Port-Hamiltonian Architectural Bias for Long-Range Propagation in Deep Graph Networks

Simon Heilig[1*], Alessio Gravina[2*], Alessandro Trenta[2], Claudio Gallicchio[2], Davide Bacciu[2]

[1] Ruhr University Bochum, Germany
[2] University of Pisa, Italy
[*] Equal Contribution

ICLR
International Conference On Learning Representations

RUHR UNIVERSITÄT BOCHUM

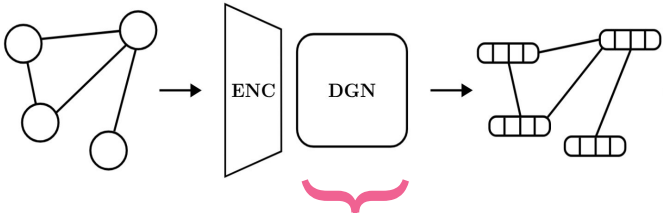
RUB

UNIVERSITÀ DI PISA

This Talk Covers

- 👉 how to exploit dynamical systems for neural graph embeddings
- 👉 a novel message passing dynamic offering
  - 💡 learnable information conservation and dissipation
  - ⚡ analytical vector field guarantees
  - 🚀 improved long-range capabilities of GNNs

Motivation

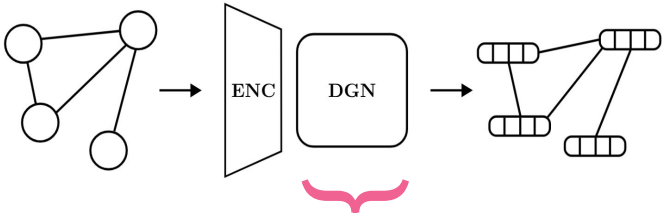


$$\mathbf{x}_u^{(l+1)} = \mathrm{UP}^{(l+1)}\left(\mathbf{x}_u^{(l)}, \mathrm{AGG}^{(l+1)}\left(\left\{\mathbf{x}_v^{(l)} : v \in \mathcal{N}_u\right\}\right)\right)$$

$$\mathbf{x}_u^{(l+1)} = \sigma\left(\mathbf{W}_{\text{self}}^{(l+1)}\mathbf{x}_u^{(l)} + \sum_{v\in\mathcal{N}_u}\mathbf{W}_{\text{msg}}^{(l+1)}\mathbf{x}_v^{(l)}\right)$$

👉 Repeat L times

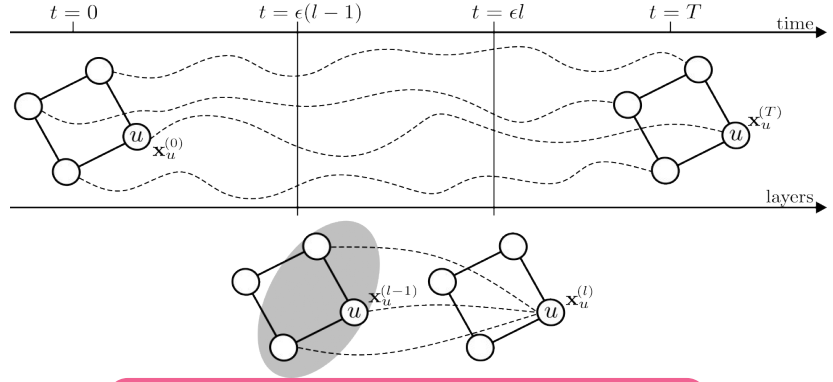
Motivation



$$\mathbf{x}_u^{(l+1)} = \mathrm{UP}^{(l+1)}\left(\mathbf{x}_u^{(l)}, \mathrm{AGG}^{(l+1)}\left(\left\{\mathbf{x}_v^{(l)} \,:\, v \in \mathcal{N}_u\right\}\right)\right)$$

$$\mathbf{x}_u^{(l+1)} = \sigma\left(\mathbf{W}_{\text{self}}^{(l+1)}\mathbf{x}_u^{(l)} + \sum_{v\in\mathcal{N}_u}\mathbf{W}_{\text{msg}}^{(l+1)}\mathbf{x}_v^{(l)}\right)$$

👉 Repeat L times

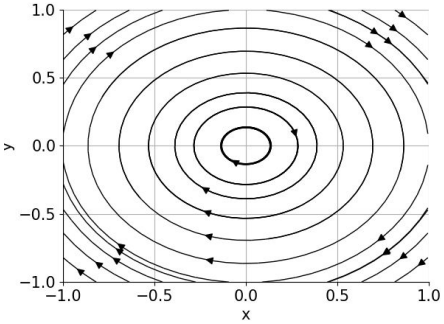


$$\frac{\mathrm{d}\mathbf{x}_u(t)}{\mathrm{d}t} = \sigma\left(\mathbf{W}_{\text{self}}(t)\mathbf{x}_u(t) + \Phi_{\mathcal{G}}(\{\mathbf{x}_v(t) \,:\, v \in \mathcal{N}_u\}, t) + \mathbf{b}(t)\right)$$

👉 Continuous Process

Di Giovanni et al. “Understanding convolution on graphs via energies”. Transactions on Machine Learning Research (2023)
Gravina, Bacciu, and Gallicchio. “Anti-Symmetric DGN: a stable architecture for Deep Graph Networks”. ICLR (2023)

Energy Conservation and Dissipation



Hamiltonian Dynamics

$$\frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = -\frac{\partial H}{\partial \mathbf{q}}(\mathbf{p}, \mathbf{q}), \quad \frac{\mathrm{d}\mathbf{q}}{\mathrm{d}t} = \frac{\partial H}{\partial \mathbf{p}}(\mathbf{p}, \mathbf{q})$$

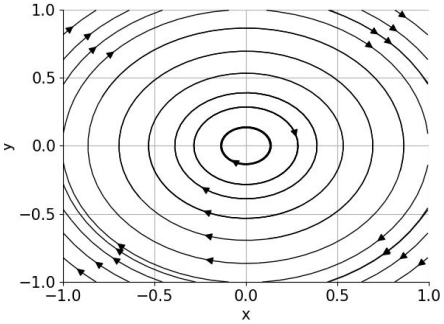
$$\Leftrightarrow$$

$$\frac{\mathrm{d}\mathbf{y}(t)}{\mathrm{d}t} = \mathcal{J}\nabla_{\mathbf{y}} H(\mathbf{y}(t)),$$

where $\mathbf{y} = (\mathbf{p}, \mathbf{q})^\top \in \mathbb{R}^{2d}$ and $\mathcal{J} = \begin{pmatrix} \mathbf{0} & -\mathbf{I}_d \\ \mathbf{I}_d & \mathbf{0} \end{pmatrix}$

Meyer, Kenneth R, and Daniel C Offin. Introduction to Hamiltonian Dynamical Systems and the N-Body Problem. 3rd ed. Springer, Cham, 2017.

Energy Conservation and Dissipation



Hamiltonian Dynamics

$$\frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = -\frac{\partial H}{\partial \mathbf{q}}(\mathbf{p},\mathbf{q}), \quad \frac{\mathrm{d}\mathbf{q}}{\mathrm{d}t} = \frac{\partial H}{\partial \mathbf{p}}(\mathbf{p},\mathbf{q})$$

$$\Leftrightarrow$$

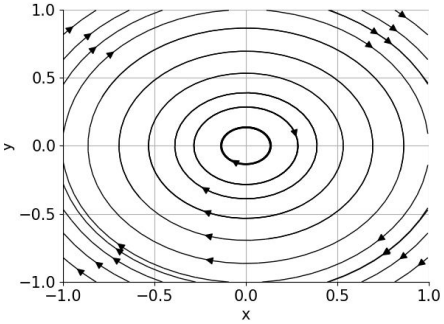
constant!

$$\frac{\mathrm{d}\mathbf{y}(t)}{\mathrm{d}t} = \mathcal{J}\nabla_{\mathbf{y}} H(\mathbf{y}(t)),$$

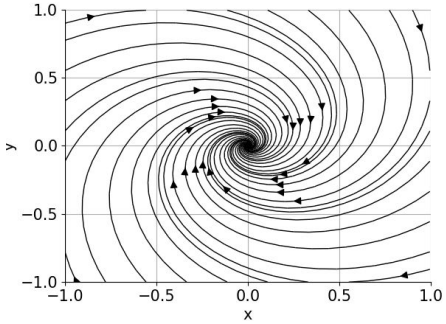
where $\mathbf{y} = (\mathbf{p},\mathbf{q})^\top \in \mathbb{R}^{2d}$ and $\mathcal{J} = \begin{pmatrix} \mathbf{0} & -\mathbf{I}_d \\ \mathbf{I}_d & \mathbf{0} \end{pmatrix}$

Meyer, Kenneth R, and Daniel C Offin. Introduction to Hamiltonian Dynamical Systems and the N-Body Problem. 3rd ed. Springer, Cham, 2017.

Energy Conservation and Dissipation



vs.



Hamiltonian Dynamics

$$\frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = -\frac{\partial H}{\partial \mathbf{q}}(\mathbf{p},\mathbf{q}),\ \frac{\mathrm{d}\mathbf{q}}{\mathrm{d}t} = \frac{\partial H}{\partial \mathbf{p}}(\mathbf{p},\mathbf{q})$$

$$\Leftrightarrow$$

constant!

$$\frac{\mathrm{d}\mathbf{y}(t)}{\mathrm{d}t} = \mathcal{J}\nabla_{\mathbf{y}} H(\mathbf{y}(t)),$$

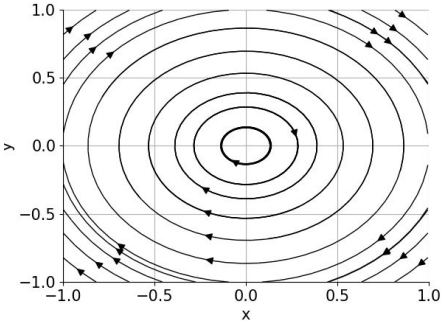
where $\mathbf{y} = (\mathbf{p},\mathbf{q})^\top \in \mathbb{R}^{2d}$ and $\mathcal{J} = \begin{pmatrix} \mathbf{0} & -\mathbf{I}_d \\ \mathbf{I}_d & \mathbf{0} \end{pmatrix}$

Port-Hamiltonian Dynamics

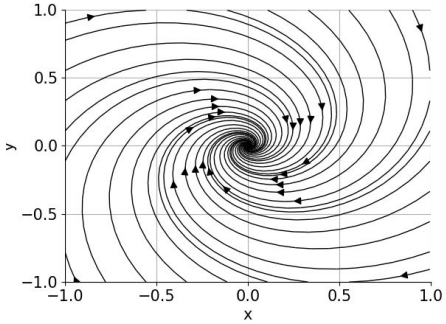
$$\frac{\mathrm{d}\mathbf{y}(t)}{\mathrm{d}t} = \left(\mathcal{J} - \begin{bmatrix} D(\mathbf{q}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}\right) \nabla_{\mathbf{y}} H(\mathbf{y}(t)) + \begin{bmatrix} F(\mathbf{q},t) \\ \mathbf{0} \end{bmatrix}$$

Desai, Shaan A., et al. “Port-Hamiltonian neural networks for learning explicit time-dependent dynamical systems”. Phys. Rev. E 104 (3 Sept. 2021): 034312.

Energy Conservation and Dissipation



vs.



Hamiltonian Dynamics

$$\frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = -\frac{\partial H}{\partial \mathbf{q}}(\mathbf{p},\mathbf{q}),\ \frac{\mathrm{d}\mathbf{q}}{\mathrm{d}t} = \frac{\partial H}{\partial \mathbf{p}}(\mathbf{p},\mathbf{q})$$

$$\Leftrightarrow$$

$$\frac{\mathrm{d}\mathbf{y}(t)}{\mathrm{d}t} = \mathcal{J}\nabla_{\mathbf{y}} H(\mathbf{y}(t)),$$

where $\mathbf{y} = (\mathbf{p},\mathbf{q})^\top \in \mathbb{R}^{2d}$ and $\mathcal{J} = \begin{pmatrix} \mathbf{0} & -\mathbf{I}_d \\ \mathbf{I}_d & \mathbf{0} \end{pmatrix}$

Port-Hamiltonian Dynamics

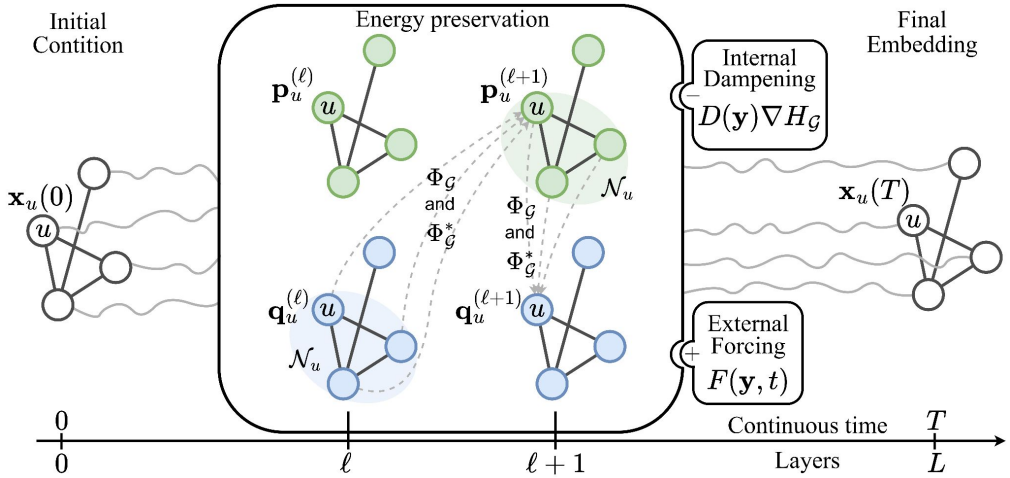
$$\frac{\mathrm{d}\mathbf{y}(t)}{\mathrm{d}t} = \left(\mathcal{J} - \begin{bmatrix} D(\mathbf{q}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}\right) \nabla_{\mathbf{y}} H(\mathbf{y}(t)) + \begin{bmatrix} F(\mathbf{q},t) \\ \mathbf{0} \end{bmatrix}$$

Desai, Shaan A., et al. “Port-Hamiltonian neural networks for learning explicit time-dependent dynamical systems”. Phys. Rev. E 104 (3 Sept. 2021): 034312.

Port-Hamiltonian DGN

The Continuous Dynamic

$$\frac{\mathrm{d}\mathbf{x}_u(t)}{\mathrm{d}t} = \mathcal{J}_u \nabla_{\mathbf{x}_u} H_{\mathcal{G}}(\mathbf{y}(t)) = \mathcal{J}_u \left[\mathbf{W}^\top \sigma(\mathbf{W}\mathbf{x}_u(t) + \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_u) + \mathbf{b}) + \sum_{v \in \mathcal{N}_u \cup \{u\}} \frac{\partial \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_v)}{\partial \mathbf{x}_u(t)}^\top \sigma(\mathbf{W}\mathbf{x}_v(t) + \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_v) + \mathbf{b}) \right]$$



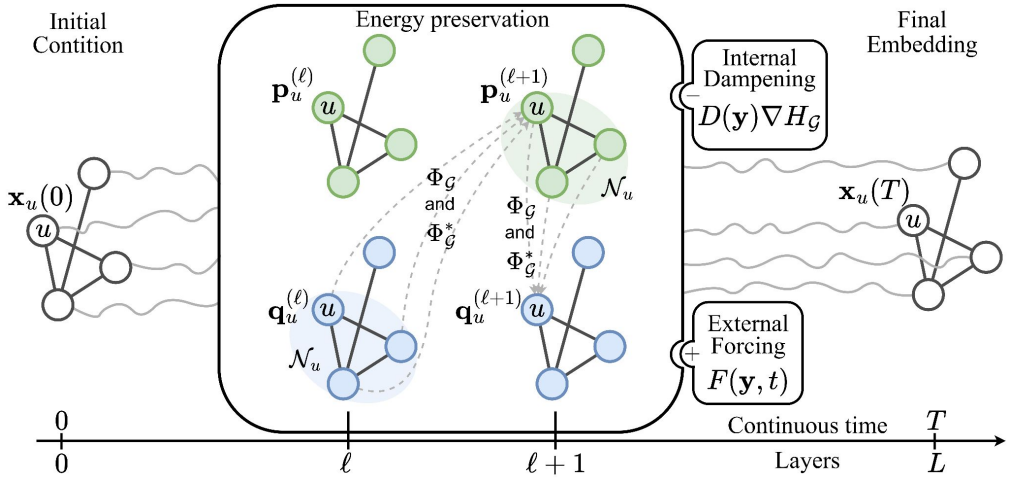
Port-Hamiltonian DGN

👉 The Hamiltonian

$$H_{\mathcal{G}}(\mathbf{y}) \coloneqq \sum_{u \in \mathcal{V}}^{n} \tilde{\sigma}(\mathbf{W}\mathbf{x}_u + \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_u) + \mathbf{b})^\top \mathbf{1}_d$$

The Continuous Dynamic

$$\frac{\mathrm{d}\mathbf{x}_u(t)}{\mathrm{d}t} = \mathcal{J}_u \nabla_{\mathbf{x}_u} H_{\mathcal{G}}(\mathbf{y}(t)) = \mathcal{J}_u \left[\mathbf{W}^\top \sigma(\mathbf{W}\mathbf{x}_u(t) + \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_u) + \mathbf{b}) + \sum_{v \in \mathcal{N}_u \cup \{u\}} \frac{\partial \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_v)}{\partial \mathbf{x}_u(t)}^\top \sigma(\mathbf{W}\mathbf{x}_v(t) + \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_v) + \mathbf{b}) \right]$$



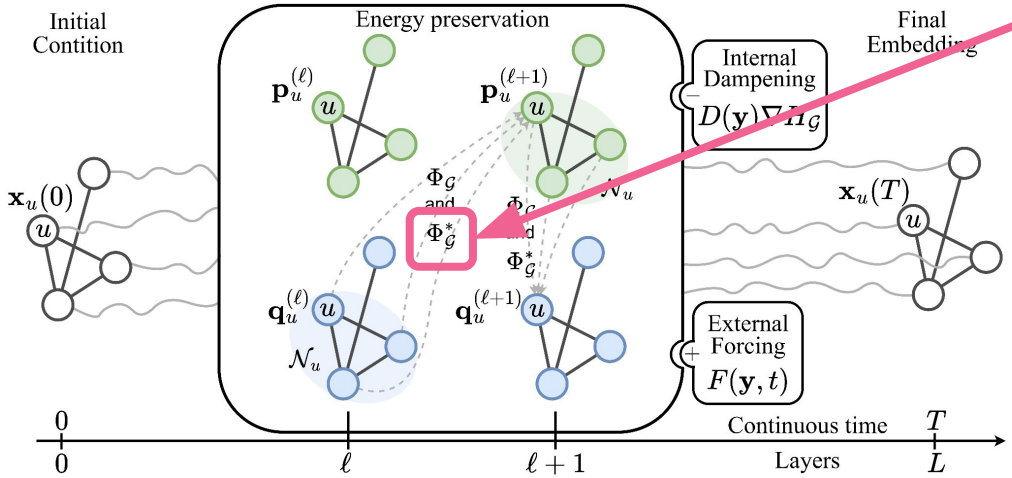
Port-Hamiltonian DGN

👉 The Hamiltonian

$$H_{\mathcal{G}}(\mathbf{y}) \coloneqq \sum_{u \in \mathcal{V}}^{n} \tilde{\sigma}(\mathbf{W}\mathbf{x}_u + \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_u) + \mathbf{b})^\top \mathbf{1}_d$$

The Continuous Dynamic

$$\frac{\mathrm{d}\mathbf{x}_u(t)}{\mathrm{d}t} = \mathcal{J}_u \nabla_{\mathbf{x}_u} H_{\mathcal{G}}(\mathbf{y}(t)) = \mathcal{J}_u \left[\mathbf{W}^\top \sigma(\mathbf{W}\mathbf{x}_u(t) + \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_u) + \mathbf{b}) + \sum_{v \in \mathcal{N}_u \cup \{u\}} \frac{\partial \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_v)}{\partial \mathbf{x}_u(t)}^\top \sigma(\mathbf{W}\mathbf{x}_v(t) + \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_v) + \mathbf{b}) \right]$$

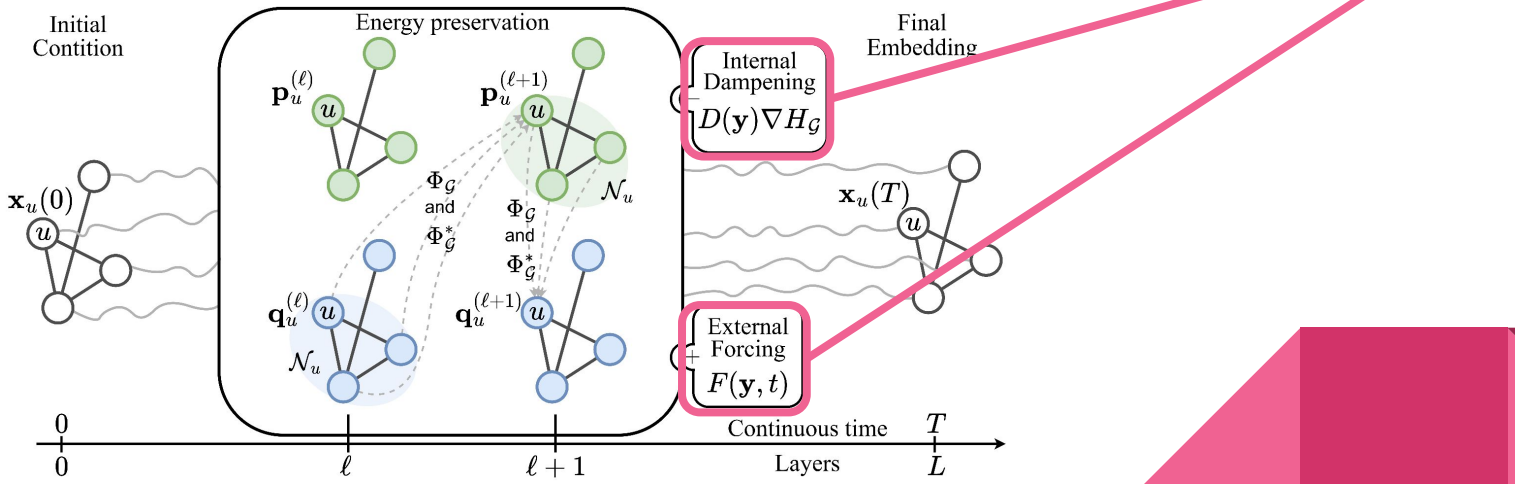


Port-Hamiltonian DGN

$$H_{\mathcal{G}}(\mathbf{y}) \coloneqq \sum_{u \in \mathcal{V}}^{n} \tilde{\sigma}(\mathbf{W}\mathbf{x}_u + \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_u) + \mathbf{b})^\top \mathbf{1}_d$$

The Continuous Dynamic

$$\frac{\mathrm{d}\mathbf{x}_u(t)}{\mathrm{d}t} = \mathcal{J}_u \nabla_{\mathbf{x}_u} H_{\mathcal{G}}(\mathbf{y}(t)) = \mathcal{J}_u \left[\mathbf{W}^\top \sigma(\mathbf{W}\mathbf{x}_u(t) + \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_u) + \mathbf{b}) + \sum_{v \in \mathcal{N}_u \cup \{u\}} \frac{\partial \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_v)}{\partial \mathbf{x}_u(t)}^\top \sigma(\mathbf{W}\mathbf{x}_v(t) + \Phi_{\mathcal{G}}(\mathbf{y}, \mathcal{N}_v) + \mathbf{b}) \right]$$



Theoretical Properties

Lemma (Eigenvalues of the Jacobian)

The local dynamics possess an Jacobian with eigenvalues purely on the imaginary axis, i.e.

$$\mathrm{Re}\left(\lambda_i\left(\frac{\partial}{\partial \mathbf{x}_u}\mathcal{J}_u\nabla_{\mathbf{x}_u}H_{\mathcal{G}}(\mathbf{y}(t))\right)\right)=0 \quad ,\forall i$$

Theoretical Properties

Lemma (Eigenvalues of the Jacobian)

The local dynamics possess an Jacobian with eigenvalues purely on the imaginary axis, i.e.

$$\mathrm{Re}\left(\lambda_i\left(\frac{\partial}{\partial \mathbf{x}_u}\mathcal{J}_u\nabla_{\mathbf{x}_u}H_{\mathcal{G}}(\mathbf{y}(t))\right)\right)=0 \quad ,\forall i$$

Lemma (Divergence-Free Hamiltonian Vector Field)

The vector field defined by the local dynamics is divergence-free everywhere, which means:

$$\nabla\cdot\mathcal{J}_u\nabla_{\mathbf{x}_u}H_{\mathcal{G}}(\mathbf{y}(t))=0, \textit{ everywhere}$$

Hence, no sources or sinks can exist in the nonlinear dynamic.

👉 Allows for long-range propagation!

Theoretical Properties cont.

Theorem (Lower Bound)

Given a Hamiltonian graph ODE, the BSM for each node can not vanish. In particular, for any sub-multiplicative matrix norm $\|\cdot\|$, the lower bound is:

$$\left\| \frac{\partial \mathbf{x}_u(T)}{\partial \mathbf{x}_u(T-t)} \right\| \geq 1, \forall t \in [0, T].$$

Galimberti, Clara Lucía, et al. “Hamiltonian Deep Neural Networks Guaranteeing Nonvanishing Gradients by Design”. IEEE Transactions on Automatic Control 68, no. 5 (2023): 3155–3162.

Theoretical Properties cont.

Theorem (Lower Bound)

Given a Hamiltonian graph ODE, the BSM for each node can not vanish. In particular, for any sub-multiplicative matrix norm $\|\cdot\|$, the lower bound is:

$$\left\| \frac{\partial \mathbf{x}_u(T)}{\partial \mathbf{x}_u(T-t)} \right\| \geq 1, \forall t \in [0, T].$$

👆Nodes can not forget!

Galimberti, Clara Lucía, et al. “Hamiltonian Deep Neural Networks Guaranteeing Nonvanishing Gradients by Design”. IEEE Transactions on Automatic Control 68, no. 5 (2023): 3155–3162.

Theoretical Properties cont.

Theorem (Lower Bound)

Given a Hamiltonian graph ODE, the BSM for each node can not vanish. In particular, for any sub-multiplicative matrix norm $\|\cdot\|$, the lower bound is:

$$\left\| \frac{\partial \mathbf{x}_u(T)}{\partial \mathbf{x}_u(T-t)} \right\| \geq 1, \forall t \in [0, T].$$

👆Nodes can not forget!

👉 full PH-DGN Picture see Thrm. B.1

$$\begin{aligned} \left\| \frac{\partial \mathbf{x}_u^{(\ell+1)}}{\partial \mathbf{x}_u^{(\ell)}} \right\|_{L_1} &\geq \left| \left\| \frac{\partial \tilde{\mathbf{x}}_u^{(\ell+1)}}{\partial \mathbf{x}_u^{(\ell)}} \right\| - \|\mathbf{D}\| \right| \\ &\geq 1 - \epsilon w B_d c'_\sigma (N + 3 + w) \end{aligned}$$

Galimberti, Clara Lucía, et al. “Hamiltonian Deep Neural Networks Guaranteeing Nonvanishing Gradients by Design”. IEEE Transactions on Automatic Control 68, no. 5 (2023): 3155–3162.

Graph Property Prediction

Table 1: Mean test $log_{10}(\text{MSE})$ and std average over 4 training seeds on the Graph Property Prediction. Our methods and DE-DGN baselines are implemented with weight sharing. The **first**, **second**, and **third** best scores are colored. *The lower, the better*

Model	Eccentricity	Diameter	SSSP
MPNNs			
GCN	$0.8468_{\pm 0.0028}$	$0.7424_{\pm 0.0466}$	$0.9499_{\pm 0.0001}$
GAT	$0.7909_{\pm 0.0222}$	$0.8221_{\pm 0.0752}$	$0.6951_{\pm 0.1499}$
GraphSAGE	$0.7863_{\pm 0.0207}$	$0.8645_{\pm 0.0401}$	$0.2863_{\pm 0.1843}$
GIN	$0.9504_{\pm 0.0007}$	$0.6131_{\pm 0.0990}$	$-0.5408_{\pm 0.4193}$
GCNII	$0.7640_{\pm 0.0355}$	$0.5287_{\pm 0.0570}$	$-1.1329_{\pm 0.0135}$
DE-DGNs			
DGC	$0.8261_{\pm 0.0032}$	$0.6028_{\pm 0.0050}$	$-0.1483_{\pm 0.0231}$
GraphCON	$0.6833_{\pm 0.0074}$	$0.0964_{\pm 0.0620}$	$-1.3836_{\pm 0.0092}$
GRAND	$0.6602_{\pm 0.1393}$	$0.6715_{\pm 0.0490}$	$-0.0942_{\pm 0.3897}$
A-DGN	$\mathbf{0.4296}_{\pm 0.1003}$	$\mathbf{-0.5188}_{\pm 0.1812}$	$\mathbf{-3.2417}_{\pm 0.0751}$
HamGNN	$0.7851_{\pm 0.0140}$	$0.6762_{\pm 0.1317}$	$0.9449_{\pm 0.0008}$
HANG	$0.8302_{\pm 0.0051}$	$1.1036_{\pm 0.1025}$	$0.1671_{\pm 0.0160}$
Ours			
PH-DGN$_C$	$\mathbf{-0.7248}_{\pm 0.1068}$	$\mathbf{-0.5473}_{\pm 0.1074}$	$\mathbf{-3.0467}_{\pm 0.1615}$
PH-DGN	$\mathbf{-0.9348}_{\pm 0.2097}$	$\mathbf{-0.5385}_{\pm 0.0187}$	$\mathbf{-4.2993}_{\pm 0.0721}$

than the best MPNN on average

than the best baseline on average

Graph Property Prediction

Table 1: Mean test log_{10}(MSE) and std average over 4 training seeds on the Graph Property Prediction. Our methods and DE-DGN baselines are implemented with weight sharing. The **first**, **second**, and **third** best scores are colored. *The lower, the better*

Model	Eccentricity	Diameter	SSSP
MPNNs			
GCN	$0.8468_{\pm 0.0028}$	$0.7424_{\pm 0.0466}$	$0.9499_{\pm 0.0001}$
GAT	$0.7909_{\pm 0.0222}$	$0.8221_{\pm 0.0752}$	$0.6951_{\pm 0.1499}$
GraphSAGE	$0.7863_{\pm 0.0207}$	$0.8645_{\pm 0.0401}$	$0.2863_{\pm 0.1843}$
GIN	$0.9504_{\pm 0.0007}$	$0.6131_{\pm 0.0990}$	$-0.5408_{\pm 0.4193}$
GCNII	$0.7640_{\pm 0.0355}$	$0.5287_{\pm 0.0570}$	$-1.1329_{\pm 0.0135}$
DE-DGNs			
DGC	$0.8261_{\pm 0.0032}$	$0.6028_{\pm 0.0050}$	$-0.1483_{\pm 0.0231}$
GraphCON	$0.6833_{\pm 0.0074}$	$0.0964_{\pm 0.0620}$	$-1.3836_{\pm 0.0092}$
GRAND	$0.6602_{\pm 0.1393}$	$0.6715_{\pm 0.0490}$	$-0.0942_{\pm 0.3897}$
A-DGN	$\mathbf{0.4296}_{\pm 0.1003}$	$\mathbf{-0.5188}_{\pm 0.1812}$	$\mathbf{-3.2417}_{\pm 0.0751}$
HamGNN	$0.7851_{\pm 0.0140}$	$0.6762_{\pm 0.1317}$	$0.9449_{\pm 0.0008}$
HANG	$0.8302_{\pm 0.0051}$	$1.1036_{\pm 0.1025}$	$0.1671_{\pm 0.0160}$
Ours			
PH-DGN$_C$	$\mathbf{-0.7248}_{\pm 0.1068}$	$\mathbf{-0.5473}_{\pm 0.1074}$	$\mathbf{-3.0467}_{\pm 0.1615}$
PH-DGN	$\mathbf{-0.9348}_{\pm 0.2097}$	$\mathbf{-0.5385}_{\pm 0.0187}$	$\mathbf{-4.2993}_{\pm 0.0721}$

than the best MPNN on average

than the best baseline on average

Long Range Graph Benchmark

include PE and SE

Model	Peptides-func AP ↑	Peptides-struct MAE ↓
Modified MPNNs, Tönshoff et al. (2023)		
GCN+PE/SE	$0.6860_{\pm 0.0050}$	**0.2460**$_{\pm 0.0007}$
GCNII+PE/SE	$0.6444_{\pm 0.0011}$	$0.2507_{\pm 0.0012}$
GINE+PE/SE	$0.6621_{\pm 0.0067}$	$0.2473_{\pm 0.0017}$
GatedGCN+PE/SE	$0.6765_{\pm 0.0047}$	$0.2477_{\pm 0.0009}$
Multi-hop DGNs, Gutteridge et al. (2023)		
DIGL+MPNN+LapPE	$0.6830_{\pm 0.0026}$	$0.2616_{\pm 0.0018}$
MixHop-GCN+LapPE	$0.6843_{\pm 0.0049}$	$0.2614_{\pm 0.0023}$
DRew-GCN+LapPE	**0.7150**$_{\pm 0.0044}$	$0.2536_{\pm 0.0015}$
Transformers, Gutteridge et al. (2023)		
Transformer+LapPE	$0.6326_{\pm 0.0126}$	$0.2529_{\pm 0.0016}$
SAN+LapPE	$0.6384_{\pm 0.0121}$	$0.2683_{\pm 0.0043}$
GraphGPS+LapPE	$0.6535_{\pm 0.0041}$	$0.2500_{\pm 0.0005}$
DE-DGNs		
GRAND	$0.5789_{\pm 0.0062}$	$0.3418_{\pm 0.0015}$
GraphCON	$0.6022_{\pm 0.0068}$	$0.2778_{\pm 0.0018}$
A-DGN	$0.5975_{\pm 0.0044}$	$0.2874_{\pm 0.0021}$
SWAN	$0.6751_{\pm 0.0039}$	**0.2485**$_{\pm 0.0009}$
Ours		
PH-DGN$_{\text{C}}$	**0.6961**$_{\pm 0.0070}$	$0.2581_{\pm 0.0020}$
PH-DGN	**0.7012**$_{\pm 0.0045}$	**0.2465**$_{\pm 0.0020}$

Long Range Graph Benchmark

include PE and SE

Model	Peptides-func AP ↑	Peptides-struct MAE ↓
Modified MPNNs, Tönshoff et al. (2023)		
GCN+PE/SE	$0.6860_{\pm 0.0050}$	**0.2460**$_{\pm 0.0007}$
GCNII+PE/SE	$0.6444_{\pm 0.0011}$	$0.2507_{\pm 0.0012}$
GINE+PE/SE	$0.6621_{\pm 0.0067}$	$0.2473_{\pm 0.0017}$
GatedGCN+PE/SE	$0.6765_{\pm 0.0047}$	$0.2477_{\pm 0.0009}$
Multi-hop DGNs, Gutteridge et al. (2023)		
DIGL+MPNN+LapPE	$0.6830_{\pm 0.0026}$	$0.2616_{\pm 0.0018}$
MixHop-GCN+LapPE	$0.6843_{\pm 0.0049}$	$0.2614_{\pm 0.0023}$
DRew-GCN+LapPE	**0.7150**$_{\pm 0.0044}$	$0.2536_{\pm 0.0015}$
Transformers, Gutteridge et al. (2023)		
Transformer+LapPE	$0.6326_{\pm 0.0126}$	$0.2529_{\pm 0.0016}$
SAN+LapPE	$0.6384_{\pm 0.0121}$	$0.2683_{\pm 0.0043}$
GraphGPS+LapPE	$0.6535_{\pm 0.0041}$	$0.2500_{\pm 0.0005}$
DE-DGNs		
GRAND	$0.5789_{\pm 0.0062}$	$0.3418_{\pm 0.0015}$
GraphCON	$0.6022_{\pm 0.0068}$	$0.2778_{\pm 0.0018}$
A-DGN	$0.5975_{\pm 0.0044}$	$0.2874_{\pm 0.0021}$
SWAN	$0.6751_{\pm 0.0039}$	**0.2485**$_{\pm 0.0009}$
Ours		
PH-DGN$_{\mathrm{C}}$	**0.6961**$_{\pm 0.0070}$	$0.2581_{\pm 0.0020}$
PH-DGN	**0.7012**$_{\pm 0.0045}$	**0.2465**$_{\pm 0.0020}$

Better than MPNN, DE-DGN, graph transformers and rewiring

Conclusions

👉 PH-DGN is a new framework for DGNs

Balancing **Non-dissipative** and **non-conservative** behaviors

Effectively exploring long-range dependencies

Conclusions

👉 PH-DGN is a new framework for DGNs

Balancing **Non-dissipative** and **non-conservative** behaviors

Effectively exploring long-range dependencies

👉 PH-DGN can be used to reinterpret and extend any classical DGN

Conclusions

👉 PH-DGN is a new framework for DGNs

Balancing **Non-dissipative** and **non-conservative** behaviors

Effectively exploring long-range dependencies

👉 PH-DGN can be used to reinterpret and extend any classical DGN

👉 Our results show that:

PH-DGN outperforms state-of-the-art models

Data-driven forces maximize long-range propagation efficacy

The Team

Simon Heilig

Alessio Gravina

Alessandro Trenta

Claudio Gallicchio

Davide Bacciu

Thank you for watching!

Github

Simon Heilig

simon.heilig@rub.de

https://simonheilig.github.io/

Dept. of Computer Science, Ruhr University Bochum, Germany

Alessio Gravina

alessio.gravina@phd.unipi.it

http://pages.di.unipi.it/gravina/

Dept. of Computer Science, University of Pisa, Italy