

RECOGNITION OF SEQUENTIAL HARMONIC AND INHARMONIC SOUND
SEQUENCES INVESTIGATED BY BEHAVIORAL AND ELECTROPHYSIOLOGICAL
STUDIES



FİLİZ TEZCAN SEMERCİ

JULY, 2020

RECOGNITION OF SEQUENTIAL HARMONIC AND INHARMONIC SOUND
SEQUENCES INVESTIGATED BY BEHAVIORAL AND ELECTROPHYSIOLOGICAL
STUDIES

BY

FİLİZ TEZCAN SEMERCİ



DISSERTATION SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
MASTER OF ARTS
IN
COGNITIVE SCIENCE

YEDİTEPE UNIVERSITY

JULY, 2020

PLAGIARISM

I hereby declare that information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Date: 22.04.2020

Name, Last name: Filiz TEZCAN SEMERCI

Signature:



ABSTRACT

Consonance rating when listening to the two consecutive or simultaneously presented sounds is associated with the ratio of their fundamental frequencies. The simpler ratio is perceived as more consonant. In this thesis, two melodies that consist of the same 10 tones were generated. Melodies had the same contour but differed with the order of tones so formed different harmonic relations between consecutive tones. Non musician participants were asked to choose which melody (harmonic or inharmonic tone sequences) was more consonant and the rating for each melody was not significantly different than each other. However, it was easier for them to notice the same amount of frequency change that occurs in one note of the harmonic sound sequence. Results suggests that better performance of the recognition task for harmonic sequence is related to better encoding and building a higher level expectation due to harmonic relations of sounds. An EEG experiment was conducted to test memory load hypothesis. The sustained anterior negativity (SAN) and power of alpha/beta oscillations were compared as a measure of working memory load. Amplitude of SAN did not reveal a difference in working memory load. However, beta and low gamma power was lower for harmonic sequences in the frontal cortex at the end of the stimulus presentation which suggests a more stable memory trace for harmonic sequences. Nonmatch stimulus elicited a P3b signal with higher amplitude for harmonic condition. This result suggests that a higher level expectation occurs for harmonic condition as all the other parameters that affect P3 signal were well controlled in the experiment.

Keywords: sequential harmony, expectation, recognition, working memory

ÖZET

Birbirini takip eden veya eş zamanlı olarak sunulan iki sesi dinlerken uyum derecesi, temel frekanslarının oranı ile ilişkilidir. Daha basit frekans oranına sahip sesler daha uyumlu olarak algılanır. Bu tezde, aynı 10 notadan oluşan ve aynı kontura sahip ancak notaların sunuluş sırası farklılık gösteren iki melodi üretildi. Böylece ardışık notalar arasında farklı harmonik ilişkiler oluşturuldu. Müzisyen olmayan katılımcılardan hangi melodinin (harmonik veya harmonik olmayan ses dizileri) daha uyumlu olduğunu seçmeleri istendi. Her iki melodinin tercih edilme oranının, birbirinden önemli ölçüde farklı olmadığı görüldü. Ancak, harmonik ses dizisinin bir notasında meydana gelen aynı miktardaki frekans değişikliğinin fark edilmesinin, harmonik olmayan ses dizilerine kıyasla daha kolay olduğu bulundu. Harmonik dizideki frekans değişikliğini fark etme performansının daha iyi olmasını, seslerin birbirleriyle olan harmonik ilişkilerinden dolayı beyinde daha iyi kodlanması ve duyulacak bir sonraki sesle ilgili daha yüksek seviye bir beklenti oluşturmasıyla ilişkili olabileceği varsayıldı. Bellek yükü hipotezini test etmek için bir EEG deneyi yapıldı ve sürekli ön negatifliği (SAN) genliği ve alfa / beta salınımlarının gücü, çalışma belleği yükünün bir ölçüsü olarak karşılaştırıldı. Her iki ses dizisi için SAN genliği, çalışma belleği yükünde bir fark ortaya koymadı. Ancak frontal korteksteki beta ve gama salınımı gücü, uyaran sunumunun sonlarında harmonik diziler için daha düşüktü. Bu da harmonik diziler için daha kararlı bir bellek izi olduğunu düşündürmektedir. Eşleşmeyen uyaranlar, harmonik durum için daha yüksek genliğe sahip bir P3b sinyalini ortaya çıkardı. Bu sonuç, P3 sinyalini etkileyen diğer tüm parametreler deneyde iyi kontrol edildiğinden, harmonik durum için daha yüksek bir seviye beklentinin oluştuğunu göstermektedir.

Anahtar kelimeler: Sıralı harmoni, beklenti, tanıma, çalışma belleği

ACKNOWLEDGEMENTS

The ability to create a model of the world in our minds and change our actions by making predictions about the reward on a long distant future is the most valuable gift. It took 11 years for me to update my prediction about the highest cumulative reward for my distant future. So the cost of the change was anxiety, excitement and low self confidence but it was worth all of it. I am grateful to all my friends who supported me for this decision of change, especially to İtir Kaşıkçı and Gökçer Eskikurt.

I would like to thank my thesis supervisor Funda Yıldırım for her patience while she was listening to my unsettled ideas, and her positive and motivating approach to my fear moments.

I would like to thank my mother Rukiye Tezcan, my father Mehmet Tezcan, and my sister Melek Tezcan. I am very lucky to have such wonderful parents who support me in every big decision in my life, and such a great sister who can talk me over for all my hesitations.

Finally, I would like to thank my husband, my best friend, and the Subject 0 of all my pilot experiments, Yusuf Can Semerci. Besides not being able to finish this thesis without his support, I even would not be able to start my academic career. No words are enough to express my gratitude to him for his never-ending effort to convince me that I am doing well.

TABLE OF CONTENTS

PLAGIARISM	i
ABSTRACT.....	ii
ÖZET	iii
ACKNOWLEDGEMENTS.....	iv
LIST OF TABLES	viii
LIST OF FIGURES.....	ix
1. INTRODUCTION.....	1
1.1. Auditory Perception	1
1.2. What is harmony in sounds?	3
1.2.1. Fundamental frequency harmonics	3
1.2.2. Dichotic listening experiments	6
1.2.3. Sequential harmonics	7
1.3. Working Memory.....	9
1.3.1. Chunking	9
1.3.2. Working memory for tonal harmony	10
1.3.3. Working memory load.....	11
1.3.4. Dynamic population coding.....	12
1.3.5. Mismatch Negativity	17
1.3.6. Predictive coding.....	18

2. EXPERIMENT 1: EFFECT OF SEQUENTIAL HARMONY ON RECOGNITION	
PERFORMANCE	23
2.1. Participants	23
2.2. Stimuli	23
2.3. Materials	24
2.4. Procedure	25
2.5. Results	27
2.6. Discussion.....	29
3. EXPERIMENT 2: EFFECT OF SEQUENTIAL HARMONY ON RECOGNITION	
PERFORMANCE WITH SELECTED FREQUENCIES	31
3.1. Participants	31
3.2. Stimuli and Procedure	31
3.3. Results	31
3.4. Discussion.....	33
4. EXPERIMENT 3: EFFECT OF SEQUENTIAL HARMONY ON AESTHETIC	
JUDGMENT	34
4.1. Participants	34
4.2. Stimuli and Procedure	34
4.3. Results	34
4.4. Discussion.....	35
5. EXPERIMENT 4: EFFECT OF SEQUENTIAL HARMONY ON NEURAL	
ACTIVATION	37

5.1. Participants	37
5.2. Stimuli and Procedure	37
5.3. Behavioral Results	38
5.4. Electrophysiological Recording and Statistical Analyses	38
5.5. Discussion.....	46
6. CONCLUSION.....	51
REFERENCES.....	57
APPENDIX A: CODES FOR EXPERIMENT.....	67
APPENDIX B: CODES FOR SOUND GENERATTION.....	86
APPENDIX C: CODES FOR EEG DATA ANALYSIS	96
APPENDIX D: ETICHAL APPROVAL FORM	112

LIST OF TABLES

Table 1. All conditions and modified notes..... 26

Table 2. Summary of conducted experiments..... 51



LIST OF FIGURES

Figure 1. Auditory neural pathway	2
Figure 2. Spectrogram of a violin sound with fundamental frequency of 497 Hz (University College London, n.d.).....	4
Figure 3. Train time vs test time accuracy matrix (Taken from Stokes, 2015)	14
Figure 4. Frequencies of harmonic (upper) and inharmonic (lower) sequences.....	24
Figure 5. Frequencies of modified harmonic (left) and modified inharmonic sequences (right) 1	24
Figure 6. Experimental Design	26
Figure 7. Recognition performance of harmonic and inharmonic sequences averaged over frequency	28
Figure 8. Recognition performance of harmonic and inharmonic sequences averaged over duration.....	29
Figure 9. Recognition performance of harmonic and inharmonic sequences.....	32
Figure 10. Consonance scores of harmonic and inharmonic sequences.	35
Figure 11. Experimental Design	38
Figure 12. Event related potentials of first melodies	41
Figure 13. Event related potentials of second (match) melodies	41
Figure 14. Event related potentials of second (nonmatch) melodies	42
Figure 15. Event related potentials of second match and nonmatch melodies	42
Figure 16. Difference ERPs of second match and nonmatch melodies	43

Figure 17. Topographies of positive cluster in harmonic difference ERP	43
Figure 18. Topographies of negative cluster of gamma power in harmonic first melodies (55-70 Hz $p=0.037$)	44
Figure 19. Topographies of (a) negative cluster and (b) positive cluster of beta power in harmonic first melodies	45
Figure 20. Topographies of positive cluster of alpha power in harmonic mismatch	45
Figure 21. Topographies of positive cluster of delta power harmonic mismatch condition	46

1. INTRODUCTION

Music is an ancient and common way to express feelings among all cultures and it is thought to be the origin of language according to some continuity based theories of language (Masataka, 2009). Due to the fact that emergence of music and language is dated to prehistoric times and that no direct evidence is left on the evolution of language and music, theories about the origin of both language and music are still controversial. Nevertheless, it is possible to observe some similarities between language and music in the way that they are perceived, in terms of their semantic components and acoustic features. Music is a well organized form of sound and silence. Studying the temporal and spectral modulations of music is considered to be more manageable compared to language. Therefore, understanding how musical acoustic components such as pitch height, pitch chroma, timbre, intensity and roughness interact with each other may shed light on understanding language processing. In this thesis, I will focus on the spectral relations and consonance of sequentially presented harmonic sounds. The aim of thesis was to investigate if there is a computational advantage in the processing of harmonic sequences.

1.1. Auditory Perception

Auditory signal is transferred from cochlea to cochlear nuclei (ventral and dorsal cochlear nucleus) on the brain stem. Basilar membrane in cochlea acts as frequency analyser. Hair cells which vibrate with the basilar membrane transform vibration into neural code (Helmholtz, 1863) Sound frequencies which are organized in a tonotopic way in cochlea (low frequencies are represented in inner cochlea, high frequencies in outer) are projected onto ventral and dorsal cochlear nuclei preserving the tonotopography. From cochlear nuclei, mainly three branches of nerves goes through different pathways (e.g. through superior olivary complex, inferior colliculi, medial geniculate body) and projects

on auditory cortex (AC) (Figure 1). Auditory pathway does not only have bottom-up but also top-down projections (Koelsch, 2011). In the primary auditory cortex (PAC), even though sound representations are more complex, tonotopography is still preserved. Low frequency sounds are represented in the rostro-lateral side of PAC and high frequencies are in the caudo-medial side.

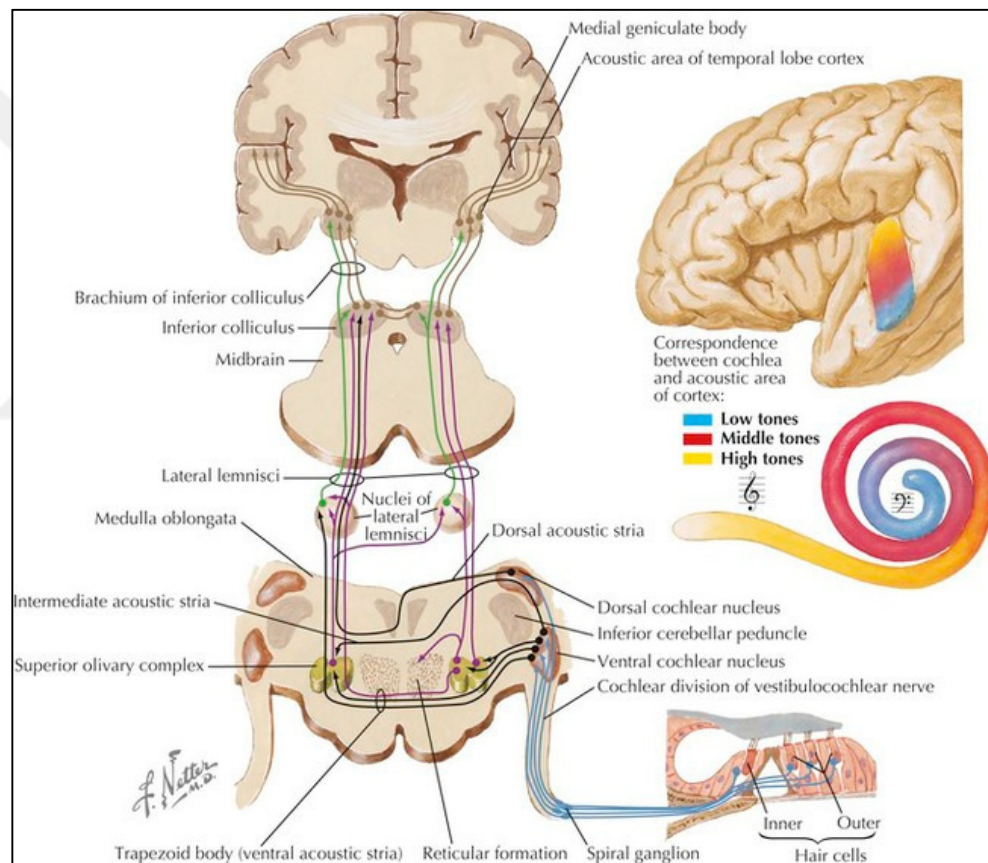


Figure 1. Auditory neural pathway (Cranial Nerve VIII | Clinical Gate (n.d.))

How and where these acoustic features are extracted is still not well understood. According to the Gestalten approach, extracted features form acoustic perception in the auditory memory. Auditory information that is processed according to Gestalt principles such as similarity, proximity and continuity is perceived as pitch, melody, timbre and rhythm (Denham and Winkler, 2014). Melodic and temporal intervals seem to be

processed independently as evident from some brain damage cases where pitch discrimination is interrupted but the interpretation of time relation is intact (Peretz, 1990). However, which brain regions are responsible for different types of interval processing is still not known well despite the effort of functional imaging studies. EEG and MEG studies show that violation of musical syntax in a melody elicits brain response with negative polarization on the right anterior frontal cortex which is called early right anterior negativity (ERAN). Detecting such irregularity requires computation of structural relationship between sequential events. According to the study of Koelsch and Siebel (2005), computation is more difficult for non harmonic tones because hearing harmonic notes is more common and non harmonic notes are unexpected because in nature most sounds are harmonic or quasi-harmonic.

1.2. What is harmony in sounds?

1.2.1. Fundamental frequency harmonics

Studies on sound harmony mostly focused on simultaneously presented sounds. In psychoacoustic literature, harmony generally refers to the frequency components of a complex sound. Pure tone is a sinusoidal sound wave with any specific frequency, phase, and amplitude. In nature almost all sounds are complex sounds which can be described as the mixture of many pure tones. It consists of signals with different frequencies which are multiples of a specific frequency that is called fundamental frequency (F_0). Multiples of the fundamental frequencies are called harmonics. In this thesis, this will be called *fundamental frequency harmonics*. If a tone has a fundamental frequency of 100 Hz, it also has frequencies of 200, 300, 400, ..., n Hz depending on the nature of the sound source. These are called the 1st, 2nd, 3rd, ... n^{th} fundamental frequency harmonics. In figure

2, spectrogram of a violin tone with fundamental frequency of 497 Hz is shown. First peak is the fundamental frequency and the following peaks are its harmonics.

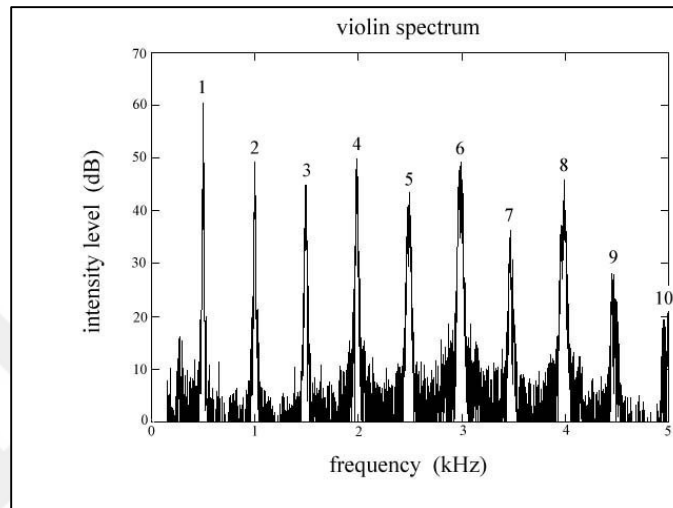


Figure 2. Spectrogram of a violin sound with fundamental frequency of 497 Hz (University College London, n.d.)

Fundamental frequency has an important role in pitch perception. When two notes have the same fundamental frequency but different numbers of harmonics (let us assume both note has a fundemantal frequency of 100 Hz; the first note has harmonic frequencies up to 500 Hz and the second tone up to 1000 Hz) they sound very similar and it is very difficult to differentiate them. Musical tones, among which their fundamental frequencies have a certain ratio, sound consonant when they are played simultaneously or in sequence. Simple ratios sound more consonant than complex ratios. When that ratio is 2:1, it is called octave, 3:2 perfect fifth and 4:3 perfect fourth. When the ratio gets more complex like in the tritone, 45:32, it sounds dissonant. Neurobiological models which attempt to explain harmony perception of simultaneously presented sounds suggest that sound waves with a simple ratio of frequencies are resolved better in the subcortical auditory system (Tramo et al., 2001, Bidelman and Heinz, 2011). These models assumed that the cochlea is doing an

autocorrelation calculation with the input signal to estimate the periodicity. It is comparing the signal itself with its different delayed versions. If the signal is periodic, correlation graph has a pick at a certain time delay which gives the period of signal. Behavioral experiments which tested the perceived pitch of complex tones consisting of three harmonics showed that periodicity is more important than the fundamental frequency. When a complex tone containing the fundamental frequency harmonics of 220, 320 and 420 Hz is presented, it is perceived as a tone close to 106 Hz pure tone, even though its missing fundamental frequency is 20 Hz. Autocorrelation function of such a signal gives a peak around delay time 9.4 ms which corresponds to 106 Hz. This shows that periodicity of the signal is more important than the fundamental frequency in determining the perceived pitch (Schnupp, et al., 2011). Periodic noise experiments also showed the same effect on pitch perception. Sound signals which are generated by repeating a short duration noise signal are called iterated repeated noise (IRN). Even though they have a wide range of frequency components, they create a pitch perception similar to a simple tone which has the frequency of repetition of the noise. Tramo et al. (2001) compared the autocorrelation functions of 4 musical dyads. Minor 2nd (A_4 440 Hz - 469 Hz $A_4^\#$), tritone (A_4 440 Hz - 619 Hz $D_5^\#$), perfect 5th (A_4 440 Hz - 660 Hz E_5) and perfect 4th (440 Hz A_4 - 587 Hz D_5). In the autocorrelation function of perfect 5th the first major peak corresponds to A_3 (220 Hz), the second major peak to A_2 (110 Hz). The first minor peak occurs at E_5 (660 Hz), then at A_4 (440 Hz), E_4 (330 Hz) and A_3 . All the corresponding peaks generated are either the octaves of the note or the fifth harmonics. Similar pattern is seen at the perfect 4th combination. However, it is not the case for tritone and minor 2nd (A_4 440 Hz - 469 Hz $A_4^\#$) combinations. For instance, in the autocorrelation function of the minor second, the largest peak occurs at 34.1 ms, which corresponds to a frequency of 29.3 Hz and it's below audible frequency limit so it creates roughness in perception. The first peaks occur at 2.20

ms, with the frequency of 455 Hz. This pitch does not correspond to any of the notes in any scale that has A4. Dyads which generate octaves or notes whose division with originating notes is a simple ratio creates regular periodicity and they are perceived as harmonic sounds.

1.2.2. Dichotic listening experiments

To eliminate the cochlear nonlinearities and to observe the neural representations of harmony perception on the cortex, dichotic listening experiments were conducted.

Researchers mostly worked with dyads of harmonic sounds and showed that musical pitch intervals are mapped on the cortex according to their perceptual features (Habibi et al., 2013; Itoh et al., 2010; Bidelman and Grall, 2014). Bidelman and Grall (2014) conducted an EEG experiment to understand if harmonic dyads and nonharmonic ones are represented in different locations in the brain. They used 13 chromatic dyads; unison (Un, $F_0 = 220$ Hz), minor 2nd (m2, 233 Hz), major 2nd (M2, 247 Hz), minor 3rd (m3, 262 Hz), major 3rd (M3, 277 Hz), perfect 4th (P4, 293 Hz), tritone (TT, 311 Hz), perfect 5th (P5, 330 Hz), minor 6th (m6, 349 Hz), major 6th (M6, 370 Hz), minor 7th (m7, 391 Hz), major 7th (M7, 415 Hz), octave (Oct, 440 Hz). Each tone of dyads was presented from different ears to prevent the peripheral confound of cochlear nonlinearity, roughness and beating. Each tone was generated by repeating a noise signal at a certain periodicity (IRN) to obtain the corresponding pitch. Dyads were rated by participants according to their consonance and the rating to each dyad was compared with the amplitude and latency of pitch onset response (POR) obtained from EEG data. Results showed that when the consonance rating increases, amplitude of POR increases and latency of POR decreases. They also calculated the possible sources of signals and showed that POR signals of harmonic dyads tend to generate on the lateral side of Heschl's gyrus compared to nonharmonic dyads. This study

showed that according to the amplitude, latency and source of the EEG signal it is possible to estimate the perception of consonance.

1.2.3. Sequential harmonics

Sequentially presented sounds with a simple ratio of frequency are also perceived as consonant. In this thesis, they will be called as *sequential harmonics*. Like dichotic listening experiments, sequentially presented tones also eliminates cochlear nonlinearities but when listeners are asked to recall sounds, the sounds need to be kept in the echoic and short term memory to be compared with the preceding tone. Micheyl and Oxenham (2010) suggested that the auditory system tends to group sounds which are presented simultaneously and are harmonically related. However, processing of sequentially presented and harmonically related sounds was not much studied.

Helmholtz's theory of beating harmonics (Helmholtz and Ellis, 1885) suggests that consonance of two simultaneously presented complex tones depends on the number of synchronized harmonics. When the fundamental frequencies of two complex tones have a ratio of 2:1 (octave) all harmonics are aligned perfectly so they sound consonant. When the ratio is complex like 16:15 harmonics are not aligned well and interacting nearby harmonics cause an unpleasant beating sensation. However this theory cannot explain the consonance of successively presented pure tones due to lack of synchronized harmonics of pure tones as they are played at different points in time.

Sensitivity to pitch relations with simple ratio and ease of neural processing is another explanation to harmony perception. In nature, sounds like wind, moving water are not periodic but animal vocalizations are periodic due to the nature of the vocal tract and it generates sounds with a fundamental frequency and its harmonics (Bowling & Purves 2015; Wang, 2013). Schwartz et. al (2003) analyzed a large database of human

vocalization. They divided audio signals into 100 ms segments and calculated the distribution of frequency ratios in segmented spectrum. They found that the human vocalization spectrum is also dominated with sound frequencies that have simple ratios. Exposure to speech from the early days of development may be the reason that the human brain is specialized in the processing of simultaneously presented harmonic sounds but like the problem of Helmholtz's theory, it still does not explain the sequential harmonics.

As an attempt to explain the nature of sequential harmony perception, Shapira Lots and Stone (2008) offered a computational model based on nonlinear oscillators. A neuron or population of neurons in the auditory cortex can be represented as nonlinear oscillators due to the excitatory and inhibitory connections with their neighbours. Each note in the melody creates a memory trace in the oscillating neural network. Neural resonance can arise from the interactive excitatory and inhibitory connections. In their model, they showed that coupled oscillators have more stable resonance regions when the frequency ratios of sequentially presented sounds are simple, so they form a more stable memory trace. A similar nonlinear oscillator model predicted the tonal stability of a melody in Western music according to calculated resonance stability in the nonlinear oscillators model. When the phase locking (resonance) of oscillators increased, melodies were perceived as more tonal (Large, 2010).

Besides its role on aesthetics judgment, sequential harmony has also a role in discrimination, learning and recognition performance. Schellenberg and Trehub (1994) showed that detecting changes in the melodic patterns, consisting of musical notes with simple frequency ratios that are sequentially presented, is easier compared to those with more complex ratios. They used melodies with 5 notes which have various sequential harmonic relations (e.g. 3:2 condition: 300 Hz, 450 Hz, 300 Hz, 450 Hz, 300 Hz and 45:32

condition: 300 Hz, 422 Hz, 300 Hz, 422 Hz, 300 Hz). They showed that detecting changes in melodies from simple ratios to complex ratios is easier than detecting changes from complex ratios to simpler ratios. However in their experimental design, change from perfect fifth to tritone was a 28 Hz decrement and change from tritone to perfect fifth was a 28 Hz increment. Frequency change in other conditions was also not equal. Recognition of descending intervals compared to ascending intervals and detection of smaller frequency change compared to higher frequency change is more difficult (Killam et al, 1975; Burns, 1999). Crespo-Bojorque and Toro (2016) compared the abstract rule learning performance of humans and rats with melodies which contain consonant and dissonant sequential harmonic relations. Advantageous processing of consonant tones was observed in abstract rule learning in humans but not in rats. It suggests that previous exposure to speech and music could play a role in how easily consonance perception is processed.

1.3. Working Memory

1.3.1. Chunking

Many studies on sequential harmonics tested the detection performance of participants where they need to recall the stimuli from their working memory. Working memory refers to the cognitive system where the information to be remembered is processed for seconds and manipulated for goal-directed behaviour, reasoning and decision making. It has a limited capacity but this can be improved by chunking the information (Miller, 1956; Gobet et al., 2001; Cowan, 2001) depending on the strength of the associations between items to be held in the memory. Miller (1956) explained the role of chunking with an example using a series of letters to be remembered; IBM CIA FBI. If the letters in the sequence are grouped into familiar abbreviations such as IBM, CIA and FBI it is easier to remember them. Most studies focused on chunking based on the long term

memory associations. However immediate memory may also chunk the series of information according to their common features. Chekaf et al. (2016) proposed 2 mechanisms for immediate chunking. Compressibility and presentation order. When the series of information can be simplified according to a rule of pattern, it is more compressible. The sequence of 2,3,4,5,6 can be remembered as numbers between 2 and 6. Presentation order is also important to extract this pattern. When the same numbers are presented in the order like 2, 4, 6, 3, 5 it is not easy to notice the same rule. To test the effects of these factors on working memory, they generated sequences of 8 simple visual objects by manipulating their features in three dimensions; shape, size and color (e.g., small white triangle, large white triangle, . . . , large black square). Sequences are grouped into 3 conditions based on the chunking strategies. 1) Rule condition: maximum similarity between 2 successive items and minimum similarity between local groups (two successive item is considered as a local group) which are formed according to the temporal categorization. 2) Similarity condition: maximum similarity through entire sequence to facilitate the one long chunk. 3) Dissimilarity condition: minimum similarity between items to hinder chunking process. They found that recall performance was best for the highest compressibility condition, especially when the presentation order facilitated the detecting the pattern in the sequence. In this thesis, I used sound sequences that have stronger consonance relations (simple ratios) that should be remembered better due to their higher compressibility and more stable memory trace.

1.3.2. Working memory for tonal harmony

Previous researchers who studied the effect of harmonic relations on working memory performance used tone sequences that are generated based on Western tonal music. In Western tonal music, harmonic relations are centered around a tonic note at the

beginning of the melody. A tonal melodic sequence has tones that are harmonically related to the tonic tone (key tone). The frequency ratio of the notes in the sequence with respect to the previous note is not always a simple ratio. Recognition task studies with tonal melodies resulted in contradictory conclusions. Dewar et. al (1977) found better performance for tonal sequences over atonal ones both with musicians and nonmusicians whereas Krumhansl (1979) found better performance for tonal sequences only with musicians. Schulze et. al (2011) found advantage of tonal sequences only with musicians for recognition of single test tone whereas they found performance advantage on tonal sequences with both musicians and nonmusicians for forward recognition task which was an easier task. Tonal and atonal sequences used in these experiments differ in terms of more abstract features than sequential harmonics and also compared sequences had different contours. Dowling (1978) showed that two melodies are relatively easy to distinguish if their contours are different. To test the effect of consonance with respect to the previous tone in a complex melodic sequence on memory performance, I generated harmonic and inharmonic sound sequences that consist of the same 10 tones. These tones had the same contour but differentiated as for their order, which forms different harmonic relations.

1.3.3. Working memory load

Researchers tried to understand the neural mechanism of working memory by studying the activity of individual neurons and neuron populations during the maintenance and recall of stimulus. They found some neurons on the lateral prefrontal cortex (PFC) had spiking rates above their baseline for a period of seconds after the stimuli offset (Funahashi et al. , 1989; Fuster and Alexander, 1971; Kubota and Niki 1971). This sustained activity is referred to as *delay spiking*.

A similar phenomenon is also observed in neural populations with M/EEG recordings. During a visual recognition memory task, it is shown that amplitude of sustained negativity during maintenance period on lateral occipital and posterior parietal regions increased when the number of items on the memory array increased (Vogel et al., 2004; Voytek et al, 2010). Working memory on nonverbal auditory working memory showed the similar sustained activity on anterior regions. Guimond et al. (2011) tested working memory load by simultaneously presenting the same pure tones to both ears (Load 1) and a different tone to each ear (Load 2). In an EEG experiment, they found higher amplitude of sustained anterior negativity (SAN) for higher memory load. Manipulating the memory load with the number of same tones which has different timbres showed the same correlation between the amplitude of SAN and memory load (Nolden et al., 2013). Lefebvre et al. (2013) showed that SAN is specific to auditory working memory by comparing the auditory memory task with a visual memory task. In all of the working memory studies mentioned up until here, memory load was manipulated by increasing the unit amount of information (total number of shapes on memory array, total number of tones) or the amount of novel information (number of tones among white noise or colored circles among black circles) in the memory. In this thesis, the information amount of conditions that the working memory load will be compared is kept intact but their compressibilities resulting from the strength of associations between successive items are different.

1.3.4. Dynamic population coding

For many years, delay spiking of small neuron groups which are tuned for specific stimuli is thought to be the neural basis of working memory. When the activity of neurons in a single trial is analysed, it was seen that activity of neurons varies over time, averaging

the activity over trials gives the trend of a stable delay activity (Miller et al., 2018). Brody et al. (2003) showed that approximately only 3% of delay active neurons have a stable activation throughout the delay. New studies with more complex analysis of larger neural population activity and computational models suggested that a sustained activity can be maintained in a dynamic neural population when different members of the population are active for a shorter time at different time points along the longer delay interval (Meyers, 2018). This model of working memory is called *dynamic population coding*. Dynamics of neural populations are analysed by cross-temporal analysis. A classifier is trained on the activity pattern of neurons at a specific time point and it is tested throughout all other time points on the delay period. A train time vs test time accuracy matrix is obtained. High accuracy on diagonal where training and test time aligned indicates that memory at each time point has different activity patterns because it is only possible to decode the activity pattern with the classifier at the trained time (Figure 3). On the other hand, cross-time generalisation which a classifier can classify the activity at other time points is a result of a stable activity pattern over time (Stokes, 2015). Instead of stable activity coding, having a dynamic population coding could facilitate the computation of elapsed time and also memory representations could be more resistant to external distraction. Biggest challenge with this model is that memory representation in neural activity at each time point needs to be read out by the downstream neural population with varying neural weights as activity at each time point has a different state (Sreenivasan, 2019). An alternative to this challenge would be to have static read out weights which integrates the information over delay period (Meyer, 2018). This means that information would be available to a subpopulation of neurons at one point in time, at the end of integrated time window. Another alternative is that there is a low-dimensional subspace that encodes information stably over time but stable activity neurons were only observed with simple tasks in which a single item is held

in memory whereas dynamic population coding was seen in more complex tasks with multiple items.

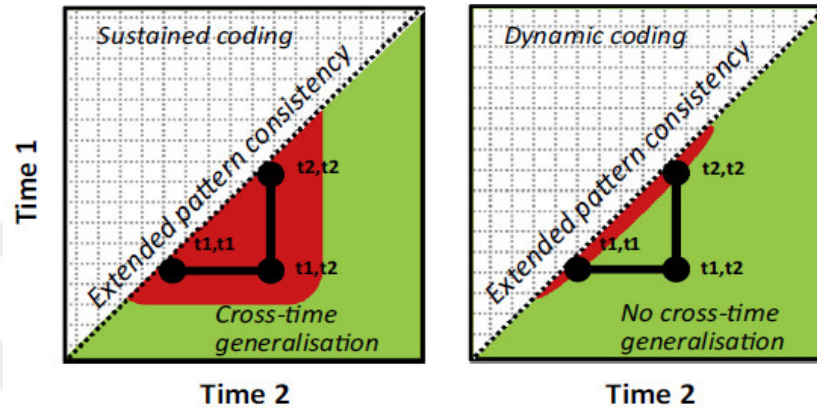


Figure 3. Train time vs test time accuracy matrix (Taken from Stokes, 2015)

Watanabe and Funahashi (2014) modified the standard memory guided saccade task that monkeys were trained to keep the location cue of a lever in their memory. They added an attention cue during the delay period so the memory cue was interrupted during the maintenance period. They observed that memory specific sustained activation abolished during competing tasks and then it was reactivated before the targeted behaviour. These results suggested that information can be kept in memory during activity silent periods in temporary synaptic memory traces. Short term synaptic plasticity (STSP) changes functional connectivity of the network to form a temporary task relevant memory trace (Mongillo et al., 2008). STSP suggests a working memory mechanism which does not require an explicit comparison of internal memory representations with the new input instead memory is stored with the synaptic connection weights like the long term memory. Another suggested mechanism for modulating the connectivity between neuron groups is phase-locking among the oscillations of communicating neuron populations. Fries (2005)

hypothesized - 'communication-through-coherence' (CTC) hypothesis- that phase-locked neuronal groups communicate more effectively because the excitable time window of receiving neurons are aligned with the output of sending neurons when both neuron groups are coherently oscillating. Dynamic population code proposes a state dependent neuronal activity. Activity of neurons are sensitive to the functional connectivity of the networks - hidden states - which dynamically changes according to the new input received by the network and new hidden states will determine the next neuronal activity. Interaction between hidden state and activity state results in a specific trajectory on state space. After a certain time of relaxation period, hidden states turn back to their baseline which is determined with long term plasticity. Same trajectory can be recreated with the same stimuli presented after the required relaxation time. Baseline hidden state can be altered with long term exposure to the stimuli. STSP can hold the memory in silent states arounds 2 seconds. For a longer duration of working memory, activity should be refreshed either with intrinsic oscillations or rehearsal (Stokes, 2015). Downside of this hypothesis is that hidden states cannot be measured directly because we cannot measure the connectivity weights of temporarily modified networks. Only way to test the hypothesis is observing the neural signals responding to well controlled inputs.

Lundqvist et al. (2011) extended a dynamic attractor network model which can hold sustained activity with intrinsic oscillation by recurrent connections with STSP. They increased the cellular adaptation and added synaptic augmentation so coding attractor states had a finite life time and they were reactivated after a refractory period. Dynamical attractors with silent period have an advantage of energy conservation because they do not need to have a continuous activation and also they are more resistant to memory interference due to the temporary synaptic changes and can hold multiple items. With their

model, they could reproduce power changes in brain oscillations in previous studies which showed increasing theta and gamma power and decreasing alpha/beta power with increasing working memory load during encoding and decoding stage (Mölle et al., 2002; Sederberg et al., 2003). Their network stored 6 memory items to be remembered. When an additional item is encoded to the memory, memory state is reactivated with theta frequency. Activation of each new oscillator occurs with a burst of gamma oscillation and the lifespan of the burst drives the slower oscillations. To test the predictions of their model they trained monkeys for a delay memory task that monkeys need to keep color and location of squares presented in a sequence while they recorded local field potentials (LFP) on lateral prefrontal cortex (PFC) and frontal eye field (FEF). They avoid cross trial averaging and performed trial-by-trial analysis to be able to observe dynamic behaviour of neural activation (Lundqvist et al., 2011). According to their model, encoding or decoding a stimulus triggers gamma oscillatory attractor state and suppresses default alpha/beta oscillatory state. First they identified gamma oscillatory local populations (modulated sites) and showed that gamma oscillations are highly correlated with the working memory related spiking and indeed suppresses beta oscillations. In addition, modulated sites showed stronger stimulus-locked 3-10 Hz power. These lower oscillations were related to the duration of each gamma burst and indicated the activation of each memory storing attractor. They also found brief irregular gamma bursts during delay time which may be needed to refresh the synaptic weights maintaining the memory. Gamma bursts increased close to the end of delay time around the time when memories are to be read out. Gamma and alpha/beta oscillations were anticorrelated during the encoding/decoding phase. They suggested that pull-push relations of gamma and alpha/beta oscillations may be regulating storage and cleaning of working memory. Decreased alpha/beta would enable sensory input to be encoded by turning the gamma up and increased alpha/beta would clean up the

memory by lowering the gamma. During the maintenance period, beta weakens to allow gamma bursts to occasionally refresh the synaptic weights. After the the information read out beta power increases again to clear the memory content.

1.3.5. Mismatch Negativity

Observing the environment and extracting the regular patterns to predict the future and adapt to change is critical for survival. There is increasing evidence that the brain is sensitive to statistical regularities of sequentially presented sensory inputs. The mismatch negativity (MMN) signal is a widely studied electrophysiological marker reflecting the violation of regularities in basic and abstract auditory stimuli features. It is elicited as a fronto-central negativity when the standard tone in the oddball paradigm is changed to a deviant tone and its latency is around 150-250 ms after the onset of the deviant tone. As reviewed in Naatanen et al. (2007), one interpretation of the cause of MMN is that deviant input is compared with the representation - memory template - of the standard stimuli. It is different from the N1 response because MMN cannot be elicited by the first sound of a sequence without any prior representation. It can neither be explained by new afferent neurons as decrement of intensity or absence of a sound in the sequence can also elicit MMN. The source of MMN is also different from N1. MMN is originating in the secondary auditory cortex and the frontal cortex while N1 originates from the primary auditory cortex (Pincze et al., 2001; Kropotov et al., 2000). Moreover, Javitt et al. (1996) showed the differential sensitivity of the MMN and N1 to N-methyl-D-aspartate (NMDA)-receptor antagonists. When NMDA receptors of monkeys were blocked, deviant tones did not elicit MMN but N1 generation remains.

When the deviant tone is different from any basic physical feature (frequency and volume) of the standard tone, on the unattentive condition, the amplitude of MMN was

found to be additive if more than one feature is deviant (Schröger, 1995). On the contrary, if the deviant tone pairs are different in more than one abstract feature (change in direction of frequency or intensity), it does not produce a larger MMN (Paavilainen et al., 2003). The results were interpreted by authors such that the processing of higher-order invariances differs from that of simple physical features.

Another view about the cause of MMN is that MMN and N1 have a similar mechanism resulting from the neuronal adaptation to frequent stimuli which is called Stimulus Specific Adaptation (SSA) (Khouri and Nelken, 2015). However, Ulanovsky et al. (2004) showed MMN does not only depend on the local history of the stimuli but also long term global history up to several seconds which cannot be explained by synaptic depression.

1.3.6. Predictive coding

As stated by Heilbron and Chait (2018) predictive coding (PC) account has a middle-ground position. Like the memory based model (MBM), PC considers the MMN as a comparison signal. However, unlike MBM this is a comparison between input and prediction and not the memory template. It does not consider MMN as a separate phenomenon than N1 but a contrast between expected and surprising response. The PC model suggests that, on hierarchically organized auditory cortex, abstracted information on the higher levels drives the activation on the lower level with feedback connections that send the predictions for the upcoming input. At the lower levels, prediction errors between the input and prediction are calculated and sent to the upper levels with feedforward connections to improve the next predictions (Rao and Ballard, 1999; Friston, 2005; Spratling, 2008; Knill and Pouget, 2004). There is an increasing amount of studies that the predictive coding approach can explain different aspects of MMN phenomena better than

adaptation. Costa-Faidella et al (2011) manipulated the interstimulus interval (ISI) in the rowing paradigm to obtain predicted and unpredicted ISI conditions which have the same average ISIs. They found larger MMN amplitude to a deviant stimulus in unpredictable timing condition. MEG studies, using repeated deviant tones (deviant AA letters in standard BB letters), showed that repetition suppression (RS) decreases when the deviant was unexpected (Wacongne et al., 2012; Todorovic et al., 2011). Wacongne et al. (2011) also investigated the neural response to local and global regularity violations with an EEG and MEG study. In the first condition, they presented xxxx sound sequence as standard and xxxY as deviant which Y sound served as both local and global deviant. In the second condition, they presented xxxY sound sequence as standard and xxxx as deviant which x served as global deviant but local standard. They found that local irregularity at the end of the xxxxY sequence in both conditions which either xxxxx or xxxxY sequences were global standards, elicited MMN but the amplitude was smaller in the second condition due to the global expectation. When they contrasted the responses to rare and frequent sequences they found a P3b (the latter component of P300) response reflecting the global novelty detection whose source was spreaded to prefrontal and parietal cortices. There was no negativity component as the last note of global deviant and local standard was the same. Predictions and prediction errors at higher levels on the hierarchy is thought to occur at longer latency through a wider network (Dehaene and Changeux, 2011). Thus, MMN is considered to be reflecting lower level and P300 is for higher level prediction errors.

Dürschmid et al. (2016) compared the high gamma activity to predicted and unpredicted deviant with ECoG recording and showed that only unpredicted deviant elicited high gamma activity at frontal electrodes whereas both deviants elicited activity at temporal electrodes. There are many other studies which correlated gamma oscillations

with prediction errors and beta oscillations with prediction change (ECoG: Chao et al., 2018; Sedley et al., 2016; MEG: Arnal et al., 2011; van Pelt et al., 2016; EEG: Lewis and Bastiaansen, 2015)

Manipulating expectations at the different hierarchical levels and developing methods different from the oddball paradigm is important to eliminate the adaptation effect and to understand the hierarchical role of prediction and prediction errors in the neural computation. Hsu et al. (2015) compared the amplitude of N1 signal elicited by predicted, mispredicted and unpredicted tones. Instead of repeating the same tone as standard stimuli such as in standard oddball paradigm, they used 5 ascending tones as standard stimulus. They replaced the last tone of the sequence with a tone that is one semitone lower than the first tone as the deviant tone. In the unpredicted condition, tones were ordered randomly. They found that predicted tone elicited lower N1 amplitude than mispredicted tone. However, the unpredicted tone's amplitude was the lowest. They concluded that predicted and mispredicted conditions revealed both prediction error and prediction signals but unpredicted condition only revealed the prediction error.

PC suggests that when a signal with regular patterns is more predictable, then its reliability is higher. Prediction errors are upweighted when the precision is high (Feldman and Friston, 2010). Precision is the inverse of uncertainty or the variance of distribution. Surprise is the divergence from the expected input. Attention can have different roles on regulation of prediction errors depending on the task. It can upregulate prediction errors (increases the weight of prediction error) when it acts as the higher precision. Alternatively, attention can downregulate prediction errors when it increases the expectation of change and thus the surprise decreases. Chennu et al. (2013) presented tone sequences to one ear and a deviant fifth tone to the same or the other ear. Attention is

modulated by asking participants to count the number of either deviant sequences (global deviant) or deviant tones (local deviant). In the control condition (no attention) participants attended to a distracting visual task. They found that MMN amplitude elicited by deviant tones increased, when participants were attending to sequences compared to no attention condition. However when participants attended to the deviant tones, MMN attenuated. Authors concluded that attending to the sequences increases the precision of prediction errors whereas attending to the tones increases the expectation of deviant tone so reduces the surprise.

In summary, the predictive coding account suggests 3 factors that affects the amplitude of MMN: (1) Distance between standard and deviant stimuli (e.g frequency difference between tones) (2) the variability of the standard stimuli (precision) (3) predictability of deviant due to the occurrence frequencies of standard and deviant stimuli. In this thesis, I compared two conditions (harmonic and inharmonic sequences) which have (1) the same frequency distance between standard and deviant stimuli (For both harmonic and inharmonic sequences, the frequency of the 4th note which corresponds to the same note, 337.8 Hz, was increased by 60 Hz to generate the deviant stimuli), (2) the same precision of standard stimuli with the same occurrence frequency for each note and (3) unpredictable deviant with the same occurrence frequency as the standard stimuli (50% standard - match, 50% deviant non-match). Only difference between the two conditions is that the harmonic sequence has a higher level regularity based on the harmonic relations between consecutive notes. Each note is either an ascending perfect 4th or descending perfect 5th of the previous note. The design of the experiment distangles all other factors except the higher level expectation on mismatch signal.

Aim of the Experiment 1 and 2 was to understand the effect of sequential harmony on recognition performance. In Experiment 3, consonance ratings of participants to harmonic and inharmonic sound sequences were compared. Lastly, in Experiment 4, electrical activity of participant's brain was measured by using an electroencephalogram (EEG) while they were performing a recognition task.



2. EXPERIMENT 1: EFFECT OF SEQUENTIAL HARMONY ON RECOGNITION PERFORMANCE

2.1. Participants

Thirty One participants between 19 and 22 years of age (mean = 19 years, SD = 0.5 years) attended the experiment. The participants reported no hearing problems and had no professional musical training. All experiments were approved by the Science-Ethics Committee of Yeditepe University.

2.2. Stimuli

The stimuli were generated in MATLAB (Mathworks, 2017) at a sampling rate of 44100. Each sound sequence consists of 10 pure tones. When generating the harmonic sequence, the frequency of the first tone in the sequence was set as one of the following frequencies which are equally spaced in logarithmic scale and not corresponding to any note in Western music; 285, 427.5, 641.25 and 961.87 Hz. The frequency of the next tone is calculated by multiplying the frequency with either (ascending perfect 4th) or 2/3 (descending perfect 5th). 126 sequences of the possible 2¹⁰ combinations were selected according to the following criteria; (1) Each frequency is used only once. (2) The ratio between the first and the last tone is between 3/2 and 2/3. Inharmonic sequences are generated by shuffling the order of tones in harmonic sequences. 1 of the 126x10! possible combinations met the following constraints; (1) The ratio between frequencies cannot be 4/3, 3/4, 3/2, 2/3 or 2) The ratio between the first and the last tone is between 3/2 and 2/3. (3) Initial frequency should be the same as in the harmonic sequence. (4) The contour of the sequence should be the same as the harmonic sequence. Modified versions of harmonic and inharmonic sequences are generated by increasing the frequency of 4th, 6th, 8th or 10th tone by 60 Hz. The duration of each tone was the same in the sequences and it was either

50 ms, 100 ms, 200 ms or 400 ms. Each sequence is repeated to have a total duration of 4 secs.

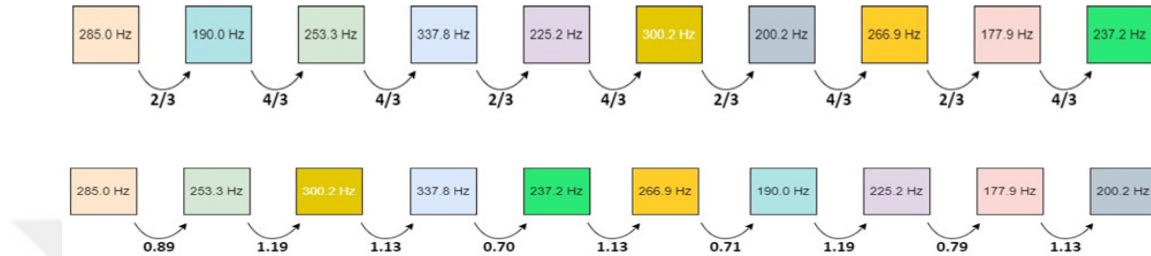


Figure 4. Frequencies of harmonic (upper) and inharmonic (lower) sequences

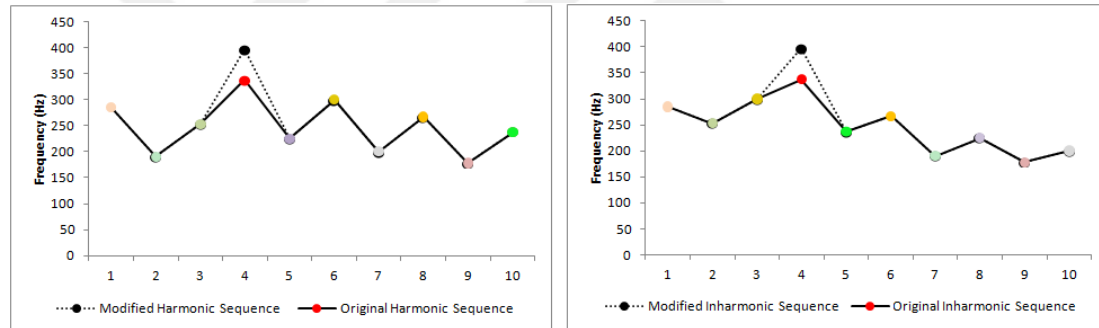


Figure 5. Frequencies of modified harmonic (left) and modified inharmonic sequences (right) 1

2.3. Materials

Participants sit 60 cm away from the computer screen and they fixated to the center of the screen throughout the session to maintain attention. Sounds are played with MATLAB Psychtoolbox (Brainard, 1997; Pelli, 1997; Kleiner et al, 2007) via SONY headphones with 60 dB sound pressure.

2.4. Procedure

The participants listened to one of the 32 (4 different stimuli for each of the 4 Duration x 2 Harmony conditions) harmonic or inharmonic sound sequences in each trial. After the first trial, they had 4 seconds of silent periods where they were instructed to rehearse the melody silently. Then they listened to 8 seconds of white noise proceeded with the second melody. The first melody was a harmonic sequence in half of the trials and inharmonic in the other half, which was selected randomly. The second melody was the same as the first melody for half of the trials and it was a modified version of the first melody for the other half. The order of trials from each condition was arranged randomly. The aim was to assess if the first melody and the second melody was the same or not. They were told to search for a difference in one of the tones in the melody. Participants responded by pressing button Y (yes) or N (No). The duration of each trial was 20 seconds and there were 64 trials in total. After every 16 trials they had a 1-minute break. Participants had a training session before the main experiment mentioned above. In the training session they received feedback as correct or false after their response. Training session had 16 trials in which the conditions were the same as the main experiment except for the frequencies of the notes in the sequences. Harmonic and inharmonic sequences started with a 1442.8 Hz tone that was not used in the main experiment to prevent familiarity effect. All conditions used in the experiment are shown on Table 1. Experimental design is illustrated in Figure 6.

Table 1.

All conditions and modified notes

Duration of each note	Freq. of Tone	1 st Modified Note	Duration of each note	Freq. of Tone	1 st Modified Note
50 ms	285 Hz	4th	200 ms	285 Hz	8th
	427.5 Hz	6th		427.5 Hz	10th
	641.25 Hz	8th		641.25 Hz	4th
	961.87 Hz	10th		961.87 Hz	6th
100 ms	285 Hz	6th	400 ms	285 Hz	10th
	427.5 Hz	8th		427.5 Hz	4th
	641.25 Hz	10th		641.25 Hz	6th
	961.87 Hz	4th		961.87 Hz	8th

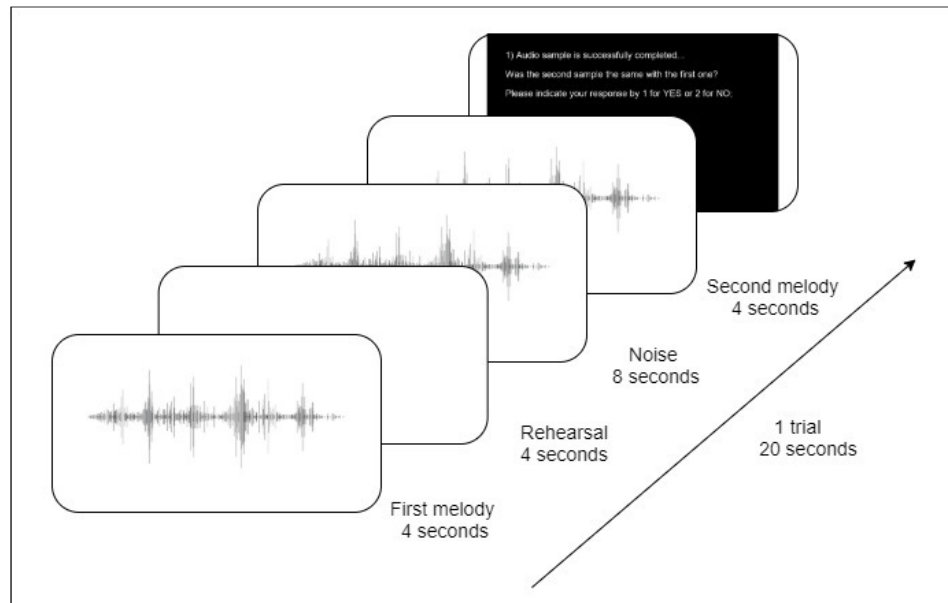


Figure 6. Experimental Design

2.5. Results

For each condition, d' values of each participant were calculated based on their hit rates and false alarm rates using signal detection theory (Elliot, 1964). Signal detection theory is used in psychophysics for more than 40 years to calculate the performance of the tasks such as discrimination, retrieval and decision making (Stanislaw and Todorov, 1999). It has an advantage over calculating average values for correct responses because it also takes the false alarms into account. When participants perform a memory task that they need to compare first and second stimulus for being identical or not, they can give a response as yes when they are the same (hit), as no when they are different (correct rejection) or as yes when they are different (miss) or no when they are the same (false alarm). d' values are calculated by subtracting Z-score of false alarm p-values from z-scores of hit rates. 0 and 1 false alarm/hit rate results in infinite d' values due to the sampling error arising from the limited number of trials. To overcome this problem, 0 values were replaced with $1/2N$ and 1 values were replaced with $1-1/2N$. One participant was excluded from analysis due to very low d' value (0.09) which indicates random answers. Maximum d' value with this correction for a perfect performance was 2.30.

1 subject was excluded from the analysis because the d' value of that subject was close to zero. For remaining 30 subjects, mean d' values for each condition were compared with repeated measure ANOVA test in SPSS (IBM Corp. Released 2013. IBM SPSS Statistics for Windows, Version 22.0. Armonk, NY: IBM Corp.). Significant main effects were found for both Duration and Harmony conditions. (Harmony $F(1,29)=16.630$, $p<0.0001$ and Duration $F(3,87)=5.779$, $p<0.01$). No interaction between Harmony and Duration was found (Harmony*Duration $F(3,87) = 0.497$, $p=0.685$). Recognition performance of harmonic sequences was better than inharmonic sequences (d' : 1.121 vs

0.611). Performance was increased as the duration of each tone was increased (d' : 50ms - 0.504, 100ms - 0.831, 200ms - 0.986, 400ms - 1.141). Pairwise comparison showed that performance of harmonic sequences was significantly better than inharmonic sequences when the duration was 100 ms, 200 ms and 400 ms. (50 ms: $p=0.062$, 100 ms: $p=0.002$, 200 ms: $p=0.039$ and 400 ms: $p=0.032$). Vertical bars on the figure below represent standard error of mean.

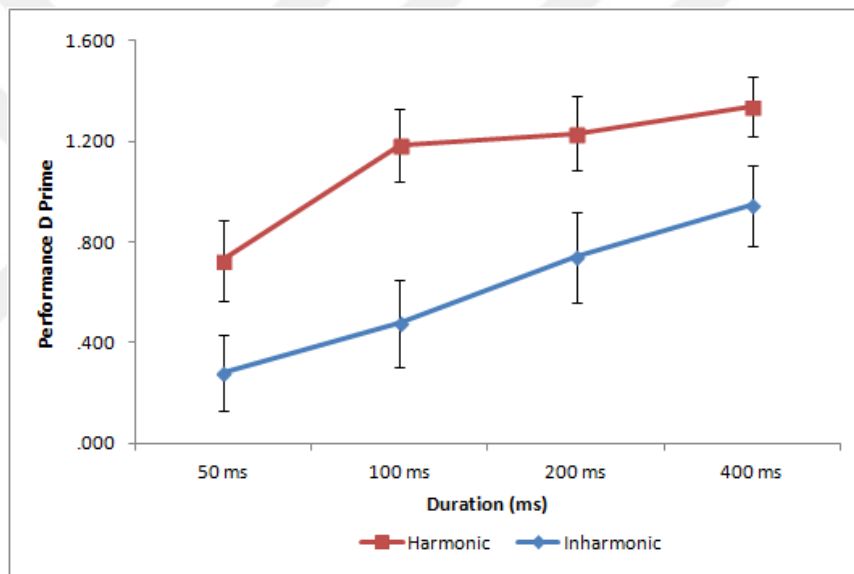


Figure 7. Recognition performance of harmonic and inharmonic sequences averaged over frequency

The response time (RT) to each condition was also analysed by averaging the RT to 4 different stimuli and matching/non matching cases. There was no significant RT difference for both Harmony ($F(1,29)=0.228$, $p=0.637$) and Duration ($F(3,87)=1.897$, $p=0.136$) conditions.

Lastly, to see if frequency of stimuli has an effect on recognition performance, another analysis was performed for Harmony vs Frequency condition. D' values were

calculated with the hit/false alarm rate of each stimulus with different durations. A significant main effect on Frequency condition was found ($F(3,87)=6.674$ $p<0.001$) but even though the mean d' value of harmonic sequence is higher than inharmonic, I could not find any significant main effect for Harmony condition (Harmonic d' : 1.053, Inharmonic d' : 0.873; $F(1,29)=1.94$, $p=0.173$). No interaction between Harmony and Frequency was found (Harmony*Frequency $F(3,87)=1.52$, $p=0.215$). Pairwise analysis showed that performance for harmonic sequence is only significantly better than inharmonic sequence in 427.5 Hz duration condition ($p=0.033$).

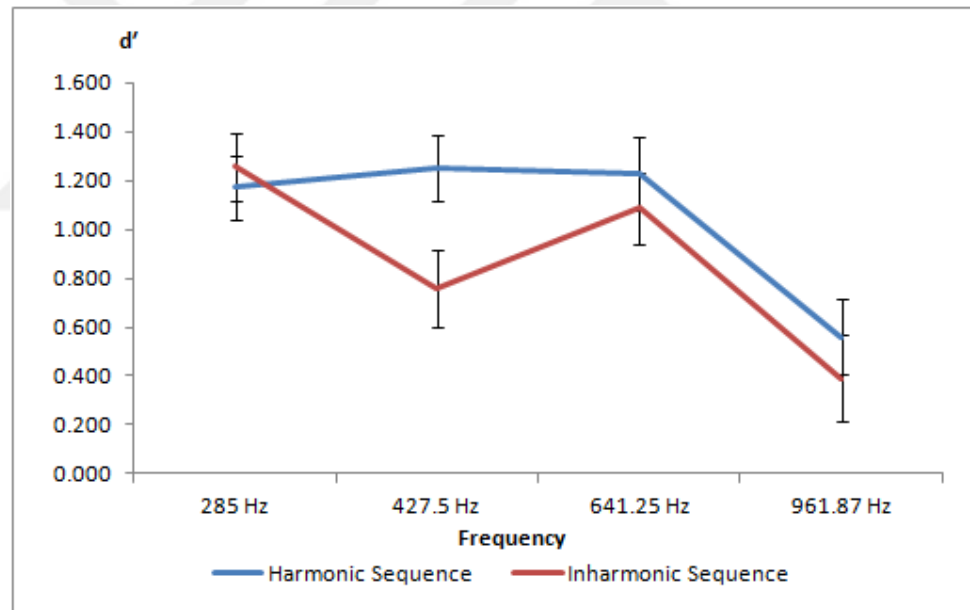


Figure 8. Recognition performance of harmonic and inharmonic sequences averaged over duration.

2.6. Discussion

Recognition performance was better for harmonic sequences even though the amount of frequency change and change direction was always the same for all conditions. If the

performance for detecting increment and decrement in frequency is different and it is a confound such as in Schellenberg and Trehub's (1994) study, I could eliminate it.

I changed the frequency of a different note (one of the 4th, 6th, 8th or 10th note) on each trial to prevent the effect of learning and searching the change on the same note. Order of modified notes on non match conditions are distributed evenly for each time and frequency conditions so that on each condition, all of the 4 notes are modified (Table 1). However, when performance according to frequency conditions was analysed, it was shown that performance was decreasing when the frequency was increasing. This may be due to the higher discrimination threshold of humans for higher frequencies (Wier et al., 1977). 60 Hz is quite above the discrimination threshold for the frequency range used in melodies; however, in many studies melodic intervals are determined according to frequency ratio, not to frequency difference (Schnupp 2011_Auditory Neuroscience). Modified notes in inharmonic condition (4th, 6th, 8th and 10th) have either same or lower frequencies than harmonic condition. This means that increment in terms of frequency ratio is higher in inharmonic condition so the discrimination performance should be better. This does not explain the better recognition performance for harmonic sequences. However, we decided to perform another experiment with a simpler design by keeping the order of modified notes the same for all conditions (4th note which corresponds to the frequency in both harmonic and inharmonic condition).

3. EXPERIMENT 2: EFFECT OF SEQUENTIAL HARMONY ON RECOGNITION PERFORMANCE WITH SELECTED FREQUENCIES

3.1. Participants

Thirty participants between 19 and 26 years of age (mean = 22.4 years, SD = 3.2 years) attended the experiment. The participants reported no hearing problems and had no professional musical training. All experiments were approved by the Science-Ethics Committee of Yeditepe University.

3.2. Stimuli and Procedure

Stimuli and procedure were similar to Experiment 1. To eliminate the possible confound from frequency and the order of modified note dependency, in Experiment 2, only one type of stimuli was presented (Sequences with initial frequency of 285 Hz) and in all trials 4th note of the sequences, which corresponds to 337.8 Hz in both harmonic and inharmonic sequences, was modified. Every stimulus from each of the 8 conditions (4 Duration x 2 Harmony) is presented 2 times in each session in a random order. Half of the trials had matching second melodies and the half was unmatching. Participants compared 64 melody couples totally. Due to Covid19, experiments were done online. Participants downloaded the Matlab executable file which is prepared with Psychtoolbox on their own computers. They used their own headphones. Instructions were given via Zoom.us. Participants were observed during the experiment from their cameras.

3.3. Results

For each condition, d' values of each participant were calculated with the same method in Experiment 1. Mean d' values for each condition were compared with repeated measure ANOVA test. Significant main effects were found for both Duration and

Harmony conditions in this experiment too. (Harmony $F(1,29)=9.286$ $p<0.01$ ($p=0.005$) and Duration $F(3,87)=6.059$, $p<0.01$ ($p=0.002$)). Recognition performance of harmonic sequences was better than inharmonic sequences (d' : 1.380 vs 0.729). Performance was increased as the duration of each tone was increased (d' : 50ms - 0.794, 100ms - 0.889, 200ms - 1.107, 400ms - 1.429). There was not any interaction between Harmony and Duration (Harmony*Duration $F(3,87)=0.658$, $p=0.580$). Pairwise comparison showed that performance of harmonic sequences was significantly better than inharmonic sequences when the duration was 200 ms. (50 ms: $p=0.087$, 100 ms: $p=0.066$, 200 ms: $p=0.001$ and 400 ms: $p=0.118$). Vertical lines on the figure below represent standard error of mean.

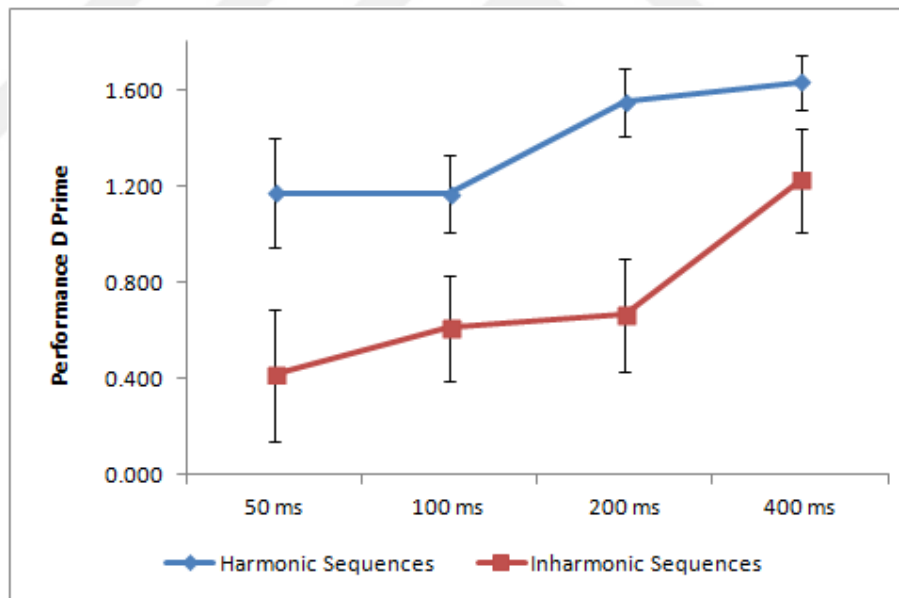


Figure 9. Recognition performance of harmonic and inharmonic sequences.

Response time (RT) to each condition was also analysed by averaging the RT to 4 different stimuli and matching/non matching cases. I could not find any significant RT difference for both Harmony ($F(1,29)=0.401$, $p=0.530$) and Duration ($F(3,87)=0.345$, $p=0.725$) conditions.

3.4. Discussion

In this experiment, only the 4th note which corresponds to the same frequency in both harmonic and inharmonic condition was modified to eliminate any confound from the order of the modified note. Mean performance for both conditions were higher than Experiment 1. This could be related to the learning effect due the repetition of the same stimuli and also expectation about which note will be changed. A significant main effect of Harmony was found but the performance difference between harmonic and inharmonic was less and it was only significant for the 200 ms duration.

4. EXPERIMENT 3: EFFECT OF SEQUENTIAL HARMONY ON AESTHETIC JUDGMENT

4.1. Participants

Thirty One participants between 19 and 44 years of age (mean = 23.5 years, SD = 4.9 years) attended the experiment. The participants reported no hearing problems and had no professional musical training. All experiments were approved by the Science-Ethics Committee of Yeditepe University.

4.2. Stimuli and Procedure

Stimuli were the similar Experiment 1 and 2 first melodies. In this experiment, participants compared the harmonic and inharmonic sound sequences and decided which one sound more consonant by pressing 1 if the first melody was more consonant and 2 if the second melody was chosen. There was 2 seconds silent interval between two melodies. The presentation order of harmonic and inharmonic melodies were randomized so half of the trials, first melody was a harmonic melody. They only compared the original sequences, so there was not any modified version of melodies. Participants compared 20 melody couples totally (4 different durations and 5 different frequencies). Due to Covid19, experiments were done online. Participants downloaded the Matlab executable file which is prepared with Psychtoolbox on their own computers. They used their own headphones. Instructions were given via Zoom.us. Participants were observed during the experiment from their cameras.

4.3. Results

For each condition, Harmony and Duration, I calculated the average scores. If a melody was chosen as more consonant, it had a score of 1, and if it was not chosen, the

score was 0. I did not find any significant main effects for both Duration and Harmony conditions and also no interaction between Harmony and Duration (Harmony $F(1,29)=2.374$, $p=0.134$; Duration $F(3,87)=0$; Harmony*Duration $F(3,87)=0.598$, $p=0.602$). Average score for each duration was always the same because participants could choose one of the stimuli for each trial so the total score was always equal to 1 and the mean score was 0.5. However, the trend of the mean scores for harmonic and inharmonic melodies showed an increasing difference as the duration increased. Mean score for harmonic sequences was higher than inharmonic ones (Harmonic=0.538, Inharmonic=0.462)

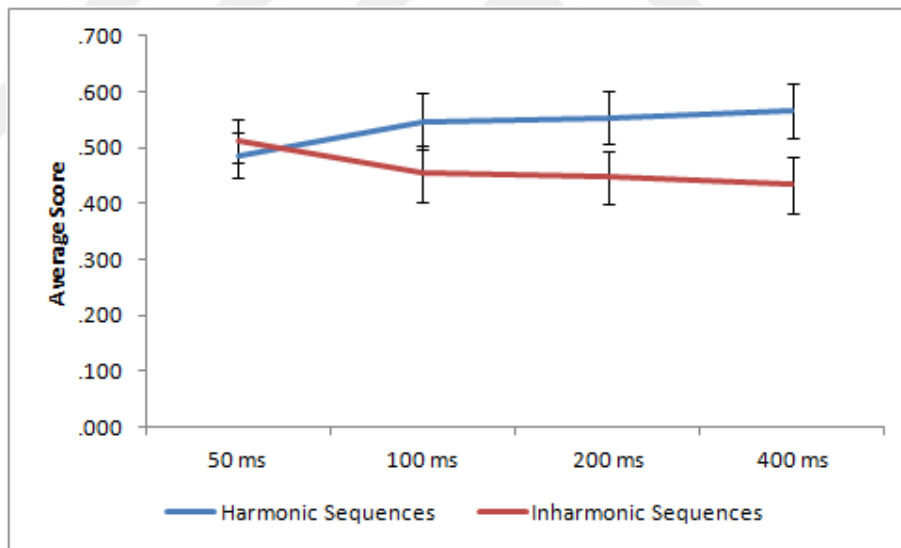


Figure 10. Consonance scores of harmonic and inharmonic sequences.

4.4. Discussion

Even though I found a trend of higher rating for harmonic sequences in 100, 200 and 400 ms durations, the difference between them was not significant. When notes were presented in pairs simultaneously or consecutively, perfect 5th and perfect 4th harmonic

relations were found to be more pleasing regardless of the participant's musical education. (Bidelman and Krishnan, 2009; Bidelman and Krishnan, 2011; McDermott et al., 2010). In this thesis, the notes in harmonic and inharmonic sequence do not have a silence period in between and they form higher level sound objects which have the same contour. When they were compared, a higher level feature like the contour may be dominating the perception so they can be perceived similarly. The results suggest that better recognition performance of harmonic sequences cannot be explained with its likability.

5. EXPERIMENT 4: EFFECT OF SEQUENTIAL HARMONY ON NEURAL ACTIVATION

5.1. Participants

Eleven subjects between 20 and 36 years of age (mean = 27 years, SD = 5.7 years) attended the experiment. The participants reported no hearing problems and had no professional musical training. All experiments were approved by the Science-Ethics Committee of Yeditepe University.

5.2. Stimuli and Procedure

Harmonic and inharmonic sequences which have 100 ms single note duration and initial frequency of 427.5 Hz were presented to participants while their EEG activity was recorded. In this experiment melodies were not repeated so they each lasted 1 second. Similar to Experiment 1 and 2, the task for participants was to decide whether the first and the second melody were the same or not. Each trial started with a 1 second noise presentation to eliminate any sensory processing remaining from the previous trial and to have an equal baseline for the first melody encoding. After the noise there was a 2 second silence interval, and then the first melody was presented for 1 second. There were 2 seconds silence period between the first and second melody and also after the second melody until the question screen appeared. Each harmonic and inharmonic sequence was presented 96 times in 4 sessions in a random order. Second melody was the same as the first melody for half of the trials (match condition) and it was modified for the rest of the trials (non-match condition). Figure 11 shows the EEG experimental design.

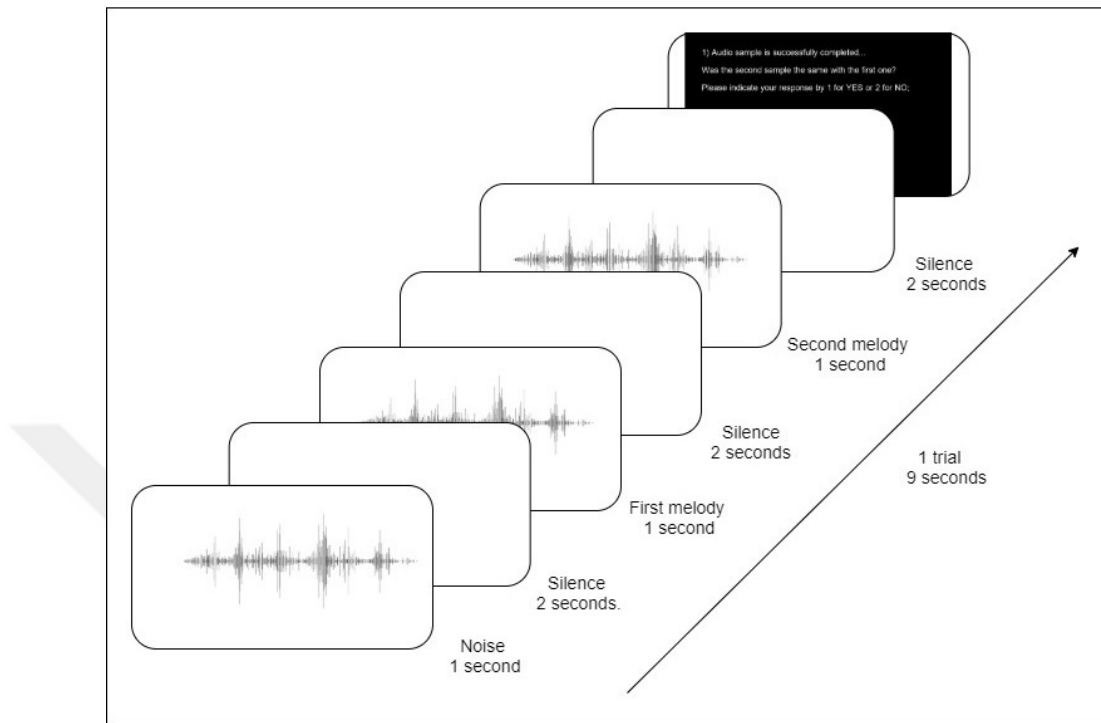


Figure 11. Experimental Design

5.3. Behavioral Results

Mean d' values of 11 participants were compared with related samples Wilcoxon signed rank test. Mean d' value for harmonic sequences was 1.30 and for inharmonic sequences 0.86 and the difference was statistically significant ($p=0.041$).

5.4. Electrophysiological Recording and Statistical Analyses

EEG was collected using the MRI-compatible BrainAmp EEG amplifier (Brain Products, Munich, Germany) and the EasyCap (Easycap, Falk Minow Services, Herrsching, Germany) electrode cap with sintered Ag/AgCl ring electrodes with 30 EEG channels and 1 EOG channel. The electrodes were positioned according to the 10/20 system. Raw EEG was sampled at 1 kHz with 70 Hz online low pass filtering and online reference to the average of both mastoids. Data was recorded using the Brain Vision

Recorder software (Brain Products). The impedance of all channels was held below 20 k Ω .

Preprocessing and statistical analysis of EEG data of all 11 subjects was done on MATLAB with the fieldtrip toolbox (Oostenveld et al, 2011). Raw data was high pass filtered at 0.3 Hz, lowpass filtered at 125 Hz and notch filtered at 50 Hz then it was segmented relative to the onset of presented stimuli including 500 ms prestimulus interval upto 3000 ms after the onset. Segments with EEG artifacts were rejected (any trial with a voltage change larger than 100 mV in a 50-ms interval). Independent components analysis (ICA) was applied and eyeblink components were removed. Clean segments were divided based on the following conditions:

- Harmonic_first: Harmonic sequence during encoding (first presented melody)
- Inharmonic_first: Inharmonic sequence during encoding (first presented melody)
- Harmonic_second_match: Harmonic match sequence during retrieval (Second presented melody)
- Inharmonic_second_match: Inharmonic match sequence during retrieval (Second presented melody)
- Harmonic_second_nonmatch: Modified (nonmatch) harmonic sequence during retrieval (Second presented melody)
- Inharmonic_second_nonmatch: Modified (nonmatch) inharmonic sequence during retrieval (Second presented melody)

Event related potentials (ERPs) for each participant were calculated for each condition by taking the average of all remaining clean trials. Then ERPs for all participants

were averaged. Figures 12, 13 and 14 show the average ERPs for 11 participants on anterior electrodes (Fp1, Fp2, F3, Fz, F4, FC3, FCz, and FC4).

Event related potentials of different conditions were statistically compared using nonparametric cluster-based permutation t-tests (Maris and Oostenveld, 2007) on Fieldtrip toolbox. When comparing many different EEG channels, signal frequencies or time points, there is risk of type 1 error (a false positive conclusion) due to multiple comparisons. Multiple comparison corrections like Bonferroni solves this problem by setting a smaller p value (e.g. dividing the p value of 0.05 with the number of comparisons). However this causes the type 2 error risk (a false negative conclusion) due to the high number of comparisons. Cluster-based permutation t-test is ideal for a M/EEG data which has many data points in different dimensions. It identifies significant clusters over time, frequency and/or channel space instead of comparing each of them pairwise.

Distributions are obtained by 5.000 random permutations. Both permutation p-value and cluster level p-value were taken as 0.05. For both encoding and retrieval periods, I did not find any significant difference on any time interval on any channel between harmonic and inharmonic sequences.

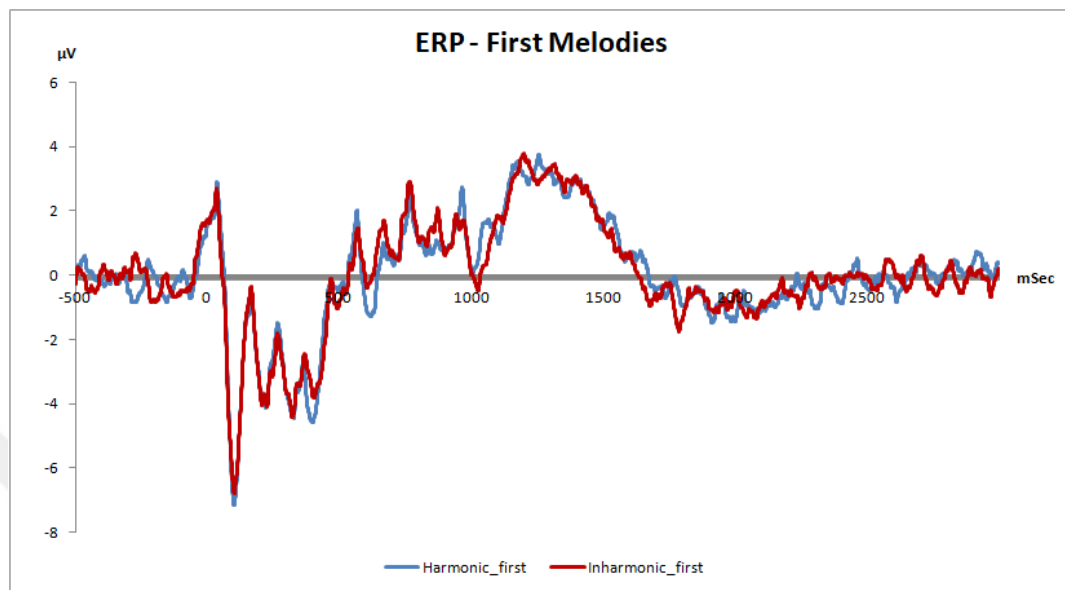


Figure 12. Event related potentials of first melodies

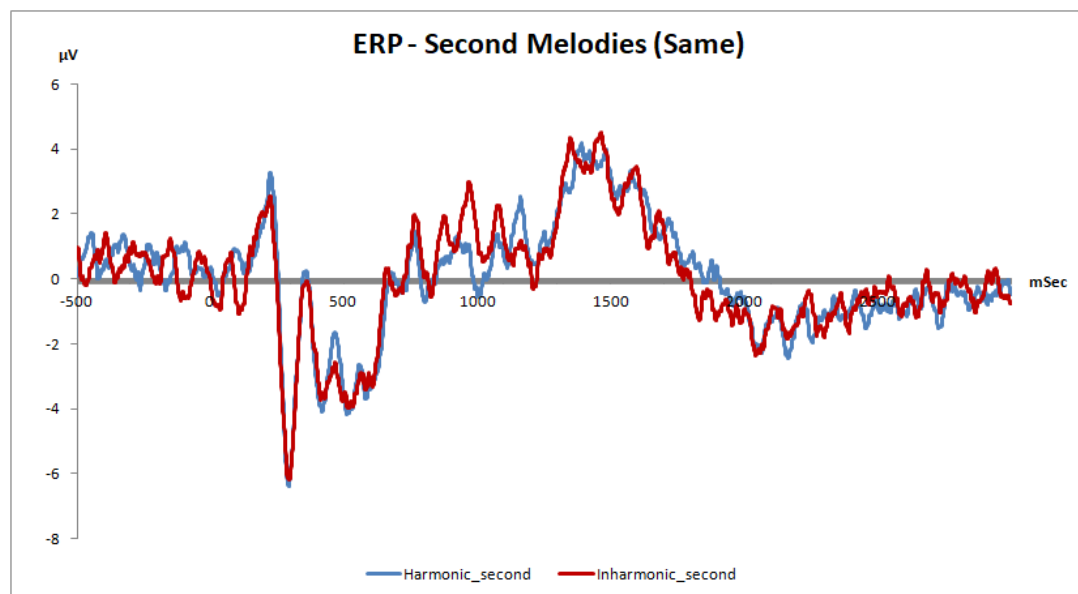


Figure 13. Event related potentials of second (match) melodies

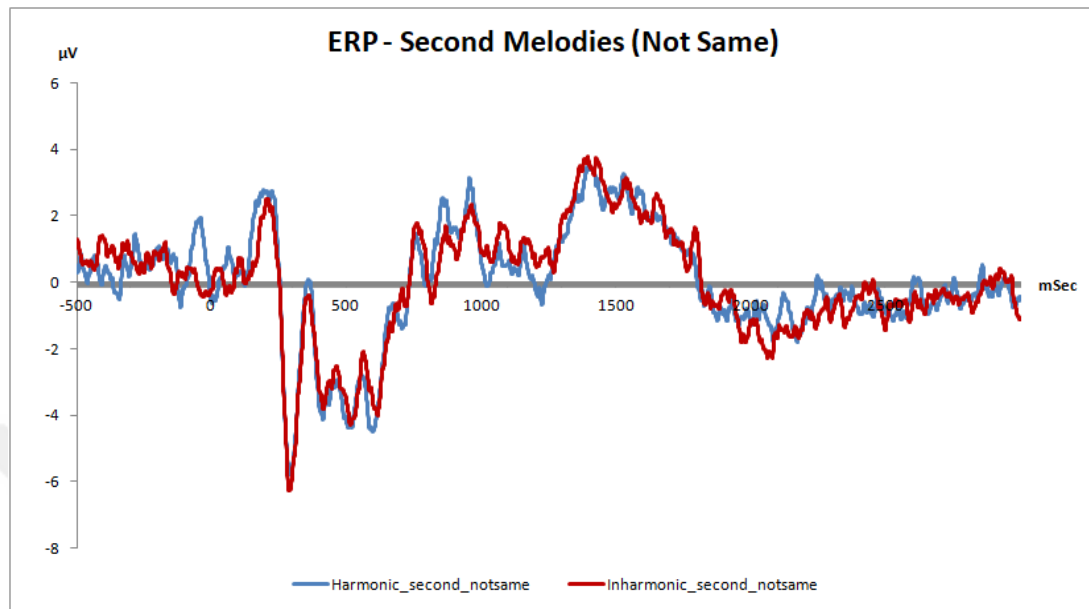


Figure 14. Event related potentials of second (nonmatch) melodies

I also compared the mismatch potentials during the retrieval phase by subtracting ERP of the deviant (second nonmatch) melody from standard (second match) melody. Figure 15 shows the ERPS averaged over right anterior and parietal channels of Harmonic_second_match / Harmonic_second_nonmatch and Inharmonic_second_match / Inharmonic_second_nonmatch conditions. In Figure 16, differences of match and nonmatch conditions for both harmonic and inharmonic melodies are shown.

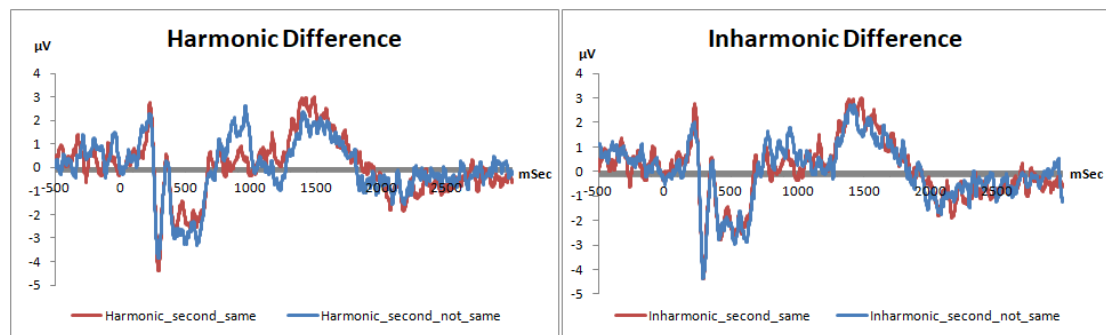


Figure 15. Event related potentials of second match and nonmatch melodies

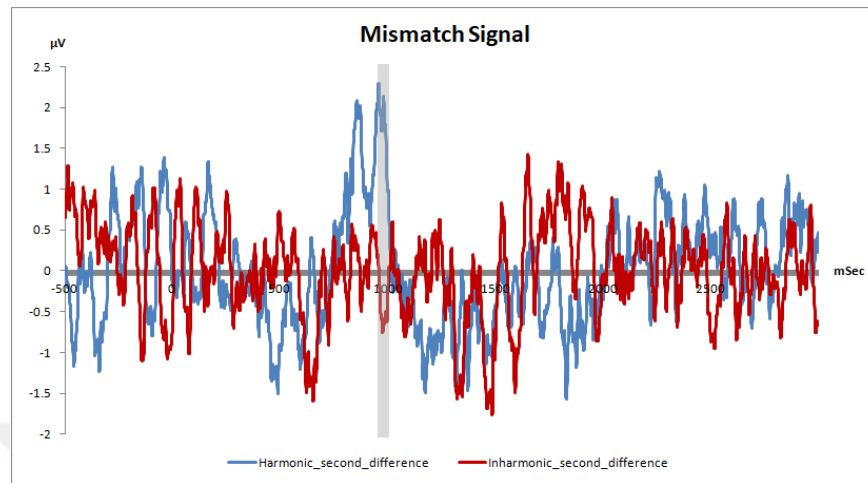


Figure 16. Difference ERPs of second match and nonmatch melodies

Visual evaluation of Figure 16 shows that harmonic melodies elicits an earlier negativity after the presentation of the deviant note ($t=400$ ms) and a greater positivity around 800 - 1000 ms after the onset of first note (around 400-600 ms after the deviant note). Statistical analysis showed that there is a significant positivity in harmonic condition between 950-1000 ms ($p=0.049$, $\Delta=1.83$ μV) on right anterior and parietal channels.

Figure 17 shows the topography of the significant positive cluster.

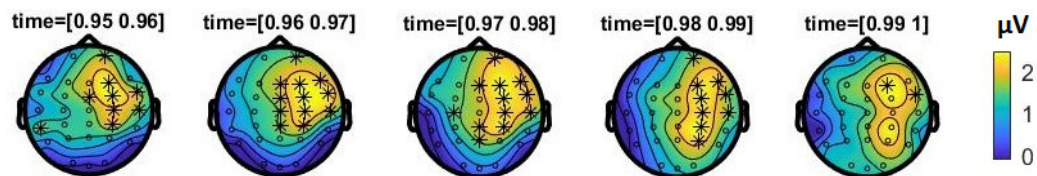


Figure 17. Topographies of positive cluster in harmonic difference ERP

Amplitude of harmonic and inharmonic mismatch signals between 950 – 1000 ms was compared to see the between subject variance using repeated measure ANOVA. There was not any significant variance between subjects ($F(1,10)=2.032$, $p=0.185$).

After ERP analysis, spectral analysis was also performed for each condition. Time–frequency representations (TFRs) were calculated using a Fourier transform with hanning window of 1000 ms. Time step was 100 ms and the frequency range was between 1-125 Hz. Calculated frequency powers were averaged between delta (1-3 Hz), theta (4-7 Hz), alpha (8-14 Hz), beta (15-32 Hz), low gamma (32-60 Hz) and high gamma (60-125 Hz) frequency bands. Averaged powers of each frequency band for each condition were compared using nonparametric cluster-based permutation t tests.

Firstly, the first melodies (encoding and maintenance period) were analysed. A negative cluster of gamma power (55-70 Hz, $p=0.037$) was found during the early maintenance period (between 1200 ms and 1500 ms after the onset of first tone) for harmonic sequences compared to inharmonic sequences.

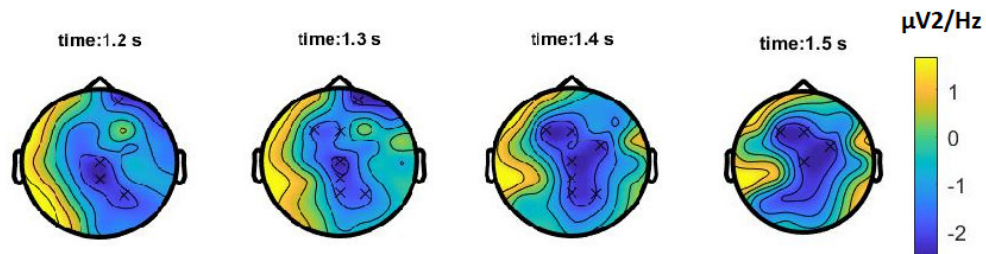


Figure 18. Topographies of negative cluster of gamma power in harmonic first melodies (55-70 Hz $p=0.037$)

Also, a negative cluster (Figure 18.a) of beta power was found around left anterior region in harmonic sequences during encoding (first melodies) between 900 - 1300 ms after the onset of first note ($p=0.042$) and a positive cluster (Figure 18.b) around parietal region between 2100 - 2400 ms ($p=0.016$).

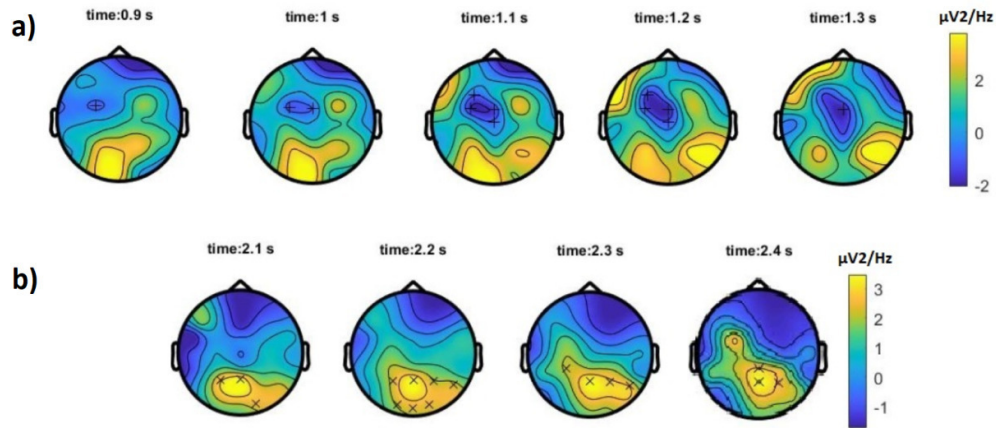


Figure 19. Topographies of (a) negative cluster and (b) positive cluster of beta power in harmonic first melodies

Then the spectral change for mismatch cases was analysed. The time-frequency response of Harmonic_second_match and Harmonic_second_nonmatch were compared and an alpha power decrease was found between 700 - 800 ms interval on anterior channels (Figure 19, $p=0.041$) for nonmatch condition. Same analysis for random second melodies did not give any significant spectral change..

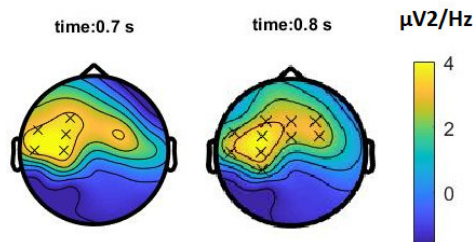


Figure 20. Topographies of positive cluster of alpha power in harmonic second match

Lastly, the spectral power difference of harmonic and inharmonic conditions (mismatch conditions) were compared by subtracting the spectral power of

second_nonmatch melodies from second_match and a significant positive delta power cluster was found for harmonic melodies between 400 and 1000 ms starting around left anterior channels and spreading to posterior regions (Figure 20, $p=0.0016$).

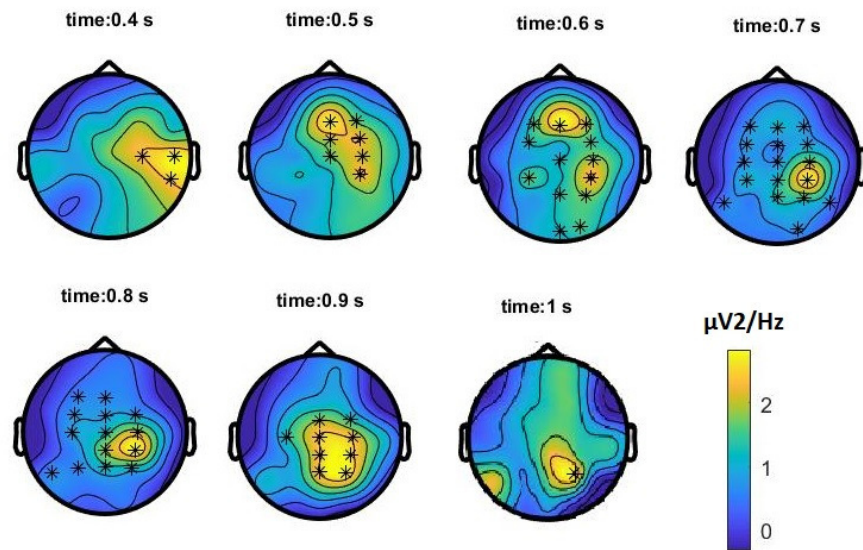


Figure 21. Topographies of positive cluster of delta power harmonic mismatch condition

To analyze the dependency of the delta power on P300 signal, average P300 signal between 250-500 ms after the onset of deviant tone was subtracted from the mismatch signal of each subject. Then delta power of harmonic mismatch and inharmonic mismatch conditions were compared after subtraction of P300 signals using cluster based permutation t-test. New analysis did not reveal any significant difference in delta band power ($p=0.108$).

5.5. Discussion

The EEG experiment was conducted to understand the neural mechanism behind the better recognition performance for harmonic sequences. Firstly the amplitudes of SAN of both conditions were compared as a marker of working memory load. When the working

memory load is higher a lower performance can be expected. However, during the maintenance period, SAN amplitudes of harmonic and inharmonic sequences are not significantly different (Figure 11). In previous studies which showed a higher SAN amplitude for higher memory load, number of items to be hold in the memory was increased (Visual: Vogel et al., 2004; Voytek et al, 2010, Auditory: Guimond et al., 2011; Nolden et al., 2013; Lefebvre et al. 2013). In this thesis, on both conditions the number of notes in the sequence was the same. It was also shown that delay activity diminishes due to learning when participants use the same search template (Carlisle et al., 2011). In the EEG experiment, the same stimulus was presented 96 times and always the 4th note was modified in the non match condition. Participants might have obtained a search strategy to only attend to the 4th note.

Then mismatched ERPs of harmonic and inharmonic sequences were compared by subtracting non match (deviant) trials from standard (match) trials during the retrieval phase (Figure 15). As the number of participants was limited to 11 people, average mismatch signal is very noisy but it is still possible to observe MMN and P300 components. Visual evaluation revealed a shorter latency of MMN for harmonic sequences but a significant amplitude or latency difference could not be found. A significant amplitude difference of P300 was found at later latencies (450-500 ms after the offset of deviant tone) which correspond to P3b signal. Topography of increased positivity in harmonic condition showed an activity on the right fronto-parietal region (Figure 16) in line with the previous findings (Dehaene et al., 1998; Dehaene and Changeux, 2011). As previous studies also suggested, higher amplitude of P300 in harmonic condition is thought to be related with the violation of a higher level expectation which is the harmonic relations between consecutive tones. Previous studies showed that MMN amplitude is

dependent on attention and expectation. As participants were asked to detect the change on the one note of the sequences, attention served as increased expectation of change so it decreased the surprise. Attenuated MMN could also have suppressed the effect of harmony. In a future study, changing the task to decrease the predictability of deviant tone, amplitude of MMN can be increased.

After ERP analysis, oscillatory dynamics when processing harmonic and inharmonic sequences were also compared. Spectro temporal analysis revealed alpha power decrease in harmonic_second_nonmatch compared to harmonic_second_match condition. Increased alpha power on frontal regions was associated with increased working memory load (Luo et al., 2005; Kaiser et al., 2007; Kawasaki et al., 2010; Sauseng et al., 2005, 2009). It is thought to reflect increased executive control or suppression of irrelevant information like beta power (Kaiser, 2015). A positive alpha power cluster in harmonic_second_match condition between 700 - 800 ms time interval shows a significant alpha power decrease in the nonmatch condition after 300 ms from the onset of deviant tone. On non match trials, memory load might have decreased after detection of the deviant tone because it is no longer needed to hold the sequence in memory. However, on the match condition participants still hold the first sequence until the end of the melody. The reason that the same significant difference in inharmonic condition was not observed could be that deviant note detection is more difficult than harmonic condition as behavioral performance and also the P300 amplitude suggests.

Spectral analysis of first melodies (encoding and maintenance period) showed that shortly after the offset of melodies, gamma power of inharmonic melodies becomes higher than harmonic ones (negative cluster of harmonic melodies) on the frontal region. Dynamic population coding suggests that to retain the information in memory occasional

gamma bursts are required to refresh the memory (Stokes, 2015). For inharmonic conditions more powerful refreshment could be required because memory traces for harmonic sequences are more stable due to resonating (phase locked) oscillations of neuron populations (Shapira Lots and Stone, 2008; Large et al, 2016). In line with the hypothesis that beta power is modulated with gamma bursts, beta power is also lower in harmonic condition in the early maintenance phase (Lundqvist et al., 2011). In the prefrontal region beta power is associated with inhibition (Bastos et al., 2018; van Ede et al., 2011). For inharmonic sequences more beta power is required to inhibit irrelevant information. In the parietal region, beta is more related with working memory maintenance (Kopell et al., 2011; Salazar et al., 2012). In the later stage of the delay time, beta power gets higher in the parietal region for the harmonic condition and there is no significant difference between gamma powers.

Both predictive coding account and dynamic population coding predicts gamma increase and beta decrease during the detection of deviant tone due to different mechanistic explanations. However there was no significant gamma or beta power change for both conditions. Due to the 70 Hz online low pass filtering, high gamma activity could not be recorded Lewis and Bastiaansen (2015) suggested that low gamma activity reflects the matching of top-down predictions with bottom up inputs and high gamma activity reflects bottom up precision errors which are transferred to higher levels to update the prediction. In our experiment, we would expect high gamma activity for deviant tone because it should reflect low level prediction errors.

Lastly, higher delta power in harmonic mismatch compared to inharmonic mismatch condition (Figure 20) after the onset of deviant tone until the offset of stimulus (between 400 ms -1000 ms) should be related with the higher amplitude of P300 signal for harmonic

condition as the same theta power difference could not be observed when P300 component was eliminated from the signal.



6. CONCLUSION

Sound dyads which have a simple frequency ratio are perceived consonant both when presented simultaneously and consecutively. Studies suggest that there is a preferential encoding for simultaneously presented harmonic sounds due to both physical properties of harmonic sounds and also being more exposed to harmonics sounds in nature. However, there are very limited studies on sequentially presented harmonic sounds. Most of the previous studies compared recognition performance for the tonal and atonal melodies which are generated according to Western tonal music. Those melodies differed in terms of more abstract features which are built up on melodic harmony and the results were contradictory. In this thesis, hypothesis was that recognition performance for harmonic melodies should be better with even nonmusicians when the compared melodies only differs in terms of harmonic relation between consecutive notes because harmonic relations between consecutive notes is a basic physical property which facilitates more efficient encoding in memory. Conducted experiments were summarized in Table 2.

Table 2.

Summary of conducted experiments

Experiment	Used Stimuli	Result
1: EFFECT OF SEQUENTIAL HARMONY ON RECOGNITION PERFORMANCE	D: 50, 100, 200, 400 F: 285, 427.5, 641.25, 961.87 M: 4th, 6th, 8th, 10th	Recognition performance is better for harmonic sequences. Performance is increasing as the duration increases and decreasing as the frequency increases.
2: EFFECT OF SEQUENTIAL HARMONY ON RECOGNITION PERFORMANCE WITH SELECTED FREQUENCIES	D: 50, 100, 200, 400 F: 427.5 M: 4th	Recognition performance is better for harmonic sequences. Performance is increasing as the duration increases.

3: EFFECT OF SEQUENTIAL HARMONY ON AESTHETIC JUDGMENT	D: 50, 100, 200, 400 F: 285, 427.5, 641.25, 961.87	There is a trend of higher scores for harmonic sequences but the difference is not statistically significant.
4: EFFECT OF SEQUENTIAL HARMONY ON NEURAL ACTIVATION	D: 100 F: 427.5 M: 4th	Recognition performance is better for harmonic sequences. ERPs and frequency powers were compared between harmonic and inharmonic sequences (see Chapter 5.5 for discussion).

* D: Duration (ms), F: Frequency (Hz), M: Modified Note

To compare the recognition performance of harmonic and inharmonic sound sequences, I generated two melodies which consist of 10 pure tones. Both melodies had the same initial note and contour; they only differed in terms of harmonic relations between consecutive notes. Behavioral experiments 1 and 2 (Chapter 2 and 3) showed that recognition was better for harmonic sequences than inharmonic sequences and the performance increased as the duration of individual note increased.

Sequential harmony perception is related with how sounds are represented and maintained in the working memory. Grouping the information according to a categorical relationship is an effective mechanism for memory storage. When the compressibility of the sequentially presented information is higher, memory performance is better (Chekaf et al., 2016; McDermott et al., 2020). Harmonic sequences in this thesis have a higher compressibility due the regular pattern of perfect 4th and perfect 5th relation between consecutive sounds throughout the melody. McPherson and McDermott (2020) showed that discrimination performance of two sounds is better when the sounds have harmonic spectra (complex sounds with fundamental frequency harmonics). They also concluded that compressibility of a sound with harmonic spectra facilitates efficient encoding in memory. Increasing recognition performance for both harmonic and inharmonic sequences with the increasing tone duration could be related to the same reason. Sequences with

longer tone duration were repeated less. If the melody of 400 ms tone duration is represented by the sequence “1 1 1 1 2 2 2 2 3 3 3 3”, 50 ms duration could be represented by “1 2 3 4 1 2 3 4 1 2 3 4”. Repetition of the melody brings another level of processing that causes beat perception. It increases the complexity so reduces the compressibility.

Lower performance for sequences with shorter note duration and with more melodic repetition can also be explained by global precedence. In vision studies, it is shown that global configuration of letters where letters are arranged to form a larger letter, shows more interference in detecting local letters. Participants are also faster to detect global letters (Navon, 1977). These findings suggest that perception of grouping is prioritized from global to local. In vision studies, global and local organization is defined by the spatial organization. Analogous studies were conducted in auditory domain and global and local organizations were defined by the temporal adjacencies. Wacongne et al. (2011) compared the MMNs to local and global deviants which are defined according to shorter or longer temporal patterns. In this thesis, higher repetition of the melody which had shorter individual tone duration could be analogous to forming a bigger global organization with smaller local details. Thus, it gets more difficult to detect a change in the local detail when the duration of each tone decreases even though the total duration of presentation time of each note does not change. Sanders and Poeppel (2007) conducted an auditory experiment where they generated congruent and incongruent sound sequences according to their local and global frequency modulation patterns. Congruent sequences had increasing or decreasing pitch direction both in local and global organization. (e.g. 100, 200 300, 150, 300, 450 Hz: both local and global frequency increase, or 300, 200, 100, 450, 300, 150 Hz: local frequency decrease, global frequency increase). Incongruent sequences had different directions in local and global organization (150, 300, 450, 100, 200 300 Hz or 450, 300,

150, 300, 200, 100 Hz). Participants were asked to detect if there was increase or decrease in global or local organization. They also found that performance for detecting the global direction was better than local direction for congruent and incongruent conditions. Performance for congruent sequences was also better than incongruent sequences when detecting both local and global change direction.

Harmonic sequences could also be forming a more stable memory trace as a result of the resonance of the neural populations that encode the frequencies of individual tones. For inharmonic sequences, memory trace is not stable due to the lack of phase synchronization between neural populations (Lots and Stone, 2008).

To investigate if the recognition performance advantage is related with the aesthetic preferences, third behavior experiment was conducted (Chapter 4). Consonance ratings for harmonic and inharmonic sequences were not significantly different even though ratings for harmonic sequences tend to be higher. Sequences were generated to have controlled features like frequency of each note and the contour of the melody. Having the same low and high level features might have caused the melodies to be perceived similarly and thus masked the aesthetical differences. Despite similar judgment of sequences regarding aesthetics, recognition performance was better for harmonic sequences (Chapter 4) so results cannot suggest that melodies which are liked more were remembered better.

EEG study was performed to investigate if there is any difference in electrical activity during encoding, maintenance and retrieval phases of harmonic and inharmonic sequences. Sustained Negativity in parietal/occipital and anterior regions was associated with working memory load in previous visual and auditory studies (Visual: Vogel et al., 2004; Voytek et al., 2010, Auditory: Guimond et al., 2011; Nolden et al., 2013; Lefebvre et al. 2013). In all these studies load was manipulated with the amount of information to be kept in the

memory. However, in this thesis both conditions had the same number of notes to be remembered. I could not find any difference in the amplitude of SAN even though recognition performance was better for harmonic sequences (Chapter 5).

New studies suggest that memory is not represented only with delayed activity of neurons tuned to the stimulus; instead dynamic activity of neuron populations enables the encoding and retrieval of information. Time domain analysis cannot reveal the dynamic relation between neuron populations, so I also performed time frequency analysis. In the frontal region, I found decreased beta power during maintenance period for harmonic sequences compared to inharmonic sequences (Chapter 5). In the literature, beta power in the frontal region is associated with inhibition of irrelevant information and gamma power is associated with bursts of activity to refresh the memory trace (Lundqvist et al., 2011; Miller et al., 2018) or with prediction errors (Chao et al., 2018; Sedley et al., 2016; Arnal et al., 2011; van Pelt et al., 2016; Lewis and Bastiaansen, 2015). Decreased beta and gamma power during early maintenance phase for harmonic sequences may be related with more stable memory trace or lower change in prediction so lower prediction errors.

When the mismatch signals (nonmatch - match condition in retrieval phase) were compared, harmonic conditions elicited a mismatch signal with higher P3b amplitude which indicates the violation of a higher level expectation.

Both behavioral experiments looking into recognition performance (Experiment 1 and 2) showed that stronger harmonic relation between consecutive tones improves the recognition performance. ERP and spectral analysis of EEG data suggest that better recognition performance is related with the higher level expectation of harmonic relations. Driving force of this expectation could be one of the following; (1) a more stable memory trace as a result of resonating neural populations, (2) stronger synaptic weights due to more

frequent exposure to harmonic sounds in nature or (3) ability to track statistical regularity of perfect 4th and perfect 5th relation.

To distangle these effects, higher level regularity of harmonic relations and harmonic relation strength should be controlled separately. To do that, two sequences with the same initial frequency and the same contour can be generated to repeat the more complex frequency ratios throughout the melody and have a random pattern of simple ratios (e.g. $4/3, 5/4, 2/3, 3/4, 4/5, 3/2$ vs $16/15, 8/9, 16/15, 8/9, 16/15, 8/9$). Surprisingness level of deviant sequences can also be manipulated to compare the effect of harmonic relations to N200 components of mismatch signals.

REFERENCES

- Arnal, L. H., Wyart, V., & Giraud, A. L. (2011). Transitions in neural oscillations reflect prediction errors generated in audiovisual speech. *Nature neuroscience*, 14(6), 797.
- Bidelman, G. M., & Heinz, M. G. (2011). Auditory-nerve responses predict pitch attributes related to musical consonance-dissonance for normal and impaired hearing. *The Journal of the Acoustical Society of America*, 130(3), 1488-1502.
- Bowling, D. L., & Purves, D. (2015). A biological rationale for musical consonance. *Proceedings of the National Academy of Sciences*, 112(36), 11155-11160.
- Burns, E. M. (1999). Intervals, scales, and tuning. In *The psychology of music* (pp. 215-264). Academic Press.
- Chao, Z. C., Takaura, K., Wang, L., Fujii, N., & Dehaene, S. (2018). Large-scale cortical networks for hierarchical prediction and prediction error in the primate brain. *Neuron*, 100(5), 1252-1266.
- Chekaf, M., Cowan, N., & Mathy, F. (2016). Chunk formation in immediate memory and how it relates to data compression. *Cognition*, 155, 96-107.
- Chennu, S., Noreika, V., Gueorguiev, D., Blenkmann, A., Kochen, S., Ibáñez, A., ... & Bekinschtein, T. A. (2013). Expectation and attention in hierarchical auditory prediction. *Journal of Neuroscience*, 33(27), 11194-11205.
- Cranial Nerve VIII | Clinical Gate (n.d.). *Afferent Auditory Pathways*. Retrieved June 20, 2020, from <http://www.clinicalgate.com/cranial-nerve-viii/>

- Costa-Faidella, J., Baldeweg, T., Grimm, S., & Escera, C. (2011). Interactions between “what” and “when” in the auditory system: temporal predictability enhances repetition suppression. *Journal of Neuroscience*, 31(50), 18590-18597.
- Cowan, N. (2001). The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and brain sciences*, 24(1), 87-114.
- Crespo-Bojorque, P., & Toro, J. M. (2016). Processing advantages for consonance: A comparison between rats (*Rattus norvegicus*) and humans (*Homo sapiens*). *Journal of Comparative Psychology*, 130(2), 97.
- Dehaene, S., & Changeux, J. P. (2011). Experimental and theoretical approaches to conscious processing. *Neuron*, 70(2), 200-227.
- Denham, S. L., & Winkler, I. (2014). *Auditory Perceptual Organisation*. In: Encyclopedia of Computational Neuroscience. SpringerReference.
- Dewar, K. M., Cuddy, L. L., & Mewhort, D. J. (1977). Recognition memory for single tones with and without context. *Journal of Experimental Psychology: Human Learning and Memory*, 3(1), 60.
- Dowling, W. J. (1978). Scale and contour: Two components of a theory of memory for melodies. *Psychological review*, 85(4), 341.
- Dürschmid, S., Edwards, E., Reichert, C., Dewar, C., Hinrichs, H., Heinze, H. J., ... & Knight, R. T. (2016). Hierarchy of prediction errors for auditory events in human temporal and frontal cortex. *Proceedings of the National Academy of Sciences*, 113(24), 6755-6760.

- Feldman, H., & Friston, K. (2010). Attention, uncertainty, and free-energy. *Frontiers in human neuroscience*, 4, 215.
- Fries, P. (2005). A mechanism for cognitive dynamics: neuronal communication through neuronal coherence. *Trends in cognitive sciences*, 9(10), 474-480.
- Friston, K. (2005). A theory of cortical responses. *Philosophical transactions of the Royal Society B: Biological sciences*, 360(1456), 815-836.
- Funahashi, S., Bruce, C. J., & Goldman-Rakic, P. S. (1989). Mnemonic coding of visual space in the monkey's dorsolateral prefrontal cortex. *Journal of neurophysiology*, 61(2), 331-349.
- Fuster, J. M., & Alexander, G. E. (1971). Neuron activity related to short-term memory. *Science*, 173(3997), 652-654.
- Gobet, F., Lane, P. C., Croker, S., Cheng, P. C., Jones, G., Oliver, I., & Pine, J. M. (2001). Chunking mechanisms in human learning. *Trends in cognitive sciences*, 5(6), 236-243.
- Guimond, S., Vachon, F., Nolden, S., Lefebvre, C., Grimault, S., & Jolicoeur, P. (2011). Electrophysiological correlates of the maintenance of the representation of pitch objects in acoustic short-term memory. *Psychophysiology*, 48(11), 1500-1509.
- Heilbron, M., & Chait, M. (2018). Great expectations: is there evidence for predictive coding in auditory cortex?. *Neuroscience*, 389, 54-73.
- Von Helmholtz, H. (1863). *Die Lehre von den Tonempfindungen als physiologische Grundlage für die Theorie der Musik*. Fr. Veeweg.

- Helmholtz, H. L., & Ellis, A. J. (1885). The sensations of tone: As a physiological basis for the theory of music, 6th ed., (pp. 290-309). New York, NY, US: Peter Smith, xix, 575 pp.
- Hsu, Y. F., Le Bars, S., Hämäläinen, J. A., & Waszak, F. (2015). Distinctive representation of mispredicted and unpredicted prediction errors in human electroencephalography. *Journal of Neuroscience*, 35(43), 14653-14660.
- Javitt, D. C., Steinschneider, M., Schroeder, C. E., & Arezzo, J. C. (1996). Role of cortical N-methyl-D-aspartate receptors in auditory sensory memory and mismatch negativity generation: implications for schizophrenia. *Proceedings of the National Academy of Sciences*, 93(21), 11962-11967.
- Khouri, L., & Nelken, I. (2015). Detecting the unexpected. *Current Opinion in Neurobiology*, 35, 142-147.
- Killam, R. N., Lorton, P. V., & Schubert, E. D. (1975). Interval recognition: Identification of harmonic and melodic intervals. *Journal of Music Theory*, 19(2), 212-234.
- Knill, D. C., & Pouget, A. (2004). The Bayesian brain: the role of uncertainty in neural coding and computation. *TRENDS in Neurosciences*, 27(12), 712-719.
- Koelsch, S., & Siebel, W. A. (2005). Towards a neural basis of music perception. *Trends in cognitive sciences*, 9(12), 578-584.
- Koelsch, S. (2011). Toward a neural basis of music perception—a review and updated model. *Frontiers in psychology*, 2, 110.

- Kropotov, J. D., Alho, K., Näätänen, R., Ponomarev, V. A., Kropotova, O. V., Anichkov, A. D., & Nechaev, V. B. (2000). Human auditory-cortex mechanisms of preattentive sound discrimination. *Neuroscience letters*, 280(2), 87-90.
- Krumhansl, C. L. (1979). The psychological representation of musical pitch in a tonal context. *Cognitive psychology*, 11(3), 346-374.
- Kubota, K., & Niki, H. (1971). Prefrontal cortical unit activity and delayed alternation performance in monkeys. *Journal of neurophysiology*, 34(3), 337-347.
- Large, E. W. (2010). Neurodynamics of music. In *Music perception* (pp. 201-231). Springer, New York, NY.
- Lefebvre, C., Vachon, F., Grimault, S., Thibault, J., Guimond, S., Peretz, I., ... & Jolicœur, P. (2013). Distinct electrophysiological indices of maintenance in auditory and visual short-term memory. *Neuropsychologia*, 51(13), 2939-2952.
- Lewis, A. G., & Bastiaansen, M. (2015). A predictive coding framework for rapid neural dynamics during sentence-level language comprehension. *Cortex*, 68, 155-168.
- Lundqvist, M., Herman, P., & Lansner, A. (2011). Theta and gamma power increases and alpha/beta power decreases with memory load in an attractor network model. *Journal of cognitive neuroscience*, 23(10), 3008-3020.
- Masataka, N. (2009). The origins of language and the evolution of music: A comparative perspective. *Physics of Life Reviews*, 6(1), 11-22.
- McPherson, M. J., & McDermott, J. H. (2020). Efficient codes for memory determine pitch representations. *bioRxiv*.

- Meyers, E. M. (2018). Dynamic population coding and its relationship to working memory. *Journal of neurophysiology*, 120(5), 2260-2268.
- Micheyl, C., & Oxenham, A. J. (2010). Pitch, harmonicity and concurrent sound segregation: Psychoacoustical and neurophysiological findings. *Hearing research*, 266(1-2), 36-51.
- Miller, E. K., Lundqvist, M., & Bastos, A. M. (2018). Working Memory 2.0. *Neuron*, 100(2), 463-475.
- Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review*, 63(2), 81.
- Mongillo, G., Barak, O., & Tsodyks, M. (2008). Synaptic theory of working memory. *Science*, 319(5869), 1543-1546.
- Mölle, M., Marshall, L., Fehm, H. L., & Born, J. (2002). EEG theta synchronization conjoined with alpha desynchronization indicate intentional encoding. *European Journal of Neuroscience*, 15(5), 923-928.
- Näätänen, R., Paavilainen, P., Rinne, T., & Alho, K. (2007). The mismatch negativity (MMN) in basic research of central auditory processing
- Nolden, S., Bermudez, P., Alunni-Menichini, K., Lefebvre, C., Grimault, S., & Jolicoeur, P. (2013). Electrophysiological correlates of the retention of tones differing in timbre in auditory short-term memory. *Neuropsychologia*, 51(13), 2740-2746.
- Paavilainen, P., Mikkonen, M., Kilpeläinen, M., Lehtinen, R., Saarela, M., & Tapola, L. (2003). Evidence for the different additivity of the temporal and frontal generators

of mismatch negativity: a human auditory event-related potential study. *Neuroscience letters*, 349(2), 79-82.

Patel, A. D., & Iversen, J. R. (2007). The linguistic benefits of musical abilities. *Trends in cognitive sciences*, 11(9), 369-372.

Peretz, I. (1990). Processing of local and global musical information by unilateral brain-damaged patients. *Brain*, 113(4), 1185-1205.

Pincze, Z., Lakatos, P., Rajkai, C., Ulbert, I., & Karmos, G. (2001). Separation of mismatch negativity and the N1 wave in the auditory cortex of the cat: a topographic study. *Clinical Neurophysiology*, 112(5), 778-784.

Rao, R. P., & Ballard, D. H. (1999). Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1), 79-87.

Sanders, L. D., & Poeppel, D. (2007). Local and global auditory processing: Behavioral and ERP evidence. *Neuropsychologia*, 45(6), 1172-1186.

Schellenberg, E. G., & Trehub, S. E. (1994). Frequency ratios and the discrimination of pure tone sequences. *Perception & Psychophysics*, 56(4), 472-478.

Schnupp, J., Nelken, I., & King, A. (2011). *Auditory neuroscience: Making sense of sound*. MIT press.

Schröger, E. (1995). Processing of auditory deviants with changes in one versus two stimulus dimensions. *Psychophysiology*, 32(1), 55-65.

- Schulze, K., Müller, K., & Koelsch, S. (2011). Neural correlates of strategy use during auditory working memory in musicians and non-musicians. *European Journal of Neuroscience*, 33(1), 189-196.
- Schwartz, D. A., Howe, C. Q., & Purves, D. (2003). The statistical structure of human speech sounds predicts musical universals. *Journal of Neuroscience*, 23(18), 7160-7168.
- Sederberg, P. B., Kahana, M. J., Howard, M. W., Donner, E. J., & Madsen, J. R. (2003). Theta and gamma oscillations during encoding predict subsequent recall. *Journal of Neuroscience*, 23(34), 10809-10814.
- Sedley, W., Gander, P. E., Kumar, S., Kovach, C. K., Oya, H., Kawasaki, H., ... & Griffiths, T. D. (2016). Neural signatures of perceptual inference. *Elife*, 5, e11476.
- Shapira Lots, I., & Stone, L. (2008). Perception of musical consonance and dissonance: an outcome of neural synchronization. *Journal of The Royal Society Interface*, 5(29), 1429-1434.
- Spratling, M. W. (2008). Predictive coding as a model of biased competition in visual attention. *Vision research*, 48(12), 1391-1408.
- Sreenivasan, K. K., & D'Esposito, M. (2019). The what, where and how of delay activity. *Nature Reviews Neuroscience*, 20(8), 466-481.
- Stokes, M. G. (2015). 'Activity-silent' working memory in prefrontal cortex: a dynamic coding framework. *Trends in cognitive sciences*, 19(7), 394-405.

- Todorovic, A., van Ede, F., Maris, E., & de Lange, F. P. (2011). Prior expectation mediates neural adaptation to repeated sounds in the auditory cortex: an MEG study. *Journal of Neuroscience*, 31(25), 9118-9123.
- Tramo, M. J., Cariani, P. A., Delgutte, B., & Braida, L. D. (2001). Neurobiological foundations for the theory of harmony in western tonal music. *Annals of the New York Academy of Sciences*, 930(1), 92-116.
- Ulanovsky, N., Las, L., Farkas, D., & Nelken, I. (2004). Multiple time scales of adaptation in auditory cortex neurons. *Journal of Neuroscience*, 24(46), 10440-10453.
- University College London. (n.d.). *Harmonics in a violin tone*. CCP14. Retrieved June 20, 2020, from <http://www.ccp14.ac.uk/ccp/web-mirrors/isotropy/~stokesh/violin.html>
- van Pelt, S., Heil, L., Kwisthout, J., Ondobaka, S., van Rooij, I., & Bekkering, H. (2016). Beta-and gamma-band activity reflect predictive coding in the processing of causal events. *Social cognitive and affective neuroscience*, 11(6), 973-980.
- Vogel, E. K., & Machizawa, M. G. (2004). Neural activity predicts individual differences in visual working memory capacity. *Nature*, 428(6984), 748-751.
- Voytek, B., & Knight, R. T. (2010). Prefrontal cortex and basal ganglia contributions to visual working memory. *Proceedings of the National Academy of Sciences*, 107(42), 18167-18172.
- Wacongne, C., Changeux, J. P., & Dehaene, S. (2012). A neuronal model of predictive coding accounting for the mismatch negativity. *Journal of Neuroscience*, 32(11), 3665-3678.

Wacongne, C., Labyt, E., van Wassenhove, V., Bekinschtein, T., Naccache, L., & Dehaene, S. (2011). Evidence for a hierarchy of predictions and prediction errors in human cortex. *Proceedings of the National Academy of Sciences*, 108(51), 20754-20759.

Wang, X. (2013). The harmonic organization of the auditory cortex. *Frontiers in systems neuroscience*, 7, 114.

Watanabe, K., & Funahashi, S. (2014). Neural mechanisms of dual-task interference and cognitive capacity limitation in the prefrontal cortex. *Nature neuroscience*, 17(4), 601.

APPENDIX A: CODES FOR EXPERIMENT

EXPERIMENT 1:

```

function [waveData,F]=loadWave(wave)
    filename=char(strcat(wave, '.wav'))
    wavfilename =
    fullfile('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment3\Final_Set\
    Longs\', filename)
    [y, freq] = psychwavread(wavfilename);
    waveData = y';
    nrchannels = size(waveData , 1);
    F=freq;

end

clear all;
clc;
rng('shuffle');

Screen('Preference', 'SkipSyncTests', 1);
hashtable = zeros(1,64);
iteration = 1;
iteration_try = 1;
freq=44100;
bits=16;

filepath =
'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment3\Final_Set\Longs';
filename = 'wavenames_working memory_sondiff.txt';
file = fullfile(filepath, filename);
fea = fopen(file);
Names=textscan(fea,'%s');

filepath =
'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment3\Final_Set\Longs\Noise';
filename = 'noises.txt';
file = fullfile(filepath, filename);
fea = fopen(file);
Names_noise=textscan(fea,'%s');

filepath =
'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment3\Final_Set\Longs';
filename = 'wavenames_working memory_deneme.txt';
file = fullfile(filepath, filename);
fea = fopen(file);
Names_try=textscan(fea,'%s');

```

```

filepath =
'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment3\Final_Set\Longs';
filename2 = 'randomizer.txt';
file2 = fullfile(filepath, filename2);
fea2 = fopen(file2, 'wt');

filepath =
'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment3\Final_Set\Longs';
filename3 = 'resultfile.txt';
file3 = fullfile(filepath, filename3);
fea3 = fopen(file3, 'wt');

while iteration ~= 65

    randomizer = randi(64);

    try

        while hashtable(randomizer) == 1

            randomizer = randi(64);

        end

        fprintf(fea2, '%d', iteration);
        fprintf(fea2, ' ');
        fprintf(fea2, '%d', randomizer);
        fprintf(fea2, '\n');

        if iteration == 1
            whichScreen = max(Screen('Screens'));
            [theWindow, theRect] = Screen('OpenWindow', whichScreen);

            Screen(theWindow, 'FillRect', 0);
            Screen(theWindow, 'DrawText', 'Welcome to the experiment.', 50, 50, 255);
            Screen(theWindow, 'DrawText', 'You will be given 64 audio sample couples to listen
to.', 50, 100, 255);
            Screen(theWindow, 'DrawText', 'Please listen to the audio carefully and indicate if the
second sample is the same with the first sample', 50, 150, 255);
            Screen(theWindow, 'DrawText', 'You should indicate your response by pressing 1 (if yes)
or 2 (if no) from the keyboard.', 50, 200, 255);
            Screen(theWindow, 'DrawText', 'Press right arrow key to continue...', 50, 250, 255);
            Screen('Flip', theWindow);
            KbWait;
        end

        HideCursor;
        theX = round(theRect(RectRight) / 2);
        theY = round(theRect(RectBottom) / 2);
        Screen(theWindow, 'DrawText', '!', theX, theY, 255);
        Screen(theWindow, 'DrawText', 'Now you are listening the first sample.', 50, 50, 255);
        Screen('Flip', theWindow);

        InitializePsychSound;

```

```

try
    pahandle = PsychPortAudio('Open', [], [], 0, freq, nrchannels);
catch
    fprintf("\nCould not open device at wanted playback frequency of %i Hz. Will retry
with device default frequency.\n", freq);
    fprintf('Sound may sound a bit out of tune, ...\n\n');

    psychlasterror('reset');
    pahandle = PsychPortAudio('Open', [], [], 0, [], nrchannels);
end
selectedsample1=randomizer*2-1;
selectedsample2=randomizer*2;
audio=string(Names{1,1}{selectedsample1,1});
[waveData ,F]=loadWave(audio);
PsychPortAudio('FillBuffer', pahandle, waveData);
PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing first sample

WaitSecs(4);

Screen(theWindow,'DrawText', '.', theX, theY, 255);
Screen(theWindow,'DrawText', 'Please rehearse the sample!',50,50,255);
Screen('Flip', theWindow);
WaitSecs(4);
Screen(theWindow,'DrawText', '.', theX, theY, 255);
Screen(theWindow,'DrawText', 'You are listening the noise!',50,50,255);
Screen('Flip', theWindow);
noise_number=mod(randomizer,16)+1;
audio=string(Names_noise{1,1}{noise_number,1});
[waveData ,F]=loadWave(audio);
PsychPortAudio('FillBuffer', pahandle, waveData);
PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing noise
PsychPortAudio('FillBuffer', pahandle, waveData);
PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing noise

WaitSecs(4);

Screen(theWindow,'DrawText', '.', theX, theY, 255);
Screen(theWindow,'DrawText', 'Now you are listening the second sample.',50,50,255);
Screen('Flip', theWindow);
audio=string(Names{1,1}{selectedsample2,1});
[waveData ,F]=loadWave(audio);
PsychPortAudio('FillBuffer', pahandle, waveData);
PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing second sample
WaitSecs(4);

Screen(theWindow,'DrawText', strcat(int2str(iteration), ' Audio sample is successfully
completed...'),50,50,255);
Screen(theWindow,'DrawText', 'Was the second sample the same with the first
one?',50,100,255);
Screen(theWindow,'DrawText', 'Please indicate your response by 1 for YES or 2 for
NO;',50,150,255);
Screen('Flip', theWindow);

```

```

secondst0=GetSecs;

while 1
    KbWait;
    [ keyIsDown, seconds, keyCode ] = KbCheck;
    rt=seconds-secondst0;
    keyCode = find(keyCode, 1);
    if keyIsDown && ((keyCode == 49) || (keyCode == 50))
        pressedKey = KbName(keyCode);
        fprintf(fea3, string(randomizer));
        fprintf(fea3, ': ');

        if keyCode == 49
            fprintf(fea3, 'YES');
            if (randomizer<17)
                fprintf(fea3, '1');
            elseif(randomizer<33)
                fprintf(fea3, '0');
            elseif(randomizer<49)
                fprintf(fea3, '1');
            else
                fprintf(fea3, '0');
            end
        end

        if keyCode == 50
            fprintf(fea3, 'NO');
            if (randomizer<17)
                fprintf(fea3, '0');
            elseif(randomizer<33)
                fprintf(fea3, '1');
            elseif(randomizer<49)
                fprintf(fea3, '0');
            else
                fprintf(fea3, '1');
            end
        end

        fprintf(fea3, ' ');
        fprintf(fea3, num2str(rt));
        fprintf(fea3, '\n');
        break;
    end
    WaitSecs(2);
end

% Stop playback:
PsychPortAudio('Stop', pahandle);

% Close the audio device:
PsychPortAudio('Close', pahandle);

if iteration == 16 || iteration == 32 || iteration == 48

```

```

        Screen(theWindow,'DrawText', 'It is time for a break...',50,50,255);
        Screen(theWindow,'DrawText', 'Press right arrow to continue after the break
        ',50,100,255);
        Screen('Flip', theWindow);

        while 1
            KbWait;
            [ keyIsDown, seconds, keyCode ] = KbCheck;
            keyCode = find(keyCode, 1);

            if keyIsDown && (keyCode == 39)
                break;
            end
        end

        iteration = iteration + 1;
        hashtable(randomizer) = 1;
        continue;

    catch
        sca;
        psychrethrow(psychlasterror);
    end
    WaitSecs(2);
end
Screen('Close', theWindow);
fprintf(fea3, '\n');
fclose(fea);
fclose(fea2);
fclose(fea3);
fclose(fea);

```

EXPERIMENT 2:

```

function [waveData,F]=loadWave(wave)
    filename=char(strcat(wave,'.wav'))
    wavfilename =
    fullfile('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment4\Final_Set\Longs\',
    filename)
    [y, freq] = psychwavread(wavfilename);
    waveData = y';
    nrchannels = size(waveData , 1);
    F=freq;
end

clear all;
clc;
rng('shuffle');

Screen('Preference', 'SkipSyncTests', 1);
freq=44100;
bits=16;

```

```

filepath =
'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment4\Final_Set\Longs';
filename = 'wavenames_working memory_sondiff.txt';
file = fullfile(filepath, filename);
fea = fopen(file);
Names=textscan(fea,'%s');

filepath =
'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment4\Final_Set\Longs\Noise';
filename = 'noises.txt';
file = fullfile(filepath, filename);
fea = fopen(file);
Names_noise=textscan(fea,'%s');

filepath =
'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment4\Final_Set\Longs';
filename2 = 'randomizer.xlsx';
file2 = fullfile(filepath, filename2);

filepath =
'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment4\Final_Set\Longs';
filename3 = 'resultfile.txt';
file3 = fullfile(filepath, filename3);
fea3 = fopen(file3, 'wt');
for rd=1:1:4
    flag=1;
    while flag==1
        hashtable=[];
        hashtable = zeros(1,16);
        for rdi=1:1:16
            randomizer(rd,rdi) = randi(16);

            while hashtable(randomizer(rd,rdi)) == 1

                randomizer(rd,rdi) = randi(16);

            end
            hashtable(randomizer(rd,rdi)) = 1;
        end
    end

    for i=1:1:15
        if ((randomizer(rd,i)+4) || (randomizer(rd,i)+8) || randomizer(rd,i)-4 || (randomizer(rd,i)-8))
            == randomizer(rd,i+1)
                flag=1;
            else
                flag=0;
            end
        end
    end
end
end
xlswrite(file2,randomizer);

```

```

whichScreen = max(Screen('Screens'));
[theWindow, theRect] = Screen('OpenWindow', whichScreen);

Screen(theWindow, 'FillRect', 0);
Screen(theWindow, 'DrawText', 'Welcome to the experiment.', 50, 50, 255);
Screen(theWindow, 'DrawText', 'You will be given 64 audio sample couples to listen
to.', 50, 100, 255);
Screen(theWindow, 'DrawText', 'Please listen to the audio carefully and indicate if the
second sample is the same with the first sample', 50, 150, 255);
Screen(theWindow, 'DrawText', 'You should indicate your response by pressing 1 (if yes)
or 2 (if no) from the keyboard.', 50, 200, 255);
Screen(theWindow, 'DrawText', 'Press right arrow key to continue...', 50, 250, 255);
Screen('Flip', theWindow);
KbWait;

HideCursor;
theX = round(theRect(RectRight) / 2);
theY = round(theRect(RectBottom) / 2);

for rd=1:1:4
    for iteration=1:1:16

        try

            Screen(theWindow, 'DrawText', '!', theX, theY, 255);
            Screen(theWindow, 'DrawText', 'Now you are listening the first sample.', 50, 50, 255);
            Screen('Flip', theWindow);

            InitializePsychSound;

            try
                pahandle = PsychPortAudio('Open', [], [], 0, freq, nrchannels);
            catch
                fprintf('\nCould not open device at wanted playback frequency of %i Hz. Will retry
with device default frequency.\n', freq);
                fprintf('Sound may sound a bit out of tune, ...\n\n');

                psychlasterror('reset');
                pahandle = PsychPortAudio('Open', [], [], 0, [], nrchannels);
            end
            selectedsample1=randomizer(rd,iteration)*2-1;
            selectedsample2=randomizer(rd,iteration)*2;
            audio=string(NAMES{1,1}{selectedsample1,1});
            [waveData ,F]=loadWave(audio);

            PsychPortAudio('FillBuffer', pahandle, waveData);
            PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing first sample
            WaitSecs(4);

            Screen(theWindow, 'DrawText', '!', theX, theY, 255);
            Screen(theWindow, 'DrawText', 'Please rehearse the sample!', 50, 50, 255);
            Screen('Flip', theWindow);
            WaitSecs(4);

```

```

Screen(theWindow,'DrawText', '.', theX, theY, 255);
Screen(theWindow,'DrawText', 'You are listening the noise!',50,50,255);
Screen('Flip', theWindow);
noise_number=randomizer(rd,iteration);
audio=string(Names_noise{1,1}{1,1});
[waveData ,F]=loadWave(audio);
PsychPortAudio('FillBuffer', pahandle, waveData);
PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing noise
PsychPortAudio('FillBuffer', pahandle, waveData);
PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing noise

WaitSecs(4);

Screen(theWindow,'DrawText', '.', theX, theY, 255);
Screen(theWindow,'DrawText', 'Now you are listening the second sample.',50,50,255);
Screen('Flip', theWindow);
audio=string(Names{1,1}{selectedsample2,1});
[waveData ,F]=loadWave(audio);
PsychPortAudio('FillBuffer', pahandle, waveData);
PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing second sample
WaitSecs(4);

Screen(theWindow,'DrawText', strcat(int2str(iteration), ') Audio sample is successfully
completed...'),50,50,255);
Screen(theWindow,'DrawText', 'Was the second sample the same with the first
one?',50,100,255);
Screen(theWindow,'DrawText', 'Please indicate your response by 1 for YES or 2 for
NO;',50,150,255);
Screen('Flip', theWindow);

secondst0=GetSecs;

while 1
    KbWait;
    [ keyIsDown, seconds, keyCode ] = KbCheck;
    rt=seconds-secondst0;
    keyCode = find(keyCode, 1);
    if keyIsDown && ((keyCode == 49) || (keyCode == 50))
        pressedKey = KbName(keyCode);
        fprintf(fea3, string(randomizer(rd,iteration)));
        fprintf(fea3, ': ');

        if keyCode == 49
            fprintf(fea3, 'YES');
            if (randomizer(rd,iteration)<5)
                fprintf(fea3, ' 1');
            elseif (randomizer(rd,iteration)<9)
                fprintf(fea3, ' 0');
            elseif (randomizer(rd,iteration)<13)
                fprintf(fea3, ' 1');
            else
                fprintf(fea3, ' 0');
        end
    end
end

```

```

        end

    end

    if keyCode == 50
        fprintf(fea3, 'NO');
        if (randomizer(rd, iteration) < 5)
            fprintf(fea3, '0');
        elseif (randomizer(rd, iteration) < 9)
            fprintf(fea3, '1');
        elseif (randomizer(rd, iteration) < 13)
            fprintf(fea3, '0');
        else
            fprintf(fea3, '1');
        end
    end

    fprintf(fea3, ' ');
    fprintf(fea3, num2str(rt));
    fprintf(fea3, '\n');
    break;
end
WaitSecs(2);
end

% Stop playback:
PsychPortAudio('Stop', pahandle);

% Close the audio device:
PsychPortAudio('Close', pahandle);

if iteration == 16 || iteration == 32 || iteration == 48
    Screen(theWindow, 'DrawText', 'It is time for a break...', 50, 50, 255);
    Screen(theWindow, 'DrawText', 'Press right arrow to continue after the break', 50, 100, 255);
    Screen('Flip', theWindow);

    while 1
        KbWait;
        [ keyIsDown, seconds, keyCode ] = KbCheck;
        keyCode = find(keyCode, 1);

        if keyIsDown && (keyCode == 39)
            break;
        end
    end

    iteration = iteration + 1;

    continue;

catch
    sca;
end

```

```

        psychrethrow(psychlasterror);
    end
    WaitSecs(2);
end

    Screen(theWindow,'DrawText', 'It is time for a break...',50,50,255);
    Screen(theWindow,'DrawText', 'Press right arrow to continue after the break
.',50,100,255);
    Screen('Flip', theWindow);

    while 1
        KbWait;
        [ keyIsDown, seconds, keyCode ] = KbCheck;
        keyCode = find(keyCode, 1);

        if keyIsDown && (keyCode == 39)
            break;
        end
    end

end

    Screen('Close', theWindow);
    fclose(fea);
    fclose(fea2);
    fclose(fea3);
    fclose(fea);

```

EXPERIMENT 3:

```

function [waveData,F]=loadWave(wave)
    filename=char(strcat(wave,'.wav'));
    wavfilename = fullfile('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound
Experiment2\Final_Set\All\', filename)
    [y, freq] = psychwavread(wavfilename);
    waveData = y';
    nrchannels = size(waveData , 1);
    F=freq;

end

clear all;
clc;
rng('shuffle');

    Screen('Preference', 'SkipSyncTests', 1);
    hashtable = zeros(1,20);
    hashtablewhichfirst = zeros(1,20);
    iteration = 1;
    freq=44100;

```

```

bits=16;

filepath = 'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound Experiment2\Final_Set';
filename = 'wavenames.txt';
file = fullfile(filepath, filename);
fea = fopen(file);
Names=textscan(fea,'%s');

```

```

harmonicfirst=0;
randomfirst=0;

```

```

filepath = 'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound Experiment2\Final_Set\';
filename2 = 'randomizer.txt';
file2 = fullfile(filepath, filename2);
fea2 = fopen(file2, 'wt');

```

```

filepath = 'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound Experiment2\Final_Set\';
filename3 = 'resultfile.txt';
file3 = fullfile(filepath, filename3);
fea3 = fopen(file3, 'wt');

```

```

while iteration ~= 21

```

```

    randomizer = randi(20);
    whichfirst=randi(20);
    try

```

```

        while hashtable(randomizer) == 1

```

```

            randomizer = randi(20);

```

```

        end

```

```

        while hashtablewhichfirst(whichfirst) == 1
            whichfirst=randi(20);
        end

```

```

        if(whichfirst<11)
            selectedsample=randomizer*2-1;
            whichfirstno=1;
            whichsecondno=2;
        else
            selectedsample=randomizer*2;
            whichfirstno=2;
            whichsecondno=1;
        end

```

```

        fprintf(fea2, '%d', iteration);
        fprintf(fea2, ' ');
        fprintf(fea2, '%d', selectedsample);
        fprintf(fea2, '\n');
    end
end

```

```

if iteration == 1
    whichScreen = max(Screen('Screens'));
    [theWindow, theRect] = Screen('OpenWindow', whichScreen);

    Screen(theWindow, 'FillRect', 0);
    Screen(theWindow, 'DrawText', 'Welcome to the experiment.', 50, 50, 255);
    Screen(theWindow, 'DrawText', 'You will be given 40 audio sample couples to listen
to.', 50, 100, 255);
    Screen(theWindow, 'DrawText', 'Please listen to the audio carefully and indicate which of
the samples sounds more harmonic to you', 50, 150, 255);
    Screen(theWindow, 'DrawText', 'You should indicate your response by pressing 1 (First
audio sample) or 2 (Second audio sample) from the keyboard.', 50, 200, 255);
    Screen(theWindow, 'DrawText', 'Press right arrow key to continue...', 50, 250, 255);
    Screen('Flip', theWindow);
    KbWait;
end

HideCursor;
theX = round(theRect(RectRight) / 2);
theY = round(theRect(RectBottom) / 2);
Screen(theWindow, 'DrawText', '.', theX, theY, 255);
Screen('Flip', theWindow);

InitializePsychSound;

try
    pahandle = PsychPortAudio('Open', [], [], 0, freq, nrchannels);
catch
    fprintf('\nCould not open device at wanted playback frequency of %i Hz. Will retry
with device default frequency.\n', freq);
    fprintf('Sound may sound a bit out of tune, ...\n\n');
    psychlasterror('reset');
    pahandle = PsychPortAudio('Open', [], [], 0, [], nrchannels);
end

audio=string(Names{1,1}{selectedsample,1});
[waveData ,F]=loadWave(audio);

PsychPortAudio('FillBuffer', pahandle, waveData);
PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing first sample

WaitSecs(6);

if (whichfirstno==1)
    selectedsample2=selectedsample+1;
else
    selectedsample2=selectedsample-1;
end

audio=string(Names{1,1}{selectedsample2,1});
[waveData ,F]=loadWave(audio);

PsychPortAudio('FillBuffer', pahandle, waveData);
PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing second sample

```

```

WaitSecs(4);

Screen(theWindow,'DrawText', strcat(int2str(iteration), ') Audio sample is successfully
completed...'),50,50,255);
Screen(theWindow,'DrawText', 'Which audio sample sounded more harmonic? 1st or
2nd?.',50,100,255);
Screen(theWindow,'DrawText', 'Please indicate your response by pressing the
corresponding number;',50,150,255);
Screen('Flip', theWindow);

secondst0=GetSecs;

while 1
    KbWait;
    [ keyIsDown, seconds, keyCode ] = KbCheck;
    rt=seconds-secondst0;
    keyCode = find(keyCode, 1);
    if keyIsDown && ((keyCode == 49) || (keyCode == 50))
        pressedKey = KbName(keyCode);
        fprintf(fea3, string(randomizer));
        fprintf(fea3, ': ');

        if keyCode == 49
            fprintf(fea3, string(Names{1,1}{selectedsample,1}));
            fprintf(fea3, ' ');
            fprintf(fea3, string(whichfirstno));

        end
        if keyCode == 50
            fprintf(fea3, string(Names{1,1}{selectedsample2,1}));
            fprintf(fea3, ' ');
            fprintf(fea3, string(whichsecondno));

        end

        fprintf(fea3, ' ');
        fprintf(fea3, num2str(rt));
        fprintf(fea3, '\n');
        break;
    end
end

% Stop playback:
PsychPortAudio('Stop', pahandle);

% Close the audio device:
PsychPortAudio('Close', pahandle);

if iteration == 10 || iteration == 40
    Screen(theWindow,'DrawText', 'It is time for a break...',50,50,255);
    Screen(theWindow,'DrawText', 'Press right arrow to continue after the break
.',50,100,255);
    Screen('Flip', theWindow);

```

```

while 1
    KbWait;
    [ keyIsDown, seconds, keyCode ] = KbCheck;
    keyCode = find(keyCode, 1);

    if keyIsDown && (keyCode == 39)
        break;
    end
end

iteration = iteration + 1;
hashtable(randomizer) = 1;
hashtablewhichfirst(randomizer) = 1;
continue;

catch
    sca;
    psychrethrow(psychlasterror);
end
end
Screen('Close', theWindow);
fprintf(fea3, '\n');
fclose(fea);
fclose(fea2);
fclose(fea3);

```

EXPERIMENT 4:

```

function [waveData,F]=loadWave(wave)
    filename=char(strcat(wave, '.wav'))
    wavfilename =
fullfile('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5\Final_Set\Shorts',
filename)
    [y, freq] = psychwavread(wavfilename);
    waveData = y';
    nrchannels = size(waveData , 1);
    F=freq;
end

clear all;
clc;
rng('shuffle');

Screen('Preference', 'SkipSyncTests', 1);
nrchannels=1;
freq=44100;
bits=16;

filepath =
'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5_EEG\Final_Set\Shorts';

```

```

filename = 'wavenames_working memory.txt';
file = fullfile(filepath, filename);
fea = fopen(file);
Names=textscan(fea,'%s');

filepath =
'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5_EEG\Final_Set\Shorts';
filename2 = 'randomizer.xlsx';
file2 = fullfile(filepath, filename2);

filepath =
'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5_EEG\Final_Set\Shorts';
filename3 = 'resultfile.txt';
file3 = fullfile(filepath, filename3);
fea3 = fopen(file3, 'wt');

% initialize access to the inpoutx64 low-level I/O driver
config_io;
% optional step: verify that the inpoutx64 driver was successfully initialized
global cogent;
if( cogent.io.status ~= 0 )
disp('inp/outp installation failed');
end
disp('oldu')
% write a value to the default LPT1 printer output port (at 0x378)
address = hex2dec('378');
% define as a 8-pin array
bits_trg1 = '0 0 0 0 0 0 1';
data_out_trg1 = bin2dec(bits_trg1); % harmonic first
bits_trg2 = '0 0 0 0 0 1 0';
data_out_trg2 = bin2dec(bits_trg2); % random first
bits_trg3 = '0 0 0 0 1 0 0';
data_out_trg3 = bin2dec(bits_trg2); %harmonic second same
bits_trg4 = '0 0 0 1 0 0 0';
data_out_trg4 = bin2dec(bits_trg2); %random second same
bits_trg5 = '0 0 1 0 0 0 0';
data_out_trg5 = bin2dec(bits_trg2); %harmonic second different
bits_trg6 = '0 0 1 0 0 0 0';
data_out_trg6 = bin2dec(bits_trg2); %random second different
byte_off = 0;
outp(address,byte_off);

for rd=1:1:4
    hashtable=[];
    hashtable = zeros(1,4);
    for rdi=1:1:48
        randomizer(rd,rdi) = randi(4);
        hashtable(randomizer(rd,rdi))=hashtable(randomizer(rd,rdi))+1;

        while hashtable(randomizer(rd,rdi)) >12

            randomizer(rd,rdi) = randi(4);
            hashtable(randomizer(rd,rdi))=hashtable(randomizer(rd,rdi))+1;

```

```

        end
    end
end

xlswrite(file2,randomizer);

    whichScreen = max(Screen('Screens'));
    [theWindow, theRect] = Screen('OpenWindow', whichScreen);
    Screen(theWindow,'FillRect', 0);
    Screen(theWindow,'DrawText', 'Welcome to the experiment.',50,50,255);
    Screen(theWindow,'DrawText', 'You will be given 48 audio sample couples to listen to in
each session. There will be 4 sessions in total.',50,100,255);
    Screen(theWindow,'DrawText', 'Please listen to the audio carefully and indicate if the
second sample is the same with the first sample',50,150,255);
    Screen(theWindow,'DrawText', 'You should indicate your response by pressing 1 (if yes)
or 2 (if no) from the keyboard.',50,200,255);
    Screen(theWindow,'DrawText', 'Press right arrow key to continue...',50,250,255);
    Screen('Flip', theWindow);
    KbWait;

    HideCursor;
    theX = round(theRect(RectRight) / 2);
    theY = round(theRect(RectBottom) / 2);
for rd=1:1:4
    for iteration=1:1:4
        try
            InitializePsychSound;
            try
                pahandle = PsychPortAudio('Open', [], [], 0, freq, nrchannels);
            catch
                fprintf('\nCould not open device at wanted playback frequency of %i Hz. Will retry
with device default frequency.\n', freq);
                fprintf('Sound may sound a bit out of tune, ...\n\n');

                psychlasterror('reset');
                pahandle = PsychPortAudio('Open', [], [], 0, [], nrchannels);
            end

            Screen(theWindow,'DrawText', 'X', theX, theY, 255);
            Screen('Flip', theWindow);
            audio='noise_100'
            [waveData ,F]=loadWave(audio);
            PsychPortAudio('FillBuffer', pahandle, waveData);
            PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing noise

            WaitSecs(3);

            Screen(theWindow,'DrawText', 'X', theX, theY, 255);
            Screen('Flip', theWindow);
            selectedsample1=randomizer(rd,iteration)*2-1;
            selectedsample2=randomizer(rd,iteration)*2;
            audio=string(Names{1,1}{selectedsample1,1});
            [waveData ,F]=loadWave(audio);

```

```

if (randomizer(rd,iteration)== 1 || randomizer(rd,iteration)==3)
    data_out=data_out_trg1; %harmonic
else
    data_out=data_out_trg2; %random
end

PsychPortAudio('FillBuffer', pahandle, waveData);
PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing first sample

outp(address,data_out);
WaitSecs(0.004);
outp(address,byte_off);

WaitSecs(3);

Screen(theWindow,'DrawText', 'X', theX, theY, 255);
Screen('Flip', theWindow);

WaitSecs(2);

Screen(theWindow,'DrawText', 'X', theX, theY, 255);
Screen('Flip', theWindow);
audio=string(Names{1,1}{selectedsample2,1});
[waveData ,F]=loadWave(audio);

if (randomizer(rd,iteration)== 1)
    data_out=data_out_trg3; %harmonic same
elseif (randomizer(rd,iteration)== 2)
    data_out=data_out_trg4; %random same
elseif (randomizer(rd,iteration)== 3)
    data_out=data_out_trg5; %harmonic different
else
    data_out=data_out_trg6; %random different
end

outp(address,data_out);
WaitSecs(0.004);
outp(address,byte_off);

PsychPortAudio('FillBuffer', pahandle, waveData);
PsychPortAudio('Start', pahandle, 1, 0, 1); % Start playing second sample

WaitSecs(3);

Screen(theWindow,'DrawText', strcat(int2str(iteration), ' Audio sample is successfully
completed...'),50,50,255);
Screen(theWindow,'DrawText', 'Was the second sample the same with the first
one?',50,100,255);
Screen(theWindow,'DrawText', 'Please indicate your response by 1 for YES or 2 for
NO;',50,150,255);
Screen('Flip', theWindow);

secondst0=GetSecs;

```

```

while 1
    KbWait;
    [ keyIsDown, seconds, keyCode ] = KbCheck;
    rt=seconds-secondst0;
    keyCode = find(keyCode, 1);
    if keyIsDown && ((keyCode == 49) || (keyCode == 50))
        pressedKey = KbName(keyCode);
        fprintf(fea3, string(randomizer(rd,iteration)));
        fprintf(fea3, ' ');

        if keyCode == 49
            fprintf(fea3, 'YES');
            if (randomizer(rd,iteration)==1 || randomizer(rd,iteration)==2)
                fprintf(fea3, '1');
            else
                fprintf(fea3, '0');
            end
        end

        if keyCode == 50
            fprintf(fea3, 'NO');
            if (randomizer(rd,iteration)== 1 || randomizer(rd,iteration)==2)
                fprintf(fea3, '0');
            else
                fprintf(fea3, '1');
            end
        end

        fprintf(fea3, ' ');
        fprintf(fea3, num2str(rt));
        fprintf(fea3, '\n');
        break;
    end
end

% Stop playback:
PsychPortAudio('Stop', pahandle);

% Close the audio device:
PsychPortAudio('Close', pahandle);
iteration = iteration + 1;

continue;

catch
    sca;
    psychrethrow(psychlasterror);
end

end

Screen(theWindow,'DrawText', 'It is time for a break...',50,50,255);
Screen(theWindow,'DrawText', 'Press right arrow to continue after the break
.',50,100,255);

```

```
Screen('Flip', theWindow);

while 1
    KbWait;
    [ keyIsDown, seconds, keyCode ] = KbCheck;
    keyCode = find(keyCode, 1);

    if keyIsDown && (keyCode == 39)
        break;
    end
end

end

Screen('Close', theWindow);
fprintf(fea3, '\n');
fclose(fea);
fclose(fea2);
fclose(fea3);
fclose(fea);
```

APPENDIX B: CODES FOR SOUND GENERATION

Harmonic sequences frequency search with C++:

```
//=====
// Name      : Sound_generator.cpp
// Author     : Filiz
// Version    :
// Copyright   : Your copyright notice
//=====

using namespace std;
#include "Sequence.h"
int main() {

    Sequence S;
    S.Fi=190;
    S.result_count=0;
    int arr[]={4/5,4/3,4/5,4/3,4/5,4/3,4/5,4/3,4/5,4/3,4/5,4/3,4/3,4/3};
    vector<int> list;
    for (int i=0;i<14;i++){
        S.list.push_back(arr[i]);
    }
    S.l=0;
    S.r = 14;
    S.permutation_harm(S.Fi, S.list, S.l, S.r, S.result_count, S.Results, S.OutM);

    int n=15;
    int k=1;
    string filename;
    ifstream fs;
    ofstream fo;
    filename="C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment3\Harmonic_"+(char)n+"\Harmonic_Seq"+(char)k+"\Frequencies_harmonic_seq_"+(char)k+".txt";
    filename= "Frequencies_harmonic_seq_1.txt";

    fo.open(filename.c_str(), fstream::app);
    for (int i=0;i<S.Results.size();i++) {
        for (int j=0;j<S.Results[0].size();j++) {
            fo << S.Results[i][j]<<" ";
        }
        fo <<endl;
    }
    fo.close(); // close the file
    return 0;
}
```

```

/*
 * Sequence.h
 *
 * Created on: Sep 25, 2019
 * Author: filiz
 */

#ifndef SEQUENCE_H_
#define SEQUENCE_H_
#include <string>
#include <iostream>
#include <fstream>
#include <vector>
#include <sstream>

using namespace std;

class Sequence {
public:
    Sequence();
    int Fi,l,r;
    vector<int> list;
    vector<vector<int> > Results, OutM;
    int result_count;
    int checkcondition_harm(int , vector<int>&, vector<int>& );
    void permutation_harm(int ,vector<int>& ,int ,int ,int ,vector<vector<int> >& ,
vector<vector<int> >& );

};

#endif /* SEQUENCE_H_ */

/*
 * Sequence.cpp
 *
 * Created on: Sep 25, 2019
 * Author: filiz
 */

#include "Sequence.h"
using namespace std;
Sequence::Sequence() {
    // TODO Auto-generated constructor stub
}

int checkcondition_harm(int Fi, vector<int>& list, vector<int>& Fresult) {
    int Fn, out;
    Fresult.push_back(Fi);
    out=1;
    for (int i=0;i<list.size();i++) {
        Fn=Fresult[i]*list[i];
        if (Fn<80 || Fn>45000) {
            out=0;
            break;

```

```

        }
        else {
            for (int j=0;j<Fresult.size();j++) {
                if (Fn == Fresult[j]) {
                    out=0;
                    break;
                }
            }
        }
        Fresult.push_back(Fn);
    }
    return out;
}

void permutation(int Fi,vector<int>& list, int l,int r,int result_count,vector<vector<int> >&
Results, vector<vector<int> >& OutM) {
    int i, temp, out;
    vector<int> Fresult;

    if (l == r) {
        out=checkcondition_harm(Fi, list, Fresult);
        if (out==1 && Fresult[0]/Fresult[ sizeof(Fresult)-1]<1.33 && Fresult[1]/Fresult[ sizeof(Fresult)-
1]>0.66) {
            result_count=result_count+1;
            for (int k=0;k<15;k++) {
                Results[k][result_count]=Fresult[k];
            }
        }
    }

    else {
        for (i=l;i<=r;i++) {
            using std::swap;
            temp=list[l];
            list[l]=list[i];
            list[i]=temp;
            permutation(Fi, list, l+1,r,result_count,Results,OutM);
            temp=list[l];
            list[l]=list[i];
            list[i]=temp;
        }
    }
}

```

Inharmonic sequences frequency search with C++:

```

//=====
//=====
// Name      : Rand_sounds.cpp
// Author     : Filiz
// Version    :
// Copyright  : Your copyright notice
// Description : Hello World in C++, Ansi-style

```

```

//=====
=====

#include <iostream>
using namespace std;

#include "Sequencerand.h"
void readfile(string, std::vector<vector <double> >& );

int main() {
    int result_count=0, K=1;
    int l=0;
    int r = 10;
    Sequencerand S;
    std::vector<vector<double> > Results;
    std::vector<double> list, listorg;
    vector <double> listorg_1;
    int Fi;
    std::vector<vector <double> > Frequencies;
    for(int Fi=285;Fi<286;Fi++) {
        // Fi=300;
        cout<<"Fi: "<<Fi<<endl;

        std::ostringstream str1,strs2;
        str1 << Fi;
        std::string str1 = str1.str();
        str2 << K;
        std::string str2 = str2.str();
        string filename=
"/home/filiz/workspace/Rand_sounds/Frequencies_harmonic_seq_"+str1+"_"+str2+".txt";

        Frequencies.empty();
        Frequencies.clear();

        readfile(filename, Frequencies);

        for (int i=0;i<Frequencies.size();i++) {
            list.empty();
            list.clear();
            list=Frequencies[i];
            listorg=list;
            listorg_1.empty();
            listorg_1.clear();

            for (int k=0; k<listorg.size()-1;k++) {
                if (listorg[k+1]/listorg[k]<1)
                    listorg_1.push_back(0);
                else
                    listorg_1.push_back(1);
            }

            if (listorg[0]/listorg[listorg.size()-1]<1)
                listorg_1.push_back(0);

```

```

        else
            listorg_1.push_back(1);

        S.permutation_rand(listorg, listorg_1, list, l, r, result_count, Results);
    }

}

return 0;
}

void readfile (string filename, std::vector<vector <double> >& Frequencies)
{
    ifstream fs;
    ofstream fo;
    string lastline;
    std::vector <double> line;

    std::ifstream in(filename.c_str());

    if(in)
    {
        while (std::getline(in, lastline)) {
            std::stringstream ss(lastline);
            for(std::string lastline; ss >> lastline; )
            {
                double temp=atof(lastline.c_str());
                line.push_back(temp);
            }
            Frequencies.push_back(line);
            line.empty();
            line.clear();
        }
    }
    else
    {
        cout << "Could not find end line character" << endl;
    }

    fs.close();
}

/*
 * Sequencerandrand.cpp
 *
 * Created on: Sep 28, 2019
 * Author: filiz
 */

#include "Sequencerand.h"
using namespace std;

Sequencerand::Sequencerand() {
    // TODO Auto-generated constructor stub

```

```

        int K=1;

    }

    int Sequencerand::checkcondition_rand(std::vector<double> & list) {
        int out;
        double Fn;

        out=1;
        for (int i=0;i<list.size()-1;i++) {
            Fn=list[i];
            if (list[i+1]/list[i] < 2.0/3.0 || list[i+1]/list[i] > 4.0/3.0 || abs(list[i+1]/list[i]-
            3.0/2.0)<0.0001 || abs(list[i+1]/list[i]- 3.0/4.0)<0.0001 || abs(list[i+1]/list[i]- 2.0/3.0)<0.0001 ||
            abs(list[i+1]/list[i]- 4.0/3.0)<0.0001 || abs(list[i+1]/list[i]- 5.0/4.0)<0.0001 || abs(list[i+1]/list[i]-
            4.0/5.0)<0.0001 || abs(list[i+1]/list[i]- 9.0/8.0)<0.001 ) {
                out=0;
                break;
            }
        }
        return out;
    }

    int Sequencerand::compare(std::vector<double>& listorg_1, std::vector<double>& listorg,
    std::vector<double>& list) {

        int flag_harm=-1, flag_rand=-1, flag_rand2=0, flag_harm2=0, out=0;
        int num_of_one_rand=0, num_of_one_harm=0;
        std::vector<int> one_rand, one_harm, list_1;;

        for (int k=0; k<list.size()-1;k++) {
            if (list[k+1]/list[k]<1)
                list_1.push_back(0);
            else
                list_1.push_back(1);
        }
        if (list[0]/list[list.size()-1]<1)
            list_1.push_back(0);
        else
            list_1.push_back(1);

        /*

        for (int i=0; i<list_1.size(); i++) {
            if (list_1[i]==1) {
                num_of_one_rand=num_of_one_rand+1;
                flag_rand=1;
                if (i==list_1.size()-1) {
                    for(int j=0; j<list_1.size(); j++) {
                        if (list_1[j]==1) {
                            num_of_one_rand=num_of_one_rand+1;
                            flag_rand2=1;
                        }
                    }
                }
            }
        }
    }

```

```

        else
            break;
    }
    one_rand.push_back(num_of_one_rand);
    if (flag_rand2==1)
        one_rand.erase(one_rand.begin());
    }
}
else
{
    if (flag_rand==1) {
        one_rand.push_back(num_of_one_rand);
    }
    flag_rand=0;
    num_of_one_rand=0;
}

if (listorg_1[i]==1) {
    num_of_one_harm=num_of_one_harm+1;
    flag_harm=1;
    if (i==listorg_1.size()-1) {
        for(int j=0; j<listorg_1.size(); j++) {
            if (listorg_1[j]==1) {
                num_of_one_harm=num_of_one_harm+1;
                flag_harm2=1;
            }
            else
                break;
        }
        one_harm.push_back(num_of_one_harm);
        if (flag_harm2==1)
            one_harm.erase(one_harm.begin());
    }
}
else {
    if (flag_harm==1) {
        one_harm.push_back(num_of_one_harm);
    }
    flag_harm=0;
    num_of_one_harm=0;
}
} */

int flag=0;
for(int i=0; i<list_1.size(); i++) {
    if (list_1[i]!=listorg_1[i]) {
        flag=1;
        break; }
}

if (flag==0 && list[0]==listorg[0]) {
    out=1;
    cout<<"Compare out=1"<<endl;
}

```

```

    }

    return out;
}

void Sequencerand::permutation_rand(std::vector<double>& listorg, std::vector<double>&
listorg_1, std::vector<double>& list, int l,int r,int& result_count,std::vector<std::vector<double>
>& Results) {
    int i, out=0;
    double temp=0.0;
    int K=1;
    int Fi;
    Fi=int(listorg[0]);
    string
filename="/home/filiz/workspace/Rand_sounds/Random/Frequencies_harmonic_vs_random_seq_ne
w_without_5over4.txt";

    if (l == r) {
        if (listorg!=list) {
            out=checkcondition_rand(list);
        }

        if ( out==1 && compare(listorg_1,listorg, list)==1 && list[0]/list[list.size()-1]<2 &&
list[0]/list[list.size()-1]>0.5 ) { //
            cout<<"out=1"<<endl;

            cout<<result_count<<endl;
            result_count=result_count+1;

            fo.open(filename.c_str(), fstream::app);
            for (int i=0;i<listorg.size();i++) {
                fo << listorg[i]<<" ";
            }

            fo <<" vs ";

            for (int i=0;i<list.size();i++) {
                fo << list[i]<<" ";
            }

            fo <<endl;
            fo.close(); // close the file

            Results.push_back(list);
        }
    }

    else {
        for (i=l;i<r;i++) {

            temp=list[l];
            list[l]=list[i];
            list[i]=temp;

```

```

        permutation_rand(listorg, listorg_1, list, l+1, r, result_count, Results);
        temp=list[l];
        list[l]=list[i];
        list[i]=temp;
    }
}

/*
 * Sequencerand.h
 *
 * Created on: Sep 28, 2019
 * Author: filiz

#ifndef SEQUENCERAND_H_
#define SEQUENCERAND_H_
#include <string>
#include <iostream>
#include <fstream>
#include <vector>
#include <sstream>
#include <stdlib.h>
#include <cmath>
using namespace std;

class Sequencerand {
public:
    Sequencerand();
    std::string filename;
    std::ifstream fs;
    std::ofstream fo;
    int checkcondition_rand(std::vector<double>& );
    void permutation_rand(std::vector<double>&, std::vector<double>&
, std::vector<double>& ,int ,int ,int& ,std::vector<std::vector<double> >& );
    int compare(std::vector<double>& , std::vector<double>& , std::vector<double>& );
};

#endif /* SEQUENCERAND_H_ */

```

Sound generation on Matlab:

```

function sound_generator(N,L,T,F,Type,k)
T=0.05;
Tramp=T*0.1;
Tname=T*1000;
N=10;
FRW=0;
fs=44100;
bits=16;

for i=1:1:N
    FRF=F(i);
    wave = oscillator('Sinusoid',T,FRF,Tramp);
    FRW=cat(1,FRW,wave);

```

```

end
    filename = strcat (
        'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment4\Final_Set\Shorts_new\',
        num2str(Tname),'ms_',num2str(N),'_', num2str(k),'_',Type,'.wav' );
    audiowrite(filename,FRW(:,1),fs,'BitsPerSample',bits);

FinalSound=0;
[S, FS]=audioread(filename);

for i=1:L
    FinalSound=cat(1,FinalSound,S);
end

filename = strcat (
    'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment4\Final_Set\Longs\',
    num2str(Tname),'ms_',num2str(N),'_', num2str(k),'_',Type,'.wav' );

audiowrite(filename,FinalSound(:,1),fs,'BitsPerSample',bits);
end

clear all;
clc;
T=0.05;
L=8;

Type2='harmonic';
Type2='inharmonic';
N=10;

for i=1:1:4
    for k=1:1:5
        Type1='harmonic';
        filename= strcat(
            'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment4\Frequencies\Frequencies_',
            Type1,'_seq','_', num2str(k),'.xlsx');
        F1=xlsread(filename);
        sound_generator(N,Lnew,T,F1,Type1,k);
        Type2='random';
        filename= strcat(
            'C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment4\Frequencies\Frequencies_',
            Type2,'_seq','_', num2str(k),'.xlsx');
        F2=xlsread(filename);
        sound_generator(N,Lnew,T,F2,Type2,k);
    end
    L=L/2;
    T=T*2;
end
end

```

APPENDIX C: CODES FOR EEG DATA ANALYSIS

Preprocessing and segmenting the EEG data:

```

function ft_comp_freq_anal(comp)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
cfg      = [];
cfg.output    = 'pow';
cfg.channel    = 'all'; %compute the power spectrum in all ICs
cfg.method     = 'mtmfft';
cfg.taper      = 'hanning';
cfg.foi        = 2:2:125;
freq = ft_freqanalysis(cfg, comp);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
nsubplots = 25;
nbyn = sqrt(nsubplots); % sqrt(nsubplots) should not contain decimals,
type doc subplot

Nfigs = ceil(size(comp.topo,1)/nsubplots);
tot = Nfigs*nsubplots;

rptvect = 1:size(comp.topo,1);
rptvect = padarray(rptvect, [0 tot-size(comp.topo,1)], 0, 'post');
rptvect = reshape(rptvect, nsubplots, Nfigs);

for r=1:size(rptvect,1)
    figure; set(gcf, 'units', 'normalized', 'outerposition', [0 0 1 1]); %full
    Screen
    k=0;
    for j=1:size(rptvect,2)
        if ~(rptvect(r,j)==0)
            k=k+1;
            cfg=[];
            cfg.channel = rptvect(r,j);
            subplot(nbyn,nbyn,k); ft_singleplotER(cfg, freq);
        end
    end
end

end

function [ICA_removed_data] = ft_ICA_artifact_remove(data)

% perform the independent component analysis (i.e., decompose the data)
cfg      = [];
cfg.method = 'runica'; % this is the default and uses the implementation from EEGLAB
cfg.channel=[1:31];
comp = ft_componentanalysis(cfg, data);

% plot the components for visual inspection

```

```

figure
cfg = [];
cfg.component = 1:31;    % specify the component(s) that should be plotted
cfg.layout = 'easycapM3.mat'; % specify the layout file that should be used for plotting
cfg.comment = 'no';
ft_topoplotIC(cfg, comp)

ft_comp_freq_anal(comp);

cfg = [];
cfg.layout = 'easycapM3.mat'; % specify the layout file that should be used for plotting
cfg.viewmode = 'component';
ft_databrowser(cfg, comp)

removed_comp=[];
x=0;
prompt = 'What are the components you want to remove?';
while x~= ' '
    x = input( prompt );
    removed_comp=[removed_comp,x];
end

% remove the bad components and backproject the data
cfg = [];
cfg.component = removed_comp; % to be removed component(s)
ICA_removed_data = ft_rejectcomponent(cfg, comp, data)
close all
end

clear all;
clc;
for n=3:1:14
    if (n~=5)

for k=1:1:4
    if(n==3 && k==1)
%read EEG data and notch filter 50 100 150 Hz default
filepath=strcat('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5_EEG\Final
_Set\Data\EEG_data\');
cfg = [];
cfg.dftfilter = 'yes';
cfg.dftfreq = [50 100 150 200];
cfg.channel = [1:31];
cfg.dataset = strcat(filepath,'Deney',num2str(n),'.vhdr');
cfg.continuous = 'yes'; % force it to be continuous
data = ft_preprocessing(cfg);

else

%read EEG data and notch filter 50 100 150 Hz default
filepath=strcat('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5_EEG\Final
_Set\Data\EEG_data\');

```

```

cfg = [];
cfg.dftfilter = 'yes';
cfg.dftfreq = [50 100 150 200];
cfg.channel = [1:31];
% cfg.hpfilter = 'yes';
% cfg.hpfiltertype = 'fir';
% cfg.hpfreq = 0.01;
cfg.dataset = strcat(filepath,'Denev',num2str(n),'_',num2str(k),'vhdr');
cfg.continuous = 'yes'; % force it to be continuous
data = ft_preprocessing(cfg);

```

end

```

cfg.trialdef.eventtype = '1';
cfg.trialdef.eventvalue = ['1' '1'];
cfg.trialdef.prestim = 0.5;
cfg.trialdef.poststim = 3;
cfg_harmonic_first = ft_definetrial(cfg);
data_harmonic_first(k) = ft_redefinetrial(cfg_harmonic_first, data);

cfg.trialdef.eventtype = '2';
cfg.trialdef.eventvalue = ['2' '1'];
cfg.trialdef.prestim = 0.5;
cfg.trialdef.poststim = 3;
cfg_random_first = ft_definetrial(cfg);
data_random_first(k) = ft_redefinetrial(cfg_random_first, data);

cfg.trialdef.eventtype = '3';
cfg.trialdef.eventvalue = ['3' '1'];
cfg.trialdef.prestim = 0.5;
cfg.trialdef.poststim = 3;
cfg_harmonic_second = ft_definetrial(cfg);
data_harmonic_second(k) = ft_redefinetrial(cfg_harmonic_second, data);

cfg.trialdef.eventtype = '4';
cfg.trialdef.eventvalue = ['4' '1'];
cfg.trialdef.prestim = 0.5;
cfg.trialdef.poststim = 3;
cfg_random_second = ft_definetrial(cfg);
data_random_second(k) = ft_redefinetrial(cfg_random_second, data);

```

end

```

cfg=[];
cfg.keeppsampleinfo = 'no';
data_harmonic_first_full = ft_appenddata(cfg, data_harmonic_first(1), data_harmonic_first(2),
data_harmonic_first(3), data_harmonic_first(4));
data_random_first_full = ft_appenddata(cfg, data_random_first(1), data_random_first(2),
data_random_first(3), data_random_first(4));
data_harmonic_second_full = ft_appenddata(cfg, data_harmonic_second(1),
data_harmonic_second(2), data_harmonic_second(3), data_harmonic_second(4));

```

```
data_random_second_full = ft_appenddata(cfg,data_random_second(1), data_random_second(2),
data_random_second(3), data_random_second(4));
```

```
cfg      = [];
cfg.method = 'summary';
cfg.channel = [1:31];
cfg.keepchannel = 'yes';
cfg.latency = 'all';
data_harmonic_first_full_artifact_removed = ft_rejectvisual(cfg, data_harmonic_first_full);
data_random_first_full_artifact_removed = ft_rejectvisual(cfg, data_random_first_full);
data_harmonic_second_full_artifact_removed = ft_rejectvisual(cfg, data_harmonic_second_full);
data_random_second_full_artifact_removed = ft_rejectvisual(cfg, data_random_second_full);

filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_harmonic_first_artifact_removed.mat'
);
save(filename, 'data_harmonic_first_full_artifact_removed')
filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_random_first_artifact_removed.mat');
save(filename, 'data_random_first_full_artifact_removed')
filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_harmonic_second_artifact_removed.
mat');
save(filename, 'data_harmonic_second_full_artifact_removed')
filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_random_second_artifact_removed.ma
t');
save(filename, 'data_random_second_full_artifact_removed')
```

```
data_harmonic_first_full_ICA_artifact_removed=ft_ICA_artifact_remove(data_harmonic_first_full_artifact_removed);
data_random_first_full_ICA_artifact_removed=ft_ICA_artifact_remove(data_random_first_full_artifact_removed);
data_harmonic_second_full_ICA_artifact_removed=ft_ICA_artifact_remove(data_harmonic_second_full_artifact_removed);
data_random_second_full_ICA_artifact_removed=ft_ICA_artifact_remove(data_random_second_full_artifact_removed);
```

```
filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_harmonic_first_ICA_removed.mat');
save(filename, 'data_harmonic_first_full_ICA_artifact_removed')
filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_random_first_ICA_removed.mat');
save(filename, 'data_random_first_full_ICA_artifact_removed')
filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_harmonic_second_ICA_removed.mat'
);
save(filename, 'data_harmonic_second_full_ICA_artifact_removed')
filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_random_second_ICA_removed.mat');
save(filename, 'data_random_second_full_ICA_artifact_removed')
end
end
```

ERP Analysis:

```

clc; clear all;
filepath=strcat('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5_EEG\Final
_Set\Data\EEG_data\');

for n=3:1:14
    if(n<5)
        filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_harmonic_first_ICA_removed.mat');
        data_harmonic_first_full_ICA_removed=load(filename);
        filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_random_first_ICA_removed.mat');
        data_random_first_full_ICA_removed=load(filename);
        filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_harmonic_second_ICA_removed.mat'
        );
        data_harmonic_second_full_ICA_removed=load(filename);
        filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_random_second_ICA_removed.mat');
        data_random_second_full_ICA_removed=load(filename);
        filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_harmonic_second_otsame_ICA_remo
        ved.mat');
        data_harmonic_second_otsame_full_ICA_removed=load(filename);
        filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_random_second_otsame_ICA_remo
        ved.mat');
        data_random_second_otsame_full_ICA_removed=load(filename);

        cfg = [];
        cfg.continuous = 'no';
        cfg.baseline = [-0.5 0];
        data1=ft_preprocessing(cfg,data_harmonic_first_full_ICA_removed.data_harmonic_first_full);
        harmonic_first_avg{n-2} = ft_timelockanalysis(cfg, data1);

        data2=ft_preprocessing(cfg,data_random_first_full_ICA_removed.data_random_first_full);
        random_first_avg{n-2} = ft_timelockanalysis(cfg, data2);

        data3=ft_preprocessing(cfg,data_harmonic_second_full_ICA_removed.data_harmonic_second_full
        );
        harmonic_second_avg{n-2} = ft_timelockanalysis(cfg, data3);

        data4=ft_preprocessing(cfg,data_random_second_full_ICA_removed.data_random_second_full);
        random_second_avg{n-2} = ft_timelockanalysis(cfg, data4 );

        data5=ft_preprocessing(cfg,data_harmonic_second_otsame_full_ICA_removed.data_harmonic_se
        cond_otsame_full);
        harmonic_second_otsame_avg{n-2} = ft_timelockanalysis(cfg, data5);

        data6=ft_preprocessing(cfg,data_random_second_otsame_full_ICA_removed.data_random_secon
        d_otsame_full);
        random_second_otsame_avg{n-2} = ft_timelockanalysis(cfg, data6 );
        end

    if(n>5)

        filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_harmonic_first_ICA_removed.mat');
        data_harmonic_first_full_ICA_removed=load(filename);

```

```

filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_random_first_ICA_removed.mat');
data_random_first_full_ICA_removed=load(filename);
filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_harmonic_second_ICA_removed.mat'
);
data_harmonic_second_full_ICA_removed=load(filename);
filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_random_second_ICA_removed.mat');
data_random_second_full_ICA_removed=load(filename);
filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_harmonic_second_otsame_ICA_remo
ved.mat');
data_harmonic_second_otsame_full_ICA_removed=load(filename);
filename=strcat(filepath,'\Fieldtrip_data\Deney',num2str(n),'_random_second_otsame_ICA_remo
ved.mat');
data_random_second_otsame_full_ICA_removed=load(filename);

cfg = [];
cfg.continuous = 'no';
cfg.baseline = [-0.5 0];
data1=ft_preprocessing(cfg,data_harmonic_first_full_ICA_removed.data_harmonic_first_full);
harmonic_first_avg{n-3} = ft_timelockanalysis(cfg, data1);

data2=ft_preprocessing(cfg,data_random_first_full_ICA_removed.data_random_first_full);
random_first_avg{n-3} = ft_timelockanalysis(cfg, data2);

data3=ft_preprocessing(cfg,data_harmonic_second_full_ICA_removed.data_harmonic_second_full
);
harmonic_second_avg{n-3} = ft_timelockanalysis(cfg, data3);

data4=ft_preprocessing(cfg,data_random_second_full_ICA_removed.data_random_second_full);
random_second_avg{n-3} = ft_timelockanalysis(cfg, data4 );

data5=ft_preprocessing(cfg,data_harmonic_second_otsame_full_ICA_removed.data_harmonic_se
cond_otsame_full);
harmonic_second_otsame_avg{n-3} = ft_timelockanalysis(cfg, data5);

data6=ft_preprocessing(cfg,data_random_second_otsame_full_ICA_removed.data_random_secon
d_otsame_full);
random_second_otsame_avg{n-3} = ft_timelockanalysis(cfg, data6 );
    end
end

filename=strcat(filepath,'\Fieldtrip_data\Harmonic_first_avg.mat');
save(filename, 'harmonic_first_avg');
filename=strcat(filepath,'\Fieldtrip_data\Random_first_avg.mat');
save(filename, 'random_first_avg');
filename=strcat(filepath,'\Fieldtrip_data\Harmonic_second_avg.mat');
save(filename, 'harmonic_second_avg');
filename=strcat(filepath,'\Fieldtrip_data\Random_second_avg.mat');
save(filename, 'random_second_avg');
filename=strcat(filepath,'\Fieldtrip_data\Harmonic_second_otsame_avg.mat');
save(filename, 'harmonic_second_otsame_avg');
filename=strcat(filepath,'\Fieldtrip_data\Random_second_otsame_avg.mat');
save(filename, 'random_second_otsame_avg');

% calculate grand average for each condition

```

```

cfg = [];
cfg.channel = 'all';
cfg.latency = 'all';
cfg.parameter = 'avg';
cfg.keeptrials = 'yes';
harmonic_first_granavg = ft_timelockgrandaverage(cfg, harmonic_first_avg{:});
random_first_granavg = ft_timelockgrandaverage(cfg, random_first_avg{:});
harmonic_second_granavg = ft_timelockgrandaverage(cfg, harmonic_second_avg{:});
random_second_granavg = ft_timelockgrandaverage(cfg, random_second_avg{:});
harmonic_second_otsame_granavg = ft_timelockgrandaverage(cfg,
harmonic_second_otsame_avg{:});
random_second_otsame_granavg = ft_timelockgrandaverage(cfg,
random_second_otsame_avg{:});

for i=1:1:11

difference_harmonic{i}= harmonic_second_avg{i};
difference_harmonic{i}.avg= harmonic_second_otsame_avg{i}.avg-
harmonic_second_avg{i}.avg;
difference_random{i}= random_second_avg{i};
difference_random{i}.avg= random_second_otsame_avg{i}.avg-random_second_avg{i}.avg;
cfg = [];
cfg.continuous = 'no';
cfg.demean = 'yes';
difference_harmonic{i}=ft_timelockanalysis(cfg, difference_harmonic{i} );
difference_random{i}=ft_timelockanalysis(cfg, difference_random{i} );
end

cfg = [];
cfg.channel = 'all';
cfg.latency = 'all';
cfg.parameter = 'avg';
difference_harmonic_granavg = ft_timelockgrandaverage(cfg, difference_harmonic{:});
difference_random_granavg = ft_timelockgrandaverage(cfg, difference_random{:});

ch=5;
for i=1:1:11
    MMN_harmonic(i,:)=difference_harmonic{1, i}.avg(ch,1000:1150);
    MMN_random(i,:)=difference_random{1, i}.avg(ch,1000:1150);
end
ch=25
for i=1:1:11
    MMN_harmonic_P300(i,:)=difference_harmonic{1, i}.avg(ch,1150:1400);
    MMN_random_P300(i,:)=difference_random{1, i}.avg(ch,1150:1400);
end

```

Statistical Analysis for ERP

```

clc;
cfg = [];
cfg.method = 'montecarlo';    % use the Monte Carlo Method to calculate the significance
                                probability

```

```

cfg.statistic = 'ft_statfun_indepsamplesT'; % use the independent samples T-statistic as a measure
to
    % evaluate the effect at the sample level
cfg.correctm = 'cluster';
cfg.correcttail = 'alpha';
cfg.clusteralpha = 0.05; % alpha level of the sample-specific test statistic that
    % will be used for thresholding
cfg.clusterstatistic = 'maxsum'; % test statistic that will be evaluated under the
    % permutation distribution.

% cfg.neighbours = neighbours; % see below
cfg.tail = 0; % -1, 1 or 0 (default = 0); one-sided or two-sided test
cfg.clustertail = 0;
cfg.alpha = 0.05; % alpha level of the permutation test
cfg.numrandomization = 5000; % number of draws from the permutation distribution

design = zeros(1,11 + 11);
design(1,1:11) = 1;
design(1,11+1:11 + 11) = 2;

cfg.design = design; % design matrix
cfg.ivar = 1; % number or list with indices indicating the independent variable(s)

cfg_neighb = [];
cfg_neighb.method = 'template';
cfg.template = 'easycapM3.mat';
cfg.minnbchan = 0;
cfg.channel = [1:30]; % cell-array with selected channel labels
% neighbours = ft_prepare_neighbours(cfg_neighb, data4);
filepath=strcat('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5_EEG\Final
_Set\Data\EEG_data\');
filename=strcat(filepath,'\Fieldtrip_data\Neighbours\easycapM3_neighbours.mat');
load(filename, 'neighbours');

cfg.neighbours = neighbours; % the neighbours specify for each sensor with
    % which other sensors it can form clusters
cfg.latency = [0 3]; % time interval over which the experimental
    % conditions must be compared (in second_notsames)

[ERP_stat] = ft_timelockstatistics(cfg, difference_harmonic{1:11}, difference_random{1:11});

cfg = [];
avgHarmonic = ft_timelockgrandaverage(cfg, difference_harmonic{1:11});
avgRandom = ft_timelockgrandaverage(cfg, difference_random{1:11});

% Then take the difference of the averages using ft_math
cfg = [];
cfg.operation = 'subtract';
cfg.parameter = 'avg';
raweffectHarmonicvsRandom = ft_math(cfg, avgHarmonic, avgRandom);

pos_cluster_pvals = [ERP_stat.posclusters(:).prob];

```

```

% Then, find which clusters are significant, outputting their indices as held in stat.posclusters
% In case you have downloaded and loaded the data, ensure stat.cfg.alpha exist
if ~isfield(ERP_stat.cfg,'alpha')
    ERP_stat.cfg.alpha = 0.05;
end % stat.cfg.alpha was moved as the downloaded data was processed by an additional FieldTrip
function to anonymize the data.

pos_signif_clust = find(pos_cluster_pvals < ERP_stat.cfg.alpha);
pos = ismember(ERP_stat.posclusterslabelmat, pos_signif_clust);

% and now for the negative clusters...
neg_cluster_pvals = [ERP_stat.negclusters(:).prob];
neg_signif_clust = find(neg_cluster_pvals < ERP_stat.cfg.alpha);
neg = ismember(ERP_stat.negclusterslabelmat, neg_signif_clust);

timestep = 0.01; % timestep between time windows for each subplot
sampling_rate = 1000; % Data has a temporal resolution of 1000 Hz
sample_count = length(ERP_stat.time);
% number of temporal samples in the statistics object
j = [0:timestep:3]; % Temporal endpoints of the ERP average computed in each subplot
m = [1:timestep*sampling_rate:sample_count]; % temporal endpoints in M/EEG samples

% ensure the channels to have the same order in the average and in the statistical output.
% This might not be the case, because ft_math might shuffle the order
[i1,i2] = match_str(raweffectHarmonicvsRandom.label, ERP_stat.label);

cfg = [];
cfg.layout = 'easycapM3.mat';
cfg.maskparameter = 'mask';
cfg.maskstyle = 'box';
difference_harmonic.mask = neg; %ERP_stat.mask; % copy the significance mask into the ERF
figure; ft_multisubplotER(cfg, difference_harmonic, difference_random);
title('cluster_neg');

```

Statistical Analysis for Time-frequency domain

```

cfg.channel = 'all';
cfg.method = 'mtmconvol';
cfg.taper = 'hanning';
cfg.foi = 1:1:125;
cfg.toi = -0.5:0.1:3;
cfg.t_ftimwin = ones(length(cfg.foi),1).*1; % length of time window = 1 sec

for i=1:1:11
    TFR_harmonic_first_avg{i} = ft_freqanalysis(cfg, harmonic_first_avg{i});
    TFR_random_first_avg{i} = ft_freqanalysis(cfg, random_first_avg{i});
    TFR_harmonic_second_notsame_avg{i} = ft_freqanalysis(cfg,
    harmonic_second_notsame_avg{i});
    TFR_random_second_notsame_avg{i} = ft_freqanalysis(cfg, random_second_notsame_avg{i});
end

cfg = [];
TFR_harmonic_first_granavg = ft_freqgrandaverage(cfg, TFR_harmonic_first_avg{:});

```

```

TFR_random_first_granavg = ft_freqgrandaverage(cfg, TFR_random_first_avg{:});
TFR_harmonic_second_otsame_granavg = ft_freqgrandaverage(cfg,
TFR_harmonic_second_otsame_avg{:});
TFR_random_second_otsame_granavg = ft_freqgrandaverage(cfg,
TFR_random_second_otsame_avg{:});

% perform the statistical test using randomization and a clustering approach
% using the ft_freqstatistics function
cfg = [];
cfg_neighb.method = 'template';
cfg.template = 'easycapM3.mat';
cfg.neighbourdist = 0;
cfg.latency = 'all';
cfg.frequency = [15 32];
cfg.channel = [1:30]; % see FT_CHANNELSELECTION
cfg.avgovertime = 'no';
cfg.avgovertfreq = 'yes';
cfg.avgovertchan = 'no';
cfg.statistic = 'ft_statfun_depsamplesT';
cfg.clusteralpha = 0.05;
cfg.clustertest = 'maxsum';
cfg.minnbchan = 0;
cfg.tail = 0;
cfg.clustertail = 0;
cfg.alpha = 0.05;
cfg.numrandomization = 5000;
cfg.correctm = 'cluster';
cfg.correcttail = 'alpha';
cfg.method = 'montecarlo';

filepath= strcat('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5_EEG\Final
_Set\Data\EEG_data\');
filename= strcat(filepath, '\Fieldtrip_data\Neighbours\easycapM3_neighbours.mat');
load(filename, 'neighbours');
cfg.neighbours = neighbours; % the neighbours specify for each sensor with
% which other sensors it can form clusters

cfg.design = [
1 2 3 4 5 6 7 8 9 10 11 1 2 3 4 5 6 7 8 9 10 11 % subject number
1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 ]; % condition number

cfg.uvar = 1; % "subject" is unit of observation
cfg.ivar = 2; % "condition" is the independent variable
stat_TFR_first = ft_freqstatistics(cfg, TFR_harmonic_first_avg{1:11},
TFR_random_first_avg{1:11});

% perform the statistical test using randomization and a clustering approach
% using the ft_freqstatistics function
cfg = [];
cfg_neighb.method = 'template';
cfg.template = 'easycapM3.mat';
cfg.neighbourdist = 0;
cfg.latency = 'all';
cfg.frequency = [15 32];

```

```

cfg.channel      = [1:30]; % see FT_CHANNELSELECTION
cfg.avgovertime  = 'no';
cfg.avgoverfreq  = 'yes';
cfg.avgoverchan  = 'no';
cfg.statistic     = 'ft_statfun_depsamplesT';
cfg.clusteralpha = 0.05;
cfg.clusterstatistic = 'maxsum';
cfg.minnbchan    = 0;
cfg.tail         = 0;
cfg.clustertail   = 0;
cfg.alpha        = 0.05;
cfg.numrandomization = 5000;
cfg.correctm     = 'cluster';
cfg.correcttail  = 'alpha';
cfg.method       = 'montecarlo';

filepath=strcat('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5_EEG\Final
_Set\Data\EEG_data\');
filename=strcat(filepath, \Fieldtrip_data\Neighbours\easycapM3_neighbours.mat');
load(filename, 'neighbours');
cfg.neighbours    = neighbours; % the neighbours specify for each sensor with
                                % which other sensors it can form clusters

cfg.design        = [
    1 2 3 4 5 6 7 8 9 10 11 1 2 3 4 5 6 7 8 9 10 11 % subject number
    1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 ]; % condition number

cfg.uvar = 1; % "subject" is unit of observation
cfg.ivar = 2; % "condition" is the independent variable
stat_TFR_second_otsame = ft_freqstatistics(cfg, TFR_harmonic_second_otsame_avg{1:11},
TFR_random_second_otsame_avg{1:11});

% perform the statistical test using randomization and a clustering approach
% using the ft_freqstatistics function
cfg = [];
cfg_neighb.method = 'template';
cfg.template      = 'easycapM3.mat';
cfg.neighbourdist  = 0;
cfg.latency       = 'all';
cfg.frequency     = [15 32];
cfg.channel       = [1:30]; % see FT_CHANNELSELECTION
cfg.avgovertime   = 'no';
cfg.avgoverfreq   = 'yes';
cfg.avgoverchan   = 'no';
cfg.statistic     = 'ft_statfun_depsamplesT';
cfg.clusteralpha  = 0.05;
cfg.clusterstatistic = 'maxsum';
cfg.minnbchan     = 0;
cfg.tail         = 0;
cfg.clustertail   = 0;
cfg.alpha        = 0.05;
cfg.numrandomization = 5000;

```

```

cfg.correctm      = 'cluster';
cfg.correcttail   = 'alpha';
cfg.method        = 'montecarlo';

filepath=strcat('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5_EEG\Final
_Set\Data\EEG_data\');
filename=strcat(filepath,'\Fieldtrip_data\Neighbours\easycapM3_neighbours.mat');
load(filename, 'neighbours');
cfg.neighbours    = neighbours; % the neighbours specify for each sensor with
                                % which other sensors it can form clusters

cfg.design        = [
    1 2 3 4 5 6 7 8 9 10 11 1 2 3 4 5 6 7 8 9 10 11 % subject number
    1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 ]; % condition number

cfg.uvar = 1; % "subject" is unit of observation
cfg.ivar = 2; % "condition" is the independent variable
stat_TFR_harmonic_MM = ft_freqstatistics(cfg, TFR_harmonic_second_otsame_avg{1:11},
TFR_harmonic_first_avg{1:11});

% perform the statistical test using randomization and a clustering approach
% using the ft_freqstatistics function
cfg = [];
cfg_neighb.method = 'template';
cfg.template      = 'easycapM3.mat';
cfg.neighbourdist  = 0;
cfg.latency       = 'all';
cfg.frequency     = [15 32];
cfg.channel       = [1:30]; % see FT_CHANNELSELECTION
cfg.avgovertime   = 'no';
cfg.avgovertfreq  = 'yes';
cfg.avgovertchan  = 'no';
cfg.statistic     = 'ft_statfun_depsamplesT';
cfg.clusteralpha  = 0.05;
cfg.clusterstatistic = 'maxsum';
cfg.minnbchan     = 0;
cfg.tail          = 0;
cfg.clustertail   = 0;
cfg.alpha         = 0.05;
cfg.numrandomization = 5000;
cfg.correctm      = 'cluster';
cfg.correcttail   = 'alpha';
cfg.method        = 'montecarlo';

filepath=strcat('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5_EEG\Final
_Set\Data\EEG_data\');
filename=strcat(filepath,'\Fieldtrip_data\Neighbours\easycapM3_neighbours.mat');
load(filename, 'neighbours');
cfg.neighbours    = neighbours; % the neighbours specify for each sensor with
                                % which other sensors it can form clusters

cfg.design        = [
    1 2 3 4 5 6 7 8 9 10 11 1 2 3 4 5 6 7 8 9 10 11 % subject number

```

```

1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2]; % condition number
%
% cfg.design      = [
cfg.uvar = 1;          % "subject" is unit of observation
cfg.ivar = 2;          % "condition" is the independent variable
stat_TFR_random_MM = ft_freqstatistics(cfg, TFR_random_second_otsame_avg{1:11},
TFR_random_first_avg{1:11});

%
for i=1:1:11
cfg = [];
cfg.parameter = 'powspectrum';
cfg.operation = '(x1-x2) / (x1+x2)';
TFR_difference_harmonic{i} = ft_math(cfg, TFR_harmonic_second_otsame_avg{i},
TFR_harmonic_first_avg{i});
TFR_difference_random{i} = ft_math(cfg, TFR_random_second_otsame_avg{i},
TFR_random_first_avg{i});

end

TFR_difference_harmonic_granavg = ft_freqgrandaverage(cfg, TFR_difference_harmonic{:});
TFR_difference_random_granavg = ft_freqgrandaverage(cfg, TFR_difference_random{:});
%

% perform the statistical test using randomization and a clustering approach
% using the ft_freqstatistics function
cfg = [];
cfg_neighb.method = 'template';
cfg.template = 'easycapM3.mat';
cfg.neighbourdist = 0;
cfg.latency = 'all';
cfg.frequency = [15 32];
cfg.channel = [1:30]; % see FT_CHANNELSELECTION
cfg.avgovertime = 'no';
cfg.avgoverfreq = 'yes';
cfg.avgoverchan = 'no';
cfg.statistic = 'ft_statfun_depsamplesT';
cfg.clusteralpha = 0.05;
cfg.clusterstatistic = 'maxsum';
cfg.minnbchan = 0;
cfg.tail = 0;
cfg.clustertail = 0;
cfg.alpha = 0.05;
cfg.numrandomization = 5000;
cfg.correctm = 'cluster';
cfg.correcttail = 'alpha';
cfg.method = 'montecarlo';

filepath=strcat('C:\Users\Filiz\Desktop\Courses\Yeditepe\Seminar\Sound_Experiment5_EEG\Final
_Set\Data\EEG_data\');
filename=strcat(filepath, 'Fieldtrip_data\Neighbours\easycapM3_neighbours.mat');
load(filename, 'neighbours');
cfg.neighbours = neighbours; % the neighbours specify for each sensor with

```

% which other sensors it can form clusters

```

cfg.design      = [
    1 2 3 4 5 6 7 8 9 10 11 1 2 3 4 5 6 7 8 9 10 11 % subject number
    1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 ]; % condition number

cfg.uvar = 1; % "subject" is unit of observation
cfg.ivar = 2; % "condition" is the independent variable
stat_TFR_difference = ft_freqstatistics(cfg, TFR_difference_harmonic{1:11},
TFR_difference_random{1:11});
%

cfg = [];
cfg.alpha = 0.05;
cfg.parameter = 'stat';
cfg.layout = 'easycapM3.mat';
ft_clusterplot(cfg, stat_TFR_first);
%

cfg = [];
cfg.alpha = 0.05;
cfg.parameter = 'stat';
cfg.layout = 'easycapM3.mat';
ft_clusterplot(cfg, stat_TFR_second);
%

cfg = [];
cfg.alpha = 0.05;
cfg.parameter = 'stat';
cfg.layout = 'easycapM3.mat';
ft_clusterplot(cfg, stat_TFR_second_notsame);
%

cfg = [];
cfg.alpha = 0.05;
cfg.parameter = 'stat';
cfg.layout = 'easycapM3.mat';
ft_clusterplot(cfg, stat_TFR_difference);

cfg = [];
cfg.alpha = 0.05;
cfg.parameter = 'stat';
cfg.layout = 'easycapM3.mat';
ft_clusterplot(cfg, stat_TFR_random_MM);

cfg = [];
cfg.alpha = 0.05;
cfg.parameter = 'stat';
cfg.layout = 'easycapM3.mat';
ft_clusterplot(cfg, stat_TFR_harmonic_MM);

```

Elimination of P300 for beta power analysis

```

P300_harmonic=zeros(30,250);
P300_random=zeros(30,250);
for i=1:1:11
    for ch=1:1:30
        for t=1:1:250
            P300_harmonic(ch,t)=difference_harmonic{1,i}.avg(ch,1149+t)+P300_harmonic(ch,t);
            P300_random(ch,t)=difference_random{1,i}.avg(ch,1149+t)+P300_random(ch,t);
        end
    end
end

for ch=1:1:30
    for t=1:1:250
        P300_harmonic(ch,t)=P300_harmonic(ch,t)/11;
        P300_random(ch,t)=P300_random(ch,t)/11;
    end
end

for i=1:1:11
    for ch=1:1:30
        for t=1:1:250
            difference_harmonic{1,i}.avg(ch,1149+t)=difference_harmonic{1,i}.avg(ch,1149+t)-
P300_harmonic(ch,t);
            difference_random{1,i}.avg(ch,1149+t)=difference_random{1,i}.avg(ch,1149+t)-
P300_random(ch,t);
        end
    end
end

```

Between Subject Variance of P300 on anterior electrodes

```

P300_harmonic1=zeros(11,30,50);
P300_random1=zeros(11,30,50);
for i=1:1:11
    for ch=1:1:30
        for t=1:1:250
            P300_harmonic1(i,ch,t)=difference_harmonic{1,i}.avg(ch,1249+t);
            P300_random1(i,ch,t)=difference_random{1,i}.avg(ch,1249+t);
        end
    end
end

P300_harmonic_avg_ch=zeros(11,250);
P300_random_avg_ch=zeros(11,250);
for i=1:1:11
    for t=1:1:250

P300_harmonic_avg_ch(i,t)=(P300_harmonic1(i,2,t)+P300_harmonic1(i,5,t)+P300_harmonic1(i,6,
t)+P300_harmonic1(i,7,t)+P300_harmonic1(i,10,t)+P300_harmonic1(i,11,t)+P300_harmonic1(i,12
,t))/7;

```

```
P300_random_avg_ch(i,t)=(P300_random1(i,2,t)+P300_random1(i,5,t)+P300_random1(i,6,t)+P300_random1(i,7,t)+P300_random1(i,10,t)+P300_random1(i,11,t)+P300_random1(i,12,t))/7;  
    end  
end
```



APPENDIX D: ETICHAL APPROVAL FORM



T.C. YEDİTEPE ÜNİVERSİTESİ

Sayı : 37068608-6100-15- 1628

12/03/2019

Konu: Klinik Araştırmalar
Etik kurul Başvurusu hk.

İlgili Makama (Funda Yıldırım)

Yeditepe Üniversitesi Mühendislik Fakültesi Bilişsel Sinirbilimi Bölümü Dr. Funda Yıldırım'ın sorumlu olduğu **“Ses Olaylarında Gürültünün ve Zamansal Sıralamanın Kaynak Tespitine Etkisi ve EEG Yardımıyla Algısal Etkilerinin Araştırılması”** isimli araştırma projesine ait Klinik Araştırmalar Etik Kurulu (KAEK) Başvuru Dosyası (1598 kayıt Numaralı KAEK Başvuru Dosyası), Yeditepe Üniversitesi Klinik Araştırmalar Etik Kurulu tarafından **11.03.2019** tarihli toplantıda incelenmiştir.

Kurul tarafından yapılan inceleme sonucu, yukarıdaki isimi belirtilen çalışmanın yapılmasının etik ve bilimsel açıdan uygun olduğuna karar verilmiştir (**KAEK Karar No: 980**).

Prof. Dr. Turgay ÇELİK

Yeditepe Üniversitesi
Klinik Araştırmalar Etik Kurulu Başkanı