FIELDTRIP TOOLBOX TUTORIAL REPORT

PRADEEP KUMAR YADAV | BM18MTECH11005

TUTORIAL 1:-

Preprocessing - Reading continuous EEG and MEG data

Introduction-

Here we are using ft_preprocessing to read continuous data fully in memory that is our brain is continuously working.it is feasible at once when our data set is relatively small and our computer have enough memory to hold the data at once.

Background-

Now we can read the data from file in memory and can apply the filte and can be cut in different segments interestingly.

Procedure-

The following steps has to be followed ,and select or cut different constant length of segments.

- read the data for the EEG channels using **ft_preprocessing**, apply a filter and re-reference to linked mastoids.
- read the data for the horizontal and vertical EOG channels using **ft_preprocessing**, and compute the horizontal and vertical bipolar EOG derivations.
- combine the EEG and EOG into a single data representation using **ft_appenddata**
- determine interesting pieces of data based on the trigger events using **ft_definetrial**
- segment the continuous data into trials using **ft_redefinetrial**
- segment the continuous data into one-second pieces using **ft_redefinetrial**

The data set used in this tutorial-

There are two data set used in this tutorial EEG dataset SubjectEEG.zip and for MEG dataset Subject01.zip. Both has been extracted and saved in folder and import in matlab workspace.

Reading continuous EEG data into memory

reading the data in memory simply call ft_preprocessing function with only data set as configuration argument.

```
FT_PREPROCESSING reads MEG and/or EEG data according to user-specified trials
 and applies several user-specified preprocessing steps to the signals.
```

```
cfg = [];
cfg.dataset     = 'subj2.vhdr';
data_eeg        = ft_preprocessing(cfg)
```

I have explained above that we have cut the data in constant segment.so here cfg variable is constant length data data .which is taken from subject2 (EEG). The variable cfg.dataset is defined as subject2 data .and further data_eeg is defined where function ft_preprocessing applied on segmented constant data.

After proceesing the below results is obtained-

processing channel { '1' '2' '3' '4' '5' '6' '7' '8' '9' '10' '11' '12' '13' '14' '15' '16' '17' '18' '19' '20' '21' '22' '23' '24' '25' '26' '27' '28' '29' '30' '31' '32' '33' '34' '35' '36' '37' '38' '39' '40' '41' '42' '43' '44' '45' '46' '47' '48' '49' '50' '51' '52' '53' '54' '55' '56' '57' '58' '59' '60' '61' '62' '63' '64' }

reading and preprocessing

reading and preprocessing trial 1 from 1

the call to "ft_preprocessing" took 48 seconds

data_eeg =

struct with fields:

hdr: [1×1 struct]

label: {64×1 cell}

time: {[1×2011220 double]}

trial: {[64×2011220 double]}

fsample: 500

sampleinfo: [1 2011220]

cfg: [1×1 struct]

it means it has 64 channel and in first trail it took 48 second to execute and read the eeg data.

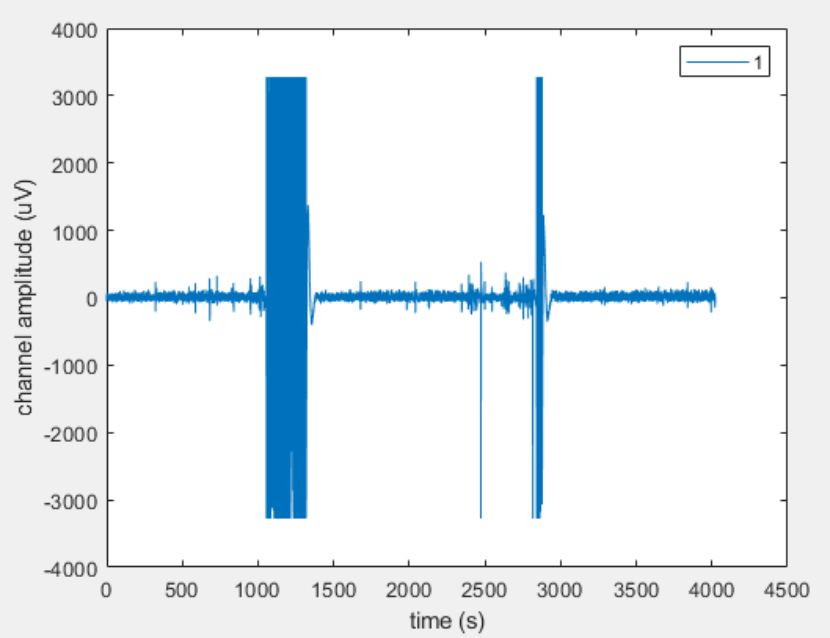
In result I have obtained header structure array and trail time measurement and total time measurement and no of sample and sampling frequency and our cut segmented data structure.

Now I am going to plot the trial segmented data which represent the potential of the trail segmented continuous data without using any additional processing filter.

```
chansel  = 1;
plot(data_eeg.time{1}, data_eeg.trial{1}(chansel, :))
xlabel('time (s)')
ylabel('channel amplitude (uV)')
legend(data_eeg.label(chansel))
```

I have selected channel 1

The plot I have obtained is below



Amplitude(uv) versus time(s) plot is obtained.

Clearly we can see the artifacts at particular interval of time. The artifact is nothing but spikes and bump in signal.

By changing the channel number we are getting different signal analysis ,its depend on the channel where the electrode are placed and some channel are duumy and some channel are connected to eog that gives the different signal analysis.

Lets take channel 62chansel = 62;

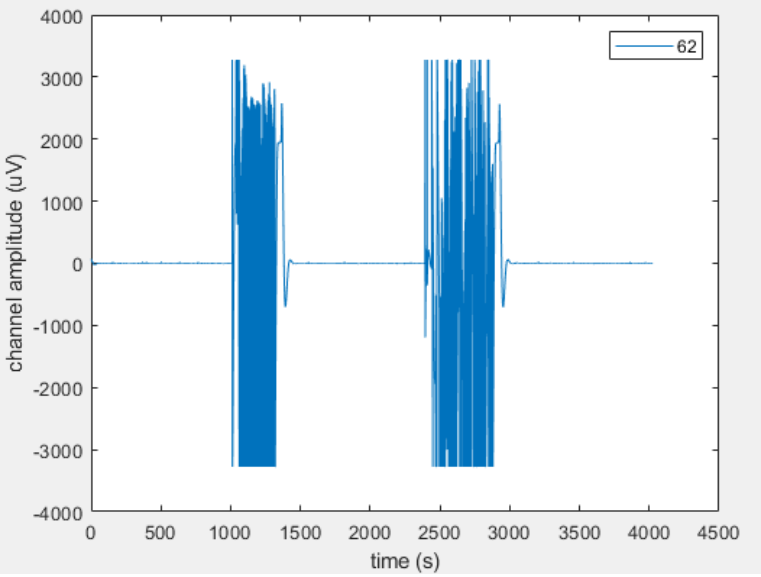
plot(data_eeg.time{1}, data_eeg.trial{1}(chansel, :))

xlabel('time (s)')

ylabel('channel amplitude (uV)')

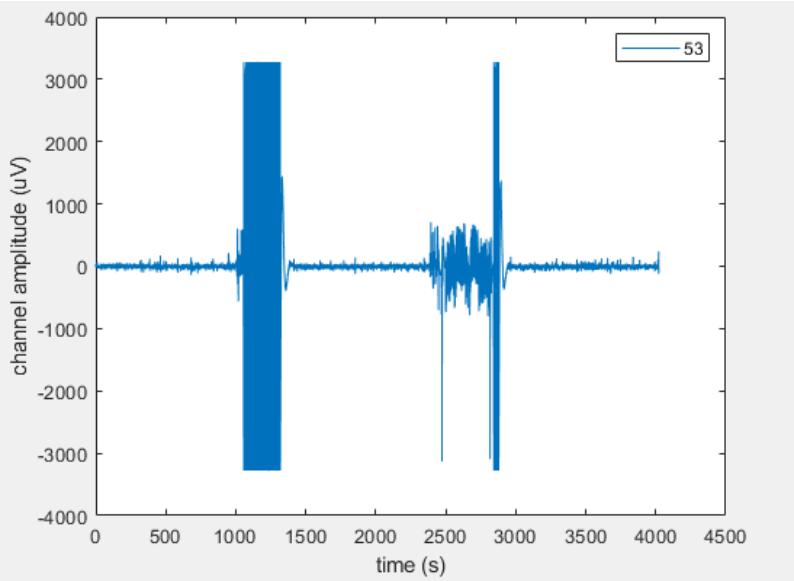
legend(data_eeg.label(chansel))

results_



Its different according to me its artifacts only because channel 61,62,63 are dummy channel which is not connected to electrode or not aimed to take the measurement so that's why only artifacts are there only.

Now check on channel 53



Let us see further that why this signal is different, it might be clear in the next page

Reading continuous MEG data into memory

Data is already stored in segmented or epoched format, the trail data will reflect when we call ft_preprocessing being read and segmented into original trail.

```
cfg = [];
cfg.dataset     = 'Subject01.ds';
data_meg        = ft_preprocessing(cfg);
```

processing channel { 'STIM' 'SCLK01' 'BG1' 'BG2' 'BG3' 'BP1' 'BP2' 'BP3' 'BQ1' 'BQ2' 'BQ3' 'BR1' 'BR2' 'BR3' 'G11' 'G12' 'G13' 'G22' 'G23' 'P11' 'P12' 'P13' 'P22' 'P23' 'Q11' 'Q12' 'Q13' 'Q21' 'Q22' 'Q23' 'R11' 'R12' 'R13' 'R22' 'R23' 'MLC11' 'MLC12' 'MLC13' 'MLC14' 'MLC15' 'MLC21' 'MLC22' 'MLC23' 'MLC24' 'MLC31' 'MLC32' 'MLC33' 'MLC41' 'MLC42' 'MLC43' 'MLF11' 'MLF12' 'MLF21' 'MLF22' 'MLF23' 'MLF31' 'MLF32' 'MLF33' 'MLF34' 'MLF41' 'MLF42' 'MLF43' 'MLF44' 'MLF45' 'MLF51' 'MLF52' 'MLO11' 'MLO12' 'MLO21' 'MLO22' 'MLO31' 'MLO32' 'MLO33' 'MLO41' 'MLO42' 'MLO43' 'MLP11' 'MLP12' 'MLP13' 'MLP21' 'MLP22' 'MLP31' 'MLP32' 'MLP33' 'MLP34' 'MLT11' 'MLT12' 'MLT13' 'MLT14' 'MLT15' 'MLT16' 'MLT21' 'MLT22' 'MLT23' 'MLT24' 'MLT25' 'MLT26' 'MLT31' 'MLT32' 'MLT33' 'MLT34' 'MLT35' 'MLT41' 'MLT42' 'MLT43' 'MLT44' 'MRC11' 'MRC12' 'MRC13' 'MRC14' 'MRC15' 'MRC21' 'MRC22' 'MRC23' 'MRC24' 'MRC31' 'MRC32' 'MRC33' 'MRC41' 'MRC42' 'MRC43' 'MRF11' 'MRF12' 'MRF21' 'MRF22' 'MRF23' 'MRF31' 'MRF32' 'MRF33' 'MRF34' 'MRF41' 'MRF42' 'MRF43' 'MRF44' 'MRF45' 'MRF51' 'MRF52' 'MRO11' 'MRO12' 'MRO21' 'MRO22' 'MRO31' 'MRO32' 'MRO33' 'MRO41' 'MRO42' 'MRO43' 'MRP11' 'MRP12' 'MRP13' 'MRP21' 'MRP22' 'MRP31' 'MRP32' 'MRP33' 'MRP34' 'MRT11' 'MRT12' 'MRT13' 'MRT14' 'MRT15' 'MRT16' 'MRT21' 'MRT22' 'MRT23' 'MRT24' 'MRT25' 'MRT26' 'MRT31' 'MRT32' 'MRT33' 'MRT34' 'MRT35' 'MRT41' 'MRT42' 'MRT43' 'MRT44' 'MZC01' 'MZC02' 'MZF01' 'MZF02' 'MZF03' 'MZO01' 'MZO02' 'MZP01' 'MZP02' 'EOG' }

reading and preprocessing

reading and preprocessing trial 266 from 266

the call to "ft_preprocessing" took 22 seconds

data_meg =

 struct with fields:

        hdr: [1×1 struct]

label: {187×1 cell}

time: {1×266 cell}

trial: {1×266 cell}

fsample: 300

sampleinfo: [266×2 double]

grad: [1×1 struct]

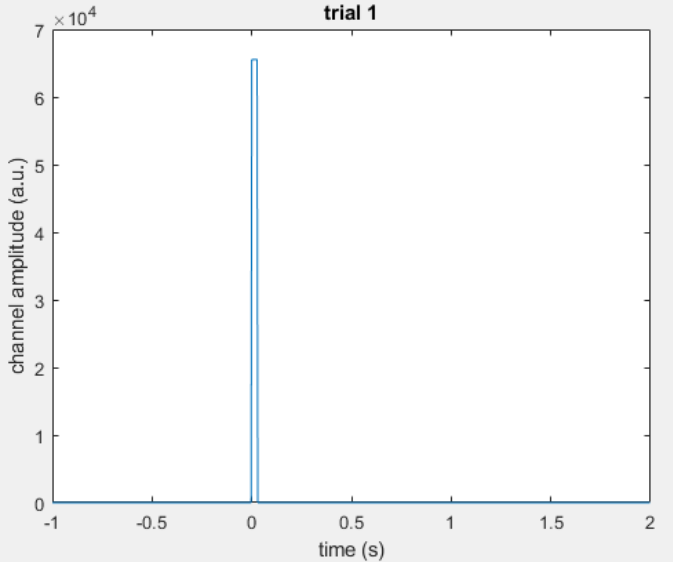
elec: [1×1 struct]

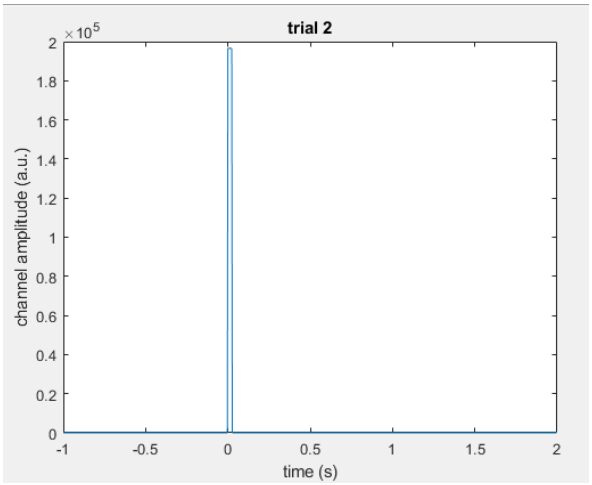
cfg: [1×1 struct]

these are 266 channel MEG it took 22 second for reading and processing.

```
for trialsel=1:2
  chansel = 1; % this is the STIM channel that contains the trigger
  figure
  plot(data_meg.time{trialsel}, data_meg.trial{trialsel}(chansel, :))
  xlabel('time (s)')
  ylabel('channel amplitude (a.u.)')
  title(sprintf('trial %d', trialsel));
end
```

trigger at same point of time but getting different channel amplitude .





These plots are subset of the trails.

If you want to force epoched data to be interpreted as continuous data, you can use the cfg.continuous option, like this

```
cfg = [];

cfg.dataset     = 'Subject01.ds';

cfg.continuous  = 'yes';              % force it to be continuous

data_meg        = ft_preprocessing(cfg);


chansel = 2;                          % this is SCLK01

plot(data_meg.time{1}, data_meg.trial{1}(chansel, :))

xlabel('time (s)')

ylabel(data_meg.label{chansel})

Warning: assuming that the units are "m"

 In ft_estimate_units at line 52

 In ft_determine_units at line 83
```

In ft_datatype_sens at line 372

In ft_datatype_sens at line 158

In ft_read_header at line 2654

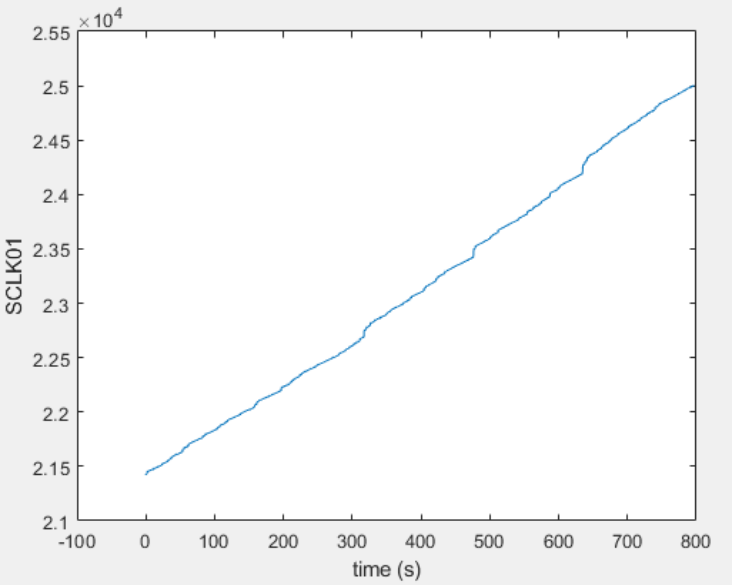
In ft_preprocessing at line 397

processing channel { 'STIM' 'SCLK01' 'BG1' 'BG2' 'BG3' 'BP1' 'BP2' 'BP3' 'BQ1' 'BQ2' 'BQ3' 'BR1' 'BR2' 'BR3' 'G11' 'G12' 'G13' 'G22' 'G23' 'P11' 'P12' 'P13' 'P22' 'P23' 'Q11' 'Q12' 'Q13' 'Q21' 'Q22' 'Q23' 'R11' 'R12' 'R13' 'R22' 'R23' 'MLC11' 'MLC12' 'MLC13' 'MLC14' 'MLC15' 'MLC21' 'MLC22' 'MLC23' 'MLC24' 'MLC31' 'MLC32' 'MLC33' 'MLC41' 'MLC42' 'MLC43' 'MLF11' 'MLF12' 'MLF21' 'MLF22' 'MLF23' 'MLF31' 'MLF32' 'MLF33' 'MLF34' 'MLF41' 'MLF42' 'MLF43' 'MLF44' 'MLF45' 'MLF51' 'MLF52' 'MLO11' 'MLO12' 'MLO21' 'MLO22' 'MLO31' 'MLO32' 'MLO33' 'MLO41' 'MLO42' 'MLO43' 'MLP11' 'MLP12' 'MLP13' 'MLP21' 'MLP22' 'MLP31' 'MLP32' 'MLP33' 'MLP34' 'MLT11' 'MLT12' 'MLT13' 'MLT14' 'MLT15' 'MLT16' 'MLT21' 'MLT22' 'MLT23' 'MLT24' 'MLT25' 'MLT26' 'MLT31' 'MLT32' 'MLT33' 'MLT34' 'MLT35' 'MLT41' 'MLT42' 'MLT43' 'MLT44' 'MRC11' 'MRC12' 'MRC13' 'MRC14' 'MRC15' 'MRC21' 'MRC22' 'MRC23' 'MRC24' 'MRC31' 'MRC32' 'MRC33' 'MRC41' 'MRC42' 'MRC43' 'MRF11' 'MRF12' 'MRF21' 'MRF22' 'MRF23' 'MRF31' 'MRF32' 'MRF33' 'MRF34' 'MRF41' 'MRF42' 'MRF43' 'MRF44' 'MRF45' 'MRF51' 'MRF52' 'MRO11' 'MRO12' 'MRO21' 'MRO22' 'MRO31' 'MRO32' 'MRO33' 'MRO41' 'MRO42' 'MRO43' 'MRP11' 'MRP12' 'MRP13' 'MRP21' 'MRP22' 'MRP31' 'MRP32' 'MRP33' 'MRP34' 'MRT11' 'MRT12' 'MRT13' 'MRT14' 'MRT15' 'MRT16' 'MRT21' 'MRT22' 'MRT23' 'MRT24' 'MRT25' 'MRT26' 'MRT31' 'MRT32' 'MRT33' 'MRT34' 'MRT35' 'MRT41' 'MRT42' 'MRT43' 'MRT44' 'MZC01' 'MZC02' 'MZF01' 'MZF02' 'MZF03' 'MZO01' 'MZO02' 'MZP01' 'MZP02' 'EOG' }

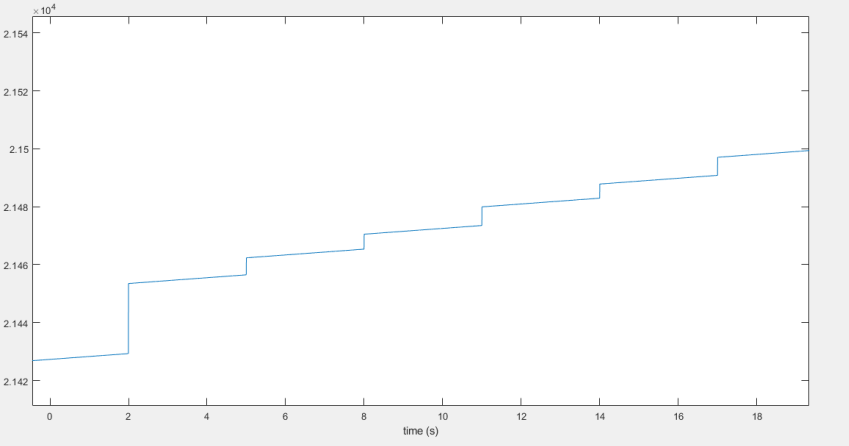
reading and preprocessing

reading and preprocessing trial 1 from 1

the call to "ft_preprocessing" took 6 seconds



If you zoom in and look in detail at the SCLK01 channel, you can see that there are small jumps every 3 seconds. These are due to the data being discontinuous on disk, i.e. only the 3 second segments around each stimulus are stored on disk.



hence this particular dataset should not be interpreted as a continuous recording.

Preprocessing, filtering and re-referencing

For preprocessing this EEG data set, the choice of the reference has to be considered. During acquisition the reference channel of the EEG amplifier was attached to the left mastoid. We

would like to analyze this data with a linked-mastoid reference (also known as an average-mastoid reference).

The electrode of EOG data is placed horizontally and vertically i.e. called bibolar electrode to measure the eye muscle movement .

Channel 1to 60 is defined and connected to amplifier and recorder and channel 53 is connected to right mastoid ,and left mastoid is not connected so it defined as a reference. it is not represented in the data file (because the Voltage at that electrode is zero by definition).

```
cfg = [];
cfg.dataset     = 'subj2.vhdr';
cfg.reref       = 'yes';
cfg.channel     = 'all';
cfg.implicitref = 'M1';         % the implicit (non-recorded) reference channel is
added to the data representation
cfg.refchannel  = {'M1', '53'}; % the average of these two is used as the new
reference, channel '53' corresponds to the right mastoid (M2)
data_eeg        = ft_preprocessing(cfg);
```

M1 is implicit(non recorded)reference channel.

M1 and channel 53(M2) are averaged and used as a new reference channel.

```
chanindx = find(strcmp(data_eeg.label, '53'));
data_eeg.label{chanindx} = 'M2';
```

whatever the data in data_eed.label that is compared with no 53 channel .it should have same matrix dimension.

to decard the channel that we don't need anymore.

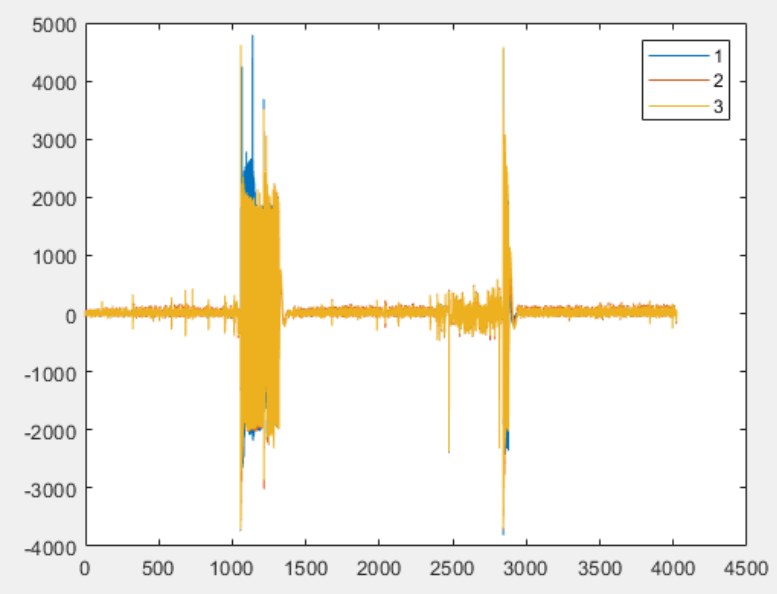
(M1 +M2)/2 would be new reference channel.

```
cfg = [];
cfg.channel     = [1:61 65];                    % keep channels 1 to 61 and the
newly inserted M1 channel
data_eeg        = ft_preprocessing(cfg, data_eeg);
```

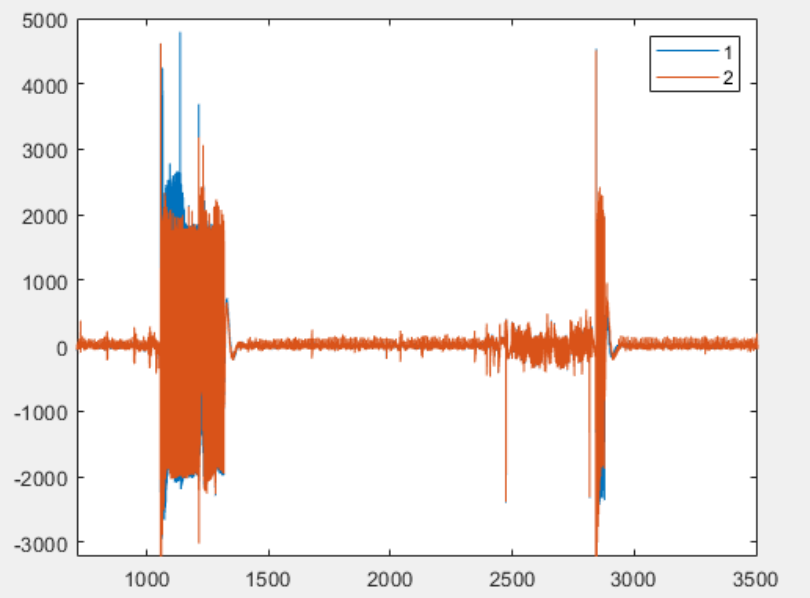
If we look that it contain sigle trail continuous recording which takes one hour long.

```
plot(data_eeg.time{1}, data_eeg.trial{1}(1:3,:));
legend(data_eeg.label(1:3));
```

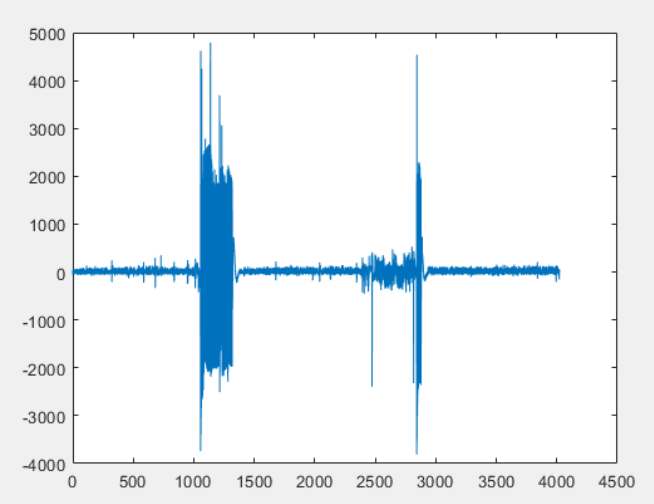
the figure below is complete one hour recording in a single trail.




```
plot(data_eeg.time{1}, data_eeg.trial{1}(1:2,:));
legend(data_eeg.label(1:2));
```



```
          plot(data_eeg.time{1}, data_eeg.trial{1}(1:1,:));
legend(data_eeg.label(1:1));
```



Subsequently we read the data for the horizontal EOG

```
cfg = [];
cfg.dataset    = 'subj2.vhdr';
cfg.channel    = {'51', '60'};
cfg.reref      = 'yes';
cfg.refchannel = '51';
data_eogh      = ft_preprocessing(cfg);
```

reading and preprocessing trial 1 from 1

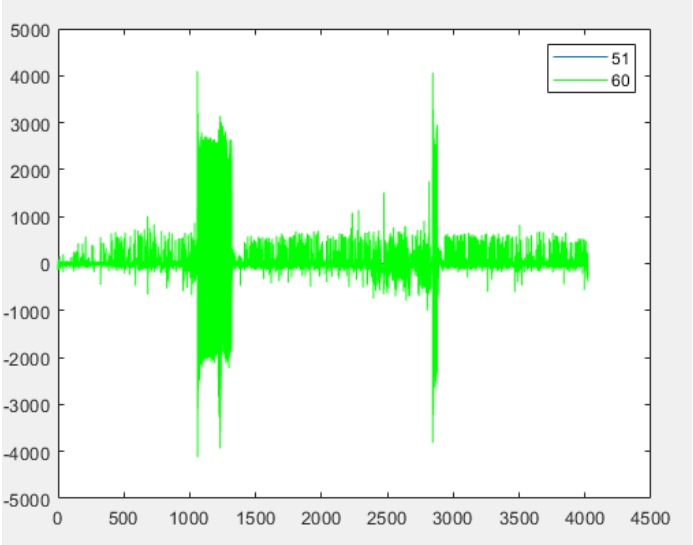
The resulting channel 51 in this representation of the data is referenced to itself, which means that it contains zero values. This can be checked by figure below

```
figure
plot(data_eogh.time{1}, data_eogh.trial{1}(1,:));
hold on
plot(data_eogh.time{1}, data_eogh.trial{1}(2,:),'g');
legend({'51' '60'});
```

results

clearly we can see that 51 is at zero amplitude and 60 have recording and response data.

That means 51 is assigned as reference and 60 is assigned as EOGH(horizontal) recorded data.



For convenience we rename channel 60 into EOGH and use the **ft_preprocessing** function once more to select the horizontal EOG channel and discard the dummy channel.

```
data_eogh.label{2} = 'EOGH';

cfg = [];
cfg.channel = 'EOGH';
data_eogh   = ft_preprocessing(cfg, data_eogh); % nothing will be done, only the
selection of the interesting channel
```

The processing of the vertical EOG is done similar, using the difference between channel 50 and 64 as the bipolar EOG

```
cfg = [];
cfg.dataset    = 'subj2.vhdr';
cfg.channel    = {'50', '64'};
cfg.reref      = 'yes';
cfg.refchannel = '50'
data_eogv      = ft_preprocessing(cfg);

data_eogv.label{2} = 'EOGV';

cfg = [];
cfg.channel = 'EOGV';
data_eogv   = ft_preprocessing(cfg, data_eogv); % nothing will be done, only the
selection of the interesting channel
```

this is for vertical EOG recording. Reading and processing the data.

Now we have EEG data referenced to linked mastoid and bipolar EOG data ,so we can combine all the raw data into a single reperesentation.

```
cfg = [];
data_all = ft_appenddata(cfg, data_eeg, data_eogh, data_eogv);
```

In the above data we have not applied any filter/rectification/re-referencing and other preprocessing function to reduce or eliminate the artifacts and other. For that we can call ft_preprocessing.

Segmenting continuous data into trials

For reading and channel specific preprocessing, you can identify interesting pieces of data based on the trigger codes and segment the continuous data into trials.

```
cfg = [];
cfg.dataset              = 'subj2.vhdr';
cfg.trialdef.eventtype = '?';
dummy                    = ft_definetrial(cfg);
```

In ft_definetrial at line 145

evaluating trialfunction 'ft_trialfun_general'

reading the header from 'subj2.vhdr'

reading the events from 'subj2.vhdr'

the following events were found in the datafile

event type: 'New Segment'

with event values:

event type: 'Response'

with event values: 'R 8'

event type: 'Stimulus'

with event values: 'S 1' 'S 12' 'S 13' 'S 21' 'S 27' 'S111' 'S112' 'S113' 'S121' 'S122' 'S123' 'S131' 'S132' 'S133' 'S141' 'S142' 'S143' 'S151' 'S152' 'S153' 'S161' 'S162' 'S163' 'S171' 'S172' 'S173' 'S181' 'S182' 'S183' 'S211' 'S212' 'S213' 'S221' 'S222' 'S223' 'S231' 'S232' 'S233' 'S241' 'S242' 'S243'

no trials have been defined yet, see FT_DEFINETRIAL for further help

found 1570 events

created 0 trials

the call to "ft_definetrial" took 4 seconds

```
FT_DEFINETRIAL defines the segments of data that will be used for
 further processing and analysis, i.e. the pieces of data that will
 be read in by FT_PREPROCESSING. Trials are defined by their begin
 and end sample in the data file and each trial has an offset that
 defines where the relative t=0 point (usually the sample at which
 the trigger is detected) is for that trial.
```

The trigger codes S111, S121, S131, S141 correspond to the presented pictures of 4 different animals respectively. The trigger codes S151, S161, S171, S181 correspond to the presented pictures of 4 different tools

Now selecting the data for further analysis

```
cfg = [];
cfg.dataset             = 'subj2.vhdr';
cfg.trialdef.eventtype = 'Stimulus';
cfg.trialdef.eventvalue = {'S111', 'S121', 'S131', 'S141'};
cfg_vis_animal          = ft_definetrial(cfg);

cfg.trialdef.eventvalue = {'S151', 'S161', 'S171', 'S181'};
cfg_vis_tool            = ft_definetrial(cfg);
```

The output configuration resulting from **ft_definetrial** contains the trial definition as a Nx3 matrix with the begin sample, the end sample and the offset of each trial.

In this configuration we have to read and analyse the data by ft_definetrial to find the offset(initial value) and end value.

But we have complete continuous trail data in memory so we have to cut the data from memory ,we'll use **ft_redefinetrial** to cut these trials out of the continuous data segment

Example-

```
. data_vis_animal = ft_redefinetrial(cfg_vis_animal, data_all);
data_vis_tool   = ft_redefinetrial(cfg_vis_tool,   data_all);
```

Subsequently we could do artifact detection with **ft_rejectvisual** to remove trials with artifacts and average the trials using **ft_timelockanalysis** to get the ERP, or use **ft_freqanalysis** to obtain an averaged time-frequency representation of the data in both conditions. If you use ft_timelockanalysis or ft_frequencyanalysis with the option cfg.keeptrials='yes', you subsequently could use ft_timelockstatistics or ft_freqstatistics for statistical comparison of the animal-tool contrast in these stimuli.

Segmenting continuous data into one-second pieces

For processing of continuous data without triggers we have to cut the data in constant length piece. This can be done from disk or it can be done after the complete continuous data is in memory.

Example-

```
cfg = [];
cfg.dataset              = 'subj2.vhdr';
cfg.trialfun             = 'ft_trialfun_general';
cfg.trialdef.triallength = 1;                       % duration in seconds
cfg.trialdef.ntrials     = inf;                     % number of trials, inf results in
as many as possible
cfg                      = ft_definetrial(cfg);

% read the data from disk and segment it into 1-second pieces
data_segmented           = ft_preprocessing(cfg);
```

cfg = [];

cfg.dataset = 'subj2.vhdr';

cfg.trialfun = 'ft_trialfun_general';

cfg.trialdef.triallength = 1; % duration in seconds

cfg.trialdef.ntrials = inf; % number of trials, inf results in as many as possible

cfg = ft_definetrial(cfg);

% read the data from disk and segment it into 1-second pieces

data_segmented = ft_preprocessing(cfg);

evaluating trialfunction 'ft_trialfun_general'

reading the header from 'subj2.vhdr'

reading the events from 'subj2.vhdr'

found 1570 events

created 4022 trials

the call to "ft_definetrial" took 3 seconds

processing channel { '1' '2' '3' '4' '5' '6' '7' '8' '9' '10' '11' '12' '13' '14' '15' '16' '17' '18' '19' '20' '21' '22' '23' '24' '25' '26' '27' '28' '29' '30' '31' '32' '33' '34' '35' '36' '37' '38' '39' '40' '41' '42' '43' '44' '45' '46' '47' '48' '49' '50' '51' '52' '53' '54' '55' '56' '57' '58' '59' '60' '61' '62' '63' '64' }

reading and preprocessing

reading and preprocessing trial 4022 from 4022

the call to "ft_preprocessing" took 78 seconds

the following example shows that how to read the data first and then after cut in to constant length piece.

```
% read it from disk as a single continuous segment
cfg = [];
cfg.dataset              = 'subj2.vhdr';
data_cont                = ft_preprocessing(cfg);

% segment it into 1-second pieces
cfg = [];
cfg.length               = 1;
data_segmented           = ft_redefinetrial(cfg, data_cont);
```

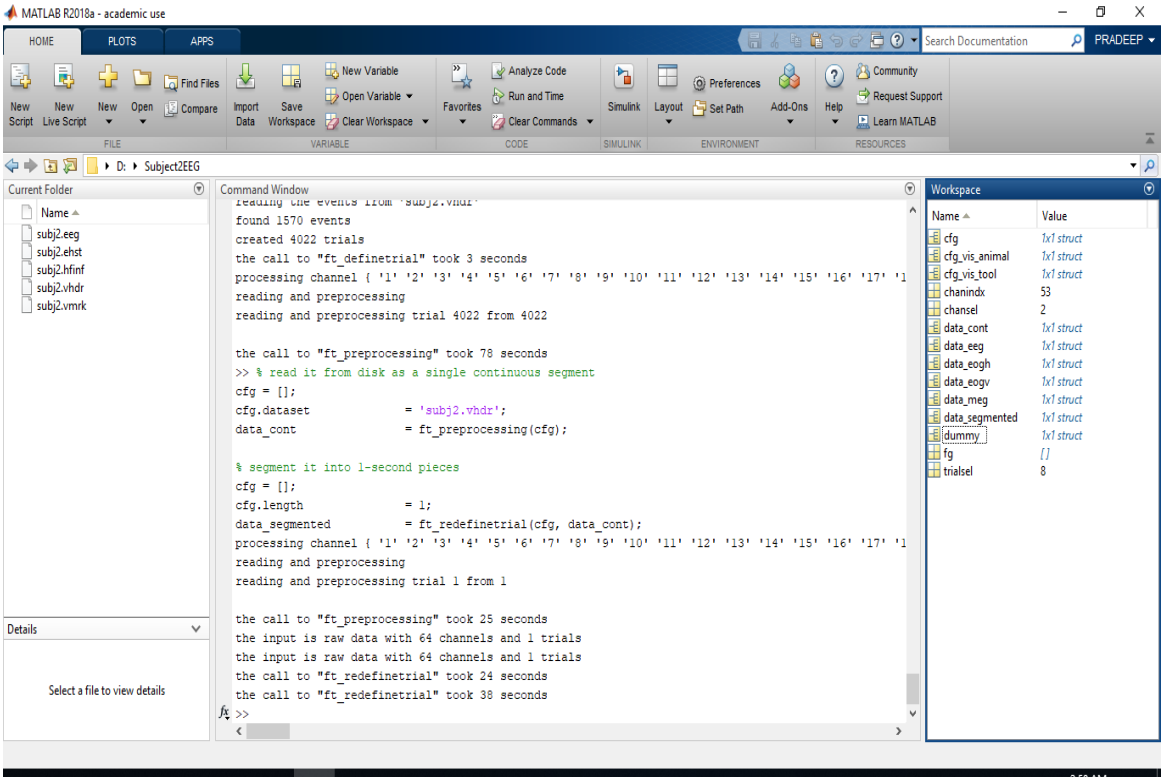
cfg = [];

cfg.dataset = 'subj2.vhdr';

data_cont = ft_preprocessing(cfg);

% segment it into 1-second pieces

```
cfg = [];
cfg.length              = 1;
data_segmented          = ft_redefinetrial(cfg, data_cont);
processing channel { '1' '2' '3' '4' '5' '6' '7' '8' '9' '10' '11' '12' '13' '14' '15' '16' '17'
'18' '19' '20' '21' '22' '23' '24' '25' '26' '27' '28' '29' '30' '31' '32' '33' '34' '35' '36' '37'
'38' '39' '40' '41' '42' '43' '44' '45' '46' '47' '48' '49' '50' '51' '52' '53' '54' '55' '56' '57'
'58' '59' '60' '61' '62' '63' '64' }
reading and preprocessing
reading and preprocessing trial 1 from 1
the call to "ft_preprocessing" took 25 seconds
the input is raw data with 64 channels and 1 trials
the input is raw data with 64 channels and 1 trials
the call to "ft_redefinetrial" took 24 seconds
the call to "ft_redefinetrial" took 38 seconds
```


MATLAB R2018a - academic use
HOME
PLOTS
APPS
Search Documentation
PRADEEP
New Script
New Live Script
New
Open
Find Files
Compare
Import Data
Save Workspace
New Variable
Open Variable
Clear Workspace
Favorites
Analyze Code
Run and Time
Clear Commands
Simulink
Layout
Preferences
Set Path
Add-Ons
Help
Community
Request Support
Learn MATLAB
FILE
VARIABLE
CODE
SIMULINK
ENVIRONMENT
RESOURCES
D: ▸ Subject2EEG
Current Folder
Name
subj2.eeg
subj2.ehst
subj2.hfinf
subj2.vhdr
subj2.vmrk
Details
Select a file to view details
Command Window
found 1570 events
created 4022 trials
the call to "ft_definetrial" took 3 seconds
processing channel { '1' '2' '3' '4' '5' '6' '7' '8' '9' '10' '11' '12' '13' '14' '15' '16' '17' '1
reading and preprocessing
reading and preprocessing trial 4022 from 4022
the call to "ft_preprocessing" took 78 seconds
>> % read it from disk as a single continuous segment
cfg = [];
cfg.dataset = 'subj2.vhdr';
data_cont = ft_preprocessing(cfg);
% segment it into 1-second pieces
cfg = [];
cfg.length = 1;
data_segmented = ft_redefinetrial(cfg, data_cont);
processing channel { '1' '2' '3' '4' '5' '6' '7' '8' '9' '10' '11' '12' '13' '14' '15' '16' '17' '1
reading and preprocessing
reading and preprocessing trial 1 from 1
the call to "ft_preprocessing" took 25 seconds
the input is raw data with 64 channels and 1 trials
the input is raw data with 64 channels and 1 trials
the call to "ft_redefinetrial" took 24 seconds
the call to "ft_redefinetrial" took 38 seconds
>>
Workspace
Name
Value
cfg 1x1 struct
cfg_vis_animal 1x1 struct
cfg_vis_tool 1x1 struct
chanindx 53
chansel 2
data_cont 1x1 struct
data_eeg 1x1 struct
data_eogh 1x1 struct
data_eogv 1x1 struct
data_meg 1x1 struct
data_segmented 1x1 struct
dummy 1x1 struct
fg []
trialsel 8