Ecole Polytechnique de Louvain

LGBIO2020 - Bioinstrumentation

Teaching staff :
Prof. Andre Moraux
Prof. Michel Verleysen
Assistant Anne-Sophie Collin

Group 1

Conet Brieuc - 11181800
Deside Guillaume - 39731800

Academic year 2021-2022

Contents

List of Figures

List of Tables

1 Introduction

In this project, we will build a binary classifier allowing us to predict if a patient has his eyes open or closed according to a specific set of data from an electroencephalogram (EEG). This EEG provided by the teaching staff is an assembly of 32 electrodes pasted on the scalp. Each electrode is named based on its relative position on the scalp, as visible in Fig. 1. The different names are included in the following variable:
ch_names = [Fp1, Fpz, Fp2, F7, F3, Fz, F4, F8, FC5, FC1, FC2, FC6, M1, T7, C3, CZ,C4, T8, M2, CP5, CP1, CP2, CP6, P7, P3, PZ, P4, P8, POz, O1, Oz, O2]

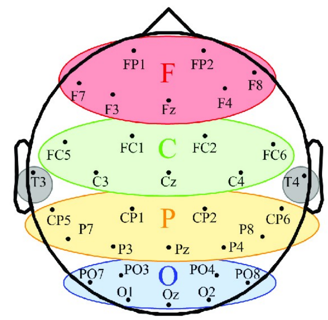


Figure 1: EEG montage (noncontractual)[2]

This report aims to explain our different choices for the processing, features extraction/selection and classifier. We will discuss the accuracy of the classifiers and their advantages and limitations. Some improvements or other options will be provided for a future project.

2 Design choices

In this section, we will discuss our choices of conception for each stage of the project.

2.1 Data preprocessing

Before applying classification, it is crucial to view and comprehend the type of data we are dealing with, as well as whether processing is necessary and what kind of processing is required.

2.1.1 Visualization data

Fig. 5 (in appendix A) presents the multichannel EEG observed. As can be seen, a temporal representation without any processing does not always allow for pattern recognition in the two situations, especially for inexperienced people. So preprocessing and features extraction are necessary.

2.1.2 Preprocessing

Some electrodes may be worthless in the case of our EEG classifier. So it is worth investigating the signal of each electrode with the target to see if we can delete some of them from the dataset based on its concordance with the target. Before removing an electrode, we also take account of its localization. We know that the visual cortex (that is the part of the brain that interests us more) is located in the occipital lobe. We deleted the signal for the following electrodes: FC1, FC2, and CZ.

When we look at Fourier transform of Fp1 signal2a and all signals B, we can observe a noise component of 50 Hz coming from digital equipment of measure. Since we use DWT to filter, we decided to filter also the delta's and gamma's waves because they do not bring any information in our specific case. In table 2, you can find the specific frequency bands used for EEG frequency spectrum analysis.

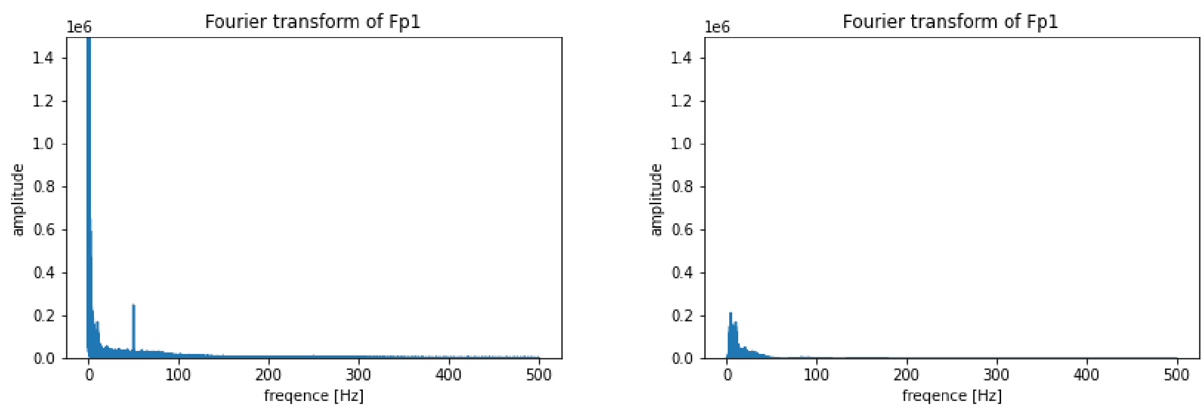
Table 1 presents frequencies corresponding to different levels of decomposition for Daubechies order four wavelets with a sampling frequency of 1000 Hz. It can be seen from Table 1 that the components A7 decomposition is within the delta range (1 Hz to 4 Hz), D7 decomposition is within the theta range (4 Hz to 8 Hz), D6 decomposition is within the alpha range (8 Hz to 13 Hz) and D5 decomposition is within the beta range (13 Hz to 30 Hz). In a normal EEG, lower level decompositions corresponding to higher frequencies have insignificant magnitudes. So we kept only D7,D6,D5. Therefore, We obtain for Fp1 the Fourier transform shown in fig. 2b and for other electrodes show in annexe B

Table 1: Frequencies corresponding to different levels of decomposition for Daubechies four filter wavelets with a sampling frequency of 1000 Hz.[4]

Decomposed signal	Frequency range (Hz)
D1	250-500
D2	125-250
D3	62.5-125
D4	31.25-62.5
D5	15.6-31.25
D6	7.8-15.6
D7	3.9-7.8
A7	0-3.9

Table 2: EEG frequency band ranges (Hz) for Dastidar reference.[1]

delta δ	theta θ	alpha α	beta β	gamma γ
0.5-4	4-8	8-13	13-25	25-40



(a) before filtering

(b) After filtering

Figure 2: Filtering with DWT

2.1.3 Construct training samples

To construct a training sample, we must divide data by segments of 500 ms.

First, we remove from the dataset data where the target was NaN because these data are useless for our classifiers.

The second thing we did was examine the latency to see whether there was a change in the EEG when the target changed from 0 to 1 or vice versa. We can observe on the Fig. 3 that there is a delay before observing a change in the EEG. This values can have a bad impact in our classifiers. We decided to do not keep values 1500 ms before and after change in the target value to avoid the influence of this data on the classifier.

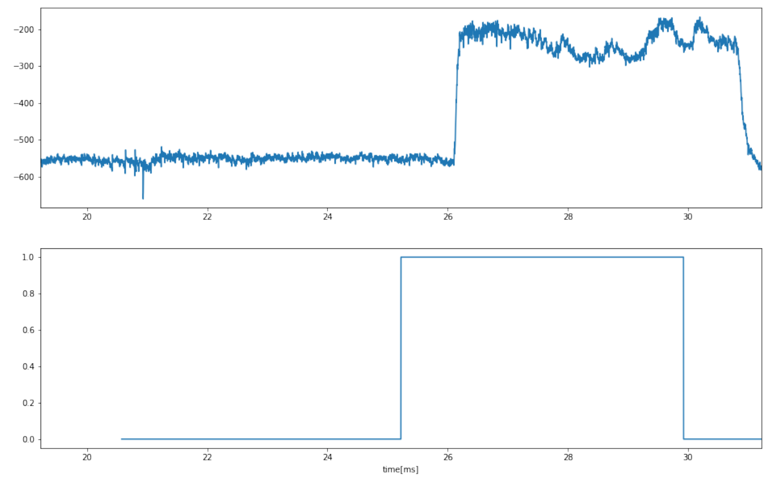


Figure 3: Delay observed between EEG and target for Fp1

The risk after the first two steps is that we do not have enough data to build robust classifiers. So, to increase the number of 500 ms segments, we create oversampling segments. This data augmentation allows for the reduction of overfitting and the stabilization of classifiers 2.4.

2.2 Features extraction

After the segmentation of our initial data, we must create the database used in the classifier. For that, we must lower the size of the training set. We have $2483 \times 500 \times 29$. Besides, feature extraction is a critical step since it has a direct impact on classification performance. When the features are not distinctive, it is less likely to achieve good classification performances.

There are several options for doing so. We choose to calculate the total powers (Equation (1)) for each kept sub-band (α,β,θ).

$$P(k) = |X(k)|^2 \tag{1}$$

When the new data set is ready, we must divide it in two different sets: Training and test sets. The test set contains 10 % of the new data set. The distribution of 0 and 1 in the new dataset is in the Table 3.

Table 3: Distribution of open and closed eyes in the new dataset

	Open eyes(0)	Closed eyes(1)
Number of	1311	1172

2.3 PCA/feature selection

In signal classification challenges, we know that the training time increases exponentially with number of features. Resources need also to be allocated for uninformative features. It can be interesting to reduce the number of features by creating new features from the existing ones (PCA) or by selecting subset of relevant features for processing, without any transformation (feature selection).

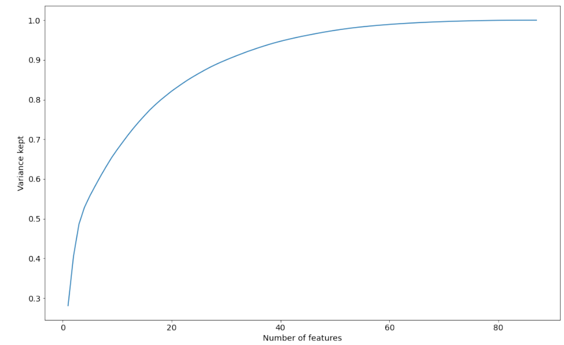


Figure 4: Explained variance ratio

In this project, we decided to focus on the *Principal Component Analysis(PCA)* but we also implemented the feature selection in the Jupyter Notebook. Even if we lose interpretability, PCA has the advantage of letting us know how much information we have kept from the original dataset. To know how much feature, we must keep, we use cumulative sum of explained variance ratio(Fig. 4). We observe we are to keep almost 90 % of the variance with just the half of the number of feature.
It seems to us important to compare different feature selections. We obtained results for five numbers of features: 2, 15, 20, 32 and 45. The variance kept for each number of features is in the Table 4.

2.4 Classification

We decide to compare results for four different classifiers for this project :

- K-nearest neighbours (kNN)
- Decision tree
- Neural network (MLP)
- Support Vector Machine (SVM)

In order to find the best hyperparameters of each classifier, we use GridSearchCV from sklearn[3]. We use the parameter *cv* to make K-fold Cross Validation. For our purposes, we used 10-fold cross-validation to train and test extracted features for all classifiers.
After finding the best hyperparameters, me measured classifier performance with the accuracy on the test set.

3 Resultats

All the classifications are performed in Python. We must test our models on the test set(10 % of dataset) after training them using the training set of data.
Looking at the classification performances given in Table 4, almost all classifier obtained 100 % of accuracy for each number of features. We can explain that by the fact that a few number of features allow for a decent separation of the two conditions. We do not need many parameters to keep a great variance between features. Besides, when we look at the dataset after the normalization step, we can observe that the sign differs between the two conditions for almost every feature.

Table 4: Accuracy of different classifiers

	Number of features	2	15	20	32	45
	Variance kept	0.49	0.78	0.83	0.92	0.97
kNN	KFold	0.91	0.99	0.99	0.99	1.00
	accuracy	0.92	0.99	1.00	1.00	1.00
Decision tree	KFold	0.91	0.97	0.97	0.96	0.96
	accuracy	0.92	0.98	0.97	0.96	0.96
MLP	KFold	0.91	0.99	1.00	1.00	1.00
	accuracy	0.92	1.00	1.00	1.00	1.00
SVM	KFold	0.91	0.99	1.00	1.00	1.00
	accuracy	0.93	0.99	1.00	1.00	1.00

For nearly every number of features, the MLP classifier provides 100 % accuracy. It appears to be the greatest model, but it does have several drawbacks that can be problematic in specific situations. In fact, the MLP classifier has a time of calculation which is rather long, that could pose problem if we want results in real time, and it often needs more resources.
Inversely, the Decision Tree is the worst model, but accuracy remains acceptable. As discussed in class, the Decision Tree has an overfitting problem.
In terms of time/resources and accuracy, the kNN classifier appears to be the best compromise.

4 Some possible improvements

Although we obtain satisfactory results, some modifications can be done to improve, simplify used method.
It is possible to use other features to create training and test set. It exists a lot of example in articles. For example, we can combine time and frequency features[1].
Data was provided by the teaching staff, so maybe we are not able to explain some details such as the origin of delay problem.
It will be interesting to know if another montage with a little number of electrodes can provide enough information to obtain as well as result.
A large public dataset may be used to allow validating the robustness of the proposed approach for classifying EEG signals.

5 Conclusion

In this report, we introduced a pattern recognition-based technique for open/closed eyes task categorization using EEG data.

Finally, the combination of DWT with PCA techniques provide a robust feature extraction approach for classification of cognitive tasks using EEG signals.

References

[1] Zafer Iscan, Zümray Dokur, and Tamer Demiralp. Classification of electroencephalogram signals with combined time and frequency features. *Expert Systems with Applications*, 38(8):10499–10505, 2011.

[2] Zhenhu Liang, Jiani Li, Xiaoyu Xia, Yong Wang, Xiaoli Li, Jianghong He, and Yang Bai. Long-range temporal correlations of patients in minimally conscious state modulated by spinal cord stimulation. *Frontiers in Physiology*, 9, 10 2018.

[3] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[4] Abdulhamit Subasi and Ergun Erçelebi. Classification of eeg signals using neural network and logistic regression. *Computer Methods and Programs in Biomedicine*, 78(2):87–99, 2005.

A EEG signals depending on the time

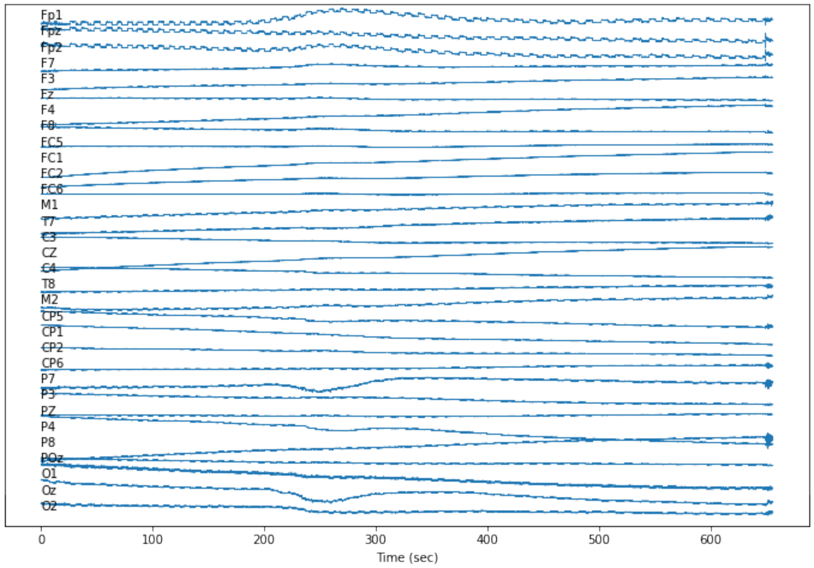


Figure 5: EEG visualization before preprocessing

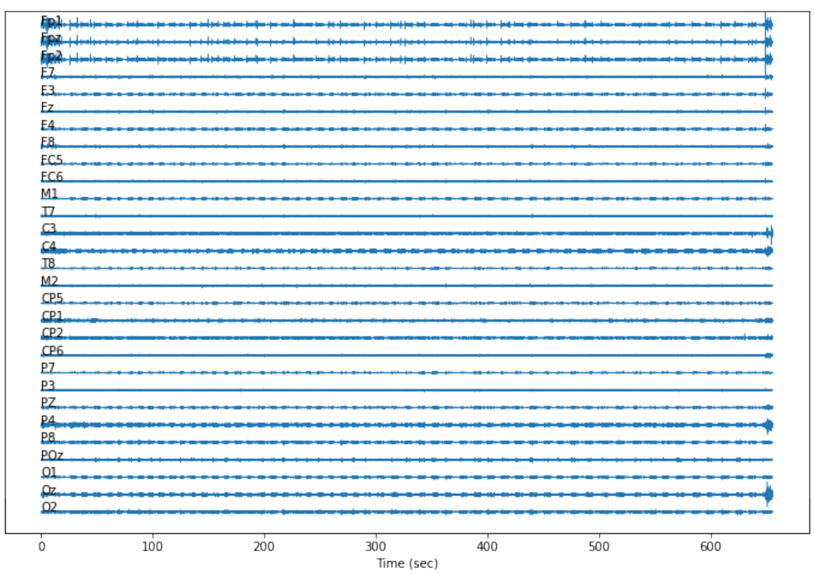


Figure 6: EEG visualization after preprocessing

B Graphs of Fourier Transform before and after filtering with DWT

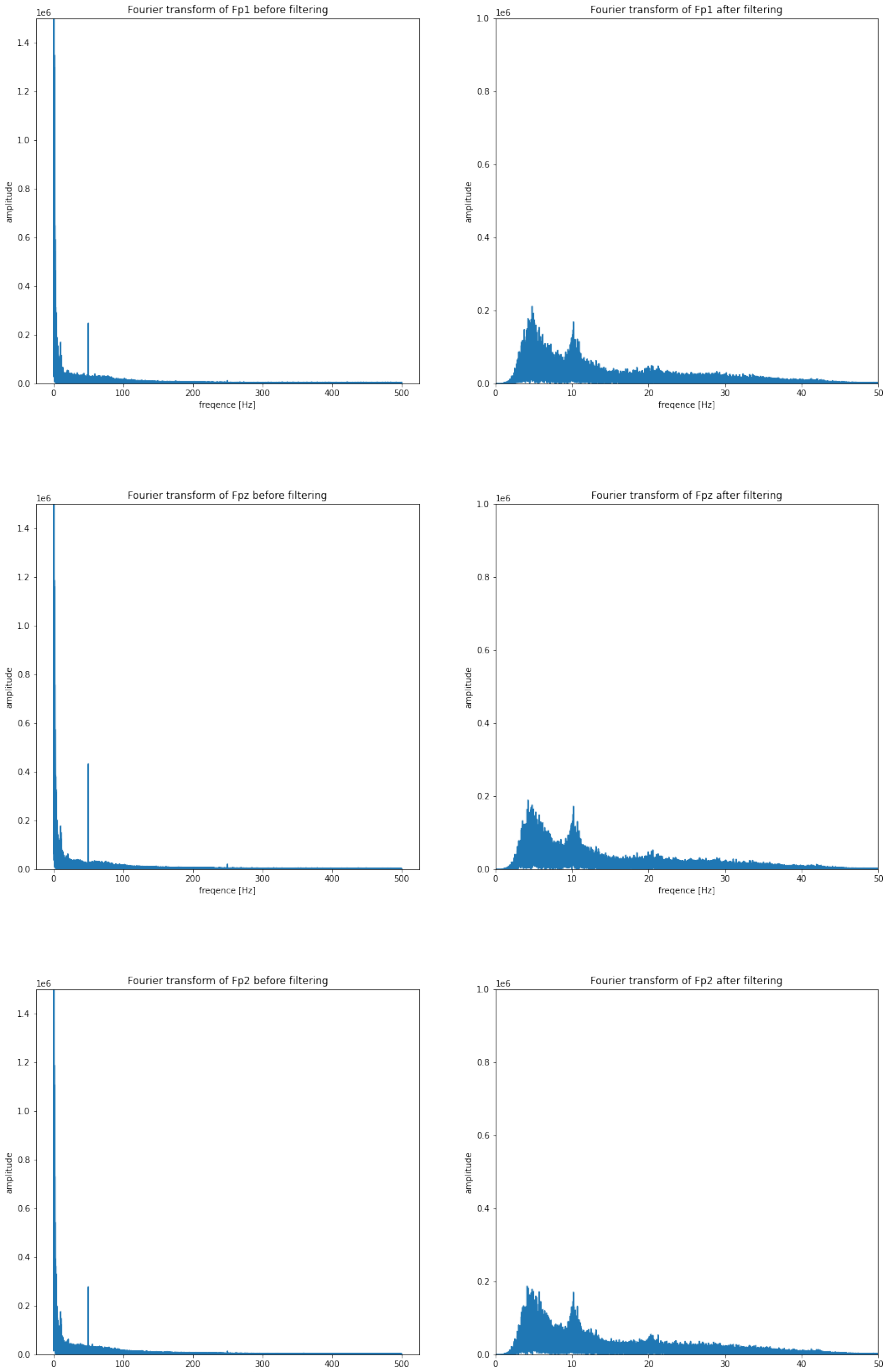


Figure 7: Comparison of Fourier transform before and after DWT (part 1)

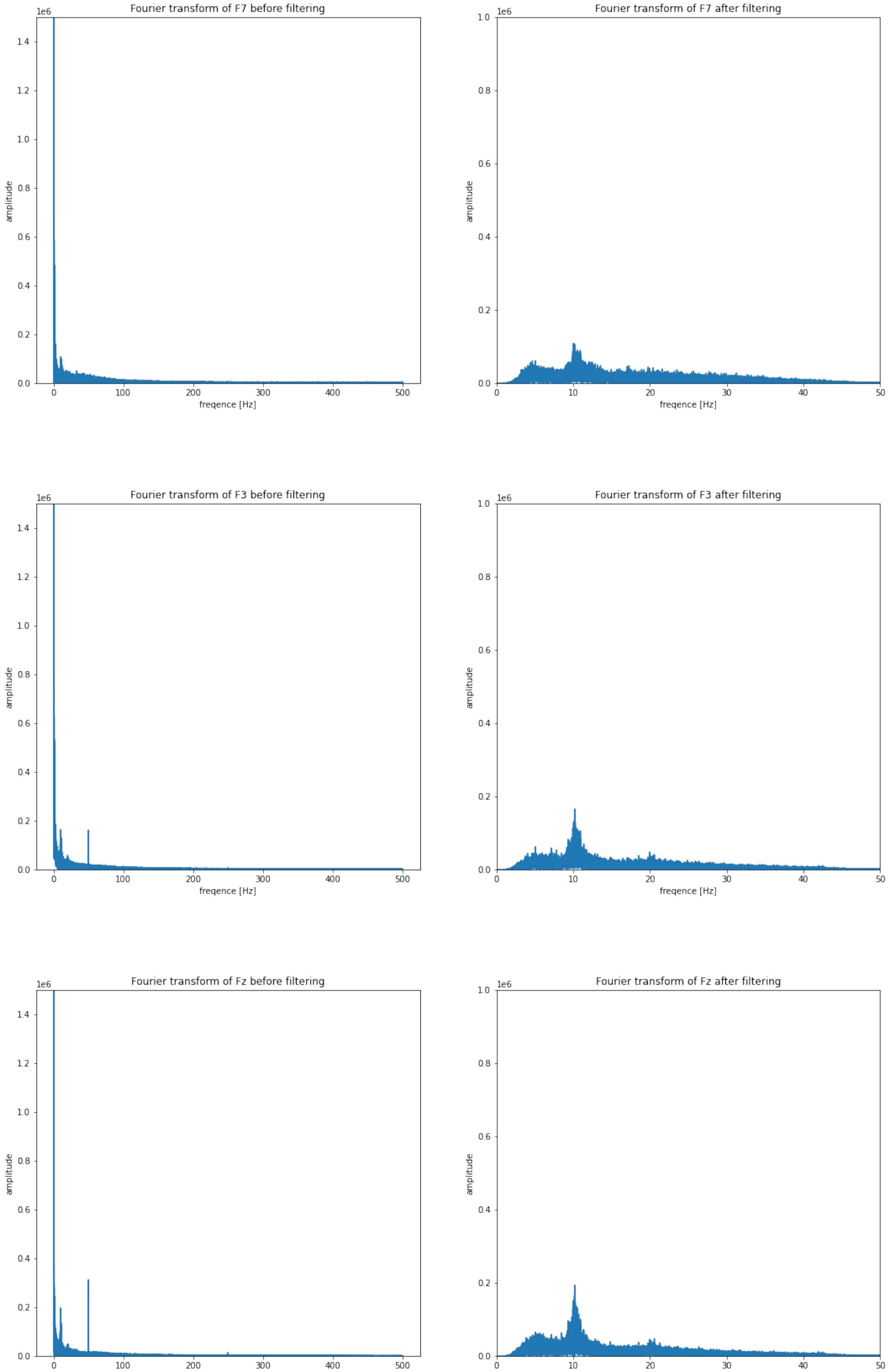


Figure 8: Comparison of Fourier transform before and after DWT (part 2)

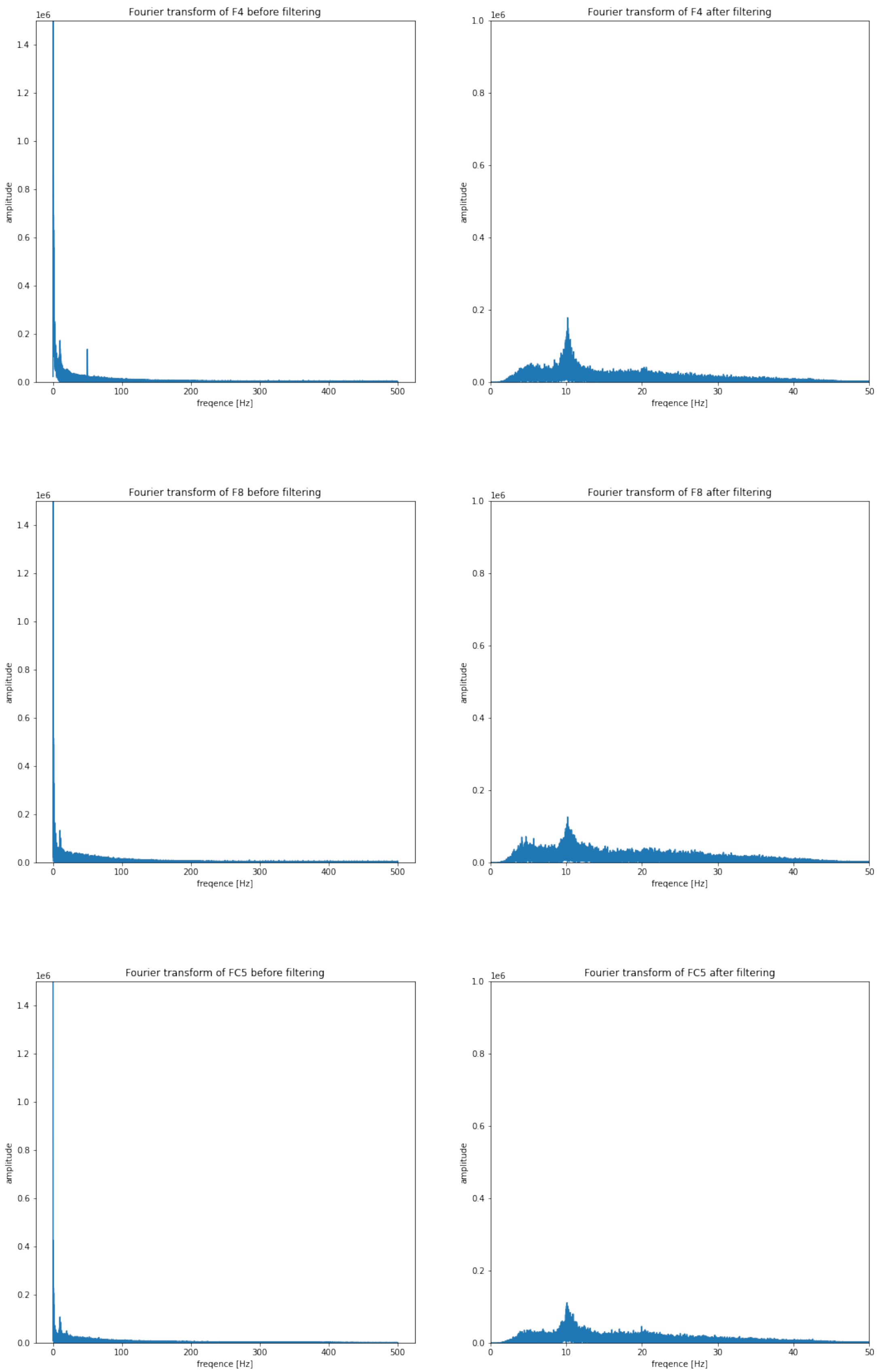


Figure 9: Comparison of Fourier transform before and after DWT (part 3)

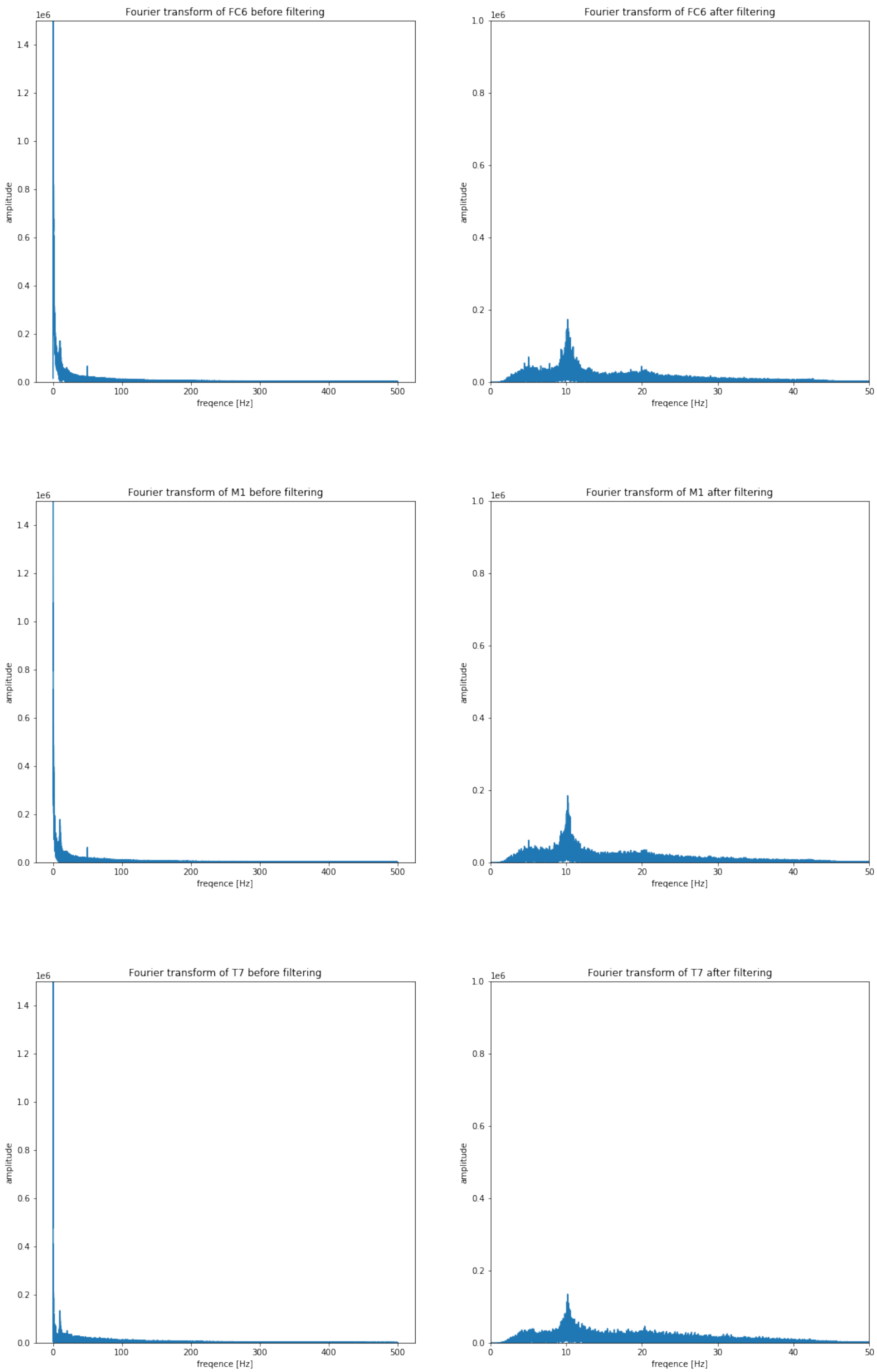


Figure 10: Comparison of Fourier transform before and after DWT (part 4)

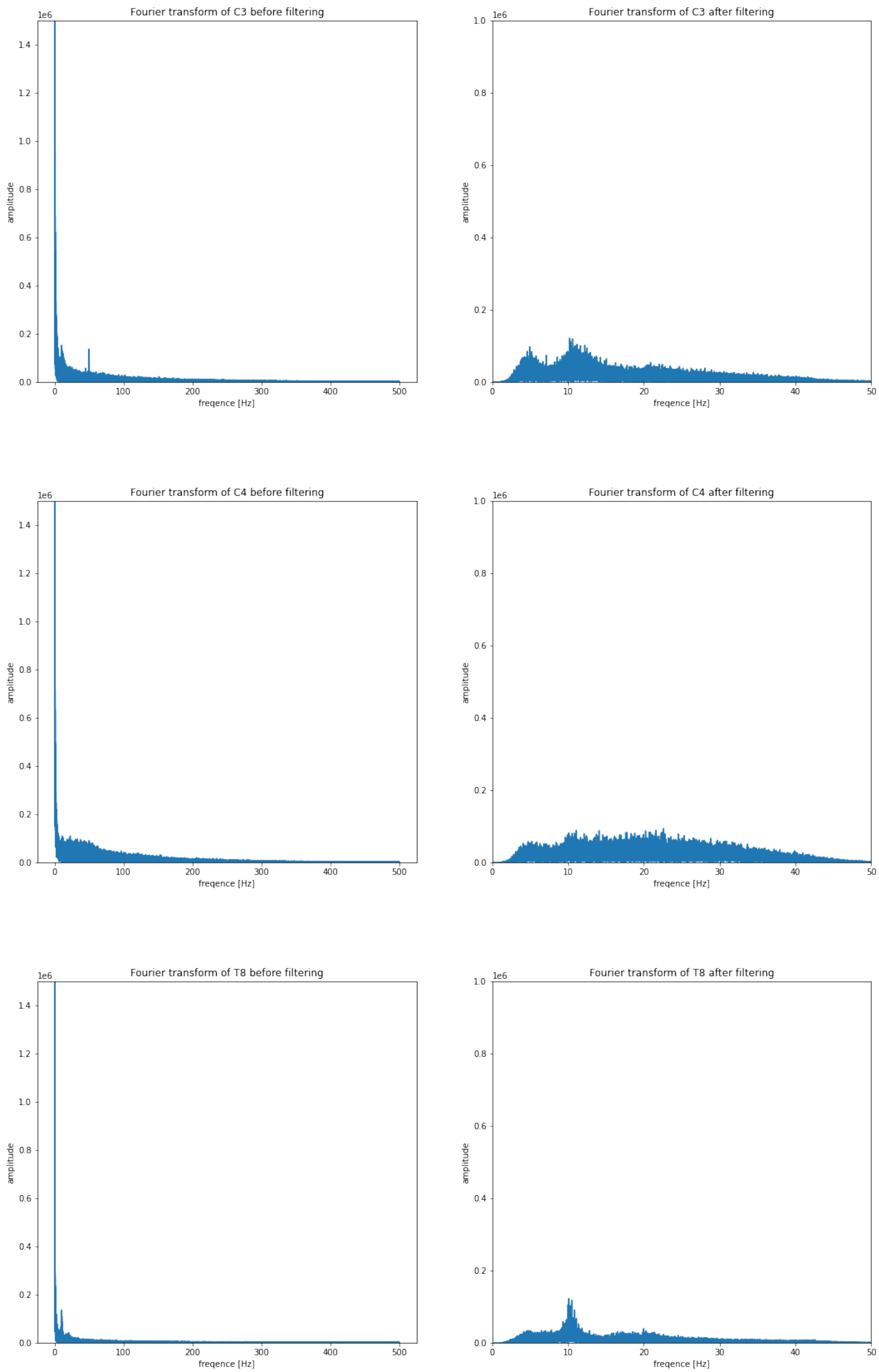


Figure 11: Comparison of Fourier transform before and after DWT (part 5)

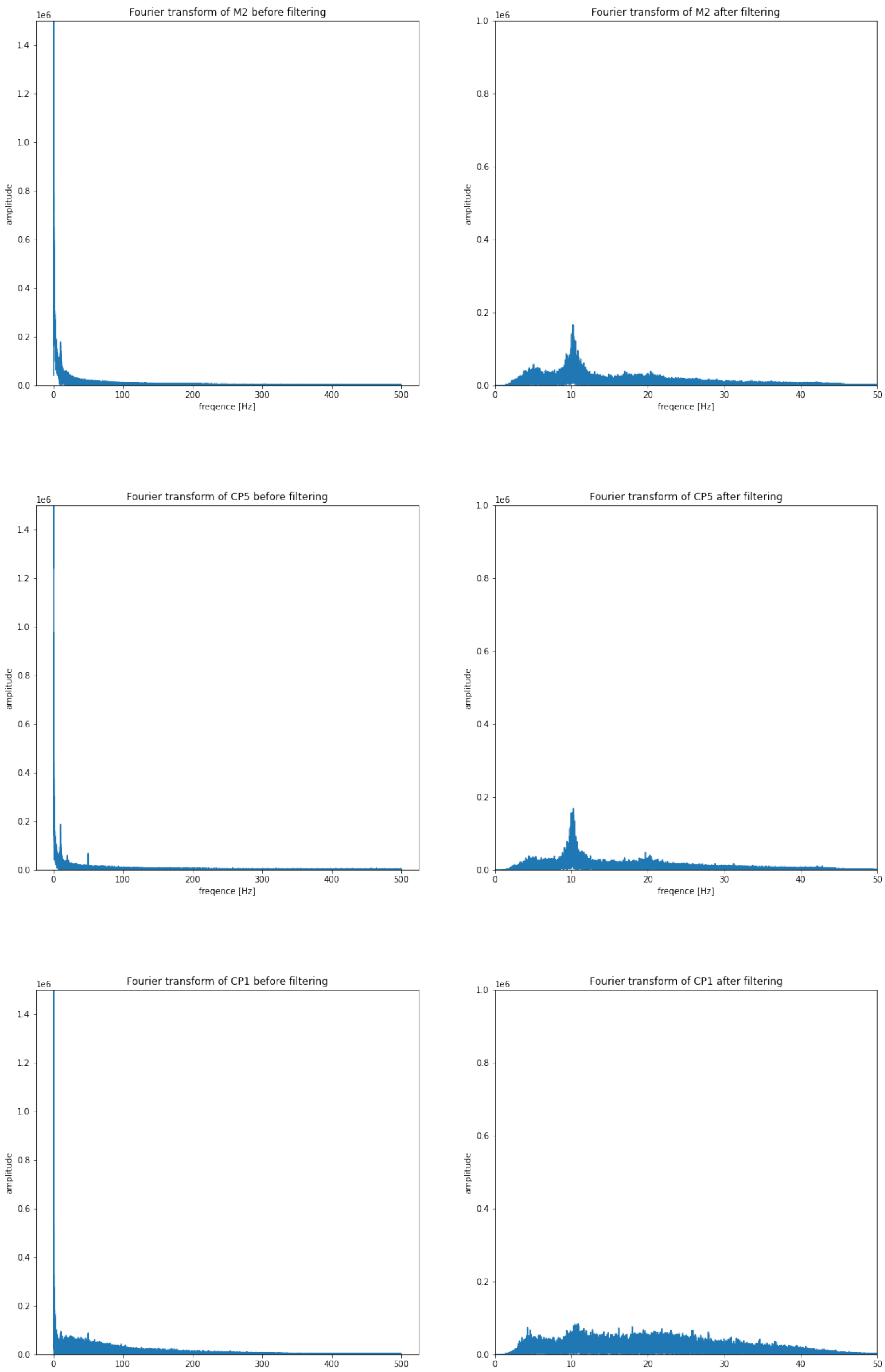


Figure 12: Comparison of Fourier transform before and after DWT (part 6)

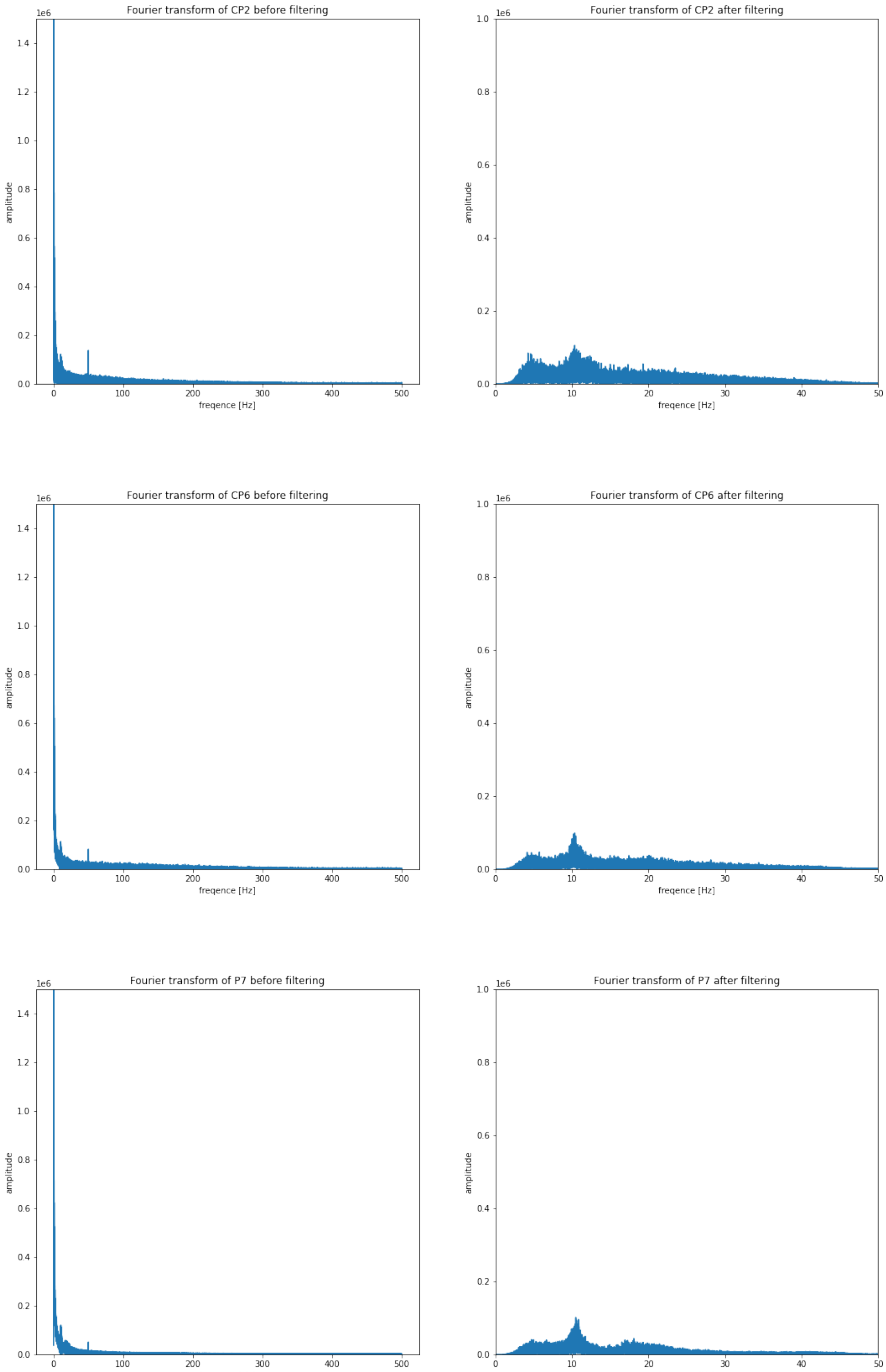


Figure 13: Comparison of Fourier transform before and after DWT (part 7)

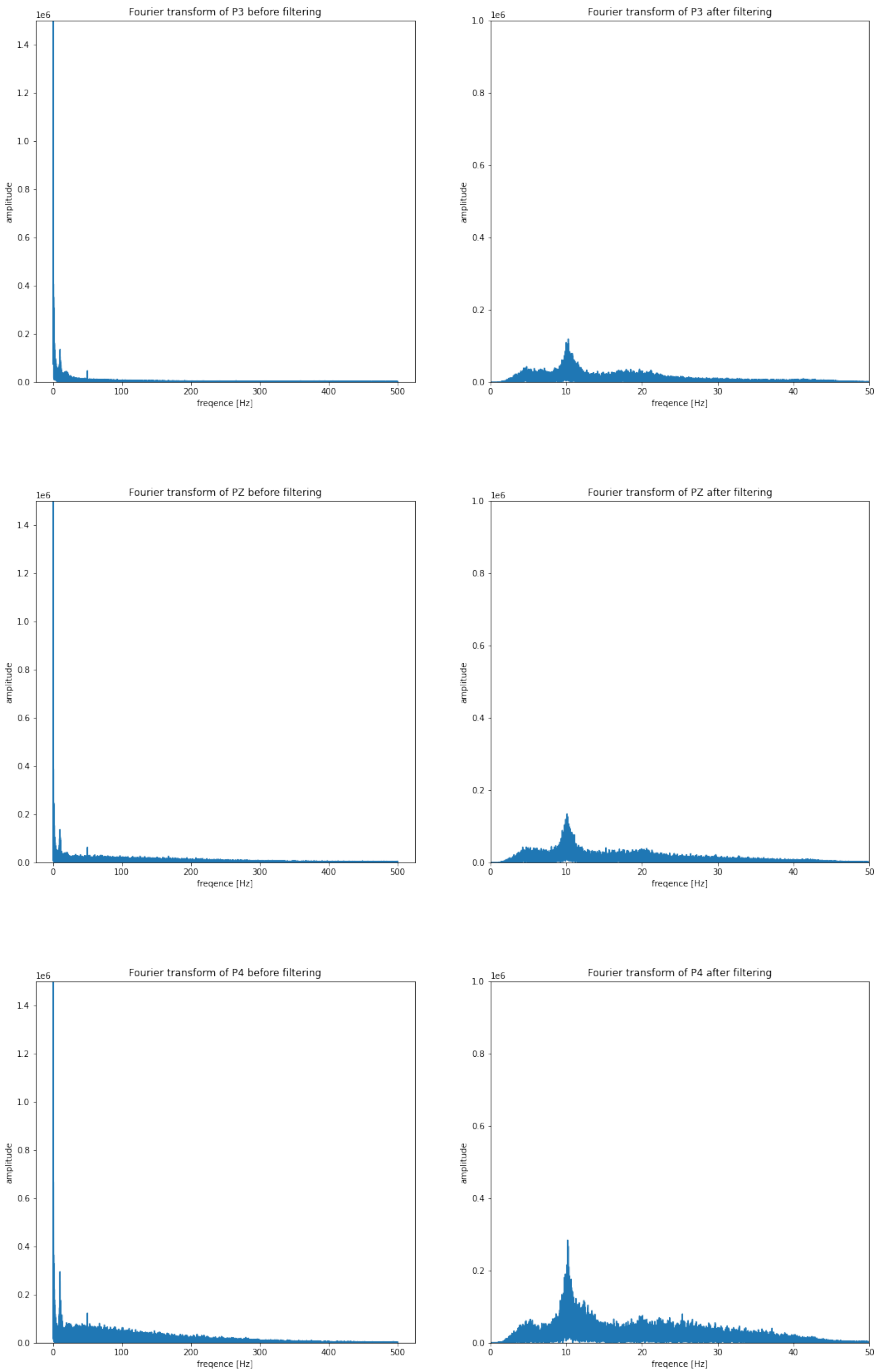


Figure 14: Comparison of Fourier transform before and after DWT (part 8)

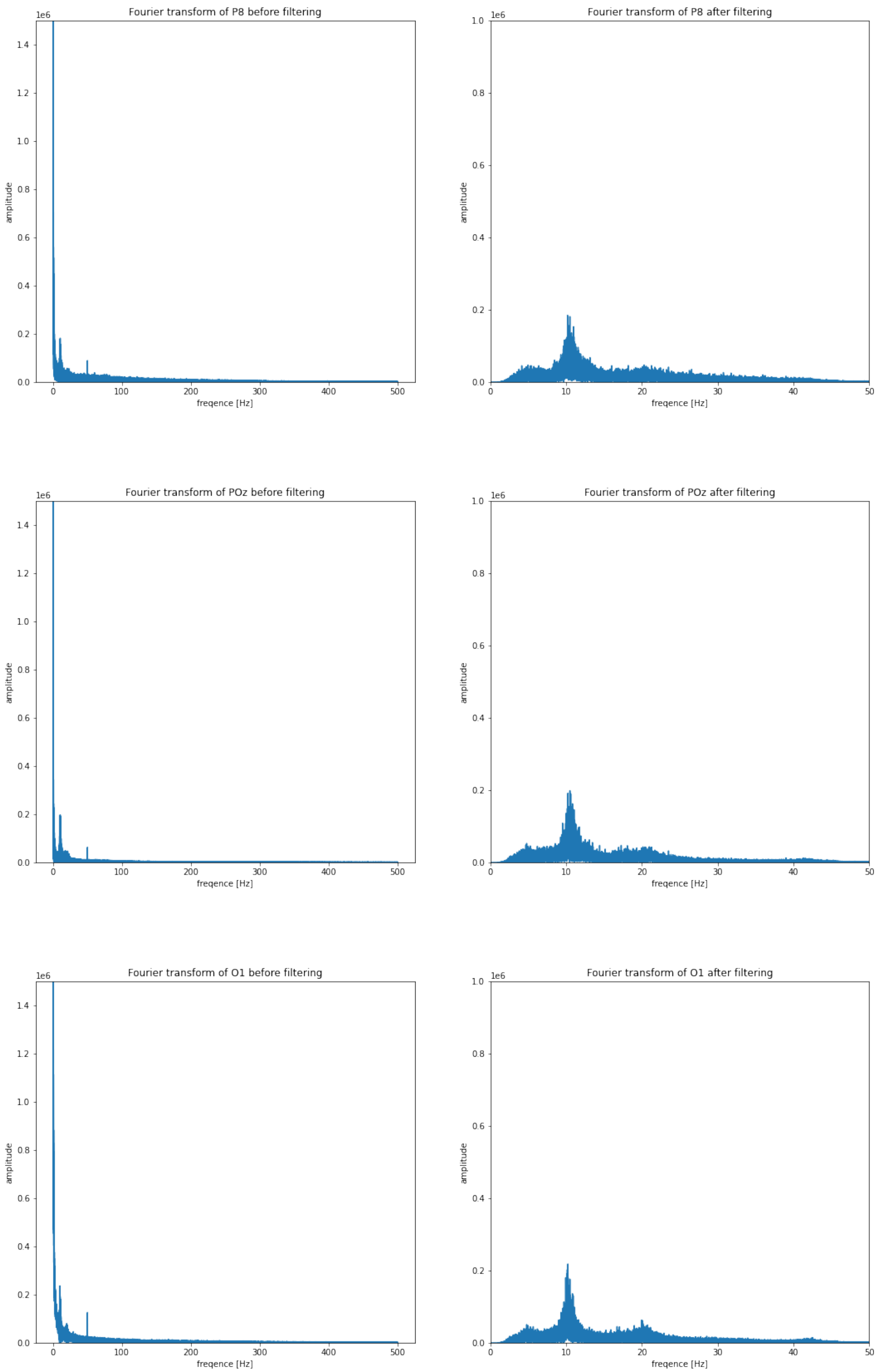


Figure 15: Comparison of Fourier transform before and after DWT (part 9)

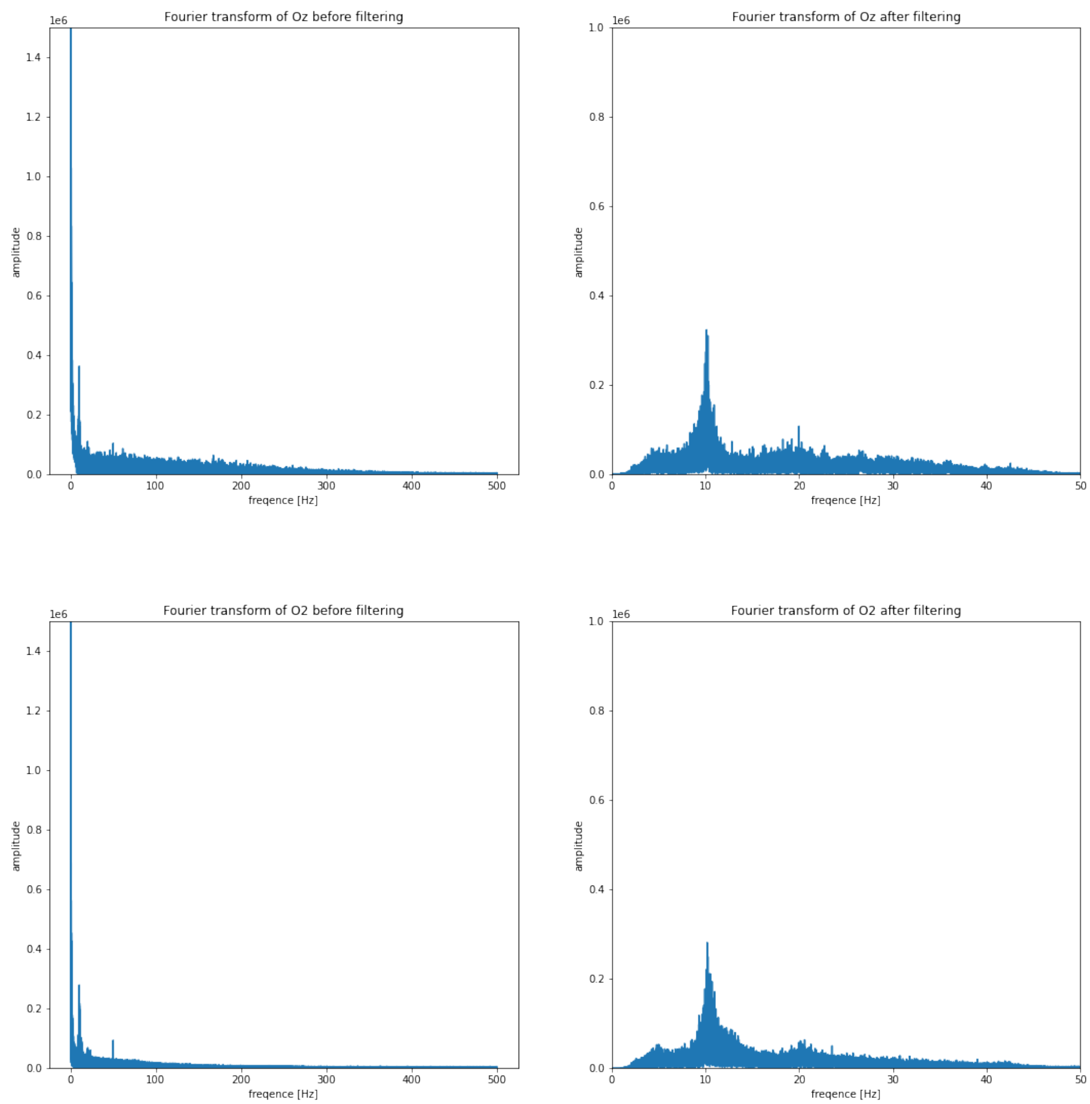


Figure 16: Comparison of Fourier transform before and after DWT (part 10)