Exploring the role of beta activity over the primary motor cortex in Motor Sequence Learning

Master Thesis by Daphne Titsing

Faculty of Behavioural, Management and Social sciences Human Factors and Engineering Psychology
University of Twente

First supervisor: Dr. Russell W. Chan
Second supervisor: Dr. Rob H.J. van der Lubbe

UNIVERSITY
OF TWENTE.

Acknowledgements

First, I would like to thank my supervisors Dr. Russell W. Chan and Dr. Rob H.J. van der Lubbe for their guidance, support, and enthusiasm. I greatly appreciate working with you and all the opportunities you have provided me with. Thank you for the inspiration and motivation that you have given me over the past two years. I think I have learned a lot about motor sequence learning and EEG. Writing this thesis, carrying out the experiment, providing workshops, and assisting in the organization of the motor learning symposium and our discussions led me to conclude that would love to keep exploring the world of motor sequence learning and EEG. Furthermore, I would like to thank everyone who has taken the time to help me with my experiment. I would especially like to thank those who have spent their hours working with me on my script or assisting in my experiment. I wish you all the best in your future studies or careers. Last but certainly not least, I would like to give my special thanks to my boyfriend, parents, and friends. Thank you for always supporting and helping me. I truly appreciate you and all you have done to support me.

Abstract

Previous studies linked enlarged beta desynchronization during motor preparation and execution (movement-related beta desynchronization; MRBD) and enlarged beta synchronization during post-movement (post-movement beta rebound; PMBR) to motor control. However, what remains unclear is how both beta synchronization and desynchronization during motor are linked to MSL. Therefore, this thesis aimed to test whether beta activity over M1 reflects motor sequence learning. Furthermore, it aimed to test whether RT correlates with beta activity over M1. If there is a relationship between RT and Beta activity over M1, this would mean that it would be possible to predict when motor learning expertise would be gained. In this thesis, participants carried out a go/nogo DSP task. Event-related desynchronization and synchronization (ERD/S) values were extracted from three separate bands: β1 (12–17Hz), β2 (18–23Hz) and β3 (24–29Hz) in 100ms time windows for the motor preparation, motor execution and post-movement phase. The results revealed a larger β2 ERD post-training compared to pre-training during motor preparation. It was also revealed that there were no changes observed in MRBD. Based on the literature it was suggested that beta activity during motor execution may reflect task difficulty. Additionally, an enlarged post-movement beta rebound (PMBR) over M1 was observed for both the left and right hand in ß1 and the left hand in ß2 and ß3. This supports previous findings in which enlarged PMBR was linked to error-based motor sequence learning. Lastly, a positive relationship was observed between ß2 ERD/S over M1 for left-hand sequences in Block 5 and a negative relationship in Block 1. The relationship between ERD/S and RT should be further investigated in future research to confirm these findings.

Contents

Abbreviations

MSL = Motor Sequence Learning

DSP = Discrete Sequence Production

M1 = Primary motor cortex

DPM =Dual-Processor Model

C-SMB = Cognitive Framework for Sequential Motor Behavior

ERD = Event Related Desynchronization

ERS = Event Related Synchronization

ERD/S = Event Related Desynchronization and Synchronization

MRBD = Movement Related Beta Desynchronization

PMBR = Post Movement Beta Rebound

ß1 = Beta 1; 12-17 Hz

ß2 = Beta 2; 18-23 Hz

ß3= Beta 3; 24-29 Hz

1. Introduction

Most of our goal-directed everyday actions, such as lacing a shoe or playing on the piano, consist of learning and executing sequential movements (Lee & Quessy, 2003). Learning to execute a series of movements is initially slow and requires attention. With practice, the execution of those movements becomes faster and requires less attention (de Kleine, 2009). This form of learning is called Motor Sequence Learning (MSL). Moreover, MSL can be defined as 'the acquisition of the skill to rapidly and accurately produce a sequence of movements with limited effort and/or attentional monitoring' (Abrahamse et al., 2013, p. 1). Experimental paradigms, like the Discrete Sequence Production (DSP) task (Verwey, 2001) and the go/nogo version of the DSP task (de Kleine & Van der Lubbe, 2011), were developed to contribute to the understanding of the underlying cognitive processes of MSL. Moreover, these paradigms gave rise to the Dual-Processor Model (DPM) (Abrahamse et al., 2013; Verwey, 2001) and, more recently, the Cognitive Framework of Sequential Motor Behaviour (C-SMB) (Verwey et al., 2015). Despite the availability of such frameworks, the underlying brain processes of motor sequence learning described are only recently being validated. Research with neuroimaging techniques could help to provide more insight into the neuro-mechanisms underlying MSL in the brain, to which this thesis aims to provide some development (Abrahamse et al. 2013).

Previous EEG studies have highlighted that beta oscillations over the primary motor cortex (M1) before, during, and after a sequence of movements can serve as important neuro-markers of sequential movement. However, the gap in the literature lies in the role of beta oscillations over M1 during the different stages of MSL (Barone & Rossiter, 2021). Therefore, the first aim of this thesis is to uncover the beta oscillations over the primary motor cortex (M1) during the three periods of MSL. In this thesis, several predictions were made on how MSL expertise is reflected in lower, middle, and upper beta oscillation changes over M1 during the preparation, execution, and post-movement phases of sequential movement. These predictions are, however, mainly exploratory to establish relationships with behavioural results. In this thesis, the go/nogo version of the DSP task was used.

Previous DSP and go/nogo DSP task studies (Abrahamse et al., 2013; de Kleine and Van der Lubbe, 2011; Verwey, 2001; Sobierajewicz et al., 2017,) have mainly investigated RT changes in relation to expertise during the different stages of motor sequence learning. In this thesis, a combination of RT and EEG measures is used to test whether there is a relationship between beta activity over M1 and RT. Such a relationship could reveal important information about when expertise would be obtained and what expertise looks like in terms of RT and Beta activity over M1. Therefore, the second aim of this thesis is to test whether there is a relationship between beta activity over M1 and RT. The introduction will be preceded by a short description of the DSP task.

1.1 The Discrete Sequence Production (DSP) task

The DSP task is a key pressing task developed by Verwey (2001) in which participants have to respond to key-specific sequences shown on a computer screen. In the original DSP task, the participant's task is to respond immediately after each stimulus by pressing the spatially corresponding key. In the go/nogo DSP task (de Kleine & Van der Lubbe, 2011) the sequence is only reproduced when all stimuli corresponding to one sequence are presented and followed by a go-signal. In case of a no-go signal, the participant is instructed not to produce the sequence. According to de Kleine & Van der Lubbe (2011) unlike the original DSP task, the go/nogo DSP task allows for distinguishing between the motor preparation, motor execution and post-movement phase. Since this thesis focuses on the three phases of MSL, the use of go/nogo DSP task would facilitate the separation of the phases. The following section will use the DPM to discover what RT pattern can be observed and the C-SMB framework to introduce the relationship between M1 and MSL.

1.2 The Dual-Processor Model and the Cognitive Framework for Sequential Motor Behavior (C-SMB)

Based on the first three experiments with the DSP task, Verwey (2001) concluded that through learning, familiar key sequences become represented in the brain as single responses rather than a set of individual responses. These single responses are referred to as motor chunks. Through the lens of the DPM, MSL arises as these motor chunks develop through practice with the DSP task, resulting in less demanding, faster, and more automatic processing and executing of key sequences. According to Abrahamse et al. (2013), sequence production can be divided into three processes: Initiation, concatenation, and execution. MSL may be reflected in changes in these three processes. After motor sequence learning a typical RT pattern can be seen (see Figure 1).

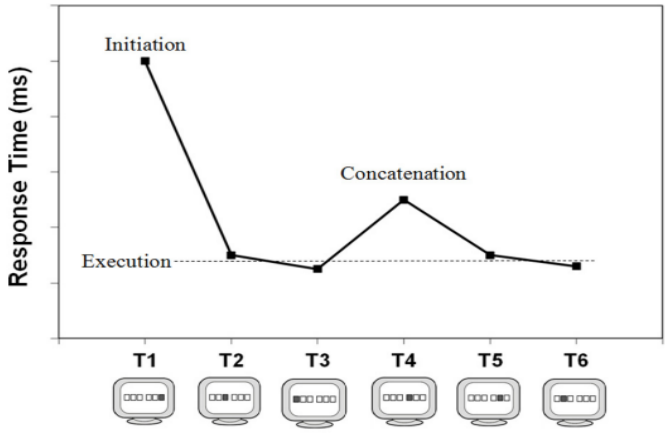


Figure 1. The three distinct phases of executing a sequence as outlined by Abrahamse et al. (2013). *Note.* Initiation is reflected in T1 a. Execution starts at T1 and ends at T6. Concatenation is reflected in T3.

The initiation phase is the start of sequence production and refers to the selection and preparation of the sequence. Here the RTs are long due to the retrieval of information from working memory. However, soon there is a large reduction of RT seen between the initiation and execution phase of the first sequence (see T1 and T2 in Figure 1). The execution phase refers to the execution of responses (T1 to T6 in Figure 1). Typically, the RTs in this phase are short. The concatenation phase refers to the process of executing distinct motor chunks within a sequence in rapid succession. Concatenation is typically characterized by longer RTs (see T3 in Figure 1). For longer sequences (exceeding 4 to 5 elements) more than one motor chunk may be needed. Because it takes time to move from one motor chunk to another, RTs are longer. For this thesis, the concatenation effect, as shown in Figure 1, is expected to be observed when individuals carry out the go/nogo DSP task.

The Cognitive Framework for Sequential Motor Behavior (C-SMB) was proposed by Verwey et al. (2015) as an updated version of the DPM. According to Verwey et al. (2015), the C-SMB provides an explanation of the underlying cognitive processes of motor sequence learning for multiple sequential motor tasks. Therefore, in this thesis, the C-SMB will be used to explain how M1 may be related to MSL. The framework holds that motor sequence execution in diverse motor tasks can be controlled by a central processor that uses central-symbolic representations, and a motor processor that uses sequence-specific motor representations (or motor chunks). According to Verwey et al. (2015), M1 acts as an interface between cortico-subcortical networks that together make up the motor processor. S1 acts as an interface between cortico-subcortical networks that form a perceptual processor. These two processors together with a distributed set of other brain regions form the central processor. These processors can operate in three modes: the reaction mode, execution mode and chunking mode. In the reaction mode, central-symbolic representations are formed by the central processor through a cognitive loop. The preceding element of the sequence is determined by the central processor and then moved on to the motor processor. The motor processor stores sequence-specific motor representations including motor chunks, in the motor buffer. Only after the motor buffer is combined with all the elements that constitute the movement sequence, movement execution takes place in the execution mode. After repeatedly carrying out the movement, motor chunks are represented and stabilized in long-term memory. In the chunking mode, the motor processor successfully searches and retrieves the motor chunks stored in the long-term memory through a motor loop, which decreases the load on the central processor. Like in the DPM, concatenation of successive subsequences is required when a sequence exceeds three to five elements. This transition from one subsequence to the other is relatively slow. With expertise, the central processor becomes less involved, and the motor processor becomes more involved (Verwey et al., 2010, 2014). This enables the individual to speed up the transition process and thus execute sequential movements faster and more automatically. One study that supports the C-SMB framework is the one by de Kleine and Van der Lubbe

(2011). In that study, it was found that when participants carried out unfamiliar sequences, the load on motor preparation and visual-working memory was increased compared to familiar sequences. De Kleine and Van der Lubbe (2011) suggested that as the load on planning and organization decreases and expertise is gained, demand on the central processor decreases and the involvement of the motor processor increases.

So far, the C-SMB has shown that the central processor and motor processor are involved in motor sequence learning that can flexibly operate on three different modes of learning. Based on the C-SMB, it is postulated that M1 is related to the motor processor, one might propose that M1 is more involved as MSL expertise increases. However, this explanation does not fully touch upon the relationship between the activity over M1 and MSL. As mentioned by Verwey et al. (2015), neuro-imaging studies are needed to provide more clarity on how activity over M1 is related to MSL. An insightful neuro-imaging study comes from Verwey et al. (2019). In that study, fMRI was used to test, under the assumption of the C-SMB, whether a specific set of brain areas was active during the execution of unfamiliar sequences, familiar sequences, and the execution of both sequences. The study showed that there was activity in the primary somatosensory cortex and left M1 (contralateral to movement) when executing unfamiliar sequences. Verwey et al. (2019) suggested that M1 activity may be related to motor sequence learning as it could reflect the learning of new hand movements (Verwey et al. (2015). This suggestion indicates that before expertise is gained activity over M1 may be shown during the production of sequences. However, since the literature on the role of M1 in MSL is scarce, it must be stressed that this suggestion is solely preliminary and needs further evidence. Besides, it remains largely unclear what the magnitude and frequency range of this activity over M1 would be before and after expertise is gained. Therefore, further focus is laid on literature that investigates the activity over M1 in specific frequency ranges both before and after sequential movement expertise. Previous EEG studies have especially focused on the relationship between the beta activity over M1 and sequential movement (expertise) as will be described in the following paragraphs.

1.3 Current EEG work on motor sequence behaviour

Motor sequence behaviour falls under the category of voluntary movement that can be divided into internally (self-paced) or externally paced (stimulated) movement **(**Pfurtscheller and Neuper, 2003). It was found that internally or externally paced movement activity within thalamocortical network dynamics leads to characteristic EEG patterns (Pfurtscheller and Neuper, 2003). When studying voluntary movements, two EEG pattern types can be observed: event-related desynchronization (ERD) and event-related synchronization, (ERS) (Pfurtscheller & Lopes da Silva, 1999). ERD refers to a decrease in power relative to the baseline. ERS refers to an increase in power relative to the baseline (Pfurtscheller & Lopes da Silva, 1999). According to Pfurtscheller and Lopes da Silva (1999), ERD and ERS (ERD/S)

phenomena are generated by changes in at least one parameter that controls brain oscillations in neuronal networks. A parameter can be the dynamics of synaptic processes and the intrinsic membrane properties of the neurons, the strength and extent of interconnections between network elements that are often formed by feedback loops (e.g., thalamocortical or cortico-cortical) controlled by general or local neurotransmitter systems. ERD and ERS patterns can be characterized over three periods of movement: motor preparation (before movement), motor execution (during movement) and post-movement (after movement) period. Previous literature has focused on unravelling ERD/S patterns to these three periods of movement. In this thesis, the focus lies only on the oscillatory beta activity (ERD/S) over M1 in relation to MSL. Before going into more depth about this relationship, the next section will provide more detail about previous studies that have investigated the beta-activity neuro markers of sequential movement over M1. This allows for the improvement of predictions on the relationship between beta activity over M1 and sequential movements for this thesis.

1.4 Beta ERD/S in relation to movement

According to Little et al. (2019) and Espenhahn et al. (2017) the beta frequency (~12-29 Hz) range has been recognized to be linked to movement for almost a century. One study that has linked activity in the beta frequency range to movement is the one by Pfurtscheller and Lopes da Silva (1999). Pfurtscheller and Lopes da Silva (1999) found that desynchronization in the lower beta (10-12 Hz) band was localized close to M1. Moreover, it was found that this desynchronization starts about two seconds before movement. ERD in the lower beta band over M1 was observed to be contralateral to the used hand from the start and to be bilateral right before movement. Similar results were found in the go/nogo study by Van der Lubbe et al. (2021) in which beta ERD was present in the preparation period over the primary motor areas in both the upper (16.0–24.0 Hz) and lower (12.2–18.4 Hz) beta bands. Beta ERD was larger for motor execution (movement-related beta desynchronization or MRBD) compared to motor preparation and motor imagery. Furthermore, Jurkiewicz et al. (2006) found that beta ERD was bilateral and started around two seconds before movement. The beta ERD was stronger on the contralateral side of movement shortly after movement execution.

Pfurtscheller and Lopes da Silva (1999) also found another movement phenomenon representing itself in lower beta synchronization over M1 shortly after movement which peaked around one second after the execution of movement. This phenomenon, also called post-movement beta rebound (PMBR), was said to be relatively robust. According to Pfurtscheller and Lopes da Silva (1999) PMBR could be observed in nearly every subject in finger, hand, arm and as well as foot movement (Pfurtscheller et al., 1998; Pfurtscheller et al., 1999). Peaks of beta activity (beta bursts) during the post-movement phase appeared to be varying in frequency within the beta range for every subject. For example, one subject revealed synchronization after arm movement at 18-23 Hz and after finger movement at 13-19 Hz.

Similar results were found by Jurkiewicz et al. (2006) as PMBR was present at 500 ms to 1000 ms after movement execution.

Based on their ERD/S findings, Pfurtscheller and Lopes da Silva (1999) suggested that beta ERS reflects a deactivated state of the motor cortex, while beta ERD reflects an activated state of the motor cortex. Beta ERD is suggested to play an inhibitory role in the execution and planning of movement. This was suggested to be important for limiting and controlling excitatory processes that do not contribute directly to movement-related outcomes. In favour of the suggestions by Pfurtscheller and Lopes da Silva (1999), Jurkiewicz et al., (2006) found that ipsilateral resynchronization over M1 is involved in the executive control of finger movements by suppressing (inhibiting) mirror movement activity. Concerning PBMR, Jurkiewicz et al. (2006) suggested that ipsilateral and contralateral PMBR may contribute differently to the control of movement.

What these studies have in common is that they highlight the inhibitory and controlling role of sequential movements that beta activity plays over M1. More recent studies have also found this connection. For example, Barone and Rossiter (2021) highlighted the 'status-quo' hypothesis in support of the study of Engel and Fries (2010). The hypothesis suggests that desynchronization over M1 during the motor preparation phase is linked to the release of inhibition and the initiation of a motor plan. Therefore, it is to be expected that there will be desynchronization over the primary cortex during the motor preparation phase of the individuals who carry out the go/nogo DSP task. Barone and Rossiter (2021) also suggested that the role of PMBR is to preserve (or control) the existing motor states from internal and external noise. In this case, one can think of noise as whatever interferes either internally or externally with the current motor state. Despite this suggestion, Barone and Rossiter (2011) mentioned that there are different views on the role of PMBR. For example, Tan et al. (2016) suggested that PMBR indicates the level of confidence one has about the prediction of a motor outcome. It has also been suggested that PMBR is involved in the resetting of the working memory to prepare cortical networks for upcoming sequences (Pfurtscheller et al. 2005). In terms of the C-SMB this would mean PMBR may be involved in updating the motor buffer to support the execution of the next sequence or the PMBR reflects the confidence in the execution of the upcoming sequence. According to Barone and Rossiter (2021), the various views on the role of beta activity within the sequential movement may be a result of the separation of beta activity into low and high beta frequencies in research (López-Azcárate et al., 2010; Litvak et al., 2011). Moreover, these low and high beta frequencies were suggested to be affected differently by different levels of dopamine (Brown et al., 2001; Priori et al., 2004; Marceglia et al., 2006). The findings by Baron and Rossiter (2021) and Jurkiewicz et al. (2006) suggest that different roles can be ascribed to the PMBR phenomenon. Despite this variety of views one can still expect to see PMBR over the M1 of individuals who carry out the go/nogo DSP task.

So far, the previous section has focused on connecting beta activity to sequential movement. The previously mentioned researchers point to the same direction in that beta ERD over M1 during motor preparation and execution must be linked to the controlling of sequential behaviour through an inhibitory process. However, how that exactly takes place in the context of learning remains suggestive. Besides, PMBR seems to have a more speculative role, with some research suggesting that PMBR acts as a process to maintain a certain motor state, while other research suggests PMBR reflects certainty about the prediction of a motor outcome. What is still missing is a more precise understanding of how ERD during the preparatory period, MRBD and PMBR over M1 look like in both novice and expert learners. The following sections serve as a guide to predict what happens with MRBD and PMBR when an individual learns to execute sequential movements.

1.5 The role of Beta ERD/S in motor sequence learning

To identify how beta ERD/S change contributes to motor learning expertise one could first make a distinction between a poor learner and a good learner. Suggestions on the difference between poor learners and experts in beta activity in motor sequence learning can be inferred from studies on Parkinson's disease (PD), ageing and stroke patients, due to the lack of studies in healthy populations. In the study by Meissner et al. (2018), PD patients and healthy controls were carrying out a serial reaction time task (SRTT; a key pressing task similar to the DSP task). Meissner et al. (2018) found that in PD patients smaller MRBD was present over M1 compared to healthy controls. Besides, PD patients showed reduced motor sequence acquisition and higher susceptibility to interference compared to healthy controls. Therefore, smaller beta MRBD over M1 was linked to diminished motor sequence learning. This thus means that a poor learner may show a smaller beta MRBD compared to an expert. More research has found this link. For example, in a study by Espenhahn et al. (2019) participants carried out a sequence-based wrist flexion and extension task. Espenhahn et al. (2019) highlighted that previous research suggested ageing is related to a reduced ability to learn new motor skills. However, Espenhahn et al. (2019) found comparable motor sequence learning between young and elderly individuals. Despite this, the younger and older individuals showed different ERD levels. The older individuals showed greater ERD before movement and MRBD compared to the younger adults. According to Espenhahn et al. (2013), this may be due to the characteristics of the task or age-related adaptations in M1. Another study by Thibaut et al. (2017) revealed that stroke patients with poor motor functioning had larger beta synchronization over M1 compared to patients with good motor functioning during the execution of movement. Based on the suggestion by Pfurtscheller and Lopes da Silva (1999), the larger beta synchronization found in stroke patients may be linked to less control and inhibition over excitatory processes resulting in poorer motor functioning. Together these studies showed that as motor sequence learning arises, MRBD may become larger. One may therefore expect that throughout the go/nogo DSP

task, individuals will show larger MRBD over M1 as they learn to execute sequential movements.

The role of PMBR in motor sequence learning has also been explored by a few researchers. Tan et al. (2014, 2016) suggested that motor sequence learning is characterized by more confidence and stable motor execution that is facilitated by greater beta activity over M1 during the post-movement phase. According to Haar and Faisel (2020), motor sequence learning is characterized by either an increase or decrease in PMBR depending on the strategy individuals use. Moreover, some tasks may prompt the individual to use one strategy over another. Haar and Faisal (2020) found that when individuals were engaged in reward-based learning PMBR would be decreased. When individuals were engaged in error-based learning an enlarged PMBR would be observed. The go/nogo DSP typically provides error-based feedback in the form of displaying text that says, 'key x is wrong' or 'good'. Based on these findings one might suggest that in the go/nogo DSP task an enlarged PMBR may serve as a neuro marker for sequential movement expertise. Based on this literature about ERD/S and RT during the three phases of sequential movement, the predictions of this thesis can be introduced.

1.6 Research goals and predictions

Firstly, for RT measures it is predicted that concatenation would be observed. Besides, it is tested whether there is a difference in RT between the dominant and non-dominant hand as ERD/S measures will be divided into left and right-hand measures. The key goals of this thesis focus on ERD/S and the relationship between ERD/S and RT. This focus is mainly of exploratory nature as literature about the role of beta activity over M1 in MSL is only recently being validated. The first key goal of this thesis is to test how motor sequence learning is reflected in beta ERD/S over M1 using the go/nogo DSP task. Specifically, the focus was on lower (12-17 Hz), middle (18-23 Hz) and upper (24-29 Hz) beta oscillations (ß1, ß2 and ß3 respectively) over M1 (electrodes C3 and C4) during the motor preparation, motor execution, and post-movement phases. Based on previous studies (Meissner et al., 2018; Thibaut et al., 2017), MRBD is expected to become larger through practice with the go/nogo DSP task during the motor preparation and motor execution phase. Confirmation of this prediction would mean that experts can be identified by a larger MRBD over M1 during the motor preparation and motor execution phase compared to a novice learner. Thus, in that case, this MRBD can serve as a neuro marker of motor sequence expertise. This also confirms the inhibitory control of movements role that has been assigned to beta activity over M1 during preparation and execution. In the post-movement phase, it was expected that PMBR would become larger through practice with the go/nogo DSP task. An enlarged PMBR may therefore serve as a neuro marker for motor sequence expertise. This will support the earlier suggestions that PMBR updates the motor buffer for incoming sequences.

The second key goal is to test whether beta ERD/S over M1 during the different stages of sequential movement correlates with RT. It was predicted that a negative relationship between ERD/S

and RT during the motor preparation and execution phase would become evident. This means that shorter reaction times were expected to correlate with more desynchronization. This was predicted as it was expected that both MRBD and PMBR becomes larger through practice with the go/nogo DSP, while RT becomes shorter. Furthermore, it was expected that there would be a negative relationship observed between ERD/S and RT during the post-movement phase, indicating that longer response times were expected to correlate with less synchronization. If the hypotheses are confirmed this means that it would be possible to pinpoint when an individual gains expertise in all three periods of motor sequence learning and in the multiple beta frequency bands.

2. Methods

2.1 Participants

For the current study thirty participants (19 females, 21.5 + 2.4 years) were recruited from the SONA System test subject pool from the Behavioural, Management and Social Sciences Faculty of the University of Twente. Participants were eligible to participate once they met all inclusion criteria: participants had to be naturally right-handed; have no professional training in musical instruments or proficiency in gaming; no neurological, psychological, or psychiatric disorders; no depression or anxiety disorders; no sleep problems; no substance addictions; no diagnosed cognitive impairments (e.g., mild attentional disorders); and no physical injuries or impairments. All participants gave their written informed consent. and the study was approved by the BMS Ethics Committee/Domain Humanities and Social Sciences of the University of Twente (no. BCE200776). Participants received SONA credits to compensate for the time they spent on their participation in the study. On average the study took 3.5 hours to complete including the application and removal of EEG and EOG equipment. One participant was excluded from data analysis as all motor preparation and motor execution epochs were dropped. Two other participants were excluded from data analysis as they shared the same data files. Lastly, one participant was excluded as the data file for Block 5 was corrupted.

2.2 Stimuli and task

Participants carried out the go/nogo DSP task as per de Kleine and Van der Lubbe (2011). In each trial, a fixation cross was presented along with four horizontally aligned squares of 30 x 25mm on either side of the fixation cross. All eight stimulus squares corresponded to the alignment of the eight response keys. In the default screen, the fixation cross and squares were drawn with the same silver colour. The screen was filled with a black background. A trial began with the trial onset which is the presentation of the default screen that lasted for 1000 ms. The period after the trial onset is referred to as the Cue period one square placeholder was filled with yellow for 750 ms. Then, the next square was filled with yellow for 750 ms and this continued until six placeholders were filled. The next period is referred to as the

Preparation interval. During this period, the default screen was shown for another 1500 ms. After the Preparation interval, the fixation cross would turn either blue or red - which was the go/nogo signal, respectively. When it was a go-trial, participants were required to replicate the sequence using either their left or right fingers, depending on which side of the fixation cross the sequence was presented. In the case of 'nogo', the participants were required to do nothing. For a visualization of the above-described periods see Figure 2.

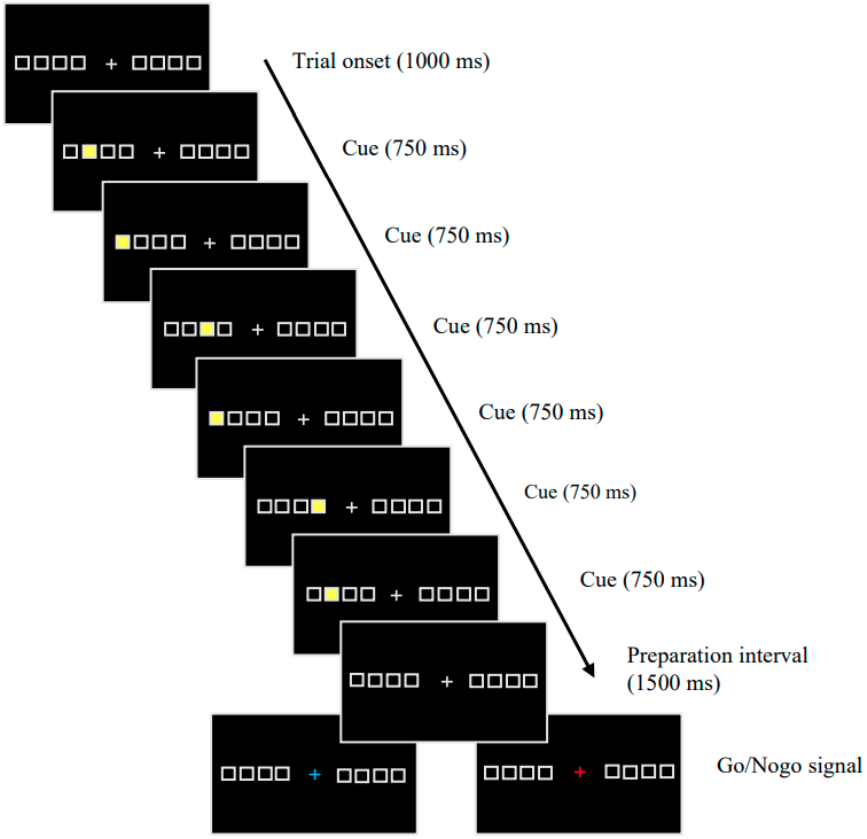


Figure 2. Six-key sequence example from trial onset to the go/nogo signal by de Kleine (2009).

All participants completed five blocks of 52 trials. Four (8%) out of the 52 trials were nogo-trials and 48 (92%) were go-trials. Additional instructions that participants received were to keep their eyes fixated on the fixation cross from the offset of the last cue until the last key press. Participants were also requested to minimize blinking during responding. Feedback was given after each response sequence. Whenever the participants responded with the correct sequence, a black screen appeared with the silver-coloured words 'Good!'. Whenever the participants responded with an incorrect sequence, they were presented with feedback showcasing which keys were incorrectly pressed. For example, if a participant incorrectly pressed the first key, then they would be presented with the words 'Key 1 was wrong'. If the participants pressed a key before the 'go' or 'nogo' signal was given, the feedback would be 'Too early!'.

After 24 trials participants were given a break of 20s. At the end of every block, the total number of errors and the average response RT were given on the end screen. 3

Each participant trained four different six-key sequences presented randomly over 48 go-trials per block, with a total of 5 blocks. There was a total of 960 sequences performed across both hands over 5 blocks. Each sequence was repeated 12 times per block. The four six-key sequences were divided into two sequences for the left hand and two mirrored sequences for the right hand. For the left hand the A, S, D and F keys were used. For the right hand, the keys J, K, L and ; were used. Counterbalancing for each keypress across positions was applied for all participants to avoid finger-specific effects on RT. For example, a left-hand sequence 'ADFSDA' was created. To create the second left-hand sequence each key of the first left-hand sequence had to be moved up one key to the right, resulting in the sequence 'SADAFS'. To create the right-hand sequences mirrored versions of the left-hand sequences were created, resulting in sequence ';KJLK;' as the first right-hand sequence and 'L;K;JL' as the second right-hand sequence. These sequences were identified with an ID number. The first participant was given the sequences from ID 1, the second from ID 2 and so on. In total eight ID numbers were created. This ensured that each participant practiced different sequences from the participant that came before or after them. For a full version of the counterbalancing please see Appendix B.

2.3 Procedure

Participants were requested to wash and dry their hair either the night before or on the day of the experiment. Participants were invited into the 'RecogNice' lab of the University of Twente. At the beginning of the experiment, participants were provided with an information sheet, an informed consent sheet, an EEG questionnaire, and a handedness questionnaire to assess their hand preferences. Once consent was given and signed, the participants were asked if their hair was washed and dry. Next, the participants were equipped with the EEG cap. To start the go/nogo DSP task, the assigned ID was selected, and the participant and block number were entered on the computer. Then, the participant's task was explained. Additionally, participants were provided with written instructions on the computer screen. Once the instructions were clear to the participants, the go/nogo DSP task started. Between every block, participants had a one-minute break.

2.4 Apparatus

During the experiment two personal computers (Optiplex 9020) running on Windows 7 with Dell UltraSharp U2518D monitors were used. One PC ran the experiment, and its monitor was placed in front of the participants at a viewing distance of about 45cm. A QWERTY keyboard was used for the participants to respond to the stimuli. The other PC was used to make EEG recordings of each block. The stimuli presented in the go/nogo DSP task, and the response registration was controlled by E-Prime (Ver.

2.0.10.356). EEG recordings were made using Brain Vision Recorder (Ver. 1.21.0403.). EEG and electro-oculogram (EOG) were amplified with an ActiChamp amplifier (32 channels) and four EEG channels were used as EOG channels. Therefore, EEG was recorded from 26 EASYCAP ActiCAP electrodes. The applied montage followed an adjusted 10/20 system. A bipolar reference was used with TP8 as the anode and TP7 as the cathode. EOG was recorded both horizontally from the outer canthi of both eyes and vertically from above and below the left eye. The electrode impedance was kept below 5kΩ. The sampling rate for the EEG and EOG data was 500Hz. Brain activity was band-pass filtered from 0.1 to 39 Hz during data analysis.

3. Data analysis

The RStudio environment (Ver. 1.4.1717) and MNE-Python package (Ver. 0.22.0.) were used for data analysis. With the lme4 package (Ver. 4.1.1), we applied linear mixed-effects models (MEMs) to account for subject-level differences in the behavioural and EEG data. There was a distinction made between 3 types of MEMs: two RT models, nine models for EEG measures and six models for the interaction between EEG measures and RT. This will be described in the following sections. To compute the significance level of effects for each interaction within each model, type II Wald chi-square tests were carried out. This was done to compare observed results to expected results to determine whether differences are observed due to chance or due to a relationship between variables. Post-hoc Tukey tests were performed on each model to determine the cause of the effect. Data analysis was performed on the data of 26 participants.

3.1 RT

Response time (RT) here was defined as the time between the go-signal to the point where the last key of the sequence was pushed down (sequence-level response time). Average Trial RT was calculated as the mean sequence RT of all participants per trial. Key press RT was calculated as the mean RT of pressing a key within a sequence. For the analysis of RT two models were created. As a distinction was made in ERD/S between left and right hands, the first RT model was used to assess how Average trial RT changes during the go/nogo DSP task between hands. In this model, the outcome variable was Average Trial RT with two predictor variables: Block (1 to 5) and Hand (left vs. right). The second model aimed to assess the concatenation effect and RT changes for each position across blocks. In this model, the outcome variable was Key Press RT with two predictor variables: Block (1 to 5) and Key-position (1 to 6).

3.2 EEG

3.2.1 Preprocessing and extraction of power components

Two Python scripts were written for the processing of EEG data (see MSL Script 1 and MSL Script 2 at https://github.com/DaphneTitsing/Sequence-learning for the full script or Appendix C for a preview of the script). Using the MNE-Python package (Ver .0.22.0.), the EEG data files for block 1 and block 5 of each participant were preprocessed one by one. The preprocessing stages first involved the selection of the reference electrodes on the raw data. A bipolar reference was used with TP8 as the anode and TP7 as the cathode. A lower bandpass filter of 0.1 Hz and an upper bandpass filter of 39 Hz was set. The next step involved a visual inspection and performing an independent component analysis (ICA) aimed to remove unwanted artefacts. The independent components that were extracted were visually inspected one by one. Components that had captured ocular movement were removed. 52 components were extracted from Block 1 and Block 5 for each participant. On average 2.7 components (9.8 %, *SD* = 1.5) were removed from Block 1 and 2.6 components were removed from Block 5 (9.6 %, *SD* = 0.9). Finally, the ICA solution was applied to the raw data. The 'ica.find_bads_eog' command was applied to automatically find ICs that match the EOG signals.

After the ICA correction, three types of epochs were selected, related to motor preparation, motor execution, and a post-movement phase. Per phase, epochs were divided into Blocks (1 and 5) and Left and Right hand (see Appendix A. for an overview of dropped epochs). For the motor preparation phase, the epochs had a length of -1100 ms to 0 ms from the go-signal. A baseline correction was applied at -6500 to -5500 ms from the go-signal (corresponding with 1000ms before the onset of the first stimulus). Motor execution epochs had a length of 0 ms to +3000 ms from the go-signal. For motor execution epochs, the same baseline correction was applied as for the motor preparation epochs. The post-movement epochs had a length of -600 to +2500 ms relative to the last button press in each trial. Here the baseline correction that had been applied was -4000 to -3000 ms from the last button pressed in each trial (corresponding with 1000ms before the go signal). For a visual representation of all epochs and baselines please see Figure 3 on the next page.

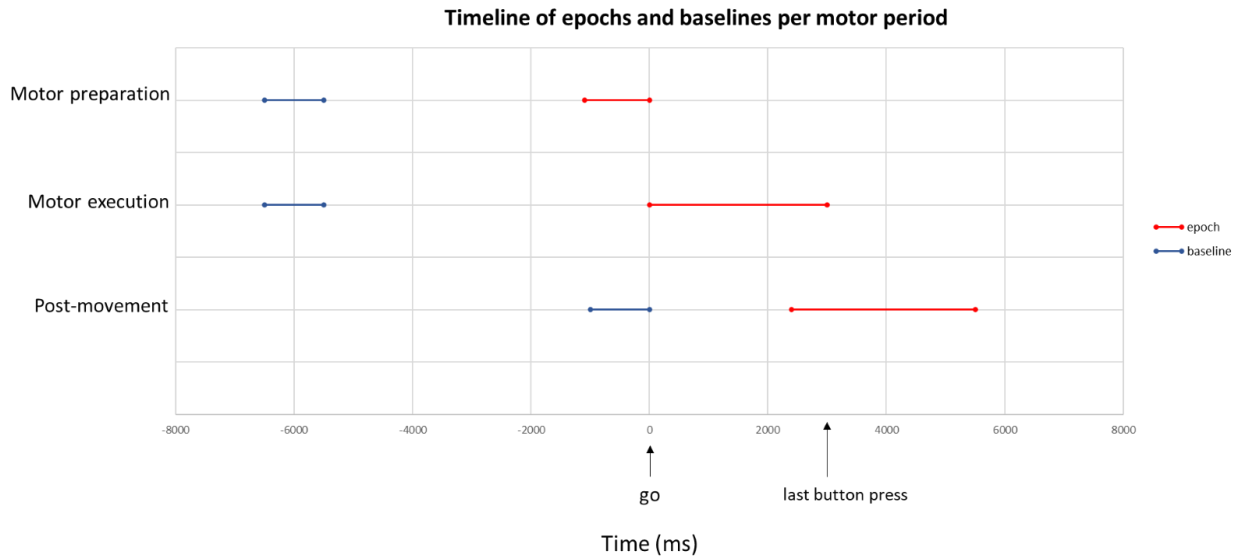


Figure 3. Timeline of epochs and baselines for each motor phase.

Finally, to perform the time-frequency analysis average Morlet wavelets were created for the C3 and C4 electrodes for the left and right hand of each motor phase. Morlet Wavelets were made of the entire beta range (12 to 29 Hz). Three cycles per frequency were used to create the Morlet Wavelets. The Morlet wavelets were expressed in Power (µV). These Power outputs were used to create a data frame of ERD/S values for analysis in R studio.

3.2.2 ERD/S

The ERD/S for each motor phase were analyzed in R studio environment Ver. 4.1.0. in several steps. First, two separate data sets for each motor phase and Block were made. Each data frame included power data computed from the Morlet wavelets. This power data included the baseline period and the time window of interest for the epochs (-1000ms to 0ms from the 'go'-signal for motor preparation, 0ms to 3000ms for motor execution and -500 to 2500ms for post-movement). Next, for each motor phase, the mean power of the baseline period was computed. The mean power was calculated for both C3 and C4 by calculating the average power over 100 ms time windows. This data was then calculated as a percentage of the baseline power to eventually create the ERD/S data in percentages. In the next step, three MEMs were created to analyse each motor phase. For each motor phase, separate models were made for β1, β2 and β3, resulting in models for frequency and phases. Specifically, across these models, the outcome variable was ERD/S (%) and the predictor variables were Block (1 vs. 5), Channel (C3 vs. C4), Hand (Left vs. Right), and Time window (ms). Block is considered the main predictor as the aim was to track the differences between pre- and post-training.

3.2.3 ERD/S across RT

Whenever an interaction of Time x Block x Hand on ERD/S was found, posthoc Tukey tests were carried out. The aim of this was to identify the time windows in which there was a significant change in ERD/S between Blocks and hands. Based on ERD/S results six MEMs for the last phase of analysis were identified: a left- and right-hand model for ß2 in the motor preparation phase, a left- and right-hand model for ß1 in the post-movement phase; and a left-hand model for ß2 and a left-hand model for ß3 in the post-movement phase. ERD/S across RT MEMs were generated for left and right hand per beta-band per motor phase (excluding ß2 and ß3 in the post-movement phase and ß3 in the motor execution phase). These models were aimed to be used as an assessment of whether ERD/S can be predictive of RT.

4. Results

4.1 Behavioural parameters

4.1.1 Hand effect

The MEM analysis on the third model on Average Trial RT performance showed no significant interaction of Hand on Average Trial RT ($p > .05$). This demonstrates that no differences between left (non-dominant hand) or right hand (dominant hand) sequences on Average Trial RT performance.

4.1.2 Concatenation

The MEM analysis of the fourth model on Key Press RT performance showed a significant effect of Position, $\chi^2(5, N= 26) = 1190.5$, $p < .001$, and Block ,$\chi^2(4, N= 26) = 908.0$, $p < .001$, on Key Press RT. Besides, the MEM analysis showed a significant Position x Block interaction on Key Press RT, $\chi^2(20, N= 26) = 53.7$, $p < .001$. The post-hoc Tukey tests within Key position showed that for Key position 1 to 6, Block 1 had longer Key Press RTs compared to Blocks 2 to 5 ($p < .0001$) (see Figure 4 on the next page).

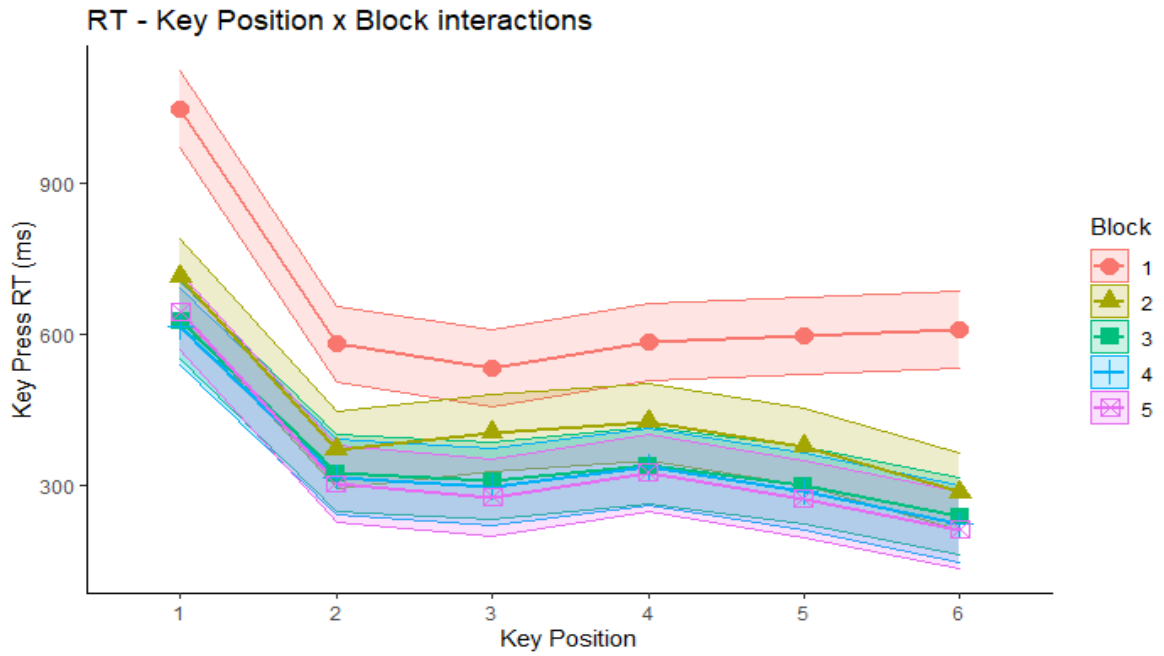


Figure 4. Key press RT as a function of Block and Trial accuracy with 95% confidence intervals. Correct trials showed faster RTs than incorrect trials.

For all Blocks, key position 1 had longer RTs compared to the remainder of the key positions ($p < .0001$). For Block 2 ($p < 0.001$), 3($p < 0.05$), 4 ($p < 0.01$) and 5 ($p < 0.01$) key position 4 had significantly longer Key Press RTs compared to key position 6 (see Figure 5).

4.2 ERD/S

4.2.1 Motor preparation

4.2.1.1 Motor preparation - ß1 (12-17 Hz)

For ß1 in the motor preparation period the model analysis revealed significant main effect of Time, $\chi^2(10, N= 26) = 32.4$, $p < .001$, Channel, $\chi^2(1, N= 26) = 24.1$, $p < .001$, and Block, $\chi^2(1, N= 26) = 49.8$, $p < .001$, on ERD/S. No further effects were found.

4.2.1.2 Motor preparation – ß2 (18-23 Hz)

The ß2 motor preparation model revealed a significant interaction of Time x Block x Hand on ERD/S, $\chi^2(1, N= 26) = 205.9$, $p < .05$. This demonstrated a significant difference of the modelled slope between left- and right-hand sequences in ERD/S across blocks and time windows. For remaining effects see Table 1.

Table 1

Model analysis of ß2 during motor preparation

Effect	**Df**	X^2	**Significance**
Time	10	222.8	*
Channel	1	48.0	*
Block	1	241.1	***
Hand	1	45.7	*
Time x Block	10	162.4	NS
Channel x Block	1	0.0	NS
Block x Hand	1	0.6	NS
Time x Channel x Block	10	19.7	NS
Time x Block x Hand	10	205.9	*
Channel x Block x Hand	1	0.3	NS
Time x Channel x Block x Hand	10	23.0	NS

Note. * $p < .05$, ** $p < .01$, *** $p < .001$, NS = Not Significant

The post-hoc Tukey tests revealed that for the left-hand ERD was significantly larger over the C3 and C4 electrodes in Block 5 compared to Block 1 from -1100 to -1000 ms ($p < .05$), -1000 to -900 ms ($p < .05$) and -900 to -800 ms ($p < .001$) from the go-signal (see Figure 6). For the right-hand ERD was significantly larger from -1100 to -1000 ms ($p < .05$) and from -400 to -300 ms ($p < .05$) and smaller from -800 to -700 ms ($p < .01$) from the go-signal (see Figure 5 on the next page.)

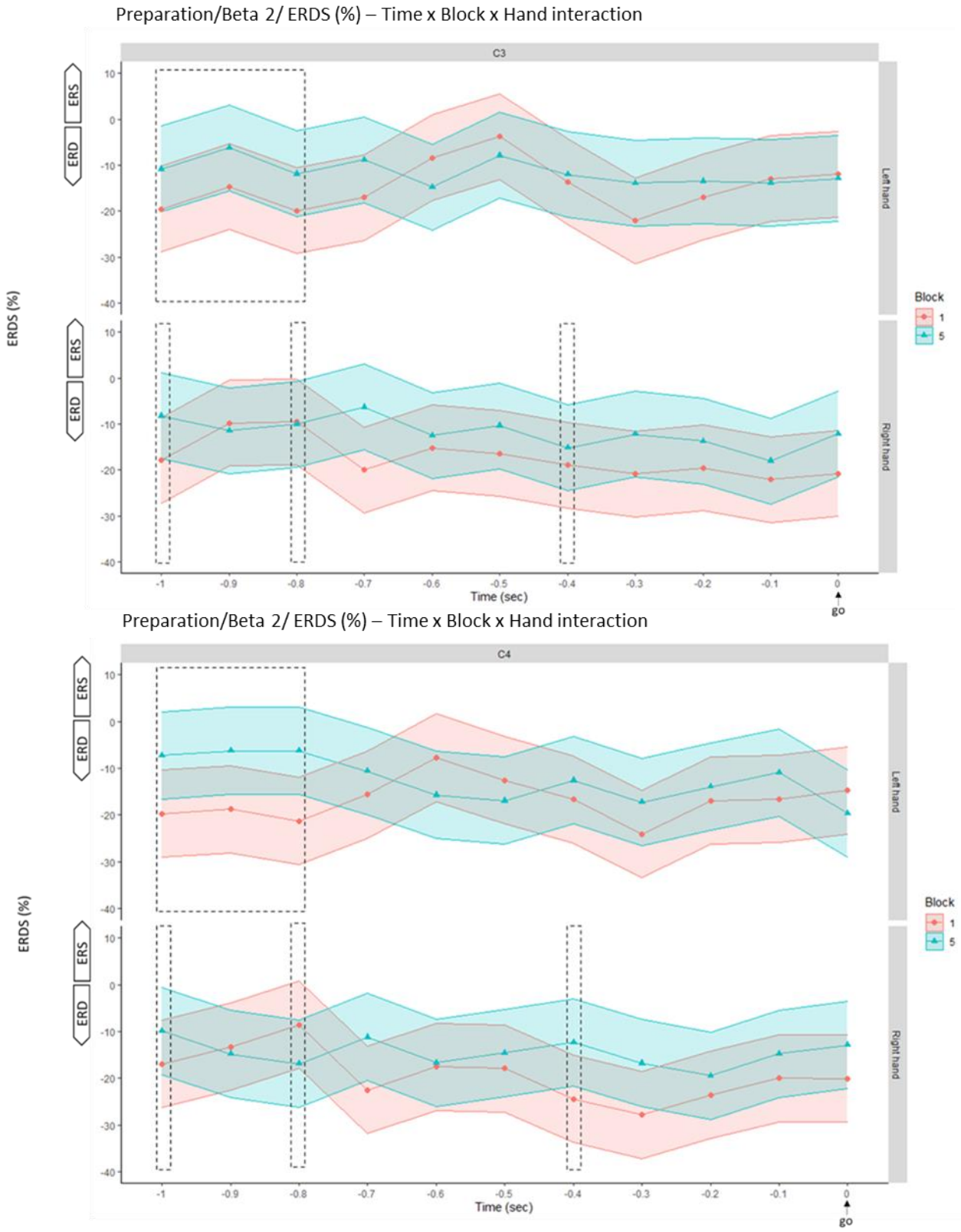
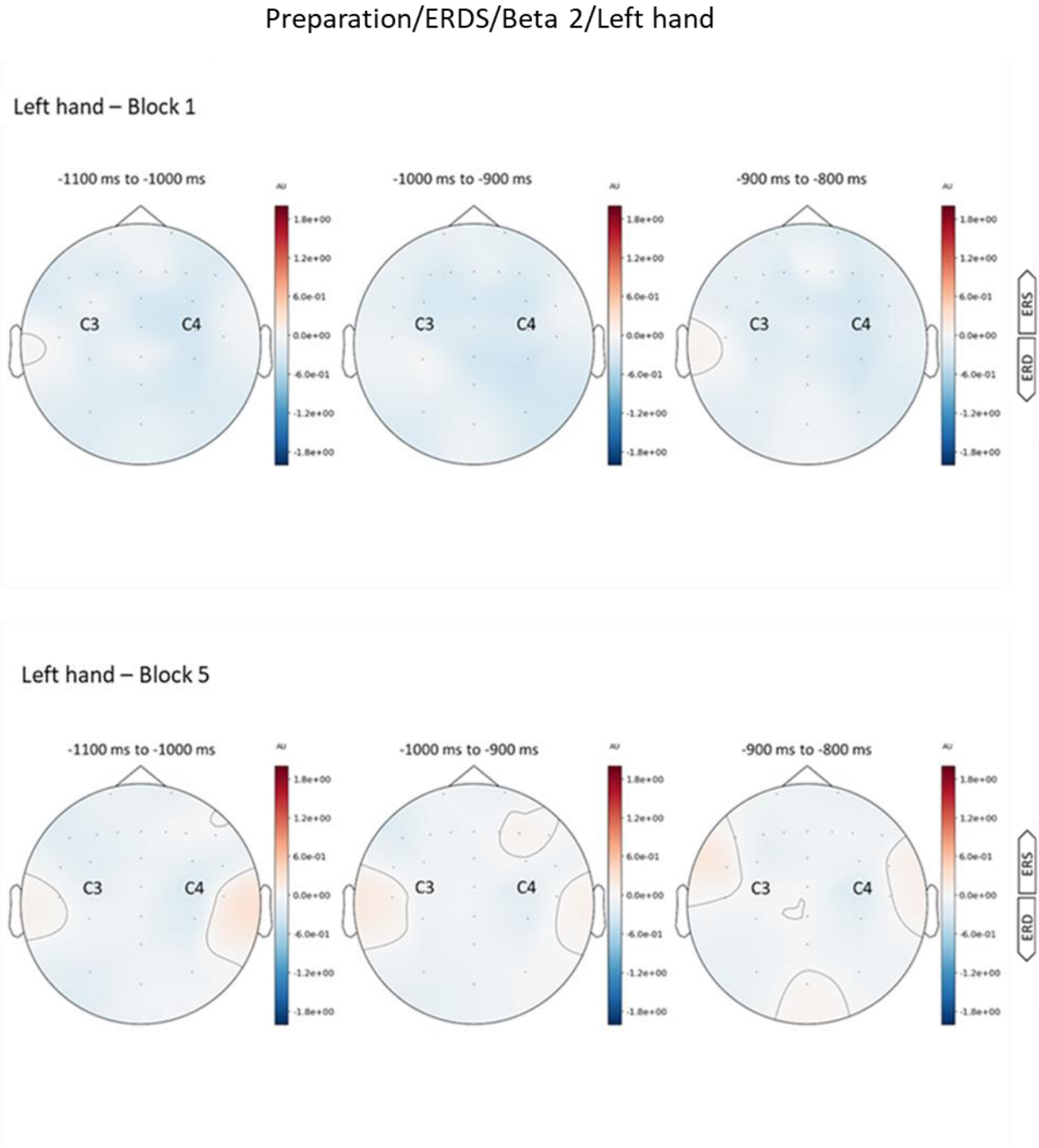


Figure 5. ß2 ERD/S (%) during motor preparation as a function of Time, Channel, Block and Hand for left and right-hand over C3 (top of the figure) and C4 (bottom of the figure) with 95 % confidence intervals. Time is presented in seconds for visual purposes. The -1 seconds epoch relates to a time window of 100 ms (from -1100 to -1000 ms) and so forth.

The topographical maps in Figure 6 reveal that left-hand ERD (ß2) was bilateral in Block 1 and Block 5 for all significant time windows. Right hand MRBD (ß2) was bilateral in Block 1. Furthermore, Right-hand MRBD (ß2) was ipsilateral from -1100 to -1000 ms and from -400 to -300 ms and contralateral from -800 to -700 ms.



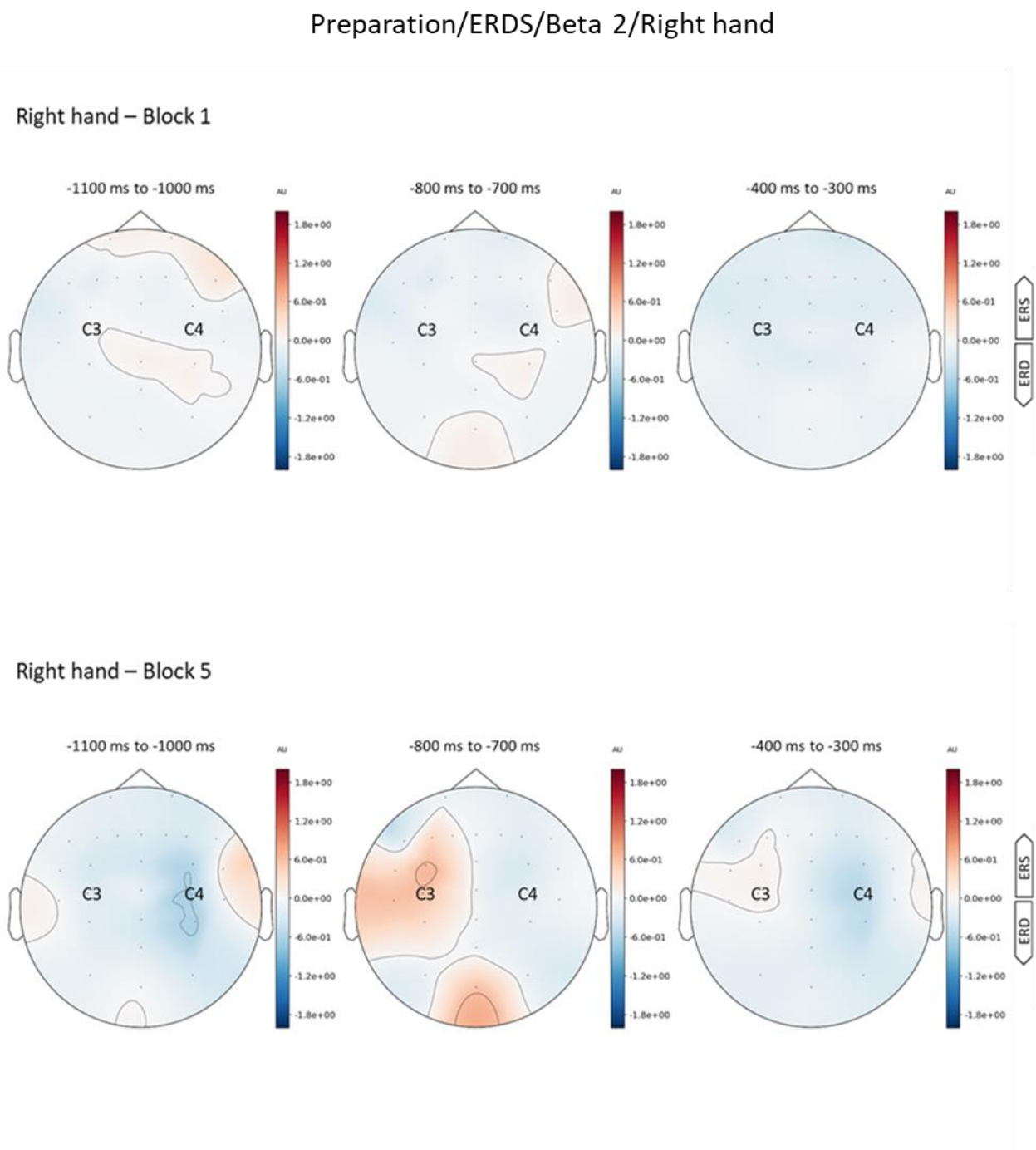


Figure 6. Topography maps of ß2 ERD/S during motor preparation. The topo maps reveal ERD/S for the left and right-hand sequence over the significant time windows between Block 1 and Block 5. *Note*. The left-hand desynchronization is bilateral in Block 1 and Block 5. The right-hand desynchronization is bilateral in Block 1. In Block 5 desynchronization is ipsilateral to the right (dominant) hand from -1100 to -1000 ms and from -400 to -300 ms and contralateral from -800 to -700 ms. In that Block 5

desynchronization is larger from -1100 to -1000 ms and from -400 to -300 ms compared to Block 1 and smaller from -800 to -700 ms compared to Block 1.

4.2.1.3 Motor preparation – ß3 (24-29 Hz)

The ß3 motor preparation period model revealed a significant main effects of Block, $\chi^2(1, N= 26) = 9.2$, $p < .01$, and Hand, $\chi^2(1, N= 26) = 32.2$, $p < .001$, on ERD/S. No further effects or interactions were found.

4.2.2. Motor execution

4.2.2.1 Motor execution - ß1 (12-17 Hz)

For the lower beta-band in the motor execution period the model analysis revealed significant main effects of Block, $\chi^2(1, N= 26) = 10.5$, $p < .01$, Time, $\chi^2(30, N= 26) = 105.8$, $p < .001$, , and Hand, $\chi^2(1, N= 26) = 12.8$, $p < .001$, on ERD/S. No further effects or interactions were found.

4.2.2.2 *Motor execution – ß2 (18-23 Hz)*

For the middle beta-band in the motor execution period the model analysis revealed a significant main effect of Block, $\chi^2(1, N= 26) = 16.1$, $p < .001$, Time, $\chi^2(30, N= 26) = 45.8$, $p < .05$, , and Hand,$\chi^2(1, N= 26) = 5.6$, $p < .05$, on ERD/S. The model analysis also showed a significant interaction of Time x Hand, $\chi^2(30, N= 26) = 44.3$, $p < .05$. This means that there was a significant difference between left- and right-hand sequences in middle beta-band ERD/S across time windows during motor execution. No further effects or interactions were found.

4.2.2.3 *Motor execution – ß3 (24-29 Hz)*

For the upper beta-band in the motor execution period the model analysis revealed significant main effects of Block, $\chi^2(1, N= 26) = 12.6$, $p < .001$, and Hand, $\chi^2(1, N= 26) = 6.1$, $p < .05$, on ERD/S. No further effects or interactions were found.

4.2.3. *Post-movement*

4.2.3.1 *Post-movement - ß1 (12-17 Hz)*

The model analysis on ß1 in the post-movement phase revealed a three-way interaction of Time x Block x Hand, $\chi^2(1, N= 26) = 53.3$, $p < .01$.This demonstrates a significant difference in ß1 ERD/S of the modelled slope between left and right hand and between Block 1 and Block 5 across time windows. For a summary of other effects and interactions see Table 2.

Table 2.

Model analysis of ß1 during post-movement

Effect	df	X^2	Significance
Time	30	152.0	***
Channel	1	11.2	***
Block	1	7.2	**
Hand	1	0.7	*
Time x Block	30	58.4	**
Channel x Block	1	0.1	NS
Block x Hand	1	10.6	**
Time x Channel x Block	30	7.7	NS
Time x Block x Hand	30	53.3	**
Channel x Block x Hand	1	0.15	NS
Time x Channel x Block x Hand	30	8.9	NS

Note. * $p < .05$, ** $p < .01$, *** $p < .001$, NS = Not Significant

The posthoc Tukey tests showed significantly larger ß1 PMBR over the C3 and C4 electrode for the left-hand sequences in block 5 compared to Block 1 from 700 to 800 ms ($p < .05$), 800 to 900 ms ($p < .05$), 1100 to 1200 ms ($p < .0001$) and 1200 to 1300 ms ($p < .0001$) from the last response (see Figure 7). For the right-hand sequences, there was a significantly larger ß1 PMBR observed at the C3 and C4 in block 5 compared to Block 1 from 2000 to 2100 ms ($p < .0001$) and 2100 to 2200 ms ($p < .05$) from the last response (see Figure 8). The topography maps in Figure 9 reveal that left-hand PMBR (ß1) was bilateral in Block 1 for all significant time windows. Left-hand PMBR (ß1) was bilateral in Block 5 from 700 to 800 ms and contralateral from 800 to 900 ms, 1100 to 1200 ms and 1200 to 1300 ms. Furthermore, right-hand PMBR (ß1) was bilateral for both significant time windows in Block 1 and ipsilateral for Block 5 from 800 to 1300 ms.

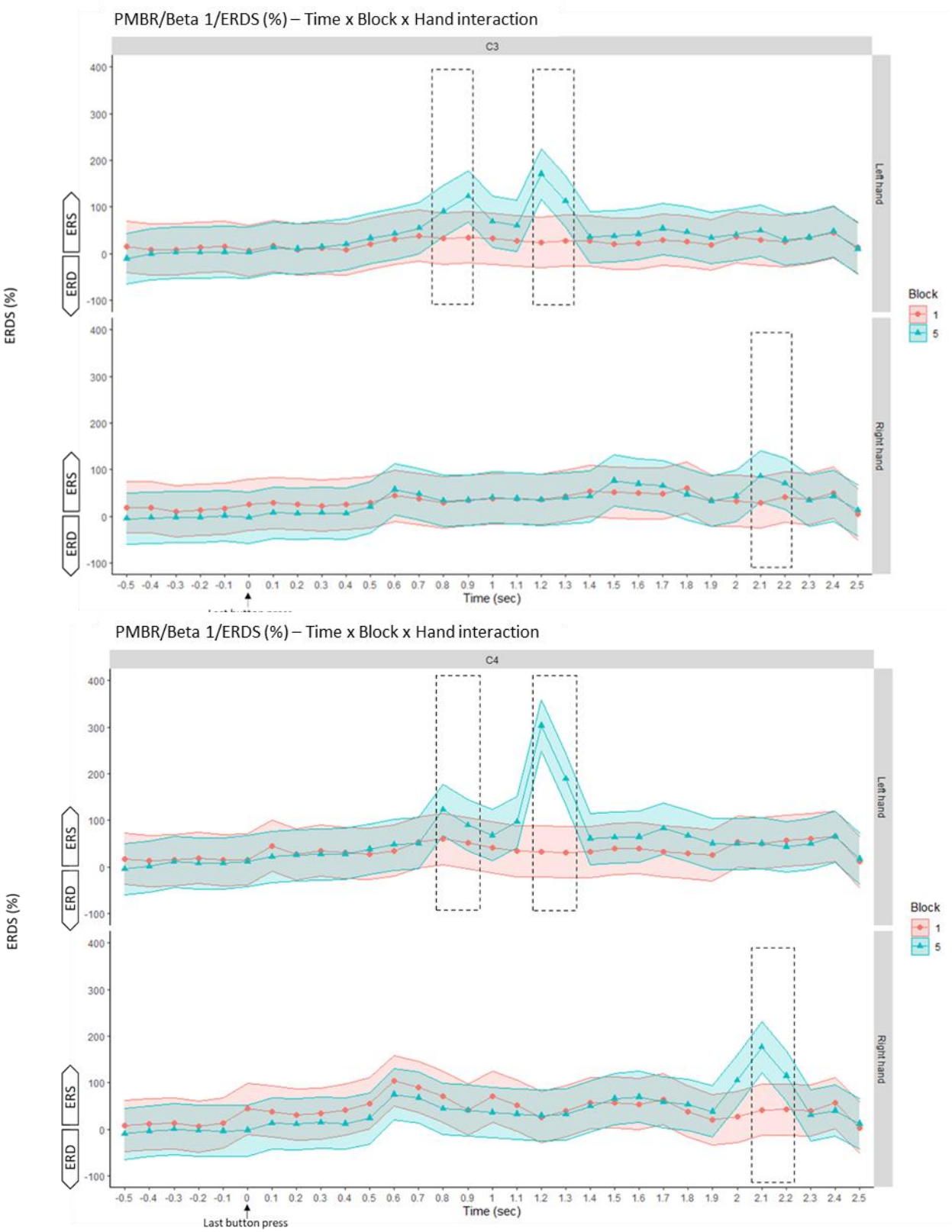


Figure 7. ß1 ERD/S (%) during post-movement as a function of Time, Channel, Block and Hand for left- and right-hand sequences over C3 (top of the figure) and C4 (bottom of the figure) with 95 % confidence

intervals. Time is presented in seconds for visual purposes. -0.5 seconds relates to a time window of 100 ms (from -600 to 500 ms) and so forth.

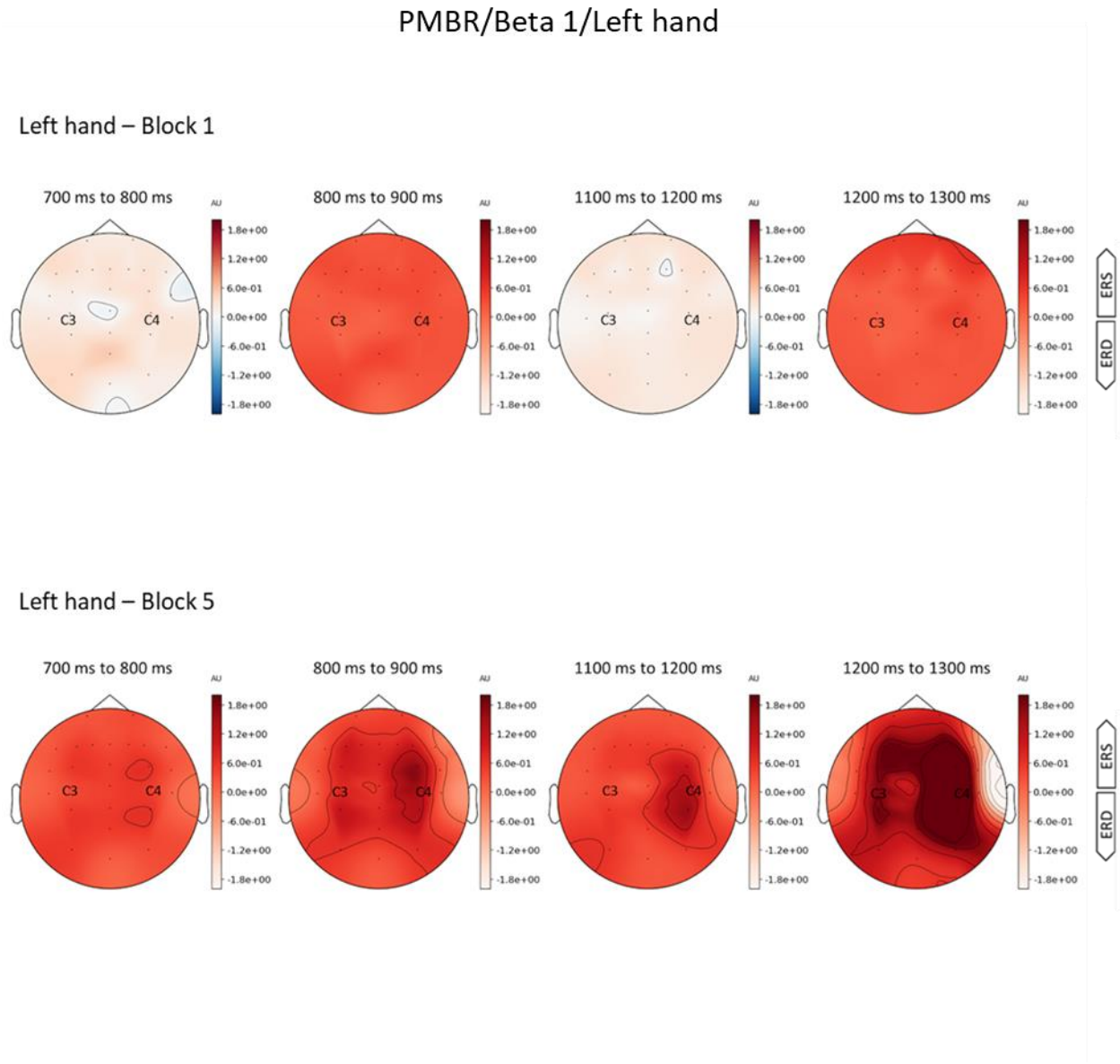


Figure 8. Topo maps of ß1 ERD/S during post-movement. The topo maps reveal ERD/S for the left hand over the significant time windows between Block 1 and Block 5. Notice that in Block 1 synchronization is bilateral and in contrast, Block 5 synchronization is contralateral to the left (non-dominant) hand performance from 800 to 1300 ms.

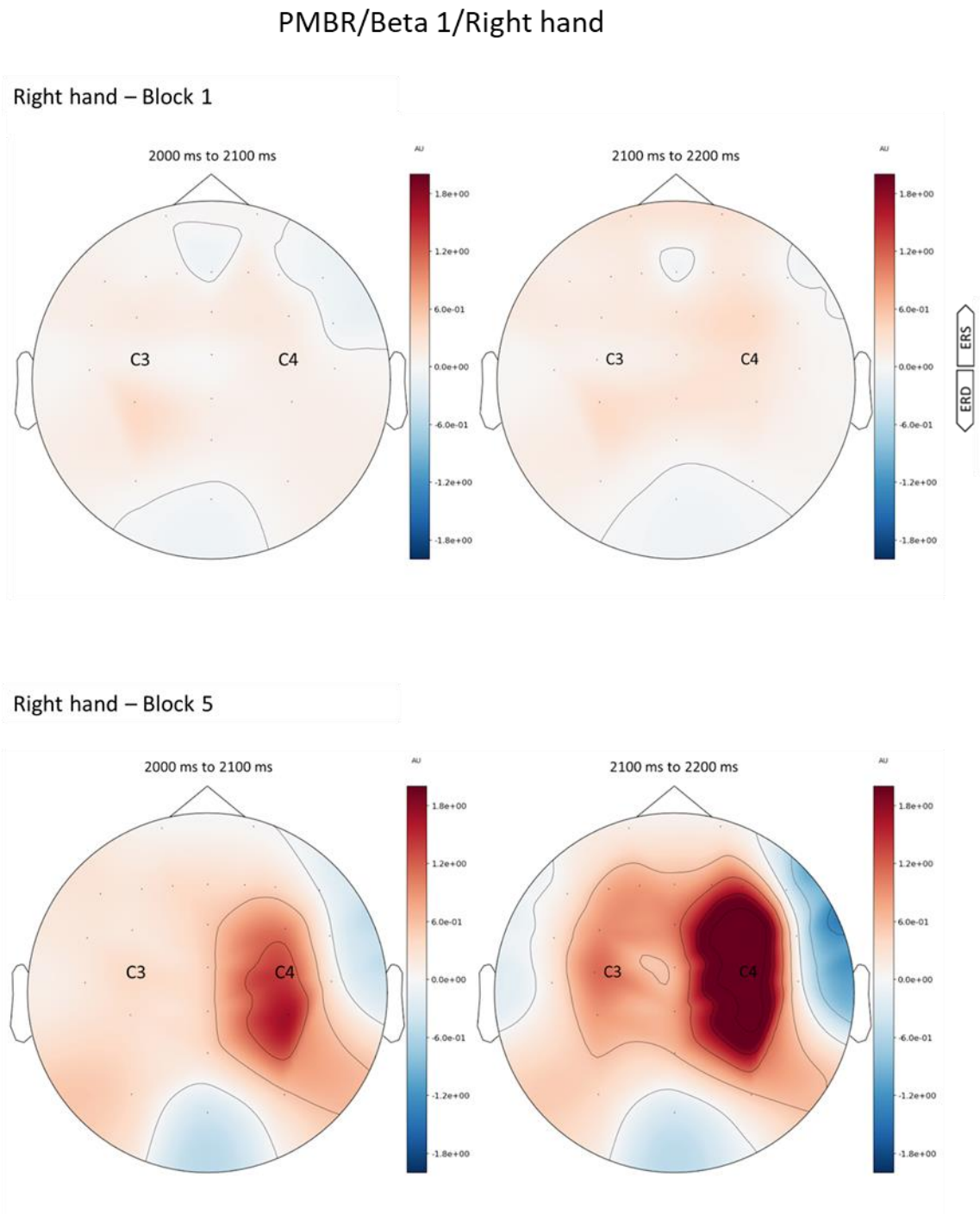


Figure 9. Topo maps of ß1 ERD/S during post-movement. The topo maps reveal ERD/S for the right hand over the significant time windows between Block 1 and Block 5. Notice that in Block 1 synchronization is bilateral and in Block 5 synchronization is l ipsilateral to the right (dominant) hand.

4.2.3.2 Post-movement – ß2 (18-23 Hz)

For ß2 in the post-movement period, the model analysis showed a three-way interaction of Time x Block x Hand, $\chi^2(1, N= 26) = 48.4$, $p < .05$. This means there was a significant difference predicted slope of ß2 ERD/S changes between left and right hand and between Block 1 and Block 5 across time windows. For other effects and interactions related to block changes see Table 3.

Table 3

Model analysis of ß2 during post-movement

Effect	**df**	X^2	**Significance**
Time	30	113.8	***
Channel	1	12.5	***
Block	1	3.7	NS
Hand	1	3.8	NS
Time x Block	30	57.1	**
Channel x Block	1	0.04	NS
Block x Hand	1	13.4	***
Time x Channel x Block	30	8.1	NS
Time x Block x Hand	30	48.4	*
Channel x Block x Hand	1	0.2	NS
Time x Channel x Block x Hand	30	9.4	NS

Note. * $p < .05$, ** $p < .01$, *** $p < .001$, NS = Not Significant

The posthoc Tukey tests showed a significantly larger ß2 PMBR over the C3 and C4 electrode for the left hand in Block 5 compared to Block 1 from 700 to 800 ms ($p < .05$), 1000 to 1100 ms ($p < .01$), 1100 to 1200 ms ($p < .001$) and 1200 ms to 1300 ms ($p < .001$) from the last response (see Figure 10). There were no significant differences between Block 1 and Block 5 for the right hand found. Figure 11 shows that PMBR was bilateral for the left hand in Block 1 and contralateral for Block 5.

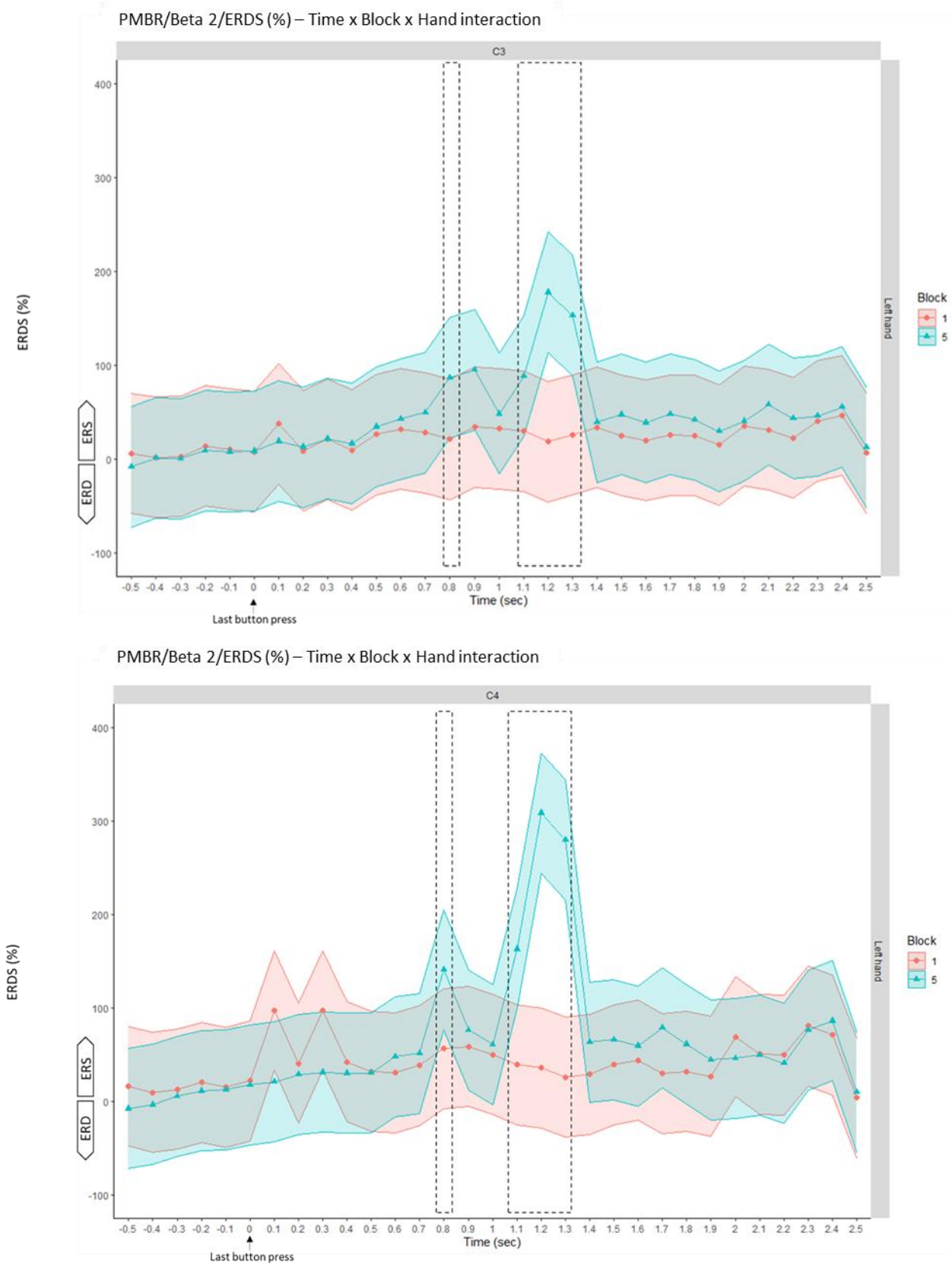


Figure 10. ß2 ERD/S (%) during post-movement as a function of Time, Channel, Block and Hand for left-hand sequences over C3 (top of the figure) and C4 (bottom of the figure) with 95 % confidence

intervals. Time is presented in seconds for visual purposes. For example, -0.5 seconds epoch relates to a time window of 100 ms (from -6000 to -500 ms) and so forth.

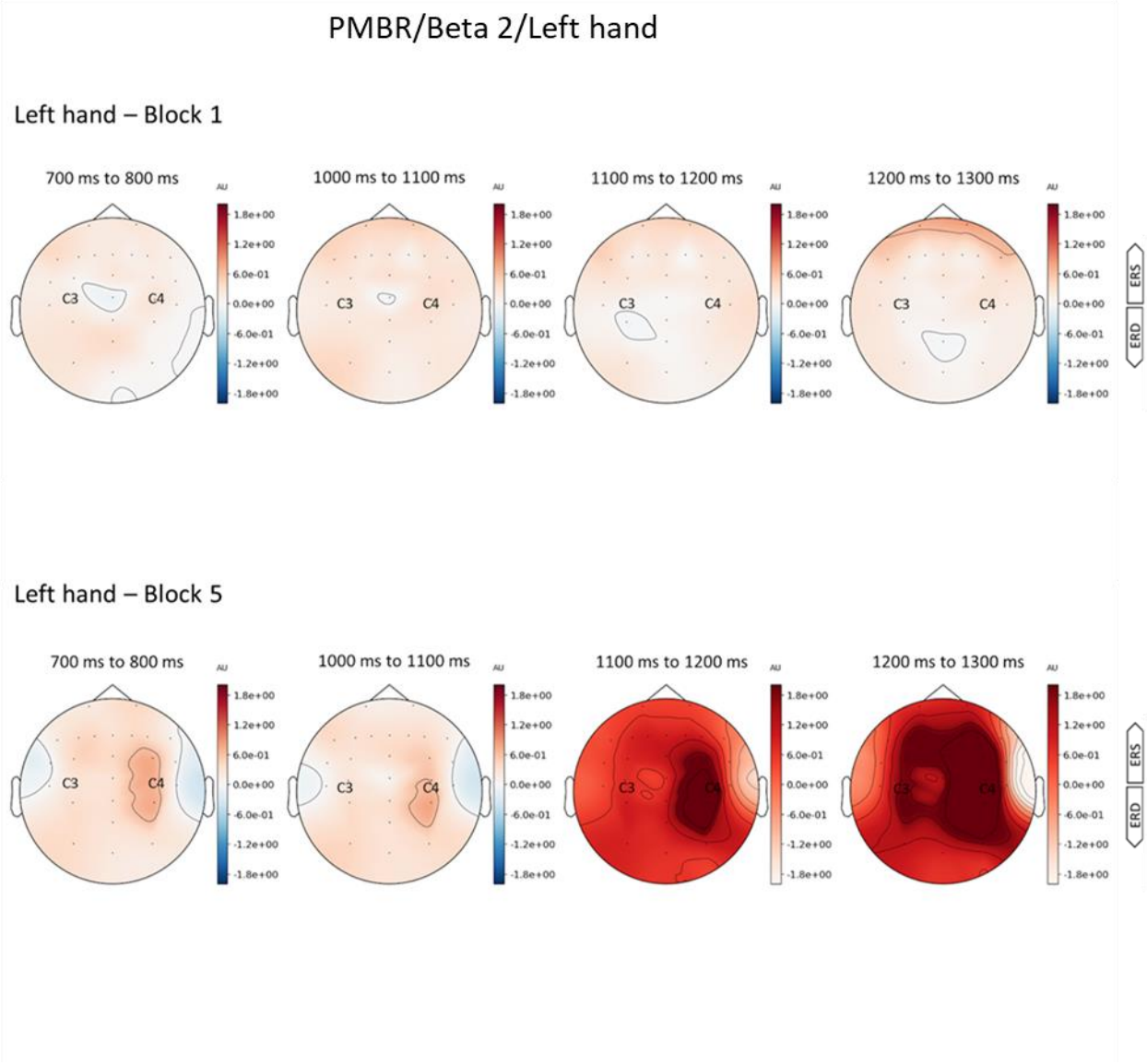


Figure 11. Topographical maps of ß2 ERD/S during post-movement. The topo maps reveal ERD/S for left-hand sequences over the significant time windows between Block 5 and Block 1. *Note.* In Block 1 synchronization is bilateral. In Block 5 synchronization is contralateral to the left (non-dominant).

4.2.3.3 Post-movement – ß3 (24-29 Hz)

For ß3 in the post-movement period, the model analysis showed a three-way interaction of Time x Block x Hand, $\chi^2(1, N= 26) = 44.9$, $p < .05$. This means that there was a significant difference in ß3 ERD/S between left and right-hand sequences and between Block 1 and Block 5 across time windows. For other effects and interactions see Table 4.

Table 4

Model analysis of ß3 during post-movement

Effect	df	X^2	Significance
Time	30	109.8	***
Channel	1	8.3	**
Block	1	1.4	NS
Hand	1	6.3	*
Time x Block	30	52.7	**
Channel x Block	1	0.3	NS
Block x Hand	1	6.2	*
Time x Channel x Block	30	10.4	NS
Time x Block x Hand	30	44.9	*
Channel x Block x Hand	1	0.0	NS
Time x Channel x Block x Hand	30	12.8	NS

Note.* $p < .05$, ** $p < .01$, *** $p < .001$, NS = Not Significant

The posthoc Tukey tests revealed a significantly larger ß3 MRBD over the C3 and C4 electrodes for the left hand in Block 5 compared to Block 1 from 900 to 1000 ms ($p < .01$). A significantly increased upper beta-band synchronization over the C3 and C4 electrodes was found for the left hand in Block 5 compared to Block 1 from 700 to 800 ms ($p < .05$), 800 to 900 ms ($p < .05$), 1000 to 1100 ms ($p < .01$), 1100 to 1200 ms ($p < .0001$) and 1200 to 1300 ms ($p < .0001$) from the last response (see Figure 12). It was revealed that there were no time windows with a significant difference in ß3 ERD/S between Block 1 and Block 5 for the right hand. Figure 13 shows that PMBR was bilateral for the left hand in Block 1 and contralateral for Block 5.

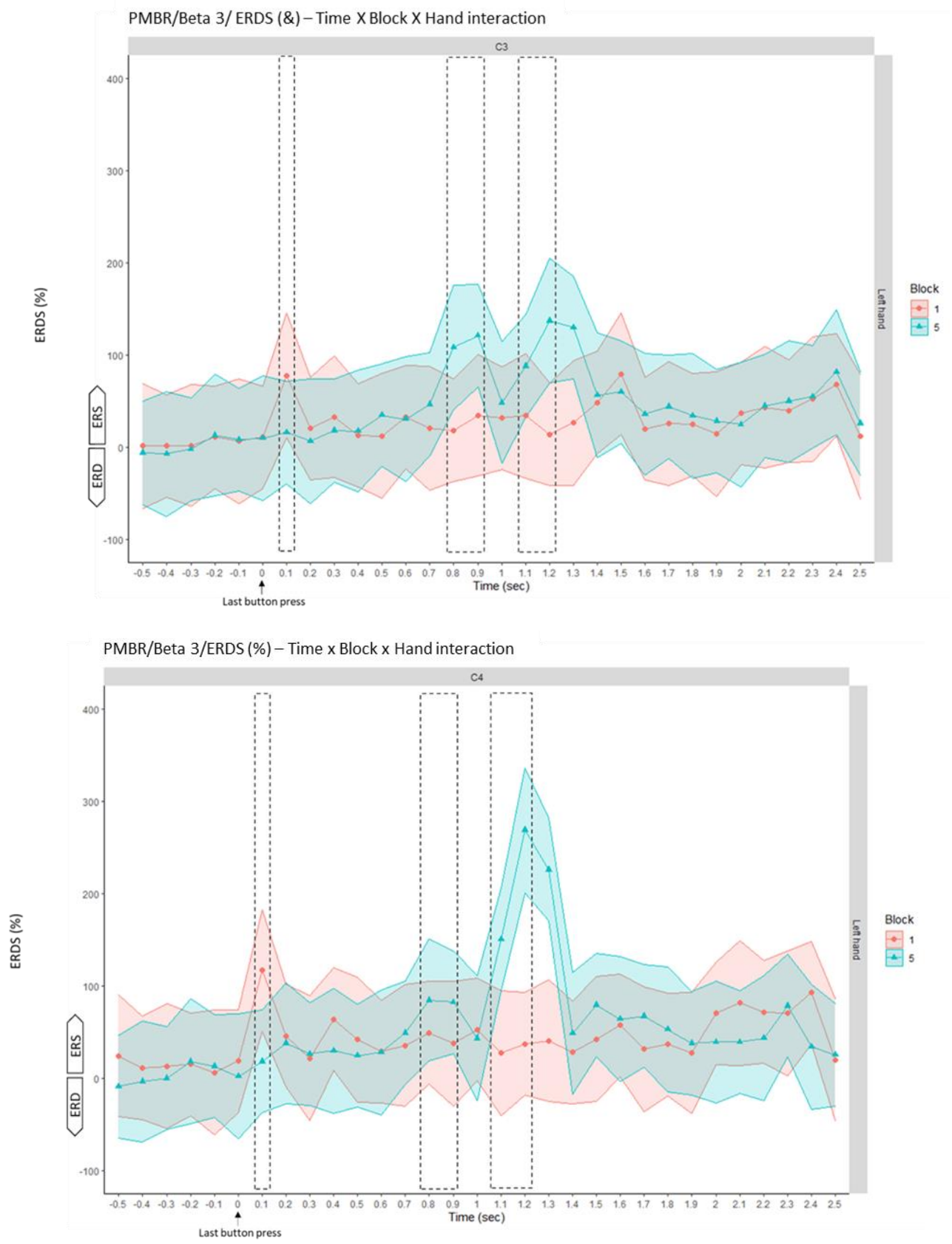


Figure 12. ß3 ERD/S (%) during post-movement as a function of Time, Channel, Block and Hand for left-hand sequences over C3 (top of the figure) and C4 (bottom of the figure) with 95 % confidence

intervals. Time is presented in seconds for visual purposes. -0.5 seconds relates to a time window of 100 ms (from -600 to -500 ms) and so forth.

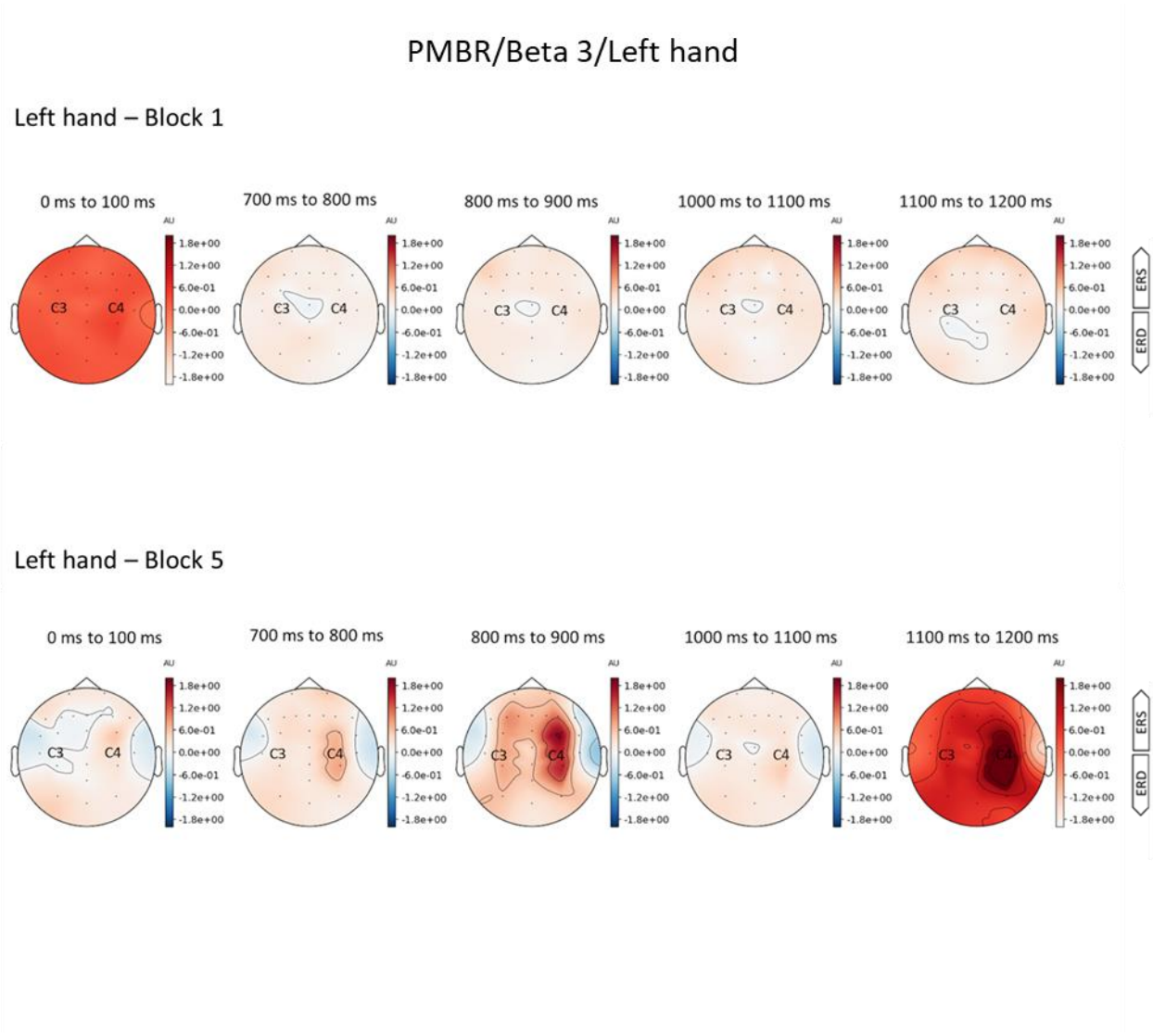


Figure 13. Topographical Maps of ß3 ERD/S during post-movement. The topo maps reveal ERD/S for left-hand sequence over the significant time windows between Block 5 and Block 1. In Block 1 synchronization is bilateral and in Block 5 synchronization is contralateral to the left (non-dominant).

4.3 Linear prediction of the relationship between ERD/S and Block RT

4.3.1 Motor preparation – ß2 (18-23 Hz)

For ß2 in the motor preparation period Block had a significant effect on ERD/S of the left hand

sequences, ,$\chi^2(1, N= 26) = 5.9$, $p < .05$, and ERD/S of the right-hand sequences in the lower beta-band, $\chi^2(1, N= 26) = 15.2$, $p < 0.001$. Besides, the model analysis revealed a significant interaction of Block x RT, $\chi^2(1, N= 26) = 7.4$, $p < .01$, on ß2 ERD/S of the left hand. Figure 14 shows that for Block 1 shorter RTs were associated with smaller MRBD in the motor preparation phase. For Block 5 shorter RTs were associated with larger ERD.

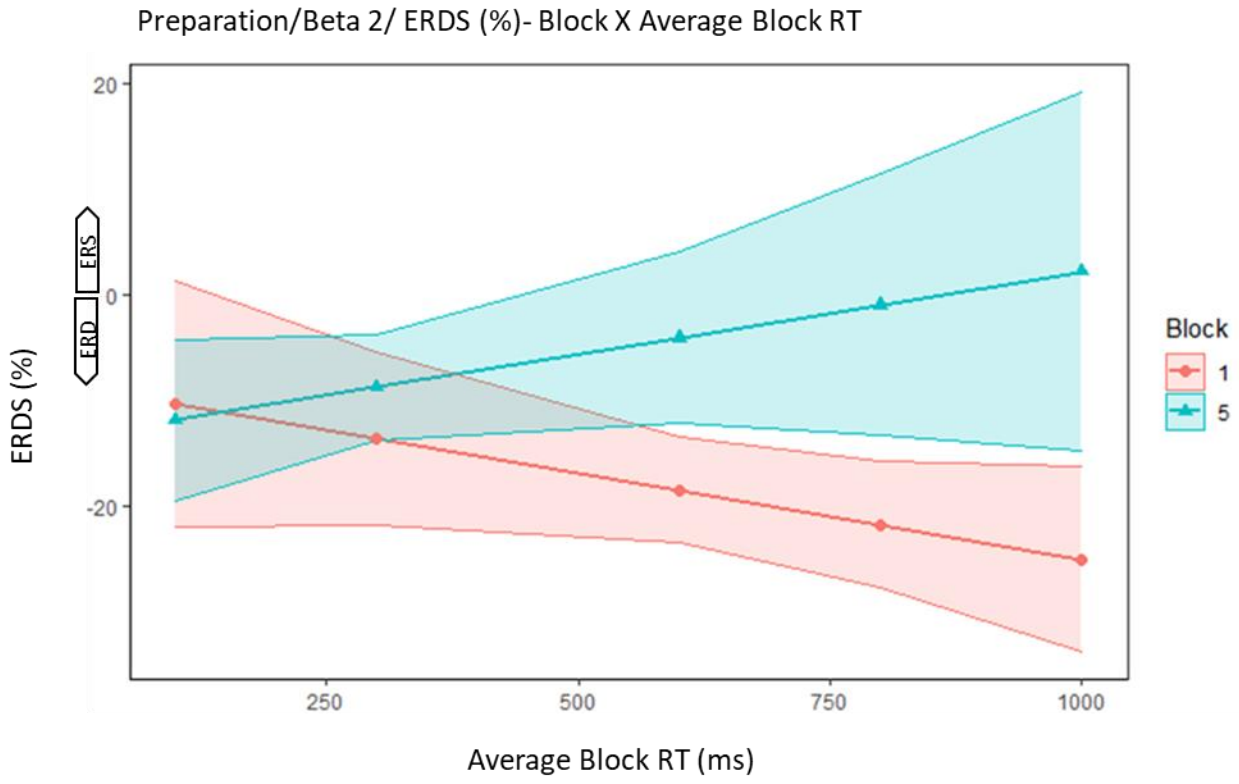


Figure 14. ß2 ERD/S as a function of Block and average block RT in the motor preparation phase. with a 95% confidence interval. A positive linear relationship can be observed between ERD/S and RT increases in Block 5 whilst a negative relationship is observed between ERD/S and Block 1. Shorter responses (< 250 ms) across both blocks were associated with similar activation levels.

4.3.2 Post-movement

4.3.2.1 Post-movement - ß1 (12-17 Hz)

For the lower beta-band in the post-movement period Block had a significant effect on ERD/S for the left hand sequences ,$\chi^2(1, N= 26) = 4.9$, $p < .05$ and right hand sequences ,$\chi^2(1, N= 26) = 5.7$, $p < .05$.

4.3.2.2 Post-movement – ß2 (18-23 Hz)

For the middle beta-band in the post-movement period only significant main Block effect on ERD/S for the left hand sequences $\chi^2(1, N= 26) = 6.5$, $p < .05$ were found.

4.3.2.3 Post-movement – ß3 (24-29 Hz)

For the upper beta-band in the post-movement period no significant effects or interactions were found for the left hand sequences.

5. Discussion

5.1 Main findings

Firstly, it was predicted that concatenation was evident during the go/nogo DSP task. The results showed that for Block 2 onwards key position 4 was associated with significantly longer Key Press RT compared to key position 6. The results were largely in line with de Kleine and Van der Lubbe (2011), and that concatenation was evident during the go/nogo DSP task except for the first Block. Besides, as beta ERD/S was divided into left- and right-hand beta ERD/S, it was tested whether there was a difference in RT between left (non-dominant hand) and right-hand (dominant hand) sequences. Contrary to the study by de Kleine and Van der Lubbe (2011), the results of this thesis revealed no differences between the left and right hand in RT.

The first key goal of this thesis was to clearly outline how motor sequence learning is reflected in ß1, ß2 and ß3 ERD/S over M1 across the different movement phases. It was predicted that through practice with the go/nogo DSP task, larger MRBD and larger ERD (in the motor preparation phase) would be found. Additionally, for the post-movement phase, a larger PMBR was expected. The results of the motor preparation phase were partly supported by the predictions. For both the dominant and non-dominant hand motor sequence learning was represented in larger ß2 ERD (but not ß1 and ß3 ERD) in the motor preparation phase. For the motor execution period, no differences were found between the modelled slopes of Block 1 and Block 5 in ß1, ß2 or ß3 throughout the experiment. This was not in line with the predictions. The results of the post-movement phase were largely in line with the expectation. Larger PMBR was observed in Block 5 compared to Block 1 (for the left and right-hand sequences in ß1 and the left-hand sequences in ß2 and ß3).

The second aim of this thesis was to investigate the relationship between RT performance and ERD/S to outline the changes that occur from learning. It was predicted that during the motor preparation, motor execution and post-movement phases there would be a negative relationship between beta-band ERD/S over M1 and RT performance. The results revealed a positive relationship between ß2

ERD/S in the motor preparation period and RT performance for Block 5 and a negative relationship for Block 1 for the left hand. In Block 5 shorter RTs were associated with larger ERD. In Block 1 shorter RTs were associated with smaller ERD.

5.2 Beta ERD/S

The finding of larger ß2 ERD (but not ß1 and ß3 ERD) for dominant and non-dominant in the motor preparation phase is in line with findings from previous studies (Meissner et al. 2018; Thibaut et al., 2017) that showed motor sequence learning is reflected by larger ERD over M1 during motor preparation. The larger ß2 ERD found in the preparatory period lends further support to previous research on the inhibitory role beta activity over M1 plays in motor sequence learning (Pfurtscheller and Lopes da Silva,1999; Jurkiewicz et al., 2006; Thibaut et al., 2017).

The finding that there were no differences between Block 1 and Block 5 in the motor execution period can be considered in line with the findings by Van der Cruijsen et al. (2021). According to Van der Cruijsen et al. (2021), theta and alpha over M1 during motor execution are related to motor sequence learning rather than beta activity. In their study, Van der Cruijsen et al. (2021) suggested that theta activity may provide for more efficient processing of visual feedback on the performance of the motor task and for the adjustment of motor control. Increased alpha activity over M1 was said to reflect an increased reliance on memory retrieval to maintain high performance in the motor task. Beta activity was suggested to merely index the level of perceived difficulty during learning. Cruijsen et al. (2021) also found that MRBD was larger during the execution of a complex motor task compared to a simple motor task. This was, however, not associated with a higher degree of motor learning (or block development in this case). This thesis supports the suggestions by Cruijsen et al. (2021) in that motor learning was reflected by the concatenation results. However, the relationship between beta activity and task difficulty was not investigated. Therefore, further conclusions about this relationship remain ambiguous.

The finding that a larger PMBR was present in Block 5 compared to Block 1 (for the left and right-hand sequences in ß1 and the left-hand sequences in ß2 and ß3) can be considered in line with the findings of a previous study by Pfurtscheller et al. (2005) and Tan et al. (2014, 2016). In those two studies, it was suggested that motor learning is reflected in M1 through enlarged PMBR. It may be suggested that PMBR relates to the level of confidence about the motor outcome (Tan et al., 2016) and/or resetting of the working memory for upcoming sequences (Pfurtscheller et al.,2005). These results are also in line with the findings by Haar and Faisal (2020) in which individual differences were said to affect changes in PMBR that come to the surface as a result of motor sequence learning. In their study, Haar and Faisal (2020) found that participants who engaged in reward-based learning showed a smaller PMBR over learning, while those who were engaged in error-based learning showed a larger PMBR throughout learning. Together these results fit well within the C-SMB perspective proposed by Verwey et al. (2015)

in which it was said that the same movement sequences can be executed with different processing strategies. The current results suggest that participants that performed the go/nogo DSP task may be effectively using the erroneous feedback to update sequence representations aligned with error-based learning strategies. The results show that the involvement of the M1 takes form in beta oscillations that can be divided into three bands (ß1, ß2, ß3). During the motor preparation phase practice is linked to more involvement of ß2 ERD over M1 for the dominant hand. During the post-movement phase more involvement of M1 translates into larger ß1 PMBR for both hands and larger ß2 and ß3 PMBR for the non-dominant hand.

5.3 ERD/S across RT

The positive relationship between ß2 ERD/S in the motor preparation period and RT performance for Block 5 and a negative relationship for Block 1 for the left-hand means that it is more costly for the brain, in terms of energy, to produce the shorter RTs in Block 1 compared to Block 5 for the left hand. A previous study by Pollok et al. (2014) revealed that larger MRBD was significantly correlated with a reduction in reaction times. This relationship was regarded as a marker for motor sequence learning. In the thesis, there were similar results for the left hand in the ß2 over M1. For shorter RTs larger ERD was observed during the motor preparation period but not the motor execution period (as in Pollok et al., 2014). There is a notable difference between the study by Pollok et al (2014) and this Thesis. For the experiment, the go/nogo DSP task was used, while in the study by Pollok et al. (2014) the Serial Reaction Time Task was used. Differences in measures and findings between this thesis and the study by Pollok (2014) make it hard to extract if there the mechanism is similar although the patterns seen are the same. However, the results can be supported by the C-SMB perspective (Verwey et al., 2015) which highlights that different processing strategies are used when performing different types of sequencing tasks. Nonetheless, one could consider that the relationships found between left-hand ß2 ERD/S over M1 during motor preparation and RT may be a marker for motor sequence learning in the go/nogo DSP task.

5.4 Conclusion

As a concluding statement it could be suggested that motor sequence learning is reflected in Beta ERD/S activity in at least two periods of sequential movement: the motor preparation and post-movement phase. This thesis revealed that motor sequence learning was reflected in larger ß2 ERD over M1 in the motor preparation phase for the dominant and non-dominant hands. Larger MRBD may support the inhibitory role of beta oscillatory activity in M1 for motor sequence learning. The findings of this thesis can aid in establishing more precise suggestions of the involvement of the role of beta oscillations for M1 in motor sequence learning. It was revealed that ß2 ERD in the motor preparation phase becomes larger as motor sequence learning takes place. This suggests that ß2 may play an inhibitory role in motor sequence

learning. The thesis also revealed that MRBD may not necessarily be reflective of motor sequence learning, but rather task difficulty. Furthermore, it was revealed that motor sequence expertise is reflected in an enlarged PMBR over M1 the (non-)dominant hands in ß1 and the non-dominant hand in ß2 and ß3. It was suggested that this phenomenon is linked to error-based learning and the resetting of the working memory and/or the level of confidence in the motor outcome. Future research may therefore focus on researching whether PMBR reflects resetting of the working memory or the level of confidence in motor the motor outcome. Furthermore, a positive relationship was observed between ß2 ERD/S over M1 for left-hand sequences in Block 5 and a negative relationship in Block 1. Based on previous literature and despite further substantial evidence, it was suggested that a negative relationship in Block 5 between left-hand ß2 ERD/s over M1 during motor preparation and RT may be a marker for motor sequence learning during the go/nogo DSP task. This indicates that less activation over M1 leads to faster execution.

5.5 Limitations and future research

The results of this thesis have to be carefully considered. There were several limitations present. The main limitation is that suggestions made as a result of consideration of the C-SMB model are less specific as imagined. Although this thesis could have significant importance to the field of motor sequence learning, it must be kept in mind that this thesis is of exploratory nature. Besides, this thesis is limited since suggestions made on ERD/S changes are usually based on healthy control groups versus Parkinson's Disease (PD)and stroke patients and young adults versus the elderly. Therefore, it is challenging to uncover whether those studies report changes in ERD/S that are related purely to MSL expertise. Another limitation of this thesis is that the focus was solely on beta oscillations. It may be interesting for future studies to investigate alpha and/or theta oscillations when focusing on the motor execution period. The reasoning behind this is that other studies have mentioned different frequency ranges to be linked to the motor execution phase. As noted before, Cruijsen et al. (2021) found that alpha and theta were more reflective of motor sequence learning during motor execution than beta. In addition, Schubert et al. (2021) found that alpha oscillations are connected to the controlling of information transfer in the premotor-cerebellar loop during motor sequence learning. The next limitation of this thesis is that the focus was laid solely on M1. This is a limitation as previous studies have mentioned that other brain areas, such as the premotor cortex may be especially relevant for revealing motor sequence learning brain mechanisms. For example, Kantak et al. (2012) found that the premotor cortex, which is part of the explicit memory system, may be engaged during the earlier stages of any motor sequence learning task in which visuospatial cues are linked to specific responses. Therefore, one may expect that the premotor cortex is engaged in the motor sequence learning that is acquired during the go/nogo DSP task. Future research may incorporate M1 as well as the premotor cortex. Lastly, the relationship between ß2 ERD and RT during motor preparation was suggested to be a possible marker for motor sequence learning.

However, research to support this suggestion is currently lacking and therefore this result remains suggestive. It could be important to investigate this relationship in future research as it could aid in predicting when motor sequence learning is established. This may be especially useful in movement rehabilitation or new training programs to decide upon the right training volume or difficulty of the task.

6. References

Abrahamse, E. L., Ruitenberg, M. F. L., de Kleine, E., & Verwey, W. B. (2013). Control of automated behaviour: insights from the discrete sequence production task. *Frontiers in human neuroscience,7*(82), https://doi.org/10.3389/fnhum.2013.00082.

Althof, H.W. (2021) *Insights into cognitive processing of the go/nogo Discrete Sequence Production task: A replication study.* Faculty: BMS: Behavioural, Management and Social Sciences - University of Twente Student Theses (utwente.nl).

Barone, J. & Rossiter, H. E. (2021). Understanding the Role of Primary motor Beta Oscillations. *Frontiers in Systems Neuroscience*, *15*, 655-886. DOI:10.3389/fnsys.2021.655886.

Brown, P., Oliviero, A., Mazzone, P., Insola, A., Tonali, P., and Di Lazzaro, V. (2001). Dopamine dependency of oscillations between subthalamic nucleus and pallidum in Parkinson's disease. *Journal of Neuroscicence, 21*, 1033-1038.

Cheyne, D., Bakhtazad, L., Gaetz, W., 2006. Spatiotemporal mapping of cortical activity accompanying voluntary movements using an event-related beamforming approach. *Human Brain Mapping, 27*,213-229

Cruijsen, J., Manoochehri, M., Jonker, Z. D., Andrinopoulou, E. R., Frens, M. A., Ribbers, G. M., Schouten, A. C., & Selles, R. W. (2021). Theta but not beta power is positively associated with better explicit motor task learning. *NeuroImage, 240*, [118373]. https://doi.org/10.1016/j.neuroimage.2021.118373.

de Kleine, E. (2009). *Cognitive control of sequential behavior*. Ipskamp Printing. https://doi.org/10.3990/1.9789036528375

de Kleine, E., & Van der Lubbe, R. H. J. (2011). Decreased load on general motor preparation and visual-working memory while preparing familiar as compared to unfamiliar movement sequences. *Brain and Cognition, 75*, 126-134. DOI:10.1016/j.bandc.2010.10.013

Espenhahn, S., van Wijk, B. C. M., Rossiter, H. E., de Berker, A. O., Redman, N. D., Rondina, J., et al. (2019). Cortical beta oscillations are associated with motor performance following visuomotor learning. *NeuroImage 195*, 340–353. doi: 10.1016/j.neuroimage.2019.03.079.

Jurkiewicz, M. T., Gaetz, W. C., Bostan, A. C. & Cheyne, D. (2006) Post-movement beta rebound is generated in motor cortex: Evidence from neuromagnetic recordings. *NeuroImage, 32*(3),1281 1289.https://doi.org/10.1016/j.neuroimage.2006.06.005.

Lee, D., & Quessy, S. (2003). Activity in the supplementary motor area related to learning and

performance during a sequential visuomotor task. *Journal of neurophysiology*, *89*(2), 1039-1056.

Little S, Bonaiuto J, Barnes G, Bestmann S (2019) Human motor cortical beta bursts relate to movement planning and response errors. *PLOS Biology*, *17*(10).https://doi.org/10.1371/journal.pbio.3000479

Litvak, V., Jha, A., Eusebio, A., Oostenveld, R., Foltynie, T., Limousin, P., et al. (2011). Resting oscillatory cortico-subthalamic connectivity in patients with Parkinson's disease. *Brain: A Journal of Neurology*, *134*, 359–374. doi: 10.1093/brain/awq332

López-Azcárate, J., Tainta, M., Rodríguez-Oroz, M. C., Valencia, M., González, R., Guridi, J., et al. (2010). Coupling between beta and high-frequency activity in the human subthalamic nucleus may be a pathophysiological mechanism in Parkinson's disease. *Journal of Neuroscience,* 30, 6667–6677. doi: 10.1523/JNEUROSCI.5459-09.2010

Marceglia, S., Foffani, G., Bianchi, A. M., Baselli, G., Tamma, F., Egidi, M., et al. (2006). Dopamine-dependent non-linear correlation between subthalamic rhythms in Parkinson's disease. *Journal of Physiology*, *571*, 579–591. doi: 10.1113/jphysiol.2005.100271

Pfurtscheller, G. & Lopes da Silva, F.H. (1999). Event-related EEG/MEG synchronization and desynchronization: basic principles. *Clinical Neurophysiology*, *110*(11), 1842-1857. doi: 10.1016/s1388-2457(99)00141-8. PMID: 10576479.

Pfurtscheller G, Pichler-Zalaudek K, Neuper C. (1998) ERD and ERS in voluntary movement of different limbs. Event-related desynchronization and related oscillatory phenomena of the brain. In: Pfurtscheller G, Pichler-Zalaudek K, Ortmayr B, Diez J, Reisecker F. Post-movement beta synchronization in patients with Parkinson's disease. *Journal of Clinical Neurophysiology*, *15*(3), 243-250.

Pfurtscheller G., Neuper C. (2003) Movement and ERD/ERS. In: Jahanshahi M., Hallett M. (eds) The Bereitschaftspotential. Springer, Boston, MA. https://doi.org/10.1007/978-1-4615-0189-3_12

Pfurtscheller, G., Neuper, C., Brunner, C & Lopes da Silva, F. (2005). Beta rebound after different types of motor imagery in man. *Neuroscience letters*, *378*, 156-9. DOI: 10.1016/j.neulet.2004.12.034.

Pollok, B., Latz, D., Krause, V., Butz, M. & Schnitzler, A (2014). Changes of motor-cortical oscillations associated with motor learning. *Neuroscience*, *275*, 47-53. DOI:10.1016/j.neuroscience.2014.06.008.

Priori, A., Foffani, G., Pesenti, A., Tamma, F., Bianchi, A. M., Pellegrini, M., et al. (2004). Rhythm-specific pharmacological modulation of subthalamic activity in Parkinson's disease. *Experimental Neurology, 189,* 369–379. doi: 10.1016/j.expneurol.2004.06.001 .

Rossiter, H. E., Davis, E. M., Clark, E. V., Boudrias, M.-H., and Ward, N. S. (2014). Beta oscillations reflect changes in motor cortex inhibition in healthy ageing. *NeuroImage, 91,* 360–365.

Meissner, V. Krause, M. Südmeyer, C.J. Hartmann, B. Pollok.The significance of brain oscillations

in motor sequence learning: insights from Parkinson's disease**.** *NeuroImage, 20*, 448-457. https://doi.org/ 10.1016/j.nicl.2018.08.009.

Sobierajewicz, J., Przekoracka-Krawczyk, A., Jaśkowski, W., Verwey, W. B., & Van der Lubbe, R. (2017). The influence of motor imagery on the learning of a fine hand motor skill. *Experimental Brain Research, 235*, 305-320. https://doi.org/10.1007/s00221-016- 4794-2.

Tan, H., Jenkinson, N., and Brown, P. (2014). Dynamic neural correlates of motor error monitoring and adaptation during trial-to-trial learning. *Journal of Neuroscience, 34*, 5678–5688. DOI: 10.1523/JNEUROSCI.4739-13.2014.

Tan, H., Wade, C., and Brown, P. (2016). Post-movement β activity in primary motor cortex indexes confidence in the estimations from internal models. *Journal of Neuroscience, 36*, 1516–1528. DOI: 10.1523/JNEUROSCI.3204-15.2016.

Taniguchi, M., Kato, A., Fujita, N., Hirata, M., Tanaka, H., Kihara, T., Ninomiya, H., Hirabuki, N., Nakamura, H., Robinson, S.E., Cheyne, D., Yoshimine, T., 2000. Movement-related desynchronization of the cerebral cortex studied with spatially filtered magnetoencephalography. *NeuroImage, 12*, 298-306.

Thibaut, A., Simis, M., Battistella, L. R., Fanciullacci, C., Bertolucci, F., Huerta-Gutierrez, R., et al. (2017). Using Brain Oscillations and Corticospinal Excitability to Understand and Predict Post-Stroke Motor Function. *Frontiers in Neurology, 8*(187). Doi: 10.3389/fneur.2017.00187.

van der Lubbe, R., Sobierajewicz, J., Jongsma, M. L. A. , Verwey, W. B., & Przekoracka-Krawczyk, A. (2021). Frontal brain areas are more involved during motor imagery than during motor execution/preparation of a response sequence. *International journal of psychophysiology, 164*, 71-86. https://doi.org/10.1016/j.ijpsycho.2021.02.020

Verwey, W. B. (2001). Concatenating familiar movement sequences: the versatile cognitive processor. *Acta psychologica, 106*(1-2), 69-95. https://doi.org/10.1016/S0001-6918(00)00027-5

Verwey, W. B., Jouen, A.-L., Dominey, P. F., & Ventre-Dominey, J. (2019). Explaining the neural 1. activity distribution associated with discrete movement sequences: Evidence for parallel functional systems. *Cognitive, Affective & Behavioral Neuroscience*, *19*(1), 138–153. https://doi.org/10.3758/s13415-018-00651-6

Verwey, W. B., Shea, C. H., & Wright, D. L. (2015). A cognitive framework for explaining serial processing and sequence execution strategies. *Psychonomic Bulletin & Review*, *22*, 54-77.https://doi.org/10.3758/s13423-014-0773-4.

Verwey, W. B., & Wright, D. L. (2014). Learning a keying sequence you never executed: Evidence for independent associative and motor chunk learning. *Acta Psychologica, 151*, 24-31.

Appendix A.

Number and percentage of dropped epochs

Condition	Left hand		Right hand	
	Block 1	**Block 5**	**Block 1**	**Block 5**
Motor prepation and execution	84 (13.5%)	109 (17.5 %)	79 (12.7 %)	112 (18.0%)
Post-movement	64 (10.3%)	63 (10.1%)	60 (9.6%)	69 (10.9%)

Note. % of 624 epochs

Appendix B.

Counterbalancing across key positions (Althof, 2021)

Participant	ID number	Six-key sequences	
		Original	Mirrored
1, 9, 14, 24	1	ADFSDA SADAFS	;KJLK; L;K;JL
2, 10. 18, 25	2	FSASDA ADSDAF	JL;LK; ;KLK;J
3, 11, 19, 28	3	SFADFS DSFSDA	LJ;KJL KLJLK;
4, 12, 26, 29	4	ADSDFS SFDFSA	;KLKJL LJKJL;
5, 15, 20, 27	5	DASFAD FDADFS	K;LJ;K JK;KJL
6, 16, 21, 30	6	SFDFAD DAFADS	LJKJ;K K;J;KL
7, 17, 22.	7	FSADAF AFSFAD	JL;K;J ;JLJ;K
8, 13, 23, 32	8	DAFASF FSASFD	K;J;LJ JL;LJK

Appendix C.

Python script

This appendix includes a part of the python script. For every participant, several events and epochs had been created due to the differences in the timing of key presses. Due to the size of the original script, the script below only includes the creation of epochs, events and Morlet wavelets averages on the data of participant 1. Furthermore, the script includes the merging of epochs and data frames of all participants and the creation of Morlet wavelets averages on the population level. For the full script please see MSL Script 1 and MSL Script 2 at https://github.com/DaphneTitsing/Sequence-learning.

```
# -*- coding: utf-8 -*-
"""
Created on Thu May 13 12:44:23 2021
@author: Daphne Titsing
"""

#-------------------------------------------------------------------------
#                       LOADING PACKAGES
#-------------------------------------------------------------------------

import os
import numpy as np
import mne
import pip
import pandas as pd
from mne.event import define_target_events
from mne.preprocessing import ICA, create_eog_epochs, create_ecg_epochs, corrmap
from mne.connectivity import spectral_connectivity
from mne.datasets import sample
from mne.viz import plot_sensors_connectivity
import matplotlib.pyplot as plt
import seaborn as sns
from mne.datasets import eegbci
from mne.io import concatenate_raws, read_raw_edf
from mne.stats import permutation_cluster_1samp_test as pcluster_test
from mne.viz.utils import center_cmap
import numpy as np
from mne import create_info, EpochsArray
from mne.baseline import rescale
from mne.time_frequency import (tfr_multitaper, tfr_stockwell, tfr_morlet,
                        tfr_array_morlet)
from mne.viz import centers_to_edges
import os.path as op
from mne.time_frequency import tfr_morlet, psd_multitaper, psd_welch




#-------------------------------------------------------------------------#
#                         PARTICIPANT 1
```

```
#-------------------------------------------------------------------------------#
#give file a name and save it#

Part_1_1 = r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\Seql_ERD_copy\ID1\Training\Part1\6
key\part1_SeqL_ERD_6_B1.vhdr'
Part_1_5 = r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\Seql_ERD_copy\ID1\Training\Part1\6
key\part1_SeqL_ERD_6_B5.vhdr'

#read file from folder

#--------B1--------#
raw_p1 = mne.io.read_raw_brainvision(Part_1_1, preload = True)

#--------B5--------#
raw5_p1 = mne.io.read_raw_brainvision(Part_1_5, preload = True)



#give channel the right type (=eeg an eog)

#--------B1---------#
raw_p1.pick_types (meg=False, eeg=True, eog=True, ecg=False)
raw_p1.set_channel_types(mapping={'vEOG_L' : 'eog'}) #ocular signals
raw_p1.set_channel_types(mapping={'vEOG_U' : 'eog'}) #ocular signals
raw_p1.set_channel_types(mapping={'hEOG_L' : 'eog'}) #ocular signals
raw_p1.set_channel_types(mapping={'hEOG_R' : 'eog'}) #ocular signals
raw_p1.drop_channels(['hEOG', 'vEOG'])              #not used
raw_p1 = mne.io.add_reference_channels(raw_p1, 'TP8')   #reference channel


#--------B5-------#
raw5_p1.pick_types (meg=False, eeg=True, eog=True, ecg=False)
raw5_p1.set_channel_types(mapping={'vEOG_L' : 'eog'}) #ocular signals
raw5_p1.set_channel_types(mapping={'vEOG_U' : 'eog'}) #ocular signals
raw5_p1.set_channel_types(mapping={'hEOG_L' : 'eog'}) #ocular signals
raw5_p1.set_channel_types(mapping={'hEOG_R' : 'eog'}) #ocular signals  '
raw5_p1.drop_channels(['hEOG', 'vEOG'])             #not used
raw5_p1 = mne.io.add_reference_channels(raw5_p1, 'TP8')  #reference channel


#plot figure of raw

#--------B1-------#

raw_p1.plot_psd(fmax = 250)#maximum frequency
raw_p1.plot(duration = 4, n_channels = 30)#duration (in  seconds) & channels shown in graph
raw_p1.plot()

#--------B5-------#
raw5_p1.plot_psd(fmax = 250)#maximum frequency
raw5_p1.plot(duration = 4, n_channels = 30)#duration (in  seconds) & channels shown in graph
raw5_p1.plot()


#set electrode location (extended 10-20system) through montage

montage = mne.channels.make_standard_montage('standard_1020')
raw_p1.set_montage(montage)
raw5_p1.set_montage(montage)


#setting bipolar reference

raw_bip_ref = mne.set_bipolar_reference(raw_p1, anode=['TP8'],
                            cathode=['TP7'])
raw_bip_ref = mne.set_bipolar_reference(raw5_p1, anode=['TP8'],
```

```
                              cathode=['TP7'])


#plot that shows the channel locations on the head
raw_p1.plot_sensors(kind='topomap', show_names=True)
raw5_p1.plot_sensors(kind='topomap', show_names=True)


#check raw information
print(raw_p1.info)
print(raw5_p1.info)


#plot to show the waves and their source on the head
raw_p1.plot_psd(fmax = 250)
raw5_p1.plot_psd(fmax = 250)




#------------------------------------------------------------------------------#
#                                ICA
#------------------------------------------------------------------------------#


#--------B1-------#

ica_p1 = mne.preprocessing.ICA()

raw_p1 = raw_p1.filter(0.1, 39)#band-pass filtering in the range 0.1Hz to 39Hz

ica_p1.fit(raw_p1)

#instead of manually selecting which ICs to exclude, we use dedicated EOG sensors as a "pattern" to check the ICs against
eog_indices_p1, eog_scores_p1 = ica_p1.find_bads_eog(raw_p1, ['vEOG_U', 'vEOG_L'])#automatically find the ICs that best match the
EOG signal
ica_p1.exclude = eog_indices_p1#excludes artefacts matching eog signals

#barpolt of ICA component "EOG" match scores
ica_p1.plot_scores(eog_scores_p1)

# plot diagnostics
ica_p1.plot_properties(raw_p1, picks=eog_indices_p1)

# plot ICs applied to raw data, with EOG matches highlighted + allows for further exclusion of components
ica_p1.plot_sources(raw_p1)


#check if raw data has been cleaned
raw_p1.plot()


#visual presentation ICA components on head
ica_p1.plot_components()

#creating eog epochs
eog_evoked_p1 = create_eog_epochs(raw_p1).average()
eog_evoked_p1.apply_baseline(baseline=(None, -0.2))
eog_evoked_p1.plot_joint()

ica_p1.plot_properties(raw_p1, [0,1])
ica_p1.exclude =[0,1]


#since ica.apply changes raw we are making a copy
reconst_raw_p1 = raw_p1.copy()
```

```
ica_p1.apply(reconst_raw_p1)#proceeds in 4 steps: 1)Unmixes the data with the unmixing matrix
#                               2)Includes ICA components based on ica.exclude
#                               3)Re-mixes the data with mixing_matrix
#                               4)Restores any data not passed to the ICA algorithm (i.e. PCA components between n_components &
n_pca_components)
reconst_raw_p1.plot()#final check of raw data, here the data should be full cleaned

reconst_raw_p1.save(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\new_part1_SeqL_ERD_6_B1.fif', overwrite=True)



#--------B5-------#

ica5_p1 = mne.preprocessing.ICA()

raw5_p1 = raw5_p1.filter(0.1, 39)#band-pass filtering in the range 0.1Hz to 39Hz

ica5_p1.fit(raw5_p1)

#instead of manually selecting which ICs to exclude, we use dedicated EOG sensors as a "pattern" to check the ICs against
eog_indices5_p1, eog_scores5_p1 = ica5_p1.find_bads_eog(raw5_p1, ['vEOG_U', 'vEOG_L'])#automatically find the ICs that best match
the EOG signal
ica5.exclude = eog_indices5#excludes artefacts matching eog signals

#barpolt of ICA component "EOG" match scores
ica5_p1.plot_scores(eog_scores5_p1)

# plot diagnostics
ica5_p1.plot_properties(raw5_p1, picks=eog_indices5_p1)

# plot ICs applied to raw data, with EOG matches highlighted + allows for further exclusion of components
ica5_p1.plot_sources(raw5_p1)

#check if raw data has been cleaned
raw5_p1.plot()

#visual presentation ICA components on head
ica5_p1.plot_components()

#creating eog epochs
eog_evoked5_p1 = create_eog_epochs(raw5_p1).average()
eog_evoked5_p1.apply_baseline(baseline=(None, -0.2))
eog_evoked5_p1.plot_joint()

ica5_p1.plot_properties(raw5_p1, [0,1,2])
ica5_p1.exclude =[0,1,2]

reconst_raw5_p1 = raw5_p1.copy()
ica5.apply(reconst_raw5_p1)#proceeds in 4 steps: 1)Unmixes the data with the unmixing matrix
#                               2)Includes ICA components based on ica.exclude
#                               3)Re-mixes the data with mixing_matrix
#                               4)Restores any data not passed to the ICA algorithm (i.e. PCA components between n_components &
n_pca_components)
reconst_raw5_p1.plot()#final check of raw data, here the data should be full cleaned

reconst_raw5_p1.save(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\new_part1_SeqL_ERD_6_B5.fif', overwrite=True)

#-------------------------------------------------------------------------------#
#                               EVENTS
#-------------------------------------------------------------------------------#

#get and save stimuli times --> make an event
#--------B1-------#
events_p1, _ = mne.events_from_annotations(raw_p1, event_id={'Stimulus/S  1': 1,'Stimulus/S  2': 2,'Stimulus/S  3': 3,'Stimulus/S  4':
4,'Stimulus/S  5': 5,'Stimulus/S  6': 6,
```

```
                                'Stimulus/S  7': 7,'Stimulus/S  8': 8,

                                'Stimulus/S  9': 9, 'Stimulus/S 10': 10,

                                'Stimulus/S 11': 11, 'Stimulus/S 12': 12,

                                'Stimulus/S 14': 14,

                                'Stimulus/S 15': 15, 'Stimulus/S 16': 16,

                                'Stimulus/S 17': 17, 'Stimulus/S 18': 18,

                                'Stimulus/S 19': 19, 'Stimulus/S 20': 20,

                                'Stimulus/S 21': 21,

                                'Stimulus/S 24': 24,

                                'Stimulus/S 25': 25, 'Stimulus/S 26': 26,

                                'Stimulus/S 27': 27, 'Stimulus/S 29': 29})


#--------B5-------#
eventsB5_p1, _ = mne.events_from_annotations(raw5_p1, event_id={'Stimulus/S  1': 1,'Stimulus/S  2': 2,'Stimulus/S  3': 3,'Stimulus/S
4': 4,'Stimulus/S  5': 5,'Stimulus/S  6': 6,

                                'Stimulus/S  7': 7,'Stimulus/S  8': 8,

                                'Stimulus/S  9': 9, 'Stimulus/S 10': 10,

                                'Stimulus/S 11': 11, 'Stimulus/S 12': 12,

                                'Stimulus/S 14': 14,

                                'Stimulus/S 15': 15, 'Stimulus/S 16': 16,

                                'Stimulus/S 17': 17, 'Stimulus/S 18': 18,

                                'Stimulus/S 19': 19, 'Stimulus/S 20': 20,

                                'Stimulus/S 21': 21,

                                'Stimulus/S 22': 22,

                                'Stimulus/S 24': 24,

                                'Stimulus/S 25': 25, 'Stimulus/S 26': 26,

                                'Stimulus/S 27': 27, 'Stimulus/S 29': 29})


#creating new events based on copy

#--------B1-------#
laststimpositionleft_b1_p1 = np.copy(events_p1)
laststimpositionright_b1_p1 = np.copy(events_p1)
feedbackleft_b1_p1 = np.copy(events_p1)
feedbackright_b1_p1 = np.copy(events_p1)
leftresponse_b1_p1 = np.copy(events_p1)
rightresponse_b1_p1 = np.copy(events_p1)
lastresponseleft_b1_p1 = np.copy(events_p1)
lastresponseright_b1_p1 = np.copy(events_p1)
preparationleft_b1_p1 = np.copy(events_p1)
preparationright_b1_p1 = np.copy(events_p1)
nogob1p1 = np.copy(events_p1)
```

```
#--------B5-------#
laststimpositionleft_b5_p1 = np.copy(eventsB5_p1)
laststimpositionright_b5_p1 = np.copy(eventsB5_p1)
feedbackleft_b5_p1 = np.copy(eventsB5_p1)
feedbackright_b5_p1 = np.copy(eventsB5_p1)
leftresponse_b5_p1 = np.copy(eventsB5_p1)
rightresponse_b5_p1 = np.copy(eventsB5_p1)
lastresponseleft_b5_p1 = np.copy(eventsB5_p1)
lastresponseright_b5_p1 = np.copy(eventsB5_p1)
preparationleft_b5_p1 = np.copy(eventsB5_p1)
preparationright_b5_p1 = np.copy(eventsB5_p1)
nogob5p1 = np.copy(eventsB5_p1)


#print to see if event times are correct

print(events_p1)

#--------B1-------#


#last  stimulus position left (34)
laststimpositionleft_b1_p1 = mne.pick_events(laststimpositionleft_b1_p1, include=[5, 6, 7, 8], exclude=[1, 2, 3, 4, 9, 10, 11, 12, 14, 15,
16, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29], step=False)
laststimpositionleft_b1_p1 = mne.merge_events(laststimpositionleft_b1_p1, [5, 6, 7, 8], 34, replace_events=True)
laststimpositionleft_b1_p1 = np.delete(laststimpositionleft_b1_p1, [77, 0, 1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 18, 19, 20, 21, 22,
24, 25, 26, 27, 28, 30, 31, 32, 33, 34, 36, 37, 38, 39, 40, 42, 43, 44, 45, 46, 48, 49, 50, 51, 52, 54, 55, 56, 57, 58, 60, 61, 62, 63, 64, 66, 67,
68, 69, 70, 72, 73, 74, 75, 76, 78, 79, 80, 81, 82, 84, 85, 86, 87, 88, 90, 91, 92, 93, 94, 96, 97, 98, 99, 100, 102, 103, 104, 105, 106,
108,109, 110, 111, 112, 114, 115, 116, 117,118, 120, 121, 122, 123, 124, 126, 127, 128, 129, 130, 132, 133, 134, 135, 136, 138, 139,
140, 141, 142, 144, 145, 146, 147, 148], axis=0)

#last  stimulus position right (35)
laststimpositionright_b1_p1 = mne.pick_events(laststimpositionright_b1_p1, include=[9, 10, 11, 12], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 14,
15, 16, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29], step=False)
laststimpositionright_b1_p1 = mne.merge_events(laststimpositionright_b1_p1, [9, 10, 11, 12], 35, replace_events=True)
laststimpositionright_b1_p1 = np.delete(laststimpositionright_b1_p1, [47, 83, 137, 0, 1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 18, 19,
20, 21, 22, 24, 25, 26, 27, 28, 30, 31, 32, 33, 34, 36, 37, 38, 39, 40, 42, 43, 44, 45, 46, 48, 49, 50, 51, 52, 54, 55, 56, 57, 58, 60, 61, 62, 63,
64, 66, 67, 68, 69, 70, 72, 73, 74, 75, 76, 78, 79, 80, 81, 82, 84, 85, 86, 87, 88, 90, 91, 92, 93, 94, 96, 97, 98, 99, 100, 102, 103, 104, 105,
106, 108,109, 110, 111, 112, 114, 115, 116, 117,118, 120, 121, 122, 123, 124, 126, 127, 128, 129, 130, 132, 133, 134, 135, 136, 138,
139, 140, 141, 142, 144, 145, 146, 147, 148, 150, 151,152, 153, 154, 156, 157, 158, 159, 160], axis=0)


#feedback left (36)
feedbackleft_b1_p1 = mne.pick_events(feedbackleft_b1_p1, include=[25,26], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17,
18, 19, 20, 21, 24, 27, 29], step=False)
feedbackleft_b1_p1 = mne.merge_events(feedbackleft_b1_p1, [25,26], 36, replace_events=True)
feedbackleft_b1_p1 = mne.event.shift_time_events(feedbackleft_b1_p1, 36, -1.000, 500)
feedbackleft_b1_p1 = np.delete(feedbackleft_b1_p1, [4, 6, 7, 8, 10, 13, 15, 17, 20, 21, 22, 23, 32, 34, 35, 36, 37, 39, 40, 41, 44, 45, 46,
47], axis=0)



#feedback right (37)
feedbackright_b1_p1 = mne.pick_events(feedbackright_b1_p1, include=[25,26], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16,
17, 18, 19, 20, 21, 24, 27, 29], step=False)
feedbackright_b1_p1 = mne.merge_events(feedbackright_b1_p1, [25,26], 37, replace_events=True)
feedbackright_b1_p1 = mne.event.shift_time_events(feedbackright_b1_p1, 37, -1.000, 500)
feedbackright_b1_p1 = np.delete(feedbackright_b1_p1, [0, 1, 2, 3, 5, 9, 11, 12, 14, 16, 18, 19, 24, 25, 26, 27, 28, 29, 30, 31, 33, 38, 42,
43], axis=0)

#left response (38)
leftresponse_b1_p1 = mne.pick_events(leftresponse_b1_p1, include=[14, 15, 16, 17], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 18, 19,
20, 21, 24, 25, 26, 27, 29], step=False)
leftresponse_b1_p1 = mne.merge_events(leftresponse_b1_p1, [14, 15, 16, 17], 38, replace_events=True)

#right response(39)
```

```
rightresponse_b1_p1 = mne.pick_events(rightresponse_b1_p1, include=[18, 19, 20, 21], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 24, 25, 26, 27, 29], step=False)
rightresponse_b1_p1 = mne.merge_events(rightresponse_b1_p1, [18, 19, 20, 21], 39, replace_events=True)

#last response left(40)
lastresponseleft_b1_p1 = mne.pick_events(lastresponseleft_b1_p1, [5, 6, 7, 8], exclude=[1, 2, 3, 4, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29] , step=False)
lastresponseleft_b1_p1 = mne.merge_events(lastresponseleft_b1_p1, [5, 6, 7, 8], 40, replace_events=True)

#last response right(41)
lastresponseright_b1_p1 = mne.pick_events(lastresponseright_b1_p1, include=[9, 10, 11, 12], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 14, 15, 16, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29], step=False)
lastresponseright_b1_p1 = mne.merge_events(lastresponseright_b1_p1, [9, 10, 11, 12], 41, replace_events=True)


#preperation left (42)
preparationleft_b1_p1 = mne.pick_events(preparationleft_b1_p1, include=[5, 6, 7, 8], exclude=[1, 2, 3, 4, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29], step=False)
preparationleft_b1_p1 = mne.merge_events(preparationleft_b1_p1, [5, 6, 7, 8], 42, replace_events=True)
preparationleft_b1_p1 = mne.event.shift_time_events(preparationleft_b1_p1, 42, 1.500, 500)
preparationleft_b1_p1  = np.delete(preparationleft_b1_p1 , [77, 0, 1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 18, 19, 20, 21, 22, 24, 25, 26, 27, 28, 30, 31, 32, 33, 34, 36, 37, 38, 39, 40, 42, 43, 44, 45, 46, 48, 49, 50, 51, 52, 54, 55, 56, 57, 58, 60, 61, 62, 63, 64, 66, 67, 68, 69, 70, 72, 73, 74, 75, 76, 78, 79, 80, 81, 82, 84, 85, 86, 87, 88, 90, 91, 92, 93, 94, 96, 97, 98, 99, 100, 102, 103, 104, 105, 106, 108,  109, 110, 111, 112, 114, 115, 116, 117,118, 120, 121, 122, 123, 124, 126, 127, 128, 129, 130, 132, 133, 134, 135, 136, 138, 139, 140, 141, 142, 144, 145, 146, 147, 148], axis=0)



#preperation right (43)
preparationright_b1_p1 = mne.pick_events(preparationright_b1_p1, include=[9, 10, 11, 12], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 14, 15, 16, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29], step=False)
preparationright_b1_p1 = mne.merge_events(preparationright_b1_p1, [9, 10, 11, 12], 43, replace_events=True)
preparationright_b1_p1 = mne.event.shift_time_events(preparationright_b1_p1, 43, 1.500, 500)
preparationright_b1_p1 = np.delete(preparationright_b1_p1, [47, 83, 137, 0, 1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 18, 19, 20, 21, 22, 24, 25, 26, 27, 28, 30, 31, 32, 33, 34, 36, 37, 38, 39, 40, 42, 43, 44, 45, 46, 48, 49, 50, 51, 52, 54, 55, 56, 57, 58, 60, 61, 62, 63, 64, 66, 67, 68, 69, 70, 72, 73, 74, 75, 76, 78, 79, 80, 81, 82, 84, 85, 86, 87, 88, 90, 91, 92, 93, 94, 96, 97, 98, 99, 100, 102, 103, 104, 105, 106, 108,109, 110, 111, 112, 114, 115, 116, 117,118, 120, 121, 122, 123, 124, 126, 127, 128, 129, 130, 132, 133, 134, 135, 136, 138, 139, 140, 141, 142, 144, 145, 146, 147, 148, 150, 151,152, 153, 154, 156, 157, 158, 159, 160], axis=0)


#nogo (44)
nogob1p1 = mne.pick_events(nogob1p1, include=[24], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 25, 26, 27, 29], step=False)
nogob1p1 = mne.merge_events(nogob1p1, [24], 44, replace_events=True)
#--------B5-------#


#last  stimulus position left (45)
laststimpositionleft_b5_p1 = mne.pick_events(laststimpositionleft_b5_p1, include=[5, 6, 7, 8], exclude=[1, 2, 3, 4, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29], step=False)
laststimpositionleft_b5_p1 = mne.merge_events(laststimpositionleft_b5_p1, [5, 6, 7, 8], 45, replace_events=True)
laststimpositionleft_b5_p1 = np.delete(laststimpositionleft_b5_p1, [119, 0, 1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 18, 19, 20, 21, 22, 24, 25, 26, 27, 28, 30, 31, 32, 33, 34, 36, 37, 38, 39, 40, 42, 43, 44, 45, 46, 48, 49, 50, 51, 52, 54, 55, 56, 57, 58, 60, 61, 62, 63, 64, 66, 67, 68, 69, 70, 72, 73, 74, 75, 76, 78, 79, 80, 81, 82, 84, 85, 86, 87, 88, 90, 91, 92, 93, 94, 96, 97, 98, 99, 100, 102, 103, 104, 105, 106, 108,109, 110, 111, 112, 114, 115, 116, 117,118, 120, 121, 122, 123, 124, 126, 127, 128, 129, 130, 132, 133, 134, 135, 136, 138, 139, 140, 141, 142, 144, 145, 146, 147, 148], axis=0)

#last  stimulus position right (46)
laststimpositionright_b5_p1 = mne.pick_events(laststimpositionright_b5_p1, include=[9, 10, 11, 12], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 14, 15, 16, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29], step=False)
laststimpositionright_b5_p1 = mne.merge_events(laststimpositionright_b5_p1, [9, 10, 11, 12], 46, replace_events=True)
laststimpositionright_b5_p1 = np.delete(laststimpositionright_b5_p1, [5, 65, 125, 0, 1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 18, 19, 20, 21, 22, 24, 25, 26, 27, 28, 30, 31, 32, 33, 34, 36, 37, 38, 39, 40, 42, 43, 44, 45, 46, 48, 49, 50, 51, 52, 54, 55, 56, 57, 58, 60, 61, 62, 63, 64, 66, 67, 68, 69, 70, 72, 73, 74, 75, 76, 78, 79, 80, 81, 82, 84, 85, 86, 87, 88, 90, 91, 92, 93, 94, 96, 97, 98, 99, 100, 102, 103, 104, 105, 106, 108,109, 110, 111, 112, 114, 115, 116, 117,118, 120, 121, 122, 123, 124, 126, 127, 128, 129, 130, 132, 133, 134, 135, 136, 138, 139, 140, 141, 142, 144, 145, 146, 147, 148, 150, 151,152, 153, 154, 156, 157, 158, 159, 160], axis=0)
```

```
#feedback (47)
feedbackleft_b5_p1 = mne.pick_events(feedbackleft_b5_p1, include=[25,26], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 24, 27, 29], step=False)
feedbackleft_b5_p1 = mne.merge_events(feedbackleft_b5_p1, [25,26], 47, replace_events=True)
feedbackleft_b5_p1 = mne.event.shift_time_events(feedbackleft_b5_p1, 47, -1.000, 500)
feedbackleft_b5_p1 = np.delete(feedbackleft_b5_p1, [0, 1, 3, 4, 10, 11, 14, 15, 16, 17, 18, 21, 25, 26, 27, 31, 34, 35, 37, 38, 39, 40, 41, 46], axis=0)


#feedback (48)
feedbackright_b5_p1 = mne.pick_events(feedbackright_b5_p1, include=[25,26], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 24, 27, 29], step=False)
feedbackright_b5_p1 = mne.merge_events(feedbackright_b5_p1, [25,26], 48, replace_events=True)
feedbackright_b5_p1 = mne.event.shift_time_events(feedbackright_b5_p1, 48, -1.000, 500)
feedbackright_b5_p1 = np.delete(feedbackright_b5_p1, [2, 5, 6, 7, 8, 9, 12, 13, 19, 20, 22, 23, 24, 28, 29, 30, 32, 33, 36, 42, 43, 44, 45, 47], axis=0)


#left response (49)
leftresponse_b5_p1 = mne.pick_events(leftresponse_b5_p1, include=[14, 15, 16, 17], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 18, 19, 20, 21, 24, 25, 26, 27, 29], step=False)
leftresponse_b5_p1 = mne.merge_events(leftresponse_b5_p1, [14, 15, 16, 17], 49, replace_events=True)

#right response(50)
rightresponse_b5_p1 = mne.pick_events(rightresponse_b5_p1, include=[18, 19, 20, 21], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 24, 25, 26, 27, 29], step=False)
rightresponse_b5_p1 = mne.merge_events(rightresponse_b5_p1, [18, 19, 20, 21], 50, replace_events=True)

#last response left(51)
lastresponseleft_b5_p1 = mne.pick_events(lastresponseleft_b5_p1, [5, 6, 7, 8], exclude=[1, 2, 3, 4, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29] , step=False)
lastresponseleft_b5_p1 = mne.merge_events(lastresponseleft_b5_p1, [5, 6, 7, 8], 51, replace_events=True)

#last response right(52)
lastresponseright_b5_p1 = mne.pick_events(lastresponseright_b5_p1, include=[9, 10, 11, 12], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 14, 15, 16, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29], step=False)
lastresponseright_b5_p1 = mne.merge_events(lastresponseright_b5_p1, [9, 10, 11, 12], 52, replace_events=True)


#preperation left (53)
preparationleft_b5_p1 = mne.pick_events(preparationleft_b5_p1, include=[5, 6, 7, 8], exclude=[1, 2, 3, 4, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29], step=False)
preparationleft_b5_p1 = mne.merge_events(preparationleft_b5_p1, [5, 6, 7, 8], 53, replace_events=True)
preparationleft_b5_p1 = mne.event.shift_time_events(preparationleft_b5_p1, 53, 1.500, 500)
preparationleft_b5_p1 = np.delete(preparationleft_b5_p1,  [119, 0, 1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 18, 19, 20, 21, 22, 24, 25, 26, 27, 28, 30, 31, 32, 33, 34, 36, 37, 38, 39, 40, 42, 43, 44, 45, 46, 48, 49, 50, 51, 52, 54, 55, 56, 57, 58, 60, 61, 62, 63, 64, 66, 67, 68, 69, 70, 72, 73, 74, 75, 76, 78, 79, 80, 81, 82, 84, 85, 86, 87, 88, 90, 91, 92, 93, 94, 96, 97, 98, 99, 100, 102, 103, 104, 105, 106, 108,109, 110, 111, 112, 114, 115, 116, 117,118, 120, 121, 122, 123, 124, 126, 127, 128, 129, 130, 132, 133, 134, 135, 136, 138, 139, 140, 141, 142, 144, 145, 146, 147, 148], axis=0)

#preperation right (54)
preparationright_b5_p1 = mne.pick_events(preparationright_b5_p1, include=[9, 10, 11, 12], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 14, 15, 16, 17, 18, 19, 20, 21, 24, 25, 26, 27, 29], step=False)
preparationright_b5_p1 = mne.merge_events(preparationright_b5_p1, [9, 10, 11, 12], 54, replace_events=True)
preparationright_b5_p1 = mne.event.shift_time_events(preparationright_b5_p1, 54, 1.500, 500)
preparationright_b5_p1 = np.delete(preparationright_b5_p1, [5, 65, 125, 0, 1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 18, 19, 20, 21, 22, 24, 25, 26, 27, 28, 30, 31, 32, 33, 34, 36, 37, 38, 39, 40, 42, 43, 44, 45, 46, 48, 49, 50, 51, 52, 54, 55, 56, 57, 58, 60, 61, 62, 63, 64, 66, 67, 68, 69, 70, 72, 73, 74, 75, 76, 78, 79, 80, 81, 82, 84, 85, 86, 87, 88, 90, 91, 92, 93, 94, 96, 97, 98, 99, 100, 102, 103, 104, 105, 106, 108,109, 110, 111, 112, 114, 115, 116, 117,118, 120, 121, 122, 123, 124, 126, 127, 128, 129, 130, 132, 133, 134, 135, 136, 138, 139, 140, 141, 142, 144, 145, 146, 147, 148, 150, 151,152, 153, 154, 156, 157, 158, 159, 160], axis=0)

#nogo (55)
nogob5p1 = mne.pick_events(nogob5p1, include=[24], exclude=[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 25, 26, 27, 29], step=False)
nogob5p1 = mne.merge_events(nogob5p1, [24], 55, replace_events=True)
```

```
#set event dictonairy with new events

#(making a event dictionary that is needed for showing the frequency of events in the plot
#depending on the block, some events are presented for one block but not for the other
#in principle the sequence indicator events are not needed as they are specified elsewhere
#if needed it can be looked up in the eXcel file and put in late)

event_dict_p1 = {'1': 1, '2': 2, '3': 3, '4': 4, '5': 5, '6': 6, '7': 7, '8': 8, '9': 9, '10': 10, '11': 11, '12': 12, '14': 14, '15': 15, '16': 16, '17': 17, '18':
18, '19': 19, '20': 20, '21': 21, '24': 24, '25': 25, '26': 26, '29': 29}
event_dictB1_p1 = {'laststimleft':34,'laststimright':35, 'feedbackL':36, 'feedbackR':37, 'leftres':38, 'rightres':39, 'lastresleft':40,
'lastresright':41, 'prepleft':42, 'prepright':43, 'nogo': 44,}
event_dictB5_p1 = {'laststimleft':45,'laststimright':46, 'feedbackL':47, 'feedbackR':48, 'leftres':49, 'rightres':50, 'lastresleft':51,
'lastresright':52, 'prepleft':53, 'prepright':54, 'nogo': 55,}



#merging events togeher into one event list

finalB1_p1 = np.concatenate((laststimpositionleft_b1_p1, laststimpositionright_b1_p1, feedbackleft_b1_p1, feedbackright_b1_p1,
leftresponse_b1_p1, rightresponse_b1_p1, lastresponseleft_b1_p1, lastresponseright_b1_p1, preparationleft_b1_p1,
preparationright_b1_p1, events_p1, nogob1p1), axis=0)
finalB5_p1 = np.concatenate((laststimpositionleft_b5_p1, laststimpositionright_b5_p1, feedbackleft_b5_p1, feedbackright_b5_p1,
leftresponse_b5_p1, rightresponse_b5_p1, lastresponseleft_b5_p1, lastresponseright_b5_p1, preparationleft_b5_p1,
preparationright_b5_p1, events_p1, nogob5p1), axis=0)


#finalepoch = np.concatenate((new_events2, new_events7, new_events16), axis=0)
#finalepoch2 = np.concatenate((new_events9, new_events15, new_events17), axis=0)
#finalepoch3 = np.concatenate((new_events16, new_events18, new_events6), axis=0)
#finalepoch4 = np.concatenate((new_events17, new_events19, new_events14), axis=0)

#checking



#create plot showing at what times selected stimuli are
fig = mne.viz.plot_events(finalB1_p1, event_id=event_dictB1_p1,

                    sfreq=raw_p1.info['sfreq'])
fig.subplots_adjust(right=0.6) #to make room for legend(description)<- smaller number bigger legend


fig = mne.viz.plot_events(finalB5_p1, event_id=event_dictB5_p1,

                    sfreq=raw5_p1.info['sfreq'])
fig.subplots_adjust(right=0.6) #to make room for legend(description)<- smaller number bigger legend

#-------------------------------------------------------------------------------#
#                            EPOCHS
#-------------------------------------------------------------------------------#

reject_criteria = dict(eeg=150e-6) #100uV

flat_criteria = dict(eeg=5e-6)#1uV

tmin, tmax = -6.5, 3 #for preparation period
tmin3, tmax3 = -4, 2.5  #for feedback period

want_chs = ['C3','C4', 'FC3', 'FC4'] #wanted channels
picks = mne.pick_channels(raw_p2.info["ch_names"], want_chs)

want_chs2 = ['Fp1', 'Fp2', 'F7', 'F3', 'F1', 'Fz', 'F2', 'F4', 'F8', 'FT7', 'FC3', 'FCz', 'FC4', 'FT8', 'T7', 'C3', 'Cz', 'C4', 'T8', 'CP3', 'CPz', 'CP4', 'Pz',
'PO7', 'Oz', 'PO8'] #wanted channels
picks2 = mne.pick_channels(raw_p2.info["ch_names"], want_chs2)
```

```
#(NOT CORRECT YET)

#_p1_#
#-------------------PREPARATION------------------#

#--------B1-------#
epochsprep1left_p1 = mne.Epochs(raw_p1, preparationleft_b1_p1, event_id=42,

                tmin=tmin, tmax=tmax, reject_tmax=0,

                reject=reject_criteria, flat=flat_criteria, baseline=(-6.5, -5.5), picks=picks, detrend=1, reject_by_annotation=True,
preload=True, event_repeated=None) #detrending is set here)

epochsprep1right_p1 = mne.Epochs(raw_p1, preparationright_b1_p1, event_id=43,

                tmin=tmin, tmax=tmax, reject_tmax=0,

                reject=reject_criteria, flat=flat_criteria, baseline=(-6.5, -5.5), picks=picks, detrend=1, reject_by_annotation=True,
preload=True, event_repeated=None) #detrending is set here)

#--------B5-------#
epochsprep5left_p1 = mne.Epochs(raw5_p1, preparationleft_b5_p1, event_id=53,

                tmin=tmin, tmax=tmax, reject_tmax=0,

                reject=reject_criteria, flat=flat_criteria, baseline=(-6.5, -5.5), picks=picks, detrend=1, reject_by_annotation=True,
preload=True, event_repeated=None) #detrending is set here)

epochsprep5right_p1 = mne.Epochs(raw5_p1, preparationright_b5_p1, event_id=54,

                tmin=tmin, tmax=tmax, reject_tmax=0,

                reject=reject_criteria, flat=flat_criteria, baseline=(-6.5, -5.5), picks=picks, detrend=1, reject_by_annotation=True,
preload=True, event_repeated=None) #detrending is set here)

#--------------------PREPARATION ALL CHANNELS-------------------#
#--------B1-------#
epochsprep1leftall_p1 = mne.Epochs(raw_p1, preparationleft_b1_p1, event_id=42,

                tmin=tmin, tmax=tmax, reject_tmax=0,

                reject=None, flat=None, baseline=(-6.5, -5.5), picks=picks2, detrend=1, reject_by_annotation=True, preload=True,
event_repeated=None) #detrending is set here)

epochsprep1rightall_p1 = mne.Epochs(raw_p1, preparationright_b1_p1, event_id=43,

                tmin=tmin, tmax=tmax, reject_tmax=0,

                reject=None, flat=None, baseline=(-6.5, -5.5), picks=picks2, detrend=1, reject_by_annotation=True, preload=True,
event_repeated=None) #detrending is set here)

#--------B5-------#
epochsprep5leftall_p1 = mne.Epochs(raw5_p1, preparationleft_b5_p1, event_id=53,

                tmin=tmin, tmax=tmax, reject_tmax=0,

                reject=None, flat=None, baseline=(-6.5, -5.5), picks=picks2, detrend=1, reject_by_annotation=True, preload=True,
event_repeated=None) #detrending is set here)

epochsprep5rightall_p1 = mne.Epochs(raw5_p1, preparationright_b5_p1, event_id=54,

                tmin=tmin, tmax=tmax, reject_tmax=0,

                reject=None, flat=None, baseline=(-6.5, -5.5), picks=picks2, detrend=1, reject_by_annotation=True, preload=True,
event_repeated=None) #detrending is set here)
```

```
#------------------FEEDBACK PERIOD------------------#
#--------B1-------#
epochsfeedback1left_p1 = mne.Epochs(raw_p1, feedbackleft_b1_p1, event_id=36,

            tmin=tmin3, tmax=tmax3, reject_tmax=0,

            reject=reject_criteria, flat=flat_criteria, baseline=(-4, -3), picks=picks, detrend=1, reject_by_annotation=True, preload=True,
event_repeated=None) #detrending is set here)

epochsfeedback1right_p1 = mne.Epochs(raw_p1, feedbackright_b1_p1, event_id=37,

            tmin=tmin3, tmax=tmax3, reject_tmax=0,

            reject=reject_criteria, flat=flat_criteria, baseline=(-4, -3), picks=picks, detrend=1, reject_by_annotation=True, preload=True,
event_repeated=None) #detrending is set here)

#--------B5-------#
epochsfeedback5left_p1 = mne.Epochs(raw5_p1, feedbackleft_b5_p1, event_id=47,

            tmin=tmin3, tmax=tmax3, reject_tmax=0,

            reject=reject_criteria, flat=flat_criteria, baseline=(-4, -3), picks=picks, detrend=1, reject_by_annotation=True, preload=True,
event_repeated=None) #detrending is set here)

epochsfeedback5right_p1 = mne.Epochs(raw5_p1, feedbackright_b5_p1, event_id=48,

            tmin=tmin3, tmax=tmax3, reject_tmax=0,

            reject=reject_criteria, flat=flat_criteria, baseline=(-4, -3), picks=picks, detrend=1, reject_by_annotation=True, preload=True,
event_repeated=None) #detrending is set here)

#------------------FEEDBACK PERIOD ALL CHANNELS------------------#
#--------B1-------#
epochsfeedback1leftall_p1 = mne.Epochs(raw_p1, feedbackleft_b1_p1, event_id=36,

            tmin=tmin3, tmax=tmax3, reject_tmax=0,

            reject=None, flat=None, baseline=(-4, -3), picks=picks2, detrend=1, reject_by_annotation=True, preload=True,
event_repeated=None) #detrending is set here)

epochsfeedback1rightall_p1 = mne.Epochs(raw_p1, feedbackright_b1_p1, event_id=37,

            tmin=tmin3, tmax=tmax3, reject_tmax=0,

            reject=None, flat=None, baseline=(-4, -3), picks=picks2, detrend=1, reject_by_annotation=True, preload=True,
event_repeated=None) #detrending is set here)

#--------B5-------#
epochsfeedback5leftall_p1 = mne.Epochs(raw5_p1, feedbackleft_b5_p1, event_id=47,

            tmin=tmin3, tmax=tmax3, reject_tmax=0,

            reject=None, flat=None, baseline=(-4, -3), picks=picks2, detrend=1, reject_by_annotation=True, preload=True,
event_repeated=None) #detrending is set here)

epochsfeedback5rightall_p1 = mne.Epochs(raw5_p1, feedbackright_b5_p1, event_id=48,

            tmin=tmin3, tmax=tmax3, reject_tmax=0,

            reject=None, flat=None, baseline=(-4, -3), picks=picks2, detrend=1, reject_by_annotation=True, preload=True,
event_repeated=None) #detrending is set here)

#concatenating epochs for erds and morlet
#all channels
#prep
```

```
concatpreparationallb1left = mne.epochs.concatenate_epochs([epochsprep1leftall_p14, epochsprep1leftall_p1,
epochsprep1leftall_p2, epochsprep1leftall_p3, epochsprep1leftall_p5, epochsprep1leftall_p6, epochsprep1leftall_p7,
epochsprep1leftall_p8, epochsprep1leftall_p9, epochsprep1leftall_p10, epochsprep1leftall_p11, epochsprep1leftall_p12,
epochsprep1leftall_p15, epochsprep1leftall_p16, epochsprep1leftall_p17, epochsprep1leftall_p18, epochsprep1leftall_p19,
epochsprep1leftall_p22, epochsprep1leftall_p23, epochsprep1leftall_p24, epochsprep1leftall_p25, epochsprep1leftall_p26,
epochsprep1leftall_p27, epochsprep1leftall_p28, epochsprep1leftall_p29, epochsprep1leftall_p30])
concatpreparationallb1right = mne.epochs.concatenate_epochs([epochsprep1rightall_p14, epochsprep1rightall_p1,
epochsprep1rightall_p2, epochsprep1rightall_p3, epochsprep1rightall_p5, epochsprep1rightall_p6, epochsprep1rightall_p7,
epochsprep1rightall_p8, epochsprep1rightall_p9, epochsprep1rightall_p10, epochsprep1rightall_p11, epochsprep1rightall_p12,
epochsprep1rightall_p15, epochsprep1rightall_p16, epochsprep1rightall_p17, epochsprep1rightall_p18, epochsprep1rightall_p19,
epochsprep1rightall_p22, epochsprep1rightall_p23, epochsprep1rightall_p24, epochsprep1rightall_p25, epochsprep1rightall_p26,
epochsprep1rightall_p27, epochsprep1rightall_p28, epochsprep1rightall_p29, epochsprep1rightall_p30])

concatpreparationallb5left = mne.epochs.concatenate_epochs([epochsprep5leftall_p14, epochsprep5leftall_p1,
epochsprep5leftall_p2, epochsprep5leftall_p3, epochsprep5leftall_p5, epochsprep5leftall_p6, epochsprep5leftall_p7,
epochsprep5leftall_p8, epochsprep5leftall_p9, epochsprep5leftall_p10, epochsprep5leftall_p11, epochsprep5leftall_p12,
epochsprep5leftall_p15, epochsprep5leftall_p16, epochsprep5leftall_p17, epochsprep5leftall_p18, epochsprep5leftall_p19,
epochsprep5leftall_p22, epochsprep5leftall_p23, epochsprep5leftall_p24, epochsprep5leftall_p25, epochsprep5leftall_p26,
epochsprep5leftall_p27, epochsprep5leftall_p28, epochsprep5leftall_p29, epochsprep5leftall_p30])
concatpreparationallb5right = mne.epochs.concatenate_epochs([epochsprep5rightall_p14, epochsprep5rightall_p1,
epochsprep5rightall_p2, epochsprep5rightall_p3, epochsprep5rightall_p5, epochsprep5rightall_p6, epochsprep5rightall_p7,
epochsprep5rightall_p8, epochsprep5rightall_p9, epochsprep5rightall_p10, epochsprep5rightall_p11, epochsprep5rightall_p12,
epochsprep5rightall_p15, epochsprep5rightall_p16, epochsprep5rightall_p17, epochsprep5rightall_p18, epochsprep5rightall_p19,
epochsprep5rightall_p22, epochsprep5rightall_p23, epochsprep5rightall_p24, epochsprep5rightall_p25, epochsprep5rightall_p26,
epochsprep5rightall_p27, epochsprep5rightall_p28, epochsprep5rightall_p29, epochsprep5rightall_p30])

#feedback
concatfeedbackallb1left = mne.epochs.concatenate_epochs([epochsfeedback1leftall_p14, epochsfeedback1leftall_p1,
epochsfeedback1leftall_p2, epochsfeedback1leftall_p3, epochsfeedback1leftall_p5, epochsfeedback1leftall_p6,
epochsfeedback1leftall_p7, epochsfeedback1leftall_p8, epochsfeedback1leftall_p9, epochsfeedback1leftall_p10,
epochsfeedback1leftall_p11, epochsfeedback1leftall_p12, epochsfeedback1leftall_p15, epochsfeedback1leftall_p16,
epochsfeedback1leftall_p17, epochsfeedback1leftall_p18, epochsfeedback1leftall_p19, epochsfeedback1leftall_p22,
epochsfeedback1leftall_p23, epochsfeedback1leftall_p24, epochsfeedback1leftall_p25, epochsfeedback1leftall_p26,
epochsfeedback1leftall_p27, epochsfeedback1leftall_p28, epochsfeedback1leftall_p29, epochsfeedback1leftall_p30])
concatfeedbackallb1right = mne.epochs.concatenate_epochs([epochsfeedback1rightall_p14, epochsfeedback1rightall_p1,
epochsfeedback1rightall_p2, epochsfeedback1rightall_p3, epochsfeedback1rightall_p5, epochsfeedback1rightall_p6,
epochsfeedback1rightall_p7, epochsfeedback1rightall_p8, epochsfeedback1rightall_p9, epochsfeedback1rightall_p10,
epochsfeedback1rightall_p11, epochsfeedback1rightall_p12, epochsfeedback1rightall_p15, epochsfeedback1rightall_p16,
epochsfeedback1rightall_p17, epochsfeedback1rightall_p18, epochsfeedback1rightall_p19, epochsfeedback1rightall_p22,
epochsfeedback1rightall_p23, epochsfeedback1rightall_p24, epochsfeedback1rightall_p25, epochsfeedback1rightall_p26,
epochsfeedback1rightall_p27, epochsfeedback1rightall_p28, epochsfeedback1rightall_p29, epochsfeedback1rightall_p30])

concatfeedbackallb5left = mne.epochs.concatenate_epochs([epochsfeedback5leftall_p14, epochsfeedback5leftall_p1,
epochsfeedback5leftall_p2, epochsfeedback5leftall_p3, epochsfeedback5leftall_p5, epochsfeedback5leftall_p6,
epochsfeedback5leftall_p7, epochsfeedback5leftall_p8, epochsfeedback5leftall_p9, epochsfeedback5leftall_p10,
epochsfeedback5leftall_p11, epochsfeedback5leftall_p12, epochsfeedback5leftall_p15, epochsfeedback5leftall_p16,
epochsfeedback5leftall_p17, epochsfeedback5leftall_p18, epochsfeedback5leftall_p19, epochsfeedback5leftall_p22,
epochsfeedback5leftall_p23, epochsfeedback5leftall_p24, epochsfeedback5leftall_p25, epochsfeedback5leftall_p26,
epochsfeedback5leftall_p27, epochsfeedback5leftall_p28, epochsfeedback5leftall_p29, epochsfeedback5leftall_p30])
concatfeedbackallb5right= mne.epochs.concatenate_epochs([epochsfeedback5rightall_p14, epochsfeedback5rightall_p1,
epochsfeedback5rightall_p2, epochsfeedback5rightall_p3, epochsfeedback5rightall_p5, epochsfeedback5rightall_p6,
epochsfeedback5rightall_p7, epochsfeedback5rightall_p8, epochsfeedback5rightall_p9, epochsfeedback5rightall_p10,
epochsfeedback5rightall_p11, epochsfeedback5rightall_p12, epochsfeedback5rightall_p15, epochsfeedback5rightall_p16,
epochsfeedback5rightall_p17, epochsfeedback5rightall_p18, epochsfeedback5rightall_p19, epochsfeedback5rightall_p22,
epochsfeedback5rightall_p23, epochsfeedback5rightall_p24, epochsfeedback5rightall_p25, epochsfeedback5rightall_p26,
epochsfeedback5rightall_p27, epochsfeedback5rightall_p28, epochsfeedback5rightall_p29, epochsfeedback5rightall_p30])


#prep
concatpreparationb1left = mne.epochs.concatenate_epochs([epochsprep1left_p14, epochsprep1left_p1, epochsprep1left_p2,
epochsprep1left_p3, epochsprep1left_p5, epochsprep1left_p6, epochsprep1left_p7, epochsprep1left_p8, epochsprep1left_p9,
epochsprep1left_p10, epochsprep1left_p11, epochsprep1left_p12, epochsprep1left_p15, epochsprep1left_p16, epochsprep1left_p17,
epochsprep1left_p18, epochsprep1left_p19, epochsprep1left_p22, epochsprep1left_p23, epochsprep1left_p24, epochsprep1left_p25,
epochsprep1left_p26, epochsprep1left_p27, epochsprep1left_p28, epochsprep1left_p29, epochsprep1left_p30])
concatpreparationb1right = mne.epochs.concatenate_epochs([epochsprep1right_p14, epochsprep1right_p1, epochsprep1right_p2,
epochsprep1right_p3, epochsprep1right_p5, epochsprep1right_p6, epochsprep1right_p7, epochsprep1right_p8,
```

```
epochsprep1right_p9, epochsprep1right_p10, epochsprep1right_p11, epochsprep1right_p12, epochsprep1right_p15,
epochsprep1right_p16, epochsprep1right_p17, epochsprep1right_p18, epochsprep1right_p19, epochsprep1right_p22,
epochsprep1right_p23, epochsprep1right_p24, epochsprep1right_p25, epochsprep1right_p26, epochsprep1right_p27,
epochsprep1right_p28, epochsprep1right_p29, epochsprep1right_p30])

concatpreparationb5left = mne.epochs.concatenate_epochs([epochsprep5left_p14, epochsprep5left_p1, epochsprep5left_p2,
epochsprep5left_p3, epochsprep5left_p5, epochsprep5left_p6, epochsprep5left_p7, epochsprep5left_p8, epochsprep5left_p9,
epochsprep5left_p10, epochsprep5left_p11, epochsprep5left_p12, epochsprep5left_p15, epochsprep5left_p16, epochsprep5left_p17,
epochsprep5left_p18, epochsprep5left_p19, epochsprep5left_p22, epochsprep5left_p23, epochsprep5left_p24, epochsprep5left_p25,
epochsprep5left_p26, epochsprep5left_p27, epochsprep5left_p28, epochsprep5left_p29, epochsprep5left_p30])
concatpreparationb5right = mne.epochs.concatenate_epochs([epochsprep5right_p14, epochsprep5right_p1, epochsprep5right_p2,
epochsprep5right_p3, epochsprep5right_p5, epochsprep5right_p6, epochsprep5right_p7, epochsprep5right_p8,
epochsprep5right_p9, epochsprep5right_p10, epochsprep5right_p11, epochsprep5right_p12, epochsprep5right_p15,
epochsprep5right_p16, epochsprep5right_p17, epochsprep5right_p18, epochsprep5right_p19, epochsprep5right_p22,
epochsprep5right_p23, epochsprep5right_p24, epochsprep5right_p25, epochsprep5right_p26, epochsprep5right_p27,
epochsprep5right_p28, epochsprep5right_p29, epochsprep5right_p30])


#feedback
concatfeedbackb1left = mne.epochs.concatenate_epochs([epochsfeedback1left_p14, epochsfeedback1left_p1,
epochsfeedback1left_p2, epochsfeedback1left_p3, epochsfeedback1left_p5, epochsfeedback1left_p6, epochsfeedback1left_p7,
epochsfeedback1left_p8, epochsfeedback1left_p9, epochsfeedback1left_p10, epochsfeedback1left_p11, epochsfeedback1left_p12,
epochsfeedback1left_p15, epochsfeedback1left_p16, epochsfeedback1left_p17, epochsfeedback1left_p18, epochsfeedback1left_p19,
epochsfeedback1left_p22, epochsfeedback1left_p23, epochsfeedback1left_p24, epochsfeedback1left_p25, epochsfeedback1left_p26,
epochsfeedback1left_p27, epochsfeedback1left_p28, epochsfeedback1left_p29, epochsfeedback1left_p30])
concatfeedbackb1right = mne.epochs.concatenate_epochs([epochsfeedback1right_p14, epochsfeedback1right_p1,
epochsfeedback1right_p2, epochsfeedback1right_p3, epochsfeedback1right_p5, epochsfeedback1right_p6, epochsfeedback1right_p7,
epochsfeedback1right_p8, epochsfeedback1right_p9, epochsfeedback1right_p10, epochsfeedback1right_p11,
epochsfeedback1right_p12, epochsfeedback1right_p15, epochsfeedback1right_p16, epochsfeedback1right_p17,
epochsfeedback1right_p18, epochsfeedback1right_p19, epochsfeedback1right_p22, epochsfeedback1right_p23,
epochsfeedback1right_p24, epochsfeedback1right_p25, epochsfeedback1right_p26, epochsfeedback1right_p27,
epochsfeedback1right_p28, epochsfeedback1right_p29, epochsfeedback1right_p30])

concatfeedbackb5left = mne.epochs.concatenate_epochs([epochsfeedback5left_p14, epochsfeedback5left_p1,
epochsfeedback5left_p2, epochsfeedback5left_p3, epochsfeedback5left_p5, epochsfeedback5left_p6, epochsfeedback5left_p7,
epochsfeedback5left_p8, epochsfeedback5left_p9, epochsfeedback5left_p10, epochsfeedback5left_p11, epochsfeedback5left_p12,
epochsfeedback5left_p15, epochsfeedback5left_p16, epochsfeedback5left_p17, epochsfeedback5left_p18, epochsfeedback5left_p19,
epochsfeedback5left_p22, epochsfeedback5left_p23, epochsfeedback5left_p24, epochsfeedback5left_p25, epochsfeedback5left_p26,
epochsfeedback5left_p27, epochsfeedback5left_p28, epochsfeedback5left_p29, epochsfeedback5left_p30])
concatfeedbackb5right= mne.epochs.concatenate_epochs([epochsfeedback5right_p14, epochsfeedback5right_p1,
epochsfeedback5right_p2, epochsfeedback5right_p3, epochsfeedback5right_p5, epochsfeedback5right_p6, epochsfeedback5right_p7,
epochsfeedback5right_p8, epochsfeedback5right_p9, epochsfeedback5right_p10, epochsfeedback5right_p11,
epochsfeedback5right_p12, epochsfeedback5right_p15, epochsfeedback5right_p16, epochsfeedback5right_p17,
epochsfeedback5right_p18, epochsfeedback5right_p19, epochsfeedback5right_p22, epochsfeedback5right_p23,
epochsfeedback5right_p24, epochsfeedback5right_p25, epochsfeedback5right_p26, epochsfeedback5right_p27,
epochsfeedback5right_p28, epochsfeedback5right_p29, epochsfeedback5right_p30])


#-------------------------------------------------------------------------------#
#                          MORLET WAVELETS
#-------------------------------------------------------------------------------#


###determining the frequencies###
freqs = np.arange(12, 29, 1)      # full range
freqsbeta1 = np.arange(12, 16, 1)  # beta1 (12-17Hz)
freqsbeta2 = np.arange(17, 20, 1)  # beta2 (17-23Hz)
freqsbeta3 = np.arange(21, 29, 1)  # beta3 (23-29Hz)

#different number of cycle per frequency
n_cycles = 3
n_cycles1 = freqsbeta1 / 2.
n_cycles2 = freqsbeta2 / 2.
n_cycles3 = freqsbeta3 / 2.

baseline = [-6.5, -5.5]  # baseline interval (in s)
```

```
baseline2 = [-7, -6]
baseline3 = [-4, -3]



#-------------------------------------------------------------------------------#
#                    MORLET PER PARTICIPANT
#-------------------------------------------------------------------------------#

#------------#
#PARTICIPANT 1
#------------#

#--------------------PREPARATION----------------#

#----B1----#
powerprep1allbandsleft_p1, itc1 = tfr_morlet(epochsprep1left_p1, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)
powerprep1allbandsright_p1, itc2 = tfr_morlet(epochsprep1right_p1, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)


#----B5----#
powerprep5allbandsleft_p1, itc3 = tfr_morlet(epochsprep5left_p1, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)
powerprep5allbandsright_p1, itc4 = tfr_morlet(epochsprep5right_p1, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)


#-------------------FEEDBACK-----------------#

#----B1----#
powerfeedback1allbandsleft_p1, itc9 = tfr_morlet(epochsfeedback1left_p1, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)
powerfeedback1allbandsright_p1, itc10 = tfr_morlet(epochsfeedback1right_p1, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)

#----B5----#
powerfeedback5allbandsleft_p1, itc11 = tfr_morlet(epochsfeedback5left_p1, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)

powerfeedback5allbandsright_p1, itc12 = tfr_morlet(epochsfeedback5right_p1, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)

#------------#
#PARTICIPANT 1
#------------#

#------prep------#
#b1
dfpowerprep1allbandsleft_p1 = powerprep1allbandsleft_p1.to_data_frame(time_format=None)
dfpowerprep1allbandsleft_p1['Block'] = 'B1'
dfpowerprep1allbandsleft_p1['Hand'] = 'left'
dfpowerprep1allbandsleft_p1['Participant'] = '1'
print(dfpowerprep1allbandsleft_p1)

dfpowerprep1allbandsright_p1 = powerprep1allbandsright_p1.to_data_frame(time_format=None)
dfpowerprep1allbandsright_p1['Block'] = 'B1'
dfpowerprep1allbandsright_p1['Hand'] = 'right'
dfpowerprep1allbandsright_p1['Participant'] = '1'
print(dfpowerprep1allbandsright_p1)

#b5
dfpowerprep5allbandsleft_p1 = powerprep5allbandsleft_p1.to_data_frame(time_format=None)
dfpowerprep5allbandsleft_p1['Block'] = 'B5'
dfpowerprep5allbandsleft_p1['Hand'] = 'left'
```

```
dfpowerprep5allbandsleft_p1['Participant'] = '1'
print(dfpowerprep5allbandsleft_p1)

dfpowerprep5allbandsright_p1 = powerprep5allbandsright_p1.to_data_frame(time_format=None)
dfpowerprep5allbandsright_p1['Block'] = 'B5'
dfpowerprep5allbandsright_p1['Hand'] = 'right'
dfpowerprep5allbandsright_p1['Participant'] = '1'
print(dfpowerprep5allbandsright_p1)



#------feedback------#
#b1
dfpowerfeedback1allbandsleft_p1 = powerfeedback1allbandsleft_p1.to_data_frame(time_format=None)
dfpowerfeedback1allbandsleft_p1['Block'] = 'B1'
dfpowerfeedback1allbandsleft_p1['Hand'] = 'left'
dfpowerfeedback1allbandsleft_p1['Participant'] = '1'
print(dfpowerfeedback1allbandsleft_p1)

dfpowerfeedback1allbandsright_p1 = powerfeedback1allbandsright_p1.to_data_frame(time_format=None)
dfpowerfeedback1allbandsright_p1['Block'] = 'B1'
dfpowerfeedback1allbandsright_p1['Hand'] = 'right'
dfpowerfeedback1allbandsright_p1['Participant'] = '1'
print(dfpowerfeedback1allbandsright_p1)

#b5
dfpowerfeedback5allbandsleft_p1 = powerfeedback5allbandsleft_p1.to_data_frame(time_format=None)
dfpowerfeedback5allbandsleft_p1['Block'] = 'B5'
dfpowerfeedback5allbandsleft_p1['Hand'] = 'left'
dfpowerfeedback5allbandsleft_p1['Participant'] = '1'
print(dfpowerfeedback5allbandsleft_p1)

dfpowerfeedback5allbandsright_p1 = powerfeedback5allbandsright_p1.to_data_frame(time_format=None)
dfpowerfeedback5allbandsright_p1['Block'] = 'B5'
dfpowerfeedback5allbandsright_p1['Hand'] = 'right'
dfpowerfeedback5allbandsright_p1['Participant'] = '1'
print(dfpowerfeedback5allbandsright_p1)

dfpowerprepindivB1_p1 = pd.concat([dfpowerprep1allbandsright_p1, dfpowerprep1allbandsleft_p1], axis=0)
dfpowerprepindivB1_p1.to_csv(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\XLS files for master
thesis\powerprepindivB1_p1.csv', index = False)
dfpowerprepindivB5_p1 = pd.concat([dfpowerprep5allbandsright_p1, dfpowerprep5allbandsleft_p1], axis=0)
dfpowerprepindivB5_p1.to_csv(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\XLS files for master
thesis\powerprepindivB5_p1.csv', index = False)


dfpowerfeedbackindivB1_p1 = pd.concat([dfpowerfeedback1allbandsright_p1, dfpowerfeedback1allbandsleft_p1], axis=0)
dfpowerfeedbackindivB1_p1.to_csv(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\XLS files for master
thesis\powerfeedbackindivB1_p1.csv', index = False)
dfpowerfeedbackindivB5_p1 = pd.concat([dfpowerfeedback5allbandsright_p1, dfpowerfeedback5allbandsleft_p1], axis=0)
dfpowerfeedbackindivB5_p1.to_csv(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\XLS files for master
thesis\powerfeedbackindivB5_p1.csv', index = False)

#-------------------------------------------------------------------------#
#                    MORLET AVERAGE OF ALL PARTICIPANTS
#-------------------------------------------------------------------------#

#-------------------PREPARATION----------------#

#-------------------PREPARATION ALL BANDS-----------------#
#----B1----#
powerprep1allbandsleft, itc321 = tfr_morlet(concatpreparationb1left, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)
powerprep1allbandsright, itc322 = tfr_morlet(concatpreparationb1right, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)
#----B5----#
```

```
powerprep5allbandsleft, itc323 = tfr_morlet(concatpreparationb5left, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)
powerprep5allbandsright, itc324 = tfr_morlet(concatpreparationb5right, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)

#-------------------PREPARATION ALL CHANNELS ALL BANDS-------------------#

#(for computing topo maps of average power)

#----B1----#
powerprep1allleft, itc325 = tfr_morlet(concatpreparationallb1left, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)
powerprep1allright, itc326 = tfr_morlet(concatpreparationallb1right, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)

#----B5----#
powerprep5allleft, itc327 = tfr_morlet(concatpreparationallb5left, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)

powerprep5allright, itc328 = tfr_morlet(concatpreparationallb5right, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)



#-------------------FEEDBACK----------------#

#-------------------FEEDBACK ALL BANDS-------------------#
#----B1----#
powerfeedback1allbandsleft, itc337 = tfr_morlet(concatfeedbackb1left, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)
powerfeedback1allbandsright, itc338 = tfr_morlet(concatfeedbackb1right, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)
#----B5----#

powerfeedback5allbandsleft, itc339 = tfr_morlet(concatfeedbackb5left, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)
powerfeedback5allbandsright, itc340 = tfr_morlet(concatfeedbackb5right, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)

#-------------------FEEDBACK ALL CHANNELS ALL BANDS-------------------#

#(for computing topo maps of average power)

#----B1----#
powerfeedback1allleft, itc341 = tfr_morlet(concatfeedbackallb1left, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)
powerfeedback1allright, itc342 = tfr_morlet(concatfeedbackallb1right, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)

#----B5----#
powerfeedback5allleft, itc343 = tfr_morlet(concatfeedbackallb5left, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)

powerfeedback5allright, itc344 = tfr_morlet(concatfeedbackallb5right, freqs=freqs, n_cycles=n_cycles, use_fft=True,
                return_itc=True, n_jobs=1)

#exporting to dataframe
#prep
#b1
dfpowerprep1allbandsleft = powerprep1allbandsleft.to_data_frame(time_format=None)
dfpowerprep1allbandsleft['Block'] = 'B1'
dfpowerprep1allbandsleft['Hand'] = 'left'
print(dfpowerprep1allbandsleft)

dfpowerprep1allbandsright = powerprep1allbandsright.to_data_frame(time_format=None)
```

```
dfpowerprep1allbandsright['Block'] = 'B1'
dfpowerprep1allbandsright['Hand'] = 'right'
print(dfpowerprep1allbandsright)

#b5
dfpowerprep5allbandsleft = powerprep5allbandsleft.to_data_frame(time_format=None)
dfpowerprep5allbandsleft['Block'] = 'B5'
dfpowerprep5allbandsleft['Hand'] = 'left'
print(dfpowerprep5allbandsleft)

dfpowerprep5allbandsright = powerprep5allbandsright.to_data_frame(time_format=None)
dfpowerprep5allbandsright['Block'] = 'B5'
dfpowerprep5allbandsright['Hand'] = 'right'
print(dfpowerprep1allbandsright)


dfpowerprep = pd.concat([dfpowerprep1allbandsleft, dfpowerprep5allbandsleft, dfpowerprep1allbandsright,
dfpowerprep5allbandsright], axis=0)
dfpowerprep.to_csv(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\XLS files for master thesis\powerprep.csv', index = False)


#feedback
#b1
dfpowerfeedback1allbandsleft = powerfeedback1allbandsleft.to_data_frame(time_format=None)
dfpowerfeedback1allbandsleft['Block'] = 'B1'
dfpowerfeedback1allbandsleft['Hand'] = 'left'
print(dfpowerfeedback1allbandsleft)

dfpowerfeedback1allbandsright = powerfeedback1allbandsright.to_data_frame(time_format=None)
dfpowerfeedback1allbandsright['Block'] = 'B1'
dfpowerfeedback1allbandsright['Hand'] = 'right'
print(dfpowerfeedback1allbandsright)

#b5
dfpowerfeedback5allbandsleft = powerfeedback5allbandsleft.to_data_frame(time_format=None)
dfpowerfeedback5allbandsleft['Block'] = 'B5'
dfpowerfeedback5allbandsleft['Hand'] = 'left'
print(dfpowerfeedback5allbandsleft)

dfpowerfeedback5allbandsright = powerfeedback5allbandsright.to_data_frame(time_format=None)
dfpowerfeedback5allbandsright['Block'] = 'B5'
dfpowerfeedback5allbandsright['Hand'] = 'right'
print(dfpowerfeedback1allbandsright)


dfpowerfeedback = pd.concat([dfpowerfeedback1allbandsleft, dfpowerfeedback5allbandsleft, dfpowerfeedback1allbandsright,
dfpowerfeedback5allbandsright], axis=0)
dfpowerfeedback.to_csv(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\XLS files for master thesis\powerfeedback.csv', index
= False)



# Computing a topomap of average power for all channels to guide which channels should be picked
#LEFT HAND
#prep
powerprep1allleft.plot_topo(tmin=-0.5, tmax=1.5, baseline=baseline, mode='percent', vmin= -2, vmax=2, title='Average power prep1
left ')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\topo\prep\powerprep1alltopoleft')

powerprep5allleft.plot_topo(tmin=-0.5, tmax=1.5, baseline=baseline, mode='percent', vmin= -2, vmax=2, title='Average power prep5
left')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\topo\prep\powerprep5alltopoleft')

#feedback
powerfeedback1allleft.plot_topo(tmin=-0.5, tmax=2.5, baseline=baseline3, mode='percent', vmin= -2, vmax=2, title='Average power
feedback1 left')
```

```
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\topo\feedback\powerfeedback1alltopoleft')


powerfeedback5allleft.plot_topo(tmin=-0.5, tmax=2.5, baseline=baseline3, mode='percent', vmin= -2, vmax=2, title='Average power
feedback5 left')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\topo\feedback\powerfeedback5alltopoleft')


#Showing the morlet wavelets in plots per frequency band
vmin, vmax = -2, 2
#----PREPARATION----#

#C3
powerprep1allbandsleft.plot(['C3'], tmin=-0.5, tmax=1.5, baseline=baseline, mode='percent', vmin=vmin, vmax=vmax, title='prep1 C3
left')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\prep\C3\powerprep1C3left')


powerprep5allbandsleft.plot(['C3'], tmin=-0.5, tmax=1.5, baseline=baseline, mode='percent', vmin=vmin, vmax=vmax, title='prep5 C3
left')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\prep\C3\powerprep5C3left')

#C4
powerprep1allbandsleft.plot(['C4'], tmin=-0.5, tmax=1.5, baseline=baseline, mode='percent', vmin=vmin, vmax=vmax, title='prep1 C4
left')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\prep\C4\powerprep1C4left')


powerprep5allbandsleft.plot(['C4'], tmin=-0.5, tmax=1.5, baseline=baseline, mode='percent', vmin=vmin, vmax=vmax, title='prep5 C4
left')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\prep\C4\powerprep5C4left')


#----FEEDBACK----#

#C3
powerfeedback1allbandsleft.plot(['C3'], tmin=-0.5, tmax=2.5, baseline=baseline3, mode='percent', vmin=vmin, vmax=vmax,
title='feedback1 C3 left')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\feedback\C3\powerfeedback1C3left')


powerfeedback5allbandsleft.plot(['C3'], tmin=-0.5, tmax=2.5, baseline=baseline3, mode='percent', vmin=vmin, vmax=vmax,
title='feedback5 C3 left')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\feedback\C3\powerfeedback5C3left')

#C4
powerfeedback1allbandsleft.plot(['C4'], tmin=-0.5, tmax=2.5, baseline=baseline3, mode='percent', vmin=vmin, vmax=vmax,
title='feedback1 C4 left')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\feedback\C4\powerfeedback1C4left')


powerfeedback5allbandsleft.plot(['C4'], tmin=-0.5, tmax=2.5, baseline=baseline3, mode='percent', vmin=vmin, vmax=vmax,
title='feedback5 C4 left')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\feedback\C4\powerfeedback5C4left')


#RIGHT HAND

# Computing a topomap of average power for all channels to guide which channels should be picked
#prep
powerprep1allright.plot_topo(tmin=-0.5, tmax=1.5, baseline=baseline, mode='percent', vmin= -2, vmax=2, title='Average power prep1
right ')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\topo\prep\powerprep1alltoporight')

powerprep5allright.plot_topo(tmin=-0.5, tmax=1.5, baseline=baseline, mode='percent', vmin= -2, vmax=2, title='Average power prep5
right')
```

```
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\topo\prep\powerprep5alltoporight')

#feedback
powerfeedback1allright.plot_topo(tmin=-0.5, tmax=2.5, baseline=baseline3, mode='percent', vmin= -2, vmax=2, title='Average power
feedback1 right')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\topo\feedback\powerfeedback1alltoporight')


powerfeedback5allright.plot_topo(tmin=-0.5, tmax=2.5, baseline=baseline3, mode='percent', vmin= -2, vmax=2, title='Average power
feedback5 right')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\topo\feedback\powerfeedback5alltoporight')


#Showing the morlet wavelets in plots per frequency band
vmin, vmax = -2, 2
#----PREPARATION----#

#C3/
powerprep1allbandsright.plot(['C3'], tmin=-0.5, tmax=1.5, baseline=baseline, mode='percent', vmin=vmin, vmax=vmax, title='prep1 C3
right')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\prep\C3\powerprep1C3right')


powerprep5allbandsright.plot(['C3'], tmin=-0.5, tmax=1.5, baseline=baseline, mode='percent', vmin=vmin, vmax=vmax, title='prep5 C3
right')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\prep\C3\powerprep5C3right')

#C4
powerprep1allbandsright.plot(['C4'], tmin=-0.5, tmax=1.5, baseline=baseline, mode='percent', vmin=vmin, vmax=vmax, title='prep1 C4
right')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\prep\C4\powerprep1C4right')


powerprep5allbandsright.plot(['C4'], tmin=-0.5, tmax=1.5, baseline=baseline, mode='percent', vmin=vmin, vmax=vmax, title='prep5 C4
right')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\prep\C4\powerprep5C4right')



#----FEEDBACK----#

#C3
powerfeedback1allbandsright.plot(['C3'], tmin=-0.5, tmax=2.5, baseline=baseline3, mode='percent', vmin=vmin, vmax=vmax,
title='feedback1 C3 right')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\feedback\C3\powerfeedback1C3right')


powerfeedback5allbandsright.plot(['C3'], tmin=-0.5, tmax=2.5, baseline=baseline3, mode='percent', vmin=vmin, vmax=vmax,
title='feedback5 C3 right')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\feedback\C3\powerfeedback5C3right')

#C4
powerfeedback1allbandsright.plot(['C4'], tmin=-0.5, tmax=2.5, baseline=baseline3, mode='percent', vmin=vmin, vmax=vmax,
title='feedback1 C4 right')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\feedback\C4\powerfeedback1C4right')


powerfeedback5allbandsright.plot(['C4'], tmin=-0.5, tmax=2.5, baseline=baseline3, mode='percent', vmin=vmin, vmax=vmax,
title='feedback5 C4 right')
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\wavelets\feedback\C4\powerfeedback5C4right')



#Topoplots over time
#LEFT HAND
```

```
#prep b1 left

fig, axis = plt.subplots(1, 3, figsize=(30, 30))
powerprep1allleft.plot_topomap(ch_type='eeg', tmin=-1.1, tmax=-1, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline, mode='percent', axes=axis[0], show=False)
powerprep1allleft.plot_topomap(ch_type='eeg', tmin=-1, tmax=-0.9, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline, mode='percent', axes=axis[1], show=False)
powerprep1allleft.plot_topomap(ch_type='eeg', tmin=-0.9, tmax=-0.8, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline, mode='percent', axes=axis[2], show=False)

plt.suptitle('Left hand - Block 1',fontsize=20, X=0.1, y=0.9)
mne.viz.tight_layout()
plt.show()
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\topomap\prep\powerprep1b2alltopomapleft')



#prep b5 left
fig2, axis2 = plt.subplots(1, 3, figsize=(30, 30))
powerprep5allleft.plot_topomap(ch_type='eeg', tmin=-1.1, tmax=-1, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline, mode='percent', axes=axis2[0], show=False)
powerprep5allleft.plot_topomap(ch_type='eeg', tmin=-1, tmax=-0.9, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline, mode='percent', axes=axis2[1], show=False)
powerprep5allleft.plot_topomap(ch_type='eeg', tmin=-0.9, tmax=-0.8, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline, mode='percent', axes=axis2[2], show=False)

plt.suptitle('Left hand - Block 5',fontsize=20, X=0.1, y=0.9)
mne.viz.tight_layout()
plt.show()
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\topomap\prep\powerprep5b2alltopomapleft')


#feedbackb1 left
fig3, axis3 = plt.subplots(1, 4, figsize=(20, 20))
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=0.7, tmax=0.8, fmin=12, fmax=18, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis3[0], show=False)
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=0.8, tmax=0.9, fmin=12, fmax=18, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis3[1], show=False)
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=1.1, tmax=1.2, fmin=12, fmax=18, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis3[2], show=False)
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=1.2, tmax=1.3, fmin=12, fmax=18, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis3[3], show=False)

plt.suptitle('Left hand - Block 1',fontsize=20, X=0.1, y=0.8)
mne.viz.tight_layout()
plt.show()
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master
Thesis\plots\morlet\topomap\feedback\powerfeedback1b1alltopomapleft')



#feedbackb5 left
fig4, axis4 = plt.subplots(1, 4, figsize=(30, 30))
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=0.7, tmax=0.8, fmin=12, fmax=18, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis4[0], show=False)
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=0.8, tmax=0.9, fmin=12, fmax=18, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis4[1], show=False)
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=1.1, tmax=1.2, fmin=12, fmax=18, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis4[2], show=False)
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=1.2, tmax=1.3, fmin=12, fmax=18, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis4[3], show=False)

plt.suptitle('Left hand - Block 5',fontsize=20, X=0.1, y=0.8)
mne.viz.tight_layout()
plt.show()
```

```
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master
Thesis\plots\morlet\topomap\feedback\powerfeedback5b1alltopomapleft')

#feedbackb1 left
fig5, axis5 = plt.subplots(1, 4, figsize=(30, 30))
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=0.7, tmax=0.8, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis5[0], show=False)
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=1.0, tmax=1.1, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis5[1], show=False)
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=1.1, tmax=1.2, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis5[2], show=False)
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=1.2, tmax=1.3, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis5[3], show=False)

plt.suptitle('Left hand - Block 1',fontsize=20, X=0.1, y=0.8)
mne.viz.tight_layout()
plt.show()
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master
Thesis\plots\morlet\topomap\feedback\powerfeedback1b2alltopomapleft')

#feedbackb5 left
fig6, axis6 = plt.subplots(1, 4, figsize=(30, 30))
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=0.7, tmax=0.8, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis6[0], show=False)
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=1.0, tmax=1.1, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis6[1], show=False)
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=1.1, tmax=1.2, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis6[2], show=False)
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=1.2, tmax=1.3, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis6[3], show=False)

plt.suptitle('Left hand - Block 5',fontsize=20, X=0.1, y=0.8)
mne.viz.tight_layout()
plt.show()
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master
Thesis\plots\morlet\topomap\feedback\powerfeedback5b2alltopomapleft')




#feedbackb1 left
fig7, axis7 = plt.subplots(1, 5, figsize=(30, 30))
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=0, tmax=0.1, fmin=24, fmax=29, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis7[0], show=False)
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=0.7, tmax=0.8, fmin=24, fmax=29, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis7[1], show=False)
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=0.8, tmax=0.9, fmin=24, fmax=29, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis7[2], show=False)
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=1, tmax=1.1, fmin=24, fmax=29, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis7[3], show=False)
powerfeedback1allleft.plot_topomap(ch_type='eeg', tmin=1.1, tmax=1.2, fmin=24, fmax=29, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis7[4], show=False)

plt.suptitle('Left hand - Block 1',fontsize=20, X=0.1, y=0.8)
mne.viz.tight_layout()
plt.show()
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master
Thesis\plots\morlet\topomap\feedback\powerfeedback1b3alltopomapleft')


#feedbackb5 left
fig8, axis8 = plt.subplots(1, 5, figsize=(30, 30))
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=0, tmax=0.1, fmin=24, fmax=29, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis8[0], show=False)
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=0.7, tmax=0.8, fmin=24, fmax=29, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis8[1], show=False)
```

```
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=0.8, tmax=0.9, fmin=24, fmax=29, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis8[2], show=False)
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=1, tmax=1.1, fmin=24, fmax=29, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis8[3], show=False)
powerfeedback5allleft.plot_topomap(ch_type='eeg', tmin=1.1, tmax=1.2, fmin=24, fmax=29, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis8[4], show=False)

plt.suptitle('Left hand - Block 5',fontsize=20, X=0.1, y=0.8)
mne.viz.tight_layout()
plt.show()
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master
Thesis\plots\morlet\topomap\feedback\powerfeedback5b3alltopomapleft')


#Topoplots over time
#RIGHT HAND

#prep b1 right

fig9, axis9 = plt.subplots(1, 3, figsize=(30, 30))
powerprep1allright.plot_topomap(ch_type='eeg', tmin=-1.1, tmax=-1, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline, mode='percent', axes=axis9[0], show=False)
powerprep1allright.plot_topomap(ch_type='eeg', tmin=-0.8, tmax=-0.7, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline, mode='percent', axes=axis9[1], show=False)
powerprep1allright.plot_topomap(ch_type='eeg', tmin=-0.4, tmax=-0.3, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline, mode='percent', axes=axis9[2], show=False)

plt.suptitle('Right hand - Block 1',fontsize=20, X=0.1, y=0.9)
mne.viz.tight_layout()
plt.show()
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\topomap\prep\powerprep1b2alltopomapright')



#prep b5 right
fig10, axis10 = plt.subplots(1, 3, figsize=(30, 30))
powerprep5allright.plot_topomap(ch_type='eeg', tmin=-1.1, tmax=-1, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline, mode='percent', axes=axis10[0], show=False)
powerprep5allright.plot_topomap(ch_type='eeg', tmin=-0.8, tmax=-0.7, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline, mode='percent', axes=axis10[1], show=False)
powerprep5allright.plot_topomap(ch_type='eeg', tmin=-0.4, tmax=-0.3, fmin=19, fmax=23, vmin=vmin, vmax=vmax,
            baseline=baseline, mode='percent', axes=axis10[2], show=False)

plt.suptitle('Right hand - Block 5',fontsize=20, X=0.1, y=0.9)
mne.viz.tight_layout()
plt.show()
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master Thesis\plots\morlet\topomap\prep\powerprep5b2alltopomapright')


#feedbackb1 right
fig11, axis11 = plt.subplots(1, 2, figsize=(30, 30))
powerfeedback1allright.plot_topomap(ch_type='eeg', tmin=2, tmax=2.1, fmin=12, fmax=18, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis11[0], show=False)
powerfeedback1allright.plot_topomap(ch_type='eeg', tmin=2.1, tmax=2.2, fmin=12, fmax=18, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis11[1], show=False)

plt.suptitle('Right hand - Block 1',fontsize=20, X=0.1, y=1)
mne.viz.tight_layout()
plt.show()
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master
Thesis\plots\morlet\topomap\feedback\powerfeedback1b1alltopomapright')


#feedbackb5 right
fig12, axis12 = plt.subplots(1, 2, figsize=(30, 30))
powerfeedback5allright.plot_topomap(ch_type='eeg', tmin=2, tmax=2.1, fmin=12, fmax=18, vmin=vmin, vmax=vmax,
```

```
            baseline=baseline3, mode='percent', axes=axis12[0], show=False)
powerfeedback5allright.plot_topomap(ch_type='eeg', tmin=2.1, tmax=2.2, fmin=12, fmax=18, vmin=vmin, vmax=vmax,
            baseline=baseline3, mode='percent', axes=axis12[1], show=False)

plt.suptitle('Right hand - Block 5',fontsize=20, X=0.1, y=1)
mne.viz.tight_layout()
plt.show()
plt.savefig(r'C:\Users\daphn\OneDrive\Bureaublad\Master
Thesis\plots\morlet\topomap\feedback\powerfeedback5b1alltopomapright')
```