Differential Deep Learning on Graphs and its Applications

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

This Tutorial

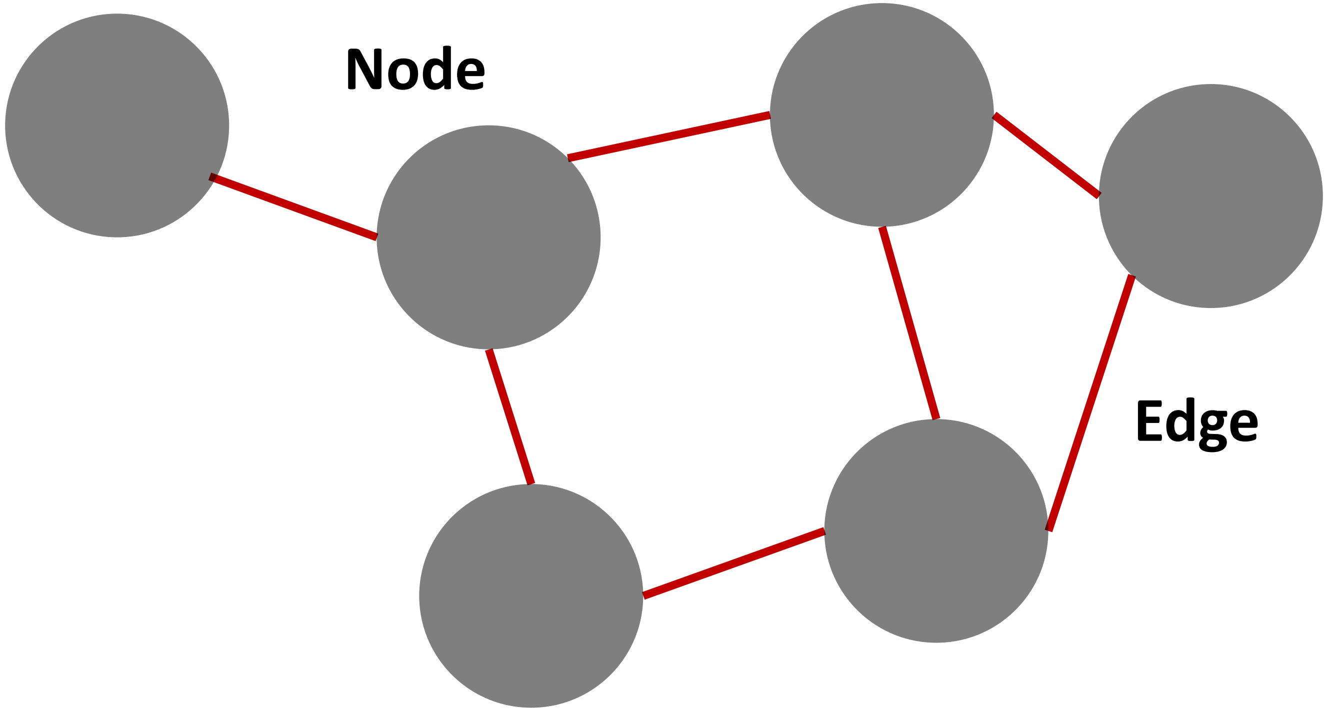
- www.calvinzang.com/DDLG_AAAI_2020.html
- AAAI-2020
- **Friday, February 7, 2020, 2:00 PM -6:00 PM**
- **Sutton North, Hilton New York Midtown, NYC**

Differential Deep Learning on Graphs

- Graphs and Differential Equations are general tools to describe structures and dynamics of complex systems

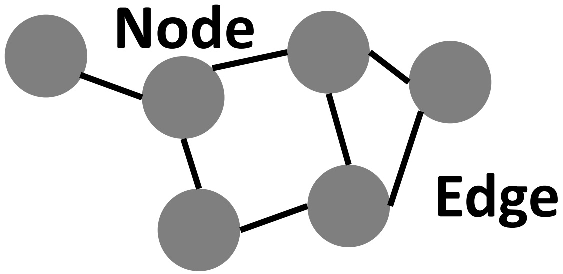
Graph

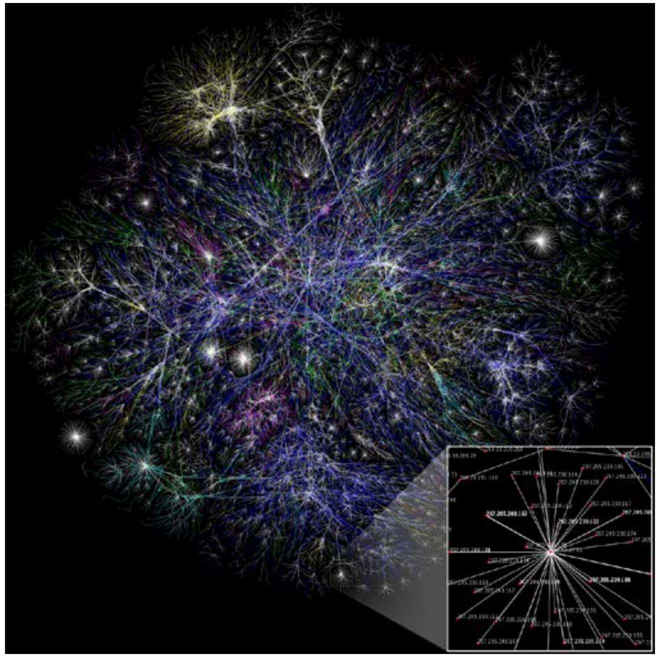
- Linked objects: nodes + edges
- Network



Graph

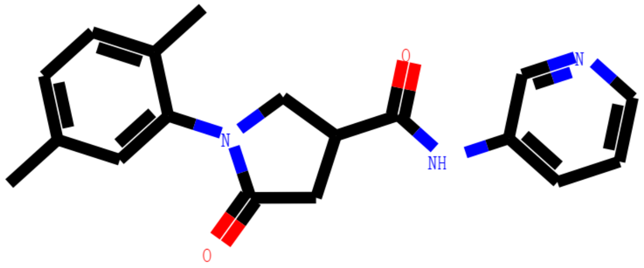
- **Linked objects: nodes + edges**
  - E.g.: Internet, social networks, molecules, etc.





Node: IPs,   Edge: Hyperlinks

Node: users,   Edge: Social links



Node: Atoms,   Edge: Bonds

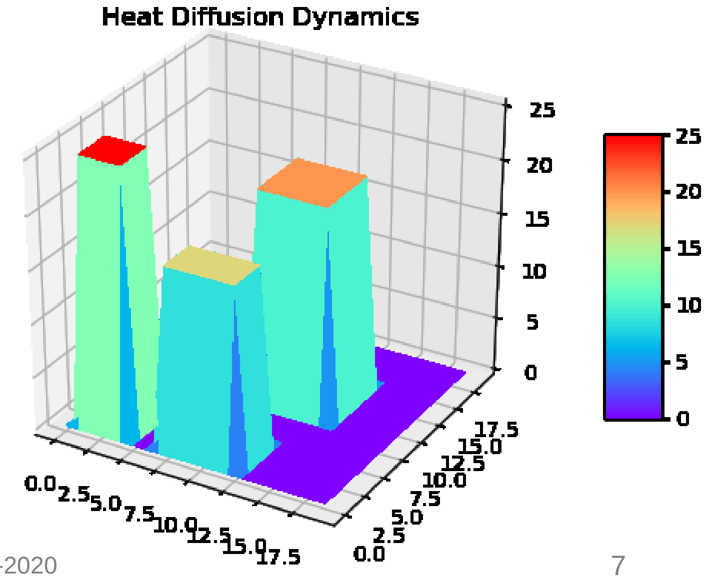
Internet Images from
https://en.wikipedia.org/wiki/Network_theory

Differential Equations

- **Equations which relates functions (physical quantities) and their derivatives (rates of change), e.g.**
  - e.g.Population growth
    - Exponential growth: $\frac{dx}{dt} = ax \rightarrow x(\mathrm{t}) = Ce^{at}$ solution by integrating
    - Power-law growth: $\frac{dx}{dt} = a\frac{x}{t} \rightarrow x(\mathrm{t}) = Ct^{a}$ solution by integrating

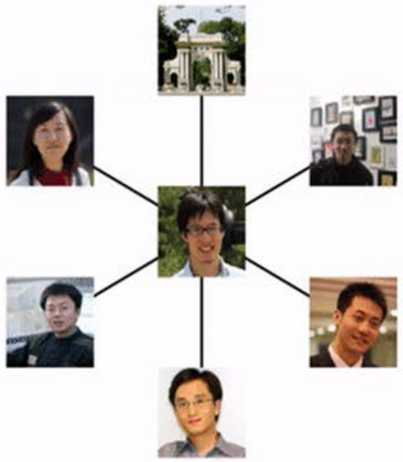
Differential Equations

- **Equations which relates functions (physical quantities) and their derivatives (rates of change), e.g.**
  - e.g.Population growth
    - Exponential growth: $\frac{dx}{dt} = ax \rightarrow x(\mathrm{t}) = Ce^{at}$ solution by integrating
    - Power-law growth: $\frac{dx}{dt} = a\frac{x}{t} \rightarrow x(\mathrm{t}) = Ct^{a}$ solution by integrating
- **Differential Equation System**
  - A system of differential equations
  - Newton's law of cooling: $\frac{dX}{dt} = -kLX$
    - Laplacian matrix: L=D-A, A: adjacency matrix



Why Graphs and Differential Equations?

- Question (Social network analysis): How does information spread in social networks? How does information flow form complex structural patterns?



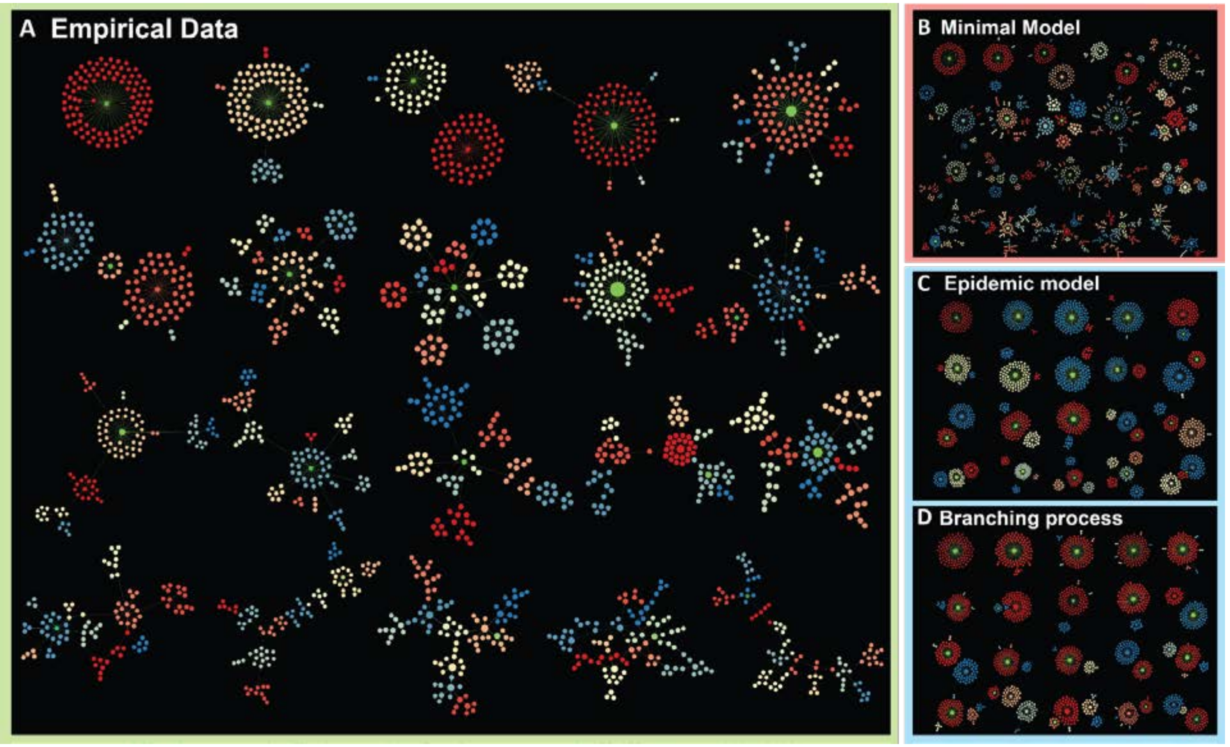
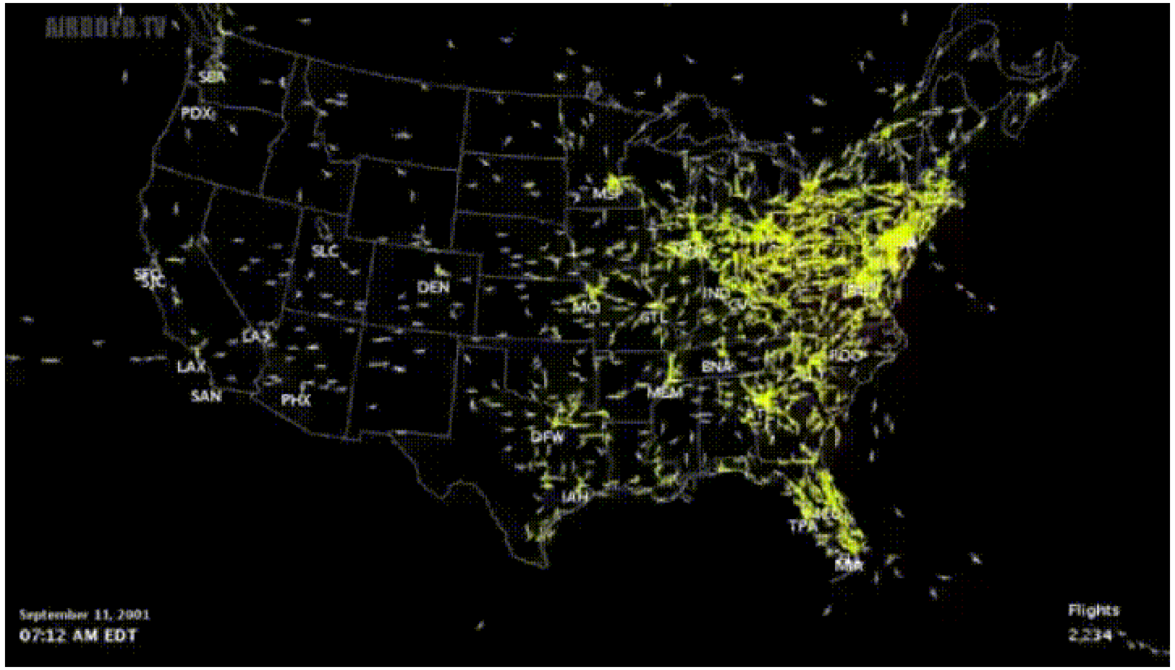
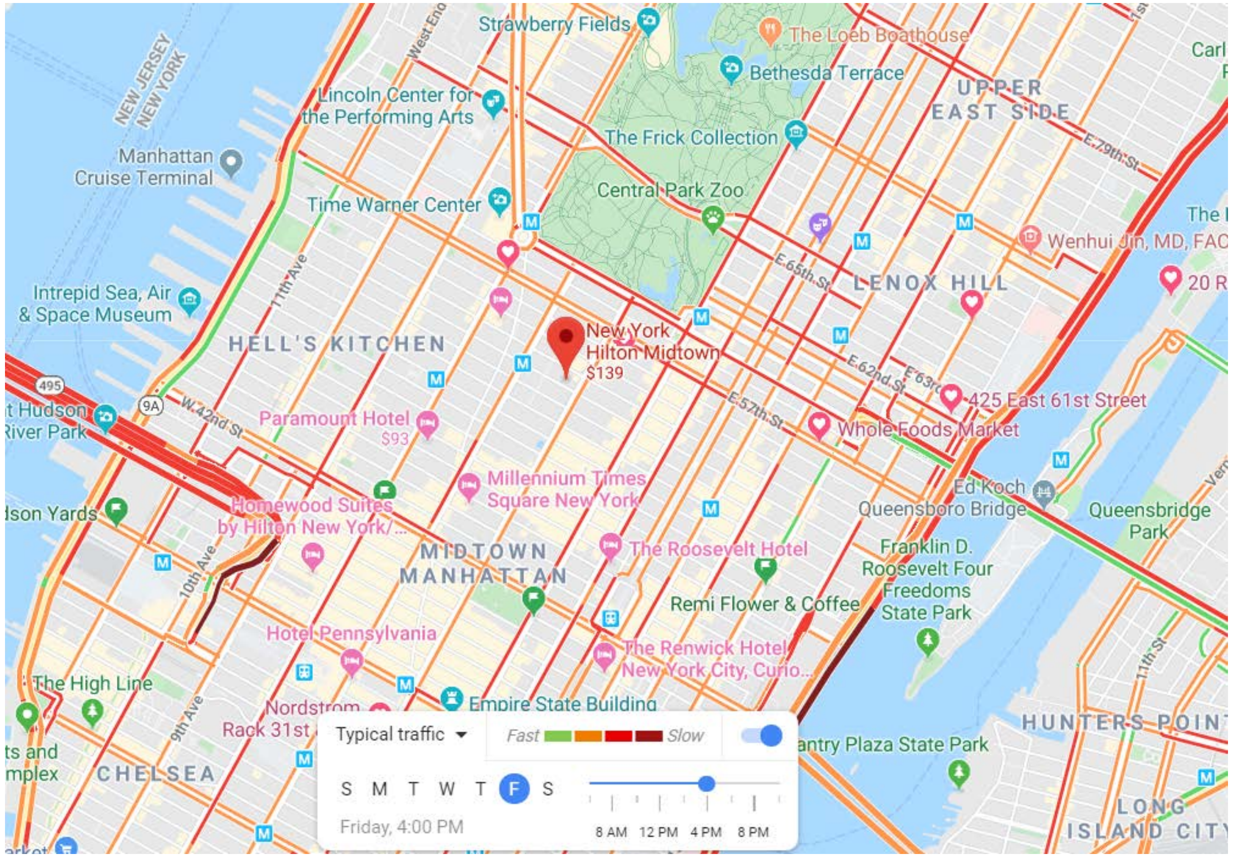


Image from: **Zang** et al. 2019. [Uncovering Pattern Formation of Information Flow.](#) *KDD*.

Why Graphs and Differential Equations?

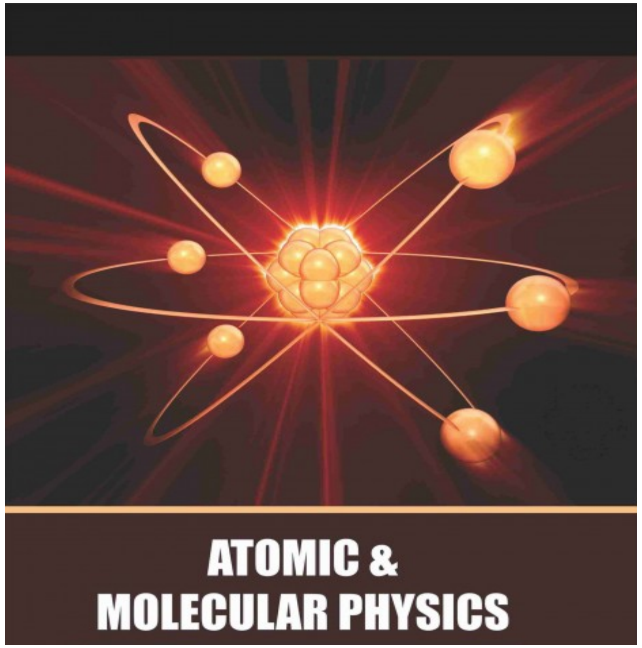
- Question (Urban computing): Can we predict and control traffic flows on road networks?

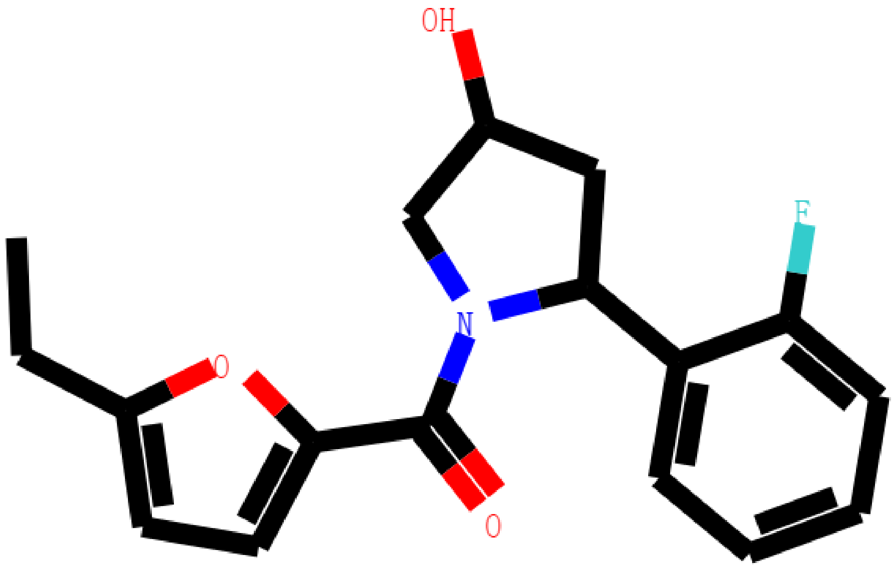




Why Graphs and Differential Equations?

- Question (Drug discovery): Can we predict molecular properties? Can we design novel drug molecule with optimized properties?





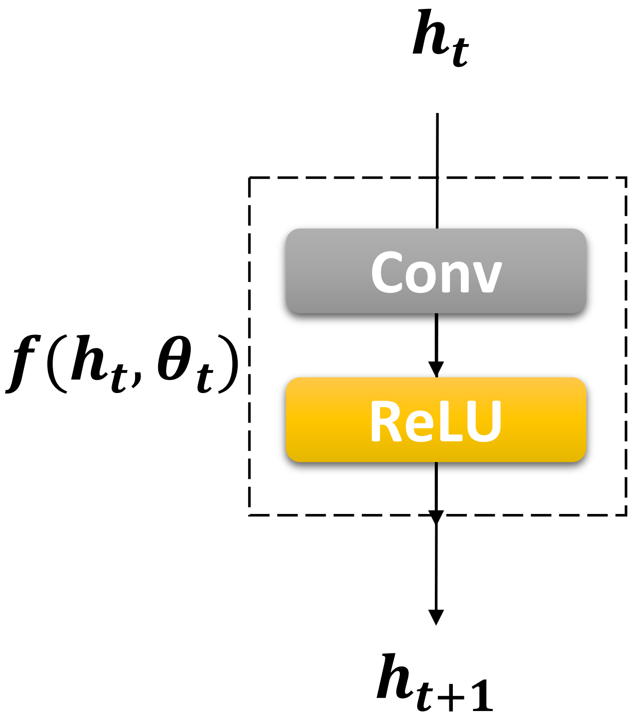
Differential Deep Learning on Graphs

- **Graphs and Differential Equations are general tools to describe structures and dynamics of complex systems**
- **Inspired by the <u>differential equations</u>, we can design and analyze <u>deep models</u>**

Residual Net.→ Differential Equations

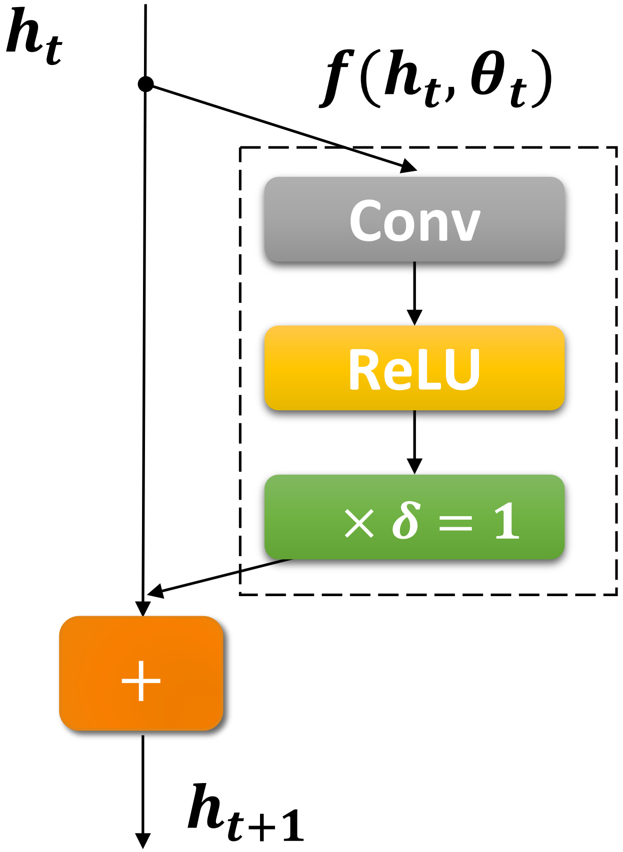
Plain Convolution Net.

$$h_{t+1} = f(h_t, \theta_t)$$



Residual Net.

$$h_{t+1} = h_t + f(h_t, \theta_t)$$

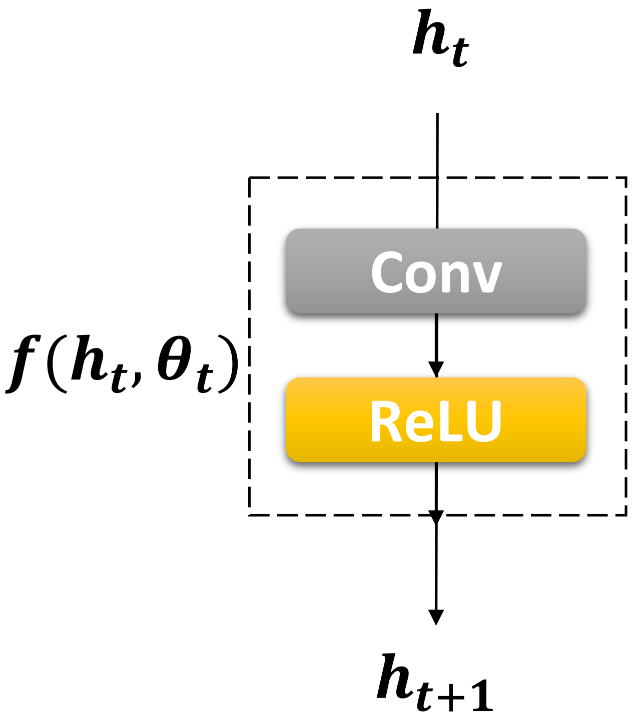


He et al. 2016. [Identity mappings in deep residual networks](#) *ECCV*.

Residual Net.→ Differential Equations

Plain Convolution Net.

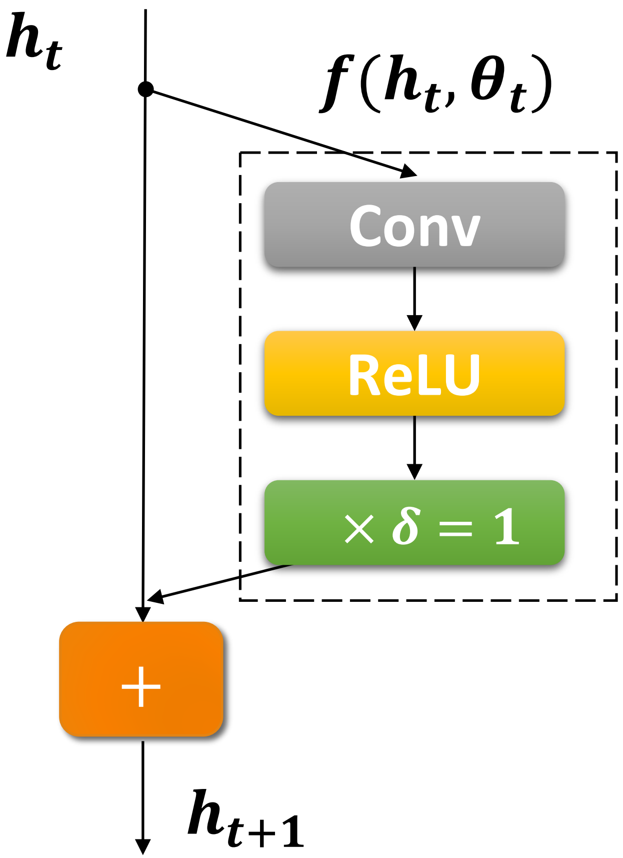
$$h_{t+1} = f(h_t, \theta_t)$$



Residual Net.

$$h_{t+1} = h_t + f(h_t, \theta_t)$$

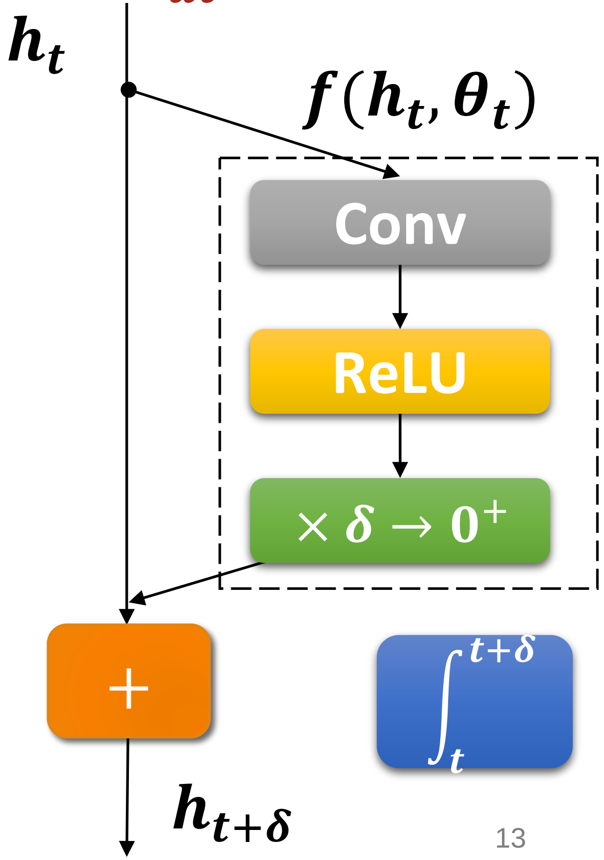
$$\frac{h_{t+1} - h_t}{\delta} = f(h_t, \theta_t), \delta = 1$$



Differential Equation Net.

$$h_{t+\delta} = h_t + \int_t^{t+\delta} f(h, \theta, \tau)\, d\tau$$

$$\frac{dh_t}{dt} = f(h_t, \theta_t)$$



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

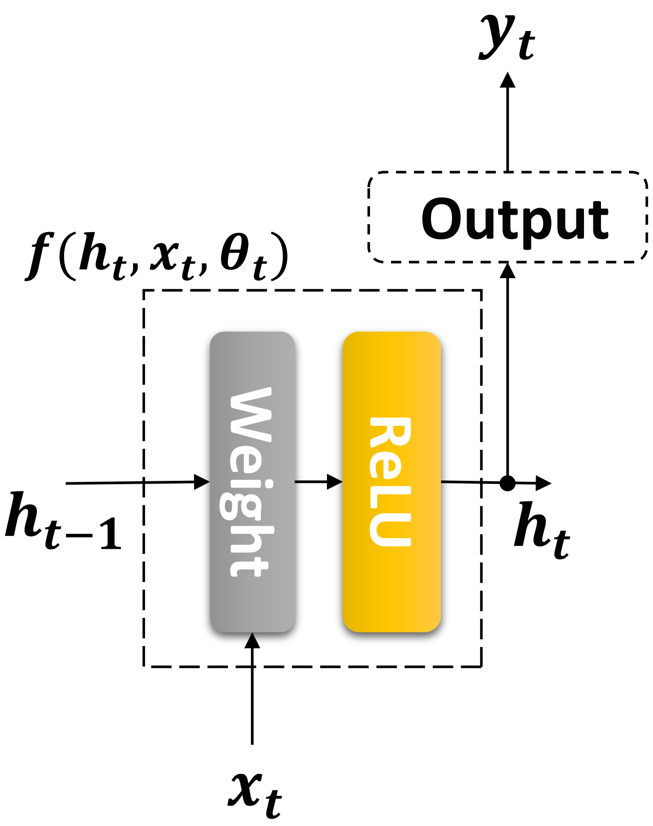
RNN → Differential Equations

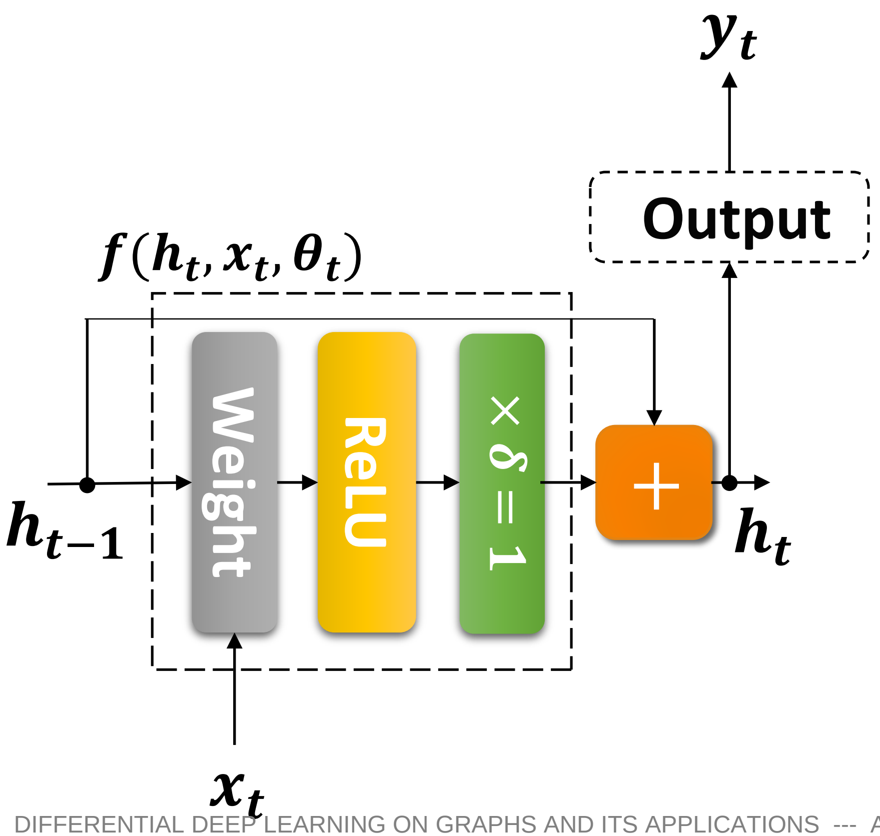
RNN

$$h_t = f(h_{t-1}, x_t, \theta_t)$$
$$y_t = o(h_t, w_t)$$

Residual RNN

$$h_t = h_{t-1} + f(h_{t-1}, x_t, \theta_t)$$
$$y_t = o(h_t, w_t)$$

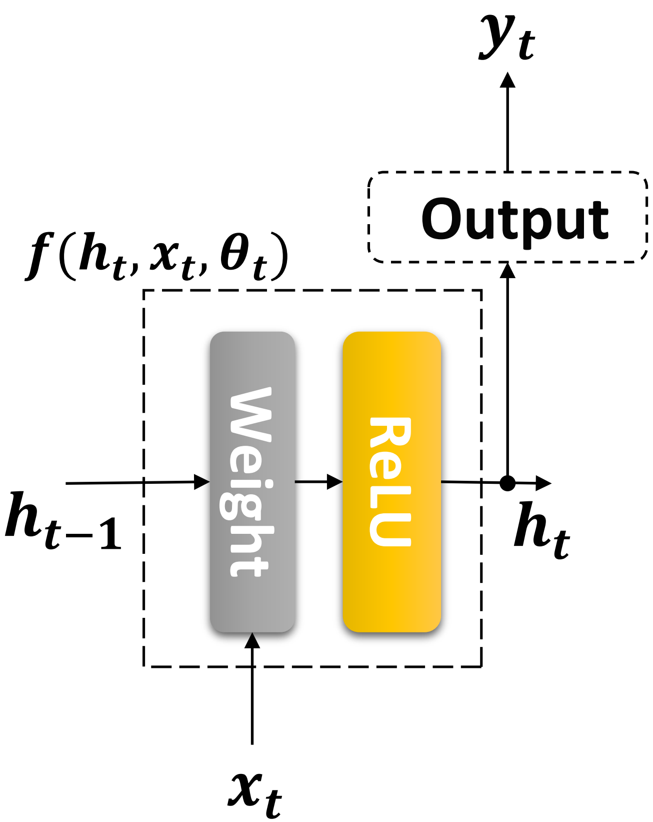




RNN → Differential Equations

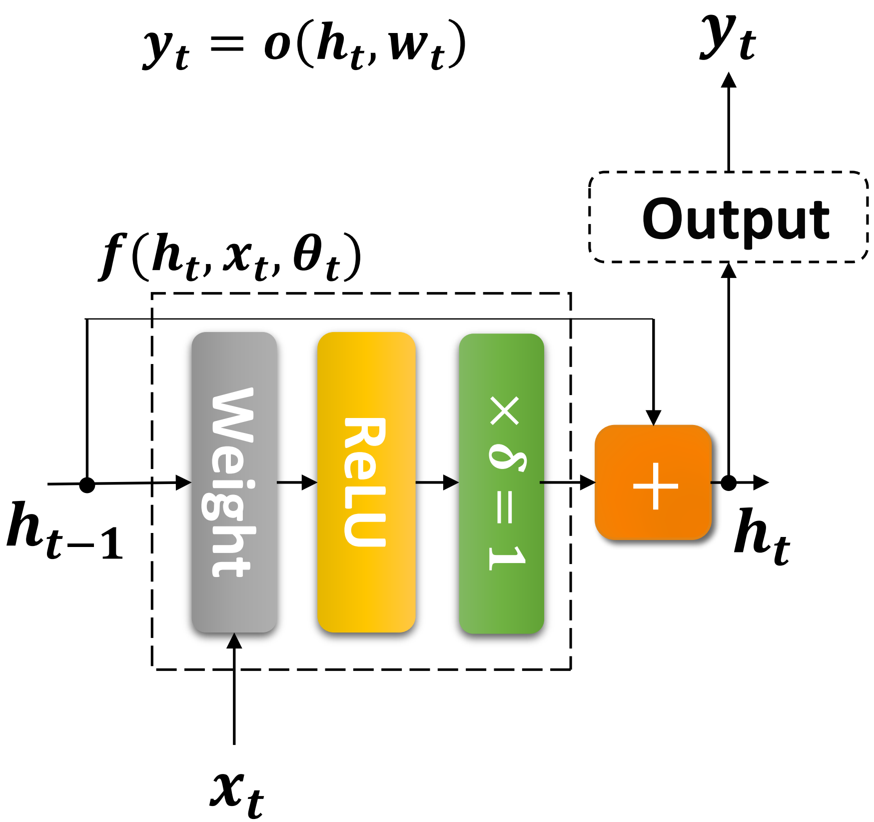
RNN

$$h_t = f(h_{t-1}, x_t, \theta_t)$$
$$y_t = o(h_t, w_t)$$



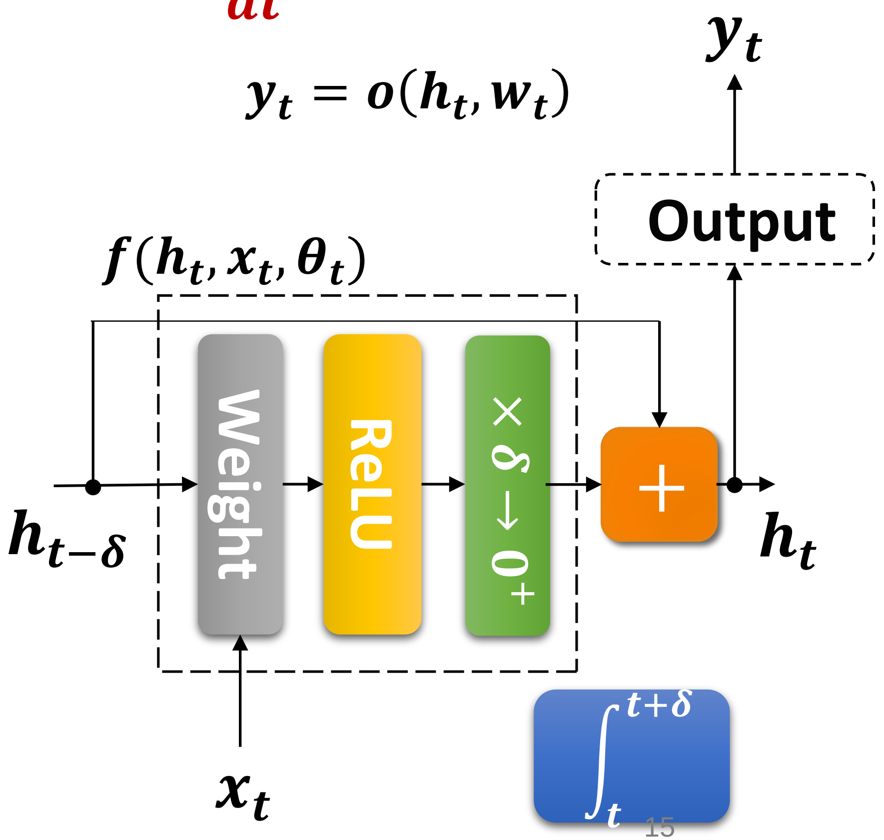
Residual RNN

$$h_t = h_{t-1} + f(h_{t-1}, x_t, \theta_t)$$
$$\frac{h_t - h_{t-1}}{\delta} = f(h_{t-1}, x_t, \theta_t), \delta=1$$
$$y_t = o(h_t, w_t)$$



Differential Equation RNN

$$h_t = h_{t-\delta} + \int_{t-\delta}^{t} f(h, x, \theta, \tau)\, d\tau$$
$$\frac{dh_t}{dt} = f(h_t, x_t, \theta_t)$$
$$y_t = o(h_t, w_t)$$

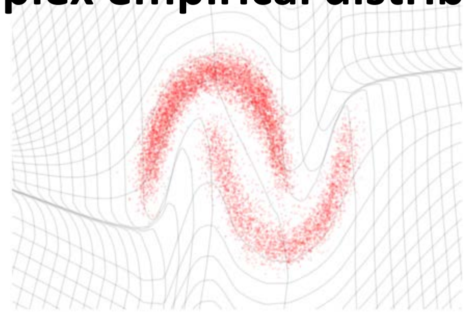


Pearlmutter1995. Gradient calculations for dynamic recurrent neural networks: A survey. *TNN*.

Normalizing flow→Differential Equations

- **An invertible generative model**
  - Goal: $X \sim \boldsymbol{P}(X)$
- **Inference**: $Z = f_\theta(X)$
  - From complex to simple
- **Generation:** $X = f_\theta^{-1}(Z)$
  - Generate complex by invertible mapping
- $\log \boldsymbol{P}(X) = \log \boldsymbol{P}(Z) + \log|\det(\frac{\partial f_\theta}{\partial Z})|$
  - Change of variable formula
  - Exact maximum likelihood training

$\boldsymbol{P(X)}$:
Complex empirical distribution



Inference ↓ ↑ Generation

$\boldsymbol{P(Z)}$:
Simple latent distribution

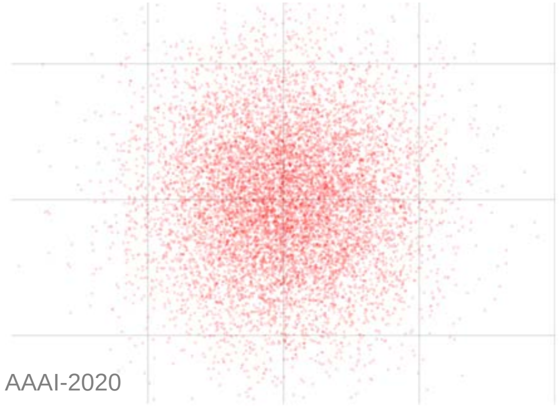


Image from: **Dinh** et al. 2017. [Density Estimation using Real NVP.](#) *ICLR*.

Normalizing flow → Differential Equations

❑ Flow

- Inference: $Z_{t+1} = f_\theta(Z_t)$, Generation: $Z_t = f_\theta^{-1}(Z_{t+1})$
- $\log \boldsymbol{P}(Z_t) = \log \boldsymbol{P}(Z_{t+1}) + \log|\det(\frac{\partial f_\theta}{\partial Z})|$

❑ Residual Flow

- Inference: $Z_{t+1} = Z_t + \delta f_\theta(Z_t)$, Generation: $Z_t = (I + \delta f_\theta)^{-1}(Z_{t+1})$, δ=1
- $\log P_M(Z_t) = \log P_Z(Z_{t+1}) + \log|\det(\frac{\partial(I+\delta f_\theta)}{\partial Z})|$

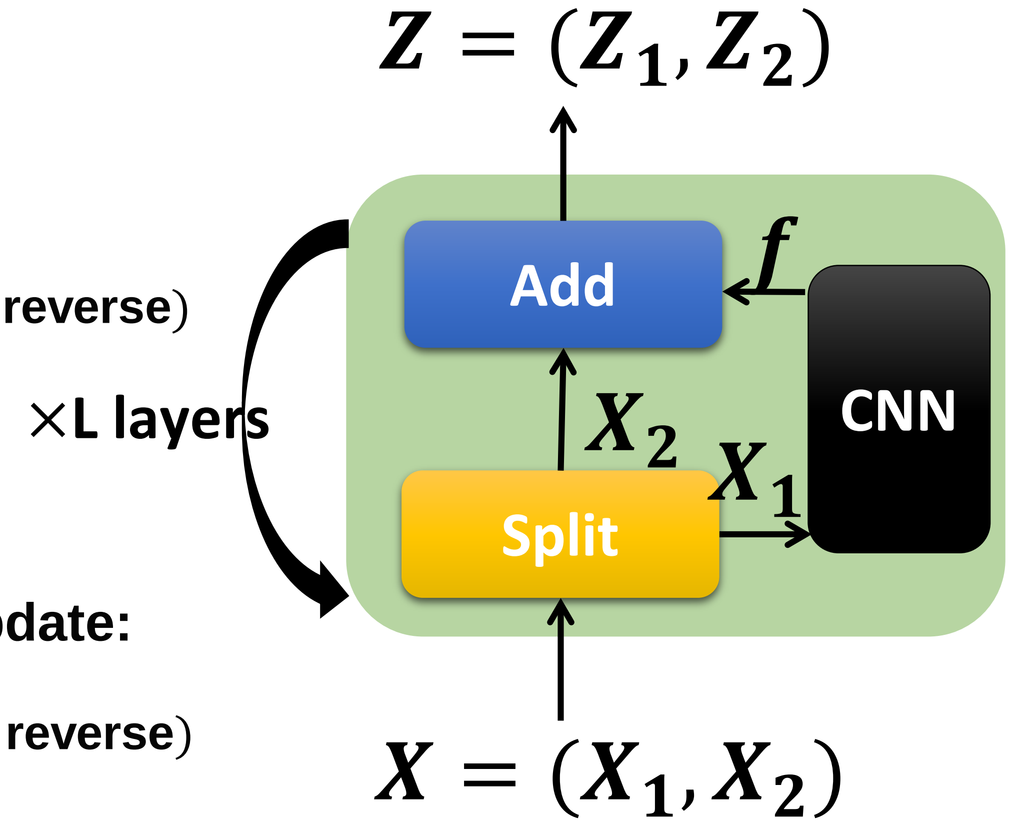
❑ Differential Eq. Flow

- Inference: $\frac{dZ(t)}{dt} = f_\theta(Z,t)$, Generation: $Z(0) = Z(t) - \int_0^t \boldsymbol{f_\theta(Z,\tau)\,d\tau}$
- $\boldsymbol{\frac{d\,logP(Z(t))}{dt} = -tr(\frac{df}{dZ(t)})}$

Chen et al. 2019. [Neural Ordinary Differential Equations.](#) *NeurIPS*.

An example: NICE v.s. Differential NICE

- **NICE or RealNVP**
  - **splitting dimensions + residual flow updated alternately**
- **Split:**
  - **X**$= (X_1, X_2)$
  - $\mathbf{Z} = (\mathbf{Z}_1, \mathbf{Z}_2)$
- **Add:**
  - $Z_1 = X_1$ (**save information for reverse**)
  - $Z_2 = X_2 + f_\theta(X_1)$ **(Residual)**
  - Reverse mapping:
    - $X_1 = Z_1$
    - $X_2 = Z_2 - f_\theta(Z_1)$
- **Next layer by alternating update:**
  - $Z_1 = X_1 + f_\theta(X_2)$ **(Residual)**
  - $Z_2 = X_2$ (**save information for reverse**)
- **…**



Hamiltonian Systems

$$\begin{bmatrix} \frac{dZ_1}{dt} \\ \frac{dZ_2}{dt} \end{bmatrix} = \begin{bmatrix} f_\theta(Z_2) \\ f_\theta(Z_1) \end{bmatrix}$$

Dinh et al. 2014. [Nice: Non-linear independent components estimation]
Dinh et al. 2017. [Density Estimation using Real NVP.] *ICLR*.

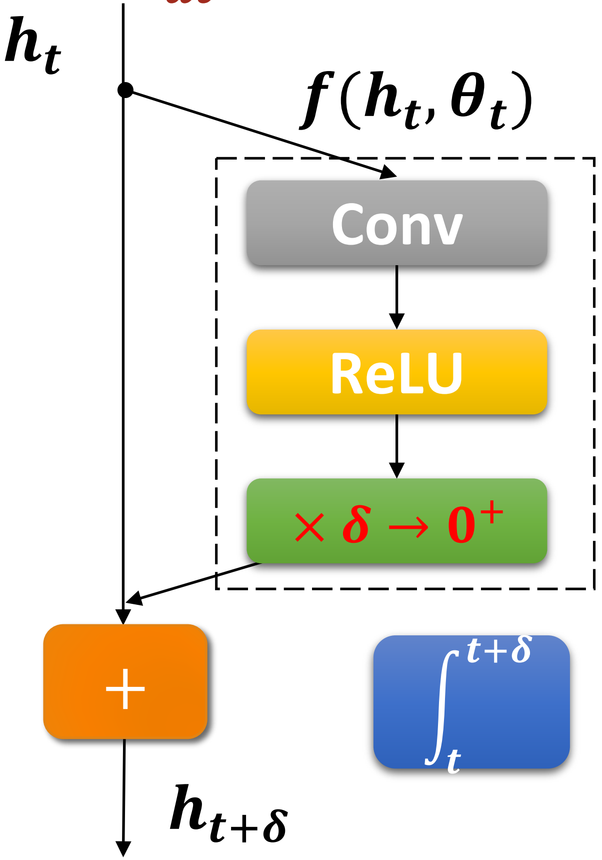
Chen et al. 2019. [Neural Ordinary Differential Equations.] *NeurIPS*.

DEs → DNNS by Numerical Methods

Differential Equation Net.

$$h_{t+\delta} = h_t + \int_t^{t+\delta} f(h,\theta,\tau)\,d\tau$$

$$\frac{dh_t}{dt} = f(h_t,\theta_t)$$

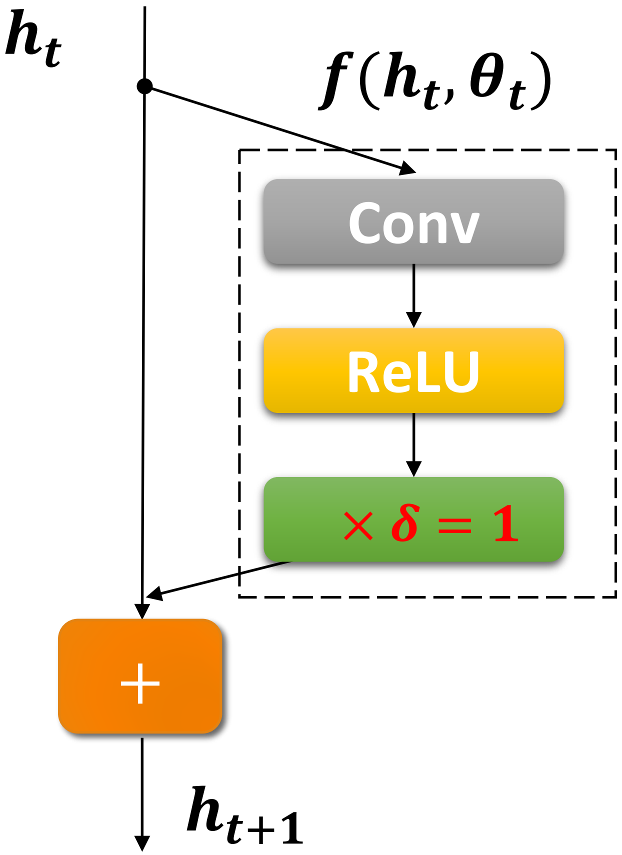


Residual Net.

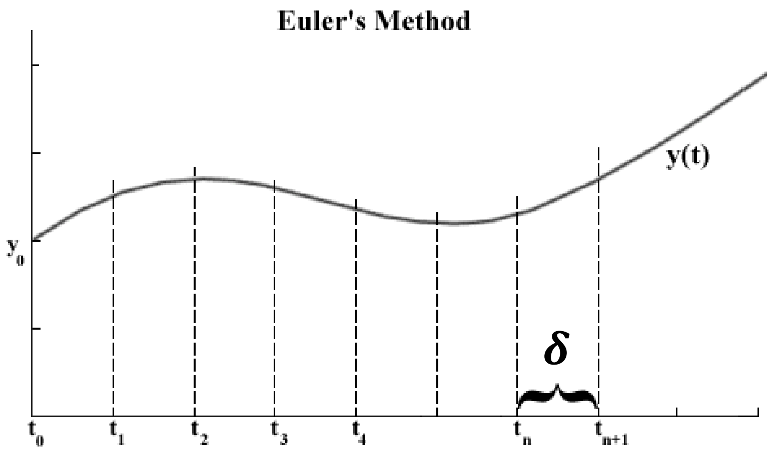
$$h_{t+1} = h_t + f(h_t,\theta_t)$$

$$h_{t+1} = h_t + \delta f(h_t,\theta_t), \delta = 1$$

output = input + step * rate of change



Numerical Methods:
Integrating DEs by discretization



$$h_{n+1} = h_0 + \sum_{t=1}^{n} \delta f(h_t,\theta_t)$$

Gif image from
https://jmahaffy.sdsu.edu/courses/f00/math122/lectures/num_method_diff_equations/nummethod_diffeq.html

Why Such Connections

- **Deep Learning → Differential Equations**
  - Analysis
    - Math analysis tools
    - Concepts in dynamic system and control: stability, robustness, complexity, resilience, etc.
  - Modeling Continuous-time process
    - Physical meaning. The laws of nature are expressed as differential equations.
- **Differential Equations → Deep Learning**

Why Such Connections

- **Deep Learning → Differential Equations**
  - Analysis
    - Math analysis tools
    - Concepts in dynamic system and control: stability, robustness, complexity, resilience, etc.
  - Modeling Continuous-time process
    - Physical meaning. The laws of nature are expressed as differential equations.
- **Differential Equations → Deep Learning**
  - Design
    - There are many dynamical systems and differential equations.
    - Discretization of continuous time-varying dynamics → Deep Neural Networks
    - DNNs implemented by modern auto-differentiation softwares are more flexible, expressive and efficient
  - Generative models and Invertible structures

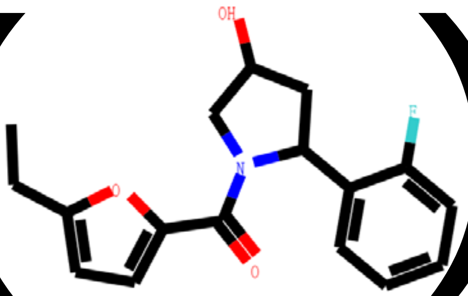
Differential Deep Learning on Graphs

- **Graphs and Differential Equations are general tools to describe structures and dynamics of complex systems**
- **Inspired by the Differential Equations, we can design and analyze Deep Models**
- **For <u>applications on graphs</u> (our focus), including:**
  - Molecular graph generation
  - Learning dynamics on graphs
  - Mechanism discovery

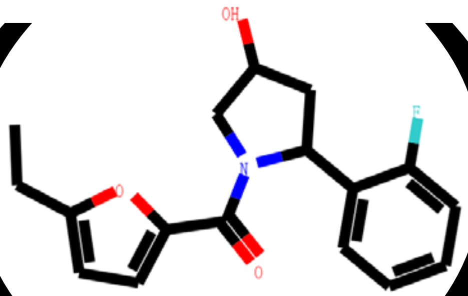
in a data-driven manner

Molecular Graph Generation

- **Goal:** To generate novel molecules with optimized properties
- **Graph Analysis tasks**
  - Graph generation: $\mathrm{G} \sim \mathrm{P}(G)$
  - Graph property prediction: $f(G)$
  - Graph optimization: $\mathrm{G} \rightarrow \mathrm{G}'$ and $\mathrm{maximizing}\ f(G') - f(G)$

P(

)?

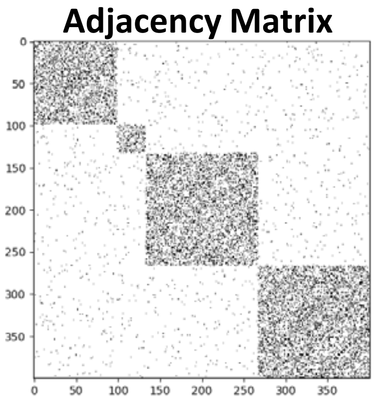
f(

)=?

Learning Dynamics on Graphs

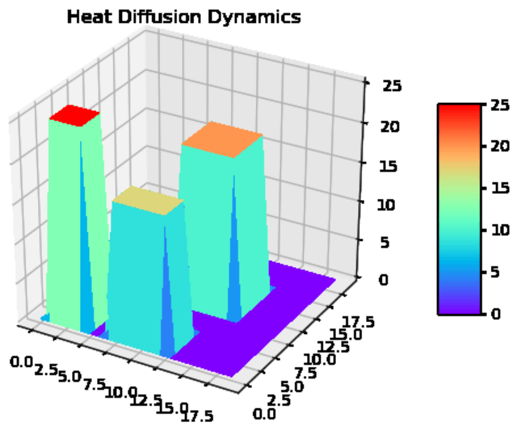
- **Goal: To predict temporal change or final states of complex systems**
- **Graph Analysis tasks**
  - Continuous-time network dynamics prediction $X(t)$
  - Structured sequence prediction $X[t+k]$
  - Node classification/regression $Y(X)$

Graph



+

Dynamic Process →

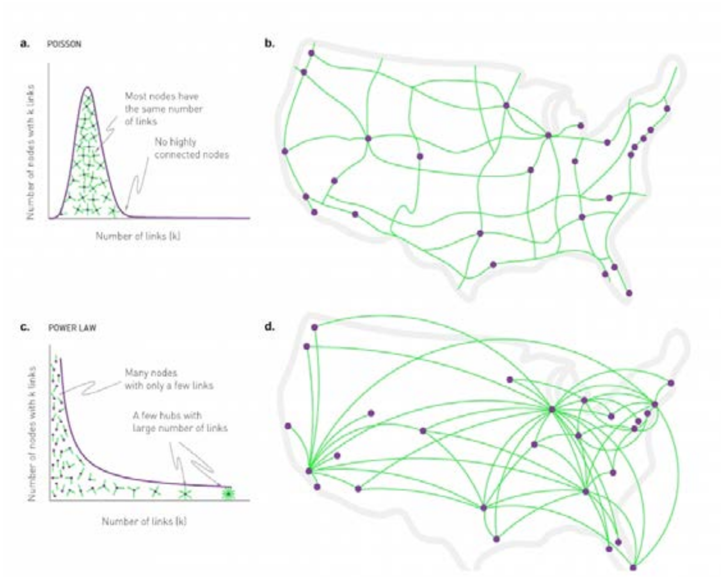


Dynamics of each nodes

?

Mechanism Discovery

- **Goals:** To find dynamical laws of complex systems
- **Graph Analysis tasks**
  - Density estimation vs. mechanism discovery
  - Data-driven discovery of differential equations



$$\xrightarrow{?}$$

$$\frac{dX}{dt}$$

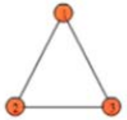
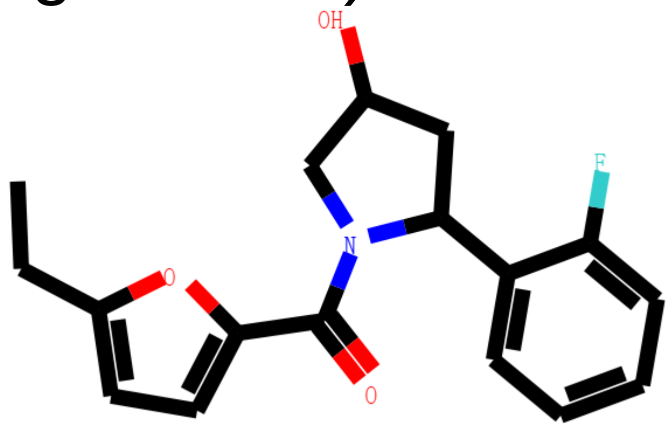


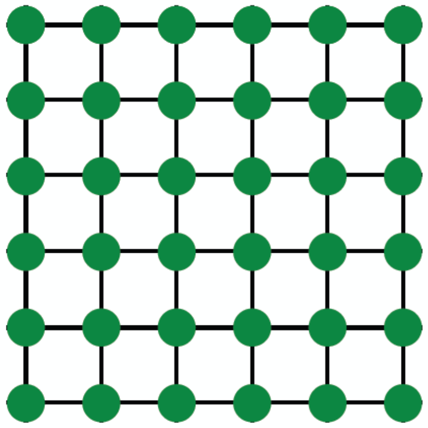
Image from http://networksciencebook.com/chapter/4#hubs

Why Is It Hard?

- **Complex combinatorial structures of graphs**
  - Due to complex combinations of node and edge sets
  - Nodes and edges can have multiple types
    - Node types: C, H, O, etc., Edge types: single, double, triple bond.
  - Complexity: the scale of drug-like graphs $\sim 10^{60}$
  - Deep models are majorly designed for regular grid structures (image or text)

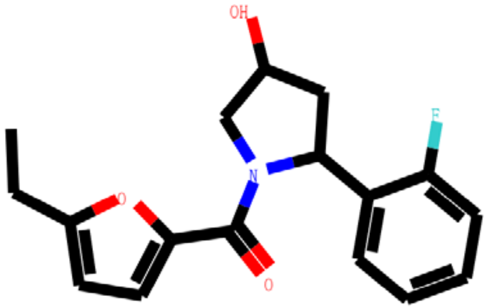


vs.



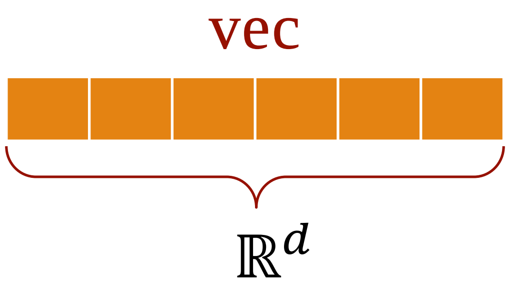
Why Is It Hard?

- **Encoding graph is hard, Decoding graph is much harder**
  - Encoding, embedding, inference with graph input

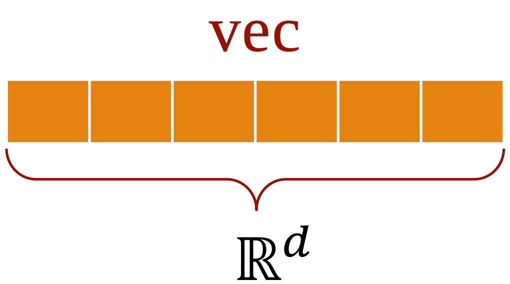


$$\xrightarrow{?}$$

$$f: G \rightarrow \mathbb{R}^d$$

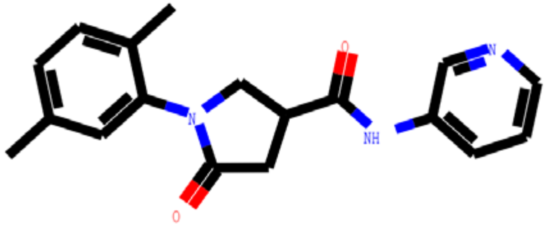


  - Decoding, generation with graph output
    - E.g. Chemically valid molecular graphs



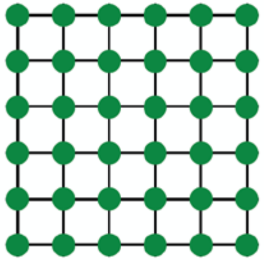
$$\xrightarrow{?}$$

$$f: \mathbb{R}^d \rightarrow G$$

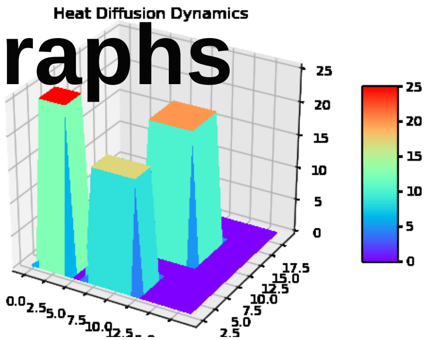


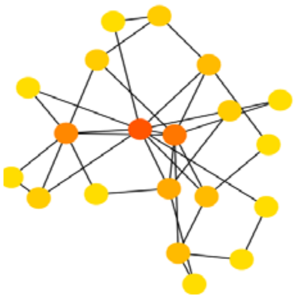
Why Is It Hard?

- Complex nonlinear dynamics on graphs

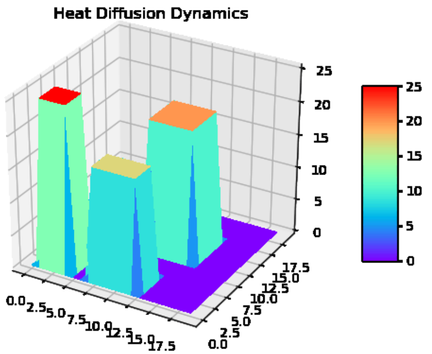


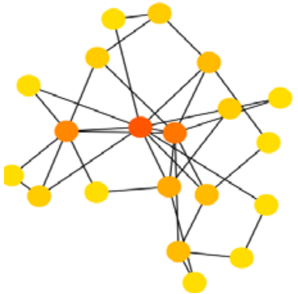
+ Linear Dynamics →



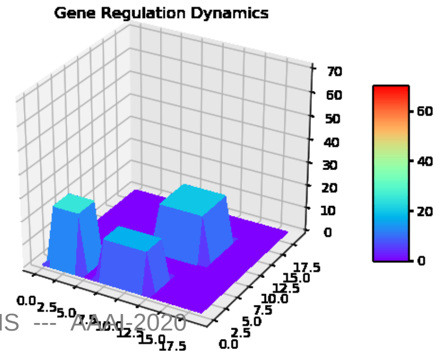


+ Linear Dynamics →





+ Non-Linear Dynamics →



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

?

This Tutorial

- **Molecular Graph Generation:** to generate novel molecules with optimized properties
  - Graph generation
  - Graph property prediction
  - Graph optimization
- **Learning Dynamics on Graphs:** to predict temporal change or final states of complex systems
  - Continuous-time network dynamics prediction
  - Structured sequence prediction
  - Node classification/regression
- **Mechanism discovery:** to find dynamical laws of complex systems
  - Density Estimation vs. Mechanism Discovery
  - Data-driven discovery of differential equations

This Tutorial

- www.calvinzang.com/DDLG_AAAI_2020.html
- AAAI-2020
- **Friday, February 7, 2020, 2:00 PM -6:00 PM**
- **Sutton North, Hilton New York Midtown, NYC**

Thank You!

Weill Cornell Medicine

Differential Deep Learning on Graphs and its Applications

Chengxi Zang and Fei Wang

Weill Cornell Medicine

www.calvinzang.com