University of Zurich[UZH]

Institute of Computational Linguistics

Bachelorarbeit
zur Erlangung des akademischen Grades
Bachelor of Arts
der Philosophischen Fakultät der Universität Zürich

Evaluation of the Application of the Open-Source Python Toolbox MNE for the Assessment of Neural Processing of Degraded Speech from Functional Near-Infrared Spectroscopy

Rebecka Fahrni
Computational Linguistics
18-735-522
November 30, 2022

Supervision
Dr. Daniel Friedrichs
MA Andrew Clark

Institute of Computational Linguistics
in collaboration with
The Linguistic Research Infrastructure (LiRI) Lab

Abstract

This work evaluates the Python toolbox for brain data analysis, called MNE-Python. For this purpose, a functional Near-Infrared Spectroscopy (fNIRS) dataset is used with the main goal of assessing the neural processing of degraded speech, which is a highly debated area of research regarding a pathway for intelligible speech. With more and more Python users and more fNIRS technology applied for inferring brain activity, it is only natural that more Python toolboxes are available for data analysis. Many of them, however, are not yet fully developed and still need many adaptations and feature enhancements. In the results, the capabilities, such as the generation of nice user-friendly fNIRS response diagrams, and other possibilities offered by MNE-Python are shown. Since it enables a first-level fNIRS analysis, the results indicates that the MNE is a good Python toolbox for fNIRS data analysis. Nevertheless, its current limitations will also be exposed.

Keywords: fNIRS · MNE-Python · degraded speech · intelligible speech

Acknowledgments

I would like to express my sincere gratitude to my supervisors Daniel Friedrichs and Andrew Clark who made this work possible. They gave me the unique opportunity to challenge myself and expand my knowledge in the field of data analysis on recorded brain data. Thanks to them I learned a lot about fNIRS, a neuroimaging technique I was previously unfamiliar with. Their guidance, advice, and patience helped me remain calm at times when I initially didn't fully understand certain matters or encountered difficulties.

Another special thanks goes to the MNE development team, including Eric Larson, Stefan Appelhoff, Richard Hoechenberger, Dan McCloy, and Robert Luke, who answered my numerous questions posted on the MNE forum[1], who added a special feature upon request, and who took the time to hold a live session where I could get instant assistance.

Finally, I would like to thank Peter Koning, Anki, Heinz, Moritz, Sebastian, Marie-louise and Raphael for supporting me.

[1]MNE forum for support and discussion on MNE and related software follow the link https://mne.discourse.group/

Contents

Chapter 1

Introduction

In this thesis the initial idea was to investigate a pathway by assessing the neuronal processing for degraded speech using functional nearinfrared spectroscopy (fNIRS). The experimental design and data collection were conducted by Daniel Friedrichs and Andrew Clark in 2017 at University College London (Friedrichs et al., 2019). They are performing an enhanced replication of a study which is intended to investigate neural activity in relation to degraded speech. My work and contribution in this thesis is primarily concerned with learning and applying fNIRS data analysis on the data they provided, whilst familiarizing myself with fNIRS and using the MNE-Python package for the analysis. Hence, to give an evaluation of the potential and capabilities of MNE-Python.

The data obtained for this project were from three main files per participant, namely the *.csv* files containing the recorded fNIRS measurements from two probes and a positional *.pos* file containing the coordinate data for the montage. Moreover, the code that generates all the plots, including the data of this project is accessible on GitHub via the following link: `https://github.com/rfahrn/fNIRS-project.git`. There are several Python scripts in the GitHub repository, including some that are only for debugging or problem solving purposes. For this project, only three major Python scripts need to be used, namely *"pos_convert.py"*, *"manuel_montage.py"*, and *"Preprocessing_individual.py"*. The first two are used for the montage, while the last one is used for first-level analysis.

The secondary underlying research question concerns the assessment of neural processing of degraded speech. The auditory cortex and its surrounding higher-level regions have gained increasing attention in recent years. Lately, the focus has been on how the auditory ventro-temporal information stream is processed along both hemispheres. At the center of attention is the underlying motive to identify a pathway for intelligible speech or, conversely, unintelligible, degraded speech, using a variety of neuroimaging techniques for purposes of verification. In this study, fNIRS was used for this purpose.

There are multiple personal reasons why I opted for this topic. One being that the study of the brain, i.e. neuroscience, has fascinated me ever since I was a young child, got hit by a car and had to learn how to walk again. Over the years, my curiosity in the human brain grew. For this reason, I enrolled in several courses related to neurolinguistics/psycholinguistics, neuroscience, and neuroinformatics during my undergraduate studies. The brain is so incredibly complex that even if we get a glimpse of the understanding of the core function of some brain areas, we have only scratched the surface. There is still much to learn about the human brain and its neural networks. A better understanding of the underlying functions of the human brain helps us to understand ourselves, others and people with cognitive impairments. One goal for the future must be to provide even better support and help for people with cognitive impairments, for example after a stroke. Hence, I was very eager to learn more about neurophysiological data analysis using fNIRS and enthusiastic to pursue this project.

Several challenges and issues arose during the implementation of the data analysis pipeline with MNE-Python. As a result, the scope of the original project had to be modified. This led to a shift in focus, away from presenting the final results of the main research question, namely the assessment of neuronal processing of degraded speech, towards an evaluation of the used MNE-Python data analysis series. Nevertheless, this work is able to deliver first-level analysis results of each subject, representing an important first step in the overall analysis of the dataset. In addition, the reader will gain a broader understanding of fNIRS technology, a

deeper understanding of the steps and pipeline required for fNIRS data analysis, along with a subsequent evaluation of the MNE Python as a suitable data analysis package used for the fNIRS data collected using a ETG-4000 optical tomography system by Hitachi High-Technologies Corporation, Tokyo, Japan.

This Chapter provides a brief introduction to the topic of the research and my motivation. In the next Chapter, *Background 2*, the reader gets a broad overview of the background on the main research question the experiment was designed to investigate. This is done by first clarifying terminology and then referring to recent research and current findings. In addition a brief introduction to the neuroimaging technique fNIRS which was used for the experiment is provided. Moreover, there is also a section describing the 2017 experimental design conducted by Daniel Friedrichs and Andrew Clark.

Next, in Chapter *Data Analysis Method 3*, the neurophysiological fNIRS data analysis method is explained, covering the steps that are necessary for the overall data analysis, where the analysis is currently at, and which specific steps remain and still need to be implemented.

Later, Chapter *Results 4*, covers the current results obtained by using MNE-Python.

Afterwards, in Chapter *Discussion and Conclusion 5*, the challenges encountered are explained and an overall evaluation of the toolbox is given.

And finally, in the last Chapter *Outlook 6*, the reader learns what remains to be done in the future to definitively assess the neural response with the collected fNIRS data.

Chapter 2

Background

For a better comprehension of the data analysis of the data collected by the fNIRS experiment and to understand the main research question of the experiment, in Section, *Topic of Research 2.1*, the reader is guided through the background of the research topic by first defining the terminology 'intelligible speech' then referring to similar research studies and their results.

Secondly, in Section, *fNIRS Technology 2.2*, a neuroimaging method is introduced that has recently become increasingly popular, functional near-infrared spectroscopy (fNIRS). This will be done by presenting the physical and physiological basics and explain how to obtain neuronal activity by fNIRS.

In Section, *Method and Experimental Design 2.3*, the utilized methodology for the investigation of the topic of research is delineated. By answering the question why functional Near-Infrared Spectroscopy (fNIRS) was the preferred neuroimaging technology for the task of discovering a pathway for intelligible speech, respectively assessing the neuronal processing of degraded speech. In addition, I also illustrate the experimental design conducted by Daniel Friedrichs and Andrew Clark in 2017 for this study.

Finally, in Section, *MNE 3.1*, background and supplementary information on the Python toolbox MNE as it is used for the overall data analysis are provided.

2.1 Topic of Research

Before discussing the experimental design some background information about speech are provided, where first the term "intelligible speech" is defined. An overview of the current research addressing the specific brain areas associated with the pathway for the perception of intelligible speech will be given.

Intelligible Speech

Human language follows a structured form of combining word forms, syntax, and word/sentence semantics, enabling us to think, relate to the environment, or even imagine things. It is therefore easy to say that language is more than just speech, the sounds or signal we produce when we speak. Yet speech must fulfill a certain criteria in order for people to understand each other perfectly. As early as 1947, French and Steinberg quantitatively analyzed the basic characteristics of speech, hearing, and noise in terms of how the ear recognizes speech sounds.

Speech is a complex stimulus, consisting of a succession of sound signals whose intensity and frequency vary rapidly from moment to moment, with large natural variations between the acoustic properties of the same sounds spoken by different people or by the same person at different times. All these acoustic phonetic features must be processed before becoming intelligible to the listener. Supposing that the various components of a speech signal are received by the ear in their initial order and at their temporal spacing, the listener's success in recognizing and interpreting these components will depend on the intensity within their ear and the intensity of unwanted sounds that may be present, both as a function of frequency (French and Steinberg, 1947).

Consequently, they quantified and evaluated several factors such as loudness, reduction of hearing sensitivity through masking, which means reducing unwanted sounds in the listener's ear and they also computed the effect of noise levels and hearing loss. Based on the findings they assessed the articulation index (AI), which is a way to quantify this relationship between acoustic speech cues and intelligibility

(French and Steinberg, 1947). The Articulation index is weighted fraction representation, for a given speech channel and noise condition, meaning it reflects the degree of segregation of speech from background or other system noise. This calculated proportion ranges form 0 to 1 and can subsequently be used to predict the speech intelligibility (Amlani et al., 2002; French and Steinberg, 1947).

Ultimately, the term speech intelligibility refers to the degree to which speech signals can be correctly recognized and understood by the listener in a given environment. Speech intelligibility must therefore encompass multiple properties of human language, such as word-form recognition, syntax and semantics.

Nevertheless, there is no single specific acoustic cue that is crucial to determine the intelligibility of speech, since skilled listeners can extract meaning even from degraded speech signals (Miller, 1951; Shannon et al., 1995). For example, Shannon et al., in 1995, observed that despite the conditions with greatly reduced spectral information and preserving only temporal cues, listeners are typically able to understand speech rapidly and accurately even if the speech has been significantly degraded. This concludes that there is no single acoustic cue essential for correct perception of speech sounds and leads to the conclusion that the higher-lever regions in the brain take on a major role in the further processing of more complex stimuli. Indeed, evidence from functional imaging indicates that the human auditory cortex has a hierarchical organization, meaning that neurons from specific brain regions respond more selectively to more complex stimuli (Okada et al., 2010).

Considering that speech can be either produced or perceived, one distinguishes between two aspects, the perception and the production of intelligible speech. Perception of intelligible speech has already been largely covered, therefore intelligible speech production will now be commented on. Nevertheless, it is to say that the presented research topic within this thesis concentrates primarily on the perception of intelligible speech.

Regarding the uniqueness of humans to produce clearly intelligible speech, despite the fact that other species, such as primates, also have the vocal anatomy to produce vocalisations, a recent study suggest that this is mainly due to the unique

evolution and structure of the human brain and is not related to vocalization-related anatomical differences between humans and primates (Fitch et al., 2016). This means that neural changes and the formation of neural circuits are fundamental for humans to produce intelligible speech. The researchers used X-ray video to quantify vocal tract dynamics during vocalisations and found that while macaques have a vocal tract capable of producing intelligible speech, they lack the neural circuits needed to control and coordinate their vocal production system (Fitch et al., 2016).

The study clearly illustrates the importance of the neuronal circuits and neuronal pathways in our brains.

All this new scientific knowledge in fields of neuroscience, psycho-linguistics, and computational neuroscience have led to increased curiosity in further research on speech production and perception regarding the brain. In the following, the findings of current research related to the perception of intelligible speech are elaborated on by highlighting a potential brain pathway for intelligible speech versus degraded speech.

Neural Pathway for Intelligible Speech Perception

There is consensus between experts that the human auditory cortex is organized in a tonotopical manner, i.e., haricells that are spatially close to each other on the basilar membrane in the cochlea are also tuned to tones of adjacent frequency (Purves et al., 2001). The auditory nerve fibers arising from the spiral ganglion cells forming the auditory nerve that transmits these action potentials to the brain, are sensitive to a certain frequency tuning curves, meaning that they have a preferred intensity across all frequencies for which the the neuron increases its firing rate of an action potential above its spontaneous firing level (Purves et al., 2001). This order is then maintained when projected onto the auditory cortex (Talavage et al., 2004).

Researchers observed with functional imaging, namely positron emission tomography (PET) and functional magnetic resonance imaging (fMRI) an organization in the human auditory cortex (Scott et al., 2000; Wessinger et al., 2001), suggesting

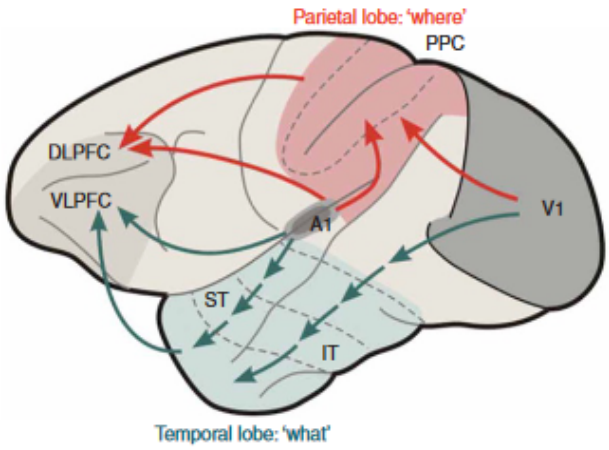


Figure 1: "What" and "where" pathway. V2, primary visual cortex; A1, primary auditory cortex; IT, inferior temporal region; ST, superior temporal region; PPC, posterior parietal cortex; VLPFC, ventrolateral prefrontal cortex; DLPFC, dorsolateral prefrontal cortex (Rauschecker and Scott, 2009).

that the human auditory cortex has a hierarchical structure similar to the hierarchy identified in the visual system. The primary visual cortex (V1) has specifically tuned neurons for a receptive field to detect contours. However, downstream, in higher-level regions of the ventral visual stream, which is also refereed to as the "What Pathway", see Figure 1, and associated to object and form recognition (Mishkin et al., 1983; Hickok and Poeppel, 2004), the features to which neurons are tuned to fire become more complex, with neurons responding to complex images in a study with monkeys (Gross et al., 1969). Quiroga et al. (2005) performed a magnetic resonance imaging (MRI) study in humans and came to the same conclusion, finding that neurons in the human medial temporal lobe (MTL) fire selectively to images of faces, animals, objects, or scenes.

Now, analogue to this ventral visual pathway leading from the primary visual cortex (V1) to the temporal lobe, an auditory pathway was discovered. More specifically Scott et al. (2000) and Wessinger et al. (2001) showed that pure tones, having a simple feature, are sufficient to drive activity in auditory cortex, which is located bilaterally in humans in the upper surface of the temporal lobe and includes

planum temporale, parts of the transverse temporal gyrus and the super temporal gyrus. However, for a maximal activity in surrounding auditory-responsive cortical regions, the stimuli presented needs to be more complex, such as band-pass noise or speech.

Nevertheless, it is more difficult to identify and design stimuli that excite tuned neurons in surrounding higher-order brain regions along the ventro-lateral stream of acoustic information to their maximum firing activity. Let me explain the reason.

Firstly, to study brain regions that are most active for intelligible speech, one cannot simply test the higher-level brain regions for images as in the visual ventral pathway, intelligible speech at the acoustic level being more complex as there is not one acoustic cue that determines intelligibility of speech (Blesser, 1972; Scott et al., 2000).

Secondly, there exists a controversy about the degree of lateralization and the precise location of responses to intelligible speech. Scott et al. (2000) have demonstrated using PET that the left superior temporal sulcus (STS) responds to phonetic information, whereas the anterior superior temporal sulcus (aSTS) preferentially responds to intelligible speech. However, in another study by Okada et al. (2010), observed using fMRI that activation was bilateral and claimed that the reason other studies were unable to show a bilateral effect was simply because they did not use enough participants. They found that the superior temporal gyrus (STG) is sensitive to auditory features. Whereas in terms of sensitivity differences to intelligibility they observed higher activation bilaterally in downstream auditory regions in both the anterior superior temporal sulcus (aSTS) and posterior superior temporal sulcus (pSTS). Nevertheless, they reported differences in acoustic invariance index between region of interest (ROI); HG, aSTS pSTS and mSTS in the left and right hemisphere.

The acoustic invariance index is a measure of the size of the intelligibility effect relative to the size of the acoustic effect using intelligibility classification contrasts (clear vs. rot + NV vs. rotNV) and subtracting the 2 acoustic classification contrasts (clear vs. NV + rot vs. rotNV) (Okada et al., 2010; Blesser, 1972). Thus, through this index they were able to quantify the significance of the sensitivity to intelligible

speech of ROI.

Interestingly, they found for example that pSTS regions have the highest degree of auditory invariance bilaterally during processing of speech, which led them to hypothesize that pSTS regions are involved in the recognition of auditory objects of speech and suggest that pSTS regions of both hemispheres give representations on phonological level.

Okada et al. (2010) conducted pattern classification in a univariate way. Another study by Evans et al. (2014) replicated but extended the work of Okada et al. (2010) by using multivariate pattern analysis. They also identified bilateral activation along the STS. Furthermore, they observed that the strongest univariate intelligibility effects were present in the left aSTS with their multivariate pattern analysis backing to this assessment. Further, they added that there must be a much wider network for intelligibility involving the inferior parietal and frontal cortex, such as Wernicke's, Broca's, and Geschwind's areas, also suggested by Abrams et al. (2013).

One commonality that all studies have regarding their approach to show differences between degraded unintelligible speech and intelligible speech is the reconstruction of acoustic speech stimuli that are as acoustically complex as speech but lack phonetic features, making them unintelligible. A well established technique for destroying comprehensibility while maintaining structural complexity is, for example spectral rotation technique, which was introduced by Blesser (1972). More on all the different types of stimuli for identifying intelligible vs. unintelligible speech and how the activation of brain areas is identified using the stimuli information is further discussed in Section *Methods and Experimental Design 2.3*, where the stimuli and the neuroimaging technique used are explained more extensively.

Similar to the studies discussed, the goal of this research topic is to identify a pathway for intelligible speech by assessing the neuronal processing of degraded speech as a preliminary step. This is done by replicating and extending the study by Scott et al. (2000). More in Section *Methods and Experimental Design 2.3*, where the experimental design of the experiment is described.

2.2 fNIRS Technology

In order to understand the collected data used for this thesis, a background on how we infer neuronal signals using fNIRS is provided. The term fNIRS is specifically used in the neuroimaging field to address NIRS applications aiming at mapping and gaining an understanding of the functioning of the human cerebral cortex.

Functional Near-Infrared Spectroscopy (fNIRS) is a portable and noninvasive optical neuroimaging technique that measures hydrodynamic alterations accompanying brain activation by exploiting changes in light properties, more specifically the changes in tissue absorbance of light at different wavelengths. A light source, called optode, sends light waves in the near-infrared range (650-950 nm) through the cortex and the tissue refracts the scattered light back to a detector (Venclove et al., 2015).

Physical Principles

Currently, there are three modalities of fNIRS spectroscopy: 1.) Continuous wave, 2.) Frequency domain, 3.) Time-domain. Only the first, continous wave modality of fNIRS will be covered, since an overwhelming majority of commercial devices developed to date are based on continuous wave technology and because it is the method used (Scholkmann et al., 2014). Generally, near-infrared spectroscopy (NIRS) relies on two characteristics of human tissue; The relative transparency of tissue to light in near-infrared (NIR) spectrum and the oxygenation-dependent light absorbing characteristics of hemoglobin. The principle that biological tissue is relatively permeable to light in the near-infrared (NIR) range enables the determination of concentration changes of oxygenated hemoglobin([HbO_2]), deoxygenated hemoglobin([HbR]), total blood volume ([HbO_2]+[HbR]= HBtot) and oxygenated cytochrome oxidase[1] (Chance, 1991). The reason being that if the absorption of light is know, the

[1]Cytochrome oxidase (CO), is a valuable endogenous metabolic marker for neurons since the nervous system strongly depends on aerobic metabolism for its energy supply and cytochrome oxidase has an intrinsic role in mitochondrial aerobic energy metabolism (Wong-Riley, 1989).

Lambert-Beer law can be used to calculate the chromophore's[2] absorption. The Lambert-Beer law is given by (Bouguer, 1729; Mayerhöfer et al., 2020; Mäntele and Deniz, 2017):

$$OD_\lambda = \log(\frac{I_0}{I}) = A = \varepsilon_\lambda \cdot c \cdot \ell$$

where

A is the amount of light absorbed, which can also be defined via incident intensity

$$A = \log(\frac{I_0}{I})$$

where I_0 is the incident light, I the transmitted light

OD_λ is the optical density of the medium, a dimensionless factor

ε is the molar absorbance coefficient or also known as chromophore's extinction coefficient, which varies with wavelength λ, but not with concentration

c is the concentration of the chromophore

ℓ is distance between light entry and exit point

λ is the wavelength

Note that OD_λ is equal to A, which means that the density of a medium is equal to absorbance of a medium.

However, the Lambert-Beer law is only meant to be used in a non-scattering and transparent medium however biological tissue is a scattering medium. Therefore a dimensionless path-length correction factor, which accounts for the increase in optical pathlength due to scattering in the tissue needs to be incorporated, this factor is called differential pathlength factor (DPF) (Essenpreis et al., 1993). The equation for a scattering medium is then known as modified Lambert-Beer law (MBLL) and is given by (Kocsis et al., 2006; Baker et al., 2014):

[2] Chromophore is a group of atoms and electrons forming part of an organic molecule that causes it to be coloured (Muller, 1994, on page 1097)

$$\Delta c = \frac{\Delta OD\lambda}{\varepsilon\lambda \cdot \ell \cdot DPF}$$

where

$\Delta OD\lambda$ is the change in optical density more specific the oxygen-independent optical losses due to scattering and absorption in the tissue. $\Delta OD\lambda$ is assumed to be constant during NIRS measurement.

Δc is the change in concentration of the chromophore

$\varepsilon\lambda$ is the molar absorbance coefficient or also known as chromophore's extinction coefficient, which varies with wavelength λ, but not with concentration

ℓ is distance between light entry and exit point

DPF is the differential pathlength factor

In summary, the following can be concluded about the fNIRS technology: The determination of light waves in the NIR range for hemoglobin concentration follows the MBLL and is the basis for continuous near-infrared tissue spectroscopy (cwNIRS) (Kocsis et al., 2006). The differential form of MBLL, dMBLL, implies that light attenuation changes are proportional to changes in tissue chromophore concentration, thus mainly due to oxy- and deo-hemoglobin changes. So, once the attenuation changes are measured at two or more wavelengths, it is possible to calculate the concentration changes (Kocsis et al., 2006).

The mechanism of neurovascular coupling is the underlying principle of fNIRS (Girouard and Iadecola, 2006). The cerebral blood flow needs to be adjusted and maintained to the level of brain activity and metabolic demand this is done by feedforward and feedback processes of autoregulation and neurovascular coupling (Girouard and Iadecola, 2006). When a particular brain region is activated, cerebral blood flow increases in a temporally and spatially coordinated pattern that is closely

associated with changes in neuronal activity through a complex sequence of coordinated events involving neurons, glia, arteries/arterioles, and signaling molecules (Girouard and Iadecola, 2006). As a result, fNIRS enables inferences about changes in neuronal activity reflected in changes in blood oxygenation in the region of the activated cortical area Devor et al. (2012). It needs to be said, that the obtained singal by fNIRS is analogous to the Blood-oxygen-level-dependent imaging (BOLD) signal measured by fMRI as it is capable of measuring changes both in oxy- and deoxyhemoglobin concentration, hence fNIRS research literature often refer to BOLD measures in their analysis.

Neurovascular coupling is a unique mechanism that controls regional cerebral blood flow (CBF) and ensures a rapid increase in the rate of CBF to activated brain structures

2.3 Method and Experimental Design

The method specification and its experimental design, including the choice of the device, participant selection data generation, and data collection, was carried out by Daniel Friedrichs and Andrew Clark in 2017 at UCL with the goal to replicate and extend the study by Scott et al. (2000) using fNIRS.

Neuroimaging Method

Reviewed in the Chapter *Background 2* there is a controversy regarding the degree of lateralization of the neuronal responses on intelligible speech. Scott et al. (2000) using PET have found activation in the left aSTS. Okada et al. (2010) using fMRI has however found bilateral activation. Every neuroimaging technique has its advantages and disadvantages. PET, for example, has the advantage that it directly measures neural activity and is less sensitive to motion artifacts, while the disadvantage is that it has low spatial and temporal resolution, whereas fMRI, on the other hand, offers good spatial resolution but similarly poor temporal resolution, is susceptible to motion artifacts, only infers neura activity by using the BOLD principle, and is

noisy. Both neuroimaging techniques have long been validated for use in auditory perception experiments. However, since they are not portable and are expensive, research has therefore emphasised the use of fNIRS.

Steinmetzger et al. (2020), for example, validated the application of fNIRS in the context of auditory perception experiments by comparing haemodynamic fNIRS data with obtained EEG electrophysiological cortical responses. The authors indicate that the spatial correspondence of the results obtained with the two methods shows that fNIRS is a valid tool for the study of auditory perception. They noted, however, that interpretation of fNIRS-HbO results is difficult because of changes in cortical blood flow, known as "blood stealing". In conclusion, they shows that the advantages of combining fNIRS and EEG go far beyond the common notion of combining the good spatial resolution of blood-based measurements with the good temporal resolution of electrophysiological data. This year, 2022, Steinmetzger et al. in a similar study of interrelationship between the hemodynamic and electrophysiological cortical responses evoked by voice pitch changes, found consistent with other neuroimaging stuides, that there is a right-lateralised activity in secondary auditory regions using again combined fNIRS and EEG. They reported similar to the Steinmetzger et al. (2020) study, that the distribution of cortical activity measured by fNIRS and EEG was largely coherent, with the sensitivity of the EEG data being higher than that of the fNIRS data.

The decision to use fNRIS neuroimaging was made based on the fact that previous research has validated the use of fNIRS for auditory perception. Moreover, it would be interesting if this replication of Scott et al. (2000) who used fMRI shows similar results when using fNRIS. Beyond that, fNIRS has several advantages: it is relatively inexpensive, portable, shows low sensitivity to motion artifacts, and has a relatively high temporal resolution.

Participants

The experiment was initially performed with 26 participants, 5 of whom were excluded, due to three of them having too many noisy channels in the end and two being left-handed. The remaining 21 participants were right-handed English speakers with no known hearing or speech impairments. Among them, 8 were female and 13 were male, all ranging in age from 18 to 54 with an overall mean age of 29.

Stimuli and Procedure

The participants were presented with five different types of auditory stimuli previously used to identify speech selective regions (Scott et al., 2000). The five stimuli were:

1. clear speech sentences (Sp)
2. Noise-vocoded speech (NV or VCo)
3. Rotated noise vocoded speech (NV-Rot or RVCo)
4. Rotated speech TS (Rot-TS)
5. Rotated speech Blesser (Rot-Blesser or RSp)

The first two types, clear speech and noise-vocoded speech, are perceived as intelligible and the last three as unintelligible. All five stimuli were based on natural sentences recorded by a single female speaker. The original unprocessed speech formed the stimuli in one condition (Sp). The second condition perceived as intelligible is termed 'noise-vocoded speech' (NV or VCo). This is a form of distortion that was developed by Bob Shannon (Shannon et al., 1995) to simulate the experience of hearing speech transduced by a cochlear implant, yielding a stimulus that sounds 'like a harsh whisper' (Davis et al., 2005). However in order for noise-vocoded speech to be intelligible, listeners have to be pre-trained.

The three last stimuli conditions, are unintelligible and involve the spectral rotation (or inversion) of a signal.

Pre-training Noise-Vocoded Speech Stimuli for Intelligibility

Prior to participating in the main experiment, participants were trained on the NV stimuli, since noise-coded speech is generally intelligible after some listening, although it may be difficult for some subjects to understand on the first hearing. Subsequently, participants were asked to engage in a task in which a short sentence or phrase was played through the participant's headphones ER (Etymotic Research headphones) and then on a display screen in front of them they had to select two heard words from a list of 6-8 words. If they chose the correct words, the next sentence would play. This routine continued for a period of 7 minutes. Finally, after completing the short task, all participants were asked whether their understanding of noise-vocoded speech has improved.

fNIRS Recording

The fNIRS signals were recorded with a continuous-wave ETG-4000 system (Hitachi high-technologies corporation, Tokyo, Japan) with a sampling rate of 10 Hz. Two 15 source optodes and 15 detector optodes in a 3 by 5 configuration were placed symmetrically over each hemisphere. The source optodes emitted infrared light with wavelengths of 695 and 830 nm. The chosen optode layout was devised to optimally cover the auditory cortex and associated areas, resulting in 22 measurement channels per hemisphere. Note that the Hitachi fNIRS system has a fixed optode spacing of 3cm therefore there are no short channels. The optode and reference positions for each individual participant were digitized with a Polhemus Patriot system (Colchester, Vermont, USA) before the experiment. The Hitachi ETG-4000 device recorded 10 trials per condition, i.e. 5 times 10 trials, making a total of 50 events. Between the events there was a recovery period of 12 seconds, because the neuronal activity must decrease again in order not to falsify the result.

Contrasts

In order to eventually identify, compare and measure specific local activities in the human auditory cortex, five contrasts are defined, which are used in the later data analysis. The five contrasts are:

1. Intelligible versus unintelligible speech:
   [(Sp + NV) − (Rot-Blesser + Rot NV)]

2. Pitch perception with Blesser's old rotation technique:
   [(Sp + Rot-Blesser) − (NV + Rot-NV)]
   ↳ Speech with and without pitch (harmonic structure not perserved)

3. Any type of phonetic information:
   [(Sp + NV + Rot-Blesser) − (Rot-NV)]
   ↳ Speech with and without phonetic information

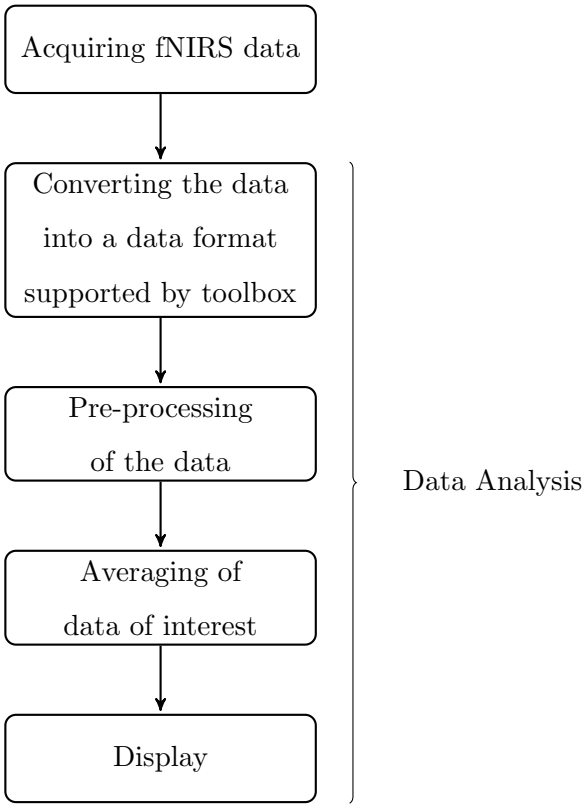
4. Comparing rotation techniques with different harmonic structure:
   [(Rot-TS) − (Rot-Blesser)]
   ↳ Rotated speech with and without preserved harmonic structure

5. Pitch perception new rotation technique of Kurt:
   [(Sp + Rot-TS) − (NV + Rot-NV)]
   ↳ Speech with and without pitch (harmonic structure preserved)

The first three contrasts were presented by Scott et al. (2000). The last two contrasts presented by Steinmetzger and Rosen. The contrasts four and five were used to assess the effect of two different rotation techniques, one with preserved harmonic structure (Kurt Steinmetzger, unpublished) and one without (Blesser, 1972). These contrasts are intended to reveal the specific differences in neuronal response of the regions of interest (ROI) and are applied in one of the final steps of the data analysis in the group-level glm analysis.

General fNIRS Experiment Workflow

The general workflow from data acquisition to the data plotting is structured in the following way:



The acquisition of the fNIRS data and the experimental design is one of the most important steps, since everything further depends on it. Okada et al. (2010), for example, shared that the reason why they found bilateral activity, unlike Scott et al. (2000), was due to the fact that Scott et al. (2000) did not have enough participants, i.e. data. Once the data is acquired, the fNIRS data analysis follows, which consists of several steps, such as converting the data to the data format supported by the toolbox used, pre-processing the data, averaging the data, and finally displaying the results.

This general data analysis workflow should be kept in mind, as the next Chapter *Data Analysis Method 3*, provides more detailed information and explanations about the steps of the data analysis.

Chapter 3

Data Analysis Method

This Chapter deals with the fNIRS data analysis method, i.e., the MNE Python package series used for this data analysis and the subsequent data analysis steps. At the beginning in Section, *MNE Data Analysis Series 3.1*, background and supplementary information on the Python toolbox series MNE is provided. In addition, reasons are given as to why MNE-Python was preferred over other methods. Next, Section *Data Analysis 3.2*, lists the specific data analysis steps of preprocessing and individual analysis that were performed on the dataset, followed by a list of the group-level GLM analysis steps required to be carried out moving forward.

3.1 MNE Data Analysis Series

The tool of the used data analysis was MNE python. Analysis software chosen for the implementation of the project is the MNE-Python is an open-source Python module for processing, analysis, and visualization of functional neuroimaging data, such as EEG, MEG, sEEG, ECoG, and fNIRS (Gramfort et al., 2013). To improve the fNIRS data analysis, MNE has provided an additional open-source toolbox in Python named MNE-NIRS, developed by Robert Luke, Erik Larson, and Alexandre Gramfort. The toolbox is handled by scripting the processing pipeline, giving users flexibility and convenience while programming in Python. However, MNE-NIRS shares common functionality with the MNE-Python toolbox, thus the documenta-

tion is split into the MNE-NIRS API Reference website for specific NIRS functionalities and for more general functions in the MNE-Python API reference.

For this work and the evaluation of MNE-Python for assessing neural processing by fNIRS, the various examples available on the MNE-Python[1] and MNE-NIRS [2] websites are heavily relied on. For Group-Level analysis the use of MNE-BIDS (Appelhoff et al., 2019) is proposed. Brain Imaging Data Structure (BIDS) is a standard describing how to organize neuroimaging and electrophysiological data, defining; file formats to use, file naming, placement of files withing a directory structure and also stores additional metadata. The advantage of using MNE-BIDS[3] is that it links BIDS and MNE-Python, this should make analysis faster to code and also supports fNIRS file formats.

One reason I chose the MNE series as the packages to do the data analysis as opposed to, for example, HOMER2 or HOMER3[4] which also provides a set of MATLAB scripts for analyzing fNIRS data to obtain estimates and maps of brain activation, is that I am more familiar and proficient in the Python programming language. Another argument for using MNE-Python was that my supervisors were very interested and motivated about using a tool for fNIRS analysis in Python, as it is relatively new. I also was very eager and fascinated, when I saw on the MNE webpage how many analysis possibilities there are.

3.2 Data Analysis

For the data analysis, which at the moment consists of preprocessing and individual analysis, I tried to follow the various instructions and tutorials that MNE provides

[1]MNE-Python toolbox webpage: https://mne.tools/stable/index.html and installation of MNE-Python package: https://pypi.org/project/mne/

[2]MNE-NIRS toolbox webpage: https://mne.tools/mne-nirs/stable/index.html and installation of MNE-NIRS toolbox: https://pypi.org/project/mne-nirs/

[3]MNE-BIDS webpage: https://mne.tools/mne-bids/stable/index.html and installation of MNE-BIDS: https://pypi.org/project/mne-bids/

[4]HOMER3 webpage: http://openfnirs.org/software/homer/

on its website. In the following list, called *First-Level Analysis*, are the specific analysis steps that were performed, with detail explanation. The next list, namely *Group-Level GLM Analysis* covers the steps that are still required to be done for an overall analysis.

Analysis Steps

First-Level Analysis (Preprocessing and Individual Analysis)

1. Conversion of the data into a data format supported by the MNE toolbox.
   1.1. Positional *pos.* file conversion
   1.2. Read and load recorded Hitachi *.csv* files
2. Plot Montage on the brain
3. Read Events, set annotation and duration of for each stimuli for Hitachi-Raw object
   3.1. Remove double triggering
4. Select channels appropriate for detecting neuronal response
   4.1. Remove motion (baseline shift & spike) artifacts by applying temporal derivative distribution repair (TDDR)
   4.2. Remove channels that are too close together, so called short channels
5. Evaluate quality of data using the scalp coupling index (SCI)
   5.1. Set for SCI less than 0.5 as bad channel
6. Convert raw intensity to optical density data and then to haemoglobin concentration using the modified Beer-Lambert law
7. Filtering
   7.1. Remove heart rate from signal

8. View consistency responses across trials
9. Plot standard fNIRS response
10. View how the topographic activity changes throughout the response
11. Plot comparison of conditions
12. Plot individual waveforms that drive to the topographic plot

The preprocessing and individual analysis starts with the conversion of the data into a data format supported by the MNE toolbox. The digitisation device has generated a positional *.pos* file for the specific localization of the channels, however MNE does not support *.pos* files yet. In order to set a montage for the channel positioning, one has to use generate a *DigMontage* instance. This is done by digging the specific data out of the *.pos* file and generating a *.csv* file containing all the relevant positional information, such as nasion fiducial point, right/left periauricular fiducial point, coordinates of the channel positions, and then use the method *mne.channels.make_dig_montage* in order to generate the *DigMontage* instance. The next step is to read and load the Hitachi raw intensity data and have a first look at the montage, by visualizing the montage in 2D or 3D.

When reading the Hitachi raw intensity data, a *Raw* instance gets generated, to this *Raw* instance, one can find events and set the according annotations, and duration for each stimuli. However, by using the *module men.find_events* 100 events are found, double as much than the initial 50 events, meaning that there is a double triggering not noticing the rest period, this has to be removed. Next, select specific channels, remove some channels or define certain channels as bad channels. The motion artifacts correction need then be applied to the optical density data, thereby convert the raw intensity to optical density and then apply the temporal derivative distribution repair on the data (TDDR). The TDDR was introduced by Fishburn et al. (2019) and is a fNIRS motion correction procedure based on robust regression, which effectively removes baseline shift and spike artifacts without the need for any

user-supplied parameters. Afterwards, the quality of the data can be evaluated using the Scalp Coupling Index (SCI). The Scalp Coupling Index determines the quality by measuring the connection between the optode and the scalp Pollonini et al. (2014). However, a threshold must be defined for the SCI, for instance 0.5, and then any channel that has less than 0.5 SCI is considered a bad channel, although it is important to ensure that the threshold is not too low to preserve the majority of the data. The acquired fNIRS signals from the optical density data may contain several types of noise, which can be classified as instrumental noise, experimental error, and physiological noise, such as heartbeat noise (1-1.5 Hz) or respiration (0.2 0.5 Hz) (Naseer and Hong, 2015).

There are several methods used to remove them, such as principle component analysis (PCA), independent component analysis (ICA) and band-pass filtering (Naseer and Hong, 2015). In this work we remove the heart rate using a band-pass filtering. In order to remove the heart rate from the signal using a filter is used with a lower pass-band edge at 0.05 Hz, upper pass-band edge at 0.7, transition band at the high cut-off frequency at 0.2 Hz and a transition band at the low cut-off at 0.02 Hz. Having done that, I can now convert the optical density data to haemoglobin concentration using the modified Beer-Lambert law. After that multiple plots can be made, one of which is the standard fNIRS response, comparison of conditions, topographic activity changes throughout the response, or give the individual waveforms that drive to the topographic plot. This was only a brief rough run through the data analysis that was made, it can however be better understood when viewing the code.

Up to this point, the code has been implemented, but a GLM (general linear model) analysis at the group level is still desired in order to assess the neural response over all participants. Therefore, I now present the steps that still need to be implemented.

Group-Level GLM Analysis

1. Create BIDS
2. Run Individual Analysis from BIDS on all participants
   - 2.1. Create Design matrix using a hemodynamic response function (HRF), either statistical parametric map (SPM) or glover and run the GLM
   - 2.2. Pick channel pairs manually and compute regions of interest (ROI)
   - 2.3. Define Contrast
   - 2.4. Visualize Results
3. Compute group level results
   - 3.1. Visualize group results
   - 3.2. Topographic visulaisation of each condition
   - 3.3. Contrast visualisation using topographic representation
   - 3.4. Cortical surface projections

The first step of a group-level GLM analysis is to create a BIDS (Brain Imaging Data Structure), since this is the easiest way to proceed with the group-level GLM analysis. It is the suggested format by MNE, as they use it in all their group-level tutorial and the package named MNE-BIDS is being applied. Furthermore, a design matrix can be created using the hemodynamic response function from SPM (statistical parametric map) and then the GLM can be performed for all subjects in the BIDS structure. Next, the aim is to pick the channels regarding the region of interest ROI, plot the responses according to the conditions and afterwards define and compute the contrasts and finally visualise the results.

Chapter 4

Results

This Chapter focuses on reporting the obtained results. In the previous Chapter *Data Analysis 3* in Section *Data Analysis Pipeline 3.2*, the sequence of steps for the data analysis, involving, preprocessing and individual analysis, namely the first-level analysis was outlined. The remaining future steps necessary for the group-level GLM analysis were discussed. Linking to the previous Chapter, the preliminary results of this first-level analysis are presented next, which includes preprocessing step and the individual analysis.

4.1 First-Level Analysis

Montage

There are several methods to view the montage using MNE Python. First, a few different variations of how the montage can be displayed are demonstrated. Whenever a 3D sample brain template is shown, it should be noted that it was provided by FreeSurfer, since MNE obtained the data from FreeSurfer and can be adapted and modified using FreeSurfer. FreeSurfer[1] is a neuroimaging toolkit for visualizing and analysing human brain MR images, however they also provide a lot of templates with region of interest (ROI) labels.

[1]FreeSurfer: `https://surfer.nmr.mgh.harvard.edu/`

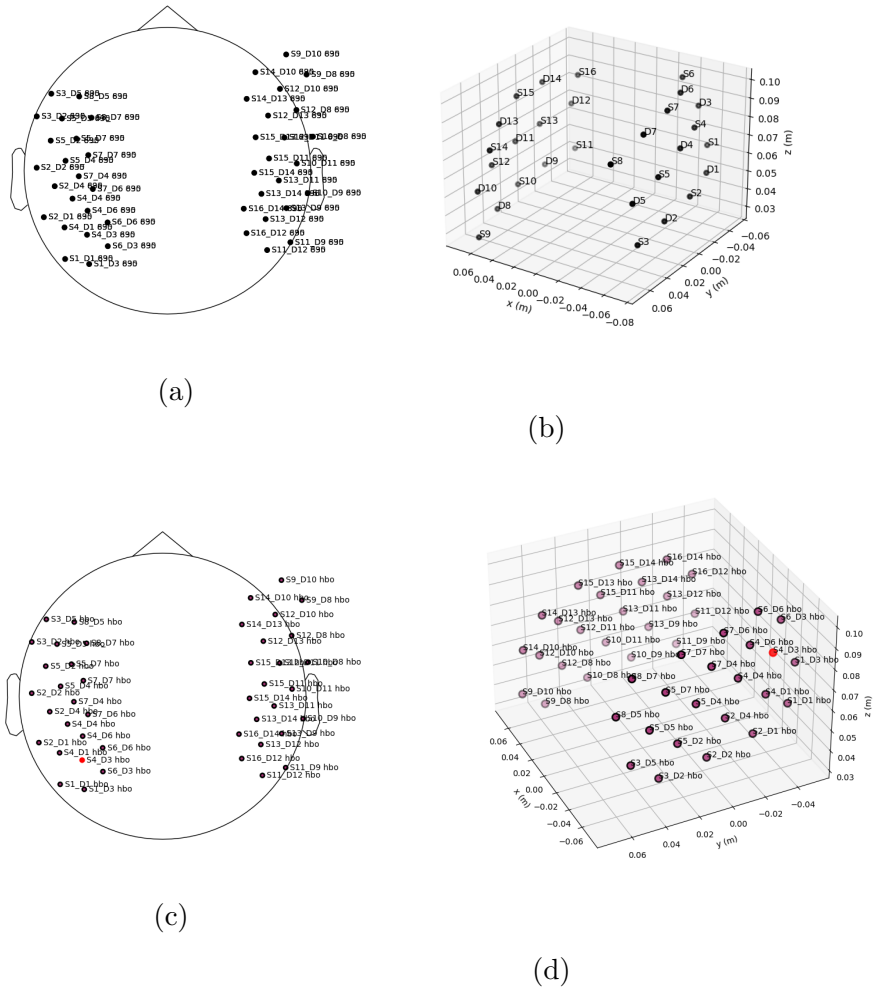


Figure 2: Tonotopical 2D (a) (b) and 3D map (c) (d) of montage *(subject S04).*

Figure 2 shows two 2D tonotpoic maps, (a) and (c) and two 3D coordinate diagram of the montage (b) and (d). The first illustrations (a) and (c) are plotted before defining or recognizing so called "bad channels", while (c) and (d) have a bad channel marked in red, namely *"S4_D3 hbo"*. It should be noted that the naming of the channels must follow a certain structure, *"S#_D# type"*, where # is replaced by the corresponding *source* and *detector* numbers and *type* is either hbo, hbr or the wavelengt. The sphere in (a) and (c), which represents the head, is automatically calculated by a fit to the digitization points, such as nasion or periauricular fiducial

points. However, the left hemisphere in the tonotopic representation (a) and (c) is clearly shifted, which is one reason why also a 3D representation is requested.

subject S04 subject S35

Figure 3: Right hemisphere with sensor-decoder pairs and channels of *subject S04* & *S35*.

In the montage illustration, Figure 3, the sensors are represented by source-detector pairs on the brain surface. The red dots are the source and the black dots symbolize the position of the detectors. The source-detector pairs are visualized by a white line whose yellow-colored center is the channel. The illustration shows the right hemisphere of the brain sample. Also, it can be seen that the montage is placed across the skull in such a way that it covers the auditory cortex, as it extends from parts of the occipital lobe through the temporal lobe to the frontal lobe. However, again the montage is not perfectly adjusted across all participants, compared to *subject S04* the montage of *subject S35* is rather skewed.

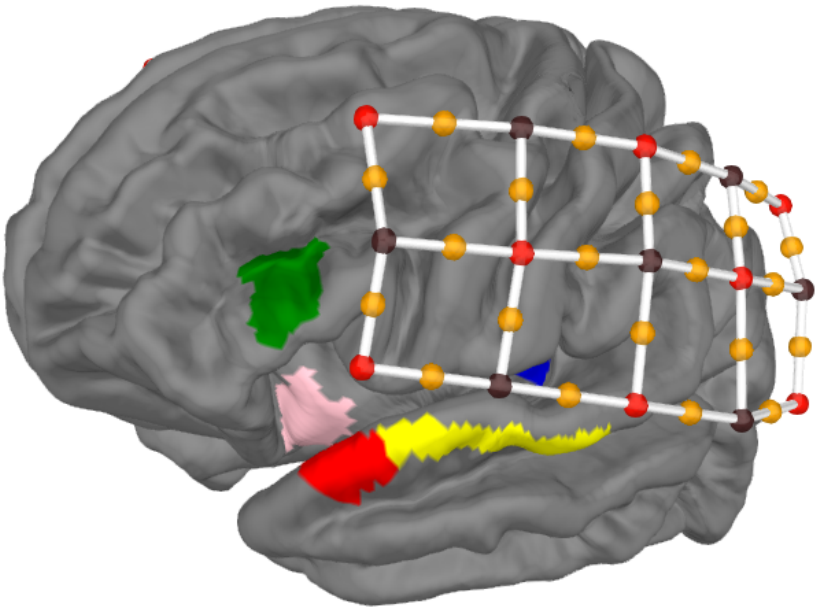


Figure 4: Sensor-decoder pairs and channels, with region of interest (ROI) of *subject S04*.

Figure 4 shows in addition to the previous Figure 3 specific regions of interests. The region colored green is around the broca's area, the area in the color blue represents the primary auditory cortex (A1), red marks the parts of the anterior superior temporal sulcus (aSTS) and the yellow are other regions of superior temporal sulcus (STS). All these brain regions are able to be colored using FreeSurfer. However, the labelling of those regions of interest is rather difficult as you need to know the abbreviations of the specific brain areas. For example, in FreeSurfer I have noticed that the left superior temporal sulcus is subdivided into multiple different labels; *"L_STSdp_ROI-lh"*, *"L_STSva_ROI-lh"* and *"L_STSvp_ROI-lh"*.

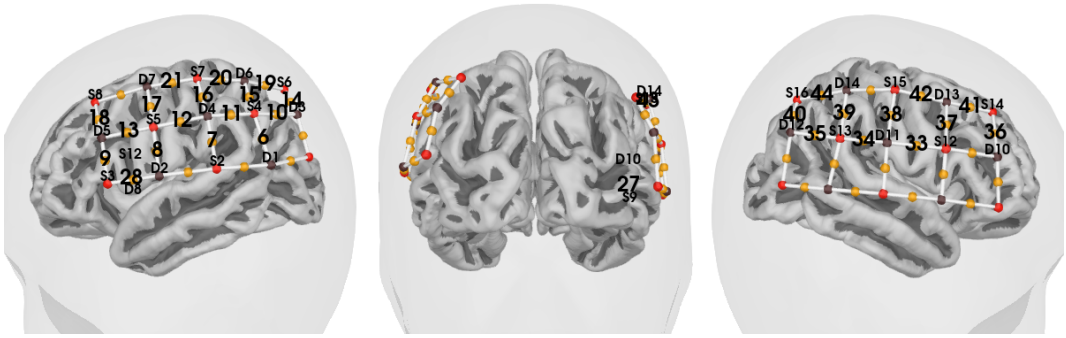


Figure 5: Left side, front and right side view of sensors, source-detector pairs with channel of *subject S04*.

Finally, Figure 5 shows the front and side view of the montage with selected channels and their naming. Given that the MNE method used to produce Figure 4 cannot display the channel names on top of the region of interest (ROI), this figure can demonstrate which channel is located in which brain area. Therefore this plot helps to find out the specific channel and their channel name in a certain region of interest.

Events

Events, or also called conditions, need to be set for the raw data. In our case we have 5 stimuli conditions, namely speech (SP), rotated speech (Rot-TS), rotated speech by Blesser (Rot-Blesser), rotated noise-vocoded speech (NV-Rot) and noise-vocoded speech (NV). The duration of each stimuli condition is set to 20s. The experimental design is conducted in such a way that each stimuli conditions has 10 trials, this makes a total of 50 events.

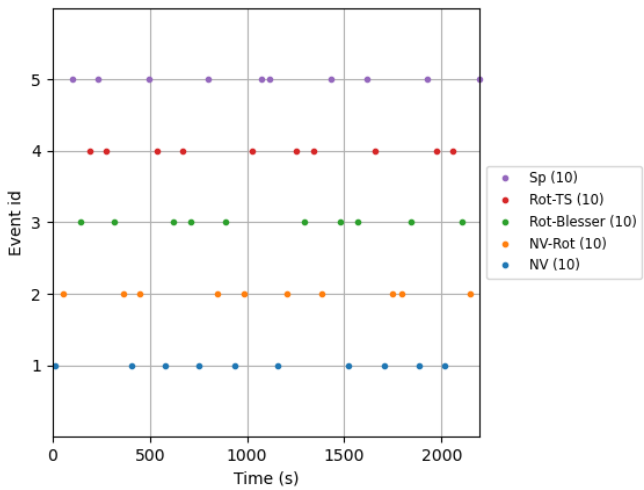


Figure 6: Event-time diagram showing all events for each condition of *subject S04*.

Figure 6 visualizes the event-time diagram, where each event (stimulus conditions) has 10 trials. Each event has its own event identifier and a different color, also the number of trials is indicated in the legend behind the condition labels.

Raw Data

MNE provides a way to interact with the plotted raw data, by for example adding annotations or selecting certain channels. In this way specific channels can be manually excluded from the later data processing.

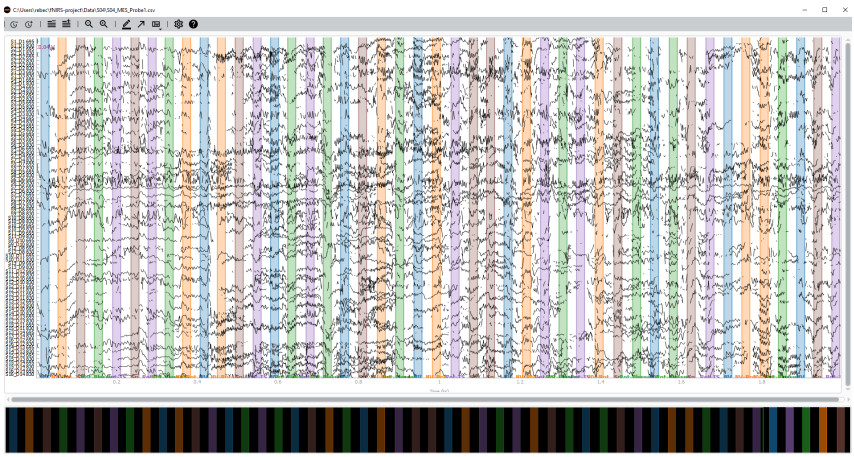


Figure 7: Raw intensity plot of *subject S04* without applied motion artifact correction (TDDR) including all channels.

Figure 7 shows the interactive raw intensity plot of MNE-Python. In the navigation bar at the very top of the plot, various adjustments can be made, such as adding annotations. In addition, all annotated events are displayed, each in a different color, and the far left lists the specific channel names from which each channel intensity originated.

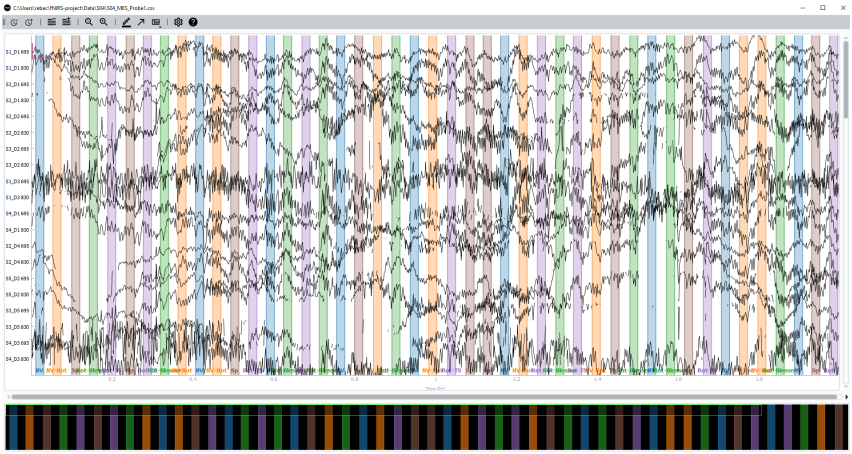


Figure 8: Optical density plot of *subject S04* with applied motion artifact correction (TDDR) of 15 channels.

Figure 8 shows the optical density plot where the time-derived distribution repair (TDDR) was previously applied to the data.

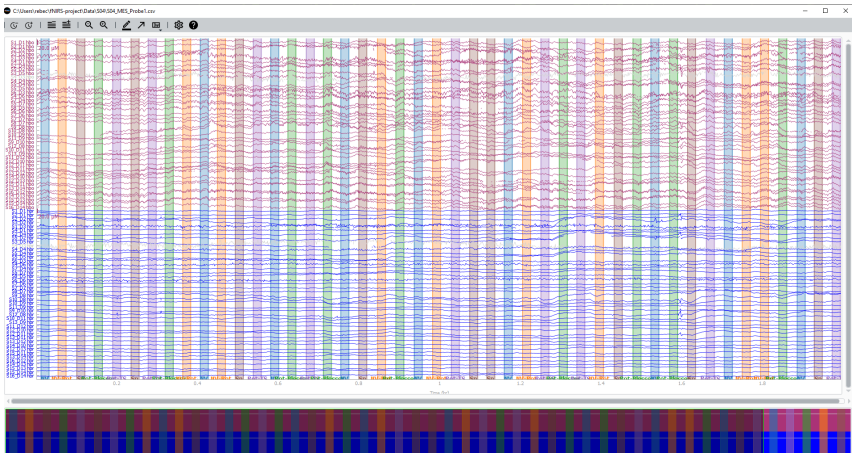


Figure 9: Haemoglobin concentrations of all channels of *subject S04* with applied motion artifact correction (TDDR).

The last raw plot in Figure 9, is the haemoglobin concentration plot. To obtain the hemoglobin concentration, the modified Beer-Lambert law must be applied to the optical density. Also, recall that a motion correction was prior performed using TDDR on optical density data, as it is recommended by Naseer and Hong (2015).

Data Quality Evaluation

In order to quantify the quality of the coupling between the scalp and the optodes, the scalp coupling index is used. The scalp coupling index is a quality check to evaluate the correlation between two wavelength channels in the cardiac band (Pollonini et al., 2014).

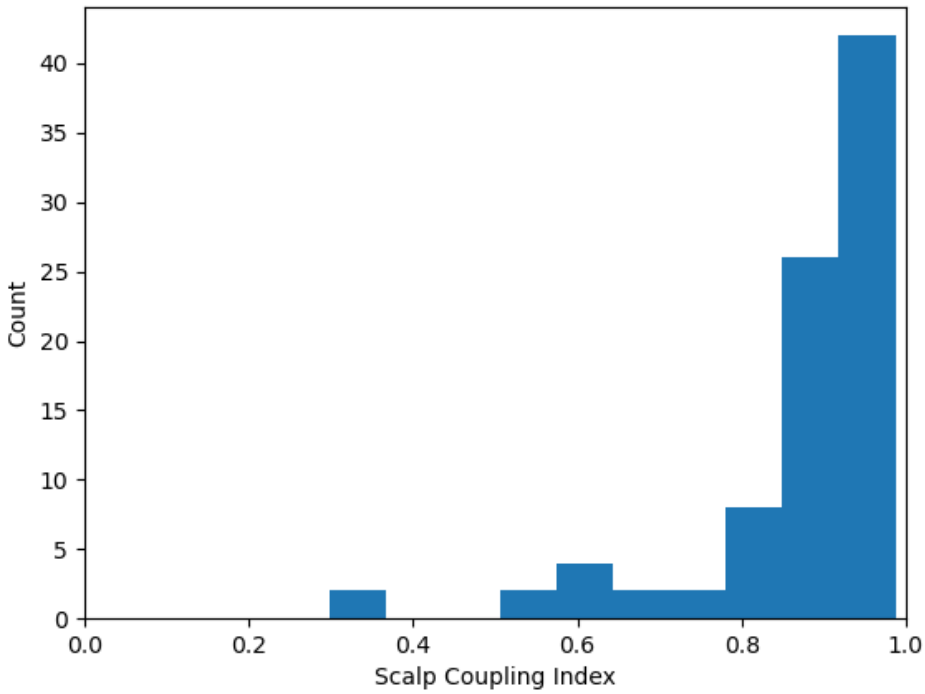


Figure 10: SCI plot of *subject S04*.

Figure 10 shows the SCI diagram. The index ranges from 0.0 to 1.0, and it can be seen that most counts are between SCI 0.8 and SCI 1.0. Depending on the threshold value set, certain channels that fall below it are removed. For example, if the threshold is SCI 0.5, only a few channels will be removed, but the larger the threshold, the more channels will be removed from the data, so one should be careful not to choose a threshold that is too high.

Filtering

Before converting optical density data to hemoglobin concentration with the modified Beer-Lambert law, apply a band-pass filter to filter out heart rate, which is recommended by Naseer and Hong (2015). In MNE-Python such a filter can be applied to the data. In order to remove the heart rate from the signal 4 parameter are used, lower pass-band edge, upper pass-band edge, transition band at the high cut-off frequency and transition band at the low cut-off. The parameters were chosen to match those in the MNE tutorial example, i.e., the lower pass-band cutoff was set to 0.05 Hz, the upper pass-band cutoff to 0.7 Hz, the transition band at the high cutoff frequency to 0.2 Hz, and the transition band at the low cutoff frequency to 0.02 Hz.

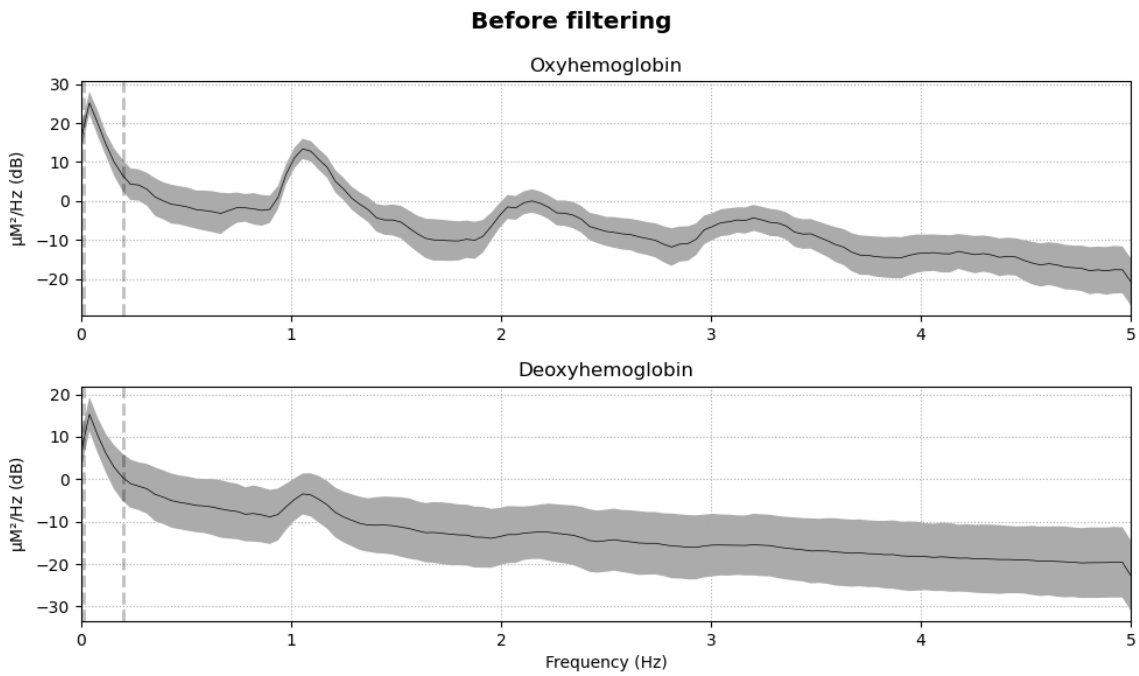


Figure 11: Haemodynamic responses and corresponding frequency before filtering of *subject S04*.

The hemodynamic response has a frequency component, displayed in Figure 11 and Figure 12. There is a 1 Hz increase in activity in the data which can be attributed to a person's heartbeat which is undesirable. Hence, the low-pass filter

is applied to remove this and the high pass filter is included in order to remove slow drifts in the data.

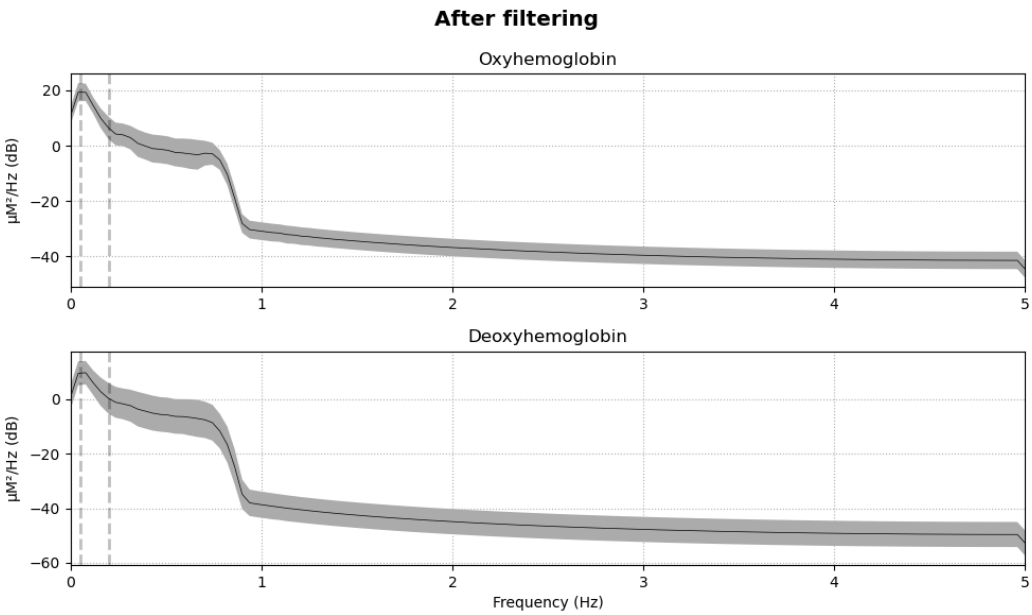


Figure 12: Haemodynamic responses and corresponding frequency after filtering of *subject S04*.

After the heartbeat is removed by the filter the haemodynamic responses and their corresponding frequency should look like the plot in Figure 12.

fNIRS Response

MNE-Python provides a very interactive way to display the fNIRS responses. The next illustrated diagrams show the graphs generated by MNE and the features they provide to the user.

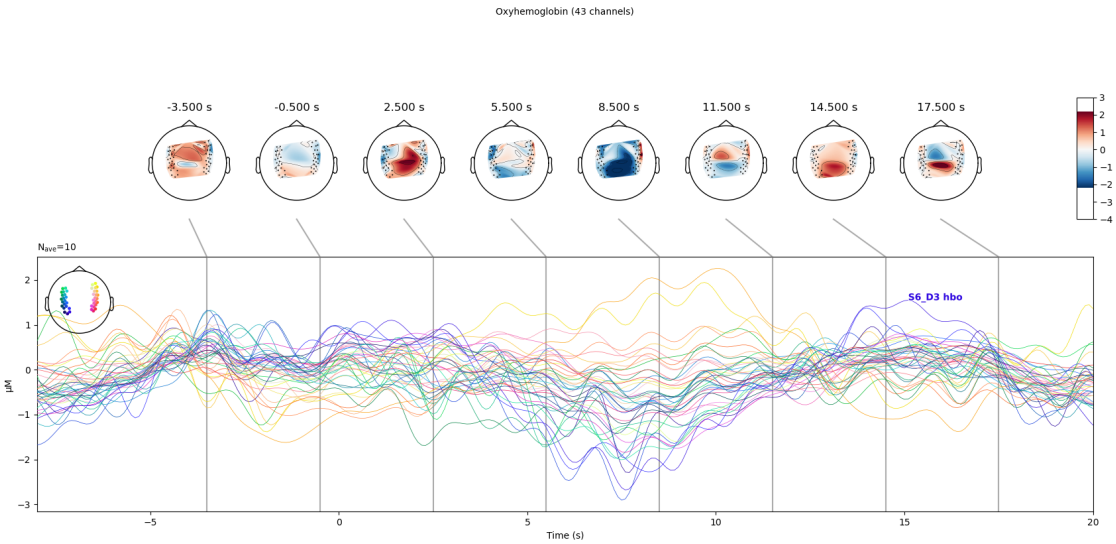


Figure 13: Average oxyhemoglobin concentration across time diagram with 44 channels of *subject S04*.

Figure 13 contains a lot of information. First, it shows the oxyhemoglobin concentration of 44 channels over the entire session. Second, the channels are color-coded with the left hemisphere having cooler colors and the right hemisphere having warmer colors assigned to the channels and within the hemispheres the channels are again differently colored. Third, you can see tonotopic diagrams above with the oxyhemoglobin responses for specific time periods. Finally, this plot is very interactive: when you click on one of the colored responses, the name of the channel is displayed. For example, in Figure 13, the blue channel *'S6_D3 hbo'* has been marked, which is located in the left posterior hemisphere and is shown on the far left of the tonotopic map. This channel has a positive oxyhemoglobin peak at around 15s and this a very

negative oxyhemoglobin response around 7s. This high negativity of the average oxyhemoglobin concentration at 7s represents the deoxyhemoglobin, which is also visualized with a blue color in a tonotopic plot on the very top of the diagram. That is why in the next illustration, Figure 14 the responses for the each conditions is plotted to further investigate this channel. Another peak of oxyhemoglobin is shown at around 10s, from a channel on the right anterior hemisphere this is further illustrated in Figure 15 and Figure 16.

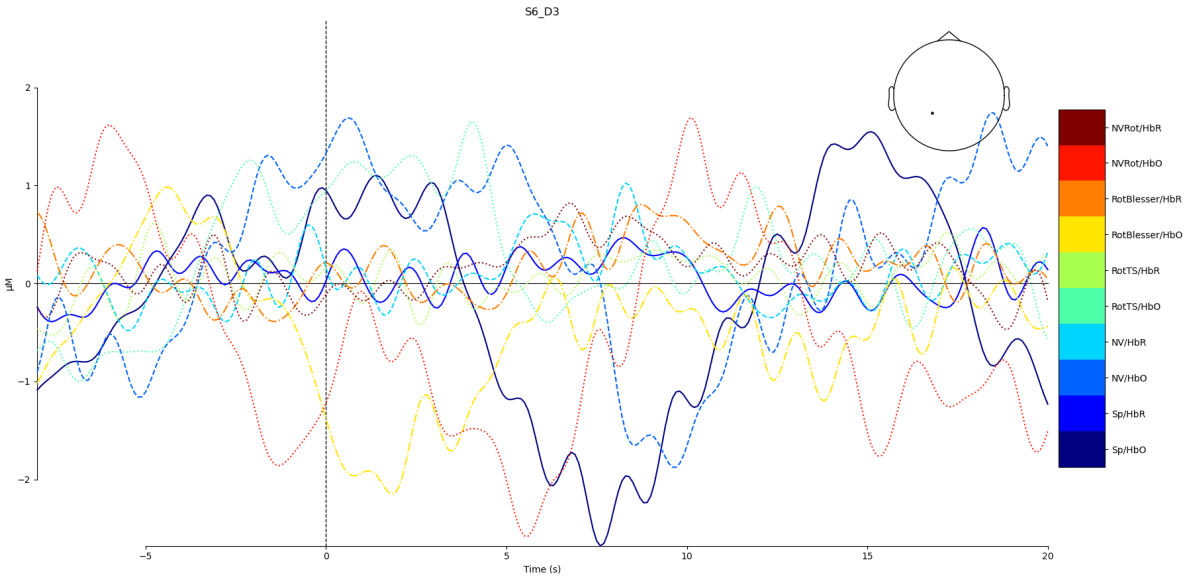


Figure 14: Standard averaged fNIRS response for all conditions for channel *S6_D3* of *subject S04*.

Figure 14 plots the standard fNIRS response, averaged oxyhemoglobin (HbO) and deoxyhemoglobin (HbR) for each of the 5 events, also called conditions, over the time axes. At the top right of the graph is a tonotopic plot showing the location of the *'S6_D3'* channel. Each condition and its individual Hbo or HbR response are shown in different colors. Remarkably, the oxyhemoglobin responses (Hbo) of two intelligible speech responses (SP) the dark blue line and noise-vocoded speech (NV) the dotted blue line are synchronous with a small shift.

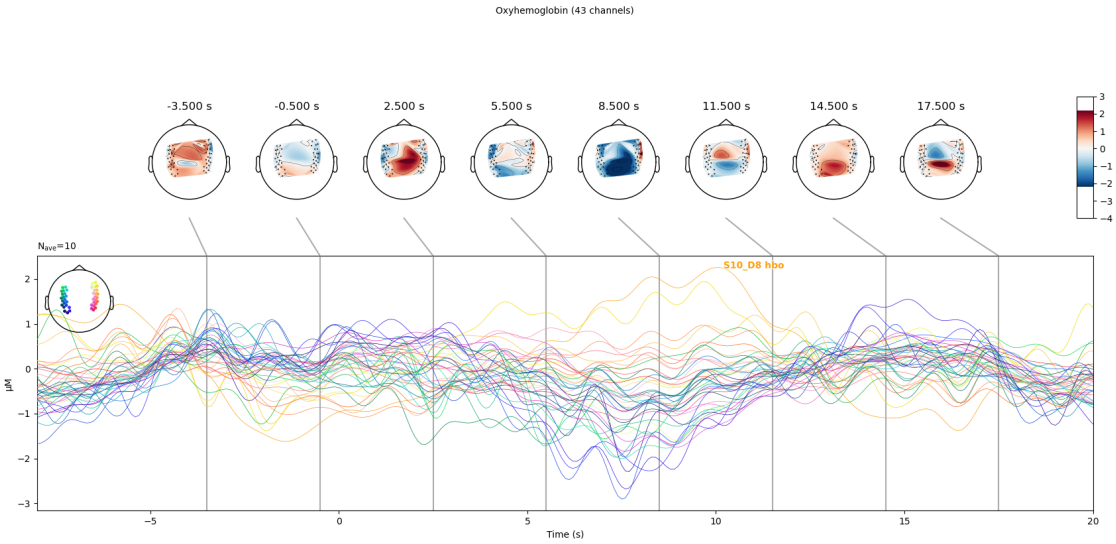


Figure 15: Oxyhemoglobin concentration across the session with 44 channels of *subject S04*.

Another channel, colored orange, exhibits a peak in oxyhemoglobin concentration at approximately 10s with a negative oxyhemoglobin response at around -1s, as shown in Figure 15. The channel is located in the anterior right hemisphere and is called *'S10_D8'*. Figure 16 examines the channel by showing the individual conditions with their responses to oxyhemoglobin (HbO) and deoxyhemoglobin (HbR). Interestingly, oxyhemoglobin concentration of speech (Sp/Hbo) and that of rotated noise-vocoded speech (NVRot/Hbo) peak at approximately 10s.

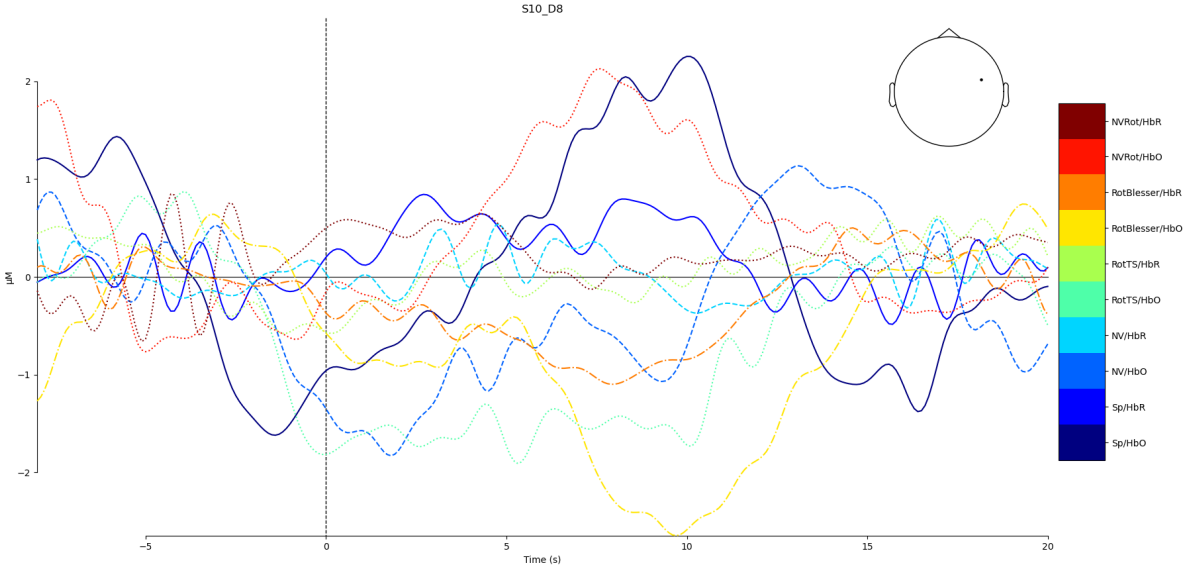


Figure 16: Standard averaged fNIRS response for all conditions from channel *S6_D3* of *subject S04*.

In Figure 16, the channel *'S10_D8'* located in the anterior right hemisphere is illustrated. In additon to the peak of oxyhemoglobin concentration of the condition speech (Sp/Hbo) and that of rotated noise-vocoded speech (NV/Rot/Hbo) peak at 10s, a sharp decrease in oxyhemoglobin concentration is seen in (Rot/Blesser/Hbo), colored yellow. Moreover, it shows that the average deoxyhemoglobin concentration of noise-vocoded speech (NV/Hbr), rotated speech TS (RotTS/Hbr), rotated noise-vocoded speech (NVRot/Hbr) and speech (Sp/Hbr) is around $0\mu M$ with some deviations.

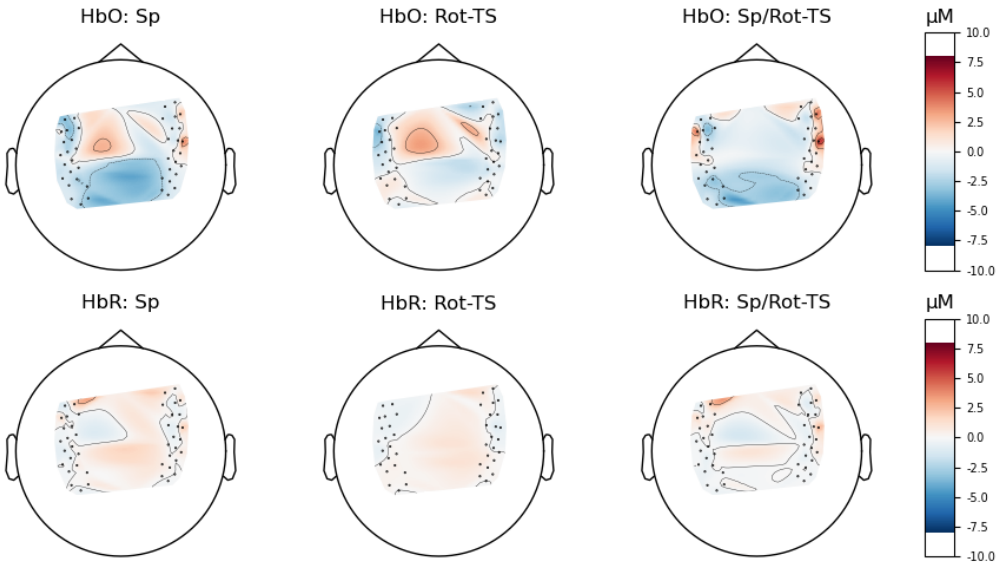


Figure 17: Tonotopic plot of the averaged fNIRS response with a condition comparison of speech (Sp) and rotated speech TS (Rot-TS) of *subject S04*.

Another possibility is to compare average fNIRS responses of a certain conditions in tonotopic plot as shown in Figure 17. This option provides the chance to view the location of activity.

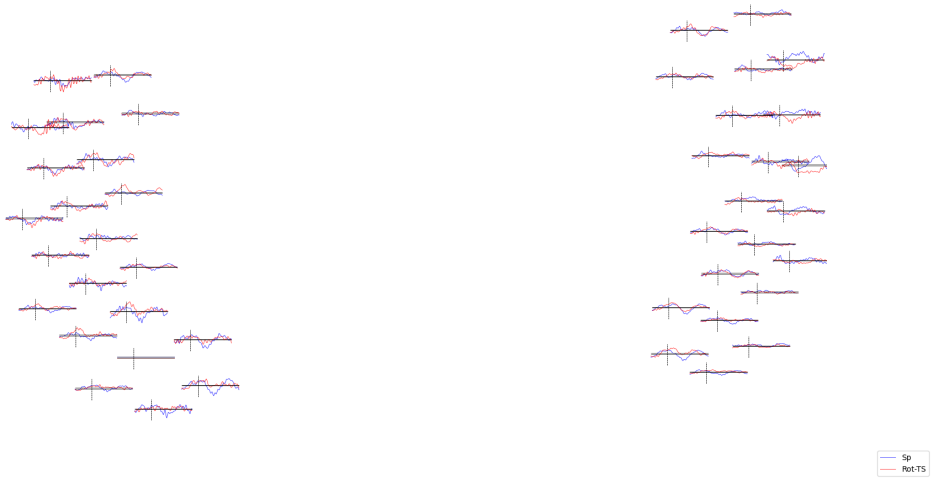
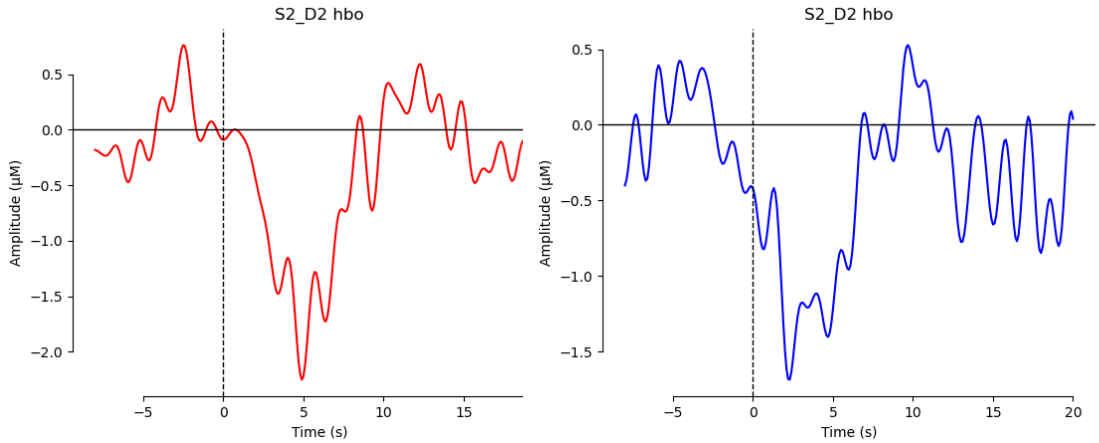


Figure 18: Individual waveforms that drive to the topographic plot 17 with blue response waves representing speech (Sp) and red response waves representing the condition rotated speech TS (Rot-TS) of *subject S04*.

The next diagram, Figure 18, is very interactive and offers the user the option to choose certain channels and display the plots with condition specific average oxyhemoglobin concentration response across time, shown in Figure 19. Here there are two individual waveforms, accounting for two conditions, speech (Sp) colored blue and rotated speech TS (Rot-TS) colored red. These individual waveforms drive the tonotopic diagram of Figure 17. Moreover, as in Figure 2, where a channel was marked as red, which means that it is a "bad channel", it is also here demonstrated that there is no data in one of the channel in the left posterior hemisphere.



(Rot-TS) (Sp)

Figure 19: Averaged oxyhemoglobin response for rotated speech TS (Rot-TS) and speech (Sp) of *subject S04*.

When clicking on one specific channel the interactive plot of Figure 18, the individual average oxyhemoglobin response for the specific condition get displayed. In this example, Figure 19, the channel 'S2_D2' was chosen and the two example conditions are in this case, Sp colored blue and Rot-TS colored red.

Chapter 5

Discussion and Conclusion

In the previous Chapter *Results* 4 it was possible to present the first-level analysis findings using the toolbox MNE-Python. Subsequently, all of the generated diagrams and figures look fine. An advantage of MNE is the high interactivity of the diagrams. For example, diagrams of Figure 18, 15 or all diagrams of the raw data, Figure 7, 8 and Figure 9, all have an interactive attribute embedded. Moreover, they follow a great visual design, with good coloring. Another strength of the MNE toolbox is that it is handled by scripting the processing pipeline which makes the programming easy and keeps the code relatively short. When evaluating the application of the open-source Python toolbox MNE for assessing neural processing of degraded speech from functional near-infrared spectroscopy, not only the usability of the code and the generated plots need to be considered, but also the MNE documentation and overall support. The MNE community made multiple tutorials with sample code and sample data available, which made the learning experience easier, especially for people who never have conducted brain data analysis. Also, their source code which is available on GitHub, is well documented and also their API references on their website follow a great documentation of code and calculations which provides a better understanding for the users. With each MNE version new features are added or old ones are changed, therefore a good documentation is essential. They for example, add and highlight their notes about additional features in visible green boxes. Moreover, the MNE community provides great online support in an online

forum and with bi-weekly live support sessions where users can share their issues with MNE developers whenever they are having problems using MNE, have questions, have noticed a bug, or have a feature request.

Let me now move on to discuss my experience with MNE-Python, the implementation phase was accompanied by multiple challenges. One of which started with the file conversion and the creation of the *DigMontage* object from the initial positional *.pos* file. Since MNE python does not support the *.pos* file generated by the digitisation system "Polhemus Patriot" to read the channels and their positions, it was necessary to convert the positional information to a for MNE readable way by a creation of a *DigMontage* object. At first I only used the x,y,z coordinates but this was not sufficient as nasion fiducial and periauricular fiducial points were also required for a good montage. Later another challenge arose, when trying to plot the sensors on a brain template, as the unite of measure was too high, when converting the *.pos* the mass unit was meters and not millimeters. Consequently, the sensors were plotted outside the brain template. After visualizing the channels and sensor-decoder pairs on a brain template, I observed that the sensor-decoder pairs in the right hemisphere were all incorrectly connected. Leading to the conclusion that the channel naming must have been wrong. In MNE, when creating an *DigMontage* object, one must convert all previous channel names to their channel naming format. The MNE channel naming needs to be converted in this structure, *"S#_D# type"*, where # is replaced by the corresponding *source* and *detector* numbers and *type* is either hbo, hbr or the wavelength. If the corresponding numbering of source and detector are wrong, the source-channel pairs will be connected incorrectly, as it happened in my case.

Another difficulty, was the reading in of the fNIRS Hitachi raw data. The Hitachi raw data was separated into two probes, where each probe represented the recording of one hemisphere. The method to read the raw data, namely *"mne.io.read_raw_hitachi"*, for the first-level analysis was problematic, since there was only the option to read in one file and no further concatenation option further downstream processing pipeline of MNE. Hence, I attended a live question session with MNE developers and received

assistance. After roughly two to three weeks the method was modified to pass in a list for multiple probes that are acquired simultaneously. This demonstrates their commitment and desire to further develop the MNE Python toolbox. Regarding the by MNE-Python generated montage diagrams even though it is able to zoom in to the montage plot in Figure 4 and in Figure 3, it does not support selecting specific channels by clicks to get their names, which is a drawback when trying to specify the region of interest (ROI) and pick certain channels. One more drawback is the information in the tracebacks of MNE-Python. I had to exclude two participants because the script did not work for them, while there were no problems with the other participants. The traceback information was too unclear for me to investigate the issue, thus I had to exclude participants S15 and S6 from further analysis. Another challenge arose when trying to start the group-level GLM analysis. Following the MNE-Python toolbox tutorial provided on the website it is suggested to convert the data to a BIDS structure using MNE-BIDS. However, there is currently a bug when trying to create a BIDS data structure using MNE-BIDS. For this reason, this work is not yet complete, as the main goal is to assess the neural processing of degraded speech as a whole, and that includes a GLM analysis at the group-level.

In conclusion, reading and converting the data was the most time consuming aspect of this project. If, however, the recorded data had been from another fNIRS system or/and the position file had been supported by an MNE Python method, such as, *"Polhemus isotrak"*, *"Polhemus fastscan"*, or another system, the analysis would have been much faster as promised by MNE-Python. Looking back, I have learned the importance of the experimental design, as the format in which the data is stored and whether it can be converted to the toolbox can either facilitate or complicate a toolbox. I am convinced that MNE is a very good analysis toobox choice if someone who is not very familiar with brain data analysis wants to analyze some data with fNIRS. The tutorials are great as they are very helpful in the learning process and with the great support of the MNE team almost any solution can be found. Therefore in my opinion the assessment of the neural processing of degraded

speech recorded by fNIRS is feasible when applying the open-source Python toolbox MNE for its overall data analysis. As Python becomes more popular, data analysis with Python also becomes more mainstream, which in turn leads to an increasing interest in toolboxes and Python packages. All in all, many such toolboxes or packages are not yet fully developed and mature, including MNE-Python, but the MNE community is very involved in fixing issues and improving the toolbox.

Chapter 6

Outlook

Looking ahead to the success of MNE-Python, the project, and the improvement of this evaluation, the use of a different toolbox, such as HOMER3, applied to the recorded fNIRS data should be taken into consideration, as this will allow for direct comparison and a more meaningful evaluation. As for the analysis of the fNIRS data for the assessment neural processing of the degraded speech, the first-level analysis is complete and the group-level GLM analysis will work once MNE fixes the bug in creating a BIDS with MNE-BIDS. Therefore, it is concluded that the results of the assessment of the neural processing of the degraded speech can be presented in the near future.

List of Figures

Appendices

Appendix A

Glossary

The following table provides an overview of all acronyms and terms.

AI	Articulatioon iindex
BIDS	Brain Imaging Data Structure
fNIRS	fuctional Near-Infrared Spectroscopy
EEG	Electroencephalogram
ECoG	Electrocorticography
MEG	Magnetoencephalography
PET	Positronen Emissions Tomographie
sEEG	Stereoelectroencephalography
SCI	Scalp Coupling Index
STG	Superior Temporal Gyrus
STC	Superior Temporal Cortex
STS	Superior Temporal Sulcus
SPM	statistical parametric map
Sp	Speech
RSp	rotated Speech
RVCo	rotated Noise-Vocoded Speech
ROI	region of interest
VCo	noise-vocoded speech
V1	primary visual cortex
MT	middle temporal area
HRF	haemodynamic response function
MR	Magnetic Resonance
MRI	Magnetic Resonance Imaging
GLM	General Linear Model
HG	Heschl's gyrus
MTL	Medial Temporal Lobe
ventro-lateral	"ventral"; towards the stomach, "lateral"; on one side, on one hemisphere
bilateral	affecting both sides of the hemispheres

Appendix B

Code

The code, the plots and the data used for this thesis are accessible on GitHub via the link `https://github.com/rfahrn/fNIRS-project.git`. However the GitHub repository also includes Python scripts which have the purpose of testing and debugging code, the three most important scripts however applied for the first-level analysis in this project are the following:

- **pos_convert.py**
  Converts the positional .pos file with its channel names and channel coordinates to a *.csv* file.

- **manuel_montage.py**
  Similar to *pos_convert.py*, this script extracts information from the position file to create an even better montage, as it also takes into account different digitization points, such as the left and right ear positions and creates a *DigMontage* instance.

- **Preprocessing_individual.py**
  This script is used for the first-level analysis and generates all the diagrams described in the results.

```
# Author: Rebecka Fahrni
# Script to convert .pos file into csv file

import csv
import pandas as pd
import os


def create_csv(read_filename, outfile):
    """
    Creates a .csv file out of .pos data, output csv:  4 columns
    containing ch_name, x, y, and z.
    :param read_filename: input file path that gets read
    :param outfile: path to which the output gets written
    :return: None (just creates an output-file)
    """

    with open(read_filename, 'r') as file:
        with open(outfile, 'w') as out_f:
            writer = csv.writer(out_f)
            lines = file.readlines()
            lines = [line.replace('\n', '') for line in lines]
            header = []
            indi = []
            for l in lines:
                if '[' and ']' in l:
                    header.append(l)
            for i in header:
                ind = lines.index(i)
                indi.append(ind)
            in_val = list(zip(header, indi))
            in_val = in_val[10:40]
            pre = in_val[0:10]
            h = ['ch_name', 'x', 'y', 'z']
            writer.writerow(h)
            for i in range(len(in_val)):
```

```
                if i != len(in_val)-1:
                    tup = in_val[i]
                    tup2 = in_val[i+1]
                    cont = lines[tup[1]:tup2[1]]
                    ch_n = cont[0]
                    ch_n = ch_n.lstrip('[').rstrip(']')
                    content = cont[1::]
                    x = content[0]
                    x = float(x.lstrip('X=')) / 1000
                    y = content[1]
                    y = float(y.lstrip('Y=')) / 1000
                    z = content[2]
                    z = float(z.lstrip('Z=')) / 1000
                    list_content = [str(ch_n), str(x), str(y), str(z)]
                    writer.writerow(list_content)
                else:
                    tup = in_val[i]
                    cont = lines[tup[1]::]
                    content = cont[1::]
                    ch_n = cont[0]
                    ch_n = ch_n.lstrip('[').rstrip(']')
                    x = content[0]
                    x = float(x.lstrip('X=')) / 1000
                    y = content[1]
                    y = float(y.lstrip('Y=')) / 1000
                    z = content[2]
                    z = float(z.lstrip('Z=')) / 1000
                    list_content = [str(ch_n), str(x), str(y), str(z)]
                    writer.writerow(list_content)

    # read .csv of positional .csv file
    df = pd.read_csv(outfile)
    df.to_csv(outfile, index=False, sep=',')  # restructure
    return


def create_0001_edit(number):
```

```
    """
    The function creates a pos to csv converted file called 0001_edit.
    csv in all subfolders
    :param number: int of a file number
    :returns None
    """
    os.makedirs('C:/Users/rebec/fNIRS-project/Data', exist_ok=True)

    # write .pos to .csv
    file_path_pos = 'Data/S' + str(number) + '/0001.pos'
    file_path_csv = 'Data/S' + str(number) + '/0001.csv'

    create_csv(file_path_pos, file_path_csv)

    # read .csv file with columns: ch_name,x,y,z and create a dataframe
    df = pd.read_csv(file_path_csv)

    # segmentation of Probe1 and Probe2 to new .csv files
    probe1 = df[df['ch_name'].str.startswith('Probe1')]  # left
    probe2 = df[df['ch_name'].str.startswith('Probe2')]

    # rename
    # left hemisphere
    new_names_1 = 'S1 D1 S2 D2 S3 D3 S4 D4 S5 D5 S6 D6 S7 D7 S8'.split
    ()

    # right hemisphere

    new_names_2 = ['S9', 'D8', 'S10', 'D9', 'S11', 'D10', 'S12', 'D11',
     'S13', 'D12', 'S14', 'D13', 'S15', 'D14', 'S16']

    # new_names_2 = 'S11 D10 S10 D9 S9 D13 S13 D12 S12 D11 S16 D15 S15
    D14 S14'.split()
    new_names = new_names_1 + new_names_2
    df.insert(0, 'ch_name', new_names)
    probe1.drop('ch_name', axis=1, inplace=True)
    probe2.drop('ch_name', axis=1, inplace=True)
```

```
    probe1.insert(0, 'ch_name', new_names_1)
    probe2.insert(0, 'ch_name', new_names_2)

    file_path1 = 'Data/S' + str(number) + '/probe1_channel_montage.csv'
    file_path2 = 'Data/S' + str(number) + '/probe2_channel_montage.csv'
    probe1.to_csv(file_path1, index=False, sep=',')
    probe2.to_csv(file_path2, index=False, sep=',')

    df.to_csv('Data/S' + str(number) + '/0001_edit.csv', index=False,
   sep=',')
    return


# list of number of participants ( all existing subfolders ) in Data
list_folders = ['01', '04', '05', '06', '07', '08', '09', 11, 12, 15,
    16, 17, 18, 30, 31, 32, 33, 34, 35, 36, 37]

for n in list_folders:
    create_0001_edit(n)
```

pos_convert.py

```

import pandas as pd
import numpy as np

def convert_to_dic(csv_file):
    """ reuse csv file we got from pos_convert to get channel names and
     pos"""
    with open(csv_file) as f:
        df = pd.read_csv(f, header=None, index_col=0).iloc[1:, :]
        df1 = df.agg(list, axis=1).to_dict()
        dic = {}
        for ch_name, ch_pos in df1.items():
            ch_pos = np.asarray([eval(i) for i in ch_pos]) # ch_name
   with array of shape (3,)
            dic[ch_name] = ch_pos
        return dic
```

```
# print(convert_to_dic('Data/S01/0001.csv'))

def read_montage(filename):
    """
    :param filename: file of montage coordinates from .pos
    we need following parameters: ch_pos, nasion, lpa, rpa, hsp (none), hpi (none), coord_frame = MRI 'mri'
    :returns right format for function make_dig_montage arrays and matrix
    """
    with open(filename, 'r') as file:
        lines = file.readlines()
        lines = [line.replace('\n', '') for line in lines]
        header = []
        indi = []
        for l in lines:
            if '[' and ']' in l:
                header.append(l)
        for i in header:
            ind = lines.index(i)
            indi.append(ind)
        in_val = list(zip(header, indi))
        info_index = in_val[5:11]  # list of tuples of header(info) and index
        dic = {}
        for i in range(len(info_index)):

            if i != len(info_index) - 1:
                tup = info_index[i]
                tup2 = info_index[i + 1]
                name = lines[tup[1]:tup2[1]][0]
                content = np.array(lines[tup[1]:tup2[1]][1::])
                name = name.lstrip('[').rstrip(']')

                if name == 'LeftEar':
                    name = 'lpa'
```

```
                elif name == 'RightEar':
                    name = 'rpa'
                elif name == 'Nasion':
                    name = 'nasion'

                x = content[0]
                x = float(x.lstrip('X=')) /1000
                y = content[1]
                y = float(y.lstrip('Y=')) /1000
                z = content[2]
                z = float(z.lstrip('Z=')) /1000
                list_content = [name, [float(x), float(y), float(z)]]
                dic[list_content[0]] = list_content[1]


        hsp = np.matrix((dic['Back'], dic['Top']))
        back = np.array(dic['Back'])
        top = np.array(dic['Top'])
        lpa = np.array(dic['lpa'])
        rpa = np.array(dic['rpa'])
        nasion = np.array(dic['nasion'])
        coord_frame = 'mri' # or 'unknown'

        return lpa, rpa, nasion, hsp, coord_frame
```

manuel_montage.py

```

# Author: Rebecka Fahrni, 18-735-522
# Project: fNIRS data analysis
# using MNE-python https://mne.tools/stable/index.html
# ------------------------------------------------------------
import sys
sys.path.append('C:/Users/rebec/fNIRS-project/manuel_montage.py')
import manuel_montage
import numpy as np
import matplotlib.pyplot as plt
from itertools import compress
```

```
import mne

# Preprocessing & Individual analysis - First-Level analysis
# ------------------------------------------------------------


# read Hitachi-files
def read_hitachi(hitachi_path):
    """
    Read Hitachi-csv file
    :param hitachi_path csv file list of strings
    :returns raw: instance of RawHitachi - raw object containing
    Hitachi data
    """
    raw = mne.io.read_raw_hitachi(hitachi_path)
    return raw


# Dig montage
def self_montage(file_path, csv_file):
    """
    :param file_path: .pos file with location of montage
    :param csv_file: enter csv file with ch_names and position
    :return: montage
    """
    lpa, rpa, nasion, hsp, coord_frame = manuel_montage.read_montage(
    file_path)
    ch_pos = manuel_montage.convert_to_dic(csv_file)
    return mne.channels.make_dig_montage(ch_pos=ch_pos, nasion=nasion,
    lpa=lpa, rpa=rpa,
                                         hsp=hsp, hpi=None, coord_frame
    ='mri')

# clean events - double triggers
def clean_events(events_numpy):
    """
    removes double triggering after found events
```

```
    :param events_numpy: events array that should be cleaned
    :return cleaned event array"""
    a = events_numpy
    ids = [0,]
    consecutives = 0
    for i in range(1, len(a)):
        if a[i,-1]!=a[i-1,-1]:
            consecutives=0
        else:
            consecutives+=1
        if consecutives%2==0:
            ids.append(i)
    return a[ids]

def get_plots(list_part):
    """creates firs-level analysis plots
    :param list_part: list with all participants
    """
    for participant in list_part:
        file_path = "C:/Users/rebec/fNIRS-project/Data/S" + str(participant) +'/S' + str(participant) +'_raw.fif'
        pos_file_path_1 = 'C:/Users/rebec/fNIRS-project/Data/S' + str(participant) +'/S' + str(participant) + '_MES_Probe1.csv'
        pos_file_path_2 = 'C:/Users/rebec/fNIRS-project/Data/S' + str(participant) +'/S' + str(participant) + '_MES_Probe2.csv'
        montage_path_csv = 'C:/Users/rebec/fNIRS-project/Data/S' + str(participant) + '/0001_edit.csv'
        montage_path_pos = 'C:/Users/rebec/fNIRS-project/Data/S' + str(participant) +'/0001.pos'
        montage_save_3d = 'C:/Users/rebec/fNIRS-project/Data/S' + str(participant) +'/montage_3d.png'
        montage_save = 'C:/Users/rebec/fNIRS-project/Data/S' + str(participant) +'/montage_top.png'
        save_plots =  'C:/Users/rebec/fNIRS-project/Plots/s_' + str(participant)
        raw = read_hitachi([pos_file_path_1, pos_file_path_2])
        montage = self_montage(file_path=montage_path_pos, csv_file=
```

```
montage_path_csv)
    raw.set_montage(montage)

    # load Data
    raw.load_data()

    # read Events from Hitachi-Raw and create an annotation
    events = mne.find_events(raw)
    events = clean_events(events)

    event_dict = {'Sp': 1,
                  'Rot-TS': 2,
                  'Rot-Blesser': 3,
                  'NV': 4,
                  'NV-Rot': 5}

    event_desc = {v: k for k, v in event_dict.items()}

    # write and set annotations
    annotation = mne.annotations_from_events(events=events, sfreq=
raw.info['sfreq'],
                                             event_desc=event_desc,
                                             orig_time=raw.info['
meas_date'])
    raw.set_annotations(annotation)
    # set stimuus duration 20s - If orig_time is None, the
annotations are synced
    # to the start of the data (0 seconds).
    raw.annotations.set_durations(20)
    raw.plot(start=0.5, duration=2000)


    # Selecting channels appropriate for detecting neural responses
:
    # remove channels that are too close together (short channels)
to detect a
    # neural response (less than 1 cm distance between optodes).
```

```
        picks = mne.pick_types(raw.info, meg=False, fnirs=True)
        dists = mne.preprocessing.nirs.source_detector_distances(
            raw.info,
            picks=picks)

        raw.pick(picks[dists > 0.01])
        raw.plot(n_channels=len(raw.ch_names), duration=2000, show_scrollbars=False)



        # Converting raw intensity to optical density
        raw_od = mne.preprocessing.nirs.optical_density(raw)
        raw_od.plot(n_channels=len(raw_od.ch_names),
                    duration=2000, show_scrollbars=False)
        #-----------------------------------------------------------------
        # Remove motion artifacts using TDDR
        raw_od = mne.preprocessing.nirs.temporal_derivative_distribution_repair(raw_od)
        raw_od.plot(duration=2000, show_scrollbars=False)

        # -----------------------------------------------------------------
        # Evaluating the quality of the data - Scalp Coupling Index (SCI)

        # SCI: This method looks for the presence of a prominent synchronous signal in
        # the frequency range of cardiac signals across both photodetected signals

        sci = mne.preprocessing.nirs.scalp_coupling_index(raw_od)
        fig, ax = plt.subplots()
        ax.hist(sci)
        ax.set(xlabel='Scalp Coupling Index', ylabel='Count', xlim=[0, 1])

        # set SCI less than 0.5 as bad channel!
```

```
    raw_od.info['bads'] = list(compress(raw_od.ch_names, sci < 0.5)
)

    # ------------------------------------------------------------
    # convert optical density data to haemoglobin concentration
using modified Beer-Lambert law

    raw_haemo = mne.preprocessing.nirs.beer_lambert_law(raw_od, ppf
=0.57)
    raw_haemo.plot(n_channels=len(raw_haemo.ch_names),
                   duration=2000, show_scrollbars=False)

    tmin,tmax =-8,20
    epochs = mne.Epochs(raw_haemo, events, tmin=tmin, tmax=tmax,
                        event_id=event_dict,proj=True,baseline=(
tmin,0),
                        preload=True, verbose=True)


    # ------------------------------------------------------------
    # Removing heart rate from signal

    # The haemodynamic response has frequency content predominantly
 below 0.5 Hz.
    # An increase in activity around 1 Hz can be seen in the data
that is due to
    # the personâ s heart beat and is unwanted. So we use a low
pass filter to
    # remove this. A high pass filter is also included to remove
slow drifts in the data.

    fig = raw_haemo.plot_psd(average=True)
    fig.suptitle('Before filtering', weight='bold', size='x-large')
    fig.subplots_adjust(top=0.88)

    # filter out heart rate
    raw_haemo = raw_haemo.filter(l_freq=0.05, h_freq=0.7,
```

```
h_trans_bandwidth=0.2,
                                    l_trans_bandwidth=0.02)

    fig = raw_haemo.plot_psd(average=True)
    fig.suptitle('After filtering', weight='bold', size='x-large')
    fig.subplots_adjust(top=0.88)


    # ----------------------------------------------------------
    # Extract epochs

    # Now that the signal has been converted to relative haemoglobin concentration,
    # and the unwanted heart rate component has been removed, we can extract epochs
    # related to each of the experimental conditions.
    # First we extract the events of interest / visualise them to ensure they are correct.

    events, event_dict = mne.events_from_annotations(raw_haemo)
    fig = mne.viz.plot_events(events, event_id=event_dict,
                              sfreq=raw_haemo.info['sfreq'])
    fig.subplots_adjust(right=0.7)  # make room for the legend

    # define range of epochs, rejection criteria, basline correction and extract epochs.
    # Visualize log of which epochs were dropped.

    reject_criteria = dict(hbo=80e-6)  # rejection criteria
    tmin, tmax = -8, 20

    epochs = mne.Epochs(raw_haemo, events, event_id=event_dict,
                        tmin=tmin, tmax=tmax,
                        reject=reject_criteria, reject_by_annotation=True,
                        proj=True, baseline=(tmin, 0), preload=True,
```

```
                                detrend=None, verbose=True)

        epochs.plot_drop_log()

        # ------------------------------------------------------------
        # View consistency of responses across trials

        # haemodynamic response for our tapping condition. We visualise the response
        # for both the oxy- and deoxyhaemoglobin, and observe the expected peak in
        # HbO at around 6 seconds consistently across trials, and the consistent dip in
        # HbR that is slightly delayed relative to the HbO peak.

        epochs['Sp'].plot_image(combine='mean', vmin=-30, vmax=30,
                                ts_args=dict(ylim=dict(hbo=[-15, 15],
                                                       hbr=[-15, 15])))

        epochs['Rot-TS'].plot_image(combine='mean', vmin=-30, vmax=30,
                                    ts_args=dict(ylim=dict(hbo=[-15, 15],
                                                           hbr=[-15, 15])))

        epochs['Rot-Blesser'].plot_image(combine='mean', vmin=-30, vmax=30,
                                         ts_args=dict(ylim=dict(hbo=[-15, 15],
                                                                hbr=[-15, 15])))
        epochs['NV'].plot_image(combine='mean', vmin=-30, vmax=30,
                                ts_args=dict(ylim=dict(hbo=[-15, 15],
                                                       hbr=[-15, 15])))
        epochs['NV-Rot'].plot_image(combine='mean', vmin=-30, vmax=30,
                                    ts_args=dict(ylim=dict(hbo=[-15, 15],
```

```
                                                                hbr=[-15, 15])))

    # ------------------------------------------------------------
    # Plot standard fNIRS response image

    # generate the most common visualisation of fNIRS data: plotting both the
    # HbO and HbR on the same figure to illustrate the relation between the two signals.


    evoked_dict = {'Sp/HbO': epochs['Sp'].average(picks='hbo'),
                   'Sp/HbR': epochs['Sp'].average(picks='hbr'),
                   'NV/HbO': epochs['NV'].average(picks='hbo'),
                   'NV/HbR': epochs['NV'].average(picks='hbr'),
                   'RotTS/HbO': epochs['Rot-TS'].average(picks='hbo'),
                   'RotTS/HbR': epochs['Rot-TS'].average(picks='hbr'),
                   'RotBlesser/HbO': epochs['Rot-Blesser'].average(picks='hbo'),
                   'RotBlesser/HbR': epochs['Rot-Blesser'].average(picks='hbr'),
                   'NVRot/HbO': epochs['NV-Rot'].average(picks='hbo'),
                   'NVRot/HbR': epochs['NV-Rot'].average(picks='hbr'),
                   }
    # Rename channels until the encoding of frequency in ch_name is fixed
    for condition in evoked_dict:
        evoked_dict[condition].rename_channels(lambda x: x[:-4])

    color_dict = dict(HbO='#AA3377', HbR='b')
    styles_dict = dict(NV= dict(linestyle='dashed'),
                       RotTS=dict(linestyle='dotted'),
```

```
                        RotBlesser =dict(linestyle='dashdot'),
                        NVRot=dict(linestyle=':')
                        )
        pick = 'S9_D8'
        mne.viz.plot_compare_evokeds(evoked_dict,picks=pick,combine="mean", ci=0.95,
                                     colors=None,cmap='jet', styles=styles_dict,show_sensors=True)

        #comp[0].savefig(save_plots)
        # ------------------------------------------------------------
        # View topographic representation of activity

        # view how the topographic activity changes throughout the response
        times = np.arange(-3.5, 20., 3.0)
        topomap_args = dict(extrapolate='local')
        epochs['Sp'].average(picks='hbo').plot_joint(times=times,
                                                     topomap_args=topomap_args)

        # ------------------------------------------------------------
        # Compare Sp and Rot-TS

        times = np.arange(4.0, 11.0, 1.0)
        epochs['Sp'].average(picks='hbo').plot_topomap(
            times=times, **topomap_args)
        epochs['Rot-TS'].average(picks='hbo').plot_topomap(
            times=times, **topomap_args)

        fig, axes = plt.subplots(nrows=2, ncols=4, figsize=(9, 5),
                                 gridspec_kw=dict(width_ratios=[1, 1, 1, 0.1]))
        vmin, vmax, ts = -8, 8, 9.0

        evoked_sp = epochs['Sp'].average()
        evoked_rot_ts = epochs['Rot-TS'].average()
```

```
    evoked_sp.plot_topomap(ch_type='hbo', times=ts, axes=axes[0, 0],
                           vmin=vmin, vmax=vmax, colorbar=False,
                           **topomap_args, image_interp='cubic')
    evoked_sp.plot_topomap(ch_type='hbr', times=ts, axes=axes[1, 0],
                           vmin=vmin, vmax=vmax, colorbar=False,
                           **topomap_args, image_interp='cubic')
    evoked_rot_ts.plot_topomap(ch_type='hbo', times=ts, axes=axes[0, 1],
                               vmin=vmin, vmax=vmax, colorbar=False,
                               **topomap_args, image_interp='cubic')
    evoked_rot_ts.plot_topomap(ch_type='hbr', times=ts, axes=axes[1, 1],
                               vmin=vmin, vmax=vmax, colorbar=False,
                               **topomap_args, image_interp='cubic')

    evoked_diff = mne.combine_evoked([evoked_sp, evoked_rot_ts], weights=[1, -1])

    evoked_diff.plot_topomap(ch_type='hbo', times=ts, axes=axes[0, 2:],
                             vmin=vmin, vmax=vmax, colorbar=True,
                             **topomap_args)
    evoked_diff.plot_topomap(ch_type='hbr', times=ts, axes=axes[1, 2:],
                             vmin=vmin, vmax=vmax, colorbar=True,
                             **topomap_args, image_interp='cubic')

    for column, condition in enumerate(
            ['Sp', 'Rot-TS', 'Sp/Rot-TS']):
        for row, chroma in enumerate(['HbO', 'HbR']):
```

```
                axes[row, column].set_title('{}: {}'.format(chroma, condition))
        fig.tight_layout()
        fig.savefig(save_plots + 'hbo_hbr')
        # ------------------------------------------------------------
        # get individual waveforms that drive topographic plot:

        fig, axes = plt.subplots(nrows=1, ncols=1, figsize=(6, 4))
        mne.viz.plot_evoked_topo(epochs['Sp'].average(picks='hbo'), color='b',
                                 axes=axes, legend=False)
        mne.viz.plot_evoked_topo(epochs['Rot-TS'].average(picks='hbo'), color='r',
                                 axes=axes, legend=False)

        # Tidy the legend:
        leg_lines = [line for line in axes.lines if line.get_c() == 'b'][:1]
        leg_lines.append([line for line in axes.lines if line.get_c() == 'r'][0])
        fig.legend(leg_lines, ['Sp', 'Rot-TS'], loc='lower right')
        fig.savefig(save_plots + '_individual waveform_sp_rot_ts')
        # ------------------------------------------------------------

        extrapolations = ['local', 'head', 'box']
        fig, axes = plt.subplots(figsize=(7.5, 4.5), nrows=2, ncols=3)

        # Here we look at channels, and use a custom head sphere to get all the
        # sensors to be well within the drawn head surface


        for axes_row, ch_type in zip(axes, ('hbo', 'hbr')):
            for ax, extr in zip(axes_row, extrapolations):
                evoked_sp.plot_topomap(0.1, ch_type=ch_type, size=2, extrapolate=extr,
                                       axes=ax, show=False, colorbar=
```

```
False,
                                        sphere=(0., 0., 0., 0.09))
                ax.set_title('%s %s' % (ch_type.upper(), extr),
fontsize=14)
    fig.tight_layout()

    evoked_sp.plot_topomap(20.0, ch_type='hbo', show_names=True,
colorbar=True,
                               size=6, res=128, title='Speech response'
,
                               time_unit='s')
    plt.subplots_adjust(left=0.01, right=0.99, bottom=0.01, top
=0.88)

    mne.viz.plot_sensors( raw_haemo.info, kind='topomap', ch_type=
None, title=None, show_names=True,
                              ch_groups=None, to_sphere=True, axes=None,
 block=False, show=True, sphere='auto', pointsize=None, linewidth=2,
 verbose=None)
    mne.viz.plot_sensors( raw_haemo.info, kind='3d', ch_type=None,
title=None, show_names=True, ch_groups=None, to_sphere=True, axes=
None, block=False, show=True, sphere=None, pointsize=None, linewidth
=2, verbose=None)
    # ------------------------------------------------------------
    # Animation
    times = np.arange(0.05, 0.151, 0.01)
    fig, anim = evoked_sp.animate_topomap(
        times=times, ch_type='hbr', frame_rate=2, time_unit='s',
blit=False)

    # ------------------------------------------------------------
    # Show montage with brain template

    subjects_dir = str(mne.datasets.sample.data_path()) + '/
subjects'
    mne.datasets.fetch_hcp_mmp_parcellation(subjects_dir=
subjects_dir, accept=True)
```

```
        labels = mne.read_labels_from_annot('fsaverage', 'HCPMMP1', 'lh', subjects_dir=subjects_dir)
        labels_combined = mne.read_labels_from_annot('fsaverage', 'HCPMMP1_combined', 'lh', subjects_dir=subjects_dir)
        brain = mne.viz.Brain('fsaverage', subjects_dir=None, background='w', cortex='0.5')
        brain.add_sensors(raw.info, trans='fsaverage', fnirs=['channels', 'pairs', 'sources', 'detectors'])

        # specify region of interest ROI
        aud_label = [label for label in labels if label.name == 'L_A1_ROI-lh'][0] #primary auditory cortex 'L_A4_ROI-lh' 'L_A5_ROI-lh' 'L_AAIC_ROI-lh', , 'L_AIP_ROI-lh'
        stg_label = [label for label in labels if label.name == 'L_STGa_ROI-lh'][0] # STG
        audi_lable = [label for label in labels if label.name == 'L_AAIC_ROI-lh'][0] #'L_STSdp_ROI-lh' 'L_STSva_ROI-lh' 'L_STSvp_ROI-lh'
        borca_lable = [label for label in labels if label.name == 'L_IFSa_ROI-lh'][0]  # 'L_IFSa_ROI-lh' nferior frontal gyrus  'L_IFSp_ROI-lh', 'L_IFJp_ROI-lh'  'L_IFJa_ROI-lh'
        aud_2_label =  aud_label = [label for label in labels if label.name == 'L_A5_ROI-lh'][0] # A5
        aud_label = [label for label in labels if label.name == 'L_A1_ROI-lh'][0]


        l = [label for label in labels if label.name] # shows all existing labels
        brain.add_label(aud_label, borders=False, color='blue')
        brain.add_label(stg_label, borders=False, color='red')
        brain.add_label(borca_lable, borders=False, color='green')
        brain.add_label(aud_2_label, borders=False, color='yellow')
        brain.add_label(audi_lable, borders=False, color='pink')
        brain.show_view(azimuth=180, elevation=80, distance=450)

# list of all participants
```

```
list_par = ['01','04','05','07','08','09',11,12,16,17,18,30,31,32,
    33,34,35,36,37] # problem 15 und 06

# participant S04 is used as an example in the BA-thesis
list_par = ['04']
get_plots(list_par)
```

Preprocessing_individual.py

Bibliography

Daniel A. Abrams, Srikanth Ryali, Tianwen Chen, Evan Balaban, Daniel J. Levitin, and Vinod Menon. Multivariate activation and connectivity patterns discriminate speech intelligibility in Wernicke's, Broca's, and Geschwind's areas. *Cerebral Cortex*, 23(7):1703–1714, jul 2013. ISSN 10473211. doi: 10.1093/cercor/bhs165.

Amyn M. Amlani, Jerry L. Punch, and Teresa Y.C. Ching. Methods and Applications of the Audibility Index in Hearing Aid Selection and Fitting. *Trends in Amplification*, 6(3):81–129, aug 2002. ISSN 10847138. doi: 10.1177/108471380200600302.

Stefan Appelhoff, Matthew Sanderson, Teon Brooks, Marijn van Vliet, Romain Quentin, Chris Holdgraf, Maximilien Chaumon, Ezequiel Mikulan, Kambiz Tavabi, Richard Höchenberger, Dominik Welke, Clemens Brunner, Alexander Rockhill, Eric Larson, Alexandre Gramfort, and Mainak Jas. MNE-BIDS: Organizing electrophysiological data into the BIDS format and facilitating their analysis. *Journal of Open Source Software*, 4(44):1896, 2019. doi: 10.21105/joss.01896.

Wesley B. Baker, Ashwin B. Parthasarathy, David R. Busch, Rickson C. Mesquita, Joel H. Greenberg, and A. G. Yodh. Modified Beer-Lambert law for blood flow. *Biomedical Optics Express*, 5(11):4053, nov 2014. ISSN 2156-7085. doi: 10.1364/boe.5.004053.

Barry Blesser. Speech Perception Under Conditions of Spectral Transformation: I. Phonetic Characteristics. *Journal of Speech and Hearing Research*, 15(1):5–41,

mar 1972. ISSN 0022-4685. doi: 10.1044/jshr.1501.05. URL http://pubs.asha.org/doi/10.1044/jshr.1501.05.

Pierre Bouguer. Essai d'optique sur la gradation de la lumiere. *Traite d'optique sur la gradation de la lumiere*, 1729.

Britton Chance. Optical Method. *Annual Review of Biophysics and Biophysical Chemistry*, 20(1):1–30, jun 1991. ISSN 0883-9182. doi: 10.1146/annurev.bb.20.060191.000245. URL https://www.annualreviews.org/doi/10.1146/annurev.bb.20.060191.000245.

Matthew H. Davis, Ingrid S. Johnsrude, Alexis Hervais-Adelman, Karen Taylor, and Carolyn McGettigan. Lexical information drives perceptual learning of distorted speech: Evidence from the comprehension of noise-vocoded sentences. *Journal of Experimental Psychology: General*, 134(2):222–241, may 2005. ISSN 00963445. doi: 10.1037/0096-3445.134.2.222.

Anna Devor, Sava Sakadžic, SakadžicÂ´ SakadžicÂ´3, Vivek J Srinivasan, Mohammad A Yaseen, Krystal Nizar, Payam A Saisan, Peifang Tian, Anders M Dale, Sergei A Vinogradov, Maria Angela Franceschini, and David A Boas. Frontiers in optical imaging of cerebral blood flow and metabolism. *Journal of Cerebral Blood Flow Metabolism*, 32(7):1259–1276, jul 2012. doi: 10.1038/jcbfm.2011.195. URL www.jcbfm.com.

M. Essenpreis, M. Cope, C. E. Elwell, S. R. Arridge, P. Van der Zee, and D. T. Delpy. Wavelength dependence of the differential pathlength factor and the log slope in time-resolved tissue spectroscopy, 1993. ISSN 0065-2598.

S. Evans, J. S. Kyong, S. Rosen, N. Golestani, J. E. Warren, C. McGettigan, J. Mourão-Miranda, R. J.S. Wise, and S. K. Scott. The pathways for intelligible speech: Multivariate and univariate perspectives. *Cerebral Cortex*, 24(9): 2350–2361, 2014. ISSN 14602199. doi: 10.1093/cercor/bht083.

Frank A. Fishburn, Ruth S. Ludlum, Chandan J. Vaidya, and Andrei V. Medvedev. Temporal Derivative Distribution Repair (TDDR): A motion correction method

for fNIRS. *NeuroImage*, 184:171–179, jan 2019. ISSN 10959572. doi: 10.1016/j.neuroimage.2018.09.025.

W Tecumseh Fitch, Bart De Boer, Neil Mathur, and Asif A Ghazanfar. Monkey vocal tracts are speech-ready. *Science advances*, 2(12):e1600723, 2016.

N. R. French and J. C. Steinberg. Factors Governing the Intelligibility of Speech Sounds. *Journal of the Acoustical Society of America*, 19(1):90–119, jan 1947. ISSN NA. doi: 10.1121/1.1916407. URL `http://asa.scitation.org/doi/10.1121/1.1916407`.

Daniel Friedrichs, Kurt Steinmetzger, Andrew Clark, and Stuart Rosen. Assessing the cortical processing of spectrally rotated speech using functional near-infrared spectroscopy. *The Journal of the Acoustical Society of America*, 146(4):2954–2954, oct 2019. ISSN 0001-4966. doi: 10.1121/1.5137257. URL `http://asa.scitation.org/doi/10.1121/1.5137257`.

Helene Girouard and Costantino Iadecola. Neurovascular coupling in the normal brain and in hypertension, stroke, and Alzheimer disease, jan 2006. ISSN 87507587.

Alexandre Gramfort, Martin Luessi, Eric Larson, Denis A. Engemann, Daniel Strohmeier, Christian Brodbeck, Roman Goj, Mainak Jas, Teon Brooks, Lauri Parkkonen, and Matti S. Hämäläinen. MEG and EEG data analysis with MNE-Python. *Frontiers in Neuroscience*, 7(267):1–13, 2013. doi: 10.3389/fnins.2013.00267.

C. G. Gross, D. B. Bender, and C. E. Rocha-Miranda. Visual receptive fields of neurons in inferotemporal cortex of the monkey. *Science*, 166(3910):1303–1306, 1969. ISSN 00368075, 10959203. URL `http://www.jstor.org/stable/1727144`.

Gregory Hickok and David Poeppel. Dorsal and ventral streams: A framework for understanding aspects of the functional anatomy of language. *Cognition*, 92(1-2):67–99, 2004. ISSN 00100277. doi: 10.1016/j.cognition.2003.10.011.

L. Kocsis, P. Herman, and A. Eke. The modified Beer-Lambert law revisited. *Physics in Medicine and Biology*, 51(5), mar 2006. ISSN 00319155. doi: 10.1088/0031-9155/51/5/N02.

Werner Mäntele and Erhan Deniz. *UVâVIS absorption spectroscopy: Lambert-Beer reloaded*, volume 173. Elsevier B.V., feb 2017. doi: 10.1016/j.saa.2016.09.037.

Thomas G. Mayerhöfer, Susanne Pahlow, and Jürgen Popp. The BouguerâBeerâLambert Law: Shining Light on the Obscure. *ChemPhysChem*, 21 (18):2029–2046, sep 2020. ISSN 1439-4235. doi: 10.1002/cphc.202000464. URL `https://onlinelibrary.wiley.com/doi/10.1002/cphc.202000464`.

George Armitage Miller. Language and communication. 1951.

Mortimer Mishkin, Leslie G. Ungerleider, and Kathleen A. Macko. Object vision and spatial vision: two cortical pathways, jan 1983. ISSN 01662236.

P. Muller. Glossary of terms used in physical organic chemistry: (IUPAC Recommendations 1994). *Pure and Applied Chemistry*, 66(5):1077–1184, jan 1994. ISSN 13653075. doi: 10.1351/pac199466051077.

Noman Naseer and Keum Shik Hong. fNIRS-based brain-computer interfaces: A review, jan 2015. ISSN 16625161.

Kayoko Okada, Feng Rong, Jon Venezia, William Matchin, I. Hui Hsieh, Kourosh Saberi, John T. Serences, and Gregory Hickok. Hierarchical organization of human auditory cortex: Evidence from acoustic invariance in the response to intelligible speech. *Cerebral Cortex*, 20(10):2486–2495, 2010. ISSN 10473211. doi: 10.1093/cercor/bhp318.

Luca Pollonini, Cristen Olds, Homer Abaya, Heather Bortfeld, Michael S. Beauchamp, and John S. Oghalai. Auditory cortex activation to natural speech and simulated cochlear implant speech measured with functional near-infrared spectroscopy. *Hearing Research*, 309:84–93, mar 2014. ISSN 18785891. doi: 10.1016/j.heares.2013.11.007.

Dale Purves, George J Augustine, David Fitzpatrick, Lawrence C Katz, Anthony-Samuel LaMantia, James O McNamara, and S Mark Williams. Tuning and Timing in the Auditory Nerve. 2001.

R. Quian Quiroga, L. Reddy, G. Kreiman, C. Koch, and I. Fried. Invariant visual representation by single neurons in the human brain. *Nature*, 435(7045):1102–1107, jun 2005. ISSN 00280836. doi: 10.1038/nature03687.

Josef P. Rauschecker and Sophie K. Scott. Maps and streams in the auditory cortex: Nonhuman primates illuminate human speech processing, jun 2009. ISSN 10976256.

Felix Scholkmann, Stefan Kleiser, Andreas Jaakko Metz, Raphael Zimmermann, Juan Mata Pavia, Ursula Wolf, and Martin Wolf. A review on continuous wave functional near-infrared spectroscopy and imaging instrumentation and methodology. *NeuroImage*, 85:6–27, 2014. ISSN 10538119. doi: 10.1016/j.neuroimage.2013.05.004. URL `http://dx.doi.org/10.1016/j.neuroimage.2013.05.004`.

Sophie K. Scott, C. Catrin Blank, Stuart Rosen, and Richard J.S. Wise. Identification of a pathway for intelligible speech in the left temporal lobe. *Brain*, 123(12): 2400–2406, 2000. ISSN 00068950. doi: 10.1093/brain/123.12.2400.

Robert V. Shannon, Fan Gang Zeng, Vivek Kamath, John Wygonski, and Michael Ekelid. Speech recognition with primarily temporal cues. *Science*, 270(5234): 303–304, 1995. ISSN 00368075. doi: 10.1126/science.270.5234.303.

Kurt Steinmetzger, Zhengzheng Shen, Helmut Riedel, and André Rupp. Auditory cortex activity measured using functional near-infrared spectroscopy (fNIRS) appears to be susceptible to masking by cortical blood stealing. *Hearing Research*, 396:108069, 2020. ISSN 18785891. doi: 10.1016/j.heares.2020.108069. URL `https://doi.org/10.1016/j.heares.2020.108069`.

Kurt Steinmetzger, Esther Megbel, Zhengzheng Shen, Martin Andermann, and André Rupp. Cortical activity evoked by voice pitch changes: A combined

fNIRS and EEG study. *Hearing Research*, 420:108483, 2022. ISSN 18785891. doi: 10.1016/j.heares.2022.108483. URL `https://doi.org/10.1016/j.heares.2022.108483`.

Thomas M. Talavage, Martin I. Sereno, Jennifer R. Melcher, Patrick J. Ledden, Bruce R. Rosen, and Anders M. Dale. Tonotopic Organization in Human Auditory Cortex Revealed by Progressions of Frequency Sensitivity. *Journal of Neurophysiology*, 91(3):1282–1296, mar 2004. ISSN 00223077. doi: 10.1152/jn.01125.2002.

Sigita Venclove, Algis Daktariunas, and Osvaldas Ruksenas. Functional near-infrared spectroscopy: A continuous wave type based system for human frontal lobe studies. *EXCLI Journal*, 14:1145–1152, oct 2015. ISSN 16112156. doi: 10.17179/excli2015-614.

C. M. Wessinger, J. Vanmeter, B. Tian, J. Van Lare, J. Pekar, and J. P. Rauschecker. Hierarchical organization of the human auditory cortex revealed by functional magnetic resonance imaging. *Journal of Cognitive Neuroscience*, 13(1):1–7, jan 2001. ISSN 0898929X. doi: 10.1162/089892901564108.

Margaret T.T. Wong-Riley. Cytochrome oxidase: an endogenous metabolic marker for neuronal activity, 1989. ISSN 01662236.