Multi-frequency progressive refinement for learned inverse scattering

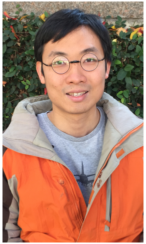
Vasilis Charisopoulos (UChicago $\rightarrow$ UW ECE)

IFDS Workshop, August 2025

My amazing collaborators

Owen Melia

Olivia Tsang



Yuehaw Khoo

Jeremy Hoskins

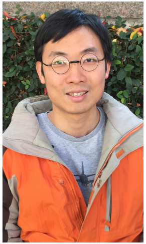
Rebecca Willett

My amazing collaborators

🎉 ✨ 💰

Owen Melia

Olivia Tsang

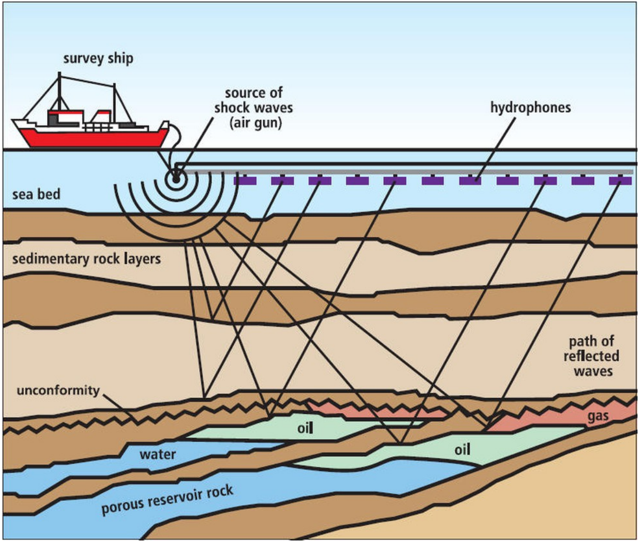


Yuehaw Khoo

Jeremy Hoskins

Rebecca Willett

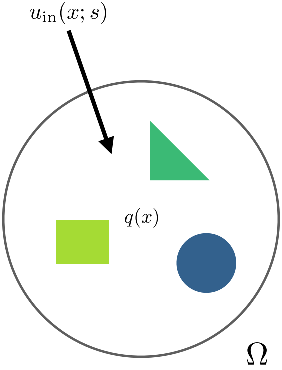
Wave scattering



Imaging setup: probing

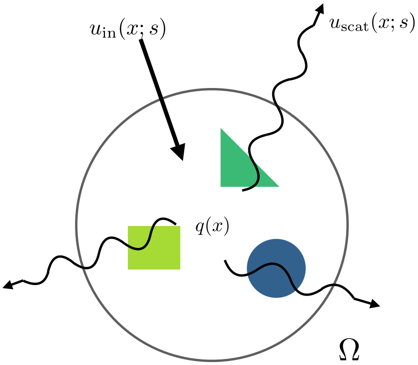
- Interested in **scattering potential** $q(x)$, $x \in \Omega$
- **Probe** object using plane waves:
  - Traveling in direction $s \in \mathbb{S}$
  - Wavelength $\lambda \to$ spatial frequency $k = \frac{2\pi}{\lambda}$
  - Constant wave speed outside Ω
- Incoming waves:

$$u_{\text{in}}(x; s) = \exp(ik \cdot \langle s, x \rangle).$$



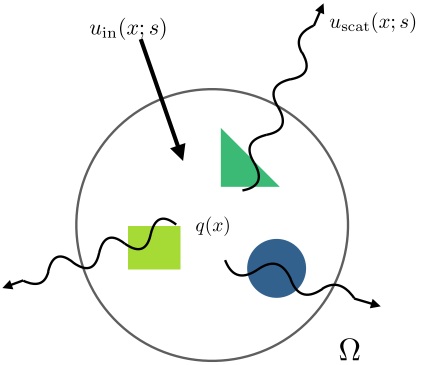
Imaging setup: scattering

- Interactions yield scattered wave $u_{\mathrm{sc}}(x; s)$
- Map $q \mapsto u_{\mathrm{sc}}(\cdot; s)$ is **nonlinear** in general!



Imaging setup: scattering

- Interactions yield scattered wave $u_{\mathrm{sc}}(x; s)$
- Map $q \mapsto u_{\mathrm{sc}}(\cdot; s)$ is **nonlinear** in general!



Lippmann-Schwinger integral equation: for appropriate G_k,

$$u_{\mathrm{sc}}(x; s) = k^2 \int_\Omega \underbrace{G_k}_{\text{Green's function}}(\|x - x'\|) q(x') \left(u_{\mathrm{in}}(x'; s) + u_{\mathrm{sc}}(x'; s)\right) \mathrm{d}x'$$

Imaging setup: measurements

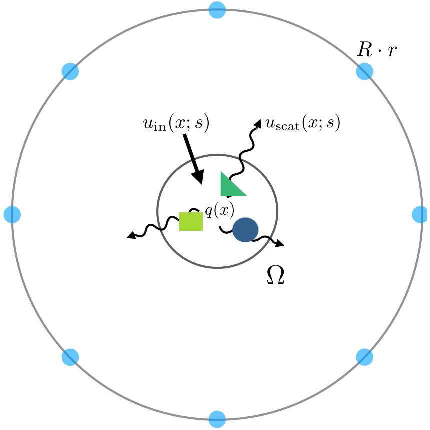
- Receiver direction $r \in \mathbb{S}$
- Observe $u_{\mathrm{sc}}(\cdot; s)$ on ring of radius $R \gg 1$:

$$\mathcal{F}_k[q](r, s) = u_{\mathrm{sc}}(R \cdot r; s)$$

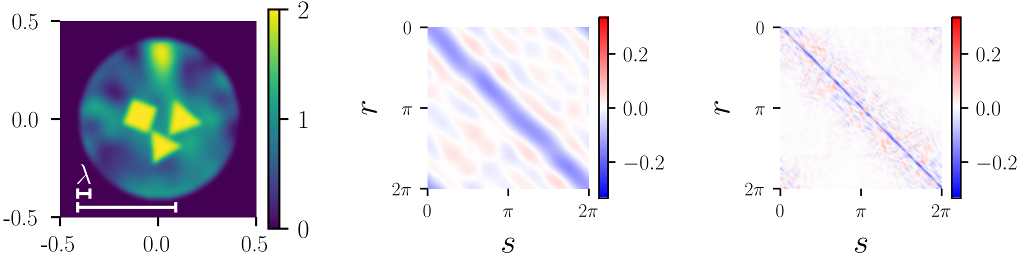
- **Far-field data**: collection of measurements

$$d_k := \{\mathcal{F}_k[q](r, s)\}_{(r,s):\text{evenly spaced on } \mathbb{S}},$$

over several spatial frequencies k.

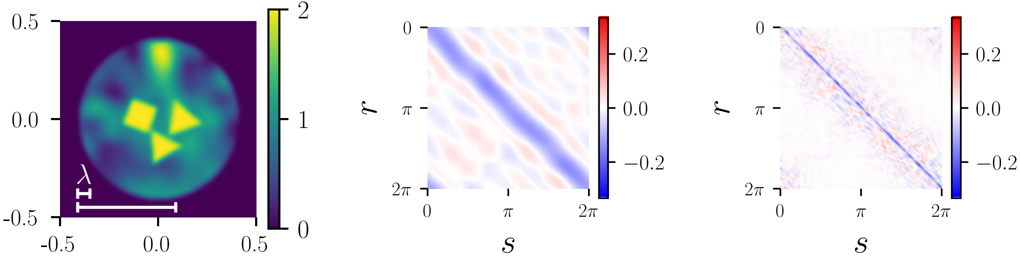


Imaging setup: measurements



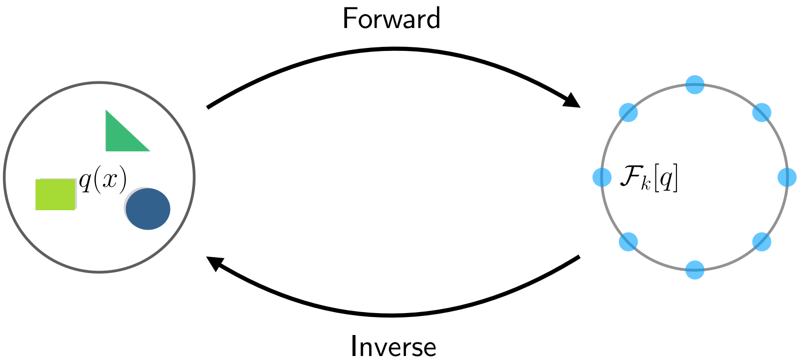
(a) Scatterer q **(b)** Measurements d_k $(k = 4\pi)$ **(c)** Measurements d_k $(k = 32\pi)$

Imaging setup: measurements



(a) Scatterer q **(b)** Measurements d_k $(k = 4\pi)$ **(c)** Measurements d_k $(k = 32\pi)$

Goal: recover q from far-field data $\{d_k\}$.


Forward
$q(x)$
$\mathcal{F}_k[q]$
Inverse

Single-scattering: the linear regime

Assumption: scattered wave depends *linearly* on scattering potential:

$$\mathcal{F}_k[q] \approx \mathcal{F}_k[\mathbf{0}] + \nabla\mathcal{F}_k[\mathbf{0}]q \equiv F_k q.$$

linear operator

Single-scattering: the linear regime

Assumption: scattered wave depends *linearly* on scattering potential:

$$\mathcal{F}_k[q] \approx \mathcal{F}_k[\mathbf{0}] + \nabla\mathcal{F}_k[\mathbf{0}]q \equiv F_k q.$$

linear operator

When is this approximation plausible?

$$u_{\mathrm{sc}}(x;s) = k^2 \int_{\Omega} G_k(\|x - x'\|)q(x')\left(u_{\mathrm{in}}(x';s) + u_{\mathrm{sc}}(x';s)\right)\mathrm{d}x' \tag{1}$$

1. **Low contrast**: small values of $q(x)$

Single-scattering: the linear regime

Assumption: scattered wave depends *linearly* on scattering potential:

$$\mathcal{F}_k[q] \approx \mathcal{F}_k[\mathbf{0}] + \nabla \mathcal{F}_k[\mathbf{0}] q \equiv F_k q.$$

linear operator

When is this approximation plausible?

$$u_{\mathrm{sc}}(x; s) = k^2 \int_{\Omega} G_k(\|x - x'\|) q(x') \left(u_{\mathrm{in}}(x'; s) + u_{\mathrm{sc}}(x'; s) \right) \mathrm{d}x' \tag{1}$$

1. **Low contrast**: small values of $q(x)$
2. **Narrow spatial support**: $q(x)$ is zero for most $x \in \Omega$

Single-scattering: the linear regime

Assumption: scattered wave depends *linearly* on scattering potential:

$$\mathcal{F}_k[q] \approx \mathcal{F}_k[\mathbf{0}] + \nabla\mathcal{F}_k[\mathbf{0}]q \equiv F_k q.$$

linear operator

When is this approximation plausible?

$$u_{\mathrm{sc}}(x;s) = k^2 \int_\Omega G_k(\|x - x'\|)q(x')\left(u_{\mathrm{in}}(x';s) + u_{\mathrm{sc}}(x';s)\right)\mathrm{d}x' \tag{1}$$

1. **Low contrast**: small values of $q(x)$
2. **Narrow spatial support**: $q(x)$ is zero for most $x \in \Omega$
3. **Low-frequency waves**: small wavenumber k.

Single-scattering: the linear regime

Assumption: scattered wave depends *linearly* on scattering potential:

$$\mathcal{F}_k[q] \approx \mathcal{F}_k[\mathbf{0}] + \nabla\mathcal{F}_k[\mathbf{0}]q \equiv F_k q.$$

linear operator

Algorithm: filtered back-projection (FBP).

$$\widehat{q} \approx (F_k^* F_k + \mu I)^{-1} F_k^* d_k. \quad (1)$$

Single-scattering: the linear regime

Assumption: scattered wave depends *linearly* on scattering potential:

$$\mathcal{F}_k[q] \approx \mathcal{F}_k[\mathbf{0}] + \nabla\mathcal{F}_k[\mathbf{0}]q \equiv F_k q.$$

linear operator

Algorithm: filtered back-projection (FBP).

$$\widehat{q} \approx (F_k^* F_k + \mu I)^{-1} F_k^* d_k. \tag{1}$$

Filtering operator (2D conv)

Single-scattering: the linear regime

Assumption: scattered wave depends *linearly* on scattering potential:

$$\mathcal{F}_k[q] \approx \mathcal{F}_k[\mathbf{0}] + \nabla \mathcal{F}_k[\mathbf{0}]q \equiv F_k q.$$

linear operator

Algorithm: filtered back-projection (FBP).

$$\widehat{q} \approx (F_k^* F_k + \mu I)^{-1} F_k^* d_k. \tag{1}$$

Filtering operator (2D conv)

Backprojection (1D conv)

Single-scattering: the linear regime

Assumption: scattered wave depends *linearly* on scattering potential:

$$\mathcal{F}_k[q] \approx \mathcal{F}_k[\mathbf{0}] + \nabla \mathcal{F}_k[\mathbf{0}]q \equiv \underbrace{F_k}_{\text{linear operator}} q.$$

Algorithm: filtered back-projection (FBP).

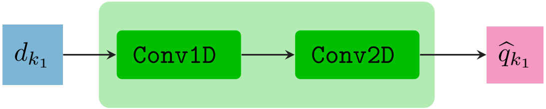
$$\widehat{q} \approx \underbrace{(F_k^* F_k + \mu I)^{-1}}_{\text{Filtering operator (2D conv)}} \underbrace{F_k^*}_{\text{Backprojection (1D conv)}} d_k. \tag{1}$$

✓ Classical method, straightforward to implement.

✗ Produces artifacts; **struggles with high-contrast data**.

Single-scattering using neural networks

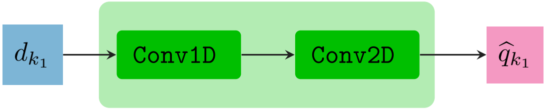
- **Idea**: replace filtering and backprojection with trainable 2D and 1D convolutions.[1]



[1] Fan & Ying, '22

Single-scattering using neural networks

- **Idea**: replace filtering and backprojection with trainable 2D and 1D convolutions.[1]

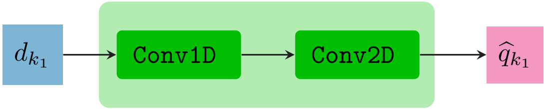


- Related works:
  - `SwitchNet`[2]: encode via product of sparse structured matrices
  - Wide-band Butterfly Net[3]: represent hierarchical structure via butterfly factorization

[1] Fan & Ying, '22
[2] Khoo & Ying, '22
[3] Li et al., '22

Single-scattering using neural networks

- **Idea**: replace filtering and backprojection with trainable 2D and 1D convolutions.[1]



- Related works:
  - `SwitchNet`[2]: encode via product of sparse structured matrices
  - Wide-band Butterfly Net[3]: represent hierarchical structure via butterfly factorization

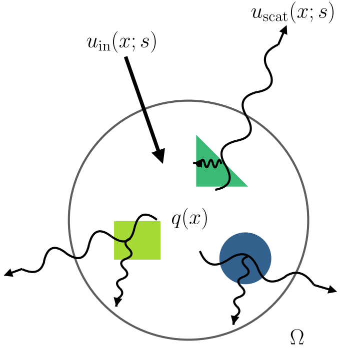
✗ Existing approaches **struggle** with high-contrast data / inhomogeneous backgrounds!

---

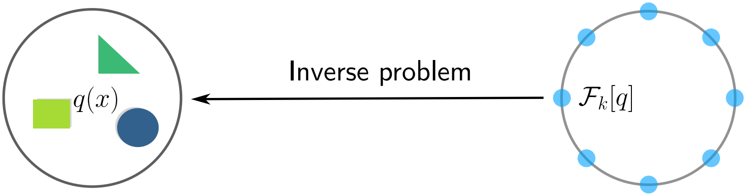
[1] Fan & Ying, '22
[2] Khoo & Ying, '22
[3] Li et al., '22

Multiple scattering: the nonlinear regime

- Scattered wave itself interacts with object
- Map from $q(x)$ to $u_{\mathrm{sc}}(x; s)$ **highly nonlinear**
- Scatterer q has high contrast / wide spatial support
- Wavenumber k is high



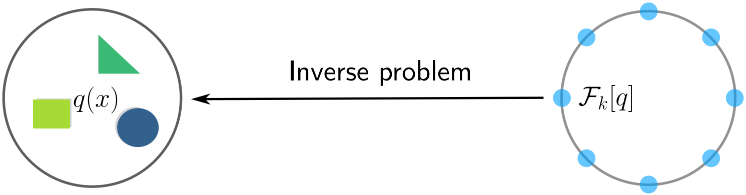
Nonlinear regime: optimization perspective



- **Approach**: estimate q by solving optimization problem

$$\widehat{q} = \underset{q}{\operatorname{argmin}} \left\| \mathcal{F}_k[q] - d_k \right\|^2$$

Nonlinear regime: optimization perspective

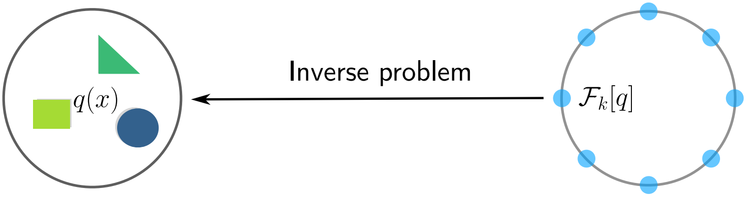


- **Approach**: estimate q by solving optimization problem

$$\widehat{q} = \underset{q}{\operatorname{argmin}} \left\| \mathcal{F}_k[q] - d_k \right\|^2$$

- Nonconvex objective $\to$ spurious local minima!

Nonlinear regime: optimization perspective

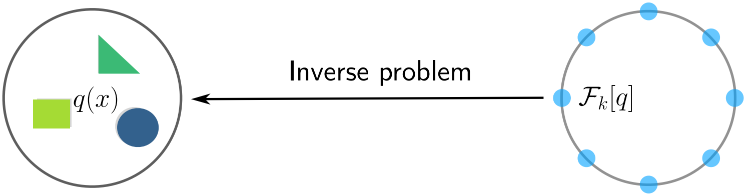


- **Approach**: estimate q by solving optimization problem

$$\widehat{q} = \underset{q}{\operatorname{argmin}} \left\| \mathcal{F}_k[q] - d_k \right\|^2$$

- Nonconvex objective $\to$ spurious local minima!
- **When** can we **avoid** "bad" solutions?
  - When the initial guess $\widehat{q}_0$ is good;
  - When wavenumber k is low (since $\mathcal{F}_k[\cdot] \approx F_k$).

Nonlinear regime: optimization perspective



- **Approach**: estimate q by solving optimization problem

$$\widehat{q} = \underset{q}{\operatorname{argmin}} \|\mathcal{F}_k[q] - d_k\|^2$$

- Nonconvex objective $\rightarrow$ spurious local minima!

- **When** can we **avoid** "bad" solutions?
  - When the initial guess $\widehat{q}_0$ is good;
  - When wavenumber k is low (since $\mathcal{F}_k[\cdot] \approx F_k$).

- **Idea**: use *low-frequency estimates* to initialize *high-frequency solves*.

Nonlinear regime: optimization perspective

Illustration: optimization over family of Gaussians.

$$\widehat{q} = \underset{q}{\operatorname{argmin}} \left\| \mathcal{F}_k[q] - d_k \right\|^2, \quad \text{where} \quad q(x) = \beta \exp\left(- \frac{\|x\|^2}{2\sigma^2} \right).$$

Nonlinear regime: optimization perspective

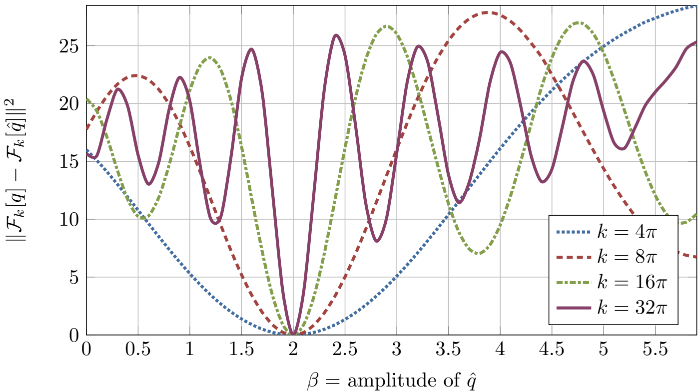
Illustration: optimization over family of Gaussians.

$$\widehat{q} = \operatorname*{argmin}_{q} \left\| \mathcal{F}_k[q] - d_k \right\|^2, \quad \text{where} \quad q(x) = \underbrace{\beta}_{\text{unknown amplitude}} \exp\left(-\frac{\|x\|^2}{2\underbrace{\sigma}_{\text{known bandwidth}}^2} \right).$$

Nonlinear regime: optimization perspective

Illustration: optimization over family of Gaussians.

$$\widehat{q} = \operatorname*{argmin}_q \|\mathcal{F}_k[q] - d_k\|^2, \quad \text{where} \quad q(x) = \underbrace{\beta}_{\text{unknown amplitude}} \exp\left(-\frac{\|x\|^2}{2\underbrace{\sigma}_{\text{known bandwidth}}^2}\right).$$



Multiple scattering: recursive linearization

- "Warm-starting" from low-frequency solutions.[4]

$$\begin{cases} \widehat{q}_{k_1} \leftarrow \left(F_{k_1}^* F_{k_1} + \mu I\right)^{-1} F_{k_1}^* d_{k_1}. \\ \delta q_{k_t} \leftarrow \underset{\delta q}{\operatorname{argmin}} \left\| d_{k_t} - \left(\mathcal{F}_{k_t}[\widehat{q}_{k_{t-1}}] + \nabla \mathcal{F}_{k_t}[\widehat{q}_{k_{t-1}}]\delta q\right) \right\|^2, \quad \widehat{q}_{k_t} = \widehat{q}_{k_{t-1}} + \delta q_{k_t}. \end{cases}$$

[4] *Recursive linearization for Inverse Scattering*, Chen '95

Multiple scattering: recursive linearization

- "Warm-starting" from low-frequency solutions.[4]

$$\begin{cases} \widehat{q}_{k_1} \leftarrow \left(F_{k_1}^* F_{k_1} + \mu I\right)^{-1} F_{k_1}^* d_{k_1} . \\ \delta q_{k_t} \leftarrow \underset{\delta q}{\operatorname{argmin}} \left\| d_{k_t} - \left(\mathcal{F}_{k_t}[\widehat{q}_{k_{t-1}}] + \nabla \mathcal{F}_{k_t}[\widehat{q}_{k_{t-1}}]\delta q\right)\right\|^2 , \quad \widehat{q}_{k_t} = \widehat{q}_{k_{t-1}} + \delta q_{k_t} . \end{cases}$$

- **Effective for high-resolution recovery** in practice.

[4] *Recursive linearization for Inverse Scattering*, Chen '95

Multiple scattering: recursive linearization

- "Warm-starting" from low-frequency solutions.[4]

$$\begin{cases} \widehat{q}_{k_1} \leftarrow \left(F_{k_1}^* F_{k_1} + \mu I\right)^{-1} F_{k_1}^* d_{k_1} . \\ \delta q_{k_t} \leftarrow \underset{\delta q}{\operatorname{argmin}} \left\| d_{k_t} - \left(\mathcal{F}_{k_t}[\widehat{q}_{k_{t-1}}] + \nabla \mathcal{F}_{k_t}[\widehat{q}_{k_{t-1}}] \delta q\right) \right\|^2 , \quad \widehat{q}_{k_t} = \widehat{q}_{k_{t-1}} + \delta q_{k_t} . \end{cases}$$

- **Effective for high-resolution recovery** in practice.
- **Computationally expensive**: multiple PDE solves for each k and direction r.

[4] *Recursive linearization for Inverse Scattering*, Chen '95

Multiple scattering: recursive linearization

- "Warm-starting" from low-frequency solutions.[4]

$$\begin{cases} \widehat{q}_{k_1} \leftarrow \left(F_{k_1}^* F_{k_1} + \mu I\right)^{-1} F_{k_1}^* d_{k_1}. \\ \delta q_{k_t} \leftarrow \underset{\delta q}{\operatorname{argmin}} \left\| d_{k_t} - \left(\mathcal{F}_{k_t}[\widehat{q}_{k_{t-1}}] + \nabla \mathcal{F}_{k_t}[\widehat{q}_{k_{t-1}}]\delta q\right) \right\|^2, \quad \widehat{q}_{k_t} = \widehat{q}_{k_{t-1}} + \delta q_{k_t}. \end{cases}$$

- **Effective for high-resolution recovery** in practice.
- **Computationally expensive**: multiple PDE solves for each k and direction r.
- Requires **many measured frequencies**:
  - Example: **277** frequencies and over **40** hours to recover single 192×192 image[5].

[4] *Recursive linearization for Inverse Scattering*, Chen '95

[5] Borges et al. '17

Multiple scattering: recursive linearization

- "Warm-starting" from low-frequency solutions.[4]

$$\begin{cases} \widehat{q}_{k_1} \leftarrow \left(F_{k_1}^* F_{k_1} + \mu I\right)^{-1} F_{k_1}^* d_{k_1}. \\ \delta q_{k_t} \leftarrow \underset{\delta q}{\operatorname{argmin}} \left\| d_{k_t} - \left(\mathcal{F}_{k_t}[\widehat{q}_{k_{t-1}}] + \nabla \mathcal{F}_{k_t}[\widehat{q}_{k_{t-1}}] \delta q\right) \right\|^2, \quad \widehat{q}_{k_t} = \widehat{q}_{k_{t-1}} + \delta q_{k_t}. \end{cases}$$

- **Effective for high-resolution recovery** in practice.
- **Computationally expensive**: multiple PDE solves for each k and direction r.
- Requires **many measured frequencies**:
  - Example: **277** frequencies and over **40** hours to recover single 192×192 image[5].

Ours: ML-based approach that emulates recursive linearization method.

[4] *Recursive linearization for Inverse Scattering*, Chen '95
[5] Borges et al. '17

Our approach

Motivation: two key features of recursive linearization.

- **Progressive refinement**: maintain intermediate estimates of the scattering potential, progressively refine them with introduction of new data.

Our approach

Motivation: two key features of recursive linearization.

- **Progressive refinement**: maintain intermediate estimates of the scattering potential, progressively refine them with introduction of new data.
- **Homotopy in frequency**: iterative refinements form a homotopy from low to high frequency measurements. Updates at step t contain high-freq information relative to k_{t-1}.

Our approach

Motivation: two key features of recursive linearization.

- **Progressive refinement**: maintain intermediate estimates of the scattering potential, progressively refine them with introduction of new data.
- **Homotopy in frequency**: iterative refinements form a homotopy from low to high frequency measurements. Updates at step t contain high-freq information relative to k_{t-1}.

Proposal: **M**ulti-**F**requency **I**nverse **S**cattering **Net**work (MFISNet).

- Composition of "refinement blocks" (trainable convolutional networks);
- **Key idea**: guide successive blocks to perform homotopy through frequency.

Progressive refinement: pseudocode

Algorithm Progressive refinement scheme

1: **Input**: multi-freq data $\left\{d_{k_1}, \ldots, d_{k_{N_f}}\right\}$
2: $\widehat{q}_{k_1} := (F_{k_1}^* F_{k_1} + \mu I)^{-1} F_{k_1}^* d_{k_1}$ ▷ Filtered backprojection
3: **for** $t = 2, \ldots, N_f$ **do**
4: $\quad \delta q := \texttt{RefinementBlock}_{\theta_t}(\widehat{q}_{k_{t-1}}, d_{k_t})$ ▷ Neural network
5: $\quad \widehat{q}_{k_t} := \widehat{q}_{k_{t-1}} + \delta q$
6: **end for**
7: **return** $\widehat{q}_{k_{N_f}}$

Trainable parameters

Progressive refinement: pseudocode

Algorithm Progressive refinement scheme

1: **Input**: multi-freq data $\{d_{k_1}, \ldots, d_{k_{N_f}}\}$
2: $\widehat{q}_{k_1} := (F^*_{k_1} F_{k_1} + \mu I)^{-1} F^*_{k_1} d_{k_1}$ ▷ Filtered backprojection
3: **for** $t = 2, \ldots, N_f$ **do**
4: $\quad \delta q := \texttt{RefinementBlock}_{\theta_t}(\widehat{q}_{k_{t-1}}, d_{k_t})$ ▷ Neural network
5: $\quad \widehat{q}_{k_t} := \widehat{q}_{k_{t-1}} + \delta q$
6: **end for**
7: **return** $\widehat{q}_{k_{N_f}}$

Trainable parameters

What makes a good `RefinementBlock`?

Implementation: refinement block

- **Design goal**: learn refinement step from data, avoid expensive PDE solves.
- Updates should contain high-frequency information.
- Residual connections to preserve low-frequency information.
- FYNet: neural net emulating FBP from (Fan & Ying, '22).

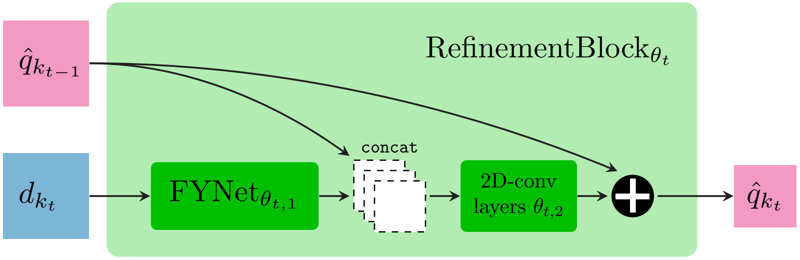


Figure: Proposed architecture; green blocks indicate trainable parameters.

Implementation: refinement block

- **Design goal**: learn refinement step from data, avoid expensive PDE solves.
- Updates should contain high-frequency information.
- Residual connections to preserve low-frequency information.
- FYNet: neural net emulating FBP from (Fan & Ying, '22).

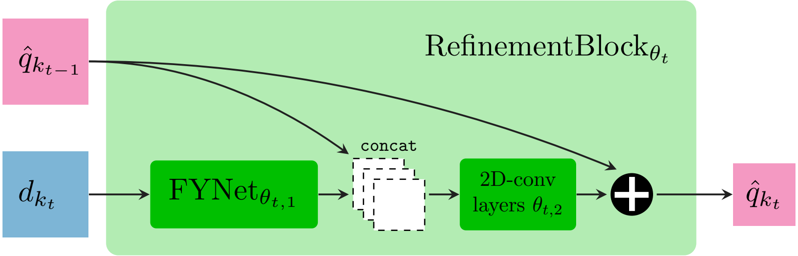
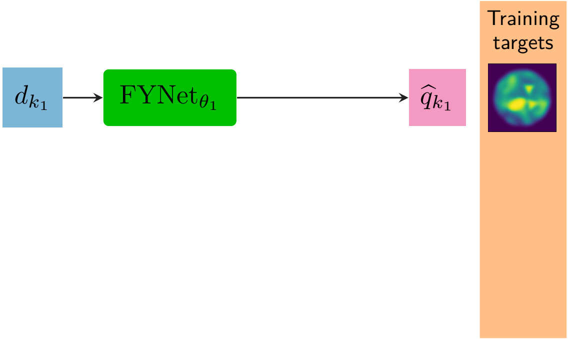


Figure: Proposed architecture; green blocks indicate trainable parameters.

How should we train the resulting network?

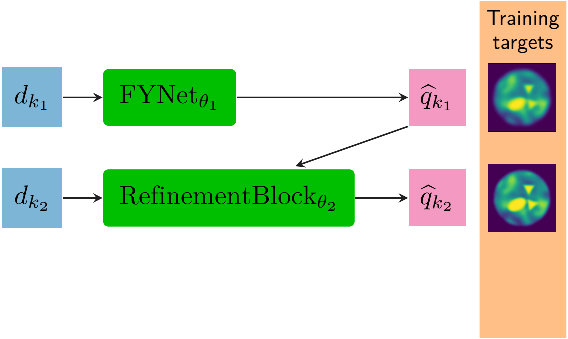
Implementation: homotopy through frequency

- **Idea**: pretrain blocks from coarser to finer scales.
- Output of t^{th} block should match $\texttt{LPF}_{k_t}(q)$, where $\texttt{LPF}$ is appropriate low-pass filter.
- After pretraining, train blocks jointly so that $\widehat{q}_{k_{\text{final}}} \to q$.



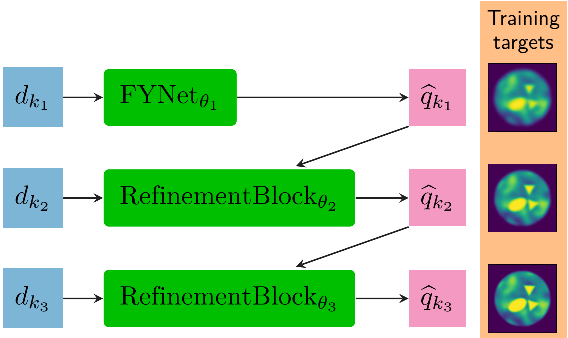
Implementation: homotopy through frequency

- **Idea**: pretrain blocks from coarser to finer scales.
- Output of t^{th} block should match $\mathtt{LPF}_{k_t}(q)$, where $\mathtt{LPF}$ is appropriate low-pass filter.
- After pretraining, train blocks jointly so that $\widehat{q}_{k_{\text{final}}} \to q$.



Implementation: homotopy through frequency

- **Idea**: pretrain blocks from coarser to finer scales.
- Output of t^{th} block should match $\texttt{LPF}_{k_t}(q)$, where $\texttt{LPF}$ is appropriate low-pass filter.
- After pretraining, train blocks jointly so that $\widehat{q}_{k_{\text{final}}} \to q$.

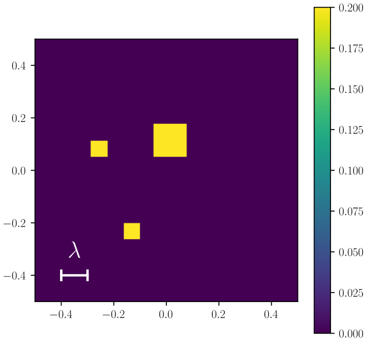


Experimental evaluation: new dataset of scatterers

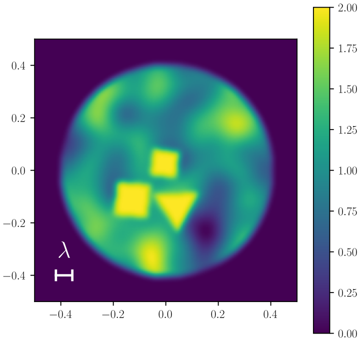
- Wider spatial support; 5x - 10x higher contrast $\|q\|_\infty$.
- Random selection of geometric shapes overlaid on smoothly-varying background
- Available from: $\boxed{\texttt{https://doi.org/10.5281/zenodo.14514353}}$.

Experimental evaluation: new dataset of scatterers

- Wider spatial support; 5x - 10x higher contrast $\|q\|_\infty$.
- Random selection of geometric shapes overlaid on smoothly-varying background
- Available from: $\boxed{\texttt{https://doi.org/10.5281/zenodo.14514353}}$.



(a) Sample from existing benchmark



(b) Sample from our dataset

Experimental evaluation: setup

Test with $N_f = 1, \ldots, 5$ different frequencies.

- Use the highest N_f from $\{2\pi, 4\pi, 8\pi, 16\pi, 32\pi\}$ as $k_1, \ldots, k_{N_f}$
- Observe $n := \lceil \frac{10000}{N_f} \rceil$ scattering potentials per frequency.
  - Selected to ensure the total number of measurements is independent of N_f.
- Compare the following methods:
  - **FYNet**: FBP-based method (Fan & Ying, '22)
  - Wide-band butterfly network (Li et al. '22)
  - Two "naive" multiscale baselines (MFISNet-Fused/Parallel)
  - The proposed method (MFISNet-Refinement).

Experimental evaluation: recovered images

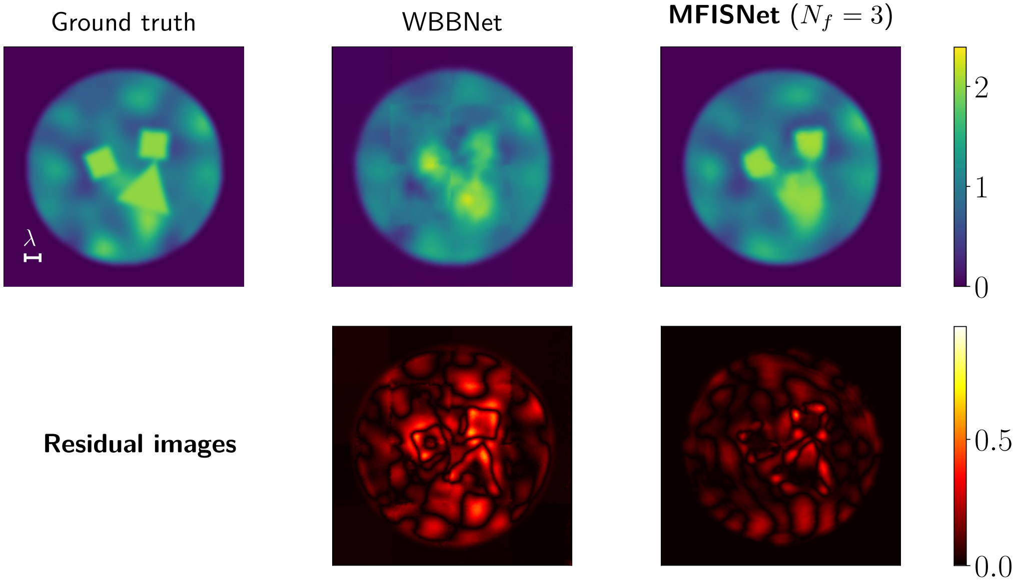


Figure: Random test sample, reconstructions and residuals

Experimental evaluation: performance comparison

N_f	$[k_1, k_2, ...]$	n	**Method Name**	**Relative ℓ_2 Error**
1	$[32\pi]$	10000	FYNet	0.261 ± 0.036
2	$[16\pi, 32\pi]$	5000	MFISNet-Fused	0.177 ± 0.033
			MFISNet-Parallel	0.168 ± 0.029
			MFISNet-Refinement (**Ours**)	$\mathbf{0.154} \pm \mathbf{0.034}$
3	$[8\pi, 16\pi, 32\pi]$	3333	Wide-Band Butterfly Network	0.160 ± 0.037
			MFISNet-Fused	0.130 ± 0.025
			MFISNet-Parallel	0.114 ± 0.021
			MFISNet-Refinement (**Ours**)	$\mathbf{0.098} \pm \mathbf{0.020}$
4	$[4\pi, 8\pi, 16\pi, 32\pi]$	2500	MFISNet-Fused	0.105 ± 0.021
			MFISNet-Parallel	0.110 ± 0.021
			MFISNet-Refinement (**Ours**)	$\mathbf{0.086} \pm \mathbf{0.017}$
5	$[2\pi, 4\pi, 8\pi, 16\pi, 32\pi]$	2000	MFISNet-Fused	0.115 ± 0.022
			MFISNet-Parallel	0.108 ± 0.021
			MFISNet-Refinement (**Ours**)	$\mathbf{0.084} \pm \mathbf{0.018}$

It's about time

Recursive linearization (2017):

- 277 forward model evaluations (**serially**) *for each new test sample*;
- reported over 40 hours to recover a single image (on 2017 computers);
- increasing frequency spacing affects convergence.

Our approach:

- 10000 forward model evaluations (**in parallel**) to generate dataset;
- ≈ 15 hours of compute to generate training samples on a **single** node;
- 1.5 hours to train model, $\leq 0.1\mathrm{s}$ to process each new image (for $N_f = 5$).

Experimental evaluation: ablation study

What is responsible for the performance gains?

Experimental evaluation: ablation study

What is responsible for the performance gains?

- **Question 1**: Is sequential pretraining necessary?
  - Alternative: train in one phase, incorporate intermediate reconstructions into loss.

$$\mathcal{L} := \|\hat{q}_{N_f} - q\|^2 + \sum_{t=1}^{N_f-1} \gamma^{-t} \|\hat{q}_{k_t} - \mathtt{LPF}_{k_t}(q)\|^2, \quad \gamma \in (0,1).$$

Experimental evaluation: ablation study

What is responsible for the performance gains?

- **Question 1**: Is sequential pretraining necessary?
  - Alternative: train in one phase, incorporate intermediate reconstructions into loss.

$$\mathcal{L} := \left\|\hat{q}_{N_f} - q\right\|^2 + \sum_{t=1}^{N_f-1} \gamma^{-t} \left\|\hat{q}_{k_t} - \mathtt{LPF}_{k_t}(q)\right\|^2, \quad \gamma \in (0,1).$$

- **Question 2**: Are intermediate reconstructions necessary?
  - Alternative: train in one phase using only the final reconstruction in the loss.

$$\mathcal{L} := \left\|\hat{q}_{N_f} - q\right\|^2.$$

Experimental evaluation: ablation study

What is responsible for the performance gains?

Table: Ablation study

Training Method	$[k_1, k_2, ...]$	**Relative L2 Error**
MFISNet-Fused	$[2\pi, 4\pi, 8\pi, 16\pi, 32\pi]$	0.115 ± 0.022
MFISNet-Parallel	$[2\pi, 4\pi, 8\pi, 16\pi, 32\pi]$	0.108 ± 0.021
No progressive refinement	$[2\pi, 4\pi, 8\pi, 16\pi, 32\pi]$	0.095 ± 0.018
No sequential pre-training	$[2\pi, 4\pi, 8\pi, 16\pi, 32\pi]$	0.090 ± 0.017
Our Method	$[2\pi, 4\pi, 8\pi, 16\pi, 32\pi]$	$\mathbf{0.084 \pm 0.018}$

Recursive linearization revisited

Why are traditional methods so expensive?

Recursive linearization revisited

Why are traditional methods so expensive?

Computational bottleneck: forward model + Jacobian evals during Gauss-Newton step.

$$\delta q \leftarrow \left(\nabla \mathcal{F}_k[\widehat{q}_{\mathsf{prev}}]^* \nabla \mathcal{F}_k[\widehat{q}_{\mathsf{prev}}] + \mu I\right)^{-1} \nabla \mathcal{F}_k[\widehat{q}_{\mathsf{prev}}]^* \left(\mathcal{F}_k[\widehat{q}_{\mathsf{prev}}] - d_k\right)$$

Recursive linearization revisited

Why are traditional methods so expensive?

Computational bottleneck: forward model + Jacobian evals during Gauss-Newton step.

$$\delta q \leftarrow \left(\nabla \mathcal{F}_k[\widehat{q}_{\mathsf{prev}}]^* \nabla \mathcal{F}_k[\widehat{q}_{\mathsf{prev}}] + \mu I\right)^{-1} \nabla \mathcal{F}_k[\widehat{q}_{\mathsf{prev}}]^* \left(\mathcal{F}_k[\widehat{q}_{\mathsf{prev}}] - d_k\right)$$

Reason: evaluating $q \mapsto \mathcal{F}_k[q]$ and the JVP / VJP primitives

$$v \mapsto \nabla \mathcal{F}_k[q] v, \quad u \mapsto \nabla \mathcal{F}_k[q]^* u$$

require solving a nonlinear Helmholtz PDE.[6]

[6] Borges et al., 2017

Recursive linearization revisited

Why are traditional methods so expensive?

Computational bottleneck: forward model + Jacobian evals during Gauss-Newton step.

$$\delta q \leftarrow \left(\nabla\mathcal{F}_k[\widehat{q}_{\text{prev}}]^*\nabla\mathcal{F}_k[\widehat{q}_{\text{prev}}] + \mu I\right)^{-1}\nabla\mathcal{F}_k[\widehat{q}_{\text{prev}}]^*\left(\mathcal{F}_k[\widehat{q}_{\text{prev}}] - d_k\right)$$

Reason: evaluating $q \mapsto \mathcal{F}_k[q]$ and the JVP / VJP primitives

$$v \mapsto \nabla\mathcal{F}_k[q]v, \quad u \mapsto \nabla\mathcal{F}_k[q]^*u$$

require solving a nonlinear Helmholtz PDE.[6]

Can we take advantage of modern hardware accelerators (GPUs)?

[6] Borges et al., 2017

Sponsored content: `jaxhps`[7]

What is it?

- GPU-accelerated solver (written in `Jax`) for elliptic PDEs.
- Modern take on Hierarchical Poincare-Steklov solvers (HPS).

Key benefit: can **autodiff** through applications of forward model $\mathcal{F}_k$ (+ speed).

`pip install jaxhps` — `github.com/meliao/jaxhps`

[7] Melia, Fortunato, Hoskins & Willett, 2025

Ongoing work: GPU-accelerated recovery methods

What to do with a GPU solver? Two options:

1. Integrate differentiable forward model with MFISNet-like solutions;

Ongoing work: GPU-accelerated recovery methods

What to do with a GPU solver? Two options:

1. Integrate differentiable forward model with MFISNet-like solutions;
2. Accelerate classical methods (e.g., prox-gradient, Gauss-Newton).

Ongoing work: GPU-accelerated recovery methods

What to do with a GPU solver? Two options:

1. Integrate differentiable forward model with MFISNet-like solutions;
2. Accelerate classical methods (e.g., prox-gradient, Gauss-Newton).

This talk: a low-overhead method based on accelerated proximal gradient.

$$\widehat{x} \in \operatorname*{argmin}_x f(x) + h(x)$$

Regularizer (non-differentiable)

Data fidelity (differentiable)

Ongoing work: GPU-accelerated recovery methods

What to do with a GPU solver? Two options:

1. Integrate differentiable forward model with MFISNet-like solutions;
2. Accelerate classical methods (e.g., prox-gradient, Gauss-Newton).

This talk: a low-overhead method based on accelerated proximal gradient.

$$\widehat{x} \in \underset{x}{\operatorname{argmin}}\, f(x) + h(x)$$

Regularizer (non-differentiable) ($h(x)$)

Data fidelity (differentiable) ($f(x)$)

For such problems, the proximal gradient method iterates:

$$(\texttt{ProxGrad}): \qquad \widehat{x}_{t+1} = \mathbf{prox}_{\tau h}\left(\widehat{x}_t - \eta_t \nabla f(\widehat{x}_t)\right)$$
$$\equiv \underset{x}{\operatorname{argmin}} \left\{ \langle \nabla f(\widehat{x}_t), x - \widehat{x}_t \rangle + \frac{1}{2\eta_t} \|x - \widehat{x}_t\|^2 + \tau h(x) \right\}.$$

Ongoing work: TV-regularized inverse scattering

$$(\texttt{ProxGrad}) \qquad \widehat{x}_{t+1} = \mathbf{prox}_{\tau h}\left(\widehat{x}_t - \eta_t \nabla f(\widehat{x}_t)\right)$$

Ongoing work: TV-regularized inverse scattering

$$(\texttt{ProxGrad}) \qquad \widehat{x}_{t+1} = \mathbf{prox}_{\tau h}\left(\widehat{x}_t - \eta_t \nabla f(\widehat{x}_t)\right)$$

Inverse scattering: $f(q) = \left\|\mathcal{F}_k[q] - d_k\right\|^2$ and $h(q) = \|q\|_{\mathsf{TV}}$ (anisotropic total variation).[8]

[8] Rudin-Osher-Fatemi, 1992

Ongoing work: TV-regularized inverse scattering

$$(\texttt{ProxGrad}) \qquad \widehat{x}_{t+1} = \mathbf{prox}_{\tau h}\left(\widehat{x}_t - \eta_t \nabla f(\widehat{x}_t)\right)$$

Inverse scattering: $f(q) = \|\mathcal{F}_k[q] - d_k\|^2$ and $h(q) = \|q\|_{\mathsf{TV}}$ (anisotropic total variation).[8]

- **Issues**: (i) no analytical form for $\mathbf{prox}_{\|\cdot\|_{\mathsf{TV}}}$, (ii) expensive to eval $f(q)$ and $\nabla f(q)$.

---

[8] Rudin-Osher-Fatemi, 1992

Ongoing work: TV-regularized inverse scattering

$$(\texttt{ProxGrad}) \qquad \widehat{x}_{t+1} = \mathbf{prox}_{\tau h}\left(\widehat{x}_t - \eta_t \nabla f(\widehat{x}_t)\right)$$

Inverse scattering: $f(q) = \|\mathcal{F}_k[q] - d_k\|^2$ and $h(q) = \|q\|_{\mathsf{TV}}$ (anisotropic total variation).[8]

- **Issues**: (i) no analytical form for $\mathbf{prox}_{\|\cdot\|_{\mathsf{TV}}}$, (ii) expensive to eval $f(q)$ and $\nabla f(q)$.
- **Past work** (`CISOR`):[9] numerical approximation to $\nabla f(q)$ via truncated Born series;

[8] Rudin-Osher-Fatemi, 1992
[9] Ma et al., 2018

Ongoing work: TV-regularized inverse scattering

$$(\texttt{ProxGrad}) \qquad \widehat{x}_{t+1} = \mathbf{prox}_{\tau h}\left(\widehat{x}_t - \eta_t \nabla f(\widehat{x}_t)\right)$$

Inverse scattering: $f(q) = \|\mathcal{F}_k[q] - d_k\|^2$ and $h(q) = \|q\|_{\mathsf{TV}}$ (anisotropic total variation).[8]

- **Issues**: (i) no analytical form for $\mathbf{prox}_{\|\cdot\|_{\mathsf{TV}}}$, (ii) expensive to eval $f(q)$ and $\nabla f(q)$.
- **Past work** (`CISOR`):[9] numerical approximation to $\nabla f(q)$ via truncated Born series;
- **Ours**: GPU-accelerated gradients, simple splitting method for $\mathbf{prox}_{\tau\|\cdot\|_{\mathsf{TV}}}$.[10]

---

[8] Rudin-Osher-Fatemi, 1992
[9] Ma et al., 2018
[10] **C**. & Willett, 2025

Ongoing work: TV-regularized inverse scattering

$$(\texttt{ProxGrad}) \qquad \widehat{x}_{t+1} = \mathbf{prox}_{\tau h}\left(\widehat{x}_t - \eta_t \nabla f(\widehat{x}_t)\right)$$

Inverse scattering: $f(q) = \|\mathcal{F}_k[q] - d_k\|^2$ and $h(q) = \|q\|_{\mathsf{TV}}$ (anisotropic total variation).[8]

- **Issues**: (i) no analytical form for $\mathbf{prox}_{\|\cdot\|_{\mathsf{TV}}}$, (ii) expensive to eval $f(q)$ and $\nabla f(q)$.
- **Past work** (`CISOR`):[9] numerical approximation to $\nabla f(q)$ via truncated Born series;
- **Ours**: GPU-accelerated gradients, simple splitting method for $\mathbf{prox}_{\tau\|\cdot\|_{\mathsf{TV}}}$.[10]

Key benefit: can use 10 - 100x larger stepsizes than `CISOR`.

[8] Rudin-Osher-Fatemi, 1992
[9] Ma et al., 2018
[10] **C**. & Willett, 2025

Experiments: TV-regularized reconstructions

Dataset: 2D inhomogeneous objects from Institut Fresnel dataset.[11]

Setup: far-field data at $4\mathrm{GHz}$; $\widehat{q}_0 = \mathbf{0}$; error tol $\varepsilon = 10^{-3}$.

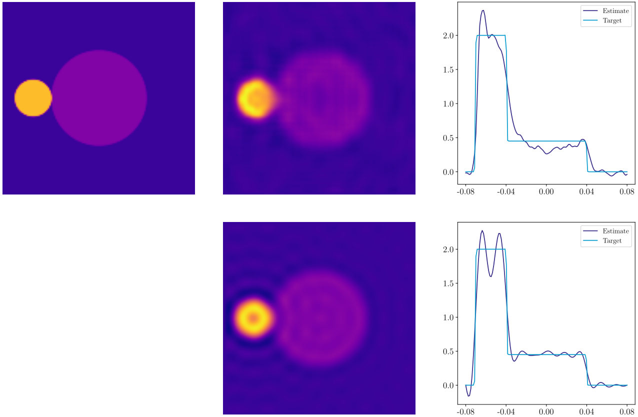
[11] Geffrin-Sabouroux-Eyraud, 2005

Experiments: TV-regularized reconstructions

Dataset: 2D inhomogeneous objects from Institut Fresnel dataset.[11]

Setup: far-field data at 4GHz; $\widehat{q}_0 = \mathbf{0}$; error tol $\varepsilon = 10^{-3}$.

Target | Reconstruction | Cross sections



⟵ **Our algorithm**

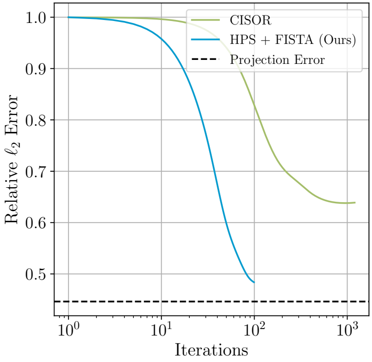
⟵ **Bandlimited projection**

[11] Geffrin-Sabouroux-Eyraud, 2005

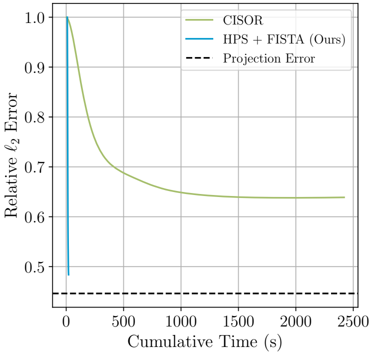
Experiments: TV-regularized reconstructions

Dataset: 2D inhomogeneous objects from Institut Fresnel dataset.[11]

Setup: far-field data at 4GHz; $\widehat{q}_0 = \mathbf{0}$; error tol $\varepsilon = 10^{-3}$.



(a) Error by iteration count



(b) Error by wall-clock time

[11] Geffrin-Sabouroux-Eyraud, 2005

Conclusion & future directions

- NN architectures inspired by recursive linearization afford key opportunities:
  - **High-resolution recovery** without test-time PDE solves ($<$0.1s “inference” time);
  - Can leverage **multiscale nature** of data;
  - **Stabilization of training** using homotopy through frequency.
- Ongoing & future work:
  - How to integrate forward model information with NN solutions?
  - How to leverage hardware acceleration in iterative frameworks? (on arXiv soon-ish).

Conclusion & future directions

- NN architectures inspired by recursive linearization afford key opportunities:
  - **High-resolution recovery** without test-time PDE solves ($<$0.1s “inference” time);
  - Can leverage **multiscale nature** of data;
  - **Stabilization of training** using homotopy through frequency.

- Ongoing & future work:
  - How to integrate forward model information with NN solutions?
  - How to leverage hardware acceleration in iterative frameworks? (on arXiv soon-ish).

Thank you! `arXiv:2405.13124`