Latent Neural Operator for Solving Forward and Inverse PDE Problems

Tian Wang
Institute of Automation,
Chinese Academy of Sciences
School of Artificial Intelligence,
University of Chinese Academy of Sciences
wangtian2022@ia.ac.cn

Chuang Wang*
Institute of Automation,
Chinese Academy of Sciences
School of Artificial Intelligence,
University of Chinese Academy of Sciences
wangchuang@ia.ac.cn

NeurIPS 2024, the Thirty-Eighth Annual Conference on Neural Information Processing Systems

Background: Concept of PDE

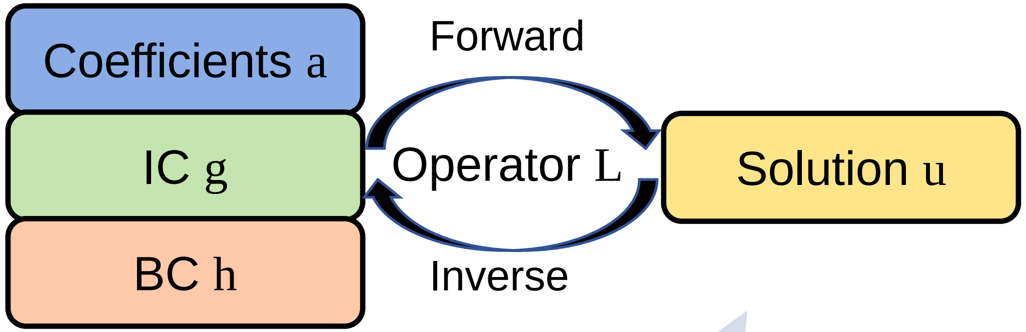
- **Forward Problem**
- Definition: given the coefficients, initial condition (IC) and boundary condition (BC), obtain the solution
- Applications: material property prediction, weather forecasting, industrial simulation
- **Inverse Problem**
- Definition:

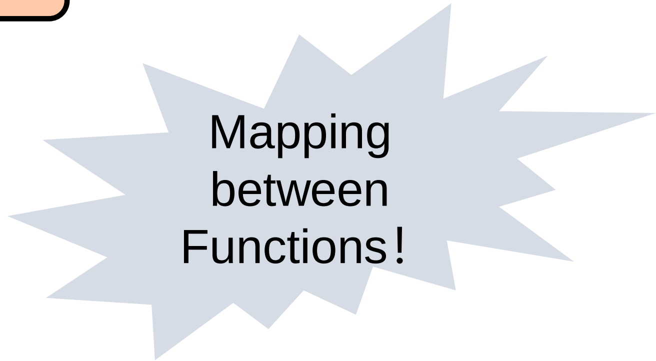
(1) System Identification: given partially observed solution, obtain the coefficients

(2) Boundary Inference: given partially observed solution, obtain the IC and BC

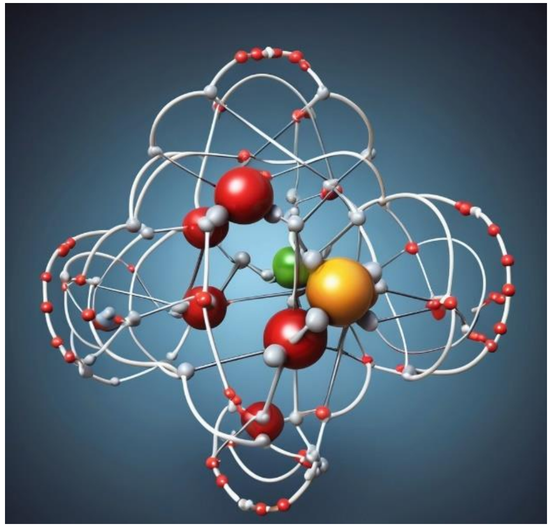
- Applications: geological exploration, pollution detection

$$L_a(u(x,t)) = 0$$
$$x,t \in D \times [0,T]$$
$$IC: u(x,t) = g(x), t = 0$$
$$BC: u(x,t) = h(x,t), x \in \partial D$$





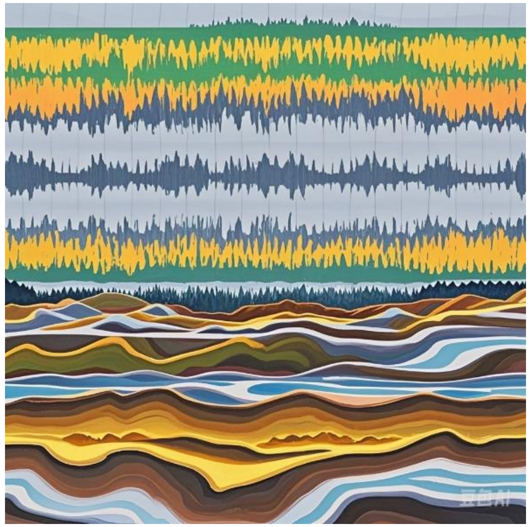
Background: Application of PDE



Material Property Prediction

Weather Forecasting

Industrial Simulation



Geological Exploration

Background: Neural Operator and Transformer

$$L_a u(x) = f(x), \qquad x \in D$$

$$u(x) = 0, \qquad x \in \partial D$$

$$L_a G(x,\cdot) = \delta_x$$

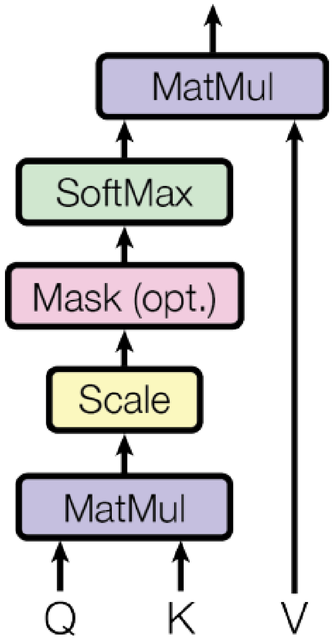
$$(L_a u)(x) = \int_D (L_a G(x,\cdot))(y) f(y) dy$$

$$= \int_D \delta_x(y) f(y) dy$$

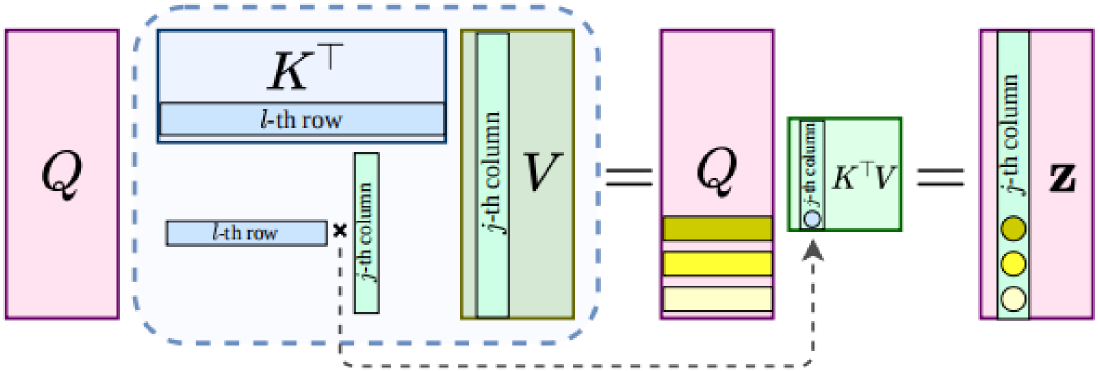
$$= f(x)$$

$$u(x) = \int_D G_a(x,y) f(y) dy$$

Green's Function



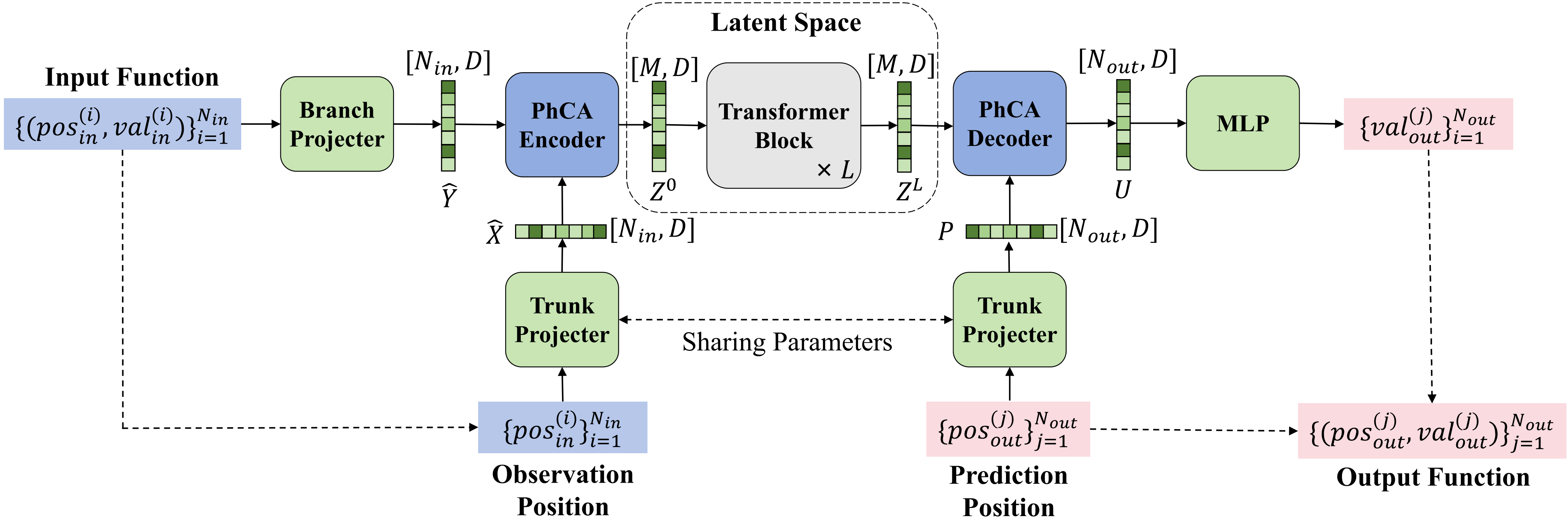
Attention (Quadratic)



Galerkin-type Attention (Linear)

Method: Latent Neural Operator (LNO)

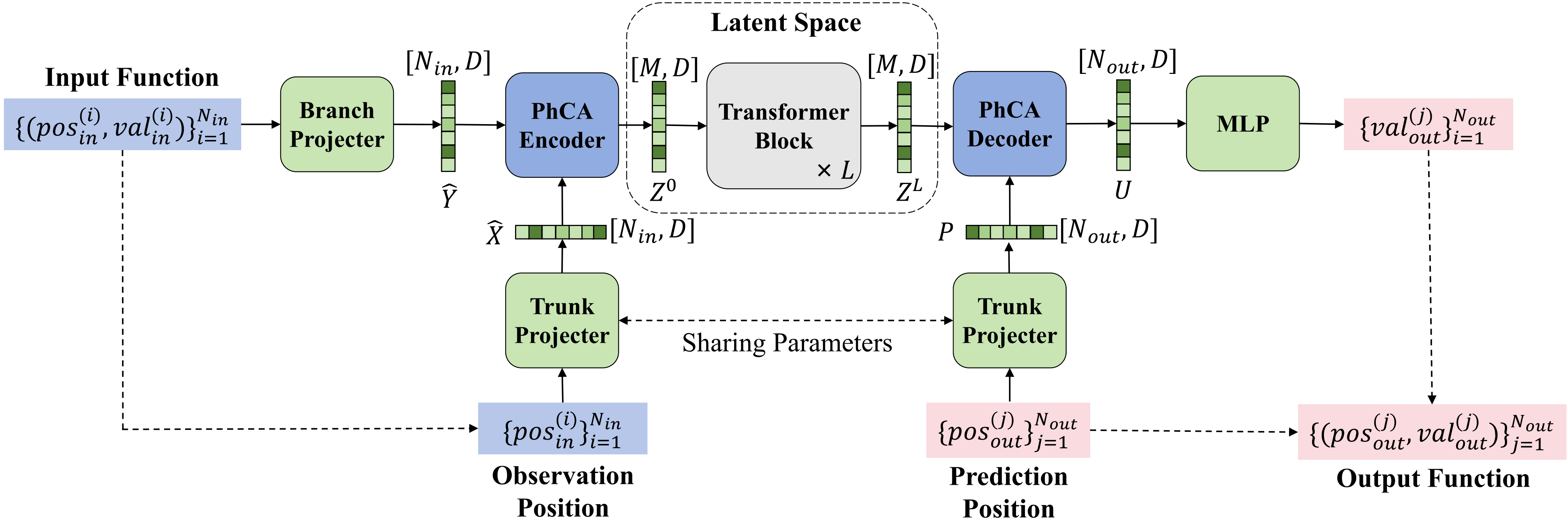
- **Motivation:** Accurate, Efficient and Flexible Neural Operator
- Accurate: We achieve SOTA accuracy on 4 out of 6 forward problem benchmarks and 1 inverse problem benchmark
- Efficient: We reduce memory usage by 50% and speed up training 1.8 times
- Flexible: We decouple the observation and prediction positions, allowing infinite resolution prediction



Method: Latent Neural Operator (LNO)

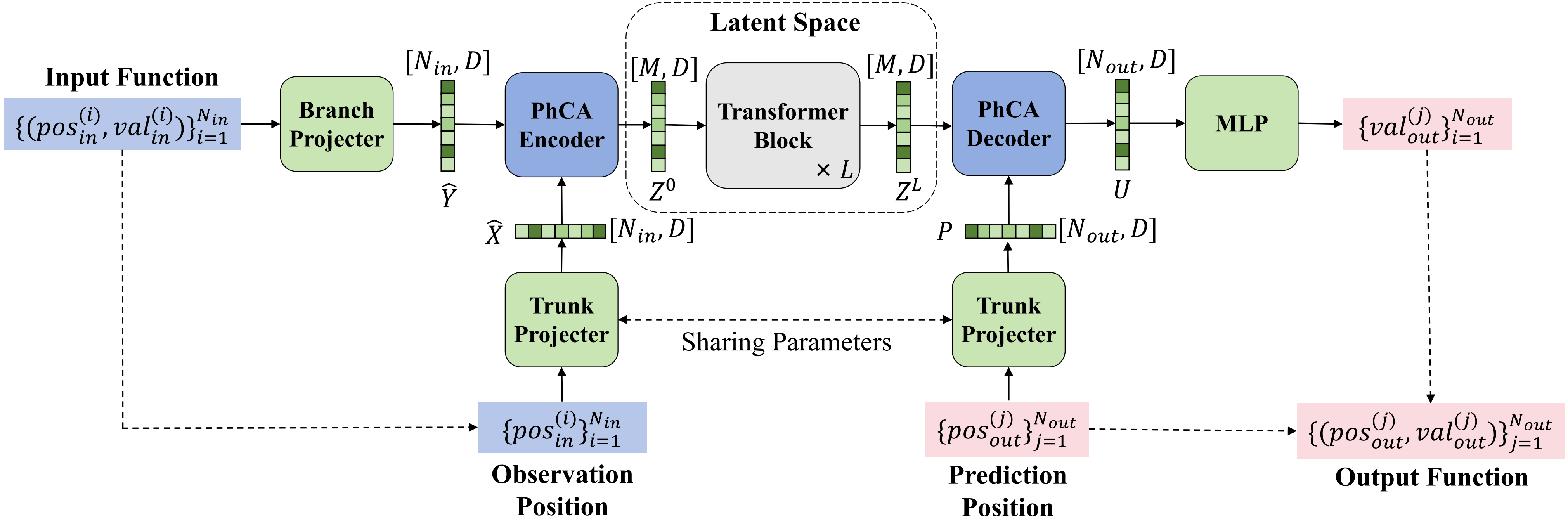
➢ **Data Format**

- Observation Sequence: $\{pos_{in}^{(i)}, val_{in}^{(i)}\}_{i=1}^{N_{in}}$, specific observation positions and the corresponding physical quantity values
- Prediction Sequence: $\{pos_{out}^{(j)}, val_{out}^{(j)}\}_{i=j}^{N_{out}}$, positions to be predicted and the corresponding ground truth physical quantity values



Method: Latent Neural Operator (LNO)

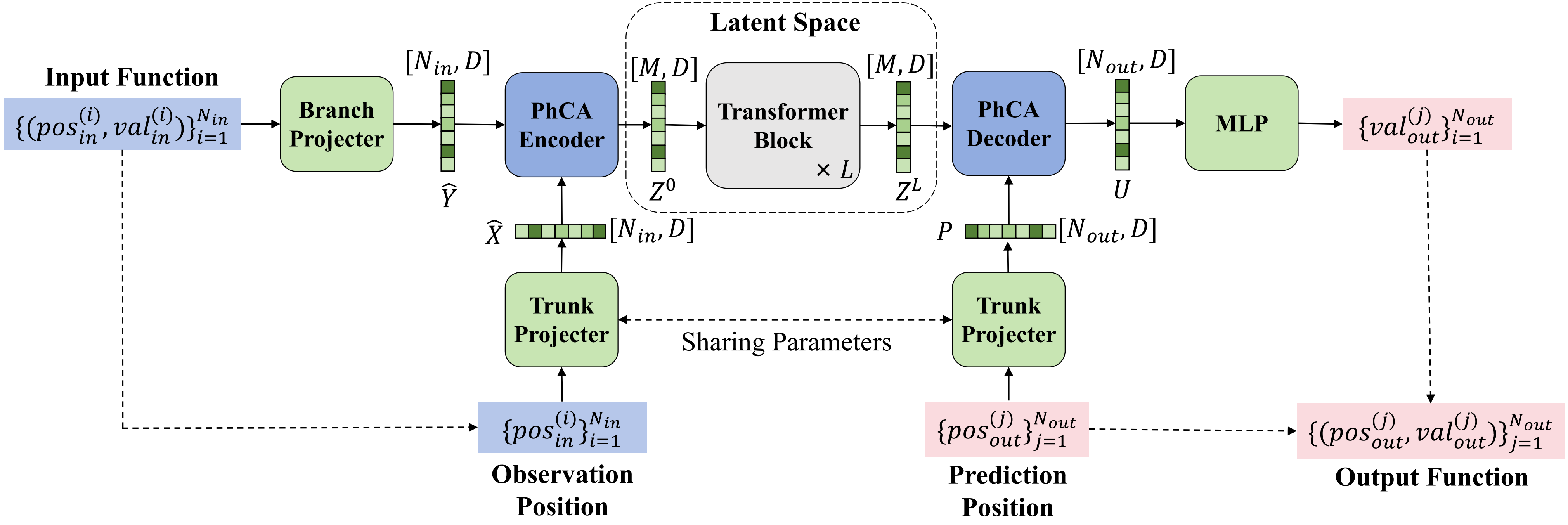
- **Latent Space**: Space where observed sampling points exist, with shape of $(N_{in}, d+n)$ for steady-state systems and $(N_{in}, d+n+1)$ for time-dependent systems. d is the dimension of spatial coordinate and n is the dimension of the physical quantity.
- **Geometric Space**: Space where representations of the hypothetical sampling points exist, with shape of (M, D). M is the number of hypothetical sampling points and D is the dimension of the representation.



Method: Latent Neural Operator (LNO)

- **1. Embedding**

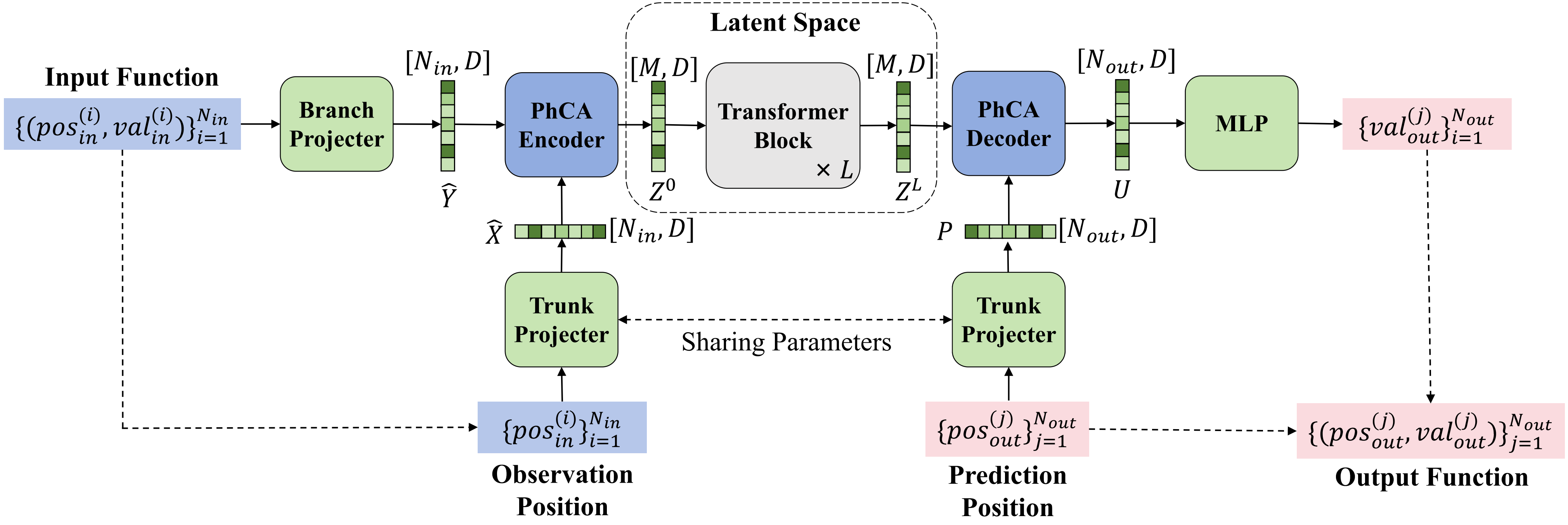
- Trunk-Projector: encoding $\{pos_{in}^{(i)}\}_{i=1}^{N_{in}}$ and $\{pos_{out}^{(j)}\}_{j=1}^{N_{out}}$ to $\hat{X} \in R^{N_{in} \times D}$ and $P \in R^{N_{out} \times D}$ respectively
- Branch-Projector: encoding $\{concat(pos_{in}^{(i)}, pos_{out}^{(i)})\}_{i=1}^{N_{in}}$ to $\hat{Y} \in R^{N_{in} \times D}$



Method: Latent Neural Operator (LNO)

➢ **Decoupling Property**

- The Trunk Projector encodes only position information, enabling the decoupling the positions of observation sequence and prediction sequence
- During inference, predictions can be made for positions without physical quantity information
- It allows for operations such as interpolation and extrapolation (key for solving inverse problem)



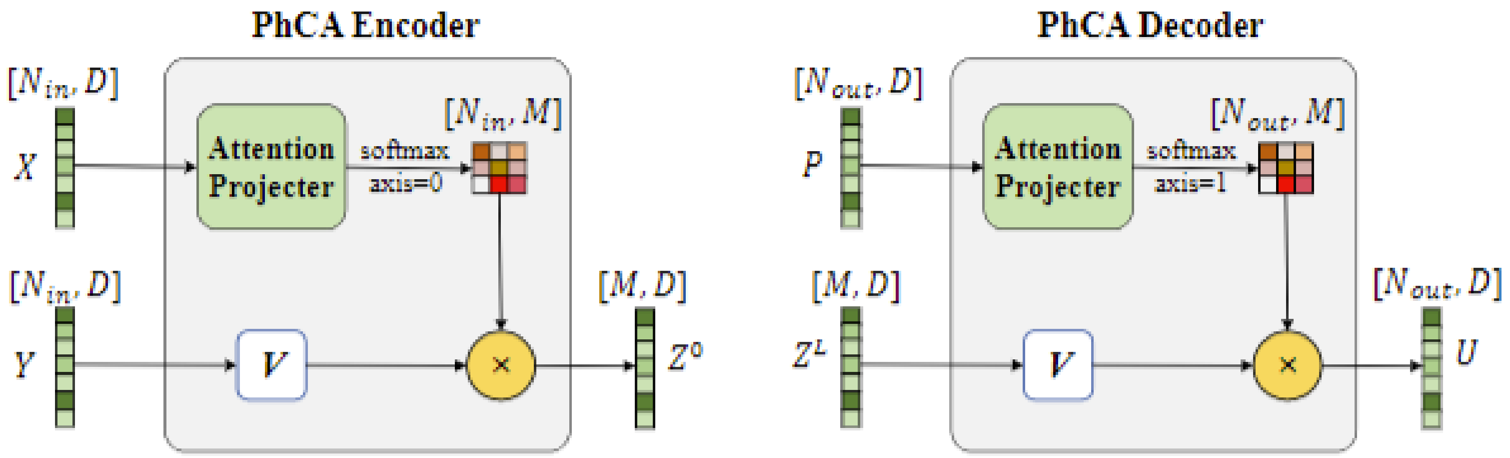
Method: Latent Neural Operator (LNO)

➢ **2. Encoding**

- We assume that the N_{in} sampling points in the geometric space can be represented as representations of M hypothetical sampling points in the latent space.
- We let the latent space positions serve as queries, the geometric space positions as keys, and the concatenation of geometric space positions and physical quantities as values.

Physics-Cross-Attention (PhCA):

$$Z^0 = \mathrm{softmax}\left(\frac{HW_QW_K^TX^T}{\sqrt{D}}\right)YW_V = \mathrm{softmax}(W_1X^T)YW_V, Z^0 \in R^{M\times D}$$

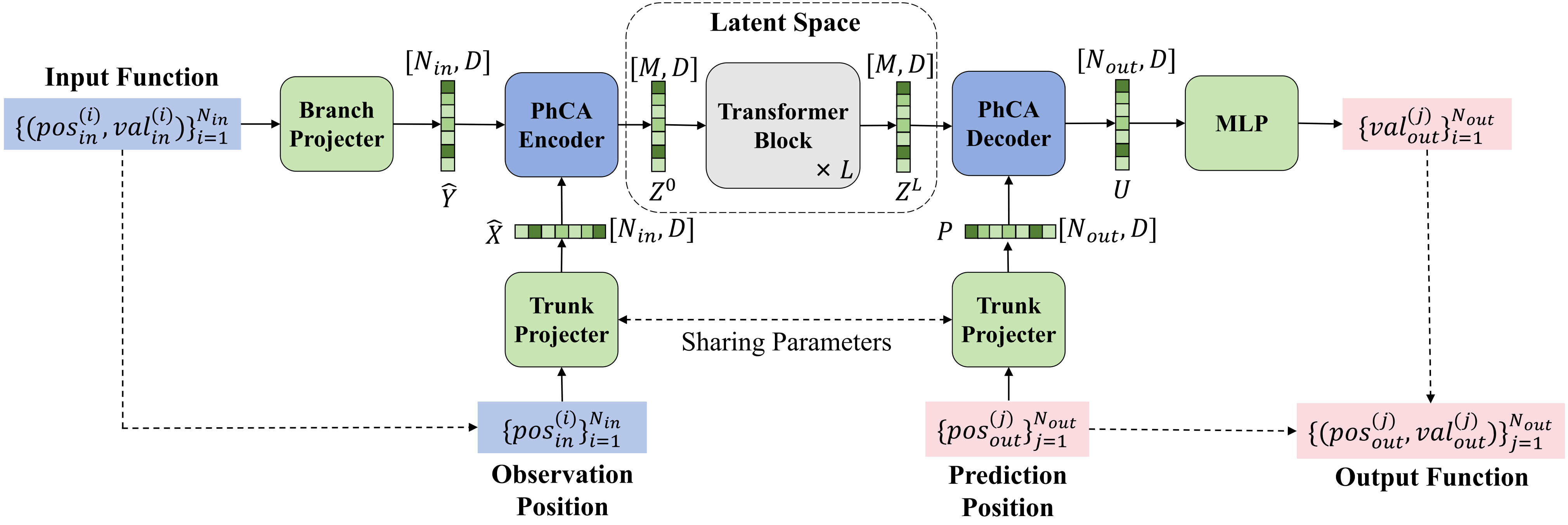


Method: Latent Neural Operator (LNO)

➢ **3. Transforming**

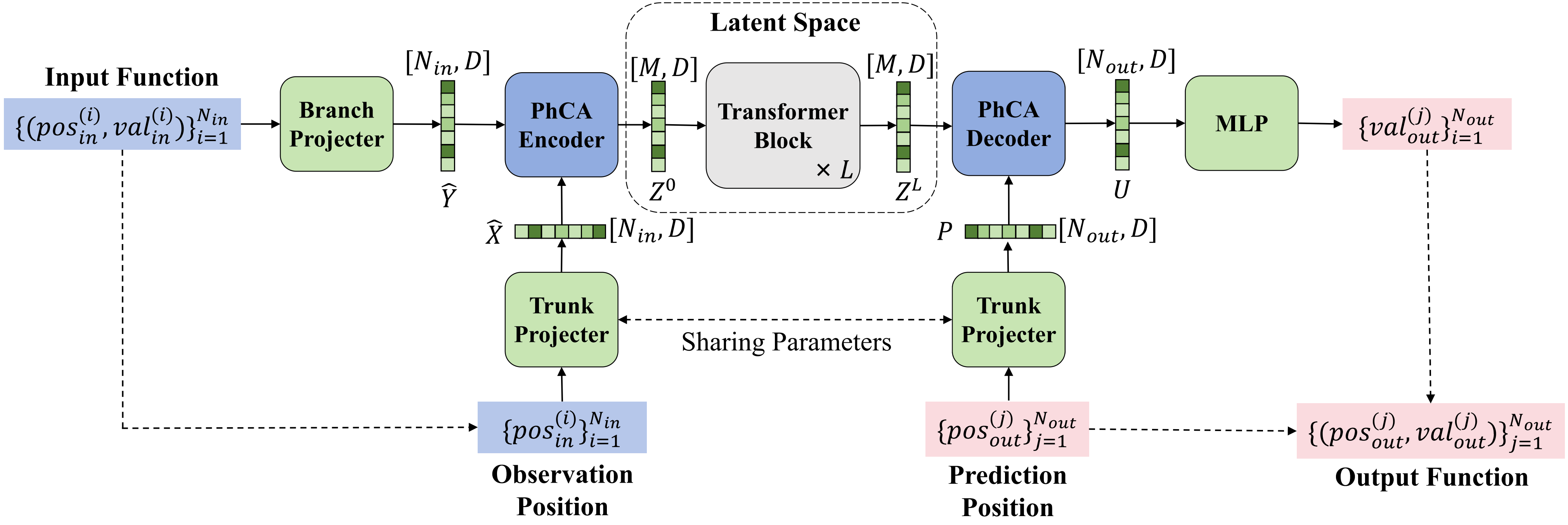
- We use the self attention mechanism as a kernel integral operator and stack Transformer blocks to learn the mapping from input functions to output functions in the latent space.

$$\hat{Z}^l = \mathrm{SelfAttention}(\mathrm{LayerNorm}(Z^{l-1})) + Z^{l-1}$$
$$Z^l = \mathrm{FeedForward}(\mathrm{LayerNorm}(\hat{Z}^l)) + \hat{Z}^l$$



Method: Latent Neural Operator (LNO)

- **Reduced complexity**
- The number of tokens (representations) in the latent space is fixes at M, and the computational complexity of self attention is $O(M^2D)$
- Compared to Transolver, there is no need for transformation in each Transformer Block, reducing the total computational complexity from $O(LMND + LM^2D)$ to $O(MND + LM^2D)$

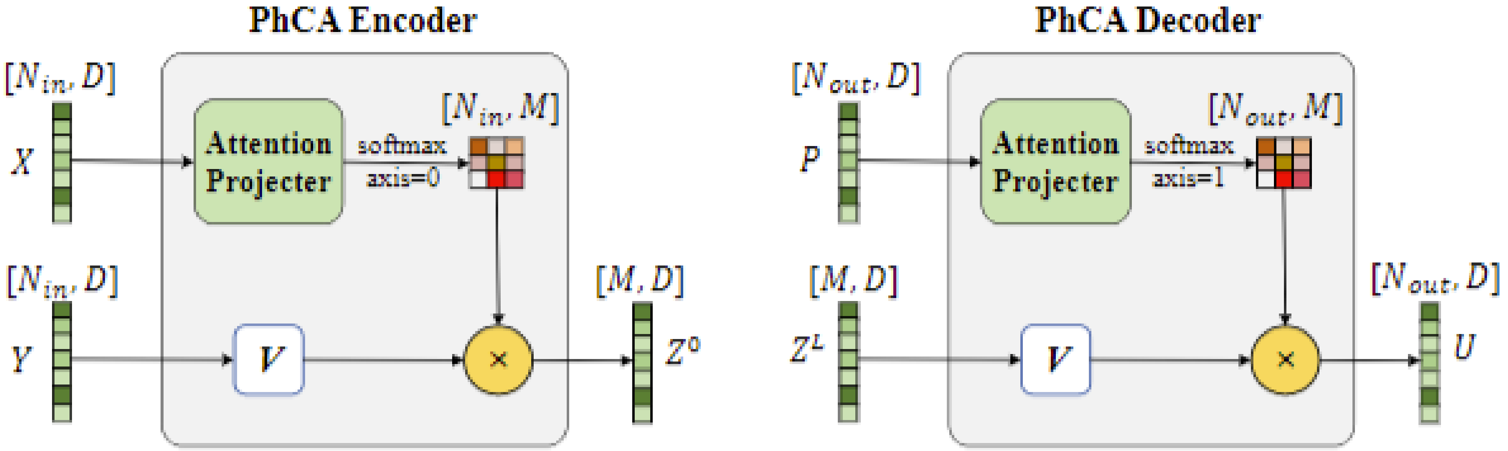


Method: Latent Neural Operator (LNO)

- **4. Decoding**
- We let the geometric space positions serve as queries, the latent space positions as keys, and the latent representations as values
- The representations of predicted physical quantities are decoded to obtain the values through another MLP

Physics-Cross-Attention (PhCA):

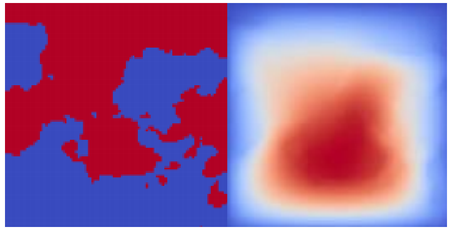
$$U = \text{softmax}\left(\frac{PW_QW_K^TH^T}{\sqrt{D}}\right)ZW_V = \text{softmax}(PW_2)YW_V, U \in R^{N_{out}\times D}$$

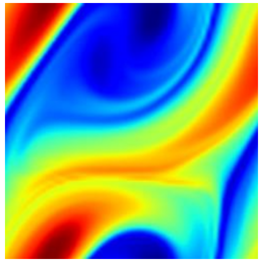


Results: Forward Problem

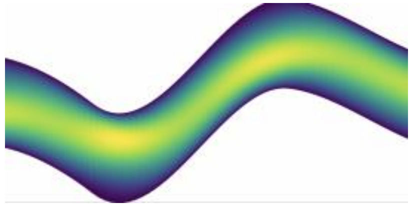
Model	D.C.	Relative L2($\times 10^{-2}$)					
		Darcy	NS2d	Airfoil	Elasticity	Plasticity	Pipe
FNO[14]	N	1.08	15.56	/	/	/	/
Geo-FNO[44]	N	1.08	15.56	1.38	2.29	0.74	0.67
F-FNO*[45]	N	0.75	11.51	0.60	1.85	0.27	0.68
U-FNO*[46]	N	1.28	17.15	1.19	2.13	0.41	0.62
LSM[22]	N	0.65	15.35	0.59	2.18	0.25	0.50
Galerkin[20]	N	0.84	14.01	1.18	2.40	1.20	0.98
OFormer[29]	Y	1.24	17.05	1.83	1.83	0.17	1.68
GNOT*[21]	N	1.04	13.40	0.75	0.88	3.19	0.45
FactFormer[30]	N	1.09	12.14	0.71	/	3.12	0.60
ONO[31]	Y	0.76	11.95	0.61	1.18	0.48	0.52
Transolver*	N	0.58	8.79	**0.47**	0.62	**0.12**	0.31
LNO(Ours)	Y	**0.49**	**8.45**	0.51	**0.52**	0.29	**0.26**

Metric	Model	Darcy	NS2d	Airfoil	Elasticity	Plasticity	Pipe
Paras Count(M)	Transolver	1.91	5.33	1.91	1.91	1.91	1.91
	LNO	**0.76**	**5.08**	**1.36**	**1.42**	**1.36**	**1.36**
Memory(GB)	Transolver	17.11	17.17	4.49	1.48	18.41	5.94
	LNO	**5.75**	**7.58**	**2.47**	**1.39**	**7.16**	**2.89**
Time(s/epoch)	Transolver	88.68	107.62	19.49	5.66	83.43	25.56
	LNO	**38.98**	**57.83**	**9.35**	**5.33**	**41.62**	**14.13**

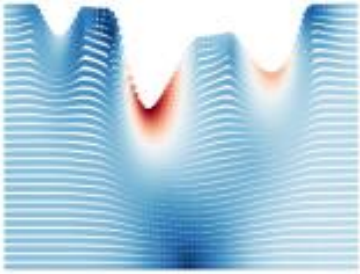

Darcy


NS2d


Airfoil

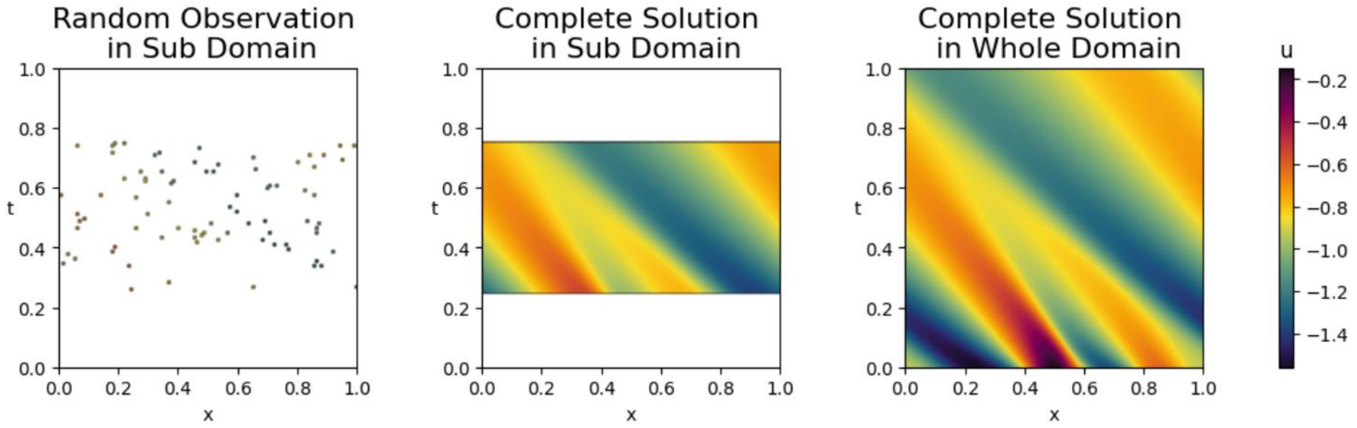

Pipe


Elasticity


Plasticity

Achieving SOTA accuracy while reducing memory usage by 50% and speeding up training 1.8 times

Results: Inverse Problem



Completer | Observation ratio	20%	10%	5%	1%	0.5%
DeepONet[13]	2.51%	2.59%	2.82%	3.25%	4.82%
GNOT[21]	1.12%	1.39%	1.62%	1.63%	2.56%
LNO(Ours)	**0.60%**	**0.74%**	**0.77%**	**1.18%**	**2.05%**

Propagator | Completer	G.T.	LNO		GNOT		DeepONet	
		10%	1%	10%	1%	10%	1%
DeepONet[13]	7.34%	8.01%	9.38%	9.09%	10.80%	11.14%	13.87%
GNOT[21]	5.45%	6.50%	8.07%	**8.04%**	**9.91%**	**10.41%**	**13.45%**
LNO(Ours)	**3.73%**	**5.69%**	**7.72%**	9.03%	10.98%	13.11%	15.50%

Achieving SOTA as both completer and propagator. Infinite resolution prediction.

Thank You!

wangtian2022@ia.ac.cn
wangchuang@ia.ac.cn

Code **Paper** **WeChat**