```
%PREDREEGEXPORT2 Prepares and exports selected EEGs (saved as *_3F.mat in
% the workspace) from local TUEG database dump to usable EEG files. The
% files are all preprocessed (cleaned, segmented, filtered, reordered,
% resampled, etc.)
%
%   Authors:
%       Michael Caiola (Michael.Caiola@fda.hhs.gov)
%       Meijun Ye (Meijun.Ye@fda.hhs.gov)
%
%   Disclaimer: This software and documentation (the "Software") were
%   developed at the Food and Drug Administration (FDA) by employees of
%   the Federal Government in the course of their official duties.
%   Pursuant to Title 17, Section 105 of the United States Code,
%   this work is not subject to copyright protection and is in the
%   public domain. Permission is hereby granted, free of charge, to any
%   person obtaining a copy of the Software, to deal in the Software
%   without restriction, including without limitation the rights to
%   use, copy, modify, merge, publish, distribute, sublicense, or sell
%   copies of the Software or derivatives, and to permit persons to
%   whom the Software is furnished to do so. FDA assumes no
%   responsibility whatsoever for use by other parties of the Software,
%   its source code, documentation or compiled executables, and makes
%   no guarantees, expressed or implied, about its quality,
%   reliability, or any other characteristic. Further, use of this code
%   in no way implies endorsement by the FDA or confers any advantage
%   in regulatory decisions. Although this software can be
%   redistributed and/or modified freely, we ask that any derivative
%   works bear some notice that they are derived from it, and any
%   modified versions bear some notice that they have been modified.
%   The Software is not intended to make clinical diagnoses or to be
%   used in any way to diagnose or treat subjects for whom the EEG is
%   taken.

path = pwd;
cd ../../eeglab %change for eeglab location
eeglab
close
cd(path)
savefolder="D:\3minSept22"; % replace with output folder
N=3;

d=dir('*_3F.mat'); % selected files
for k=1:length(d)
    F=load(d(k).name);
    pre=d(k).name(1:3);
    n=PreDREEGd(F.(d(k).name(1:6)),.5);
    T=table(n(:,1),str2double(n(:,2)),'VariableNames',{'files','n'});
    names=sortrows(T);
    names.files = strrep(names.files,"D:\","F:\");
    [sub,subidx]=unique(names.files,"stable");
    subidx(end+1)=height(names)+1;
```

```
    i_start=1;
    for i=i_start:length(sub)
        dd=dir(fullfile(sub(i),'*.edf'));
        fnames="";
        for j=1:length(dd)
            fnames(j)=string(fullfile(dd(j).folder,dd(j).name));
        end
        %%
        [rawEEG,join_flag]=joinEEG(fnames);
        if isempty(rawEEG.data)
            %clear names
            warning(['Bad EDF: ',d(j).folder])
            continue
        end
        data=cutEEG(rawEEG,N);
        if isempty(data)
            %clear names
            warning(['Bad EDF: ',d(j).folder])
            continue
        end
        minNums = (names.n(subidx(i):subidx(i+1)-1))-join_flag;
        data=data(:,:,minNums);
        try
            clndata=filtEEG(rawEEG,data);
        catch
            warning(['Dirty EDF: ',d(j).folder])
            continue
        end
        s=split(sub(i),'\');
        for kk=1:size(clndata,3)
            y=clndata(:,:,kk);
            if or(isempty(y),y==zeros(size(y)))
                continue
            end
            try
                save(fullfile(savefolder,pre,s(end-1),s(end)+"-"+string(minNums(kk))+".↙
mat"),'y')
            catch
                mkdir(fullfile(savefolder,pre,s(end-1)))
                save(fullfile(savefolder,pre,s(end-1),s(end)+"-"+string(minNums(kk))+".↙
mat"),'y')
            end
        end
    end
end

%% Functions
function [rawEEG,out]=joinEEG(names)
out=0;
cnt = 0;
%if verLessThan('matlab','9.9')
```

```
try
    [rawEEG,cnt1] = FindCh(names{1},1);
    cnt = cnt+cnt1;
    %     if cnt1
    %         disp("Fixed " + cnt + " records!")
    %     end
    param_flag = 0;
catch
    rawEEG.comments=[];
    rawEEG.data=[];
    param_flag=1;
end
%else
%end
for k=2:length(names)
    try
        [newEEG,cnt1] = FindCh(names{k},0);
        cnt = cnt+cnt1;
        %         if cnt1
        %             disp("Fixed " + cnt + " records!")
        %         end
        if param_flag
            rawEEG=newEEG;
            out=1;
            param_flag=0;
        else
            rawEEG.comments=[rawEEG.comments;newEEG.comments];
            rawEEG.data=[rawEEG.data,newEEG.data];
        end
    catch
    end
end
if param_flag
    rawEGG=[];
else
    rawEEG.data=rawEEG.data(:,1:250*60*(floor(length(rawEEG.data)/250/60)-1));
    rawEEG.xmax=length(rawEEG.data)/250;
    rawEEG.pnts=length(rawEEG.data);
    rawEEG.times=0:4:4*(rawEEG.pnts-1);
end
end
function data=cutEEG(EEG,n)
cuts=floor(EEG.xmax/60/n);
data=zeros(19,n*60*250,cuts);
for i=1:cuts
    data(:,:,i)=EEG.data(:,1+(i-1)*n*60*250:i*n*60*250);
end
end
function [clndata]=filtEEG(EEG,data)
for i=1:size(data,3)
    rawEEG=EEG;
```

```
    rawEEG.data=data(:,:,i);
    rawEEG.pnts=length(rawEEG.data);
    filtEEG = pop_eegfiltnew(rawEEG,1,100);
    EEG = pop_reref(filtEEG,[]);
    [~,W] = fastica(EEG.data,'verbose','off');
    EEG = pop_editset(EEG,'icaweights',W);
    EEG = iclabel(EEG);
    [~,ictype] = max(EEG.etc.ic_classification.ICLabel.classifications,[],2);
    icreject = find(ictype~=1);
    % EEG.reject.gcompreject(1,icreject') = 1;
    if ~(length(icreject) == size(EEG.icaweights,1))
        clean_EEG = pop_subcomp(EEG,icreject',0,0);

        %rawdata(:,:,i) = EEG.data;
        clndata(:,:,i) = clean_EEG.data;
    end
end
end
function [rawEEG,cnt] = FindCh(file,first)
cnt = 0;
load('chlocs2.mat','channel_locations');
ch_locs = struct2table(channel_locations);
ch_locs = string(ch_locs.labels);
if first
    rawEEG = pop_biosig(file,'blockrange',[60↙
Inf],'importevent','off','importannot','off');
else
    rawEEG = pop_biosig(file,'importevent','off','importannot','off');
end
chs = struct2table(rawEEG.chanlocs);
chs = string(chs.labels);
chs = erase(chs,"-REF"|"-LE");
in = [];
for i = 1:length(ch_locs)
    in1 = find(strcmpi(chs,ch_locs(i)));
    in = [in in1];
    if and(isempty(in1),ismember(ch_locs(i),["T3";"T4";"T5";"T6"]))
        switch ch_locs(i)
            case "T3"
                in2 = find(strcmpi(chs,"T7"));
            case "T4"
                in2 = find(strcmpi(chs,"T8"));
            case "T5"
                in2 = find(strcmpi(chs,"P7"));
            case "T6"
                in2 = find(strcmpi(chs,"P8"));
        end
        in = [in in2];
    end
end
if length(in)~=19
```

```
    error("Could not locate all channels!");
end
rawEEG.data = rawEEG.data(in,:);
rawEEG.chanlocs = channel_locations;
rawEEG.nbchan = 19;
if ~isequal(in,[1:16,19:21])
    cnt = 1;
    if ~isequal(in,1:19)
        cnt = 2;
    end
end
end
```