AuditoryStimuli.jl: A Julia package for generating real-time auditory stimuli

Robert Luke[1]

1 Macquarie University, Macquarie University Hearing & Department of Linguistics, Australian Hearing Hub, Sydney, New South Wales, Australia

Summary

The `AuditoryStimuli.jl` software package provides researchers with a framework to generate real-time audio signals in the Julia programming language. The package is designed for use in auditory research programs, neurofeedback applications, and audio signal processing development. The package is developed on top of the SampledSignals library (Russell, 2016) to provide auditory specific functionality and encourage best practices in real-time audio generation. As such, modules are provided to generate auditory signals, modify these signals, and output the resulting waveforms. The package can be used to generate offline audio signals, but when used as a real-time audio system, it provides safe guards for common mistakes which can cause signal distortions and undesired auditory percepts.

Statement of need

There has been great improvement in standardising the analysis and post processing of research data in the scientific fields of auditory perception, auditory neuroscience, and brain computer interface research (Gramfort et al., 2014; Oostenveld et al., 2011). However, despite being critical for conducting reproducible experiments, standardised tools for the generation of auditory stimuli have typically not been developed with the same software engineering rigour. A large portion of stimulus signals for auditory research are generated with privately shared code fragments, without version control or efficient means of reporting errors. As such, this package provides a documented and version-controlled open-source framework for generating auditory stimuli.

`AuditoryStimuli.jl` is specifically developed for real-time audio research applications. A variety of software packages already exist for controlling the presentation of traditional block-design psychoacoustic experiments (Carcagno, 2012; Peirce et al., 2019; Schönwiesner & Bialas, 2021), and the post-processing the analysis of acoustic signals (Kim et al., 2017; Lossner, 2019; McFee et al., 2015). As such, this package does not focus on providing the scaffolding for traditional experimentation such as block design experiments or alternative forced choice procedures; users are directed to existing tools such as (Carcagno, 2012) for this purpose. Instead this package fills the need for a frame-based real-time signal processing framework. `AuditoryStimuli.jl` provides tools to generate real-time audio signals that can be dynamically adapted to the participant's responses or state. I.e., the stimulus properties may change continuously and dynamically on a scale of milliseconds, rather than in pre-generated signals on the scale of seconds.

Real-time audio stimulus generation is also required for brain computer interface applications. For example, methods have been developed to measure the brains response to sounds in real time (Cheveigné et al., 2018; Luke & Wouters, 2016). This package provides a real-time

framework which can be used to adapt the resulting audio according to an external signal, such as the brain state.

Real-time audio frameworks are required for developing audio signal-processing algorithms. For example, noise reduction, wind noise detection (Sapozhnykov, Harvey, & Luke, 2019), acoustic state detection (Luke et al., 2020; Sapozhnykov, Harvey, Erfaniansaeedi, et al., 2019), and speech enhancement algorithms typically run using frame based processing, with frames of 2-4 ms. `AuditoryStimuli.jl` provides the user with the ability to dynamically set processing parameters for each audio frame, enabling real-time signal processing development. Algorithm parameters can be varied and processing steps can be adaptively enabled.

Due to the real-time nature of the software package, fast mathematical computation is required. Audio signal processing is usually deployed using assembly or C programming languages, as these provide excellent processing speed. However, these low level languages require complex tooling and management of complex data structures, which increases development time and decreases the rate at which researchers can iterate on signal-processing designs. Instead `AuditoryStimuli.jl` is written in the Julia programming language (Bezanson et al., 2017), which provides the convenience of a high level language while providing excellent computational speed. The package is developed on top of the the SampledSignals library (Russell, 2016), which provides the sample- and frame-based infrastructure. `AuditoryStimuli.jl` builds on top of this framework to provide auditory specific features. `AuditoryStimuli.jl` also utilises the Unitful (Keller, 2016) framework for automatic handling of units, all quantities must be passed with units (e.g., sample rate and modulation rate must be passed as Hz, kHz, etc) and will be automatically validated. This automatic conversion and checking of units further reduces the chance of user error when using `AuditoryStimuli.jl`.

`AuditoryStimuli.jl` provides functions to generate real-time audio signals and avoid abrupt changes in the stimulus that cause unwanted perceptual distortions. For example, when modifier functions are instantiated, the user can specify the maximum rate of change of the parameters. Then during processing, the modifier functions will ensure that parameters can not change too quickly, which causes distortions such as clicks. The package provides signal generators for common auditory stimuli such as noise, tones, and multi tone complexes. Similarly, the package provides for both single channel and multi channel audio processing, e.g., binaural stimuli with user specified interaural coherence. Modifier functions are provided for modifying the amplitude, phase, frequency content, and amplitude modulation of the signals. And tutorials are provided in the documentation which demonstrate how these generators and modifiers can be used together to produce common research stimuli.

Taken together, the `AuditoryStimuli.jl` package fills a need within the auditory research community for a open-source real-time audio framework. The package addresses the communities need for a frame-based audio framework, that is computationally efficient and can be used in neuro feedback applications as well as audio processing development. An issue tracker and means for code contribution are provided, and the package provides installation instructions, documentation, and tutorials.

Audience

`AuditoryStimuli.jl` is designed for students and researchers of all levels in the fields of auditory perception science, auditory neuroscience, neurofeedback research, and audio signal processing engineering.

Notes

This software was used to generate stimuli for (Luke et al., 2021).

References

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Review*, *59*(1), 65–98. https://doi.org/10.1137/141000671

Carcagno, S. (2012). Pychoacoustics. In *GitHub repository*. GitHub. https://github.com/sam81/pychoacoustics

Cheveigné, A. de, Wong, D. D. E., Di Liberto, G. M., J., H., M., S., & Lalor, E. (2018). Decoding the auditory brain with canonical component analysis. *NeuroImage*, *172*, 206–216. https://doi.org/10.1016/j.neuroimage.2018.01.033

Gramfort, A., Luessi, M., Larson, E., Engemann, D. A., Strohmeier, D., Brodbeck, C., Parkkonen, L., & Hämäläinen, M. S. (2014). MNE software for processing MEG and EEG data. *Neuroimage*, *86*, 446–460. https://doi.org/10.1016/j.neuroimage.2013.10.027

Keller, A. (2016). Unitful.jl. In *GitHub repository*. GitHub. https://github.com/PainterQubits/Unitful.jl

Kim, J. W., Russell, S., & Bello, J. (2017). Fast music and audio processing using the julia language. *Audio Engineering Society Conference: 2017 AES International Conference on Semantic Audio*.

Lossner, J. (2019). Spatially oriented format for acoustics (SOFA) API for python. In *GitHub repository*. GitHub. https://github.com/spatialaudio/python-sofa

Luke, R., Larson, E., Shader, M. J., Innes-Brown, H., Van Yper, L., Lee, A. K., Sowman, P. F., & McAlpine, D. (2021). Analysis methods for measuring passive auditory fNIRS responses generated by a block-design paradigm. *Neurophotonics*, *8*(2), 025008. https://doi.org/10.1117/1.NPh.8.2.025008

Luke, R., Sapozhnykov, V., & Harvey, T. I. (2020). *Blocked microphone detection* (Patent No. 10,412,518). https://patents.google.com/patent/US10412518B2/en

Luke, R., & Wouters, J. (2016). Kalman filter based estimation of auditory steady state response parameters. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *25*(3), 196–204. https://doi.org/10.1109/TNSRE.2016.2551302

McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., & Nieto, O. (2015). Librosa: Audio and music signal analysis in python. *Proceedings of the 14th Python in Science Conference*, 8. https://doi.org/10.25080/majora-7b98e3ed-003

Oostenveld, R., Fries, P., Maris, E., & Schoffelen, J.-M. (2011). FieldTrip: Open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data. *Computational Intelligence and Neuroscience*, *2011*. https://doi.org/10.1155/2011/156869

Peirce, J., Gray, J. R., Simpson, S., MacAskill, M., Höchenberger, R., Sogo, H., Kastman, E., & Lindeløv, J. K. (2019). PsychoPy2: Experiments in behavior made easy. *Behavior Research Methods*, *51*(1), 195–203. https://doi.org/10.3758/s13428-018-01193-y

Russell, S. (2016). SampledSignals.jl. In *GitHub repository*. GitHub. https://github.com/JuliaAudio/SampledSignals.jl

Sapozhnykov, V., Harvey, T. I., Erfaniansaeedi, N., & Luke, R. (2019). *Headset on ear state detection* (Patent No. 10,812,889). https://patents.google.com/patent/US10812889B2/en

Sapozhnykov, V., Harvey, T. I., & Luke, R. (2019). *Wind noise measurement* (Patent No. 10,504,537). https://patents.google.com/patent/US10504537B2/en

Schönwiesner, M., & Bialas, O. (2021). S(ound)lab: An easy to learn python package for designing and running psychoacoustic experiments. *Journal of Open Source Software*, *6*(62), 3284. https://doi.org/10.21105/joss.03284