klopotek, and Ziming Liu. Interpretable machine learning in physics: A review. *arXiv preprint arXiv:2503.23616*, 2025.

[47] Kaze WK Wong and Miles Cranmer. Automated discovery of interpretable gravitational-wave population models. *arXiv preprint arXiv:2207.12409*, 2022.

[48] Ruizi Wu, Liming Pan, and Linyuan Lü. Riva: Efficient relational inference with variate attention. *Neural Networks*, page 107748, 2025.

[49] Tailin Wu, Thomas Breuel, Michael Skuhersky, and Jan Kautz. Discovering nonlinear relations with minimum predictive information regularization, 2020.

[50] Zihan Yu, Jingtao Ding, and Yong Li. Discover network dynamics with neural symbolic regression. *Nature Computational Science*, pages 1–13, 2025.

[51] Shangyu Zhao, Changming Cheng, Miaomiao Lin, and Zhike Peng. Physics-informed deep sparse regression network for nonlinear dynamical system identification. *Journal of Sound and Vibration*, 595:118796, 2025.

[52] Jingxuan Zhu, Juexin Wang, Weiwei Han, and Dong Xu. Neural relational inference to learn long-range allosteric interactions in proteins from molecular dynamics simulations. *Nature Communications*, 13(1):1–16, 2022.

A Details of Method

A.1 Training Process of COSINE

The overall training procedure of COSINE is summarized in Algorithm 1. The framework follows a nested co-optimization structure consisting of an outer symbolic discovery loop and an inner joint optimization loop.

Inner Loop: Joint Structure–Equation Optimization. Within each outer round, with the basis function library Θ fixed, the model seeks to simultaneously recover the network structure and the governing dynamical equations. The graph generator parameters $\mathbf{\Psi}$ and the sparse regression coefficients $\mathbf{W}$ are jointly updated via gradient descent to minimize the total loss $\mathcal{L}_{\text{total}}$. This inner loop leverages the differentiability of the Gumbel–Softmax edge sampling and the linear basis expansion to efficiently navigate the high-dimensional parameter space, identifying the most probable interaction structure and the corresponding sparse weights that characterize the system's evolution.

Outer Loop: LLM-Guided Library Evolution. After the inner loop converges, the LLM $\mathcal{M}$ acts as a symbolic supervisor to refine the library Θ. It analyzes the training results—including validation performance scores, basis term weights (importance measures), and residual patterns—to decide which terms should be pruned due to negligible contributions or augmented to capture remaining nonlinearities. This iterative process of hypothesis generation, evaluation, and reflection allows COSINE to escape the "closed-world" limitation of traditional sparse regression, progressively evolving towards a parsimonious yet accurate symbolic representation of the underlying mechanisms.

Algorithm 1 Overall Training Procedure of COSINE

Input: Node trajectories $\mathbf{X}$, initial library $\Theta^{(0)}$, max rounds R, max epochs E, weights $\beta_{\text{KL}}, \lambda_W$
Output: Learned adjacency $\mathbf{A}$, coefficients $\mathbf{W}$, discovered library $\Theta^\star$
Initialize structure parameters $\mathbf{\Psi}$ and regression weights $\mathbf{W}$
Set best library $\Theta^\star \leftarrow \Theta^{(0)}$ and best validation loss $\mathcal{L}_{\text{val}}^\star \leftarrow +\infty$
for $r = 0$ **to** $R - 1$ **do**
  Fix current library $\Theta^{(r)}$
  for $e = 1$ **to** E **do**
    Sample mini-batch $(\mathbf{x}^t, \mathbf{x}^{t+1})$ from $\mathbf{X}$
    Compute soft adjacency $\mathbf{A}^{\text{soft}}$ from $\mathbf{\Psi}$ using Gumbel–Softmax
    Compute message and update features using $\Theta^{(r)}$ and predict $\hat{\mathbf{x}}^{t+1}$
    Compute losses $\mathcal{L}_{\text{nll}}, \mathcal{L}_A, \mathcal{L}_W$ and total loss $\mathcal{L}_{\text{total}}$
    Update $\mathbf{\Psi}$ and $\mathbf{W}$ by gradient descent on $\mathcal{L}_{\text{total}}$
  end for
  Evaluate validation loss $\mathcal{L}_{\text{val}}^{(r)}$ and record mean absolute weights for each basis term
  if $\mathcal{L}_{\text{val}}^{(r)} < \mathcal{L}_{\text{val}}^\star$ **then**
    $\mathcal{L}_{\text{val}}^\star \leftarrow \mathcal{L}_{\text{val}}^{(r)}$, $\Theta^\star \leftarrow \Theta^{(r)}$, and store parameters $\mathbf{\Psi}, \mathbf{W}$
  end if
  Build LLM prompt from $\Theta^\star$, importance statistics, and residual patterns
  Query LLM $\mathcal{M}$ to obtain a modified library $\widetilde{\Theta}$ via pruning and augmentation
  Set $\Theta^{(r+1)} \leftarrow \widetilde{\Theta}$ and reinitialize $\mathbf{W}$; optionally warm-start $\mathbf{\Psi}$
end for
return $\Theta^\star, \mathbf{\Psi}, \mathbf{W}$

A.2 Basis Search Implementation Details

The outer-loop optimization of COSINE can be viewed as a search process over the discrete space of symbolic structures via *basis library editing*. Each round of LLM-guided refinement, involving the augmentation or pruning of basis terms, essentially alters the feasible subspace of dynamical expressions. While sophisticated search algorithms such as Monte Carlo Tree Search (MCTS) could be employed to explicitly manage a multi-branch search tree and balance exploration versus exploitation, such approaches entail significant computational overhead due to recursive branching

and the requirement for multiple full model training and evaluation cycles. In settings that require frequent LLM interactions, MCTS introduces unnecessary complexity that may not be justified by the marginal gains.

Instead, we adopt a more lightweight and reproducible **best-so-far** strategy. This approach maintains a single current optimal basis library and only accepts a proposed candidate if it yields a strictly **lower** validation loss. Formally, let $\Theta^{(r)}$ denote the basis library at outer iteration r. Its performance is evaluated by the inner-loop joint optimization (of structural parameters $\mathbf{\Psi}$ and regression coefficients $\mathbf{W}$) as the total validation loss:

$$\mathcal{L}_{\text{val}}(\Theta) = \mathcal{L}_{\text{total}} \quad \text{(evaluated on the validation set)}. \tag{18}$$

We maintain the optimal library $\Theta^\star$ identified throughout the search:

$$\Theta^\star = \arg \min_{\Theta \in \{\Theta^{(0)}, \ldots, \Theta^{(r)}\}} \mathcal{L}_{\text{val}}(\Theta). \tag{19}$$

In each subsequent outer iteration, $\Theta^\star$along with its corresponding importance statistics and training residual patternsis provided as core context in the prompt to generate a candidate library $\widetilde{\Theta}$. Following a round of inner-loop joint training, we obtain $\mathcal{L}_{\text{val}}(\widetilde{\Theta})$. If $\mathcal{L}_{\text{val}}(\widetilde{\Theta}) < \mathcal{L}_{\text{val}}(\Theta^\star)$, we update $\Theta^\star \leftarrow \widetilde{\Theta}$. Otherwise, the candidate is rejected, and $\Theta^\star$ is preserved for the next iteration.

This strategy serves as a "greedy update with rejection" mechanism rather than a full tree search. By not explicitly maintaining multi-branch states, we mitigate the risk of performance degradation arising from LLM pruning errors or the introduction of noisy augmentation terms. The primary advantages of this approach are its controllable computational cost and implementation simplicity, which effectively leverages the strong feedback signals from numerical optimization and sparsity constraints to steer the symbolic search towards physically consistent and parsimonious mechanism regions.

A.3 LLM Details

For the core experiments presented in this work, we utilize the **GPT-OSS (20B)** model as the default Large Language Model. This model is deployed locally using the Ollama framework and runs on an NVIDIA RTX 4090 GPU, ensuring data privacy and low-latency feedback during the outer-loop library evolution. For the analysis of real-world epidemic data, we employ the **MiMo-V2-Flash** model provided via the OpenRouter API.

A.4 Prompt Details

In the COSINE framework, we utilize three distinct prompt templates to coordinate the interaction between numerical optimization and LLM-based symbolic discovery: the *System Prompt*, the *Initial Basis Prompt*, and the *Refine Basis Prompt*. The *System Prompt* establishes the foundational operational framework, including strict variable scoping, tensor dimension constraints, and numerical stability expectations. The *Initial Basis Prompt* is designed for the zero-shot generation of the starting candidate library. The *Refine Basis Prompt* incorporates multi-faceted feedback from the inner-loop trainingincluding validation performance, coefficient importance statistics, and residual patternsto guide the iterative evolution of the library.

A.4.1 System Prompt

To ensure the symbolic consistency and numerical stability of the generated basis functions, we provide the LLM supervisor with a comprehensive system prompt. This prompt establishes a strict operational framework, defining available tensor variables, mathematical primitives, and dimension-matching constraints for both message-passing and node-update modules. By explicitly outlining forbidden operations and stability expectations (e.g., handling singularities via small epsilon values), the prompt guides the LLM to propose parsimonious yet physically grounded candidate terms while adhering to the differentiable PyTorch-based execution environment.

The complete system prompt used to initialize and guide the basis library evolution is presented in Listing 1.

```
You are an expert in discovering dynamical systems and their interaction structures using
     sparse symbolic regression.

Your Task:
- Act as an expert in dynamical systems, sparse regression, and PyTorch expressions.
- Only design / refine the basis function library.

Output Requirements (MUST be strictly followed):
1) Respond with a single JSON object and nothing else (no markdown or explanatory text).
2) The JSON must contain exactly two array fields: "message_terms" and "update_terms".
   - Functions in message_terms are used to model interactions between nodes.
   - Functions in update_terms are used to model each node's own state update.
3) Each item must contain exactly three fields: {"name", "expr", "type"}, where type in
    {"vector", "scalar"}.
   - The field name must be exactly "expr".
   - No extra fields are allowed, no empty field is allowed.
4) Within the same stream, names must be unique.

Available tensors and helpers:
- Message stream only:
  xi, xj shape: [B, N, N, D]; diff = xj - xi
- Update stream only:
  x in [B, N, D]; h in [B, N, D]; deg in [B, N, 1]
- Shared helpers: torch, F, pi, inf

Strict variable scoping rules (NO EXCEPTIONS):
- Expressions in message_terms may only use: xi, xj, diff (and shared helpers); never use
     x / h / deg.
- Expressions in update_terms may only use: x, h, deg (and shared helpers); never use xi
    / xj / diff.
- Never introduce new symbols.

Shape and type rules:
- Outputs of message_terms must be broadcastable to:
  - Vector terms: [B, N, N, D]
  - Scalar terms: [B, N, N, 1]
- Outputs of update_terms must be:
  - Vector terms: [B, N, D]
  - Scalar terms: [B, N, 1]
- Scalar terms MUST keep a trailing singleton dimension (..., 1).

Strictly forbidden patterns:
- In update_terms: NEVER reference variables such as "msg" or "message"; only "h" exists.
- In update_terms: NEVER create an extra N dimension (no [B, N, N, *]).
- NEVER use reshape/reordering operations:
  view / reshape / permute / transpose / unsqueeze / squeeze.
- Forbidden constructors / stacking operations:
  torch.tensor / torch.zeros / torch.ones / torch.cat / torch.stack.
  If constants are needed, use torch.zeros_like(x) or torch.ones_like(x).

Expression rules:
- PyTorch-only expressions (torch.* / F.* and basic arithmetic).
- No numpy, no math.*, no control flow.
- No attributes containing "__" or starting with an underscore.

Recommended primitives and templates (allowed if they obey scoping + shape rules):
- Polynomial / power: x, x*x, torch.pow(x, 2), torch.pow(x, 3), torch.sqrt(torch.clamp(x,
     min=0)).
- Trigonometric: torch.sin(x), torch.cos(x), torch.tan(x).
- Exponential / log: torch.exp(x), torch.log(torch.clamp(x, min=1e-6)).
- Activations / saturation: torch.sigmoid(x), torch.tanh(x), F.relu(x), F.softplus(x).
- Rational / fractional (numerically stable):
  - Inverse: 1.0 / (x + 1e-6)
  - Saturation (Hill-like): x / (1.0 + torch.abs(x))
  - Squared saturation: x / (1.0 + x*x)
  Use small eps (e.g., 1e-6) to avoid division-by-zero.

Numerical stability expectations:
- Avoid singularities: add eps in denominators; clamp inputs to log/sqrt.
- Prefer bounded transforms (tanh/sigmoid) when exploring complex nonlinearity.
- Occam's Razor: Prefer simpler expressions. Only introduce complexity (e.g., rational
    functions, high-order polynomials) if simple linear or low-order terms fail to
    capture the dynamics.
- expr must be a single valid Python expression parseable by:
  ast.parse(expr, mode="eval").
- No placeholders are allowed inside expr.
- Expression length must be less than 500 characters.
- Numerical safety: when dividing by a potentially small quantity, add + 1e-6.
- torch.norm MUST be written as:
  torch.norm(x, dim=-1, keepdim=True)
```

```
  and NEVER as torch.norm(x, -1, ...).

Multi-channel (D > 1) channel slicing rules (CRITICAL):
- NEVER use x[..., c] / xj[..., c] (this drops the last dimension).
- ALWAYS use x[..., c:c+1] / xj[..., c:c+1], and never write c:c.
- Cross-channel interactions should be designed as scalar terms (then broadcast to D).

Budget constraints:
- Each call specifies max_terms.
- Must satisfy:
  - len(message_terms) <= max_terms
  - len(update_terms) <= max_terms
  - message_terms and update_terms must each contain at least one term.

Minimal example (FORMAT ONLY):
{
  "message_terms": [
    {"name": "diff", "expr": "xj - xi", "type": "vector"}
  ],
  "update_terms": [
    {"name": "h", "expr": "h", "type": "vector"}
  ]
}
```

Listing 1: System Prompt for the LLM Basis Library Supervisor

A.4.2 Init Basis Prompt

The *Initial Basis Prompt* (shown in Listing 2) is used to initialize the symbolic library $\Theta^{(0)}$. It instructs the LLM to prioritize parsimony and core physical primitives (e.g., linear differences and basic state terms) before exploring higher-order nonlinearities. The prompt incorporates several key environment variables:

- `D`: Represents the feature dimension of node states.
- `system description`: Provides a natural language functional description of the underlying system. Notably, this variable is **exclusively** utilized for the real-world epidemic data experiments to provide domain-specific grounding; in all synthetic benchmarks, it is omitted.
- `max_terms`: Specifies the maximum number of basis functions permitted for each of the message and update streams, ensuring complexity control.

```
You are designing the initial basis function library for COSINE dynamics discovery.

Follow all hard rules in `system_prompt.txt`. This file only adds initialization-phase
    guidance.

Problem snapshot:
- State dimension D = {feature_dim}
- System description: {description}
- Budget: at most {max_terms} terms in each stream

Initialization strategy (Simplicity First):
1) Message function (message_terms):
    - Start with simple and plausible interaction mechanisms (e.g., linear difference `xj
        - xi`).
2) Update function (update_terms):
    - Include core baseline terms (e.g., `x`, `h`) to represent basic self-dynamics and
        neighbor influence.
    - Keep terms low-order (linear or quadratic) initially.
3) Multi-channel (D>1):
    - Use simple channel-wise terms before exploring cross-channel interactions.

Advanced Exploration (if the system description suggests high complexity):
- If the system is known to be highly non-linear (e.g., biological, chemical), you may
    include terms from the following categories:
  - Trigonometric: sin(.) / cos(.) on diff or state.
  - Activation/saturation: tanh(.) / sigmoid(.).
  - Rational: x/(1+|x|) or 1/(x+eps).

Graph-specific normalization (update_terms only):
- If the system likely depends on node degree, include `h / (deg + 1e-6)`.

Avoid near-duplicate terms; keep names stable and unique.
```

Listing 2: Initial Basis Prompt Template

A.4.3 Refine Basis Prompt

The *Refine Basis Prompt* (shown in Listing 3) governs the iterative evolution of the symbolic library during the outer-loop optimization. In alignment with the "best-so-far" strategy described in Section A.2, this prompt acts as a symbolic editor that modifies the current optimal library based on numerical feedback. It incorporates two types of historical context:

- `recent_rounds`: A summary of the symbolic libraries and training results (NLL and coefficient importance statistics) from the most recent 3 iterations. This allows the LLM to identify persistent low-weight terms for pruning or recognize when the search has stalled in a specific functional subspace.
- `best_round_details`: The complete specification of the global optimal library $\Theta^\star$ identified thus far. This includes naming, mathematical expressions, and the learned importance of each term, serving as the definitive baseline for subsequent mutations and complexity escalation.

```
You are optimizing basis function selection based on training feedback. Reasoning: High.

All hard constraints (JSON format, tensors, scoping, shapes, forbidden operations, etc.)
    are specified in system_prompt.txt and must be strictly followed.

Inputs provided with each call include:
- Recent 3 rounds (each round has nll + Message terms + Update terms; each term includes
    name, mean|w|, expr):
{recent_rounds}
- Best round so far (nll + Message terms + Update terms; each term includes name, mean|w
    |, expr):
{best_round_details}

=== DECISION LOGIC ===

Step 1: Optimize and explore based on historical feedback, using the best round's basis
    functions as the starting point.

Step 2: Weight-guided refinement and Complexity Escalation
Use your reasoning to refine the library based on weights and NLL:
- Retain high-weight terms: If a simple term (e.g., linear) has a high weight, KEEP it.
    Do not replace it with a complex version unless the NLL is high and progress has
    stalled.
- Prune low-weight terms: Remove terms that consistently show near-zero weights.
- Progressive Complexity: introduce more complex terms (e.g., rational, activations, high
    -order polynomials) if the current simple terms are insufficient to reduce the NLL.

Step 2b: Structure-guided mutation (Escalate only when needed)
If the fit plateaus or NLL remains high, try MUTATING weak terms into slightly more
    complex forms:
- Escalation Path: Linear -> Polynomial (x^2) -> Non-linear (sin/tanh) -> Rational (x/(1+
    x)).
- Degree normalization (update_terms only): If neighbor influence `h` is weak, try `h/(
    deg+1e-6)`.
- Interaction nonlinearity (message_terms): If `diff` is weak, try `sin(diff)` or `tanh(
    diff)`.

=== KEY INSIGHT: Occam's Razor ===
Prefer the simplest expression that explains the data. A simple term with a high weight
    is much better than a complex term with a similar weight.

=== BUDGET ===
- Budget: at least 1 term per stream, at most {max_terms} terms per stream.
- Ensure the basis function library can adequately and clearly express the dynamical
    equations.
- Critical requirement: within the same stream, every term name must be UNIQUE.
```

Listing 3: Refine Basis Prompt Template

Table 2: Relational inference performance (AUPR %) on different graphs with $N = 50$ nodes across various dynamical systems. Boldface marks the highest accuracy.

DYN.	GRAPH	GC	MI	TE	NRI	GDP	RIVA	COSINE
MM	ER-50	15.51	31.72	13.27	88.90±0.30	12.27±0.51	12.20±0.47	**98.32±1.68**
	BA-50	12.12	48.95	16.82	71.91±0.43	9.08±0.00	8.00±0.29	**93.72±0.75**
	WS-50	14.50	76.87	5.87	**97.80±0.26**	4.98±0.11	4.52±0.15	91.92±0.97
DIFF	ER-50	18.06	16.54	19.35	96.43±0.31	61.88±12.92	12.34±0.28	**97.91±0.64**
	BA-50	24.73	15.29	18.66	52.62±27.23	25.69±2.97	8.04±0.25	**90.15±7.11**
	WS-50	12.00	17.19	47.85	82.84±11.18	6.06±0.84	4.53±0.23	**100.00±0.10**
SPR	ER-50	11.91	12.18	13.14	55.79±16.66	**96.03±2.90**	12.46±0.79	91.87±0.98
	BA-50	8.99	10.64	14.57	25.13±1.47	90.36±7.52	9.28±0.31	**100.00±0.10**
	WS-50	4.83	4.98	4.82	23.25±15.31	52.98±41.72	4.75±0.21	**100.00±0.10**
KURA	ER-50	13.57	48.21	44.25	85.78±7.13	18.85±1.72	29.98±2.76	**99.87±0.07**
	BA-50	9.01	78.43	53.03	87.34±2.06	31.30±4.58	23.48±1.26	**99.10±0.19**
	WS-50	10.90	**100.00**	68.47	52.75±47.25	**100.00±0.00**	36.22±9.04	**100.00±0.10**
FJ	ER-50	75.28	14.91	48.70	49.87±37.97	15.60±2.69	12.71±0.57	**98.37±1.63**
	BA-50	73.91	9.34	51.01	51.32±9.13	11.84±3.67	7.99±0.07	**92.93±7.07**
	WS-50	97.02	11.70	73.88	20.34±11.38	6.41±1.21	4.50±0.26	**100.00±0.10**
CMN	ER-50	14.23	55.15	28.56	41.59±0.26	51.42±5.83	12.89±0.25	**100.00±0.10**
	BA-50	11.51	**71.95**	40.45	33.77±0.89	31.03±2.88	8.46±0.46	70.66±3.52
	WS-50	14.92	**85.96**	56.24	16.68±5.68	4.70±0.07	5.36±0.53	83.09±2.03

Table 3: PCC of COSINE between A_{ij}^{soft} and true edge weights in weighted synthetic systems. Values are mean ± standard deviation.

Dyn.	PCC
CMN	0.7327±0.0415
Diff	0.9774±0.0142
FJ	0.9845±0.0105
Kuramoto	0.9962±0.0028
MM	0.9581±0.0216
Spring	0.9032±0.0348

B Details of Synthetic Experiment

B.1 Additional Relational Inference Metrics

Table 2 provides an AUPR-based view of relational inference. The trend is consistent with the AUC results in Table 1: COSINE is usually strongest or close to the strongest method, especially on ER and BA graphs and on nonlinear systems where statistical baselines are less reliable. AUPR also exposes several graph-specific exceptions, such as WS-MM and some CMN settings, where neural or statistical baselines can be competitive. This suggests that COSINE's advantage is broad but still depends on the graph family and the strength of the relational signal.

B.2 Weighted synthetic graphs

We test whether the learned edge score A_{ij}^{soft} tracks continuous interaction weights by sampling each nonzero synthetic edge weight from $[0.5, 1.5]$. Table 3 reports Pearson correlation with the true weights. The mean PCC exceeds 0.7 for all six dynamics, supporting the use of the score as a ranking proxy in this weighted setting.

B.3 Discovered Terms and Data Sensitivity

Analysis of discovered terms. Table 4 shows that COSINE recovers the main symbolic primitives for most systems under the default data budget. The harder cases are MM and CMN, where the message terms are only partially recovered with limited observations. This is expected: both systems involve nonlinear compositional mechanisms, and the model must identify symbolic interactions while also recovering the unknown graph.

Table 4: Dominant terms discovered by COSINE on ER graphs (Table 1), ranked by absolute coefficient magnitude. *Term 1–3* denote the top three terms for *Message* and *Update* modules. *Term Acc.* reports row-wise primitive coverage scores. x: node state; h_i: aggregated message; k_i: degree; $\epsilon = 10^{-6}$.

DYN.	GROUND-TRUTH DYNAMICS	COMPONENT	TERM 1	TERM 2	TERM 3	TERM ACC.
MM	$\dot{x}_i = -x_i + \sum_{j\in\mathcal{N}(i)} \frac{A_{ij}}{k_i}\frac{x_j}{1+x_j}$	MESSAGE	$(x_j - x_i)^2$ (0.31)	$\lvert x_j - x_i\rvert$ (0.28)	$x_i x_j$ (0.20)	0.00
		UPDATE	$\frac{h_i}{k_i+\epsilon}$ (0.70)	x (0.42)	$\sin(x)$ (0.25)	1.00
DIFF	$\dot{x} = -\beta L x, \quad L = D_{\text{IN}} - A$	MESSAGE	$x_j - x_i$ (0.17)	$x_i x_j$ (0.12)	$\cos(x_j - x_i)$ (0.11)	1.00
		UPDATE	$\frac{x+h_i}{k_i+1+\epsilon}$ (0.31)	$\frac{h_i}{k_i+1+\epsilon}$ (0.30)	$\cos(x)$ (0.08)	1.00
SPR	$\ddot{\mathbf{p}}_i = -k\sum_{j\in\mathcal{N}(i)} A_{ij}(\mathbf{p}_i - \mathbf{p}_j)$	MESSAGE	$x_j - x_i$ (0.02)	$x_i^{(0)}x_j^{(1)}$ (0.01)	$\sin(x_i^{(0)}x_j^{(1)})$ (0.01)	1.00
		UPDATE	$\frac{h_i}{k_i+1}$ (0.04)	$\frac{x}{k_i+1}$ (0.03)	x (0.02)	1.00
KURA	$\dot{\phi}_i = \omega_i + \sum_{j\in\mathcal{N}(i)} A_{ij}\kappa\sin(\phi_j - \phi_i)$	MESSAGE	$\sin(x_j - x_i)$ (0.53)	$\lvert x_j - x_i\rvert$ (0.16)	$x_i^{(1)}x_j^{(2)}$ (0.14)	1.00
		UPDATE	$\frac{h_i}{k_i+1}$ (0.48)	$\frac{xh_i}{k_i+1}$ (0.19)	h_i (0.08)	1.00
FJ	$x_i^{t+1} = \frac{1}{k_i+1}\left(\sum_{j\in\mathcal{N}(i)} A_{ij}x_j^t + s_i\right)$	MESSAGE	$x_j - x_i$ (0.25)	$x_i^{(0)}x_j^{(1)}$ (0.03)	$x_i x_j$ (0.03)	1.00
		UPDATE	$\frac{h_i}{k_i+1}$ (0.97)	x (0.06)	h_i (0.06)	1.00
CMN	$\theta_i^{t+1} = (1-s)f(\theta_i^t) + \frac{s}{k_i}\sum_{j\in\mathcal{N}(i)} A_{ij}f(\theta_j^t)$	MESSAGE	$x_i x_j$ (0.33)	$(x_j - x_i)^2$ (0.29)	$x_j - x_i$ (0.12)	0.50
		UPDATE	$\frac{h_i}{k_i+1}$ (2.66)	$\tanh(h_i)$ (1.18)	x (1.12)	1.00

Table 5: Symbolic recovery for MM and CMN with 200×10 (trajectories $\times$ sampled steps). x: node state; h_i: aggregated message; k_i: degree; $\varepsilon = 10^{-6}$.

DYN.	MESSAGE TERMS	UPDATE TERMS	MSE
MM	$x_j/(1+x_j)$	x_i, $h_i/(k_i+\varepsilon)$	9.52×10^{-7}
CMN	x_j, x_j^2	h_i, x_i, $h_i/(k_i+\varepsilon)$	9.78×10^{-5}

Table 6: Prediction error (MSE ↓) on different graphs with $N = 50$ nodes across various dynamical systems. Boldface marks the lowest MSE.

DYN.	GRAPH	NRI	GDP	RIVA	COSINE
MM	ER-50	$1.7018\times10^{-5}\pm2.8853\times10^{-7}$	$\mathbf{3.3862\times10^{-6}\pm8.9822\times10^{-8}}$	$9.4050\times10^{-5}\pm1.4024\times10^{-5}$	$4.4400\times10^{-6}\pm2.2700\times10^{-6}$
	BA-50	$3.5539\times10^{-5}\pm1.0712\times10^{-6}$	$\mathbf{5.6843\times10^{-6}\pm4.7960\times10^{-8}}$	$1.6669\times10^{-4}\pm1.6766\times10^{-5}$	$1.7600\times10^{-5}\pm1.7600\times10^{-6}$
	WS-50	$4.7501\times10^{-5}\pm3.8091\times10^{-7}$	$\mathbf{8.5702\times10^{-6}\pm3.3663\times10^{-7}}$	$3.4491\times10^{-4}\pm4.7818\times10^{-6}$	$4.5900\times10^{-5}\pm4.5900\times10^{-6}$
DIFF	ER-50	$7.7768\times10^{-5}\pm5.4743\times10^{-6}$	$8.0824\times10^{-5}\pm8.6647\times10^{-6}$	$4.9044\times10^{-4}\pm2.0191\times10^{-4}$	$\mathbf{3.3400\times10^{-5}\pm8.7700\times10^{-9}}$
	BA-50	$1.2277\times10^{-4}\pm2.6557\times10^{-6}$	$\mathbf{6.3466\times10^{-5}\pm1.4402\times10^{-6}}$	$7.8219\times10^{-4}\pm4.5359\times10^{-4}$	$6.6300\times10^{-5}\pm4.6900\times10^{-5}$
	WS-50	$3.1724\times10^{-4}\pm1.9039\times10^{-5}$	$6.8072\times10^{-5}\pm4.6132\times10^{-6}$	$8.2823\times10^{-4}\pm6.5361\times10^{-5}$	$\mathbf{2.3300\times10^{-6}\pm2.3300\times10^{-7}}$
SPR	ER-50	$2.6906\times10^{-5}\pm3.0112\times10^{-6}$	$1.5338\times10^{-4}\pm1.1216\times10^{-5}$	$2.4972\times10^{-3}\pm8.1489\times10^{-5}$	$\mathbf{7.4200\times10^{-6}\pm7.4200\times10^{-7}}$
	BA-50	$1.5901\times10^{-5}\pm5.5379\times10^{-7}$	$1.6035\times10^{-4}\pm1.3170\times10^{-5}$	$1.5239\times10^{-3}\pm5.6256\times10^{-5}$	$\mathbf{7.9500\times10^{-8}\pm7.9500\times10^{-9}}$
	WS-50	$3.2927\times10^{-5}\pm1.8676\times10^{-6}$	$8.1005\times10^{-4}\pm1.8911\times10^{-4}$	$3.0290\times10^{-3}\pm6.9017\times10^{-5}$	$\mathbf{9.7300\times10^{-8}\pm9.7300\times10^{-9}}$
KURA	ER-50	$3.6740\times10^{-3}\pm1.0097\times10^{-4}$	$6.4481\times10^{-3}\pm3.4350\times10^{-5}$	$\mathbf{6.3492\times10^{-4}\pm6.3422\times10^{-5}}$	$9.7200\times10^{-3}\pm1.1300\times10^{-3}$
	BA-50	$3.7515\times10^{-3}\pm1.6245\times10^{-4}$	$8.2731\times10^{-3}\pm1.7709\times10^{-4}$	$\mathbf{8.4725\times10^{-4}\pm5.6965\times10^{-6}}$	$8.1900\times10^{-3}\pm5.6200\times10^{-4}$
	WS-50	$3.0970\times10^{-3}\pm2.8471\times10^{-3}$	$4.4286\times10^{-3}\pm2.5312\times10^{-5}$	$\mathbf{1.9890\times10^{-3}\pm3.7917\times10^{-5}}$	$7.1600\times10^{-3}\pm7.1600\times10^{-4}$
FJ	ER-50	$2.5622\times10^{-3}\pm9.1344\times10^{-4}$	$\mathbf{1.8907\times10^{-4}\pm1.2330\times10^{-5}}$	$4.2587\times10^{-3}\pm2.3276\times10^{-4}$	$4.2100\times10^{-4}\pm4.1400\times10^{-4}$
	BA-50	$5.2834\times10^{-3}\pm3.6400\times10^{-4}$	$\mathbf{4.4081\times10^{-4}\pm3.0002\times10^{-5}}$	$2.6749\times10^{-3}\pm1.1484\times10^{-3}$	$7.9000\times10^{-4}\pm7.5300\times10^{-4}$
	WS-50	$9.7899\times10^{-3}\pm3.1362\times10^{-4}$	$1.3288\times10^{-3}\pm1.6783\times10^{-5}$	$2.2123\times10^{-3}\pm9.7345\times10^{-4}$	$\mathbf{7.9000\times10^{-5}\pm7.9000\times10^{-6}}$
CMN	ER-50	$1.5585\times10^{-3}\pm1.5474\times10^{-6}$	$9.7297\times10^{-3}\pm1.6899\times10^{-4}$	$2.8771\times10^{-3}\pm7.3826\times10^{-4}$	$\mathbf{3.6400\times10^{-6}\pm3.6400\times10^{-7}}$
	BA-50	$1.9743\times10^{-3}\pm4.4331\times10^{-5}$	$1.0513\times10^{-2}\pm2.2569\times10^{-4}$	$3.6853\times10^{-3}\pm4.1975\times10^{-4}$	$\mathbf{9.7800\times10^{-5}\pm9.7800\times10^{-6}}$
	WS-50	$2.4850\times10^{-3}\pm4.5972\times10^{-5}$	$1.4522\times10^{-2}\pm1.7174\times10^{-5}$	$4.9462\times10^{-3}\pm1.9855\times10^{-4}$	$\mathbf{1.0700\times10^{-3}\pm1.0700\times10^{-4}}$

For Kuramoto, the recovered sinusoidal primitive combines with the fixed intrinsic frequency encoded in the node state to give the interaction and self-dynamics:

$$h_i^{\text{Kura}} = \sum_j A_{ij}^{\text{soft}} \sin(x_j - x_i), \qquad \dot{\phi}_i = \omega_i + h_i^{\text{Kura}}.$$

For MM and CMN, we increase the training trajectories with independent initial conditions from 50×10 to 200×10. COSINE then recovers the saturation term for MM and the linear and quadratic message terms for CMN, with the corresponding update terms and MSE reported in Table 5.

Table 6 reports prediction error for the learned dynamics. COSINE obtains the lowest MSE on several systems, especially Diff, Spring, and CMN, showing that the discovered symbolic mechanisms can also predict accurately. At the same time, MSE does not always favor COSINE: in regimes such as MM, FJ, and Kuramoto, high-capacity neural baselines such as NRI and GDP can achieve lower prediction error. This is expected because COSINE uses a sparse symbolic model with substantially

Table 7: MSE before and after thresholding the inferred graph and symbolic coefficients at inference.

Dyn.	Differentiable model	Thresholded structure
MM	4.44×10^{-6}	2.7919×10^{-6}
Kuramoto	9.72×10^{-3}	1.0629×10^{-3}

Table 8: Performance (AUC %) of COSINE with different LLMs. Results are mean $\pm$ standard deviation over 10 runs. Models are ordered by increasing parameter scale.

MODEL	PARAMS	SPR	KURA	MM	CMN
DEEPSEEK-R1	8B	100.00±0.00	99.97±0.00	54.65±2.97	54.33±2.65
QWEN3	8B	100.00±0.00	99.90±0.03	58.21±0.59	63.42±1.40
DEEPSEEK-R1	14B	100.00±0.00	99.93±0.01	88.07±4.02	81.05±18.95
GPT-OSS	20B	100.00±0.00	99.97±0.02	97.79±1.73	96.20±3.42
DEEPSEEK-R1	32B	100.00±0.00	99.97±0.01	92.62±1.16	91.84±8.12
QWEN3	32B	99.66±0.34	83.49±1.98	97.11±1.67	95.70±4.27
QWEN2.5	72B	99.15±0.01	99.94±0.02	98.12±1.66	99.98±0.02
GPT-OSS	120B	100.00±0.00	99.84±0.02	92.29±4.06	93.55±6.01

Table 9: End-to-end runtime in seconds with GPT-OSS (20B), reported as mean $\pm$ standard deviation over 10 runs.

Dyn.	NRI	GDP	RIVA	COSINE (local)	COSINE (API)
MM	163.13±2.15	153.81±1.84	21.66±0.38	150.20±1.62	177.76±15.42
Diff	179.77±2.41	153.33±1.95	21.43±0.31	51.89±0.85	114.55±11.68
Spring	104.26±1.58	80.57±1.12	19.95±0.26	58.50±0.94	54.89±9.85

fewer functional degrees of freedom while also solving the unknown graph recovery problem. The performance gap is generally modest, and we view it as a reasonable trade-off between mechanistic interpretability and predictive accuracy. The MSE results therefore complement the term-recovery results: they measure predictive fit, whereas the discovered terms measure mechanistic interpretability.

We also evaluate the inferred graph and W after thresholding both at inference. Table 7 distinguishes this selected symbolic structure from the continuous model used in training.

B.4 Impact of LLM Scale

Analysis of LLM scale. Table 8 suggests that LLM scale is not the main bottleneck for simpler symbolic mechanisms. Spring and Kuramoto remain near saturation for most models, indicating that once the required primitives are proposed, the inner-loop optimization can recover the graph reliably. In contrast, MM and CMN are more sensitive to the quality of symbolic proposals: small 8B models struggle, while medium-to-large models generally recover much stronger structures. The trend is not strictly monotonic with parameter count, so these results point to both model capacity and proposal quality, rather than scale alone, as the relevant factors.

B.5 End-to-end computational cost

We measure complete runtime for the GPT-OSS (20B) backend, using local deployment through Ollama on an NVIDIA RTX 4090 and API access through OpenRouter. COSINE is timed from initialization to final-library acceptance, including inner-loop training, validation, and LLM inference. NRI, GDP, and RIVA are timed over their complete training procedures.

Including the outer loop, local COSINE is faster than NRI and GDP on all three dynamics in Table 9. With API access, it remains faster on Diffusion and Spring, while MM takes longer. RIVA has the shortest runtime on all three. Spring and Diffusion usually converge within one or two outer rounds,

while MM requires more rounds of library evolution. API runtime also includes transfer, queueing, and inference-response latency. These complete-run measurements complement the epoch-level time and memory comparison in Figure 5.

B.6 Dynamical Systems

Details of the six representative classes of networked dynamical systems used in experiments:

B.6.1 Michaelis–Menten kinetics (MM)

Michaelis–Menten (MM) kinetics [24] is a fundamental continuous-time nonlinear model widely used to describe gene-regulatory interactions and biochemical reaction networks. In this model, the state evolution of each node is governed by a combination of basal production, nonlinear cooperative interactions, and first-order degradation. The dynamic equation is given by:

$$\dot{x}_i = -x_i + \sum_{j \in \mathcal{N}(i)} \frac{A_{ij}}{k_i} \frac{x_j}{1 + x_j} \tag{20}$$

where x_i represents the activity level of node i, and $k_i = \sum_j A_{ij}$ is the in-degree of node i. This formulation assumes unit degradation rate and saturation constants, with normalized interaction strengths across neighbors. This model effectively captures the saturation effects inherent in biological signaling. Figure 6 illustrates the representative trajectories and the underlying ground-truth adjacency matrix for the Michaelis–Menten system.

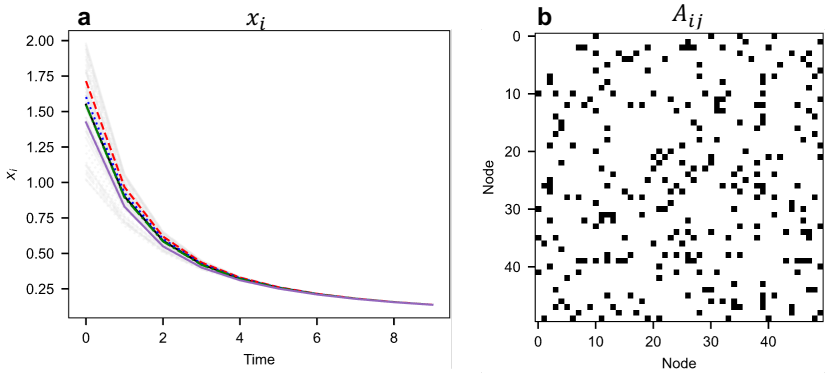

Figure 6: Visual illustration of Michaelis–Menten kinetics. **a**, Node activities (with the trajectories of the first 5 nodes highlighted by colored lines). **b**, Adjacency matrix of the network.

B.6.2 Diffusion (Diff)

The diffusion process represents a classic continuous-time linear mechanism for conservative flow or heat distribution over a network. It assumes that the rate of change of a node's state is proportional to the difference between its state and those of its neighbors. The global system dynamics are expressed as:

$$\dot{x} = -\beta L x, \quad L = D_{\text{in}} - A \tag{21}$$

where $\beta > 0$ denotes the global diffusion rate (or thermal conductivity). In our experiments, we set $\beta = 1.0$. L is the directed graph Laplacian defined as $L = \mathbf{D}_{\text{in}} - \mathbf{A}$, where $\mathbf{D}_{\text{in}}$ is the in-degree diagonal matrix and $\mathbf{A}$ is the adjacency matrix. This model is a cornerstone for studying spreading processes and consensus algorithms. Figure 7 visualizes the diffusing states alongside the network.

B.6.3 Network of Springs (Spring)

The network of springs (Spring or Spr) is a continuous-time second-order dynamical system derived from classical mechanics. It treats each node as a unit mass coupled to its neighbors by linear springs

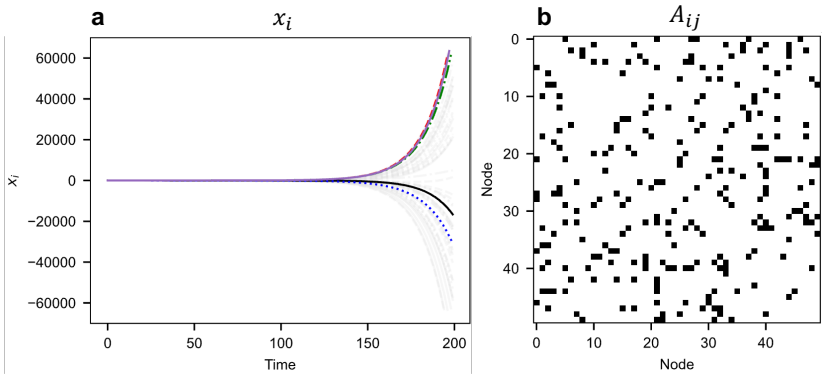

Figure 7: Visual illustration of Diffusion dynamics. **a**, Node activities (with the trajectories of the first 5 nodes highlighted by colored lines). **b**, Adjacency matrix of the network.

satisfying Hookes law, often including a global damping effect. The dynamics follow:

$$\ddot{\mathbf{p}}_i = -\gamma \dot{\mathbf{p}}_i - k \sum_{j \in \mathcal{N}(i)} A_{ij}(\mathbf{p}_i - \mathbf{p}_j) \tag{22}$$

where $\mathbf{p}_i$ denotes the position of node i, k is the spring constant, and γ is a global damping coefficient. Figure 8 illustrates the representative trajectories and the underlying ground-truth adjacency matrix for the Spring system.

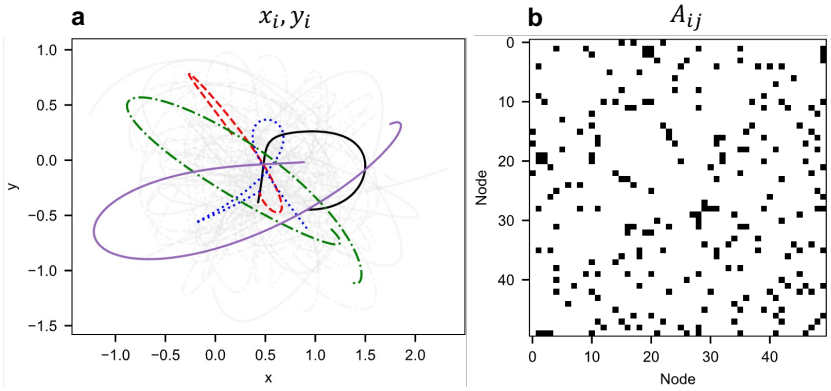

Figure 8: Visual illustration of Spring dynamics. **a**, Node activities (with the trajectories of the first 5 nodes highlighted by colored lines). **b**, Adjacency matrix of the network.

B.6.4 Kuramoto Model (Kuramoto)

The Kuramoto model [26] is a canonical framework for studying synchronization phenomena in populations of coupled oscillators. It describes the evolution of the phases of N oscillators that interact through a nonlinear sine-coupling term. The governing equation is:

$$\dot{\phi}_i = \omega_i + \sum_{j \in \mathcal{N}(i)} A_{ij} \kappa \sin(\phi_j - \phi_i) \tag{23}$$

where ϕ_i is the phase of oscillator i, ω_i is its natural frequency, and κ denotes the coupling strength. Figure 9 visualizes the phase evolution, network, and natural frequencies.

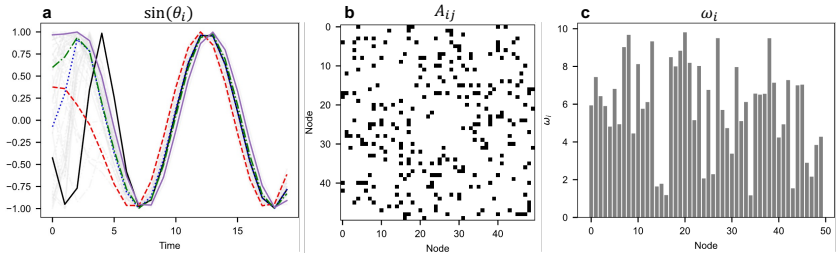

Figure 9: Visual illustration of Kuramoto model. **a**, Node activities (with the trajectories of the first 5 nodes highlighted by colored lines). **b**, Adjacency matrix of the network. **c**, Natural frequencies ω_i of each node.

B.6.5 Friedkin–Johnsen Dynamics (FJ)

Friedkin–Johnsen (FJ) dynamics [12] is a classic discrete-time opinion formation model on graphs, balancing social influence and innate opinions (stubbornness). The dynamics are defined as:

$$x_i^{t+1} = \lambda_i \frac{1}{k_i} \sum_{j \in \mathcal{N}(i)} A_{ij} x_j^t + (1 - \lambda_i) s_i \tag{24}$$

where x_i^t is the opinion of node i at time t, $k_i = \sum_j A_{ij}$ is the in-degree, and s_i represents the innate opinion. In our setup, $x_i(0), s_i \sim \mathcal{U}(-1, 1)$ and we adopt $\lambda_i = k_i/(k_i + 1)$ with degree-normalized influence. Figure 10 illustrates the opinion evolution and the stubbornness biases present in the network.

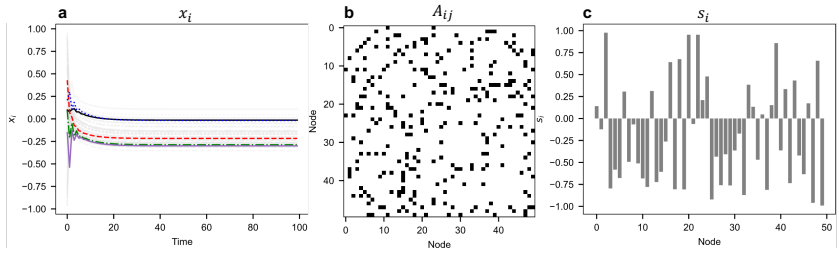

Figure 10: Visual illustration of Friedkin–Johnsen dynamics. **a**, Node activities (with the trajectories of the first 5 nodes highlighted by colored lines). **b**, Adjacency matrix of the network. **c**, Innate opinions s_i of each node.

B.6.6 Coupled Map Network (CMN)

Coupled map networks (CMN) [14] are discrete-time dynamical systems used to model spatiotemporal chaos and self-organization in complex systems. It couples local chaotic maps through a network structure. The dynamics are defined as:

$$\theta_i^{t+1} = (1 - s) f(\theta_i^t) + \frac{s}{k_i} \sum_{j \in \mathcal{N}(i)} A_{ij} f(\theta_j^t) \tag{25}$$

where θ_i^t is the state of node i at time t and $s \in [0, 1]$ is the coupling parameter. We use the logistic map $f(x) = \lambda x(1 - x)$ as the local nonlinear map with parameters $\lambda = 3.5$ and $s = 0.2$. The system balances local mapping with degree-normalized neighborhood interactions. CMNs are powerful tools for investigating the balance between local chaotic dynamics and global synchronization. Figure 11 shows the dynamic patterns and the interaction matrix for the coupled map network.

B.7 Baselines

Granger Causality (GC) Granger Causality [17] evaluates the directed functional connectivity between nodes by measuring the predictive gain. Specifically, node j is said to "Granger-cause"

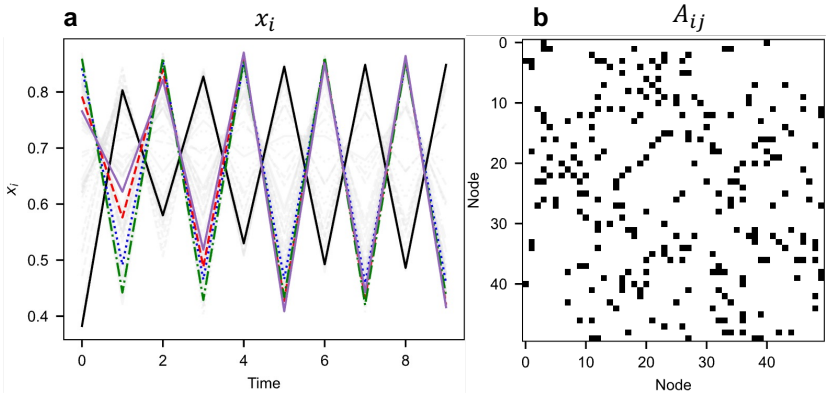

Figure 11: Visual illustration of Coupled map network. **a**, Node activities (with the trajectories of the first 5 nodes highlighted by colored lines). **b**, Adjacency matrix of the network.

node i if the future of x_i can be better predicted using the past of both x_i and x_j than using the past of x_i alone. The GC score is typically computed as the log-ratio of the residual variances: $\mathcal{G}_{j\to i} = \ln(\sigma^2_{\text{restricted}}/\sigma^2_{\text{unrestricted}})$. In our experiments, we implement GC using the `netrd` library, which fits a linear vector autoregressive (VAR) model to the time series data.

Mutual Information (MI) Mutual Information [7] is a non-parametric measure of statistical dependence between two random variables. For two node trajectories X_i and X_j, it quantifies the amount of information obtained about X_i through observing X_j, defined as: $I(X_i; X_j) = \sum_{x_i,x_j} p(x_i, x_j) \ln \frac{p(x_i,x_j)}{p(x_i)p(x_j)}$. We utilize the `netrd` implementation, which estimates the probability densities to construct a symmetric adjacency matrix representing pairwise dependencies.

Transfer Entropy (TE) Transfer Entropy [40] is an information-theoretic measure of directed interaction that quantifies the reduction in uncertainty of the future of X_i given the past of X_j, over and above the reduction provided by the past of X_i itself. The formula is given by: $TE_{j\to i} = \sum p(x_{i,t+1}, x_{i,t}, x_{j,t}) \ln \frac{p(x_{i,t+1}|x_{i,t},x_{j,t})}{p(x_{i,t+1}|x_{i,t})}$. This method is implemented via `netrd` to capture directed, potentially nonlinear information flows.

Neural Relational Inference (NRI) NRI [25] is a variational autoencoder-based framework that jointly infers latent interactions and system dynamics. We employ a GNN-based encoder to infer the edge distribution and an MLP-based decoder for state prediction. The model is trained for 500 epochs with a batch size of 64. We set the number of hidden channels to 512. The optimization uses a dual-learning rate strategy: $lr = 0.0005$ for the decoder and $lr_a = 0.1$ for the structural inference. The learning rate is decayed by a factor of 0.5 every 200 epochs to ensure convergence.

Graph Dynamics Prior (GDP) GDP [35] introduces a generative prior to model the graph distribution, which is optimized alongside the dynamics via variational inference. We configure GDP with 512 hidden channels and a batch size of 64. For spectral modeling, we use a `power` polynomial filter of order $K = 4$. Similar to NRI, we use $lr = 0.0005$ for the dynamics parameters and a higher learning rate $lr_a = 0.1$ for the distribution estimation to accelerate structural recovery.

Relational Inference via Variational Attention (RIVA) RIVA [48] is a recent Transformer-based model that interprets attention weights as latent interaction strengths within a variational framework. For the unweighted task, we retain the public implementation's attention-based interaction scores and evaluate them against binary adjacency using AUC. The hidden dimension is set to 512 with a batch size of 64. The dynamics learning rate is set to $lr = 0.00005$, while the structural learning rate is $lr_a = 0.1$. This configuration allows the model to capture fine-grained asymmetric dependencies between node trajectories.

Table 10: LaSR expressions and prediction errors with the true graph supplied. Spring-10 has 10 nodes and Kuramoto-50 has 50 nodes.

System	Node	Target	MSE	Recovered expression
Spring-10	6	`loc_x`	7.94×10^{-8}	$x_6^{t+1} \approx x_6^t + 0.10\, v_{x,6}^t$
Spring-10	6	`loc_y`	4.87×10^{-8}	$y_6^{t+1} \approx y_6^t + 0.10\, v_{y,6}^t$
Spring-10	7	`loc_x`	3.92×10^{-8}	$x_7^{t+1} \approx x_7^t + 0.10\, v_{x,7}^t$
Spring-10	7	`loc_y`	5.25×10^{-8}	$y_7^{t+1} \approx y_7^t + 0.10\, v_{y,7}^t$
Kuramoto-50	12	ϕ	5.15×10^{-2}	$x_{12}^{t+1} \approx x_{12}^t + 0.54$
Kuramoto-50	19	ϕ	5.73×10^{-2}	$x_{19}^{t+1} \approx x_{19}^t + 0.56$
Kuramoto-50	24	ϕ	4.31×10^{-2}	$x_{24}^{t+1} \approx x_{24}^t + 0.56$
Kuramoto-50	33	ϕ	4.92×10^{-2}	$x_{33}^{t+1} \approx x_{33}^t + 0.54$

Table 11: ND2 expressions and prediction errors on ER-50 with graph information supplied.

Dyn.	Target	MSE	R^2	Recovered expression
Spring	`dx`	5.36×10^{-6}	1.000	$v_x + 0.01\sum_{j\in\mathcal{N}(i)}(x_j - x_i)$
Kuramoto	`dphi`	2.48×10^{-1}	0.073	$4.80\,\sigma\big(0.03\,\phi^3\omega\big)$

Table 12: Module ablation on ER graphs ($N = 50$). Performance is relational inference AUC (%).

VARIANT	MM	DIFF	SPR	KURA	FJ	CMN
COSINE	99.68±0.32	99.32±0.05	99.49±0.26	99.98±0.01	99.60±0.40	100.00
W/O LLM EVOLUTION	96.44±0.96	99.80±0.14	65.36±0.99	59.02±0.84	99.53±0.42	82.23±1.34
W/O SPARSE	94.33±2.91	99.11±0.87	99.62±0.38	97.99±1.97	99.93±0.03	63.71±6.37
W/O MSG/UPD	53.14	52.45	53.62	55.11	55.46	53.25±0.64

B.8 Symbolic recovery with graph information

We evaluate LaSR [18] and ND2 [50] with graph information to examine their recovered interaction terms. LaSR receives the true graph as an oracle input.

Table 10 reports LaSR's recovered expressions for Spring-10 and Kuramoto-50. The Spring expressions recover local position updates. The Kuramoto expressions consist of the current phase plus a constant and omit the sinusoidal coupling. It recovers local updates on Spring-10 but produces phase-plus-constant expressions on Kura-50 and omits coupling terms. This illustrates the added difficulty of larger neighborhoods and permutation-invariant dynamics. For ND2 on ER-50, Table 11 shows a neighbor-difference update for Spring and a Kuramoto expression without the required $\sin(\phi_j - \phi_i)$ interaction.

B.9 Module Ablation Study

We further isolate the contribution of each module under a unified setting: relation inference on ER graphs, evaluated by AUC (%). Since graph learning is required for relation inference, we ablate the remaining components. *w/o LLM Evolution* uses the LLM-generated library only at initialization. *w/o Sparse* sets the sparse regularization weight to zero. *w/o Msg/Upd* removes the message/update decomposition.

Table 12 shows that the message/update decomposition is the most critical inductive bias: removing it drops AUC to near-chance levels across all systems. LLM-guided evolution is especially important when the initial symbolic library misses key mechanisms, as seen on Spring, Kuramoto, and CMN. Sparse regularization has a more selective role. It improves stability on harder systems such as MM and CMN, while on Diff and Spring the AUC can remain close to the full model. Overall, the gains come from complementary components rather than a single isolated module.

Table 13: Library search controls on ER-50. Performance is relational inference AUC (%), reported as mean ± standard deviation. The λ columns denote threshold-pruning variants.

Dyn.	$\lambda = 0$	$\lambda = 0.01$	Random	Oracle	COSINE
Spring	22.51±1.34	22.51±1.34	100.00±0.00	100.00±0.00	99.87±0.18
FJ	96.11±3.62	99.84±0.01	83.51±1.25	99.99±0.02	98.37±1.63
CMN	48.41±2.67	58.8±0.58	58.52±2.18	99.38±0.14	100±0.1
MM	85.47±0.49	85.47±0.49	77.42±1.65	99.57±0.08	98.32±1.68

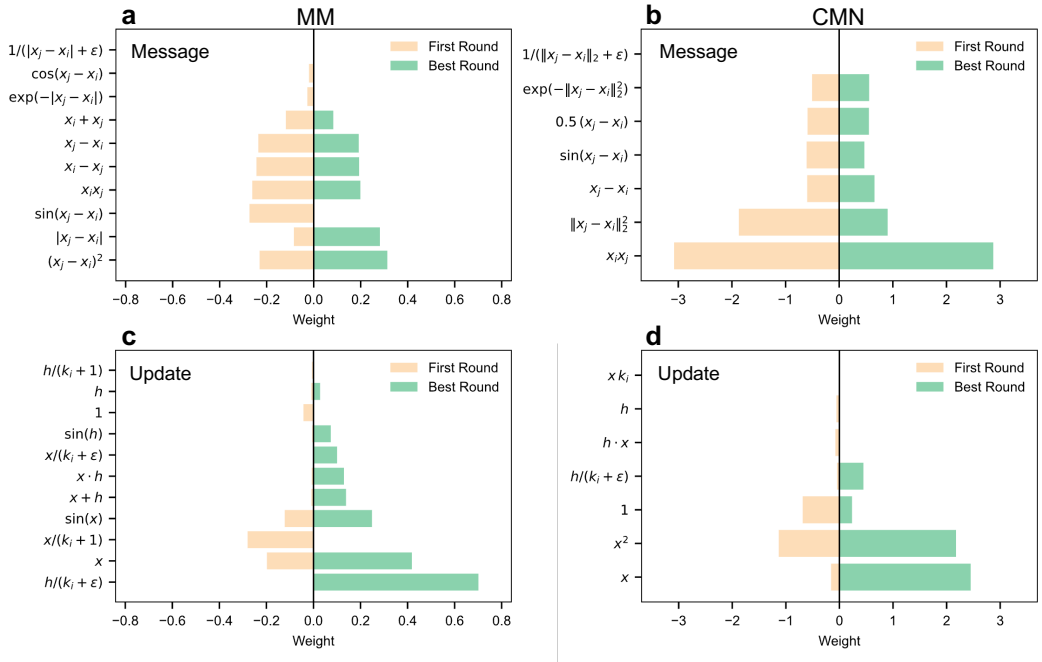

Figure 12: Evolution of discovered basis terms for MM and CMN systems. Panels (a, b) show Message term weights, while (c, d) show Update term weights for the first (blue) and best (red) rounds.

B.10 Library search controls

We compare LLM-guided evolution with random library evolution, threshold pruning with $\lambda = 0$ and $\lambda = 0.01$ [13], and an oracle library on ER-50 graphs. Table 13 shows that COSINE remains close to the oracle across the four dynamics. Random evolution reaches 100% on Spring, while its AUC ranges from 58.52% to 83.51% on the other three systems. This comparison supports more reliable library search with LLM-guided evolution in these experiments.

B.11 Evolution of Basis Library

Figure 12 illustrates the evolution of weight distributions within the message and update modules for the *Michaelis–Menten* (MM) and *CMN* systems, comparing the first round of training with the best-performing round. In the first round, weights are dispersed across multiple candidate terms, including redundant nonlinearities that are weakly correlated with the true dynamics.

As the LLM-guided update process converges, the weight distributions focus sharply on core dynamical primitives in the best round. Specifically, in the *MM* system, weights focus on terms consistent with degree-normalized reaction inputs (e.g., $h/(k_i + \epsilon)$ in panel c), while in the *CMN* system, the identified terms accurately capture the coupling (e.g., $x_i x_j$ in panel b) and local mapping mechanisms (e.g., x and x^2 in panel d). While minor redundant terms may persist, their coefficients are significantly suppressed compared to the dominant, mechanism-aligned terms. This directional

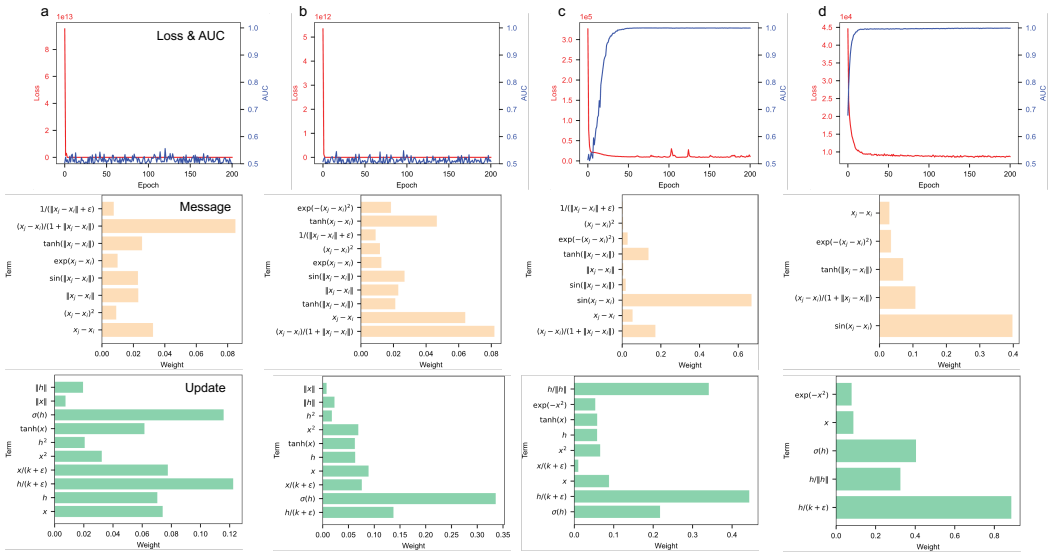

Figure 13: Closed-loop evolution of the basis library in the Kuramoto system in the first 4 rounds (a-d). The first row displays the training loss (red) and structural inference AUC (blue) over four optimization rounds. The second and third rows illustrate the distribution of learned coefficients (importance) for candidate terms in the message and update libraries, respectively. Note the sharp rise in AUC following the introduction of the $\sin(x_j - x_i)$ term and the subsequent pruning of redundant basis functions.

convergence demonstrates that the LLM effectively leverages training feedback to bridge the gap between initial symbolic hypotheses and parsimonious physical representations, which is crucial for modeling complex nonlinear networked dynamics.

B.12 Mechanistic Discovery via Iterative Library Refinement: A Case Study

To further illustrate the dynamics of the LLM-guided basis library evolution, we track the performance and symbolic composition over four consecutive optimization rounds in the Kuramoto system. As shown in Figure 13, the process demonstrates a clear transition from exploratory hypothesis testing to precise mechanism discovery.

In the initial rounds (Rounds 1 and 2), the message library consists of various candidate terms such as linear differences, tanh, and exponential forms, but lacks the true sinusoidal coupling. During this phase, the structural inference performance (AUC) remains stagnant and suppressed, and the training loss fails to converge to a low steady state. This performance bottleneck confirms that without the correct symbolic primitives, simply increasing the complexity of non-linear fits or structural parameters is insufficient to recover the underlying network.

A significant breakthrough occurs in subsequent rounds when the LLM-guided outer loop introduces the $\sin(x_j - x_i)$ term into the message library. As observed in the training plots, the inclusion of this physically consistent primitive triggers a sharp increase in AUC, rapidly approaching near-perfect recovery (AUC ≈ 1.0), accompanied by a drastic reduction in prediction loss. In the update module, the model simultaneously identifies that the dynamics are primarily governed by the aggregated messages and degree-normalized terms, effectively capturing the intrinsic natural frequency and coupling effects.

In the final round, the framework performs a pruning operation based on the importance statistics (learned weights) from the previous round. Redundant terms with negligible coefficients are removed, resulting in a highly parsimonious and interpretable library. This evolution highlights the effectiveness of COSINE's closed-loop paradigm: the inner loop provides a quantifiable signal of "how well a symbolic hypothesis works," while the outer loop utilizes this feedback to iteratively refine the expression space toward mechanism-aligned governing terms.

Table 14: Relational inference performance (AUC %) under low-data regime (10×10 samples) across different dynamics and networks. Results are reported as mean $\pm$ standard deviation over multiple runs.

DYN.	GRAPH	GC	MI	TE	GDP	NRI	RIVA	COSINE
MM	BA50	51.63	72.08	55.35	50.98±1.34	63.83±2.12	52.37±0.96	**72.89±2.31**
	ER50	54.65	64.90	53.84	52.85±1.45	59.84±2.35	52.31±1.04	**73.21±2.18**
	WS50	59.48	**91.23**	55.98	56.97±2.22	88.30±3.10	54.19±1.20	87.49±2.65
DIFF	BA50	64.22	68.07	56.56	54.59±1.52	59.84±3.28	52.19±1.08	**91.98±1.46**
	ER50	58.65	56.18	53.13	63.52±2.74	81.57±4.36	52.69±1.15	**81.72±1.82**
	WS50	63.66	55.68	67.76	56.16±1.68	91.17±2.57	54.97±1.31	**98.21±0.74**
SPR	BA50	54.23	58.86	59.26	98.62±0.86	92.13±3.95	50.62±0.88	**99.28±0.42**
	ER50	50.15	50.13	52.51	98.67±0.72	78.00±4.85	51.18±0.95	**99.39±0.31**
	WS50	53.81	51.40	51.50	95.18±1.92	81.53±5.14	56.64±1.44	**95.78±1.08**
KURA	BA50	60.03	92.25	68.86	58.02±3.46	67.39±5.12	55.45±1.62	**96.06±1.22**
	ER50	56.67	86.72	68.16	53.73±2.88	84.67±4.24	52.76±1.34	**91.10±2.06**
	WS50	56.30	99.59	91.09	89.82±4.73	52.48±1.72	62.63±2.20	**99.98±0.02**
FJ	BA50	83.80	57.86	79.26	51.90±1.18	53.89±1.66	52.86±1.03	**94.74±1.21**
	ER50	87.34	54.87	75.65	53.19±1.44	51.00±1.28	50.55±0.91	**96.26±0.92**
	WS50	99.00	59.73	91.02	70.76±3.16	66.27±2.85	52.37±1.17	**100.00±0.00**
CMN	BA50	51.47	**86.56**	71.73	78.60±3.95	76.99±4.28	53.83±1.25	60.37±2.76
	ER50	54.43	**77.47**	67.91	53.99±2.14	68.56±3.64	51.04±1.08	70.09±2.42
	WS50	58.63	**96.12**	81.11	81.02±3.72	62.45±3.88	56.43±1.53	85.35±2.18

B.13 Low-Data Performance

Table 14 summarizes relational inference under an extreme low-data regime, where only 10×10 samples are available for each dynamical system and network. This setting is particularly challenging for approaches that rely on stable estimates of statistical dependence or long-horizon temporal correlations.

Overall, most baselines degrade substantially. Classical statistical and information-theoretic methods (GC, MI, TE) often approach near-chance performance on heterogeneous graphs such as BA50 and ER50, consistent with the difficulty of estimating dependencies from severely limited observations. Model-based and regression-driven approaches (GDP, NRI) are also sensitive to the specific pairing of dynamics and network, performing well in selected cases but lacking uniform robustness.

In contrast, COSINE maintains higher and more stable AUC across most tested systems and networks. The gap is most pronounced for nonlinear and oscillatory dynamics (Diff and Kuramoto), where COSINE achieves near-saturated accuracy even when baselines fluctuate. For mean-field-like dynamics (CMN and MM), where relational signals are intrinsically weaker, COSINE still delivers clear improvements, indicating robustness rather than isolated best-case wins.

These results suggest that COSINE depends less on large sample sizes or long-term statistical convergence. Unlike neural surrogates that require abundant data to constrain model capacity and thus risk overfitting noise in low-data regimes, COSINE imposes strong structural and dynamical inductive biases through explicit symbolic representations, enabling reliable parameter identification from limited observations and improving data efficiency.

From a bias–variance perspective, COSINE deliberately trades expressive flexibility for stronger inductive constraints, which suppress variance when observations are scarce. This bias toward parsimonious, physically meaningful interaction patterns reduces spurious fits and yields more reliable relational inference under severe data scarcity. Although low-data settings are not the primary focus of this work, the consistent gains here further underscore COSINE's robustness and generalization when observations are limited or noisy.

Table 15: Discovered terms from round 0 for COVID datasets. Terms are ordered by absolute coefficient magnitude. Each region has Message and Update rows. x: node state; h_i: aggregated message; k_i: degree; $\epsilon = 10^{-6}$.

Region	Component	Term 1	Term 2	Term 3	Term 4	Term 5	Term 6
Arizona	Message	$x_i \cdot x_j$ (0.90)	$x_j^{(1)} \cdot (x_j^{(0)} - x_i^{(0)})$ (0.56)	$\|x_j - x_i\|_2$ (0.10)	$\sigma(x_j - x_i)$ (0.05)	$x_j - x_i$ (0.04)	-
	Update	$\tanh(x)$ (0.58)	$x \cdot h_i$ (0.13)	$\frac{h_i}{k_i+\epsilon}$ (0.06)	$x^{(2)} \cdot h_i^{(0)}$ (0.05)	$h_i^{(1)}$ (0.03)	-
Connecticut	Message	$(x_j - x_i)^2$ (0.54)	$(x_j - x_i)$ (0.10)	$x_j^{(0)}$ (0.07)	$\sigma(\|x_j - x_i\|_2)$ (0.05)	$x_j^{(0)} \cdot \sigma((x_j^{(0)} - x_i^{(0)}))$ (0.04)	-
	Update	$x^{(1)}$ (0.23)	$x^{(0)}$ (0.13)	$x^{(2)}$ (0.13)	$x^{(1)} \cdot x^{(2)}$ (0.10)	$\tanh(h_i)$ (0.09)	$\frac{h_i}{k_i+\epsilon}$ (0.07)
Illinois	Message	$x_i \cdot x_j$ (0.52)	$\frac{(x_i \cdot (x_j - x_i))}{(1+\|x_j - x_i\|_2)}$ (0.28)	$\|x_j - x_i\|_2$ (0.07)	$x_j^{(2)} - x_i^{(2)}$ (0.07)	$x_j^{(1)} - x_i^{(1)}$ (0.07)	$2 \cdot (x_j - x_i)$ (0.07)
	Update	$-0.1x$ (0.43)	$-0.2x^{(1)}$ (0.38)	$\frac{h_i}{k_i+\epsilon}$ (0.10)	$\tanh(h_i)$ (0.09)	$x^{(0)} + 0.5 \cdot x^{(1)}$ (0.08)	h_i (0.04)
Michigan	Message	$x_i \cdot (x_j - x_i)$ (0.27)	$x_j^{(1)} \cdot (x_j^{(0)} - x_i^{(0)})$ (0.08)	$(x_j - x_i)^2$ (0.06)	$x_j - x_i$ (0.03)	$\tanh(x_j - x_i)$ (0.01)	-
	Update	$-x$ (0.47)	$\frac{h_i}{k_i+\epsilon}$ (0.11)	$x \cdot (1 - x)$ (0.10)	$h_i^{(1)}$ (0.07)	h_i (0.05)	$h_i^{(0)} + h_i^{(1)} + h_i^{(2)}$ (0.03)

Table 16: PCC against the mobility graph on the four COVID-19 datasets. Neural baselines are reported as mean $\pm$ standard deviation; COSINE values are those shown in Figure 3.

State	NRI	GDP	RIVA	COSINE
Arizona	0.1991 ± 0.1324	-0.0350 ± 0.2147	0.0051 ± 0.0630	0.8116
Illinois	0.0427 ± 0.1942	0.0026 ± 0.0131	0.0207 ± 0.0380	0.7100
Connecticut	0.0633 ± 0.1194	0.2029 ± 0.2072	0.0452 ± 0.1533	0.6227
Michigan	0.0408 ± 0.2333	0.0013 ± 0.0176	0.0243 ± 0.0797	0.7435

C Details of empirical experiment

C.1 Discovered Terms

To elucidate the impact of the LLM-guided hypothesis refinement, we present the initial basis configuration for the COVID-19 dataset in Table 15. By contrasting the first-round candidates with the final converged library, we observe how the LLM-guided agent iteratively prunes redundant terms and augments missing mechanisms. This evolutionary process effectively streamlines the initially cluttered symbolic space, converging toward an optimal and parsimonious basis library that accurately represents the underlying dynamical landscape of the epidemic.

C.2 Agreement with the mobility graph

We evaluate NRI, GDP, and RIVA on the same four-state COVID-19 datasets by computing PCC against the mobility graph used in Figure 3. Table 16 includes the COSINE values from that figure for comparison. COSINE has higher PCC than the neural-baseline means in each state. Here PCC measures agreement with an external mobility graph for regional interactions.

C.3 Dataset

In this section, we provide detailed visualizations and descriptions of the four real-world COVID-19 datasets used in our experiments. Each dataset corresponds to a specific U.S. state, comprising county-level daily confirmed cases and geographic connectivity.

C.3.1 Arizona

The Arizona dataset consists of 15 counties. As shown in Figure 14a, the geographic network is characterized by large spatial distances between nodes, with a few central hubs (notably Maricopa and Pima counties) exhibiting high connectivity. Figure 14b illustrates the temporal distribution of confirmed cases. The state experienced two major synchronized waves: the Winter 2020-2021 wave and the prominent Omicron spike in early 2022. Despite the vast geographic areas, the ridgeline plot reveals a high degree of temporal correlation across counties, suggesting that the primary population centers drive the overall dynamical trend.

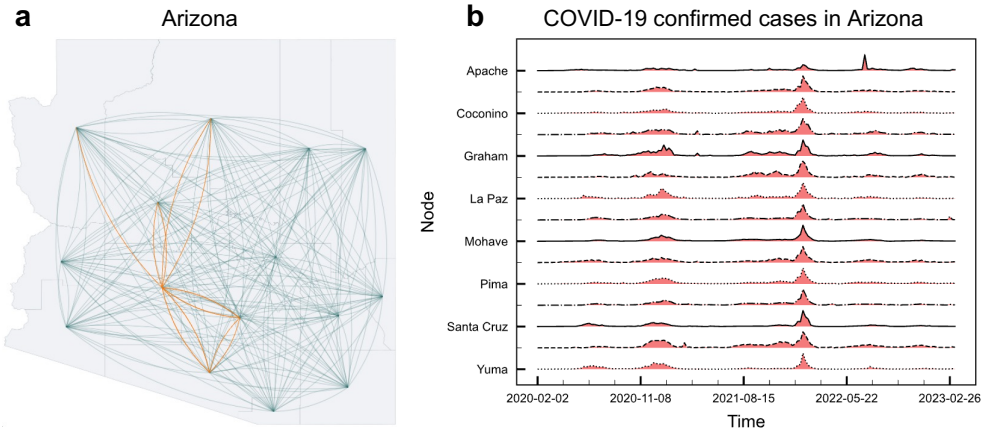

Figure 14: **Epidemiological data for Arizona.** (a) Geographic map and county-level interaction network. (b) Ridgeline plot of daily COVID-19 confirmed cases across counties from February 2020 to February 2023.

C.3.2 Connecticut

Connecticut represents a compact, high-density system with only 8 counties. The network in Figure 15a is relatively dense given its small size, with strong inter-county ties reflecting frequent cross-region commuting. The case distribution in Figure 15b is notable for its extreme synchronization; while earlier waves were relatively subdued, the early 2022 Omicron peak is exceptionally sharp and uniform across all counties (e.g., Fairfield, Hartford, New Haven). This dataset serves as a benchmark for testing relational inference in small-scale, highly coupled systems.

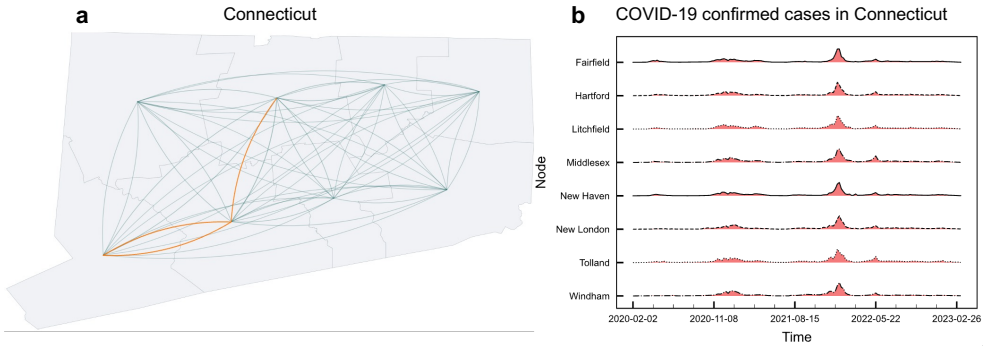

Figure 15: **Epidemiological data for Connecticut.** (a) Geographic map and county-level interaction network. (b) Ridgeline plot of daily COVID-19 confirmed cases across its 8 counties.

C.3.3 Illinois

Illinois provides a large-scale dataset with 102 counties. The interaction network (Figure 16a) displays a distinct core-periphery structure, with massive connectivity concentrated in the northeast corner around the Chicago metropolitan area. The temporal data in Figure 16b is highly complex, showing multiple distinct regional surges. The large number of nodes (Adams, Clinton, Ford, etc.) creates a dense ridgeline pattern, where the propagation of waves from urban centers to rural peripheries provides a challenging testbed for capturing long-range spatial dependencies.

C.3.4 Michigan

The Michigan dataset includes 83 counties, with its geographic structure constrained by the state's unique peninsula topography. As seen in Figure 17a, the network density is highest in the southeast

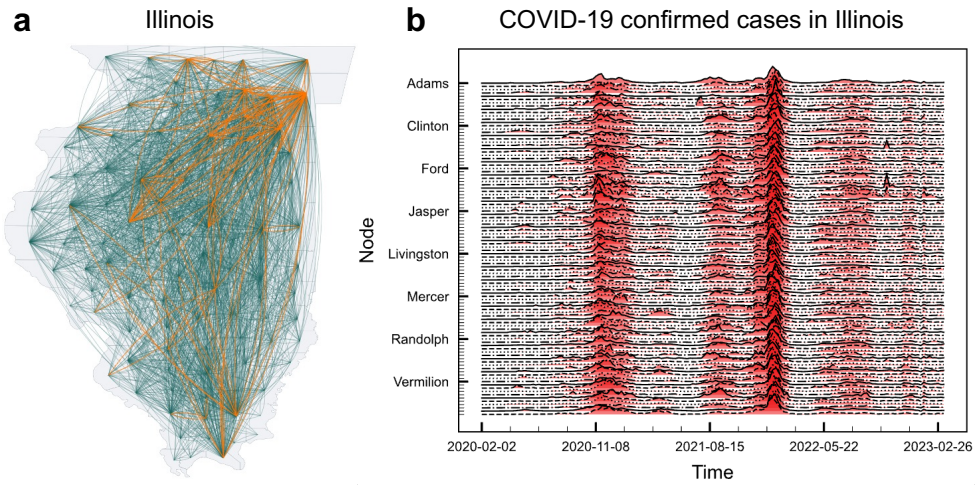

Figure 16: **Epidemiological data for Illinois.** (a) Geographic map and county-level interaction network. (b) Ridgeline plot showing the high-density case distribution across over 100 counties.

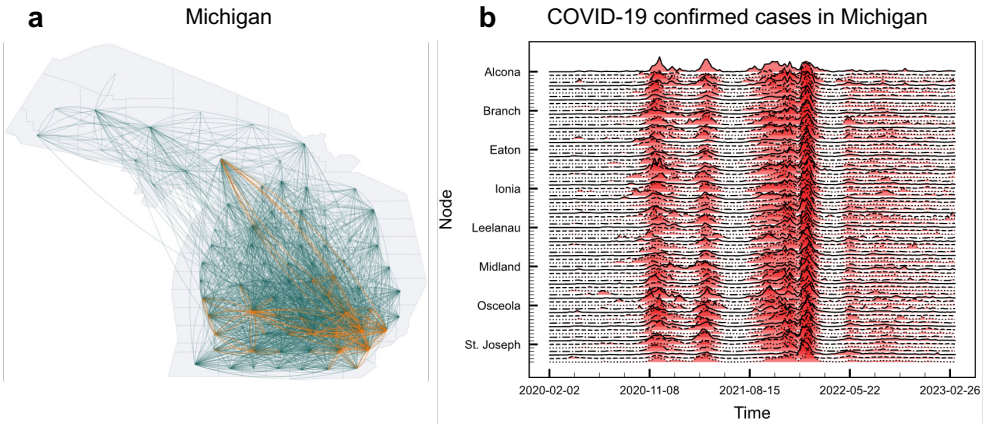

Figure 17: **Epidemiological data for Michigan.** (a) Geographic map and county-level interaction network. (b) Ridgeline plot of daily COVID-19 confirmed cases illustrating regional seasonal variations.

(the Detroit area), with orange-highlighted edges indicating strong local interaction clusters. Figure 17b shows that Michigan faced unique seasonal patterns, including a significant surge in Spring 2021 that preceded the nationwide Omicron wave. The variation in peak timing and magnitude among counties (e.g., Alcona vs. St. Joseph) allows us to evaluate the model's ability to disentangle local update dynamics from inter-county message passing.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the scope of COSINE as joint graph and symbolic dynamics discovery, and the claims are supported by the synthetic, ablation, efficiency, and epidemic-data experiments reported in the paper and appendix.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper includes a separate Limitations section (Section 5) discussing LLM dependence, bounded symbolic expressivity, pairwise coupling assumptions, non-identifiability under noisy data, and cautions for epidemic-data interpretation.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not present formal theoretical results requiring theorem statements or proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The method, objective, training procedure, hyperparameters, synthetic systems, baselines, evaluation metrics, and additional experimental details are described in the main text and appendix.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: A code repository URL is provided in a footnote on the first page. The paper describes the data, model, baselines, metrics, and training settings needed to reproduce the main results.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental settings, graph families, dynamical systems, baselines, optimizer, hyperparameters, LLM backend, and evaluation metrics are specified in the Experiments section and appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The main synthetic results and additional AUPR, MSE, ablation, and LLM-scale results report mean and standard deviation where repeated runs are used.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports the GPU used for LLM deployment and includes an efficiency analysis of epoch time and GPU memory across graph sizes.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses public aggregate epidemic data and includes an impact statement discussing the societal implications of the research.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The impact statement discusses the use of public, aggregate COVID-19 epidemic data and cautions against unsupported public-health interpretation or deployment.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release high-risk pretrained models, scraped sensitive datasets, or other assets that require special safeguards; the real-world experiment uses public aggregate epidemic data.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper cites the existing datasets, baselines, and models used, and the accompanying anonymous code release documents third-party asset sources and licenses.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The paper introduces COSINE and provides an anonymized code release; the method, training procedure, prompts, limitations, and experimental protocol are documented in the paper and appendix.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing experiments or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing experiments or human-subject research; the epidemic experiment uses public aggregate data.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: The paper describes LLM usage as a core component of COSINE, including LLM-guided library evolution, prompt details, and the LLM backends used in experiments.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.