BeBeC-2022-D09

A Trainable Iterative Soft Thresholding Algorithm for Microphone Array Source Mapping

Can Kayser, Adam Kujawski and Ennes Sarradj
TU Berlin, FG Technische Akustik
Einsteinufer 25, 10587 Berlin, Germany

Abstract

Microphone arrays have long been considered powerful tools for acoustic measurements and as such in the low-noise design of technologies such as aircraft, wind turbines and automobiles. Processing the microphone data to yield accurate results quickly has therefore been an active area of research. However, existing methods suffer from suboptimal spatial resolution, undesirable artifacts and high computational cost and consequently new techniques need to be explored. In this contribution, a recently proposed method coined Trained Iterative Soft Thresholding Algorithm (TISTA) is applied to the processing of microphone array measurements to create source maps using a discretized grid. The TISTA merges iterative algorithms designed to solve sparse linear inverse problems with data driven machine learning algorithms and was found to yield promising results. Synthetic source maps and corresponding microphone data are specifically computed on demand for training as well as evaluation and for a wide range of examples the sound sources are located and characterized with considerable accuracy at a Helmholtz number of 16 (5488 Hz). The technique is compared to the well established CLEAN-SC beamforming algorithm and while CLEAN-SC tends to overlook sources, particularly when many are present in the sound field, the TISTA can accurately identify a comparably large number of sources, while also being significantly faster.

1 Introduction

Noise is a stress factor and thus to be reduced to a minimum [8]. For this purpose it is helpful to know how much noise emerges exactly where - in other words to characterize and locate sound sources. This can be achieved using Microphone Arrays (MAs) - a set of microphones arranged along a deliberate geometry. However, processing the MA signals is associated with various issues, including high computational cost, low spatial resolution and undesirable artifacts [5, 11, 19]. It has thus been suggested to use machine learning techniques in order to improve the signal processing with respect to quality and efficiency [15].

One way to approach the problem at hand is referred to as compressed sensing, which has been a field of ongoing interest in the signal processing research community [7] and a wide variety of algorithms have been proposed for this task. Based on the Iterative Soft Thresholding Algorithm (ISTA) [3], a number of subsequent improvements have been suggested, the earlier of which retain the iterative nature, while later ones like the Learned Iterative Soft Thresholding Algorithm (LISTA) [4, 9], as well as TISTA [12, 23] discussed in this work, reframe the approach in a machine learning setting, where key parameters are optimized using data driven methods. These algorithms are designed such that they are applicable in a variety of scenarios and the Fast Iterative Soft Thresholding Algorithm (FISTA) [3] - an earlier iterative instance - has already been applied to MA data [16]. However, to the best of the authors knowledge, TISTA has not yet been tested on MA setups.

Thus, the purpose of the work presented here is to give a first informed assessment on the feasibility of TISTA with respect to MA signal processing. Namely, it is to be shown

- that it can - in principle - retrieve positions and characteristics of sound sources,
- how accurate the resulting source maps are,
- how much time this takes and
- whether or not TISTA can improve upon existing methods.

For this purpose it is compared to the CLEAN-SC algorithm, proposed in [22].

In Section 2 the theoretical framework is laid out, starting with the information theoretical fundamentals. First, the central linear inverse problem is outlined, followed by TISTA intended to solve it, including the employed principles of machine learning. Finally, the underlying physical model is introduced alongside the CLEAN-SC algorithm.

In Section 3 the methods used to approach the problem at hand are described, starting with the assumed measuring environment - that is, the MA geometry and the discretized source space, also referred to as grid. Subsequently the generation of training data and the training itself are detailed. Furthermore, a brief clarification of how complex numbers are represented in computer memory is given.

In Section 4, the results are laid out, including example source maps, computation time statistics and the training history.

In Section 5, the results are put into context with a particular focus on the inherent uncertainties of the methodology.

The entirety of this work is implemented using the programming language Python [24], particularly its modules NumPy [10], TensorFlow [1], Pandas [25] and Acoular [21].

2 Theoretical Framework

2.1 Linear Inverse Problems

This section generally draws from the ideas presented in [2, 7, 9]. Linear inverse problems arise when, based on an observable phenomenon, its cause is to be derived and their relation can be assumed to be linear. The problem can be stated as

$$\mathbf{m} = \mathbf{A}\mathbf{s} + \mathbf{n}, \tag{1}$$

where $\mathbf{s} \in \mathbb{C}^S$ is an *unkown* signal observed through a linear system $\mathbf{A} \in \mathbb{C}^{M\times S}$ to yield a *known* measurement $\mathbf{m} \in \mathbb{C}^M$ which may be corrupted by noise $\mathbf{n} \in \mathbb{C}^M$. The goal is then to recover the original signal $\mathbf{s}$. In cases where $M < S$ the problem is considered ill-posed, as a definitive solution may not and often does not exist [2]. However, if the signal $\mathbf{s}$ can be assumed to be sparse, a reasonable estimate - that is, a vector $\tilde{\mathbf{s}} \in \mathbb{C}^S$ that minimizes an appropriate error metric - may be found. One possible approach is presented in the following section. [1]

2.2 TISTA

TISTA is constructed as suggested in [12], also drawing from [4, 23]. It is intended to derive from a known measurement $\mathbf{m}$ an estimate $\tilde{\mathbf{s}}$ for the original signal $\mathbf{s}$. This is achieved recursively - the output of each layer depending on the result of the one before it - formally

$$\tilde{\mathbf{s}}_{t+1} = \eta(\tilde{\mathbf{s}}_t + \beta_t \mathbf{B}(\mathbf{m} - \mathbf{A}\tilde{\mathbf{s}}_t), \lambda_t) \tag{2}$$

with the initial condition $\tilde{\mathbf{s}}_0 = 0$. The number of layers is set to a constant $T \geq t \geq 0$ and consequently the derived estimate is the output of the final layer $\tilde{\mathbf{s}} = \tilde{\mathbf{s}}_T$. The operation $\mathbf{m} - \mathbf{A}\tilde{\mathbf{s}}_t$ can be thought of as a linear error computation, where the measurement that *would* result from the estimate of the respective layer $\mathbf{A}\tilde{\mathbf{s}}_t$ is compared to the *actual* input measurement $\mathbf{m}$. This is then mapped back to the signal vector space $\mathbb{C}^S$ by a pseudo-inverse linear mapping $\mathbf{B} = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T)^{-1} \in \mathbb{C}^{S\times M}$ and weighted with some step size $\beta_t \in \mathbb{R}$. Finally, the output of each layer is subject to a shrinkage function η with a shrinkage parameter $\lambda_t \in \mathbb{R}$. The shrinkage function is defined elementwise as

$$\eta(x, \lambda) = \mathrm{sgn}\,(x) \cdot \max\,(|x|, \lambda) \tag{3}$$

and can be thought of as a sieve, where λ represents its mesh size. More and more elements of $\tilde{\mathbf{s}}_t$ are set to zero with each layer, exploiting the sparsity of the original signal. The shrinkage function is shown in Fig. 1 with a shrinkage parameter $\lambda = 1$. This shrinkage function is different from the one that was originally proposed for TISTA, since the latter relies on statistical assumptions that have not yet been shown to be valid for the problem at hand. The one used here is taken from [4]. Lastly - again deviating from the original formulation of TISTA - the final output $\tilde{\mathbf{s}}_T$ is subject to a *ReLU* function, defined as

$$ReLU(x) = \max(0, x) \tag{4}$$

[1] See Section 3.5 for details on how complex numbers are represented in computer memory.

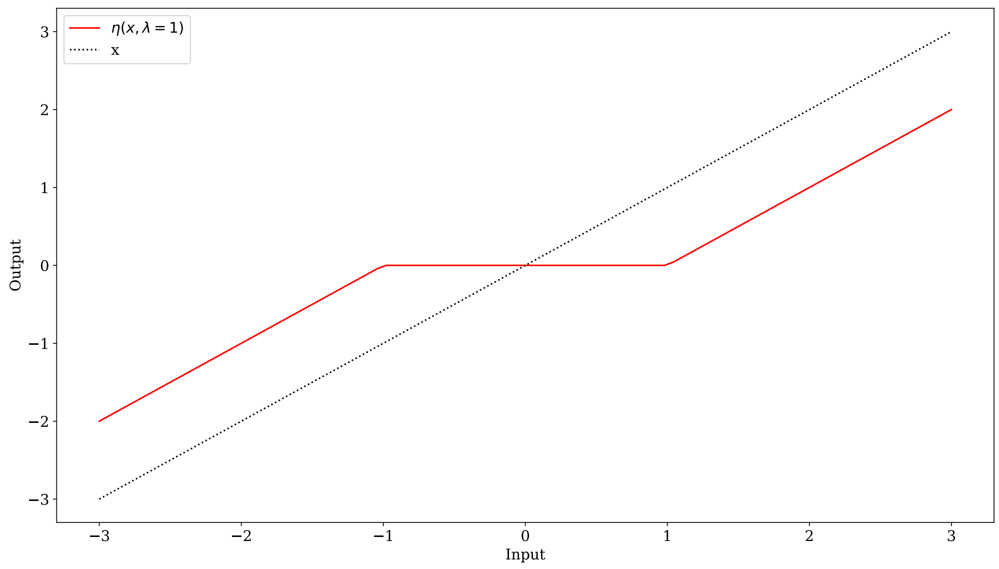

Figure 1: Shrinkage Function η with $\lambda = 1$

setting all negative values to zero, since source strength is strictly non-negative.

Now, the key idea is to subject the step size β_t and the shrinkage parameter λ_t to data driven optimization methods, which are discussed in the following section.

2.3 Machine Learning

This section is based on [4, 14, 18], according to which machine learning, namely *supervised learning* as it is employed here, is based on a set of training data: As mentioned above, the signal **s** that caused the measurement **m** is generally *unknown*, but for machine learning techniques to work instances of *known* signals and their corresponding measurements $\phi = \{\mathbf{s}_d, \mathbf{m}_d\}_{d=1}^{D}$ are necessary. Based on such data and the set of trainable parameters $\theta = \{\beta_t, \lambda_t\}_{t=0}^{T}$, a so called *mean squared error function* can be defined, formally

$$\mathcal{L}(\theta, \phi) = \frac{1}{D} \sum_{d=1}^{D} \|\tilde{\mathbf{s}}(\mathbf{m}_d, \theta) \; - \; \mathbf{s}_d\|_2^2 \, , \tag{5}$$

where $\tilde{\mathbf{s}}$ is, as stated in Section 2.2, the output of the algorithm and, as such, a function of its input **m** and the parameters θ, while D is the size of the dataset ϕ, or a subset of it, also called batch. This loss function maps the entire set of parameters θ onto a scalar, which is a measure of the algorithms output error with respect to the data ϕ, hence the term loss. Consequently, it is possible to find a set of parameters $\tilde{\theta}$ that minimizes the loss, formally

$$\tilde{\theta} = \arg\min_{\theta} \mathcal{L}(\theta), \tag{6}$$

which may be achieved by algorithms based on an approach called *gradient descent*: The gradient of the loss is calculated with respect to each parameter and repeatedly subtracted from them, causing the loss to approach a local minimum. Starting with some initial values $\tilde{\theta}_0$, every subsequent update of the parameters can be formalized as

$$\tilde{\theta}_{t+1} = \tilde{\theta}_t - \rho \, \nabla_\theta \mathcal{L}(\tilde{\theta}_t), \tag{7}$$

where ∇_θ denotes the gradient with respect to the parameters θ, while $\rho \in \mathbb{R}$ is a tunable parameter called learning rate. This process is subsequently referred to as training. There are a variety of algorithms designed for this optimization task, a thorough elaboration of which is beyond the scope of this work, but is provided in [18]. The Adam optimization algorithm presented in [14] is used for the entirety of this work.

Implicitly, the approach presented here is based on the central assumption that minimizing the loss for *known* data will also minimize it for *unknown* data, which in turn relies on the supposition that they are sufficiently similar in structure.

2.4 Cross-Spectral Matrix

Before the physical model is introduced, this section shall establish the Cross-Spectral Matrix (*CSM*), according to the definitions in [6]. Assuming an acoustic measurement, in other words sound pressures at a set of microphone positions p, the *CSM* is the covariance matrix of this measurement, formally

$$CSM = \mathbb{E}(pp^H), \tag{8}$$

where $\mathbb{E}$ is the expected value and H denotes complex conjugation and transposition. Note that in this setting, sound pressure is considered a stochastic process, implying that real measurements do not necessarily coincide with this expected value. Rather they are assumed to approach it with increasing accuracy, the longer the time frame over which they are averaged. However the physical model introduced in the following section assumes perfectly accurate instances of the *CSM*, each of their elements representing the actual expected value respectively. [2]

2.5 Physical Model

The physical model follows the one presented in [13], also drawing from [5, 11, 19]. Sound sources are assumed to be uncorrelated monopoles in a motionless fluid under free-field conditions. Assuming a stationary sound field, acoustic quantities are represented in the frequency domain.

A number of potential source locations - also called focus points or grid points - x_s are examined. With these, the signal vector $\mathbf{s}$ is defined such that each element corresponds to one

[2]See Section 5 for details on the possible repercussions of this assumption.

of the locations and its value is equal to the source strength [3] at that location, formally

$$\mathbf{s} = \begin{bmatrix} s_1(x_1) \\ s_2(x_2) \\ \vdots \\ s_S(x_S) \end{bmatrix}, \tag{9}$$

where if no source is present, the source strength is zero. Since - as discussed in Section 2.1 - **s** is assumed to be sparse, the number of *potential* source locations is supposed to significantly exceed the *actual* number of sources.

Subsequently, the measurement vector **m** is defined using the *CSM*, of which only the upper triangle elements are used, since they contain all the linear independent information [13]. Formally this can be expressed as

$$CSM = \begin{bmatrix} c_{1,1} & c_{1,2} & c_{1,3} & \cdots & c_{1,M} \\ c_{2,1} & c_{2,2} & c_{2,3} & \cdots & c_{2,M} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{M,1} & c_{M,2} & c_{M,3} & \cdots & c_{M,M} \end{bmatrix} \quad \rightarrow \quad \mathbf{m} = \begin{bmatrix} c_{1,1} \\ c_{1,2} \\ \vdots \\ c_{1,M} \\ c_{2,2} \\ \vdots \\ c_{2,M} \\ \vdots \\ c_{M,M} \end{bmatrix}, \tag{10}$$

where $c_{i,j} = p_i p_j^*$, with p_i representing the sound pressure at the i-th microphone and * denoting complex conjugation.

When **s** and **m** are defined, the sensing matrix needs to be constructed such that $\mathbf{m} = \mathbf{As}$ holds. This is achieved using the transfer vector $\mathbf{v}_{m,s}$, which associates microphone positions x_m with focus points x_s. If the assumptions stated at the beginning of this section hold, the transfer vector can be formalized as

$$\mathbf{v}_{m,s}(x_m, x_0, x_s) = \frac{r_{s,0}}{r_{s,m}} e^{-ik(r_{s,m} - r_{s,0})}, \tag{11}$$

where x_0 is an arbitrary but constant reference point, which is usually set to the position of one of the microphones, as specified in Section 3.1. The distance between two points x_i, x_j is denoted by $r_{i,j} = |x_i - x_j|$ and $k = \frac{c}{f}$ is the wave number, while i denotes the imaginary unit.

[3]Source strength, as defined here, is the squared sound pressure at a reference location due to the source.

The sensing matrix **A** can hence be defined columnwise as

$$\mathbf{A}_i = \begin{bmatrix} \mathbf{v}_{i,1}\mathbf{v}^*_{i,1} \\ \mathbf{v}_{i,1}\mathbf{v}^*_{i,2} \\ \vdots \\ \mathbf{v}_{i,1}\mathbf{v}^*_{i,M} \\ \mathbf{v}_{i,2}\mathbf{v}^*_{i,2} \\ \vdots \\ \mathbf{v}_{i,2}\mathbf{v}^*_{i,M} \\ \vdots \\ \mathbf{v}_{i,M}\mathbf{v}^*_{i,M} \end{bmatrix}, \tag{12}$$

where * denotes complex conjugation. Notably, Eq. (11) implies that **v** and - as a consequence - **A** are frequency-dependent. A thorough proof that $\mathbf{m} = \mathbf{As}$ holds for the above definitions can be found in [13].

Lastly, in order to describe the frequency f more generally, the dimensionless Helmholtz number is defined as

$$He = f\frac{a}{c}, \tag{13}$$

where $c = 343\,\mathrm{m\,s^{-1}}$ is the speed of sound and a is the aperture of the MA, defined as the maximum distance between two microphones.

2.6 CLEAN-SC

In this section, a brief and qualitative introduction of the CLEAN-SC algorithm used for comparison shall be given, drawing from [5, 11, 22].

CLEAN-SC is based on conventional beamforming, where sound sources are characterized using the *CSM* and steering vectors - elements of the transfer vector introduced in Eq. (11) - in a delay-and-sum approach. However, the results of conventional beamforming are not sufficiently precise, mostly because of so called side lobes - artifacts contaminating the derived source map. The CLEAN-SC algorithm seeks to remove these side lobes. Each iteration finds the maximum in the source map, replaces it by a clean beam and removes the spatially coherent side lobes. Since the source map resulting from conventional beamforming is thought of as a convolution of the true source map and the measurement system, CLEAN-SC is classified as a deconvolution algorithm [17]. The implementation provided in Acoular [21] is used in this work and - as originally proposed - the diagonal elements of the *CSM* are omitted.

3 Methods

3.1 Measurement Environment

The MA geometry follows a Vogel's spiral - distances being normalized, resulting in an aperture of $a = 1$ - as suggested in [20] and consists of 16 microphones. The reference position x_0 discussed in Section 2.5 is set to the position of the microphone closest to the origin.

A number of potential source locations are considered and make up the discretized signal space, also called grid. It consists of a total of 676 points, arranged parallel to the MA plane along a regular square with a side length of 1 m and a resolution of 0.04 m. It is shifted 0.5 m perpendicular to the MA plane and centered around the same point as the MA, as illustrated in Fig. 2.

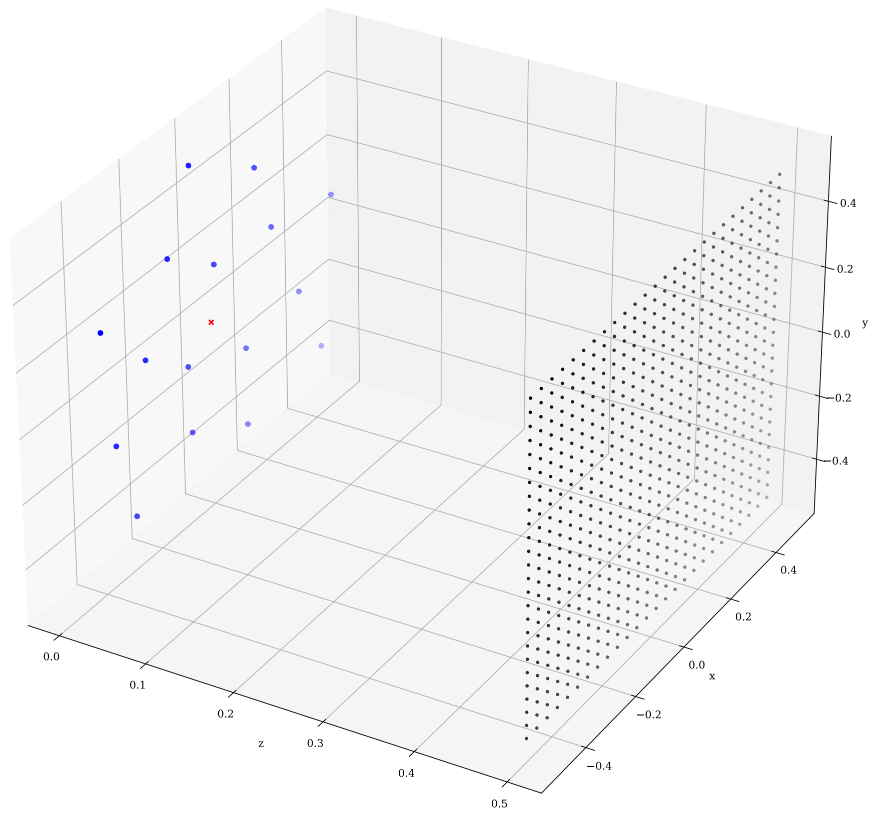

Figure 2: Microphone Array (blue dots), Grid (black dots) and Origin (red x)

3.2 Data

As stated in Section 2.3, the training process requires a data set of *known* signals and the resulting measurements, subsequently referred to as training data. Following the methodology in [4, 12, 23], training data is generated on demand during training. Notably, the data is not designed to model real measurements as accurately as possible. Rather, it is intended to be generated quickly and easily while being reasonably representative of actual use cases. [4]

First, instances of signal vectors $\mathbf{s}_d$ are generated. Using the random variables $r_1 \in \mathbb{R}^S \sim \mathcal{U}(0,1)$ drawn from a uniform distribution and $r_2 \in \mathbb{R}^S \sim \mathcal{R}(1)$ drawn from a Rayleigh distribution in accordance with [11], the desired sparsity $q = 0.01$ as well as the step

[4]See Section 5 for details on the limitations of this approach.

function

$$\mathcal{H}(x,q) = \begin{cases} 1 & \text{if } x < q \\ 0 & \text{if } x \geq q \end{cases}, \tag{14}$$

instances of $\mathbf{s}_d$ can be formalized as

$$\mathbf{s}_d = \mathcal{H}(r_1,q)|r_2| + 0i, \tag{15}$$

where i is the imaginary unit clarifying that source strength is modeled as purely real. The process described above is visualized in Fig. 3.

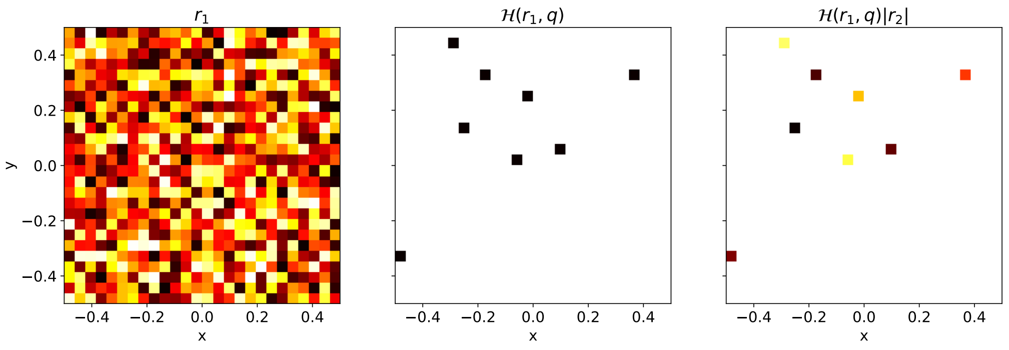

Figure 3: Example Case of Randomly Generated Source Distribution

Finally, $\mathbf{s}_d$ is multiplied by the sensing matrix $\mathbf{A}$ at a Helmholtz number of $He = 16$, yielding the associated measurement vector $\mathbf{m}_d = \mathbf{A}\mathbf{s}_d$. Note that the noise vector introduced in Section 2.1 is assumed to be $\mathbf{n} = 0$.

3.3 Training

As stated in Section 2.3, TISTA is supposed to be trained, that is the parameters β and λ need to be optimized such that the loss becomes minimal. Therefore they are initialized with $\beta_t = 1$ and $\lambda_t = 0$ for $0 \leq t \leq T$, while the number of layers is set to $T = 30$.

The Adam optimizer presented in [14] is used for training, with a learning rate $\rho = 8 \cdot 10^{-4}$, a batch size $D = 200$ and a total of 200 batches per epoch. After each epoch, the loss is calculated for one batch of *separate* but identically generated data, also called evaluation data. Once this evaluation loss does not improve for 10 consecutive epochs - a value referred to as patience - training is concluded, assuring that it does not run indefinitely on an otherwise theoretically infinite data set. After each epoch the intermediate loss for both the training data and the evaluation data are documented.

Aforementioned hyperparameters are summarized in Table 1.

3.4 Computation Time Measurement

TISTA as well as CLEAN-SC are run on an *Intel ® Core ™ i7-8665U* CPU using a single thread. Wall-clock-time is measured for 10^5 randomly generated cases, with $He = 16$. Note that computation time is measured for the successfully trained model *after* training.

Table 1: Training Hyperparameters

Hyperparameter	Symbol	Value
Number of Layers	T	30
Initial Step Size	β	1
Initial Shrinkage Parameter	λ	0
Learning Rate	ρ	$8 \cdot 10^{-4}$
Batch Size	D	200
Batches per Epoch	-	200
Patience	-	10 Epochs

3.5 Complex Number Representation

As the implementation of the Adam optimizer does not allow the processing of complex numbers, the data used here needs to be represented using real numbers only. This is achieved by stacking vectors and matrices, as follows: For a complex vector **v**, real and imaginary parts are written sequentially, which can be formally expressed as

$$\mathbf{v} = \begin{bmatrix} \mathrm{Re}\,(\mathbf{v}) \\ \mathrm{Im}\,(\mathbf{v}) \end{bmatrix}, \tag{16}$$

and requires a similar approach for complex matrices, such that matrix-vector multiplication yields results analogous to the original representation. For a complex matrix **V**, this is achieved with

$$\mathbf{V} = \begin{bmatrix} \mathrm{Re}\,(\mathbf{V}) & -\mathrm{Im}\,(\mathbf{V}) \\ \mathrm{Im}\,(\mathbf{V}) & \mathrm{Re}\,(\mathbf{V}) \end{bmatrix}, \tag{17}$$

an approach based on [13]. All quantities introduced as complex in this work are represented this way at all stages of computation.

4 Results

4.1 Source Maps

A selection of source maps is shown in Figs. 4 to 6, including the true source distribution along with predictions for both TISTA and CLEAN-SC. Three cases are shown with four, nine and 20 sources respectively, all of them at $He = 16$. Notably, with many sources present in the sound field, CLEAN-SC tends to overlook some of them, whereas TISTA can correctly locate all of them.

4.2 Computation Time

Computation times are presented as histograms in Fig. 7 and statistical values in Table 2 for 10^5 cases with $He = 16$. TISTA outperforms CLEAN-SC by roughly a factor of five on average, while also having a smaller standard deviation.

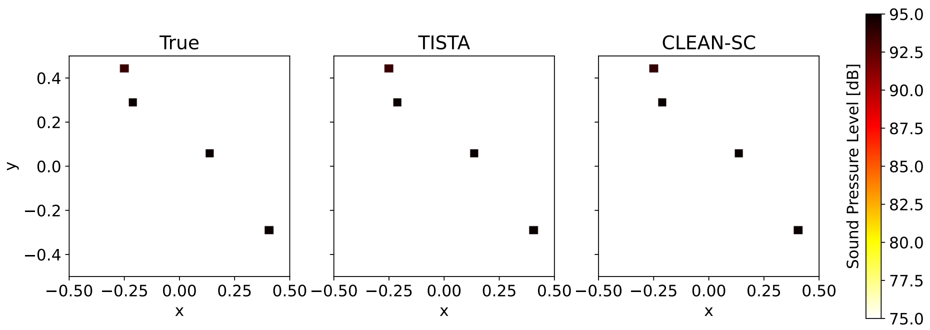

Figure 4: True Source Distribution With Four Sources Along With Predictions of TISTA and CLEAN-SC at Helmholtz number 16

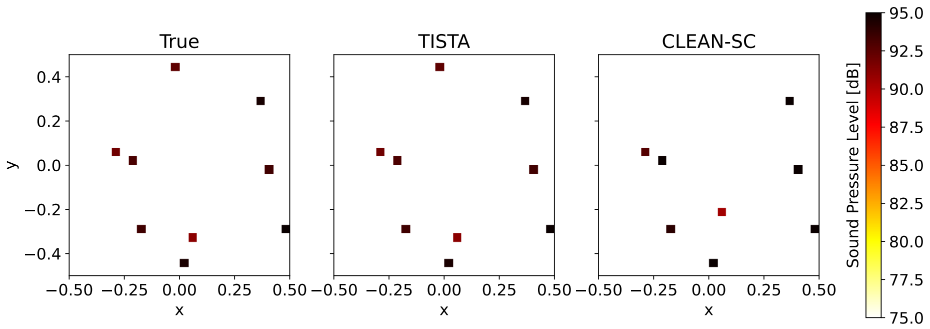

Figure 5: True Source Distribution With Nine Sources Along With Predictions of TISTA and CLEAN-SC at Helmholtz number 16

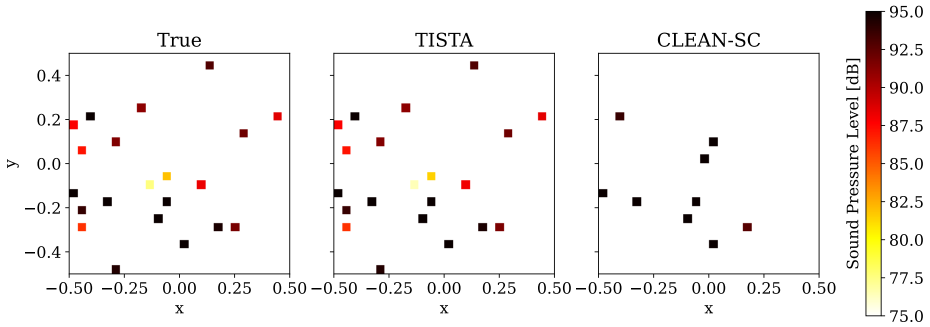

Figure 6: True Source Distribution With 21 Sources Along Predictions of TISTA and CLEAN-SC at Helmholtz number 16

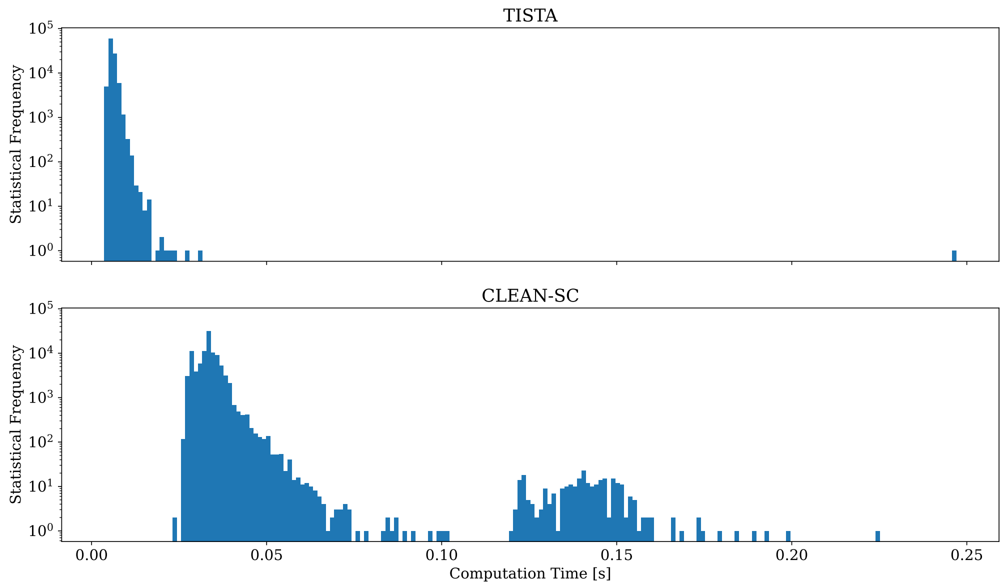

Figure 7: Computation Time Histograms for TISTA and CLEAN-SC (logarithmic y-Axis, 10^5 *Cases, 200 Bins)*

Table 2: Computation Time Statistics for TISTA and CLEAN-SC (10^5 *Cases, He* = 16*)*

	TISTA [s]	CLEAN-SC [s]
Min	0.004	0.024
Mean	0.006	0.034
Std. Deviation	0.001	0.007

4.3 Training History

In Fig. 8, the training history, that is, how the loss changes over the course of the training process is presented for both the training data and the evaluation data. The training process shows typical convergent behavior as discussed in [14] and the errors for the evaluation data mostly coincide with the errors for the training data. Using the hardware introduced in Section 3.4, the time required for training is in the order of several hours.

5 Discussion

5.1 Cross Spectral Matrix and Noise

Following the paradigm of quick and easy data production discussed in Section 3.2, noise is not considered in the model employed here.

Real measurements on the other hand, come with perturbations that can be assumed to notably impact the *CSM* in various ways. For instance, [6] states that flow-induced noise -

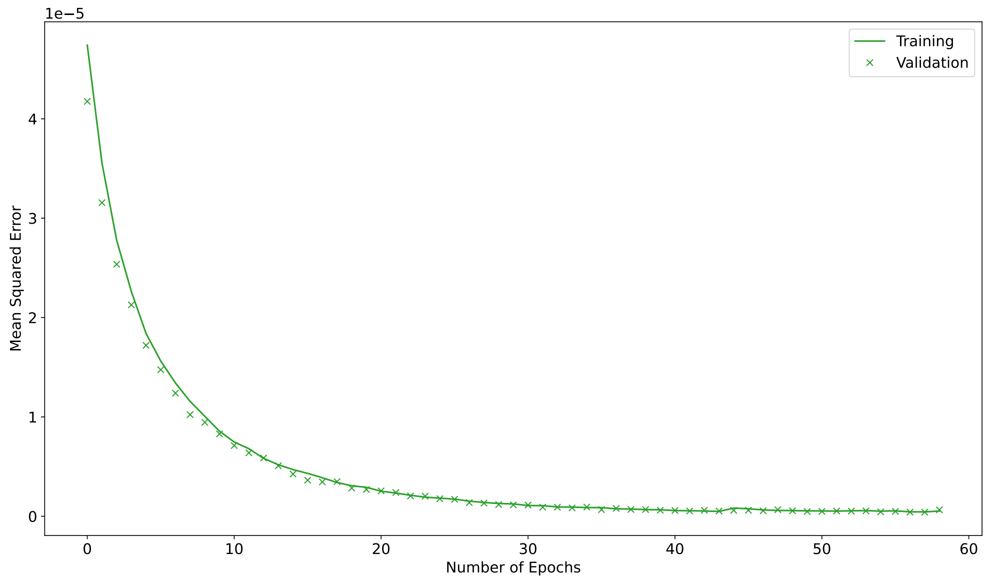

Figure 8: Training history

or more generally, perturbations that are uncorrelated over the microphones - can be assumed to primarily affect diagonal elements of the *CSM*. Furthermore, as discussed in Section 2.4, the *CSM* is assumed to perfectly represent the expected value of a stochastic process, which is generally not the case for real measurements averaged over a limited time frame.

Sound sources may also - in most use cases - not be located at the exact position of a grid point. Instead, they are more probably located *between* grid points. Acoustic, electric and digital noise are also not represented in the generated data and since free-field conditions are assumed, inaccuracies resulting from reverberation are not considered either.

In short it cannot be reliably concluded that TISTA will yield similar results in actual use cases, since measurement noise may result in various artifacts in the source maps, or in greater output errors than would be expected from the results shown here. However, in [12, 23], the robustness of TISTA is tested and it is shown that it can perform reasonably well even with noisy data, as long as the *condition number* of **A** is not too high.

5.2 Computation Time

On average TISTA is faster than CLEAN-SC by roughly a factor of five - a significant result as CLEAN-SC is already considered a comparably fast method [11, 17]. Notably, numbers of layers smaller than $T = 30$ may suffice for TISTA to converge, in which case the computational load would be even lower, so TISTA appears to be an exceptionally fast technique.

However, the methods used to assess the computation time come with a number of uncertainties. Since Python is a high-level programming language and different modules where used for TISTA and CLEAN-SC, it is not clear to what extent the implementation might be responsible for the difference in computation time. Furthermore, computational cost generally depends on different variables for different algorithms. Namely, the number of sources can be assumed to significantly impact computation time of CLEAN-SC, since it is iterated once for every source it finds. Consequently, the more sources are present in the sound field, the longer CLEAN-SC is kept running, which immediately affects computation time. TISTA on the other hand has a set number of operations, so it can be expected that computation time is not affected by the number of sources. Since the data set contains samples with variable numbers of sources, computation times vary more notably for CLEAN-SC, which is reflected in the higher standard deviation displayed in Table 2 and the visibly broader histogram in Fig. 7. Consequently the difference in computation time may be less evident for data sets with smaller numbers of sources. Moreover, the resolution and resulting number of focus points as well as the number of microphones may affect computation times to different degrees.

A theoretical assessment of computational complexity - as given in [4, 12, 23] - may be more insightful than the experimental measurement of computation time that was done here, but is not provided in the literature for CLEAN-SC [22].

5.3 Training History

Errors for the evaluation data mostly coincide with the errors for the training data, which is not surprising, since for a continually generated data set, overfitting - as discussed in [4] - is not to be expected, as each sample is used only once. While this can generally be considered a good indicator, the number of training samples used is in the order of 10^6, a scale not easily attained in real measurements and it is not yet clear how the training process evolves for data sets of limited size. However - as stated in [12, 23] - the training process of TISTA is generally very robust due to the remarkably small number of trainable parameters and consequently it can be assumed that training shows convergent behavior for smaller numbers of training samples than were used here.

5.4 Resolution and Sparsity

As stated in Section 2.1, a key assumption is that the signal vector $\mathbf{s}$ is sparse and consequently TISTA performs better the more elements of $\mathbf{s}$ are zero [12, 23]. For any given number of sources, higher resolutions will result in lower relative numbers of non-zero elements in $\mathbf{s}$ and consequently TISTA can be expected to perform increasingly well, the higher the resolution of the underlying grid.

Naturally, higher resolutions also result in longer computation times. However, as discussed earlier, TISTA is a remarkably fast algorithm, so increasing the grid size can be considered a worthwhile trade off.

6 Conclusion

TISTA is an outstandingly fast technique, yielding accurate results with a particular strength in detecting large numbers of sources. Downsides include its dependency on training data and a time consuming training process. However, the methods employed in this work do not necessarily represent actual use cases and further research is required to assess the fitness of TISTA.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. "TensorFlow: Large-scale machine learning on heterogeneous systems.", 2015. URL `https://www.tensorflow.org/`, software available from tensorflow.org.

[2] J. Adler and O. Öktem. "Solving ill-posed inverse problems using iterative deep neural networks." *Inverse Problems*, 33(12), 2017. ISSN 13616420. doi:10.1088/1361-6420/aa9581.

[3] A. Beck and M. Teboulle. "A fast iterative shrinkage-thresholding algorithm for linear inverse problems." *SIAM Journal on Imaging Sciences*, 2(1), 183–202, 2009. ISSN 19364954. doi:10.1137/080716542.

[4] M. Borgerding, P. Schniter, and S. Rangan. "AMP-Inspired Deep Networks for Sparse Linear Inverse Problems." *IEEE Transactions on Signal Processing*, 65(16), 4293–4308, 2017. ISSN 1053587X. doi:10.1109/TSP.2017.2708040.

[5] T. F. Brooks and W. M. Humphreys. "A deconvolution approach for the mapping of acoustic sources (DAMAS) determined from phased microphone arrays." *Journal of Sound and Vibration*, 294(4), 856–879, 2006. ISSN 10958568. doi:10.1016/j.jsv.2005.12.046.

[6] A. Dinsenmeyer, J. Antoni, Q. Leclère, and A. Pereira. "A probabilistic approach for cross-spectral matrix denoising: Benchmarking with some recent methods." *The Journal of the Acoustical Society of America*, 147(5), 3108–3123, 2020. ISSN 0001-4966. doi:10.1121/10.0001098.

[7] D. L. Donoho. "Compressed sensing." *IEEE Transactions on Information Theory*, 52(4), 1289–1306, 2006. ISSN 00189448. doi:10.1109/TIT.2006.871582.

[8] W. H. O. R. O. for Europe. *Burden of disease from environmental noise: quantification of healthy life years lost in Europe*. World Health Organization. Regional Office for Europe, 2011.

[9] K. Gregor and Y. LeCun. "Learning fast approximations of sparse coding." In *ICML 2010 - Proceedings, 27th International Conference on Machine Learning*, pages 399–406. 2010.

[10] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. "Array programming with NumPy." *Nature*, 585(7825), 357–362, 2020. doi:10.1038/s41586-020-2649-2. URL `https://doi.org/10.1038/s41586-020-2649-2`.

[11] G. Herold and E. Sarradj. "Performance analysis of microphone array methods." *Journal of Sound and Vibration*, 401, 152–168, 2017. ISSN 10958568. doi:10.1016/j.jsv.2017.04.030.

[12] D. Ito, S. Takabe, and T. Wadayama. "Trainable ISTA for Sparse Signal Recovery." *IEEE Transactions on Signal Processing*, 67(12), 3113–3125, 2019. ISSN 19410476. doi: 10.1109/TSP.2019.2912879. URL `http://arxiv.org/abs/1801.01978http://dx.doi.org/10.1109/TSP.2019.2912879`.

[13] A. Jahnke. *Untersuchung ausgewählter Compressed-Sensing-Ansätze zur Anwendung für inverse Mikrofonarrayverfahren*. Master's thesis, TU Berlin, 2018.

[14] D. P. Kingma and J. L. Ba. "Adam: A method for stochastic optimization." In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. 2015. URL `http://arxiv.org/abs/1412.6980`.

[15] A. Kujawski, G. Herold, and E. Sarradj. "A deep learning method for grid-free localization and quantification of sound sources." *The Journal of the Acoustical Society of America*, 146(3), EL225–EL231, 2019. ISSN 0001-4966. doi:10.1121/1.5126020.

[16] O. A. Lylloff, E. Fernandez-Grande, F. T. Agerkvist, J. Hald, E. T. Roig, and M. S. Andersen. "Improving the efficiency of deconvolution algorithms for sound source localization." *The Journal of the Acoustical Society of America*, 138 1, 172–80, 2015.

[17] R. Merino-Martínez, P. Sijtsma, M. Snellen, T. Ahlefeldt, J. Antoni, C. J. Bahr, D. Blacodon, D. Ernst, A. Finez, S. Funke, T. F. Geyer, S. Haxter, G. Herold, X. Huang, W. M. Humphreys, Q. Leclère, A. Malgoezar, U. Michel, T. Padois, A. Pereira, C. Picard, E. Sarradj, H. Siller, D. G. Simons, and C. Spehr. "A review of acoustic imaging methods using phased microphone arrays: Part of the "Aircraft Noise Generation and Assessment" Special Issue." *CEAS Aeronautical Journal*, 10(1), 197–230, 2019. ISSN 18695590. doi:10.1007/s13272-019-00383-4.

[18] S. Ruder. "An overview of gradient descent optimization algorithms." *arXiv*, pages 1–14, 2016. URL `http://arxiv.org/abs/1609.04747`.

[19] E. Sarradj. "Three-dimensional acoustic source mapping with different beamforming steering vector formulations." *Advances in Acoustics and Vibration*, 2012. ISSN 16876261. doi:10.1155/2012/292695.

[20] E. Sarradj. "A Generic Approach To Synthesize Optimal Array Microphone Arrangements." *Proceedings of the 6th Berlin Beamforming Conference*, pages 1–12, 2016. URL `http://www.bebec.eu/Downloads/BeBeC2016/Papers/BeBeC-2016-S4.pdf`.

[21] E. Sarradj and G. Herold. "A python framework for microphone array data processing." *Applied Acoustics*, 116, 50–58, 2017. ISSN 0003-682X. doi:https://doi.org/10.1016/j.apacoust.2016.09.015. URL `https://www.sciencedirect.com/science/article/pii/S0003682X16302808`.

[22] P. Sijtsma. “Clean based on spatial source coherence.” *International Journal of Aeroacoustics*, 6, 357–374, 2007. doi:10.1260/147547207783359459.

[23] S. Takabe, T. Wadayama, and Y. C. Eldar. “Complex Trainable Ista for Linear and Nonlinear Inverse Problems.” In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, volume 2020-May, pages 5020–5024. 2020. ISSN 15206149. doi:10.1109/ICASSP40776.2020.9053161.

[24] G. van Rossum. “Python tutorial.” Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI), Amsterdam, 1995.

[25] Wes McKinney. “Data Structures for Statistical Computing in Python.” In *Proceedings of the 9th Python in Science Conference* (edited by Stéfan van der Walt and Jarrod Millman), pages 56 – 61. 2010. doi:10.25080/Majora-92bf1922-00a.