KDD2020

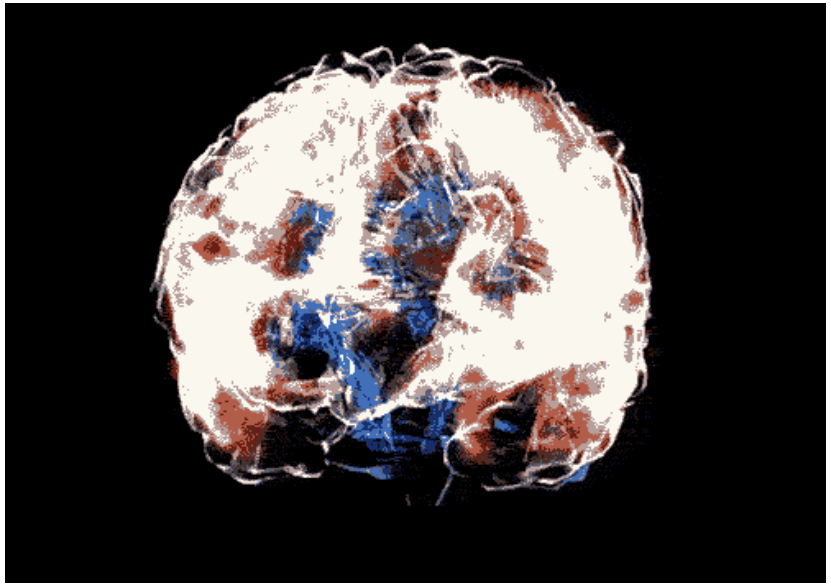
Weill Cornell Medicine

Neural Dynamics on Complex Networks

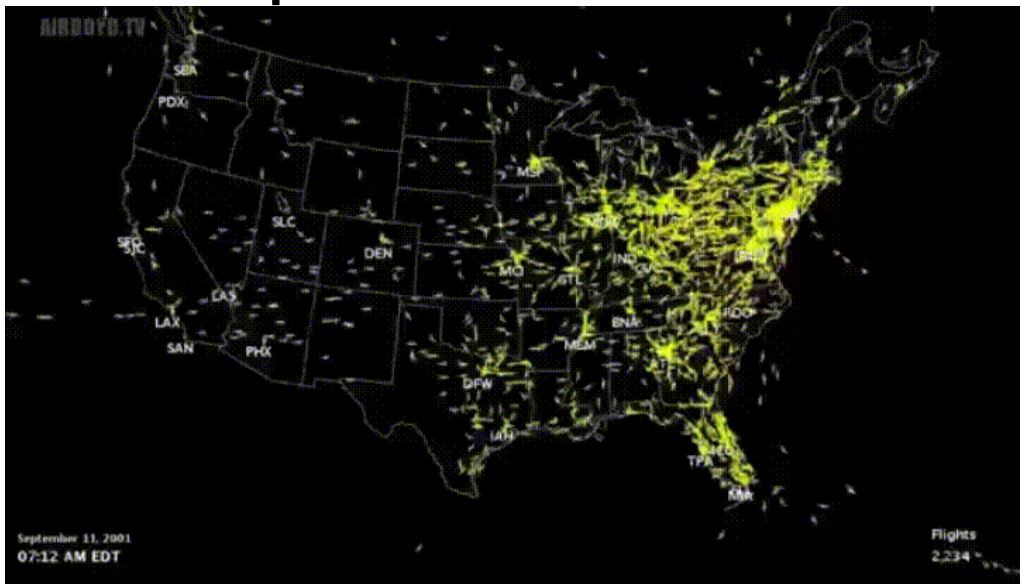
Chengxi Zang and Fei Wang
Weill Cornell Medicine
www.calvinzang.com

Network Dynamics of Complex Systems

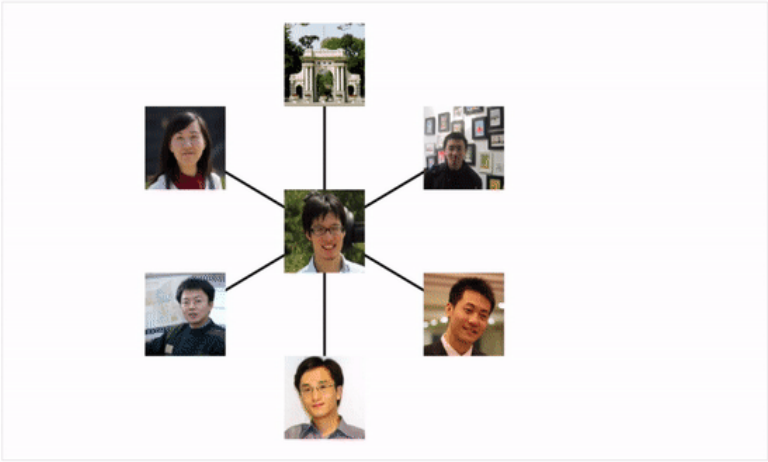
Brain and Bioelectrical flow



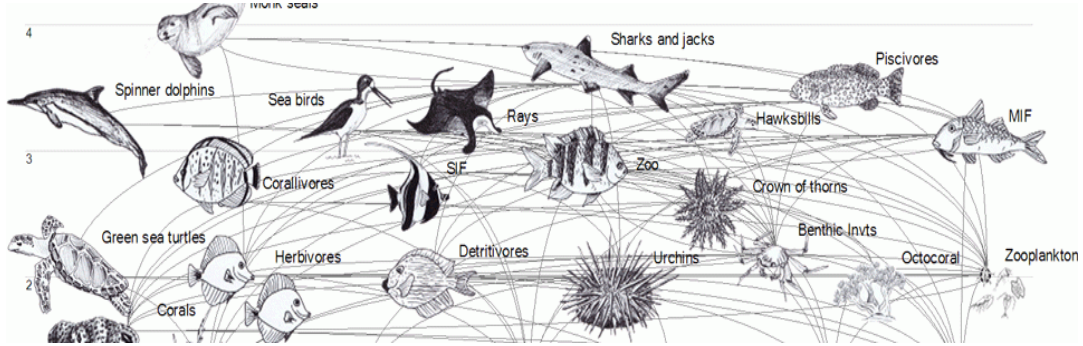
Transportation and Traffic flow



Social Networks and Information flow

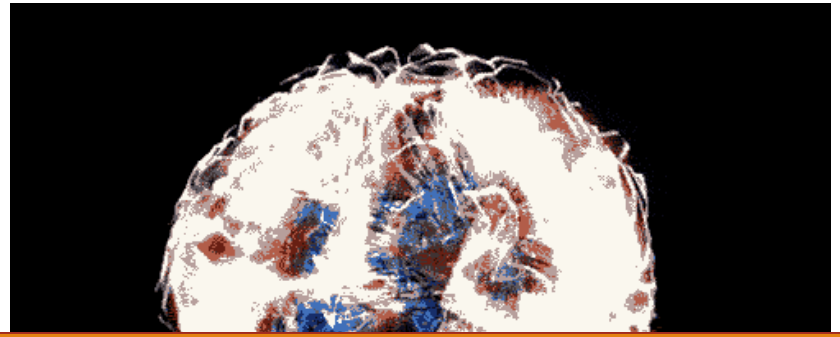


Ecological Systems and Energy flow



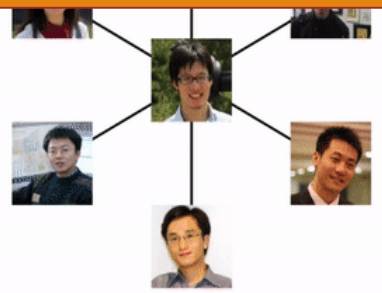
Problem

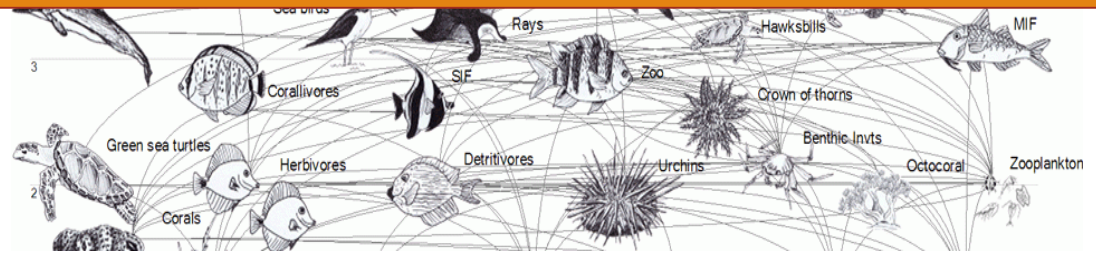
Brain and Bioelectrical flow



Transportation and Traffic flow

How to model and predict these network dynamics?





Problem: Math Formulation

- **Learning Dynamics on Networks/Graphs**
  - Graph: $G = (V, E)$, V represents nodes, E represents edges.
  - Dynamics of nodes: $X(t) \in \mathbb{R}^{n \times d}$ changes over continuous time t,
    - where n is the number of nodes, d is the number of features
  - How dynamics $\frac{\boldsymbol{dX(t)}}{\boldsymbol{dt}} = \boldsymbol{f(X(t), G, \theta, t)}$ change over continuous-time on the graph G?

Problem as a Prediction Task

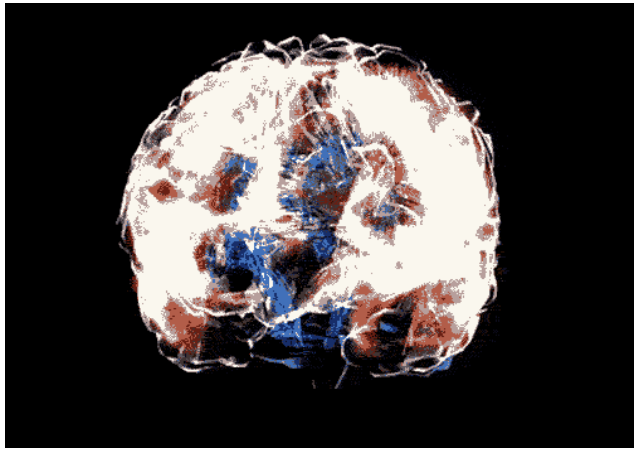
- **Continuous-time Network Dynamics Prediction:**
  - **Input**: G, $\left\{\widehat{X(t_1)}, \widehat{X(t_2)}, \ldots, \widehat{X(t_T)} \middle| 0 \leq t_1 < \cdots < t_T\right\}, t_1 < \cdots < t_T$ are **arbitrary time moments with irregular time intervals.**
  - ?A model of dynamics on graphs $\frac{\boldsymbol{dX(t)}}{\boldsymbol{dt}} = \boldsymbol{f}(\boldsymbol{X(t)}, \boldsymbol{G}, \boldsymbol{\theta}, \boldsymbol{t})$
  - **Output**: to predict $X(t)$ at an arbitrary time moment

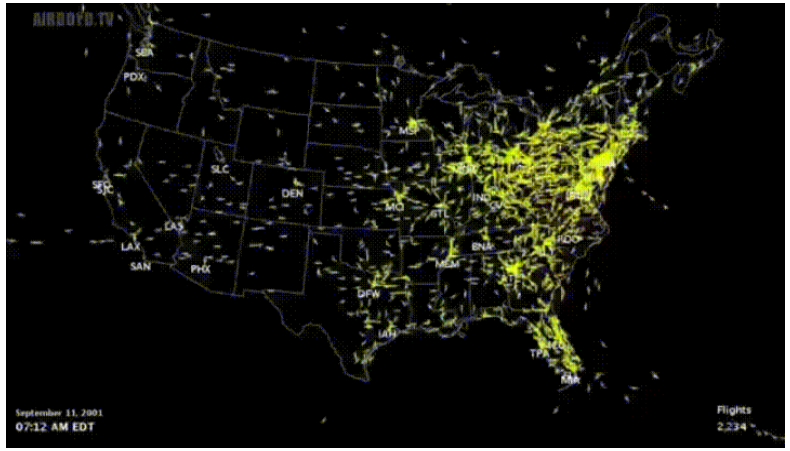
Problem: Prediction Tasks

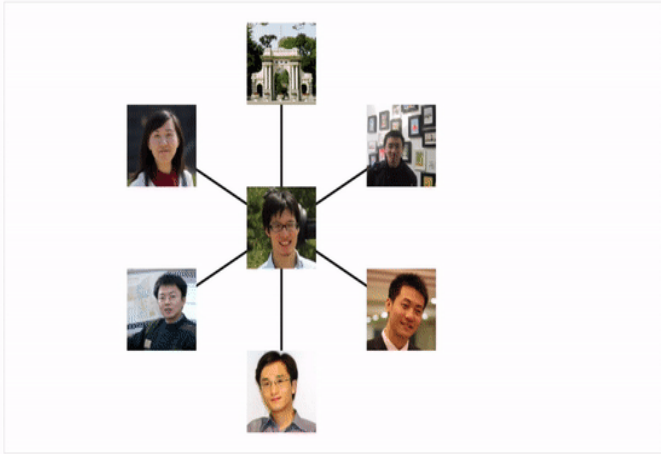
- **Continuous-time network dynamics prediction:**
  - Input: G, $\left\{\widehat{X(t_1)}, \widehat{X(t_2)}, \dots, \widehat{X(t_T)} \middle| 0 \le t_1 < \cdots < t_T\right\}, t_1 < \cdots < t_T$ are **arbitrary time moments**
  - A model of dynamics on graphs $\frac{dX(t)}{dt} = f(X(t), G, \theta, t)$
  - Output: to predict $X(t)$ at an arbitrary time moment
- **(Special case) Structured sequence prediction**
  - Input: G, $\left\{\widehat{X[1]}, \widehat{X[2]}, \dots, \widehat{X[T]} \middle| 0 \le 1 < \cdots < T\right\}$, an **ordered sequence**
  - Output: to predict next k steps $X[T+k]$?
- **(Special case) Node (semi-supervised) classification**
  - Input: G, $\hat{X} = [\hat{X}, Mask \odot \hat{Y}]$ **a snapshot** of features and node labels
  - Output: to complete $[\hat{X}, \hat{Y}]$?

Why Network Dynamics Matter?

- **To understand, predict, and control real-world complex systems in engineering and science.**
  - Brain dynamics, traffic dynamics, social dynamics, etc.

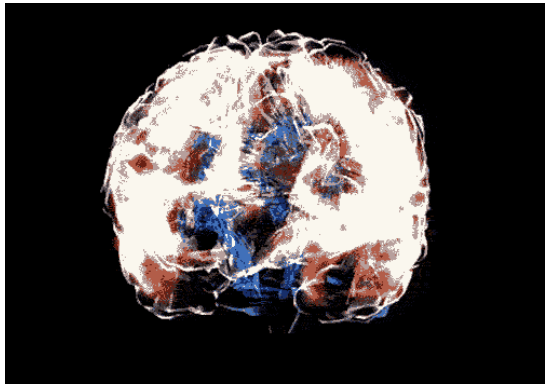


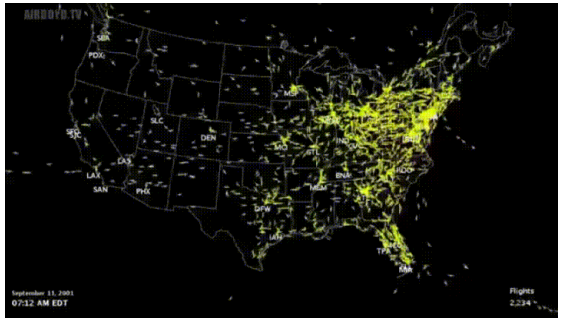


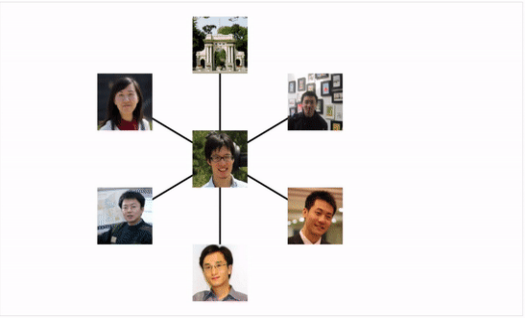


Why is it Hard?

- **Complex systems:**
  - High-dimensionality and Complex interactions
  - ≥ 100 nodes, ≥ 1000 interactions
- **Dynamics:**
  - Continuous-time, Nonlinear
- **Structural-dynamic dependencies**:
  - Difficult to be modeled by simple mechanistic models, no clear knowledge of their mechanisms

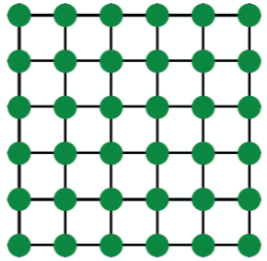




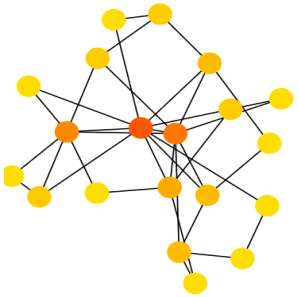


Why is it Hard?

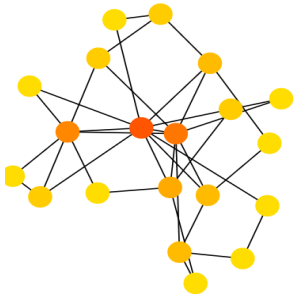
- Examples of Dynamics on Graphs



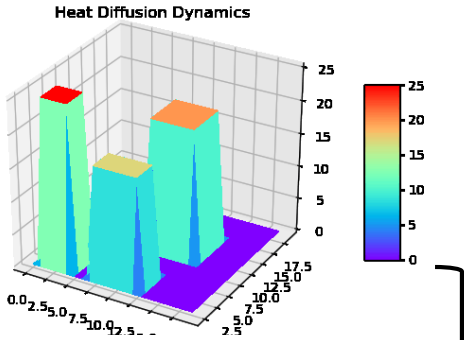
+ Linear Dynamics →

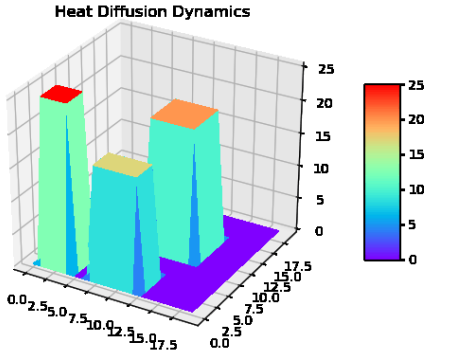


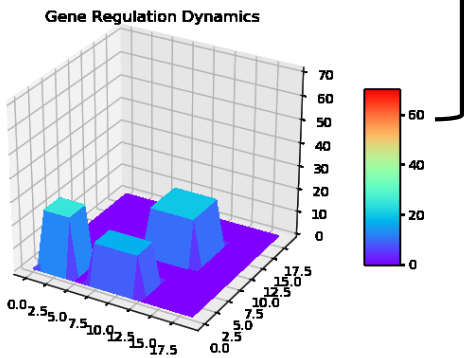
+ Linear Dynamics →



+ Non-Linear Dynamics →





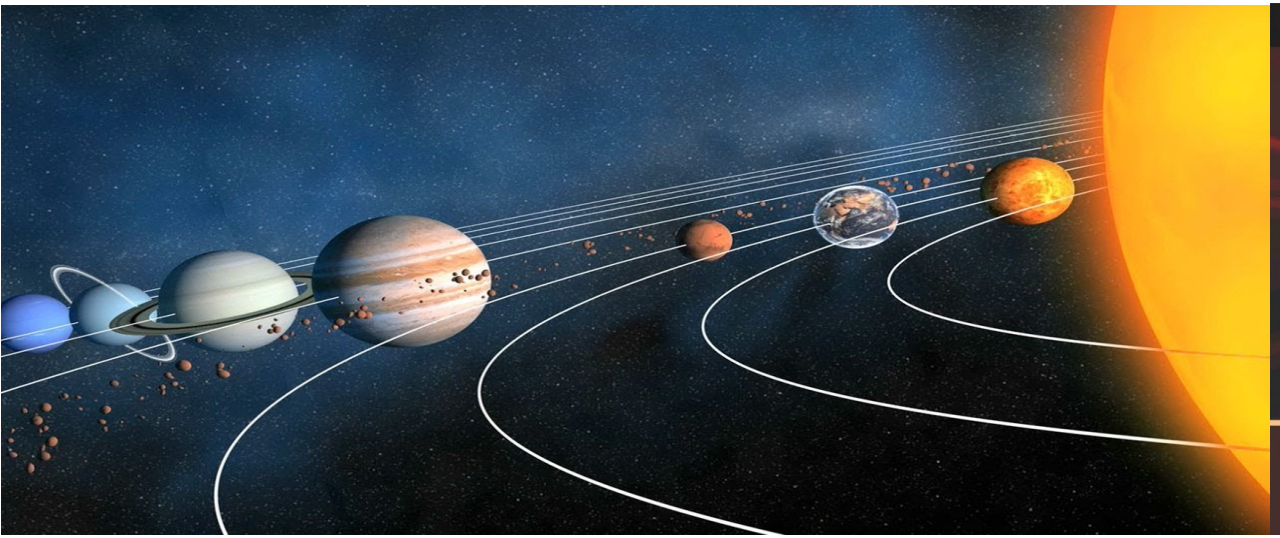


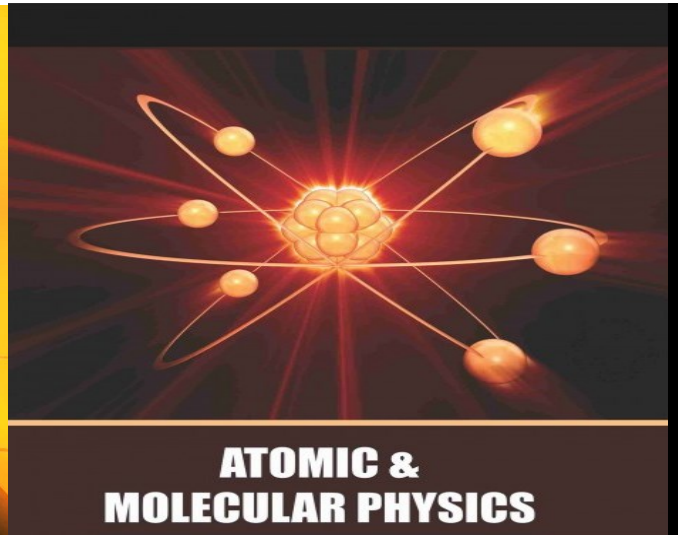
?

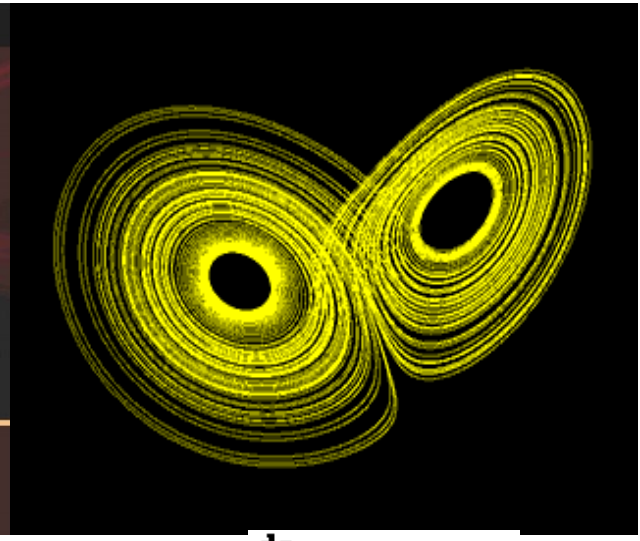
$f(X(t), G, \theta, t)$

Related Works I: Learning Continuous Time Dynamics

- **To learn continuous-time dynamics**
  - A clear knowledge of the mechanisms, small systems, few interaction terms, first principle from physical laws, mechanistic models,







$$\textbf{Macro}: \quad F = \frac{d(mv)}{dt}$$

$$\textbf{Micro}: \quad i\hbar\frac{\partial}{\partial t}\Psi(\mathbf{r},t) = \left[\frac{-\hbar^2}{2m}\nabla^2 + V(\mathbf{r},t)\right]\Psi(\mathbf{r},t)$$

$$\textbf{Chaos}: \quad \begin{aligned} \frac{dx}{dt} &= \sigma(y-x), \\ \frac{dy}{dt} &= x(\rho-z)-y, \\ \frac{dz}{dt} &= xy-\beta z. \end{aligned}$$

Related Works I: Dynamical Origins of Distribution Functions

❑1-D distribution ←→ Ordinary Differential Equation

	Data Distribution		Survival Analysis		Dynamic System		
	$f(x)$	$F(x)$	$\lambda(x)$	$\Lambda(x)$	$x_i(t)$	Dynamics $\frac{dx_i(t)}{dt}$	Interpretation
Exponential	$\alpha e^{-\alpha x}$	$1-e^{-\alpha x}$	α	αx	$\frac{\ln(\frac{t}{t_i})}{\alpha}$	$\frac{1}{\alpha t}$	GC
Power law	$\alpha x_0^{\alpha} x^{-(\alpha+1)}$	$1-(\frac{x_0}{x})^{\alpha}$	$\frac{\alpha}{x}$	$a \ln \frac{x}{x_0}$	$x_0(\frac{t}{t_i})^{\frac{1}{\alpha}}$	$\frac{x_i(t)}{\alpha t}$	PA + GC
Stretched exponential	$\frac{\alpha}{x^{\theta}} e^{-\frac{\alpha(x^{1-\theta}-x_0^{1-\theta})}{1-\theta}}$	$1-e^{-\frac{\alpha(x^{1-\theta}-x_0^{1-\theta})}{1-\theta}}$	$\frac{\alpha}{x^{\theta}}$	$\frac{\alpha}{1-\theta}(x^{1-\theta}-x_0^{1-\theta})$	$[\ln(\frac{t}{t_i})\frac{1-\theta}{\alpha}+x_0^{1-\theta}]^{\frac{1}{1-\theta}}$	$\frac{x_i^{\theta}(t)}{\alpha t}$	Non-linear PA + GC
Weibull	$\alpha\lambda^{\alpha}x^{\alpha-1}e^{-(\lambda x)^{\alpha}}$	$1-e^{-(\lambda x)^{\alpha}}$	$\alpha\lambda^{\alpha}x^{\alpha-1}$	$(\lambda x)^{\alpha}$	$\frac{(\ln \frac{t}{t_i})^{\frac{1}{\alpha}}}{\lambda}$	$\frac{x_i^{1-\alpha}(t)}{\lambda^{\alpha}\alpha t}$	Non-linear PA + GC
Log-logistic	$\frac{\lambda\alpha(\lambda x)^{\alpha-1}}{[1+(\lambda x)^{\alpha}]^2}$	$1-\frac{1}{1+(\lambda x)^{\alpha}}$	$\frac{\lambda\alpha(\lambda x)^{\alpha-1}}{1+(\lambda x)^{\alpha}}$	$\ln[1+(\lambda x)^{\alpha}]$	$\frac{(\frac{t}{t_i}-1)^{\frac{1}{\alpha}}}{\lambda}$	$\frac{x_i(t)}{\alpha(t-t_i)}$	PA + Since then GC
Sigmoid	$\frac{e^x}{(1+e^x)^2}$	$1-\frac{1}{1+e^x}$	$\frac{e^x}{1+e^x}$	$\ln(1+e^x)$	$\ln(\frac{t}{t_i}-1)$	$\frac{1}{t-t_i}$	Since then GC
Log-normal *	$\frac{1}{x\sqrt{2\pi}}e^{-\frac{(\ln x)^2}{2}}$	$\Phi(\ln x)$	$\frac{f(x)}{1-\Phi(\ln x)}$	$-\ln[1-\Phi(\ln x)]$	$e^{\Phi^{-1}(1-\frac{t_i}{t})}$	$x_i\frac{d\Phi^{-1}(z)}{dz}\frac{t_i}{t^2}$	PA + Square GC
Normal *	$\frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}$	$\Phi(x)$	$\frac{f(x)}{1-\Phi(x)}$	$-\ln[1-\Phi(x)]$	$\Phi^{-1}(1-\frac{t_i}{t})$	$\frac{d\Phi^{-1}(z)}{dz}\frac{t_i}{t^2}$	Square GC
Uniform	$\frac{1}{b-a}$	$\frac{x-a}{b-a}$	$\frac{1}{b-x}$	$\ln\frac{b-a}{b-x}$	$b-(b-a)\frac{t_i}{t}$	$\frac{b-x_i(t)}{t}$	EL + GC

Table from Zang etc., KDD2019, [Dynamical Origins of Distribution Functions]

Data-driven Dynamics for Small Systems

- **Data-driven discovery of ODEs/ PDEs**
  - **Sparse Regression**
  - Residual Network
  - Etc.
- **Small systems!**
  - <10 nodes & interactions
  - Combinatorial complexity
  - Not for complex systems

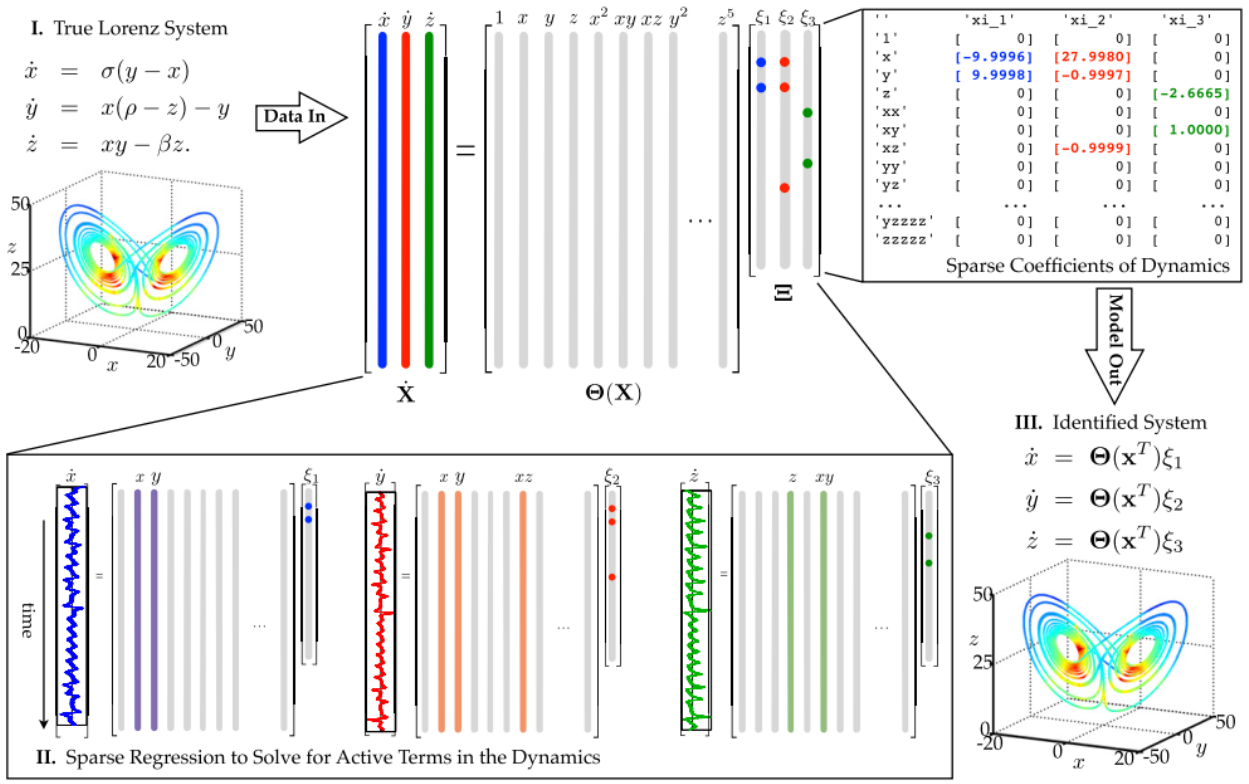


Image from: Brunton et al. 2016. [Discovering governing equations from data by sparse identification of nonlinear dynamical systems.](#) PNAS

Related Works II: Structured Sequence Learning

- **Defined characteristics**
  - Dynamics on graphs are regularly-sampled with same time intervals, **an ordered sequence** instead of physical time.
- **Temporal Graph Neural Networks**
  - RNN + CNN
  - RNN + GNN
    - X[t+1]=LSTM(GCN([t], G))
- **Limitations:**
  - Only ordered sequence instead of continuous physical time

Seo et al. 2016. Structured Sequence Modeling with Graph Convolutional Recurrent Networks.
Wu et al. 2019. A Comprehensive Survey on Graph Neural Networks

Related Works III: Node (Semi-supervised) Classification

- **Defined characteristics**
  - **One-snapshot** features and some nodes' labels on graphs
  - Goal: to infer labels of each node
- **Graph Neural Networks**
  - GCN,
  - GAT, etc.

$$Z = f(X, A) = \mathrm{softmax}\Big(\hat{A}\ \mathrm{ReLU}\Big(\hat{A}XW^{(0)}\Big)\,W^{(1)}\Big)$$

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^{k}\mathbf{W}^{k}\vec{h}_j\right)$$

- **Limitations**
  - 1 or 2 layers, over-smoothing when deep
  - Lacking a continuous-time dynamics view
    - To spread features or labels on graphs
    - Continuous-time: more fine-grained capturing diffusion

Kipf et al. 2016. Semi-Supervised Classification with Graph Convolutional Networks
Velickovic et al. 2017. Graph Attention Networks

Goal: A Unified Framework

- **Once we LEARN a model of dynamics on graphs** $\frac{dX(t)}{dt} = f(X(t), G, \theta, t)$
- **Continuous-time network dynamics prediction:**
  - Input: G, $\{\widehat{X(t_1)}, \widehat{X(t_2)}, \dots, \widehat{X(t_T)} | 0 \leq t_1 < \cdots < t_T\}, t_1 < \cdots < t_T$ are **arbitrary time moments**
  - Output: to predict $X(t)$ at an arbitrary time moment
- **(Special case) Structured sequence prediction**
  - Input: G, $\{\widehat{X[1]}, \widehat{X[2]}, \dots, \widehat{X[T]} | 0 \leq 1 < \cdots < T\}$, an **ordered sequence**
  - Output: to predict next k steps $X[T + k]$
- **(Special case) Node (semi-supervised) classification**
  - Input: G, $\hat{X} = [\hat{X}, Mask \odot \hat{Y}]$ **a snapshot** of features and node labels
  - Output: to complete $[\hat{X}, \hat{Y}]$

Our Idea

- **Differential Equation Systems**
  - Differential Equations are general tools to describe the dynamics of systems
- **Graph Neural Networks**
  - Graphs are general tools to describe the structures of systems
  - GNNs, Temporal GNNs, are the state-of-the-art computational tools driven by data for linked data
- **How to leverage both Differential Equation systems and Graph Neural Networks?**

Neural Dynamics on Complex Networks (NDCN)

❑ Differential Equations + Graph Neural Networks

- Differential Equation Systems $\frac{dX(t)}{dt} = f(X(t), G, W, t)$ modeled by a Graph Neural Network.
- "Deep" and Continuous Depth: $X(t) = X(0) + \int_0^t f(X(\tau), G, W, \tau) d\tau$ for states at arbitrary time $t \in [0, T]$
- Or modeling network dynamics in a latent space $X_h(t)$, with encoding and decoding layers
- Learned as an optimization problem for running dynamics and terminal loss

$$\underset{W(t),\Theta(T)}{\arg\min} \quad \mathcal{L} = \int_0^T \mathcal{R}\left(X, G, W, t\right) dt + \mathcal{S}\left(Y(X(T), \Theta)\right)$$

$$\text{subject to} \quad X_h(t) = f_e\left(X(t), W_e\right), \quad X(0) = X_0$$

$$X_h(t) = X_h(0) + \int_0^t f\left(X_h, G, W_h, \tau\right) d\tau$$

$$X(t) = f_d\left(X_h(t), W_d\right)$$

A NDCN Instance

❑ODEs + GNN

- Differential Equation systems: $\frac{dX(t)}{dt} = f(X(t), G, W, t)$ is modeled by a Graph Neural Network.
- Continuous-depth Deep model: $X(t) = X(0) + \int_0^t f(X(\tau), G, W, \tau)d\tau$ for dynamics at arbitrary time t

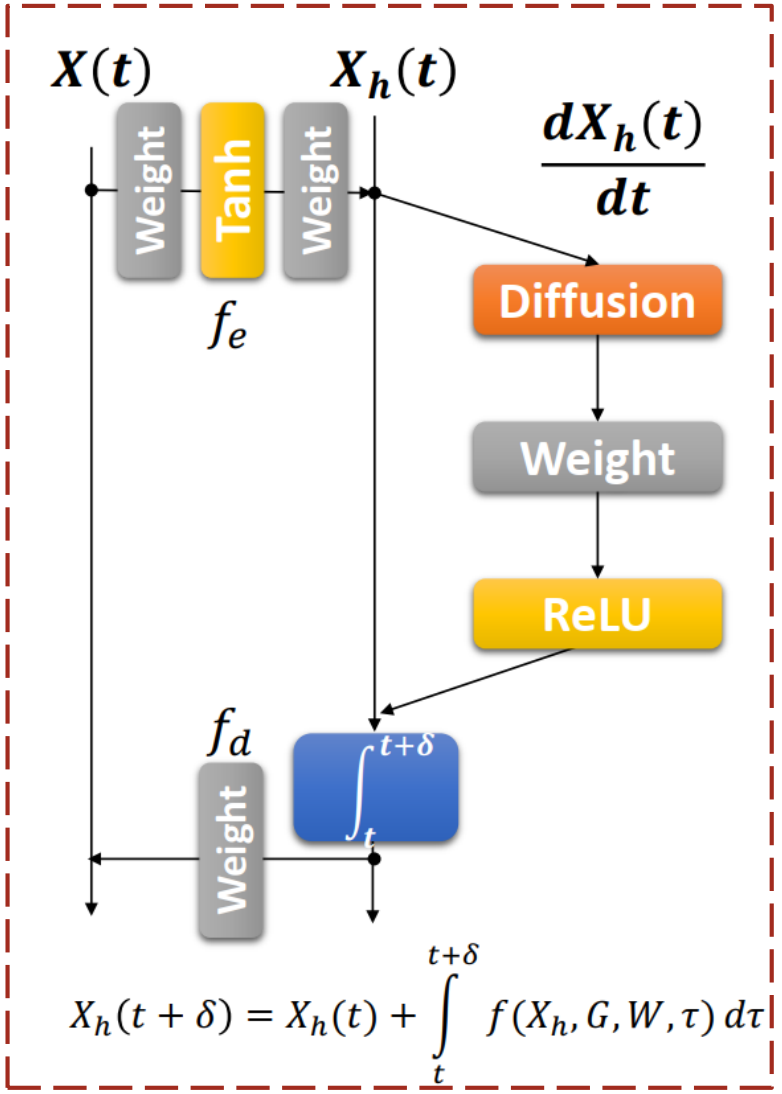
$$\underset{W_*, b_*}{\text{argmin}} \quad \mathcal{L} = \int_0^T |X(t) - \hat{X(t)}|\, dt$$

$$\text{subject to} \quad X_h(t) = \tanh\left(X(t)W_e + b_e\right)W_0 + b_0$$

$$\frac{dX_h(t)}{dt} = \text{ReLU}\left(\Phi X_h(t)W + b\right), X_h(0)$$

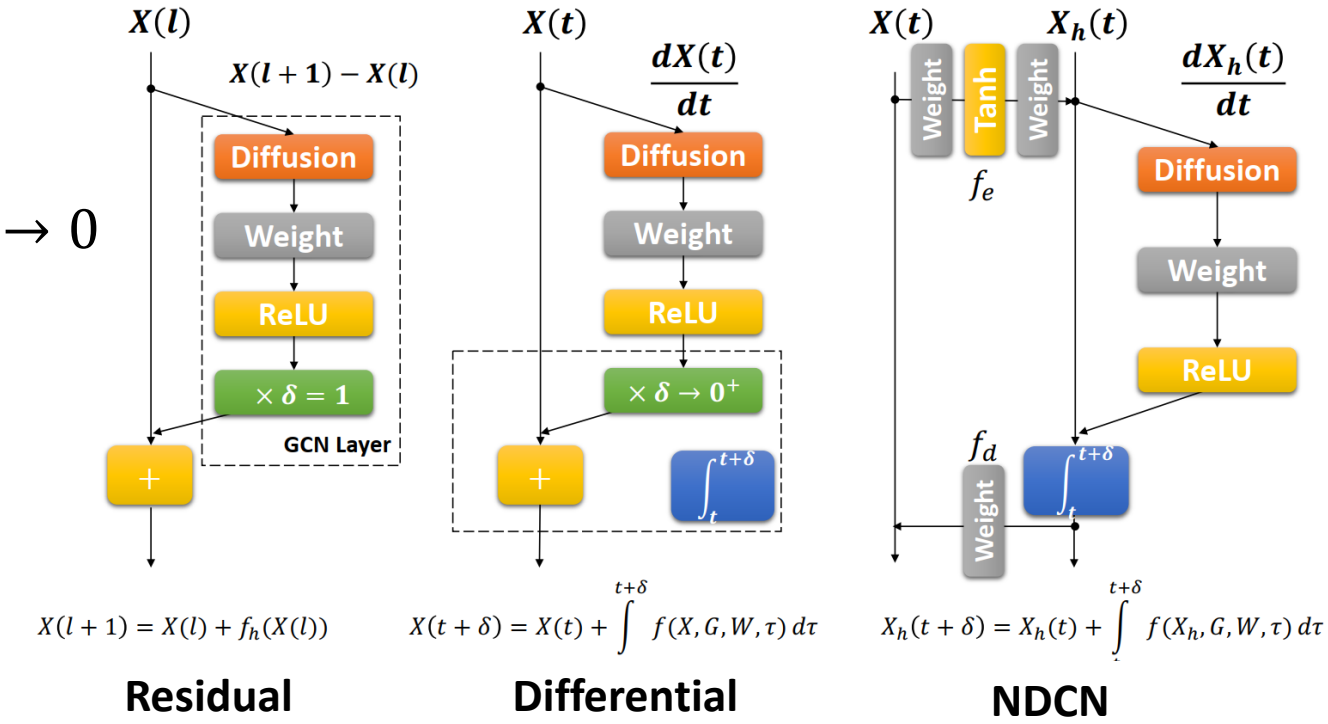
$$X(t) = X_h(t)W_d + b_d$$

$$\Phi = D^{-\frac{1}{2}}(D - A)D^{-\frac{1}{2}} \in \mathbb{R}^{n \times n}$$



Interpretation from Graph Neural Networks

- **GNN, Residual-GNN, ODE-GNN, NDCN**
  - GNN: $X_{t+1} = f(G, X_t, \theta_t)$
  - Residual-GNN: $X_{t+1} = X_t + f(G, X_t, \theta_t)$
  - ODE-GNN: $X_{t+\delta} = X_t + \delta * f(G, X_t, \theta_t), \delta \to 0$
    - $\frac{dX}{dt} = f(G, X_t, \theta_t)$
  - ODE-GNN in a latent space: NDCN
- **Our model is an ODE-GNN**
  - **Continuous layers/time**
  - **A real (~~integer~~) number of depth → time**
  - **Latent space dynamics**



$X(l+1) = X(l) + f_h(X(l))$

Residual

$X(t+\delta) = X(t) + \int_t^{t+\delta} f(X, G, W, \tau)\, d\tau$

Differential

$X_h(t+\delta) = X_h(t) + \int_t^{t+\delta} f(X_h, G, W, \tau)\, d\tau$

NDCN

Interpretation from RNN and Temporal GNN

- **RNN, Temporal GNN and our NDCN**
  - RNN or Temporal GNN
    - $h_t = f(h_{t-1}, x_t, \theta_t)$ or $h_t = f(h_{t-1}, G * x_t, \theta_t)$
    - $y_t = o(h_t, w_t)$
  - Residual RNN or Residual Temporal GNN with skip connection
    - $h_t = h_{t-1} + f(h_{t-1}, x_t, \theta_t)$ or $h_t = h_{t-1} + f(h_{t-1}, G * x_t, \theta_t)$
    - $y_t = o(h_t, w_t)$
  - ODE-RNN or ODE-GNN
    - $\frac{dh_t}{dt} = f(h_t, x_t, \theta_t)$ or $\frac{dh_t}{dt} = f(h_t, G * x_t, \theta_t)$
    - $y_t = o(h_t, w_t)$
- **Our model is an ODE-GNN**
  - **Learning continuous-time network dynamics**
  - **Encompassing Temporal GNN by discretization**
  - **Encompassing RNN by not using graph convolution**

Exp1: Learning Continuous-time Network Dynamics

The Problem:

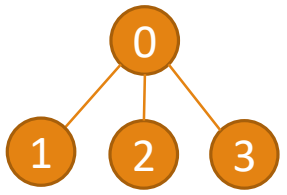
- Input: G, $\{\widehat{X(t_1)}, \widehat{X(t_2)}, \ldots, \widehat{X(t_T)} | 0 \leq t_1 < \cdots < t_T\}, t_1 < \cdots < t_T$ are **arbitrary time moments with different time intervals**
- Output: $X(t)$, t is an arbitrary time moment
  - **interpolation** prediction: $t < t_T$ and $\neq \{t_1 < \cdots < t_T\}$
  - **extrapolation** prediction: $t > t_T$

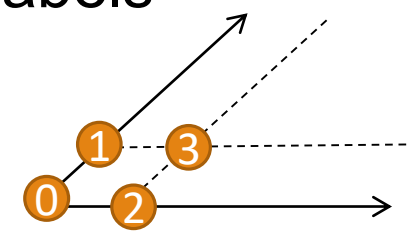
Setups:

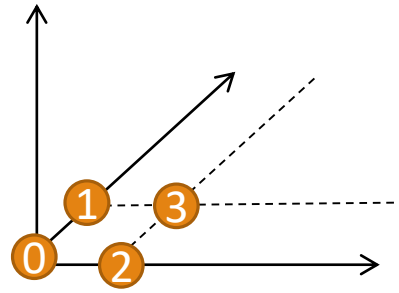
- 120 irregularly sampled snapshots of network dynamics
- First 100: 80 for train 20 for testing interpolation
- Last 20: testing for extrapolation

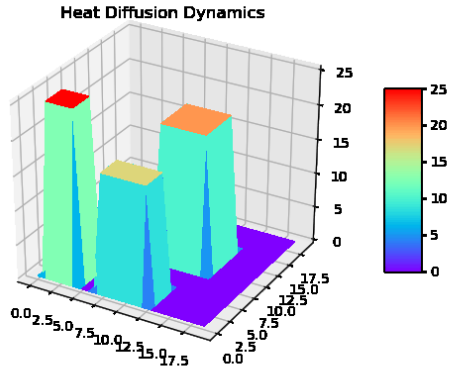
Canonical Network Dynamics in Physics and Biology

- **Real-world Dynamics on Graph (adjacency matrix A)**
  - **Heat diffusion**: $\frac{d\overrightarrow{x_i(t)}}{dt} = -k_{i,j}\sum_{j=1}^{n} A_{i,j}(\overrightarrow{x_i(t)} - \overrightarrow{x_j(t)})$
  - **Mutualistic interaction**: $\frac{d\overrightarrow{x_i(t)}}{dt} = b_i + \overrightarrow{x_i(t)}\left(1 - \frac{\overrightarrow{x_i(t)}}{k_i}\right)\left(\frac{\overrightarrow{x_i(t)}}{c_i} - 1\right) + \sum_{j=1}^{n} A_{i,j}\frac{\overrightarrow{x_i(t)} * \overrightarrow{x_j(t)}}{d_i + \overrightarrow{e_i x_i(t)} + h_j\overrightarrow{x_j(t)}}$
  - **Gene regulatory**: $\frac{d\overrightarrow{x_i(t)}}{dt} = -b_i\overrightarrow{x_i(t)}^f + \sum_{j=1}^{n} A_{i,j}\frac{\overrightarrow{x_j(t)}^h}{\overrightarrow{x_j(t)}^h + 1}$
- **Graphs**
  - Grid, Random, power-law, small-world, community, etc.
- **Visualizing dynamics on graph**
  - Nodes are numbered by community labels
  - Mapped into a $\mathbb{N}^2$ grid
  - $X(t)^{n*1}$: $\mathbb{N}^2 \rightarrow \mathbb{R}$









Exp1: Learning Continuous-time Network Dynamics

Baselines: ablation models

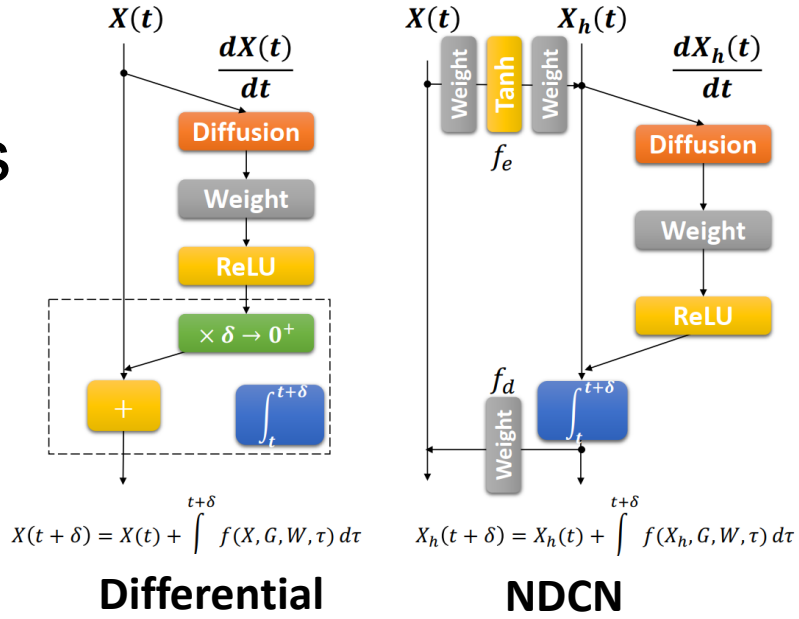
- ODE-GNN without encoding/decoding layers
- Neural ODE Network
  - No graph diffusion
- NDCN without control parameter W
  - Determined dynamics

$$\underset{W_*, b_*}{\text{argmin}} \quad \mathcal{L} = \int_0^T |X(t) - \hat{X}(t)|\, dt$$

$$\text{subject to} \quad X_h(t) = \tanh\left(X(t)W_e + b_e\right)W_0 + b_0$$

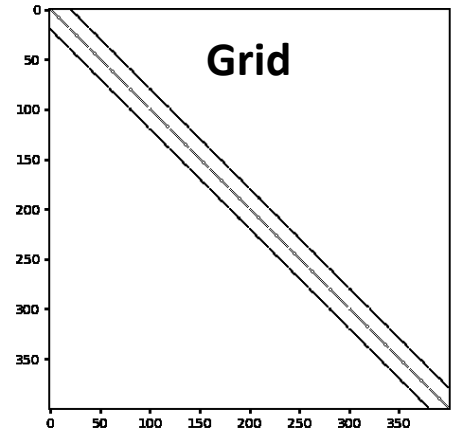
$$\frac{dX_h(t)}{dt} = \text{ReLU}\left(\Phi X_h(t)W + b\right), X_h(0)$$

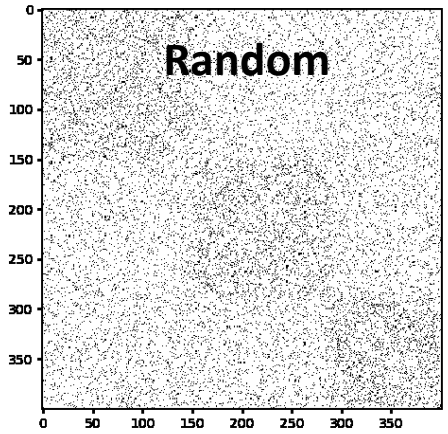
$$X(t) = X_h(t)W_d + b_d$$

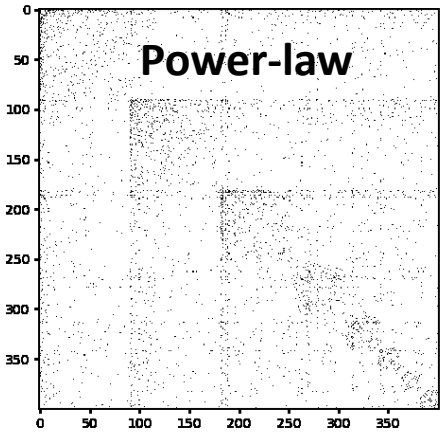


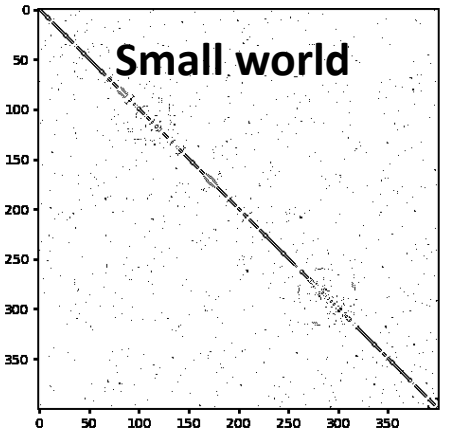
Chen et al. 2019. Neural Ordinary Differential Equations. *NeurIPS*.

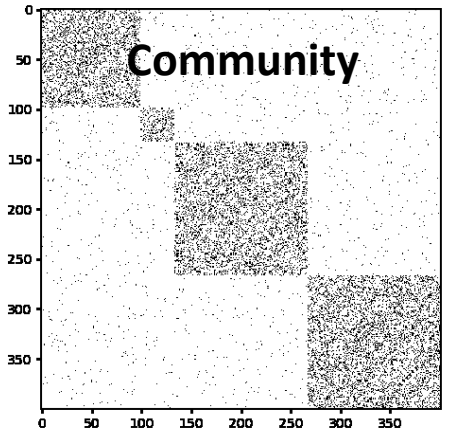
Exp1: Heat Diffusion on Different Graphs



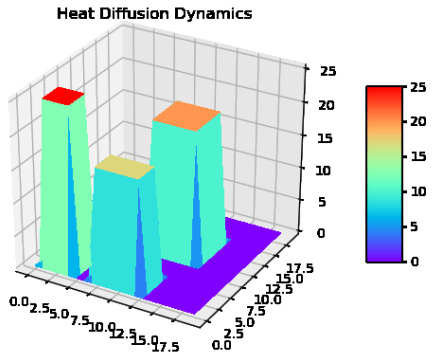


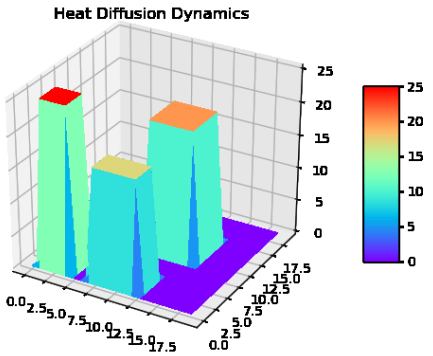


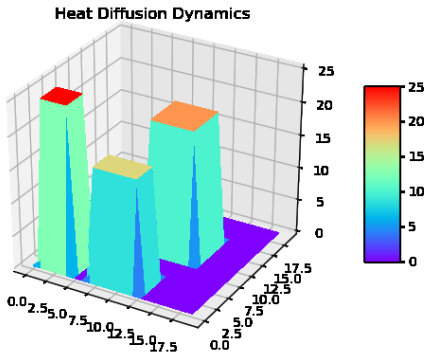


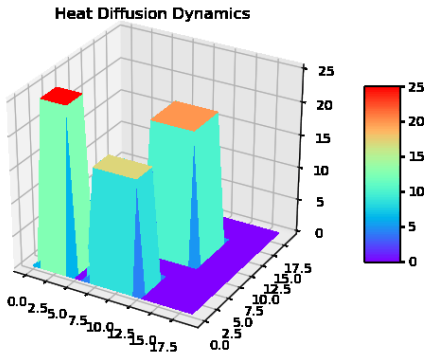


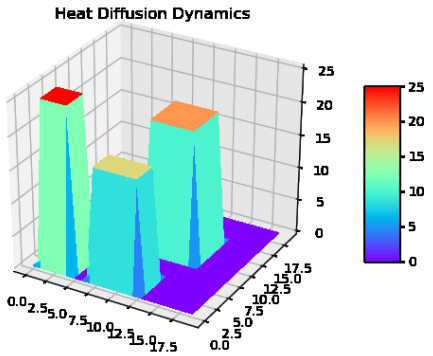
Ground Truth



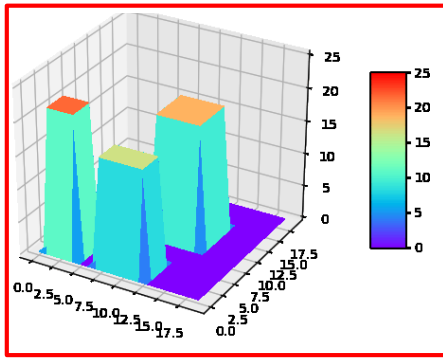


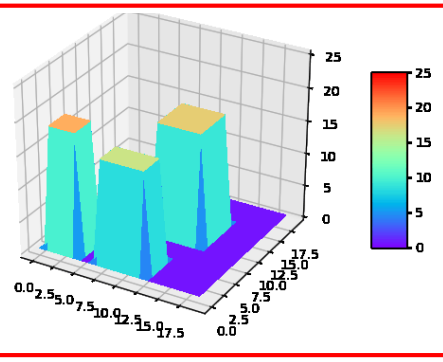


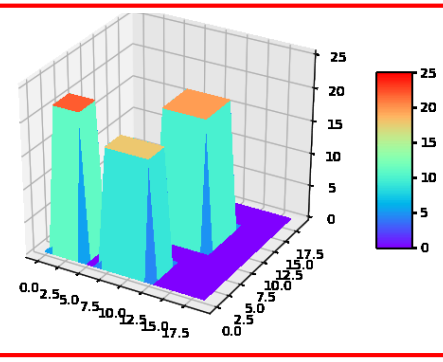


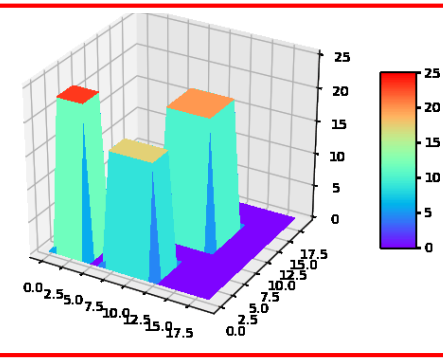


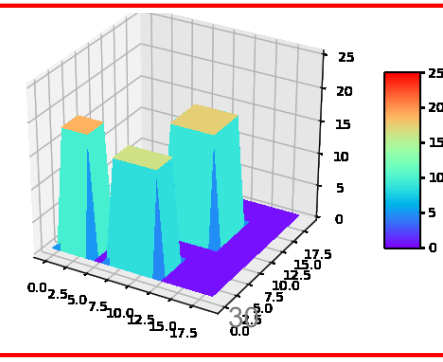
NDCN



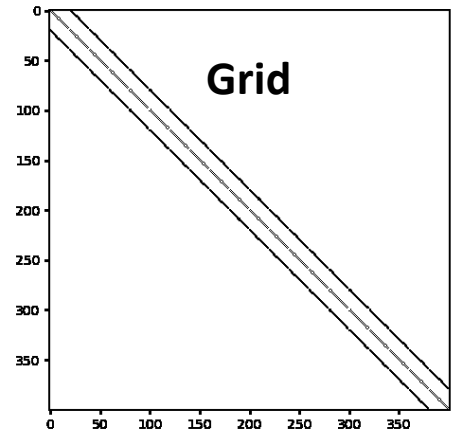


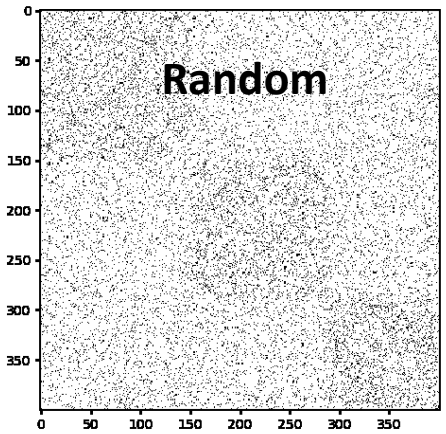


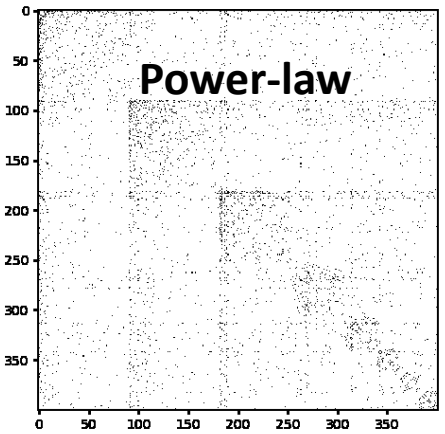


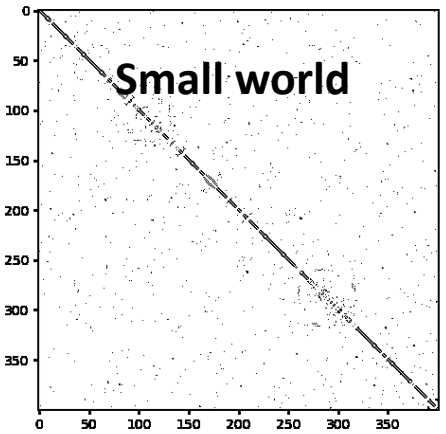


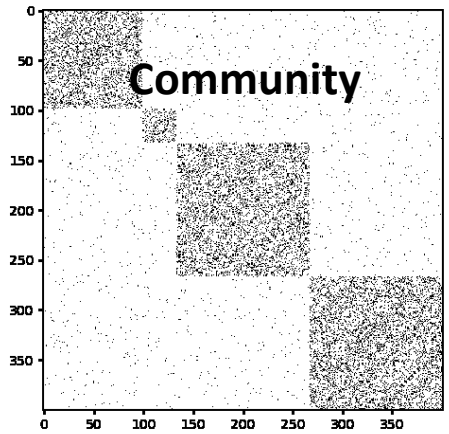
Exp1: Mutualistic Dynamics on Different Graphs



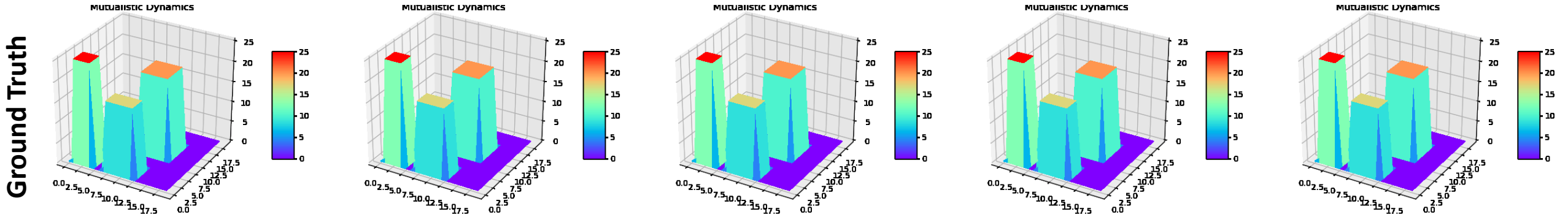




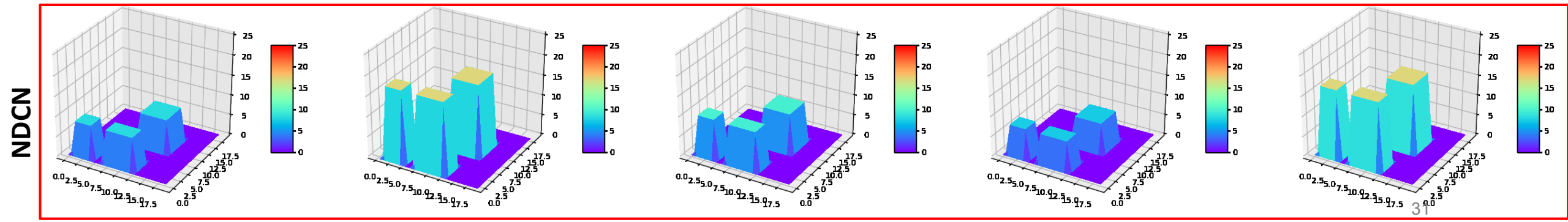




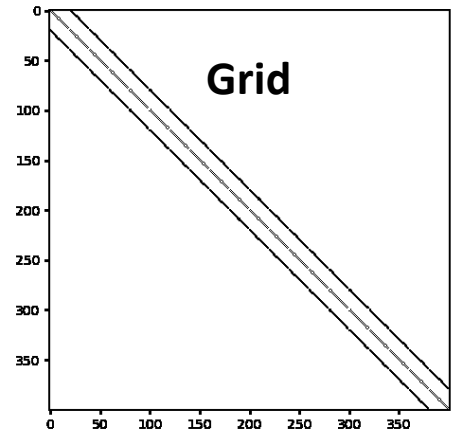
Ground Truth

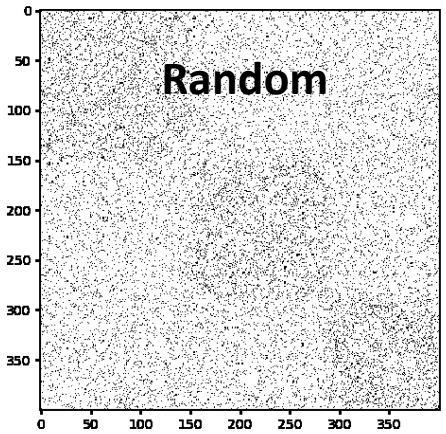


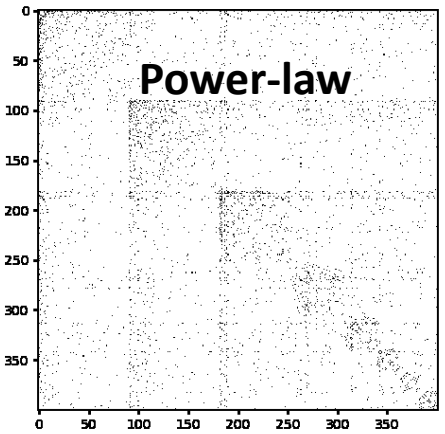
NDCN

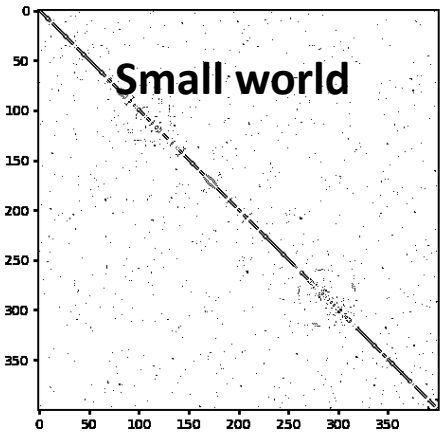


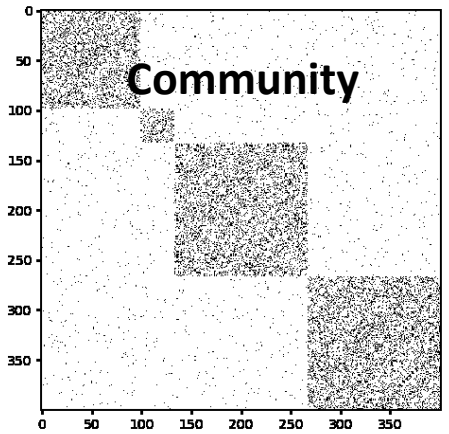
Exp1: Gene Dynamics on Different Graphs



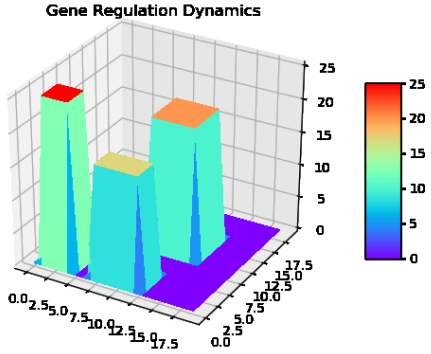


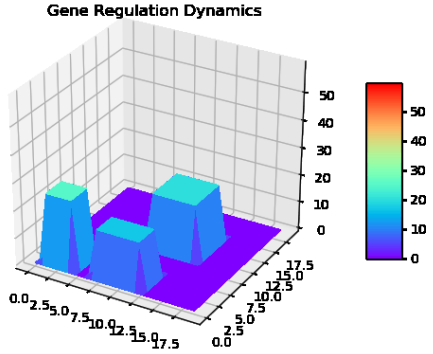


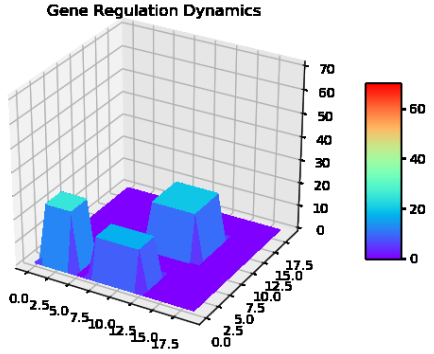


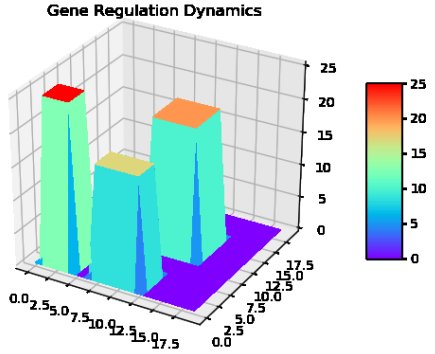


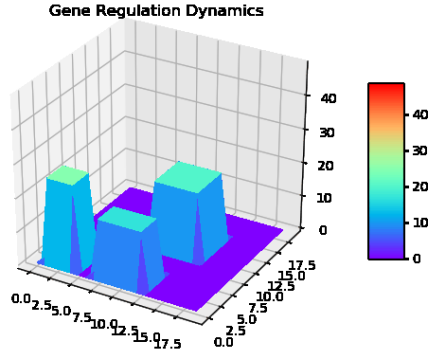
Ground Truth



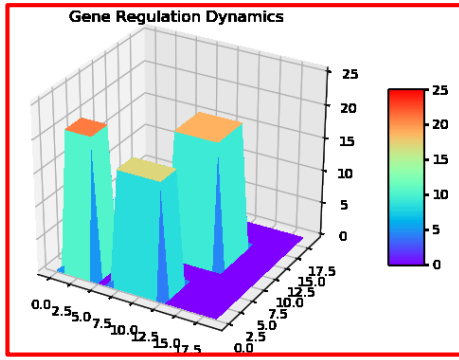


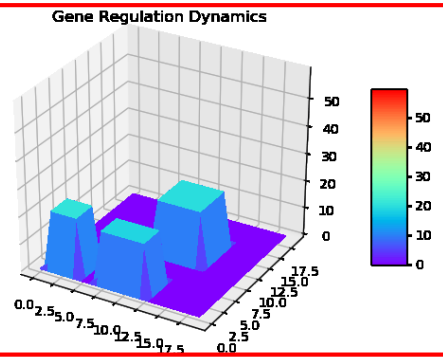


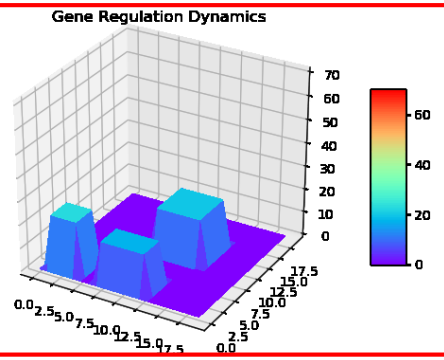


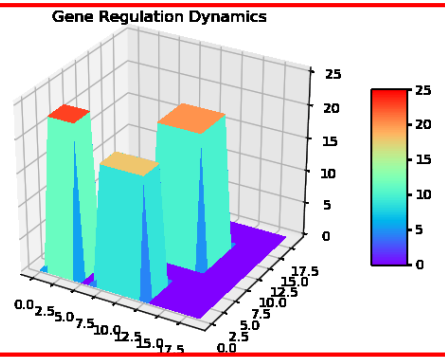


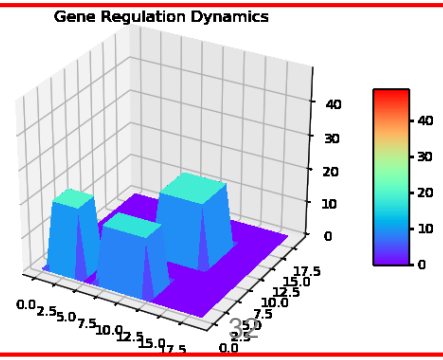
NDCN











Exp1: Results for Continuous-time Extrapolation

- **Mean Absolute Percentage Error**
- **20 runs for 3 dynamics on 5 graphs**
- **Our model achieves lowest error**

Table 1: **Continuous-time Extrapolation Prediction.** Our NDCN predicts different continuous-time network dynamics accurately. Each result is the normalized ℓ_1 error with standard deviation (in percentage %) from 20 runs for 3 dynamics on 5 networks by each method.

		Grid	Random	Power Law	Small World	Community
Heat Diffusion	No-Encode	29.9 ± 7.3	27.8 ± 5.1	24.9 ± 5.2	24.8 ± 3.2	30.2 ± 4.4
	No-Graph	30.5 ± 1.7	5.8 ± 1.3	6.8 ± 0.5	10.7 ± 0.6	24.3 ± 3.0
	No-Control	73.4 ± 14.4	28.2 ± 4.0	25.2 ± 4.3	30.8 ± 4.7	37.1 ± 3.7
	NDCN	$\mathbf{4.1 \pm 1.2}$	$\mathbf{4.3 \pm 1.6}$	$\mathbf{4.9 \pm 0.5}$	$\mathbf{2.5 \pm 0.4}$	$\mathbf{4.8 \pm 1.0}$
Mutualistic Interaction	No-Encode	45.3 ± 3.7	9.1 ± 2.9	29.9 ± 8.8	54.5 ± 3.6	14.5 ± 5.0
	No-Graph	56.4 ± 1.1	6.7 ± 2.8	14.8 ± 6.3	54.5 ± 1.0	9.5 ± 1.5
	No-Control	140.7 ± 13.0	10.8 ± 4.3	106.2 ± 42.6	115.8 ± 12.9	16.9 ± 3.1
	NDCN	$\mathbf{26.7 \pm 4.7}$	$\mathbf{3.8 \pm 1.8}$	$\mathbf{7.4 \pm 2.6}$	$\mathbf{14.4 \pm 3.3}$	$\mathbf{3.6 \pm 1.5}$
Gene Regulation	No-Encode	31.7 ± 14.1	17.5 ± 13.0	33.7 ± 9.9	25.5 ± 7.0	26.3 ± 10.4
	No-Graph	13.3 ± 0.9	12.2 ± 0.2	43.7 ± 0.3	15.4 ± 0.3	19.6 ± 0.5
	No-Control	65.2 ± 14.2	68.2 ± 6.6	70.3 ± 7.7	58.6 ± 17.4	64.2 ± 7.0
	NDCN	$\mathbf{16.0 \pm 7.2}$	$\mathbf{1.8 \pm 0.5}$	$\mathbf{3.6 \pm 0.9}$	$\mathbf{4.3 \pm 0.9}$	$\mathbf{2.5 \pm 0.6}$

Exp1: Results for Continuous-time Interpolation

- Interpolation is easier than extrapolation
- Our model achieves lowest error

Table 2: **Continuous-time Interpolation Prediction.** Our NDCN predicts different continuous-time network dynamics accurately. Each result is the normalized ℓ_1 error with standard deviation (in percentage %) from 20 runs for 3 dynamics on 5 networks by each method.

		Grid	Random	Power Law	Small World	Community
Heat Diffusion	No-Encode	32.0 ± 12.7	26.7 ± 4.4	25.7 ± 3.8	27.9 ± 7.3	35.0 ± 6.3
	No-Graph	41.9 ± 1.8	9.4 ± 0.6	18.2 ± 1.5	25.0 ± 2.1	25.0 ± 1.4
	No-Control	56.8 ± 2.8	32.2 ± 7.0	33.5 ± 5.7	40.4 ± 3.4	39.1 ± 4.5
	NDCN	$\mathbf{3.2 \pm 0.6}$	$\mathbf{3.2 \pm 0.4}$	$\mathbf{5.6 \pm 0.6}$	$\mathbf{3.4 \pm 0.4}$	$\mathbf{4.3 \pm 0.5}$
Mutualistic Interaction	No-Encode	28.9 ± 2.0	19.9 ± 6.5	34.5 ± 13.4	27.6 ± 2.6	25.5 ± 8.7
	No-Graph	28.7 ± 4.5	7.8 ± 2.4	23.2 ± 4.2	26.9 ± 3.8	14.1 ± 2.4
	No-Control	72.2 ± 4.1	22.5 ± 10.2	63.8 ± 3.9	67.9 ± 2.9	33.9 ± 12.3
	NDCN	$\mathbf{7.6 \pm 1.1}$	$\mathbf{6.6 \pm 2.4}$	$\mathbf{6.5 \pm 1.3}$	$\mathbf{4.7 \pm 0.7}$	$\mathbf{7.9 \pm 2.9}$
Gene Regulation	No-Encode	39.2 ± 13.0	14.5 ± 12.4	33.6 ± 10.1	27.7 ± 9.4	21.2 ± 10.4
	No-Graph	25.2 ± 2.3	11.9 ± 0.2	39.4 ± 1.3	15.7 ± 0.7	18.9 ± 0.3
	No-Control	66.9 ± 8.8	31.7 ± 5.2	40.3 ± 6.6	49.0 ± 8.0	35.5 ± 5.3
	NDCN	$\mathbf{5.8 \pm 1.0}$	$\mathbf{1.5 \pm 0.6}$	$\mathbf{2.9 \pm 0.5}$	$\mathbf{4.2 \pm 0.9}$	$\mathbf{2.3 \pm 0.6}$

Exp2: Structured Sequence Prediction

- **The Problem (Structured sequence prediction):**
  - Input: G, $\{\widehat{X[1]}, \widehat{X[2]}, \ldots, \widehat{X[T]} | 0 \leq 1 < \cdots < T\}, 1, .. T$ are regularly-sampled with the same time intervals
    - with an emphasis on an ordered sequence rather than time
  - Output: $X[\mathrm{T} + \mathrm{M}]$, next M steps
    - extrapolation prediction
- **Setups:**
  - 100 regularly sampled snapshots of network dynamics
  - First 80 for training, last 20 for testing

Exp2: Structured Sequence Prediction

- **Baselines: temporal-GNN models**
  - LSTM-GNN
    - X[t+1]=LSTM(GCN([t], G))
  - GRU-GNN
    - X[t+1]=GRU(GCN([t], G))
  - RNN-GNN
    - X[t+1]=RNN(GCN([t], G))

Seo et al. 2016. Structured Sequence Modeling with Graph Convolutional Recurrent Networks.
Wu et al. 2019. A Comprehensive Survey on Graph Neural Networks

Exp2: Structured Sequence Prediction

❑Results:

- ○Our model achieves lowest error with much fewer parameters

❑The learnable parameters:

- ○LSTM-GNN: 84,890, GRU-GNN: 64,770, RNN-GNN: 24,530
- ○**NDCN: 901**

Table 3: **Regularly-sampled Extrapolation Prediction.** Our NDCN predicts different structured sequences accurately. Each result is the normalized ℓ_1 error with standard deviation (in percentage %) from 20 runs for 3 dynamics on 5 networks by each method.

		Grid	Random	Power Law	Small World	Community
Heat Diffusion	LSTM-GNN	12.8 ± 2.1	21.6 ± 7.7	12.4 ± 5.1	11.6 ± 2.2	13.5 ± 4.2
	GRU-GNN	11.2 ± 2.2	9.1 ± 2.3	8.8 ± 1.3	9.3 ± 1.7	7.9 ± 0.8
	RNN-GNN	18.8 ± 5.9	25.0 ± 5.6	18.9 ± 6.5	21.8 ± 3.8	16.1 ± 0.0
	NDCN	$\mathbf{4.3 \pm 0.7}$	$\mathbf{4.7 \pm 1.7}$	$\mathbf{5.4 \pm 0.4}$	$\mathbf{2.7 \pm 0.4}$	$\mathbf{5.3 \pm 0.7}$
Mutualistic Interaction	LSTM-GNN	51.4 ± 3.3	24.2 ± 24.2	27.0 ± 7.1	58.2 ± 2.4	25.0 ± 22.3
	GRU-GNN	49.8 ± 4.1	$\mathbf{1.0 \pm 3.6}$	12.2 ± 0.8	51.1 ± 4.7	$\mathbf{3.7 \pm 4.0}$
	RNN-GNN	56.6 ± 0.1	8.4 ± 11.3	12.0 ± 0.4	57.4 ± 1.9	8.2 ± 6.4
	NDCN	$\mathbf{29.8 \pm 1.6}$	$\mathbf{4.7 \pm 1.1}$	$\mathbf{11.2 \pm 5.0}$	$\mathbf{15.9 \pm 2.2}$	$\mathbf{3.8 \pm 0.9}$
Gene Regulation	LSTM-GNN	27.7 ± 3.2	67.3 ± 14.2	38.8 ± 12.7	13.1 ± 2.0	53.1 ± 16.4
	GRU-GNN	24.2 ± 2.8	50.9 ± 6.4	35.1 ± 15.1	11.1 ± 1.8	46.2 ± 7.6
	RNN-GNN	28.0 ± 6.8	56.5 ± 5.7	42.0 ± 12.8	14.0 ± 5.3	46.5 ± 3.5
	NDCN	$\mathbf{18.6 \pm 9.9}$	$\mathbf{2.4 \pm 0.9}$	$\mathbf{4.1 \pm 1.4}$	$\mathbf{5.5 \pm 0.8}$	$\mathbf{2.9 \pm 0.5}$

Exp3. Node Semi-supervised Classification

- **Problem: One-snapshot case**
  - Input: G, X, a **part of** labels $Y(X)$
  - Output: To Complete $Y(X)$
- **Datasets:**
  -

Table 11: Statistics for three real-world citation network datasets. N, E, D, C represent number of nodes, edges, features, classes respectively.

Dataset	N	E	D	C	Train/Valid/Test
Cora	2,708	5,429	1,433	7	140/500/1,000
Citeseer	3,327	4,732	3,703	6	120/500/1,000
Pubmed	19,717	44,338	500	3	60/500/1,000

Exp3. Node Semi-supervised Classification

- **Baselines**
  - Graph Convolution Network (GCN)
  - Attention-based GNN (AGNN)
  - Graph Attention Networks (GAT)

$$Z = f(X, A) = \mathrm{softmax}\Big(\hat{A}\ \mathrm{ReLU}\Big(\hat{A}XW^{(0)}\Big)\,W^{(1)}\Big)$$

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^{k}\mathbf{W}^{k}\vec{h}_j\right)$$

Kipf et al. 2016. Semi-Supervised Classification with Graph Convolutional Networks
Velickovic et al. 2017. Graph Attention Networks

Exp3. Node Semi-supervised Classification

Interpretation of our model

- Input: G, $[X, Mask \odot Y]$, features and some node labels
- Output: To complete Y
- Model: A graph dynamics to spread features and labels on graph over continuous time T
  - $\frac{d[X,Y]}{dt} = \mathrm{f}(\mathrm{G}, \mathrm{X}, \mathrm{Y}, \mathrm{W}, \mathrm{t})$

$$\underset{W_e, b_e, W_d, b_d}{\operatorname{argmin}} \quad \mathcal{L} = \int_0^T \mathcal{R}(t)\, dt - \sum_{i=1}^{n} \sum_{k=1}^{c} \hat{Y}_{i,k}(T) \log Y_{i,k}(T)$$

$$\text{subject to} \quad X_h(0) = \tanh\left(X(0) W_e + b_e\right)$$

$$\frac{dX_h(t)}{dt} = \mathrm{ReLU}\left(\Phi X_h(t)\right)$$

$$Y(T) = \mathrm{softmax}(X_h(T) W_d + b_d)$$

Exp3. Node Semi-supervised Classification

- **Metrics**
  - Accuracy over 100 runs
- **Results**
  - Very competitive results
  - Continuous-time dynamics on graphs
  - Best results at time T=1.2
    - Continuous depth/time

Table 4: Test mean accuracy with standard deviation in percentage (%) over 100 runs. Our NDCN model gives very competitive results compared with many GNN models.

Model	Cora	Citeseer	Pubmed
GCN	81.5	70.3	79.0
AGNN	83.1 ± 0.1	71.7 ± 0.1	$\mathbf{79.9 \pm 0.1}$
GAT	83.0 ± 0.7	72.5 ± 0.7	79.0 ± 0.3
NDCN	$\mathbf{83.3 \pm 0.6}$	$\mathbf{73.1 \pm 0.6}$	$\mathbf{79.8 \pm 0.4}$

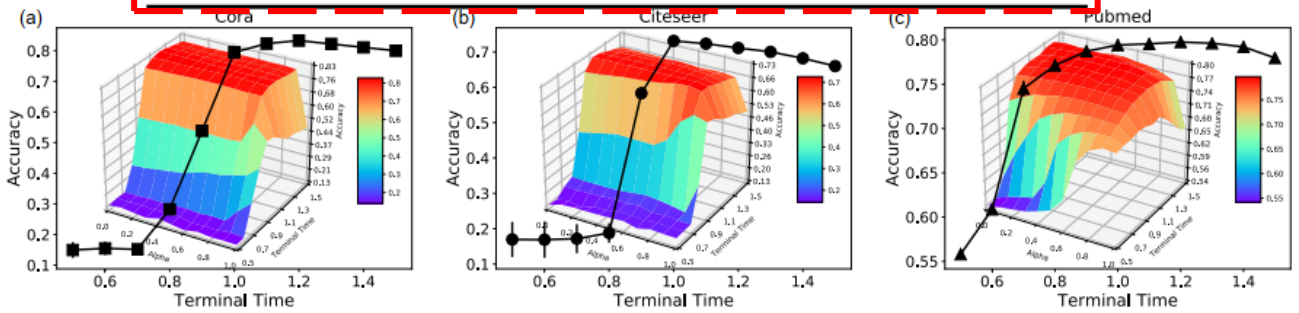


Figure 5: *Our NDCN model captures continuous-time dynamics.* Mean classification accuracy of 100 runs over terminal time when given a specific α. Insets are the accuracy over the two-dimensional space of terminal time and α

Summary

- **A novel NDCN model, which is a**
  - Continuous-depth GNN
  - Continuous-time (temporal) GNN
  - Graph Neural ODE
- **Our NDCN, a unified framework to solve**
  - Continuous-time network dynamics prediction:
  - Structured sequence prediction
  - Node (semi-supervised) regression/classification at one-snapshot

  good performance with less parameters.
- **ODE-GNN model**
  - A potential data-driven method to model structure and dynamics of complex systems in a unified framework

Thank You!

Weill Cornell Medicine

Neural Dynamics on Complex Networks

Chengxi Zang and Fei Wang
Weill Cornell Medicine
www.calvinzang.com