Guide to EEG_TF App

by Alina Beliakova, CRNL, France

September 2021

Contents:

Introduction

The EEG_TF is a MATLAB toolbox designed to visualize time-frequency maps (spectrograms and scalograms) of the EEG signals. The app uses MATLAB and EEGLAB to operate through the GUI.

EEG_TF toolbox can upload the data in the specific EEGLAB format (.set), any other data (2- or 3D) that can be passed to the GUI through the command line. Nevertheless, this toolbox is specifically designed to operate with EEG data where several subjects and several conditions are present. It allows comparison or merging of conditions, individual or group analysis, different settings of spectral decomposition and plotting.

This guide gives a brief description of the functionality of the toolbox, the details about functions and their interaction can be found in the scripts themselves.

Requirements

- MATLAB R2018a and higher

- MATLAB Signal processing toolbox (for Short-Time Fourier Transform, wavelet transform don't require any toolbox)

- EEGLAB for .set files processing

- Before starting ensure that the EEG_TF is in the MATLAB path

Basic functionality

Launching the app

- Type *eeg_tf()* in the MATLAB command line to launch the GUI without any input data. Alternatively, it is also possible to pass input arguments from the current MATLAB workspace by typing *eeg_tf(data, fs, time)*, where the *data* is 3D signal with dimensions [channels, time, trials], *fs* is the sampling frequency in Hz (e.g. 128 Hz), *time* is start and end times in seconds (e.g. [-0.5, 1] sec), or full time vector in seconds. If you pass the dataset as an argument, skip the next step **LOAD .SET DATA**.

The GUI contains three main panels (see Figure 1):

- Main app controls (right panel) to load, visualize and save data, quit the app and info buttons
- EEG dataset info panel
- Analysis' parameters panel

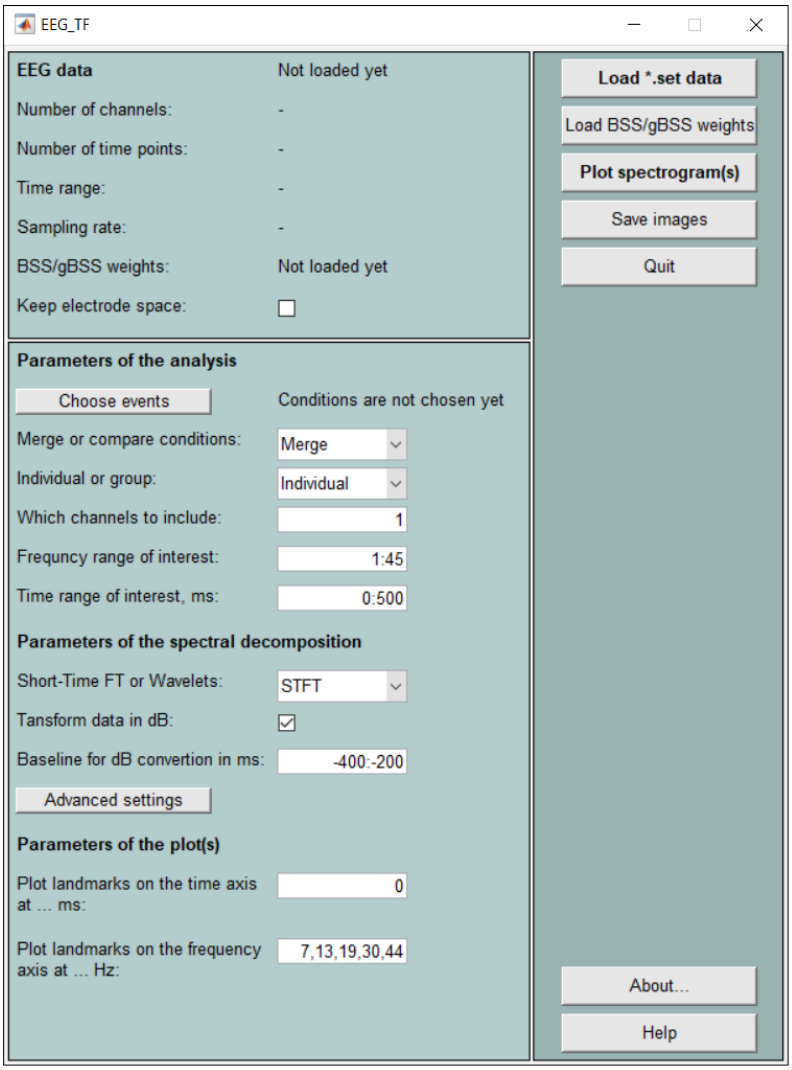


Figure 1 - The GUI with no data loaded

Load the dataset(s) and check ICA weights

- Click **LOAD .SET DATA** to upload the existing EEGLAB dataset(s), you can choose several datasets but ensure they have the same time range, conditions and sampling frequency. If these parameters are not consistent between datasets you can launch analysis for each dataset separately by loading them separately. In the EEG data region you will see the information about the (first) dataset uploaded: number of channels, number of time points, time range and

sampling rate. If the (ICA) weights for source space (inverse) projection are included in the EEGLAB dataset you will see the corresponding message in front of the BSS/gBSS weights line:

EEG data	2 dataset(s)
Number of channels:	134
Number of time points:	129
Time range:	[-500:500]ms
Sampling rate:	128
BSS/gBSS weights:	Included
Keep electrode space:	☐

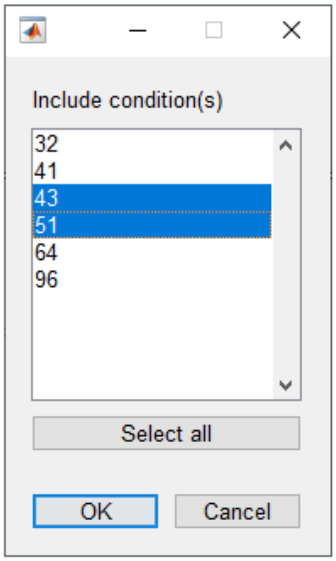
- Click **LOAD BSS/GBSS WEIGHTS** to load the pre-saved weights, calculated not in the framework of EEGLAB. This function is useful if the (g)BSS was done using other tools than EEGLAB[1]. If there are no weights available in the dataset and no weights uploaded, the analysis will be done in the electrode space data.

- Check '**Keep electrode space**' to avoid automatic projection data to the source space. It is needed if there are weights in the dataset but you don't want to use them.

Set parameters of the analysis

- Click **CHOOSE EVENTS** to choose the condition(s) of interest (if unset, all conditions will be included):

[1] Theoretically it is possible to add the (g)BSS data (weights and sphere matrices) and recalculate the EEG.icaact in the EEGLAB data structure manually but after saving the dataset (pop_saveset()) this information is not saved. If it is not the case in your situation or if this issue is solved in future releases of EEGLAB you can ignore the function of loading presaved weights to EEG_TF



- Choose **COMPARE** (2 conditions only) or **MERGE** (any number of conditions) the data corresponding to the conditions

- Choose **INDIVIDUAL** or **GROUP** to perform analysis on each dataset or on the datasets grouped together (datasets have to be compatible, e.g. to have the same sampling rate, time range and conditions)

- Set which channel(s) to take for analysis, each channel is treated independently, a separate spectrogram will be generated for each channel

- Set frequency range of interest

- Set time range of interest

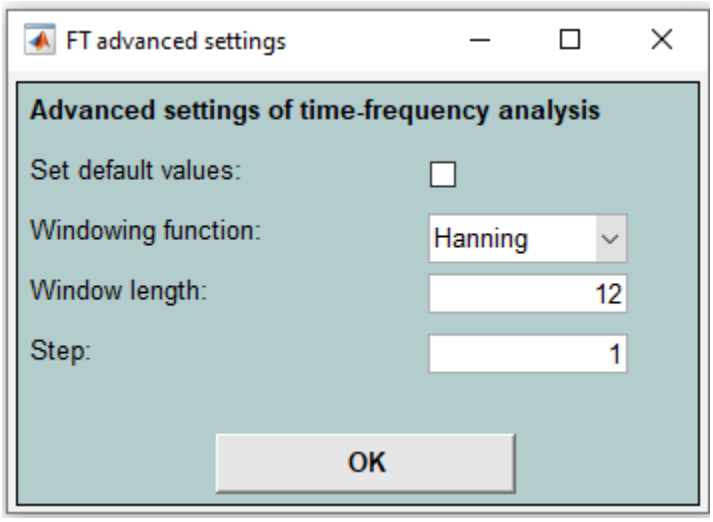
Set the parameters of the spectral decomposition

Note: The default parameters in this section are calculated automatically based on the given dataset, these parameters can be kept for the fast start. Nevertheless, it is recommended to tune them depending on the task (see 'Practical tips' section for additional information).

- Choose the method of the spectral decomposition: Short-Time Fourier Transform[2] or Wavelets[3]
- Check if you want to transform data to dB (this function is useful to compensate for the 1/f law, e.g. unify the resolution across frequencies)
- Set the baseline for dB transform

Set advanced parameters of spectral decomposition (optional)

Short-Time Fourier Transform

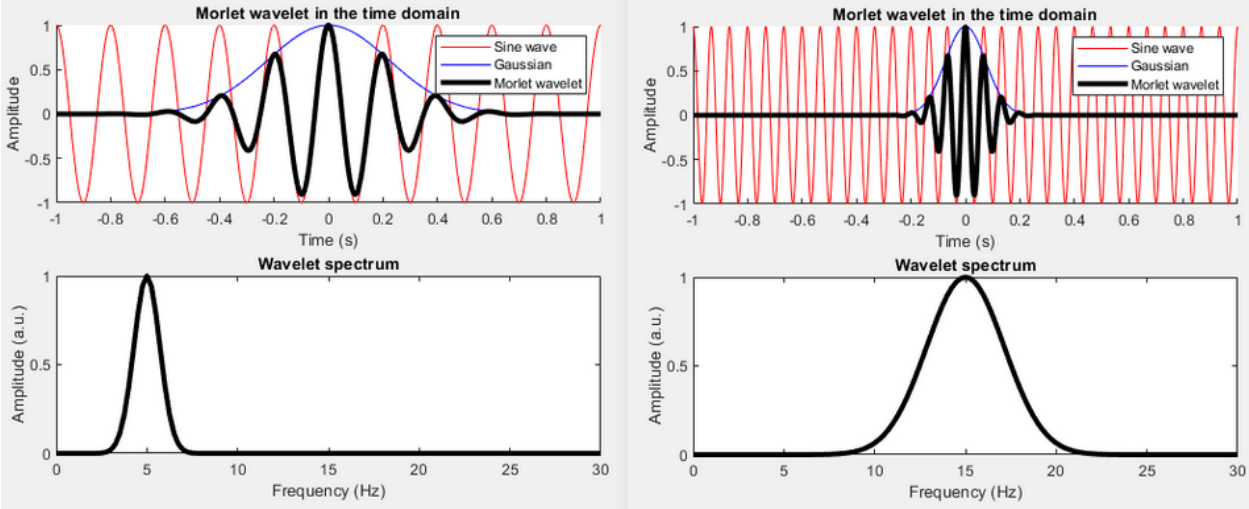


- Uncheck the 'Set default values' option to be able to change the settings
- Choose the window function: Hanning (default), Blackman or none. It is recommended to use window function to avoid artifacts on the tf-maps
- Set the window length, note that the larger the window the better is the frequency resolution but worse is the time resolution, plus the signal is more cut on the sides
- Set the step with witch the window will be moving, smaller step increases the time resolution but may result in the longer computation time

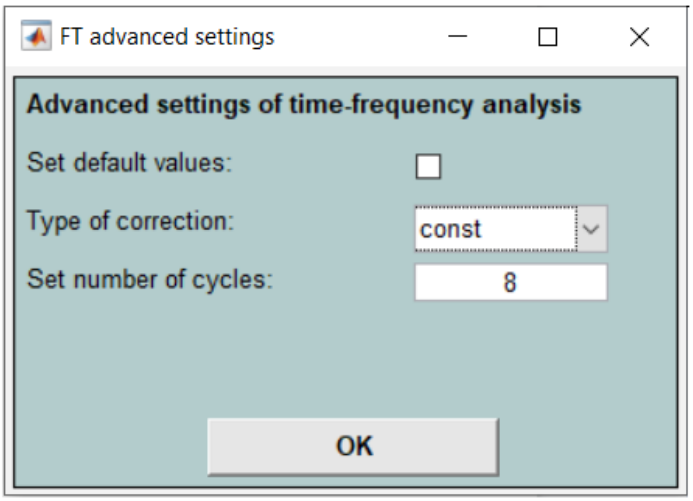
[2] uses the MATLAB function *spectrogram()*, check https://www.mathworks.com/help/signal/ref/spectrogram.html for more info
[3] based on the code from Mike X Cohen (2014) Fundamentals of Time-Frequency Analyses in Matlab/Octave. 1st ed, sinc(x) Press, check https://www.sincxpress.com for more information

Wavelet Transform

Advanced settings of wavelet transform are used to correct the inconsistency which arises from the properties of wavelets. Wavelets are normalized to always have unit amplitude in the frequency domain, but their width varies in both domains - if number of cycles is constant, for higher frequencies the gaussian envelope gets narrower (to still fit the same number of cycles) and the wavelet gets narrower. In the frequency domain such wavelet will look like a gaussian which on contrary gets wider, so it covers more power than a wavelet corresponding to a lower frequency:



In the picture above, wavelets have the fixed number of cycles = 7 for two frequencies - 5 and 15 Hz. Obviously, their width differs a lot in the frequency domain. This is the reason why scalograms have better frequency resolution at the lower frequencies and worse at the higher. While the process of normalizing wavelets is complicated, this inconsistency can be partially compensated by varying the number of cycles over the range of frequencies - from smaller number of cycles for smaller frequencies to larger for the larger ones. It allows to get the time-frequency resolution more homogeneous over the scalogram. The best solution is to try different options and to choose that one which allows you to see the region of interest in the best resolution.



- Uncheck the 'Set default values' option to be able to change the settings
- Change the type of correction of non-linearity: **const** (default) means that wavelets with the same number of cycles will be applied at each frequency, **linear** means that the the number of cycles will change linearly in the range indicated below, **log** leads to logarithmic change of the number of cycles in the range
- Set the number of cycles as a single integer number or as a range

Set parameters of plotting

These settings allow plotting the guiding vertical and horizontal lines on the time-frequency maps.

- Set at which time points you prefer to have landmarks (for example at the time of stimulus presentation, e.g. 0ms), can be left blank

- Set at which frequencies you prefer to have landmarks (for example to separate the alpha and beta activity, etc.), can be left blank

Plot and save time-frequency maps

- Click **PLOT SPECTROGRAM(S)** to get the graphical results of the analysis

- Click **SAVE IMAGES** to automatically save images in .fig and .jpg formats in the directory which you choose at demand (names are automatically generated in the form 'EEG_TF_Image1.jpg'). After saving the figures are closed. You can save images manually (though the figure interface File->Save as) and ignore this option.

Sample analysis

There is a simulated EEG event-related toy data *epoched_data.set* in the EEGLAB format included in the app distribution. This dataset is produced using the SEREEGA toolbox and the exact code in the *generate_epoched_data.m* file can be used to acquire a similar dataset (difference maybe due to the randomly generated noise). For this aim the activation signals from 2 brain components are projected to the scalp space. As the forward model the pre-generated lead-field (New York Head ICBM-NY) is used, 64 electrode montage. Data is sampled at 500Hz, the epoch lasts for 0.5sec. The first activation signal comes from the eyes, covers the frequency range between 60 and 80Hz and it appears in the first half of the epoch. The second activation signal comes from the cuneus area in the range of 20-30Hz and appears in the second half of the epoch. Figure 2 shows the described characteristics of the signals on the example of one of the electrodes, where both signal are mixed in different proportions:

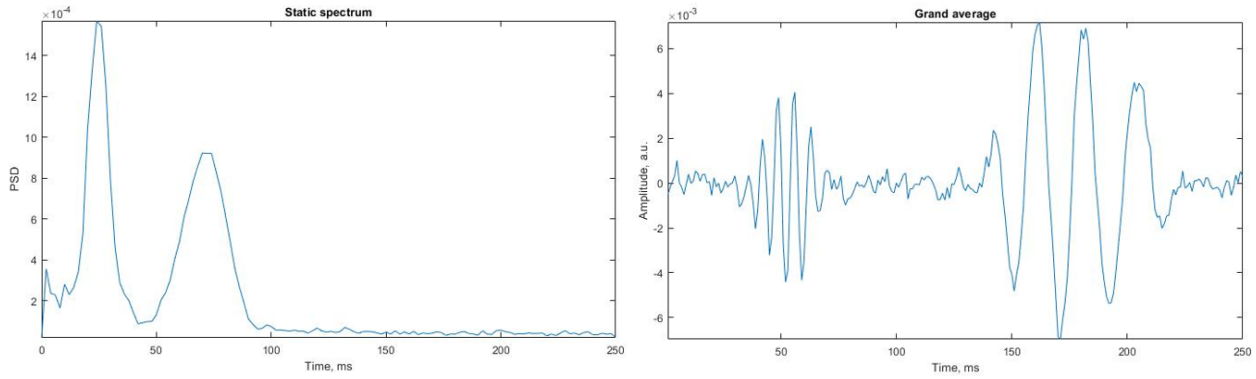
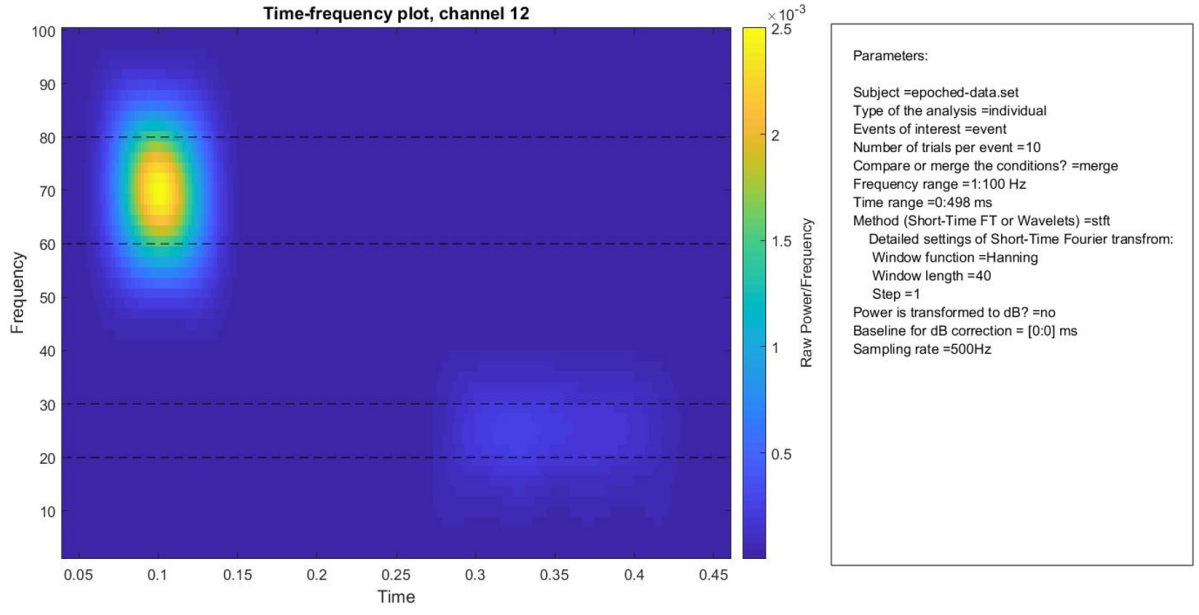


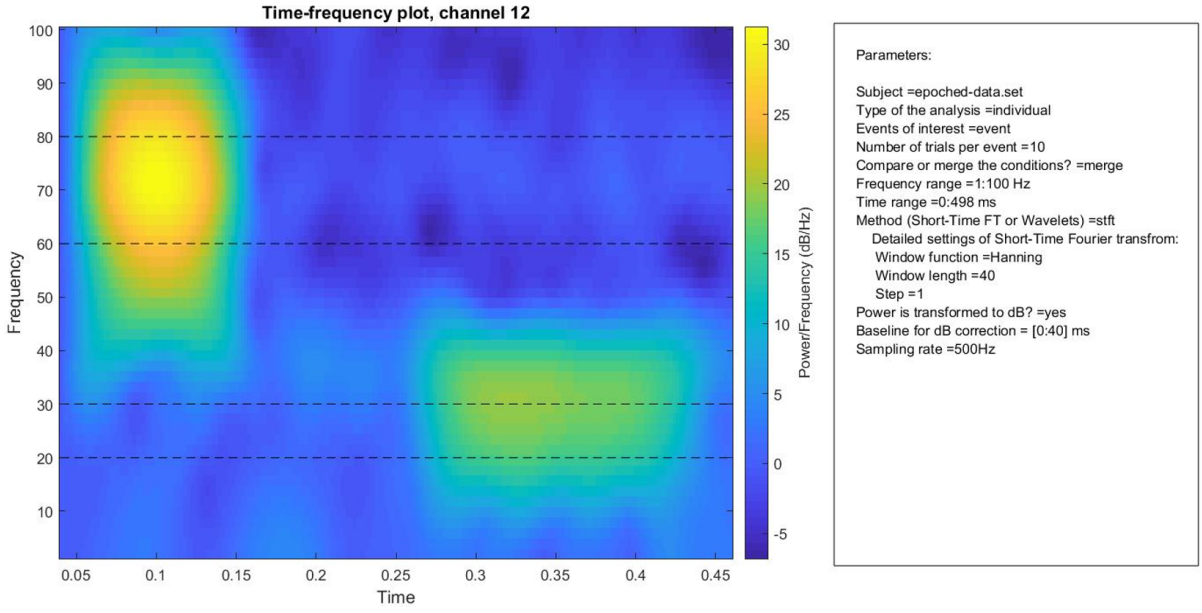
Figure 2 - left: static spectra of the 12th channel obtained with Welch's method; right: grand average of the 12th channel over trials

Several examples of the analysis are done to demonstrate the effect of dB transformation with baseline correction and the difference between STFT and wavelet transform results. The resulting time-frequency maps for the channel 12 in the electrode space of the sample dataset obtained with the EEG_TF toolbox

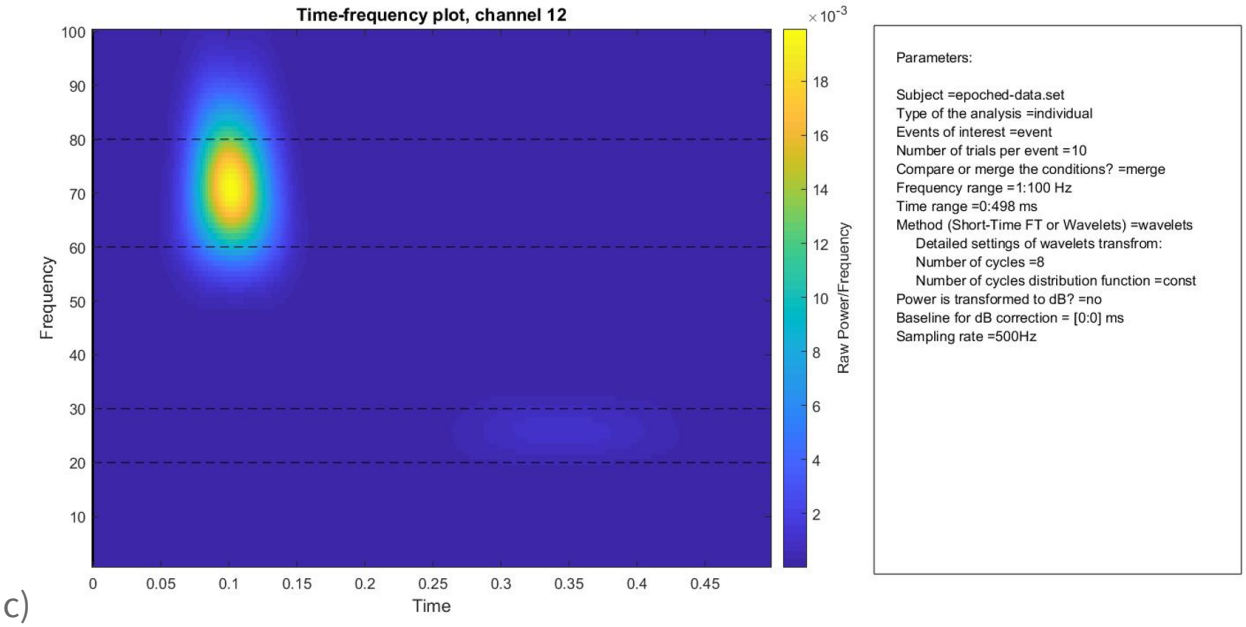
are shown on the Figure 3. Note that the main information about the data and analysis parameters are shown in the textbox on the right from the plot in order to simplify the process.



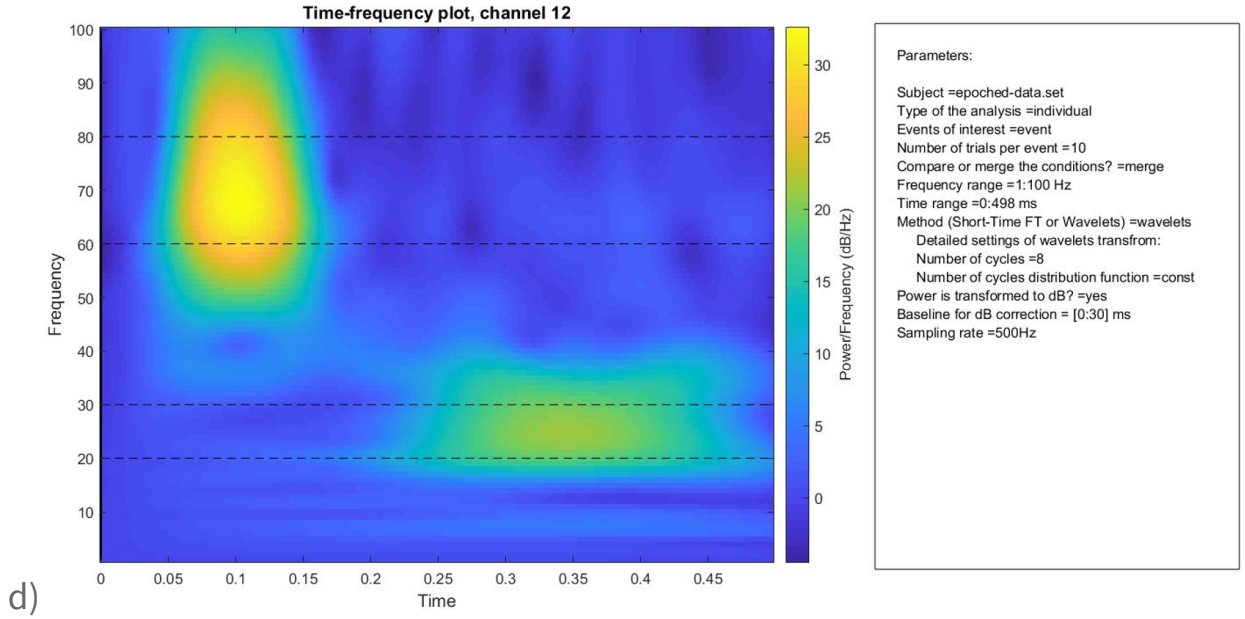
a)



b)



c)

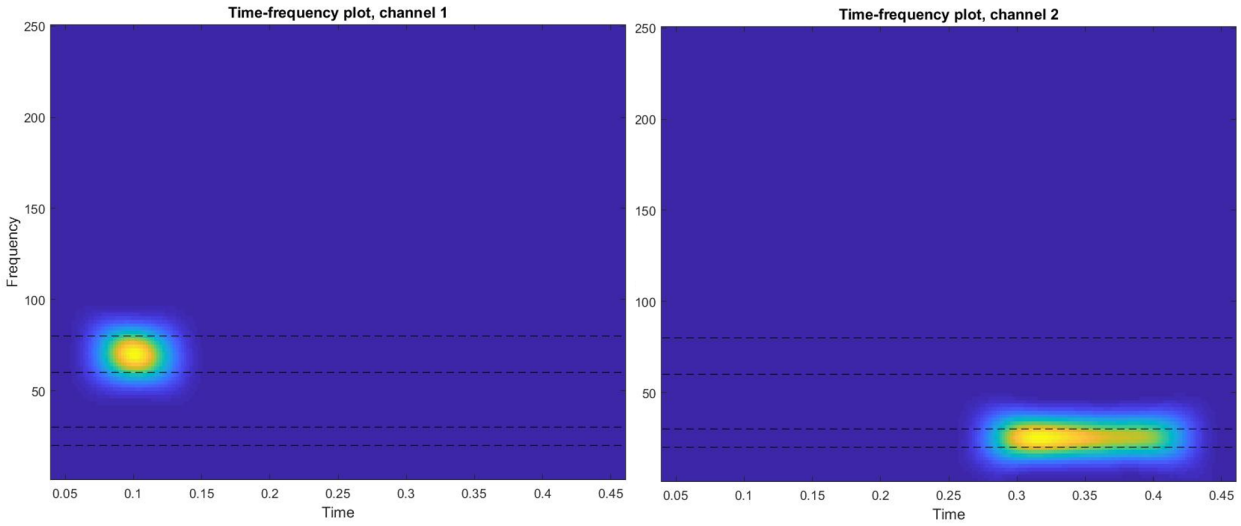


d)

Figure 3 - The time-frequency plot of the signal at the 12th channel produced with different parameters: a) STFT with 40 ms window length, no baseline correction, no dB conversion; b) STFT with baseline correction (0:40ms) and dB conversion; c) wavelet transform no baseline correction, no dB conversion; d) wavelet transform with baseline correction (0:40ms) and dB conversion

From the Figure 3 it is seen that the dB conversion with baseline correction allows to reach better contrast. Nevertheless the time period considered as the baseline has to be chosen carefully as it has to be the period where no important activity (by the experimental design) is present, for example some time before stimulus presentation. If there is no possibility to define a baseline period (for example, resting state data), it can be set to 0 and dB conversion can be done anyway. Also, it can be seen that the STFT and wavelet transform can be used interchangeably at the particular parameters, as in the present example there is almost no difference in the results.

The previous examples were done for the single channel in the electrode space. If we switch to the source space (remove the check after 'Keep electrode space' in the EEG dataset info panel) by using the demixing matrix, we get the two perfectly (because it is the inverse of the forward model used to create these signals) separated signals at each of the two initial sources:



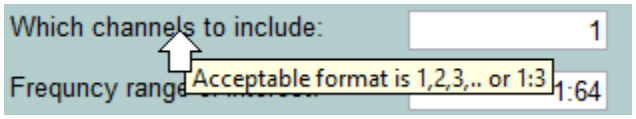
Practical tips

Using the app

➔ At the bottom of the app window there is the status/error line which provides you with tips if a problem occurred or some action has been launched.

➔ It is recommended to quit the app by hitting the **QUIT** button, not by closing the window. Quitting this way allows you to wipe up the variables from the memory and close all the app windows at once. Note: This action closes all the previously open MATLAB windows, not only EEG_TF related.

➔ For the parameters which consist of numerical values ('Which channels to include', 'Frequency range of interest', etc.) typed by the user, you can get a tip (what format of the input is expected) by pointing the mouse on the parameter's line:



Spectral analysis

➔ As an alternative to the present toolbox, check the EEGLAB function newtimef() for time-frequency analysis, there are such useful options as multitapers, different types of wavelets, detrending and others. The tutorial videos can be found on Youtube.

➔ It is important to keep in mind that there is always a trade-off between time and frequency resolution. Therefore, it may be useful to try different settings (the window length in case of STFT or number of cycles in case of wavelets) to detect fine details in the particular area of time-frequency map, and analyse the ensemble of images obtained with different settings rather than a single image.

➔ Due to the non-linear spectral resolution of the time-frequency maps obtained with wavelets they require correction of this nonlinearity by using a variable number of cycles. There are three options addressing this issue in the EEG_TF toolbox: constant (no correction of nonlinearity, usually numbers between 2 and 9 provide acceptable result), linear (same range of numbers applies) and logarithmic (same numbers apply). See the section ‘Wavelet transform’ for detailed explanation.

➔ Note that Short-Time Fourier Transform always cuts the time axis on the value of the window size