CCEP preprocessing Procedure

Skye

2021.05.31

資料處理以prep\_CCEP\_v6.m分析, 每個步驟結束人工檢查

##### 1. 讀進單段或多段資料, 存成set檔

• prep\_CCEP\_v6.m檔中第一段:新增需要的資料夾後, 以readedf\_saveset.m讀進edf檔後, 其中會以trans4event.m讀進event, 再存成set檔, **需檢查**

%% filename: prep\_CCEP\_v6.m--------------------------------  
anapath = uigetdir('/Users/nissen/Documents/CCEP', 'Pick a Directory');  
  
if ~exist('prep\_dir','dir')  
 mkdir prep\_dir  
end  
  
if ~exist('erp','dir')  
 mkdir erp  
end  
  
if ~exist('epoch\_mat','dir')  
 mkdir epoch\_mat  
end  
  
prep\_dir=([anapath '/prep\_dir']);  
epoch\_dir=([anapath '/epoch\_mat']);  
erp\_dir=([anapath '/erp']);  
cd(['/Users/nissen/Documents/CCEP/script/'])  
readedf\_saveset  
%% ---------------------------------------------------  
  
%% filename: readedf\_saveset.m--------------------------------  
filename=dir([anapath '/\*.edf']);  
filedir=anapath;  
if isempty(filename)  
 filename=dir([anapath '/edf/\*.edf']);  
 filedir=[anapath '/edf'];  
end  
filename={filename.name}.';  
filename=order4file(filename); % sort  
sprintf('find %d files',size(filename,1))  
  
eeglab  
for x=1:size(filename,1)  
 sprintf('read %d/%d files: %s',x,size(filename,1),filename(x,:))  
 [~,name,~]=fileparts(filename(x,:));  
 expression = '[1-9A-Za-z]+[\_]\*#[\_]\*[1-9][\_]\*';  
 startIndex = regexp(name,expression);  
   
 EEG=pop\_biosig([filedir '/' deblank(filename(x,:))]);  
 if ~isempty(startIndex)  
 EEG.setname= strrep(name(startIndex:end),'#','');  
 else  
 EEG.setname= name;  
 end  
 [ALLEEG EEG CURRENTSET] = pop\_newset(ALLEEG, EEG, CURRENTSET,'setname',EEG.setname,'gui','off');  
 eeglab redraw  
 if exist([filedir '/' strrep(deblank(filename(x,:)),'.edf','.txt')],'file')  
 ev=trans4event([filedir '/' strrep(deblank(filename(x,:)),'.edf','.txt')]);   
 if isempty(EEG.event)  
 EEG.event(1).latency=0;  
 EEG.event(1).type=1;  
 EEG.event(1).duration=0;  
 end  
 EEG = pop\_importevent( EEG, 'append','no','event',ev,'fields',{'type' 'latency' 'duration'},'timeunit',1,'align',0,'optimalign','off');  
 EEG = eeg\_checkset( EEG );  
 for x=1:length(EEG.event)  
 EEG.event(x).latency=round((EEG.event(x).init\_time-1)\*2048);  
 end  
 eeglab redraw  
 end  
   
 chanlocs=permute(struct2cell(EEG.chanlocs),[2 1 3]);  
 EEG = pop\_select( EEG,'channel',find(strncmp(chanlocs(:,1),'s',1)));  
 eeglab redraw  
 EEG = pop\_saveset(EEG, 'filename',[EEG.setname '.set'],'filepath', prep\_dir);  
end  
%% ---------------------------------------------------

##### 2. 讀進set檔找出給電的時間, notch filter, segment seeg into [-0.5 0.99]

• prep\_CCEP\_v6.m檔中第二段: 讀進set檔, 根據event分段處理, 找出每段給電的二個電極的差異, 再找差異值的正負波峰, 最後以給電的頻率校正, 新增event後另存成\*\_mrked.set檔, **需檢查**

• notch filter at 60 Hz, segment seeg into [-0.5 0.99], 另存成\*\_EPCH.set檔

%% filename: prep\_CCEP\_v6.m--------------------------------  
set\_files=dir([prep\_dir '/\*.set']);  
set\_files = char({set\_files.name}.');  
for n=1:size(set\_files,1)  
 eeglab  
 EEG=pop\_loadset('filename',[ deblank(set\_files(n,:))],'filepath',prep\_dir);  
 eeglab redraw  
   
 %% process the CCEP data with multiple sess  
 ev=[];clear charge\*  
 charge\_ch=cell(1,2);  
 charge\_name\_idx=[];  
 F=[];  
 PD=[];  
 TD=[];  
 for x=1:length(EEG.event)  
 ev=strsplit(EEG.event(x).type,'(');  
 charge\_ch(x,:)=strtrim(strsplit(strrep(ev{1}(regexp(ev{1},'from'):end),'from',''),'to'));  
 charge\_name\_idx(x,1)=x;  
 ev{2}=cellfun(@(x) strrep(strrep(x(~isstrprop(x,'alpha')),'=',''),')',''),strsplit(ev{2},','),'un',0);  
 charge\_F(x,1)=str2num(ev{2}{1});  
 charge\_PD(x,1)=str2num(strrep(ev{2}{2},'?',''));  
 charge\_TD(x,1)=str2num(ev{2}{3});  
 charge\_I(x,:)=strtrim(ev{2}{4});  
 end  
  
 chanlocs=permute(struct2cell(EEG.chanlocs),[2 1 3]);  
 [SEEG\_probe,~,~]=unique(cellfun(@(x) x(~isstrprop(x,'digit')),chanlocs(strncmp(chanlocs(:,1),'s',1),1),'un',0));  
 SEEG\_ch\_idx=strncmp(chanlocs(:,1),'s',1);  
 SEEG\_ch=chanlocs(strncmp(chanlocs(:,1),'s',1),1);  
  
 ch\_used=cellfun(@(x) find(ismember(chanlocs(:,1),x)),charge\_ch);  
  
 %% For each section identify events of charges, edit by syh  
 cd /Users/nissen/Documents/CCEP/script  
 ev\_onsets=[];  
 ev\_ch=[];  
 ev\_chid=[];  
 % ev\_chset=[];  
 for idxx=1:length(EEG.event)  
 %% get event range  
 if idxx<length(EEG.event)  
 ev\_rng=round(EEG.event(idxx).latency:EEG.event(idxx+1).latency);  
 else  
 ev\_rng=round(EEG.event(idxx).latency):EEG.pnts;  
 end  
 if ev\_rng(1)==0  
 ev\_rng(1)=ev\_rng(1)+1;  
 end  
 % plot(SEEG.data(chids(idxx,:),ev\_rng)'-10000\*idxx);  
 %% get 2 charged channels's differential  
 DATA\_A=double(diff(EEG.data(ch\_used(idxx,:),ev\_rng)'));  
% DATA\_A=EEG.data(ch\_used(idxx,:),ev\_rng)'; % modify by yt 20200325  
 temp=sort(DATA\_A);  
   
 %% find peaks in the two charged channles  
 [pkvA,pklA]=findpeaks(DATA\_A(:,1),'Threshold',std(DATA\_A(:,1))\*1,'MinPeakDistance',0.005\*EEG.srate);  
 [pkvB,pklB]=findpeaks(-DATA\_A(:,2),'Threshold',std(DATA\_A(:,2))\*1,'MinPeakDistance',0.005\*EEG.srate);  
 %% find intersecting peaks  
 mk\_flt=[];  
 plp=pklA;  
 for pp=1:length(pklA)  
 if ~any(ismember([round(-0.02\*EEG.srate):round(0.02\*EEG.srate)]+pklA(pp),pklB))  
 mk\_flt=[mk\_flt pp];  
 end  
 end  
 plp(mk\_flt)=[];  
 plp=deal\_wth\_pks(plp,EEG.srate,charge\_F(idxx));  
 if find(diff(plp)>EEG.srate\*1.03)  
 plp=plp(1:find(diff(plp)>EEG.srate\*1.03));  
 end  
 %% find vallies in the two charged channles  
 [pkvA,pklA]=findpeaks(-DATA\_A(:,1),'Threshold',std(DATA\_A(:,1))\*1,'MinPeakDistance',0.005\*EEG.srate);  
 [pkvB,pklB]=findpeaks(DATA\_A(:,2),'Threshold',std(DATA\_A(:,2))\*1,'MinPeakDistance',0.005\*EEG.srate);  
 %% find intersecting vallies  
 mk\_flt=[];  
 pln=pklA;  
 for pp=1:length(pklA)  
 if ~any(ismember([round(-0.02\*EEG.srate):round(0.02\*EEG.srate)]+pklA(pp),pklB))  
 mk\_flt=[mk\_flt pp];  
 end  
 end  
 pln(mk\_flt)=[];  
 pln=deal\_wth\_pks(pln,EEG.srate,charge\_F(idxx));  
 if find(diff(pln)>EEG.srate\*1.03)  
 pln=pln(1:find(diff(pln)>EEG.srate\*1.03));  
 end  
 %% Decide whether peaks or vallies to be used  
 if length(pln)>length(plp)  
 plbase=pln;  
 else  
 plbase=plp;  
 end  
 stpl=sort([pln;plp]);  
  
 pl=round([0:length(plbase)-1]\*EEG.srate/charge\_F(idxx))'+stpl(1);  
  
 %% Calibrate the peaks to the initiate point of charge onset 20190807 syh  
 pl=calibrate\_pks(pl,DATA\_A,EEG.srate);  
% %  
 figure;  
 plot(DATA\_A);hold on;  
 plot(pl,DATA\_A(pl,1),'b+','linewidth',2);hold off;  
 length(pl);  
 ev\_onsets=[ev\_onsets;EEG.event(idxx).latency+pl-1];  
 ev\_ch=[ev\_ch;repmat({[charge\_ch{idxx,1},'-',charge\_ch{idxx,2} '-' charge\_I(idxx,:)]},[length(pl) 1])];  
 % ev\_chset=[ev\_chset;repmat(ch\_name\_used(idxx,:),[length(pl) 1])];  
 ev\_chid=[ev\_chid;repmat(ch\_used(idxx,:),[length(pl) 1])];  
 end  
  
 %% save the charge locations in eeglab format  
 cd(anapath)  
 n\_event=length(EEG.event);  
 for mm=1:length(ev\_onsets)  
 EEG.event(n\_event+mm).latency=ev\_onsets(mm);  
 EEG.event(n\_event+mm).duration=1;  
  
 EEG.event(n\_event+mm).label=ev\_ch{mm};  
 EEG.event(n\_event+mm).type=ev\_ch{mm};  
 EEG.event(n\_event+mm).code='Stimulus';  
 EEG.event(n\_event+mm).urevent=n\_event+mm;  
 EEG.event(n\_event+mm).channel=ev\_chid(mm,1);  
 end  
 EEG.urevent=EEG.event;  
 % Save files  
 EEG=eeg\_checkset(EEG);  
 EEG = pop\_saveset( EEG, 'filename',[EEG.setname '\_mrked.set'],'filepath',prep\_dir);  
  
 %% Print event counts  
 [s,IA,IC]=unique(ev\_ch,'stable');  
 [N,~] = hist(IC,unique(IC));  
 fidA=fopen(fullfile(prep\_dir,['EvNbr\_' EEG.setname '.csv']),'w');  
 cellfun(@(x,y,z) fprintf(fidA,'%d, %s, %d\n',x,y,z),mat2cell(unique(IC),ones(length(unique(IC)),1),ones(1)),...  
 s,...  
 mat2cell(N',ones(length(N),1),ones(1)),'UniformOutput',0);  
 fclose(fidA);  
  
 fidA=fopen(fullfile(prep\_dir,['ERPLAB\_eventlist\_' EEG.setname '.txt']),'w');  
 cellfun(@(o,p,q,r) fprintf(fidA,'%d \t "%s" \t %d \t "%s" \n',o,p,q,r),...  
 mat2cell(unique(IC),ones(length(unique(IC)),1),ones(1)),...  
 s,...%unique(codelabel,'stable'),...  
 mat2cell(unique(IC),ones(length(unique(IC)),1),ones(1)),...  
 s,'UniformOutput',0);  
 fclose(fidA);  
end  
  
set\_files=dir([prep\_dir '/\*\_mrked.set']);  
set\_files = char({set\_files.name}.');  
for n=1:size(set\_files,1)  
 eeglab  
 EEG=pop\_loadset('filename',[ deblank(set\_files(n,:))],'filepath',prep\_dir);  
 eeglab redraw  
 %% notch filter  
 EEG = pop\_basicfilter( EEG, 1:EEG.nbchan , 'Cutoff', 60, 'Design', 'notch', 'Filter', 'PMnotch', 'Order', 180 );  
 EEG.setname=[EEG.setname '\_notch'];  
 eeglab redraw  
 EEG = pop\_saveset( EEG, 'filename',[EEG.setname '.set'],'filepath',pwd);  
   
 %% epoch  
 EEG = pop\_editeventlist( EEG , 'AlphanumericCleaning', 'off', 'BoundaryNumeric', { -99}, ...  
 'BoundaryString', { 'boundary' }, 'List', [prep\_dir '/ERPLAB\_eventlist\_' strrep(strrep(EEG.setname,'\_mrked',''),'\_filt','') '.txt'], ...  
 'SendEL2', 'EEG', 'UpdateEEG', 'code', 'Warning', 'off');  
  
 EEG = eeg\_checkset( EEG );  
 EPCH = pop\_epochbin( EEG , [-500.0 990.0], [ -500 -50]);

EPCH.setname=[EEG.setname '\_EPCH'];  
 binlabel={EPCH.event.binlabel}.';  
 if find(strcmp(binlabel,'""'))  
 er\_binlabel=find(strcmp(binlabel,'""'));  
 EPCH.event(find(strcmp(binlabel,'""')))=[];  
   
 er\_idx=find(cellfun(@(x) length(x),{EEG.epoch.eventcodelabel}.')==2);  
 for x=1:length(er\_idx)  
 EEG.epoch(er\_idx(x)).event=EEG.epoch(er\_idx(x)).event(~strncmp(EEG.epoch(er\_idx(x)).eventcodelabel,'Stimu',5));  
 EEG.epoch(er\_idx(x)).eventbepoch=EEG.epoch(er\_idx(x)).eventbepoch{~strncmp(EEG.epoch(er\_idx(x)).eventcodelabel,'Stimu',5)};  
 EEG.epoch(er\_idx(x)).eventchannel=EEG.epoch(er\_idx(x)).eventchannel(~strncmp(EEG.epoch(er\_idx(x)).eventcodelabel,'Stimu',5));  
 EEG.epoch(er\_idx(x)).eventduration=EEG.epoch(er\_idx(x)).eventduration(~strncmp(EEG.epoch(er\_idx(x)).eventcodelabel,'Stimu',5));  
 EEG.epoch(er\_idx(x)).eventenable=EEG.epoch(er\_idx(x)).eventenable(~strncmp(EEG.epoch(er\_idx(x)).eventcodelabel,'Stimu',5));  
 EEG.epoch(er\_idx(x)).eventflag=EEG.epoch(er\_idx(x)).eventflag(~strncmp(EEG.epoch(er\_idx(x)).eventcodelabel,'Stimu',5));  
 EEG.epoch(er\_idx(x)).eventinit\_index=EEG.epoch(er\_idx(x)).eventinit\_index(~strncmp(EEG.epoch(er\_idx(x)).eventcodelabel,'Stimu',5));  
 EEG.epoch(er\_idx(x)).eventinit\_time=EEG.epoch(er\_idx(x)).eventinit\_time(~strncmp(EEG.epoch(er\_idx(x)).eventcodelabel,'Stimu',5));  
 EEG.epoch(er\_idx(x)).eventitem=EEG.epoch(er\_idx(x)).eventitem(~strncmp(EEG.epoch(er\_idx(x)).eventcodelabel,'Stimu',5));  
 EEG.epoch(er\_idx(x)).eventlabel=EEG.epoch(er\_idx(x)).eventlabel(~strncmp(EEG.epoch(er\_idx(x)).eventcodelabel,'Stimu',5));  
 EEG.epoch(er\_idx(x)).eventlatency=EEG.epoch(er\_idx(x)).eventlatency(~strncmp(EEG.epoch(er\_idx(x)).eventcodelabel,'Stimu',5));  
 EEG.epoch(er\_idx(x)).eventcodelabel=EEG.epoch(er\_idx(x)).eventcodelabel(~strncmp(EEG.epoch(er\_idx(x)).eventcodelabel,'Stimu',5));  
 end  
 end  
 [EPCH.event.urevent] = EPCH.event.epoch;   
 EPCH = eeg\_checkset( EPCH );  
 EPCH = pop\_saveset(EPCH, 'filename',[EPCH.setname '.set'],'filepath', prep\_dir);  
end  
%% ---------------------------------------------------

##### 3. 合併所有的EPCH檔

合併後檢查epoch, 刪掉bad channel及epoch

%% filename: prep\_CCEP\_v6.m--------------------------------  
%% combine all epoch files  
cd(anapath)  
[~,subno]=fileparts(anapath);  
eeglab  
EPCH\_files=dir([prep\_dir '/\*\_EPCH.set']);  
EPCH\_files=char({EPCH\_files.name}.');  
EPCH\_files=deblank(EPCH\_files);  
  
if size(EPCH\_files,1)>1  
 EEG = pop\_loadset('filename',deblank(EPCH\_files(1,:)),'filepath',prep\_dir);  
 [ALLEEG, EEG, CURRENTSET] = eeg\_store( ALLEEG, EEG, 0 );  
 chanlocs=permute(struct2cell(EEG.chanlocs),[2,1,3]);  
 SEEG\_CH=find(strncmp(chanlocs(:,1),'s',1));  
 EEG = pop\_select( EEG,'channel',SEEG\_CH);  
 eeglab redraw  
  
 for x=2:size(EPCH\_files,1)  
 EEG = pop\_loadset('filename',deblank(EPCH\_files(x,:)),'filepath',prep\_dir);  
 [ALLEEG, EEG, CURRENTSET] = eeg\_store( ALLEEG, EEG, 0 );  
 chanlocs=permute(struct2cell(EEG.chanlocs),[2,1,3]);  
 SEEG\_CH=find(strncmp(chanlocs(:,1),'s',1));  
 EEG = pop\_select( EEG,'channel',SEEG\_CH);  
 eeglab redraw  
 [ALLEEG EEG CURRENTSET] = pop\_newset(ALLEEG, EEG, CURRENTSET,'retrieve',CURRENTSET-1,'study',0);  
 EEG.data=cat(3,ALLEEG(1).data,ALLEEG(2).data);  
 EEG.trials=ALLEEG(1).trials+ALLEEG(2).trials;  
 EEG.EVENTLIST.eventinfo=[ALLEEG(1).EVENTLIST.eventinfo ALLEEG(2).EVENTLIST.eventinfo];  
 EEG.epoch=[ALLEEG(1).epoch ALLEEG(2).epoch];  
 EEG.event=[ALLEEG(1).event ALLEEG(2).event];  
 eeglab redraw  
 ALLEEG = pop\_delset( ALLEEG, [2] );  
 end  
 EEG.setname=[subno '\_CCEP\_EPCH'];  
 for x=1:length(EEG.event)  
 EEG.event(x).latency=EEG.event(1).latency+EEG.pnts\*(x-1);  
 EEG.event(x).epoch=x;  
 end  
 [~,index] = sortrows([EEG.event.latency].'); EEG.event = EEG.event(index); clear index  
 eeglab redraw  
 EEG = pop\_saveset(EEG, 'filename',[EEG.setname '.set'],'filepath', prep\_dir);  
else  
 EEG = pop\_loadset('filename',EPCH\_files(1,:),'filepath',prep\_dir);  
 EEG.setname=[subno '\_CCEP\_EPCH'];  
 eeglab redraw  
 EEG = pop\_saveset(EEG, 'filename',[EEG.setname '.set'],'filepath', prep\_dir);  
end  
%% ---------------------------------------------------

##### 4. 輸出subno eventlist, 表格, epoch並存erp file

表格包含電極對及電極次數

%% filename: prep\_CCEP\_v6.m--------------------------------

%% after combine all epoch files  
codelabel = {EEG.event.codelabel}.';  
[s,~,IC]=unique(codelabel,'stable');  
[N,~] = hist(IC,unique(IC));  
  
EEG.EVENTLIST.nbin=length(s);  
for x=1:EEG.EVENTLIST.nbin  
 EEG.EVENTLIST.bdf(x).description=s{x};  
 EEG.EVENTLIST.bdf(x).namebin=['Bin' num2str(x)];  
end  
EEG.EVENTLIST.trialsperbin=N;  
eeglab redraw  
  
fidA=fopen(fullfile(prep\_dir,['ERPLAB\_eventlist\_' EEG.setname '.txt']),'w');  
cellfun(@(o,p,q,r) fprintf(fidA,'%d \t "%s" \t %d \t "%s" \n',o,p,q,r),...  
 mat2cell((1:length(unique(codelabel,'stable')))',ones(length(unique(codelabel)),1),ones(1)),...  
 unique(codelabel,'stable'),...%unique(codelabel,'stable'),...  
 mat2cell((1:length(unique(codelabel,'stable')))',ones(length(unique(codelabel)),1),ones(1)),...  
 unique(codelabel,'stable'),'UniformOutput',0);  
fclose(fidA);  
  
  
[s,IA,IC]=unique(codelabel,'stable');  
[N,~] = hist(IC,unique(IC));  
fidA=fopen(fullfile(prep\_dir,['EvNbr\_' EEG.setname '.csv']),'w');  
cellfun(@(x,y,z) fprintf(fidA,'%d, %s, %d\n',x,y,z),...  
 mat2cell((1:length(unique(codelabel,'stable')))',ones(length(unique(codelabel)),1),ones(1)),...  
 unique(codelabel,'stable'),...  
 mat2cell(N',ones(length(N),1),ones(1)),'UniformOutput',0);  
fclose(fidA);  
  
s(:,2)=mat2cell((1:length(s))',ones(length(s),1),ones(1));  
codelabel(:,2)=cellfun(@(x) find(strcmp(x,s(:,1))),codelabel,'un',0);  
codelabel(:,3)=cellfun(@(x) num2str(x,['%0' num2str(floor(log10(length(s)))+1) 'd']),codelabel(:,2),'un',0);  
code = [EEG.EVENTLIST.eventinfo.code].';  
idx=find(~isnan(code));  
for x=1:length(EEG.event)  
 EEG.event(x).bini=codelabel{x,2};  
 EEG.epoch(x).eventbini=codelabel{x,2};  
 EEG.EVENTLIST.eventinfo(idx(x)).code=codelabel{x,2};  
 EEG.event(x).binlabel=strrep(EEG.event(x).binlabel,EEG.event(x).binlabel(1:regexp(EEG.event(x).binlabel,'(')-1),['B',codelabel{x,3}]);  
 EEG.event(x).type=EEG.event(x).binlabel;  
 EEG.epoch(x).eventbinlabel=EEG.event(x).binlabel;  
 EEG.epoch(x).eventtype=EEG.event(x).binlabel;  
 EEG.EVENTLIST.eventinfo(idx(x)).binlabel=EEG.event(x).binlabel;  
end  
  
eeglab redraw  
EEG = pop\_saveset(EEG, 'filename',[EEG.setname '.set'],'filepath', prep\_dir);  
  
bini = [EEG.event.bini].';  
for ii=1:EEG.EVENTLIST.nbin  
 idx=find(bini==ii);  
 for x=1:length(idx)  
 epoch=EEG.data(:,:,idx(x));  
 save(fullfile(epoch\_dir,sprintf('%s\_epoch%04d.mat',EEG.event(idx(x)).codelabel,x)),'epoch');  
 end  
end  
  
f=fopen([epoch\_dir '/seeg\_channels.txt'],'w');  
fprintf(f,'%s\t%s\n','ch.no','ch. name');  
for x=1:EEG.nbchan  
 fprintf(f,'%s\t%s\n',num2str(x),EEG.chanlocs(x).labels);  
end  
fclose(f);  
  
% write seeg\_epoch\_timepoints.txt  
f=fopen([epoch\_dir '/seeg\_epoch\_timepoints.txt'],'w');  
fprintf(f,'%s\t%s\n','time point','time (ms)');  
for x=1:length(EEG.times)  
 fprintf(f,'%s\t%s\n',num2str(x),num2str(EEG.times(x)));  
end  
fclose(f);  
  
% save erp file  
ERP = pop\_averager( EEG , 'Criterion', 'all', 'ExcludeBoundary', 'on', 'SEM', 'on' );  
erplab redraw  
ERP = pop\_savemyerp(ERP, 'erpname', EEG.setname , 'filename', ...  
 [EEG.setname '.erp'], 'filepath', erp\_dir);  
%% ---------------------------------------------------