Source-aware soundfield estimation based on Physics-constrained Kernel interpolation

Mattia Marella

27 novembre 2025

1 Introduction

Sound-field estimation aims at reconstructing the complex acoustic pressure $p(\mathbf{x}, \omega)$ in a region of space from a limited number of microphone measurements. This problem arises in spatial audio, sound-zone control, active noise control, and acoustic holography. When only few microphones are available, classical interpolation methods often suffer from spatial aliasing, poor generalization, and strong dependence on microphone placement.

Physics-informed kernel methods provide a powerful alternative: instead of learning an arbitrary regression function, the estimator is constrained to satisfy (at least approximately) the Helmholtz equation, which governs monochromatic acoustic wave propagation:

$$(\nabla^2 + k^2)p(\mathbf{x}) = 0, \quad k = \frac{\omega}{c}.$$

Here, k is the wavenumber, ω is the angular frequency, and c is the speed of sound. By incorporating this physical knowledge into the kernel design, these methods can achieve better accuracy and robustness with fewer measurements.

Koyama et al. introduced *Physics-Constrained Neural Kernels* (PCNK), where the spatial kernel consists of a superposition of plane waves whose weights are produced by a neural network, but the kernel itself still satisfies the Helmholtz PDE. This hybrid approach combines (i) the universal approximation capability of neural networks, and (ii) strong physics priors that regularize learning and reduce overfitting.

The work considered here extends this framework by making the kernel source-aware, meaning that the position of the sound source is explicitly incorporated into the kernel structure. The idea is that acoustic fields are not arbitrary solutions of the Helmholtz equation: they originate from actual sources, and their spatial characteristics (directivity, decay, wavefront orientation) depend on the relative geometry between source and receivers. So we want the kernel functions to take advantage of this additional information, in order to obtain superior generalization power.

2 Problem Statement and Preliminaries

This section introduces the notation, geometric assumptions, and mathematical background used throughout the kernel formulations.

We consider a three-dimensional region $\Omega \subset \mathbb{R}^3$ in which the acoustic field is assumed to be *source-free.* Suppose also that there is an unknown number of sources outside of Ω generating sound waves that combine into a pressure distribution denoted as $u(\mathbf{x}, k)$ For a fixed angular frequency $\omega = 2\pi f$, the complex acoustic pressure field $u(\mathbf{x}, k) \in \mathbb{C}$ satisfies the homogeneous Helmholtz equation

$$\nabla^2 u(\mathbf{x}, k) + k^2 u(\mathbf{x}, k) = 0, \qquad \forall x \in \Omega, \tag{1}$$

where $k = \omega/c$ is the wavenumber and c is the speed of sound.

Our objective is to estimate the sound field $u(\mathbf{x}, k)$ at any position $x \in \Omega$ from a *snapshot* consisting of M microphone measurements

$$\{s_m(k)\}_{m=1}^{M}, \qquad s_m(k) = u(\mathbf{x}_m, k),$$

measured at known microphone positions $\{\mathbf{x}_m\}_{m=1}^{M} \subset \Omega$. An illustration of this problem setting is given in Fig. 1 of the original article.

For notational simplicity, we omit k from the arguments whenever no ambiguity arises.

2.1 Estimation Objective

Following the standard formulation in kernel-based sound field estimation, we seek an estimate $u \colon \Omega \to \mathbb{C}$ that minimizes the regularized empirical loss

$$L(u) := \sum_{m=1}^{M} |s_m - u(\mathbf{x}_m)|^2 \; + \; \lambda \|u\|_H^2, \tag{2}$$

where $\lambda > 0$ is a regularization constant, and $\|\cdot\|_H$ is the norm associated with a functional space H to be defined. The estimator is therefore the unique minimizer

$$\hat{u} = \arg\min_{u \in H} L(u). \tag{3}$$

The choice of the space H is crucial: it must encode the physical constraint (1), possess a well-defined norm, and allow tractable computation of $\hat{u}$. Moreover, we can guarantee a unique solution with a closed form if H is a *reproducing kernel Hilbert space* (RKHS).

2.2 Representation of Helmholtz Solutions

To ensure physical admissibility, the authors construct H using the classical representation of Helmholtz solutions as *Herglotz wave functions.* Any solution u of (1) in a source-free region can be written as

$$u(\mathbf{x}) = T(\tilde{u}) := \int_{S^2} e^{ik\eta \cdot \vec{x}} \tilde{u}(\eta) \, d\eta, \tag{4}$$

where S^2 denotes the unit sphere, $\eta \in S^2$ is a directional variable, and $\tilde{u}$ is a square-integrable function over the sphere (the *directional representation* of

u). This integral representation ensures that u always satisfies the Helmholtz equation, since $(\nabla^2 + k^2)e^{ik\eta\cdot\vec{x}} = 0$ for every η.

The authors introduce a *weight function* $w\colon S^2 \to \mathbb{R}_{>0}$ and require

$$\int_{S^2} \frac{|\tilde{u}(\eta)|^2}{w(\eta)}\, d\eta < \infty. \tag{5}$$

In this case we write $\tilde{u} \in L^2(S^2, w)$. The weight w plays an essential role in shaping the directional characteristics of the field and will later become the object to be *adapted* from data.

Using the mapping T in (4), the functional space is defined as

$$H := \left\{ u = T(\tilde{u}) \;\middle|\; \tilde{u} \in L^2(S^2, w) \right\}. \tag{6}$$

2.3 Inner Product and Structure of the Space

Since $L^2(S^2, w)$ is a Hilbert space, the induced inner product on H is

$$\langle u_1, u_2 \rangle_H = 4\pi \int_{S^2} \frac{\tilde{u}_1(\eta)^* \tilde{u}_2(\eta)}{w(\eta)}\, d\eta. \tag{7}$$

The completeness of H under this inner product follows from the proportionality between the norms of $T(\tilde{u})$ and $\tilde{u}$, proven in Appendix B of the original paper.

Importantly, for any continuous weight function w, the space H is guaranteed to be a reproducing kernel Hilbert space (RKHS), which implies: 1) the minimizer (3) exists and is unique, and 2) the solution can be written via a kernel of the form $\kappa(\cdot, \mathbf{x}_m)$.

The RKHS machinery itself is not detailed here, but its consequence is the kernel ridge regression form of the solution.

2.4 Kernel Ridge Regression Form

Once the functional space H is chosen as in the previous subsection, the minimizer of (2) must belong to H and must satisfy the Helmholtz equation by construction. To guarantee the existence of a unique minimizer of (3) and to obtain it in closed form, we require H to be a reproducing kernel Hilbert space (RKHS). This means that there exists a *reproducing kernel* $\kappa : \Omega \times \Omega \to \mathbb{C}$ such that

$$\langle \kappa(\cdot, \mathbf{x}), u \rangle_H = u(\mathbf{x}), \qquad \forall\, \mathbf{x} \in \Omega,\ \forall\, u \in H. \tag{8}$$

Under this assumption, the classical representer theorem ensures that the solution of (3) is a finite linear combination of kernel sections evaluated at the microphone positions. Hence, the estimated sound field $\hat{u}$ can be written as

$$\hat{u}(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m\, \kappa(\mathbf{x}, \mathbf{x}_m) = \kappa(\mathbf{x})\, \alpha, \tag{9}$$

where the vector of coefficients $\alpha = [\alpha_1, \ldots, \alpha_M]^\top$ depends solely on the microphone measurements. Here we use the shorthand

$$\kappa(\mathbf{x}) := \big[\kappa(\mathbf{x}, \mathbf{x}_1), \ldots, \kappa(\mathbf{x}, \mathbf{x}_M)\big].$$

Substituting this form into the loss function (2) yields the standard kernel ridge regression (KRR) linear system. We define the Gram matrix

$$\mathbf{K} \in \mathbb{C}^{M \times M}, \qquad K_{mn} = \kappa(\mathbf{x}_m, \mathbf{x}_n), \tag{10}$$

and the measurement vector

$$s = [s_1, \ldots, s_M]^\top.$$

Then the coefficients α minimizing (2) are given by

$$\alpha = (\mathbf{K} + \lambda I)^{-1} s. \tag{11}$$

Combining (9) and (11), the resulting KRR estimate is

$$\hat{u}(\mathbf{x}) = \kappa(\mathbf{x})\,(\mathbf{K} + \lambda I)^{-1} s. \tag{12}$$

Physically, the kernel κ encodes the directional and structural properties of all admissible sound fields in the space H, while the Gram matrix $\mathbf{K}$ expresses the pairwise similarity of microphone observations under that model. Thus, once κ is chosen (i.e., equivalently, once the weight function w determining the kernel is set) sound-field reconstruction reduces to solving a single linear system (11). In the remainder of this manuscript, the focus is on designing and comparing different physically motivated kernels κ and investigating how well each one models the spatial structure of the true sound field.

3 Related Work

Research on sound-field reconstruction from sparse measurements has produced a variety of models and kernel constructions, many of which exploit the structure of solutions to the Helmholtz equation or are based on physically motivated parametrizations.

3.1 Plane-Wave Representations

A classical representation of Helmholtz solutions is the plane-wave (Herglotz) expansion

$$u(x) = \int_{S^2} a(\eta)\, e^{ik\,\eta\cdot x}\, d\eta, \tag{13}$$

where $a(\eta)$ is a directional amplitude function. Many spatial audio and field-interpolation methods approximate (13) using a finite set of directions $\{\eta_d\}_{d=1}^{N}$ and quadrature weights $\{\sigma_d\}_{d=1}^{N}$, leading to

$$u(x) \approx \sum_{d=1}^{N} \sigma_d\, e^{ik\,\eta_d\cdot x}. \tag{14}$$

Such discrete plane-wave models form the basis of uniform kernels, neural plane-wave kernels, and residual models used in data-driven acoustics.

3.2 Isotropic and Bessel-Type Kernels

Integrating plane waves uniformly over the sphere yields an isotropic kernel that depends only on the distance $\|x - x'\|$:

$$\int_{S^2} e^{ik\,\eta\cdot(x-x')} d\eta = 4\pi\, j_0(k\|x - x'\|), \tag{15}$$

where j_0 is the spherical Bessel function of order zero,

$$j_0(t) = \frac{\sin t}{t}.$$

Related formulations employ the Bessel function of the first kind J_0, and in anisotropic settings the modified Bessel function I_0. These kernels are widely used in wave-field synthesis, near-field holography, and spatial filtering, as they provide rotationally symmetric reproducing kernels consistent with the Helmholtz equation.

3.3 Directional and Exponential Kernels

To extend isotropic models and represent anisotropic propagation, several works introduce *directionally biased* kernels obtained by modifying the uniform (isotropic) plane-wave distribution with an exponential weight. Starting from the uniform weight $w(\eta) = 1$, one introduces the directional emphasis

$$w(\eta) \;\propto\; \exp\bigl(\beta\,\eta \cdot d\bigr), \tag{16}$$

where $d \in S^2$ is a preferred direction and $\beta > 0$ controls how sharply the weight concentrates around d:

- when $\beta = 0$, $w(\eta) = 1$ and the kernel reduces to the uniform isotropic model;
- as β increases, directions η aligned with d are exponentially amplified, while directions opposite to d are attenuated.

Thus, (16) acts as a tunable "directional spotlight" on the sphere of propagation directions. This idea naturally fits the physics of point-source radiation, where the wavefield exhibits stronger coherence along outward-propagating directions.

When this weighted distribution is substituted into the Herglotz representation, the resulting kernel becomes a *directionally biased* variant of the isotropic Bessel-type kernel. A representative form arising in the literature is

$$\kappa(x, x') = J_0\bigl(\bigl\|k(x - x') - i\beta\, d\bigr\|\bigr)\,,$$

which can be interpreted as a complex-shifted Bessel function whose argument is displaced along the preferred direction d. This complex displacement encodes the exponential weighting (16) and produces a kernel whose correlation is strongest along d and decreases for directions orthogonal or opposite to it.

Directional exponential kernels of this form have been widely used in sound-field control, directional beamforming, and propagation modeling in both free-field and room-acoustic scenarios, as they offer a principled mechanism for emphasizing a specific propagation direction while retaining the Helmholtz-compliant structure of Bessel-based kernels.

3.4 Learning-Based and Adaptive Kernels

Recent approaches propose to learn directional weights directly from data, often by modeling $a(\eta)$ or $w(\eta)$ with a neural network. In these methods, the kernel takes the form

$$\kappa(x, x') = \sum_{d=1}^{N} \sigma_d \, W(k\eta_d) e^{ik\,\eta_d \cdot (x-x')},$$

where W is a trainable positive function (typically an MLP). This strategy enables data-driven adaptation of directional patterns while preserving the Helmholtz structure of the kernel, and has shown promising performance for acoustic field reconstruction, especially in sparse or irregular microphone configurations.

3.5 Relation to the Present Work

The approaches above motivate the kernel families evaluated in this paper:

- *uniform kernels* based on isotropic spherical Bessel functions,
- *directed kernels* obtained from exponential directional weights,
- *neural plane-wave kernels* that learn anisotropic weights from data,
- and *hybrid (directed + neural) kernels* combining strong analytic directionality with learned residual components.

All these kernels remain consistent with the Helmholtz equation through their plane-wave or Bessel-based construction, and their comparative performance in sound-field estimation is the primary focus of this study.

4 Evaluation Metric

To quantitatively assess the accuracy of the reconstructed sound field, we employ the *Normalized Mean Squared Error* (NMSE), a widely used metric in acoustic field estimation, spatial audio, and inverse problems. NMSE measures the relative discrepancy between the reconstructed complex pressure and the ground-truth reference field.

4.1 Definition

Let $u_{\text{ref}}(x)$ denote the reference sound field and $\hat{u}(x)$ the estimated field, both evaluated at a set of validation positions $\{x_m^{(v)}\}_{m=1}^{M_v}$. The NMSE is defined as

$$\text{NMSE} = \frac{\sum_{m=1}^{M_v} \left| u_{\text{ref}}(x_m^{(v)}) - \hat{u}(x_m^{(v)}) \right|^2}{\sum_{m=1}^{M_v} \left| u_{\text{ref}}(x_m^{(v)}) \right|^2 + \epsilon}, \tag{17}$$

where $\epsilon > 0$ is a small constant included only to avoid division by zero in degenerate cases. The numerator corresponds to the reconstruction error, while the denominator normalizes the value by the total energy of the true sound field.

For reporting convenience, NMSE values are often expressed in decibels:

$$\text{NMSE}_{\text{dB}} = 10 \log_{10}(\text{NMSE}). \tag{18}$$

4.2 Why NMSE for Acoustic Pressure Fields?

NMSE is particularly well suited for evaluating complex acoustic pressure fields for several reasons:

1) Scale invariance. The denominator normalizes by the energy of the reference field. This ensures that the metric is unaffected by the absolute amplitude of the sound pressure, which may vary with frequency, microphone configuration, or source–receiver distance. Thus, NMSE allows fair comparison across frequencies and across distinct kernels.

2) Physically meaningful for complex-valued fields. Acoustic pressures are complex quantities, encoding both magnitude and phase. The squared modulus $|u_{\text{ref}}(x) - \hat{u}(x)|^2$ captures phase errors and amplitude errors simultaneously, and therefore quantifies the discrepancy in the physically relevant quantity.

3) Robustness across heterogeneous validation geometries. In many sound-field estimation tasks, validation microphones are placed in regions with different energy levels (e.g. inside vs. outside measurement surfaces). By normalizing by $\|u_{\text{ref}}\|^2$, NMSE remains comparable across evaluation regions with different field magnitudes.

Moreover, NMSE has been adopted extensively in the spatial acoustics literature, including spherical harmonics interpolation, Herglotz-based reconstructions, plane-wave decomposition methods, and prior kernel-based approaches. Using NMSE thus allows direct comparison with related methods.

4.3 Interpretation

An NMSE of 0 corresponds to perfect reconstruction, while larger values indicate poorer accuracy. Expressed in decibels, negative NMSE values (e.g. -15 dB) indicate that the error power is much smaller than the signal power, whereas values close to 0 dB suggest errors comparable to the signal itself.

Since NMSE aggregates errors over all validation points, it provides a global measure of reconstruction quality, sensitive to global shape accuracy of the wavefront.

For these reasons, NMSE serves as a principled and physically meaningful metric for comparing the uniform, directed, neural, and hybrid kernels investigated in this work.

5 Initial Experimental Setup

This section describes the numerical setup used to compare the kernel constructions introduced in the previous sections. All experiments are performed independently for each frequency. The aim is to evaluate how accurately each kernel reconstructs the acoustic field throughout the validation region, given sparse pressure measurements on a microphone array.

5.1 Measurement Geometry

We consider a source located at a fixed position $x_s \in \mathbb{R}^3$. Microphone measurements are collected at M receiver positions

$$X = \{x_1, \dots, x_M\} \subset \Omega,$$

forming the *training set*. These microphones lie on a spherical measurement surface (see 1). For evaluation, an additional set of M_v positions

$$X_v = \{x_1^{(v)}, \dots, x_{M_v}^{(v)}\}$$

is placed inside the spherical surface, forming the *validation set*. This two-region configuration assesses the ability of each kernel to extrapolate the field into an interior region not covered by the measurements.

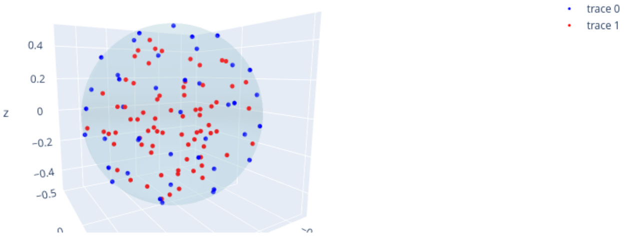

Figura 1: Illustration of the microphone array geometry used in experiments.

When available, a dense Cartesian grid $\{x_g\}_{g=1}^{G}$ is used for visualization of the reconstructed magnitude field and for qualitative inspection of spatial artifacts.

5.2 Frequencies and Data

Experiments are performed at discrete frequencies

$$\{f_1, \dots, f_F\}, \qquad k_f = \frac{2\pi f}{c}$$

For each frequency f, the complex pressure measurements

$$s_m(f) = u(x_m, k_f)$$

are obtained from the dataset contained in the provided HDF5 file, which includes:

- the discrete frequencies array,
- the training microphone positions X,
- the validation microphone positions X_v,

- the training pressures on X,
- the validation pressures on X_v,
- (optionally) the ground-truth field on a dense grid.

To match the measurement conditions described in the reference, additive Gaussian noise is injected into the training data at a prescribed signal-to-noise ratio (SNR). The validation and grid data remain noise-free.

5.3 Kernel Constructions

Four kernel families are evaluated:

1) Uniform kernel. Isotropic, derived from the spherical Bessel function $j_0(k\|x - x'\|)$, corresponding to uniform weighting on the sphere.

2) Directed kernel. Source-aware kernel with exponential directional weighting, leading to complex-argument Bessel functions of the form $J_0(\|k(x - x') - i\beta d\|)$.

3) Neural plane-wave kernel. Data-driven kernel whose weights are modulated by a neural network $W(k\eta_d)$ acting on plane-wave directions.

4) Proposed hybrid kernel. Sum of the directed analytical component and a neural residual plane-wave component.

Each kernel is instantiated independently at each frequency using the corresponding wavenumber k_f.

5.4 Training Procedure and Optimization Methods

For each frequency, the parameters of the kernel are optimized by minimizing the leave-one-out (LOO) objective introduced in the original formulation. The LOO loss evaluates the predictive performance obtained when each training microphone is virtually removed in turn, thereby providing a robust, data-driven criterion for tuning the kernel parameters without requiring an explicit validation set during training.

Because different kernels contain different types of parameters and structural constraints, two classes of optimization algorithms are employed.

LBFGS for unconstrained neural parameters. The uniform kernel contains no trainable parameters, while the PCNK kernel includes only neural plane-wave weights, which are free real-valued parameters apart from a simple non-negativity condition. For such smooth, unconstrained problems, a quasi-Newton method such as LBFGS is highly effective. LBFGS estimates curvature information from past gradients, resulting in rapid convergence even for moderately sized neural parametrizations. This choice mirrors the original SPM implementation, in which LBFGS is used whenever the kernel parameters live in an unconstrained Euclidean space.

IP-based Newton methods for constrained analytical parameters. The directed kernel and the proposed hybrid kernel contain analytical directional weights that must satisfy specific structural constraints, such as non-negativity and a simplex condition (the weights must sum to one). These constraints arise naturally from the underlying physical model: the analytical part of the kernel represents a mixture of directional components, and therefore must behave as a normalized distribution over directions.

To handle these constraints consistently, the training of these kernels uses an interior-point Newton method (IPNewton). This method enforces equality and inequality constraints through barrier functions and solves a sequence of Newton systems that remain within the feasible region. It is particularly well suited for problems where a subset of parameters must stay strictly positive or lie on a simplex, and where curvature information is essential for stability.

Rationale for the two-optimizer strategy. The combination of LBFGS and IPNewton reflects the heterogeneous nature of the kernel families:

- Neural residual components are flexible, unconstrained, and smooth: $\Rightarrow$ LBFGS is efficient and reliable.
- Analytical directional components represent probability-like weights with physically meaningful constraints: $\Rightarrow$ constrained Newton methods are required.
- Hybrid kernels inherit both structures, and IPNewton ensures that the analytical part remains valid while still allowing neural terms to adapt freely.

This separation of optimization roles mirrors the structure of the original methodology and ensures that each kernel is trained under the most appropriate numerical strategy for its parameterization. By relying on second-order information (either approximated or exact), both optimizers achieve stable convergence across frequencies, enabling a fair and consistent comparison between the kernel families evaluated in this work.

In all cases, the loss function is the frequency-wise LOO objective:

$$L_{\mathrm{LOO}} = \sum_{m=1}^{M} \left| \frac{\alpha_{f,m}}{[\mathrm{diag}(A_f^{-1})]_m} \right|^2 ,$$

where

$$A_f = K_f + \lambda R_f,$$

with K_f the Gram matrix and R_f the regularizer derived from the directional and/or neural components of the kernel.

Meaning of the regularization terms. The regularization in the original formulation does not arise from an external hyperparameter added to the loss, but is instead a *model-internal* term that emerges naturally from the structure of the kernel. Each kernel contains directional weights, analytical or neural, that determine how strongly individual plane-wave components contribute to the field.

These weights enter the reproducing kernel through the integral representation and give rise to two scalar quantities:

$$\mathrm{reg}_a = \sum_d \sigma_d^{(a)}, \tag{19}$$

$$\mathrm{reg}_n = \sum_d \sigma_d^{(n)}\, W(k\eta_d), \tag{20}$$

where $\sigma_d^{(a)}$ are the analytical (directed) mixture weights and $\sigma_d^{(n)} W(k\eta_d)$ are the neural residual weights. In the original theory, these sums correspond to the squared H-norm of the function in the underlying Herglotz-based RKHS. Thus, they represent the *intrinsic smoothness* or *energy* of the function class modeled by the kernel.

During training, these quantities appear in the expression

$$A_f = K_f + \lambda(\mathrm{reg}_a + \mathrm{reg}_n)I,$$

where the scalar $\lambda > 0$ modulates the influence of the RKHS norm on the estimator. Because reg_a and reg_n depend on the kernel parameters, the regularization term becomes part of the optimization objective, acting as a natural penalty that discourages excessively large weights and enforces physically meaningful directional distributions.

This explains why the same λ is used across all kernels: the term $\mathrm{reg}_a + \mathrm{reg}_n$ already adapts to the structure and parametrization of each kernel family, making the regularization *kernel-aware* rather than manually tuned.

5.5 Field Reconstruction

Once training is complete at frequency f, the reconstructed field is evaluated at the validation positions by

$$\hat{u}_f(x_m^{(v)}) = \sum_{n=1}^{M} \alpha_{f,n}\, \kappa_f(x_m^{(v)}, x_n).$$

When the dataset provides a pointwise reference field on a rectangular grid, the same expression is used to generate 2-D field maps, enabling visual comparison among kernels in terms of spatial smoothness, wavefront shape, and extrapolation behavior.

5.6 Evaluation Metric

Quantitative accuracy is measured using the normalized mean squared error (NMSE) defined in Section 17. For each frequency, NMSE is computed between the ground-truth validation pressures and the reconstructed values. This yields an NMSE curve as a function of frequency for every kernel family.

These curves serve as the primary numerical indicator of reconstruction quality and are used to compare the kernels under identical geometric and measurement conditions.

6 Training Procedures

In this section we describe in detail how the kernels are trained in the original single-source implementation and in the new multi-source, source-aware Python implementation. In both cases, the goal is to adapt the kernel parameters so as to minimize the leave-one-out (LOO) prediction error at the microphone positions, under the Helmholtz-consistent kernel ridge regression framework introduced in the previous sections.

6.1 Single-Source Training (Julia)

In the original code, training is carried out *separately for each frequency* and for each kernel type. The source is **always external** to the measurement region (this is a key assumption since it keeps homogeneous the Helmholtz equation), and a single set of microphones is used for both training and evaluation, with a train/test split. For a fixed frequency f with wavenumber k_f, and a chosen kernel family κ_θ parametrized by θ, the training problem can be written schematically as

$$\theta^\star(f) = \arg\min_\theta L_{\mathrm{LOO}}(\theta; f), \tag{21}$$

where L_{LOO} is the closed-form leave-one-out loss derived from the KRR solution.

Given the Gram matrix $K_f(\theta)$ and the regularization terms $\mathrm{reg}_a(\theta)$ and $\mathrm{reg}_n(\theta)$, the regularized system matrix is constructed as

$$A_f(\theta) = K_f(\theta) + \lambda\big(\mathrm{reg}_a(\theta) + \mathrm{reg}_n(\theta)\big)I,$$

and expresses the LOO loss using the diagonal of $A_f(\theta)^{-1}$ and the KRR coefficients $\alpha_f(\theta)$. This yields a scalar objective $L_{\mathrm{LOO}}(\theta; f)$ that is smooth with respect to θ.

Different kernel families induce different parameterizations and constraints:

- **Uniform kernel.** Has no trainable parameters; L_{LOO} is evaluated only once and serves as a baseline.
- **Neural plane-wave kernel (PCNK).** The parameters θ are the weights and biases of the neural network W that modulates the plane-wave amplitudes. These parameters are unconstrained real numbers (up to the non-negativity enforced by the final activation), so $L_{\mathrm{LOO}}(\theta; f)$ is minimized using an unconstrained quasi-Newton method (LBFGS).
- **Directed and hybrid kernels.** Besides possible neural parameters, these kernels contain analytical directional weights that form a simplex: they must be non-negative and sum to one. As a result, θ lives in a constrained domain. In the original Julia code, these constraints are handled explicitly via an interior-point Newton method (IPNewton), which enforces the simplex constraints while minimizing L_{LOO}.

For each frequency and kernel, the optimizer (LBFGS or IPNewton, depending on the parameterization) is run until convergence of L_{LOO}. Once the optimizer has converged, the resulting parameters $\theta^\star(f)$ define the kernel $\kappa_{\theta^\star(f)}$ for frequency f. These parameters are then used to compute the KRR coefficients based on the *training microphones* only. In the single-source setup, the microphone array is intentionally split into two sets with very different sizes:

- a **small set of training microphones**, placed on a spherical surface. These represent the practically affordable sensing setup: in real applications the number of microphones is limited, so the model must be able to reconstruct the sound field from a sparse set of measurements.

- a **much larger set of test** microphones, located inside the spherical surface. These dense interior measurements are used *only for evaluation* to assess how accurately the trained kernel reconstructs the field in space.

After solving the regularized linear system

$$\alpha_f = \big(K_f(\theta^\star(f)) + \lambda(\mathrm{reg}_a + \mathrm{reg}_n)I\big)^{-1} s_f,$$

where s_f contains the training pressures at frequency f, the estimator

$$\hat{u}_f(x) = \sum_{m=1}^{M_{\mathrm{train}}} \alpha_{f,m}\, \kappa_{\theta^\star(f)}(x, x_m)$$

is evaluated at the *validation microphones.* These microphones are never used during optimization: they serve solely to quantify reconstruction performance.

The reconstruction error is measured via the normalized mean squared error (NMSE), computed between $\hat{u}_f$ and the ground-truth field on the dense validation set.

6.2 Multi-Source Training in the Source-Aware Python Implementation

The new Python implementation extends the original SPM framework in two directions: (i) it introduces *source-aware kernels* whose parameters depend explicitly on the source position; and (ii) it trains these kernels on *multiple source datasets* with an explicit train/validation/test split. This is implemented in the function `run_spm_multisource`, whose logic we now describe in detail.

6.2.1 Source-Level Train/Validation/Test Split

The input directory contains one HDF5 file per source, each file providing: the source position x_s, the receiver positions X and X_v, and the frequency-dependent pressures (train and validation). If there are N sources, the code first generates a random permutation of $\{1, \ldots, N\}$ and assigns the corresponding HDF5 files to three disjoint sets:

- a *training set* of sources (comprising 80% of the files),

- a *validation set* of sources (comprising 10% of the files),

- a *test set* of sources (remaining files).

This means that, unlike the original SPM, the model is encouraged to learn kernel parameters that generalize *across different source positions and room realizations*, not only across microphones for a single source.

6.2.2 Per-Frequency Kernel Instantiation

The frequency axis is read from the first training file. For each frequency f with wavenumber k_f, and for each kernel type, the code:

1. instantiates a fresh kernel object κ_{θ_f} via the provided factory function, with wavenumber set to k_f;
2. initializes a record of the best parameters according to validation loss (early stopping).

Thus, the training remains frequency-wise, but now each kernel at frequency f is trained on *many* sources.

6.2.3 Epoch Loop and Shuffled Source Order

Training proceeds for a fixed number of epochs, before early stopping gets triggered. At each epoch:

1. The list of training source files is randomly shuffled.
2. For each training source in this shuffled order:
   (a) The receiver positions X and complex pressures Y_f at frequency index f are loaded.
   (b) Additive white Gaussian noise with prescribed SNR is added to Y_f to emulate noisy measurements; this is done *only* on the training sources.
   (c) The kernel is informed of the source position x_s.
   (d) A short optimization run is performed on this source: depending on the configuration, either
      - **Adam** is used, with a fixed number of gradient steps, or
      - **LBFGS** is used, where LBFGS is run for a small number of iterations.

      In both cases, the objective being minimized is the same LOO-based loss used in the original formulation, but now evaluated on the current source dataset. The kernel parameters θ_f are updated *in place* after each source, so over one epoch the kernel gradually adapts to all sources in the training set.

This training scheme can be seen as a form of mini-batch optimization where each "batch" consists of all microphones for a single source. The shuffling at each epoch prevents the optimizer from overfitting to a fixed source order and improves robustness.

6.2.4 Validation-Based Early Stopping Across Sources

At the end of each epoch, the kernel is evaluated on the *validation* sources, which were never used for parameter updates. For each validation source:

1. The corresponding X and Y_f are loaded (without noise).

2. The source position is passed to the kernel.
3. The LOO loss is computed using the current kernel parameters, but without performing gradient updates.

The average validation LOO loss over all validation sources is then recorded. If this average loss is lower than the best value observed so far, the current kernel state is saved as the *best* model for that frequency. This implements an early-stopping-like mechanism: even though the number of epochs is fixed, the final model used for testing is the one that performed best on unseen validation sources.

6.2.5 Test-Time Reconstruction and NMSE

After training at frequency f is complete, the kernel parameters are restored to the best validation state. The model is then evaluated on the *test* sources:

1. For each test source, X, X_v, and the corresponding pressures Y_f and Y_v are loaded.
2. The kernel is updated with the source position x_s.
3. The Gram matrix K_f on the training microphones is built, regularized using the current regularization term, and the KRR coefficients α_f are computed by solving the linear system.
4. The field is reconstructed at the test microphones: $\hat{u}_f(x_m^{(v)})$ for all m.
5. The NMSE between $\hat{u}_f$ and u_{ref} on the test microphones is computed for that source.

The NMSE values are then averaged over all test sources to yield a frequency-dependent NMSE curve for each kernel. These curves summarize the ability of each kernel to generalize *across both space and sources.*

6.2.6 Summary of Differences w.r.t. the Original code

Compared to the original single-source training, the new Python implementation introduces:

- a **multi-source** perspective, where a single set of kernel parameters must account for many different source positions;
- an explicit **train/validation/test split at the source level**, enabling validation-based model selection and unbiased test evaluation;
- **epoch-based optimization** with shuffled sources, using Adam or incremental LBFGS, which gradually refines the kernel over many source realizations;
- explicit **source-aware training**, in the sense that the kernel is always informed of the current source position before each optimization or evaluation step.

In this way, the Python framework preserves the Helmholtz-consistent LOO training principle of the original SPM method, while extending it to a more realistic and challenging multi-source scenario.