Real-Time Solutions to PDEs with Neural Operators in SciML

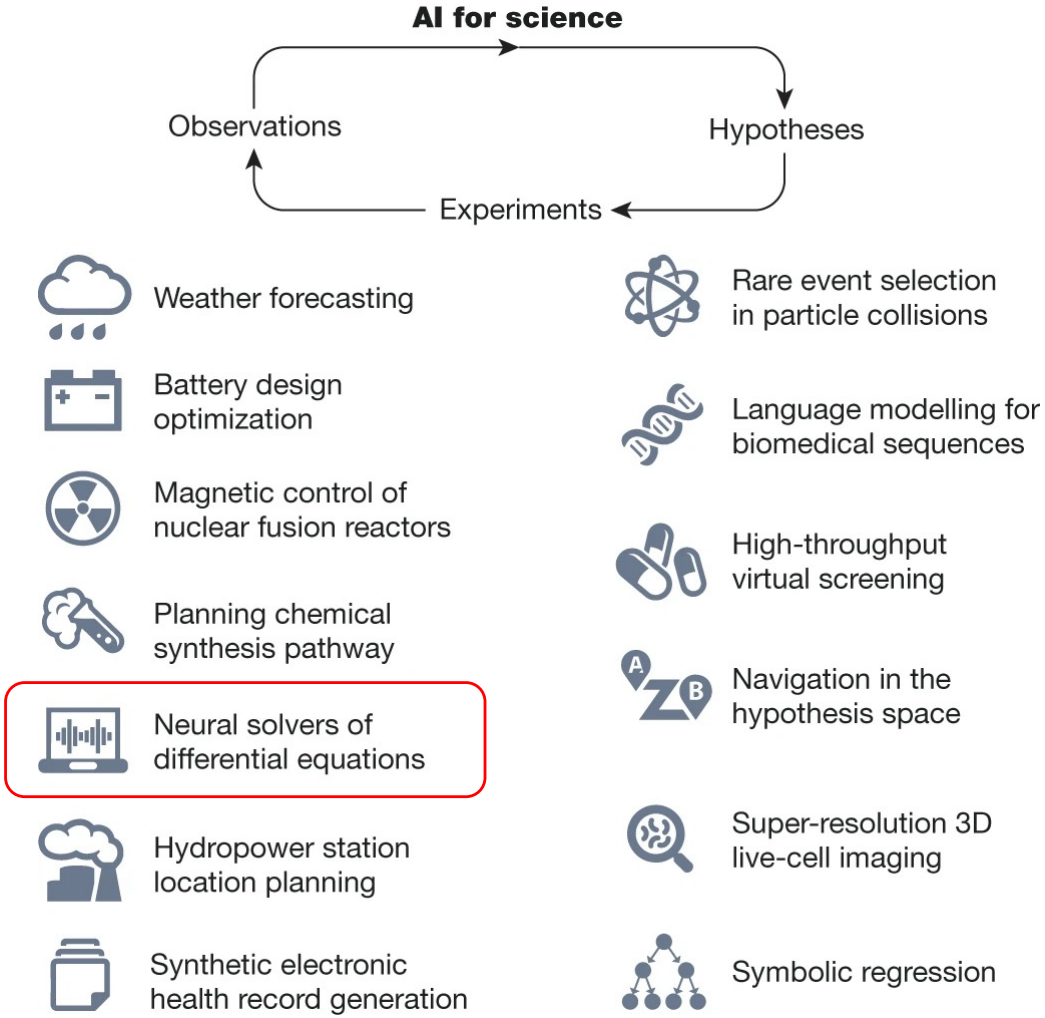
Department of Artificial Intelligence
Chung-Ang University
Jaeyong Lee
https://www.jaeyong-lee.com/

KIAS Center for AI and Natural Sciences Fall Workshop
2024.11.06.

Contents

- **Introduction**
  - AI for Science (AI4Science)
  - Scientific Machine Learning (SciML)
  - Neural Operator
- **Well-known Neural Operator Models and Related Researches**
  - Deep Operator Network (DeepONet)
    - Concept and Methodology
    - Overcoming the limitations of DeepONet [1, ICLR 2023]
  - Fourier Neural Operator (FNO)
    - Concept and Methodology
    - Generalizing the FNO [2, TMLR 2024]
  - Solving PDE-constrained control problem using operator learning [3, AAAI 2022]
  - FEONet: Data-free FEM-based neural operator [4, preprint 2023]
- **Conclusion**

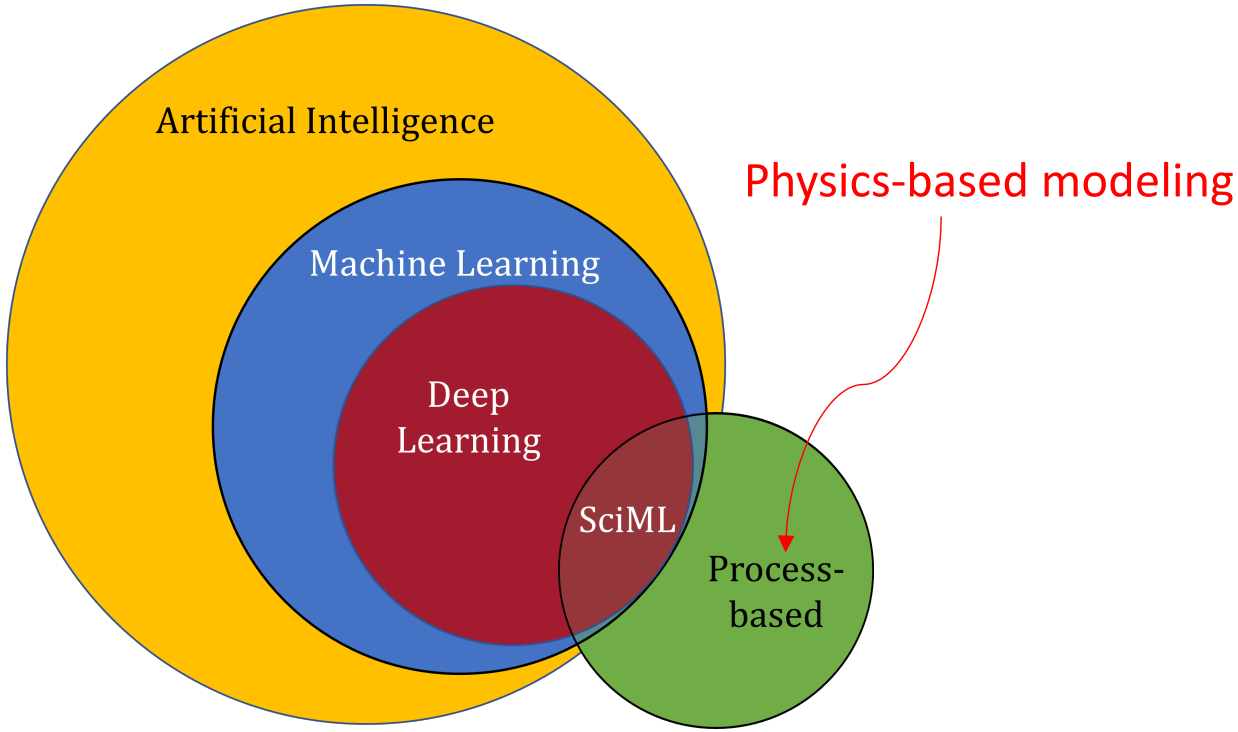
AI for Science (AI4Science)



- Accelerate scientific observation and discovery
- Interpret large datatsets
- Gain insights that might not have been possible using traditional scientific methods alone.

https://ai4sciencecommunity.github.io/

Scientific Machine Learning (SciML)

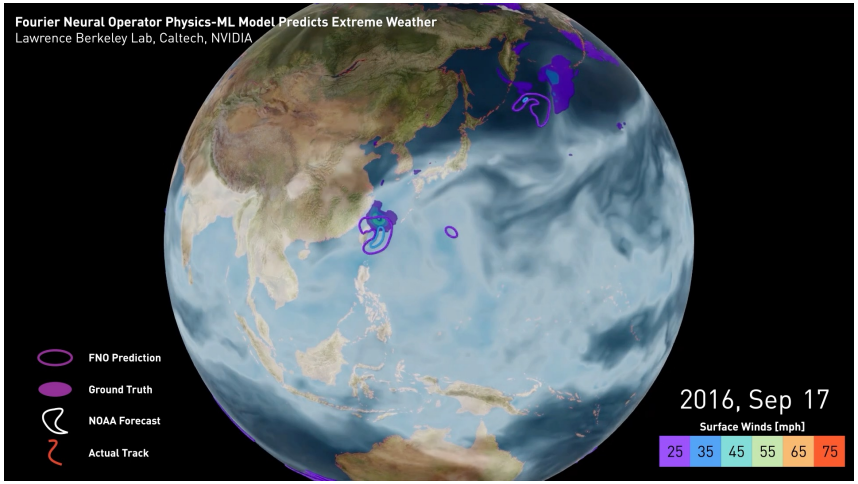


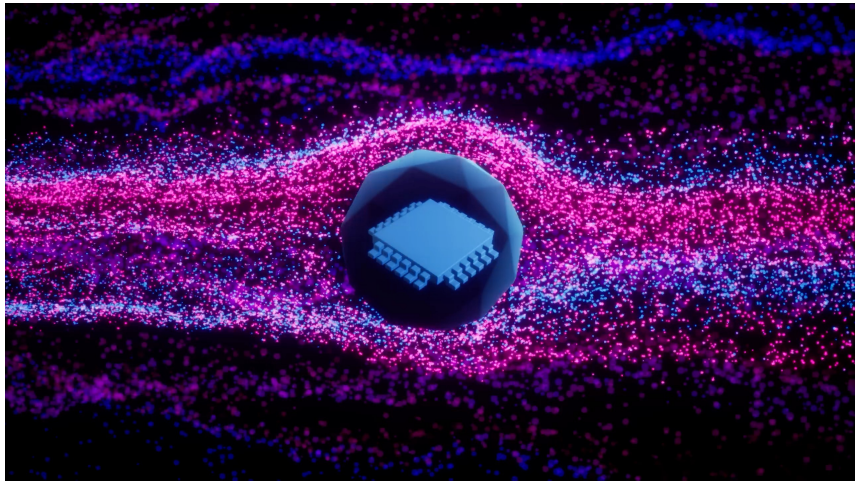
SciML = Machine learning + Science(Physics-based modeling)

- Physics-based models contain mathematical formula's that are based on scientific knowledge and physical laws.
- To build these models, observational studies and experiments were carried out.

https://sciml.wur.nl/reviews/sciml/sciml.html

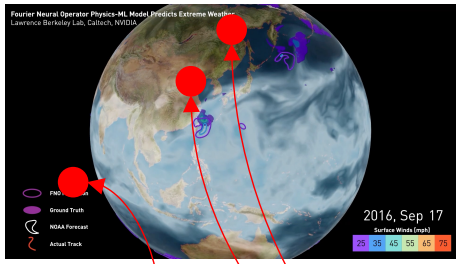
Scientific Machine Learning (SciML)



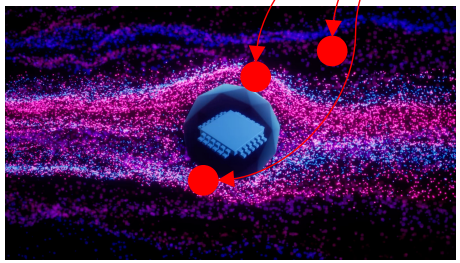


https://www.quantamagazine.org/latest-neural-nets-solve-worlds-hardest-equations-faster-than-ever-before-20210419/

Scientific Machine Learning (SciML)

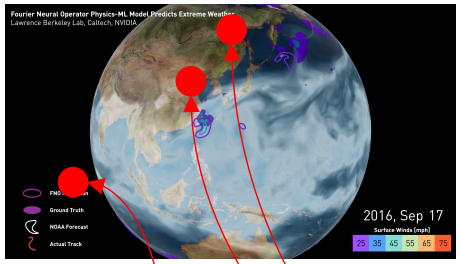


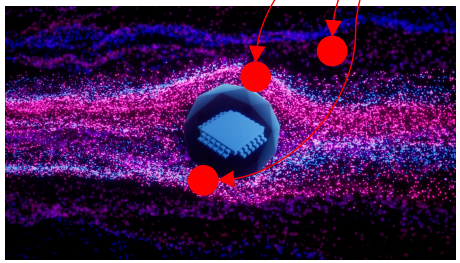
Sensors



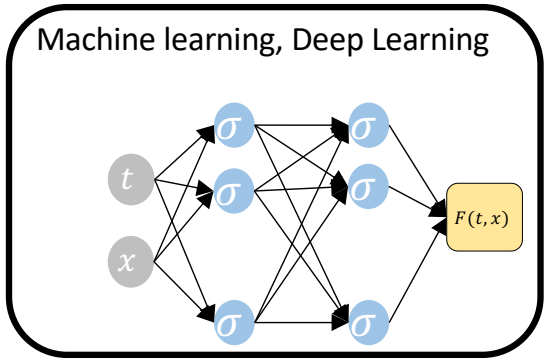
https://www.quantamagazine.org/latest-neural-nets-solve-worlds-hardest-equations-faster-than-ever-before-20210419/

Scientific Machine Learning (SciML)





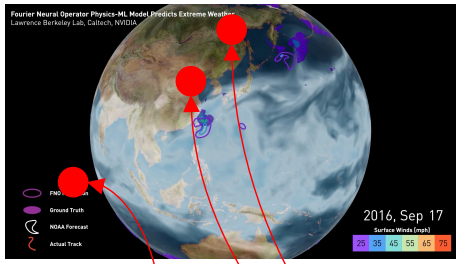
Sensors → **Data science**

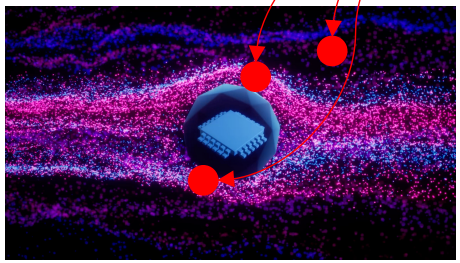


Understand and Predict

https://www.quantamagazine.org/latest-neural-nets-solve-worlds-hardest-equations-faster-than-ever-before-20210419/

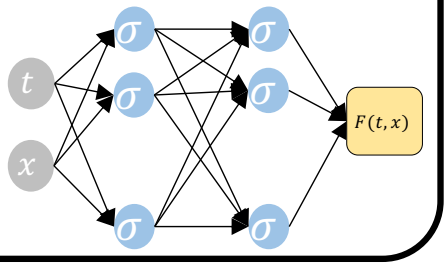
Scientific Machine Learning (SciML)





Sensors → **Data science**

Machine learning, Deep Learning



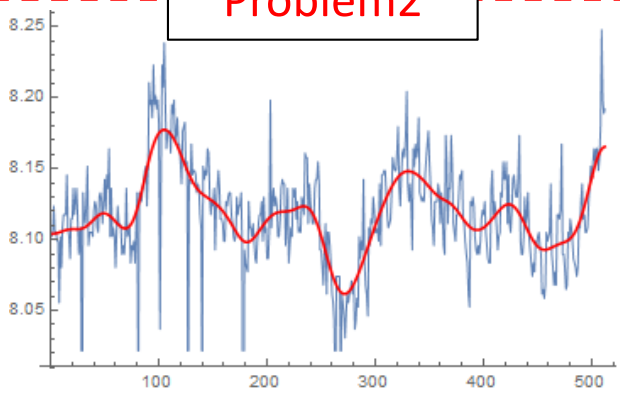
Understand and Predict

Problem1



Insufficient amount of data

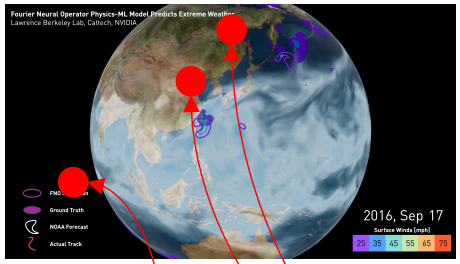
Problem2

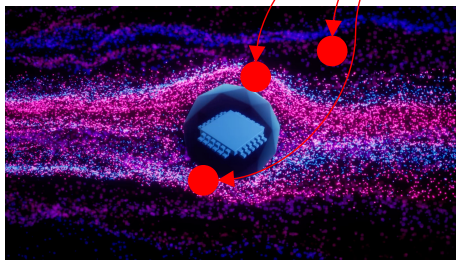


Noisy data

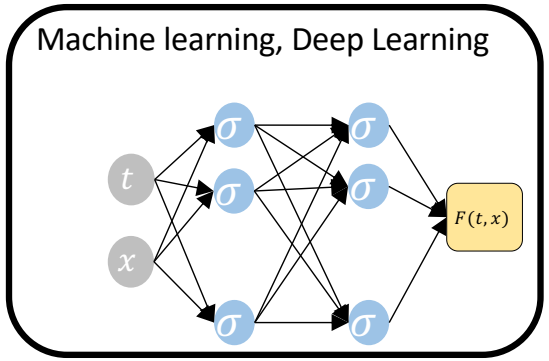
https://www.quantamagazine.org/latest-neural-nets-solve-worlds-hardest-equations-faster-than-ever-before-20210419/

Scientific Machine Learning (SciML)





Sensors → **Data science**

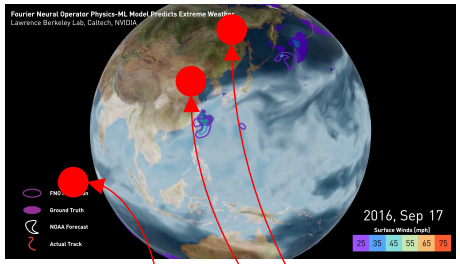


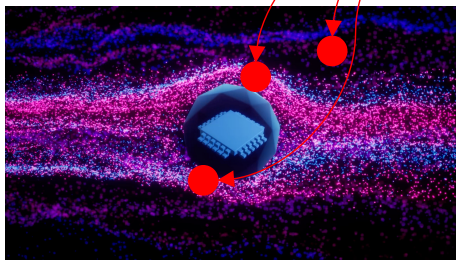
Understand and Predict

Physics-based modeling

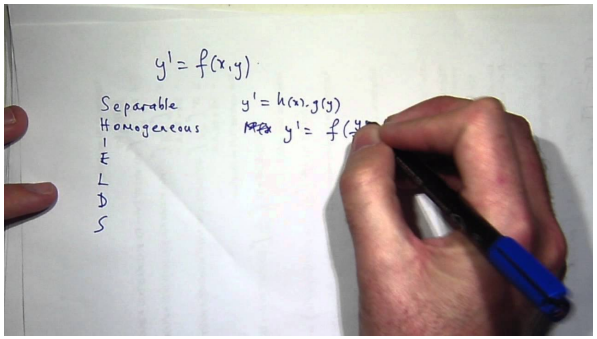
https://www.quantamagazine.org/latest-neural-nets-solve-worlds-hardest-equations-faster-than-ever-before-20210419/

Scientific Machine Learning (SciML)





Sensors → **Data science**



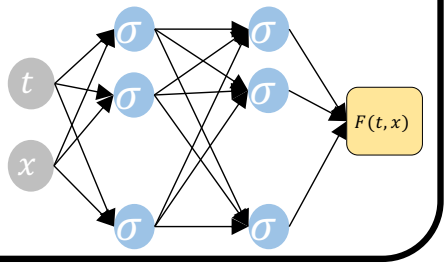
Physical and mathematical meaning

$u(t,x)$: Temperature at location x at time t.

→ Heat equation $\frac{\partial u(t,x)}{\partial t} = \frac{\partial^2 u(t,x)}{\partial x^2}$

Partial **D**ifferential **E**quation (**PDE**)

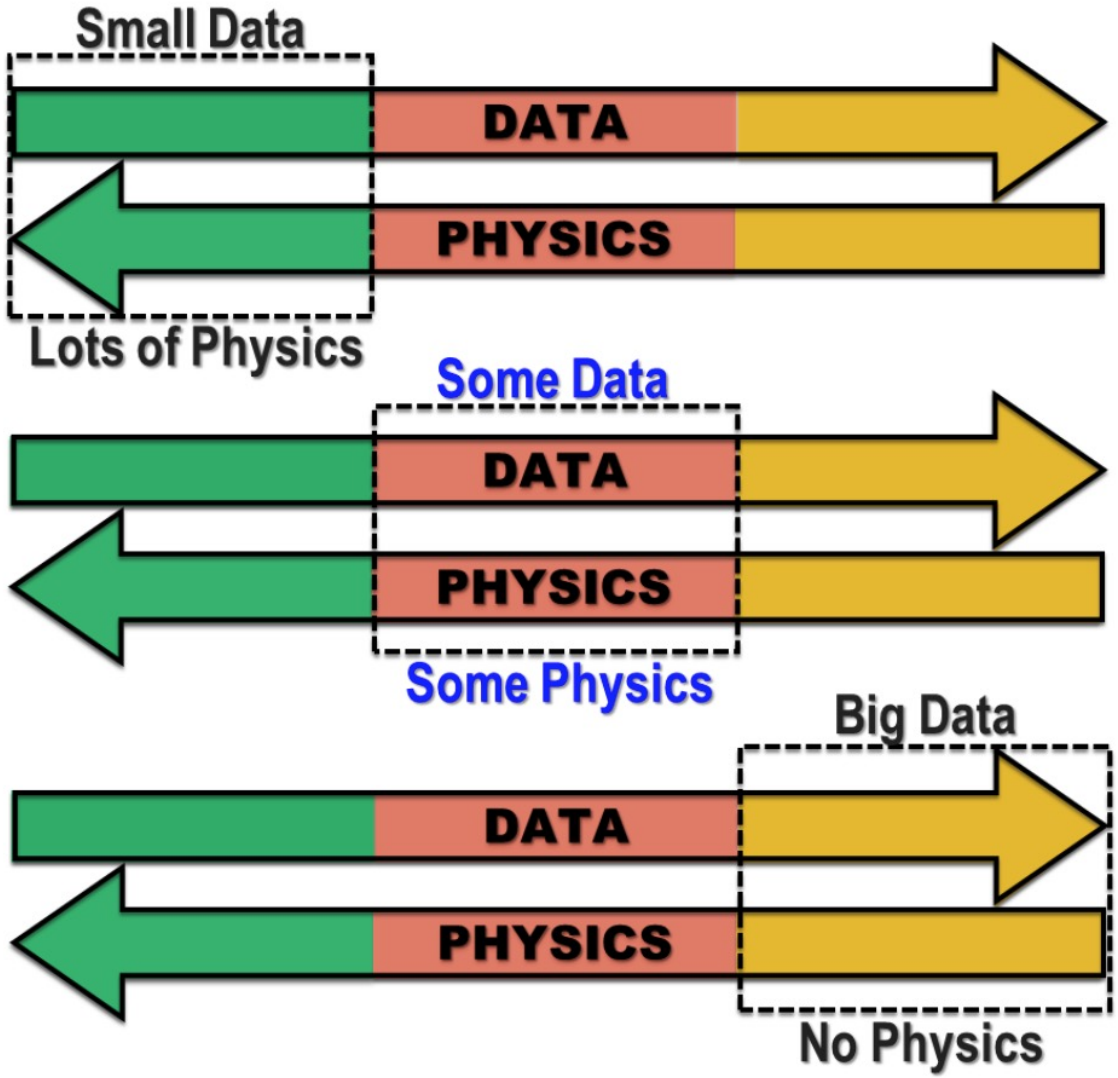
Machine learning, Deep Learning



Understand and Predict

https://www.quantamagazine.org/latest-neural-nets-solve-worlds-hardest-equations-faster-than-ever-before-20210419/

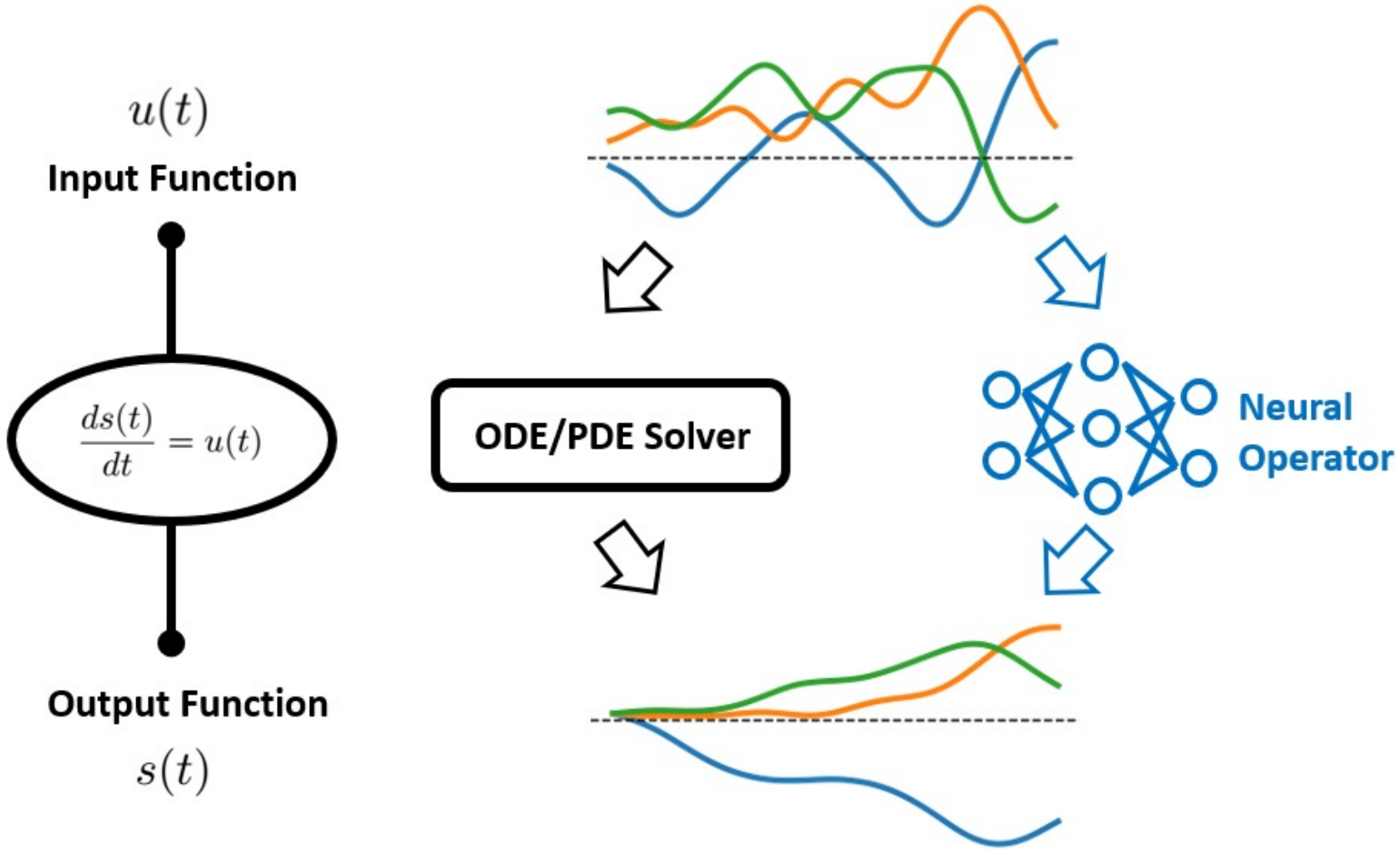
Scientific Machine Learning (SciML)



Yang, Liu, Dongkun Zhang, and George Em Karniadakis. "Physics-informed generative adversarial networks for stochastic differential equations." *SIAM Journal on Scientific Computing* 42.1 (2020): A292-A317.

What is Neural Operator for real-time solutions to PDEs

- Learning a mapping from the parameters (e.g. external force, initial, and boundary conditions) of the PDEs to the corresponding solution.



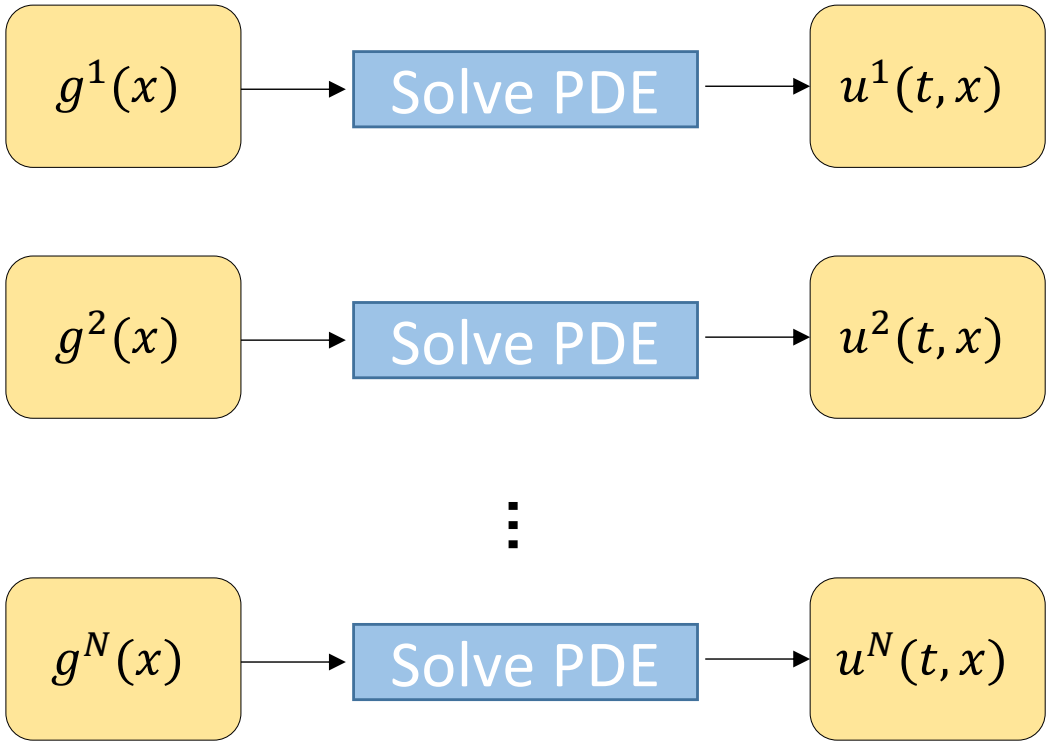
https://towardsdatascience.com/operator-learning-via-physics-informed-deeponet-lets-implement-it-from-scratch-6659f3179887

What is Neural Operator for real-time solutions to PDEs

- Traditional numerical methods had to **repeat** similar calculations for each input function to derive its output function.

Find a map $\boldsymbol{\mathcal{G}}: \boldsymbol{g(x)} \mapsto \boldsymbol{u(t,x)}$ satisfying

$$\mathcal{L}_{PDE} = f(u, u_t, u_x, u_{xx}, \dots) = g(x)$$
$$\mathcal{L}_{IC} = u(0,x) - k(x) = 0$$
$$\mathcal{L}_{BC} = u|_{\partial\Omega} - h(t,x)|_{\partial\Omega}$$

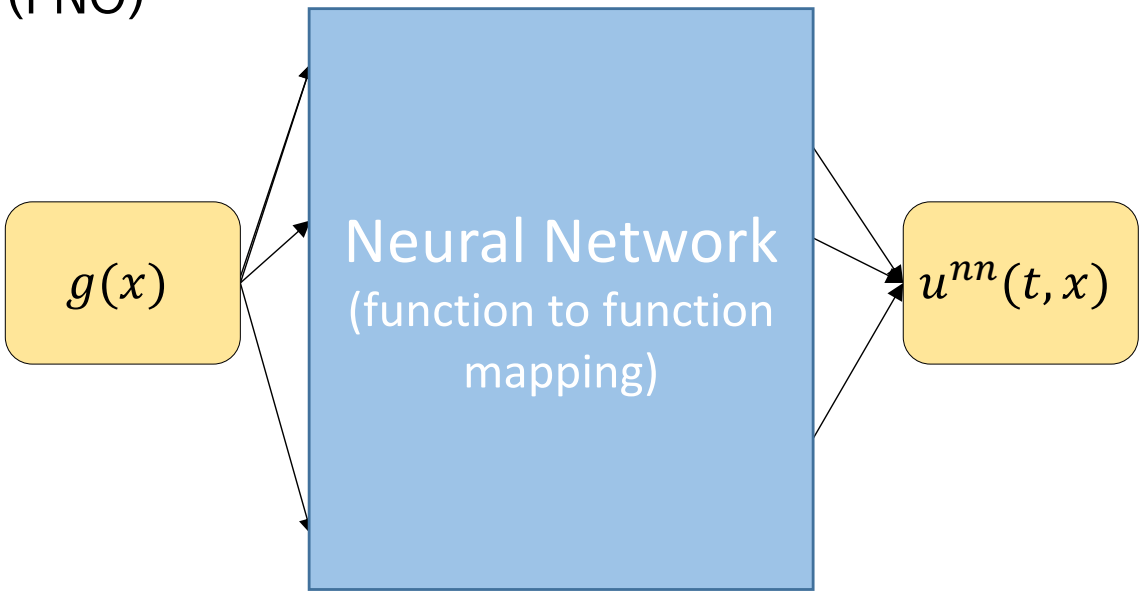


What is Neural Operator for real-time solutions to PDEs

- **Deep learning-based studies** have been emerged to learn the operator from input function to the output function.
- We **need a dataset** $\{\boldsymbol{g^i(x)}, \boldsymbol{u^i(t,x)}\}_{\boldsymbol{i=1}}^{\boldsymbol{N}}$.
- Predict $u^{new}(t,x)$ for a new $g^{new}(x)$ in a real-time.
- Models
  - Deep Operator Network (DeepONet)
  - Fourier Neural Operator (FNO)

Find a map $\boldsymbol{\mathcal{G}}: \boldsymbol{g(x)} \mapsto \boldsymbol{u(t,x)}$ satisfying

$$\mathcal{L}_{PDE} = f(u, u_t, u_x, u_{xx}, \dots) = g(x)$$
$$\mathcal{L}_{IC} = u(0,x) - k(x) = 0$$
$$\mathcal{L}_{BC} = u|_{\partial\Omega} - h(t,x)|_{\partial\Omega}$$

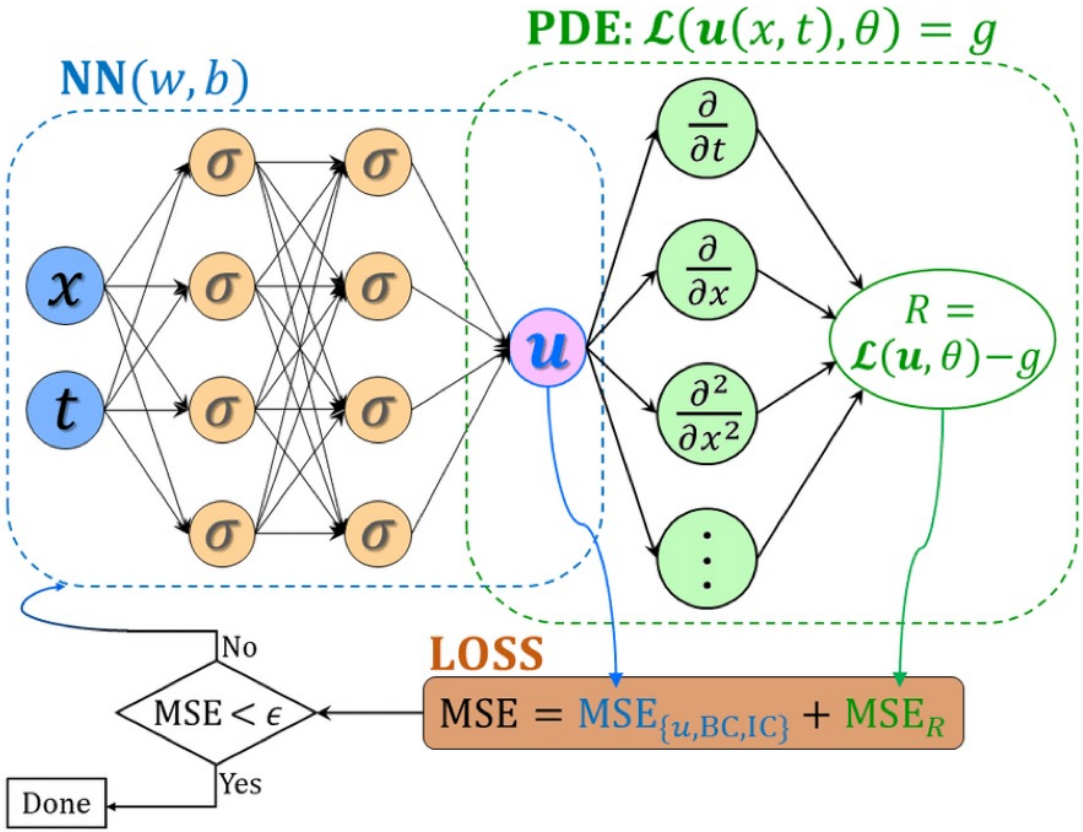


Physics-informed neural network (PINN)

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computation al physics*, *378*, 686-707.

- **Find a solution $u(t,x)$ satisfying**
- $\mathcal{L}(u, u_t, u_x, u_{xx}, \dots) = g(x)$
- $\mathcal{L}_{IC} = u(0,x) - k(x) = 0$
- $\mathcal{L}_{BC} = u|_{\partial\Omega} - h(t,x)|_{\partial\Omega}$

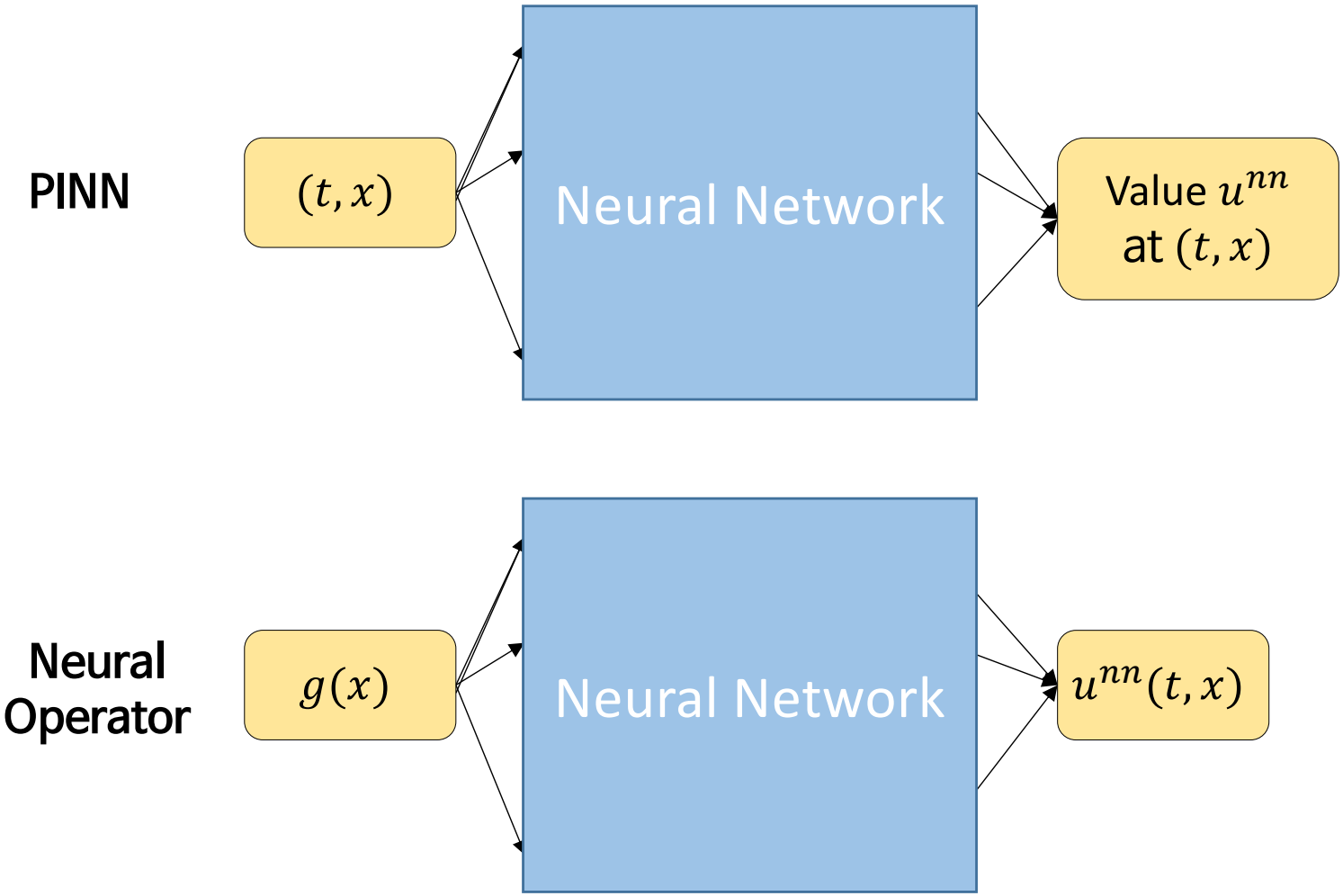
[Schematic of a PINN]



Lu, L., Meng, X., Mao, Z., & Karniadakis, G. E. (2021). DeepXDE: A deep learning library for solving differential equations. *SIAM Review*, *63*(1), 208-228

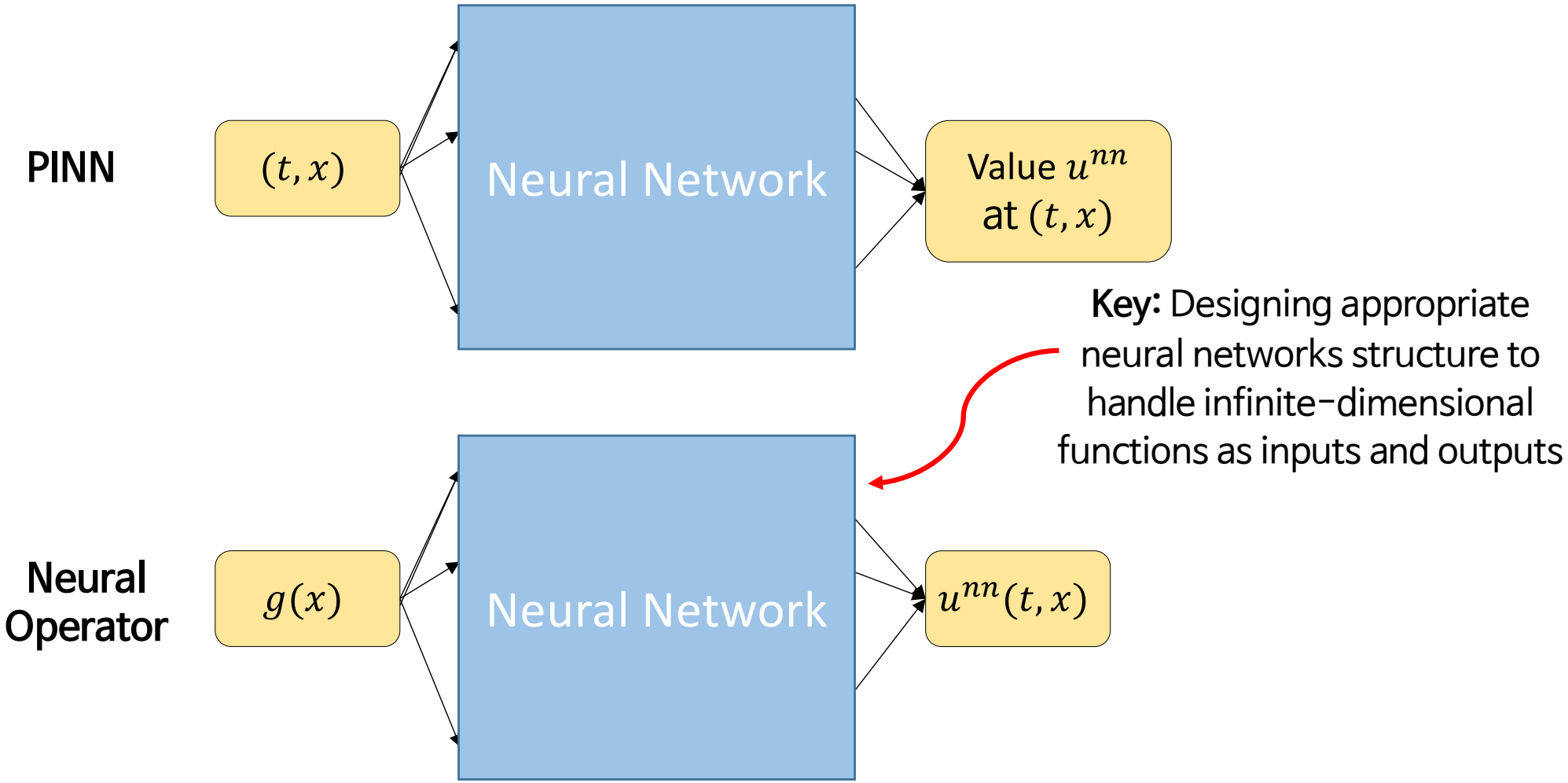
Comparison between PINN and Neural Operator

- Compared to Physics-Informed Neural Network (PINN)



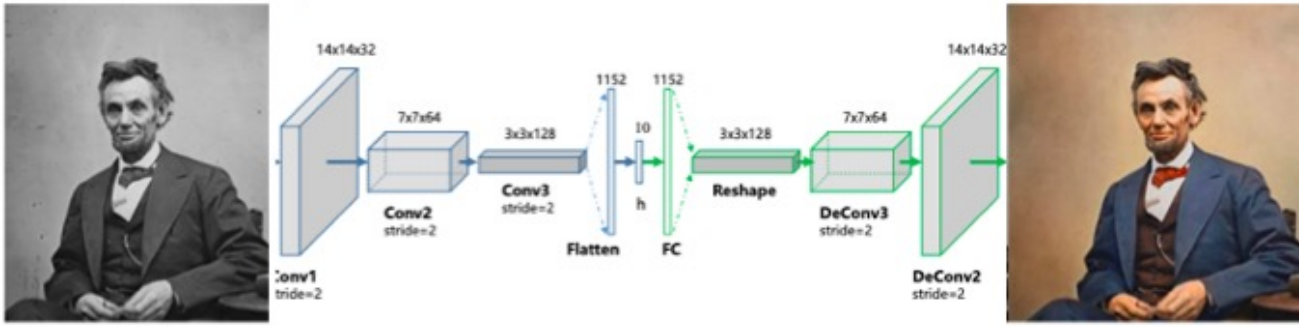
Comparison between PINN and Neural Operator

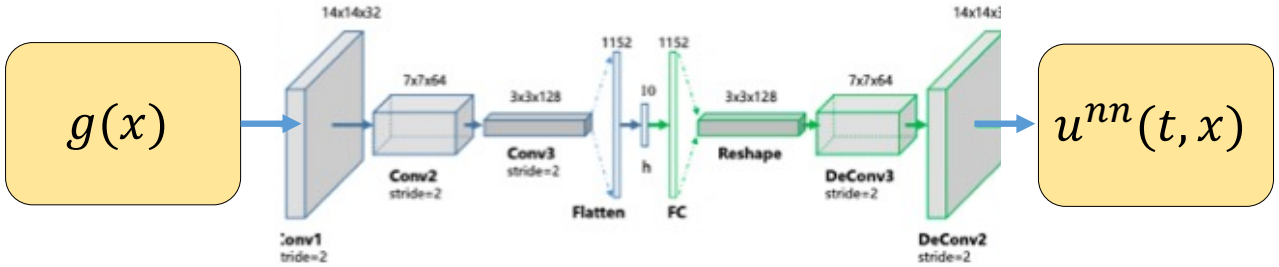
- Compared to Physics-Informed Neural Network (PINN)



Natural approach : image-to-image regression

- **Convolutional Neural Network encoder-decoder (Image-to-image regression)**
  - Image segmentation, Image Noise Reduction, Image coloring

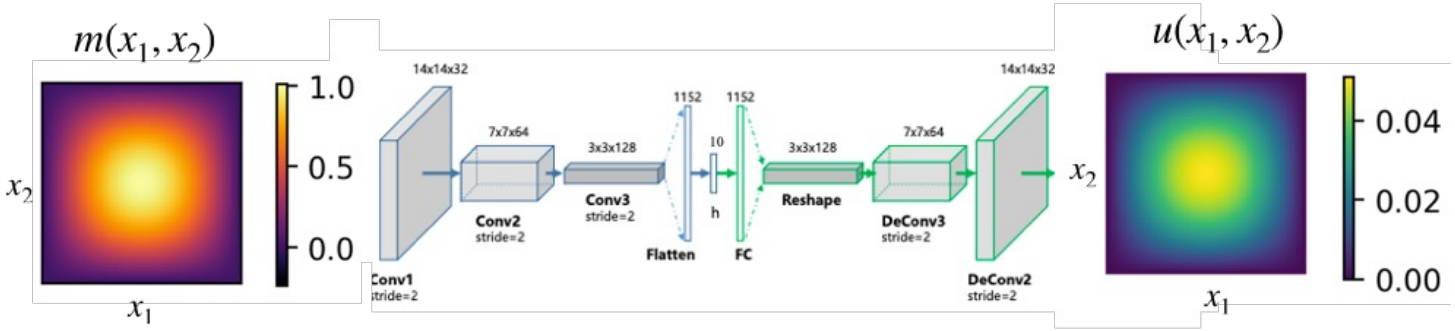




https://towardsdatascience.com/convolutional-autoencoders-for-image-noise-reduction-32fce9fc1763

Natural approach : image-to-image regression

- **Operator learning (function-to-function regression)**



Ex) Poisson equation

$$-\Delta u - m = 0 \text{ in } \Omega \subset \mathbb{R}^2$$
$$u = 0 \text{ in } \partial\Omega$$

Goal) Find a solution $u(x_1, x_2)$ for different source term $m(x_1, x_2)$.
(Learning operator from the source term to the solution)

$$m(x_1, x_2) \mapsto u(x_1, x_2)$$

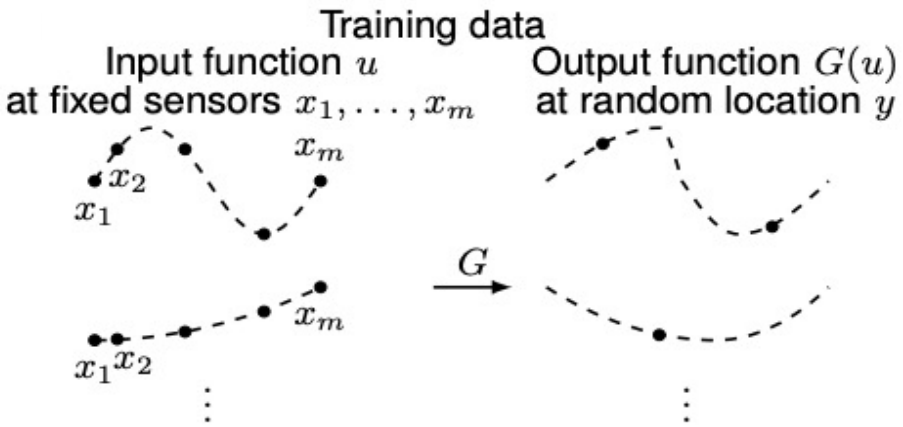
- We can consider the functions m and u as 2d images $m(x_1, x_2)$ and $u(x_1, x_2)$.
- Limitation
  - Only when input function values are known in a uniform grid.
  - Only when all input function values are known in all grids.

Deep Operator Network (DeepONet)

Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021). Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature machine intelligence*, *3*(3), 218-229.

Learning an operator $G: u(x) \mapsto G(u)(y)$ using the network

Data

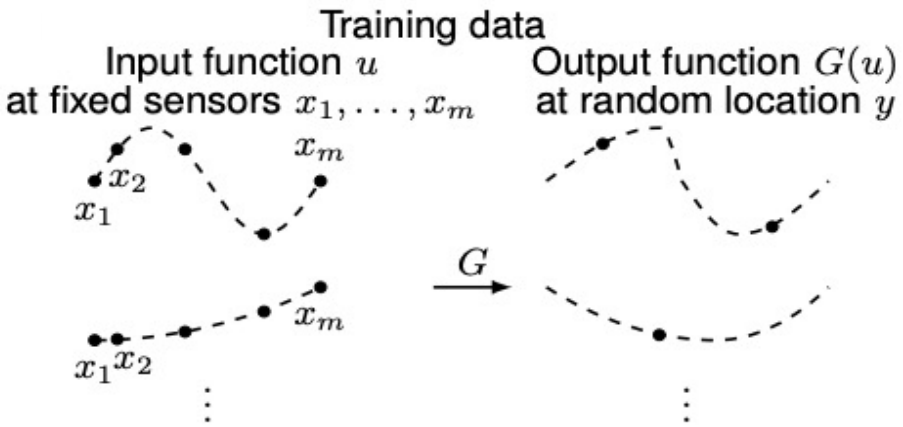


Deep Operator Network (DeepONet)

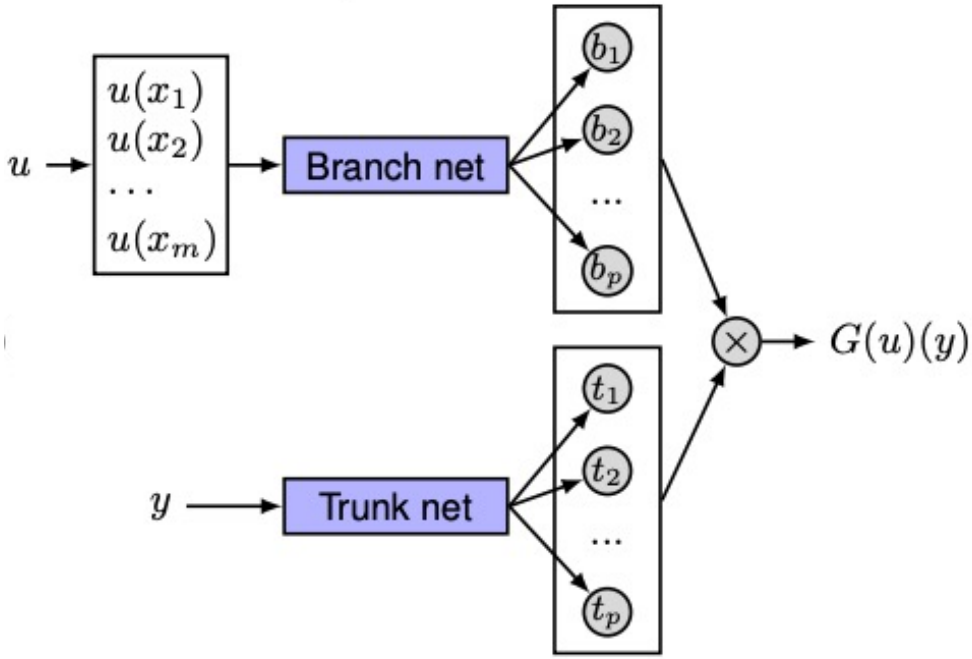
Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021). Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature machine intelligence*, *3*(3), 218-229.

Learning an operator $G: u(x) \mapsto G(u)(y)$ using the network

Data



Model



Deep Operator Network (DeepONet)

Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021). Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature machine intelligence*, *3*(3), 218-229.

Learning an operator $G: u(x) \mapsto G(u)(y)$ using the network

[Chen and Chen, 1995]

Theorem 1 (Universal Approximation Theorem for Operator). *Suppose that σ is a continuous non-polynomial function, X is a Banach Space, $K_1 \subset X$, $K_2 \subset \mathbb{R}^d$ are two compact sets in X and $\mathbb{R}^d$, respectively, V is a compact set in $C(K_1)$, G is a nonlinear continuous operator, which maps V into $C(K_2)$. Then for any $\epsilon > 0$, there are positive integers n, p, m, constants $c_i^k, \xi_{ij}^k, \theta_i^k, \zeta_k \in \mathbb{R}$, $w_k \in \mathbb{R}^d$, $x_j \in K_1$, $i = 1, \ldots, n$, $k = 1, \ldots, p$, $j = 1, \ldots, m$, such that*

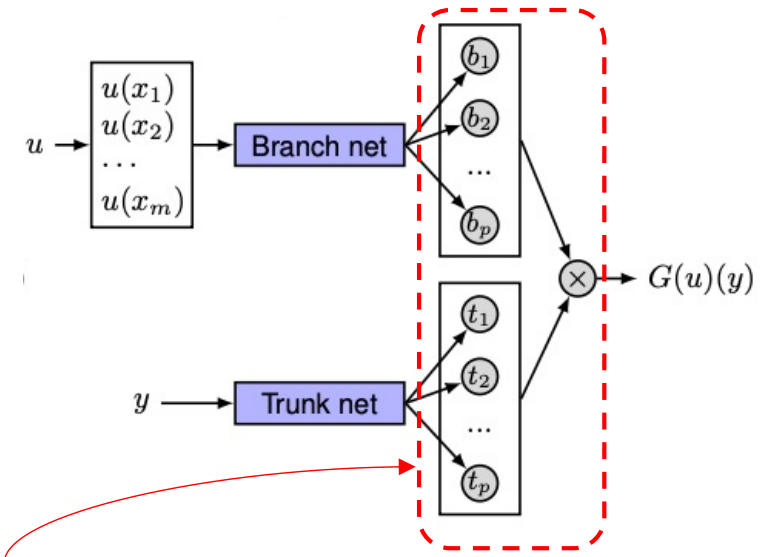
$$\left| G(u)(y) - \sum_{k=1}^{p} \underbrace{\sum_{i=1}^{n} c_i^k \sigma \left(\sum_{j=1}^{m} \xi_{ij}^k u(x_j) + \theta_i^k \right)}_{branch} \underbrace{\sigma(w_k \cdot y + \zeta_k)}_{trunk} \right| < \epsilon \tag{1}$$

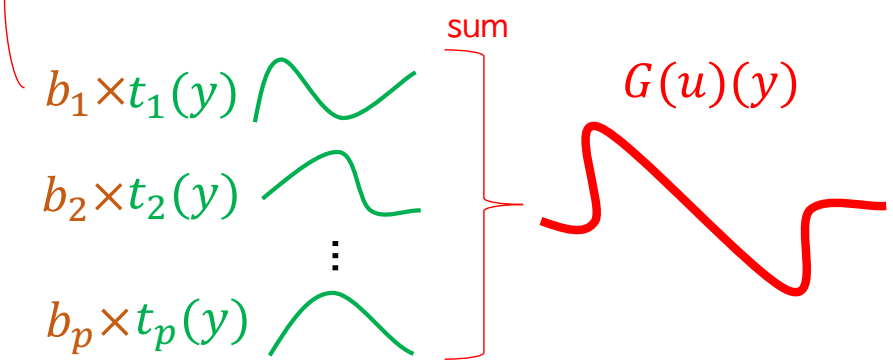
holds for all $u \in V$ and $y \in K_2$.

Understanding of DeepONet: linear approximation

Learning an operator $G: u(x) \mapsto G(u)(y)$ using the network

Model

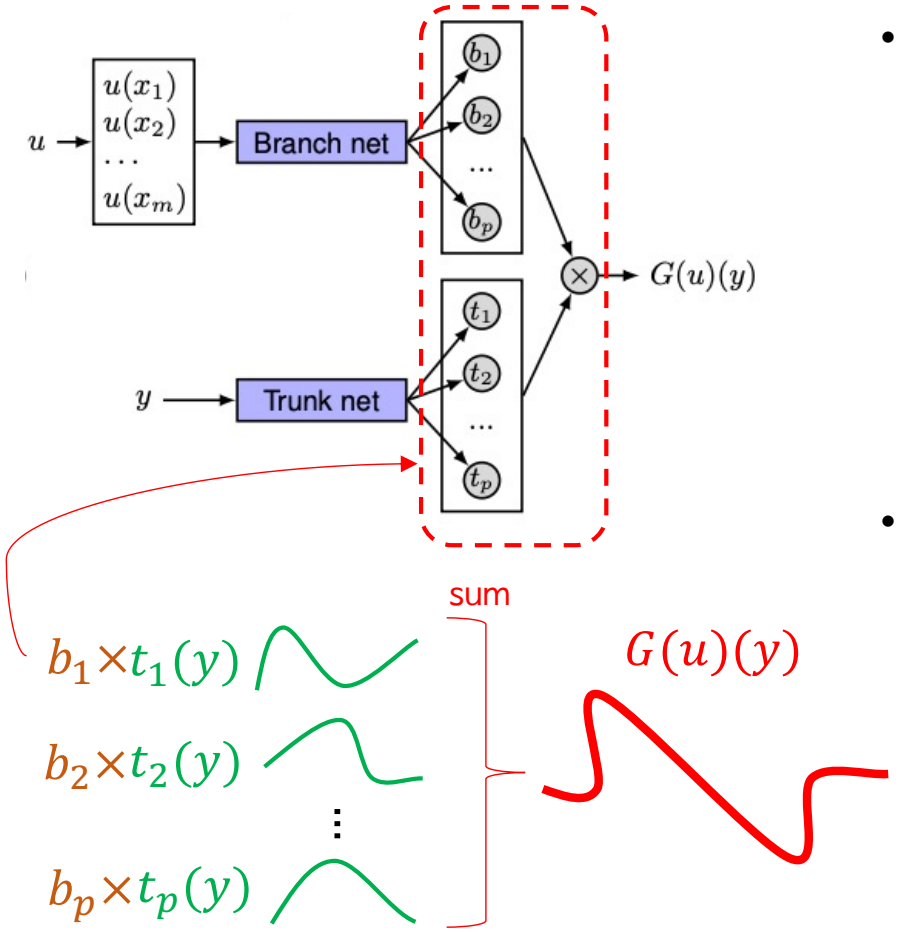




Understanding of DeepONet: linear approximation

Learning an operator $G: u(x) \mapsto G(u)(y)$ using the network

Model



- It approximates the target function $G(u)(y)$ as

$$G(u)(y) \approx b_1 t_1 + b_2 t_2 + \cdots + b_p t_p$$

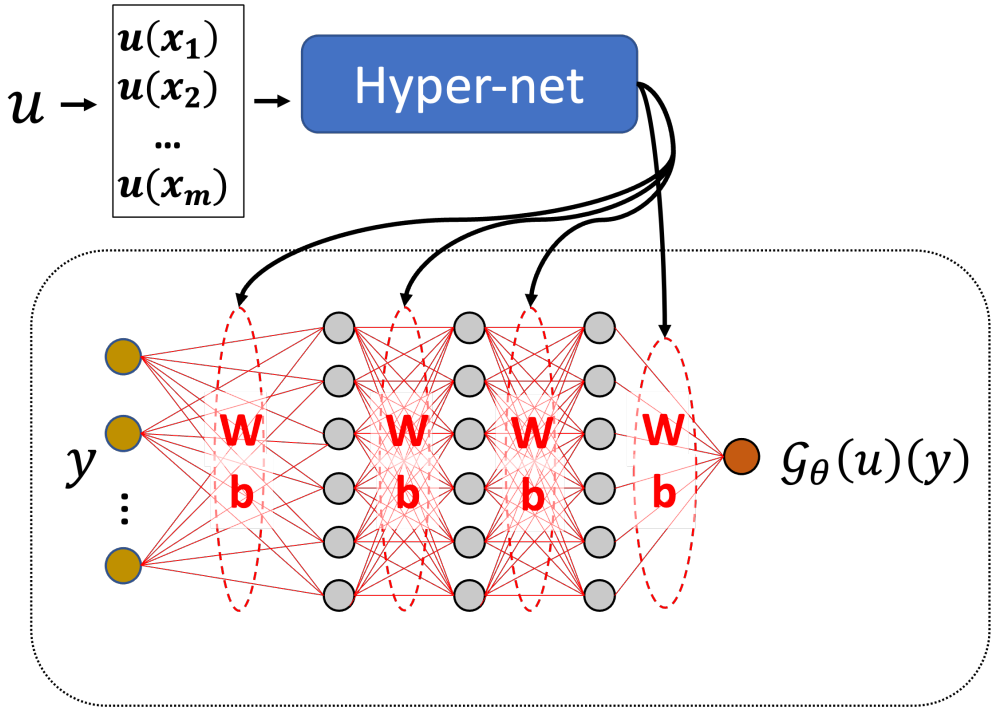
  - Here, $t_1(y), t_2(y), \ldots, t_p(y)$ are the **p-basis functions** for the target function space domain $\mathcal{Y}$ $(y \in \mathcal{Y})$.
  - $b_1, b_2, \ldots, b_p$ are the **p-coefficient values** for the target function $G(u)$ (depends on the input function u)

- The linear approximation of the DeepONet has limitations when approximating discontinuous or sharp **target function.**
  - The DeepONet needs large number of parameters (complexity).

Proposed model: HyperDeepONet [1, ICLR 2023]

Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021). Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature machine intelligence*, *3*(3), 218-229.

- We propose a general model **HyperDeepONet**, which contains all the previous models, as a special case of the HyperDeepONet.
- The **hypernetwork** generates all parameters of the target network.



Target network

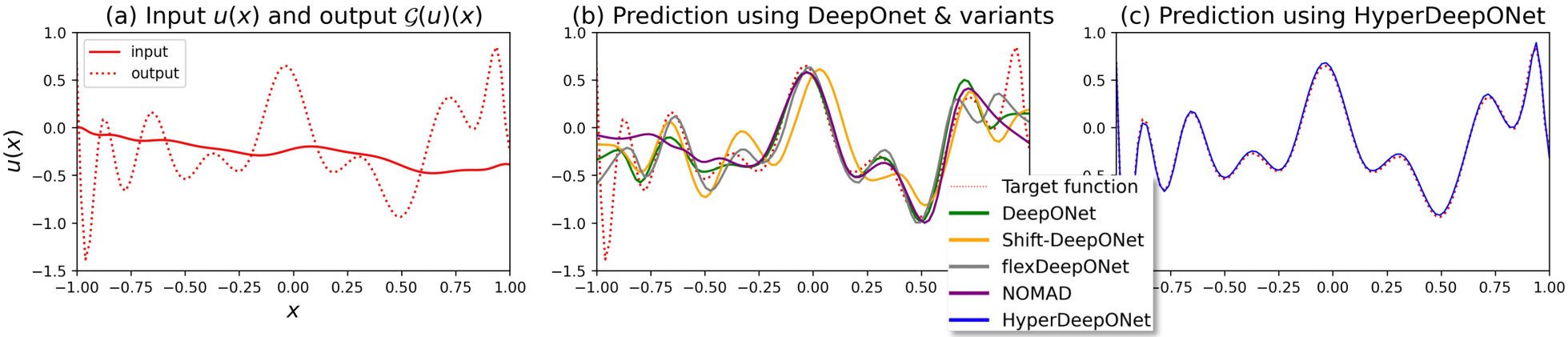
HyperDeepONet (Our model)

Results [1, ICLR 2023]

Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021). Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature machine intelligence*, *3*(3), 218-229.

Toy example

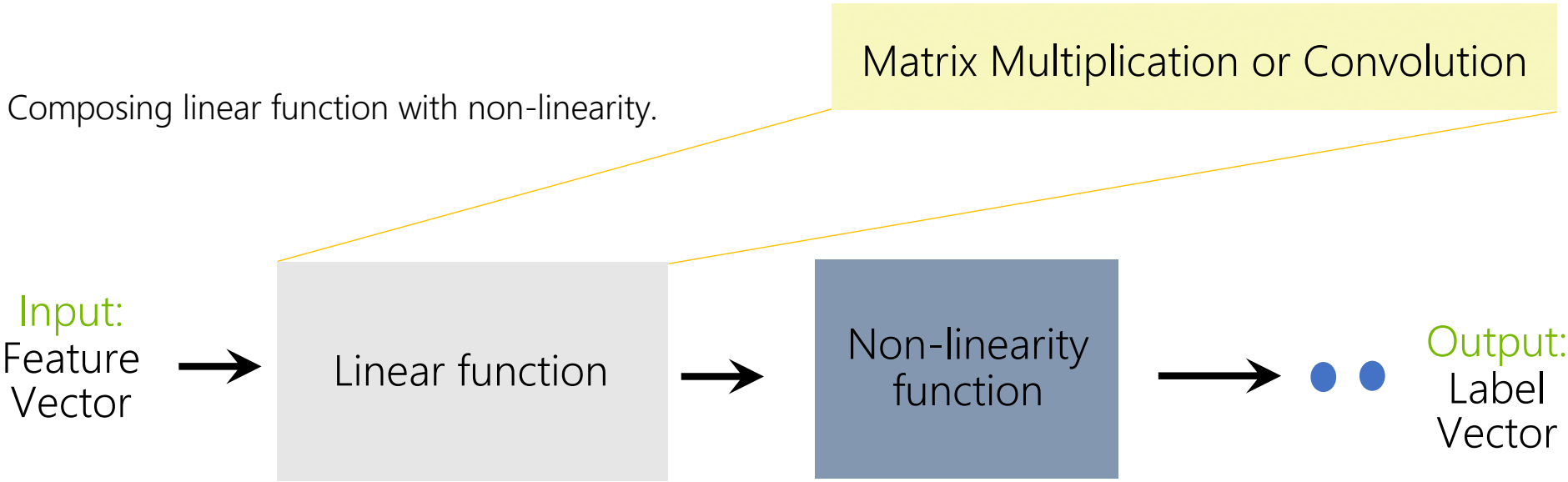
- $G: u(x) \mapsto \frac{d}{dx} u(x)$ ($u \sim$ **Chebyshev degree 20**)
- **We choose the differentiation operator to focus on operator learning when the operator's output function space is complicated.**



Fourier Neural Operator

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2020). Fourier neural operator for parametric partial di fferential equations. *arXiv preprint arXiv:2010.08895*.

Neural networks -> Learning in fixed dimensions

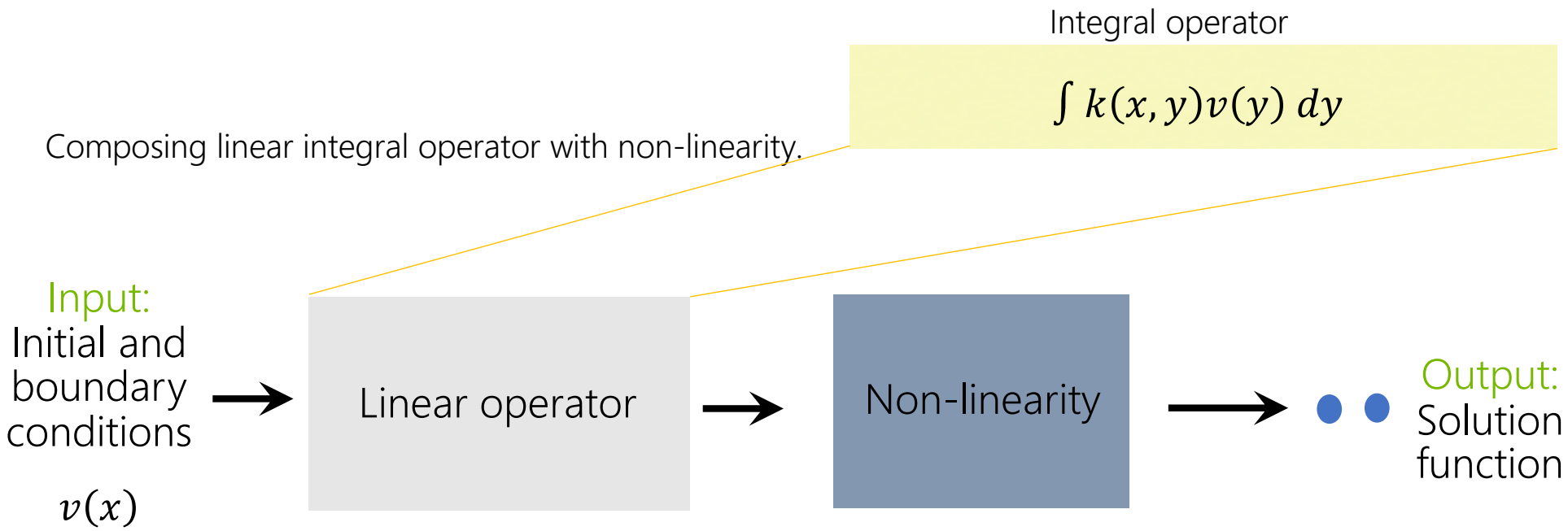


Neural Network can approximate any continuous function.

Fourier Neural Operator

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2020). Fourier neural operator for parametric partial di fferential equations. *arXiv preprint arXiv:2010.08895*.

Neural operator -> Learning in infinite dimensions



Neural Operator can approximate any continuous operator

Fourier Neural Operator

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2020). Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*.

- **Neural Operator - Iterative Kernel Integration**

$$v_{t+1}(x) = \sigma\left(W_\theta v_t(x) + \boldsymbol{K_\phi v_t(x)}\right)$$

where $\boldsymbol{K_\phi v_t(x)} = \int_D \boldsymbol{\kappa_\phi(x,y) v_t(y) dy} \approx \boldsymbol{\mathcal{F}^{-1}\left[\mathcal{F}[\kappa_\phi]\mathcal{F}[v_t]\right]}$.

$\boldsymbol{= R_\phi}$

Fourier Neural Operator

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2020). Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*.

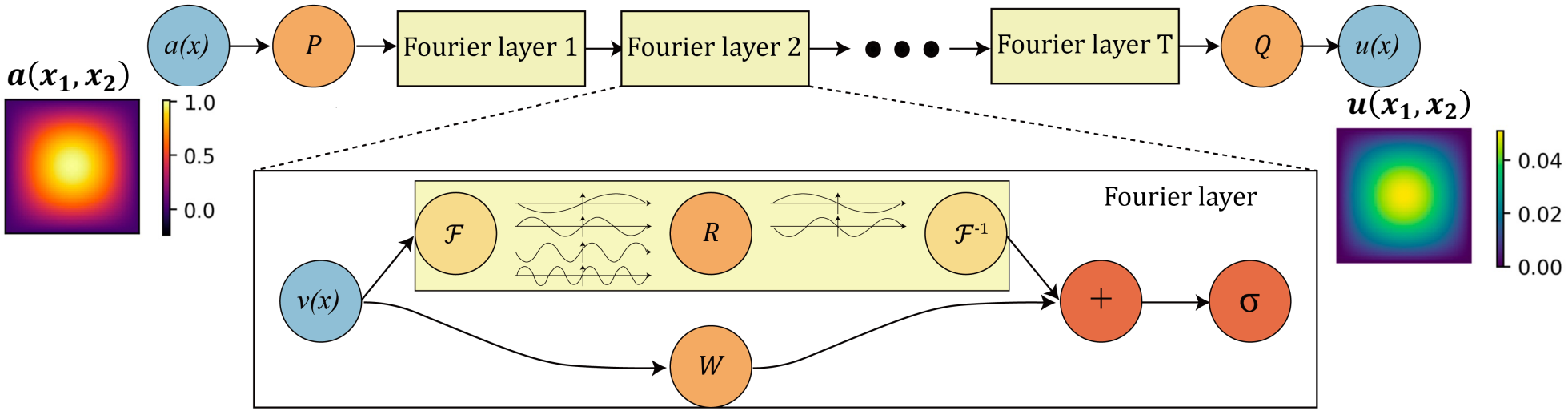
- **Neural Operator - Iterative Kernel Integration**

$$v_{t+1}(x) = \sigma\left(W_\theta v_t(x) + \boldsymbol{K_\phi v_t(x)}\right)$$

where $\boldsymbol{K_\phi v_t(x)} = \int_D \boldsymbol{\kappa_\phi(x,y) v_t(y) dy} \approx \boldsymbol{\mathcal{F}^{-1}\left[\mathcal{F}[\kappa_\phi]\mathcal{F}[v_t]\right]}$.

$\boldsymbol{= R_\phi}$

Learning an operator $\mathcal{G}: a(x) \mapsto \mathcal{G}(a)(x) = u(x)$ using the network

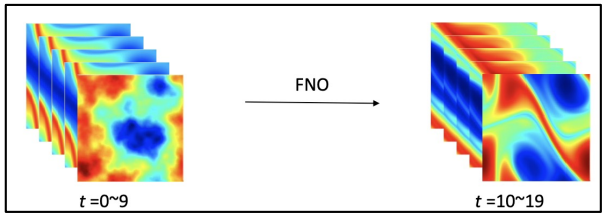


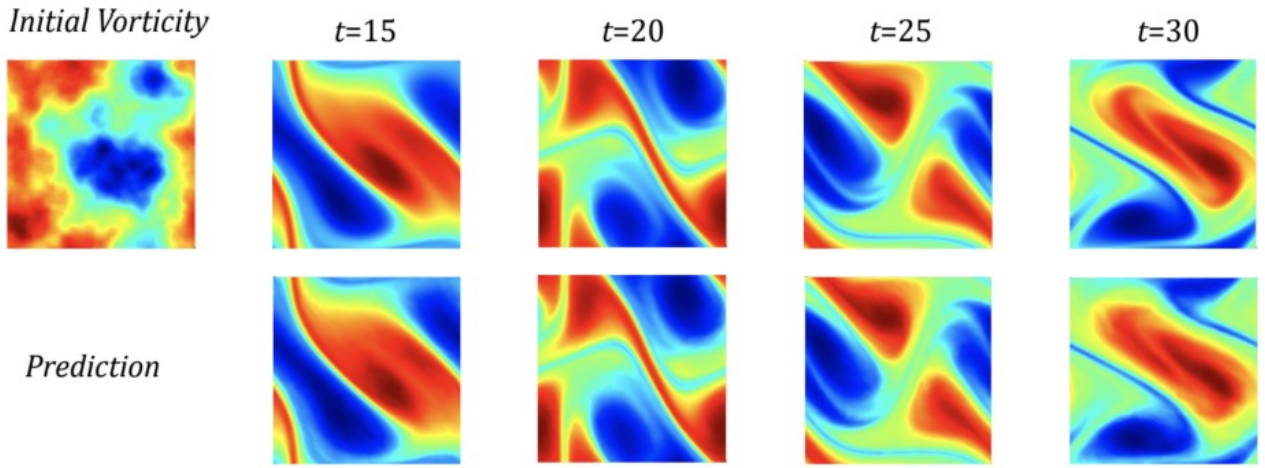
Fourier Neural Operator

(ex) 2-D Navier-Stokes equation (viscous, incompressible fluid in vorticity form)

$$\begin{cases} \partial_t w(x,t) + u(x,t)\cdot\nabla w(x,t) = \nu\Delta w(x,t) + f(x), & x \in (0,1)^2 \\ \nabla\cdot u(x,t) = 0, & t \in (0,T] \\ w(x,0) = w_0(x), & \\ w(x,t) = \nabla\times u(x,t), & \end{cases}$$

Learning an operator $w|_{(0,1)^2\times[0,9]} \mapsto w|_{(0,1)^2\times[10,T]}$





Zero-shot super-resolution: Navier-Stokes Equation with viscosity $\nu = 1e{-}4$; Ground truth on top and prediction on bottom; trained on $64 \times 64 \times 20$ dataset; evaluated on $256 \times 256 \times 80$ (see Section 5.4).

Generalize FNO [2, TMLR 2024]

Shin, J. Y., **Lee, J. Y.**, & Hwang, H. J. Pseudo-Differential Neural Operator: Generalize Fourier Neural operator for Learning Solution Operators of Partia l Differential Equations. Transactions on Machine Learning Research.

Fourier Neural Operator

$$K_{FNO}v_t(x) = \mathcal{F}^{-1}\left(R_\phi(\xi)\cdot(\mathcal{F}v_t)\right)(x)$$

(Ours) Pseudo-differential integral operator

$$K_{PDIO}v_t(x) = \mathcal{F}^{-1}(a_\theta^{nn}(x,\xi)\cdot(\mathcal{F}v_t))(x)$$

- **Operator learning based on the theory of Pseudo-differential operator**
  - Pseudo-differential operator

$$T_a(f) = \int_{\mathbb{R}^n} a(x,\xi)\hat{f}(\xi)e^{2\pi i\xi x}d\xi$$

  - (Ours) Pseudo-differential integral operator (PDIO) with $a_\theta^{nn}(x,\xi) = a_{\theta_1}^{nn}(x)a_{\theta_2}^{nn}(\xi)$

$$T_a(f) \approx T_{a_\theta}^{nn} = \int_{\mathbb{R}^n} a_\theta^{nn}(x,\xi)\hat{f}(\xi)e^{2\pi i\xi x}d\xi = \mathcal{F}^{-1}[a_{\theta_1}^{nn}(x)a_{\theta_2}^{nn}(\xi)\mathcal{F}[f](\xi)]$$

=⟩ Generalization of the Fourier Neural Operator (FNO)

Generalize FNO [2, TMLR 2024]

Shin, J. Y., **Lee, J. Y.**, & Hwang, H. J. Pseudo-Differential Neural Operator: Generalize Fourier Neural operator for Learning Solution Operators of Partia l Differential Equations. Transactions on Machine Learning Research.

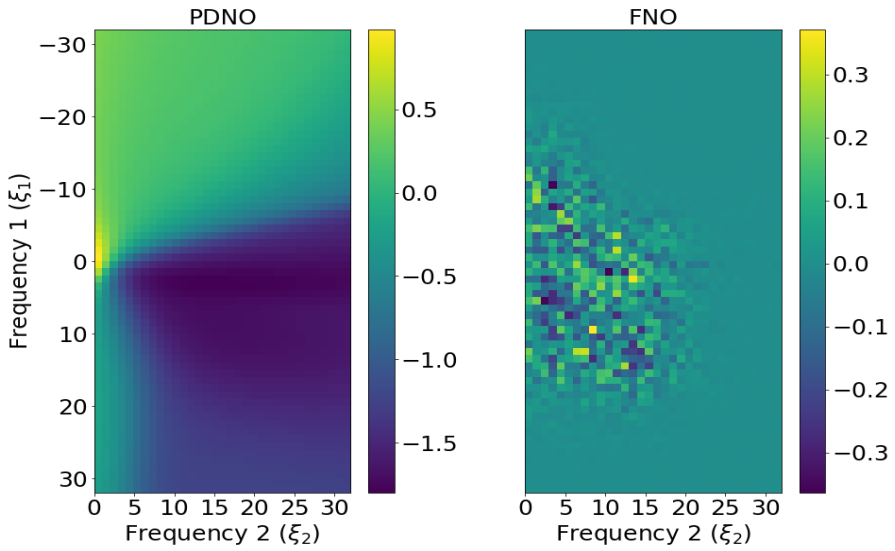
Fourier Neural Operator

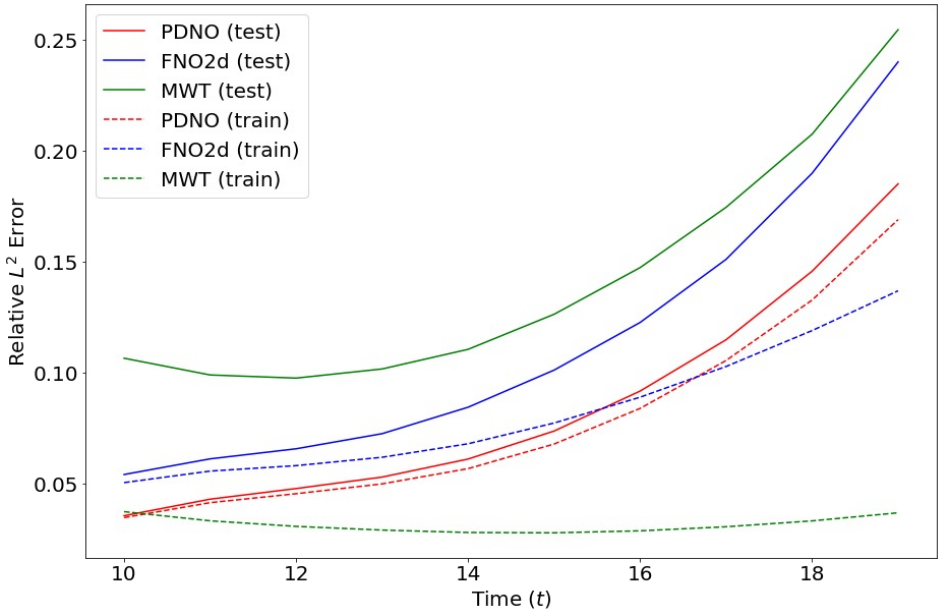
$$K_{FNO}v_t(x) = \mathcal{F}^{-1}\left(R_\phi \cdot (\mathcal{F}v_t)\right)(x)$$

(Ours) Pseudo-differential integral operator

$$K_{PDIO}v_t(x) = \mathcal{F}^{-1}(a_\theta^{nn}(x,\xi) \cdot (\mathcal{F}v_t))(x)$$

$a_\theta^{nn}(\xi)$ $R_\phi(\xi)$





The parameter $a_\theta^{nn}(\xi)$ is smooth!

This prevent overfitting of the FNO.

DeepONet vs FNO

Lu, L., Meng, X., Cai, S., Mao, Z., Goswami, S., Zhang, Z., & Karniadakis, G. E. (2022). A comprehensive and fair comparison of two neural operators (wi th practical extensions) based on fair data. *Computer Methods in Applied Mechanics and Engineering*, *393*, 114778.

	DeepONet	FNO
Input domain D' & Output domain D'	Arbitrary	Cuboid, $D = D'$
Discretization of output function u	No	Yes
Mesh	Arbitrary	Grid
Prediction location	Arbitrary	Grid points
Full field observation data	No	Yes

Table 1: **Comparison between vanilla DeepONet and vanilla FNO.**

Solving PDE-constrained control problem using operator learning [3, AAAI 2022]

Hwang, R., **Lee, J. Y.**, Shin, J. Y., & Hwang, H. J. (2022, June). Solving pde-constrained control problems using operator learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 36, No. 4, pp. 4504-4512).

- **Solving control problems using operator learning.**

Poisson equation with zero Dirichlet boundary conditions

$$\begin{cases} -\Delta u(x) - m(x) = 0, & x \in \Omega, \\ u = 0, & x \in \partial\Omega, \\ m_a \le m \le m_b, & x \in \Omega, \end{cases}$$

- Ω : domain of interest
- $u : \Omega \to \mathbb{R}$ (unknown temperature)
- $u_d : \Omega \to \mathbb{R}$ (desired temperature)
- α : (unknown temperature)
- $m : \Omega \to \mathbb{R}$ (source term : control function)

Solving PDE-constrained control problem using operator learning [3, AAAI 2022]

Hwang, R., **Lee, J. Y.**, Shin, J. Y., & Hwang, H. J. (2022, June). Solving pde-constrained control problems using operator learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 36, No. 4, pp. 4504-4512).

- **Solving control problems using operator learning.**

Poisson equation with zero Dirichlet boundary conditions

$$\begin{cases} -\Delta u(x) - m(x) = 0, & x \in \Omega, \\ u = 0, & x \in \partial\Omega, \\ m_a \le m \le m_b, & x \in \Omega, \end{cases}$$

- Ω : domain of interest
- $u : \Omega \to \mathbb{R}$ (unknown temperature)
- $u_d : \Omega \to \mathbb{R}$ (desired temperature)
- α : (unknown temperature)
- $m : \Omega \to \mathbb{R}$ (source term : control function)

- We want to control $m(x)$ which minimize

$$J_{total}(m(x), u(x))$$

$$= \min \frac{1}{2} \int_{\Omega} (u - u_d)^2 \, dx + \frac{\alpha}{2} \int_{\Omega} m^2 dx$$

Solving PDE-constrained control problem using operator learning [3, AAAI 2022]

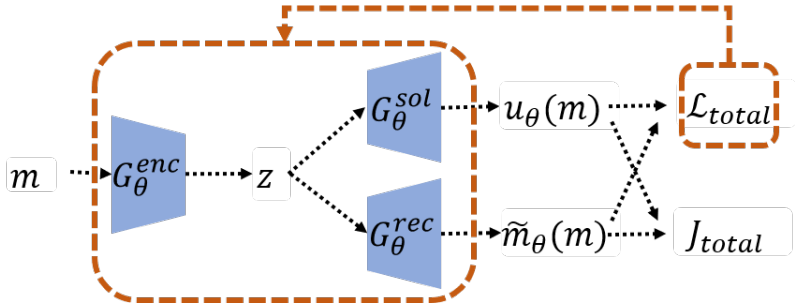
Hwang, R., **Lee, J. Y.**, Shin, J. Y., & Hwang, H. J. (2022, June). Solving pde-constrained control problems using operator learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 36, No. 4, pp. 4504-4512).

- **Solving control problems using operator learning.**

Poisson equation with zero Dirichlet boundary conditions

$$\begin{cases} -\Delta u(x) - m(x) = 0, & x \in \Omega, \\ u = 0, & x \in \partial\Omega, \\ m_a \le m \le m_b, & x \in \Omega, \end{cases}$$

- Ω : domain of interest
- $u : \Omega \to \mathbb{R}$ (unknown temperature)
- $u_d : \Omega \to \mathbb{R}$ (desired temperature)
- α : (unknown temperature)
- $m : \Omega \to \mathbb{R}$ (source term : control function)



Phase 1 : Operator learning
(Learning $m(x) \mapsto u(x)$)

- We want to control $m(x)$ which minimize

$$J_{total}(m(x), u(x))$$

$$= \min \frac{1}{2}\int_\Omega (u - u_d)^2\, dx + \frac{\alpha}{2}\int_\Omega m^2 dx$$

Solving PDE-constrained control problem using operator learning [3, AAAI 2022]

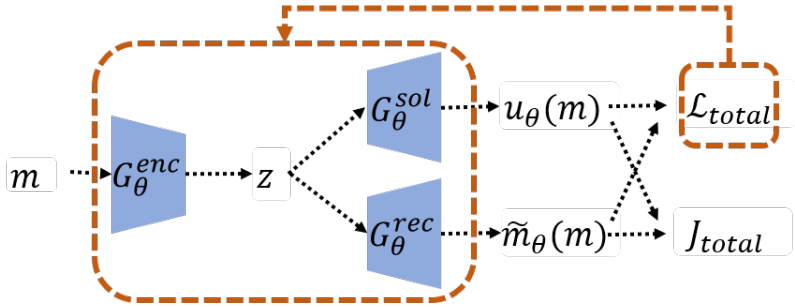
Hwang, R., **Lee, J. Y.**, Shin, J. Y., & Hwang, H. J. (2022, June). Solving pde-constrained control problems using operator learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 36, No. 4, pp. 4504-4512).

- **Solving control problems using operator learning.**

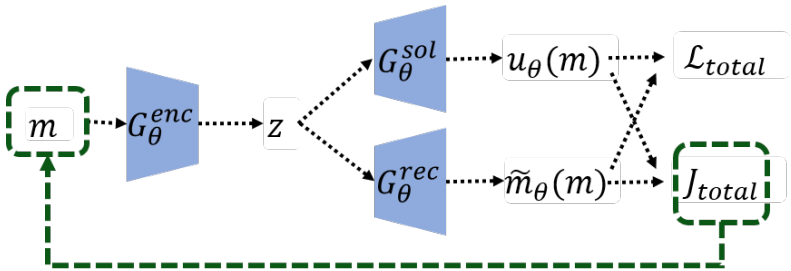
Poisson equation with zero Dirichlet boundary conditions

$$\begin{cases} -\Delta u(x) - m(x) = 0, & x \in \Omega, \\ u = 0, & x \in \partial\Omega, \\ m_a \le m \le m_b, & x \in \Omega, \end{cases}$$

- Ω : domain of interest
- $u : \Omega \to \mathbb{R}$ (unknown temperature)
- $u_d : \Omega \to \mathbb{R}$ (desired temperature)
- α : (unknown temperature)
- $m : \Omega \to \mathbb{R}$ (source term : control function)



Phase 1 : Operator learning
(Learning $m(x) \mapsto u(x)$)



Phase 2 : Solving control problem
(Minimize J_{total} to find optimal $m(x)$)

- We want to control $m(x)$ which minimize

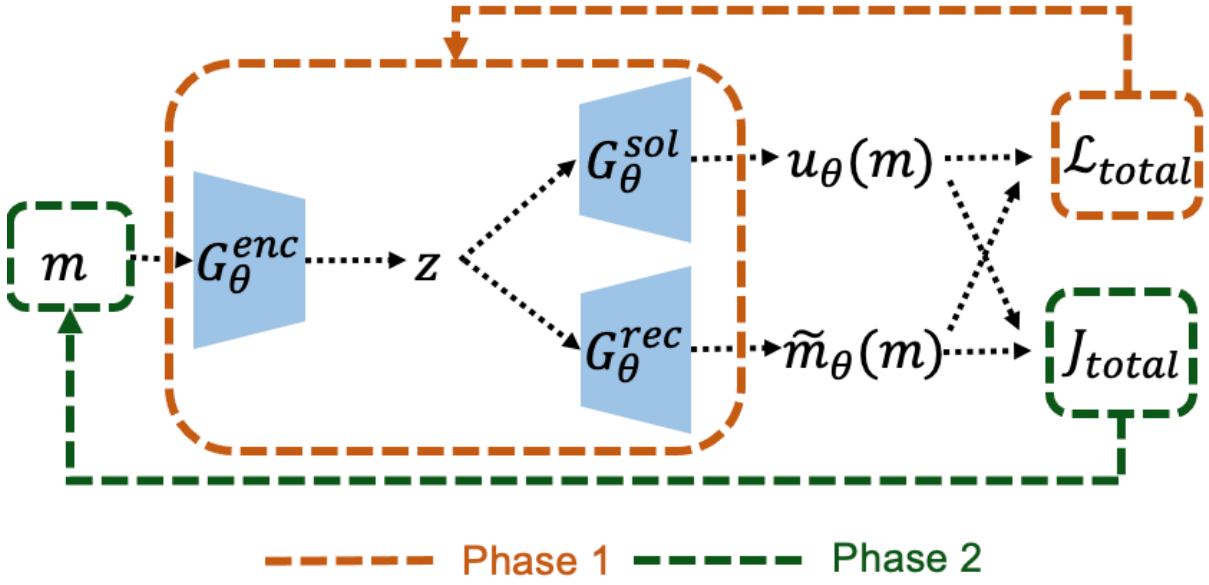
$$J_{total}(m(x), u(x))$$

$$= \min \frac{1}{2}\int_\Omega (u - u_d)^2\, dx + \frac{\alpha}{2}\int_\Omega m^2 dx$$

Solving PDE-constrained control problem using operator learning [3, AAAI 2022]

Hwang, R., **Lee, J. Y.**, Shin, J. Y., & Hwang, H. J. (2022, June). Solving pde-constrained control problems using operator learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 36, No. 4, pp. 4504-4512).

- **Solving control problems using operator learning.**
  - Our framework approximates a PDE solution with **sufficiently high accur acy** to search optimal controls while taking significantly **less time for infe rence.**

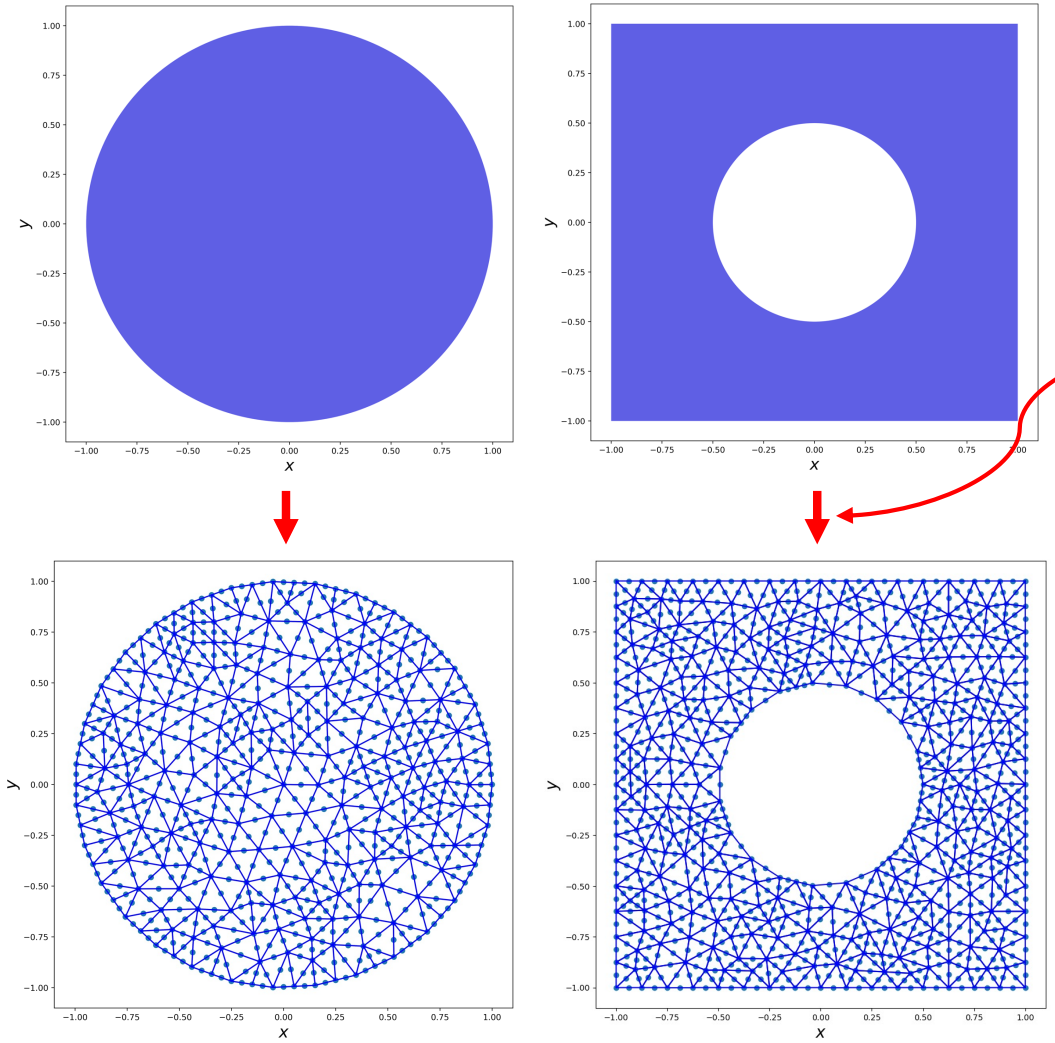


FEONet: Data-free FEM-based neural operator [4, preprint 2023]

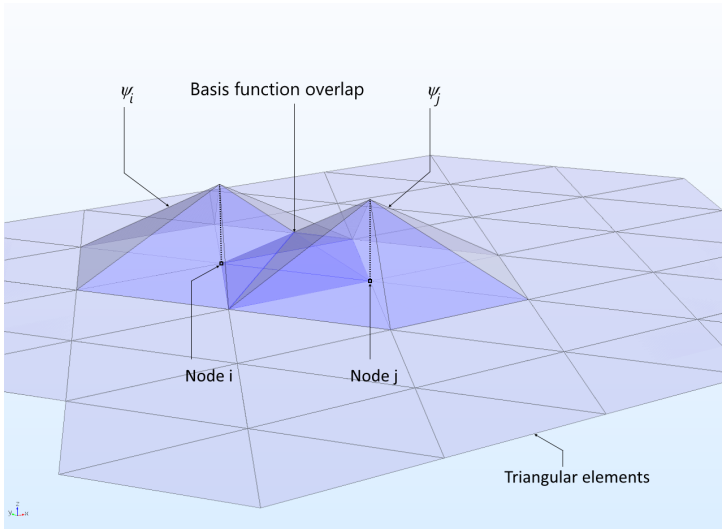
Lee, J. Y., Ko, S., & Hong, Y. (2023). Finite Element Operator Network for Solving Parametric PDEs. arXiv preprint arXiv:2308.04690.

- Neural Operator on complex domains based on **Finite Element Method (FEM)**

It's **difficult** to use Deep Learning directly on complex domains!



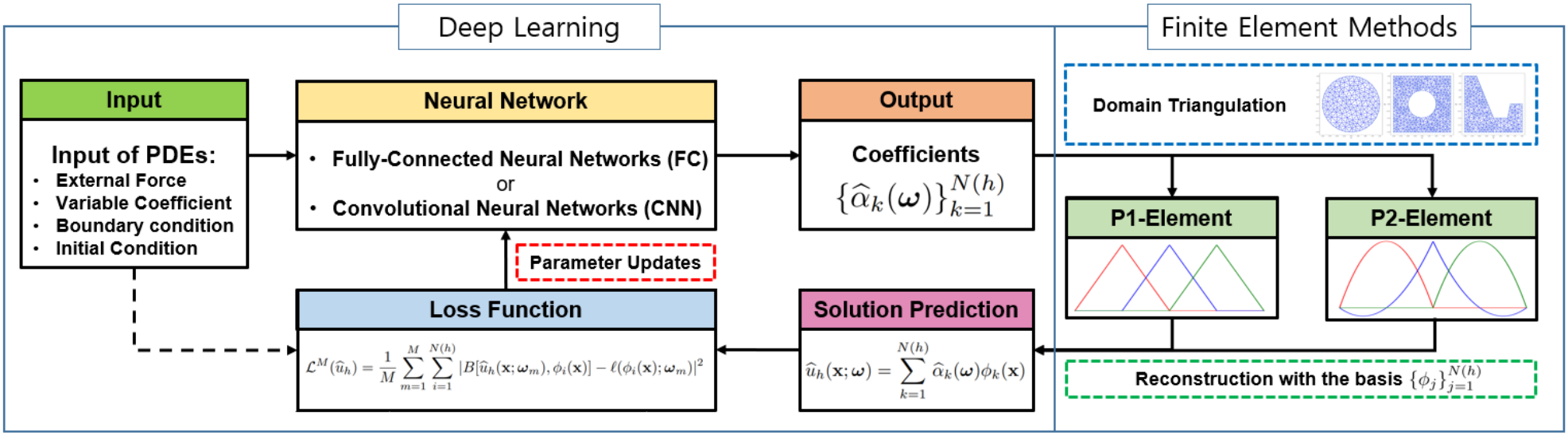
We can develop **FEM**-based Deep Learning!!



FEONet: Data-free FEM-based neural operator [4, preprint 2023]

Lee, J. Y., Ko, S., & Hong, Y. (2023). Finite Element Operator Network for Solving Parametric PDEs. arXiv preprint arXiv:2308.04690.

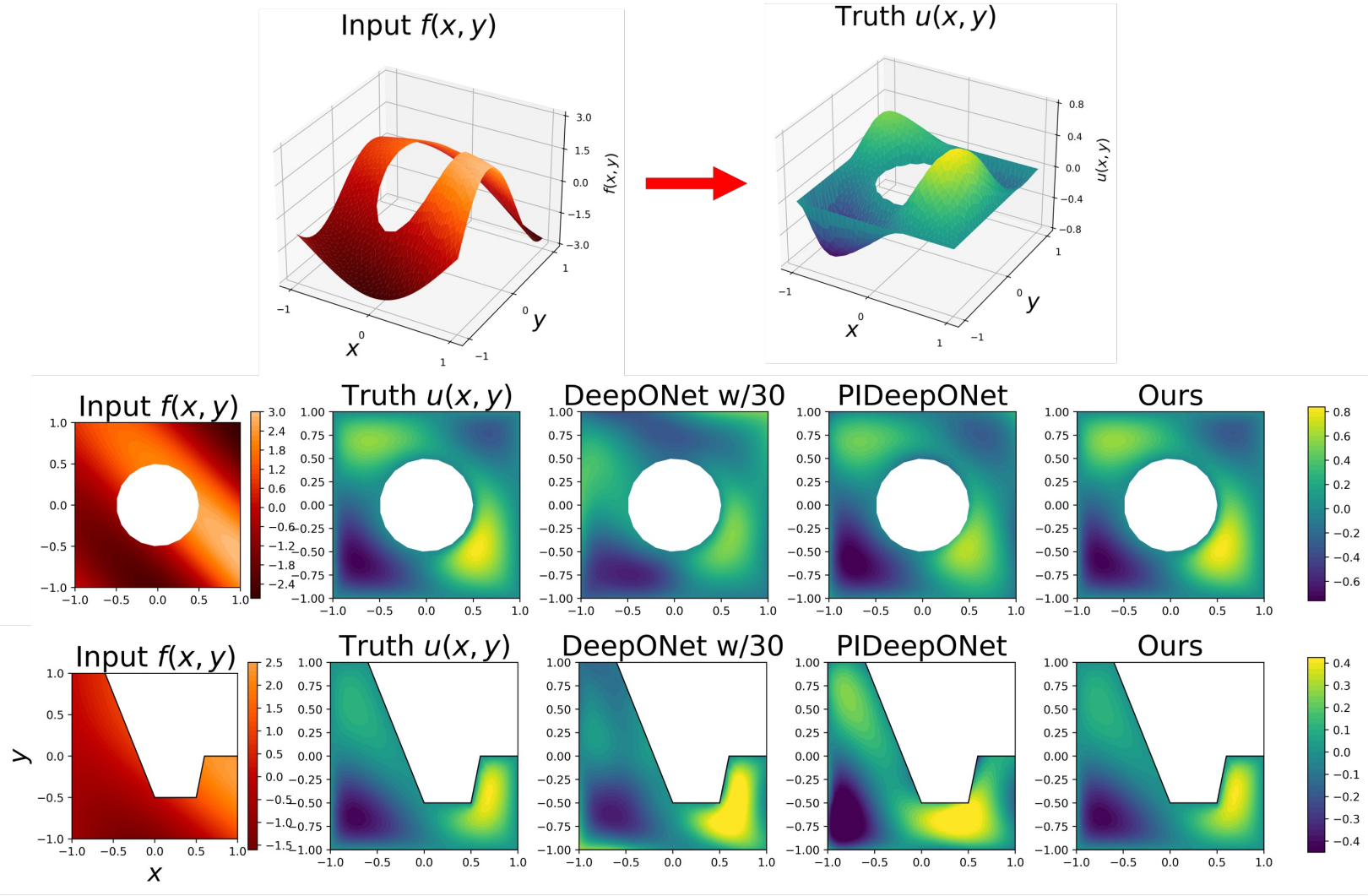
- Neural Operator on complex domains based on **Finite Element Method (FEM)**



FEONet: Data-free FEM-based neural operator [4, preprint 2023]

Lee, J. Y., Ko, S., & Hong, Y. (2023). Finite Element Operator Network for Solving Parametric PDEs. arXiv preprint arXiv:2308.04690.

- Neural Operator on complex domains based on **Finite Element Method (FEM)**



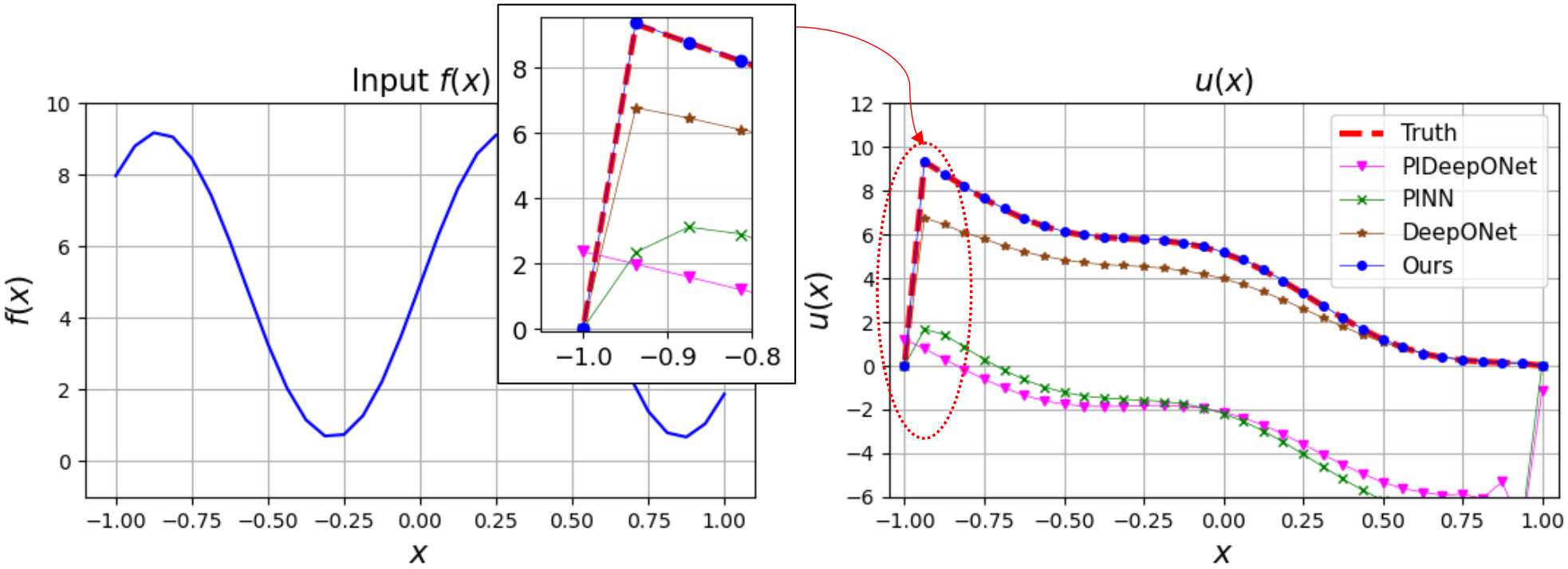
FEONet: Data-free FEM-based neural operator [4, preprint 2023]

Lee, J. Y., Ko, S., & Hong, Y. (2023). Finite Element Operator Network for Solving Parametric PDEs. arXiv preprint arXiv:2308.04690.

- Neural Operator on complex domains based on **Finite Element Method (FEM)**

Apply to singular perturbation and boundary layer problems

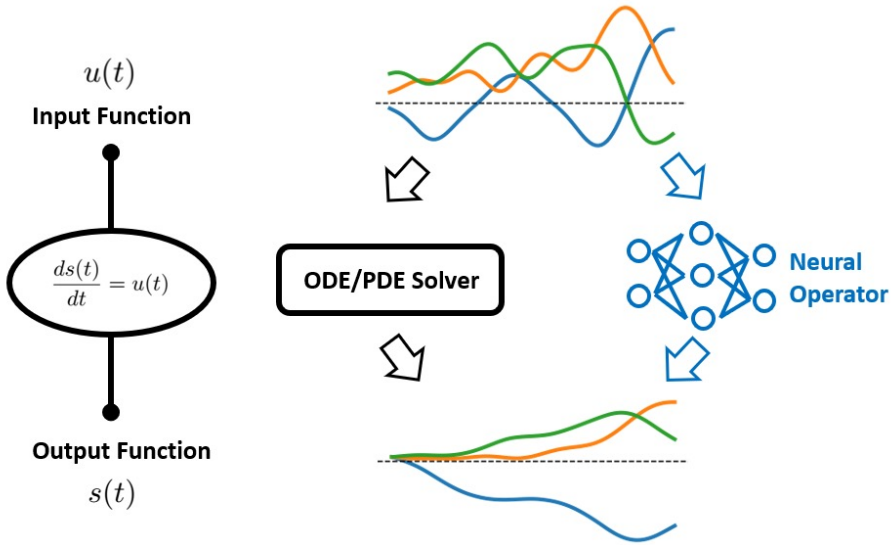
$$-\varepsilon u_{xx} + b u_x = f(x), \quad x \in D,$$
$$u(x) = 0, \qquad x \in \partial D,$$



Conclusion

Neural Operator

Learning in infinite dimensions



Key: Designing appropriate neural network structure to handle infinite-dimensional functions as inputs and outputs

[1] **Lee, J. Y.**, SungWoong, C. H. O., & Hwang, H. J. (2023). HyperDeepONet: learning operator with complex target function space using the limited resources via hypernetwork. In International Conference on Learning Representations.

[2] Shin, J. Y., **Lee, J. Y**., & Hwang, H. J. Pseudo-Differential Neural Operator: Generalize Fourier Neural operator for Learning Solution Operators of Partial Differential Equations. Transactions on Machine Learning Research.

[3] Hwang, R., Lee, J. Y., Shin, J. Y., & Hwang, H. J. (2022, June). Solving pde-constrained control problems using operator learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 36, No. 4, pp. 4504-4512).

[4] **Lee, J. Y.**, Ko, S., & Hong, Y. (2023). Finite element operator network for solving parametric pdes. *arXiv preprint arXiv:2308.04690*.