< BRL MEGA project >

A Literature Survey on Physics-informed GNNs

MIDaS Lab

Presenter: Jin-Duk Park
Supervisor: Won-Yong Shin

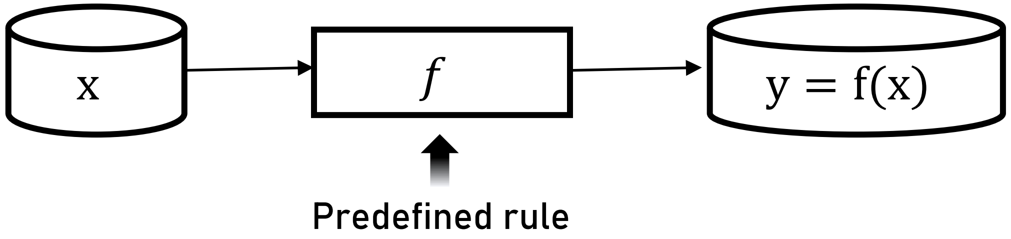
MLP is Universal Approximator

Theorem 2.4 [Hornik et al., 1989]

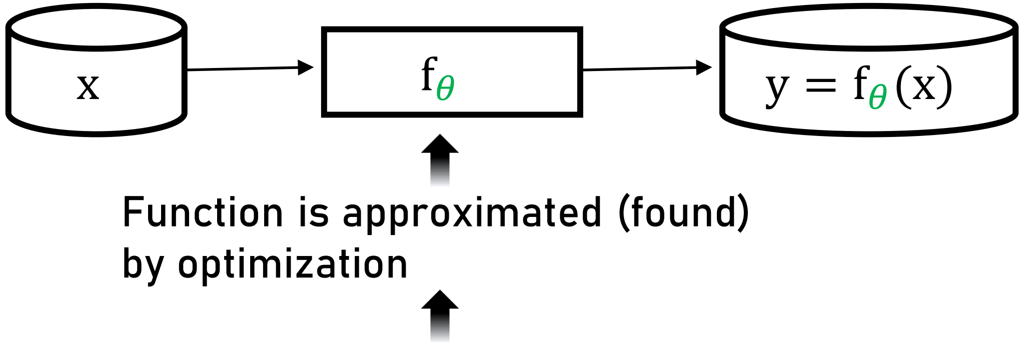
For every squashing function Ψ, every r, and every probability measure μ on (R^r, B^r), $\Sigma^r(\Psi)$ is uniformly dense on compacta in C^r and ρ_μ-dense in M^r. □

In other words, standard feedforward networks with only a single hidden layer can approximate any continuous function uniformly on any compact set and any measurable function arbitrarily well in the ρ_μ metric, regardless of the squashing function Ψ (continuous or not), regardless of the dimension of the input space r, and regardless of the input space

Rule-based Algorithm



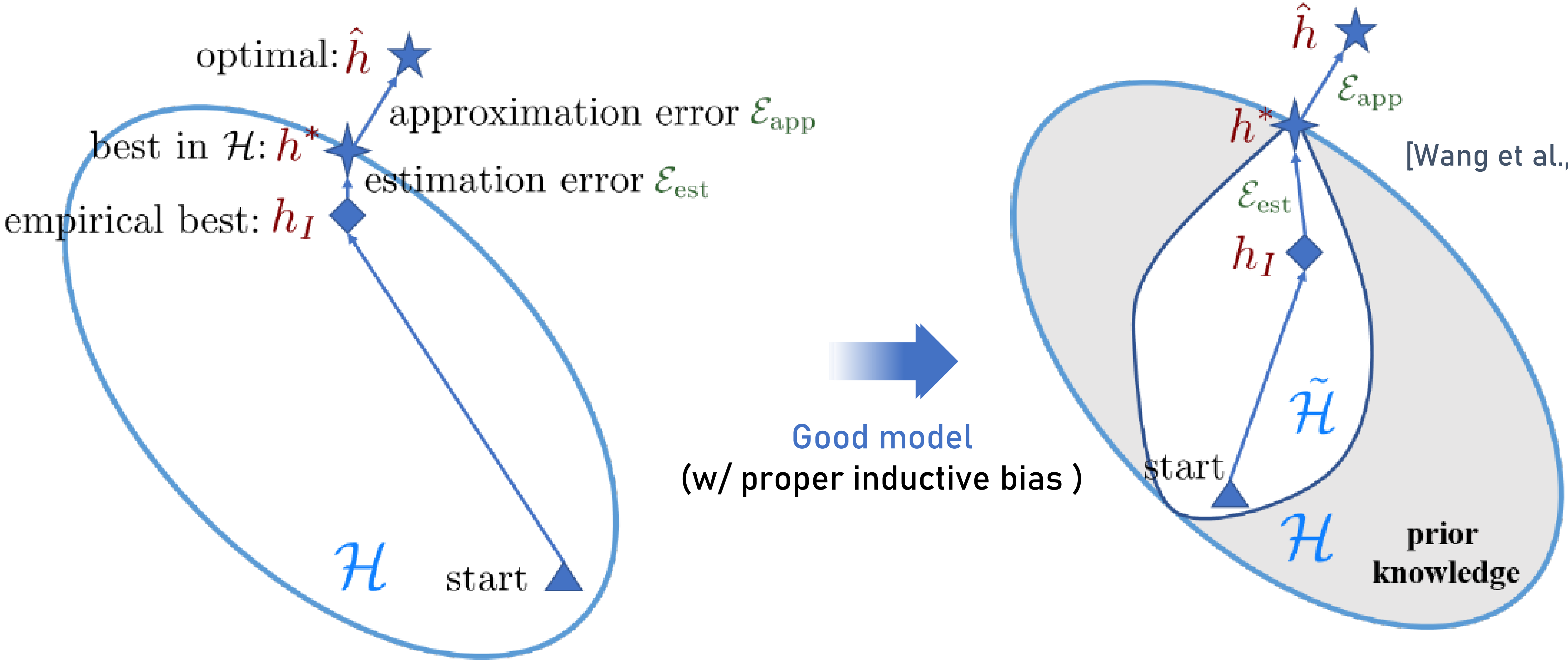
Machine Learning Algorithm



Choosing f_θ as MLP have capability of approximating any functions!

Hornik, Kurt, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators." *Neural networks* 2.5 (1989): 359-366.

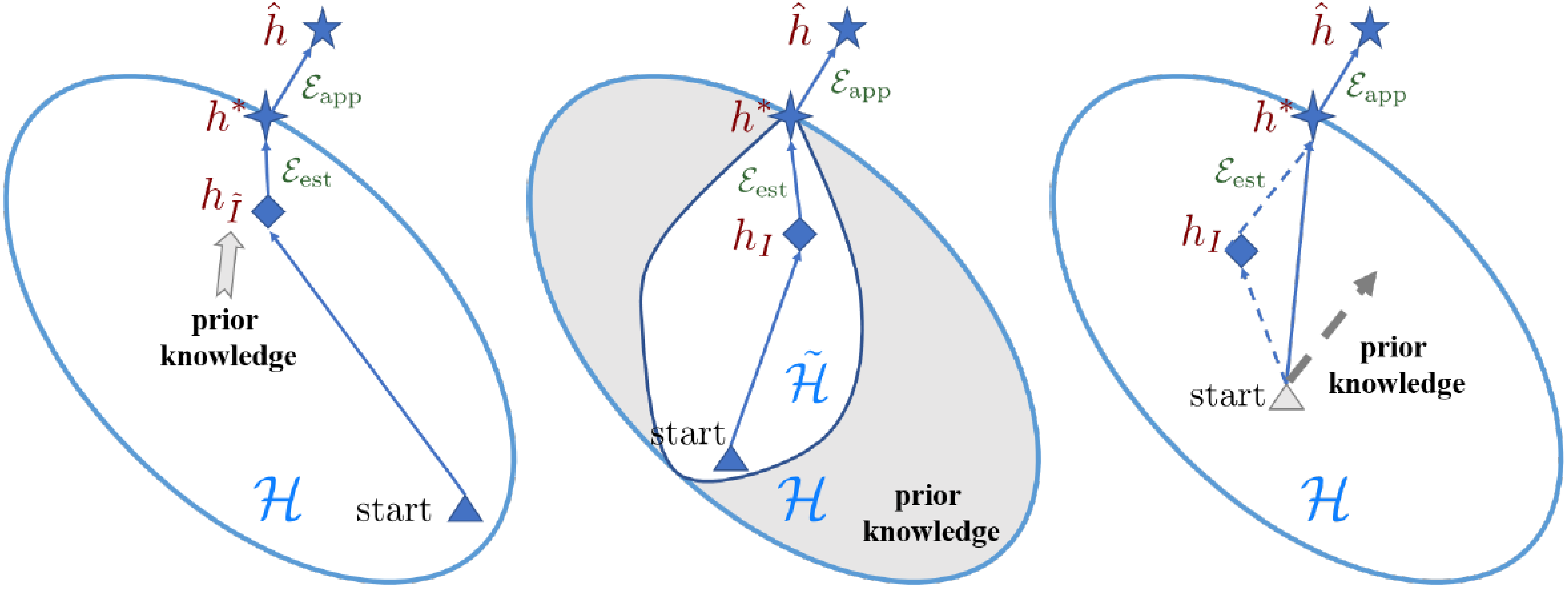
Empirical Risk Minimization (ERM) Perspective



[Wang et al., 2020]

Wang, Yaqing, et al. "Generalizing from a few examples: A survey on few-shot learning." ACM computing surveys (csur) 53.3 (2020): 1-34.

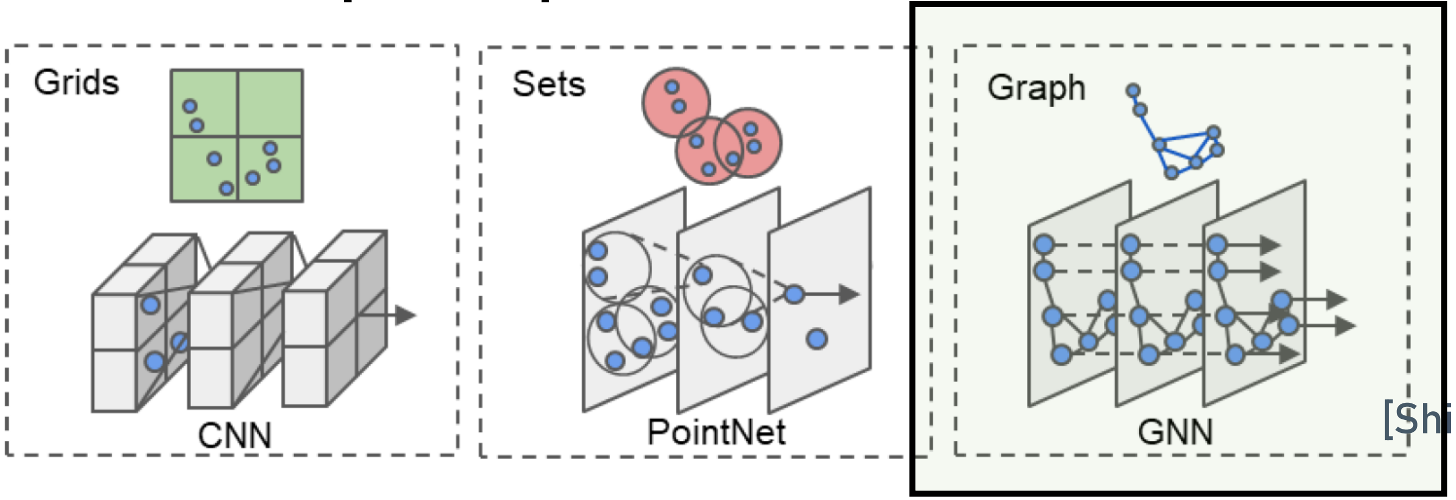
Empirical Risk Minimization (ERM) Perspective



(a) Data. (b) Model. (c) Algorithm.

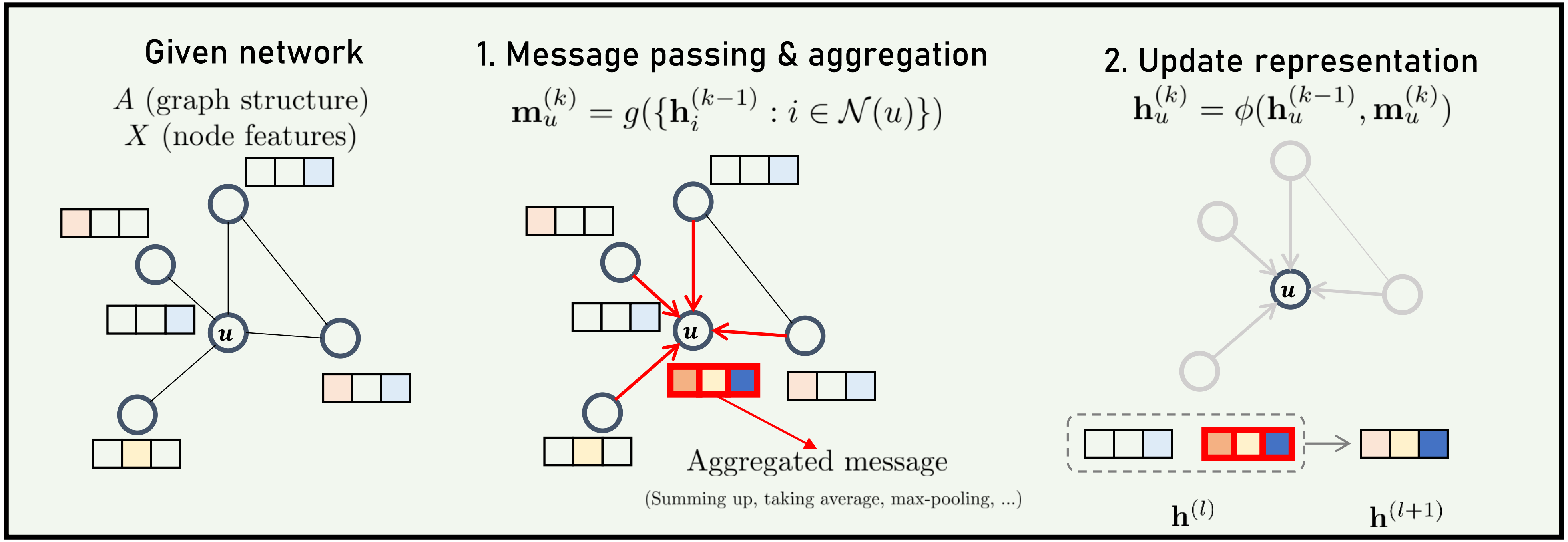
Wang, Yaqing, et al. "Generalizing from a few examples: A survey on few-shot learning." ACM computing surveys (csur) 53.3 (2020): 1-34.

Short Recap: Graph Neural Network



[Shi et al., 2020]

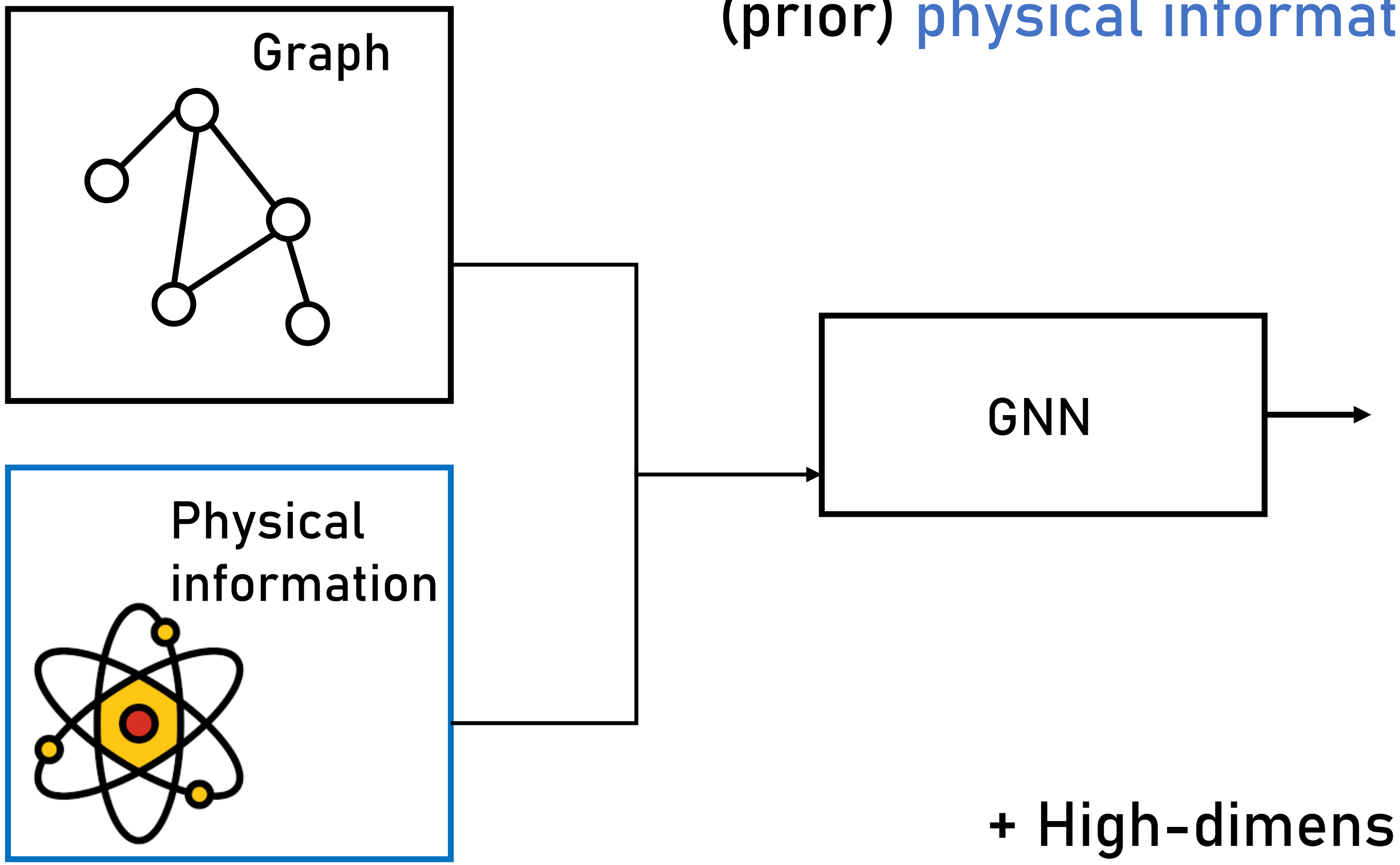
Graph neural network (GNN) is a "good model" for graph data

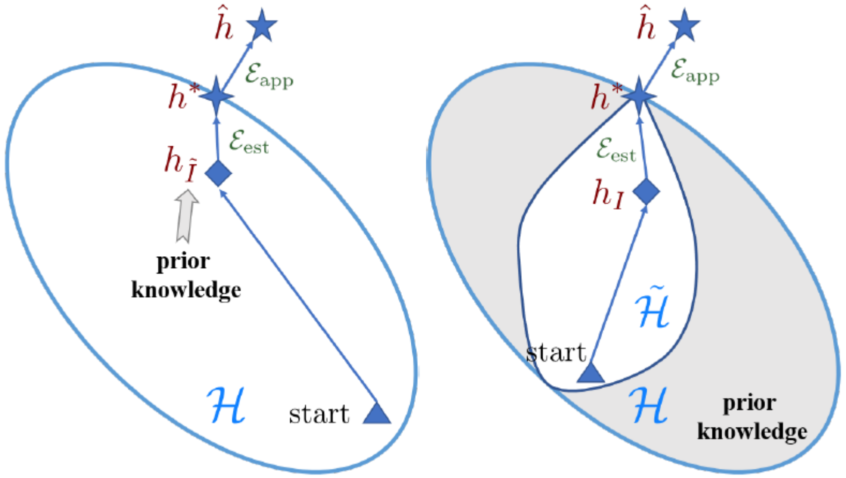


Shi, Weijing, and Raj Rajkumar. "Point-gnn: Graph neural network for 3d object detection in a point cloud." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.

Physics-Informed GNNs

Further improve model inductive bias or data utilizing (prior) physical information.





(a) Data. (b) Model.

+ High-dimensional scientific simulations: expensive, parameter tuning for each system

1. Equivariant Graph Neural Network

Satorras, Vıctor Garcia, Emiel Hoogeboom, and Max Welling. "E (n) equivariant graph neural networks." International conference on machine learning. PMLR, 2021.

2. MeshGraphNets

Pfaff, Tobias, et al. "Learning mesh-based simulation with graph networks." ICLR 2021

1. Equivariant Graph Neural Network

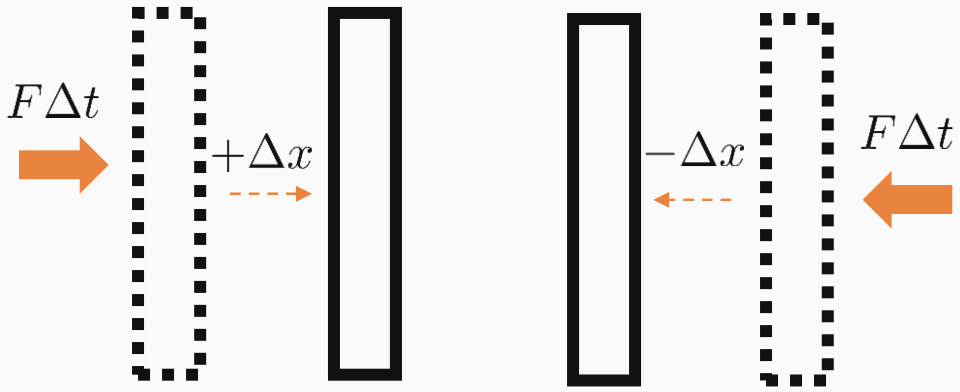
Satorras, Vıctor Garcia, Emiel Hoogeboom, and Max Welling. "E (n) equivariant graph neural networks." International conference on machine learning. PMLR, 2021.

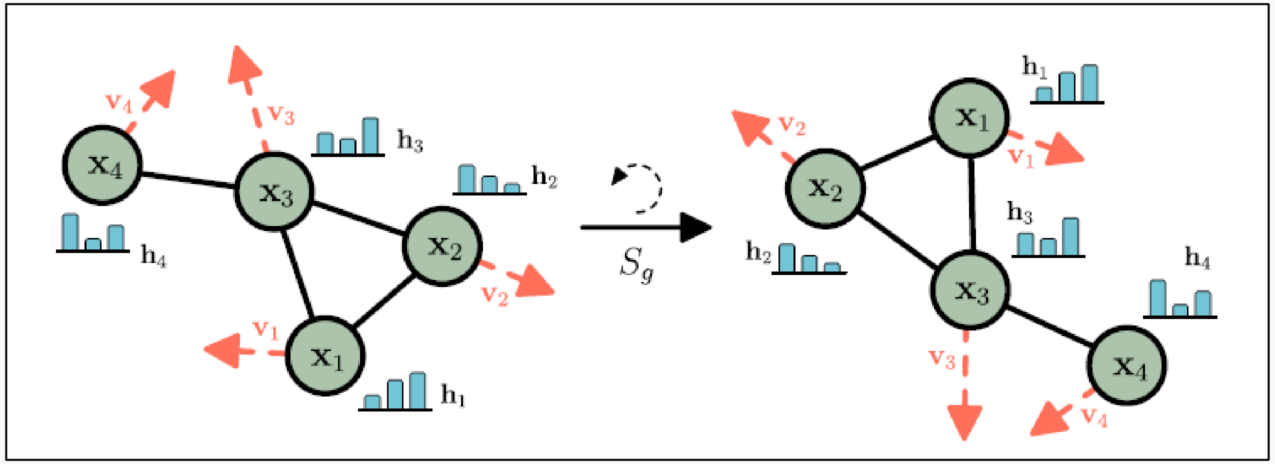
2. MeshGraphNets

Pfaff, Tobias, et al. "Learning mesh-based simulation with graph networks." ICLR 2021

Physics in Real-worlds and Equivariance of Models

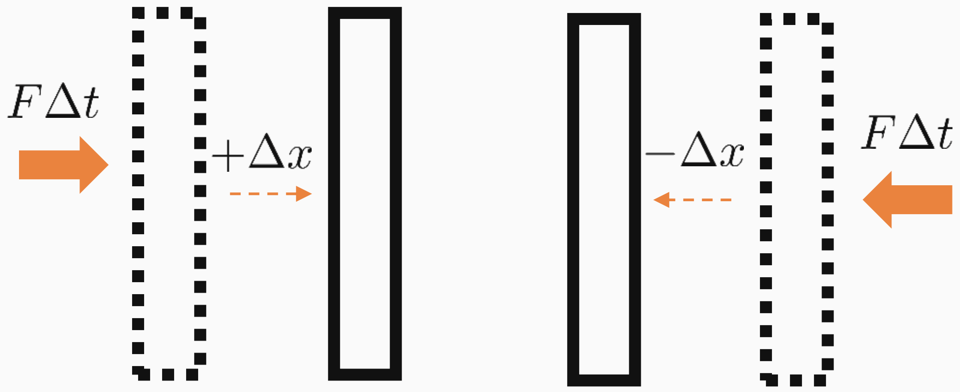
Symmetry in physics

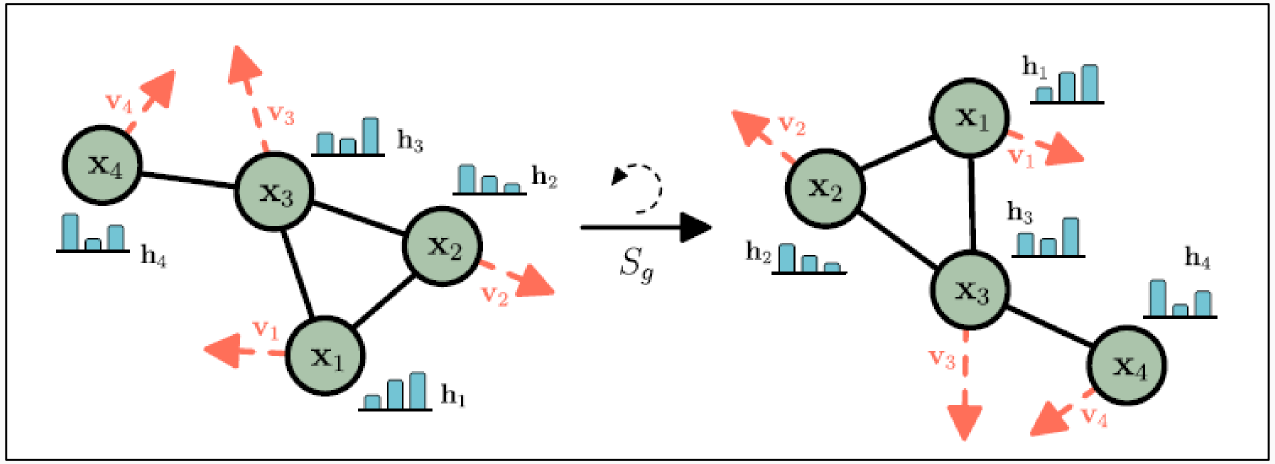




Physics in Real-worlds and Equivariance of Models

Symmetry in physics





- Let a model aware physical patterns
- A model with good inductive bias
- *Equivariant* neural network

* What is Equivariance?

Let $T_g : X \to X$ be a set of transformations on X for the abstract group $g \in G$. We say a function $\phi : X \to Y$ is equivariant to g if there exists an equivalent transformation on its output space $S_g : Y \to Y$ such that:

$$\phi(T_g(\mathbf{x})) = S_g(\phi(\mathbf{x})) \quad (1)$$

Types of Equivariance

3 Types of equivariance in particles

$\phi(\cdot)$: equivariant model

1. Translation equivariance

$$y + g = \phi(x + g)$$

2. Rotation equivariance

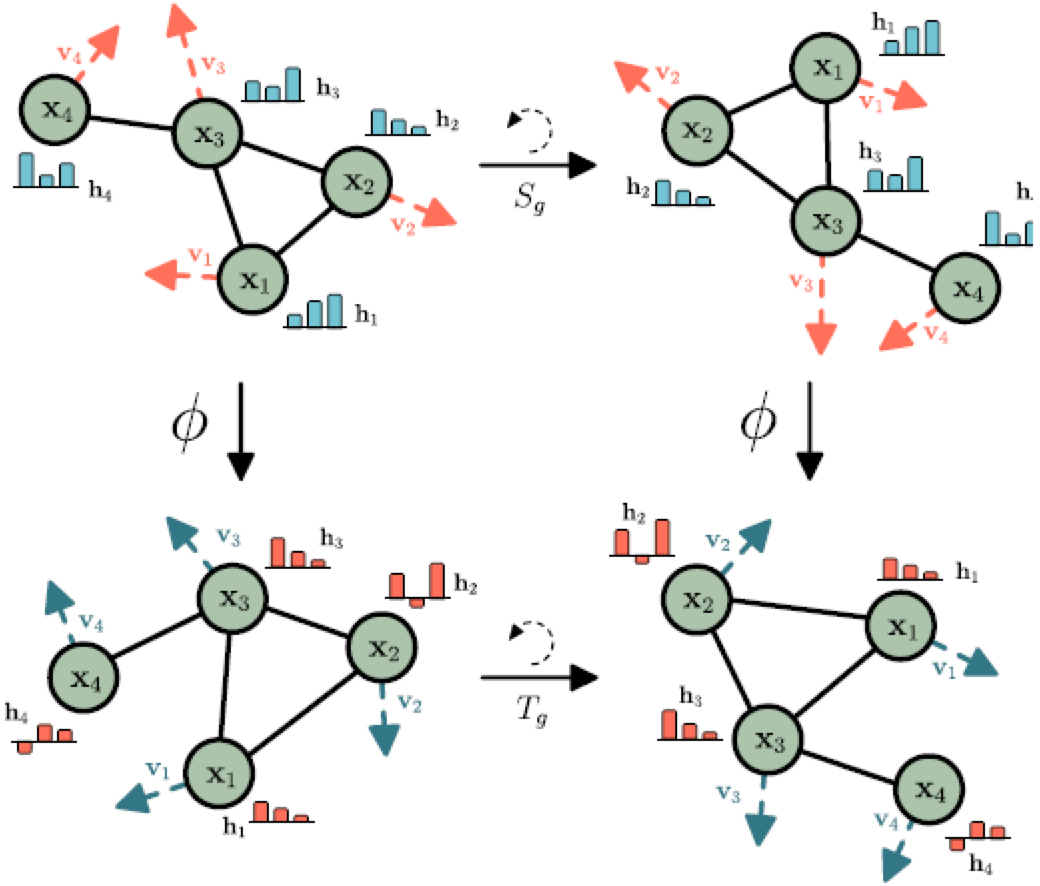
Q: orthogonal matrix

$$Qy = \phi(Qx)$$

3. Permutation equivariance

P: permutation matrix

$$Py = \phi(Px)$$



EGNN: Equivariant Graph Neural Network

$$\mathbf{m}_{ij} = \phi_e \left(\mathbf{h}_i^l, \mathbf{h}_j^l, \left\| \mathbf{x}_i^l - \mathbf{x}_j^l \right\|^2, a_{ij} \right)$$

$$\mathbf{x}_i^{l+1} = \mathbf{x}_i^l + C \sum_{j \neq i} \left(\mathbf{x}_i^l - \mathbf{x}_j^l \right) \phi_x \left(\mathbf{m}_{ij} \right)$$

$$\mathbf{m}_i = \sum_{j \in \mathcal{N}(i)} \mathbf{m}_{ij}$$

$$\mathbf{h}_i^{l+1} = \phi_h \left(\mathbf{h}_i^l, \mathbf{m}_i \right)$$

EGNN: Equivariant Graph Neural Network

Novel proposition

$$\mathbf{m}_{ij} = \phi_e\left(\mathbf{h}_i^l, \mathbf{h}_j^l, \left\|\mathbf{x}_i^l - \mathbf{x}_j^l\right\|^2, a_{ij}\right)$$

$$\mathbf{x}_i^{l+1} = \mathbf{x}_i^l + C\sum_{j\neq i}\left(\mathbf{x}_i^l - \mathbf{x}_j^l\right)\phi_x\left(\mathbf{m}_{ij}\right)$$

"Model aware *relative* distance between two coordinates"

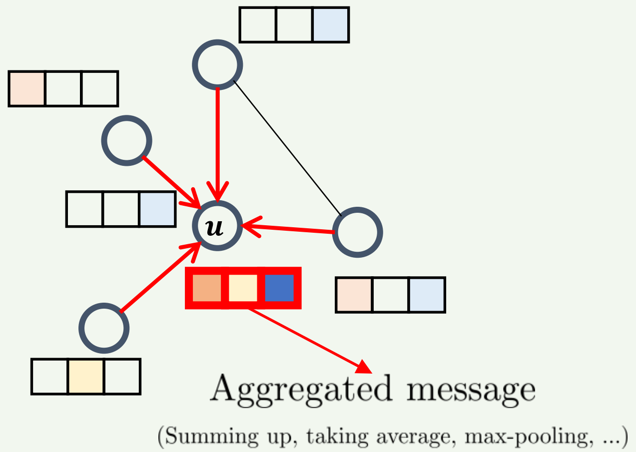
$$\mathbf{m}_i = \sum_{j\in\mathcal{N}(i)} \mathbf{m}_{ij}$$

$$\mathbf{h}_i^{l+1} = \phi_h\left(\mathbf{h}_i^l, \mathbf{m}_i\right)$$

Conventional GNNs

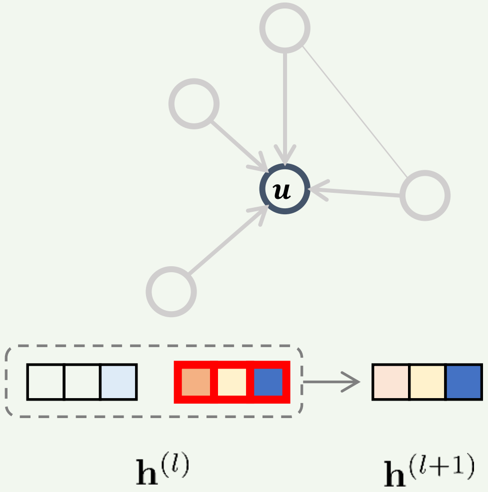
1. Message passing & aggregation

$$\mathbf{m}_u^{(k)} = g(\{\mathbf{h}_i^{(k-1)} : i \in \mathcal{N}(u)\})$$



2. Update representation

$$\mathbf{h}_u^{(k)} = \phi(\mathbf{h}_u^{(k-1)}, \mathbf{m}_u^{(k)})$$



EGNN: Equivariant Graph Neural Network

Novel proposition

$$\mathbf{m}_{ij} = \phi_e\left(\mathbf{h}_i^l, \mathbf{h}_j^l, \left\|\mathbf{x}_i^l - \mathbf{x}_j^l\right\|^2,\right.$$

$$\mathbf{x}_i^{l+1} = \mathbf{x}_i^l + C\sum_{j\neq i}\left(\mathbf{x}_i^l - \mathbf{x}_j^l\right)\phi_x$$

$$\mathbf{m}_i = \sum_{j\in\mathcal{N}(i)} \mathbf{m}_{ij}$$

$$\mathbf{h}_i^{l+1} = \phi_h\left(\mathbf{h}_i^l, \mathbf{m}_i\right)$$

B.1. Equivariance proof for velocity type inputs

In this subsection we prove that the velocity types input formulation of our model is also $\mathrm{E}(n)$ equivariant on $\mathbf{x}$. More formally, for any translation vector $g \in \mathbb{R}^n$ and for any orthogonal matrix $Q \in \mathbb{R}^{n\times n}$, the model should satisfy:

$$\mathbf{h}^{l+1}, Q\mathbf{x}^{l+1} + g, Q\mathbf{v}^{l+1} = \mathrm{EGCL}[\mathbf{h}^l, Q\mathbf{x}^l + g, Q\mathbf{v}^{\text{init}}, \mathcal{E}]$$

In Appendix A we already proved the equivariance of our EGNN (Section 3) when not including vector type inputs. In its velocity type inputs variant we only replaced its coordinate updates (eq. 4) by Equation 7 that includes velocity. Since this is the only modification we will only prove that Equation 7 re-written below is equivariant.

$$\mathbf{v}_i^{l+1} = \phi_v\left(\mathbf{h}_i^l\right)\mathbf{v}_i^{\text{init}} + C\sum_{j\neq i}\left(\mathbf{x}_i^l - \mathbf{x}_j^l\right)\phi_x\left(\mathbf{m}_{ij}\right)$$

$$\mathbf{x}_i^{l+1} = \mathbf{x}_i^l + \mathbf{v}_i^{l+1}$$

First, we prove the first line preserves equivariance, that is we want to show:

$$Q\mathbf{v}_i^{l+1} = \phi_v\left(\mathbf{h}_i^l\right)Q\mathbf{v}_i^{\text{init}} + C\sum_{j\neq i}\left(Q\mathbf{x}_i^l + g - [Q\mathbf{x}_j^l + g]\right)\phi_x\left(\mathbf{m}_{ij}\right)$$

Derivation.

$$+ g - [Q\mathbf{x}_j^l + g])\,\phi_x\left(\mathbf{m}_{ij}\right) = Q\phi_v\left(\mathbf{h}_i^l\right)\mathbf{v}_i^{\text{init}} + QC\sum_{j\neq i}\left(\mathbf{x}_i^l - \mathbf{x}_j^l\right)\phi_x\left(\mathbf{m}_{ij}\right) \quad (10)$$

$$= Q\left(\phi_v\left(\mathbf{h}_i^l\right)\mathbf{v}_i^{\text{init}} + C\sum_{j\neq i}\left(\mathbf{x}_i^l - \mathbf{x}_j^l\right)\phi_x\left(\mathbf{m}_{ij}\right)\right) \quad (11)$$

$$= Q\mathbf{v}_i^{l+1} \quad (12)$$

Finally, it is straightforward to show the second equation is also equivariant, that is we want to show $Q\mathbf{x}_i^{l+1} + g = Q\mathbf{x}_i^l + g + Q\mathbf{v}_i^{l+1}$

Derivation.

$$Q\mathbf{x}_i^l + g + Q\mathbf{v}_i^{l+1} = Q\left(\mathbf{x}_i^l + \mathbf{v}_i^{l+1}\right) + g$$
$$= Q\mathbf{x}_i^{l+1} + g$$

Concluding we showed that an $\mathrm{E}(n)$ transformation on the input set of points results in the same transformation on the output set of points such that $\mathbf{h}^{l+1}, Q\mathbf{x}^{l+1} + g, Q\mathbf{v}^{l+1} = \mathrm{EGCL}[\mathbf{h}^l, Q\mathbf{x}^l + g, Q\mathbf{v}^{\text{init}}, \mathcal{E}]$ is satisfied.

Full proof included in the arXiv version:
https://arxiv.org/abs/2102.09844

Results

Task	α	$\Delta\varepsilon$	$\varepsilon_{\mathrm{HOMO}}$	$\varepsilon_{\mathrm{LUMO}}$	μ	C_ν	G	H	R^2	U	U_0	ZPVE
Units	bohr^3	meV	meV	meV	D	cal/mol K	meV	meV	bohr^3	meV	meV	meV
NMP	.092	69	43	38	.030	.040	19	17	.180	20	20	1.50
Schnet	.235	63	41	34	.033	.033	14	14	.073	19	14	1.70
Cormorant	.085	61	34	38	.038	.026	20	21	.961	21	22	2.03
L1Net	.088	68	46	35	.043	.031	14	14	.354	14	13	1.56
LieConv	.084	49	30	25	.032	.038	22	24	.800	19	19	2.28
DimeNet++*	.044	33	25	20	.030	.023	8	7	.331	6	6	1.21
TFN	.223	58	40	38	.064	.101	-	-	-	-	-	-
SE(3)-Tr.	.142	53	35	33	.051	.054	-	-	-	-	-	-
EGNN	.071	48	29	25	.029	.031	12	12	.106	12	11	1.55

Table 3. Mean Absolute Error for the molecular property prediction benchmark in QM9 dataset. *DimeNet++ uses slightly different train/val/test partitions than the other papers listed here.

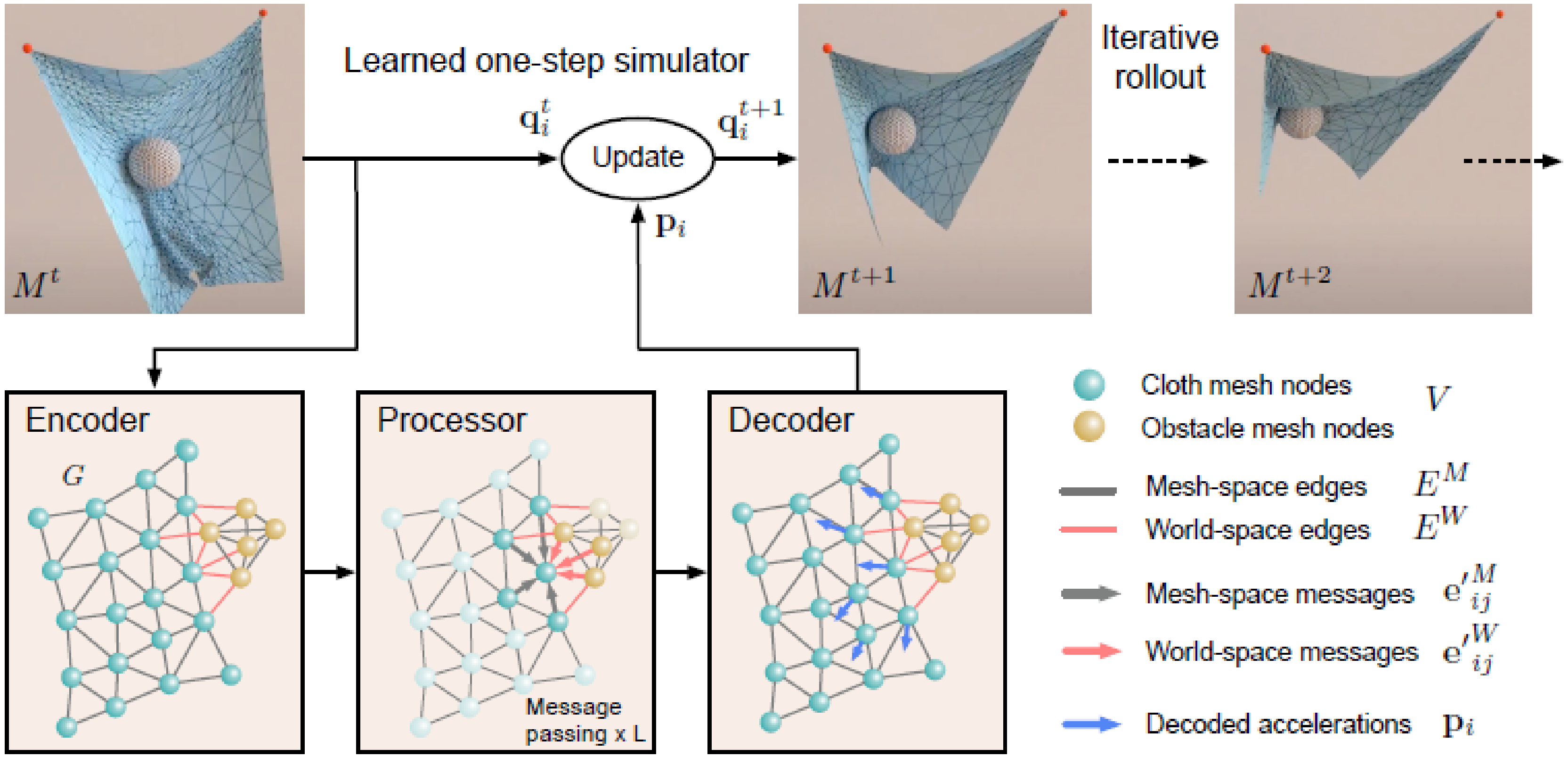
1. Equivariant Graph Neural Network

Satorras, Vıctor Garcia, Emiel Hoogeboom, and Max Welling. "E (n) equivariant graph neural networks." International conference on machine learning. PMLR, 2021.

2. MeshGraphNets

Pfaff, Tobias, et al. "Learning mesh-based simulation with graph networks." ICLR 2021

MeshGraphNets: Overview



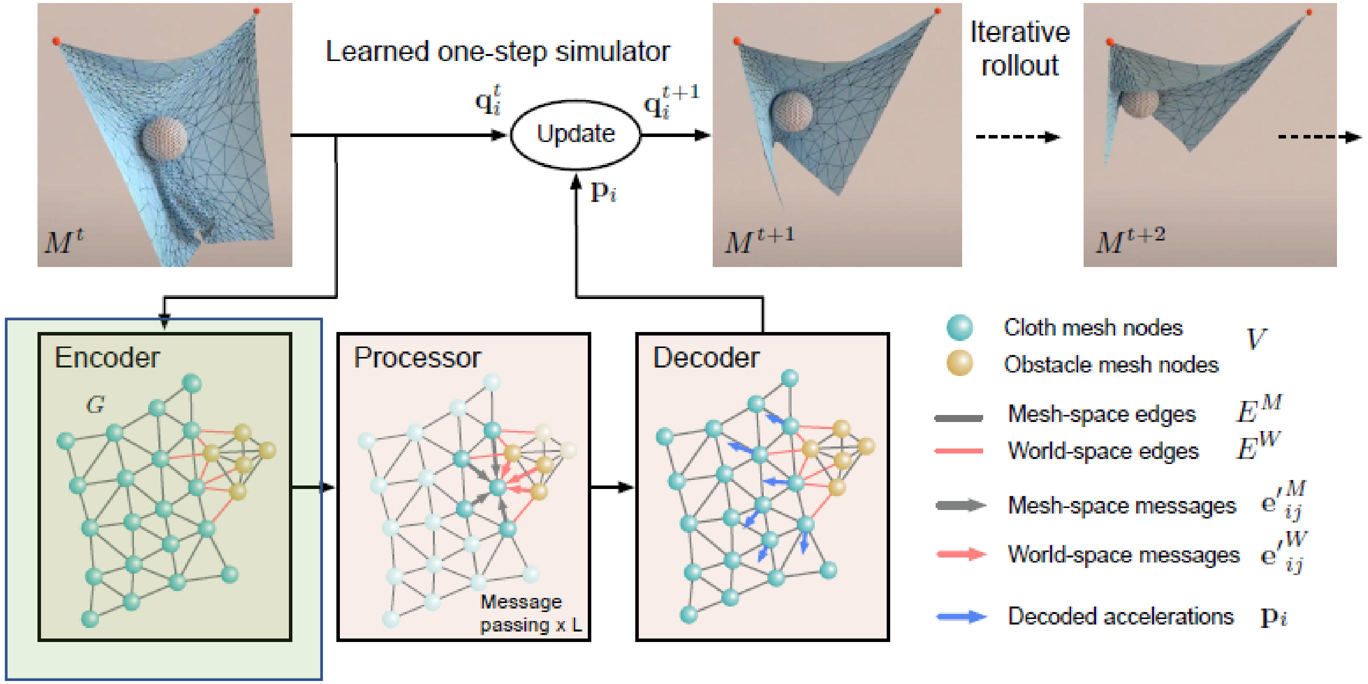
Watch <video>

Graph Construction

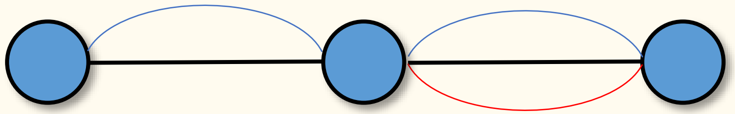
Encoder: Graph Construction

- Encoder encodes the current mesh M^t into a mutigraph $G = (V, E^M, E^W)$

- Two-types of edges (meshs) are constructed
  1. regular edges
  2. world space edges



* What is multi-graph?



Nodes are connected with multiple types of relations (edges)

Graph Construction

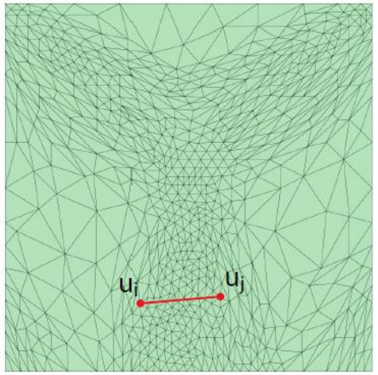
Encoder: Graph Construction

Given

- $\boldsymbol{u}_i$: mesh coordinate vector, for node i
- $\boldsymbol{x}_i$: world-space coordinate vector , for node i

1. Graph structure encoding

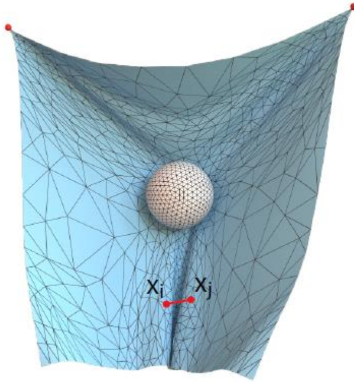
Regular edges



mesh space u

World-space edges

- Euclidean (spatial) proximity
- Add edges: $|\boldsymbol{x}_i - \boldsymbol{x}_j| < r_W$
- External dynamics that are non-local in mesh space, can be captured



world space x

2. Edge feature encoding

- $\boldsymbol{u}_i - \boldsymbol{u}_j$, $|\boldsymbol{u}_i|$: displacement vector and its norm
- $\boldsymbol{x}_i - \boldsymbol{x}_j$, $|\boldsymbol{x}_i|$: displacement vector and its norm

GNN in MeshGraphNets

Processer: GNNs

- L-identical message passing blocks are used

* $f(.)$: MLP

- Mesh edge update

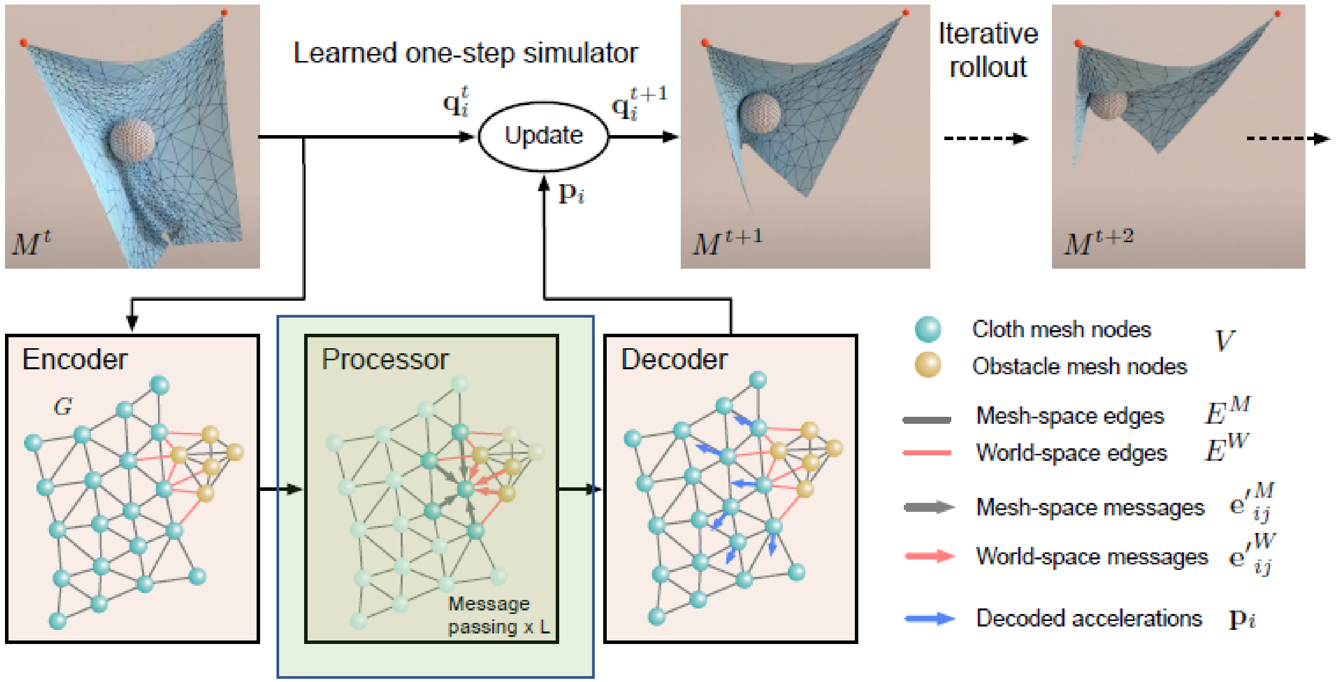
$$\mathbf{e}'^{M}_{ij} \leftarrow f^{M}(\mathbf{e}^{M}_{ij}, \mathbf{v}_i, \mathbf{v}_j)$$

- World edge update

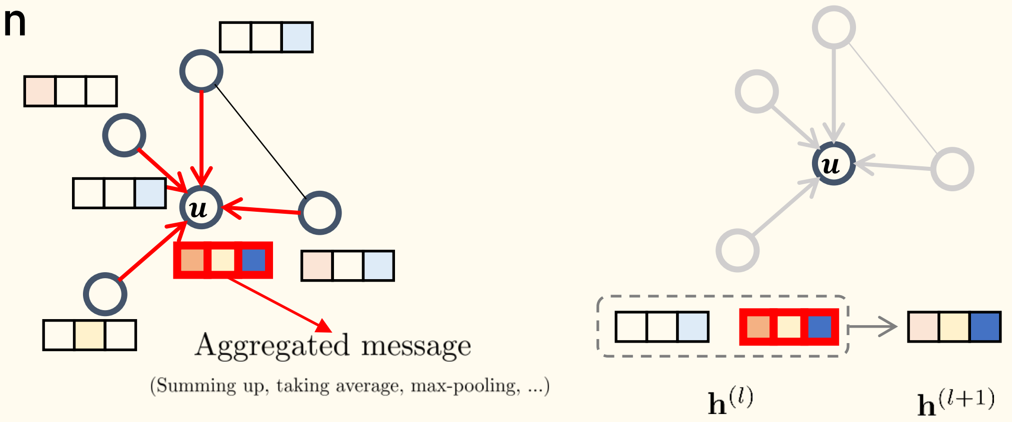
$$\mathbf{e}'^{W}_{ij} \leftarrow f^{W}(\mathbf{e}^{W}_{ij}, \mathbf{v}_i, \mathbf{v}_j)$$

- Node embedding update

$$\mathbf{v}'_i \leftarrow f^{V}(\mathbf{v}_i, \sum_j \mathbf{e}'^{M}_{ij}, \sum_j \mathbf{e}'^{W}_{ij})$$



* Aggregation & Update in GNNs



Next Step Prediction

Decoder: predict next step

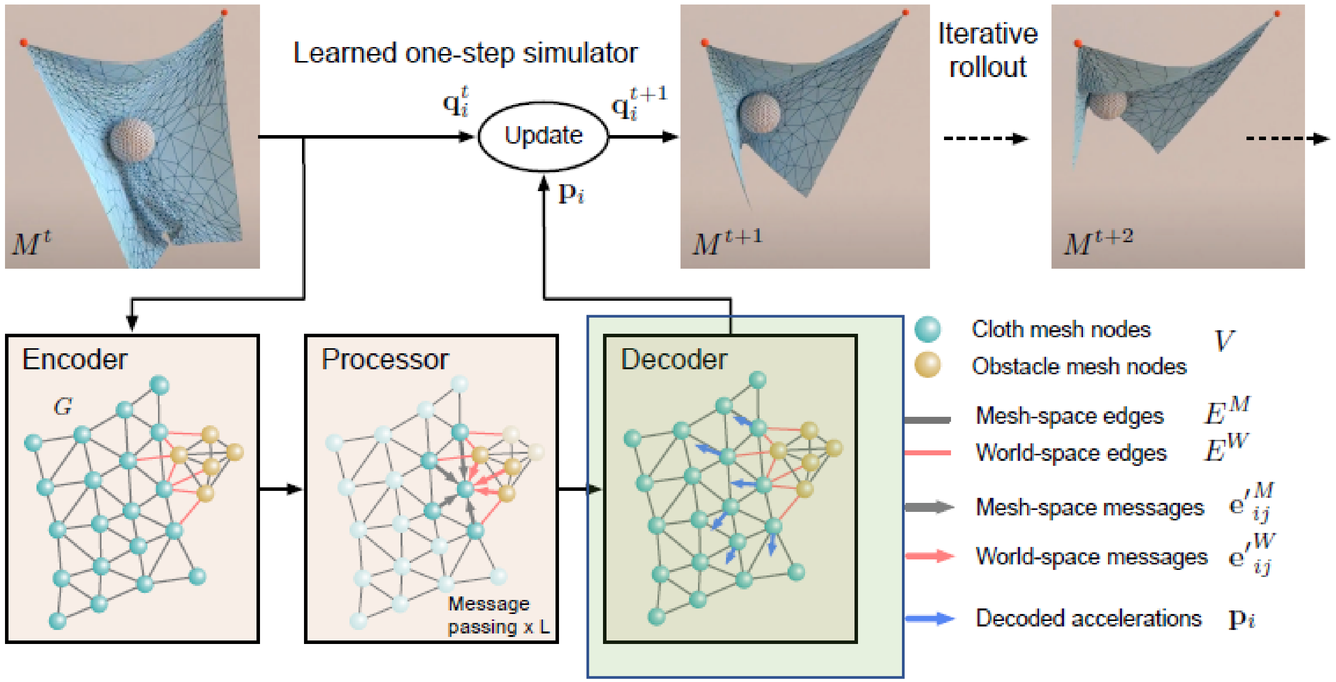
- MLP is used to decode next step output
- Interpret output features $\boldsymbol{p}_i$ as derivatives of $\boldsymbol{q}_i$

  - First order system

$$\mathrm{q}_i^{t+1} = \mathrm{p}_i + \mathrm{q}_i^t$$

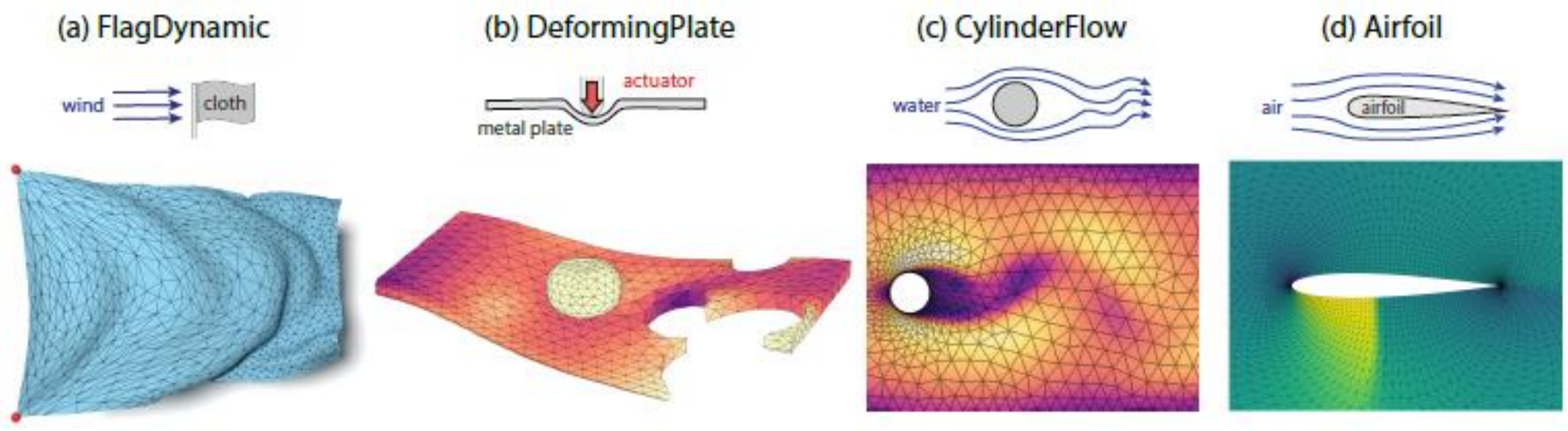
  - Second order system

$$\mathrm{q}_i^{t+1} = \mathrm{p}_i + 2\mathrm{q}_i^t - \mathrm{q}^{t-1}$$



Simulation Results

The Results



https://sites.google.com/view/meshgraphnets

< BRL MEGA project >

Thank you for your attention!

jindeok6@yonsei.ac.kr