MITSUBISHI ELECTRIC
Changes for the Better

Physics-Inspired Operator Learning for Inverse Scattering with Application to Ground Penetrating Radar

Yanting Ma

Joint work with Qingqing Zhao (Standford, previous intern at MERL),

Petros Boufounos (MERL), Saleh Nabi (MERL), Hassan Mansour (MERL)

Brown AI Winter School, January 2025

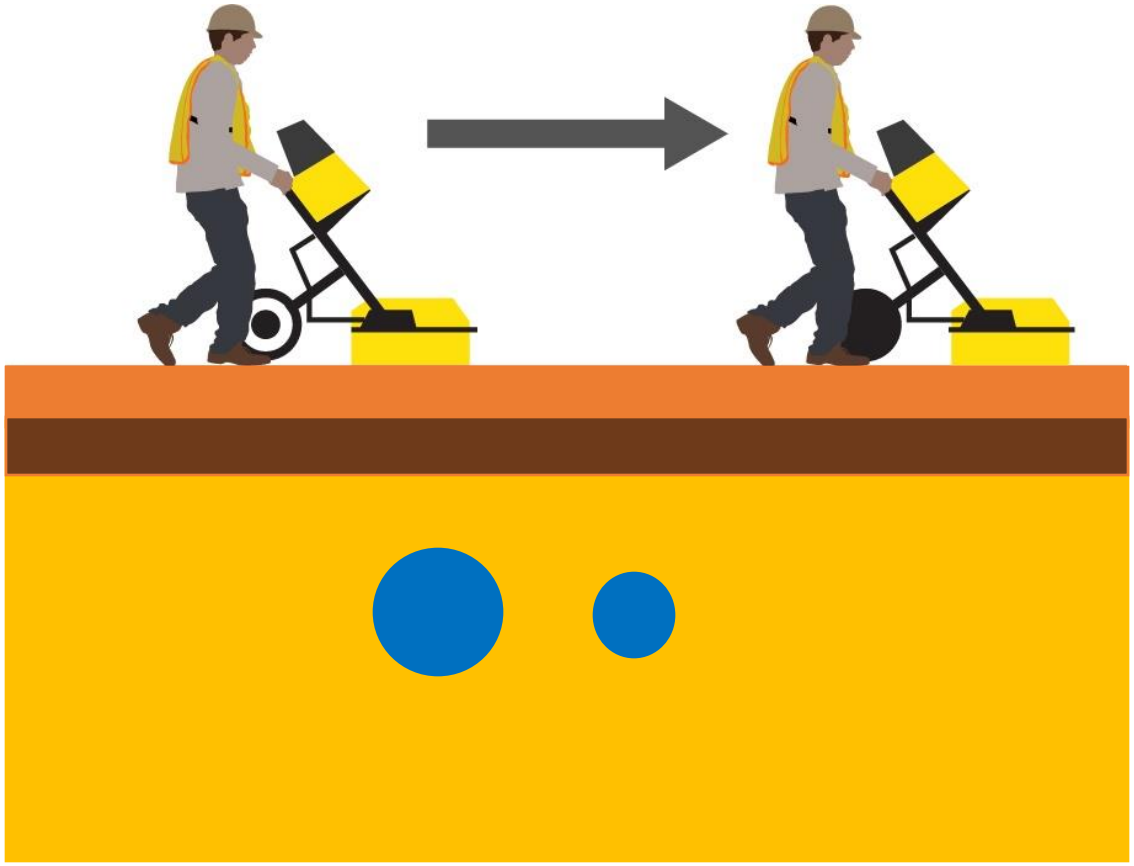
MITSUBISHI ELECTRIC RESEARCH LABORATORIES (MERL)
Cambridge, Massachusetts, USA
http://www.merl.com

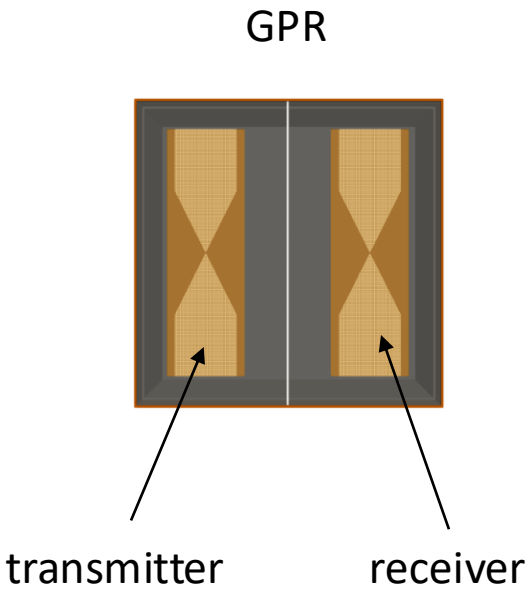
MITSUBISHI ELECTRIC
Changes for the Better

Ground Penetrating Radar

Ground Penetrating Radar (GPR)

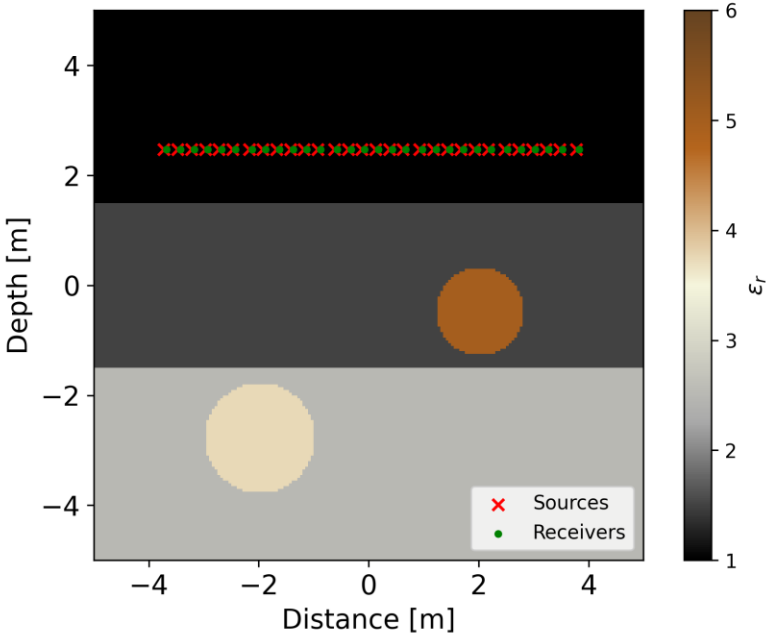
- Purpose: non-destructive underground mapping
- Challenges:
  - very limited measurements (highly ill-posed inverse problem)
  - Unknown objects extend to infinity, need to handle boundary condition carefully



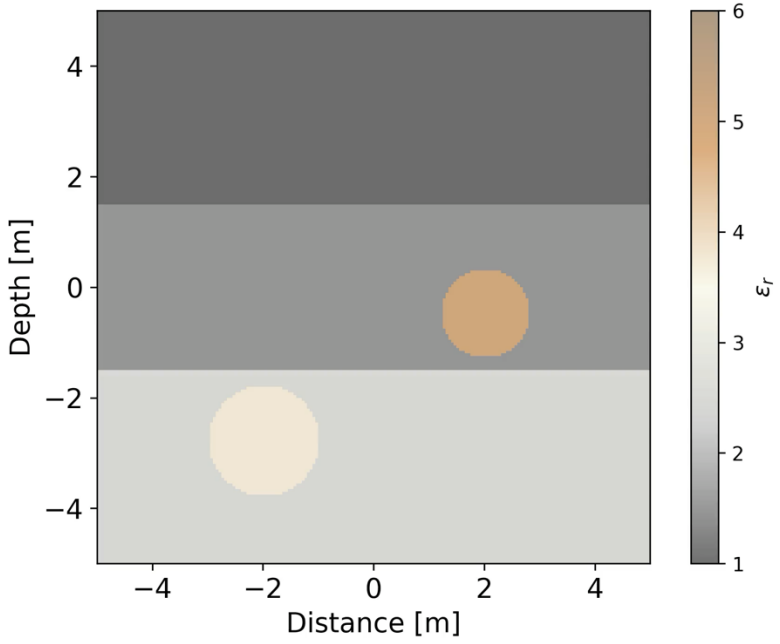


An Example for Wavefield in Time Domain

True Model



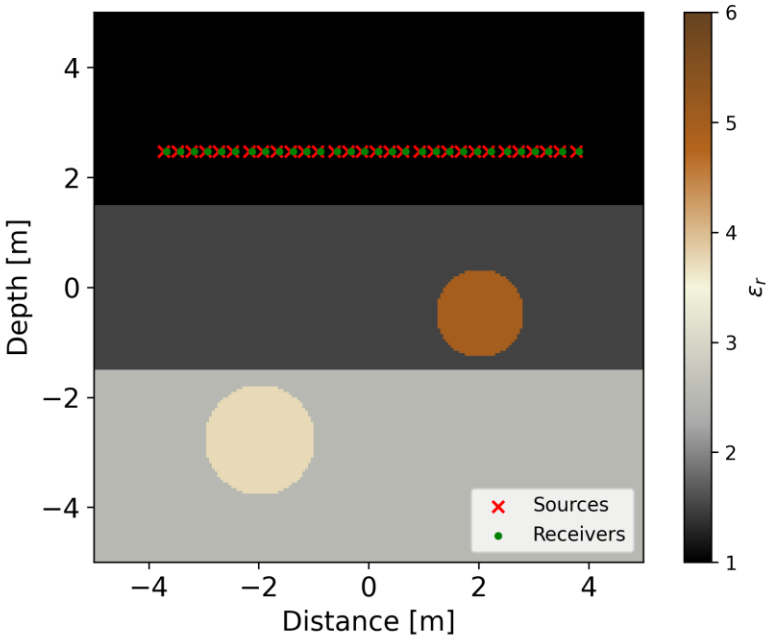
Wavefield from src 0



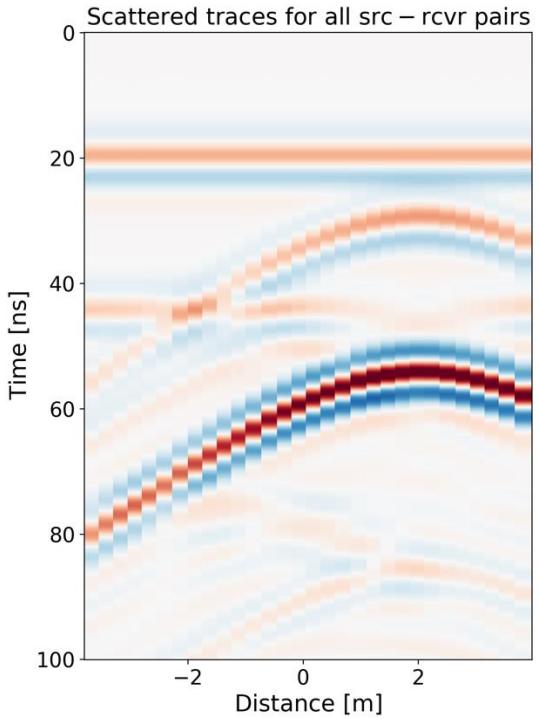
(video in PowerPoint)

An Example for GPR Measurements

True Model

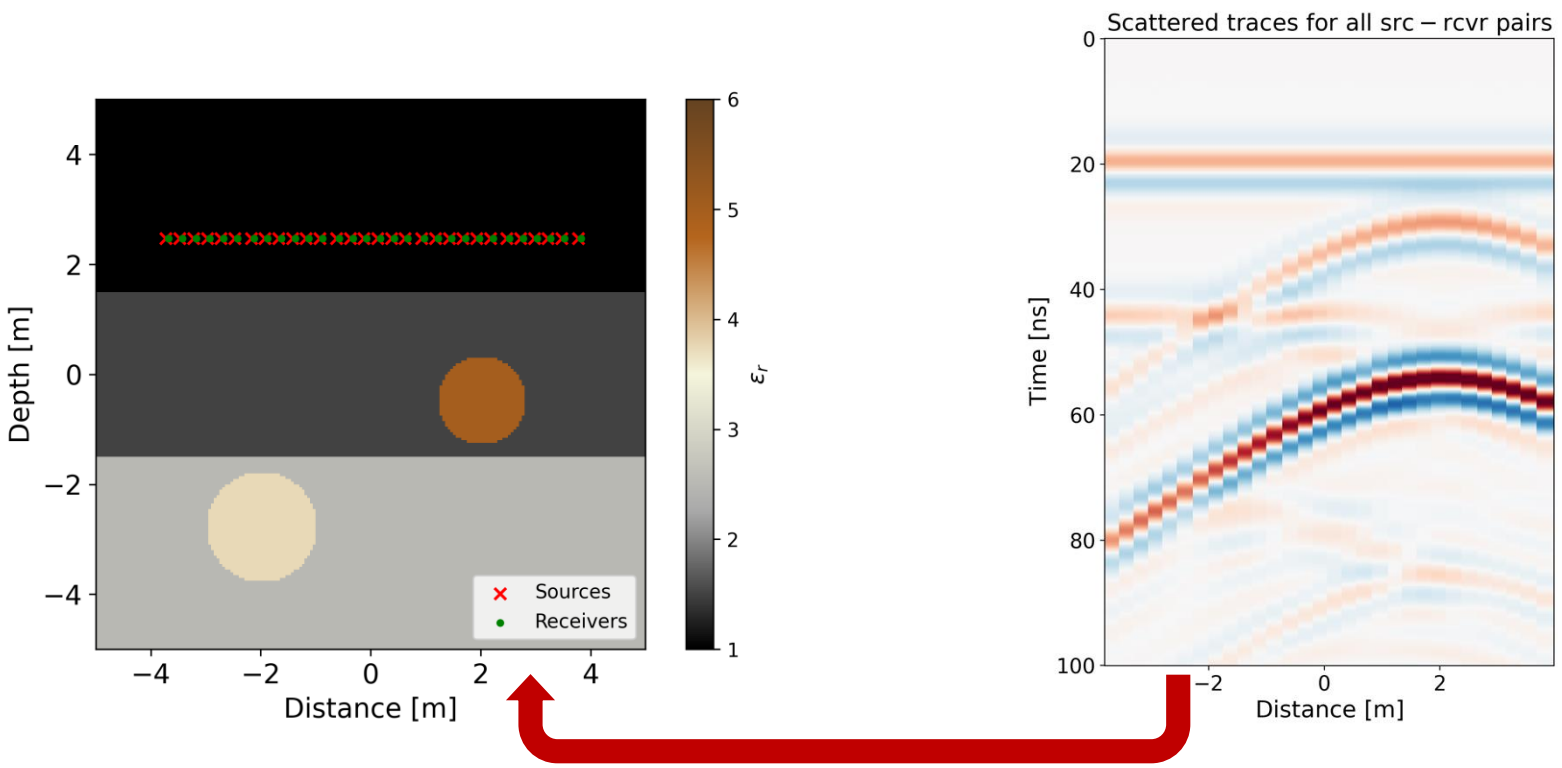


GPR data



The Forward and Inverse Problems

Forward: compute wavefield at the receiver given a permittivity map



Inverse: reconstruct permittivity map of underground scene given GPR data

MITSUBISHI ELECTRIC
Changes for the Better

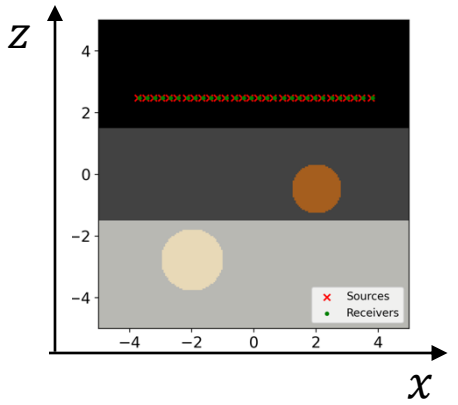
The Physics

Maxwell's Equations for a 2D Setup

- Assume all materials are non-magnetic
- Assume permittivity distribution $\epsilon(x, y, z)$ is inhomogeneous in xz-plane, but homogeneous along y-axis
- Assume time-harmonic fields (i.e., $\vec{E}(x, y, z, t) = \mathcal{R}\{\vec{E}(x, y, z; \omega)e^{-j\omega t}\}$)
- The source is an electrical current line source (a point in xz-plane and infinitely long along y-axis), and is linearly polarized in y direction
- In this setup, Maxwell's equations reduce to a scalar wave equation

$$\nabla^2 E_y(x, z; \omega) + k^2(x, z)E_y(x, z; \omega) = -i\omega\mu_0 J_y,$$

  where ω is angular frequency, $k^2(x, z) = \omega^2\mu_0\epsilon(x, z)$, J_y is current density of the source, and $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial z^2}$
- In the following, we let $\rho = (x, z)$, $u(\rho) = E_y(\rho; \omega)$



MITSUBISHI ELECTRIC
Changes for the Better

Integral Equation

- It can be shown that the solution of the differential equation satisfies the integral equation:

$$u(\rho) = u_{\text{in}}(\rho) + k_b^2 \int_{\mathbb{R}^2} g(\rho, \rho') f(\rho') u(\rho') d\rho'$$

  where

  - $f(\rho) = \epsilon(\rho)/\epsilon_b - 1$ is the permittivity contrast w.r.t. background ϵ_b
  - $u_{\text{in}}(\rho)$ is the incident field, which depends on the source and the background
  - $g(\rho, \rho')$ is the Green's function, which is the fundamental solution of the differential equation that satisfies

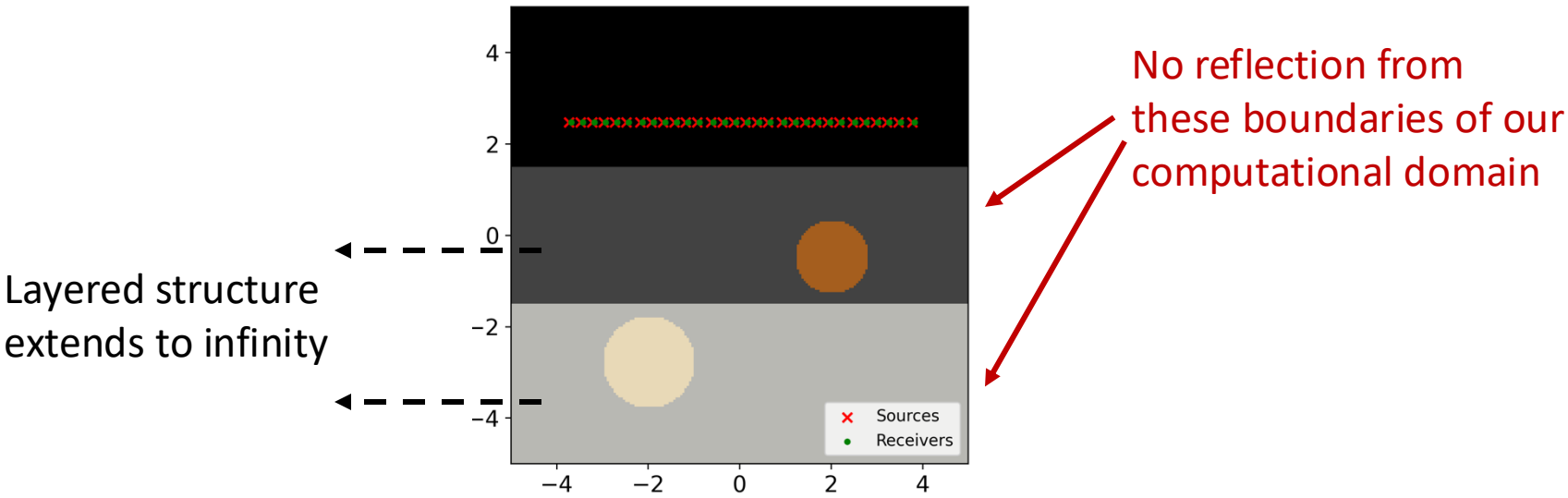
$$\nabla_\rho^2 g(\rho, \rho') + k_b^2 g(\rho, \rho') = \delta(\rho - \rho'),$$

- The integral equation is known as the Lippmann Schwinger equation and is commonly used for solving inverse scattering problems

Choosing Background

- If we choose **freespace** as the background, then the Green's function has closed form solution, but we need to construct absorbing boundary condition such that no reflection is coming from our artificial computational domain boundaries. (This is not easy to do, and we use **learning-based method**.)

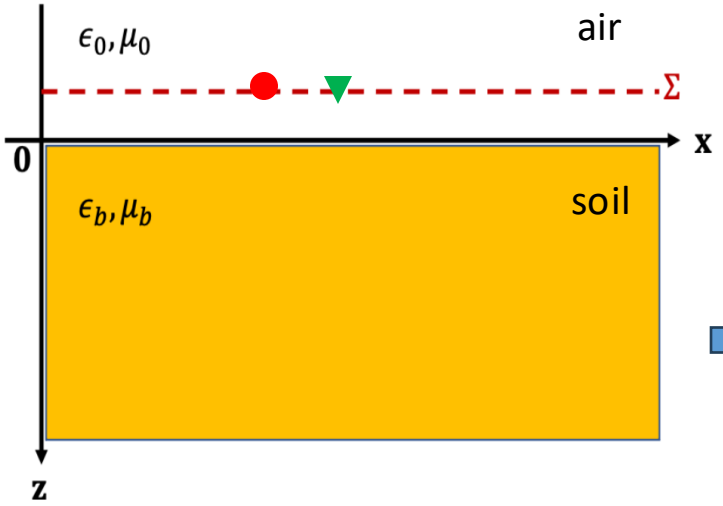
- If we choose infinitely large **layered structure** as the background, then the Green's function is harder to compute. (This is doable with **model-based method** if we have prior knowledge of the background and computational cost is not a concern; we will show an example.)



MITSUBISHI ELECTRIC
Changes for the Better

Model-Based Methods

A Halfspace Background Setup

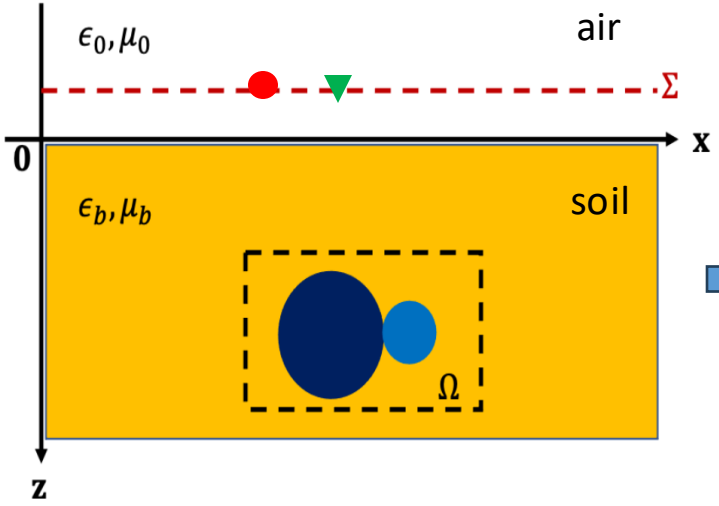


Σ: sensor locations

source

receiver

- Incident field $u_{\text{in}}(\rho)$, $\rho = (x, y)$
- Green's function $g(\rho, \rho')$ [1]



- Different colors represent different permittivity values (thus different materials)
- Forward problem: given permittivity distribution $\epsilon(\rho)$, compute total field $u(\rho)$
- Inverse problem: given total field $u(\rho)$ at receiver locations, compute permittivity distribution $\epsilon(\rho)$

[1] Persico, Raffaele. *Introduction to ground penetrating radar: inverse scattering and data processing*. John Wiley & Sons, 2014.

Forward Problem in Discrete Form

- Assume: all objects are located within Ω, hence, permittivtity contrast

$$f(\rho) := \frac{\epsilon(\rho) - \epsilon_b(\rho)}{\epsilon_b(\rho)} = 0 \quad \text{for} \quad \rho \notin \Omega$$

- The limits of integration can therefore be restricted to Ω:

$$u(\rho) = u_{\text{in}}(\rho) + \int_{\Omega} g(\rho, \rho') f(\rho') u(\rho') d\rho', \quad \rho \in \mathbb{R}^2$$

- Discretize Ω into N points and Σ into M points
- In discrete form, we can compute the total field at the receivers by

$$\mathbf{u}(\Sigma) = \mathbf{u}_{\text{in}}(\Sigma) + \mathbf{G}(\Sigma, \Omega)\text{diag}(\mathbf{f}(\Omega))\mathbf{u}(\Omega),$$

  where $\mathbf{u}(\Sigma), \mathbf{u}_{\text{in}}(\Sigma) \in \mathbb{C}^M$ are discretization of $u(\rho), u_{\text{in}}(\rho)$ with $\rho \in \Sigma$, $\mathbf{f}(\Omega), \mathbf{u}(\Omega) \in \mathbb{C}^N$ are discretization of $f(\rho), u(\rho)$ with ρ in Ω, and $\mathbf{G}(\Sigma, \Omega) \in \mathbb{C}^{M \times N}$ is discretization of $g(\rho, \rho')$ with $\rho \in \Sigma, \rho' \in \Omega$

- $\mathbf{u}(\Omega)$ in the equation above needs to satisfy

$$\mathbf{u}(\Omega) = \mathbf{u}_{\text{in}}(\Omega) + \mathbf{G}(\Omega, \Omega)\text{diag}(\mathbf{f}(\Omega))\mathbf{u}(\Omega),$$

MITSUBISHI ELECTRIC
Changes for the Better

PDE-Constrained Optimization for Inversion

- Measurements: $\mathbf{u}(\Sigma)$
- We can estimate $\mathbf{f}(\Omega)$ from $\mathbf{u}(\Sigma)$ by

$$\widehat{\mathbf{f}} = \underset{\mathbf{f}\in\mathbb{R}^N}{\arg\min}\ \mathcal{C}(\mathbf{f}) := \left\{ \|\mathbf{u}_{\text{in}}(\Sigma) + \mathbf{G}(\Sigma,\Omega)\text{diag}(\mathbf{f})\mathbf{u}(\Omega) - \mathbf{u}(\Sigma)\|_2^2 + \mathsf{Reg}(\mathbf{f}) \right\}$$
$$\text{subject to} \quad \mathbf{u}(\Omega) = \mathbf{u}_{\text{in}}(\Omega) + \mathbf{G}(\Omega,\Omega)\text{diag}(\mathbf{f})\mathbf{u}(\Omega)$$

where $\mathsf{Reg}(\mathbf{f})$ denotes some regularization of $\mathbf{f}(\Omega)$ (e.g., total variation semi-norm which promotes piecewise constant structure)

- Gradient-based algorithms can be used to solve the optimization problem, where the key step is to compute the gradient $\nabla\mathcal{C}(\mathbf{f})$, which can be obtained via one of the following:
  - Plug the constraint $\mathbf{u}(\Omega) = \big(\mathbf{I} - \mathbf{G}(\Omega,\Omega)\text{diag}(\mathbf{f})\big)^{-1}\mathbf{u}_{\text{in}}(\Omega)$ into $\mathcal{C}(\mathbf{f})$ and apply the chain rule
  - Use the adjoint-state method [1]
  - Code it with some differentiable program (e.g., Pytorch, JAX) and use automatic differentiation

[1] Plessix, R-E. "A review of the adjoint-state method for computing the gradient of a functional with geophysical applications." *Geophysical Journal International* 167.2 (2006): 495-503.

MITSUBISHI ELECTRIC
Changes for the Better

Evaluating Convolution in Fourier Domain

- Key idea for Fourier Neural Operator (used in our learning-based method)
- Recall that we need to evaluate the integration:

$$v(\rho) = \int_{\Omega} g(\rho, \rho') h(\rho') d\rho', \quad \forall \rho = (x, y) \in \Omega \subset \mathbb{R}^2, \quad \text{where } h(\rho) = f(\rho) u(\rho)$$

- It can be shown

$$g(\rho, \rho') = \begin{cases} g(\rho - \rho'), & \text{freespace (convolution)} \\ g_1(\rho - \rho') + g_2(x - x', z + z'), & \text{layered (mixed convolution-correlation)} \end{cases}$$

- In both cases, the integration can be approximated by discrete Fourier transform (DFT) and inverse DFT (IDFT)
- For simple illustration, let us consider convolution in 1D. In discrete form:

$$\mathbf{v}[n] = \sum_{n'=0}^{N-1} \mathbf{g}[n - n'] \mathbf{h}[n']$$

Evaluating Convolution in Fourier Domain (cont.)

- For $\mathbf{h} \in \mathbb{C}^N$, DFT $\widetilde{\mathbf{h}} = \mathcal{F}(\mathbf{h})$ and IDFT $\mathbf{h} = \mathcal{F}^{-1}(\widetilde{\mathbf{h}})$ are defined as:

$$\widetilde{\mathbf{h}}[k] = \sum_{n=0}^{N-1} \mathbf{h}[n] \exp\left(-i2\pi \frac{kn}{N}\right), \quad k = 0, \ldots, N-1$$

$$\mathbf{h}[n] = \frac{1}{N} \sum_{k=0}^{N-1} \widetilde{\mathbf{h}}[k] \exp\left(i2\pi \frac{kn}{N}\right), \quad k = 0, \ldots, N-1$$

- For $\mathbf{h}, \mathbf{g} \in \mathbb{C}^N$, circular convolution $\mathbf{y} = \mathbf{g} \circledast \mathbf{h}$ is defined as

$$\mathbf{y}[n] = \sum_{n'=0}^{N-1} \mathbf{g}[(n-n')_N]\mathbf{h}[n'], \quad n = 0, \ldots, N-1$$

where $(\cdot)_N$ denotes modulo N (e.g., $(2)_6 = 2, (6)_6 = 0, (-2)_6 = 4$)

- We have

$$\widetilde{\mathbf{y}}[k] = \sum_{n=0}^{N-1} \mathbf{y}[n] \exp\left(-i2\pi \frac{kn}{N}\right) = \sum_{n=0}^{N-1} \sum_{n'=0}^{N-1} \mathbf{g}[(n-n')_N]\mathbf{h}[n'] \exp\left(-i2\pi \frac{kn}{N}\right)$$

$$= \underbrace{\sum_{n'=0}^{N-1} \mathbf{h}[n'] \exp\left(-i2\pi \frac{kn'}{N}\right)}_{\widetilde{\mathbf{h}}[k]} \underbrace{\sum_{n=0}^{N-1} \mathbf{g}[(n-n')_N] \exp\left(-i2\pi \frac{k(n-n')}{N}\right)}_{\widetilde{\mathbf{g}}[k]} = \exp\left(-i2\pi \frac{k(n-n')_N}{N}\right)$$

Evaluating Convolution in Fourier Domain (cont.)

- We can therefore compute circular convolution by DFT and IDFT:

$$\mathbf{y} = \mathbf{g} \circledast \mathbf{h} = \underbrace{\mathcal{F}^{-1}\{\mathcal{F}\{\mathbf{g}\} \odot \mathcal{F}\{\mathbf{h}\}\}}, \quad \text{where } \odot \text{ denotes pointwise multiplication}$$

(We will see this again when we discuss Fourier neural operator later!)

- Recall that we wanted to compute linear convolution. We need to express it as a circular convolution. Let us use $N = 3$ for illustration

Linear convolution

$$\mathbf{v}[n] = \sum_{n'=0}^{N-1} \mathbf{g}[n-n']\mathbf{h}[n']$$

$$\begin{aligned}
\mathbf{v}[0] &= \mathbf{g}[0]\mathbf{h}[0] + \mathbf{g}[-1]\mathbf{h}[1] + \mathbf{g}[-2]\mathbf{h}[2] \\
\mathbf{v}[1] &= \mathbf{g}[1]\mathbf{h}[0] + \mathbf{g}[0]\mathbf{h}[1] + \mathbf{g}[-1]\mathbf{h}[2] \\
\mathbf{v}[2] &= \mathbf{g}[2]\mathbf{h}[0] + \mathbf{g}[1]\mathbf{h}[1] + \mathbf{g}[0]\mathbf{h}[2]
\end{aligned}$$

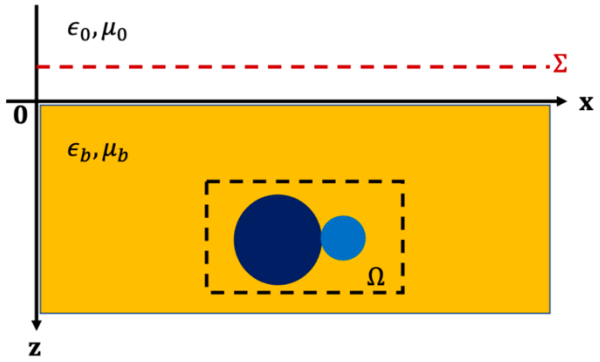
Circular convolution

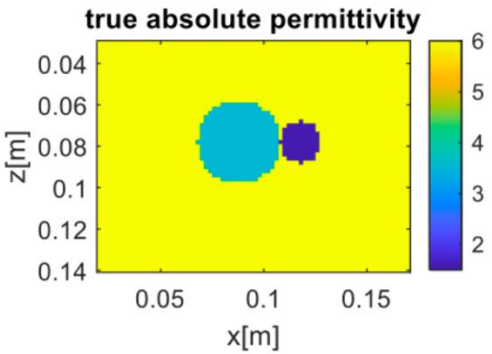
$$\mathbf{y} = (\mathbf{g}[0], \mathbf{g}[-1], \mathbf{g}[-2], \mathbf{g}[-3], \mathbf{g}[2], \mathbf{g}[1]) \circledast (\mathbf{h}[0], \mathbf{h}[1], \mathbf{h}[2], 0, 0, 0)$$

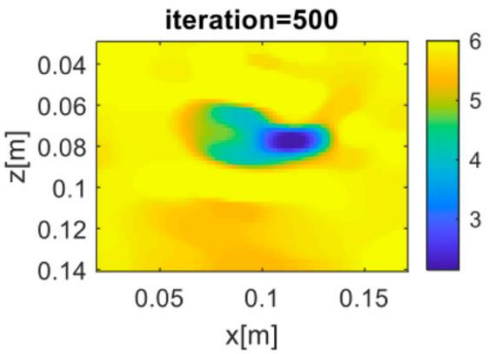
$$\begin{pmatrix} \mathbf{y}[0] \\ \mathbf{y}[1] \\ \mathbf{y}[2] \\ \mathbf{y}[3] \\ \mathbf{y}[4] \\ \mathbf{y}[5] \end{pmatrix} = \begin{pmatrix} \mathbf{g}[0] & \mathbf{g}[-1] & \mathbf{g}[-2] & \mathbf{g}[-3] & \mathbf{g}[2] & \mathbf{g}[1] \\ \mathbf{g}[1] & \mathbf{g}[0] & \mathbf{g}[-1] & \mathbf{g}[-2] & \mathbf{g}[-3] & \mathbf{g}[2] \\ \mathbf{g}[2] & \mathbf{g}[1] & \mathbf{g}[0] & \mathbf{g}[-1] & \mathbf{g}[-2] & \mathbf{g}[-3] \\ \mathbf{g}[-3] & \mathbf{g}[2] & \mathbf{g}[1] & \mathbf{g}[0] & \mathbf{g}[-1] & \mathbf{g}[-2] \\ \mathbf{g}[-2] & \mathbf{g}[-3] & \mathbf{g}[2] & \mathbf{g}[1] & \mathbf{g}[0] & \mathbf{g}[-1] \\ \mathbf{g}[-1] & \mathbf{g}[-2] & \mathbf{g}[-3] & \mathbf{g}[2] & \mathbf{g}[1] & \mathbf{g}[0] \end{pmatrix} \begin{pmatrix} \mathbf{h}[0] \\ \mathbf{h}[1] \\ \mathbf{h}[2] \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

1. Pad N zeros to h
2. Compute $2N$-point DFT and IDFT to get y
3. Take the first N elements of y to get v

Simulation Results







Limitations:

1. Only applies to layered background
2. Need to know or estimate the background to build Green's function
3. Computing Green's function is computationally expensive. For N x N computational domain, we need to evaluate the Sommerfeld integral 2N x 2N times for each temporal frequency

MITSUBISHI ELECTRIC
Changes for the Better

Learning-Based Methods

Neural Operator: Mapping between Functions

- Let $D \subset \mathbb{R}^d$ be a bounded open set, $\mathcal{A} = \mathcal{A}(D; \mathbb{R}^{d_a})$ be the input function space, and $\mathcal{U} = \mathcal{U}(D; \mathbb{R}^{d_u})$ be the output function space. For $x \in D$, we have $x \mapsto a(x) \in \mathbb{R}^{d_a}$ and $x \mapsto u(x) \in \mathbb{R}^{d_u}$, where $a \in \mathcal{A}$ and $u \in \mathcal{U}$.
- We want to learn an operator $G : \mathcal{A} \to \mathcal{U}$ given a dataset $\{a_j, u_j\}_{j=1}^N$ where $u_j = G(a_j)$.
- An network architecture proposed in [1]: let $v_0(x) = P(a(x))$, and recursively define

$$v_{t+1}(x) = \sigma\left(W v_t(x) + [\mathcal{K}_\phi(a) v_t](x)\right), \forall x \in D, t = 0, 1, \ldots, T-1,$$

  then $u(x) = Q(v_T(x))$ is the output. P lifts $a(x)$ to a higher dimensional representation space, Q projects the representation $v_T(x)$ to the output function space $\mathcal{U}$.
- The kernel integral operator $\mathcal{K}_\phi(a)$ (parameterized by ϕ) is defined as

$$[\mathcal{K}_\phi(a) v_t](x) = \int_D \kappa_\phi(x, y, a(x), a(y)) v_t(y) dy$$

[1] Li, Zongyi, et al. "Neural operator: Graph kernel network for partial differential equations." *arXiv preprint arXiv:2003.03485* (2020).

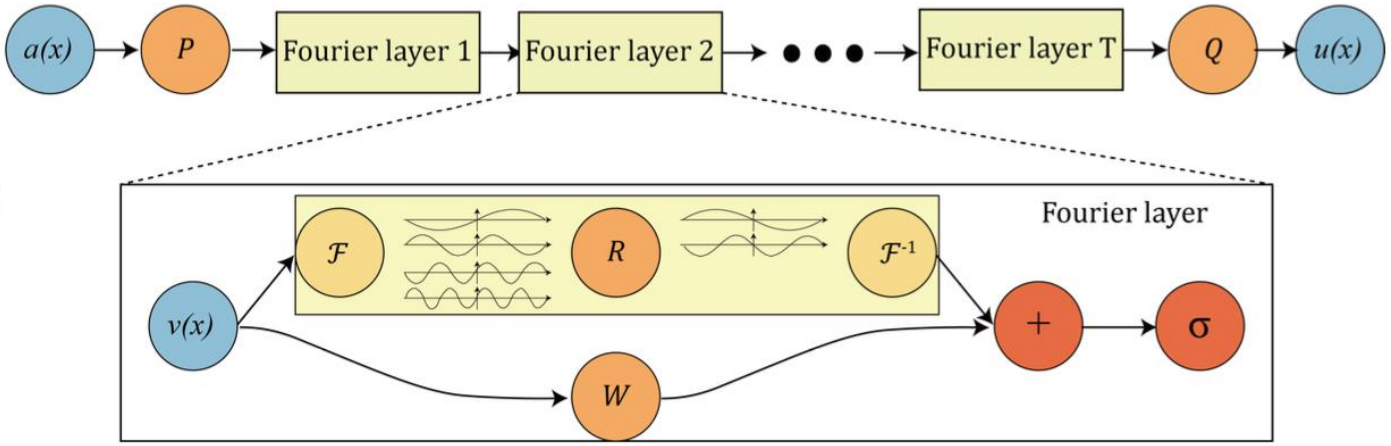
Fourier Neural Operator (FNO)

- When the kernel integral operator is a convolution operator, that is

$$[\mathcal{K}_\phi(a)v_t](x) = \int_D \kappa_\phi(x-y)v_t(y)dy,$$

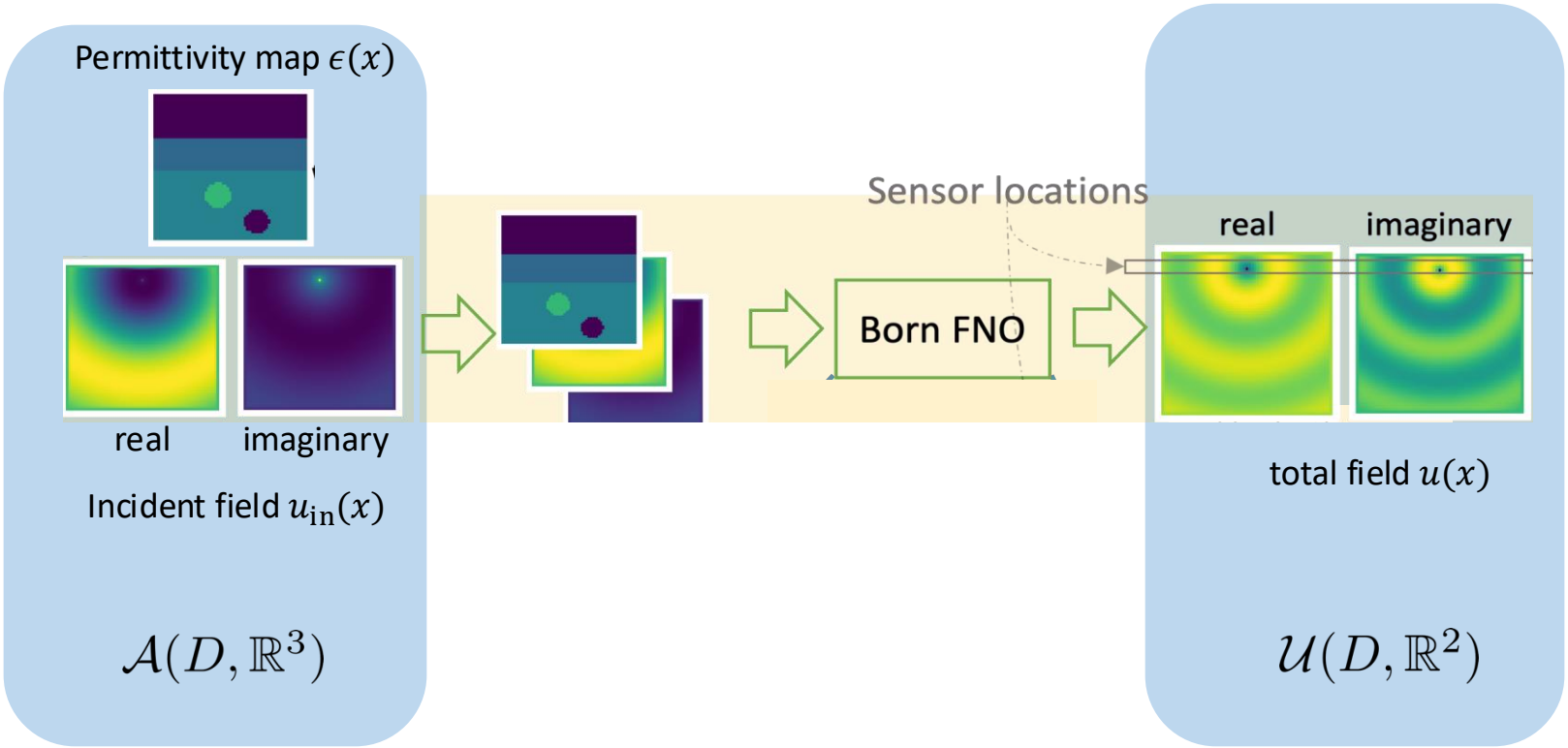
FNO [2] parameterizes κ_ϕ in Fourier space where convolution becomees multiplication (more computationally efficient)

- Specifically, $[\mathcal{K}_\phi(a)v_t](x) = \mathcal{F}^{-1}(\mathcal{F}(\kappa_\phi) \cdot \mathcal{F}(v_t))(x)$. R_ϕ
- A graphic representation of FNO is provided in [2]:



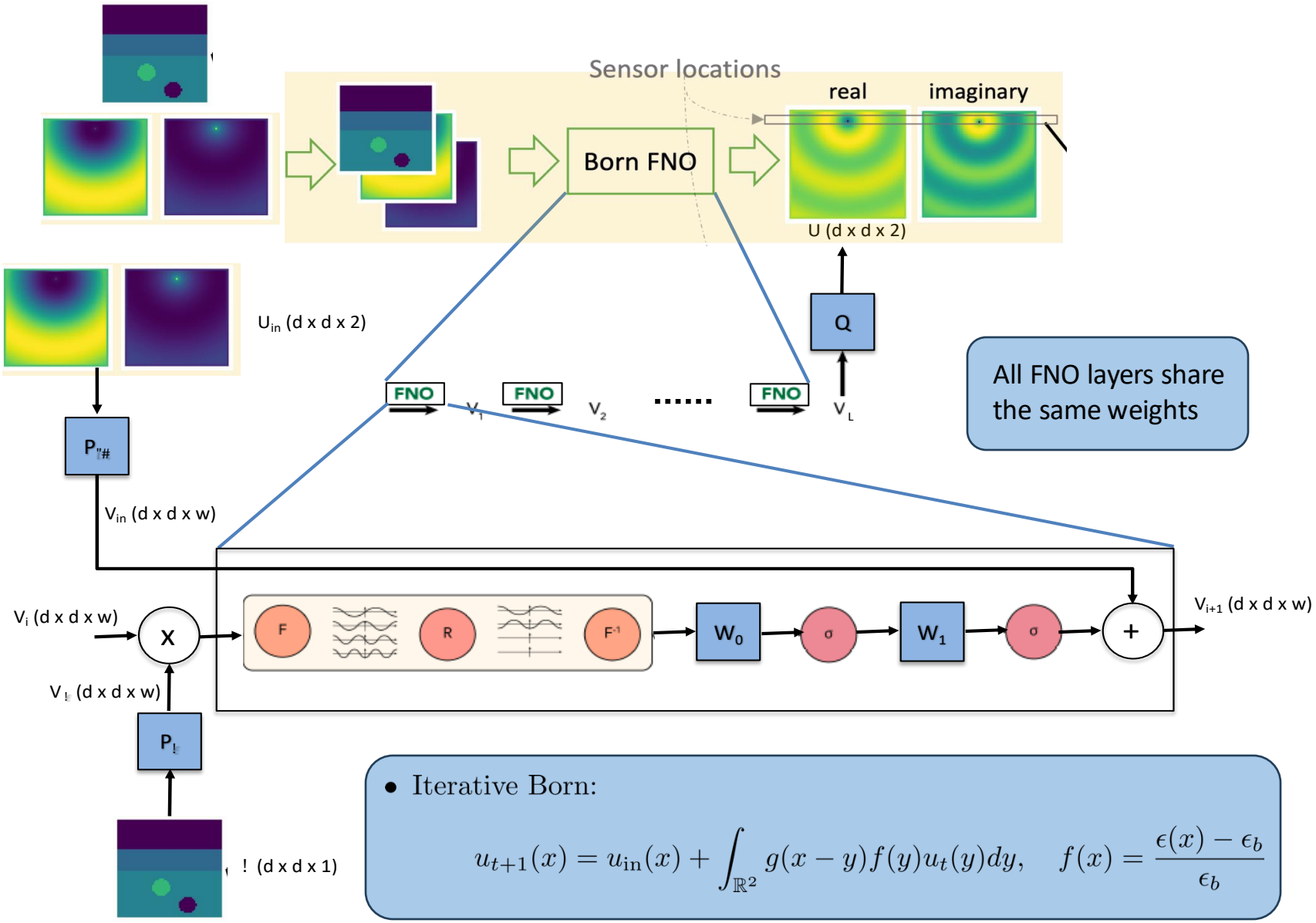
[2] Li, Zongyi, et al. "Fourier neural operator for parametric partial differential equations, arXiv." *arXiv preprint arXiv:2010.08895* (2020).

Born FNO for Scattering



- For 2D problems, $D \subset \mathbb{R}^2$
- Input function space $\mathcal{A}(D, \mathbb{R}^3)$: for $x \in D$, $x \mapsto a(x) := (\epsilon(x), u_{\text{in}}^{\mathcal{R}}(x), u_{\text{in}}^{\mathcal{I}}(x))$
- Output function space $\mathcal{U}(D, \mathbb{R}^2)$: for $x \in D$, $x \mapsto u(x) := (u^{\mathcal{R}}(x), u^{\mathcal{I}}(x))$

Born Series Inspired Network Structure



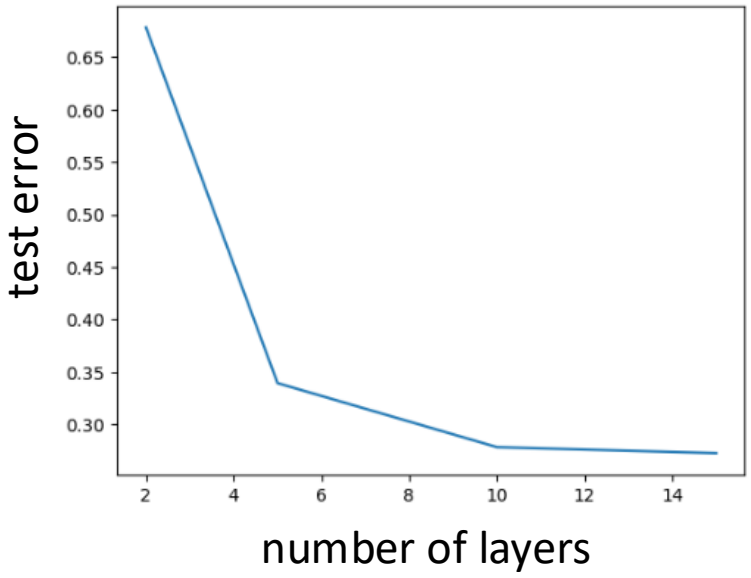
- Iterative Born:

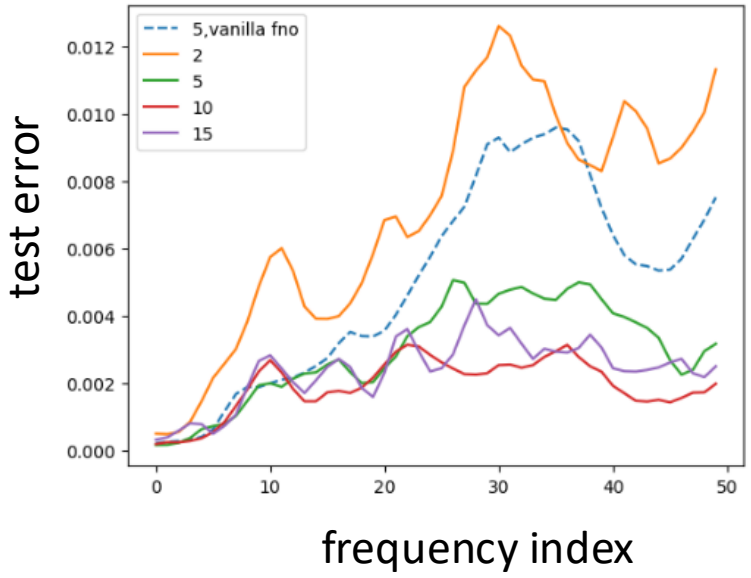
$$u_{t+1}(x) = u_{\text{in}}(x) + \int_{\mathbb{R}^2} g(x-y) f(y) u_t(y) dy, \quad f(x) = \frac{\epsilon(x) - \epsilon_b}{\epsilon_b}$$

Learned wave-object interaction - Born FNO

- We observe that **with more layers, we obtain better test time accuracy**
  - more layers could be interpretated as higher order born approximation

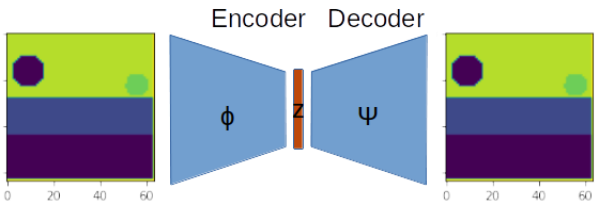
- We compare our proposed Born FNO with the Vanilla FNO that does not leverage the Born series approximation structure.
  - Our model achieves better performance given same number of layers.





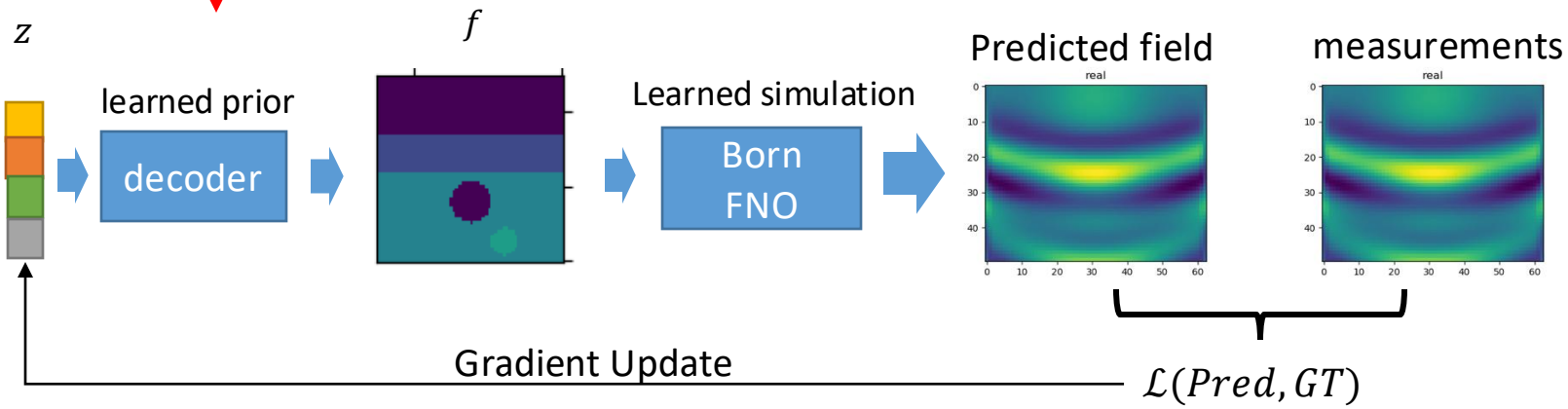
Born FNO for solving inverse problems

Prior Training:



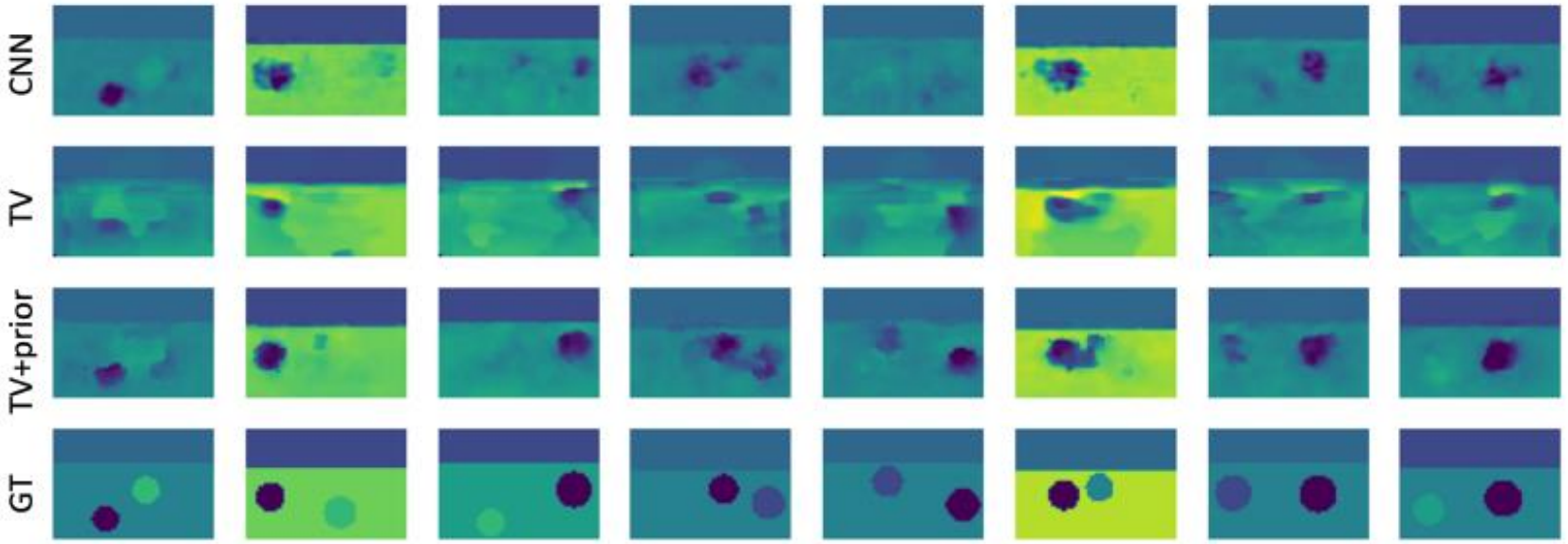
Why autoencoder prior:

- Learn parametric model that maps a low dimensional vector z to the target structure f
- during optimization, optimize $z \in R^n$, *rather than* $f \in R^{w \times h}$ *- constrain the solution space to be close to the target dataset distribution learned by the decoder*



Born FNO for solving the inverse problem

- Test results averaged over 50 samples



Zhao, Q., Ma, Y., Boufounos, P., Nabi, S., & Mansour, H. (2023, June). Deep born operator learning for reflection tomographic imaging. In *2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*