

Physics-Informed Machine Learning For Audio Processing

Mirco Pezzoli¹, Diego Di Carlo², Shoichi Koyama³

¹Politecnico di Milano, Milan, Italy

²RIKEN, Kyoto, Japan

³National Institute of Informatics, Tokyo, Japan



POLITECNICO
MILANO 1863



NII S.Koyama's Lab
Audio Processing Research Group

Outline

- Introduction
- Part I – Overview of PIML for audio processing
- Part II – Current studies of PIML for audio processing
- Part III – Practical session of PINNs for audio processing
- Conclusion

Outline

- Introduction
 - **Part I – Overview of PIML for audio processing**
 - Part II – Current studies of PIML for audio processing
 - Part III – Practical session of PINNs for audio processing
 - Conclusion
-
- Introduce the problem of sound field estimation
 - Explain the main strategies for including physical priors
 - Provide examples of PIML for sound field estimation



Shoichi Koyama^{DP}, Juliano G. C. Ribeiro^{DP}, Tomohiko Nakamura^{IN},
Natsuki Ueno^{DP}, and Mirco Pezzoli^{IN}

Physics-Informed Machine Learning for Sound Field Estimation

Fundamentals, state of the art, and challenges

Outline

- Introduction
 - Part I – Overview of PIML for audio processing
 - **Part II – Current studies of PIML for audio processing**
 - Part III – Practical session of PINNs for audio processing
 - Conclusion
-
- Classify current literature on sound field estimation
 - Explain two example works of PIML
 - Provide details on PINNs for sound field estimation



Shoichi Koyama¹, Juliano G. C. Ribeiro², Tomohiko Nakamura¹,
Natsuki Ueno³, and Mirco Pezzoli¹

Physics-Informed Machine Learning for Sound Field Estimation

Fundamentals, state of the art, and challenges

Outline

- Introduction
 - Part I – Overview of PIML for audio processing
 - Part II – Current studies of PIML for audio processing
 - **Part III – Practical session of PINNs for audio processing**
 - Conclusion
-
- Hands-on session on PINNs
 - Example of PINNs for forward and inverse problem
 - Analysis of the limitations of PINNs

https://github.com/Chutlhu/EUSIPCO25_PIML_tutorial

Link to repo →



Outline

- Introduction
- Part I – Overview of PIML for audio processing
- Part II – Current studies of PIML for audio processing
- Part III – Practical session of PIML for audio processing
- Conclusion

Download the material from here!



Introduction

PHYSICS-INFORMED MACHINE LEARNING FOR AUDIO PROCESSING

The rise of DL

This causes the component to re-render, and the updated "count" value is displayed in the `p` element.

I hope this helps! Let me know if you have any other questions.

Can you add App.tsx that uses this component?

Sure! Here's an example of an `'App'` component that uses the `'Counter'` component from the previous example:

```
import React from 'react';
import Counter from './Counter';

function App() {
  return (
    <div>
      <Counter />
    </div>
  );
}

export default App;
```

In this example, the `'App'` component simply renders the `'Counter'` component. When the `'Counter'` component is rendered, it will display a button and a paragraph that shows the current count. When the button is clicked, the `'Counter'` component will update its state and re-render, showing an updated count in the paragraph.

Free Research Preview: ChatGPT is optimized for dialogue. Our goal is to make AI systems more natural to interact with, and your feedback will help us improve our systems and make them safer.

Explaining How End-to-End Deep Learning Steers a Self-Driving Car. <https://developer.nvidia.com/blog/explaining-deep-learning-self-driving-car/>



Brown, T., et al. Language models are few-shot learners. NEURIPS 2020



"A horse galloping through van Gogh's 'Starry Night'"

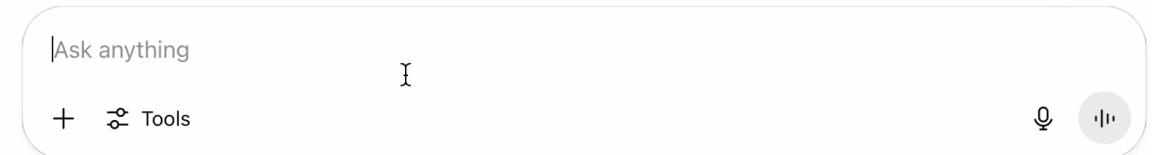
Blattmann, Andreas, et al. "Align your latents: High-resolution video synthesis with latent diffusion models." CVPR. 2023.

Machine learning for acoustics

➤ Why using Machine learning in our field?

- Access to a lot of data
- Identify novel patterns
- Deal with noise
- Automate workflows
- And many more!

What can I help with?



Deep learning challenges

2.1 Model and Architectures

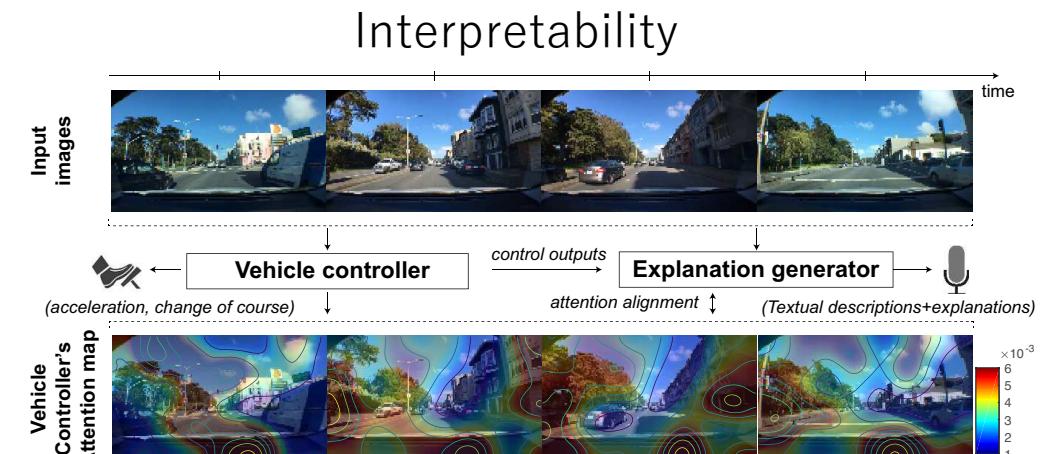
We use the same model and architecture as GPT-2 [RWC⁺19], including the modified initialization, pre-normalization, and reversible tokenization described therein, with the exception that we use alternating dense and locally banded sparse attention patterns in the layers of the transformer, similar to the Sparse Transformer [CGRS19]. To study the dependence of ML performance on model size, we train 8 different sizes of model, from 125 million parameters to 175 billion parameters, with the last being the model we call GPT-3. This range of model sizes allows us to test the scaling laws introduced in [KMH⁺20].

Brown, T., et al. Language models are few-shot learners. NEURIPS 2020

Table 2.2 shows the final mixture of datasets that we used in training. The CommonCrawl data was downloaded from 41 shards of monthly CommonCrawl covering 2016 to 2019, constituting 45TB of compressed plaintext before filtering and 570GB after filtering, roughly equivalent to 400 billion byte-pair-encoded tokens. Note that during training, datasets are not sampled in proportion to their size, but rather datasets we view as higher-quality are sampled more frequently, such that CommonCrawl and Books2 datasets are sampled less than once during training, but the other datasets are sampled 2-3 times. This essentially accepts a small amount of overfitting in exchange for higher quality training data.

Computational cost & Data

Kim, J., Rohrbach, A., Darrell, T., Canny, J., & Akata, Z. (2018). Textual explanations for self-driving vehicles. In Proceedings of the European conference on computer vision (ECCV)



Example of textual descriptions + explanations:

Ours: "The car is driving forward + because there are no other cars in its lane"
Human annotator: "The car heads down the street + because the street is clear."

Deep learning challenges

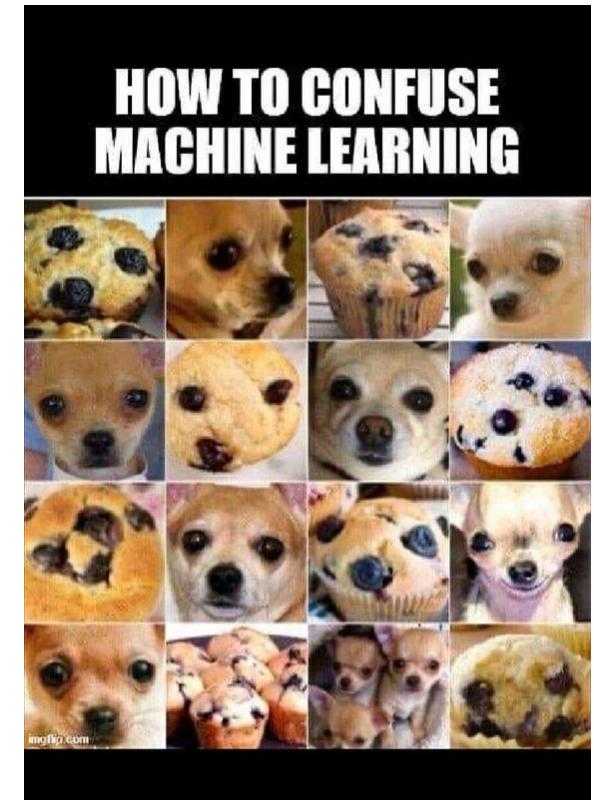
Generalization



Siamese Cat 1.0

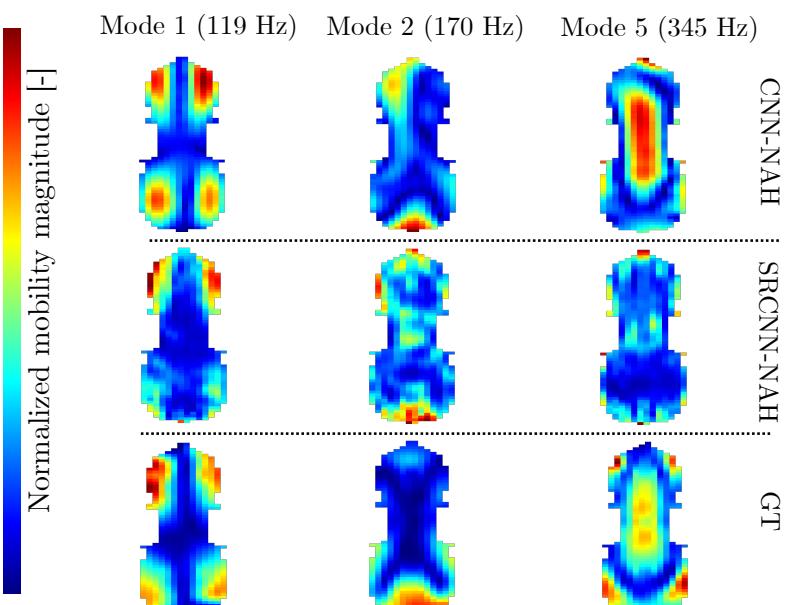
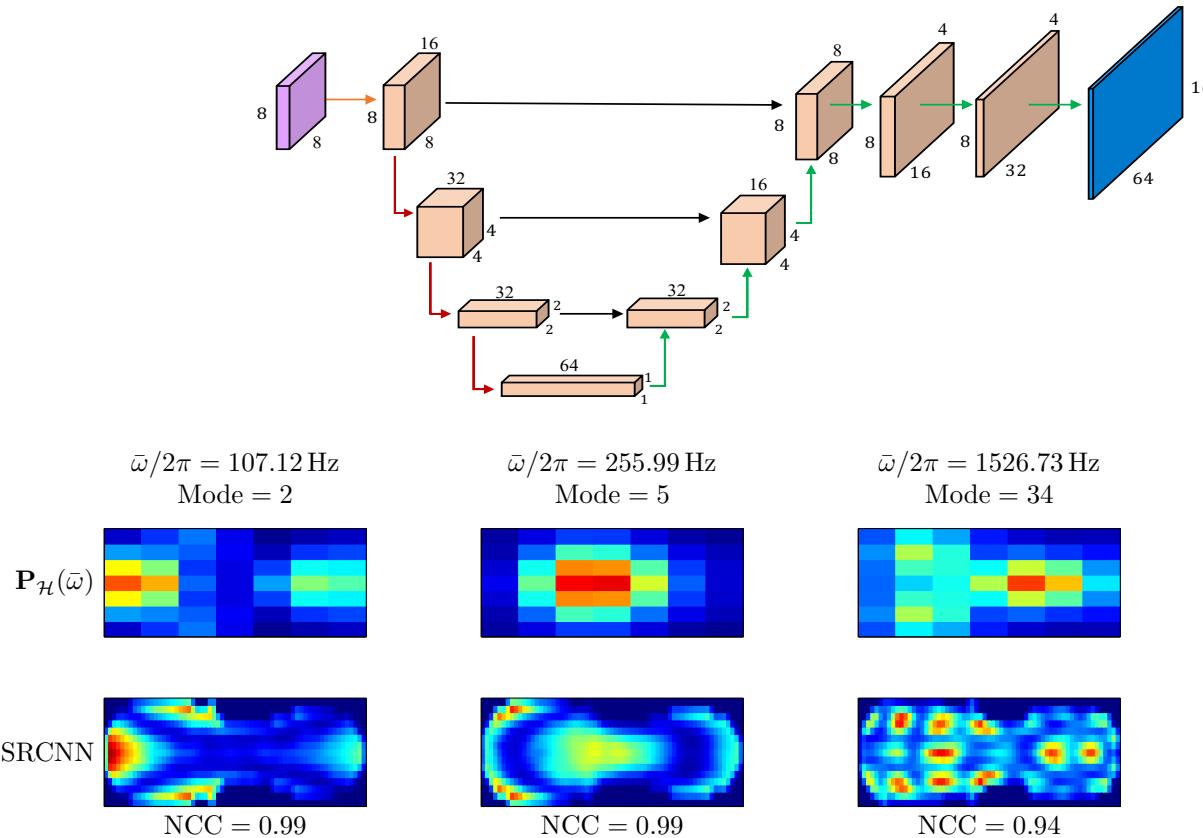


Siamese Cat 0.1



Deep learning challenges

Generalization



Lampis, Alessio. "Experimental validation for data driven Near-Field Acoustic Holography." Thesis in M. Sc. Music and Acoustic Engineering, Politecnico di Milano (2020).

Olivieri, Marco, et al. "Near field acoustic holography on arbitrary shapes using convolutional neural network." 29th European Signal Processing Conference (EUSIPCO). IEEE, 2021.

Scientific machine learning

Naively using **machine learning** for scientific tasks (acoustics) usually leads to:

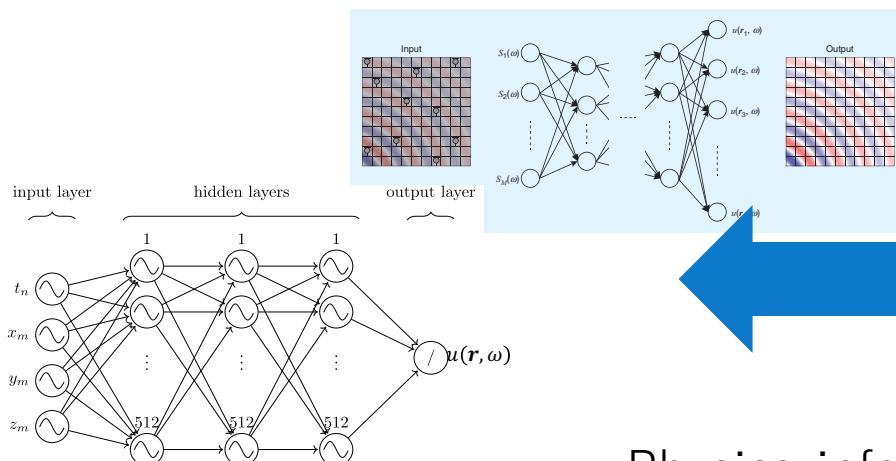
- Lack of **interpretability**
- **Generalization** issues
- Huge amount of required **data**

Do neural networks really “understand” the task?

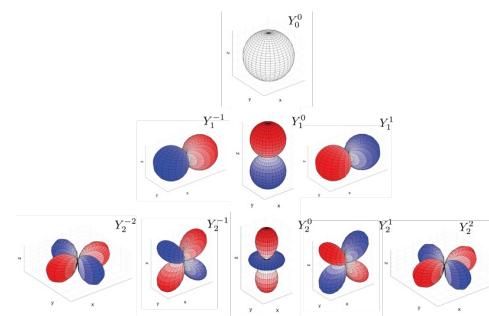
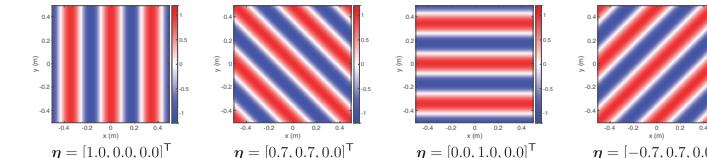
Physics-informed machine learning

Solution: bridge **machine learning** with **scientific knowledge**

Machine learning 



Domain knowledge 



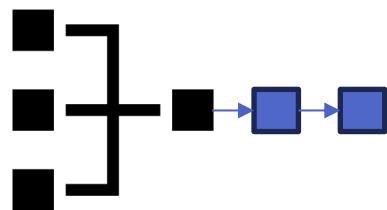
Physics-informed machine learning

More powerful, robust and interpretable

Ways to add physics principles to ML

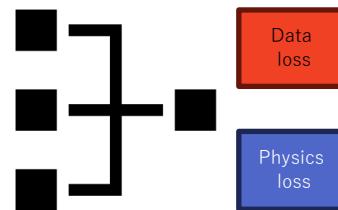
Constrained

- Forced to adhere to **physical model**
- Impose known solutions of governing equation



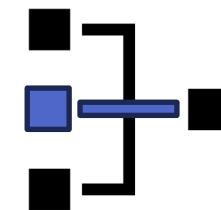
Penalized (informed)

- Penalization through **loss function**
- Add governing equation but small deviation are allowed



Architecture

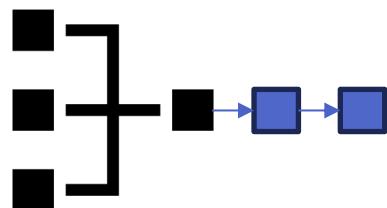
- Impose **physics properties** to the model
- Encoding general properties e.g., symmetry



Ways to add physics principles to ML

Constrained

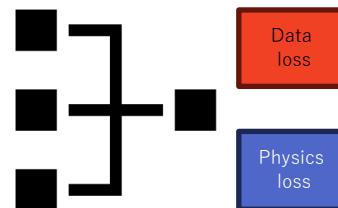
- Forced to adhere to **physical model**
- Impose known solutions of governing equation



We will cover these two approaches!

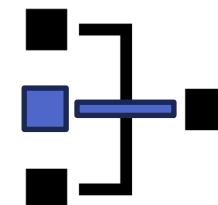
Penalized (informed)

- Penalization through **loss function**
- Add governing equation but small deviation are allowed



Architecture

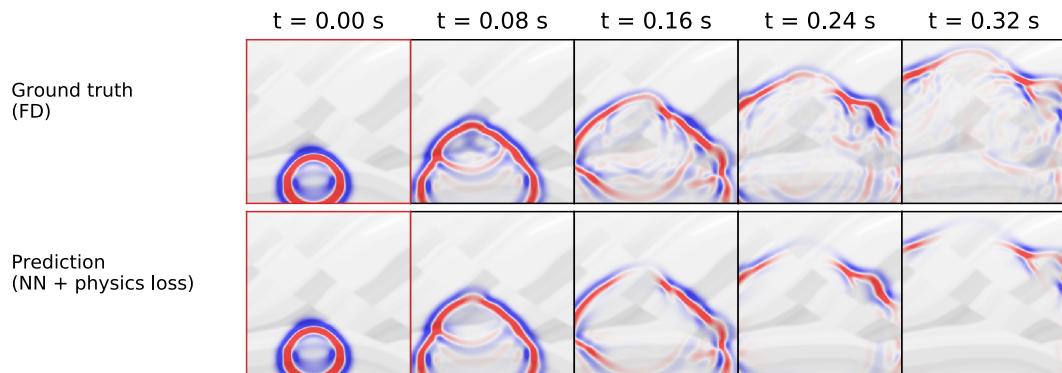
- Impose **physics properties** to the model
- Encoding general properties e.g., symmetry



Key tasks for PIML in Acoustics

➤ Forward problem

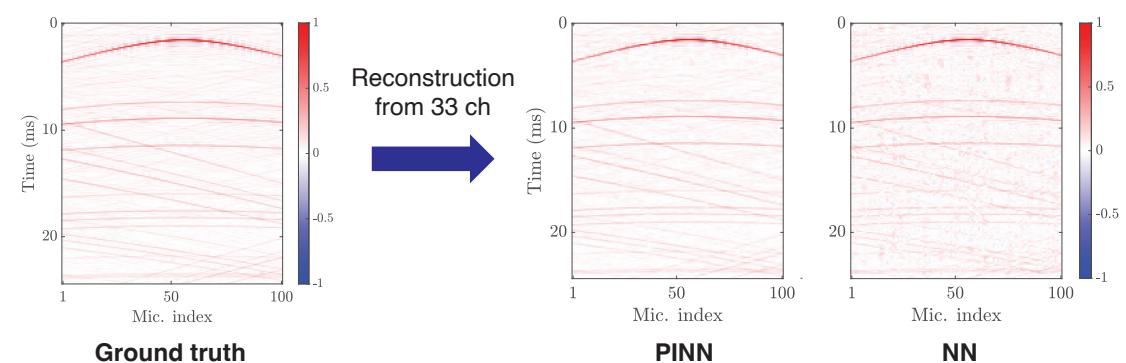
- **Simulation**
- Directly solve **Partial Differential Equations (PDE)**
- Essential for understanding behavior of systems
- Alternative to numerical methods



Moseley, B., Markham, A., & Nissen-Meyer, T. (2020). Solving the wave equation with physics-informed deep learning. *arXiv preprint arXiv:2006.11894*.

➤ Inverse problem

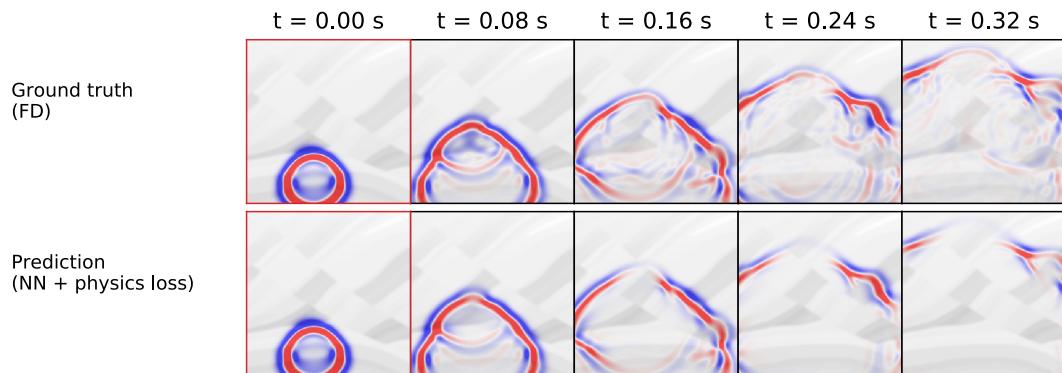
- **Estimation**
- Infer **parameters** or data
- Essential in several applications
- "Alternative" to **optimization methods**



Key tasks for PIML in Acoustics

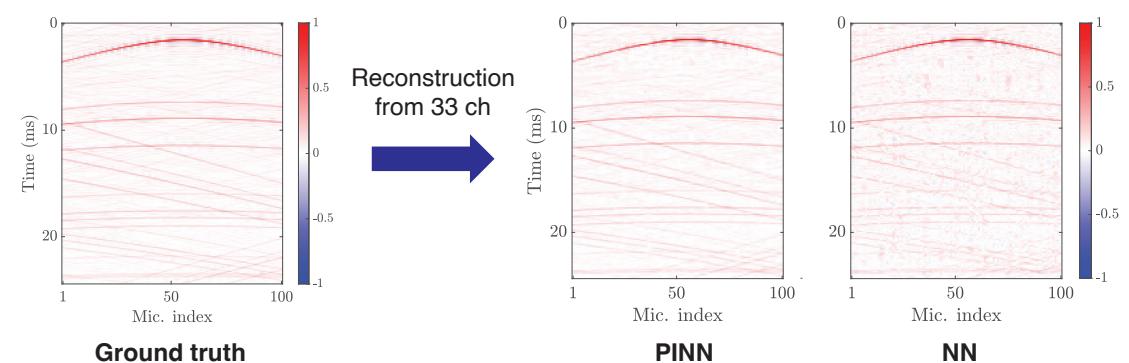
➤ Forward problem

- **Simulation**
- Directly solve **Partial Differential Equations (PDE)**
- Essential for understanding behavior of systems
- Alternative to numerical methods



➤ Inverse problem

- **Estimation**
- Infer **parameters** or data
- Essential in several applications
- "Alternative" to **optimization methods**



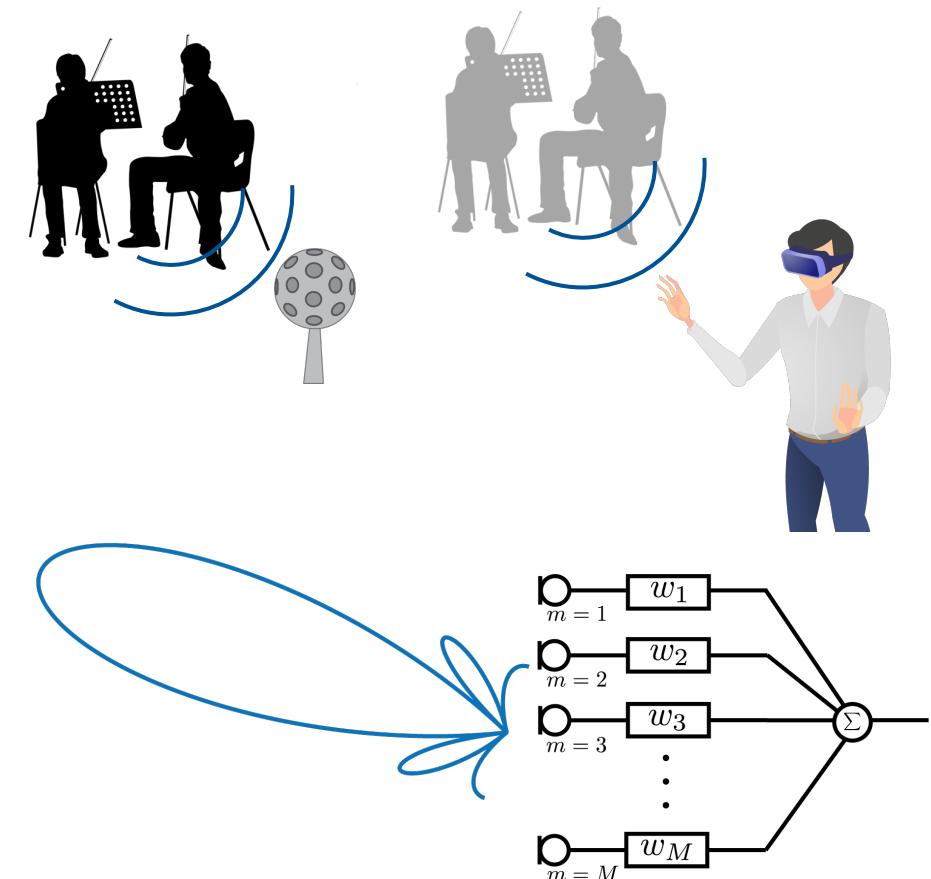
Part I – Overview of PIML for audio processing

PHYSICS-INFORMED MACHINE LEARNING FOR AUDIO PROCESSING

Audio Processing Based on Sound Field Estimation

Many spatial audio processing techniques are based on estimation of sound field using multiple mics

- **Spatial audio capturing and reproduction**
 - Sound field is captured by multiple mics and reproduced by headphones or loudspeakers
 - Head-related transfer function personalization
- **Room acoustic analysis and control**
 - Visualization and auralization of spatial sound
 - Estimation of room acoustic impulse responses/transfer functions
- **Source enhancement and separation**
 - Beamforming techniques require accurate steering vectors (array manifold vectors)
 - Source enhancement using wearable devices is more challenging



About part 1&2

SPECIAL ISSUE ON MODEL-BASED AND
DATA-DRIVEN AUDIO SIGNAL PROCESSING

Shoichi Koyama[✉], Juliano G. C. Ribeiro[✉], Tomohiko Nakamura[✉],
Natsuki Ueno[✉], and Mirco Pezzoli[✉]

Physics-Informed Machine Learning for Sound Field Estimation

Fundamentals, state of the art, and challenges

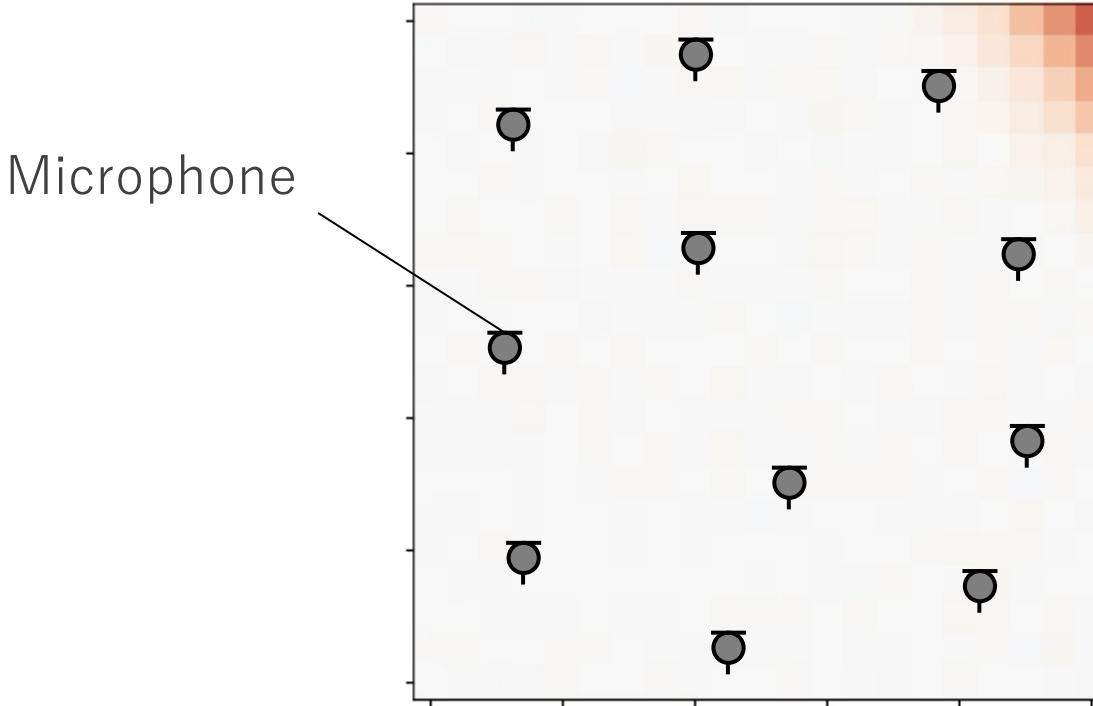
Paper



Based on our article published in IEEE Signal Processing Magazine

What is Sound Field Estimation?

Estimating sound field inside target region using multiple mics



Fundamental technology in various audio processing tasks
and has variety of applications

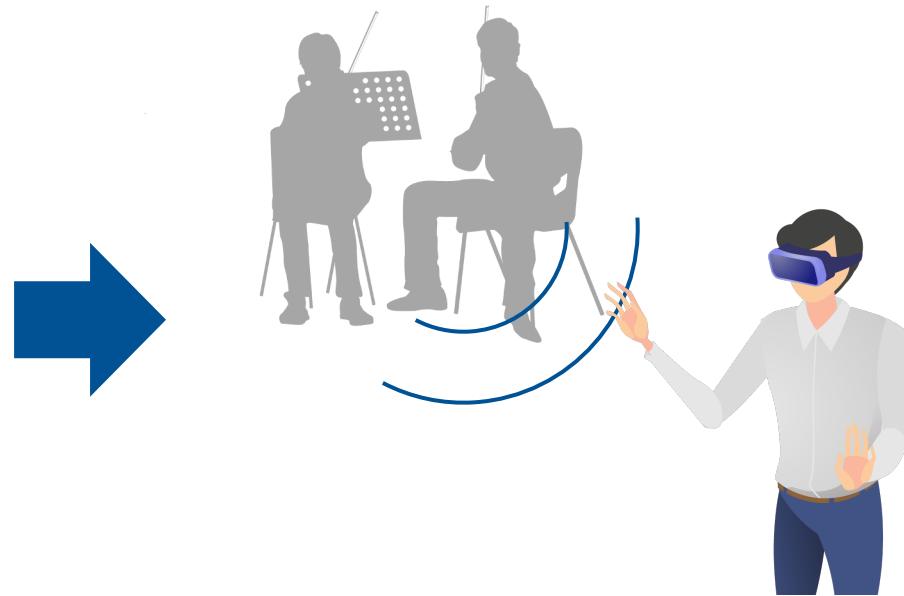
Application #1: Binaural Reproduction

Binaural reproduction from mic array recordings for VR audio

Recording



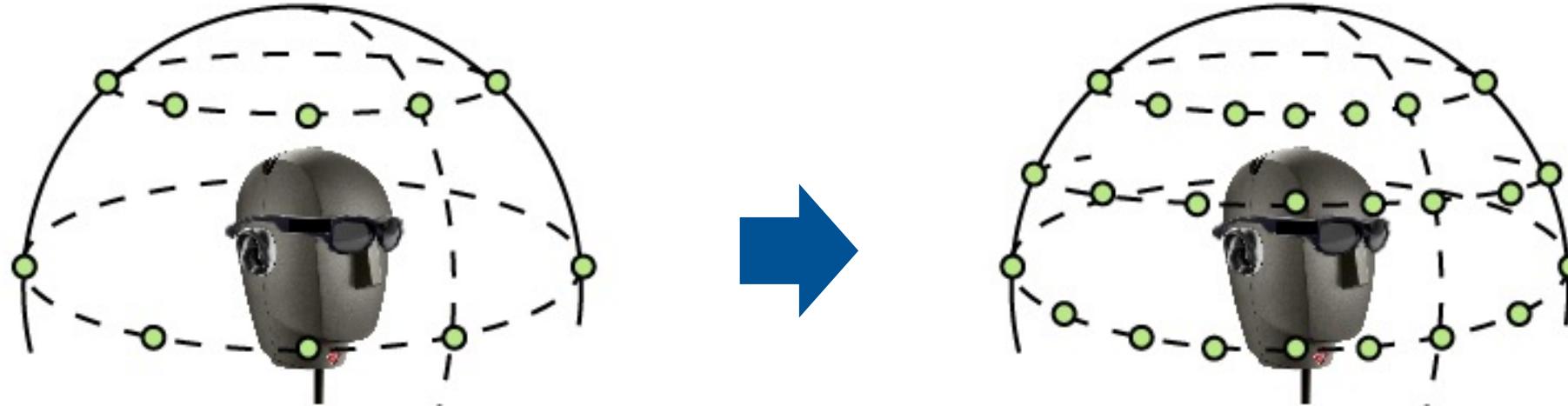
Reproduction



- Unlike binaural synthesis in VR space, binaural reproduction in real environments requires spatial audio capturing by using multiple mics
- Required to estimate spatial sound in a wide area to achieve a wide listening area, e.g., 6DoF reproduction

Application #2: Steering Vector Upsampling

Interpolating measured steering vectors



- Estimation of steering vectors for wearable devices with multiple mics is crucial for source enhancement compared to simple-shaped arrays
- Upsampling techniques for steering vectors will simplify the measurement of steering vectors

Application #3: Spatial Active Noise Control

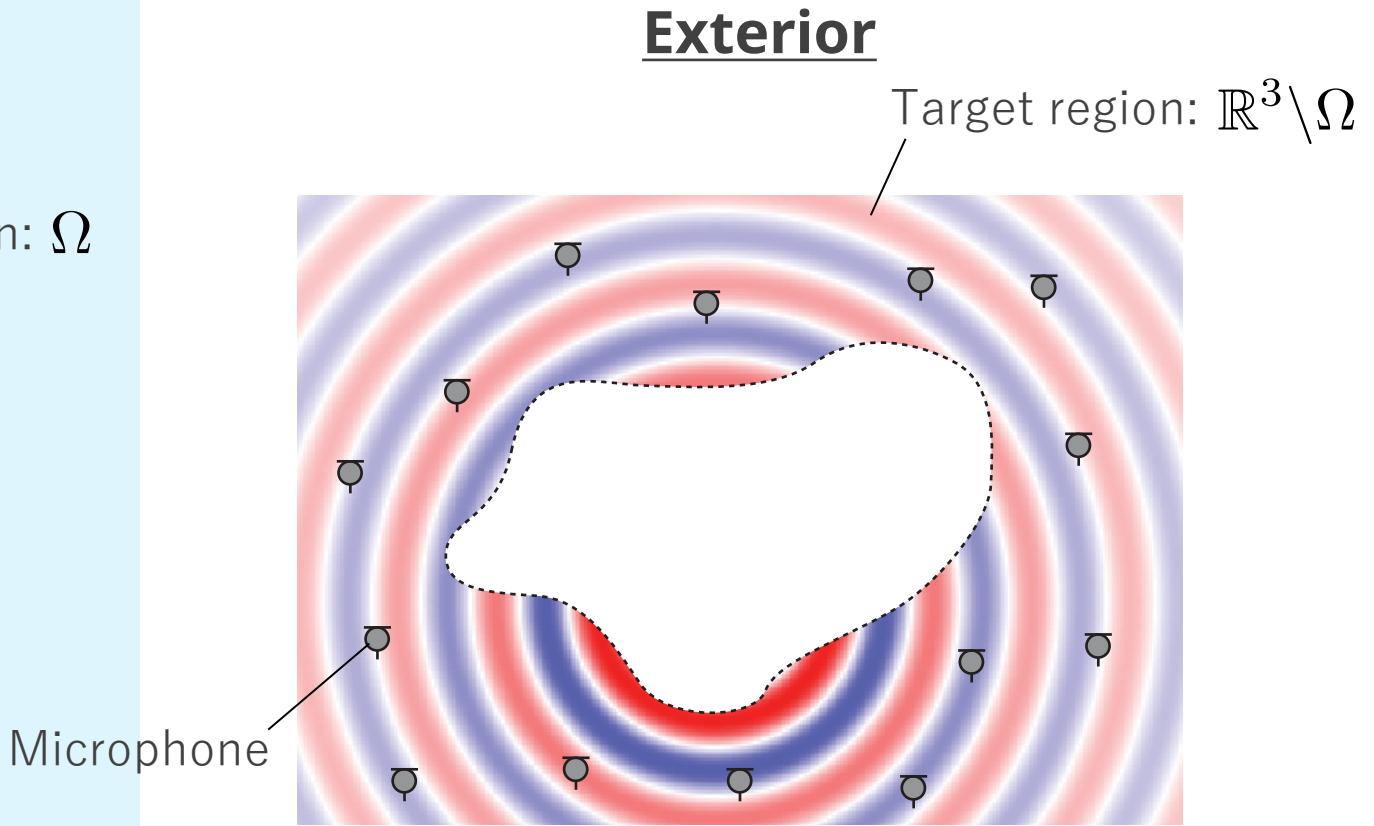
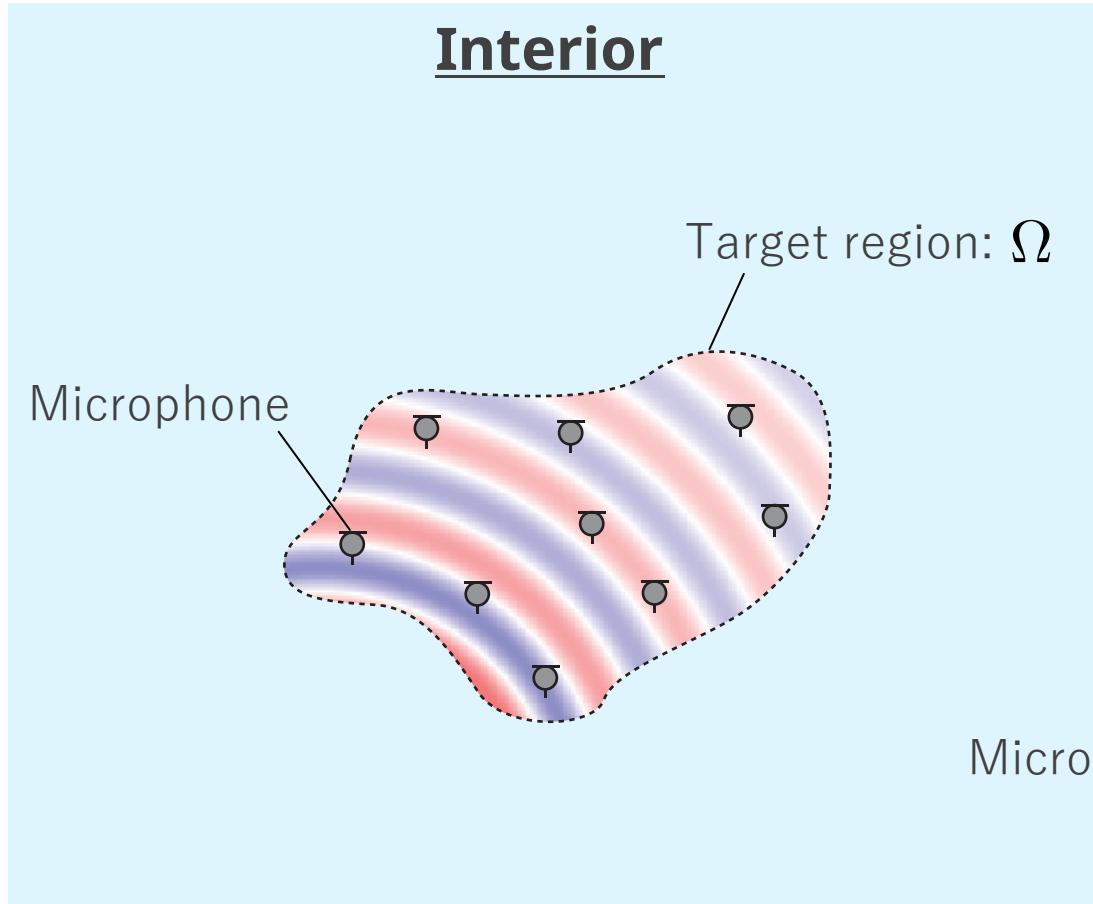
Noise suppression over 3D space by loudspeaker signals



- Active noise control (ANC) aims to cancel noise by using loudspeaker signals, but its effect is limited to local region
- Spatial ANC by estimating spatial sound using multiple mics and synthesizing anti-spatial sound using multiple loudspeakers

Sound field estimation

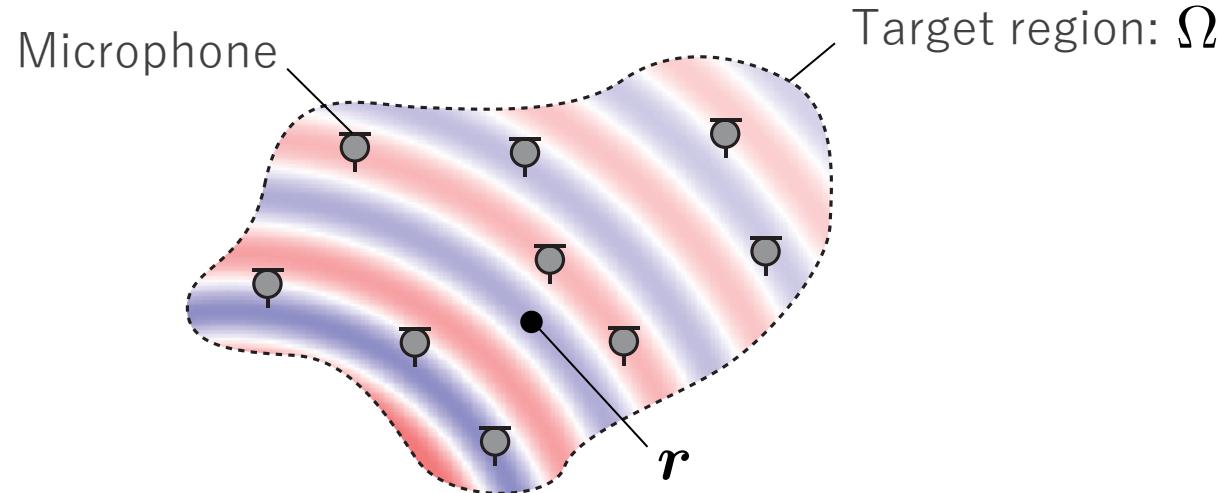
Interior and exterior sound field estimation



Focusing mainly on estimation in interior free field

Sound field estimation

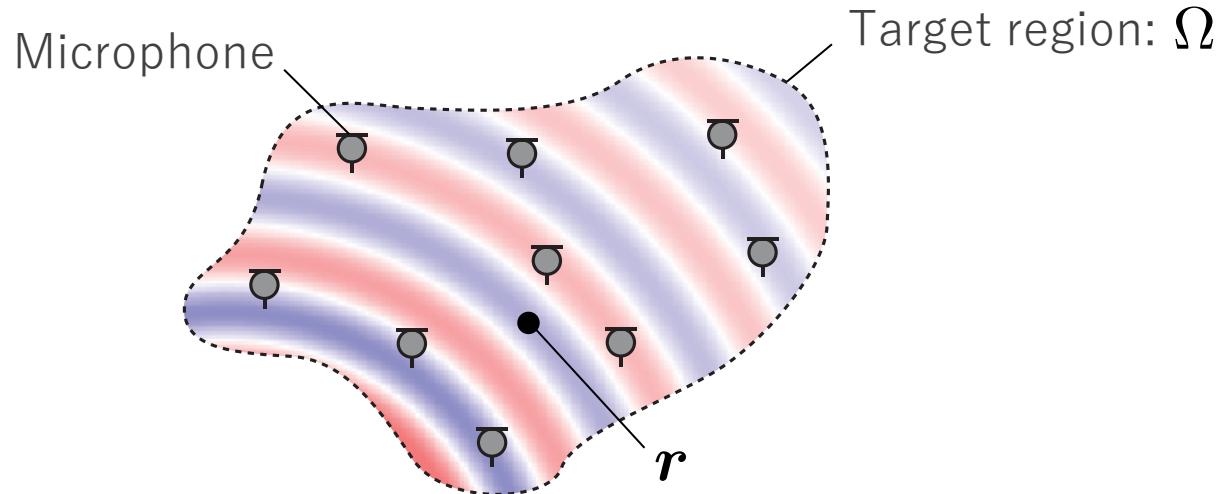
Formulation of sound field estimation problem



Estimate pressure distribution $U(\mathbf{r}, t)$ ($\mathbf{r} \in \Omega$) in the time domain or $u(\mathbf{r}, \omega)$ in frequency domain with M omnidirectional mics at $\{\mathbf{r}_m\}_{m=1}^M$

Sound field estimation

Formulation of sound field estimation problem



- Problem to be solved in general interpolation techniques
 - f (U or u) is represented by model parameters θ

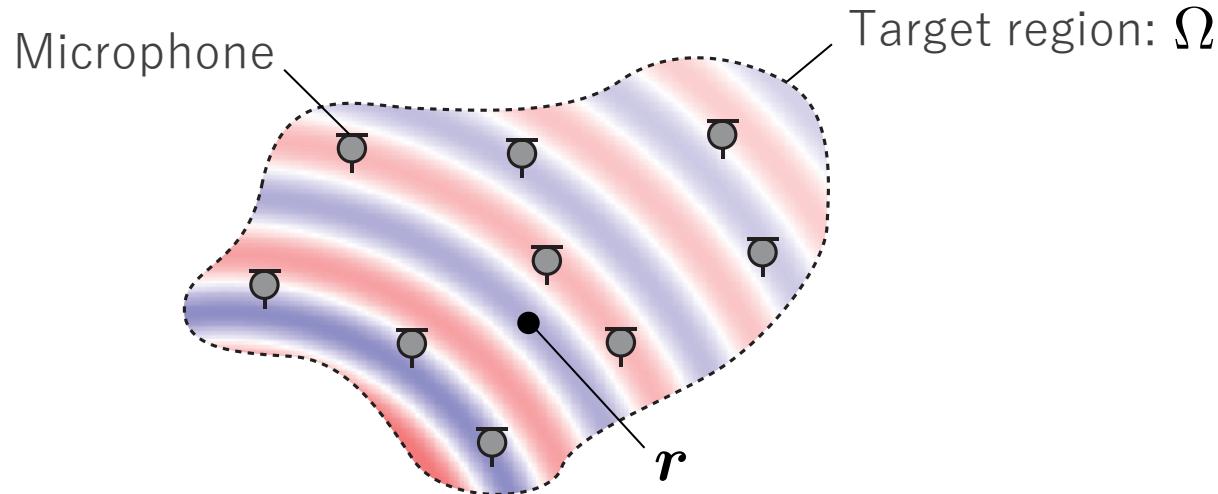
$$\underset{\theta}{\text{minimize}} \mathcal{L}(\mathbf{y}, f(\{\mathbf{x}_i\}_{i=1}^I; \theta)) + \mathcal{R}(\theta)$$

Observation Regularization term

Loss term Samples in space/time/freq

Sound field estimation

Formulation of sound field estimation problem



- Problem to be solved in general interpolation techniques
 - f (U or u) is represented by model parameters θ

$$\underset{\theta}{\text{minimize}} \left\| \mathbf{y} - f(\{\mathbf{x}_i\}_{i=1}^I; \theta) \right\|^2 + \lambda \|\theta\|^2$$

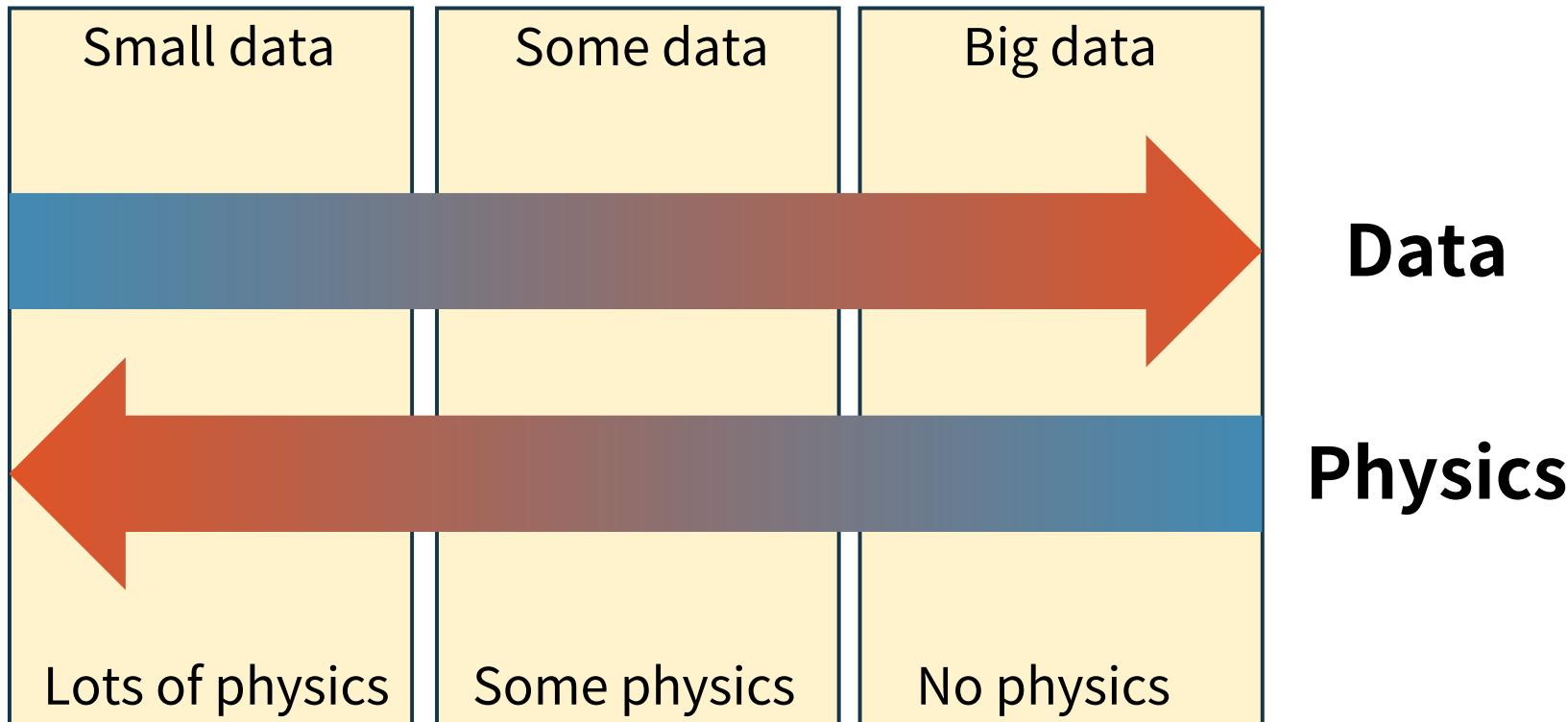
Squared ℓ_2 -norm penalty

Squared error loss

EMBEDDING PHYSICAL PROPERTIES IN INTERPOLATION TECHNIQUES

Embedding physical properties in interpolation techniques

Purely data-driven approaches may suffer from overfitting



[Karniadakis+ 2021]

Physical properties will be useful prior information
in sound field estimation

Embedding physical properties in interpolation techniques

What kind of physical properties can be embedded?

- Function to be estimated should satisfy governing PDE
 - **Wave equation** in time domain

$$\left(\nabla_{\boldsymbol{r}}^2 - \frac{1}{c^2} \frac{\partial^2}{\partial t^2} \right) U(\boldsymbol{r}, t) = 0$$

- **Helmholtz equation** in freq domain

$$(\nabla_{\boldsymbol{r}}^2 + k^2) u(\boldsymbol{r}, \omega) = 0$$

➡ Techniques incorporating constraints on the governing PDEs are introduced

Basis expansion into element solutions

Linear combination of finite number of basis functions

- Function f is modeled by basis functions $\{\varphi_l(\mathbf{x})\}_{l=1}^L$ and their weights $\{\gamma_l\}_{l=1}^L$

$$f(\mathbf{x}; \boldsymbol{\gamma}) = \sum_{l=1}^L \gamma_l \varphi_l(\mathbf{x})$$

$$= \boldsymbol{\varphi}(\mathbf{x})^\top \boldsymbol{\gamma}$$

$$\boldsymbol{\varphi} = [\varphi_1, \dots, \varphi_L]^\top$$

$$\boldsymbol{\gamma} = [\gamma_1, \dots, \gamma_L]^\top$$

- **Basis functions as element solutions of wave/Helmholtz eq** [Williams+ 1999, Colton&Kress 2019]

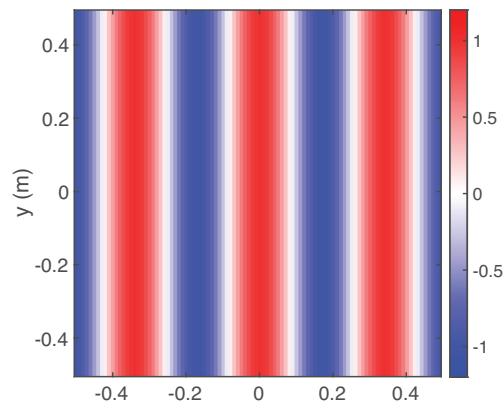
- Plane wave expansion (Herglotz wave function)
- Spherical wave function expansion
- Equivalent source distribution (single-layer potential)

Basis expansion into element solutions

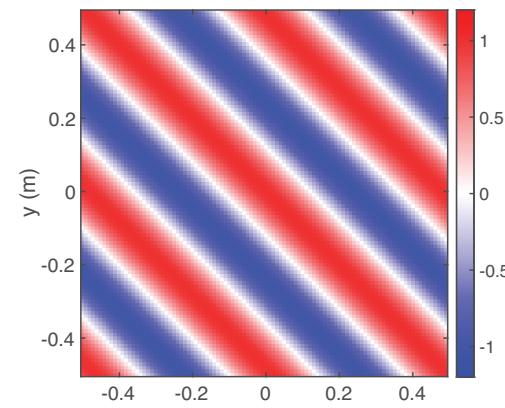
➤ Plane wave expansion (or Herglotz wave function)

Plane wave arrival direction

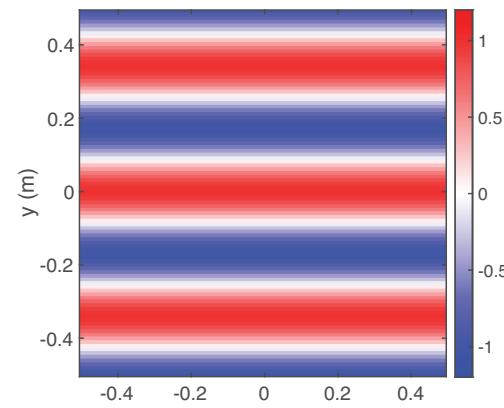
$$u(\mathbf{r}, \omega) = \int_{\mathbb{S}_2} \tilde{u}(\boldsymbol{\eta}, \omega) e^{jk\langle \boldsymbol{\eta}, \mathbf{r} \rangle} d\boldsymbol{\eta}$$



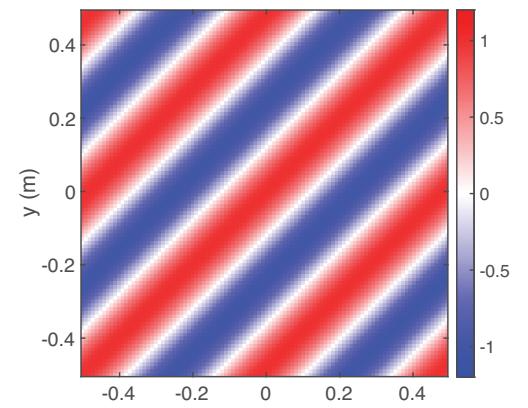
$$\boldsymbol{\eta} = [1.0, 0.0, 0.0]^T$$



$$\boldsymbol{\eta} = [0.7, 0.7, 0.0]^T$$



$$\boldsymbol{\eta} = [0.0, 1.0, 0.0]^T$$



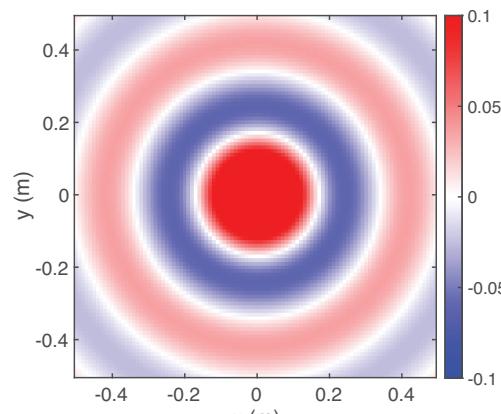
$$\boldsymbol{\eta} = [-0.7, 0.7, 0.0]^T$$

Basis expansion into element solutions

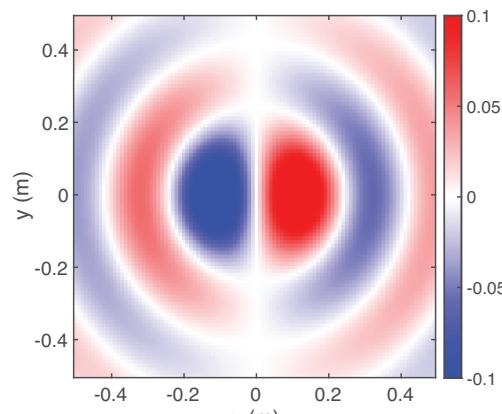
➤ Spherical wave function expansion

$$u(\mathbf{r}, \omega) = \sum_{\nu=0}^{\infty} \sum_{\mu=-\infty}^{\infty} \ddot{u}_{\nu,\mu}(\mathbf{r}_o, \omega) j_{\nu}(k \|\mathbf{r} - \mathbf{r}_o\|) Y_{\nu,\mu}\left(\frac{\mathbf{r} - \mathbf{r}_o}{\|\mathbf{r} - \mathbf{r}_o\|}\right)$$

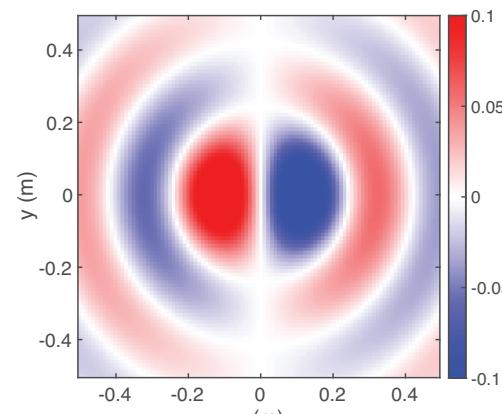
Expansion center Spherical harmonic function
 Spherical Bessel function



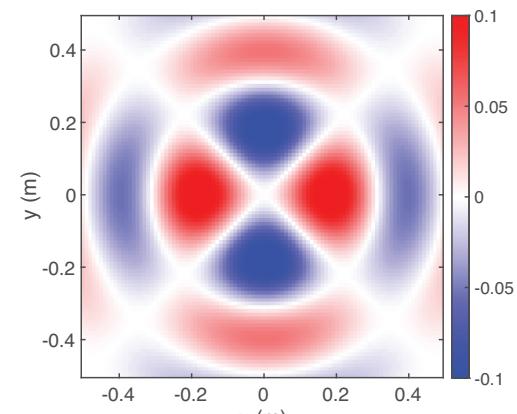
$(\nu, \mu) = (0, 0)$



$(\nu, \mu) = (1, 1)$



$(\nu, \mu) = (1, -1)$



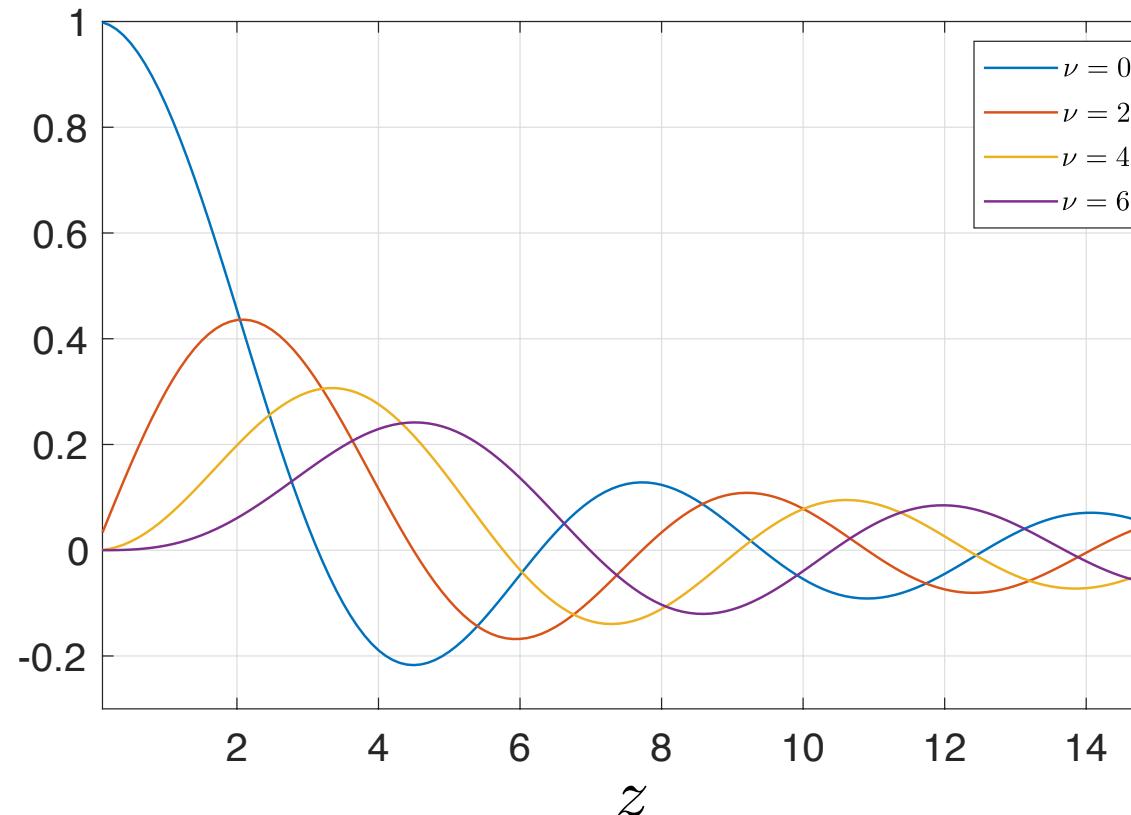
$(\nu, \mu) = (2, 2)$

Basis expansion into element solutions

- Spherical Bessel function

$$j_\nu(z) = \sqrt{\frac{\pi}{2z}} J_{\nu+1/2}(z)$$

Bessel function

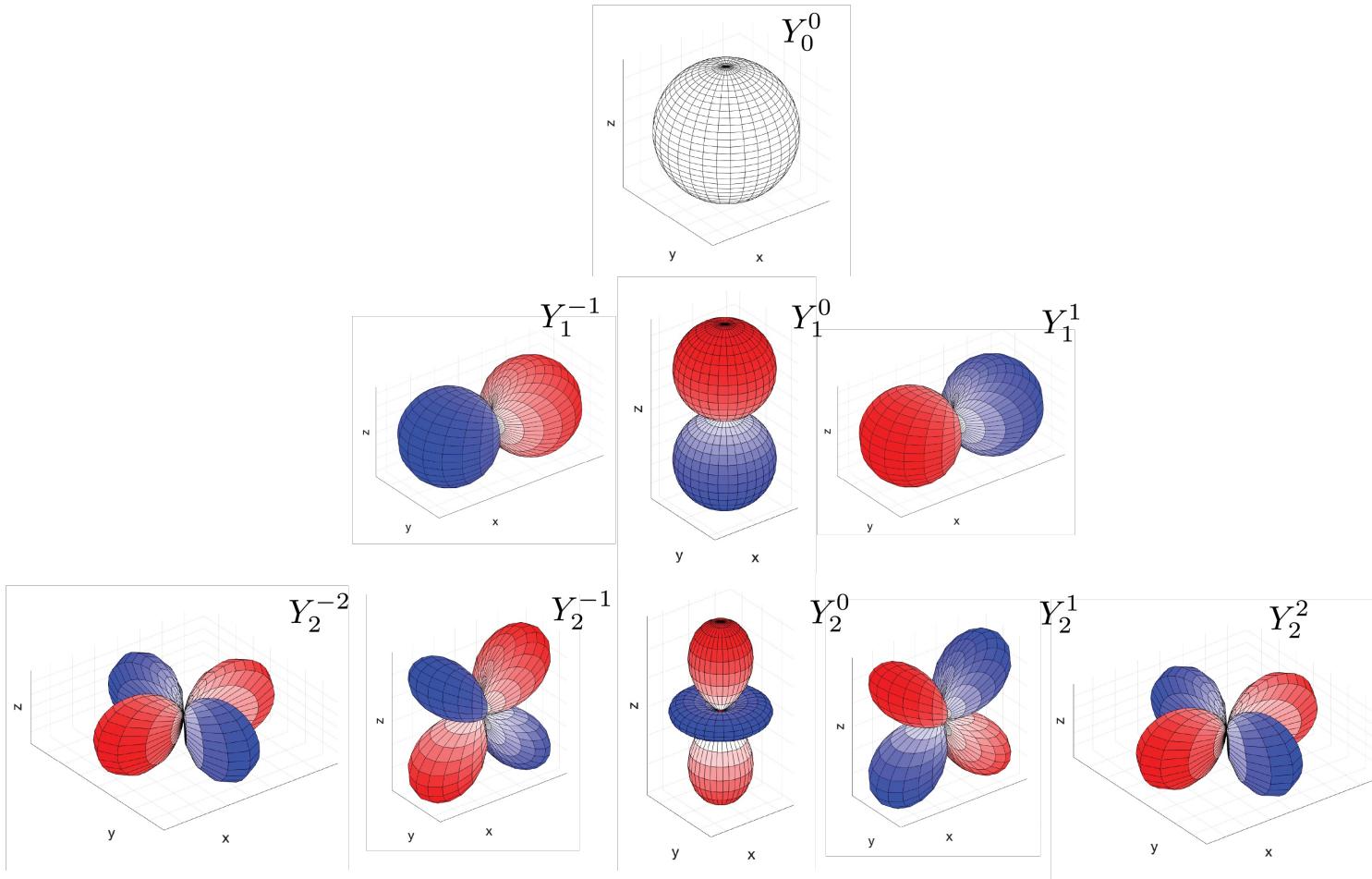


Basis expansion into element solutions

► Spherical harmonic function

Associated Legendre function

$$Y_{\nu,\mu}(\theta, \phi) = \sqrt{\frac{(2\nu + 1)}{4\pi} \frac{(\nu - \mu)!}{(\nu + \mu)!}} P_{\nu}^{\mu}(\cos \theta) e^{j\mu\phi}$$

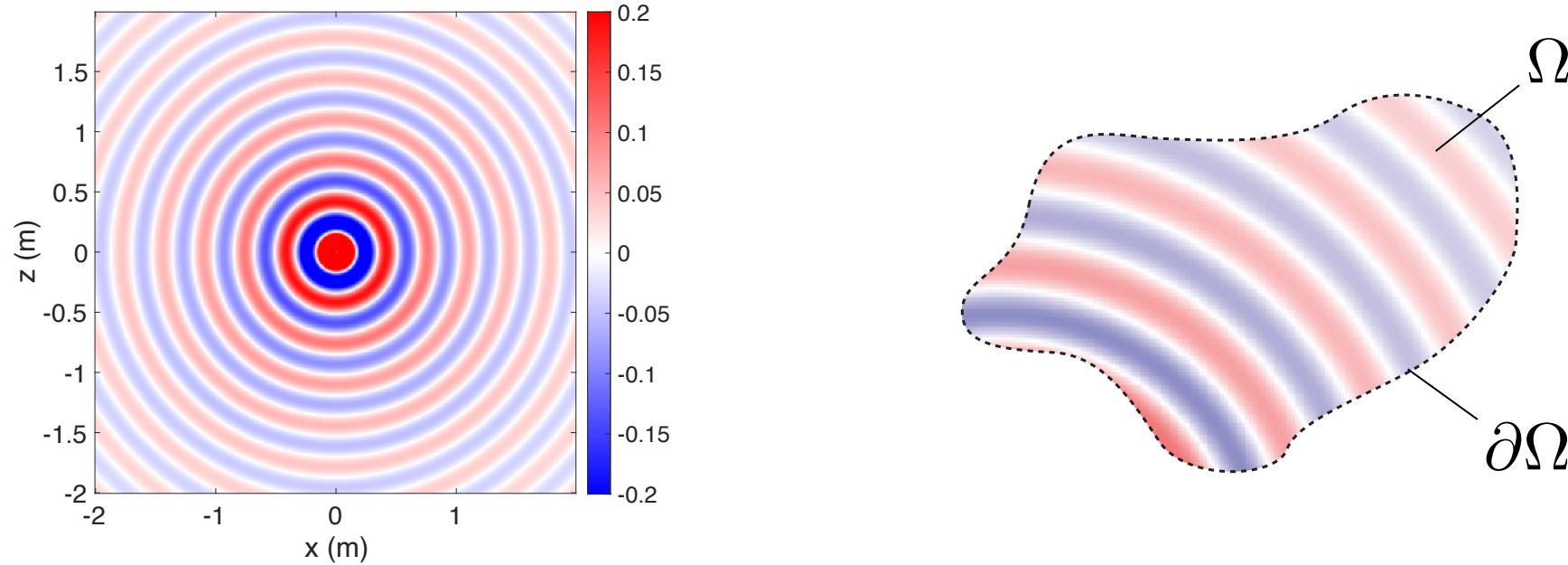


Basis expansion into element solutions

- Equivalent source distribution (or single layer potential)

$$u(\mathbf{r}, \omega) = \int_{\partial\Omega} \check{u}(\mathbf{r}', \omega) \frac{e^{-jk(\mathbf{r}-\mathbf{r}')}}{4\pi \|\mathbf{r} - \mathbf{r}'\|} d\mathbf{r}'$$

Point source



Basis expansion into element solutions

- Linear regression with finite-dimensional basis expansion
 - Regularized least squares solution of expansion coeffs

$$\begin{aligned}\hat{\gamma} &= \arg \min_{\gamma} \|\mathbf{y} - \Phi \gamma\|^2 + \lambda \|\gamma\|^2 \\ &= (\Phi^H \Phi + \lambda I)^{-1} \Phi^H \mathbf{y}\end{aligned}$$

$\Phi = [\varphi(x_1), \dots, \varphi(x_I)]^T$

- Estimate the function f

$$\hat{f}(x; \hat{\gamma}) = \varphi(x)^T \hat{\gamma}$$

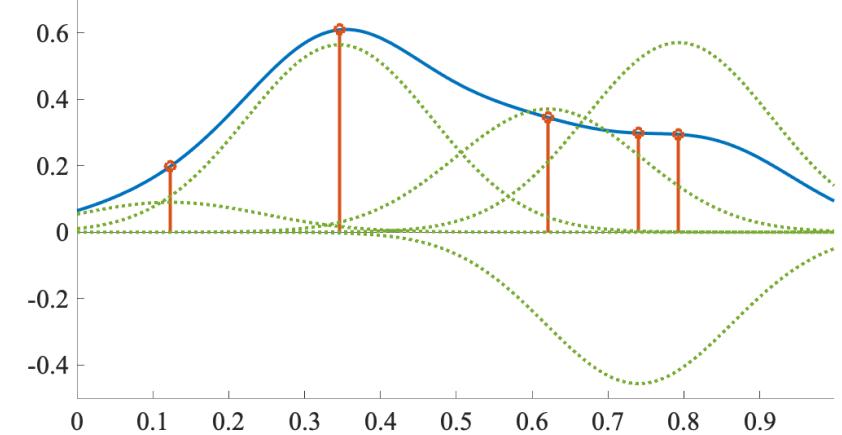
→ Number of basis functions (and expansion center for spherical wave function expansion) have to be properly set

Kernel regression with constraint of governing PDE

- f is represented by weighted sum of kernel function κ

$$\begin{aligned} f(\mathbf{x}; \boldsymbol{\alpha}) &= \sum_{i=1}^I \alpha_i \kappa(\mathbf{x}, \mathbf{x}_i) \\ &= \boldsymbol{\kappa}(\mathbf{x})^\top \boldsymbol{\alpha} \end{aligned}$$

$\boldsymbol{\kappa}(\mathbf{x}) = [\kappa(\mathbf{x}, \mathbf{x}_1), \dots, \kappa(\mathbf{x}, \mathbf{x}_I)]^\top$ $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_I]^\top$



- Kernel function κ is a similarity function expressed as inner product on some functional space \mathcal{H}

$$\kappa(\mathbf{x}, \mathbf{x}') = \langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle_{\mathcal{H}}$$

➡ $\varphi(\mathbf{x})$ can be infinite-dimensional or κ can be directly designed

Kernel regression with constraint of governing PDE

- In kernel ridge regression, $\hat{\alpha}$ is obtained as

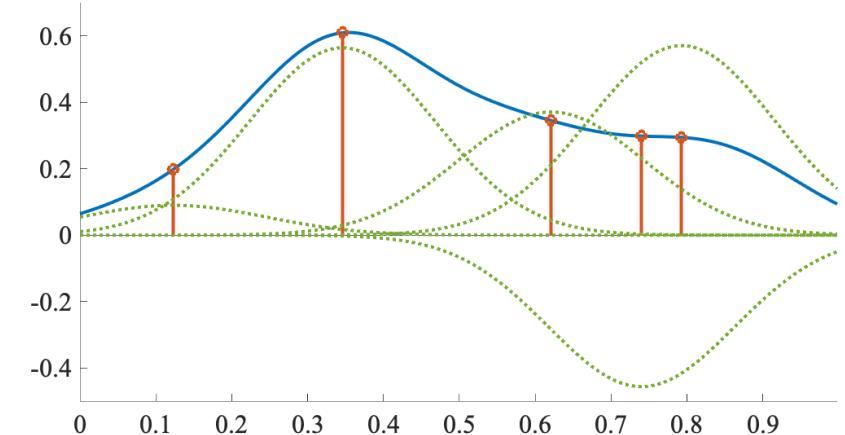
$$\hat{\alpha} = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{y}$$

with Gram matrix defined as

$$\mathbf{K} = \begin{bmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) & \dots & \kappa(\mathbf{x}_1, \mathbf{x}_I) \\ \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}_I, \mathbf{x}_1) & \dots & \kappa(\mathbf{x}_I, \mathbf{x}_I) \end{bmatrix}$$

- Estimate the function

$$\hat{f}(\mathbf{x}; \hat{\alpha}) = \kappa(\mathbf{x})^\top \hat{\alpha}$$



➡ Function space \mathcal{H} , which also defines κ , must be properly defined

Kernel regression with constraint of governing PDE

Kernel function to constrain the solution to satisfy Helmholtz eq

- Inner product and norm over \mathcal{H} are defined by plane wave expansion with positive directional weighting w [Ueno+ 2021]

$$\langle u_1, u_2 \rangle_{\mathcal{H}} = 4\pi \int_{\mathbb{S}_2} \frac{1}{w(\boldsymbol{\eta})} \tilde{u}_1(\boldsymbol{\eta})^* \tilde{u}_2(\boldsymbol{\eta}) d\boldsymbol{\eta}$$

$$\|u\|_{\mathcal{H}} = \sqrt{\langle u, u \rangle_{\mathcal{H}}}$$

Directional weighting w is designed to incorporate prior knowledge of sound field directivity

Kernel regression with constraint of governing PDE

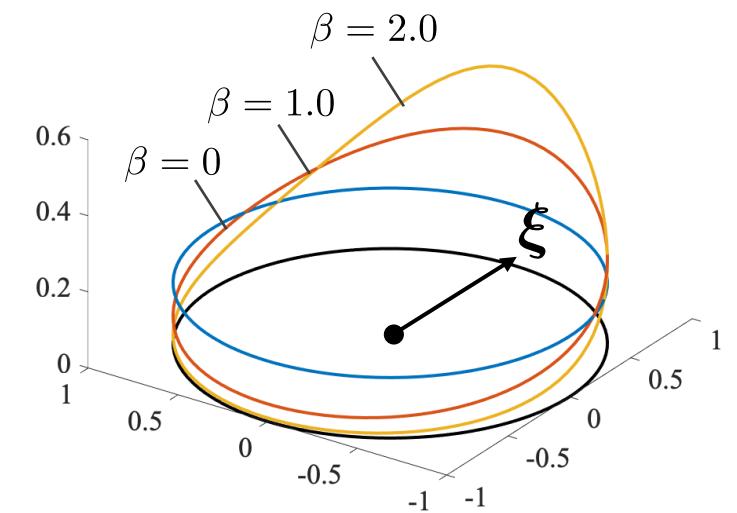
Kernel function to constrain the solution to satisfy Helmholtz eq

- Kernel function when w is defined by using von Mises–Fisher distribution

$$w(\boldsymbol{\eta}) = \frac{1}{C(\beta)} e^{\beta \langle \boldsymbol{\eta}, \boldsymbol{\xi} \rangle}$$

$$\kappa(\mathbf{r}_1, \mathbf{r}_2) = \frac{1}{C(\beta)} j_0 \left(\sqrt{(k\mathbf{r}_{12} - j\beta\boldsymbol{\xi})^\top (k\mathbf{r}_{12} - j\beta\boldsymbol{\xi})} \right)$$

$$\text{with } \mathbf{r}_{12} = \mathbf{r}_2 - \mathbf{r}_1$$

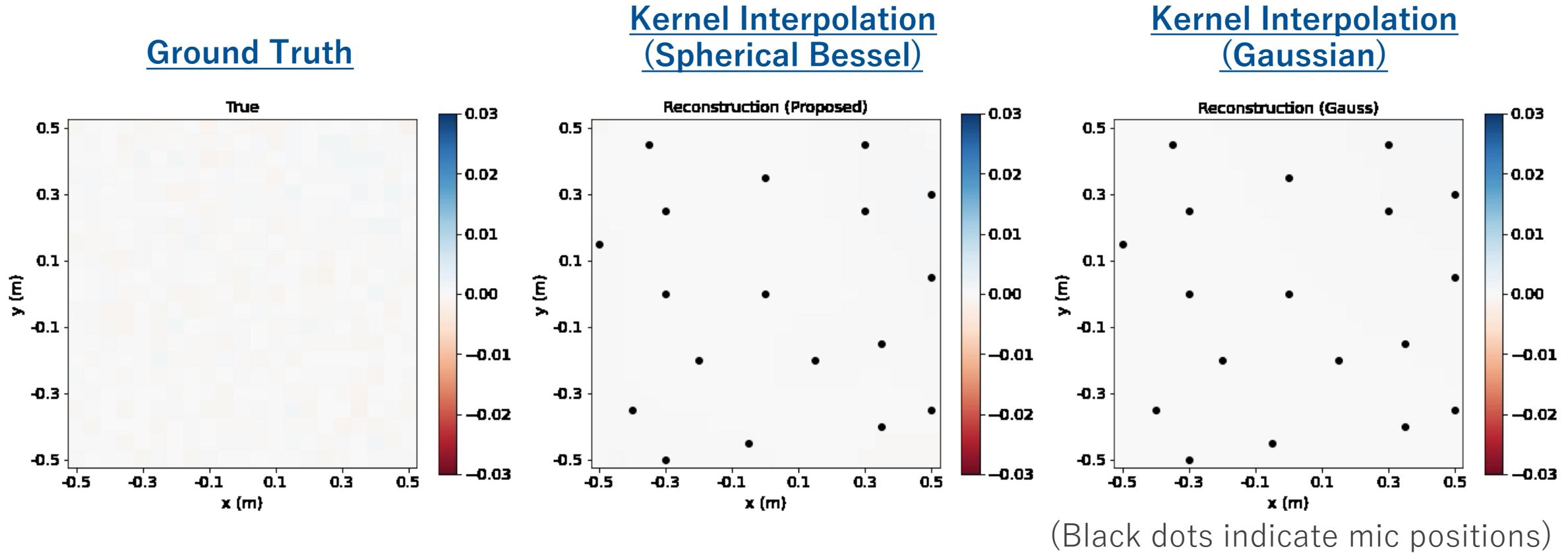


- When no prior information, i.e., uniform weight $w(\boldsymbol{\eta}) = 1$,

$$\kappa(\mathbf{r}_1, \mathbf{r}_2) = j_0(k\|\mathbf{r}_2 - \mathbf{r}_1\|)$$

Kernel regression with constraint of governing PDE

- Experimental results using real data from MeshRIR dataset [Koyama+ 2021]
 - Reconstructing pulse signal from single loudspeaker w/ 18 mic



Why NNs in sound field estimation?

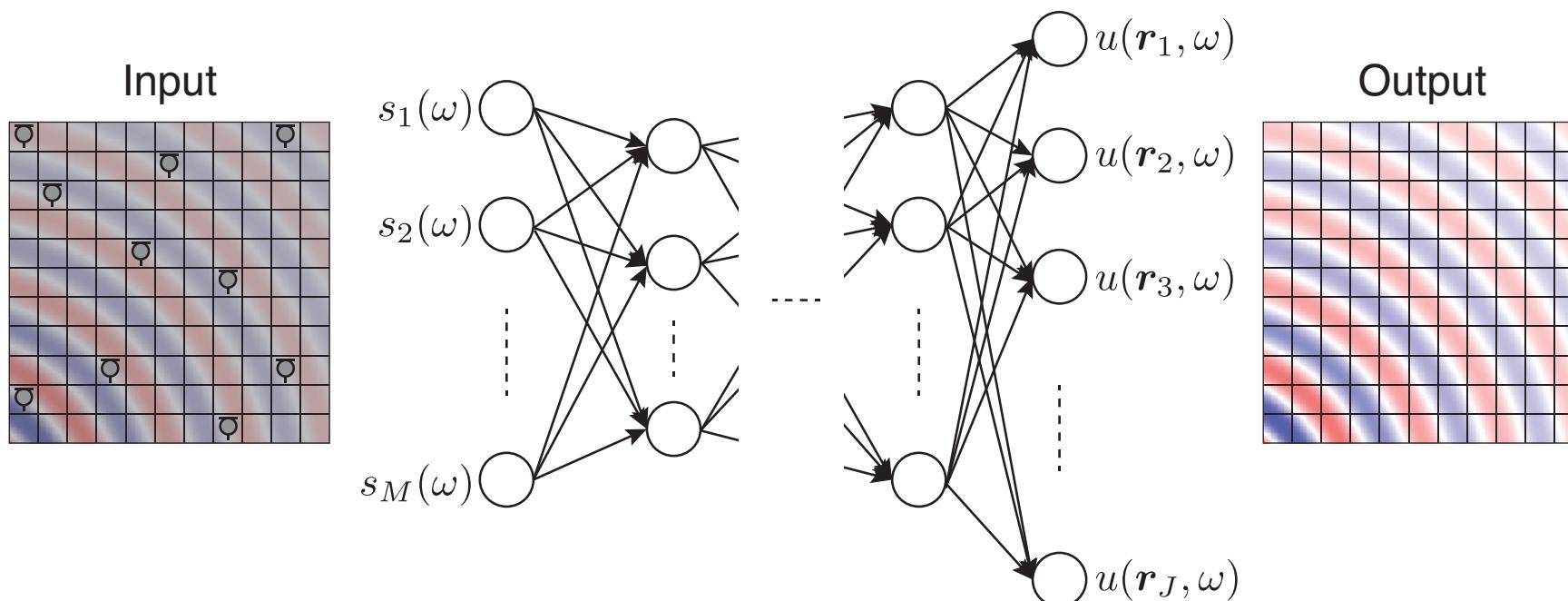
- High representational power
 - Solution space in basis expansion and kernel regression is highly constrained
 - High adaptability to the target acoustic environment can be expected by using NNs
 - From snapshot-based (**unsupervised**) to learning-based (**supervised**)
 - Basically, linear and kernel regressions use only a snapshot observation
 - Properties of the target acoustic environment can be learned from training data
-  **Highly accurate estimation can be expected, especially when the number of mics is extremely small**

Feedforward NNs incorporating governing PDEs

➤ Regression by feedforward NNs

- Target output is discretized as $\mathbf{t} = [f(\mathbf{x}_1), \dots, f(\mathbf{x}_J)]^\top$
- NN with input \mathbf{y} and output $\mathbf{g}(\mathbf{y}; \boldsymbol{\theta}_{\text{NN}})$ is designed with NN params $\boldsymbol{\theta}_{\text{NN}}$
- NN is trained using a pair of datasets $\{(\mathbf{y}_d, \mathbf{t}_d)\}_{d=1}^D$ to minimize the loss, e.g.,

$$\mathcal{J}(\boldsymbol{\theta}_{\text{NN}}) = \sum_{d=1}^D \|\mathbf{t}_d - \mathbf{g}(\mathbf{y}_d; \boldsymbol{\theta}_{\text{NN}})\|^2$$



How to embed governing PDEs to feedforward NNs?

➤ Estimating weights of basis expansion using NNs

- Train a NN estimating weights of basis expansion
- Continuous function can be reconstructed by using estimated expansion coeffs
- Can be regarded as **physics-constrained neural network (PCNN)** [Karakonstantis+ 2023, Lobato+ 2024]

➤ Incorporating (approximate) PDE loss

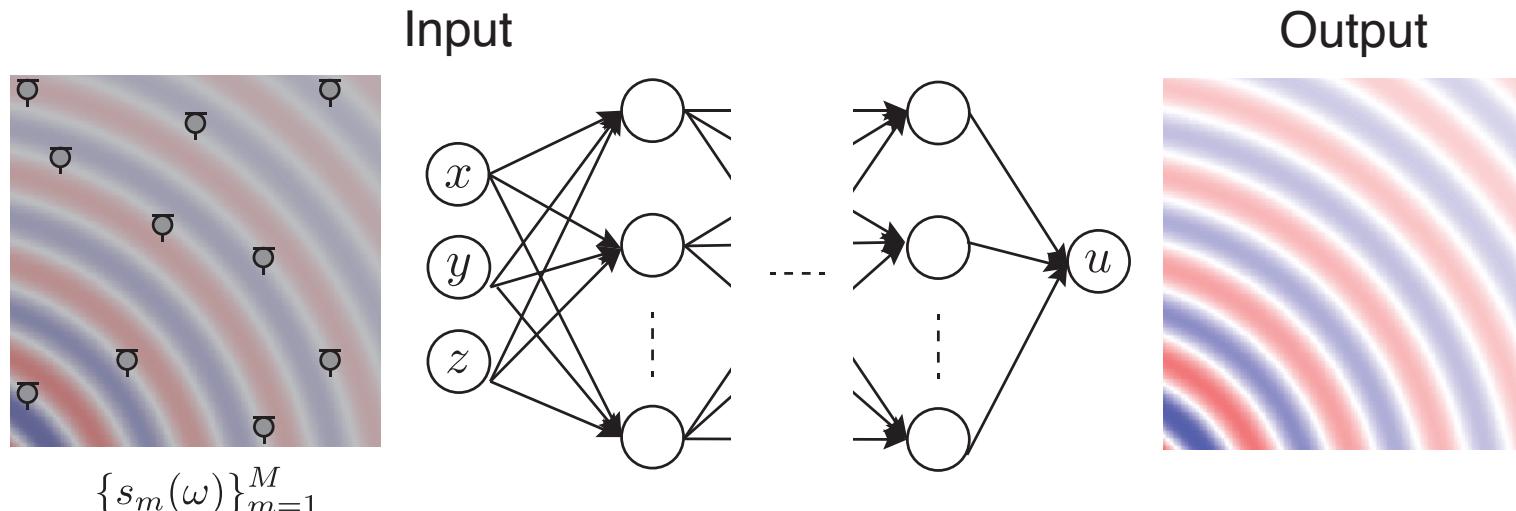
- Loss function evaluating deviation from governing PDEs: **PDE loss**
- Because of discrete output values, PDE loss is computed by finite difference or interpolation
- In [Shigemi+ 2022], **physics-informed convolutional neural network (PICNN)** using bi-cubic spline interpolation is proposed

PINNs based on implicit neural representation

➤ Implicit neural representation [Sitzmann+ 2020]

- NNs are used to implicitly represent a continuous function f
- NN with input \mathbf{x} and output $g(\mathbf{x}; \boldsymbol{\theta}_{\text{NN}})$ is designed with NN params $\boldsymbol{\theta}_{\text{NN}}$
- NN is trained for approximating $f(\mathbf{x})$ by using training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^I$

$$\mathcal{J}_{\text{INR}}(\boldsymbol{\theta}_{\text{NN}}) = \sum_{i=1}^I |y_i - g(\mathbf{x}_i; \boldsymbol{\theta}_{\text{NN}})|^2$$



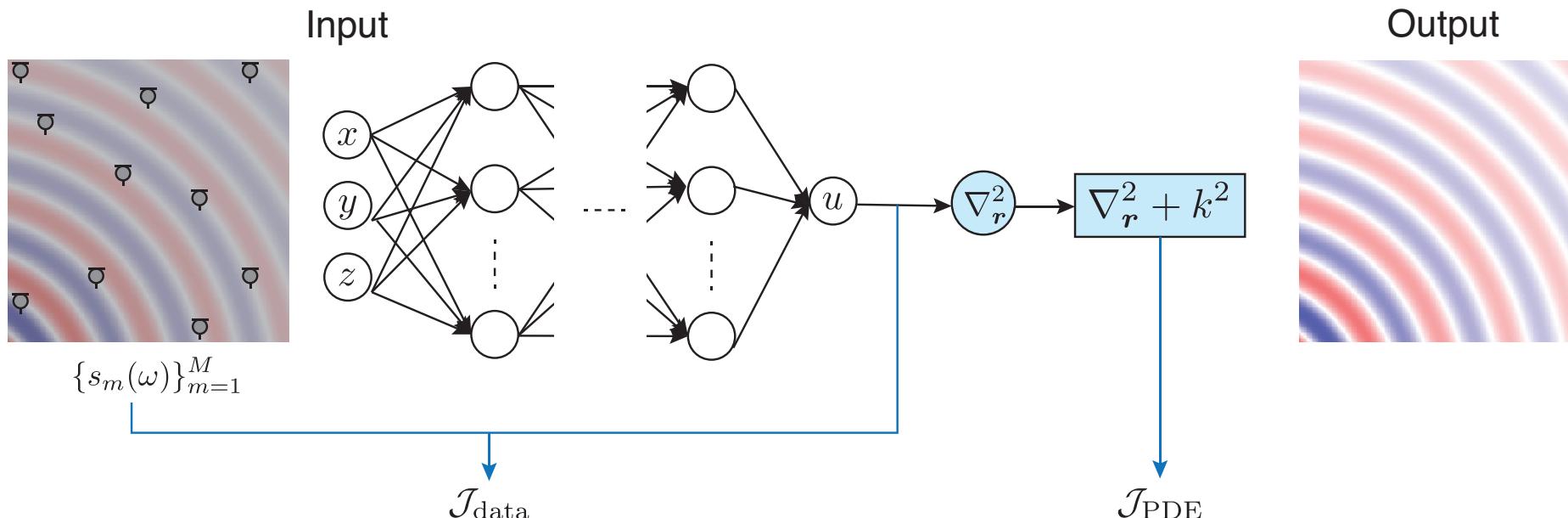
PINNs based on implicit neural representation

➤ Physics-informed neural network (PINN) [Raissi+ 2019]

- Implicit neural representation allows incorporating constraints on g including its (partial) derivatives in loss function

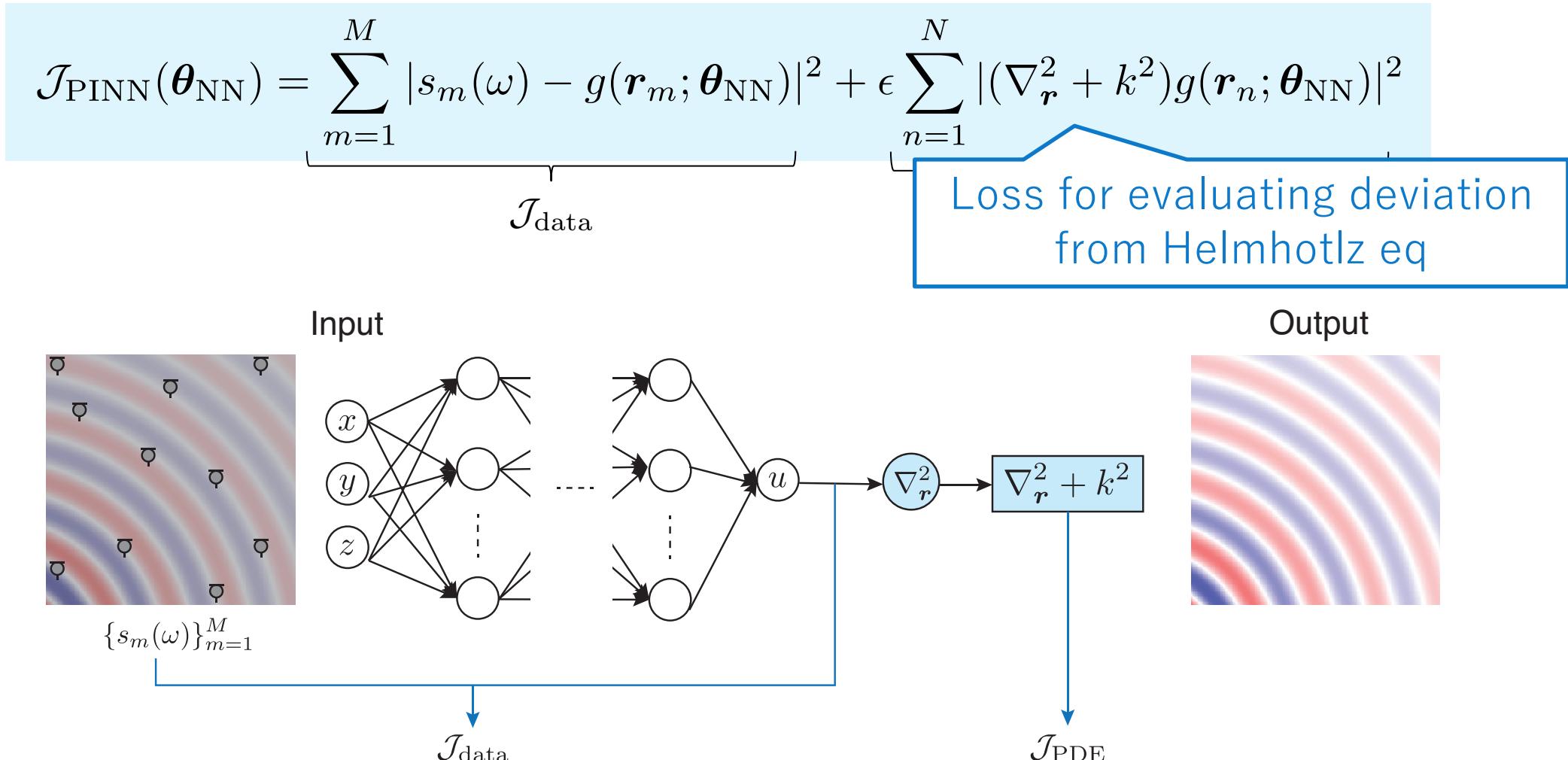
$$\mathcal{J}_{\text{INR}}(\boldsymbol{\theta}_{\text{NN}}) = \sum_{i=1}^I |y_i - g(\mathbf{x}_i; \boldsymbol{\theta}_{\text{NN}})|^2 + \epsilon \sum_{n=1}^N |H(g(\mathbf{x}_n), \nabla_{\mathbf{x}}g(\mathbf{x}_n), \nabla_{\mathbf{x}}^2g(\mathbf{x}_n), \dots)|^2$$

Usually computed by
automatic differentiation



PINNs based on implicit neural representation

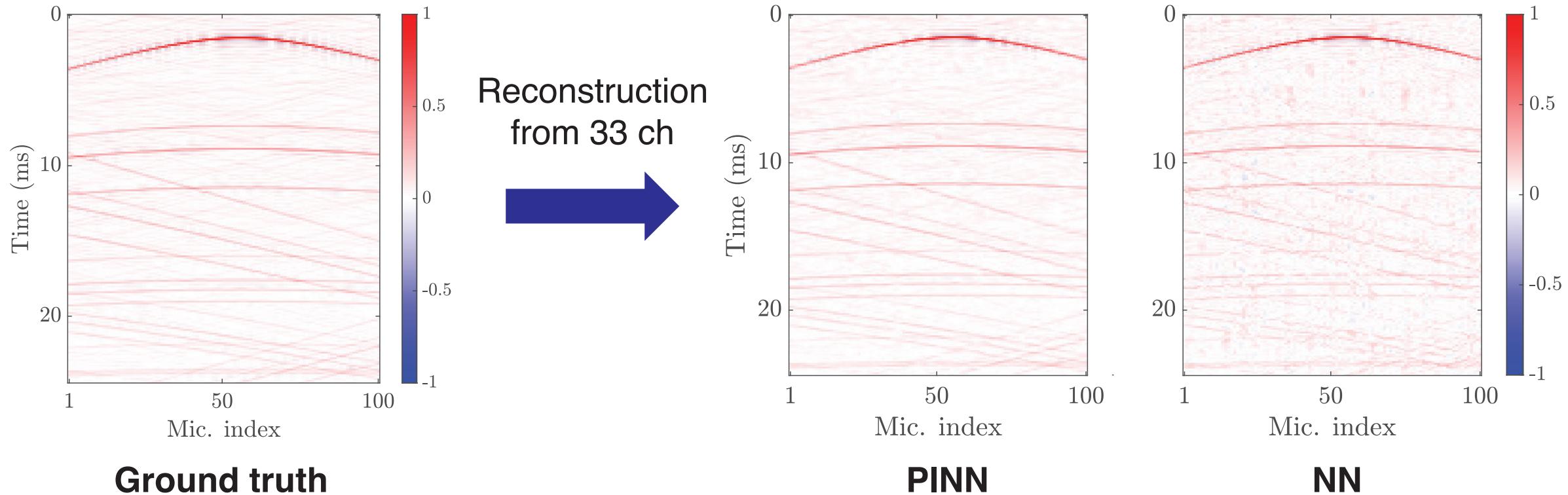
- Physics-informed neural network (PINN) [Raissi+ 2019]
 - Case when estimating function approximately satisfying Helmholtz eq



PINNs based on implicit neural representation

➤ PINNs for reconstructing RIRs in time domain [Pezzoli+ 2023]

- RIRs measured by array of 100 mics are reconstructed using only 33 channels



Embedding physical properties in interpolation techniques

Four techniques to incorporate governing PDEs

➤ Basis expansion into element solutions

- Plane wave expansion, spherical wave function expansion, equivalent source distribution
- Expansion coefficients are obtained by linear regression

➤ Kernel regression with constraint of governing PDEs

- Infinite dimensional extension of basis expansion
- Kernel function to constrain the solution to satisfy Helmholtz eq

➤ Feedforward NNs incorporating governing PDEs

- Feedforward NNs to estimate discrete target output
- Setting output to expansion coeffs or using approximate PDE loss

➤ PINNs based on implicit neural representation

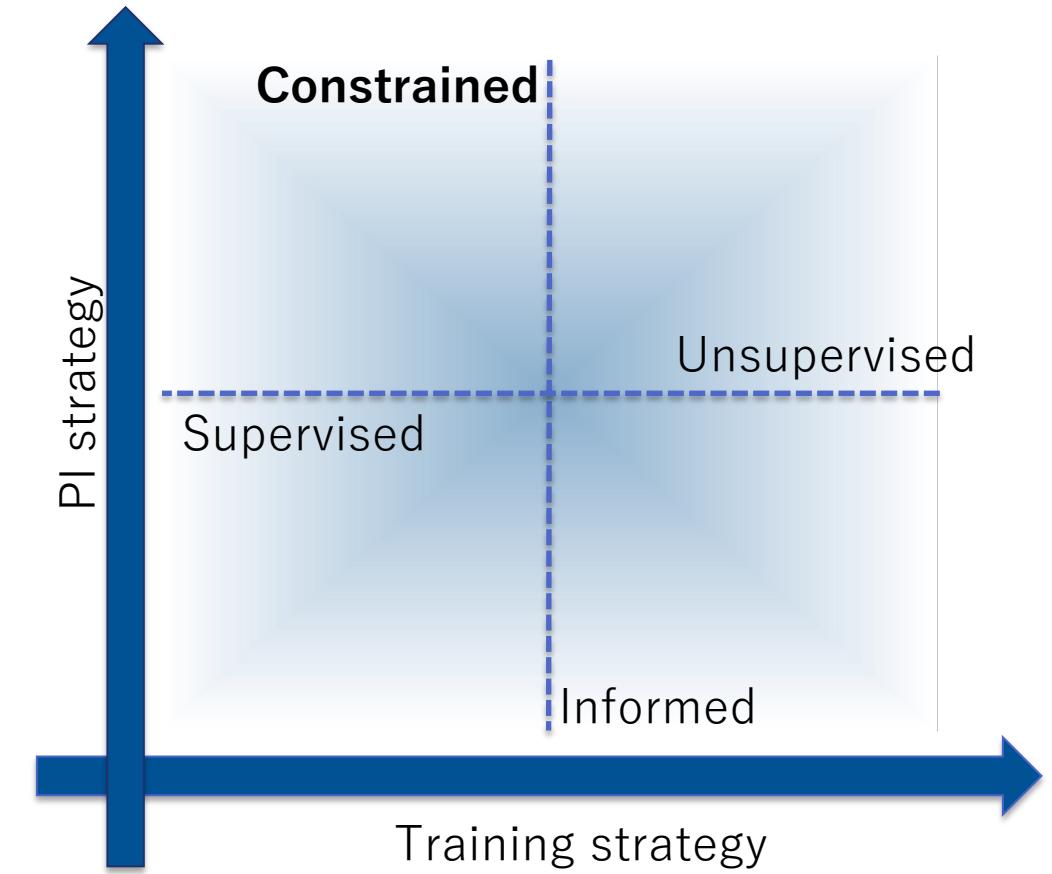
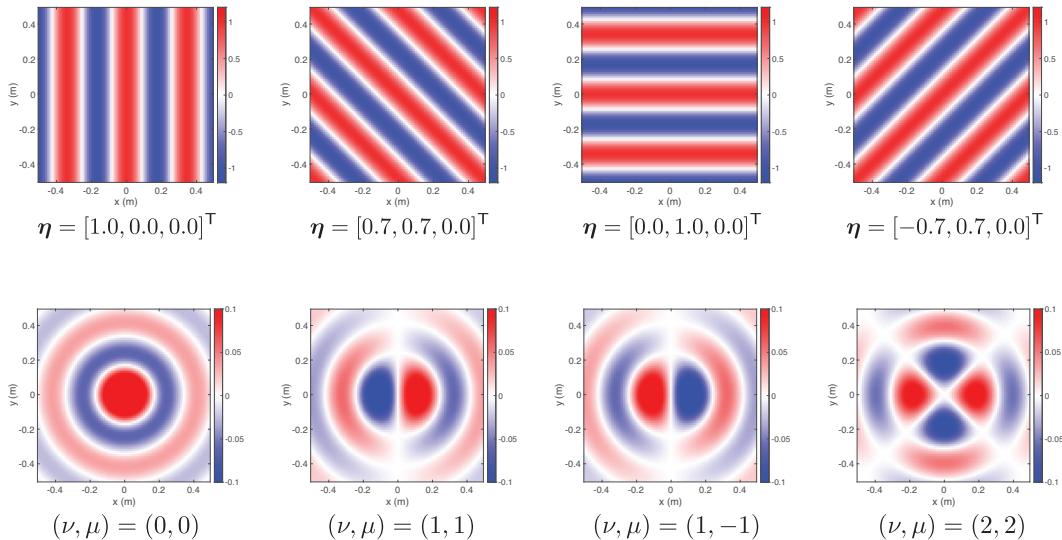
- NNs to implicitly represent continuous function
- PDE loss computed by automatic differentiation

Part II – Current studies of PIML for audio processing

PHYSICS-INFORMED MACHINE LEARNING FOR AUDIO PROCESSING

Physics prior is introduced as:

- **Forced** to adhere to physical model
 - Solutions of wave equation
- **No deviations** of the solution are allowed
 - Less flexibility in challenging scenarios



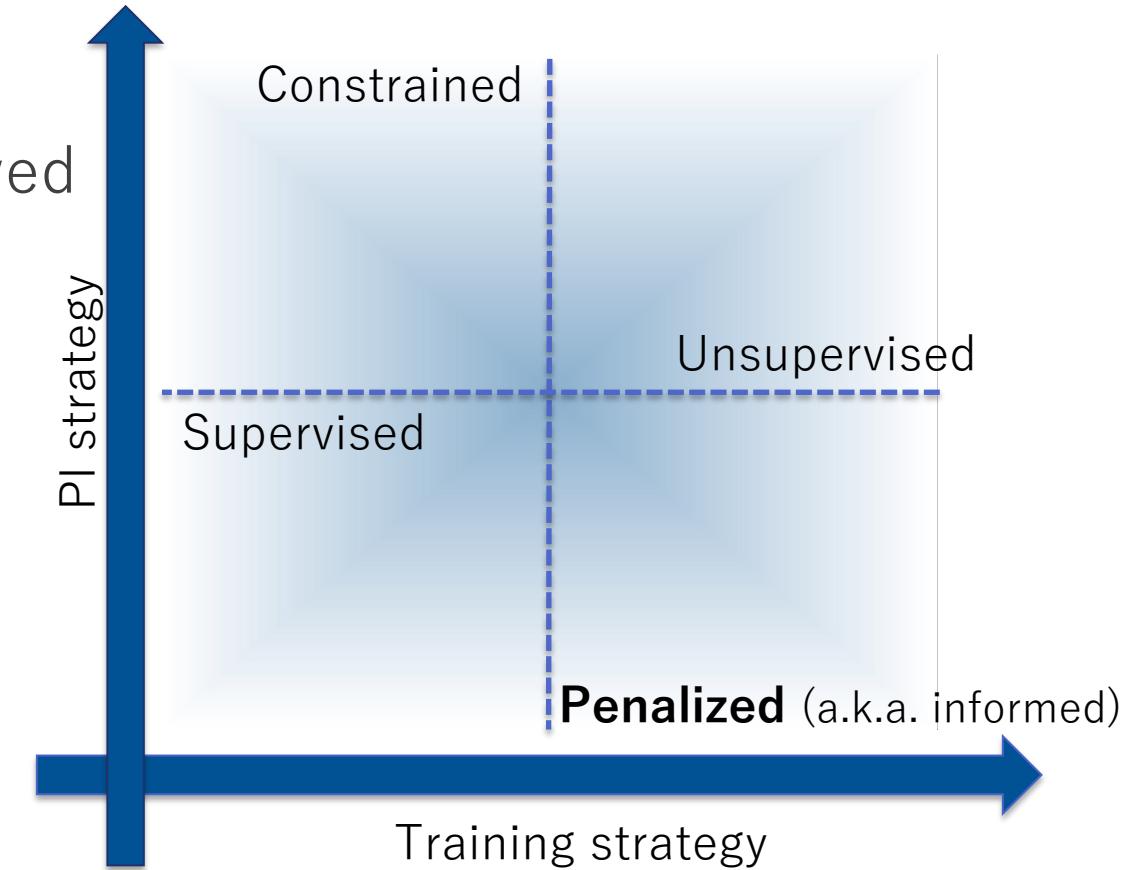
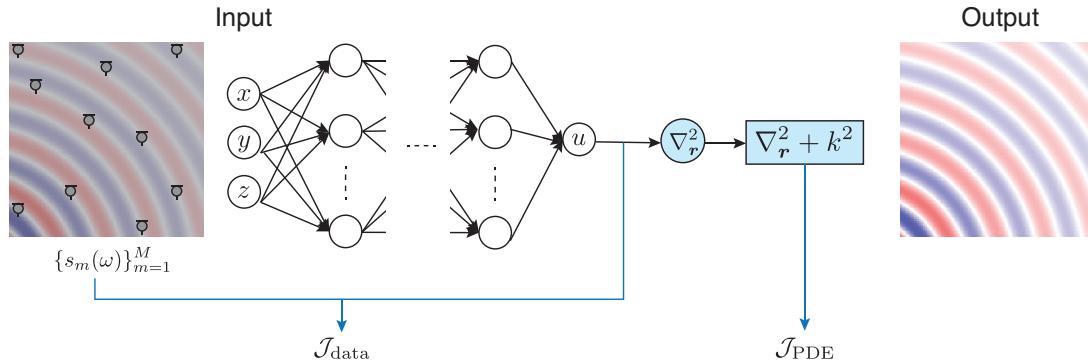
Physics prior is introduced as:

➤ **Penalization** term of the optimization

- Residual on wave/Helmholtz equation

➤ **Small deviations** of the solution are allowed

- More flexibility in challenging scenarios



PIML **training approach** is similar to standard ML

➤ **Supervised:** we have access to “ground truth”

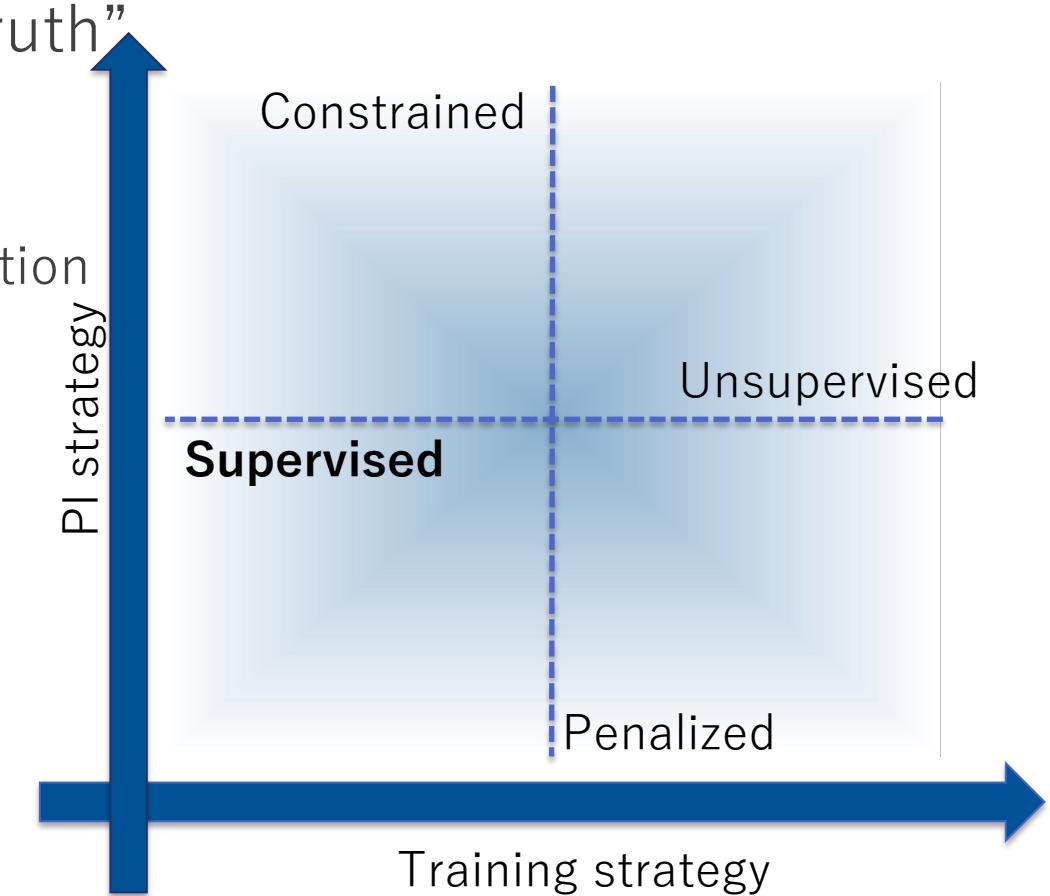
- Direct comparison between NN output and GT
- Training dataset \neq Test dataset
- Common scenario for regression and classification

➤ **Training stage**

- Dataset of measurements or simulations

➤ **Test stage**

- Inference on new data



PIML **training approach** is similar to standard ML

➤ **Supervised:** we have access to “ground truth”

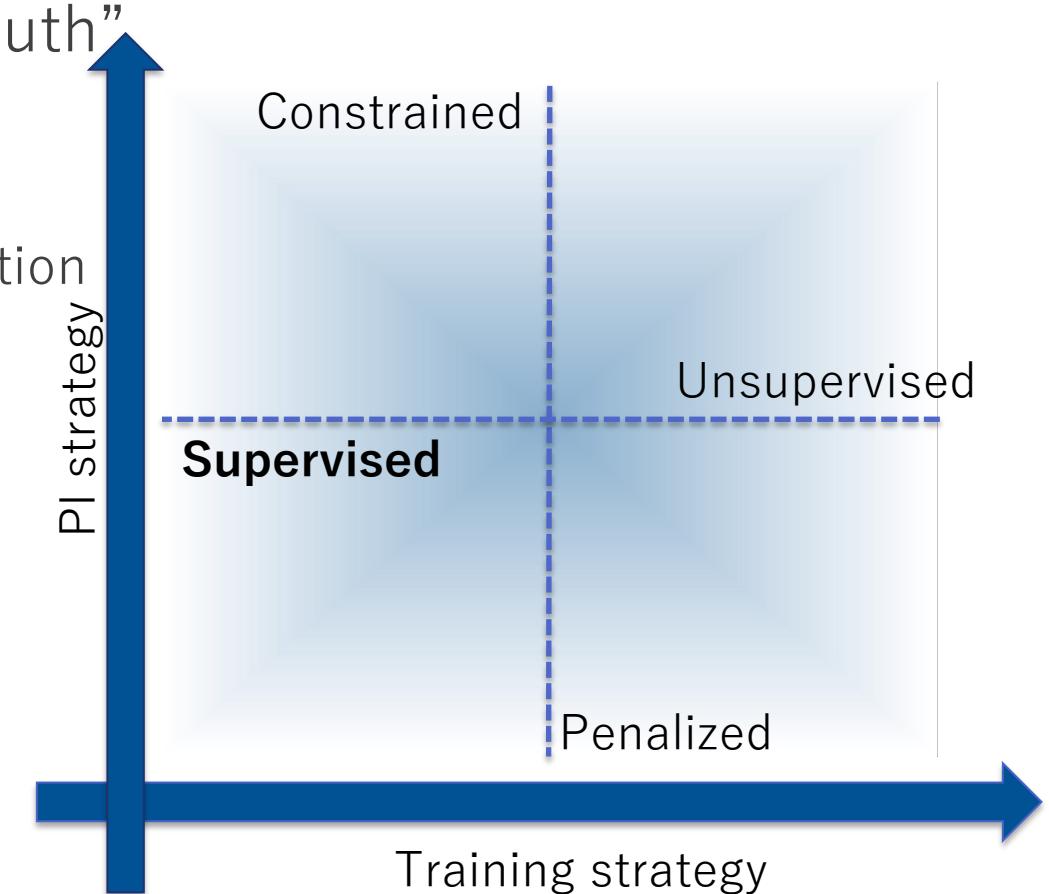
- Direct comparison between NN output and GT
- Training dataset \neq Test dataset
- Common scenario for regression and classification

➤ Pros:

- **Exploit available data**
- **Fast inference**

➤ Cons:

- **Generalization** is difficult



PIML **training approach** is different from standard ML

➤ Standard meaning for **unsupervised** in ML:

- No access to “ground truth”
- Common scenario for clustering

➤ **Unsupervised**

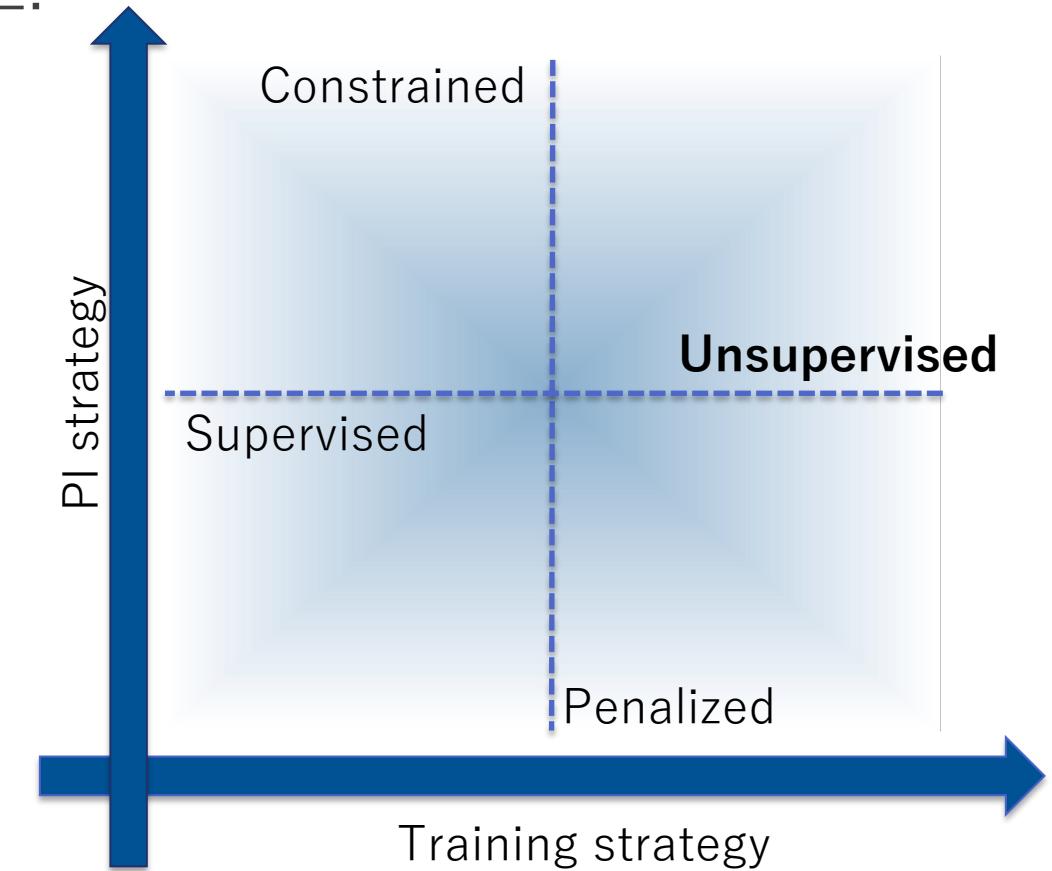
- no training dataset, “single-shot” training
- **Same conditions** for training and testing
- “overfit” the data

➤ **Training stage**

- Only available measurements are used

➤ **Test stage**

- Model applied on the same data



PIML training approach is different from standard ML

➤ **Unsupervised**

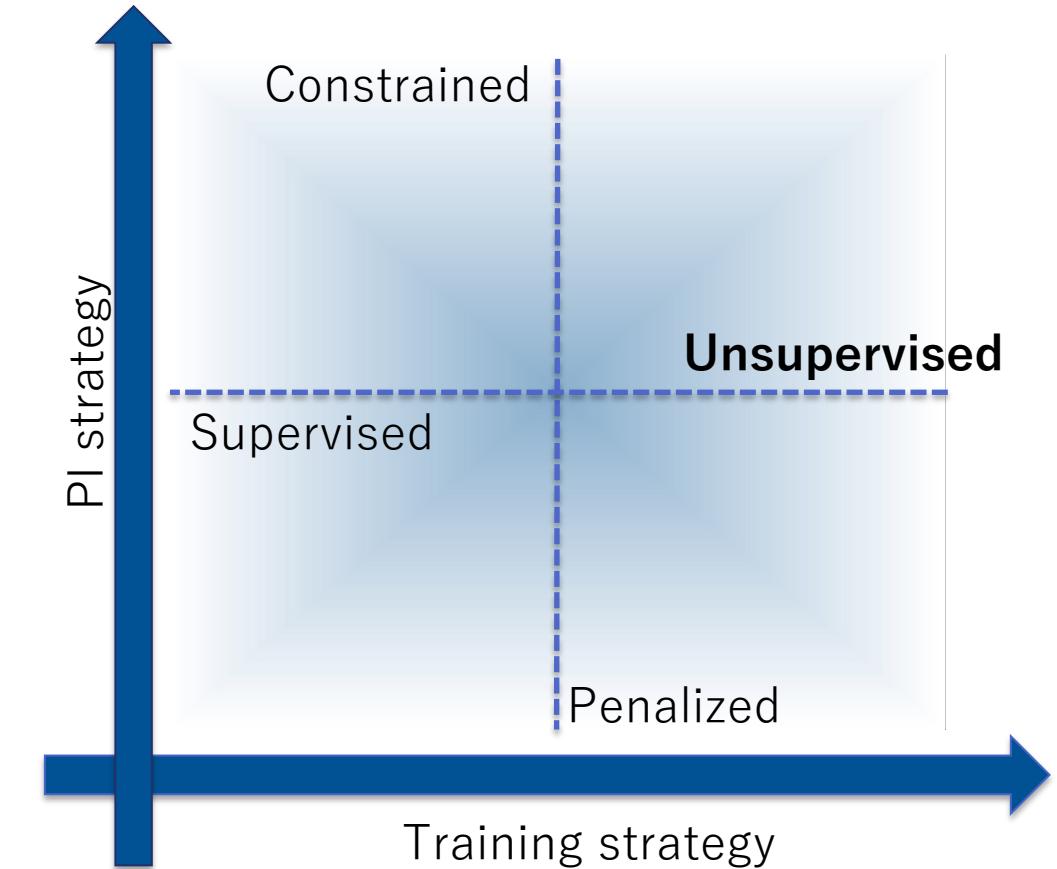
- no training dataset, “single-shot” training
- **Same conditions** for training and testing
- “overfit” the data

➤ Pros

- **No** need for **big** training **dataset**
- **No** “generalization” issues

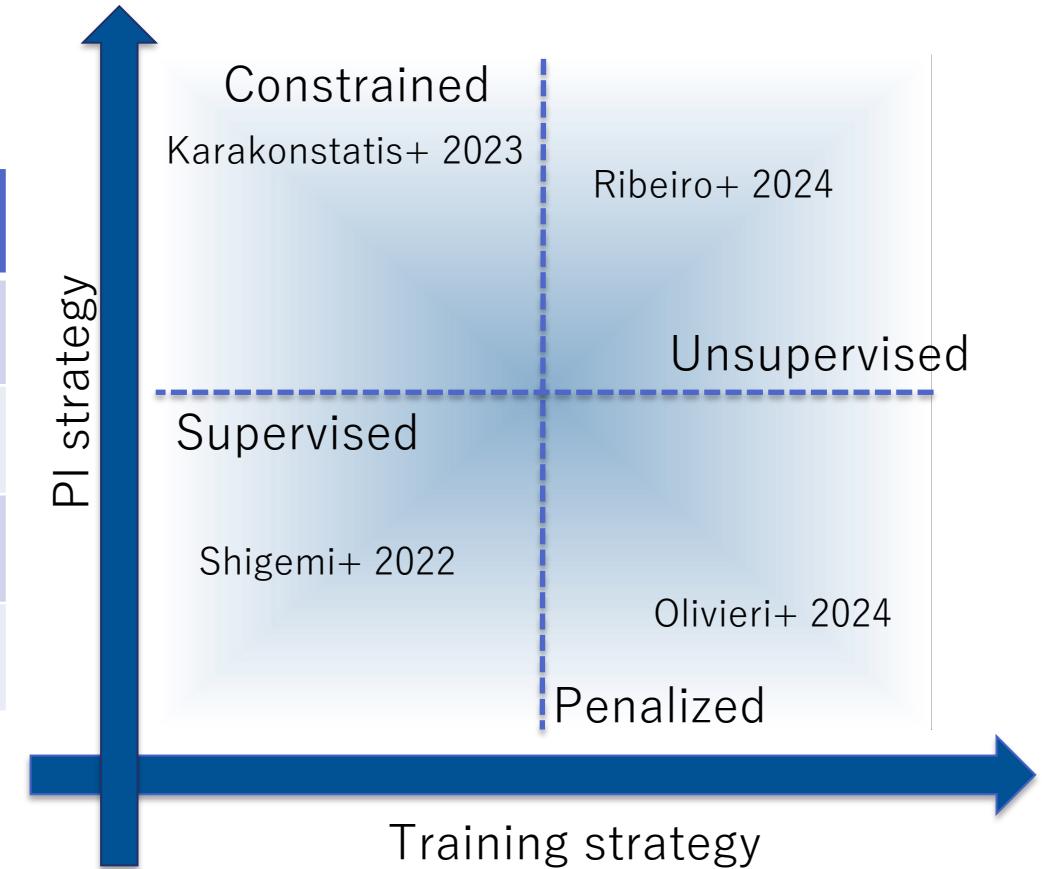
➤ Cons

- Does **not exploit** other available **datasets**
- Needs **re-training** for new scenarios



PIML techniques

Paper*	Supervised/Unsupervised	Estimator	Domain	Physical Property
Shigemi+ 2022	Supervised	Nonlinear	Frequency	Penalized
Karakonstatis + 2023	Supervised	Linear	Frequency	Constrained
Olivieri+ 2024	Unsupervised	Nonlinear	Time	Penalized
Ribeiro+ 2024	Unsupervised	Linear	Frequency	Constrained



*selection from our IEEE SPM paper

PIML techniques

Paper*	Supervised/Unsupervised	Estimator	Domain	Physical Property
Shigemi+ 2022	Supervised	Nonlinear	Frequency	Penalized
Karakonstatis + 2023	Supervised	Linear	Frequency	Constrained
Olivieri+ 2024	Unsupervised	Nonlinear	Time	Penalized
Ribeiro+ 2024	Unsupervised	Linear	Frequency	Constrained



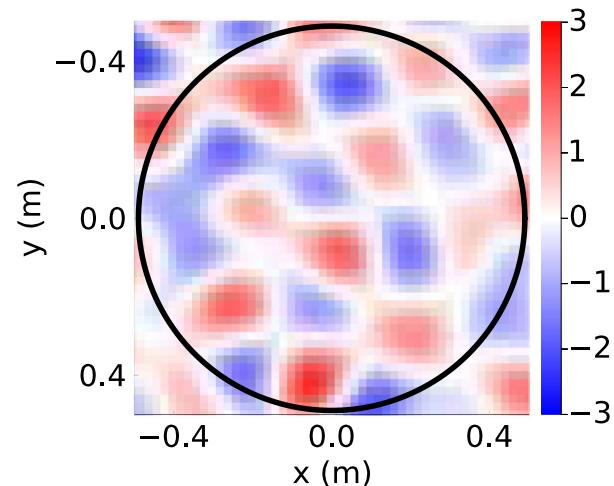
*selection from our IEEE SPM paper

“Sound Field Estimation Based on Physics-Constrained Kernel Interpolation Adapted to Environment”

Juliano G. C. Ribeiro, Shoichi Koyama, Ryosuke Horiuchi, and Hiroshi Saruwatari

IEEE/ACM Trans. Audio, Speech, Lang. Process.

- KRR in combination with NN for carefully model the sound field
- Frequency-domain model, unsupervised, constrained



Reproducing kernel function adapted to acoustic environment using neural networks

- Kernel function with constraint of Helmholtz eq is optimized to acoustic environment with the aid of neural networks [Ribeiro+ 2024]
 - Superposition of two kernel functions

$$\kappa = \kappa_{\text{dir}} + \kappa_{\text{res}}$$

The diagram illustrates the decomposition of the total kernel κ into its components. A central equation shows $\kappa = \kappa_{\text{dir}} + \kappa_{\text{res}}$. Two arrows point from boxes below the equation to the terms: a blue arrow points to the term κ_{dir} and a red arrow points to the term κ_{res} . The box for the directed kernel is blue, and the box for the residual kernel is red.

- **Directed kernel**: direct source and early reflections
- **Residual kernel**: late reverberations and residual components

Reproducing kernel function adapted to acoustic environment using neural networks

➤ Directed kernel

- Directional weighting with weighted sum of (sparse) von Mises—Fisher distribution [Horiuchi+ 2021]

$$w_{\text{dir}}(\boldsymbol{\eta}; \boldsymbol{\beta}, \boldsymbol{\gamma}) = \sum_{n=1}^N \gamma_n \frac{e^{\beta_n \langle \boldsymbol{\eta}, \mathbf{d}_n \rangle}}{C(\beta_n)} \quad (\|\boldsymbol{\gamma}\|_1 = 1)$$

→ Sparsity constraint

$$\kappa_{\text{dir}}(\mathbf{r}_1, \mathbf{r}_2) = \sum_{n=1}^N \gamma_n \frac{j_0 \left(\sqrt{(\mathbf{j}\beta\boldsymbol{\eta} - k\mathbf{r}_{12})^\top (\mathbf{j}\beta\boldsymbol{\eta} - k\mathbf{r}_{12})} \right)}{C(\beta_n)}$$

Normalization constant

Reproducing kernel function adapted to acoustic environment using neural networks

➤ Residual kernel

- Directional weighting with implicit neural representation

$$w_{\text{res}}(\boldsymbol{\eta}; \boldsymbol{\theta}) = \text{NN}(\boldsymbol{\eta}; \boldsymbol{\theta}) : \text{Implicit neural representation}$$

$$\kappa_{\text{res}}(\mathbf{r}_1, \mathbf{r}_2) = \int_{\mathbb{S}_2} w_{\text{res}}(\boldsymbol{\eta}; \boldsymbol{\theta}) e^{-jk\langle \boldsymbol{\eta}, \mathbf{r} \rangle} d\boldsymbol{\eta}$$

→ Computed by numerical integration

Reproducing kernel function adapted to acoustic environment using neural networks

- Again, (positive-definite) kernel function is the sum of directed and residual kernels

$$\kappa = \kappa_{\text{dir}} + \kappa_{\text{res}}$$

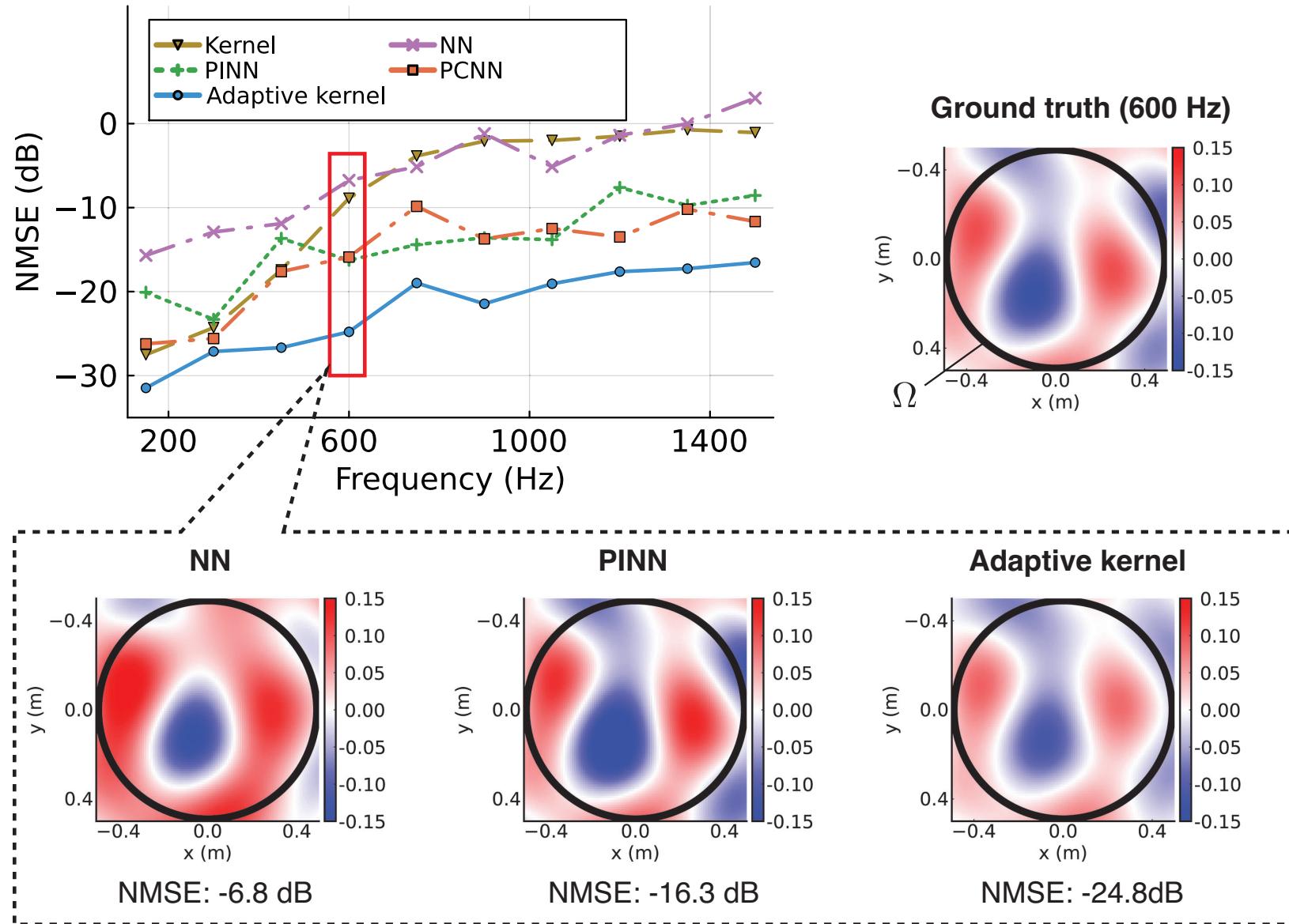
Directed kernel **Residual kernel**

The diagram illustrates the decomposition of a kernel function. At the top, the symbol κ is followed by an equals sign. Below the equals sign, there is a blue line segment pointing to a blue rectangular box containing the text "Directed kernel". To the right of the plus sign, there is a red line segment pointing to a red rectangular box containing the text "Residual kernel".

- Hyperparameters β, γ, θ are jointly optimized by a steepest-descent-based algorithm
- The method is physics-constrained
- Estimation process is still linear operation in freq domain based on kernel ridge regression

Neural kernel for sound field estimation

➤ Numerical experiment: T60: 400 ms, # mics: 41, spherical shell array



PIML techniques

Paper*	Supervised/Unsupervised	Estimator	Domain	Physical Property
Shigemi+ 2022	Supervised	Nonlinear	Frequency	Penalized
Karakonstatis + 2023	Supervised	Linear	Frequency	Constrained
Olivieri+ 2024	Unsupervised	Nonlinear	Time	Penalized
Ribeiro+ 2024	Unsupervised	Linear	Frequency	Constrained

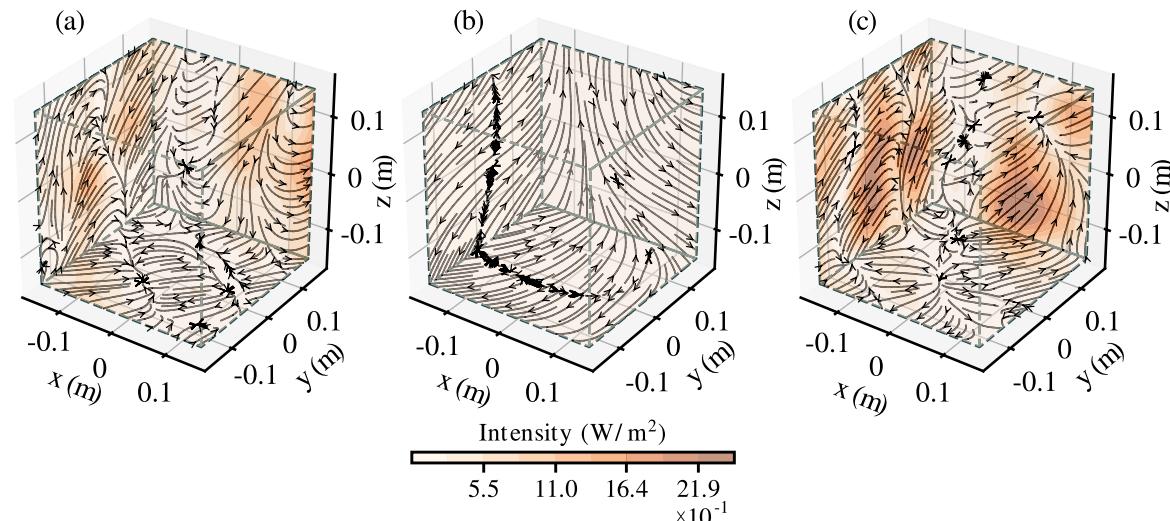


*selection from our IEEE SPM paper

“Physics-informed neural network for volumetric sound field reconstruction of speech signals”

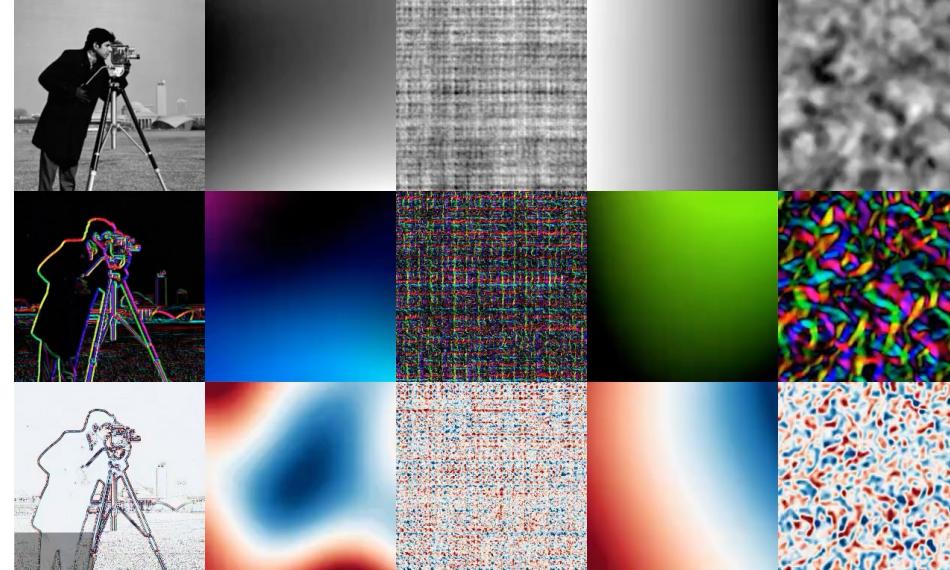
Marco Olivieri, Xenofon Karakonstantis, Mirco Pezzoli, Fabio Antonacci, Augusto Sarti and Efren Fernandez-Grande
EURASIP J. Audio, Speech, Music Process.

- Physics-informed neural network for sound field reconstruction
- Time domain model, unsupervised, penalized



Implicit Neural Representation (INR)

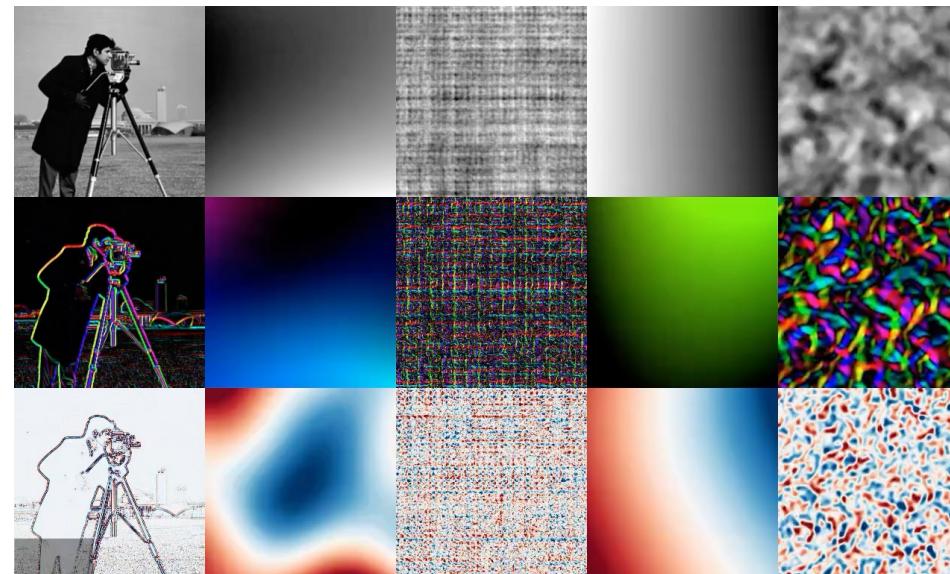
- Implicit continuous differentiable representation of function f (a.k.a. Neural Field)
- Proved to be effective for different classes of signals (images, videos, point clouds etc.)



Implicit Neural Representation (INR)

INR is used to implicitly represent the continuous function f

- Input is the domain \mathbf{x} of f sampled in $\{(\mathbf{x}_i, y_i)\}_{i=1}^I$
- Output are the value of f in $\{(\mathbf{x}_i, y_i)\}_{i=1}^I$
- Typically, small MLPs are used



Sinusoidal representation networks (SIREN) [Sitzmann+ 2020]

- MLP structure with sinusoidal activations

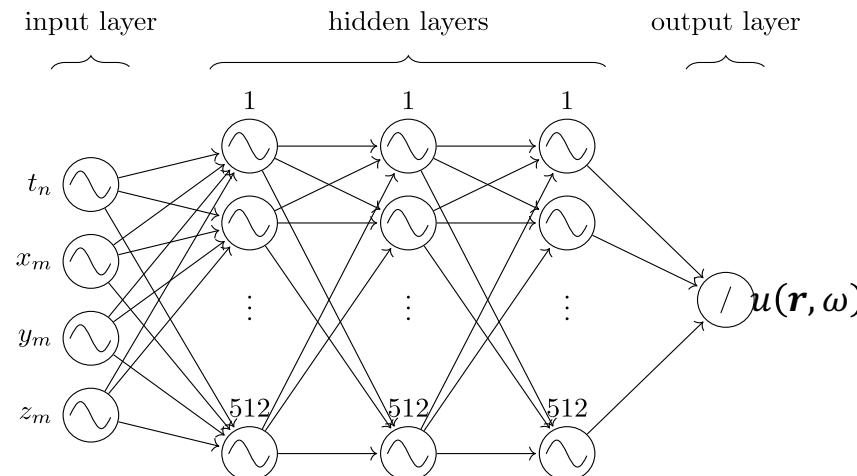
$$g(\mathbf{x}; \boldsymbol{\theta}) = (\phi_L \circ \phi_{L-1} \circ \dots \circ \phi_1)(\mathbf{x})$$

Learnable parameters

- Sine layer

$$\phi_i(\mathbf{x}_i) = \sin(\omega_0 \mathbf{x}_i^T \boldsymbol{\theta}_i + b_i)$$

→ Derivatives of SIREN are still SIREN

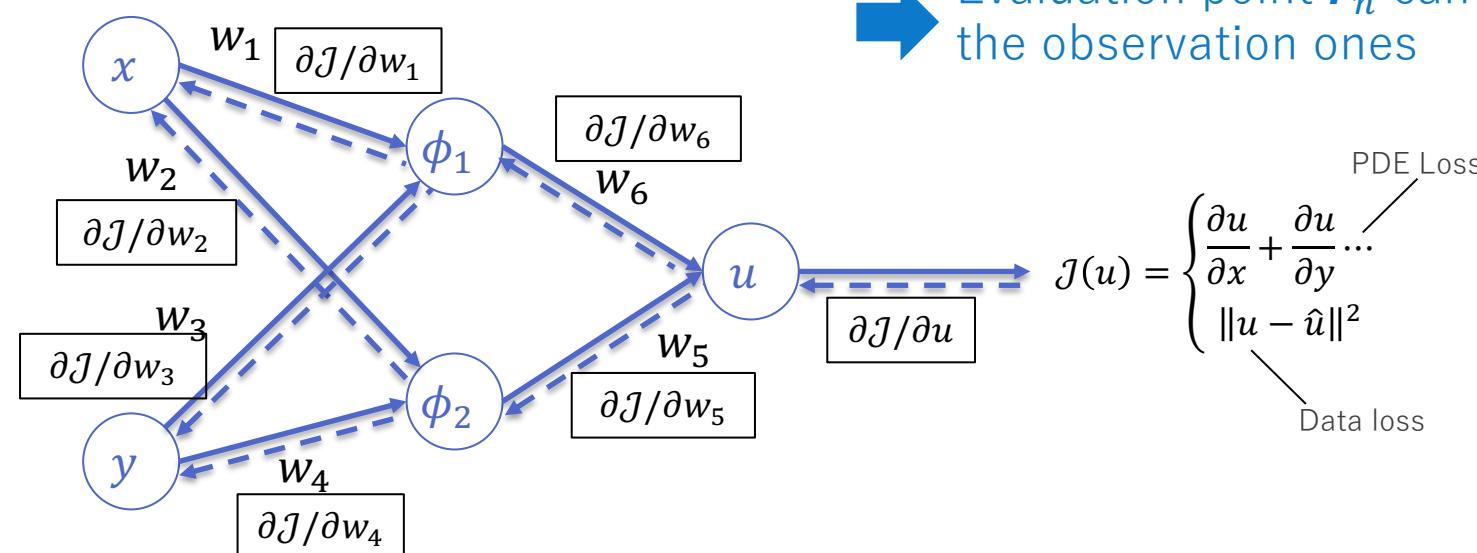


Physics-informed SIREN (PI-SIREN)

- Being INR, SIREN allows for imposing constraints on its derivatives
 - Derivatives are implemented using automatic differentiation
- Penalizing reconstruction using the residual of wave equation

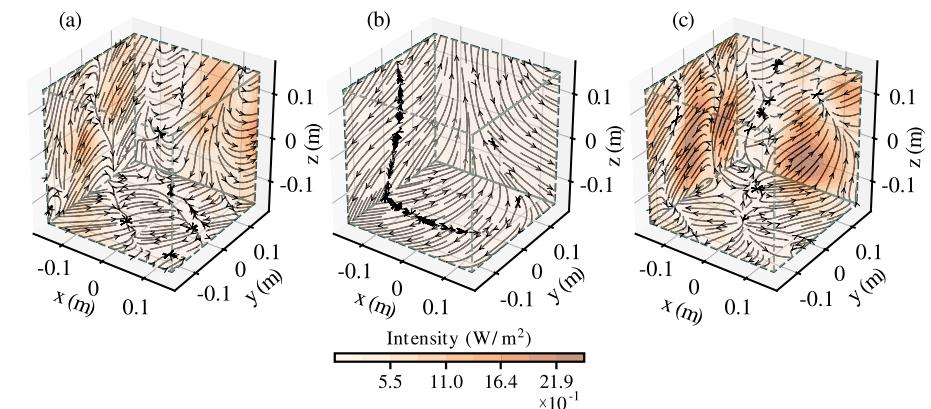
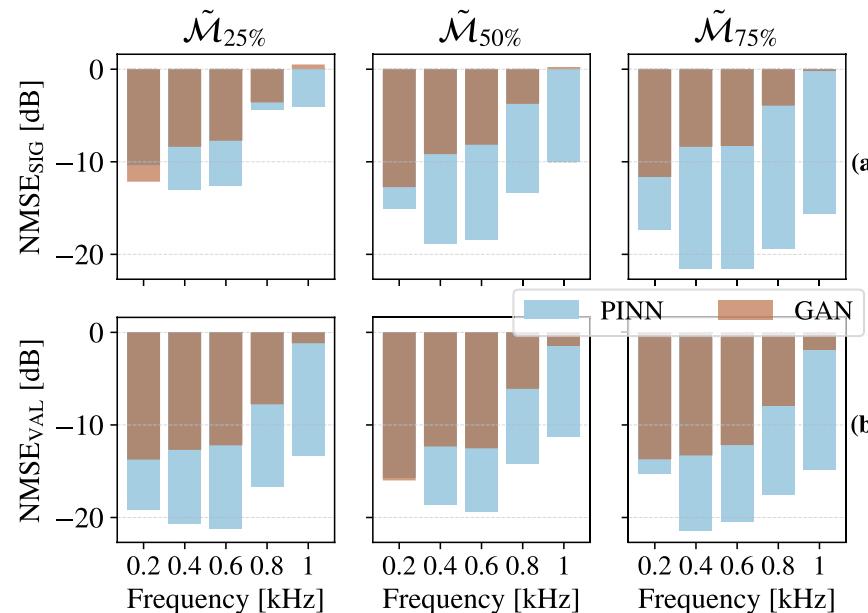
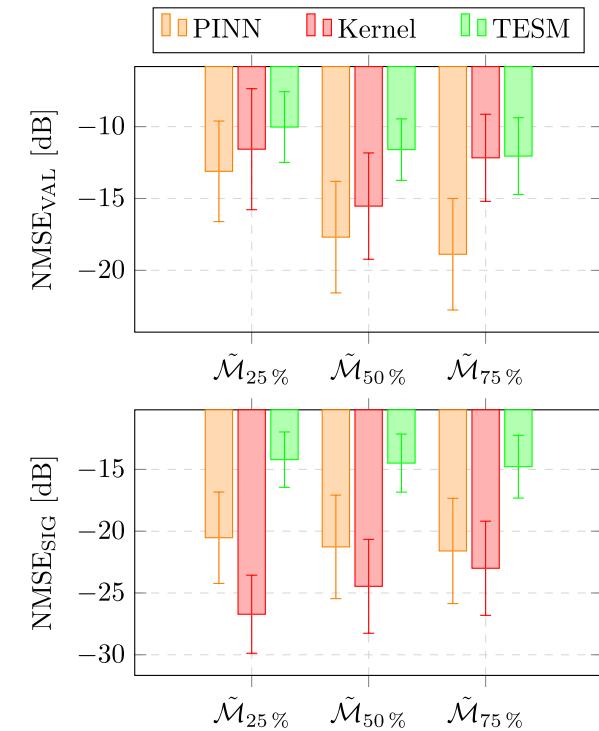
$$\mathcal{J}_{\text{PDE}} = \sum_{n=1}^N \left| (\nabla_r^2 + \frac{1}{c^2} \frac{\partial^2}{\partial t^2}) g(\mathbf{r}_n, t; \boldsymbol{\theta}_{NN}) \right|^2$$

→ Evaluation point \mathbf{r}_n can be different from the observation ones



PINN for sound field estimation

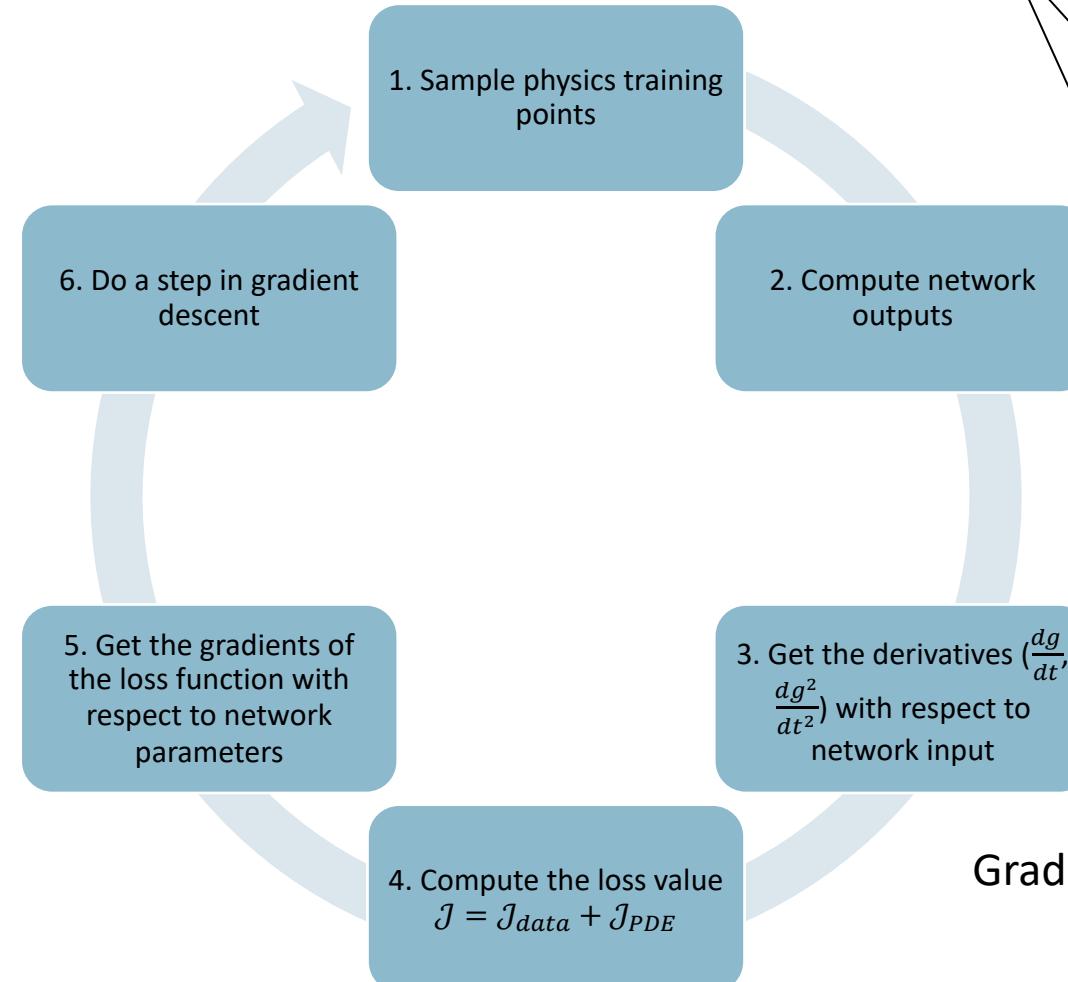
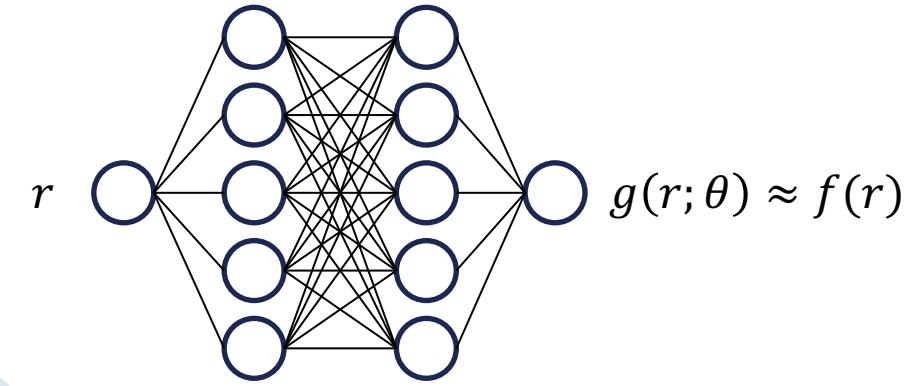
- Evaluation on speech sound field using real measurements from MeshRIR [Koyama+ 2021]



[Olivieri+ 2024]

Training of PINN

- Data loss $\mathcal{J}_{\text{Data}} = \sum_i (y_i - g(x_i; \theta))^2$
- Physics loss $\mathcal{J}_{\text{PDE}} = \sum_{n=1}^N \left| (\nabla_r^2 + \frac{1}{c^2} \frac{\partial^2}{\partial t^2}) g(\mathbf{r}_n, t; \theta_{NN}) \right|^2$



Training of PINN

- Data loss $\mathcal{J}_{\text{Data}} = \sum_i (y_i - g(x_i; \theta))^2$
- Physics loss $\mathcal{J}_{\text{PDE}} = \sum_{n=1}^N \left| (\nabla_r^2 + \frac{1}{c^2} \frac{\partial^2}{\partial t^2}) g(\mathbf{r}_n, t; \theta_{NN}) \right|^2$

Forward problem (solving PDE)
→ we need more loss terms:

- Boundary loss
- Initial condition loss

Data loss is optional!

More info later in part III

1. Sample physics training points

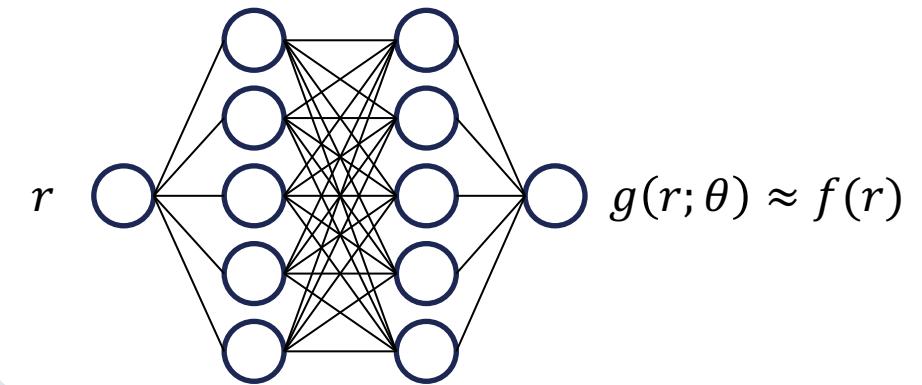
a step in gradient descent

2. Compute network outputs

5. Get the gradients of the loss function with respect to network parameters

3. Get the derivatives ($\frac{dg}{dt}$, $\frac{dg^2}{dt^2}$) with respect to network input

4. Compute the loss value $\mathcal{J} = \mathcal{J}_{\text{data}} + \mathcal{J}_{\text{PDE}}$



Gradients are computed using autodiff

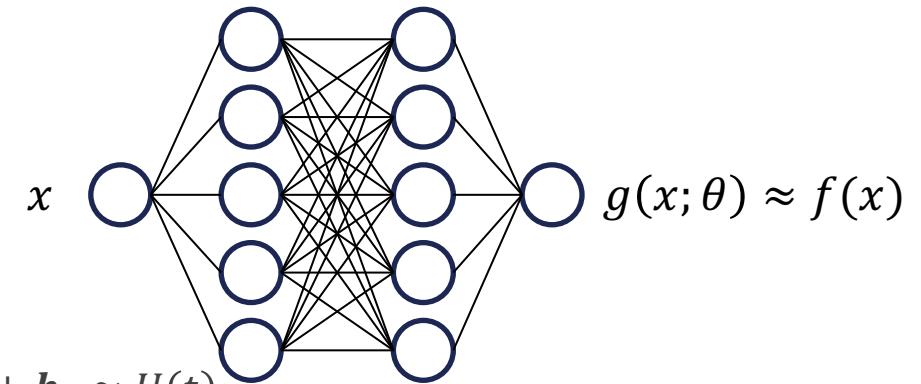
Gradient computation

- NN are trained using the so-called backpropagation
- Basically, the multivariate chain rule of differentiation

- Assuming MLP:

$$g(t; \theta) = \mathbf{W}_3 (\sigma(\mathbf{W}_2 \sigma(\mathbf{W}_1 t + \mathbf{b}_1) + \mathbf{b}_2) + \mathbf{b}_3) \approx U(t)$$

- \mathbf{W} are the weights of a layer
- \mathbf{b} are the biases
- σ is the nonlinear function



$$\begin{aligned} g(t; \theta) &= h_3 \circ h_2 \circ h_1(t; \theta) \\ h_3 &= \mathbf{W}_3 \sigma(h_2) + \mathbf{b}_3 \\ h_2 &= \mathbf{W}_2 \sigma(h_1) + \mathbf{b}_2 \\ h_1 &= \mathbf{W}_1 t + \mathbf{b}_1 \\ \frac{\partial g}{\partial t} &= \frac{\partial h_3}{\partial h_2} \frac{\partial h_2}{\partial h_1} \frac{\partial h_1}{\partial t} \end{aligned}$$

Chain rule to compute gradients

$$\frac{\partial g}{\partial t} = \mathbf{W}_3 diag(\sigma'(h_2)) \mathbf{W}_2 diag(\sigma'(h_1)) \mathbf{W}_1$$

Computational graph

- Computational graph is extended using gradients

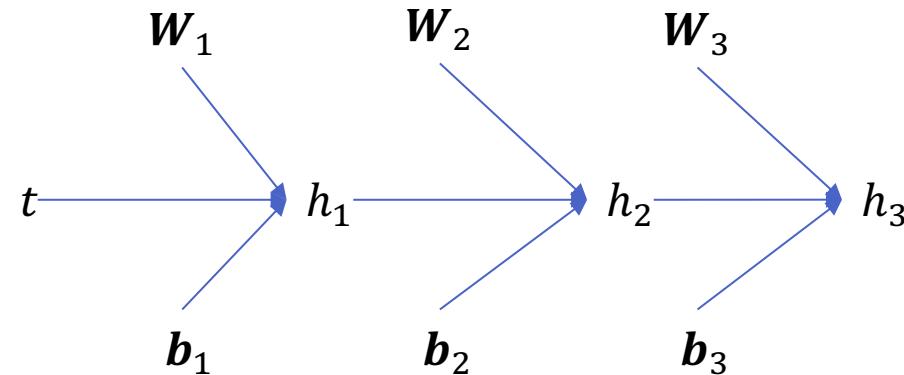
$$g(t; \theta) = \mathbf{W}_3(\sigma(\mathbf{W}_2\sigma(\mathbf{W}_1t + \mathbf{b}_1) + \mathbf{b}_2) + \mathbf{b}_3$$

$$h_3 = \mathbf{W}_3\sigma(h_2) + \mathbf{b}_3$$

$$h_2 = \mathbf{W}_2\sigma(h_1) + \mathbf{b}_2$$

$$h_1 = \mathbf{W}_1t + \mathbf{b}_1$$

$$\frac{\partial g}{\partial t} = \mathbf{W}_3 \text{diag}(\sigma'(h_2)) \mathbf{W}_2 \text{diag}(\sigma'(h_1)) \mathbf{W}_1$$



Computational graph

- Computational graph is extended using gradients
- We can compute the gradient with respect to the loss function

$$g(t; \theta) = \mathbf{W}_3(\sigma(\mathbf{W}_2\sigma(\mathbf{W}_1t + \mathbf{b}_1) + \mathbf{b}_2) + \mathbf{b}_3)$$

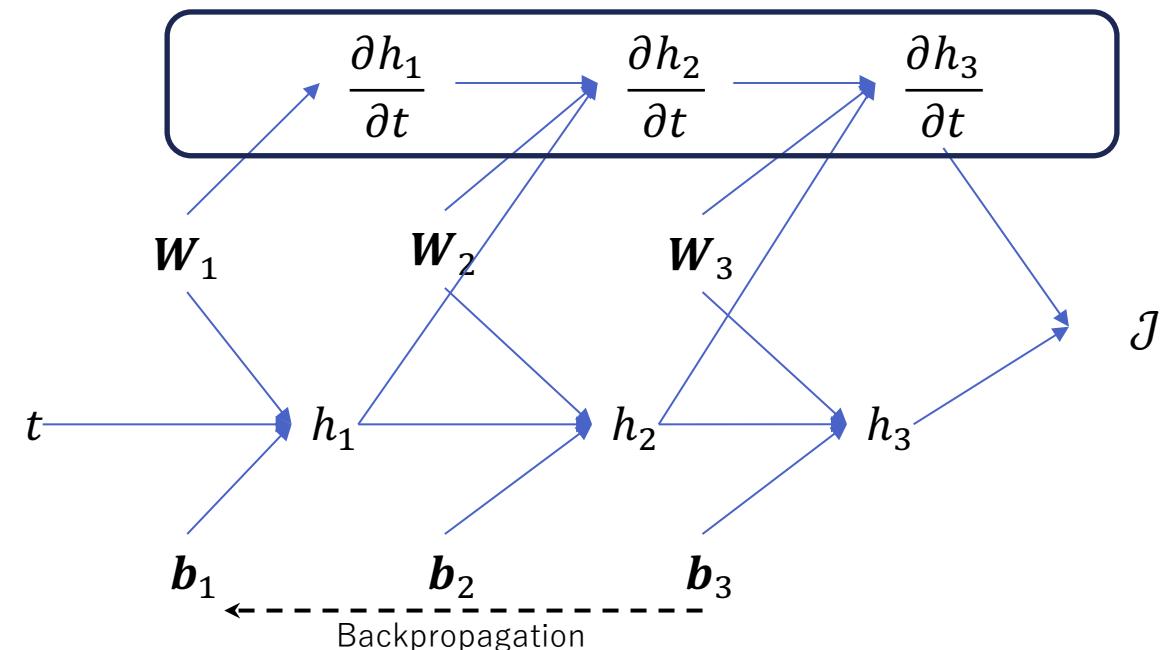
$$h_3 = \mathbf{W}_3\sigma(h_2) + \mathbf{b}_3$$

$$h_2 = \mathbf{W}_2\sigma(h_1) + \mathbf{b}_2$$

$$h_1 = \mathbf{W}_1t + \mathbf{b}_1$$

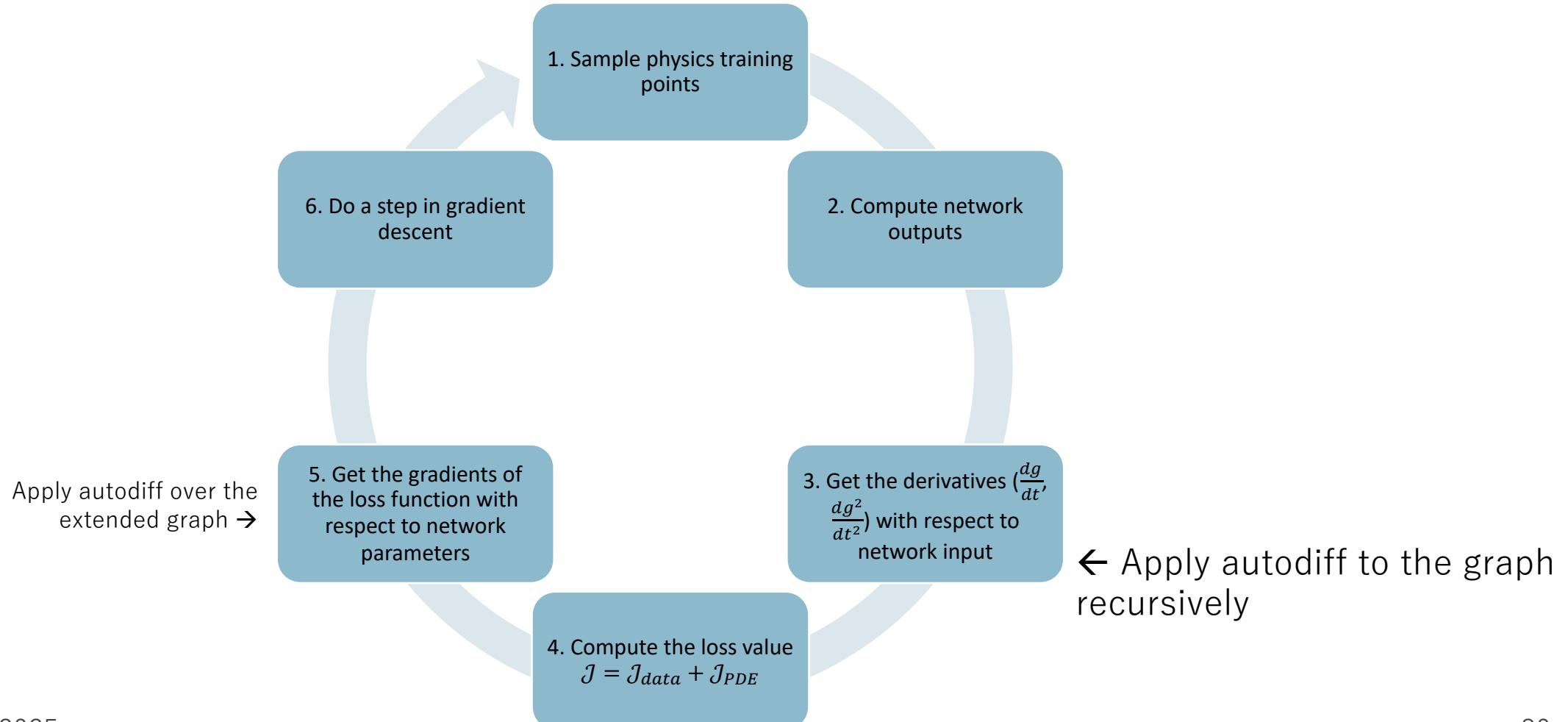
$$\frac{\partial g}{\partial t} = \mathbf{W}_3 \text{diag}(\sigma'(h_2)) \mathbf{W}_2 \text{diag}(\sigma'(h_1)) \mathbf{W}_1$$

Extend graph using autodiff

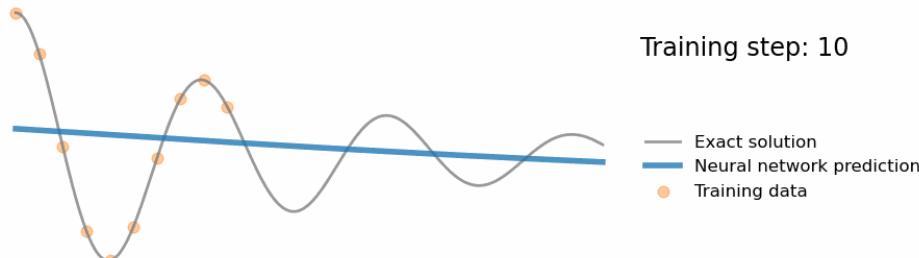


Training of PINN

- We can recursively apply automatic differentiation to get the higher order gradients!

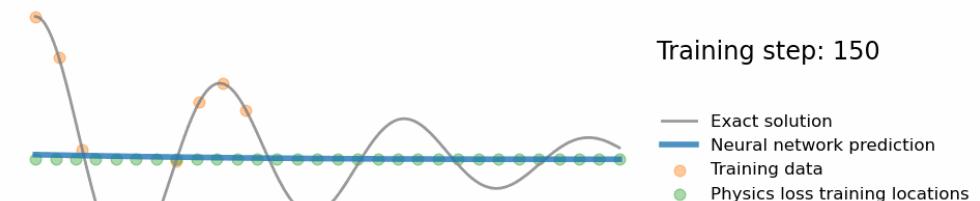


Training of PINN



Supervised loss

$$\mathcal{J}(\theta) = \sum_i^I |y_i - g(x_i; \theta)|^2$$



Moseley, B. So, what is a physics-informed neural network? www.benmoseley.blog

Supervised loss

$$\begin{aligned} \mathcal{J}(\theta) = & \sum_i^I |y_i - g(x_i; \theta)|^2 + \\ & + \frac{1}{N_p} \sum_i^{N_p} \left(\left[\frac{1}{c^2} \frac{d^2}{dt^2} + \nabla_r^2 \right] g(x; \theta) \right)^2 \end{aligned}$$

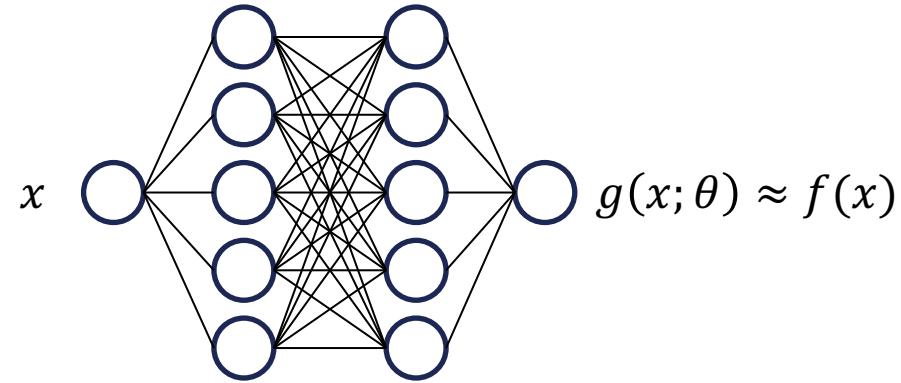
Regularization effect of PDE loss

PINN for sound field estimation

PINNs are at the base of different sound field estimation works

- RIR Reconstruction using PI-SIREN
 - [Karakonstantis+ 2023, Pezzoli+ 2023, Karakonstantis+ 2024] First works using PI-SIREN
- Spherical microphones
 - [Chen+ 2023, Ma+ 2024, Miotello 2024+] Variation in the architectures, frequency domain
- Nearfield acoustic holography
 - [Olivieri+ 2021] Physics loss with Kirchhoff-Helmholtz integral
- Sound field simulation
 - [Borrel-Jensen+ 2024] Solving forward problem using DeepONet

PINN for sound field estimation



Questions?

https://github.com/Chutlhu/EUSIPCO25_PIML_tutorial

Link to repo →



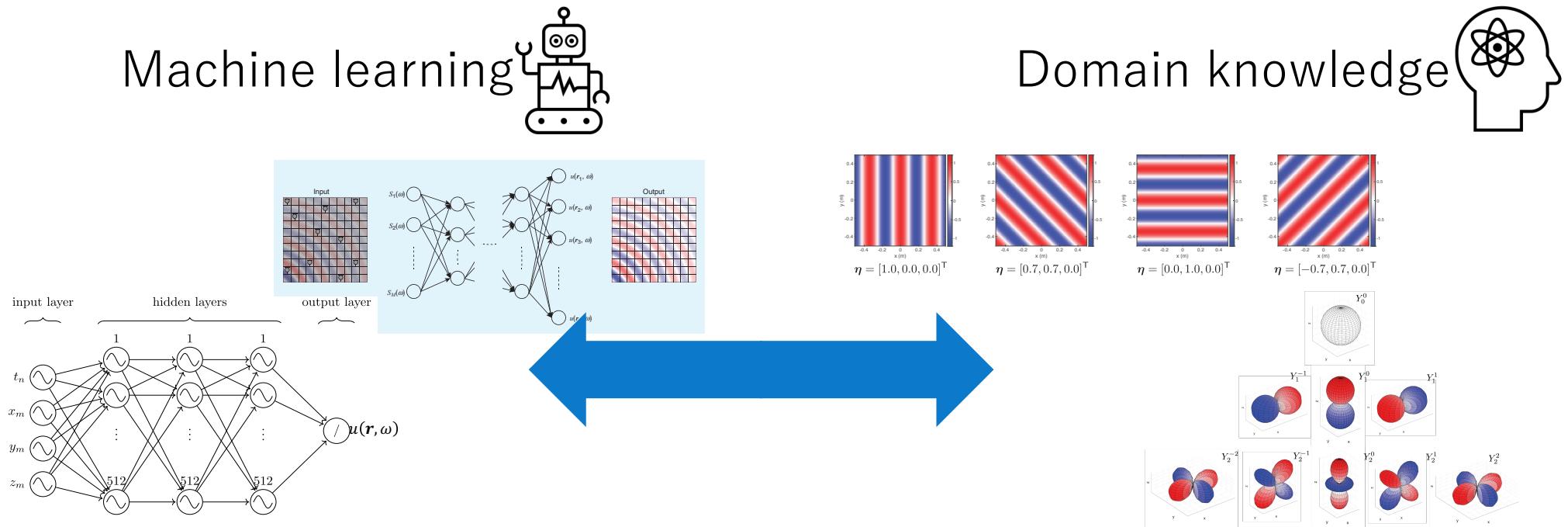
Part III – Practical session of PINNs for audio processing

PHYSICS-INFORMED MACHINE LEARNING FOR AUDIO PROCESSING

Conclusion

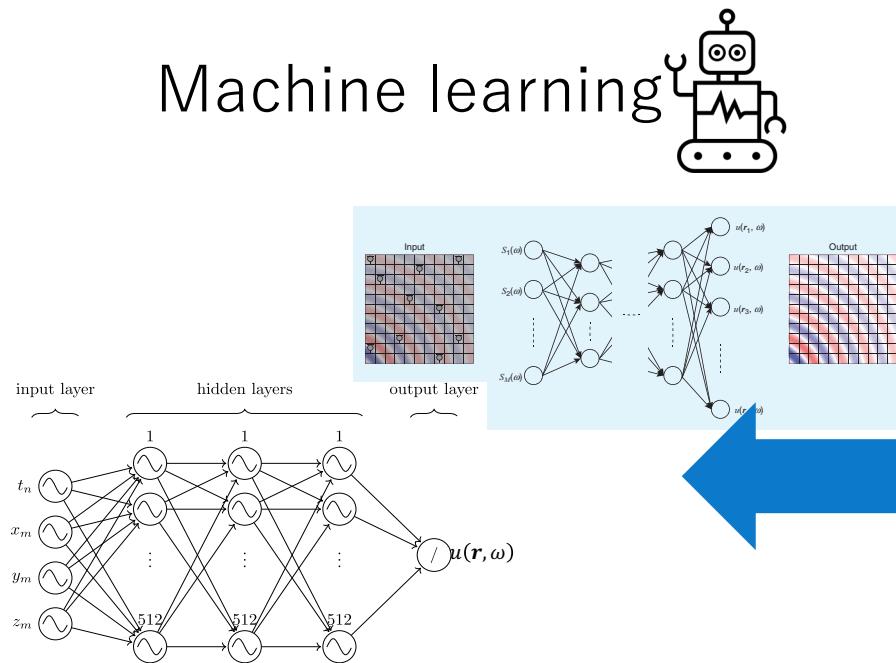
PHYSICS-INFORMED MACHINE LEARNING FOR AUDIO PROCESSING

- Bridging ML with physical prior proved to be a **winning** approach
- Help with common **problems**
 - Generalization
 - Interpretability
 - Data **scarcity**

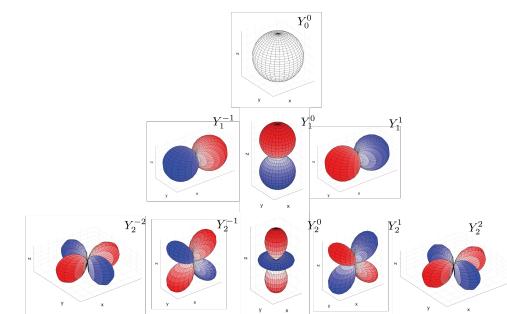
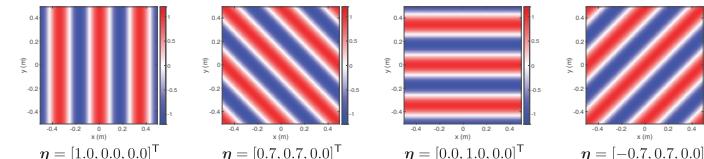


- Bridging ML with physical prior proved to be a **winning** approach
- Help with common **problems**
 - Generalization

Especially for sound field estimation and related tasks



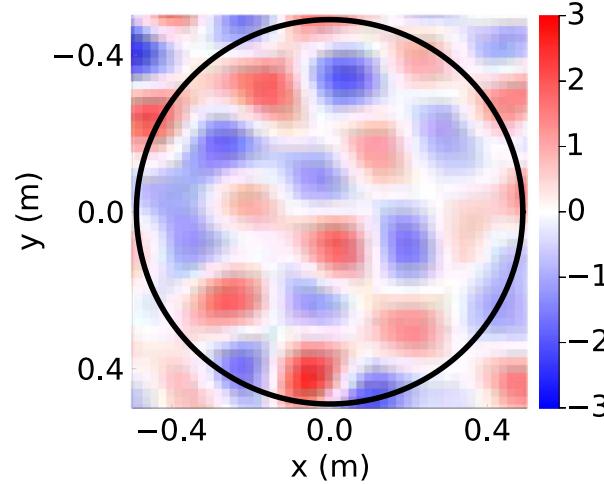
Domain knowledge



We have seen two strategies for including **acoustics priors** into ML

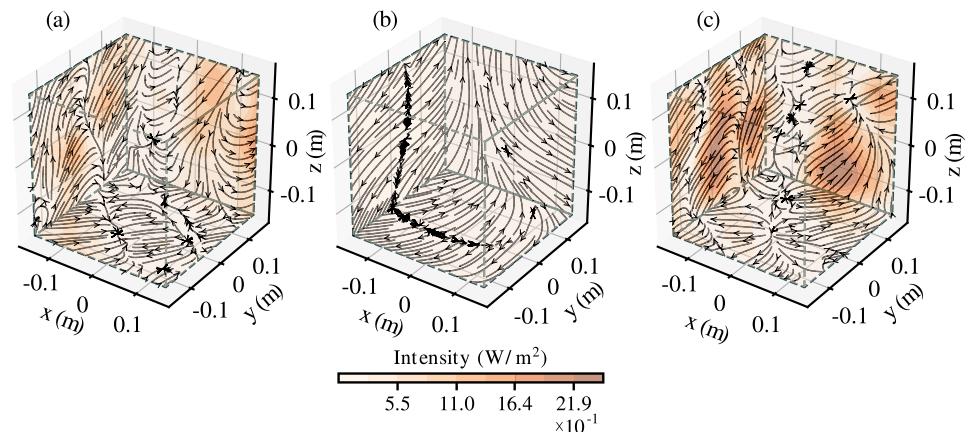
➤ **Constrained** models

- Basis expansions
- KRR



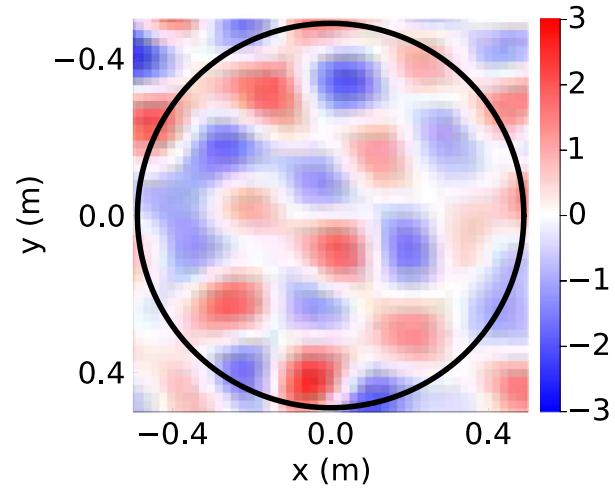
➤ **Informed** (penalized) models

- Feedforward NNs with PDEs
- PINNs



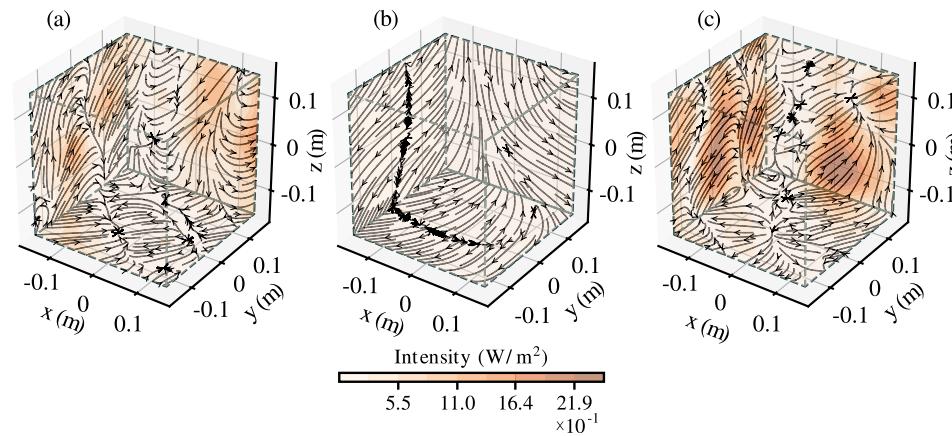
Why choose **constrained models**?

- **Simpler** and **interpretable** models
- Great **accuracy** (if **assumptions** are matched)
- Most of them are **linear** estimators



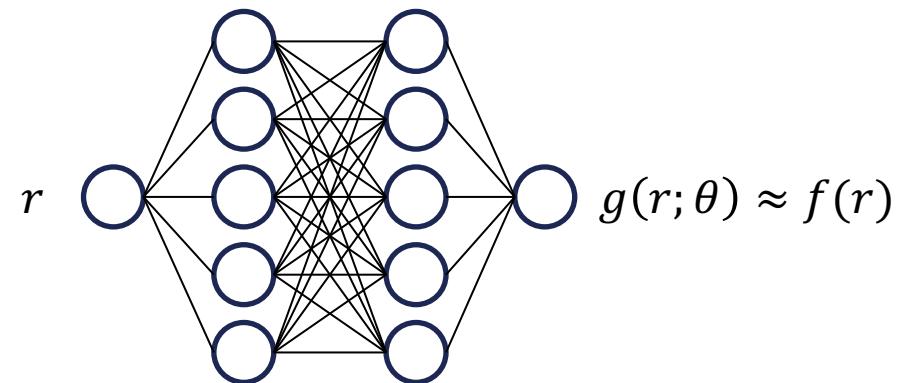
Why choose **informed** models?

- Great **accuracy** (even if **assumptions are not matched**)
- More **flexibility**
- **Lightweight** models



Why choose PINNs?

- Can deal with both **forward** and **inverse** problems
- Ideally “**discretization-free**” model
- **Small training set (single-shot training)**

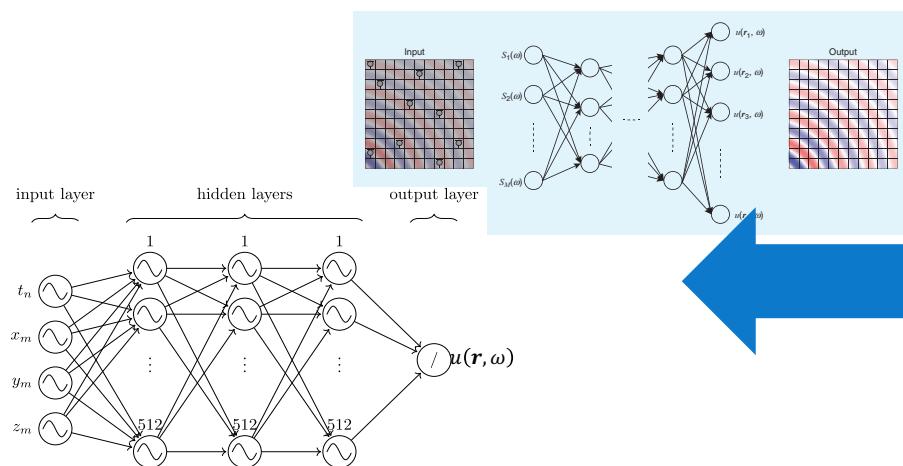


- Bridging ML with physical prior proved to be a **winning** approach
- Help with common **problems**
 - Generalization

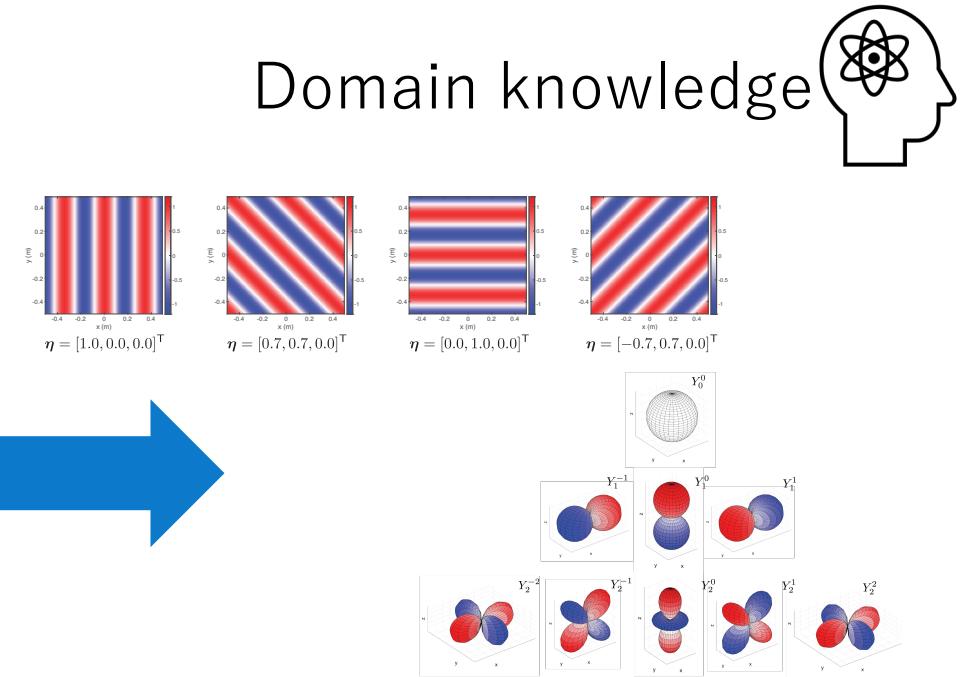
However, there are still major problems to be addressed

Especially for sound field estimation and similar tasks

Machine learning



Domain knowledge



PIML sound field estimation: open challenges

We identified three main **open challenges**:



Preparation of training data



Mismatch between training and test data



Neural network architecture design

PIML sound field estimation: open challenges

Preparation of training data

➤ **Supervised methods** potentially extract more information from data

However,

High spatiotemporal resolution is required

➤ Large **acoustic variations** in different environments

➡ **Simulations** could be used but they have **high computational cost**

PIML sound field estimation: open challenges

Mismatch between **training** and **test** data

- Many **parameters** influence the acoustics
 - Source-receiver location
 - Environment geometry Cover extended ranges is **unpractical**
- **Deviation** between **simulations** and **real world**
 - Wave phenomena
 - Nonlinearities
 - Noise

PIML sound field estimation: open challenges

Neural network **architecture** design

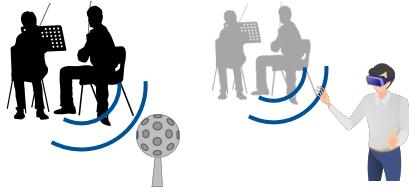
- **Not always** there is a clear methodology for architecture design
- **Unsupervised** methods mainly **MLPs**
 - Number of layers
 - Activations

→ SIREN for spectral bias
Loss term balancing strategies
- **Supervised** methods
 - CNN
 - Generative models
- Application-dependent solutions

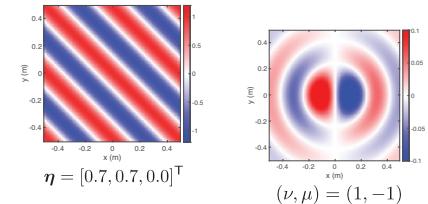
PIML sound field estimation: further observations

- Application of PIML for **interior/mixed** sound fields
 - Some techniques could be applied here
- Dependency on **number** and **distribution** of mics
 - Cover large areas with smallest number of microphones
 - Optimal placement is unclear
- Computational cost
 - Affects several applications e.g., noise control or HRTF interpolation
 - NN-based are mainly offline

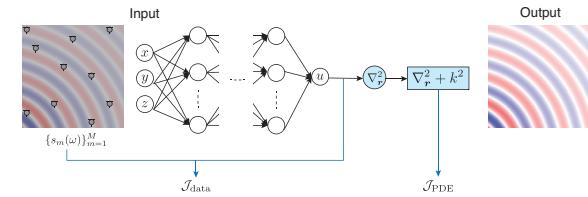
Physics-informed Machine Learning for Audio Processing



Introduce PIML in the context of audio processing



State of the art in sound field estimation based on PIML



Practical implementation of PIML algorithms

➤ List of useful resources

- S. Koyama, J. G. Ribeiro, T. Nakamura, N. Ueno, and M. Pezzoli, “Physics-informed machine learning for sound field estimation: Fundamentals, state of the art, and challenges [special issue on model-based and data-driven audio signal processing],” vol. 41, no. 6, pp. 60–71, 2025.
- S. Koyama and M. Pezzoli “Physics-Informed Machine Learning in Sound Field Estimation: Fundamentals, State of the Art, and Challenges” IEEE SPS SLTC/AASP WEBINAR
- B. Moseley and S. Mishra “AI in the Sciences and Engineering” ETH course ([online](#))
- Nascimento, R. G., Fricke, K., & Viana, F. A. (2020). A tutorial on solving ordinary differential equations using Python and hybrid physics-informed neural network. *Engineering Applications of Artificial Intelligence*, 96, 103996.
- Wang, S., Sankaran, S., Wang, H., & Perdikaris, P. (2023). An expert's guide to training physics-informed neural networks. arXiv preprint arXiv:2308.08468.
- V. Sitzmann SIREN repository <https://github.com/vsitzmann/siren>
- Dagrada, Dario. "Introduction to Physics-informed Neural Networks" <https://github.com/madagra/basic-pinn>

Physics-Informed Machine Learning For Audio Processing

Thanks for attending!

Mirco Pezzoli¹, Diego Di Carlo², Shoichi Koyama³

¹Politecnico di Milano, Milan, Italy

²RIKEN, Kyoto, Japan

³National Institute of Informatics, Tokyo, Japan



POLITECNICO
MILANO 1863



NII S.Koyama's Lab
Audio Processing Research Group

IWAENC 2026



The 19th International Workshop on Acoustic Signal Enhancement will be held in Cremona Italy 



**POLITECNICO
MILANO 1863**

