FieldTrip Beamformer Manual

Sara Jamil

Table of Contents

Beamforming Background 3

FieldTrip Manual 6

FieldTrip Basics 6

Script Walkthrough 8

Preprocessing 8

Covariance Calculation 10

Preparing the MRI 11

Preparing the Head model 11

Aligning the Electrodes 13

Preparing the Lead field Matrix 14

Source Analysis using Beamforming 14

Splitting into Time Windows 16

Plotting 17

Future Improvements 20

Beamforming Background

This section provides is intended to provide a brief review of beamforming. Beamforming is a spatial filter operation that isolates neural sources of activity at a specific virtual brain source location. While there are different techniques for applying beamforming to EEG data, such as frequency vs. time domain analysis and vector vs. scalar beamformers, this manual focuses mainly on the application of time domain methods in vector beamforming.

The main idea behind beamforming is that the neural signal at a location of interest is constructed as a weighted sum of EEG channels forming a virtual source. The weights are chosen such that only the signal from that specified location contributes to the beamformer output, thus acting as a spatial filter.

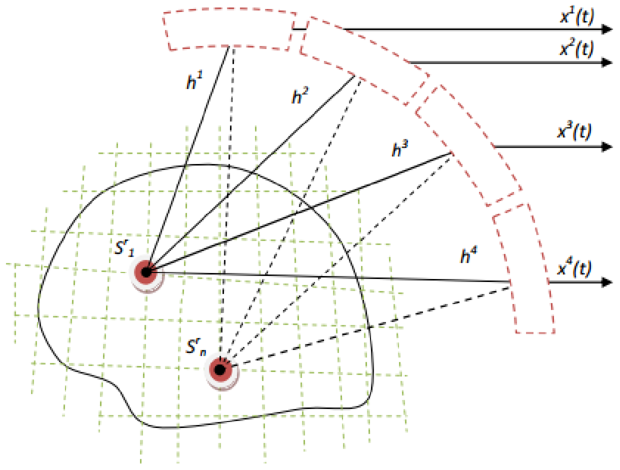


Figure 1: Electrodes and virtual sources

Beamforming is a technique used to solve the ill-posed inverse problem, which also requires the calculation of a forward model. The forward model predicts the electric potentials measured at the sensors if some source was active inside the brain while the inverse model estimates the activity values of the source that is generated given the measured electric potential at the electrodes.

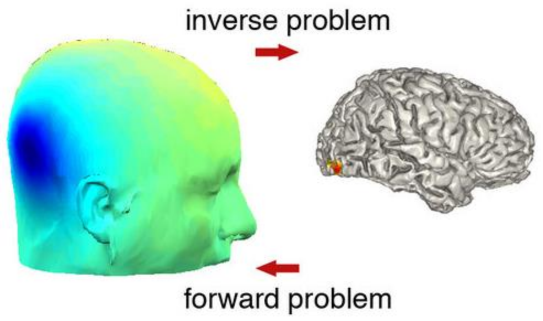
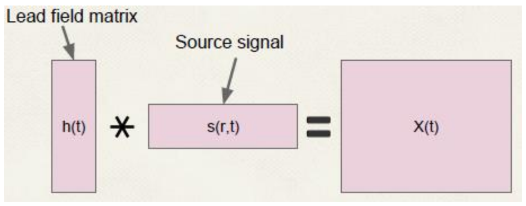
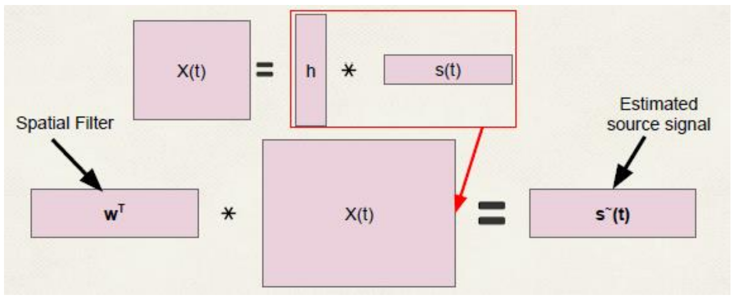


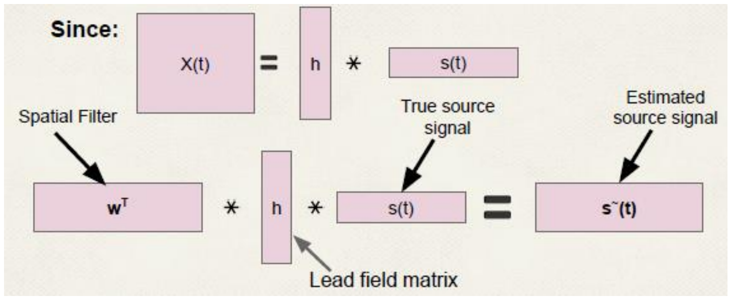
Figure 2: Forward vs. Inverse problem

The following equation describes the forward problem calculation where X(t) is the electric potential measured at the electrodes, h(t) the lead field matrix is the forward model, and s(r,t) is the source signal.



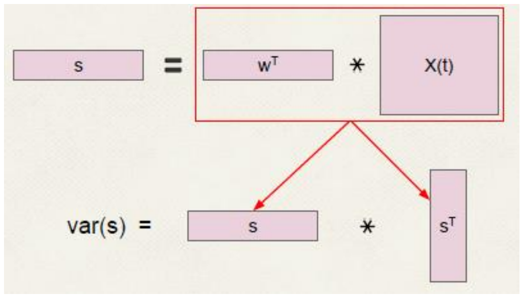
The inverse problem can then be solved by multiplying the calculated spatial filter weightings, w, to the recorded data.

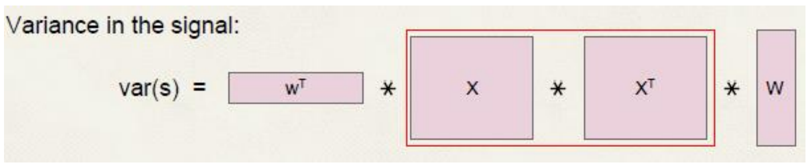




To simplify the equation you can divide both sides by the true source signal. This shows that, ideally, if the actual and estimated source signals are the same, the product of the spatial filter and lead field for a particular location must equal 1. However, the spatial filter of one source multiplied by the lead field of another should equal to zero. This corresponds to the two constraints of spatial filters: unity gain and minimum variance.

The power of the source signal can be calculated by the variance.





The box highlighted in the equation above corresponds to the covariance, Cov, calculated from the recorded data. Minimizing this variance gives us the following equation for the weights:

$$w^T = [h^T Cov^{-1} h]^{-1} h^T Cov^{-1}$$

Some assumptions are made in using beamforming. The first is that no two distant cortical areas generate coherent local field potentials over long periods of time. This causes limitations in beamforming when two or more sources are highly correlated. Another assumption is that the activity at each channel is the linear superposition of all the source activity. Contribution of each source depends on its detectability at the channel. Thus, the linear combination of data collected at the channel level is able to reconstruct the time course activity at a specified location.

Link to beamforming lecture video: https://www.youtube.com/watch?v=7eS11DtbIPw

Link to source localization article: http://www.scholarpedia.org/article/Source_localization

Link to Brainstorm beamforming tutorial:
http://neuroimage.usc.edu/brainstorm/Tutorials/Beamformers

Links to related beamforming papers: http://www.ncbi.nlm.nih.gov/pubmed/9282479
http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4041989/

FieldTrip Manual

FieldTrip Basics

Before you begin, it is important to download the latest version of FieldTrip from their website. Once you have provided your email address, it will take you to the index page where you can download the toolbox as well as many other files from their tutorials, modules, examples, and workshops. Some of these available examples can provide useful data that you can easily download. You will notice that there is a new version of the toolbox uploaded every day. This is because the toolbox does not update from MATLAB, however you must check to see if there are any relevant updates and download the latest version if you think it would be useful. It is recommended to download the latest version of the toolbox and to download the full version rather than the lite version.

Link to download FieldTrip: http://www.fieldtriptoolbox.org/download.php

After downloading and unzipping the toolbox, ensure you have correctly added the folder to the main path using the "Set Path" key or addpath command in the beginning of your script. Next, you will need to execute the **ft_defaults** function which sets the defaults. You may need to run this function every time you want to use FieldTrip, otherwise you may get an error when trying to execute certain functions like reading MRI data.

It is useful to understand the basic method for calling FieldTrip functions. Every Fieldtrip function begins with "ft_" with the exception of some of the lower level functions (which you will not need to call often). The higher level functions require you to input a configuration variable "cfg" as well as the stated input data. The functions may have certain configuration parameters that require your input but most of these parameters have defaults if you do not enter anything. It is important to check what the default parameters by checking the help documentation for the function or the source code, and it is recommended to input all the parameters in case the defaults change between the different versions. Also, not all of the functions require an output variable (ex. plotting functions). Here is an example of how to call a function:

```
cfg = [];
cfg.covariance = 'yes';
cfg.channel = 'eeg';
cfg.vartrllength = 0;
cfg.covariancewindow = 'all';
cfg.keeptrials = 'yes';
tlock = ft_timelockanalysis(cfg, data);
```

Since analysis using FieldTrip usually requires multiple steps in order to get the output, the series of steps is referred to as the analysis protocol or pipeline. If there is saved Fieldtrip data, you can use the function **ft_analysispipeline** to visualize the pipeline. Here is an example of a pipeline you may use to perform source analysis on FieldTrip (taken from my script – details below):

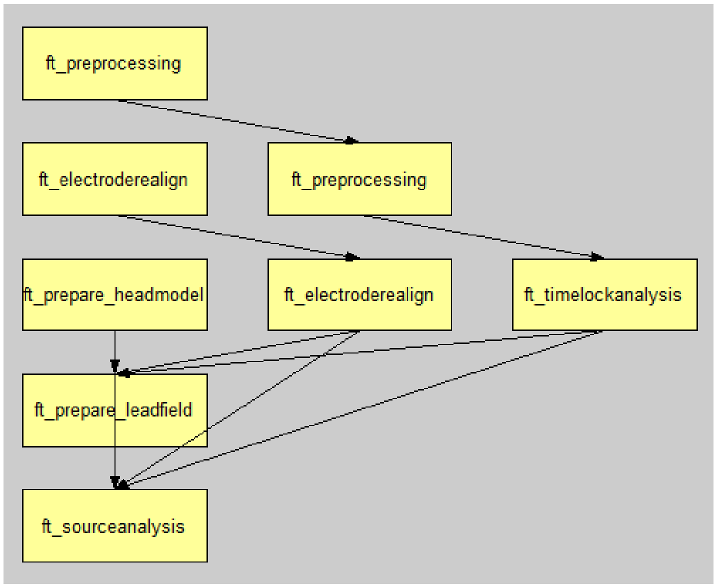


Figure 3: source_lcmv pipeline

It is important to save your output continuously in order to avoid wasting time re-running the entire script whenever you lose the data stored in your workspace. If you are trying to save large amounts of data at once, it may take an excessively long time to save the data or you may encounter an error like this:

```
Warning: Variable 'interpwin16' cannot be saved to a MAT-file whose version
is older than 7.3. To save this variable, use the -v7.3 switch.
Skipping...
```

While there are functions online that try to reduce the save time (ex. fastsave), it may not always work and increases the file size by forgoing compression. The easiest solution is to split up your data and save it separately, if possible (more details below).

The most useful thing to do when encountering a problem on Fieldtrip is to look up the help documentation, either online or in MATLAB, and carefully read through the details of the function.

```
help ft_sourceanalysis
```

It is easy to miss the details written in the documentation but it is important to get familiar with the functions you are using. Sometimes the help function does not give enough information, in which case you may want to open the script for the function and look through the code. It is more difficult to extract information this way due to multiple layers of function calls within each function, and this may require some time before you get used to the way the code is written.

Link to introduction tutorial: http://www.fieldtriptoolbox.org/tutorial/introduction

Link to walkthrough tutorial: http://www.fieldtriptoolbox.org/walkthrough

Script Walkthrough

This script walkthrough provides background information on every section of the *beamformer_dipoli.m* script which is included in the appendix of this report. It may be helpful to examine the code alongside the walkthrough. Useful data files are also provided and referred to below.

Preprocessing

The preprocessing phase of the analysis can take quite a bit of time in order to correctly preprocess the EEG data and is sometimes tedious. Since this is the first step you must take before analysis of the data, it may take longer to figure out than the subsequent steps. It is very important to know exactly what kind of data you are working with, so it is highly recommended to ask someone who works in the lab where you are getting the data from.

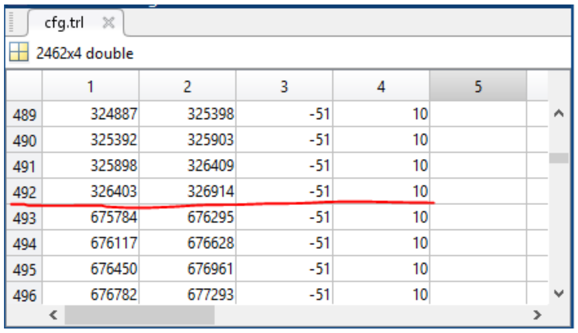
The first step is to import the data and use the **ft_definetrial** function to extract the relevant data trials from the EEG file. The EEG file will come as one long continuous dataset with markers in time indicating when different events occurred that will pertain to different trials. In order to view all the different event types in the data, you can change the following code in the script:

```
cfg.trialdef.eventtype = '?';
cfg = ft_definetrial(cfg);
```

This will display the event types in the command window (in text) what the different events are as a set of numbers (ex. 10, 20, 30, 40… etc.). You may encounter a problem with some of the markers being named incorrectly due to the binary number being concatenated with another marker in the data that will result in large numbers for the events (ex. 1034, 1044, 1054, 1064… etc.). In this case you may need to look at the binary representation of these numbers and only look at the 10 least significant bits in order to get the correct event number. This may differ between dataset, so it is best to talk to someone who knows about your particular data.

Once you know which markers you would like to analyse, you can choose it in **cfg.trialdef.eventvalue**. You must also provide details for how long all the trials should be. In the script, I have chosen the trial length to be 1 second long with 0.1 seconds taken before the event has occurred.

The EEG data file may contain a large amount of information from EEG tests that were conducted for an hour's length. There may be different tests pertaining to different event-related potentials (ERP) responses. For example, in the data used in the script, only the first 11 minutes contained the relevant trials for the P300 ERP's that were meant to be examined. Hence, after the call to **ft_definetrial**, you may need to open **cfg.trl** to examine which trials fall into the time window you wish to analyse.



cfg.trl

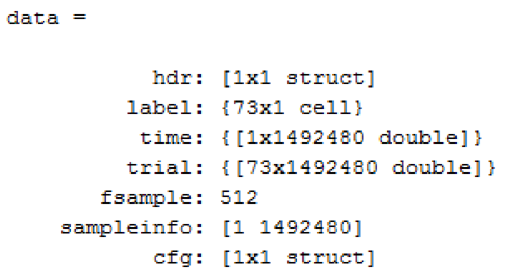
2462x4 double

	1	2	3	4	5
489	324887	325398	-51	10	
490	325392	325903	-51	10	
491	325898	326409	-51	10	
492	326403	326914	-51	10	
493	675784	676295	-51	10	
494	676117	676628	-51	10	
495	676450	676961	-51	10	
496	676782	677293	-51	10	

Figure 4: Selecting trials to remove manually

The first column and second column are the start and end sample for each trial you have defined. It is important to note that this is in terms of samples, not time, so you will need to do some quick calculations once you know what the sampling rate is. The sample rate is usually 512 Hz; however, the easiest way to check this is to make a call to **ft_preprocessing** and examining the output data.

```
fn = <insert filename here>
cfg = [];
cfg.dataset = fn;
cfg.trialfun = 'ft_trialfun_general';
data = ft_preprocessing(cfg);
```



```
data =

           hdr: [1x1 struct]
         label: {73x1 cell}
          time: {[1x1492480 double]}
         trial: {[73x1492480 double]}
       fsample: 512
    sampleinfo: [1 1492480]
           cfg: [1x1 struct]
```

Figure 5: Information on the data

Once you have selected the trials to remove, you can continue with the actual preprocessing of the data using **ft_preprocessing**. There are many options to choose from in terms of preprocessing the EEG data, but some filtering is usually required for the data. The chosen cut-off frequencies for the filters will depend on how you intend to analyse the data. In the script, I have chosen to apply a high-pass and low-pass filter, where the order of the filters can also be chosen, as well as subtracting the mean from the data.

The next step is to ensure that the channels are renamed into a format that is recognizable as EEG data by Fieldtrip. It is important to make sure you have the correct naming scheme for the data you are working with. The data used in the script contained 64 channel EEG data with the 10-20 naming scheme. This will be important for choosing the correct electrode placement later on.

It is important in the early stages to examine your data before analysis using browsing functions available on FieldTrip, such as **ft_databrowser**. This will help in determining the quality of the data and if more preprocessing is required. In some cases you may need to remove bad trials or bad channels altogether. You can visually inspect the data and remove trials or channels using **ft_rejectvisual**. Independent Component Analysis (ICA) is a commonly used tool for artifact rejection of blink, EOG, or ECG artifact, which can be done using **ft_componentanalysis.** However, using ICA may be unnecessary when preforming beamforming and removing channels from the data may lead to issues with applying ICA (and possible source reconstruction). Links to visual and automatic artifact rejection are available below.

Link to ft_definetrial help: http://www.fieldtriptoolbox.org/reference/ft_definetrial

Link to ft_preprocessing help: http://www.fieldtriptoolbox.org/reference/ft_preprocessing

Link to ft_databrowser help: http://www.fieldtriptoolbox.org/reference/ft_databrowser

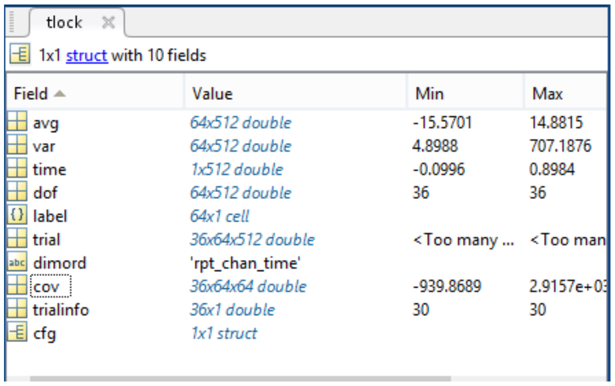
Link to visual artifact rejection tutorial:
http://www.fieldtriptoolbox.org/tutorial/visual_artifact_rejection

Link to automatic artifact rejection tutorial:
http://www.fieldtriptoolbox.org/tutorial/automatic_artifact_rejection

Covariance Calculation

In order to perform source analysis, you must use **ft_timelockanalysis** in order to obtain the covariance matrix. The covariance matrix must be calculated for the LCMV beamformer; however, you must use **ft_freqanalysis** for methods that involve frequency domain analysis rather than time domain.

The **ft_timelockanalysis** computes the timelocked average ERP and the covariance matrix. You can also choose the time window for which you wish to take the covariance in all of the trials using **cfg.covariancewindow**. There are also different methods for handling the covariance matrix where the trials are of different lengths, but by default it will not accept variable length trials. It is recommended to keep all the trials at the same length when defining them.



tlock

1x1 struct with 10 fields

Field	Value	Min	Max
avg	64x512 double	-15.5701	14.8815
var	64x512 double	4.8988	707.1876
time	1x512 double	-0.0996	0.8984
dof	64x512 double	36	36
label	64x1 cell		
trial	36x64x512 double	<Too many ...	<Too man
dimord	'rpt_chan_time'		
cov	36x64x64 double	-939.8689	2.9157e+0
trialinfo	36x1 double	30	30
cfg	1x1 struct		

Figure 6: tlock data structure

The function will return the covariance matrix as well as the average of all the trials. In some applications you may wish to keep all of the trials. For example, if you wish to project each trial through a common spatial filter, you will need to keep the trial information in the timelock data for processing using **ft_sourceanalysis** later on. In this case, set **cfg.keeptrials** to 'yes'.

Link to ft_timelockanalysis help: http://www.fieldtriptoolbox.org/reference/ft_timelockanalysis

Link to ft_freqanalysis help: http://www.fieldtriptoolbox.org/reference/ft_freqanalysis

Preparing the MRI

Preparing the MRI data is important if you wish to get visual feedback on where the sources are located in the brain by overlaying the source information onto a standard MRI. There is a set of template MRI scans that can be used for this purpose that are downloaded with the FieldTrip toolbox or you can download the Subject01 data which was used in the script from the FieldTrip index (link below).

In preparing the MRI scan for the head model, you will need to use **ft_volumesegment** to segment the different regions in the head (i.e. brain, skull, and scalp). This will return probabilistic tissue maps of the different regions that will be required in the calculation of the forward model. If you are using the same template MRI for all your data, you will only need to calculate this once.

Some of the methods for preparing the head model require the calculation of a triangulated surface mesh for the volume conduction model using **ft_prepare_mesh**. This requires you to choose the number of vertices of the surface mesh you wish to have for each compartment.

Link to download Subject01 data: ftp://ftp.fieldtriptoolbox.org/pub/fieldtrip/tutorial/Subject01/

Link to ft_read_mri help: http://www.fieldtriptoolbox.org/reference/ft_read_mri

Link to ft_volumesegment help: http://www.fieldtriptoolbox.org/reference/volumesegment

Link to ft_prepare_mesh help: http://www.fieldtriptoolbox.org/reference/ft_prepare_mesh

Preparing the Head model

The preparation of the head model is required for the calculation of the forward model and can be done using **ft_prepare_headmodel**. This function constructs a volume conduction model that specifies how the currents in the brain are propagated through the tissue to create externally measurable electric potentials. This requires the input of the estimated conductivities of the different tissues in the model (i.e. scalp, skull, brain conductivities). Standard values of conductivities were used in the code.

There are many methods available for the head model calculation for EEG data on FieldTrip, the ones that were tested in this project were the ‘bemcp’ and ‘dipoli’ methods. The different methods used for calculating the head models either use boundary element methods (BEM) or finite element methods (FEM), however, both the bemcp and dipoli methods use BEM. BEM uses the triangulated surface mesh calculation for the head model while FEM uses a volumetric grid.

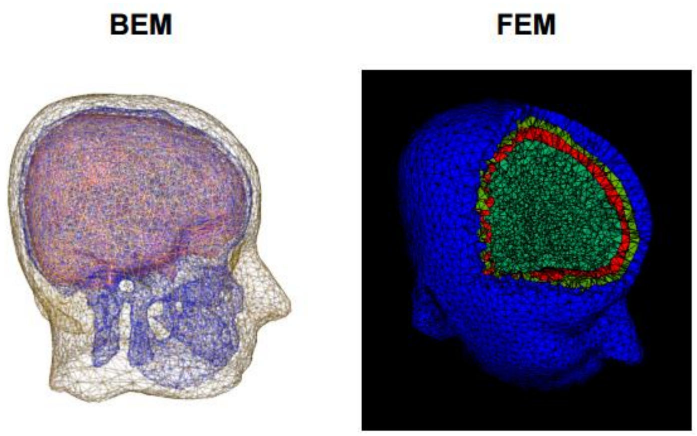


Figure 7: BEM vs. FEM forward models

After testing these forward models using a simulated source, it was found that the bemcp model did not perform adequately. The dipoli model seems to provide a more accurate representation of the source. However, this method cannot run on a Windows PC (only Linux or Mac) which may cause some issues. *Note: the dipoli head model is provided in the data files.*

There are papers that compare the performance between the different types of forward models. Some claim that OpenMEEG provides the best forward model using BEM techniques. While methods that use FEM techniques, such as SimBio, are considered to outperform BEM techniques, they also require more computational power. The software for both these methods are available online and must be downloaded separately from FieldTrip.

Link to ft_prepare_headmodel help: http://www.fieldtriptoolbox.org/reference/ft_prepare_headmodel

Link to ft_headmodel_bemcp help: http://www.fieldtriptoolbox.org/reference/ft_headmodel_bemcp

Link to ft_headmodel_dipoli help: http://www.fieldtriptoolbox.org/reference/ft_headmodel_dipoli

Link to headmodel EEG BEM tutorial: http://www.fieldtriptoolbox.org/tutorial/headmodel_eeg_bem

Link to download OpenMEEG: http://openmeeg.github.io/

Link to relevant papers on EEG forward modelling:
http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3683142/
http://www.montefiore.ulg.ac.be/~geuzaine/preprints/eeg_preprint.pdf

Aligning the Electrodes

Before proceeding, it is important to ensure that the electrode placements are correctly aligned with the head model. The function **ft_read_sens** is used to input standard electrode sensor data; you must ensure that the correct sensor naming scheme for your dataset is used. The script attempts to automatically align the electrode placements using **ft_warp_apply** first before using **ft_electroderealign** to visually align the electrodes. Automatic electrode alignment may not provide correct placements, which is why it is important to check the alignment using **ft_plot_mesh**. Visual realignment will require your input on rotation, translation, and scaling in the x, y, and z directions. This may take some time to correctly align the electrode using visual realignment. The final electrode placement should look like the following figure.

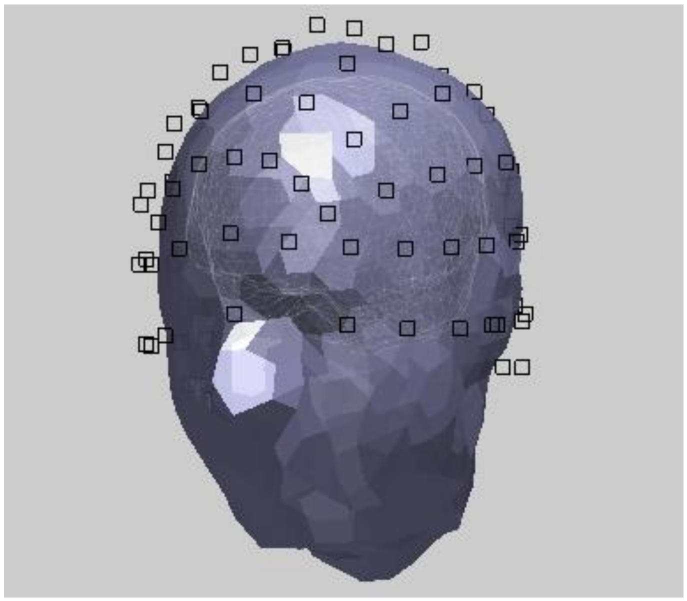


Figure 8: Electrode placement

A useful tutorial for aligning electrodes is found in the tutorial on creating a BEM volume conduction model for the head for source-reconstruction of EEG data (link below). *Note: If you are using the dipoli head model, the electrode alignment data is provided in the data files. You will need to realign the electrodes if you are using a different electrode placement scheme or different MRI.*

Link to ft_plot_mesh help: http://www.fieldtriptoolbox.org/reference/ft_plot_mesh

Link to ft_electroderealign help: http://www.fieldtriptoolbox.org/reference/ft_electroderealign

Template 3D electrode sets information: http://www.fieldtriptoolbox.org/template/electrode

Link to headmodel EEG BEM tutorial: http://www.fieldtriptoolbox.org/tutorial/headmodel_eeg_bem

Preparing the Lead field Matrix

The next step prior to source analysis is preparing the lead field matrix using **ft_prepare_leadfield**. In order to implement source analysis, you must provide a grid in **cfg.grid** that is a structure produced using **ft_prepare_leadfield** or **ft_prepare_sourcemodel**. If you do not specify a grid of pre-computed lead fields, the lead field will be generated in the source analysis step. The function to prepare the lead field also computes the source model, so it is unnecessary to call that function. This function requires the input of the previously computed timelocked data that includes the covariance matrix.

The **ft_prepare_leadfield** function computes the forward model for many dipole locations on a 3D grid in advance to efficiently compute the inverse model, since this computation takes some time to complete. This function requires the choice of the grid resolution. The script uses a 3D grid with a resolution of 5 mm for the virtual sources. The parameter **cfg.reducerank** can be used to remove the weakest orientation (the default is to keep all 3 orientations for EEG data). The parameter **cfg.normalize** can be used to normalize the columns to help remove the center-of-head bias (discussed in source analysis section). However, it is not recommended if you are attempting to contrast the activity of interest against another condition or baseline time-window.

Link to ft_prepare_leadfield help: http://www.fieldtriptoolbox.org/reference/ft_prepare_leadfield

Link to ft_prepare_sourcemodel help:
http://www.fieldtriptoolbox.org/reference/ft_prepare_sourcemodel

Link to source modelling tutorial: http://www.fieldtriptoolbox.org/tutorial/sourcemodel

Source Analysis using Beamforming

Congratulations, you have finally reached the beamforming stage. The function used to perform beamforming on the data is **ft_sourceanalysis**. As mentioned before, there are different methods of beamforming available on Fieldtrip that include time domain analysis and frequency domain analysis. The method that was used in the script is the Linearly Constrained Minimum Variance (LCMV) beamformer.

In the configuration parameters, you can include the previously computed head model, lead field matrix, and electrode alignment positions. This function also requires the input of the data structure obtained from the timelock analysis function. The output of this function, source_lcmv, will include the filter for each virtual source location as well as the average power of the each of these virtual sources over time. You can also choose to keep the trials, the lead field matrix, and the filter created by the beamformer for further processing. You will need to keep the trials, **cfg.keeptrials**, if you intend to project each trial through the common spatial filter. You will also need to keep the filter and the lead field matrix, **cfg.lcmv.keepfilter** and **cfg.keepleadfield**, if you intend to split the trials into time windows to analyse the sources over time.

This brings us to the main issue encountered with beamforming using FieldTrip. In theory, each source signal can be computed with respect to time by multiplying the lead field matrix by the spatial filter. However, the output of the source analysis function on FieldTrip outputs the averaged power of the signal across time for each virtual source location. When examining the **ft_sourceanalysis** code, you will find that there is a method for obtaining the sources with respect to time called 'trace'. However, this code is commented out saying that it has been disabled due to Bugzilla bug 2395. More comments on this problem are provided in the Future Improvements section. The solution I have implemented in order to get around this problem is to split the trials into time windows, as described in the following section.

Another issue that arises when applying beamforming is the center-of-head bias where the sources appear only in the center of the head as shown in the figure below.

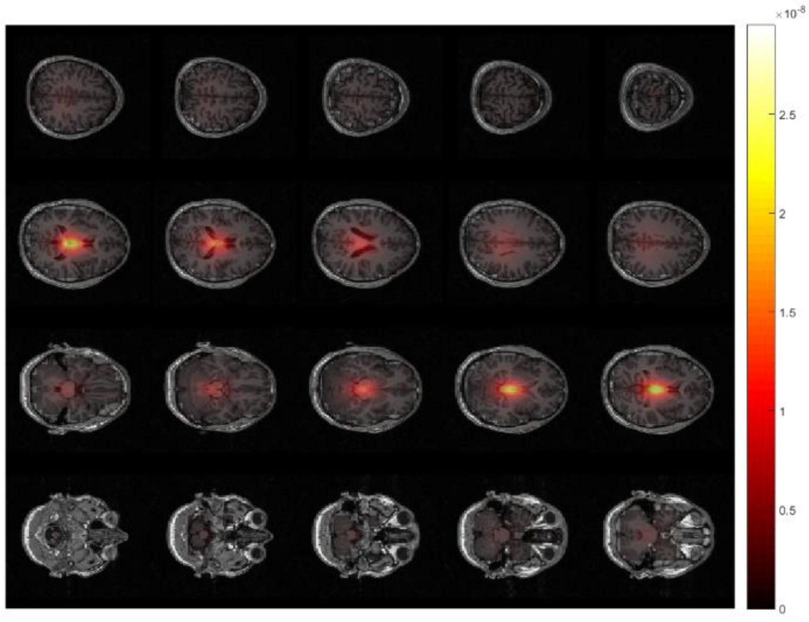


Figure 9: Center of head bias

This issue can be resolved by either normalizing the columns (as previously mentioned) or by applying the neural activity index (NAI). The NAI is the power normalized with respect to the estimated spatially inhomogeneous noise. The noise estimate is done by setting the **cfg.lcmv.projectnoise** parameter to 'yes' in the source analysis function. The noise is estimated on the bases of the smallest eigenvalue of the covariance matrix. The NAI is performed in the script after the time windows are applied and more details about it can be found in the beamformer tutorial.

After the neural activity index is applied, the center-of-head bias is removed and when plotted the sources appear as shown in the following figure.

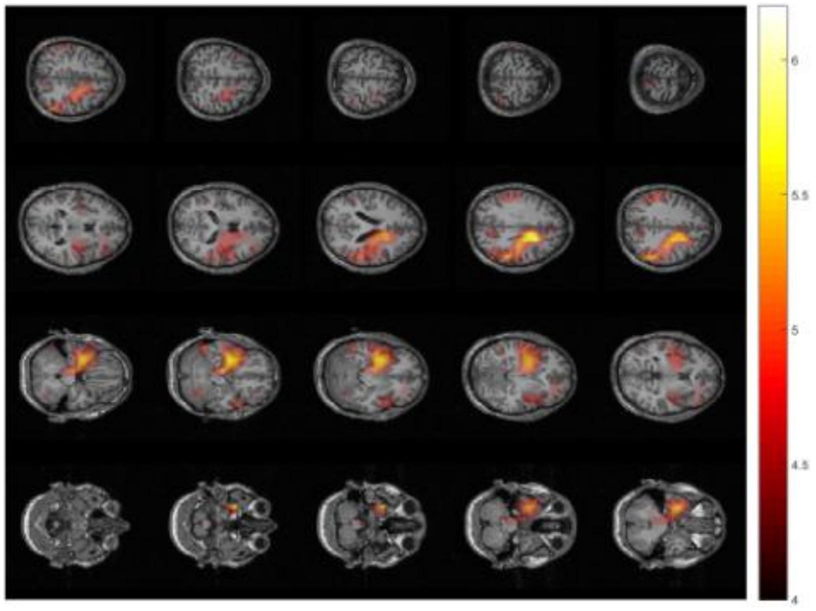


Figure 10: NAI plotted for the entire trial length (using the bemcp head model)

Note: you will find some comments of failed trials after the beamformer experimenting with ft_sourcedescriptives, ft_sourcemovie, and projecting the trials through a common spatial filter. You can attempt to use this or ignore it if you wish.

Link to ft_sourceanalysis help: http://www.fieldtriptoolbox.org/reference/ft_sourceanalysis

Links to beamformer tutorials: http://www.fieldtriptoolbox.org/tutorial/beamformer
http://www.fieldtriptoolbox.org/tutorial/beamformingextended
http://www.fieldtriptoolbox.org/tutorial/natmeg/beamforming
http://www.fieldtriptoolbox.org/tutorial/beamformer_lcmv

Splitting into Time Windows

This section describes the method used to deal with the beamformer 'trace' problem highlighted above. Since the source analysis function can only calculate the average power over the trial length rather than trace through the source with respect to time, the trials were further split into shorter time windows in order to find the average power of the sources and track them along the specified time windows. This method also calculates the average of all the trials with respect to each time window.

In order to further split the already defined trials into smaller time windows, **ft_redefinetrial** was used. The script splits the trials into 16 non-overlapping windows with a length of 32 samples. This was calculated to ensure that it divides exactly into the 512 samples of the trials to avoid having any of the data in the trials truncated or removed. You may also choose to have overlap in the time windows to create a moving window effect resulting in more time windows.

The timelock analysis is then performed again to implement source analysis on each time window. There is no need to calculate the lead field matrix for each time window since the forward model is expected to be the same throughout the length of the trial. Since the time windows have been averaged for a large number of trials, this does not pose an issue of having a rank deficient matrix due to the sample size (32 samples) being less than the number of channels (64 channels). However, this may pose an issue depending on the dataset size, therefore some experimentation on window size and overlap may be needed. The script did not encounter a rank deficiency error while running the beamformer, so it was assumed that this method of splitting into time windows is valid.

Finally, the previously mentioned neural activity index (NAI) was taken to obtain accurate sources (as previously mentioned) before the sources were plotted. This process requires the **ft_sourceinterpolation** function which interpolates the source activity onto the anatomical representation of the brain. The interpolation process usually requires more computational power and a special method of splitting the data cell was implemented to avoid the issue of not being able to save large amounts of data on MATLAB.

Link to ft_redefinetrial help: http://www.fieldtriptoolbox.org/reference/ft_redefinetrial

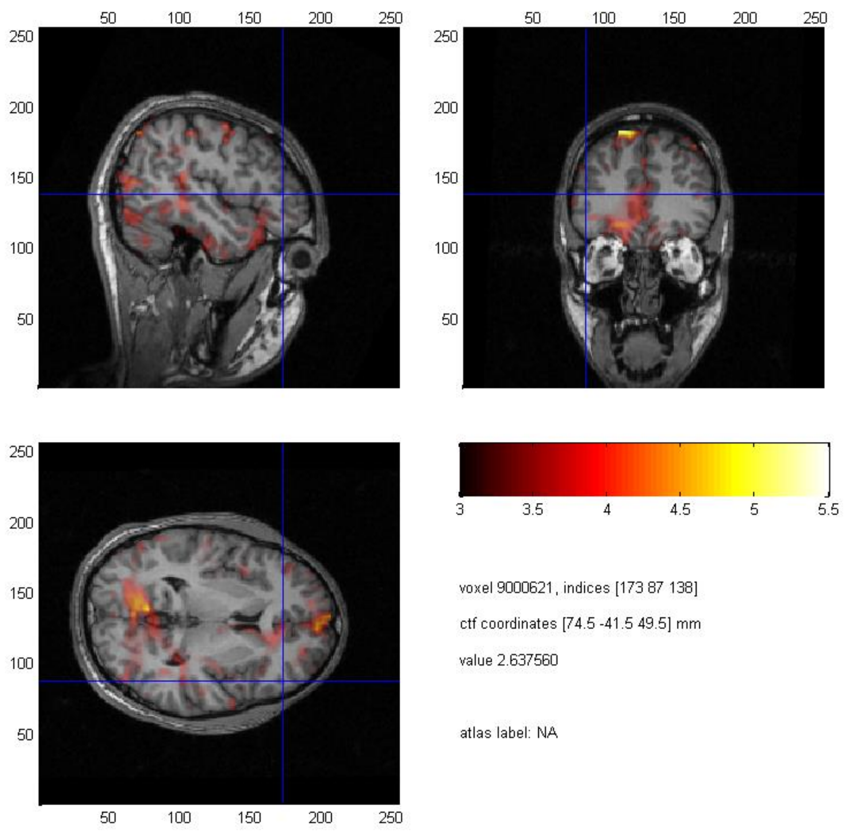
Link to ft_sourceinterpolate help: http://www.fieldtriptoolbox.org/reference/ft_sourceinterpolate

Plotting

The final stage in this process is to plot the sources to be able to visually inspect the sources in the brain. The function that can be used to do this is the **ft_sourceplot** function which has five different methods for plotting the data. Two different methods are shown here as examples; the 'ortho' and 'slice' methods.

The ortho method is an interactive process that allows you to inspect the 3D scan of the brain by choosing the desired coronal, sagittal, and transverse planes. This method is not ideal when you are trying to examine the sources across time since you would need to look at the source plot of each time window separately (with respect to a particular coronal, sagittal, and transverse plane).

The slice method shows the source power information overlaid onto transverse slices of the MRI. This effectively shows the power of the sources for the entire brain at once. Therefore, this method is preferred in order to show the brain sources across time. The script generates plots for each time window and combines them into a single movie to show if there are changes in source trajectories over time.



voxel 9000621, indices [173 87 138]
ctf coordinates [74.5 -41.5 49.5] mm
value 2.637560

Figure 11: Ortho source plot

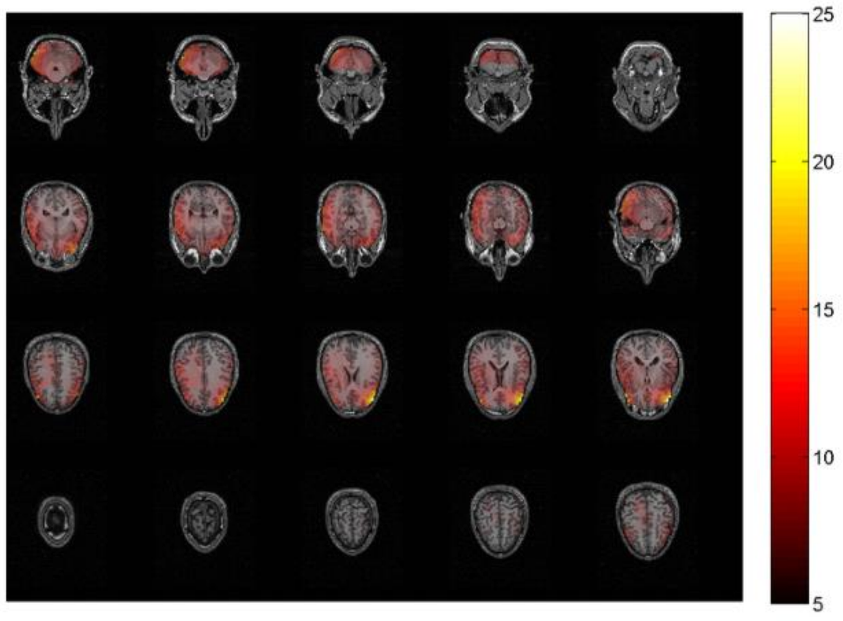


Figure 12: Slice source plot

The parameters used are similar for both methods. The **cfg.funparameter** specifies the actual information you wish to overlay onto the anatomical scan. The **cfg.maskparameter** can be used to apply varied opacity depending on the strength of the source signal. It is recommended to first examine the limits of the plots using ‘zeromax’ to have an idea of the min and max range for the output, then apply that range to all of your data for better comparison between the different conditions in the dataset. The tutorial on plotting provides more detailed information on these parameters.

Link to plotting tutorial: http://www.fieldtriptoolbox.org/tutorial/plotting

Link to ft_sourceplot help: http://www.fieldtriptoolbox.org/reference/ft_sourceplot

Future Improvements

The code included in the appendix has reached a point where it can be applied to accurately find brain sources for different control and concussion datasets (or any dataset) if the proper changes are applied in the preprocessing stage. However, I have included this list of suggested improvements and guidance for the next steps required in the process.

The first issue that should be investigated is the inability to trace the sources through time, as mentioned in the source analysis section above. It would be an improvement to be able to obtain the output signal with respect to time instead of the average power across the trial length (i.e. the "trace" parameter). This may require examining the **ft_sourceanalysis** function more closely and determining the issues with the trace method (Bugzilla bug 2395). This issue may be resolved in a newer version of the FieldTrip toolbox, so keep an eye out for that. It is possible to take the filters generated using source analysis and multiplying it with the lead field matrix to obtain the source power with respect to time for each virtual source on MATLAB. However, this poses the issue of how you will plot the data without using the FieldTrip toolbox. If this issue remains unsolved, it should be fine to continue using the time windowing method and possible attempting to improve on it by experimenting with the window size and overlap parameters.

The next stage in the process would be to analyse multiple EEG data recordings and find the average among the different conditions. This process will involve the use of **ft_sourcegrandaverage** and **ft_sourcestatistics** as shown in the analysis protocol figure below. You may also need to use **ft_sourcedescriptives** to extract useful information from the source analysis. The function **ft_sourcemovie** may also be used to create movies of the sources over time, but may also depend on the trace issue mentioned above.

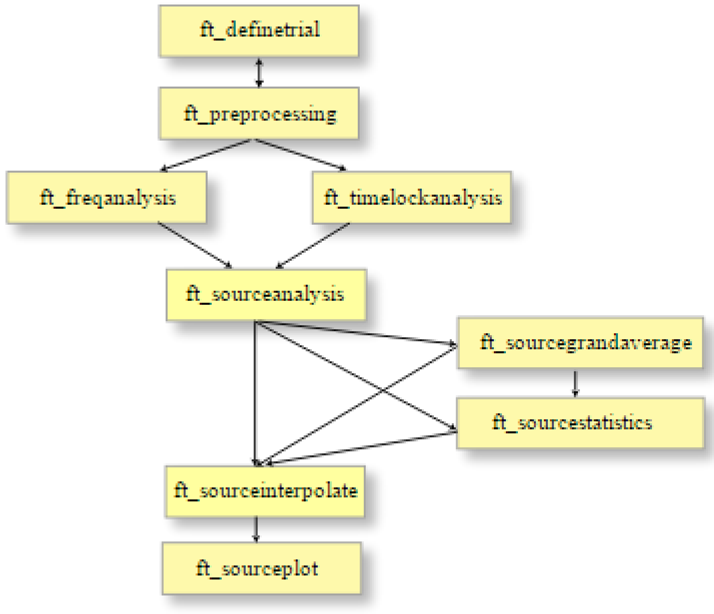


Figure 13: Example of Analysis Protocol for Beamforming

Link to ft_sourceanalysis help: http://www.fieldtriptoolbox.org/reference/ft_sourceanalysis

Link to ft_sourcegrandaverage help: http://www.fieldtriptoolbox.org/reference/ft_sourcegrandaverage

Link to ft_sourcestatistics help: http://www.fieldtriptoolbox.org/reference/ft_sourcestatistics

Link to ft_sourcedescriptives help: http://www.fieldtriptoolbox.org/reference/ft_sourcedescriptives

Link to ft_sourcemovie help: http://www.fieldtriptoolbox.org/reference/ft_sourcemovie