Optimal Range of Frequencies for Estimating Tempo

Hiromi Kageyama and Garrett Archer

CSCE A470 - Project Write-up

May 2, 2024

Table of Contents

Abstract

The ability to accurately predict song tempo from a subject's electroencephalogram (EEG) data plays a crucial role in understanding music perception and cognition. This project set out to determine if this data could be used for accurate prediction without using machine learning methods, which require training and testing data that may not be available, and which frequency range is the most effective for this prediction. We used data recorded from 20 participants who listened to a selection of 10 songs. We created a program in MatLab that optimized this data for every song and plotted the most significant peaks for each of the three ranges that we had chosen on a categorical scatter plot. The results were then compared to the actual tempo of that song. For most songs we found this method to be quite accurate, especially in the 3-7 Hz range, though a couple songs proved to have more difficult tempos to predict. This program not only can find use in its current state for determining tempo from pre-recorded data, but future developments could see it being adapted for real-time tempo estimation or for use in EEG-based music composition.

1. Introduction

This project was developed for use by Dr. Blair Kaneshiro. Dr. Kaneshiro and Dr. Heidari were interested in trying to find out if a tool could be developed that would be able to predict the tempo of a song using the EEG data of a participant and, if it was possible, what the optimal range of frequencies would be for this prediction. This data would be useful for their further study in the field of neuroscience.

Both Dr. Kaneshiro and Dr. Heidari has experience using MatLab. We also communicated with them frequently to fulfill their requirements for this project.

2. Project Overview

2.1 Additional Involved Parties

This project was developed with the assistance of our clients/project advisors Dr. Blair Kaneshiro and Dr. Masoumeh Heidari Kapourchali.

Dr. Blair Kaneshiro is currently the Director of Research & Development with the Stanford Educational Neuroscience Initiative (SENSI) in the Graduate School of Education and an Adjunct Professor at Stanford's Center for Computer Research in Music and Acoustics (CCRMA). Additionally she is an adjunct professor at the University of Alaska Anchorage in the College of Engineering and Senior Scientific Consultant with soundBrilliance, LLC.

Dr. Masoumeh Heidari Kapourchali is an assistant professor in the Department of Computer Science and Computer Engineering at University of Alaska Anchorage.

2.2 Planning Process

At the beginning of our project we created a gantt chart to outline our schedule, where we set aside the first half of the semester (approximately eight weeks) for research into EEG data structures and theory, machine learning theory and implementation, and for learning how to code in MATLAB. Additionally, we planned weekly Zoom meetings every Wednesday from 5:30 - 6:00 PM with Dr. Kaneshiro and Dr. Heidari, which were attended throughout the duration of the project's development. The second half of the semester we set aside for writing the code of the program as well as any additional research as needed.

3. Project Requirements

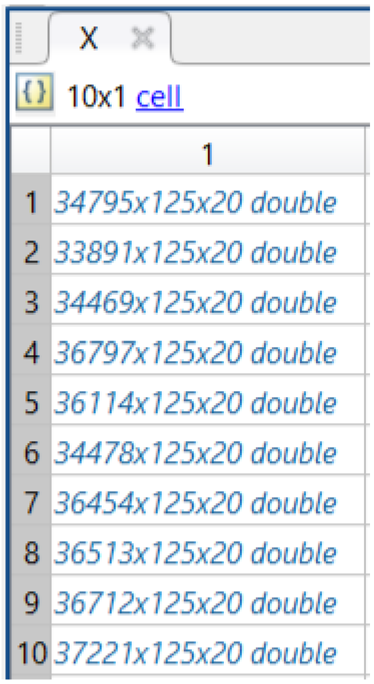
We used MatLab for our coding environment. The files of the cleaned EEG data are in .mat format. Our code will be able to analyze and use the EEG data to find the most optimal range of frequencies, in hertz, to improve the accuracy of estimating the tempo of a song.

#	Song Title	Artist	Tempo (BPM)	Tempo (Hz)	min:sec
1	First Fires	Bonobo	55.97	0.9328	4:38
2	Oino	LA Priest	69.44	1.1574	4:31
3	Tiptoes	Daedelus	74.26	1.2376	4:36
4	Careless Love	Croquet Club	82.42	1.3736	4:54
5	Lebanese Blonde	Thievery Corporation	91.46	1.5244	4:49
6	Canopée	Polo & Pan	96.15	1.6026	4:36
7	Doing Yoga	Kazy Lambist	108.70	1.8116	4:52
8	Until the Sun Needs to Rise	Rüfüs du Sol	120.00	2.0000	4:52
9	Silent Shout	The Knife	128.21	2.1368	4:54
10	The Last Thing You Should Do	David Bowie	150.00	2.5000	4:58

Table 3.1 Basic information of each song [8]

4. Design

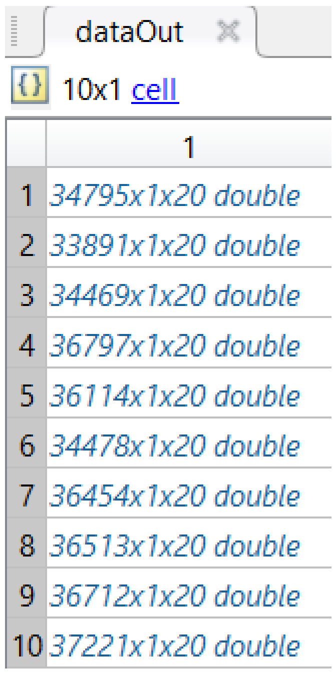
After loading in the EEG data [13] – format of the data shown in figure 4.1 – we spatially optimized it to limit the data from 125 electrodes to 1 electrode [14], shown in figure 4.2. Next, we converted the 3D matrix into a 2D matrix by using a function called squeeze so we could pass the data through other functions, shown in figure 4.3. Once the data was ready, we created a 3D matrix holding the largest peak of the participant between 3-7, 3-8, and 3-9 Hz for each song; the largest peak of each participant for each song in range 3-7 Hz shown in figure 4.4.


X

10x1 cell

	1
1	34795x125x20 double
2	33891x125x20 double
3	34469x125x20 double
4	36797x125x20 double
5	36114x125x20 double
6	34478x125x20 double
7	36454x125x20 double
8	36513x125x20 double
9	36712x125x20 double
10	37221x125x20 double

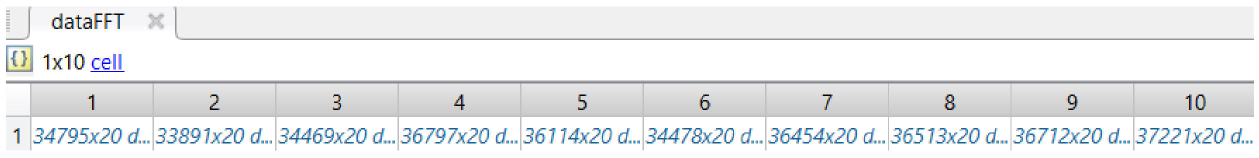
Figure 4.1 Loaded song data (song data x electrode count x participant number)


dataOut

10x1 cell

	1
1	34795x1x20 double
2	33891x1x20 double
3	34469x1x20 double
4	36797x1x20 double
5	36114x1x20 double
6	34478x1x20 double
7	36454x1x20 double
8	36513x1x20 double
9	36712x1x20 double
10	37221x1x20 double

Figure 4.2 Spatially optimized the data (song data x electrode count x participant number)

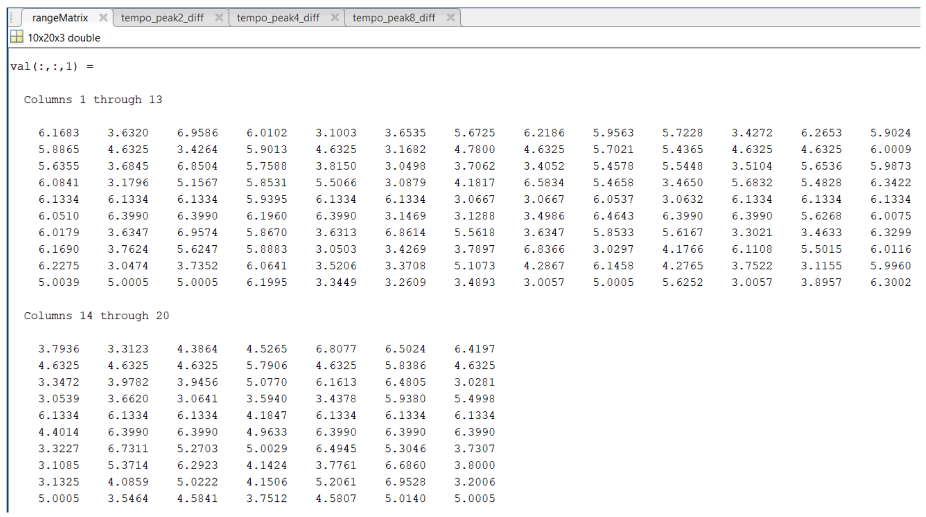


dataFFT

1x10 cell

	1	2	3	4	5	6	7	8	9	10
1	34795x20 d...	33891x20 d...	34469x20 d...	36797x20 d...	36114x20 d...	34478x20 d...	36454x20 d...	36513x20 d...	36712x20 d...	37221x20 d...

Figure 4.3 Converted the data from a 3D matrix into a 2D matrix (song data x participant number)



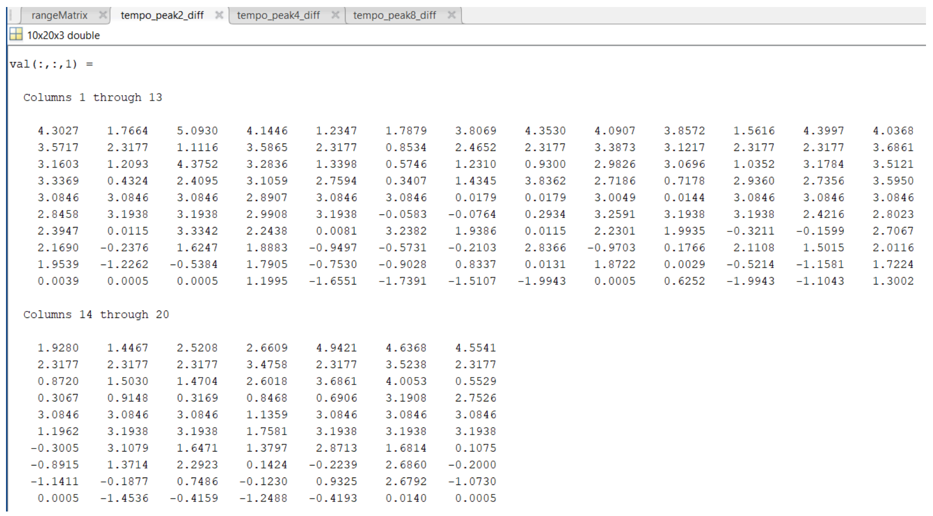
rangeMatrix | tempo_peak2_diff | tempo_peak4_diff | tempo_peak8_diff

10x20x3 double

```
val(:,:,1) =

  Columns 1 through 13

    6.1683    3.6320    6.9586    6.0102    3.1003    3.6535    5.6725    6.2186    5.9563    5.7228    3.4272    6.2653    5.9024
    5.8865    4.6325    3.4264    5.9013    4.6325    3.1682    4.7800    4.6325    5.7021    5.4365    4.6325    4.6325    6.0009
    5.6355    3.6845    6.8504    5.7588    3.8150    3.0498    3.7062    3.4052    5.4578    5.5448    3.5104    5.6536    5.9873
    6.0841    3.1796    5.1567    5.8531    5.5066    3.0879    4.1817    6.5834    5.4658    3.4650    5.6832    5.4828    6.3422
    6.1334    6.1334    6.1334    5.9395    6.1334    6.1334    3.0667    3.0667    6.0537    3.0632    6.1334    6.1334    6.1334
    6.0510    6.3990    6.3990    6.1960    6.3990    3.1469    3.1288    3.4986    6.4643    6.3990    6.3990    5.6268    6.0075
    6.0179    3.6347    6.9574    5.8670    3.6313    6.8614    5.5618    3.6347    5.8533    5.6167    3.3021    3.4633    6.3299
    6.1690    3.7624    5.6247    5.8883    3.0503    3.4269    3.7897    6.8366    3.0297    4.1766    6.1108    5.5015    6.0116
    6.2275    3.0474    3.7352    6.0641    3.5206    3.3708    5.1073    4.2867    6.1458    4.2765    3.7522    3.1155    5.9960
    5.0039    5.0005    5.0005    6.1995    3.3449    3.2609    3.4893    3.0057    5.0005    5.6252    3.0057    3.8957    6.3002

  Columns 14 through 20

    3.7936    3.3123    4.3864    4.5265    6.8077    6.5024    6.4197
    4.6325    4.6325    4.6325    5.7906    4.6325    5.8386    4.6325
    3.3472    3.9782    3.9456    5.0770    6.1613    6.4805    3.0281
    3.0539    3.6620    3.0641    3.5940    3.4378    5.9380    5.4998
    6.1334    6.1334    6.1334    4.1847    6.1334    6.1334    6.1334
    4.4014    6.3990    6.3990    4.9633    6.3990    6.3990    6.3990
    3.3227    6.7311    5.2703    5.0029    6.4945    5.3046    3.7307
    3.1085    5.3714    6.2923    4.1424    3.7761    6.6860    3.8000
    3.1325    4.0859    5.0222    4.1506    5.2061    6.9528    3.2006
    5.0005    3.5464    4.5841    3.7512    4.5807    5.0140    5.0005
```

Figure 4.4 Largest peak of each participant for each song between 3-7 Hz

The next step involved getting the difference of each participant's peak with the actual tempo of the song multiplied by 2, 4, and 8, since a participant can tap along to the beat of the song at twice the speed or four times the speed – shown in figure 4.5. The difference was recorded in three 3D matrices for each tempo multiplied by 2, 4, and 8, in the format of (song number x participant number x range). Next, we compared the values of each peak subtracted by the tempo of the song multiplied by 2, 4, and 8 and put the value closest to zero, from the 3 matrices, in a new 3D matrix called predictedTempo – shown in figure 4.6.

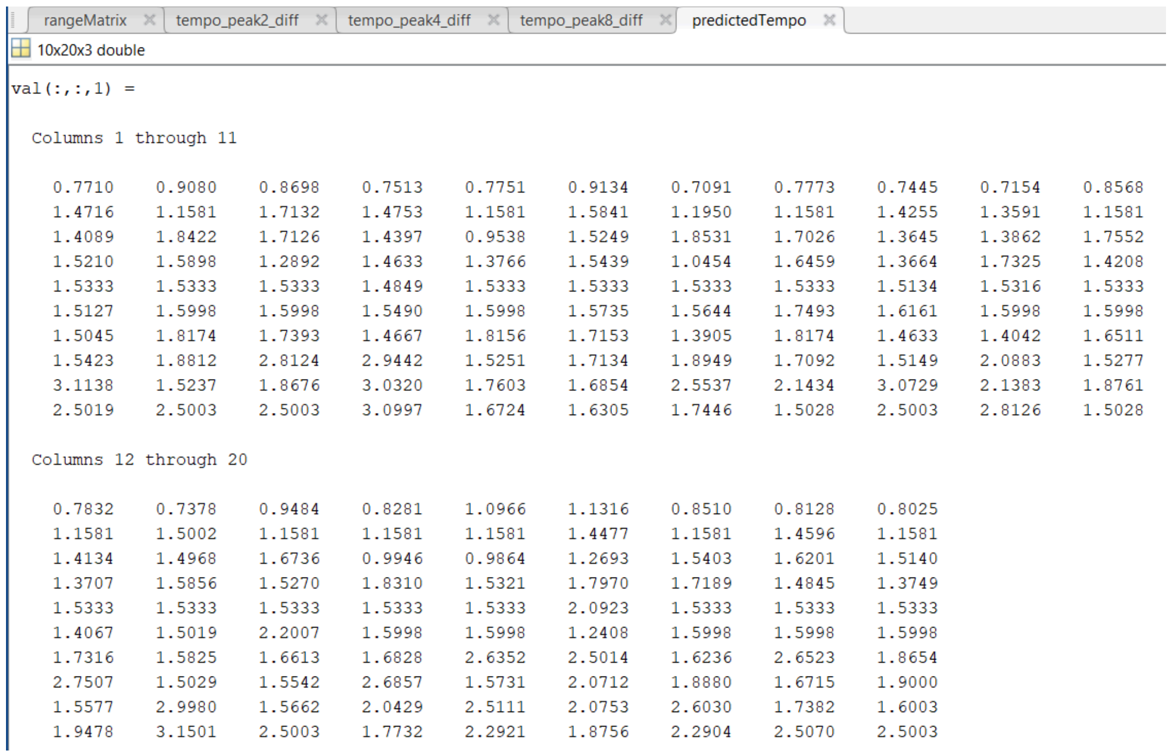


rangeMatrix | tempo_peak2_diff | tempo_peak4_diff | tempo_peak8_diff

10x20x3 double

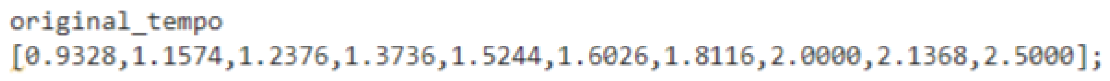
```
val(:,:,1) =

  Columns 1 through 13

    4.3027    1.7664    5.0930    4.1446    1.2347    1.7879    3.8069    4.3530    4.0907    3.8572    1.5616    4.3997    4.0368
    3.5717    2.3177    1.1116    3.5865    2.3177    0.8534    2.4652    2.3177    3.3873    3.1217    2.3177    2.3177    3.6861
    3.1603    1.2093    4.3752    3.2836    1.3398    0.5746    1.2310    0.9300    2.9826    3.0696    1.0352    3.1784    3.5121
    3.3369    0.4324    2.4095    3.1059    2.7594    0.3407    1.4345    3.8362    2.7186    0.7178    2.9360    2.7356    3.5950
    3.0846    3.0846    3.0846    2.8907    3.0846    3.0846    0.0179    0.0179    3.0049    0.0144    3.0846    3.0846    3.0846
    2.8458    3.1938    3.1938    2.9908    3.1938   -0.0583   -0.0764    0.2934    3.2591    3.1938    3.1938    2.4216    2.8023
    2.3947    0.0115    3.3342    2.2438    0.0081    3.2382    1.9386    0.0115    2.2301    1.9935   -0.3211   -0.1599    2.7067
    2.1690   -0.2376    1.6247    1.8883   -0.9497   -0.5731   -0.2103    2.8366   -0.9703    0.1766    2.1108    1.5015    2.0116
    1.9539   -1.2262   -0.5384    1.7905   -0.7530   -0.9028    0.8337    0.0131    1.8722    0.0029   -0.5214   -1.1581    1.7224
    0.0039    0.0005    0.0005    1.1995   -1.6551   -1.7391   -1.5107   -1.9943    0.0005    0.6252   -1.9943   -1.1043    1.3002

  Columns 14 through 20

    1.9280    1.4467    2.5208    2.6609    4.9421    4.6368    4.5541
    2.3177    2.3177    2.3177    3.4758    2.3177    3.5238    2.3177
    0.8720    1.5030    1.4704    2.6018    3.6861    4.0053    0.5529
    0.3067    0.9148    0.3169    0.8468    0.6906    3.1908    2.7526
    3.0846    3.0846    3.0846    1.1359    3.0846    3.0846    3.0846
    1.1962    3.1938    3.1938    1.7581    3.1938    3.1938    3.1938
   -0.3005    3.1079    1.6471    1.3797    2.8713    1.6814    0.1075
   -0.8915    1.3714    2.2923    0.1424   -0.2239    2.6860   -0.2000
   -1.1411   -0.1877    0.7486   -0.1230    0.9325    2.6792   -1.0730
    0.0005   -1.4536   -0.4159   -1.2488   -0.4193    0.0140    0.0005
```

Figure 4.5 Difference of largest peak subtracted by twice the tempo of the song



```
val(:,:,1) =

  Columns 1 through 11

    0.7710    0.9080    0.8698    0.7513    0.7751    0.9134    0.7091    0.7773    0.7445    0.7154    0.8568
    1.4716    1.1581    1.7132    1.4753    1.1581    1.5841    1.1950    1.1581    1.4255    1.3591    1.1581
    1.4089    1.8422    1.7126    1.4397    0.9538    1.5249    1.8531    1.7026    1.3645    1.3862    1.7552
    1.5210    1.5898    1.2892    1.4633    1.3766    1.5439    1.0454    1.6459    1.3664    1.7325    1.4208
    1.5333    1.5333    1.5333    1.4849    1.5333    1.5333    1.5333    1.5333    1.5134    1.5316    1.5333
    1.5127    1.5998    1.5998    1.5490    1.5998    1.5735    1.5644    1.7493    1.6161    1.5998    1.5998
    1.5045    1.8174    1.7393    1.4667    1.8156    1.7153    1.3905    1.8174    1.4633    1.4042    1.6511
    1.5423    1.8812    2.8124    2.9442    1.5251    1.7134    1.8949    1.7092    1.5149    2.0883    1.5277
    3.1138    1.5237    1.8676    3.0320    1.7603    1.6854    2.5537    2.1434    3.0729    2.1383    1.8761
    2.5019    2.5003    2.5003    3.0997    1.6724    1.6305    1.7446    1.5028    2.5003    2.8126    1.5028

  Columns 12 through 20

    0.7832    0.7378    0.9484    0.8281    1.0966    1.1316    0.8510    0.8128    0.8025
    1.1581    1.5002    1.1581    1.1581    1.1581    1.4477    1.1581    1.4596    1.1581
    1.4134    1.4968    1.6736    0.9946    0.9864    1.2693    1.5403    1.6201    1.5140
    1.3707    1.5856    1.5270    1.8310    1.5321    1.7970    1.7189    1.4845    1.3749
    1.5333    1.5333    1.5333    1.5333    1.5333    2.0923    1.5333    1.5333    1.5333
    1.4067    1.5019    2.2007    1.5998    1.5998    1.2408    1.5998    1.5998    1.5998
    1.7316    1.5825    1.6613    1.6828    2.6352    2.5014    1.6236    2.6523    1.8654
    2.7507    1.5029    1.5542    2.6857    1.5731    2.0712    1.8880    1.6715    1.9000
    1.5577    2.9980    1.5662    2.0429    2.5111    2.0753    2.6030    1.7382    1.6003
    1.9478    3.1501    2.5003    1.7732    2.2921    1.8756    2.2904    2.5070    2.5003
```

Figure 4.6 Predicted tempo of each song using EEG data gathered from the participants



```
original_tempo
[0.9328,1.1574,1.2376,1.3736,1.5244,1.6026,1.8116,2.0000,2.1368,2.5000];
```

Figure 4.7 The actual tempo of each song where the first element is song 1's tempo and then song 2's tempo continuing until song 10

The final step is to plot the results in the predictedTempo matrix. We used a categorical scatterplot with dots representing the participant's predicted tempo overlaid on the boxplot [15]. The y-axis represents the predicted tempo and the x-axis represents the range of frequencies in hertz. Samples of the results can be found on figures 6.3 - 6.5.

5. Software Development Process

We started coding towards the second half of the semester. The amount of time spent coding, testing, and debugging was about the same. A few challenges we faced were learning how to code in MatLab and researching machine learning models only to change the direction of our project in the final quarter of the semester. We met with our project advisors every Wednesday on Zoom from 5:30 - 6:30 PM and received help when needed.

6. Results

Our program was completed on time and, overall, it works quite well. We were able to get an overview of the data for all participants for each song, an example of which can be seen in figure 6.1. Additionally, we also have a view of the median over 0-15 Hz range, an example of which is shown in figure 6.2. Most songs found a reasonably close fit of predicted to actual tempo in the 3-7 range. A couple were even nearly a perfect fit over all three ranges, as seen in figure 6.3. However, we did have a couple songs that didn't fit as well as we had hoped, as can be seen in figure 6.4. These were most likely caused by the subjects having difficulty finding the tempo of the song, though noise might also be a culprit in the larger 3-9 range. The majority of the songs, however, showed a fit that was very near to the actual tempo without being a perfect fit, such as what is shown in figure 6.5.

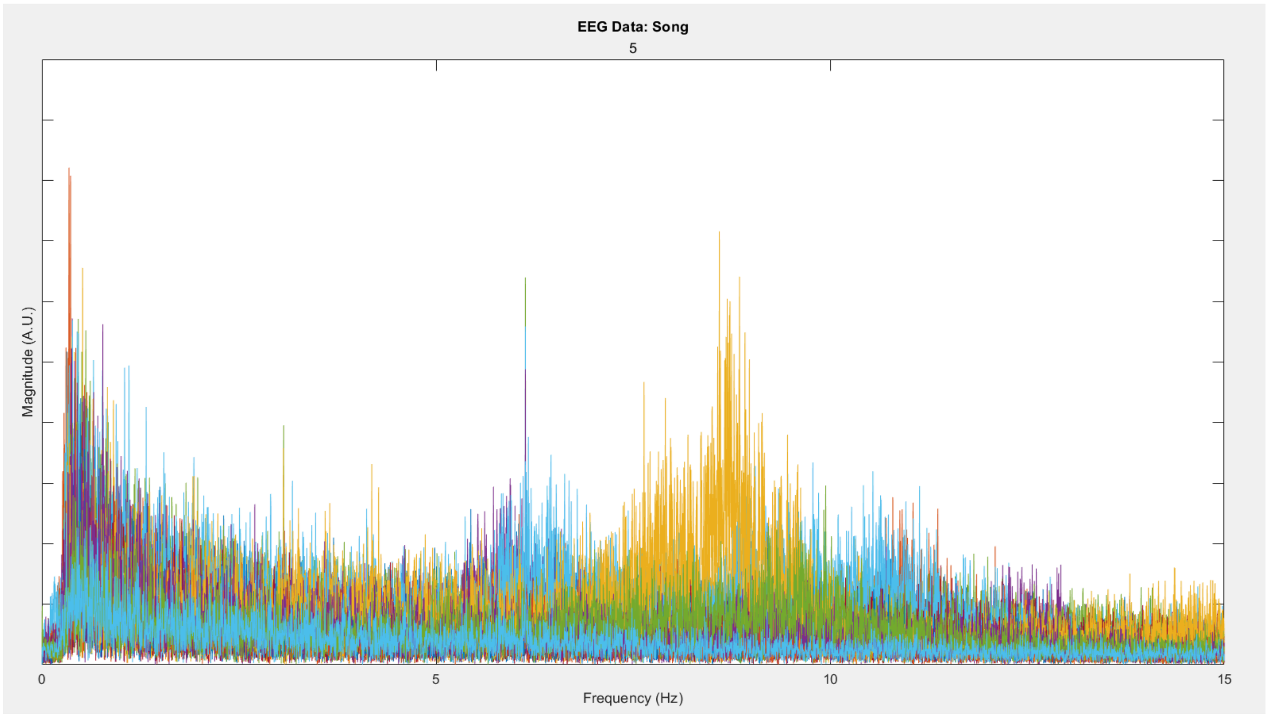


Figure 6.1. All participants' EEG data for Song 5. Each color represents a different participant

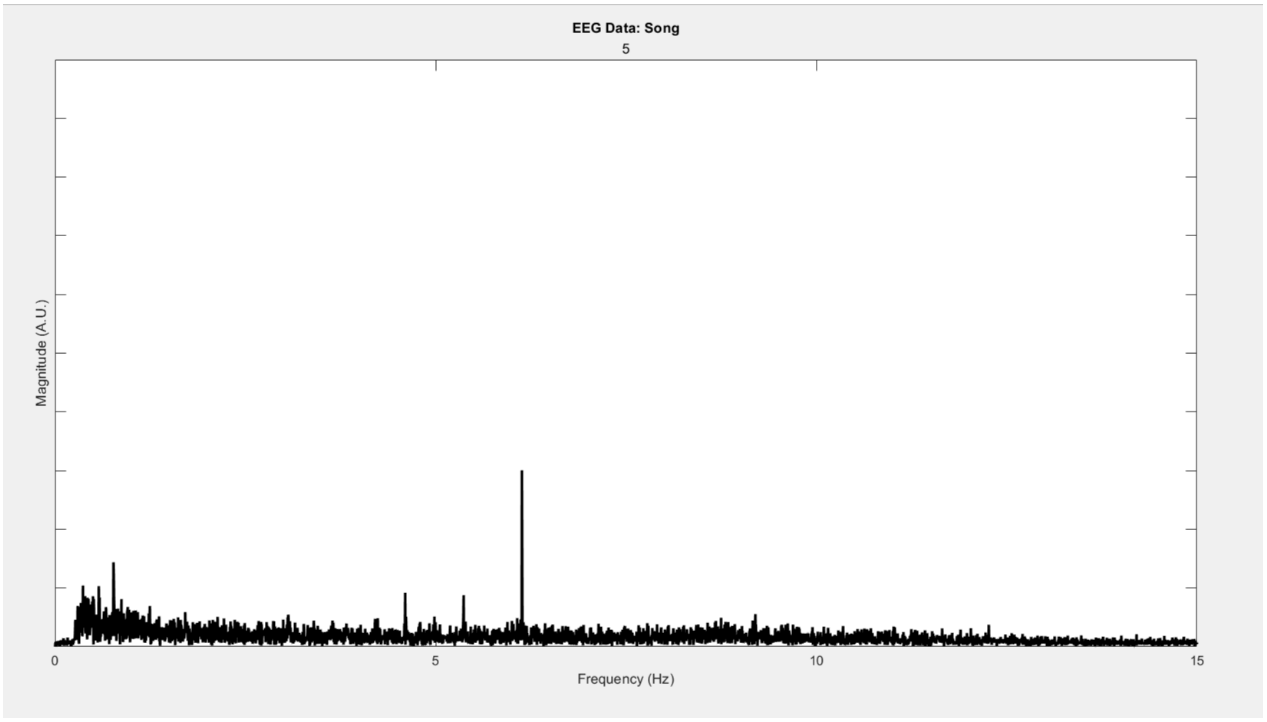


Figure 6.2. Song 5’s median line showing peaks that were used for tempo estimation

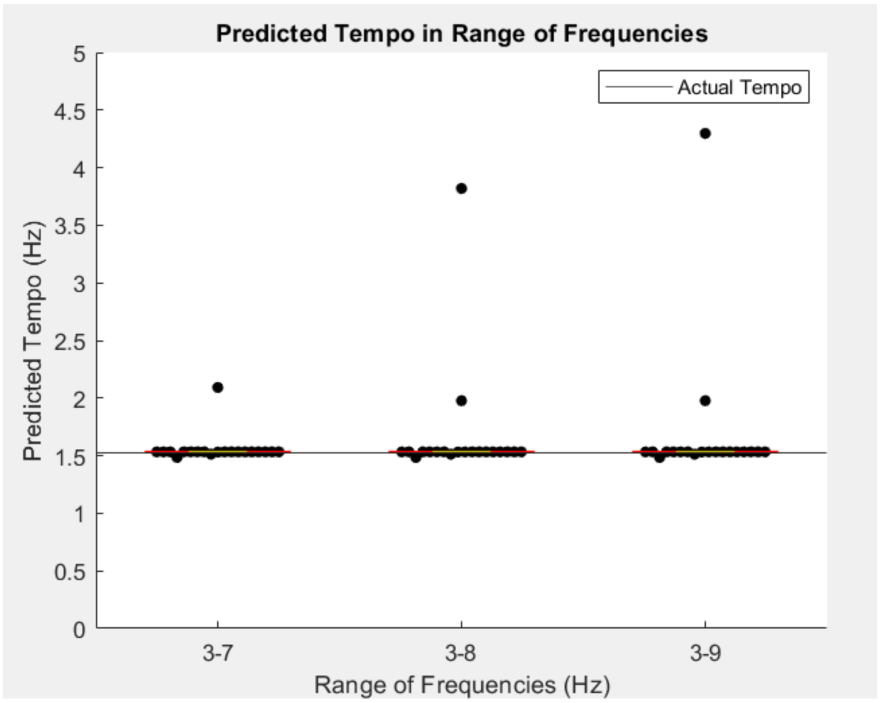


Figure 6.3. Song 5 shows the majority of the subjects’ peaks being a near-perfect fit over all three selected ranges

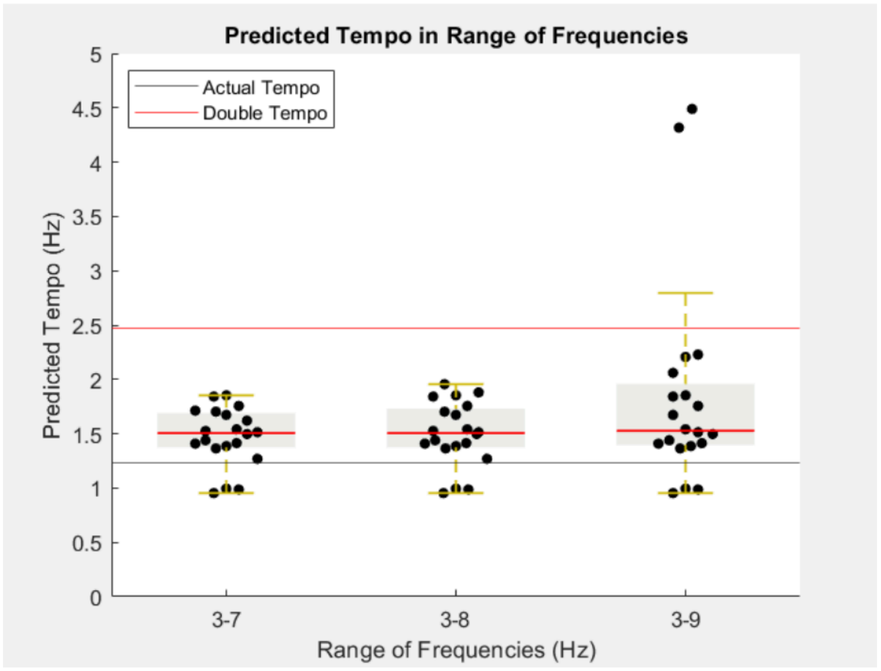


Figure 6.4. Song 3 shows the majority of the subjects' peaks at a higher frequency than the song's tempo.

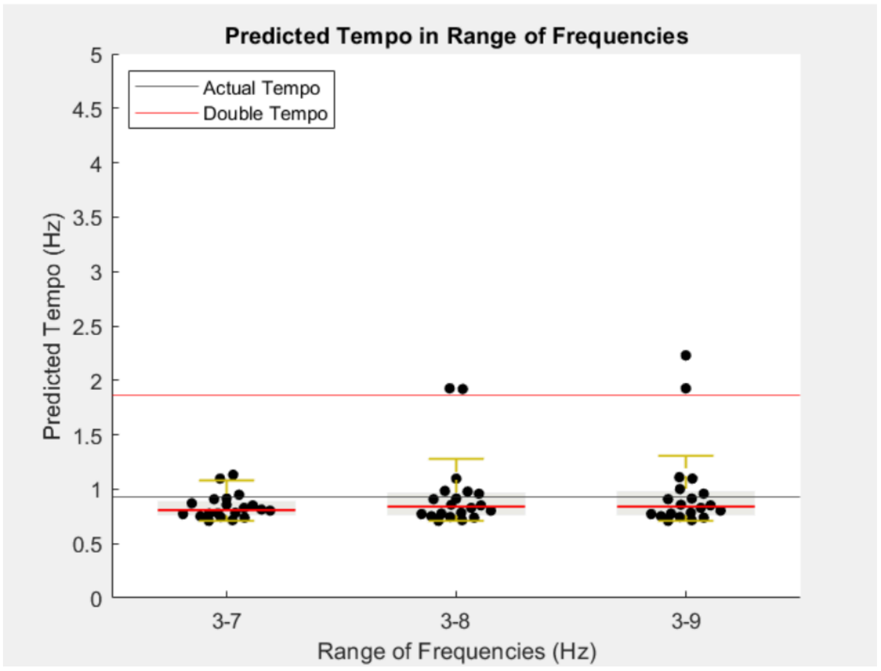


Figure 6.5. Song 1 shows more average results with a close, but not perfect, fit.

7. Summary and Conclusions

The best range of frequency to improve the accuracy of estimating the tempo of a song using EEG data was between 3-7 Hz.

8. References

8.1 Documents

[1] Schreiber, Hendrik, et al. “Music Tempo Estimation: Are We Done Yet?” *Transactions of the International Society for Music Information Retrieval*, vol. 3, no. 1, 24 Aug. 2020, pp. 111–125, doi: 10.5334/tismir.43.

[2] Nozaradan, S., et al. “Tagging the Neuronal Entrainment to Beat and Meter.” *The Journal of Neuroscience*, vol. 31, no. 28, 13 July 2011, pp. 10234–10240, doi: 10.1523/jneurosci.0411-11.2011.

[3] Pandey, Pankaj, et al. “Predicting dominant beat frequency from brain responses while listening to music.” *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 9 Dec. 2021, pp. 3058–3064, doi:10.1109/bibm52615.2021.9669750.

[4] Vinay, Ashvala, et al. “Mind the Beat: Detecting Audio Onsets from EEG Recordings of Music Listening.” *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6 June 2021, pp. 231–235, doi: 10.1109/icassp39728.2021.9414245.

[5] Dmochowski, Jacek P., et al. “Maximally Reliable Spatial Filtering of Steady State Visual Evoked Potentials.” *NeuroImage*, vol. 109, Apr. 2015, pp. 63–72, doi: 10.1016/j.neuroimage.2014.12.078.

[6] Kaneshiro, Blair, et al. “Natural Music Evokes Correlated EEG Responses Reflecting Temporal Structure and Beat.” *NeuroImage*, vol. 214, July 2020, p. 116559, doi:10.1016/j.neuroimage.2020.116559.

[7] Dmochowski, Jacek P., Jason J. Ki, et al. “Extracting Multidimensional Stimulus-Response Correlations Using Hybrid Encoding-Decoding of Neural Activity.” *NeuroImage*, vol. 180, Oct. 2018, pp. 134–146, doi:10.1016/j.neuroimage.2017.05.037.

[8] Losorelli, Steven, et al. "NMED-T: A Tempo-Focused Dataset of Cortical and Behavioral Responses to Naturalistic Music." ISMIR. Vol. 3. 2017.

[9] Kaneshiro, Blair, and Jacek P. Dmochowski. "Neuroimaging Methods for Music Information Retrieval: Current Findings and Future Prospects." ISMIR. 2015.

[10] Stober, Sebastian, Thomas Prätzlich, and Meinard Müller. "Brain Beats: Tempo Extraction from EEG Data." ISMIR. 2016.

[11] Dmochowski, Jacek P., Paul Sajda, et al. “Correlated Components of Ongoing EEG Point to Emotionally Laden Attention – A Possible Marker of Engagement?” *Frontiers in Human Neuroscience*, vol. 6, 16 May 2012, p. 112, doi: 10.3389/fnhum.2012.00112.

8.2 Code Repositories

[12] Blair Kaneshiro. computeFFTFrequencyAxis.m *BKanMatEEGToolbox*, commit fcbabea49fe3b4018e11efabd8be57a3c85a5c23, 2020. *GitHub*, github.com/blairkan/BKanMatEEGToolbox/blob/master/computeFFTFrequencyAxis.m

[13] Blair Kaneshiro. loadMultipleFiles.m *NMED_T_intro_matlab*, commit d1f9b69acf7f28f2b4ea06b5bb30b8d98135d069, 2024. *GitHub*, github.com/blairkan/NMED_T_intro_matlab/blob/main/loadMultipleFiles.m

[14] Blair Kaneshiro. rcaRun125_parpoolAlready2021.m *NMED_T_intro_matlab*, commit 65ab5eed1a0be2ab3e72946e8444ad0ecce93dfa, 2024. *GitHub*, github.com/blairkan/NMED_T_intro_matlab/blob/main/HelperFiles/rcaRun125_parpoolAlready2021.m

[15] AbstractGeek. CategoricalScatterplot.m *CategoricalScatterplot*, commit 138a4f5c91aab19f6f32d65882375a9a5c6d08e0, 2018. *GitHub*, github.com/AbstractGeek/CategoricalScatterplot/blob/master/CategoricalScatterplot.m

[16] Dmochowski, Jacek P. rca *rca*, 2019. *GitHub*, https://github.com/dmochow/rca/

Appendix A: Actions Taken From Code Review/Heuristic Evaluation

Defect	**Action Taken**
“No units for magnitude”	No action taken. Units are Arbitrary Units
“for loop without indentation or statements”	All for loops are now indented
“Function names are confusing, not sure what they do (rcaRun125_parpoolAlready2021)”	No action taken. Function name defined in repo
“It would be helpful to see the plots of actual vs predicted values or the evaluation metrics.”	Charts now show both predicted values for each range plus a line for actual value
“Multiple colors are reused in graph, each color should be unique”	No action taken. Chart is intended for rough overview, not final analysis
“comment formatting is a little messy and inconsistent”	Most comment formatting was revised. A few in-line comments remain at beginning for setting variables
“Training and testing sets are manually split -- MATLAB has functions to partition / cross-validate so data is fully utilized and results are more accurate”	No action taken. No longer needed after pivot in design
“magic numbers”	No action taken. Any magic numbers were taken from referenced papers

Appendix B: User manual

Step 1:
Install MatLab to run the code, git clone the github repositories necessary to run the code, and make sure you have EEG data available. You can find the link of every repository we used in the reference section. The data we used can be found on the link below. Download the first 10 items with “song” in the beginning of the file name.

https://exhibits.stanford.edu/data/catalog/jn859kj8079

Step 2:
Add the repositories into MatLab’s pathway and make sure to select “Add with subfolders” to include the path of every file in the repository. Search how to add paths on MatLab using YouTube if needed.

Step 3:
Run each section of the code from top to bottom.

- The first section of the code loads in the EEG data of each song and creates a 3D matrix in the format of (song data x electrode count x participant number).

- The second section spatially optimizes the data and calculates the weighted sum of each electrode choosing the electrode best suited for this project.

- The third section converts the 3D matrix into a 2D matrix so we could pass the data through useful functions. You could also uncomment lines 26-35 to see the EEG data of every participant overlaid on each other for each song.

- The fourth section gets the largest peak between 3 to 6+k, where k starts from 1 and goes up to 3 to increase the range of the x-axis. You could uncomment lines 59-64 to see the EEG data of each participant. Changing the value of k and i to an integer is recommended when trying to look at each participant’s EEG data in a specific range of frequencies for a specific song.

- The fifth section gets the difference of the predicted peak (largest peak of a participant) with the actual tempo of the song multiplied by 2, 4, and 8 and records each number in a separate matrix for each multiplier.

- The sixth section compares the 3 matrices that hold the difference between the predicted tempo and the actual tempo multiplied by 2, 4, and 8 and puts the peak of the difference closest to 0 in a new matrix called predictedTempo, (e.g, the difference in matrix tempo_peak2_diff is closest to 0 for participant j=7 for song 6 rather than tempo_peak4_diff or tempo_peak8_diff).

- The seventh and last section plots down the results of each song

Appendix C: Source Code

Hiromi wrote most of the code with help from Dr. Kaneshiro and Dr. Kapourchali. Garrett wrote the code from line 118 onwards, for plotting the results, with little edits by Hiromi. Additional code for the original machine learning design was written by Garrett with edits by Hiromi, but was later depreciated then removed in favor of the new direction of the program.

Before switching the direction of our project, Garrett wrote code for the linear regression model; however, due to changing our direction the code was not included in the project.

Dependencies:
We used 4 git repositories to be able to spatially filter the EEG data, use some functions, and to plot the results.

We included the BKanMatEEGToolbox repo for a function that computes the FFT frequency axis. This converts the x-axis from time to frequency.

We included the CategoricalScatterplot repo for plotting the results of our code.

We included the NMED_T_intro_matlab repo for loading data and spatially filtering the data.

We included the rca repo since some of the functions in the NMED_T_intro_matlab repo require functions in the rca repo.