DeepWave: A Recurrent Neural-Network for Real-Time Acoustic Imaging

Matthieu Simeoni [*]
IBM Zurich Research Laboratory
`meo@zurich.ibm.com`

Sepand Kashani [†]
École Polytechnique Fédérale de Lausanne (EPFL)
`sepand.kashani@epfl.ch`

Paul Hurley
Western Sydney University
`paul.hurley@westernsydney.edu.au`

Martin Vetterli
École Polytechnique Fédérale de Lausanne (EPFL)
`martin.vetterli@epfl.ch`

Abstract

We propose a recurrent neural-network for real-time reconstruction of acoustic camera spherical maps. The network, dubbed DeepWave, is both physically and algorithmically motivated: its recurrent architecture mimics iterative solvers from convex optimisation, and its parsimonious parametrisation is based on the natural structure of acoustic imaging problems. Each network layer applies successive filtering, biasing and activation steps to its input, which can be interpreted as generalised deblurring and sparsification steps. To comply with the irregular geometry of spherical maps, filtering operations are implemented efficiently by means of graph signal processing techniques. Unlike commonly-used imaging network architectures, DeepWave is moreover capable of directly processing the complex-valued raw microphone correlations, learning how to optimally back-project these into a spherical map. We propose moreover a smart physically-inspired initialisation scheme that attains much faster training and higher performance than random initialisation. Our real-data experiments show DeepWave has similar computational speed to the state-of-the-art delay-and-sum imager with vastly superior resolution. While developed primarily for acoustic cameras, DeepWave could easily be adapted to neighbouring signal processing fields, such as radio astronomy, radar and sonar.

1 Introduction

Motivation An *acoustic camera (AC)* [26, 8, 18, 24] is a multi-modal imaging device that allows one to visualise in real-time sound emissions from every *direction* in space. This is typically achieved by overlaying on the live video from an optical camera a heatmap representing the intensity of the ambient directional sound field, recovered from the simultaneous recordings of a *microphone array* [3, 42]. Most commercial acoustic cameras recover the sound intensity field by combining linearly the correlated microphone recordings with a *Delay-And-Sum (DAS) beamformer* [42, Chapter 5]. The beamformer acts as an *angular filter* [20, 21], steering sequentially the array sensitivity pattern –or *beamshape*– towards various directions where the sound intensity field is probed. Acoustic images obtained this way are cheap to compute, but are *blurred* by the beamshape of the microphone array,

[*]Corresponding author. Matthieu Simeoni is also affiliated to the École Polytechnique Fédérale de Lausanne (EPFL), with email address `matthieu.simeoni@epfl.ch`

[†]Matthieu Simeoni and Sepand Kashani have contributed equally to this work. Sepand Kashani was in part supported by the Swiss National Science Foundation grant number 200021 181978/1, “SESAM - Sensing and Sampling: Theory and Algorithms".

Preprint. Under review.

and hence exhibit poor angular resolution [49, 7, 48]. The severity of this blur can be shown [53] to be proportional to the ratio λ/D, where D is the diameter of the microphone array and λ the sound wavelength. Because of the relatively large wavelengths of acoustic waves in the audible range, this blur can be significant in practice: a 30 cm diameter microphone array has an angular resolution at 5 kHz (an $E\flat$) of approximately 10 degrees, against $7{\cdot}10^{-4}$ degrees for a standard optical camera at 790 THz (violet). Moreover, acoustic cameras are often deployed in confined environments [34], requiring them to be as *compact* and *portable* as possible, which limits[3] further the achievable angular resolution.

The advent of *compressed sensing* techniques [14, 44] –and their wide adoption in imaging sciences [54, 4, 32]– have inspired algorithmic solutions [48, 7, 11, 12] to the acoustic imaging problem, promising vastly improved angular resolutions. Unfortunately, these methods proved ill-suited for real-time purposes. Indeed, they often rely on iterative solvers, such as *proximal gradient descent (PGD)* [37] or its accelerated variants [2, 31]. While exhibiting a fast convergence rate [2], such methods still require on the order of a few dozen iterations to converge in practice, making them unable to cope with the high refresh-rate[4] of acoustic cameras. For this reason, and despite their clear superiority in terms of resolving power, nonlinear imaging methods have not yet replaced the suboptimal DAS imager in the software stack of commercial acoustic cameras.
The recent eruption of *deep learning* [33, 56, 10] in the field of imaging sciences may however seal the fate of DAS for good. Indeed, this new imaging paradigm leverages *neural-networks* [28] to reduce dramatically the image formation time. Unlike compressed-sensing methods which proceed iteratively, neural networks *encode* the image reconstruction process in a cascade of linear and nonlinear transformations *trained* on a very large number of input/output example pairs. Once properly trained, a neural-network can be efficiently evaluated for some input data to produce images of high quality, with similar accuracy and resolution as state-of-the-art compressed-sensing methods [33]. Network architectures used for inverse imaging [22, 17, 56, 10, 40] are most often *convolutional neural-networks (CNNs)*, directly adapted from generic architectures developed for *image classification* and *segmentation* [45]. While suitable for image processing tasks such as *denoising*, *super-resolution* or *deblurring* [38, 6], such architectures are ill-suited [33] for more complex image reconstruction problems where the input data may not consist of an image, as is the case in *biomedical imagery* [4, 32], *interferometry* [54] or acoustic imaging. Moreover, and particularly limiting for our current purposes, standard convolutional architectures cannot handle images with non-Euclidean domains [13] such as *spherical* maps [41] produced by omnidirectional acoustic or optical cameras.
To overcome these limitations, *recurrent* architectures [16, 50, 30, 33] have been proposed, by *unrolling* iterative convex optimisation algorithms. Such networks are not only able to handle non-image inputs, but also have greater interpretability than generic CNNs. For example, Gregor and LeCun proposed in their pioneering work [16] a *recurrent neural-network (RNN)* dubbed *LISTA*[5], inspired from the popular *iterative soft-thresholding algorithm (ISTA)*[2].[6] Their network can be seen as generalising ISTA, allowing for the normally fixed gradient and proximal steps occurring at each iteration of the algorithm to be learnt from the data: update steps of ISTA are replaced by a cascade of recurrent layers with trainable parameters. The depth of the resulting RNN is typically much smaller than the number of iterations required for ISTA to converge. Roughly speaking, the network is learning *shortcuts* in the reconstruction space, allowing it to achieve a prescribed reconstruction accuracy faster than gradient-based iterative methods.[7]
While the effectiveness of LISTA was verified on small images from the MNIST dataset (784 pixels) [16], its application to large-scale imaging problems remains challenging. This is mainly due to the huge number of weights parametrising the network which, in the fully-connected case, grows as the number of pixels to the square. Storing[8] –let alone learning– all those weights quickly becomes intractable for increasing resolutions. As a potential fix, Gregor and LeCun recommended sparsifying the network by *pruning* layer connections. While they showed that such a pruning could reduce the number of parameters in the network by 80% without affecting too much the performance of the

[3]Remember that the blur spread is inversely proportional to the microphone array diameter.

[4]An acoustic camera typically updates the acoustic image a dozen times per second.

[5]LISTA stands for learned iterative soft-thresholding algorithm.

[6]ISTA is an instance of proximal gradient descent for *penalised basis pursuit* problems [52].

[7]Of course, such shortcuts will most likely only be valid for the distribution of inputs and outputs implicitly defined by the training set, which should hence be carefully crafted for the network to generalise well in practice.

[8]For a 1 megapixel image, the weights parametrising the network would be approximately 8 Gb in size.

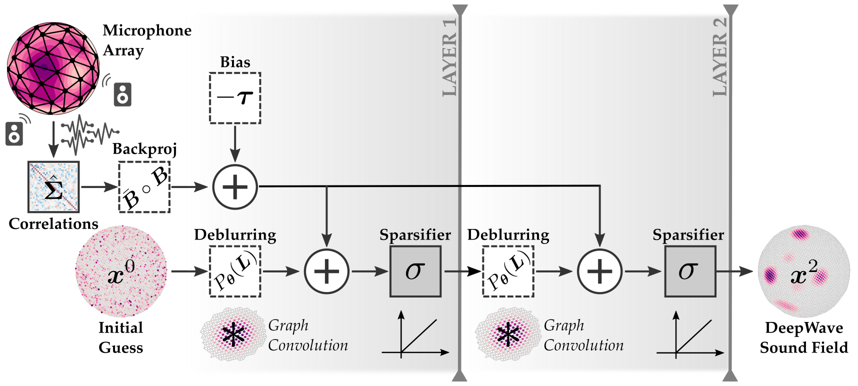


Figure 1: DeepWave's recurrent architecture (1) for $L = 2$ layers and random initialisation. Learnable parameters of the network are denoted by dashed boxes. Affine operations are denoted by white boxes and nonlinear activations by grey boxes.

latter, this is still insufficient for large-scale problems, and additional structure must be considered on network layers. Such structure is however often very dependent on the problem at hand.

Contributions In this work, we propose the first realistic architecture of a LISTA neural-network adapted to acoustic imaging. Our custom architecture, dubbed *DeepWave*, is capable of rendering high-resolution spherical maps of real-life sound intensity fields in milliseconds. DeepWave is tailored to the acoustic imaging problem, leveraging fully its underlying structure so as to minimise the number of network parameters. The latter is easy to train, with a typical training time of less than an hour on a general-purpose CPU. Unlike most state-of-the-art neural-network architectures, it moreover readily supports *complex*-valued input vectors, making it capable of directly processing the raw correlated microphone recordings. Assuming a microphone array with M microphones, the instantaneous covariance matrix $\hat{\mathbf{\Sigma}} \in \mathbb{C}^{M\times M}$ of the microphone recordings is processed by the network as follows (see also fig. 1):

$$\mathbf{x}^l = \sigma\left(P_{\boldsymbol{\theta}}(\mathbf{L})\,\mathbf{x}^{l-1} + \left[\overline{\mathbf{B}} \circ \mathbf{B}\right]^H \mathrm{vec}(\hat{\mathbf{\Sigma}}) - \boldsymbol{\tau}\right), \qquad l = 1,\ldots,L, \tag{1}$$

where $\mathrm{vec} : \mathbb{C}^{M\times M} \to \mathbb{C}^{M^2}$ is the *vectorisation operator* and $\circ$ denotes the *Khatri-Rao product* (see appendix A[9] for definitions). The neurons $\{\mathbf{x}^1, \ldots, \mathbf{x}^L\} \subset \mathbb{R}^N_+$ at the output of each layer l of the depth L neural-network correspond to the acoustic image as it is processed by the network, with N the number of pixels. The neuron $\mathbf{x}^0 \in \mathbb{R}^N_+$ defines the initial state of the network. The nonlinear *activation function*[10] $\sigma : \mathbb{R} \to \mathbb{R}$ induces *sparsity* in the acoustic image, and is inspired by the proximal operator of an elastic-net penalty [37]. The remaining quantities, namely $P_{\boldsymbol{\theta}}(\mathbf{L})$, $\mathbf{B}$ and $\boldsymbol{\tau}$ are trainable parameters of the network, with various roles:

- *Deblurring:* the matrix $P_{\boldsymbol{\theta}}(\mathbf{L}) := \sum_{k=0}^{K} \theta_k \mathbf{L}^k \in \mathbb{R}^{N\times N}$ can be interpreted as a *deblurring matrix*, cleaning potential artefacts from the array beamshape. Following the approach of [41], it is defined as a polynomial of the *graph Laplacian* $\mathbf{L} \in \mathbb{R}^{N\times N}$ based on the *connectivity graph* of the spherical tessellation in use, with learnable coefficients $\boldsymbol{\theta} = [\theta_0, \ldots, \theta_K] \in \mathbb{R}^{K+1}$. Such parametrisation permits notably the interpretation of $P_{\boldsymbol{\theta}}(\mathbf{L})$ as a finite-support filter defined on the tessellation graph. Moreover, fast graph convolution algorithms are available for such filters [13].

- *Back-projection:* the operation $\left[\overline{\mathbf{B}} \circ \mathbf{B}\right]^H \mathrm{vec}(\hat{\mathbf{\Sigma}}) = \mathrm{diag}\left(\mathbf{B}^H \hat{\mathbf{\Sigma}} \mathbf{B}\right)$ (A.8) is a *back-projection*, mapping the raw microphone correlations to the image domain. Thanks to the convenient Khatri-Rao structure, this linear operation depends only on the matrix $\mathbf{B} \in \mathbb{C}^{M\times N}$.

[9]In all that follows, labels prefixed with roman letters refer to elements of the supplementary material.
[10]Typified by a rectilinear unit.

- *Bias:* the vector $\boldsymbol{\tau} \in \mathbb{R}^N$ is a non-uniform *bias*, boosting or shrinking the neurons of the network. Since only positive neurons are activated by the nonlinearity σ, this biasing operation helps sparsify the final acoustic image.

The total number of learnable coefficients in DeepWave is *linear* in the number of pixels. The rationale behind DeepWave's architecture is detailed in section 2, with theoretical justifications for the structures of the deblurring and back-projection linear operators. In section 3, we discuss network training, including initialisation and regularisation. We moreover derive the *forward-* and *backward-propagation* recursions[11] for our custom architecture, required for forming gradient steps. Finally, we test the architecture on synthetic as well as real data acquired with the *Pyramic array* [5, 46]. DeepWave is shown to have similar resolving power as state-of-the-art compressed-sensing methods, with a computational overhead similar to the DAS imager. To our knowledge, this is the first time a nonlinear imager of the kind achieves real-time performance on a standard computing platform. While developed primarily for acoustic cameras, DeepWave can easily be applied in neighbouring array signal processing fields [27], including radio astronomy, radar and sonar technologies.

2 Network architecture

In this section, we proceed similarly to [16, 50, 30] and construct DeepWave by studying the update equations of an iterative solver, namely proximal gradient descent applied to acoustic imaging.

2.1 Proximal gradient descent for acoustic imaging

In all that follows, we model the sound intensity field as a discrete *spherical map* with resolution N, specified by an *intensity vector* $\mathbf{x} \in \mathbb{R}_+^N$ and a *tessellation* $\Theta = \{\mathbf{r}_1, \ldots, \mathbf{r}_N\} \subset \mathbb{S}^2$. Spherical tessellations [19, 15] can be viewed as pixelation schemes for spherical geometries (see appendix B.1). As is customary in compressed-sensing, we propose to recover the sound intensity map by solving a convex optimisation problem (see appendix C):

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}\in\mathbb{R}_+^N} \frac{1}{2}\left\|\hat{\boldsymbol{\Sigma}} - \mathbf{A}\,\mathrm{diag}(\mathbf{x})\mathbf{A}^H\right\|_F^2 \quad + \quad \lambda\left[\gamma\|\mathbf{x}\|_1 + (1-\gamma)\|\mathbf{x}\|_2^2\right], \tag{2}$$

where $\|\cdot\|_F$ denotes the *Frobenius norm*, $\gamma \in]0,1[$ and $\lambda > 0$ are hyperparameters, and $\hat{\boldsymbol{\Sigma}} \in \mathbb{C}^{M\times M}$ is the empirical covariance matrix of the microphone recordings. In a far-field context, the *forward map* $\mathbf{A} \in \mathbb{C}^{M\times N}$ –linking the intensity vector to the microphone recordings– is commonly modelled by the so-called *steering matrix* [27]: $[\mathbf{A}]_{mn} := \exp\left(-2\pi j\langle \mathbf{p}_m, \mathbf{r}_n\rangle/\lambda_0\right)$, where $\{\mathbf{p}_1, \ldots, \mathbf{p}_M\} \subset \mathbb{R}^3$ are the microphone locations and $\lambda_0 > 0$ the sound wavelength. Using properties (A.5) and (A.6) of the *vectorisation operator* and the Frobenius norm [23, 53], problem (2) can be re-written in vectorised form as:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}\in\mathbb{R}_+^N} \frac{1}{2}\left\|\mathrm{vec}\left(\hat{\boldsymbol{\Sigma}}\right) - \left(\overline{\mathbf{A}} \circ \mathbf{A}\right)\mathbf{x}\right\|_2^2 \quad + \quad \lambda\left[\gamma\|\mathbf{x}\|_1 + (1-\gamma)\|\mathbf{x}\|_2^2\right], \tag{3}$$

where $\circ$ denotes the *Khatri-Rao product* (see definition A.3). Problem (3) is an *elastic-net penalised least-squares problem* [57], which seeks an optimal[12] trade-off between *data-fidelity* and *group-sparsity*. Group-sparsity is in this context better suited than traditional sparsity since acoustic sources are often diffuse. It is worth noting that, since the elastic-net functional is strictly convex for $\gamma \in [0, 1[$, problem (3) admits a unique solution. The latter can moreover be approximated by means of *proximal gradient descent (PGD)* [2], whose update equations are given here by (see appendix D):

$$\mathbf{x}^k = \mathrm{ReLu}\left(\frac{\mathbf{x}^{k-1} - \alpha\left(\overline{\mathbf{A}} \circ \mathbf{A}\right)^H\left[\left(\overline{\mathbf{A}} \circ \mathbf{A}\right)\mathbf{x}^{k-1} - \mathrm{vec}\left(\hat{\boldsymbol{\Sigma}}\right)\right] - \lambda\alpha\gamma}{2\lambda\alpha(1-\gamma) + 1}\right), \quad k \geq 1, \tag{4}$$

where $\mathbf{x}^0 \in \mathbb{R}^N$ is arbitrary, $\alpha \leq 1/\left\|\overline{\mathbf{A}} \circ \mathbf{A}\right\|_2^2$ is the *step size* and $\mathrm{ReLu}(x) := \max(x, 0)$ is the *rectified linear unit* [29], applied element-wise to a real vector.[13] The sequence of iterates $\{\mathbf{x}^k\}_{k\in\mathbb{N}}$

[11] DeepWave implementation can be found on `https://github.com/imagingofthings/DeepWave`.

[12] The notion of optimality is defined here by the penalty parameter λ.

[13] Note that with $\mathbf{x}^0 \in \mathbb{R}^N$, every gradient step produces a real vector.

defined in (4) reduces the objective function in (3) at a rate $O(1/k)$ [2]. Accelerated variants of proximal gradient descent have been proposed [2], which modify (4) with an extra *momentum term*:

$$\begin{cases} \mathbf{y}^k &= \mathrm{ReLu}\left(\dfrac{\mathbf{x}^{k-1} - \alpha\left(\overline{\mathbf{A}} \circ \mathbf{A}\right)^H \left[\left(\overline{\mathbf{A}} \circ \mathbf{A}\right)\mathbf{x}^{k-1} - \mathrm{vec}\left(\hat{\boldsymbol{\Sigma}}\right)\right] - \lambda\alpha\gamma}{2\lambda\alpha(1-\gamma)+1}\right), \quad k \geq 1, \\ \mathbf{x}^k &= \mathbf{y}^k + \omega^k\left(\mathbf{y}^k - \mathbf{y}^{k-1}\right) \end{cases} \tag{5}$$

where the *momentum sequence* $\{\omega^k\}_{k\in\mathbb{N}}$ can be designed in various ways [31, 9]. In our experiments, we will use (5) as a baseline for speed comparisons, where ω^k is updated according to Chambolle and Dossal's strategy [9]: $\omega^k = (k-1)/(k+d), \quad k \geq 0$, with $d = 50$ [31]. The *accelerated proximal gradient descent (APGD)* method thus obtained is the fastest reported in the literature, with convergence rate $o(1/k^2)$ [31]. Finally, we leverage the formulae $\left(\overline{\mathbf{A}} \circ \mathbf{A}\right)\mathbf{x} = \mathrm{vec}(\mathbf{A}\,\mathrm{diag}(\mathbf{x})\mathbf{A}^H)$ (A.5), and $\left(\overline{\mathbf{A}} \circ \mathbf{A}\right)^H \mathrm{vec}(\mathbf{R}) = \mathrm{diag}(\mathbf{A}^H\mathbf{R}\mathbf{A})$ (A.8), to compute gradient steps efficiently in (5).

2.2 DeepWave : a PGD-inspired RNN for fast acoustic imaging

In practice PGD is terminated according to some stopping criterion. The intensity map $\mathbf{x}^L$ obtained after L iterations of (4) can then be seen as the output of an RNN with depth L and intermediate neurons linked by the recursion formula:

$$\mathbf{x}^l = \mathrm{ReLu}\left(\boldsymbol{\mathcal{D}}\mathbf{x}^{l-1} + \boldsymbol{\mathcal{B}}\,\mathrm{vec}\left(\hat{\boldsymbol{\Sigma}}\right) - \boldsymbol{\tau}\right), \qquad l = 1, \dots, L. \tag{6}$$

We call this RNN the *oracle RNN*, since its weights $\boldsymbol{\mathcal{D}} \in \mathbb{R}^{N\times N}$, $\boldsymbol{\mathcal{B}} \in \mathbb{C}^{N\times M^2}$ and $\boldsymbol{\tau} \in \mathbb{R}^N$ are not learnt but simply *given* to us by identifying (6) with (4):

$$\boldsymbol{\mathcal{D}} = \frac{1}{\beta}\left[I - \alpha\left(\overline{\mathbf{A}} \circ \mathbf{A}\right)^H \left(\overline{\mathbf{A}} \circ \mathbf{A}\right)\right], \quad \boldsymbol{\mathcal{B}} = \frac{\alpha}{\beta}\left(\overline{\mathbf{A}} \circ \mathbf{A}\right)^H, \quad \boldsymbol{\tau} = \frac{\lambda\alpha\gamma}{\beta}\mathbf{1}_N, \tag{7}$$

where $\beta = 2\lambda\alpha(1-\gamma)+1$. An analysis of (7) allows us moreover to interpret physically the affine operations performed by the oracle RNN. The matrix $\boldsymbol{\mathcal{B}}$ first is a *back-projection* operator, mapping the vectorised correlation matrix into a spherical map by applying the adjoint of the forward operator used in (3). The resulting spherical map is called a *dirty map*, and is equivalent to the DAS image [53, Section 5.2][55]. The matrix $\boldsymbol{\mathcal{D}}$ then is a *deblurring operator*, which subtracts at each iteration a fraction of the array beamshape from the spherical map, hence *cleaning* the latter of blur artefacts. The vector $\boldsymbol{\tau}$ finally is an affine *shrinkage operator*, which biases uniformly the spherical map. The latter permits –in conjunction with the rectified linear unit– the *sparsification* of the spherical map and hence improve its angular resolution.
Since the oracle RNN is merely a reinterpretation of PGD, it inherits all its properties. In particular, it is capable of solving (3) with high accuracy for arbitrary input correlation matrices. Unfortunately, this great generalisability is typically obtained at the price of a very large number[14] of layers L, resulting in impractical reconstruction times. If one is however willing to sacrifice some of this generalisability, it is possible to reduce drastically the network depth by unfreezing the weights $\boldsymbol{\mathcal{D}}$, $\boldsymbol{\mathcal{B}}$, $\boldsymbol{\tau}$ in (6), and allowing them to be *learnt* for some specific input distribution. This idea was first explored in the context of sparse coding by Gregor and LeCun [16], resulting in the LISTA network. A fully-connected architecture, corresponding to unconstrained $\boldsymbol{\mathcal{D}}$, $\boldsymbol{\mathcal{B}}$ and $\boldsymbol{\tau}$, would however result in $\mathcal{O}(N^2)$ weights to be learnt, which is unfeasible in large-scale acoustic imaging problems. To overcome this issue, we propose in the next paragraphs a parsimonious parametrisation of $\boldsymbol{\mathcal{D}}$ and $\boldsymbol{\mathcal{B}}$. The resulting RNN architecture, dubbed DeepWave, is given in (1) and depicted in fig. 1.

Parametrisation of $\boldsymbol{\mathcal{D}}$ Our parametrisation of $\boldsymbol{\mathcal{D}}$ is motivated by the following result, characterising the oracle deblurring kernel for *spherical microphone arrays*[42] (see proof in appendix E).

Proposition 1. *Consider a* spherical microphone array*, with diameter D and microphone directions $\{\tilde{\mathbf{p}}_1, \dots, \tilde{\mathbf{p}}_M\} \subset \mathbb{S}^2$, forming a* near-regular tessellation *of the sphere. Then, we have*

$$\left[I - \alpha\left(\overline{\mathbf{A}} \circ \mathbf{A}\right)^H \left(\overline{\mathbf{A}} \circ \mathbf{A}\right)\right]_{ij} \simeq \left[\delta_{ij} - \alpha M^2 \,\mathrm{sinc}^2\left(\frac{D}{\lambda_0}\|\mathbf{r}_i - \mathbf{r}_j\|\right)\right], \ \forall i,j \in \{1,\dots,N\} \tag{8}$$

[14]Even with momentum acceleration, PGD typically requires more than 50 iterations to converge. The oracle RNN obtained by unrolling PGD will consequently be very deep.

Algorithm 1 DeepWave forward propagation

```
1: Input: $\hat{\mathbf{\Sigma}}_t$, $\mathbf{x}_t^0$, $\hat{\mathbf{x}}_t$, $\boldsymbol{\theta}$, $\mathbf{B}$, $\boldsymbol{\tau}$, $\sigma$
2: Output: $\mathcal{L}_t \in \mathbb{R}_+$, $\{\mathbf{s}_t^l\}_{l=1,\ldots,L} \subset \mathbb{R}^N$
3:
4: $\mathbf{y}_t \leftarrow \mathrm{diag}(\mathbf{B}^H \hat{\mathbf{\Sigma}}_t \mathbf{B}) - \boldsymbol{\tau}$
5: for $l$ in $[1,\ldots,L]$ do
6:     $\mathbf{s}_t^l \leftarrow P_{\boldsymbol{\theta}}(\mathbf{L})\mathbf{x}_t^{l-1} + \mathbf{y}_t$
7:     $\mathbf{x}_t^l \leftarrow \sigma(\mathbf{s}_t^l)$
8: $\mathcal{L}_t \leftarrow \frac{1}{2}\left\|\hat{\mathbf{x}}_t - \mathbf{x}_t^L\right\|_2^2 / \|\hat{\mathbf{x}}_t\|_2^2$
```

Algorithm 2 DeepWave backward propagation

```
1: Input: $\hat{\mathbf{\Sigma}}_t$, $\mathbf{x}_t^0$, $\hat{\mathbf{x}}_t$, $\boldsymbol{\theta}$, $\mathbf{B}$, $\sigma$, $\{\mathbf{s}_t^l\}_{l=1,\ldots,L}$
2: Output: $\partial\boldsymbol{\theta} \in \mathbb{R}^{K+1}$, $\partial\mathbf{B} \in \mathbb{C}^{M\times N}$, $\partial\boldsymbol{\tau} \in \mathbb{R}^N$
3: $(\partial\mathbf{x}, \partial\boldsymbol{\theta}, \partial\boldsymbol{\tau}) \leftarrow \left(\left(\sigma(\mathbf{s}_t^L) - \hat{\mathbf{x}}_t\right) / \|\hat{\mathbf{x}}_t\|_2^2, \mathbf{0}, \mathbf{0}\right)$
4: for $l$ in $[L,\ldots,1]$ do
5:     $\partial\mathbf{s} \leftarrow \mathrm{diag}\left(\sigma'(\mathbf{s}_t^l)\right)\partial\mathbf{x}$
6:     $\partial\mathbf{x} \leftarrow P_{\boldsymbol{\theta}}(\mathbf{L})\partial\mathbf{s}$
7:     $\partial\boldsymbol{\tau} \leftarrow \partial\boldsymbol{\tau} - \partial\mathbf{s}$
8:     $[\partial\boldsymbol{\theta}]_k \leftarrow [\partial\boldsymbol{\theta}]_k + \partial\mathbf{s}^T T_k(\mathbf{L})\,\sigma(\mathbf{s}_t^{l-1})$
9: $\partial\mathbf{B} \leftarrow -2\hat{\mathbf{\Sigma}}_t\mathbf{B}\,\mathrm{diag}\left(\partial\boldsymbol{\tau}\right)$
```

Figure 2: Forward and backward algorithms to compute gradients of $\mathcal{L}_t$ with respect to $\boldsymbol{\theta}, \mathbf{B}, \boldsymbol{\tau}$. For notational simplicity we use the shorthand $\partial\boldsymbol{\alpha} = \partial\mathcal{L}_t/\partial\boldsymbol{\alpha}$, and assume $\sigma(\mathbf{s}_t^0) = \mathbf{x}_t^0$.

where λ_0 *is the* wavelength*,* δ_{ij} *denotes the* Kronecker delta *and* $\mathrm{sinc}(x) := \sin(\pi x)/\pi x$ *is the* cardinal sine*. Moreover, the approximation* (8) *is* extremely good *for* $M \geq 3\lfloor \frac{2\pi D}{\lambda_0} \rfloor^2$.

Proposition 1 tells us that, for spherical arrays with sufficient number of microphones[15], the oracle deblurring operator $\boldsymbol{\mathcal{D}}$ in (7) corresponds actually to a sampled *zonal kernel* [35]: $[\boldsymbol{\mathcal{D}}]_{ij} = \kappa(\|\mathbf{r}_i - \mathbf{r}_j\|)$ for some $\kappa : \mathbb{R}_+ \to \mathbb{R}$. Since zonal kernels are used to define spherical convolutions [35], $\boldsymbol{\mathcal{D}}$ can hence be seen as a *discrete convolution operator* over the tessellation in use $\Theta = \{\mathbf{r}_1, \ldots, \mathbf{r}_N\}$. Its bandwidth is moreover essentially finite, since coefficients $[\boldsymbol{\mathcal{D}}]_{ij}$ decay as $1/\|\mathbf{r}_i - \mathbf{r}_j\|^2$. As discussed in [41, 13], discrete spherical convolution operators with finite scope can be efficiently represented and implemented by means of *graph signal processing* [47] techniques. This leads us to consider the following parametrisation (see appendix B.3 for details): $\boldsymbol{\mathcal{D}} = P_{\boldsymbol{\theta}}(\mathbf{L}) := \sum_{k=0}^K \theta_k \mathbf{L}^k$, where $\boldsymbol{\theta} = [\theta_0, \ldots, \theta_K] \in \mathbb{R}^{K+1}$, K controls the scope of the discrete convolution and $\mathbf{L} \in \mathbb{R}^{N\times N}$ is the *Laplacian* [47] associated to the convex-hull graph of Θ. Note that with this parametrisation, the number of parameters characterising $\boldsymbol{\mathcal{D}}$ drops from N^2 to $K+1$, with $K \ll N$.

Parametrisation of $\boldsymbol{\mathcal{B}}$ The oracle back-projection operator (7) admits a factorisation in terms of the Khatri-Rao product. We decide hence to equip $\boldsymbol{\mathcal{B}}$ with a similar structure: $\boldsymbol{\mathcal{B}} = (\overline{\mathbf{B}} \circ \mathbf{B})^H$ for some learnable matrix $\mathbf{B} \in \mathbb{C}^{M\times N}$. With such a parametrisation, the number of parameters characterising $\boldsymbol{\mathcal{B}}$ drops from NM^2 to NM. The Khatri-Rao structure guarantees moreover real-valued –and hence physically-interpretable– dirty maps.

3 Network training

To facilitate the description of the training procedure, we adopt the following shorthand notations.

- $\mathrm{DeepWave}(\mathbf{\Omega}, L)$ denotes a specific instance of the DeepWave network (1) with parameters $\mathbf{\Omega} := \{\boldsymbol{\theta}, \mathbf{B}, \boldsymbol{\tau}\}$ and depth L.
- $\mathrm{APGD}(\alpha, \lambda, \gamma)$ denotes an instance of APGD (5), with tuning parameters $(\alpha, \lambda, \gamma) \in \mathbb{R}_+^3$.

The network parameters are chosen as minimisers of the following optimisation problem:

$$\hat{\mathbf{\Omega}} \in \underset{\substack{\boldsymbol{\theta}\in\mathbb{R}^{K+1}\\ \mathbf{B}\in\mathbb{C}^{M\times N}\\ \boldsymbol{\tau}\in\mathbb{R}^N}}{\arg\min}\; \frac{1}{T}\sum_{t=1}^T \underbrace{\frac{\left\|\hat{\mathbf{x}}_t - \mathbf{x}_t^L(\mathbf{\Omega})\right\|_2^2}{2\left\|\hat{\mathbf{x}}_t\right\|_2^2}}_{:=\mathcal{L}_t} + \underbrace{\frac{\lambda_{\boldsymbol{\theta}}}{2(K+1)}\|\boldsymbol{\theta}\|_2^2}_{:=\mathcal{L}_{\boldsymbol{\theta}}} + \underbrace{\frac{\lambda_{\mathbf{B}}}{2MN}\|\mathbf{B}\|_F^2}_{:=\mathcal{L}_{\mathbf{B}}} + \underbrace{\frac{\lambda_{\boldsymbol{\tau}}}{2N}\left\|\mathbf{L}^{1/2}\boldsymbol{\tau}\right\|_2^2}_{:=\mathcal{L}_{\boldsymbol{\tau}}}. \tag{9}$$

The quantities $\{\mathbf{x}_t^L(\mathbf{\Omega})\}_t$ and $\{\hat{\mathbf{x}}_t\}_t$ in (9) correspond respectively to the outputs of $\mathrm{DeepWave}(\mathbf{\Omega}, L)$ and $\mathrm{APGD}(\alpha, \lambda, \gamma)$ with identical example input data $\{(\hat{\mathbf{\Sigma}}_t, \mathbf{x}_t^0)\}_t$. The first term $\frac{1}{T}\sum_{t=1}^T \mathcal{L}_t$ is a

[15]For a spherical array with diameter $D = 30$ cm operating at 1 kHz, $M \geq 90$ is sufficient.

data-fidelity term, which attempts to bring $\hat{\mathbf{x}}_t$ and $\mathbf{x}_t^L(\mathbf{\Omega})$ as close as possible from one another.[16] The additional terms $\mathcal{L}_{\boldsymbol{\theta}}, \mathcal{L}_{\mathbf{B}}, \mathcal{L}_{\boldsymbol{\tau}}$ are smoothing *regularisers*, fighting against *overfitting*, a common issue in deep learning. Since the shrinkage operator $\boldsymbol{\tau}$ is defined over an irregular spherical tessellation, the smoothing term $\mathcal{L}_{\boldsymbol{\tau}}$ is defined via the Laplacian $\mathbf{L} \in \mathbb{R}^{N\times N}$ associated to the connectivity graph of the tessellation, as is customary in graph signal processing (see appendix B.3).
Optimisation of (9) is carried out by *stochastic gradient descent (SGD)* with momentum acceleration [51]. Gradients of $\mathcal{L}_t$ with respect to $\boldsymbol{\theta}, \mathbf{B}, \boldsymbol{\tau}$ are efficiently evaluated using *reverse-mode algorithmic differentiation* [1, 25] and are given in algorithms 1 and 2 (see appendix F for a derivation). While random initialisation of neural-networks is a common practice in deep learning [51], this strategy failed for our specific architecture, leading to poor validation loss and considerably increased training times. Instead, we hence use the oracle parameters (7) to initialise SGD:

$$\boldsymbol{\theta}^0 := \underset{\boldsymbol{\theta}\in\mathbb{R}^{K+1}}{\arg\min} \|P_{\boldsymbol{\theta}}(\mathbf{L}) - \boldsymbol{\mathcal{D}}\|_F^2, \quad \mathbf{B}^0 := \sqrt{\frac{\alpha}{\beta}}\mathbf{A}, \quad \boldsymbol{\tau}^0 := \frac{\lambda\alpha\gamma}{\beta}\mathbf{1}_N. \tag{10}$$

For greater numerical stability during training, we follow [41] and reparameterise the deblurring filter as $P_{\boldsymbol{\theta}}(\tilde{\mathbf{L}}) = \sum_{k=0}^{K} \theta_k T_k(\tilde{\mathbf{L}})$, where $T_k(\cdot)$ is the Chebychev polynomial of order k and $\tilde{\mathbf{L}}$ is the normalised Laplacian with spectrum in $[-1, 1]$ (see appendix B.3 for implementation details). Finally, we substitute the ReLu activation function by a scaled rectified $\tanh$ to avoid the exploding gradient problem [39].[17]

4 Experimental results

In this section, we compare the accuracy, resolution and runtime performance of DeepWave to DAS and APGD on real-world (RW) and simulated (SIM) datasets. More comprehensive dataset descriptions and additional results, including an ablation study, are provided in appendices G to I.

Dataset 1 [36] (RW) reproduces a conference room setup depicted in figs. 3a and 3b, where 8 people[18] are gathered around a table and speak either in turns or simultaneously (with at most 3 concurrent speakers). Recordings of the conversation are collected by the 48-element Pyramic array [46] (fig. 3f) positioned at the centre of the table. Since human speech is wide-band, the audible range [1500, 4500] Hz in the latter are pre-processed every 100 ms and split into 9 uniform bins to form a suitable training set $\{(\hat{\boldsymbol{\Sigma}}_t, \hat{\mathbf{x}}_t, \mathbf{x}_t^0)\}_t$ of 2760 data points per frequency band for DeepWave (with $N = 2234$). (See appendix G.2.) Frequency channels are processed independently by each algorithm. DeepWave is trained by splitting the data points into a training and validation set (respectively 80% and 20% in size). For each frequency band, we chose an architecture with 5 layers.
In fig. 3, figs. G.4 and H.3 respectively, we compare the accuracy and runtime of DeepWave, DAS and APGD. A video showing the evolution in time of DeepWave and DAS azimuthal sound fields (as in figs. 3a and 3b) is also available.[19] In terms of resolution, DeepWave and APGD perform similarly, outperforming DAS by approximately 27%. The mean contrast scores for DeepWave and DAS over the test set of Dataset 1 are 0.99 ($\pm$0.0081) and 0.89 ($\pm$0.07), respectively. Note that since the metrics used for assessing resolution and contrast[20] are not perfectly reflective of human-eye perception, the reported image quality improvements appear even more striking through visual inspection of the sound intensity fields (see for example fig. 3).

Dataset 2 [43] (RW) consists of 2700 template recordings from the Pyramic array taken in an anechoic chambre at an angular resolution of 2 degrees in azimuth and three different elevations (-15, 0, 15 degrees). Recordings contain both male and female speech samples to cover a wide audible range. The audio samples can be combined to simulate complex multi-source sound fields, hence we leverage this property to augment the dataset to 5700 distinct recordings with one, two, or three active speakers simultaneously. The raw time-series are then pre-processed as for Dataset 1 to obtain a

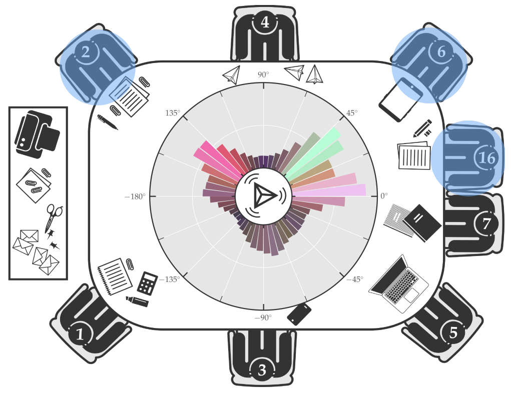
[16]in a mean relative squared-error sense.

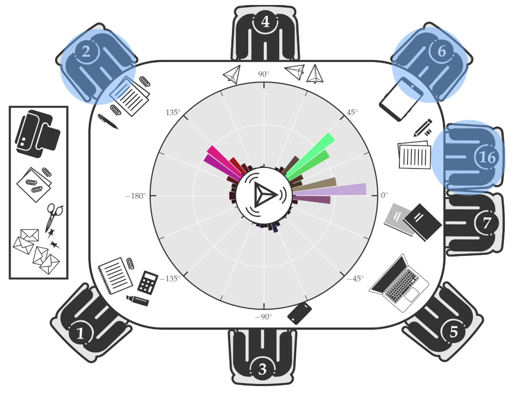
[17]An alternative is to use a *truncated* ReLu. Given initialisation strategy 10, network training will still converge with similar step sizes as those used with $\tanh$ non-linearities.

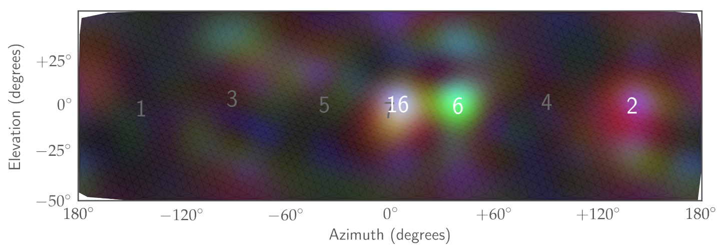
[18]The 8 people are represented in the experiment by loadspeakers playing male and female speech samples.

[19]Available online: `https://www.youtube.com/watch?v=PwB3CS2rHdI`

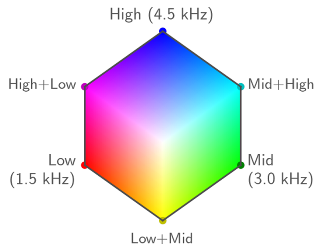
[20]As is customary, resolution is measured as the *width at half-maximum* of the impulse response of the algorithms. Contrast is measured as the difference between the maximum and mean of the greyscale image.


(a) DAS azimuthal sound field.

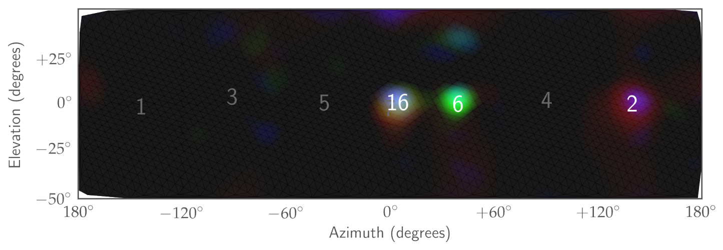

(b) DeepWave azimuthal sound field.



(c) DAS spherical sound field (resolution: 25.3° , RMS contrast: 0.78).



(d) Frequency-colour mapping.



(e) DeepWave spherical sound field (resolution: 18.5° , contrast: 0.97).

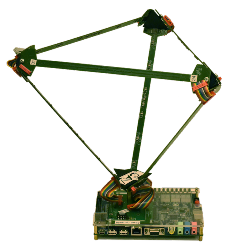
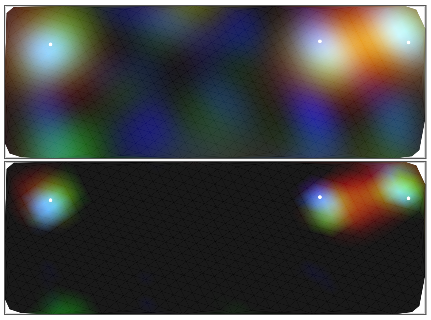
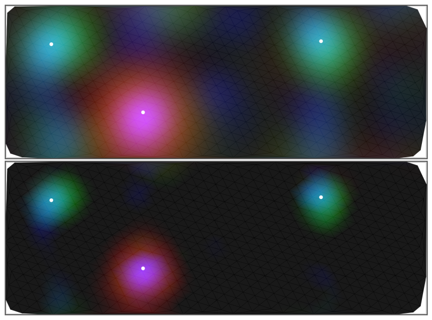

(f) Pyramic array.

Figure 3: Snapshots at time $t = 1.7$ s of the sound intensity fields produced by DeepWave and DAS for the Pyramic recordings with speakers 2, 6 and 16 active. Sound frequencies range from 1.5 to 4.5 kHz and were mapped to true colours (see fig. 3d, colour shades correspond to lower intensities). The spherical maps of DAS and DeepWave are plotted in figs. 3c and 3e, respectively. In figs. 3a and 3b we plot the azimuthal projections of figs. 3c and 3e, respectively.

training set of 151980 data points per frequency band (with $N = 1568$). Network training is identical to that of Dataset 1, except that 10 azimuth directions are also witheld from the training set to assess how well the network generalises to emissions from unseen directions.
Figures 4a and 4b show sample DAS and DeepWave reconstructions with real sources from directions withheld from the training set. Similarly, fig. 4c shows sample reconstructions when the network is trained on real data but tested on synthetic narrow-band covariance matrices induced by sources from directions absent from the training set. In both cases we see that DeepWave outperforms DAS in resolution and contrast (i.e. sharper blobs and darker background).

Dataset 3 (SIM) finally is a dataset with recordings from a spherical microphone array using a narrow-band point-source data-model at 2 kHz [53]. The sources are randomly positioned over a 120° field-of-view, with up to 10 concurrent sources per recording. Experiment results available in fig. H.1 corroborate the real-data results, hence showing that DeepWave generalises well to a large number of sources with unconstrained positions. We further investigated in fig. H.2 the influence of network depth, and concluded that 5 or 6 layers are generally sufficient for the investigated dataset.


(a) DAS/DeepWave sound fields for Dataset 2.


(b) DAS/DeepWave sound fields for Dataset 2.

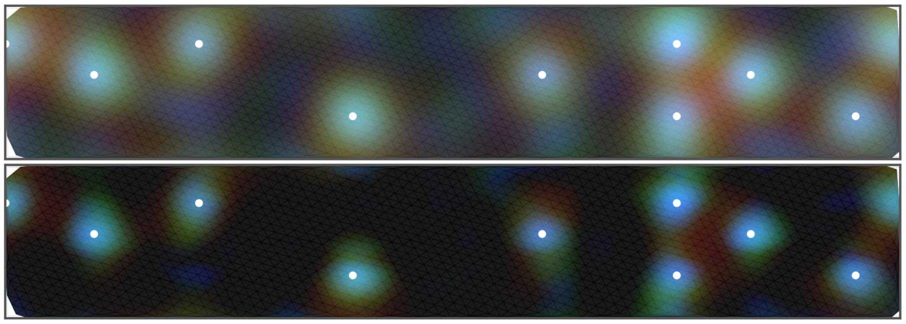

(c) DAS/DeepWave sound fields for synthetic data trained on Dataset 2.

Figure 4: Snapshots of the sound intensity fields produced by DeepWave and DAS when trained on Dataset 2 (with 10 held-out source directions). Each subplot contains a DAS image (top) and a DeepWave image (bottom). The frequency color mapping is identical to fig. 3d. Figures 4a and 4b show azimuthal sound field slices on $[-20^\circ, 150^\circ]$ using real-world covariance matrices with sources from unseen directions during training. Figure 4c shows a full 360° sound field on a synthetic covariance matrix from unseen directions during training. Elevations span $[-15^\circ, +15^\circ]$.

In terms of runtimes finally, DeepWave and DAS both reach real-time requirements (6.5 ms and 2.0 ms respectively), largely outperforming APGD (211 ms). (See fig. H.3 for more details.)

5 Conclusion

We introduced DeepWave, the first recurrent neural-network for real-time and high resolution acoustic imaging. It mimics iterative solvers from convex optimisation, while using the natural structure of acoustic imaging problems for efficient training and operation. Our real and simulated data experiments show DeepWave has similar computational speed to the state-of-the-art DAS imager with vastly superior resolution and contrast.

For future work, one of our goals is to make DeepWave *time-aware*, by training it on sequences of consecutive measurements in time. To this end, we plan to connect multiple DeepWave networks together, one for each time, and train them end-to-end. In such an architecture, the output neurons from one network would serve as initial neural state $\mathbf{x}^0$ for the next network in line. This can be interpreted as warm-starting the network with the sound field estimated at the previous time instant. Additionally, we would like to propose a frequency-invariant DeepWave architecture, allowing to train a single network for all frequency bands. Properties of the oracle weights (7) suggest that this should be possible. This would considerably facilitate the training of the network, since the training set would be augmented and the number of trainable parameters reduced.

Acknowledgments We thank Erwan Zerhouni for useful discussions regarding network training and implementation details; and Ivan Dokmanić for insights on related works that inspired our approach. Finally we express our gratitude towards Robin Scheibler and Hanjie Pan for their openly-accessible real-world datasets [36, 43].

References

[1] Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of Marchine Learning Research*, 18:1–43, 2018.

[2] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[3] Jacob Benesty, Jingdong Chen, and Yiteng Huang. *Microphone array signal processing*, volume 1. Springer Science & Business Media, 2008.

[4] Adrien Besson, Lucien Roquette, Dimitris Perdios, Matthieu Simeoni, Marcel Arditi, Paul Hurley, Yves Wiaux, and Jean-Philippe Thiran. A physical model of non-stationary blur in ultrasound imaging. *IEEE Transactions on Computational Imaging*, 2019.

[5] Eric Bezzam, Robin Scheibler, Juan Azcarreta, Hanjie Pan, Matthieu Simeoni, René Beuchat, Paul Hurley, Basile Bruneau, Corentin Ferry, and Sepand Kashani. Hardware and software for reproducible research in audio array signal processing. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6591–6592. Ieee, 2017.

[6] Jan Biemond, Reginald L Lagendijk, and Russell M Mersereau. Iterative methods for image deblurring. *Proceedings of the IEEE*, 78(5):856–883, 1990.

[7] Thomas F Brooks and William M Humphreys. A deconvolution approach for the mapping of acoustic sources (damas) determined from phased microphone arrays. *Journal of Sound and Vibration*, 294(4-5):856–879, 2006.

[8] Leon Brusniak, James Underbrink, and Robert Stoker. Acoustic imaging of aircraft noise sources using large aperture phased arrays. In *12th AIAA/CEAS Aeroacoustics Conference (27th AIAA Aeroacoustics Conference)*, page 2715, 2006.

[9] Antonin Chambolle and Ch Dossal. On the convergence of the iterates of the "fast iterative shrinkage/thresholding algorithm". *Journal of Optimization theory and Applications*, 166(3):968–982, 2015.

[10] Yunjin Chen and Thomas Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1256–1272, 2017.

[11] Ning Chu, José Picheral, Ali Mohammad-Djafari, and Nicolas Gac. A robust super-resolution approach with sparsity constraint in acoustic imaging. *Applied Acoustics*, 76:197–208, 2014.

[12] Zhigang Chu and Yang Yang. Comparison of deconvolution methods for the visualization of acoustic sources based on cross-spectral imaging function beamforming. *Mechanical Systems and Signal Processing*, 48(1-2):404–422, 2014.

[13] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[14] Simon Foucart and Holger Rauhut. A mathematical introduction to compressive sensing. *Bull. Am. Math*, 54:151–165, 2017.

[15] Krzysztof M Gorski, Eric Hivon, Anthony J Banday, Benjamin D Wandelt, Frode K Hansen, Mstvos Reinecke, and Matthia Bartelmann. Healpix: a framework for high-resolution discretization and fast analysis of data distributed on the sphere. *The Astrophysical Journal*, 622(2):759, 2005.

[16] Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pages 399–406. Omnipress, 2010.

[17] Harshit Gupta, Kyong Hwan Jin, Ha Q Nguyen, Michael T McCann, and Michael Unser. Cnn-based projected gradient descent for consistent ct image reconstruction. *IEEE transactions on medical imaging*, 37(6):1440–1453, 2018.

[18] RK Hansen and PA Andersen. A 3d underwater acoustic camera—properties and applications. In *Acoustical Imaging*, pages 607–611. Springer, 1996.

[19] Doug P Hardin, Timothy Michaels, and Edward B Saff. A comparison of popular point configurations on s^ 2. *Dolomites Research Notes on Approximation*, 9(1), 2016.

[20] Paul Hurley and Matthieu Simeoni. Flexibeam: analytic spatial filtering by beamforming. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2877–2880. Ieee, 2016.

[21] Paul Hurley and Matthieu Simeoni. Flexarray: Random phased array layouts for analytical spatial filtering. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3380–3384. Ieee, 2017.

[22] Kyong Hwan Jin, Michael T McCann, Emmanuel Froustey, and Michael Unser. Deep convolutional neural network for inverse problems in imaging. *IEEE Transactions on Image Processing*, 26(9):4509–4522, 2017.

[23] KG Jinadasa. Applications of the matrix operators vech and vec. *Linear Algebra and its Applications*, 101:73–79, 1988.

[24] Charles H Jones and George A Gilmour. Acoustic camera apparatus, August 5 1975. US Patent 3,898,608.

[25] Sepand Kashani. Optimization notes. page 6, 2019.

[26] K Kim, N Neretti, and N Intrator. Mosaicing of acoustic camera images. *IEE Proceedings-Radar, Sonar and Navigation*, 152(4):263–270, 2005.

[27] Hamid Krim and Mats Viberg. Two decades of array signal processing research. *IEEE signal processing magazine*, 1996.

[28] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[29] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[30] Huan Li, Yibo Yang, Dongmin Chen, and Zhouchen Lin. Optimization algorithm inspired deep neural network structure design. *arXiv preprint arXiv:1810.01638*, 2018.

[31] Jingwei Liang and Carola-Bibiane Schönlieb. Faster fista. *arXiv preprint arXiv:1807.04005*, 2018.

[32] Michael Lustig, David Donoho, and John M Pauly. Sparse mri: The application of compressed sensing for rapid mr imaging. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 58(6):1182–1195, 2007.

[33] Michael T McCann, Kyong Hwan Jin, and Michael Unser. Convolutional neural networks for inverse problems in imaging: A review. *IEEE Signal Processing Magazine*, 34(6):85–95, 2017.

[34] Andy Meyer and Dirk Döbler. Noise source localization within a car interior using 3d-microphone arrays. *Proceedings of the BeBeC*, pages 1–7, 2006.

[35] Volker Michel. Lectures on constructive approximation. *AMC*, 10:12, 2013.

[36] Hanjie Pan, Robin Scheibler, Eric Bezzam, Ivan Dokmanić, and Martin Vetterli. Audio speech recordings used in the paper FRIDA: FRI-based DOA Estimation for Arbitrary Array Layout, March 2017. This work was supported by the Swiss National Science Foundation grant 20FP-1 151073, LABEX WIFI under references ANR-10-LABX-24 and ANR-10-IDEX-0001-02 PSL* and by Agence Nationale de la Recherche under reference ANR-13-JS09-0001-01.

[37] Neal Parikh, Stephen Boyd, et al. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, 2014.

[38] Sung Cheol Park, Min Kyu Park, and Moon Gi Kang. Super-resolution image reconstruction: a technical overview. *IEEE signal processing magazine*, 20(3):21–36, 2003.

[39] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318, 2013.

[40] Dimitris Perdios, Adrien Besson, Marcel Arditi, and Jean-Philippe Thiran. A deep learning approach to ultrasound image recovery. In *2017 IEEE International Ultrasonics Symposium (IUS)*, pages 1–4. Ieee, 2017.

[41] Nathanaël Perraudin, Michaël Defferrard, Tomasz Kacprzak, and Raphael Sgier. Deepsphere: Efficient spherical convolutional neural network with healpix sampling for cosmological applications. *Astronomy and Computing*, 2019.

[42] Boaz Rafaely. *Fundamentals of spherical array processing*, volume 8. Springer, 2015.

[43] Scheibler Robin. Pyramic Dataset : 48-Channel Anechoic Audio Recordings of 3D Sources, March 2018. The author would like to acknowledge Juan Azcarreta Ortiz, Corentin Ferry, and René Beuchat for their help in the design and usage of the Pyramic array. Hanjie Pan, Miranda Kreković, Mihailo Kolundzija, and Dalia El Badawy for lending a hand, or even two, during experiments. Finally, Juan Azcarreta Ortiz, Eric Bezzam, Hanjie Pan and Ivan Dokmanić for feedback on the documentation and dataset organization.

[44] Justin Romberg. Imaging via compressive sampling. *IEEE Signal Processing Magazine*, 25(2):14–20, 2008.

[45] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[46] Robin Scheibler, Juan Azcarreta, René Beuchat, and Corentin Ferry. Pyramic: Full stack open microphone array architecture and dataset. In *2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)*, pages 226–230. IEEE, 2018.

[47] David Shuman, Sunil Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 3(30):83–98, 2013.

[48] Pieter Sijtsma. Clean based on spatial source coherence. *International journal of aeroacoustics*, 6(4):357–374, 2007.

[49] Matthieu Martin Jean-Andre Simeoni and Paul Hurley. Graph spectral clustering of convolution artefacts in radio interferometric images. Technical report, 2019.

[50] Jian Sun, Huibin Li, Zongben Xu, et al. Deep admm-net for compressive sensing mri. In *Advances in neural information processing systems*, pages 10–18, 2016.

[51] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[52] Michael Unser, Julien Fageot, and Harshit Gupta. Representer theorems for sparsity-promoting l1 regularization. *IEEE Transactions on Information Theory*, 62(9):5167–5180, 2016.

[53] Alle-Jan van der Veen and Stefan J Wijnholds. Signal processing tools for radio astronomy. In *Handbook of Signal Processing Systems*, pages 421–463. Springer, 2013.

[54] Yves Wiaux, Laurent Jacques, Gilles Puy, Anna MM Scaife, and Pierre Vandergheynst. Compressed sensing imaging techniques for radio interferometry. *Monthly Notices of the Royal Astronomical Society*, 395(3):1733–1742, 2009.

[55] Stefan J Wijnholds and Alle-Jan van der Veen. Fundamental imaging limits of radio telescope arrays. *IEEE Journal of Selected Topics in Signal Processing*, 2(5):613–623, 2008.

[56] Li Xu, Jimmy SJ Ren, Ce Liu, and Jiaya Jia. Deep convolutional neural network for image deconvolution. In *Advances in Neural Information Processing Systems*, pages 1790–1798, 2014.

[57] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

DeepWave: A Recurrent Neural-Network for Real-Time Acoustic Imaging *Supplementary Material*

Matthieu Simeoni [*]
IBM Zurich Research Laboratory
meo@zurich.ibm.com

Sepand Kashani [†]
École Polytechnique Fédérale de Lausanne (EPFL)
sepand.kashani@epfl.ch

Paul Hurley
Western Sydney University
paul.hurley@westernsydney.edu.au

Martin Vetterli
École Polytechnique Fédérale de Lausanne (EPFL)
martin.vetterli@epfl.ch

Contents

[*]Corresponding author. Matthieu Simeoni is also affiliated to the École Polytechnique Fédérale de Lausanne (EPFL), with email address matthieu.simeoni@epfl.ch

[†]Matthieu Simeoni and Sepand Kashani have contributed equally to this work. Sepand Kashani was in part supported by the Swiss National Science Foundation grant number 200021 181978/1, “SESAM - Sensing and Sampling: Theory and Algorithms".

Preprint. Under review.

A Linear algebra primer

This work makes heavy use of the Kronecker product and related operators. To ease the user's understanding, we provide a short description of these operators along with proofs of common transforms used throughout the text. Useful references for this section are [9, 5].

A.1 Conventions

Throuhought this document, we adopt the following conventions:

- Vectors are denoted with bold lowercase letters: $\mathbf{y}$.
- Matrices are denoted with bold uppercase letters: $\mathbf{A}$.
- If $\mathbf{A} \in \mathbb{C}^{M\times N}$, $\mathbf{a}_k \in \mathbb{C}^M$ denotes the k-th column of $\mathbf{A}$.
- The i-th entry of vector $\mathbf{y}$ is denoted $[\mathbf{y}]_i$.
- The (i, j)-th entry of matrix $\mathbf{A}$ is denoted $[\mathbf{A}]_{ij}$.
- The conjugation operation is denoted by overlining a vector or a matrix respectively: $\overline{\mathbf{a}}$, $\overline{\mathbf{A}}$.
- The modulus of a complex number $z \in \mathbb{C}$ is denoted by $|z|$.

A.2 Hadamard, Kronecker and Khatri-Rao products

The Hadamard product is the element-wise multiplication operator:

Definition A.1 (Hadamard product)**.** *Let* $\mathbf{A} \in \mathbb{C}^{M\times N}$ *and* $\mathbf{B} \in \mathbb{C}^{M\times N}$. *The* Hadamard product $\mathbf{A} \odot \mathbf{B} \in \mathbb{C}^{M\times N}$ *is defined as*

$$[\mathbf{A} \odot \mathbf{B}]_{ij} = [\mathbf{A}]_{ij} [\mathbf{B}]_{ij} .$$

Moreover, we denote by $\mathbf{A}^{\odot 2}$ *the Hadamard square of a matrix:* $\mathbf{A} \odot \mathbf{A}$.

The Kronecker product generalises the vector outer product to matrices, and represents the tensor product between two finite-dimensional linear maps:

Definition A.2 (Kronecker product)**.** *Let* $\mathbf{A} \in \mathbb{C}^{M_1\times N_1}$ *and* $\mathbf{B} \in \mathbb{C}^{M_2\times N_2}$. *The* Kronecker product $\mathbf{A} \otimes \mathbf{B} \in \mathbb{C}^{M_1M_2\times N_1N_2}$ *is defined as*

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} [\mathbf{A}]_{11}\mathbf{B} & \cdots & [\mathbf{A}]_{1N_1}\mathbf{B} \\ \vdots & \ddots & \vdots \\ [\mathbf{A}]_{M_11}\mathbf{B} & \cdots & [\mathbf{A}]_{M_1N_1}\mathbf{B} \end{bmatrix}.$$

The main properties of the Kronecker product are [9]:

$$(\mathbf{A} \otimes \mathbf{B})^H = \mathbf{A}^H \otimes \mathbf{B}^H, \tag{A.1}$$
$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC}) \otimes (\mathbf{BD}), \tag{A.2}$$
$$(\mathbf{A} \otimes \mathbf{B}) \odot (\mathbf{C} \otimes \mathbf{D}) = (\mathbf{A} \odot \mathbf{C}) \otimes (\mathbf{B} \odot \mathbf{D}). \tag{A.3}$$

The Khatri-Rao product finally, is a column-wise Kronecker product:

Definition A.3 (Khatri-Rao product)**.** *Let* $\mathbf{A} \in \mathbb{C}^{M_1\times N}$ *and* $\mathbf{B} \in \mathbb{C}^{M_2\times N}$. *The* Khatri-Rao product $\mathbf{A} \circ \mathbf{B} \in \mathbb{C}^{M_1M_2\times N}$ *is defined as*

$$\mathbf{A} \circ \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1, \ldots, \mathbf{a}_N \otimes \mathbf{b}_N].$$

A.3 Matrix identities

In imaging problems, $\mathbf{A} \otimes \mathbf{B}$ and $\mathbf{A} \circ \mathbf{B}$ are often too large to be stored in memory. However it is not the matrix itself that is of interest in many circumstances, but rather the effect of a linear map such as $f(\mathbf{x}) = (\mathbf{A} \otimes \mathbf{B})\mathbf{x}$. The matrix identities below allow us to evaluate $f(\mathbf{x})$ without ever having to compute large intermediate arrays. They make use of the vectorisation operator, defined hereafter:

Definition A.4 (Vectorisation)**.** *Let* $\mathbf{A} \in \mathbb{C}^{M\times N}$. *The* vectorisation operator $\operatorname{vec}(\cdot)$ *reshapes a matrix into a vector by stacking its columns:*

$$[\operatorname{vec}(\mathbf{A})]_{M(j-1)+i} = [\mathbf{A}]_{ij}\,.$$

Conversely, the matricisation operator $\operatorname{mat}_{M,N}(\cdot)$ *reshapes a vector into a matrix:*

$$[\operatorname{mat}_{M,N}(\mathbf{a})]_{ij} = [\mathbf{a}]_{M(j-1)+i}\,.$$

Commonly used matrix identities are the following [5, 23]:

$$\operatorname{vec}(\mathbf{ABC}) = \left(\mathbf{C}^T \otimes \mathbf{A}\right)\operatorname{vec}(\mathbf{B}) \tag{A.4}$$

$$\operatorname{vec}(\mathbf{A}\operatorname{diag}(\mathbf{b})\mathbf{C}) = \left(\mathbf{C}^T \circ \mathbf{A}\right)\mathbf{b} \tag{A.5}$$

$$\langle \mathbf{A}, \mathbf{B}\rangle_F = \operatorname{tr}\left(\mathbf{A}^H\mathbf{B}\right) = \operatorname{vec}(\mathbf{A})^H\operatorname{vec}(\mathbf{B}) \tag{A.6}$$

$$\operatorname{vec}(\mathbf{b}\mathbf{a}^T) = \mathbf{a}\otimes\mathbf{b} \tag{A.7}$$

In this work, we furthermore make use of the following nonstandard matrix identities, proven hereafter:

$$(\mathbf{A}\circ\mathbf{B})^H\operatorname{vec}(\mathbf{C}) = \operatorname{diag}\left(\mathbf{B}^H\mathbf{C}\overline{\mathbf{A}}\right) \tag{A.8}$$

$$(\mathbf{A}\otimes\mathbf{B})^H(\mathbf{A}\otimes\mathbf{B})\operatorname{vec}(\mathbf{C}) = \operatorname{vec}(\mathbf{B}^H\mathbf{B}\mathbf{C}\mathbf{A}^T\overline{\mathbf{A}}) \tag{A.9}$$

$$(\mathbf{A}\circ\mathbf{B})^H(\mathbf{A}\circ\mathbf{B})\,\mathbf{c} = \operatorname{diag}(\mathbf{B}^H\mathbf{B}\operatorname{diag}(\mathbf{c})\mathbf{A}^T\overline{\mathbf{A}}) \tag{A.10}$$

$$(\mathbf{A}\circ\mathbf{B})^H(\mathbf{A}\circ\mathbf{B}) = \mathbf{A}^H\mathbf{A}\odot\mathbf{B}^H\mathbf{B}. \tag{A.11}$$

Proof. (A.8)

$$\begin{aligned}\left[(\mathbf{A}\circ\mathbf{B})^H\operatorname{vec}(\mathbf{C})\right]_i &= \langle[\mathbf{A}\circ\mathbf{B}]_i, \operatorname{vec}(\mathbf{C})\rangle = (\mathbf{a}_i\otimes\mathbf{b}_i)^H\operatorname{vec}(\mathbf{C})\\ &\overset{\text{(A.7)}}{=} \operatorname{vec}(\mathbf{b}_i\mathbf{a}_i^T)^H\operatorname{vec}(\mathbf{C}) \overset{\text{(A.6)}}{=} \operatorname{tr}\left(\overline{\mathbf{a}}_i\mathbf{b}_i^H\mathbf{C}\right)\\ &= \operatorname{tr}\left(\mathbf{b}_i^H\mathbf{C}\overline{\mathbf{a}}_i\right) = \left[\mathbf{B}^H\mathbf{C}\overline{\mathbf{A}}\right]_{ii} = \left[\operatorname{diag}\left(\mathbf{B}^H\mathbf{C}\overline{\mathbf{A}}\right)\right]_i\end{aligned}$$

□

Proof. (A.9)

$$\begin{aligned}(\mathbf{A}\otimes\mathbf{B})^H(\mathbf{A}\otimes\mathbf{B})\operatorname{vec}(\mathbf{C}) &\overset{\text{(A.1)}}{=} \left(\mathbf{A}^H\otimes\mathbf{B}^H\right)(\mathbf{A}\otimes\mathbf{B})\operatorname{vec}(\mathbf{C})\\ &\overset{\text{(A.3)}}{=} \left[\left(\mathbf{A}^H\mathbf{A}\right)\otimes\left(\mathbf{B}^H\mathbf{B}\right)\right]\operatorname{vec}(\mathbf{C})\\ &\overset{\text{(A.4)}}{=} \operatorname{vec}(\mathbf{B}^H\mathbf{B}\mathbf{C}\mathbf{A}^T\overline{\mathbf{A}})\end{aligned}$$

□

Proof. (A.10)

$$\begin{aligned}(\mathbf{A}\circ\mathbf{B})^H(\mathbf{A}\circ\mathbf{B})\,\mathbf{c} &\overset{\text{(A.5)}}{=} (\mathbf{A}\circ\mathbf{B})^H\operatorname{vec}\left(\mathbf{B}\operatorname{diag}(\mathbf{c})\mathbf{A}^T\right)\\ &\overset{\text{(A.8)}}{=} \operatorname{diag}\left(\mathbf{B}^H\mathbf{B}\operatorname{diag}(\mathbf{c})\mathbf{A}^T\overline{\mathbf{A}}\right)\end{aligned}$$

□

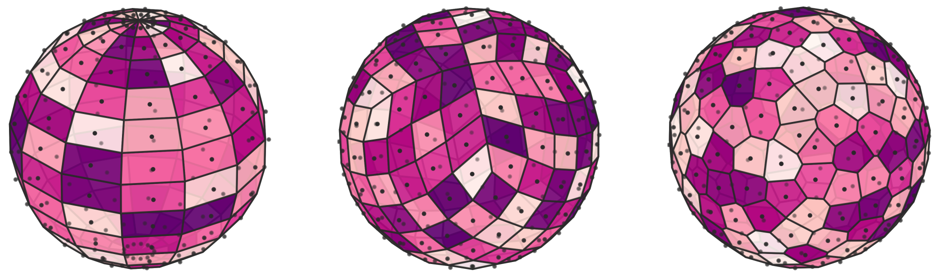
Proof. (A.11)

$$\begin{aligned}\left[(\mathbf{A}\circ\mathbf{B})^H(\mathbf{A}\circ\mathbf{B})\right]_{ij} &= \langle\mathbf{a}_i\otimes\mathbf{b}_i, \mathbf{a}_j\otimes\mathbf{b}_j\rangle \overset{\text{(A.7)}}{=} \langle\operatorname{vec}(\mathbf{b}_i\mathbf{a}_i^T), \operatorname{vec}(\mathbf{b}_j\mathbf{a}_j^T)\rangle\\ &\overset{\text{(A.6)}}{=} \operatorname{tr}\left(\overline{\mathbf{a}_i}\mathbf{b}_i^H\mathbf{b}_j\mathbf{a}_j^T\right) = \operatorname{tr}\left(\mathbf{b}_i^H\mathbf{b}_j\mathbf{a}_j^T\overline{\mathbf{a}_i}\right)\\ &= \langle\mathbf{b}_i, \mathbf{b}_j\rangle\langle\mathbf{a}_i, \mathbf{a}_j\rangle\end{aligned}$$

When put in matrix form, the above yields

$$(\mathbf{A}\circ\mathbf{B})^H(\mathbf{A}\circ\mathbf{B}) = \mathbf{A}^H\mathbf{A}\odot\mathbf{B}^H\mathbf{B}$$

□



(a) Equal-angle tessellation. (b) HEALPix tessellation. (c) Fibonacci tessellation.

Figure B.1: Examples of spherical maps defined over tessellations on the sphere, with an approximate resolution of $N = 200$ for each scheme. Cell centres are marked by black dots. The equal-angle tessellation fig. B.1a is obtained by pixelating the azimuth-elevation domain. The Fibonacci tessellation fig. B.1c is obtained by constructing the spherical Voronoi tessellation of the Fibonacci lattice (B.1). The HEALPix tessellation fig. B.1b finally, very popular in cosmology and astronomy, is constructed by hierarchical subdivision of the Voronoi cells of the dodecahedron vertices [4].

B Signal processing tools for spherical maps

B.1 Spherical tessellations and spherical maps

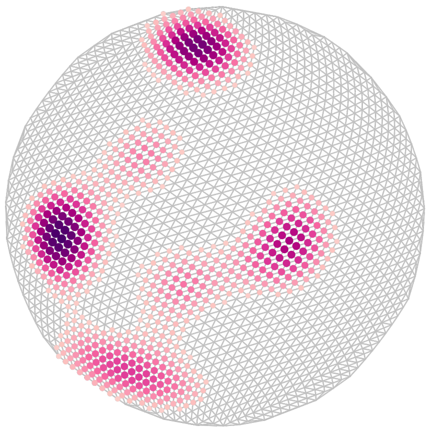
In the Euclidean setting, images are commonly stored, manipulated and displayed as multi-dimensional arrays. This convenient data-structure implicitly assumes a uniform partitioning of the image support into rectangular tiles or *pixels*. Spherical maps on the other hand are stored as arbitrarily ordered intensity vectors $\mathbf{x} \in \mathbb{R}^N$ associated to a *lattice* of directions $\Theta = \{\mathbf{r}_1, \ldots, \mathbf{r}_N\} \subset \mathbb{S}^2$ sampling irregularly the sphere. The pixels are obtained as polygonal Voronoi cells of the set Θ, and form a *spherical tessellation* [7]. For example, the equal-angle tessellation in fig. B.1a is obtained by pixelating the azimuth-elevation domain with rectangular tiles. This tessellation is unfortunately impractical since *irregular*: its polygonal cells have varying areas and shapes (large rectangular cells at the equator and elongated triangular cells near the poles). For practical purposes, regular tessellations –with equal-area and identical polygonal cells– are often preferred. The latter are however only available for fixed resolutions ($N = 4$, 6, 8, 12 and 20) and are obtained from the five Platonic solid vertices [14, Chapter 3]: the tetrahedron, cube, octahedron, dodecahedron and icosahedron. For arbitrary number of tiles, there exist many *near-regular* tilings of the sphere with almost uniform polygonal tiles [14, Chapter 3]. Among them, one counts notably the *HEALPix tessellation* [4], primarily used in cosmological applications [12]. It is obtained by hierarchically subdividing the Voronoi cells of the dodecahedron vertices into equal-area elements (see fig. B.1b). In this work, we consider moreover the *Fibonacci tessellation* [7]. Points in the Fibonacci lattice are arranged uniformly along a spiral pattern on the sphere linking the two poles (see fig. B.1c). The lattice can very easily be generated from the following formulae:

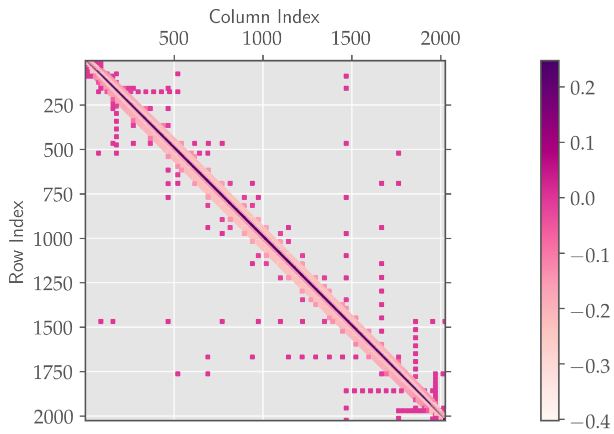
$$\begin{cases} \boldsymbol{r}_n = [\cos(\varphi_n)\sin(\theta_n), \sin(\varphi_n)\sin(\theta_n), \cos(\theta_n)]\,, \\ \text{where} \quad \varphi_n = 2\pi n\left(1 - \frac{2}{1+\sqrt{5}}\right) \quad \& \quad \theta_n = \arccos\left(1 - \frac{2n}{N}\right), \end{cases} \qquad n = 1, \ldots, N. \tag{B.1}$$

B.2 Spherical maps as signals on graphs

Multi-dimensional arrays are particularly convenient data-structures, since their *connectivity graph* is implicitly defined[3] and preserves the notion of spatial *locality*: Euclidean distances are proportional to index offsets. Labels of spherical maps on the other hand are often arbitrary, resulting in a fundamental mismatch between connectivity in the intensity vector $\mathbf{x}$ and locality in the lattice Θ. To properly account for the dependencies in $\mathbf{x}$ arising from the underlying domain geometry, one possibility [3, 12] is to define an explicit connectivity graph $\mathcal{G} = (\Theta, \mathcal{E}, \mathbf{W})$, where $\mathcal{E} \subset \Theta^2$ is an *edge set* defining neighbouring vertices in Θ and $\mathbf{W} \in \mathbb{R}^{N\times N}$ a *weighting matrix*, defining the *similarity* between two connected vertices (see fig. B.2). Given an arbitrary lattice Θ, the edge set

[3]For example, two entries (i, j) and (k, l) in a 2D-array are neighbours if $\max(|i - k|, |j - l|) = 1$.


(a) Spherical map on its connectivity graph.



(b) Sparsity pattern of the associated graph Laplacian (only $\simeq 0.4\%$ of nonzero values).

Figure B.2: Spherical map seen as a signal on a graph.

can for example be defined as its Delaunay triangulation, obtained from the convex-hull of Θ. The edge weights are moreover commonly defined as a function of the distance separating two vertices in the lattice Θ. In [12], the authors recommend the following weighting scheme:

$$[\mathbf{W}]_{nm} := \begin{cases} \exp\left(-\frac{\|\mathbf{r}_n - \mathbf{r}_m\|_2^2}{\rho^2}\right) & \text{if } (\mathbf{r}_n, \mathbf{r}_m) \in \mathcal{E}, \\ 0 & \text{otherwise,} \end{cases}$$

where $\rho > 0$ is given by $\rho = \frac{1}{|\mathcal{E}|}\sum_{(\mathbf{r}_n,\mathbf{r}_m)\in\mathcal{E}} \|\mathbf{r}_n - \mathbf{r}_m\|_2$. With this additional structure, a spherical map can be seen as a signal on a graph (see fig. B.2a), which can be processed by means of *graph signal processing* tools [17].

B.3 Discrete spherical convolutions

The DeepWave RNN described in (1) performs *filtering* operations at each layer, implemented by means of *graph convolution operators* [17]. Graph convolution operators generalise Euclidean convolution operators to irregular domains such as spherical tessellations where shifts are not naturally defined. By analogy to the Euclidean setting, graph convolutions are defined as operators diagonalised by the Fourier basis, obtained from the eigenvectors of the *Laplacian* of the graph $\mathcal{G}$. As explained in [17], there exist many possible definitions of the graph Laplacian. In this paper, we proceed as in [12] and choose to work with the *normalised Laplacian* given by:

$$\mathbf{L} := \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}, \tag{B.2}$$

where $\mathbf{I} \in \mathbb{R}^{N\times N}$ denotes the *identity matrix* and $\mathbf{D} \in \mathbb{R}^{N\times N}$ is a diagonal matrix defined as:

$$[\mathbf{D}]_{ii} = \sum_{n=1}^{N} [\mathbf{W}]_{in}\,.$$

The Laplacian operator (B.2) has many useful properties [17]. In particular, it is often extremely sparse[4] (see fig. B.2b) and its induced norm

$$\|\mathbf{x}\|_{\mathbf{L}} := \|\mathbf{L}^{1/2}\mathbf{x}\|_2^2 = \mathbf{x}^T\mathbf{L}\mathbf{x}, \tag{B.3}$$

can be seen as a measure of *smoothness* [17, Example 2] for a signal $\mathbf{x} \in \mathbb{R}^N$ defined on the vertex set Θ of $\mathcal{G}$. If $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$ is the eigendecomposition of $\mathbf{L}$, a filter $h(\mathbf{L}) \in \mathbb{R}^{N\times N}$ is a linear operator acting on a graph signal $\mathbf{x} \in \mathbb{R}^N$ as

$$h(\mathbf{L})\mathbf{x} := \mathbf{U}h(\mathbf{\Lambda})\mathbf{U}^T\mathbf{x},$$

[4]At least in the context of sparse connectivity graphs explored here.

for some function $h : \mathbb{R}_+ \to \mathbb{R}$. In this work, we will consider specific graph filters for which h is an order-K polynomial:

$$h(\mathbf{L}) = \sum_{k=0}^{K} \theta_k \mathbf{L}^k.$$

Such filters can indeed be shown [17, 12] to have finite support in the graph domain, with radius at most K vertices. Moreover, they can be efficiently implemented as a cascade of multiplications between the sparse matrix $\mathbf{L}$ and the vector $\mathbf{x}$ to be filtered. In particular, if $\mathbf{z}^0 = \mathbf{x}$ and $\mathbf{x}^0 = \theta_0 \mathbf{x}$, then the filtered vector $\tilde{\mathbf{x}}$ is given by the outcome $\mathbf{x}^K$ of the following recursion:

$$\begin{cases} \mathbf{z}^k & = \mathbf{L}\mathbf{z}^{k-1} \\ \mathbf{x}^k & = \mathbf{x}^{k-1} + \theta_k \mathbf{z}^k \end{cases}, \qquad k = 1, \ldots, K. \tag{B.4}$$

For stability reasons, we consider in practice an equivalent version of (B.4), as recommended in [17, 12]:

$$\begin{cases} \mathbf{z}^k & = 2\tilde{\mathbf{L}}\mathbf{z}^{k-1} - \mathbf{z}^{k-2} \\ \mathbf{x}^k & = \mathbf{x}^{k-1} + \tilde{\theta}_k \mathbf{z}^k \end{cases}, \qquad k = 2, \ldots, K, \tag{B.5}$$

with $\mathbf{x}^0 = \mathbf{z}^0 = \mathbf{x}$, $\mathbf{z}^1 = \tilde{\mathbf{L}}\mathbf{x}$ and $\mathbf{x}^1 = \tilde{\theta}_1 \mathbf{z}^1 + \tilde{\theta}_2 \mathbf{z}^2$. The weights $\{\tilde{\theta}_0, \ldots, \tilde{\theta}_K\} \subset \mathbb{R}$ in (B.5) are such that

$$\sum_{k=0}^{K} \theta_k \mathbf{L}^k = \sum_{k=0}^{K} \tilde{\theta}_k T_k(\tilde{\mathbf{L}}),$$

where $T_k : [-1, 1] \to \mathbb{R}$ are *Chebyshev polynomials* and $\tilde{\mathbf{L}}$ is the Laplacian with rescaled and shifted spectrum in the interval $[-1, 1]$ [12]:

$$\tilde{\mathbf{L}} = \frac{2}{\lambda_{max}} \mathbf{L} - \mathbf{I}.$$

Note that eq. (B.5) is obtained from the recursion formula defining the Chebyshev polynomials: $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$, with $T_1(x) = x$ and $T_0(x) = 1$.

C Background: far-field point source reconstruction problem 2

This section provides background material on classical objective function (2).

Narrow-band far-field acoustic perturbations are modeled in baseband-equivalent form [19] as realisations of a random function

$$\mathcal{S} = \left\{ S(\mathbf{r}) : \Omega \to \mathbb{C}, \ \mathbf{r} \in \mathbb{S}^2 \right\},$$

where $S(\mathbf{r})$ follows a zero-mean, spatially-uncorrelated complex Gaussian distribution of variance (i.e. intensity) $x(\mathbf{r}) = \mathbb{E}\left[S(\mathbf{r})S^*(\mathbf{r})\right] \in \mathbb{R}_+$. Concretely, when mapped onto a discrete spherical tesselation of resoluton N, $S(\mathbf{r})$ is given by

$$S(\mathbf{r}) = \sum_{q=1}^{N} \xi_q \delta(\mathbf{r} - \mathbf{r}_q), \tag{C.1}$$

where $\xi_q \sim \mathbb{C}\mathcal{N}(0, x_q)$ and $\Theta = \{\mathbf{r}_1, \ldots, \mathbf{r}_N\} \subset \mathbb{S}^2$ are the tesselation support points. The goal in the acoustic imaging problem is to estimate the intensity field $\mathbf{x} = [x_1, \ldots, x_N] \in \mathbb{R}_+^N$.

Knowledge of $\mathbf{x}$ can be gathered by observing realisations of $\mathcal{S}$ using a microphone array. Let $\mathbf{y} : \Omega \to \mathbb{C}^M$ denote the random samples measured at the output of an M-element array. Then microphone samples are linked to $\mathcal{S}$ through the relation [20]

$$[\mathbf{y}]_m = \int_{\mathbb{S}^2} S(\mathbf{r}) \exp\left(-j\frac{2\pi}{\lambda_0}\langle \mathbf{r}, \mathbf{p}_m \rangle\right) d\mathbf{r} \overset{\text{(C.1)}}{=} \sum_{q=1}^{N} \xi_q \exp\left(-j\frac{2\pi}{\lambda_0}\langle \mathbf{r}_q, \mathbf{p}_m \rangle\right), \tag{C.2}$$

where $\mathbf{p}_m \in \mathbb{R}^3$ denotes the position of the m-th microphone, and $\lambda_0 > 0$ is the wavelength of the impeding plane wave. The covariance matrix $\mathbf{\Sigma} \in \mathbb{C}^{M\times M}$ then directly links $\mathbf{x}$ to the measurements as

$$\begin{aligned}[\mathbf{\Sigma}]_{ij} = \mathbb{E}\left[[\mathbf{y}]_i\,[\mathbf{y}]_j^*\right] &= \sum_{q=1}^{N}\sum_{l=1}^{N}\mathbb{E}\left[\xi_q\xi_l^*\right]\exp\left(-j\frac{2\pi}{\lambda_0}\langle\mathbf{r}_q,\mathbf{p}_i\rangle\right)\exp\left(j\frac{2\pi}{\lambda_0}\langle\mathbf{r}_l,\mathbf{p}_j\rangle\right)\\ &= \sum_{q=1}^{N}[\mathbf{x}]_q\exp\left(j\frac{2\pi}{\lambda_0}\langle\mathbf{r}_q,\mathbf{p}_j-\mathbf{p}_i\rangle\right),\end{aligned}$$

which can be succintely described by matrix equation

$$\mathbf{\Sigma} = \mathbf{A}\,\mathrm{diag}(\mathbf{x})\mathbf{A}^H, \quad \mathbf{A} = \exp\left(-j\frac{2\pi}{\lambda_0}\mathbf{P}^T\mathbf{R}\right) \in \mathbb{C}^{M\times N}, \tag{C.3}$$

where $\mathbf{P} = [\mathbf{p}_1,\ldots,\mathbf{p}_M] \in \mathbb{R}^{3\times M}$ and $\mathbf{R} = [\mathbf{r}_1,\ldots,\mathbf{r}_N] \in \mathbb{R}^{3\times N}$. Replacing $\mathbf{\Sigma}$ in (C.3) by the empirical covariance matrix $\hat{\mathbf{\Sigma}} \in \mathbb{C}^{M\times M}$ then gives rise to the data-fidelity term

$$\frac{1}{2}\left\|\hat{\mathbf{\Sigma}} - \mathbf{A}\,\mathrm{diag}(\mathbf{x})\mathbf{A}^H\right\|_F^2. \tag{C.4}$$

Minimisation of (C.4) alone is not enough to obtain a unique minimiser due to limited data availability, i.e. $\binom{M}{2} \ll N$. To overcome this limitation and allow group-sparse reconstructions characteristic of acoustic scenes, (C.4) is augmented with the elastic-net regulariser [25]

$$\lambda\left[\gamma\left\|\mathbf{x}\right\|_1 + (1-\gamma)\left\|\mathbf{x}\right\|_2^2\right]. \tag{C.5}$$

Combining (C.4) and (C.5) leads to (2).

D Derivation: proximal gradient descent for elastic-net problem 3

This section shows how to obtain proximal iteration (4) from (3).

Recall that the sound intensity map is obtained by solving the convex optimisation problem:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}\in\mathbb{R}_+^N} f(\mathbf{x}) + g(\mathbf{x}), \tag{D.1}$$

$$f(\mathbf{x}) = \frac{1}{2}\left\|\hat{\mathbf{\Sigma}} - \mathbf{A}\,\mathrm{diag}(\mathbf{x})\mathbf{A}^H\right\|_F^2 \overset{(A.5)}{=} \frac{1}{2}\left\|\mathrm{vec}(\hat{\mathbf{\Sigma}}) - \left(\overline{\mathbf{A}}\circ\mathbf{A}\right)\mathbf{x}\right\|_2^2, \tag{D.2}$$

$$g(\mathbf{x}) = \lambda\left[\gamma\left\|\mathbf{x}\right\|_1 + (1-\gamma)\left\|\mathbf{x}\right\|_2^2\right], \tag{D.3}$$

where g is an elastic-net regularizer with $\lambda \geq 0$ and $\gamma \in]0,1[$.

Proximal gradient descent (PGD) is a fixed-point method to solve problems of the form (D.1) where f, g are closed proper convex with f differentiable. It consists of iterating the proximal update equation until convergence:

$$\mathbf{x}^k = \mathrm{prox}_{\alpha g}\left(\mathbf{x}^{k-1} - \alpha\nabla f(\mathbf{x}^{k-1})\right), \tag{D.4}$$

where $\alpha > 0$ is the step size and $\mathrm{prox}_{\alpha g}$ is the proximal operator associated with (D.3), given by (see proof below):

$$\mathrm{prox}_{\alpha g}(\mathbf{x}) = \arg\min_{\mathbf{u}\in\mathbb{R}_+^N} g(\mathbf{u}) + \frac{1}{2\alpha}\left\|\mathbf{u}-\mathbf{x}\right\|_2^2, \tag{D.5}$$

$$= \mathrm{ReLu}\left(\frac{\mathbf{x}-\lambda\alpha\gamma}{2\lambda\alpha(1-\gamma)+1}\right), \qquad \forall\mathbf{x}\in\mathbb{R}^N. \tag{D.6}$$

The quantity $\nabla f \in \mathbb{R}^N$ finally is obtained using the rules of vector calculus [13]:

$$\begin{aligned}\nabla f(\mathbf{x}) &= \left\{\frac{\partial}{\partial\mathbf{x}}\left[\mathrm{vec}(\hat{\mathbf{\Sigma}}) - \left(\overline{\mathbf{A}}\circ\mathbf{A}\right)\mathbf{x}\right]\right\}\cdot\left[\mathrm{vec}(\hat{\mathbf{\Sigma}}) - \left(\overline{\mathbf{A}}\circ\mathbf{A}\right)\mathbf{x}\right]\\ &= \left(\overline{\mathbf{A}}\circ\mathbf{A}\right)^H\left[\left(\overline{\mathbf{A}}\circ\mathbf{A}\right)\mathbf{x} - \mathrm{vec}(\hat{\mathbf{\Sigma}})\right].\end{aligned} \tag{D.7}$$

Combining (D.4), (D.6) and (D.7) leads to (4).

Proof: (Analytic expression for $\mathrm{prox}_{\alpha g}$*).* Replacing (D.3) in (D.5), we get for $\mathbf{x} \in \mathbb{R}^N$:

$$
\begin{aligned}
\mathrm{prox}_{\alpha g}(\mathbf{x}) &= \arg\min_{\mathbf{u}\in\mathbb{R}_+^N} \lambda \left[\gamma \|\mathbf{u}\|_1 + (1-\gamma)\|\mathbf{u}\|_2^2\right] + \frac{1}{2\alpha}\|\mathbf{u}-\mathbf{x}\|_2^2 \\
&= \arg\min_{(u_1,\ldots,u_N)\in\mathbb{R}_+^N} \sum_{n=1}^N \lambda\left[\gamma|u_n| + (1-\gamma)u_n^2\right] + \frac{1}{2\alpha}(u_n - x_n)^2 \\
&= \arg\min_{(u_1,\ldots,u_N)\in\mathbb{R}_+^N} \sum_{n=1}^N \lambda\left[\gamma u_n + (1-\gamma)u_n^2\right] + \frac{1}{2\alpha}\left[u_n^2 + x_n^2 - 2u_n x_n\right] \\
&= \arg\min_{(u_1,\ldots,u_N)\in\mathbb{R}_+^N} \sum_{n=1}^N \varphi_n(u_n). \qquad \text{(D.8)}
\end{aligned}
$$

Notice that (D.8) is the sum of N independent objective functionals, hence each can be independently minimised. (We drop the subscript of φ_n below for simplicity.) Let $\hat{u}$ be the minimiser:[5]

$$
\hat{u} = \arg\min_{u\geq 0}\varphi(u) = \arg\min_{u\geq 0}\lambda\left[\gamma u + (1-\gamma)u^2\right] + \frac{1}{2\alpha}\left[u^2 + x^2 - 2ux\right], \qquad \text{(D.9)}
$$

for some fixed $x \in \mathbb{R}$. Then two cases can occur:

- $x \leq 0$: the objective functional being composed of positive terms only, any $\hat{u} > 0$ will increase the objective. Therefore $\hat{u} = 0$.
- $x > 0$: In this case the Karush Kuhn Tucker (KKT) conditions [18, 2] tell us that $\hat{u}$ is a minimizer of (D.9) if

$$
\begin{aligned}
\hat{u}\varphi'(\hat{u}) &= 0 \\
\varphi'(\hat{u}) &\geq 0 \quad \text{if } \hat{u} = 0.
\end{aligned}
$$

  Plugging $\varphi'(u) = \lambda\gamma + \left(2\lambda(1-\gamma) + \alpha^{-1}\right)u - \alpha^{-1}x$ and solving the above yields

$$
\hat{u} = \begin{cases} \frac{x-\lambda\alpha\gamma}{2\lambda\alpha(1-\gamma)+1} & x > \lambda\alpha\gamma, \\ 0 & x \leq \lambda\alpha\gamma \end{cases}.
$$

Both cases can be written in short as

$$
\hat{u} = \arg\min_{u\geq 0}\varphi(u) = \left[\frac{x-\lambda\alpha\gamma}{2\lambda\alpha(1-\gamma)+1}\right]_+, \qquad \forall x \in \mathbb{R},
$$

leading to an element-wise proximal operator of the form

$$
\mathrm{prox}_{\alpha g}(\mathbf{x}) = \left[\frac{\mathbf{x}-\lambda\alpha\gamma}{2\lambda\alpha(1-\gamma)+1}\right]_+ = \mathrm{ReLu}\left(\frac{\mathbf{x}-\lambda\alpha\gamma}{2\lambda\alpha(1-\gamma)+1}\right), \qquad \forall \mathbf{x} \in \mathbb{R}^N.
$$

□

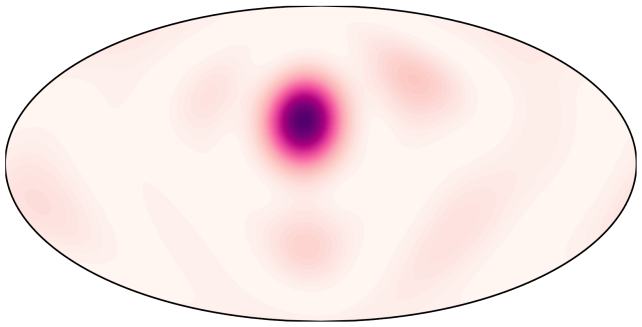
E Proof: proposition 1

In this section, we prove proposition 1 of the main paper:

Proposition. *Consider a* spherical microphone array*, with diameter D and microphone directions* $\{\tilde{\mathbf{p}}_1, \ldots, \tilde{\mathbf{p}}_M\} \subset \mathbb{S}^2$*, forming a* near-regular tessellation *of the sphere. Then, we have*

$$
\left[I - \alpha\left(\overline{\mathbf{A}}\circ\mathbf{A}\right)^H\left(\overline{\mathbf{A}}\circ\mathbf{A}\right)\right]_{ij} \simeq \left[\delta_{ij} - \alpha M^2\,\mathrm{sinc}^2\left(\frac{D}{\lambda_0}\|\mathbf{r}_i - \mathbf{r}_j\|\right)\right], \ \forall i,j \in \{1,\ldots,N\} \qquad \text{(E.1)}
$$

where λ_0 is the wavelength*, δ_{ij} denotes the* Kronecker delta *and* $\mathrm{sinc}(x) := \sin(\pi x)/\pi x$ *is the* cardinal sine*. Moreover, the approximation* (E.1) *is* extremely good *for $M \geq 3\lfloor\frac{2\pi D}{\lambda_0}\rfloor^2$.*


(a) Beamshape of the Pyramic array.

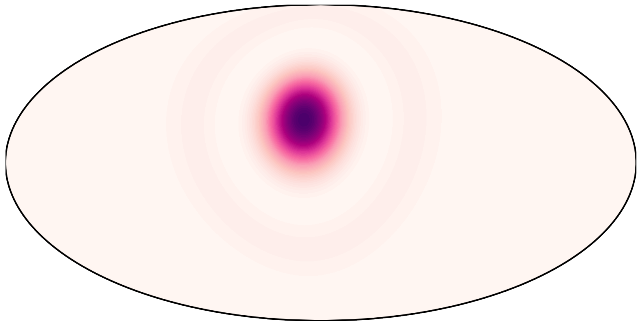

(b) Approximate beamshape obtained with (E.2).

Figure E.1: Accuracy of approximation (E.2) for the Pyramic array [16] ($D = 30$[cm], $M = 48$) at 1 kHz.

Proof. To prove (E.1), it is sufficient to show that

$$\left[\left(\overline{\mathbf{A}} \circ \mathbf{A}\right)^H \left(\overline{\mathbf{A}} \circ \mathbf{A}\right)\right]_{ij} \simeq M^2 \operatorname{sinc}^2\left(\frac{D}{\lambda_0}\|\mathbf{r}_i - \mathbf{r}_j\|\right). \tag{E.2}$$

To this end, we first use (A.11) and obtain:

$$\left(\overline{\mathbf{A}} \circ \mathbf{A}\right)^H \left(\overline{\mathbf{A}} \circ \mathbf{A}\right) = \left|\mathbf{A}^H\mathbf{A}\right|^{\odot 2}. \tag{E.3}$$

For a spherical array with diameter D and microphone directions $\{\tilde{\mathbf{p}}_1, \ldots, \tilde{\mathbf{p}}_M\} \subset \mathbb{S}^2$, we get moreover from the definition of the steering matrix that:

$$[\mathbf{A}^H\mathbf{A}]_{ij} = \sum_{m=1}^{M} \exp\left(j\frac{\pi D}{\lambda_0}\langle \mathbf{r}_i - \mathbf{r}_j, \tilde{\mathbf{p}}_m\rangle\right), \qquad i, j = 1, \ldots, N. \tag{E.4}$$

Since the microphone directions are assumed to form a near-regular tessellation over the sphere (such as the Fibonacci or HEALPix tessellations discussed in appendix B.1), we can interpret (E.4) as a quadrature rule on the sphere, yielding:

$$\frac{4\pi}{M}\sum_{m=1}^{M} \exp\left(j\frac{\pi D}{\lambda_0}\langle \mathbf{r}_i - \mathbf{r}_j, \tilde{\mathbf{p}}_m\rangle\right) \simeq \int_{\mathbb{S}^2} \exp\left(j\frac{\pi D}{\lambda_0}\langle \mathbf{r}_i - \mathbf{r}_j, \tilde{\mathbf{p}}\rangle\right) d\tilde{\mathbf{p}} \tag{E.5}$$

$$= 4\pi \operatorname{sinc}\left(\frac{D}{\lambda_0}\|\mathbf{r}_i - \mathbf{r}_j\|\right), \tag{E.6}$$

where the second equality (E.6) follows from the result on [21, p. 154]. From (E.6), (E.4) and (E.3) we obtain (E.2) from which (E.1) trivially follows.
Regarding the quality of the approximation (E.2) finally, we use the approximate bandlimitedness of complex plane-waves in the spherical domain [14, Chapter 2]. Indeed, quadrature rules such as

[5]which exists since the optimisation problem is convex.

(E.5) are almost exact for bandlimited functions [14, Chapter 3], provided a high-enough number of quadrature points M. For example, a function with spherical harmonic bandwidth $L \in \mathbb{N}$ is extremely well approximated by the HEALPix quadrature rule for $M \geq 3L^2$ [4]. In our case, the *plane-wave expansion* [14, Chapter 2] gives us

$$\exp\left(j\frac{\pi D}{\lambda_0}\langle \mathbf{r}_i - \mathbf{r}_j, \tilde{\mathbf{p}}\rangle\right) = 4\pi \sum_{l=0}^{+\infty}\sum_{k=-l}^{l} j^l(2l+1)j_l\left(\frac{\pi D}{\lambda_0}\|\mathbf{r}_i - \mathbf{r}_j\|_2\right)\overline{Y_l^k}(\tilde{\mathbf{r}}_{ij})Y_l^k(\tilde{\mathbf{p}}),$$

where j_l are *spherical Bessel functions*, Y_l^k *spherical harmonics*, and $\tilde{\mathbf{r}}_{ij} = (\mathbf{r}_i - \mathbf{r}_j)/\|\mathbf{r}_i - \mathbf{r}_j\|_2^2$ [14]. Since $j_l(x) \simeq 0$ for $l \geq x$ [14, Chapter 2] we have hence that complex plane-waves are approximately bandlimited with bandwidth $L = \lfloor \frac{\pi D}{\lambda_0}\|\mathbf{r}_i - \mathbf{r}_j\|_2 \rfloor \leq \lfloor \frac{2\pi D}{\lambda_0} \rfloor$. As a result, choosing $M \geq 3\lfloor \frac{2\pi D}{\lambda_0} \rfloor^2$ makes the approximation (E.2) very accurate. □

While proven for spherical arrays only, approximation (E.2) (and hence (E.1)) remains quite accurate in practice, even for non spherical microphone arrays such as the Pyramic array used in our real-world experiments [16]. In fig. E.1, we investigated visually the quality of the approximation (E.2) for the Pyramic array at 1 kHZ. To this end, we plotted a row of $\left|\mathbf{A}^H\mathbf{A}\right|^{\odot 2}$ (which corresponds to the beamshape of the instrument for a particular direction [23]) with and without approximation. We observe that the approximation is already very good, even if the Pyramic array possesses only $M = 48$ microphones against the 90 required by proposition 1 for an optimal approximation accuracy at this frequency.

F Network gradient evaluation

This section shows how to obtain derivatives of data-fidelity term $\mathcal{L}_t$ from eq. (9) w.r.t. network parameters $\boldsymbol{\theta}, \mathbf{B}, \boldsymbol{\tau}$.[6]

F.1 Problem statement

Recall that

$$\nabla\mathcal{L}(\boldsymbol{\Omega}) = \left\{\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}} \in \mathbb{R}^{K+1}, \ \frac{\partial \mathcal{L}}{\partial \mathbf{B}} \in \mathbb{C}^{M\times N}, \ \frac{\partial \mathcal{L}}{\partial \boldsymbol{\tau}} \in \mathbb{R}^{N}\right\},$$

$$\mathcal{L}(\boldsymbol{\Omega}) = \frac{1}{2}\frac{\left\|\hat{\mathbf{x}} - \mathbf{x}^L(\boldsymbol{\Omega})\right\|_2^2}{\|\hat{\mathbf{x}}\|_2^2}, \tag{F.1}$$

where $\mathbf{x}^L(\boldsymbol{\Omega}) \in \mathbb{R}_+^N$ is given by recurrence relation (1):

$$\begin{aligned}
\mathbf{x}^l(\boldsymbol{\Omega}) &= \sigma\left[P_{\boldsymbol{\theta}}(\mathbf{L})\mathbf{x}^{l-1} + \left(\overline{\mathbf{B}} \circ \mathbf{B}\right)^H \operatorname{vec}(\hat{\boldsymbol{\Sigma}}) - \boldsymbol{\tau}\right] && \text{(F.2)}\\
&= \sigma\left[\mathbf{u}^l + \mathbf{w} - \boldsymbol{\tau}\right] && \text{(F.3)}\\
&= \sigma\left[\mathbf{s}^l\right], \qquad l = 1, \ldots, L && \text{(F.4)}
\end{aligned}$$

with $\mathbf{x}^0 \in \mathbb{R}_+^N$ some arbitrary constant, $\sigma : \mathbb{R} \to \mathbb{R}$ a point-wise non-linearity, and $P_{\boldsymbol{\theta}}(\mathbf{L}) = \sum_{k=0}^K \theta_k T_k(\mathbf{L})$ a polynomial filter of order K expressed in terms of Chebychev polynomials.

$\nabla\mathcal{L}$ can be efficiently evaluated using *reverse-mode algorithmic differentiation*[1, 6] in a two-stage process:

- Forward pass: evaluate eq. (F.1) while storing all intermediate values $\mathbf{w}, \boldsymbol{\tau}, \left\{\mathbf{s}^l\right\}_{l=1,\ldots,L}$;
- Backward pass: walk the computational graph (fig. F.1) backwards to evaluate derivatives w.r.t. $\boldsymbol{\theta}, \mathbf{B}, \boldsymbol{\tau}$.

[6]For notational simplicity, this section drops the subscript in $\mathcal{L}_t$.

$$\begin{tikzcd}
\boldsymbol{\tau} \arrow[rrd] \arrow[rrrrrd] & & & & & & & \\
\mathbf{x}^0 \arrow[r] & \mathbf{u}^1 \arrow[r] & \mathbf{s}^1 \arrow[r] & \mathbf{x}^1 \arrow[r, dashed] & \mathbf{u}^L \arrow[r] & \mathbf{s}^L \arrow[r] & \mathbf{x}^L \arrow[r] & \mathcal{L} \\
\boldsymbol{\theta} \arrow[ru] \arrow[rrrru] & & & & & & & \hat{\mathbf{x}} \arrow[u] \\
\mathbf{B} \arrow[r] & \mathbf{w} \arrow[ruu] \arrow[rrrruu] & & & & & & \\
\hat{\boldsymbol{\Sigma}} \arrow[ru] & & & & & & &
\end{tikzcd}$$

Figure F.1: L-layer computational graph of $\mathcal{L}$.

F.2 Conventions

- If $\mathbf{u} \in \mathbb{R}^N$, $\mathbf{v} \in \mathbb{R}^M$, the *Jacobian matrix* $\frac{\partial \mathbf{u}}{\partial \mathbf{v}} \in \mathbb{R}^{N \times M}$ is defined as
$$\left[\frac{\partial \mathbf{u}}{\partial \mathbf{v}}\right]_{ij} = \frac{\partial \left[\mathbf{u}\right]_i}{\partial \left[\mathbf{v}\right]_j}.$$
Gradients of scalar-valued functions are therefore row vectors.
- If $\mathbf{u} \in \mathbb{R}^N$, $\mathbf{V} \in \mathbb{R}^{M \times Q}$, the *Jacobian tensor* $\frac{\partial \mathbf{u}}{\partial \mathbf{V}} \in \mathbb{R}^{N \times M \times Q}$ is defined as
$$\left[\frac{\partial \mathbf{u}}{\partial \mathbf{V}}\right]_{ijk} = \frac{\partial \left[\mathbf{u}\right]_i}{\partial \left[\mathbf{V}\right]_{jk}}.$$

F.3 Common intermediate gradients

$$\left[\frac{\partial \mathcal{L}}{\partial \mathbf{x}^L}\right]_i = \frac{\partial \mathcal{L}}{\partial \left[\mathbf{x}^L\right]_i} = \left[\frac{\mathbf{x}^L - \hat{\mathbf{x}}}{\|\hat{\mathbf{x}}\|_2^2}\right]_i \tag{F.5}$$

$$\left[\frac{\partial \mathbf{x}^l}{\partial \mathbf{s}^l}\right]_{ij} = \frac{\partial \left[\mathbf{x}^l\right]_i}{\partial \left[\mathbf{s}^l\right]_j} = \delta_{i-j}\sigma'\left(\left[\mathbf{s}^l\right]_j\right) = \left[\operatorname{diag}\left(\sigma'\left(\mathbf{s}^l\right)\right)\right]_{ij}, \qquad l = 1, \ldots, L \tag{F.6}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{s}^l} = \frac{\partial \mathcal{L}}{\partial \mathbf{x}^l}\frac{\partial \mathbf{x}^l}{\partial \mathbf{s}^l} \overset{\text{(F.6)}}{=} \frac{\partial \mathcal{L}}{\partial \mathbf{x}^l}\operatorname{diag}\left(\sigma'\left(\mathbf{s}^l\right)\right), \qquad l = 1, \ldots, L \tag{F.7}$$

$$\left[\frac{\partial \mathbf{s}^l}{\partial \mathbf{u}^l}\right]_{ij} = \frac{\partial \left[\mathbf{s}^l\right]_i}{\partial \left[\mathbf{u}^l\right]_j} = \frac{\partial}{\partial \left[\mathbf{u}^l\right]_j}\left[\mathbf{u}^l + \mathbf{w} - \boldsymbol{\tau}\right]_i = \delta_{i-j} = \left[\mathbf{I}_N\right]_{ij} \tag{F.8}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{u}^l} = \frac{\partial \mathcal{L}}{\partial \mathbf{s}^l}\frac{\partial \mathbf{s}^l}{\partial \mathbf{u}^l} \overset{\text{(F.8)}}{=} \frac{\partial \mathcal{L}}{\partial \mathbf{s}^l}, \qquad l = 1, \ldots, L \tag{F.9}$$

$$\left[\frac{\partial \mathbf{u}^l}{\partial \mathbf{x}^{l-1}}\right]_{ij} = \left[\frac{\partial}{\partial \mathbf{x}^{l-1}}P_{\boldsymbol{\theta}}(\mathbf{L})\mathbf{x}^{l-1}\right]_{ij} = \left[P_{\boldsymbol{\theta}}(\mathbf{L})\right]_{ij}, \qquad l = 1, \ldots, L \tag{F.10}$$

$$\left[\frac{\partial \mathbf{s}^l}{\partial \mathbf{w}}\right]_{ij} = \frac{\partial \left[\mathbf{s}^l\right]_i}{\partial \left[\mathbf{w}\right]_j} = \frac{\partial}{\partial \left[\mathbf{w}\right]_j}\left[\mathbf{u}^l + \mathbf{w} - \boldsymbol{\tau}\right]_i = \delta_{i-j} = \left[\mathbf{I}_N\right]_{ij} \tag{F.11}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \sum_{l=1}^{L}\frac{\partial \mathcal{L}}{\partial \mathbf{s}^l}\frac{\partial \mathbf{s}^l}{\partial \mathbf{w}} \overset{\text{(F.11)}}{=} \sum_{l=1}^{L}\frac{\partial \mathcal{L}}{\partial \mathbf{s}^l} \tag{F.12}$$

F.4 $\partial\mathcal{L}/\partial\theta$

$$\left[\frac{\partial \mathbf{u}^l}{\partial \boldsymbol{\theta}}\right]_{ij} = \frac{\partial}{\partial\left[\boldsymbol{\theta}\right]_j}\sum_{k=0}^{K}[\boldsymbol{\theta}]_k\left[T_k(\mathbf{L})\mathbf{x}^{l-1}\right]_i = \left[T_j(\mathbf{L})\mathbf{x}^{l-1}\right]_i, \qquad l=1,\ldots,L \tag{F.13}$$

$$\left[\frac{\partial\mathcal{L}}{\partial\boldsymbol{\theta}}\right]_i = \sum_{l=1}^{L}\left[\frac{\partial\mathcal{L}}{\partial\mathbf{u}^l}\frac{\partial\mathbf{u}^l}{\partial\boldsymbol{\theta}}\right]_i \underset{\text{(F.13)}}{\overset{\text{(F.9)}}{=}} \sum_{i=1}^{L}\frac{\partial\mathcal{L}}{\partial\mathbf{s}^l}T_i(\mathbf{L})\mathbf{x}^{l-1}, \qquad i=0,\ldots,K \tag{F.14}$$

F.5 $\partial\mathcal{L}/\partial\mathbf{B}$

$\frac{\partial\mathcal{L}}{\partial\mathbf{B}}$ can be obtained by evaluating $\frac{\partial\mathcal{L}}{\partial\mathbf{w}}\frac{\partial\mathbf{w}}{\partial\mathbf{B}}$, but $\frac{\partial\mathbf{w}}{\partial\mathbf{B}}\in\mathbb{C}^{N\times M\times N}$ is difficult to obtain directly. We therefore proceed in multiple steps:

1. Decompose $\mathbf{w}$ as $(\mathbf{w}_1+\mathbf{w}_2+\mathbf{w}_3)$ and express $\{\mathbf{w}_k\}_{k=1,2,3}$ explicitly in terms of $\hat{\boldsymbol{\Sigma}}_R,\hat{\boldsymbol{\Sigma}}_I,\mathbf{B}_R,\mathbf{B}_I$:

$$\begin{aligned}
\mathbf{w} &= \left(\overline{\mathbf{B}}\circ\mathbf{B}\right)^H\operatorname{vec}(\hat{\boldsymbol{\Sigma}}) \overset{\text{(A.8)}}{=} \operatorname{diag}\left(\mathbf{B}^H\hat{\boldsymbol{\Sigma}}\mathbf{B}\right) \\
&= \operatorname{diag}\left(\left[\mathbf{B}_R+j\mathbf{B}_I\right]^H\left[\hat{\boldsymbol{\Sigma}}_R+j\hat{\boldsymbol{\Sigma}}_I\right]\left[\mathbf{B}_R+j\mathbf{B}_I\right]\right) \\
&= \operatorname{diag}\left(\left[\mathbf{B}_R^T-j\mathbf{B}_I^T\right]\left[\hat{\boldsymbol{\Sigma}}_R+j\hat{\boldsymbol{\Sigma}}_I\right]\left[\mathbf{B}_R+j\mathbf{B}_I\right]\right) \\
&= \operatorname{diag}\left(\mathbf{B}_R^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_R+\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_I+\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_I\mathbf{B}_R-\mathbf{B}_R^T\hat{\boldsymbol{\Sigma}}_I\mathbf{B}_I\right) \\
&\quad + j\operatorname{diag}\left(\mathbf{B}_R^T\hat{\boldsymbol{\Sigma}}_I\mathbf{B}_R+\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_I\mathbf{B}_I+\mathbf{B}_R^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_I-\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_R\right) \\
&\overset{\mathbf{w}\in\mathbb{R}_+^N}{=} \operatorname{diag}\left(\mathbf{B}_R^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_R+\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_I+\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_I\mathbf{B}_R\right)-\operatorname{diag}\left(\mathbf{B}_R^T\hat{\boldsymbol{\Sigma}}_I\mathbf{B}_I\right) \\
&= \operatorname{diag}\left(\mathbf{B}_R^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_R+\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_I+\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_I\mathbf{B}_R\right)-\operatorname{diag}\left(\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_I^T\mathbf{B}_R\right) \\
&\overset{\hat{\boldsymbol{\Sigma}}_I=-\hat{\boldsymbol{\Sigma}}_I^T}{=} \operatorname{diag}\left(\mathbf{B}_R^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_R+\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_I+\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_I\mathbf{B}_R\right)+\operatorname{diag}\left(\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_I\mathbf{B}_R\right) \\
&= \operatorname{diag}\left(\mathbf{B}_R^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_R+\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_I+2\mathbf{B}_I^T\hat{\boldsymbol{\Sigma}}_I\mathbf{B}_R\right) \\
&\overset{\text{(A.8)}}{=} \underbrace{\left(\mathbf{B}_R\circ\mathbf{B}_R\right)^T\operatorname{vec}(\hat{\boldsymbol{\Sigma}}_R)}_{\mathbf{w}_1}+\underbrace{\left(\mathbf{B}_I\circ\mathbf{B}_I\right)^T\operatorname{vec}(\hat{\boldsymbol{\Sigma}}_R)}_{\mathbf{w}_2}+\underbrace{2\left(\mathbf{B}_R\circ\mathbf{B}_I\right)^T\operatorname{vec}(\hat{\boldsymbol{\Sigma}}_I)}_{\mathbf{w}_3}.
\end{aligned} \tag{F.15}$$

2. Derive analytic forms for $\left\{\frac{\partial\mathbf{w}_k}{\partial\mathbf{B}_{R/I}}\right\}_{k=1,2,3}$:

$$\begin{aligned}
\left[\frac{\partial\mathbf{w}_1}{\partial\mathbf{B}_R}\right]_{ijk} &= \frac{\partial\left[\mathbf{w}_1\right]_i}{\partial\left[\mathbf{B}_R\right]_{jk}} \overset{\text{(A.8)}}{=} \frac{\partial}{\partial\left[\mathbf{B}_R\right]_{jk}}\left[\operatorname{diag}\left(\mathbf{B}_R^T\hat{\boldsymbol{\Sigma}}_R\mathbf{B}_R\right)\right]_i \\
&= \frac{\partial}{\partial\left[\mathbf{B}_R\right]_{jk}}(\mathbf{b}_i^R)^T\hat{\boldsymbol{\Sigma}}_R\mathbf{b}_i^R \\
&= \delta_{i-k}\frac{\partial}{\partial\left[\mathbf{B}_R\right]_{jk}}(\mathbf{b}_k^R)^T\hat{\boldsymbol{\Sigma}}_R\mathbf{b}_k^R \\
&= \delta_{i-k}\sum_{q=1}^{M}\sum_{g=1}^{M}\left[\hat{\boldsymbol{\Sigma}}_R\right]_{qg}\frac{\partial}{\partial\left[\mathbf{B}_R\right]_{jk}}\left\{\left[\mathbf{B}_R\right]_{qk}\left[\mathbf{B}_R\right]_{gk}\right\} \\
&= \delta_{i-k}\sum_{q=1}^{M}\left(\left[\hat{\boldsymbol{\Sigma}}_R\right]_{jq}+\left[\hat{\boldsymbol{\Sigma}}_R\right]_{qj}\right)\left[\mathbf{B}_R\right]_{qk} \\
&\overset{\hat{\boldsymbol{\Sigma}}_R=\hat{\boldsymbol{\Sigma}}_R^T}{=} 2\delta_{i-k}(\mathbf{b}_k^R)^T\sigma_j^R
\end{aligned} \tag{F.16}$$

$$\left[\frac{\partial \mathbf{w}_2}{\partial \mathbf{B}_I}\right]_{ijk} = \frac{\partial \left[\mathbf{w}_2\right]_i}{\partial \left[\mathbf{B}_I\right]_{jk}} \overset{\text{(A.8)}}{=} \frac{\partial}{\partial \left[\mathbf{B}_I\right]_{jk}} \left[\text{diag}\left(\mathbf{B}_I^T \hat{\mathbf{\Sigma}}_R \mathbf{B}_I\right)\right]_i \tag{F.17}$$
$$= \frac{\partial}{\partial \left[\mathbf{B}_I\right]_{jk}} (\mathbf{b}_i^I)^T \hat{\mathbf{\Sigma}}_R \mathbf{b}_i^I$$
$$= \delta_{i-k} \frac{\partial}{\partial \left[\mathbf{B}_I\right]_{jk}} (\mathbf{b}_k^I)^T \hat{\mathbf{\Sigma}}_R \mathbf{b}_k^I$$
$$= \delta_{i-k} \sum_{q=1}^{M} \sum_{g=1}^{M} \left[\hat{\mathbf{\Sigma}}_R\right]_{qg} \frac{\partial}{\partial \left[\mathbf{B}_I\right]_{jk}} \left\{\left[\mathbf{B}_I\right]_{qk} \left[\mathbf{B}_I\right]_{gk}\right\}$$
$$= \delta_{i-k} \sum_{q=1}^{M} \left(\left[\hat{\mathbf{\Sigma}}_R\right]_{jq} + \left[\hat{\mathbf{\Sigma}}_R\right]_{qj}\right) \left[\mathbf{B}_I\right]_{qk}$$
$$\overset{\hat{\mathbf{\Sigma}}_R = \hat{\mathbf{\Sigma}}_R^T}{=} 2\delta_{i-k} (\mathbf{b}_k^I)^T \sigma_j^R$$

$$\left[\frac{\partial \mathbf{w}_3}{\partial \mathbf{B}_R}\right]_{ijk} = \frac{\partial \left[\mathbf{w}_3\right]_i}{\partial \left[\mathbf{B}_R\right]_{jk}} \overset{\text{(A.8)}}{=} 2\frac{\partial}{\partial \left[\mathbf{B}_R\right]_{jk}} \left[\text{diag}\left(\mathbf{B}_I^T \hat{\mathbf{\Sigma}}_I \mathbf{B}_R\right)\right]_i \tag{F.18}$$
$$= 2\frac{\partial}{\partial \left[\mathbf{B}_R\right]_{jk}} (\mathbf{b}_i^I)^T \hat{\mathbf{\Sigma}}_I \mathbf{b}_i^R$$
$$= 2\delta_{i-k} \frac{\partial}{\partial \left[\mathbf{B}_R\right]_{jk}} (\mathbf{b}_k^I)^T \hat{\mathbf{\Sigma}}_I \mathbf{b}_k^R$$
$$= 2\delta_{i-k} (\mathbf{b}_k^I)^T \sigma_j^I$$

$$\left[\frac{\partial \mathbf{w}_3}{\partial \mathbf{B}_I}\right]_{ijk} = \frac{\partial \left[\mathbf{w}_3\right]_i}{\partial \left[\mathbf{B}_I\right]_{jk}} \overset{\text{(A.8)}}{=} 2\frac{\partial}{\partial \left[\mathbf{B}_I\right]_{jk}} \left[\text{diag}\left(\mathbf{B}_I^T \hat{\mathbf{\Sigma}}_I \mathbf{B}_R\right)\right]_i \tag{F.19}$$
$$= 2\frac{\partial}{\partial \left[\mathbf{B}_I\right]_{jk}} (\mathbf{b}_i^I)^T \hat{\mathbf{\Sigma}}_I \mathbf{b}_i^R$$
$$= 2\delta_{i-k} \frac{\partial}{\partial \left[\mathbf{B}_I\right]_{jk}} (\mathbf{b}_k^I)^T \hat{\mathbf{\Sigma}}_I \mathbf{b}_k^R$$
$$\overset{\hat{\mathbf{\Sigma}}_I = -\hat{\mathbf{\Sigma}}_I^T}{=} -2\delta_{i-k} \frac{\partial}{\partial \left[\mathbf{B}_I\right]_{jk}} (\mathbf{b}_k^R)^T \hat{\mathbf{\Sigma}}_I \mathbf{b}_k^I$$
$$= -2\delta_{i-k} (\mathbf{b}_k^R)^T \sigma_j^I$$

3. Combine $\left\{\frac{\partial \mathbf{w}_k}{\partial \mathbf{B}_{R/I}}\right\}_{k=1,2,3}$ with $\frac{\partial \mathcal{L}}{\partial \mathbf{w}}$ to obtain $\frac{\partial \mathcal{L}}{\partial \mathbf{B}} \in \mathbb{C}^{M \times N}$:

$$\left[\frac{\partial \mathbf{w}}{\partial \mathbf{w}_k}\right]_{ij} = \frac{\partial \left[\mathbf{w}\right]_i}{\partial \left[\mathbf{w}_k\right]_j} = \frac{\partial}{\partial \left[\mathbf{w}_k\right]_j} \left[\mathbf{w}_1 + \mathbf{w}_2 + \mathbf{w}_3\right]_i = \delta_{i-j} = \left[\mathbf{I}_N\right]_{ij}, \qquad k = 1, 2, 3 \tag{F.20}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}_k} = \frac{\partial \mathcal{L}}{\partial \mathbf{w}} \frac{\partial \mathbf{w}}{\partial \mathbf{w}_k} \underset{\text{(F.20)}}{\overset{\text{(F.12)}}{=}} \sum_{l=1}^{L} \frac{\partial \mathcal{L}}{\partial \mathbf{s}^l}, \qquad k = 1, 2, 3 \tag{F.21}$$

$$\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_1}\frac{\partial\mathbf{w}_1}{\partial\mathbf{B}_R}\right]_{jk} = \sum_{i=1}^{N}\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_1}\right]_i\left[\frac{\partial\mathbf{w}_1}{\partial\mathbf{B}_R}\right]_{ijk} \overset{\text{(F.16)}}{=} 2\sum_{i=1}^{N}\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_1}\right]_i \delta_{i-k}(\mathbf{b}_k^R)^T\sigma_j^R \tag{F.22}$$

$$= 2\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_1}\right]_k(\mathbf{b}_k^R)^T\sigma_j^R = \left[2\hat{\mathbf{\Sigma}}_R^T\mathbf{B}_R\,\mathrm{diag}\left(\frac{\partial\mathcal{L}}{\partial\mathbf{w}_1}\right)\right]_{jk}$$

$$\overset{\hat{\mathbf{\Sigma}}_R=\hat{\mathbf{\Sigma}}_R^T}{=} \left[2\hat{\mathbf{\Sigma}}_R\mathbf{B}_R\,\mathrm{diag}\left(\frac{\partial\mathcal{L}}{\partial\mathbf{w}_1}\right)\right]_{jk}$$

$$\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_2}\frac{\partial\mathbf{w}_2}{\partial\mathbf{B}_I}\right]_{jk} = \sum_{i=1}^{N}\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_2}\right]_i\left[\frac{\partial\mathbf{w}_2}{\partial\mathbf{B}_I}\right]_{ijk} \overset{\text{(F.17)}}{=} 2\sum_{i=1}^{N}\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_2}\right]_i \delta_{i-k}(\mathbf{b}_k^I)^T\sigma_j^R \tag{F.23}$$

$$= 2\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_2}\right]_k(\mathbf{b}_k^I)^T\sigma_j^R = \left[2\hat{\mathbf{\Sigma}}_R^T\mathbf{B}_I\,\mathrm{diag}\left(\frac{\partial\mathcal{L}}{\partial\mathbf{w}_2}\right)\right]_{jk}$$

$$\overset{\hat{\mathbf{\Sigma}}_R=\hat{\mathbf{\Sigma}}_R^T}{=} \left[2\hat{\mathbf{\Sigma}}_R\mathbf{B}_I\,\mathrm{diag}\left(\frac{\partial\mathcal{L}}{\partial\mathbf{w}_2}\right)\right]_{jk}$$

$$\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\frac{\partial\mathbf{w}_3}{\partial\mathbf{B}_R}\right]_{jk} = \sum_{i=1}^{N}\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\right]_i\left[\frac{\partial\mathbf{w}_3}{\partial\mathbf{B}_R}\right]_{ijk} \overset{\text{(F.18)}}{=} 2\sum_{i=1}^{N}\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\right]_i \delta_{i-k}(\mathbf{b}_k^I)^T\sigma_j^I \tag{F.24}$$

$$= 2\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\right]_k(\mathbf{b}_k^I)^T\sigma_j^I = \left[2\hat{\mathbf{\Sigma}}_I^T\mathbf{B}_I\,\mathrm{diag}\left(\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\right)\right]_{jk}$$

$$\overset{\hat{\mathbf{\Sigma}}_I=-\hat{\mathbf{\Sigma}}_I^T}{=} \left[-2\hat{\mathbf{\Sigma}}_I\mathbf{B}_I\,\mathrm{diag}\left(\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\right)\right]_{jk}$$

$$\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\frac{\partial\mathbf{w}_3}{\partial\mathbf{B}_I}\right]_{jk} = \sum_{i=1}^{N}\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\right]_i\left[\frac{\partial\mathbf{w}_3}{\partial\mathbf{B}_I}\right]_{ijk} \overset{\text{(F.19)}}{=} -2\sum_{i=1}^{N}\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\right]_i \delta_{i-k}(\mathbf{b}_k^R)^T\sigma_j^I \tag{F.25}$$

$$= -2\left[\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\right]_k(\mathbf{b}_k^R)^T\sigma_j^I = \left[-2\hat{\mathbf{\Sigma}}_I^T\mathbf{B}_R\,\mathrm{diag}\left(\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\right)\right]_{jk}$$

$$\overset{\hat{\mathbf{\Sigma}}_I=-\hat{\mathbf{\Sigma}}_I^T}{=} \left[2\hat{\mathbf{\Sigma}}_I\mathbf{B}_R\,\mathrm{diag}\left(\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\right)\right]_{jk}$$

$$\frac{\partial\mathcal{L}}{\partial\mathbf{B}_R} = \frac{\partial\mathcal{L}}{\partial\mathbf{w}_1}\frac{\partial\mathbf{w}_1}{\partial\mathbf{B}_R} + \frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\frac{\partial\mathbf{w}_3}{\partial\mathbf{B}_R} \tag{F.26}$$

$$\underset{\text{(F.24)}}{\overset{\text{(F.22)}}{=}} 2\left\{\hat{\mathbf{\Sigma}}_R\mathbf{B}_R\,\mathrm{diag}\left(\frac{\partial\mathcal{L}}{\partial\mathbf{w}_1}\right) - \hat{\mathbf{\Sigma}}_I\mathbf{B}_I\,\mathrm{diag}\left(\frac{\partial\mathcal{L}}{\partial\mathbf{w}_3}\right)\right\}$$

$$\overset{\text{(F.21)}}{=} 2\left\{\hat{\mathbf{\Sigma}}_R\mathbf{B}_R - \hat{\mathbf{\Sigma}}_I\mathbf{B}_I\right\}\mathrm{diag}\left(\sum_{l=1}^{L}\frac{\partial\mathcal{L}}{\partial\mathbf{s}^l}\right)$$

$$= 2\Re\left\{\hat{\mathbf{\Sigma}}\mathbf{B}\right\}\mathrm{diag}\left(\sum_{l=1}^{L}\frac{\partial\mathcal{L}}{\partial\mathbf{s}^l}\right)$$

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \mathbf{B}_I} &= \frac{\partial \mathcal{L}}{\partial \mathbf{w}_2}\frac{\partial \mathbf{w}_2}{\partial \mathbf{B}_I} + \frac{\partial \mathcal{L}}{\partial \mathbf{w}_3}\frac{\partial \mathbf{w}_3}{\partial \mathbf{B}_I} \\ &\overset{\text{(F.23)}}{\underset{\text{(F.25)}}{=}} 2\left\{\hat{\mathbf{\Sigma}}_R \mathbf{B}_I \operatorname{diag}\left(\frac{\partial \mathcal{L}}{\partial \mathbf{w}_2}\right) + \hat{\mathbf{\Sigma}}_I \mathbf{B}_R \operatorname{diag}\left(\frac{\partial \mathcal{L}}{\partial \mathbf{w}_3}\right)\right\} \\ &\overset{\text{(F.21)}}{=} 2\left\{\hat{\mathbf{\Sigma}}_R \mathbf{B}_I + \hat{\mathbf{\Sigma}}_I \mathbf{B}_R\right\} \operatorname{diag}\left(\sum_{l=1}^{L}\frac{\partial \mathcal{L}}{\partial \mathbf{s}^l}\right) \\ &= 2\Im\left\{\hat{\mathbf{\Sigma}}\mathbf{B}\right\} \operatorname{diag}\left(\sum_{l=1}^{L}\frac{\partial \mathcal{L}}{\partial \mathbf{s}^l}\right)\end{aligned} \tag{F.27}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{B}} = \frac{\partial \mathcal{L}}{\partial \mathbf{B}_R} + j\frac{\partial \mathcal{L}}{\partial \mathbf{B}_I} = 2\hat{\mathbf{\Sigma}}\mathbf{B} \operatorname{diag}\left(\sum_{l=1}^{L}\frac{\partial \mathcal{L}}{\partial \mathbf{s}^l}\right) \tag{F.28}$$

F.6 $\partial \mathcal{L}/\partial \tau$

$$\left[\frac{\partial \mathbf{s}^l}{\partial \boldsymbol{\tau}}\right]_{ij} = \frac{\partial \left[\mathbf{s}^l\right]_i}{\partial \left[\boldsymbol{\tau}\right]_j} = \frac{\partial}{\partial \left[\boldsymbol{\tau}\right]_j}\left[\mathbf{u}^l + \mathbf{w} - \boldsymbol{\tau}\right]_i = -\delta_{i-j} = \left[-\mathbf{I}_N\right]_{ij}, \qquad l = 1, \ldots, L \tag{F.29}$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\tau}} = \sum_{l=1}^{L}\frac{\partial \mathcal{L}}{\partial \mathbf{s}^l}\frac{\partial \mathbf{s}^l}{\partial \boldsymbol{\tau}} \overset{\text{(F.29)}}{=} -\sum_{l=1}^{L}\frac{\partial \mathcal{L}}{\partial \mathbf{s}^l} \tag{F.30}$$

Combining eqs. (F.14), (F.28) and (F.30) leads to algorithms 1 and 2.

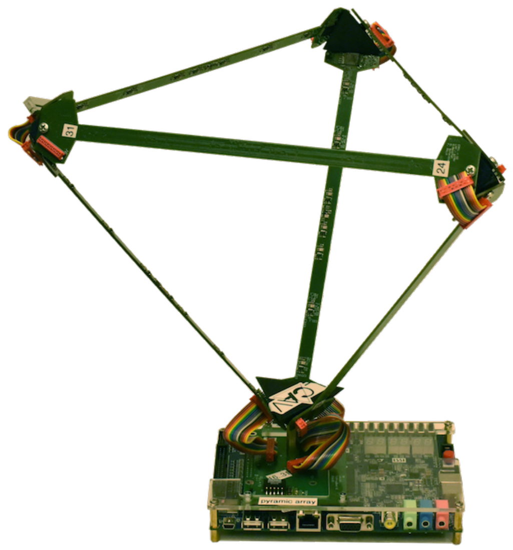


Figure G.1: Pyramic 48-element microphone array [16] used to acquire real-world dataset [10]. Eight microphones are mounted on six PCBs that form the edges of a tetrahedron.

G Real-data experiments (supplement)

Results in the main text present a summary of DeepWave's performance on two real-world datasets. The goal of this section is to provide a more elaborate description of the datasets, training process, and emphasise interesting observations.

G.1 Dataset description

Two real-world datasets are considered:

Dataset 1 [10] consists of a series of 92 microphone recordings from the *Pyramic*[16] array (fig. G.1) taken in an anechoic chamber to evaluate the performance of different direction-of-arrival algorithms [11]. Specifically, the dataset contains a series of 3 second recordings of human speech emitted by loudspeakers positioned around the edge of the chamber and located at the same height. Each recording has one, two, or three speakers active simultaneously. Recordings contain both male and female speech samples to cover a wide audible range.

Dataset 2 [15] consists of a larger collection of microphone recordings from the *Pyramic*[16] array (fig. G.1) taken in an anechoic chamber. The goal of this dataset is to provide a generic dataset on which to evaluate the performance of array processing algorithms on real-life recordings with all the non-idealities involved. Specifically, the dataset contains 2700 recordings of human speech emitted from every direction of the anechoic chamber at a resolution of 2 degrees in azimuth and three different elevations ({-15, 0, 15} degrees). Recordings contain both male and female speech samples to cover a wide audible range. While the total number of recordings is significant, since each recording contains emissions from a single source, different audio samples can be combined to simulate complex multi-source sound fields. This data-augmentation task therefore allows us to assess the generalizability of DeepWave to such setups. Concretely, we construct a synthetic dataset of 5700 distinct microphone recordings with one, two, or three active speakers simultaneously.

G.2 Data pre-processing

The raw time-series are pre-processed to get a suitable training set for DeepWave as follows:

- Instantaneous empirical covariances $\left\{\hat{\boldsymbol{\Sigma}}_t\right\}_t$ are obtained for 9 equi-spaced frequency bands spanning $[1500, 4500]$ Hz every 100 ms using *Short-Time Fourier Transforms (STFT)* [24, 8].

- APGD ground truths $\{\hat{\mathbf{x}}_t\}_t$ were estimated by solving eq. (3) with $\gamma = 0.5$, step size $\alpha = 1/\left\|\overline{\mathbf{A}} \circ \mathbf{A}\right\|_2^2$, and $\lambda_t = \max([\mathbf{x}_t^1]_1, \ldots, [\mathbf{x}_t^1]_N)/(\alpha\gamma)$, where $\mathbf{x}_t^1 \in \mathbb{R}^N$ is the APGD estimate obtained after one iteration of eq. (5).

After pre-processing, we obtain 2760 training samples $\mathcal{T} = \left\{\left(\hat{\mathbf{\Sigma}}_t, \hat{\mathbf{x}}_t\right)\right\}_t$ per frequency band for Dataset 1. The same process applied to Dataset 2 gives 151980 training samples $\mathcal{T} = \left\{\left(\hat{\mathbf{\Sigma}}_t, \hat{\mathbf{x}}_t\right)\right\}_t$ per frequency band.

G.3 Network training

DeepWave is trained by solving eq. (9) using stochastic gradient descent (SGD) with momentum acceleration [22]. The optimisation problem is initialised as given in eq. (10). Dataset 1 is trained on an 80% random subset of $\mathcal{T}$ using mini-batches of size $N_{batch} = 100$, with the remaining 20% serving as a validation set. The learning rate was set to 10^{-8}. Dataset 2 is also trained as above, except that 10 source directions are also witheld from the training set to assess how well DeepWave generalizes to emissions from unseen directions.

Regularisation parameters were chosen based on a grid search with optimal values $\lambda_{\boldsymbol{\theta}} = \lambda_{\mathbf{B}} = \lambda_{\boldsymbol{\tau}} = 0.1$. It was noticed during our experiments that regularising $\boldsymbol{\theta}$ and $\mathbf{B}$ provides little benefit to generalisation error and hence can be omitted. Regularisation of $\boldsymbol{\tau}$ is important however to ensure convergence to smooth biases. This is particularly relevant for rich acoustic fields where sources have weak spatial constraints, i.e. Dataset 2. (See also appendix H.)

Training and validation losses converged in less than 10 epochs for the optimal parameterisation, i.e. when $L = 5$ and K ranges from 10 to 23 depending on the frequency band. Total training time for Dataset 1 was 10 minutes per band on an i7-8550U CPU with 32GB memory. Due to disk space constraints, Dataset 2 was trained on a dual-socket Intel E5-2680v3 with 256GB memory. Total training time for Dataset 2 was roughly 3 hours per band.

G.4 Experimental results

In this section, we provide the supporting plots for the claims made in section 4 of the main paper:

- Figure G.2 shows DeepWave's learnt bias parameter on Dataset 1. Unlike APGD, the latter is highly nonuniform in space, and slightly stronger in magnitude.
- Figure G.3 shows the impulse response of DAS and DeepWave trained on Dataset 1 at 3.5 kHz, obtained by simulating the data from a single point-source in the field. Such plots were used to compute resolution scores of all algorithms across frequency bands.
- Figure G.4 shows example spherical fields obtained with DeepWave, DAS, APGD and APGD prematurely terminated applied to recordings in the validation set of Dataset 1. Resolution and contrast comparisons are moreover carried out. The true colour images displayed in fig. G.4 were obtained by mapping frequency channels into a colour spectrum (see the color-frequency mapping in fig. 3d).
- A video showing the evolution in time of DeepWave and DAS azimuthal sound fields (as in figs. 3a and 3b) is also available as supplementary material.[7]

[7]Also available online: `https://www.youtube.com/watch?v=PwB3CS2rHdI`

H Further experiments in simulation

Results in the main text present a summary of DeepWave's performance on two real-world datasets. Though the datasets represent a particular real-world scenario (i.e. Dataset 1) and realistic complex sound fields (i.e. Dataset 2), the downside is that sound emissions are assumed to come from fixed spatial directions. It is therefore challenging to test DeepWave's generalisability on these datasets alone. The goal of this section is to investigate how well DeepWave generalises to richer datasets through simulation.

H.1 Dataset description

The simulated dataset is designed to mimic a key application of acoustic cameras: accurate mapping of the sound field in an open-air setting from a given direction. To this end, the setup is modelled as follows:

- The scene is assumed to be a 120° spherical viewport in which sources are uniformly distributed.
- Source emissions follow a narrow-band point-source model at 2 kHz [23, 8], where their intensities are either uniform or Rayleigh-distributed with rate parameter $r = 1$. All images below show equi-amplitude visualisations only as they are easier to assess through visual inspection.
- Emissions from the scene are captured by a 64-element spherical microphone array of radius $r = 20$ cm.
- Empirical covariances matrices $\hat{\mathbf{\Sigma}} \in \mathbb{C}^{64\times 64}$ are synthesised using the traditional far-field measurement equation [23, eq.(12)] for point sources.
- APGD ground truths are obtained as described in appendix G.2.

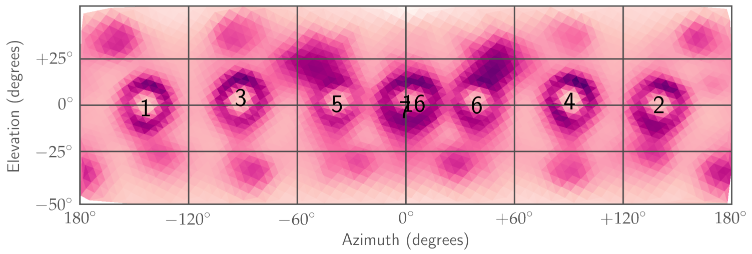


Figure G.2: Bias parameter $\boldsymbol{\tau}$ learnt by SGD run on Dataset 1 described in appendix G.1. We observe that the biasing is more prominent at sidelobes and around actual sources. This results in an increased angular resolution with fewer artefacts.

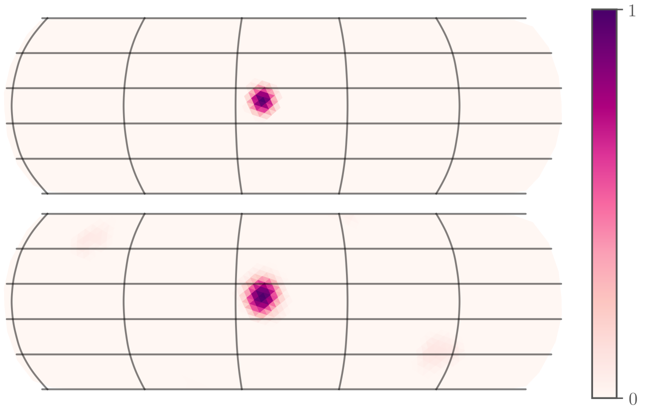
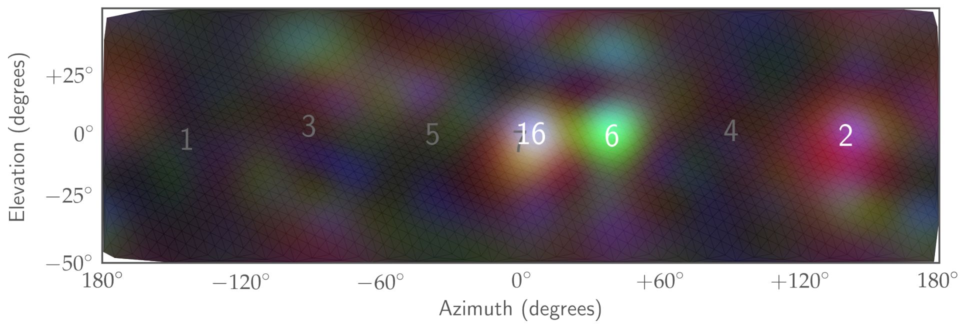
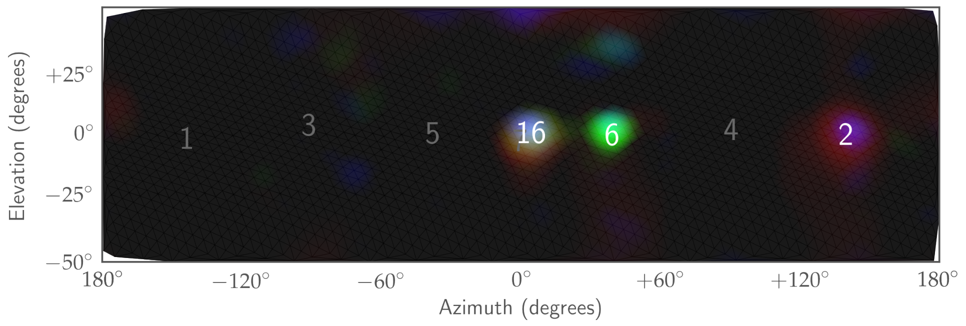


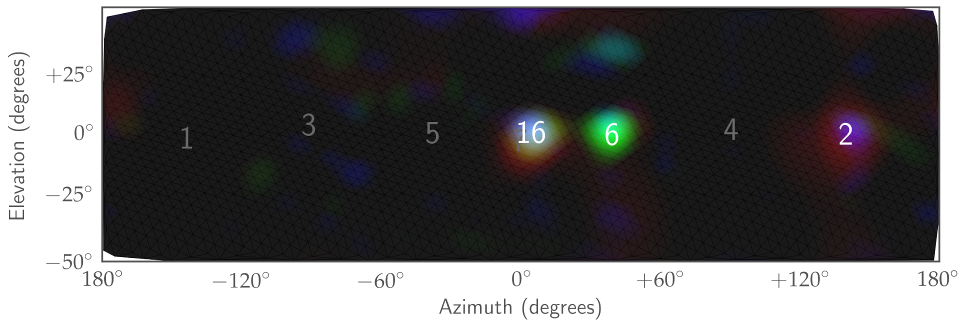
Figure G.3: Impulse response of DeepWave (top) vs DAS (bottom). We notice a shrinkage of the main lobe, resulting in increased angular resolution.



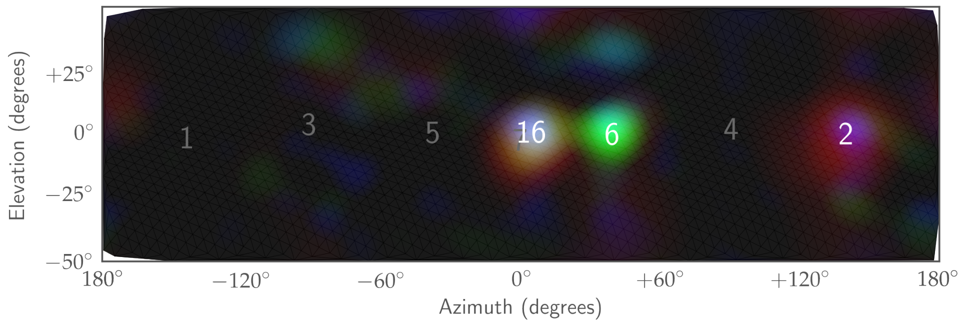
(a) DAS spherical sound field (resolution: 25.3° , RMS contrast: 0.78).



(b) DeepWave spherical sound field (resolution: 18.5° , RMS contrast: 0.97).



(c) APGD spherical sound field (resolution: 13° , RMS contrast: 0.97).



(d) APGD (terminated) spherical sound field (resolution: 21.4° , RMS contrast: 0.94).

Figure G.4: Intensity field reconstruction comparison between DAS, DeepWave ($L = 5$), APGD (converged, $N_{iter} = 17$), APGD (premature termination, $N_{iter} = 5$) on Dataset 1. In terms of resolution, DeepWave and APGD perform similarly, outperforming DAS by approximately 27%. The mean contrast scores for DeepWave and DAS over the test set of Dataset 1 are 0.99 (±0.0081) and 0.89 (±0.07), respectively. When limited to a number of iterations equal to the depth L of DeepWave, APGD's performance degrades considerably.

The final dataset consists of 20'000 images that contain up to 10 sources in the field. Training the network is identical to appendix G.3, except for the batch-size which increases to 200 and the learning rate that is set to 10^{-7}. In particular training converges in less than 10 epochs under an hour. The optimal parameterisation of the network is achieved with $L = 6$ and $K = 18$.

H.2 Experimental results

- Figure H.1 shows example spherical fields obtained with DeepWave, DAS and APGD applied to recordings in the validation set of DeepWave.
- Figure H.2 investigates the influence of DeepWave's depth on the validation loss. Profiles show that 5 or 6 layers are sufficient for the investigated dataset.
- Figure H.3 investigates the runtime of APGD, DAS and DeepWave for different depths. DAS and DeepWave execute several orders of magnitude faster than APGD, regardless of network depths. Similar conclusions apply to Dataset 1 investigated in appendix G.

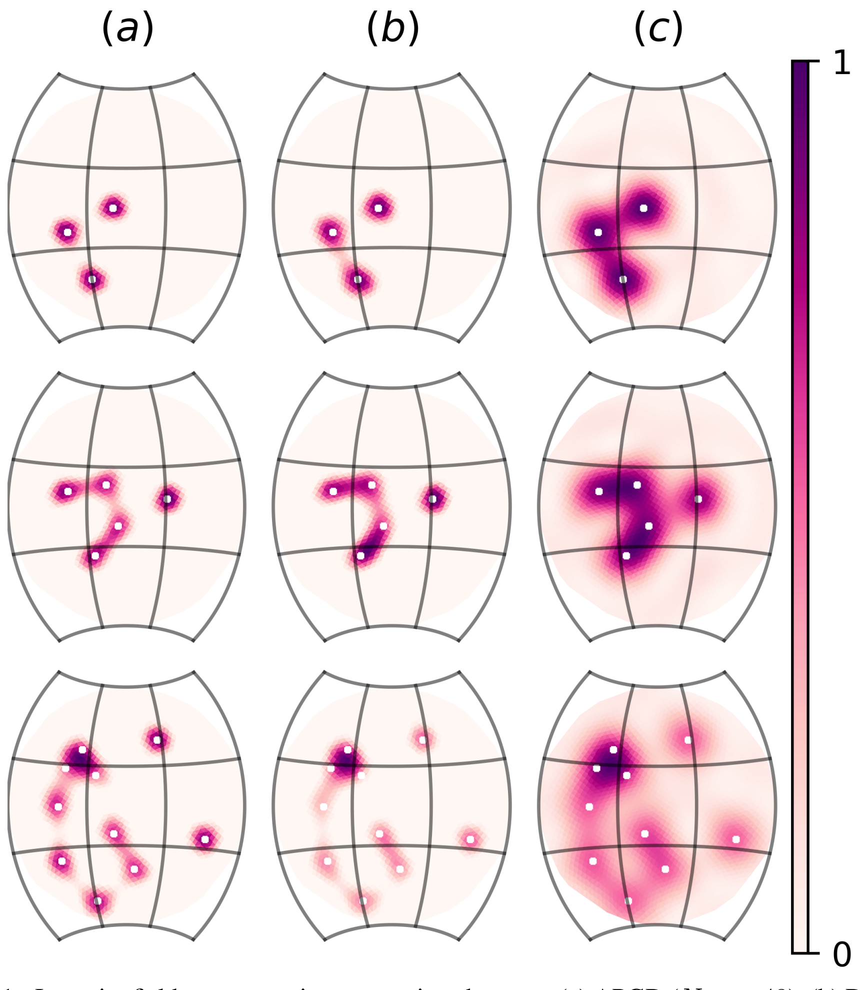


Figure H.1: Intensity field reconstruction comparison between (a) APGD ($N_{\text{iter}} = 48$), (b) DeepWave ($L = 5$), and (c) DAS. The image quality results corroborate with the observations made in fig. G.4.

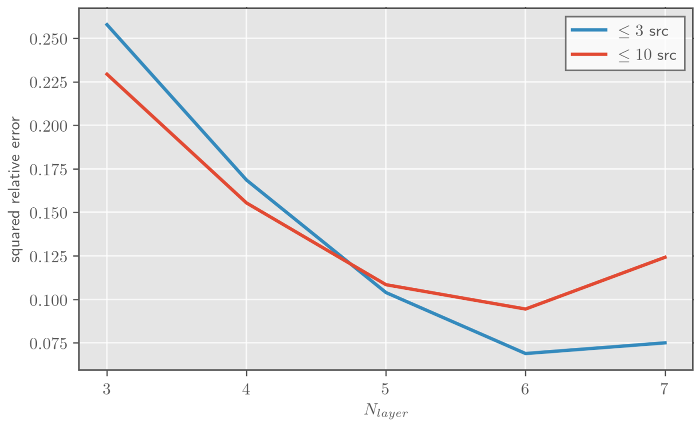


Figure H.2: Influence of network depth on validation loss. The plot shows the relative squared-error on the validation set between APGD ground truth $\hat{\mathbf{x}}$ and DeepWave output $\mathbf{x}^L$ as a function of network depth L using simulated data. The red curve corresponds to the full unconstrained dataset with up to 10 sources present in the field. The blue curve is obtained by retraining the network on a subset of the dataset where only up to 3 sources are present. Precision loss for small L comes from insufficient sparsification of network output w.r.t. ground truth. On the other hand error increase for L large are due to amplitude mismatches between ground truth and network output. This is presumably caused by the use of the rectified $\tanh$ activation function to avoid gradient explosion during training.

Method	N_{iter}/L	Execution time [s]
APGD (converged)	48	0.2118 $\pm$ 24e-3
DeepWave	6	0.0070 $\pm$ 80e-6
DeepWave	5	0.0065 $\pm$ 95e-6
DeepWave	3	0.0046 $\pm$ 44e-6
DeepWave	1	0.0031 $\pm$ 46e-6
DAS		0.0020 $\pm$ 41e-6

Figure H.3: Runtime comparison of imaging methods on simulated dataset. Execution times averaged over 50 runs for a specific $\hat{\boldsymbol{\Sigma}} \in \mathbb{C}^{64\times 64}$. DeepWave inference time is comparable to Delay-and-Sum and is adequate to obtain a fluid framerate on an acoustic camera. Runtimes in DeepWave weakly depends on network depth L due to strong sparsity of the deblurring operator $\boldsymbol{\mathcal{D}}$: the main contributor to the former is evaluation of the backprojection term $\boldsymbol{\mathcal{B}}\,\mathrm{vec}(\hat{\boldsymbol{\Sigma}})$. In stark contrast to DeepWave, APGD requires orders of magnitude more time to reach similar accuracy.

$\boldsymbol{\theta}$	$\mathbf{B}$	$\boldsymbol{\tau}$	$\mathcal{L}_{\text{test}}$	rel. improv. [%]	rel. improv. [%]
✗	✗	✗	0.160417		
✗	✗	✓	0.054927	65.76 (✗✗✗)	
✗	✓	✗	0.159698	0.45 (✗✗✗)	
✓	✗	✗	0.159948	0.29 (✗✗✗)	
✓	✗	✓	0.054910	65.77 (✗✗✗)	0.03 (✗✗✓)
✓	✓	✗	0.159234	0.74 (✗✗✗)	0.29 (✗✓✗)
✗	✓	✓	0.054917	65.77 (✗✗✗)	0.02 (✗✗✓)
✓	✓	✓	0.054900	65.78 (✗✗✗)	0.03 (✗✓✓)

Figure I.1: DeepWave performance comparison on simulated dataset described in appendix H.1 as a function of parameter degrees of freedom. A ✗ in the first three columns means that the associated parameter was frozen during training. In contrast a ✓ means that the parameter is optimized during training. $\mathcal{L}_{\text{test}}$ represents the data-fidelity loss term of eq. (9) evaluated over the test set. Finally, the last two columns show the relative improvement of $\mathcal{L}_{\text{test}}$ w.r.t. the baseline parameterisation given in parentheses. The results show that learning the shrinkage operator $\boldsymbol{\tau}$ has the strongest net effect on improving predictive performance, while *for this setup* the deblurring $P_{\boldsymbol{\theta}}(\mathbf{L})$ and backprojection operators $\mathbf{B}$ provide marginal gains.

I Ablation study

Results in the main text and above present a summary of DeepWave's performance after optimal tuning of network parameters $\boldsymbol{\theta}$, $\mathbf{B}$, $\boldsymbol{\tau}$ during training. Given the physical interpretation of these parameters as deblurring, backprojection and shrinkage operators respectively, we carry out an ablation study to investigate the relative importance of each parameter on DeepWave's ability to reconstruct ground truth APGD images.

Concretely, eight instances of DeepWave with $L = 6$ are trained on the simulated dataset described in appendix H.1. Each instance corresponds to a particular combination of free/frozen parameters such that all possible parameter triplets are taken into consideration. Frozen parameters remain at the initialisation point eq. (10) of SGD. Network performance is assessed by computing the data-fidelity term $\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_t$ from eq. (9) over the test set. The results are shown in Figure I.1.

As expected, freezing all three parameters (✗✗✗) produces the worst reconstructions as the network fails to converge to the ground truth after so few iterations. At the other end of the spectrum, learning all parameters (✓✓✓) leads to the best predictive performance, with a relative improvement of 65.78% over not learning anything. However the contributions of each parameter vary significantly: Learning $\boldsymbol{\tau}$ (✗✗✓) has the strongest net effect (65.76%), whereas learning $\boldsymbol{\theta}$ (✓✗✗), $\mathbf{B}$ (✗✓✗) provide minimal gains over no learning (✗✗✗). The second half of Figure I.1 shows similar observations hold when training parameter pairs, where learning any parameter in addition to $\boldsymbol{\tau}$ only provides small marginal gains over just learning the latter (✗✗✓). The reason for the marginal gains obtained when learning $\boldsymbol{\theta}$ and $\mathbf{B}$ is that the deblurring and backprojection operators are, for the specific experimental conditions investigated (point sources, non-reverberant environments (i.e. anechoic chambre), near-spherical geometries), very well modelled by initialisation scheme eq. (10). However, for environments containing reverberation and non-spherical array geometries, the observations above may differ significantly. In these contexts, learning $\boldsymbol{\theta}$ and $\mathbf{B}$ may lead to better predictive performance.

References

[1] Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of Marchine Learning Research*, 18:1–43, 2018.

[2] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[3] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[4] Krzysztof M Gorski, Eric Hivon, Anthony J Banday, Benjamin D Wandelt, Frode K Hansen, Mstvos Reinecke, and Matthia Bartelmann. Healpix: a framework for high-resolution discretization and fast analysis of data distributed on the sphere. *The Astrophysical Journal*, 622(2):759, 2005.

[5] KG Jinadasa. Applications of the matrix operators vech and vec. *Linear Algebra and its Applications*, 101:73–79, 1988.

[6] Sepand Kashani. Optimization notes. page 6, 2019.

[7] Benjamin Keinert, Matthias Innmann, Michael Sänger, and Marc Stamminger. Spherical fibonacci mapping. *ACM Transactions on Graphics (TOG)*, 34(6):193, 2015.

[8] Hamid Krim and Mats Viberg. Two decades of array signal processing research. *IEEE signal processing magazine*, 1996.

[9] Shuangzhe Liu and Gõtz Trenkler. Hadamard, khatri-rao, kronecker and other matrix products. *International Journal of Information and Systems Sciences*, 4(1):160–177, 2008.

[10] Hanjie Pan, Robin Scheibler, Eric Bezzam, Ivan Dokmanić, and Martin Vetterli. Audio speech recordings used in the paper FRIDA: FRI-based DOA Estimation for Arbitrary Array Layout, March 2017. This work was supported by the Swiss National Science Foundation grant 20FP-1 151073, LABEX WIFI under references ANR-10-LABX-24 and ANR-10-IDEX-0001-02 PSL* and by Agence Nationale de la Recherche under reference ANR-13-JS09-0001-01.

[11] Hanjie Pan, Robin Scheibler, Eric Francis Bezzam, Ivan Dokmanic, and Martin Vetterli. Frida: Fri-based doa estimation for arbitrary array layouts. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5. 3186–3190, 2017.

[12] Nathanaël Perraudin, Michaël Defferrard, Tomasz Kacprzak, and Raphael Sgier. Deepsphere: Efficient spherical convolutional neural network with healpix sampling for cosmological applications. *Astronomy and Computing*, 2019.

[13] Kaare Brandt Petersen, Michael Syskind Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7(15):510, 2008.

[14] Boaz Rafaely. *Fundamentals of spherical array processing*, volume 8. Springer, 2015.

[15] Scheibler Robin. Pyramic Dataset : 48-Channel Anechoic Audio Recordings of 3D Sources, March 2018. The author would like to acknowledge Juan Azcarreta Ortiz, Corentin Ferry, and René Beuchat for their help in the design and usage of the Pyramic array. Hanjie Pan, Miranda Kreković, Mihailo Kolundzija, and Dalia El Badawy for lending a hand, or even two, during experiments. Finally, Juan Azcarreta Ortiz, Eric Bezzam, Hanjie Pan and Ivan Dokmanić for feedback on the documentation and dataset organization.

[16] Robin Scheibler, Juan Azcarreta, René Beuchat, and Corentin Ferry. Pyramic: Full stack open microphone array architecture and dataset. In *2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)*, pages 226–230. IEEE, 2018.

[17] David Shuman, Sunil Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 3(30):83–98, 2013.

[18] Matthieu SIMEONI. Statistical inference in positron emission tomography. Technical report, SWISS FEDERAL INSTITUTE OF TECHNOLOGY LAUSANNE, 2014.

[19] Matthieu Martin Jean-Andre Simeoni. Towards more accurate and efficient beamformed radio interferometry imaging. Technical report, 2015.

[20] Matthieu Martin Jean-Andre Simeoni. Deconvolution of gaussian random fields using the graph fourier transform. Technical report, 2016.

[21] Elias M Stein and Guido Weiss. *Introduction to Fourier analysis on Euclidean spaces (PMS-32)*, volume 32. Princeton university press, 2016.

[22] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[23] Alle-Jan van der Veen and Stefan J Wijnholds. Signal processing tools for radio astronomy. In *Handbook of Signal Processing Systems*, pages 421–463. Springer, 2013.

[24] Martin Vetterli, Jelena Kovačević, and Vivek K Goyal. *Foundations of signal processing*. Cambridge University Press, 2014.

[25] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.