BeBeC-2024-S11

State of open-source software for microphone array processing

Oliver Lylloff[1], Gert Herold[2], Adam Kujawski[2], and Ennes Sarradj[2]
[1]Department of Wind and Energy Systems, Technical University of Denmark (DTU)
Frederiksborgvej 399, 4000 Roskilde, Denmark
[2]Department of Engineering Acoustics, Technische Universität Berlin
Einsteinufer 25, D-10587 Berlin, Germany

Abstract

Open-source software has become increasingly important in research and recognized through funding and peer-reviewing. This paper outlines the current state of openly available software for microphone array processing and describes two libraries in detail: the well-established Acoular, and newer addition AeroAcoustics.jl. For both libraries, the design choices as well as possible application is discussed. Moreover, it is detailed how these open-source libraries allow a multi-leveled approach to quality assurance through the possibility of software peer review, regression tests as well as open benchmarks. The latter is also shown to be instrumental for reproducible research.

This is further exemplified by demonstrating a benchmark exercise with measured data from a wind tunnel test. The task is performed with both libraries. The presented results highlight the possibility to get the same results from the same data using different software.

Finally, lessons learned from the development of these tools are discussed, i.e., funding, peer-reviewing, student and researcher collaboration between research teams, and community involvement. It is concluded that open-source scientific software can be a key for the scientific community to improve not only the quality and reproducibility of research and application, but also the general availability of new algorithms and results.

1 INTRODUCTION

Acoustic imaging using microphone arrays has been immensely successful in enhancing the spatial resolution of noise mapping in aeroacoustic applications over the past 40 years [27]. A great diversity of algorithms has been developed, with continuous improvements regularly documented; two recent reviews are available in Refs. [9, 26]. The aeroacoustic community has also led efforts to benchmark various post-processing and acoustic imaging techniques through several benchmark cases, resulting in collaborative papers [6, 34] (the remaining cases are shared online, benefiting the aeroacoustic community by providing resources for testing and developing software and techniques [3]). These studies focused on comparing results from different research groups using the same datasets and algorithms (in the mathematical sense) but using different software. One conclusion was that despite using seemingly identical algorithms, the results varied due to differences in software implementation [6, 34]. The software codes used to produce these results were not shared, preventing identification of the source of discrepancies. If all participants had used publicly shared software, it would have been possible to trace the differences, reach a consensus on the correct implementation, and share these developments for the community's benefit. This is the spirit of open-source software, which is the topic of this paper.

Free and open-source software (FOSS) refers to software that is both freely available for use and redistribution, with source code openly accessible [1, 4, 12]. This accessibility allows users to review and suggest modifications to the code, promoting knowledge sharing and new developments. FOSS is based on a community-driven development model and is an integral part of the modern digital ecosystem. The Linux operating system is one prominent example of this [15].

Open-source software can be defined by the following characteristics[1]: 1) publicly available source code, 2) a software license approved by the Open Source Initiative (OSI [28]) or similar (e.g., GPL, MIT, BSD), 3) a documentation describing installation and usage, 4) automated tests (e.g., unit and regression tests), and 5) community collaboration guidelines (e.g., contributing, reporting issues, seeking help).

Unlike general-purpose software, which benefits from a large user base and the potential to establish substantial organizations, open-source *research* software faces challenges due to its limited user base and lack of academic recognition. Despite this fact, within the aeroacoustic community, several software packages have been shared online. A comprehensive review is outside the scope of this paper, but a curated list is available online [11]. Notable examples include Acoular [32], Augen [2], AcouPipe [18], and AeroAcoustics.jl [22]. Acoular, well-known due to it's lifespan and community interaction, is described in more detail later. Augen and AcouPipe integrate with Acoular, with the former enabling aeroacoustic simulations via Amiet's turbulence-aerofoil interaction implemented in Amiet-Tools [16], and the latter generating simulated datasets for machine learning applications. AeroAcoustics.jl, written in Julia [7], is also detailed later.

A substantial part of the software codes are written in Matlab, ranging from short pieces of code to extensive libraries. Although Matlab is widely used in academic institutions, its proprietary nature and licensing requirements disqualify it as open-source software. Nonetheless, shared Matlab code remains valuable for understanding and implementing specific methodologies. It is important to note that educational licenses for Matlab typically prohibit commercial use, unlike most open-source software licenses.

This paper introduces two open-source software libraries and demonstrates how a benchmark study can be conducted using shared code to facilitate further software development. The paper is structured as follows: Section 2 describes the two open-source software packages, which are used for the benchmark exercise detailed in Section 3. Section 4 presents the results, while

[1]Using the review guidelines `https://joss.readthedocs.io/en/latest/review_criteria.html` of the Journal of Open Source Software [37].

Section 5 discusses peer-review, teaching, and community involvement. The paper is concluded in Section 6.

2 Software

In this section the two open-source software packages (Acoular [32] and AeroAcoustics.jl [22]) are described briefly. A more detailed explanation can be found in their respective references and online repositories.

2.1 Acoular

Acoular[2] is an open software package written in Python for processing multichannel signals from microphone arrays [32]. Its first openly available version was released in 2015 under a permissive New BSD 3-Clause license. While the 2017 published article about Acoular already describes many still relevant concepts, such as following an object-oriented programming paradigm to ease the addition of new processing capabilities, the software has been constantly extended and improved since then. Currently, Acoular is being further developed as part of a research project[3] to improve collaboration possibilities, software quality, and reproducibility.

With the latest release [33], Acoular offers 15 microphone array methods, of which 13 are performed in the frequency domain and two in the time domain. Apart from the microphone array processing methods, functionalities such as data acquisition and data import, time-domain linear and nonlinear filtering, spectrum estimation, array layout and mapping grids definition, the ability to account for different sound propagating environments, moving sources and data synthesis are provided.

To allow the processing of computationally demanding tasks, Acoular uses out-of-core processing, lazy evaluation techniques, and transparent and persistent caching of intermediate and final results to avoid repeated computation. To enable parallel computation, Acoular relies on Numba, which is a just-in-time compiler for Python that translates Python functions to optimized machine code at runtime [19].

2.2 AeroAcoustics.jl

AeroAcoustics.jl[4] is an open source software package written in Julia [7]. It was developed as part of the PhD study [21], and has since been further extended to be the primary software tool for post-processing of aeroacoustic measurements at the Poul la Cour Tunnel (PLCT, DTU Wind, Risø Campus).

AeroAcoustics.jl utilizes key features of Julia, such as multiple dispatch, multi-threading, and just-in-time compilation, to provide a simple API with fast execution. The current acoustic imaging techniques are frequency-domain specific and include conventional beamforming [17], DAMAS [8], Clean-SC [35], and FISTA [24]. Utility functions for computing cross-spectral matrices from time-domain data, source integration, and octave band analysis are also provided.

[2] `https://www.acoular.org`
[3] Deutsche Forschungsgemeinschaft (DFG), project number: 528753521
[4] `https://github.com/1oly/AeroAcoustics.jl`

3 Benchmark

A new microphone array benchmarking exercise is introduced with this paper. To highlight the capabilities of the two software packages presented above, the benchmark case is designed to include multiple tasks, covering a typical post-processing chain of an aeroacoustic wind tunnel measurement campaign. The benchmark is summarized here, but a detailed version is available online [25]. It is the authors' hope that other institutions will join the benchmarking exercise online. This effort can be seen as an extension of the benchmarking initiatives put forward by the AIAA aeroacoustics community, resulting in the two papers [6, 34]. The purpose of this benchmark study, "Airfoil in a Kevlar-walled wind tunnel", is to demonstrate the benefits of open-source software for reproducing results and hence developing more robust software.

The benchmark exercise is presented below in a number of tasks. The same tasks are provided in the online repository [25] with additional instructions, and solutions to each task are given in separate documents by the two software packages, Acoular and AeroAcoustics.jl. In each task, a dataset is exported to a commonly readable file format (comma-separated values, or CSV), and in a third document, these results are imported and compared. This approach has the benefit of being extensible, allowing other open-source software packages to contribute to the benchmark and have their results compared to the existing ones. Another benefit is that users can reproduce results in incremental steps and compare them to their own code, even if it is not shared publicly.

The dataset for this exercise is shared in the AIAA HDF5 file format, along with metadata describing the measurement setup, ambient conditions, and microphone array geometry. This provides sufficient information to be reproducible and can also be used for purposes other than the benchmarking exercise. A summary of the measurement setup is given below. For further details, the reader is referred to Ref. [23].

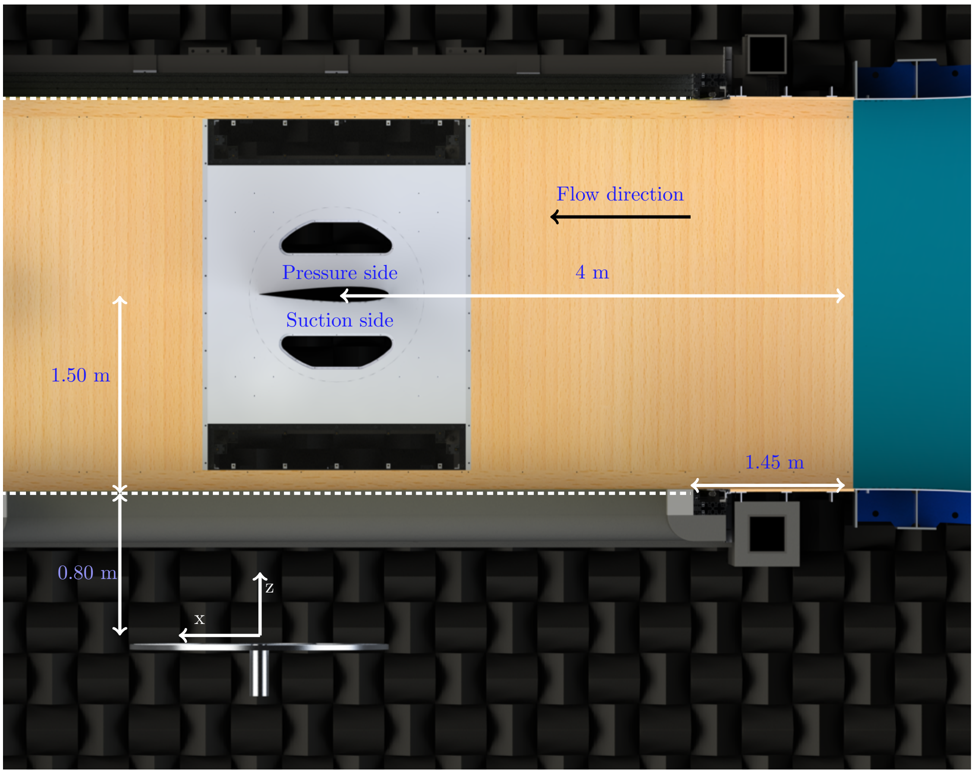
A NACA63018 airfoil with a chord length of 0.9 m and a span of 2 m was placed in a wind tunnel (Poul la Cour Tunnel (PLCT), DTU, Risø Campus) with Kevlar side walls (see Figure 1a). The microphone array, a B&K 84-channel array equipped with 1/4-inch microphones, was positioned such that the array plane is parallel to the center plane of the test section. The microphones are distributed pseudo-randomly in a circular plane with a diameter of 1.96 m (see Figure 1b). The free-stream velocity considered is $U_0 = 50$ m/s, and the airfoil angle of attack is AoA = 0°. The flow direction is in the positive x-direction (right to left in acoustic maps).

3.1 Tasks

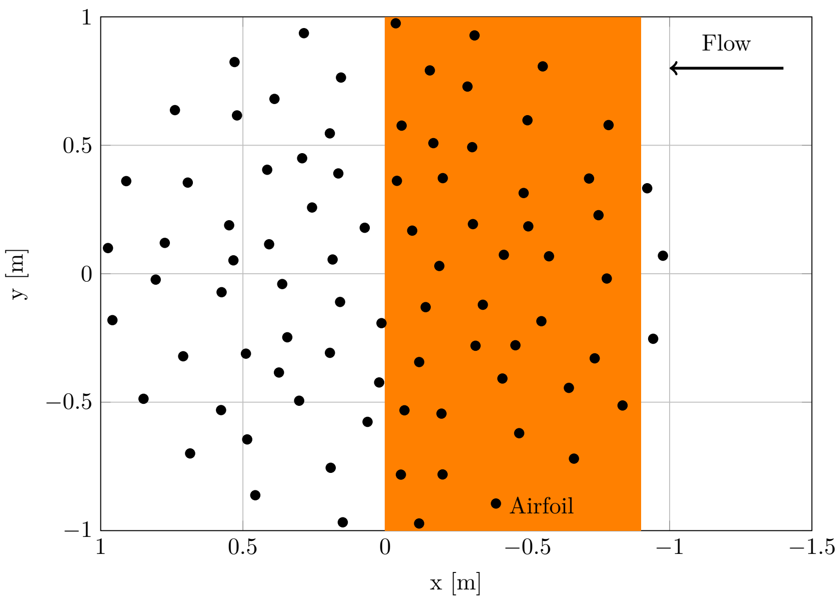
The benchmark exercise consists of 8 sub-tasks, which are stated below. The input data and import procedure are described in detail online [25].

1. Given the time-domain data file and associated sample rate: Compute the cross-spectral matrix (CSM) using Welch's method [40] with $n = 4096$, 50% overlap, and a Hanning window. Store real and imaginary parts of the CSM for frequency bins 500 Hz and 1000 Hz in CSV files.

2. Given the microphone array geometry found in the time-domain data file: Compute the Point-spread function (PSF) from a reference point $(x_0, y_0) = (0.0, 0.0)$ in the domain $x = [-1; 1]$ m and $y = [-1; 1]$ m using 41 points in each dimension for the plane at $z_0 =$



(a) Top view of test section.



(b) Microphone array geometry and airfoil location.

Figure 1: Schematic overview of measurement for benchmark dataset.

2.3 m. Use the steering vector formulation III as defined in [30] and store the PSF for frequencies 500 Hz, 1000 Hz, 2000 Hz, and 4000 Hz in CSV files.

3. Given the cross-spectral matrix data file, associated microphone array geometry, and source distance $z_0 = 2.3$: Apply diagonal removal, and compute the conventional beamforming acoustic image in the domain $x = [-2;2]$ m and $y = [-1;1]$ m using 41 points in the y-dimension and 81 points in the x-dimension, in the frequency range $(fmin, fmax) = (400, 4000)$. Store the acoustic images in dB for frequencies 500 Hz, 1000 Hz, 2000 Hz, and 4000 Hz in CSV files.

4. Using the above setup, compute acoustic maps with Clean-SC [35] using a loop-gain of 0.5, and maximum iterations of 100. Store the acoustic images in dB for frequencies 500 Hz, 1000 Hz, 2000 Hz, and 4000 Hz in CSV files.

5. Compute Task 3 (CBF) again, but with a shear-layer correction, given the position of the Kevlar wall 1.5 m from the trailing edge and 0.8 m from the microphone array, and the flow-speed given in the dataset. Assume a plane, thin shear-layer if using Amiet's correction [5]. Store the acoustic images in dB for frequencies 500 Hz, 1000 Hz, 2000 Hz, and 4000 Hz in CSV files.

6. Using the acoustic maps from Task 5, compute source integration for the region $(xmin, xmax, ymin, ymax) = (-0.5, 0.5, -0.4, 0.4)$ in the frequency range $(fmin, fmax) = (400, 4000)$. Store the result as two columns, first column 'fc' and second column 'dB' in CSV files.

7. Compute Task 4 (Clean-SC) again, but with a shear-layer correction, given the position of the Kevlar wall 1.5 m from the trailing edge and 0.8 m from the microphone array, and the flow-speed given in the dataset. Assume a plane, thin shear-layer if using Amiet's correction [5]. Store the acoustic images in dB for frequencies 500 Hz, 1000 Hz, 2000 Hz, and 4000 Hz in CSV files.

8. Using the acoustic maps from Task 7, compute source integration for the region $(xmin, xmax, ymin, ymax) = (-0.5, 0.5, -0.4, 0.4)$ in the frequency range $(fmin, fmax) = (400, 4000)$. Store the result as two columns, first column 'fc' and second column 'dB' in CSV files.

4 Results

In this section, the main results of the benchmarking exercise comparing the two software codes are shown. The complete set of results can be found online [25].

In task 1, the cross-spectral matrix was computed from time-domain data. Two of the frequency bins were stored in CSV files and the result for 500 Hz is shown in Figure 2 with the real and imaginary parts separated. Visually, it is difficult to assess any specific differences, but the level, as shown by the colorbar, is off. By further inspection, it is clear, that there is a constant difference (approximately 1.5) between the results produced by AeroAcoustics.jl and Acoular. The reason can be found by inspection of the source code, and is given by that fact, that two different scaling conventions for the compensation of the window influence are used. While AeroAcoustics.jl follows the "spectrum" convention used by SciPy [38], Acoular uses a different scaling. This scaling ensures that the overall power of the signal given by the mean square of the signal samples $p(n)$ equals the sum over all M components of the sampled autopower spectrum $G_{xx}(f_k)$:

$$p_{\text{rms}}^2 = \frac{1}{N}\sum_n^N p(n)^2 = \sum_k^M G_{xx}(f_k). \tag{1}$$

This holds regardless of the type of window applied.

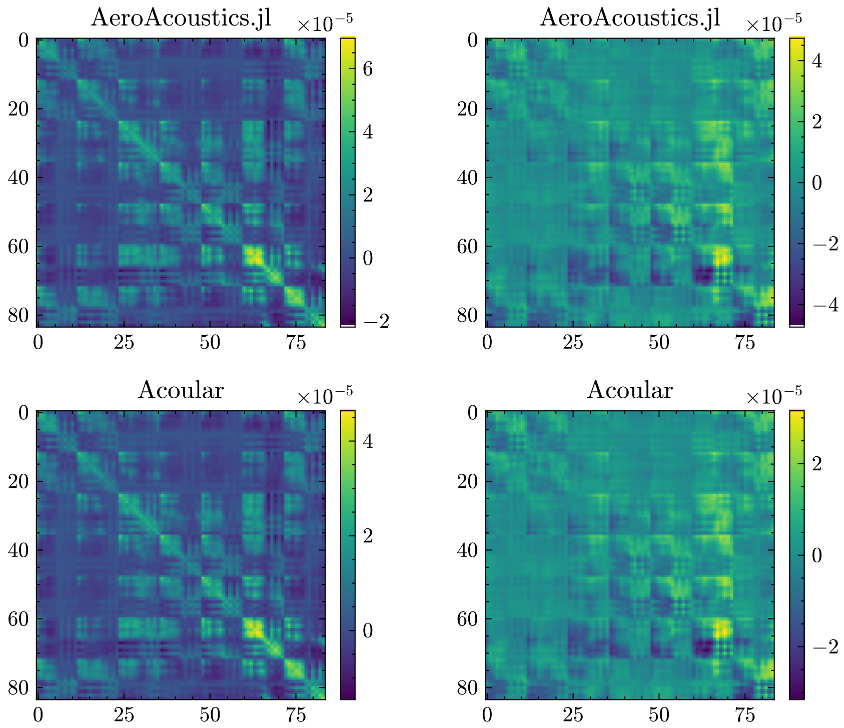
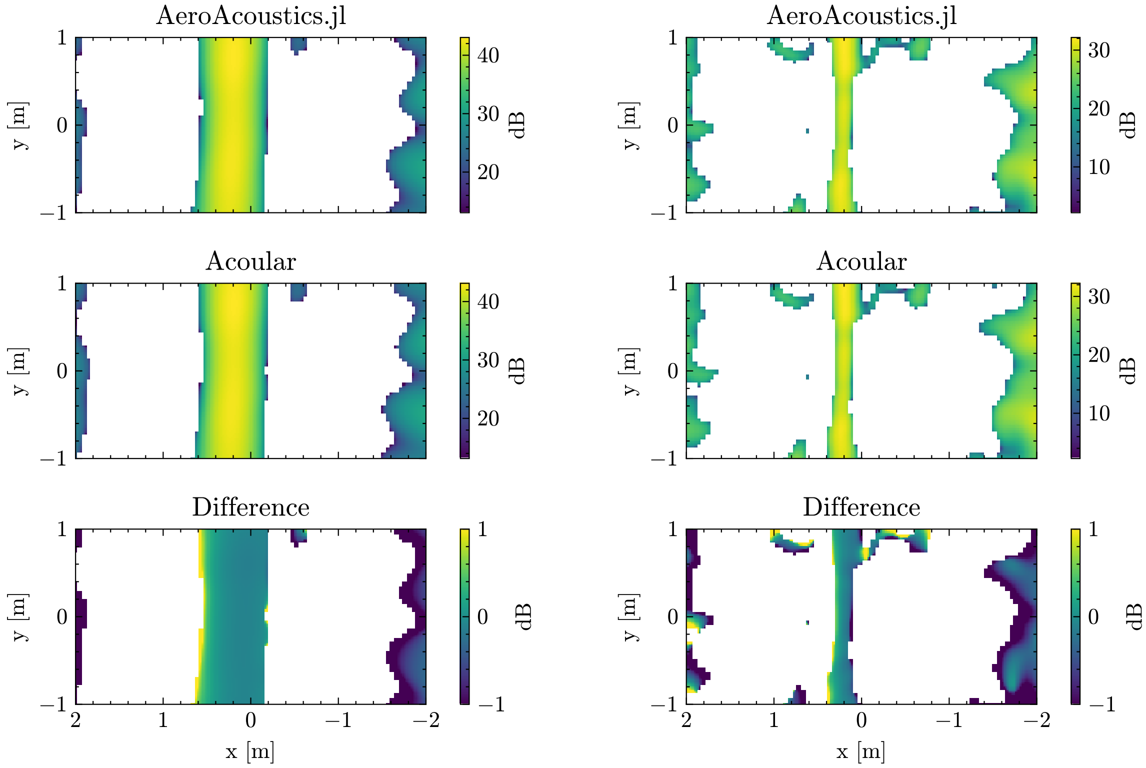


Figure 2: Cross-spectral matrix (CSM) at 500Hz. Left: Real part of CSM, Right: Imaginary part of CSM.

In task 3, conventional beamforming maps were computed based on a pre-computed cross-spectral matrix. The offset observed in task 1 is therefore not propagated to the following results. Acoustic images at several frequency bins were stored, and two of those (1000 Hz and 2000 Hz) are shown in Figure 3, with one plot pane showing the level difference in dB. The difference at the airfoil trailing edge is less than 1 dB; however, at the very outer edge, there is a larger difference. The discrepancy in this particular region was not investigated further, but with both source codes available, it is possible to identify the cause with some additional analysis.

In task 7, acoustic images were computed using the Clean-SC algorithm. In addition, a shear-layer correction was applied to account for sound wave refraction occurring at the Kevlar-wall interface. This correction, typically based on the work by Amiet [5], was not a requirement. The shear-layer correction implemented in AeroAcoustics.jl follows the derivation given in [14], and Acoular use a slot jet flow environment to mimic the (thicker) shear layer. Shear layer refraction is then considered using a ray-casting approach [31]. The acoustic maps are shown in Figure 4. In both cases, the position of the trailing edge has been successfully corrected for refraction effects, and are now located at $x = 0$, which is the correct physical position (see Figure 2). The



(a) 1000Hz *(b) 2000Hz*

Figure 3: Conventional beamforming without shear layer correction. Flow is right to left.

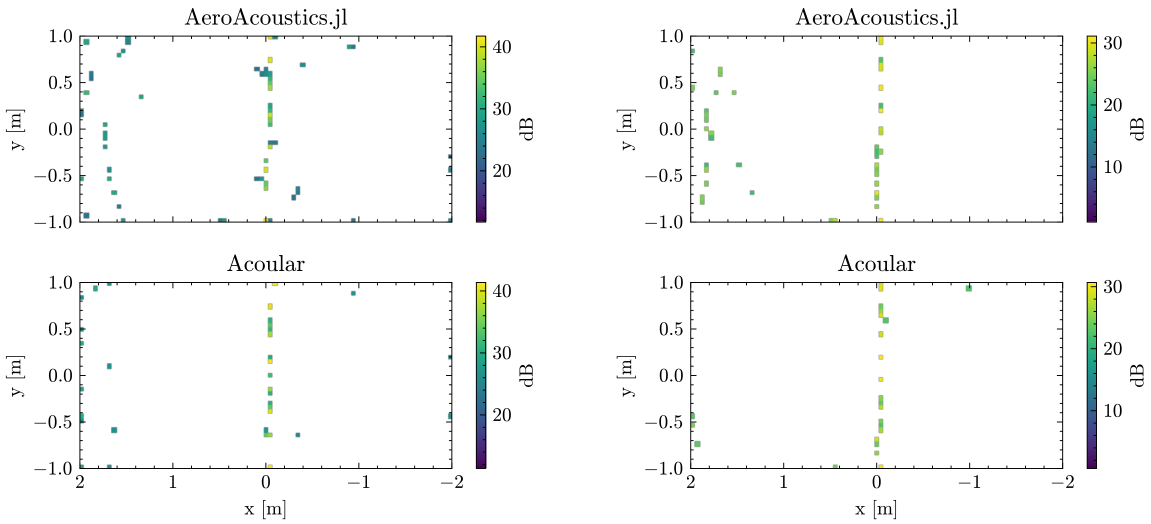
acoustic maps computed in task 3 (Figure 3) were computed without a shear-layer correction causing the trailing edge to appear convected down stream (around $x = 0.2$).

In tasks 6 and 8, source integration of the acoustic maps was performed for conventional beamforming and Clean-SC, respectively. A comparison of results is shown in Figure 5. For conventional beamforming, the acoustic maps were not compensated for the effect of the point-spread function (typically referred to as source power integration, SPI [36]) and therefore overestimate levels compared to Clean-SC. Overall, the results are very similar despite different implementations and software. The small discrepancies observed can be explained by studying the source code and could potentially be corrected if they are found unacceptable.

5 Discussion

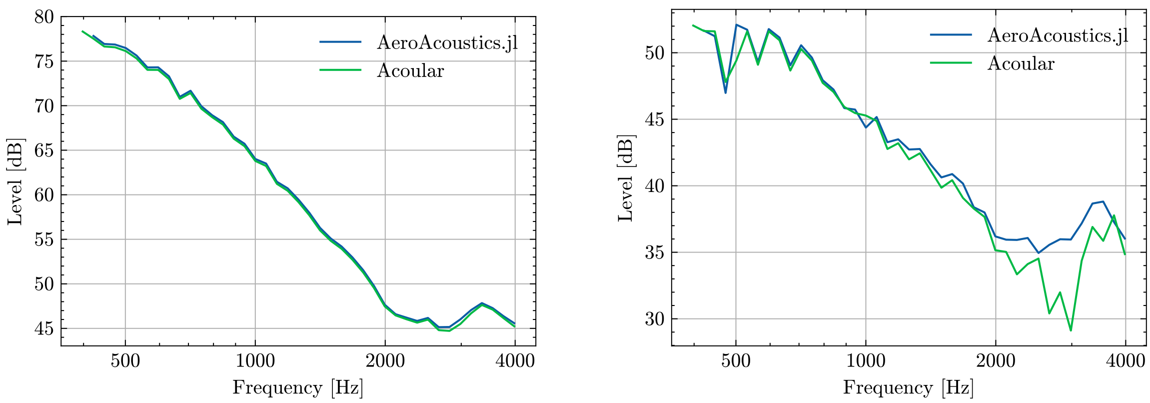
In this section, key topics related to open-source software are discussed: Peer-reviewing, teaching and community engagement.

Gaining recognition for developing open-source software within academia is challenging, as value is typically placed on peer-reviewed papers, which can be incompatible with constantly evolving and community-driven source code. This is, however, starting to change. The two software packages discussed in this paper have been through two different types of peer-reviewing. Acoular was described in a conventional journal paper [32], that introduced the software, but



(a) 1000Hz *(b) 2000Hz*

Figure 4: Clean-SC with shear layer correction. Flow is right to left.



(a) Conventional beamforming *(b) CleanSC*

Figure 5: Source integration results.

had to follow a conventional paper format and add an experimental part to get it accepted. Moreover, it turned out that some journals are not prepared to accept papers discussing tools instead of original research findings.

AeroAcoustics.jl has recently been through peer-review in the Journal of Open Source Software (JOSS)[37], which is a completely different format that traditional journals. The peer-review process is public and entirely online[5]. It is comprised of a short paper, that introduces the software, including a statement of need. Reviewers are recruited from the open-source community via their Github accounts and specfic domain-knowledge. A predefined review checklist is generated, that covers the online repository, including licence, installation, documentation, reproducibility of examples, and paper. The reviewers are also asked to assess whether a sub-

[5]https://github.com/openjournals/joss-reviews

stantial scholarly effort was made. At time of writing, the ten most cited papers from JOSS has over 20000 citations combined, this includes for instance Tidyverse [41], a data-manipulation and plotting package written in R [29] and Seaborn [39], a statistical-plotting library written in Python.

Open-source software can significantly enhance the teaching and learning experience. Installation instructions, comprehensive documentation, and practical examples are typically accessible online, facilitating the introduction and adaptation of the software for new users. As demonstrated by D'Andrea Fonseca et al. [10], employing open-source software in a teaching context can be highly beneficial.

Well-documented and freely available software enables students to apply what they have learned more quickly, allowing them to achieve tangible results. This immediate application can be both rewarding and motivating, enhancing the educational experience. Moreover, open-source software allows for the teaching of courses that might otherwise be too complex to manage. For instance, at TU Berlin, the "Microphone Array Project" course allows students to conceptualize, plan, and execute a small project related to microphone array applications within a single semester. Notably, some students have even published the outcomes of their group projects [13, 20], highlighting the value of incorporating open-source software into the curriculum.

Finally, adopting an open-source approach for research software, along with maintaining a dedicated online presence, can attract potential scientific collaborators and facilitate the initiation of joint projects. A recent example is the present paper, which significantly came into being because of the authors' commitment to the open-source philosophy and their shared interest in research involving microphone array signal processing.

6 Conclusion

The current state of open-source software for microphone array processing has been outlined, with two software packages described and several others mentioned. There is a growing movement towards open-source software and the establishment of an online community for microphone array techniques. A new benchmark exercise has been introduced and shared online, with the results of comparing two different software libraries presented. These results demonstrate a high degree of agreement in a range of typical post-processing tasks. Where discrepancies were observed, explanations were found through evaluation of the source code. Reproducibility of results, community engagement, and teaching are specific areas where open-source software proves beneficial. A broader recognition of open-source software is emerging in academia, with new forms of peer review and research grants being awarded for software development. The authors hope this paper inspires the aeroacoustic community to further embrace and adopt open-source practices, advancing collaboration and innovation within the field.

References

[1] M. Aberdour. "Achieving quality in open-source software." *IEEE Software*, 24, 58–64, 2007. doi:10.1109/MS.2007.2. URL `https://doi.org/10.1109/MS.2007.2`.

[2] Ackermann, Michael Markus and Fonseca, William D'Andrea, and Mareze, Paulo Henrique and Casagrande Hirono, Fábio. "Integration of multiple toolboxes for application in beamforming and aeroacoustics (original: Integracão de múltiplas toolboxes para aplicação em beamforming e aeroacústica)." In *In 12th Iberoamerican Congress of Acoustics (FIA 2020/22) & Meeting of the Brazilian Society of Acoustics - Sobrac*. Florianópolis, SC, Brazil, 2022. URL `https://bit.ly/fia2022-augen`.

[3] Aeroacoustics community. *Benchmarking Array Analysis Methods*. URL `https://www.b-tu.de/fg-akustik/lehre/aktuelles/arraybenchmark`, (Retrieved: 2024-05-20).

[4] A. Aksulu and M. Wade. "A comprehensive review and synthesis of open source research." *Journal of the Association for Information Systems*, 11, 6, 2010. doi:10.17705/1jais.00245. URL `https://doi.org/10.17705/1jais.00245`.

[5] R. Amiet. "Refraction of sound by a shear layer." *Journal of Sound and Vibration*, 58, 467–482, 1978. doi:10.1016/0022-460X(78)90353-X.

[6] C. J. Bahr, W. M. Humphreys, D. Ernst, T. Ahlefeldt, C. Spehr, A. Pereira, Q. Leclère, C. Picard, R. Porteous, D. J. Moreau, J. Fischer, and C. J. Doolan. "A comparison of microphone phased array methods applied to the study of airframe noise in wind tunnel testing." In *23rd AIAA/CEAS Aeroacoustics Conference, 2017*, pages 1–19. American Institute of Aeronautics and Astronautics, 2017. doi:10.2514/6.2017-3718. URL `https://doi.org/10.2514/6.2017-3718`.

[7] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. "Julia: A fresh approach to numerical computing." *SIAM Review*, 59, 65–98, 2014. doi:10.1137/141000671. URL `https://doi.org/10.1137/141000671`.

[8] 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, 856–879, 2006. doi:10.1016/j.jsv.2005.12.046. URL `https://doi.org/10.1016/j.jsv.2005.12.046`.

[9] P. Chiariotti, M. Martarelli, and P. Castellini. "Acoustic beamforming for noise source localization – reviews, methodology and applications." *Mechanical Systems and Signal Processing*, 120, 422–448, 2019. doi:10.1016/j.ymssp.2018.09.019. URL `https://doi.org/10.1016/j.ymssp.2018.09.019`.

[10] W. D'Andrea Fonseca, P. H. Mareze, F. Ramos de Mello, and C. C. da Fonseca. "Teaching Acoustical Beamforming via Active Learning." In *Proceedings of the 9th Berlin Beamforming Conference*. URL `https://www.bebec.eu/fileadmin/bebec/downloads/bebec-2022/papers/BeBeC-2022-D04.pdf`.

[11] W. D. Fonseca. *Beamforming-tools*, 2024. URL https://github.com/eac-ufsm/beamforming-tools, (Retrieved: 2024-05-20).

[12] A. Fuggetta. "Open source software—an evaluation." *Journal of Systems and Software*, 66, 77–90, 2003. doi:10.1016/S0164-1212(02)00065-1. URL https://doi.org/10.1016/S0164-1212(02)00065-1.

[13] S. Gareayaghi, I. Hagenmaier, H. Henze, and G. Herold. "Filtering of separate audio tracks from synchronously playing musical instruments using beamforming in the time domain." In *Proceedings of the 9th Berlin Beamforming Conference*. URL https://www.bebec.eu/fileadmin/bebec/downloads/bebec-2022/papers/BeBeC-2022-D05.pdf.

[14] S. Glegg and W. Devenport. *Aeroacoustics of low mach number flows: Fundamentals, analysis, and measurement*. 2017. ISBN 9780128097939.

[15] M. W. Godfrey and Q. Tu. "Evolution in open source software: a case study." *Conference on Software Maintenance*, pages 131–142, 2000. doi:10.1109/ICSM.2000.883030. URL https://doi.org/10.1109/ICSM.2000.883030.

[16] F. C. Hirono, P. Joseph, and F. M. Fazi. "An open-source implementation of analytical turbulence-airfoil interaction noise model." In *AIAA AVIATION 2020 FORUM*. 2020. doi:10.2514/6.2020-2544. URL https://doi.org/10.2514/6.2020-2544.

[17] D. H. D. H. Johnson and D. E. Dudgeon. *Array signal processing: concepts and techniques*. PTR Prentice Hall, 1993. ISBN 0130485136.

[18] A. Kujawski, A. J. R. Pelling, S. Jekosch, and E. Sarradj. "A framework for generating large-scale microphone array data for machine learning." *Multimedia Tools and Applications*, 2023. doi:10.1007/s11042-023-16947-w. URL https://doi.org/10.1007/s11042-023-16947-w.

[19] S. K. Lam, A. Pitrou, and S. Seibert. "Numba: A llvm-based python jit compiler." In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, LLVM '15, pages 1–6. Association for Computing Machinery, New York, NY, USA, 2015. ISBN 9781450340052. doi:10.1145/2833157.2833162. URL https://doi.org/10.1145/2833157.2833162.

[20] L. Liebich, A. Philipp, M. Strobl, T.-N. Wennemann, and G. Herold. "Acoustic investigation of wave-field-synthesised virtual sound sources using a 3D microphone array." In *Proceedings of the 9th Berlin Beamforming Conference*. URL https://www.bebec.eu/fileadmin/bebec/downloads/bebec-2022/papers/BeBeC-2022-D14.pdf.

[21] O. Lylloff. *Aeroacoustic wind tunnel tests*. Ph.D. thesis, Technical University of Denmark, Denmark, 2020. doi:10.11581/dtu:00000102. URL https://doi.org/10.11581/dtu:00000102.

[22] O. Lylloff. "Aeroacoustics.jl: A julia package for aeroacoustics." *Journal of Open Source Software*, 9(97), 6390, 2024. doi:10.21105.joss.06390. URL `https://doi.org/10.21105/joss.06390`.

[23] O. Lylloff, C. Bak, A. Fischer, A. S. Olsen, S. L. Ildvedsen, J. S. Beckerlee, M. Gaunaa, and R. F. Mikkelsen. "Plct-data: Naca63018 aeroacoustic - microphone array.", 2024. doi:10.11583/DTU.c.7222614. URL `https://data.dtu.dk/collections/PLCT-data_NACA63018_aeroacoustic_-_microphone_array/7222614/1`.

[24] O. Lylloff, E. Fernández-Grande, F. 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, 172–180, 2015. doi:10.1121/1.4922516. URL `https://doi.org/10.1121/1.4922516`.

[25] O. Lylloff, G. Herold, A. Kujawski, and E. Sarradj. *Airfoil in a Kevlar-walled Wind Tunnel*. URL `https://github.com/MicrophoneArrayBenchmarking/airfoil-in-kevlar-walled-windtunnel`, (Retrieved: 2024-05-20).

[26] 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." *CEAS Aeronautical Journal*, 10, 197–230, 2019. doi:10.1007/s13272-019-00383-4. URL `https://doi.org/10.1007/s13272-019-00383-4`.

[27] U. Michel. "History of acoustic beamforming." In *Proceedings of the 1st Berlin Beamforming Conference (BeBeC)*, pages 1–17. 2006. URL `https://www.bebec.eu/fileadmin/bebec/downloads/bebec-2006/papers/BeBeC-2006-01_Michel.pdf`.

[28] OSI. *The Open Source Definition*. URL `https://opensource.org/osd`, (Retrieved: 2024-05-20).

[29] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2023. URL `https://www.R-project.org/`, (Retrieved: 2024-05-20).

[30] E. Sarradj. "Three-dimensional acoustic source mapping with different beamforming steering vector formulations." *Advances in Acoustics and Vibration*, 2012, 1–12, 2012. doi:10.1155/2012/292695. URL `https://doi.org/10.1155/2012/292695`.

[31] E. Sarradj. "A fast ray casting method for sound refraction at shear layers." *International Journal of Aeroacoustics*, 16(1-2), 65–77, 2017.

[32] E. Sarradj and G. Herold. "A python framework for microphone array data processing." *Applied Acoustics*, 116, 50–58, 2017. doi:10.1016/j.apacoust.2016.09.015. URL `https://doi.org/10.1016/j.apacoust.2016.09.015`.

[33] E. Sarradj, G. Herold, A. Kujawski, S. Jekosch, A. J. R. Pelling, M. Czuchaj, T. Gensch, and S. Oertwig. “Acoular – Acoustic testing and source mapping software.”, 2024. doi:10.5281/zenodo.11234980. URL `https://doi.org/10.5281/zenodo.11234980`.

[34] E. Sarradj, G. Herold, P. Sijtsma, R. M. Martinez, A. Malgoezar, M. Snellen, T. F. Geyer, C. J. Bahr, R. Porteous, D. J. Moreau, and C. J. Doolan. “A microphone array method benchmarking exercise using synthesized input data.” In *23rd AIAA/CEAS Aeroacoustics Conference, 2017*. American Institute of Aeronautics and Astronautics, 2017. doi:10.2514/6.2017-3719. URL `https://doi.org/10.2514/6.2017-3719`, doi:10.2514/6.2017-3719.

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

[36] P. Sijtsma. “Phased array beamforming applied to wind tunnel and fly-over tests.” 2010. ISSN 01487191. doi:10.4271/2010-36-0514. URL `http://doi.org/10.4271/2010-36-0514`.

[37] A. M. Smith, K. E. Niemeyer, D. S. Katz, L. A. Barba, G. Githinji, M. Gymrek, K. D. Huff, C. R. Madan, A. C. Mayes, K. M. Moerman, P. Prins, K. Ram, A. Rokem, T. K. Teal, R. V. Guimera, and J. T. Vanderplas. “Journal of open source software (joss): design and first-year review.” *PeerJ Computer Science*, 2018, 2017. doi:10.7717/peerj-cs.147. URL `http://dx.doi.org/10.7717/peerj-cs.147`.

[38] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python.” *Nature Methods*, 17, 261–272, 2020. doi:10.1038/s41592-019-0686-2.

[39] M. L. Waskom. “seaborn: statistical data visualization.” *Journal of Open Source Software*, 6(60), 3021, 2021. doi:10.21105/joss.03021. URL `https://doi.org/10.21105/joss.03021`.

[40] P. Welch. “The use of fast fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms.” *IEEE Transactions on Audio and Electroacoustics*, 15, 70–73, 1967. doi:10.1109/TAU.1967.1161901. URL `http://doi.org/10.1109/TAU.1967.1161901`.

[41] H. Wickham, M. Averick, J. Bryan, W. Chang, L. D. McGowan, R. François, G. Grolemund, A. Hayes, L. Henry, J. Hester, M. Kuhn, T. L. Pedersen, E. Miller, S. M. Bache, K. Müller, J. Ooms, D. Robinson, D. P. Seidel, V. Spinu, K. Takahashi, D. Vaughan, C. Wilke, K. Woo, and H. Yutani. “Welcome to the tidyverse.” *Journal of Open Source Software*, 4(43), 1686, 2019. doi:10.21105/joss.01686. URL `https://doi.org/10.21105/joss.01686`.