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) [25, 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, and hence exhibit poor angular resolution [48, 7, 47]. The severity of this blur can be shown [52] to be proportional to the ratio $\lambda/D$, where $D$ is the diameter of the microphone array and $\lambda$ the sound wavelength.

*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.

Preprint. Under review.
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 E5) of approximately 10 degrees, against 7·10^{-4} degrees for a standard optical camera at 790 THz (violet). Moreover, acoustic cameras are often deployed in confined environments [33], requiring them to be as compact and portable as possible, which limits further the achievable angular resolution.

The advent of compressed sensing techniques [14, 43]—and their wide adoption in imaging sciences [53, 4, 31]—have inspired algorithmic solutions [47, 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) [57] or its accelerated variants [2, 30]. 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 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 [32, 55, 10] in the field of imaging sciences may however seal the fate of DAS for good. Indeed, this new imaging paradigm leverages neural-networks [27] 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 outstanding quality, with similar accuracy and resolution as state-of-the-art compressed-sensing methods [32]. Network architectures used for inverse imaging [22, 17, 55, 10, 40] are most often convolutional neural-networks (CNNs), directly adapted from generic architectures developed for image classification and segmentation [44]. While suitable for image processing tasks such as denoising, super-resolution or deblurring [38, 6], such architectures are ill-suited [32] for more complex image reconstruction problems where the input data may not consist of an image, as in the case of biomedical imagery [4, 31], interferometry [53] 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, 49, 29, 32] 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^4 inspired from the popular iterative soft-thresholding algorithm (ISTA) [2]. 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.\footnote{Remember that the blur spread is inversely proportional to the microphone array diameter.}

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\footnote{An acoustic camera typically updates the acoustic image dozen of times per second.}—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 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.

\footnote{LISTA stands for learned iterative soft-thresholding algorithm.}

\footnote{ISTA is an instance of proximal gradient descent for penalised basis pursuit problems [51].}

\footnote{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.}

\footnote{For a 1 megapixel image, the weights parametrising the network would be approximately 8 Gb in practice.}
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 $\hat{\Sigma} \in \mathbb{C}^{M \times M}$ of the microphone recordings is processed by the network as follows (see also fig. 1):

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

where $\text{vec} : \mathbb{C}^{M \times M} \to \mathbb{C}^{M^2}$ is the vectorisation operator and $\circ$ denotes the Khatri-Rao product (see appendix A for definitions). The neurons $\{x^1, \ldots, x^L\} \subset \mathbb{R}^N$ 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 $x^0 \in \mathbb{R}^N$ defines the initial state of the network. The nonlinear activation function $\sigma : \mathbb{R} \to \mathbb{R}$ induces sparsity in the acoustic image, and is inspired by the proximal operator of an elastic-net penalty \[37\]. The remaining quantities, namely $P_\theta(L)$, $\mathbf{B}$ and $\tau$ are trainable parameters of the network, with various roles:

- **Deblurring**: the matrix $P_\theta(L) := \sum_{k=0}^K \theta_k L^k \in \mathbb{R}^{N \times N}$ 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 $L \in \mathbb{R}^{N \times N}$ based on the connectivity graph of the spherical tessellation in use, with learnable coefficients $\theta = [\theta_0, \ldots, \theta_K] \in \mathbb{R}^{K+1}$. Such parametrisation permits notably the interpretation of $P_\theta(L)$ 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 $\left[ \mathbf{B} \circ \mathbf{B} \right]^H \text{vec}(\hat{\Sigma}) = \text{diag} \left( \mathbf{B}^H \hat{\Sigma} \mathbf{B} \right)$ \[A.8\] 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 $\mathbf{B} \in \mathbb{C}^{M \times N}$.

- **Bias**: the vector $\tau \in \mathbb{R}^N$ is a non-uniform bias, boosting or shrinking the neurons of the network. Since only positive neurons are activated by the nonlinearity $\sigma$, 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.

---

9 In all that follows, labels prefixed with roman letters refer to elements of the supplementary material.

10 Typified by a rectilinear unit.
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] [45]. 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 [26], including radio astronomy, radar and sonar technologies.

2 Network architecture

In this section, we proceed similarly to [16, 39, 29] 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:

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

where \( \| \cdot \|_F \) denotes the Frobenius norm, \( \gamma \in [0, 1] \) and \( \lambda > 0 \) are hyperparameters, and \( \Sigma \in \mathbb{C}^{M \times M} \)
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—commonly modelled by
the so-called steering matrix [26]: \( [\mathbf{A}]_{mn} := \exp((-2\pi i / \mathbf{p}_m) \cdot \mathbf{r}_n / \lambda_0) \), 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] [52], problem (2) can be re-written in
vectorised form as:

\[
\hat{\mathbf{x}} = \arg \min_{\mathbf{x} \in \mathbb{R}^N} \frac{1}{2} \left\| \mathbf{vec}(\Sigma) - (\mathbf{A}^\dagger \mathbf{A}) \mathbf{x} \right\|_2^2 + \lambda [\gamma \|\mathbf{x}\|_1 + (1 - \gamma) \|\mathbf{x}\|_2^2],
\]

where \( \circ \) denotes the Khatri-Rao product (see definition A.3). Problem (3) is an elastic-net penalised
least-squares problem [56], which seeks an optimal trade-off between data-fidelity and group-sparsey.
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) [22], whose update equations are given here by (see appendix C):

\[
\mathbf{x}^k = \text{ReLu} \left( \frac{\mathbf{x}^{k-1} - \alpha (\mathbf{A}^\dagger \mathbf{A})^H \left[ (\mathbf{A}^\dagger \mathbf{A}) \mathbf{x}^{k-1} - \mathbf{vec}(\Sigma) \right] - \lambda \alpha \gamma}{2 \lambda \alpha (1 - \gamma) + 1} \right), \quad k \geq 1,
\]

where \( \mathbf{x}^0 \in \mathbb{R}^N \) is arbitrary, \( \alpha \leq 1/\|\mathbf{A}^\dagger \mathbf{A}\|_2^2 \) is the step size and \( \text{ReLu}(x) := \max(x, 0) \) is the
rectified linear unit [28], applied element-wise to a real vector [1]. The sequence of iterates \( \{\mathbf{x}^k\}_{k \in \mathbb{N}} \)
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:

\[
\begin{aligned}
\mathbf{y}^k &= \text{ReLu} \left( \frac{\mathbf{x}^{k-1} - \alpha (\mathbf{A}^\dagger \mathbf{A})^H \left[ (\mathbf{A}^\dagger \mathbf{A}) \mathbf{x}^{k-1} - \mathbf{vec}(\Sigma) \right] - \lambda \alpha \gamma}{2 \lambda \alpha (1 - \gamma) + 1} \right), \quad k \geq 1, \\
\mathbf{x}^k &= \mathbf{y}^k + \omega^k (\mathbf{y}^k - \mathbf{y}^{k-1})
\end{aligned}
\]

[11] Python implementation provided in supplementary material.
[12] The notion of optimality is defined here by the penalty parameter \( \lambda \).
[13] Note that with \( \mathbf{x}^0 \in \mathbb{R}^N \), every gradient step produces a real vector.
where the momentum sequence \( \{\omega^k\}_{k \in \mathbb{N}} \) can be designed in various ways \([30, 9]\). In our experiments, we will use \((5)\) as a baseline for speed comparisons, where \(\omega^k = (k-1)/(k+d), \quad k \geq 0, \) with \(d = 50\) \([30]\). The accelerated proximal gradient descent (APGD) method thus obtained is the fastest reported in the literature, with convergence rate \(o(1/k^2)\) \([30]\). Finally, we leverage the formulae \((\tilde{A} \circ A) x = \mathrm{vec}(A \mathrm{diag}(x) A^H) \{A\} \{S\},\) and \((\tilde{A} \circ A)^H \mathrm{vec}(R) = \mathrm{diag}(A^H R A) \{A\} \{S\}\(,\) 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 \(x^L\) obtained after \(L\) iterations of \((3)\) can then be seen as the output of an RNN with depth \(L\) and intermediate neurons linked by the recursion formula:

\[
x^l = \text{ReLU} \left( \mathcal{D} x^{l-1} + B \text{vec} \left( \Sigma \right) - \tau \right), \quad l = 1, \ldots, L.
\]

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

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

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 \(B\) 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 \([52, \text{Section 5.2}][54]\). The matrix \(\mathcal{D}\) 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 \(\tau\) 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 \(\mathcal{D}, B, \tau\) 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 \(\mathcal{D}, B\) and \(\tau\), would however result in \(O(N^2)\) 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 \(\mathcal{D}\) and \(B\). The resulting RNN architecture, dubbed DeepWave, is given in \([1]\) and depicted in fig. \([1]\).

**Parametrisation of \(\mathcal{D}\)** Our parametrisation of \(\mathcal{D}\) is motivated by the following result, characterising the oracle deblurring kernel for spherical microphone arrays\(^{15}\) (see proof in appendix \([D]\)

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

\[
\left[ I - \alpha \left( \tilde{A} \circ A \right)^H \left( \tilde{A} \circ A \right) \right]_{ij} \simeq \left[ \delta_{ij} - \alpha M^2 \text{sinc}^2 \left( \frac{D}{\lambda_0} ||r_i - r_j|| \right) \right], \quad \forall i, j \in \{1, \ldots, N\}
\]

where \(\lambda_0\) is the wavelength, \(\delta_{ij}\) denotes the Kronecker delta and \(\text{sinc}(x) := \sin(\pi x)/\pi x\) is the cardinal sine. Moreover, the approximation \([5]\) is extremely good for \(M \geq 3\left| \frac{2\pi D}{\lambda_0} \right|^2\).

Proposition \([1]\) tells us that, for spherical arrays with sufficient number of microphones\(^{16}\), the oracle deblurring operator \(\mathcal{D}\) in \([7]\) corresponds actually to a sampled zonal kernel \([14]\): \(\mathcal{D}_{ij} = \kappa(||r_i - r_j||)\). 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.

\(^{14}\) Even for a spherical array with diameter \(D = 30\) cm operating at \(1\) kHz, \(M \geq 90\) is sufficient.
Algorithm 1 DeepWave forward propagation

1: Input: $\hat{\Sigma}_t$, $x_t^0$, $\hat{x}_t$, $\theta$, $B$, $\tau$, $\sigma$
2: Output: $L_t \in \mathbb{R}_+$, $\{s_l^t\}_{l=1}^{\tau}$
3: $y_t \leftarrow \text{diag}(B^H \hat{\Sigma}_t B) - \tau$
4: for $l$ in $\{1, \ldots, \tau\}$ do
5: $\sigma_l^t \leftarrow P_0(L) x_l^{t-1} + y_t$
6: $x_l^t \leftarrow \sigma_l^t$
7: $L_t \leftarrow \frac{1}{2} \| \hat{x}_t - x_l^t \|^2_2 / \| \hat{x}_t \|^2_2$

Algorithm 2 DeepWave backward propagation

1: Input: $\hat{\Sigma}_t$, $x_t^0$, $\hat{x}_t$, $\theta$, $B$, $\sigma$, $\{s_l^t\}_{l=1}^{\tau}$
2: Output: $\partial \theta \in \mathbb{R}^{K+1}$, $\partial B \in \mathbb{C}^{M \times N}$, $\partial \tau \in \mathbb{R}$
3: $(\partial x, \partial \theta, \partial \tau) \leftarrow (\sigma(s_l^t) - \hat{x}_t) / \| \hat{x}_t \|^2_2, 0, 0$
4: for $l$ in $\{L, \ldots, 1\}$ do
5: $\partial s_l \leftarrow \text{diag}(\sigma'(s_l^t)) \partial x$
6: $\partial x \leftarrow P_0(L) \partial s$
7: $\partial \tau \leftarrow \partial \tau - \partial s$
8: $\partial \theta_k \leftarrow [\partial \theta]_k + \partial s^T T_k(L) \sigma(s_l^{t-1})$
9: $\partial B \leftarrow -2 \hat{\Sigma}_t B \text{diag}(\partial \tau)$

Figure 2: Forward and backward algorithms to compute gradients of $L_t$ with respect to $\theta$, $B$, $\tau$. For notational simplicity we use the shorthand $\partial \alpha = \partial L_t/\partial \alpha$, and assume $\sigma(s_l^0) = x_l^0$. $r_j(\|)$ for some $\kappa : \mathbb{R}_+ \rightarrow \mathbb{R}$. Since zonal kernels are used to define spherical convolutions [34], $D$ can hence be seen as a discrete convolution operator over the tessellation in use $\Theta = \{r_1, \ldots, r_N\}$. Its bandwidth is moreover essentially finite, since coefficients $D_{ij}$ decay as $1/\|r_i - r_j\|^2$. As discussed in [41] [13], discrete spherical convolution operators with finite scope can be efficiently represented and implemented by means of graph signal processing [46] techniques. This leads us to consider the following parametrisation (see appendix B.3 for details): $D = P_0(L) := \sum_{k=0}^{K} q_k L^k$, where $\theta = [\theta_0, \ldots, \theta_K] \in \mathbb{R}^{K+1}$, $K$ controls the scope of the discrete convolution and $L \in \mathbb{R}^{N \times N}$ is the Laplacian [46] associated to the convex-hull graph of $\Theta$. Note that with this parametrisation, the number of parameters characterising $D$ drops from $N^2$ to $K + 1$, with $K \ll N$.

Parametrisation of $B$ The oracle back-projection operator (7) admits a factorisation in terms of the Khatri-Rao product. We decide hence to equip $B$ with a similar structure: $B = (B \circ B)^H$ for some learnable matrix $\hat{B} \in \mathbb{C}^{M \times N}$. With such a parametrisation, the number of parameters characterising $B$ 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.

- DeepWave($\Omega$, $L$) denotes a specific instance of the DeepWave network [1] with parameters $\Omega := \{\theta, B, \tau\}$ and depth $L$.
- 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:

$$
\hat{\Omega} \in \underset{\theta \in \mathbb{R}^{K+1}}{\arg \min} \underset{B \in \mathbb{C}^{M \times N}}{\frac{1}{T} \sum_{t=1}^{T} \| x_t - x_l^t(\Omega) \|^2_2 / \| x_t \|^2_2 : L_t := L_0 } + \frac{\lambda_\theta}{2(K+1)} \| \theta \|^2_2 + \frac{\lambda_B}{2M} \| B \|^2_F + \frac{\lambda_{\tau}}{2N} \| L^{1/2} \|^2_2.
$$

The quantities $\{ x_l^t(\Omega) \}_t$ and $\{ \hat{x}_t \}_t$ in (9) correspond respectively to the outputs of DeepWave($\Omega$, $L$) and APGD($\alpha, \lambda, \gamma$) with identical example input data $\{(\hat{\Sigma}_t, x_t^0)\}_t$. The first term $\frac{1}{T} \sum_{t=1}^{T} L_t$ is a data-fidelity term, which attempts to bring $\hat{x}_t$ and $x_l^t(\Omega)$ as close as possible from one another. The additional terms $L_0$, $L_B$, $L_\tau$ are smoothing regularisers, fighting against overfitting, a common issue in deep learning. Since the shrinkage operator $\tau$ is defined over an irregular spherical tessellation, the smoothing term $L_\tau$ is defined via the Laplacian $L \in \mathbb{R}^{N \times N}$ 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 [50]. Gradients of $\mathcal{L}_t$ with respect to $\theta, B, \tau$ are efficiently evaluated using reverse-mode algorithmic differentiation [11] and are given in algorithms [1] and [2] (see appendix [E] for a derivation). While random initialisation of neural-networks is a common practice in deep learning [50], 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:

$$\theta^0 := \arg \min_{\theta \in \mathbb{R}^{K+1}} \| P\theta(L) - D \|^2_F, \quad B^0 := \sqrt{\frac{\alpha}{\beta}} A, \quad \tau^0 := \lambda \alpha \gamma^{1/N}.$$  (10)

For greater numerical stability during training, we follow [41] and reparameterise the deblurring filter as $P\theta(L) = \sum_{k=0}^{K} \theta_k T_k(L)$, where $T_k(\cdot)$ is the Chebychev polynomial of order $k$ and $L$ 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].

4 Experimental results

In this section, we compare the accuracy, resolution and runtime performances of DeepWave to DAS and APGD on both real and simulated recordings from various microphone array geometries.

Real-data experiments We tested DeepWave on the dataset [35] (see appendix [F.1] for a detailed description), originally introduced in [36] for direction-of-arrival estimation purposes. This dataset reproduces a conference room setup depicted in figs. 3a and 3b: 8 people are gathered around a big table and speak either in turns or simultaneously (with at most 3 concurrent speakers). Recordings of the conversation are collected by the Pyramic array [45], positioned at the centre of the table. The latter has a tetrahedral geometry, with a total of 48 microphones and an approximate diameter of 30 cm (see fig. 3f). Since human speech is wide-band, we split the audible range $[1500, 4500]$ Hz in 9 uniform bins, which are processed independently by each algorithm. Moreover, empirical covariance matrices of the microphone recording are formed over 100 ms time intervals (see appendix [F.2]). For the training of DeepWave, the 2760 data points are split into a training and validation set (respectively 80% and 20% in size). For each frequency band, we chose an architecture with 5 layers, leading to a per-band training time of approximately 10 min on an i7-8550U CPU with 32 Gb of RAM (see appendix [F.3]).

In fig. 3, figs. F.4 and G.3 respectively, we compare the accuracy and runtimes 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 as supplementary material. In terms of resolution and contrast, DeepWave and APGD have similar performances, outperforming DAS by approximately 27% resolution-wise and 20% contrast-wise. Note that since the metrics used for assessing resolution and contrast 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).

In terms of runtimes finally, DeepWave and DAS both reach real-time requirements (6.3 ms and 2.4 ms respectively), outperforming APGD (7.26 s) by several orders of magnitude (see appendix [F.4]).

Further experiments In appendix [G] we tested DeepWave on a simulated dataset. The recordings from a spherical microphone array are simulated, using a narrow-band point-source data-model at 2 kHz [52]. The sources are randomly positioned over a $120^\circ$ field-of-view, with up to 10 concurrent sources per recording. Experiment results available in fig. [G.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. [G.2] the influence of network depth, and concluded that 5 or 6 layers are generally sufficient for the investigated dataset.

---

17 The 8 people are represented in the experiment by loudspeakers playing male and female speech samples.

18 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.
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.

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 $x^0$ 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. Finally, we would like to adapt DeepWave for real-time imaging in radio astronomy.

References


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


