ReSync Toolbox Manual 1.0

Guang Ouyang, 2020 Feb

I. General

ReSync is an EEG toolbox for correcting ERP waveform that is attenuated due to trial-to-trial latency jitter. More details about how it works can be found in ReSync paper. In brief, due to latency jitter, the ERP components in single trials is not fully synchronized with respect to stimulus onset, making the ERP waveform a 'blurred' version. ReSync identifies the latency of the latency-variable components in single trials, using General Linear Model method to decompose and resynchronize them. One of the advantage of ReSync method is that it can resynchronize different components with different degree of jitter (e.g., early and late) separately, without affecting each other. One can follow the instruction below to see how it works.

II. Installation

ReSync is an extension(plugin) toolbox under EEGLab toolbox. You can directly download ReSync from within EEGLab (File → Manage Extensions) or from the GitHub website (https://github.com/guangouyang/ReSync). If you download the zip from GitHub, upzip the folder and simply put the ReSync1.0 folder under the 'plugins' folder in EEGLab directories. Make sure that it is the folder that contains all the matlab scripts (named 'ReSync1.0') is put under 'plugins'. After this is set, launch EEGLab and you will see ReSync under Tools after you load some EEG dataset into EEGlab.

III. Resyncing ERP on a single participant using GUI

A. Load data and examine ERP

1. Launce EEGLab by typing 'eeglab' in Matlab command line.
2. File → Load existing data. Navigate to the ReSync1.0 folder and select the file 'vp26_clean' under the folder 'sample_data', note that this sample data in not included in the toolbox downloaded from EEGLAB server, but you can download the sample data from GitHub.

   This is from a face recognition task in which the participant made gender judgement from the facial pictures with different emotion expressions – happy (marker S 11), neutral (marker S 12), and angry (marker S 13). The data has been preprocessed with major artifacts removed.
3. Click Tools → ReSync. You will see the interface as shown in Figure 1.
4. Select Electrode(s) → Select Oz.
5. Select Marker(s) → Select S 11, S 12, S 13. Hold 'Shift' or 'Ctrl' to select multiple items.
6. Click 'Plot average ERP'.

   After clicking this button, you will see the single trials and average ERP averaged from the electrode(s), time locked to the markers that you just selected (Figure 2). The time range of the ERP and the baseline window are specified in the parameter as shown below. This is for you to get an impression on the single trial variability and decide which time window you want to ReSync (correct the latency jitter). We will start with correcting the late component (P3).

B. ReSync ERP and visually examine the results

7. From the average ERP waveform, we can see a clear P3-like component ranging from 400 to 800 ms. So we now change the default ReSync time window from '300 800' to '400 800', and click the button ReSync.

We are going to start correcting ERP component within 400 to 800 ms. Click the button 'ReSync' and you will get the results as shown in Figure 3. Please refer to Figure 3 caption of description of the results. As you can see, the ERP component within (400 ms, 800 ms) time window has been corrected for jitter effect (Figure 3d). But the components in other time windows are not affected.

The Matlab command window will show some parameters related to the data feature. SNR is the estimated signal to ratio signal of the ERP within the selected ReSync time window. RLV is the estimated relative latency variability (see paper for detailed explanation) of the ERP within the selected ReSync time window. And Dominant Frequency is the estimated dominant frequency of it. The ReSync Flag indicates if this ReSync is meaningful or not (1 means yes, 0 means no, see the paper for how this judgement is made).

8. Now we can try a new ReSync time window (200, 400) to see how it would look like to correct the ERP in this window. Change the original ReSync time window to '200 400' and then click ReSync, you will get Figure 4. As you can see in Figure 4d, the component in (200 ms, 400 ms) was substantially corrected without affecting others.
9. The above two steps, namely, correcting the early ERP (200 – 400ms) and late ERP (400 – 800ms) can be done iteratively in GUI. This requires overwriting the existing EEG with jitter-corrected (ReSynced) ERP in previous steps. See the next steps on how to operate.

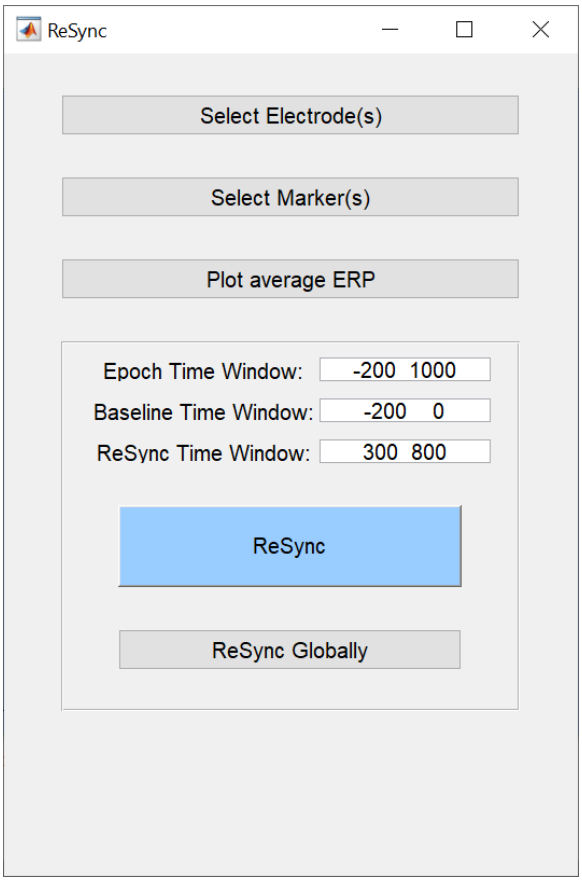

Figure 1. ReSync UI.

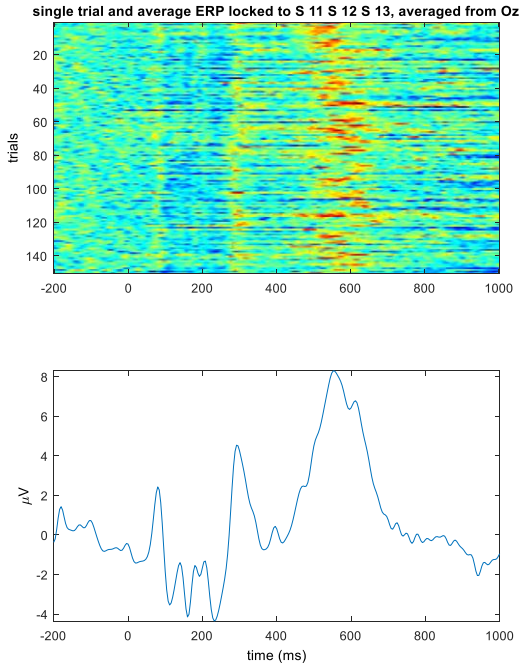

Figure 2. Plot single trial and average ERP.

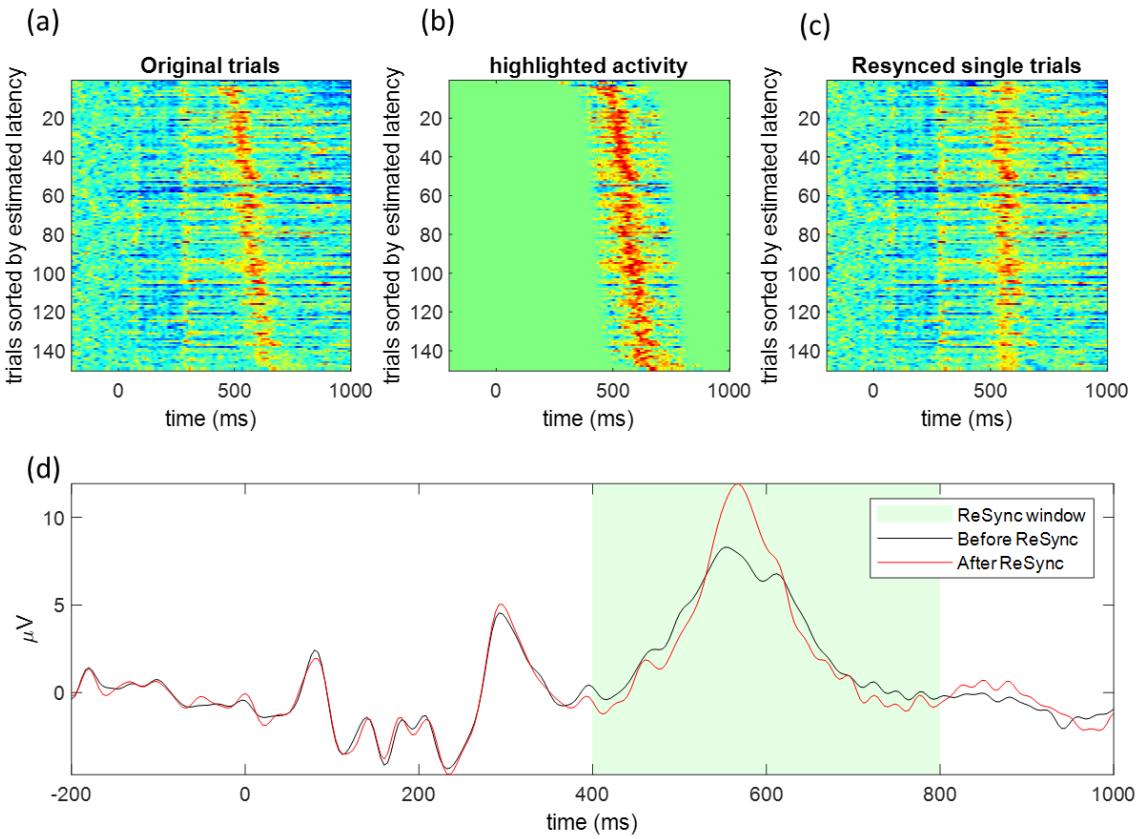

Figure 3. Visual summary of ReSync results. (a) Single trials ERP sorted by estimated latency. (b) Highlighted component activities that are with latency jitter. (c) Jitter-corrected single trials. (d) Comparison of standard average ERP and jitter-corrected ERP (specified time window is indicated by color background).

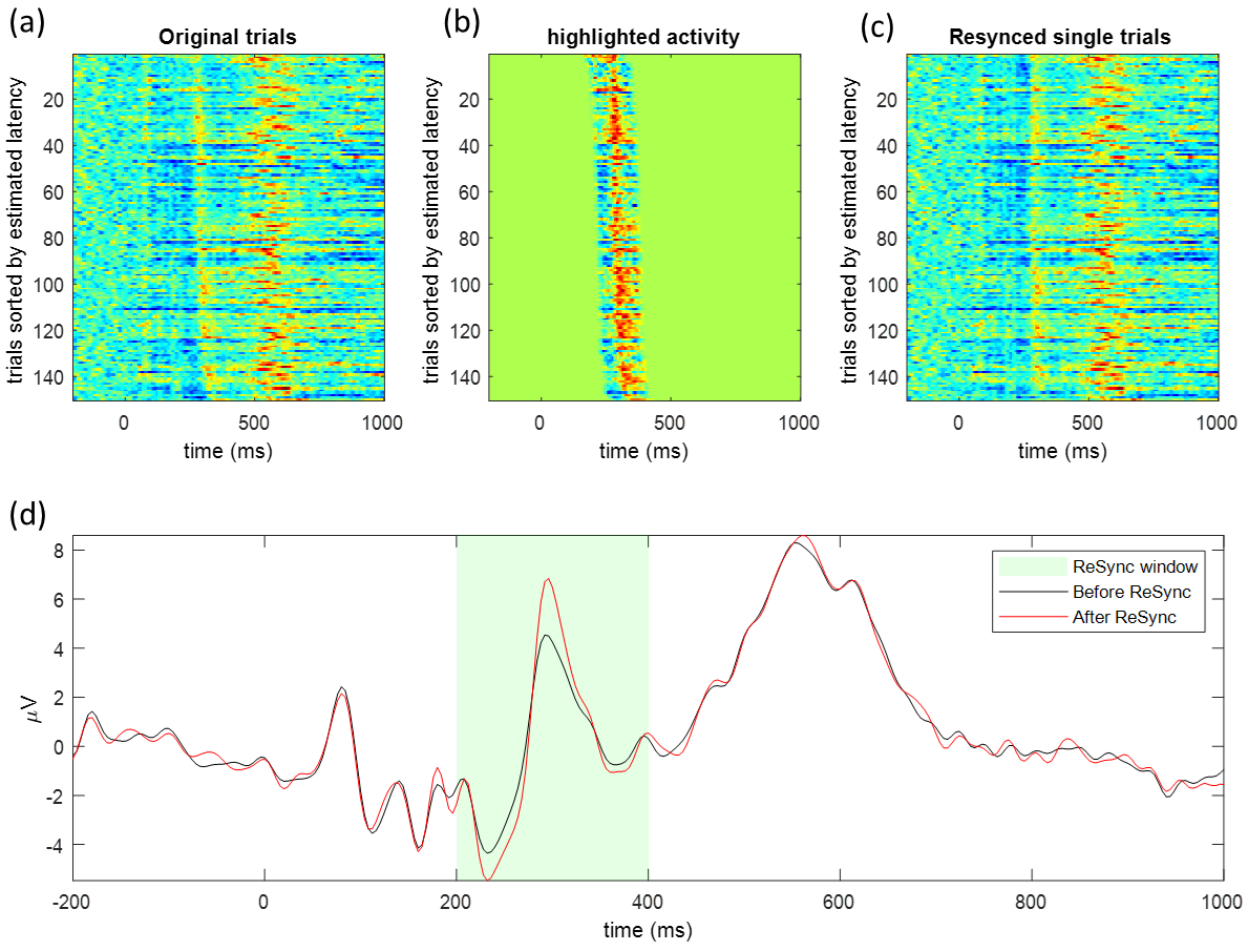

Figure 4. Same as Figure 3 but for early component correction (200 – 400 ms).

C: ReSync ERP globally (across all electrodes)

10. The function of the blue button (ReSync), i.e., steps in B, is only for visualizing the before-and-after effect of ReSyncing. It does not change anything in the EEG data structure, and it only process the data from the selected electrode(s). In reality, the latency jitter issue exists in every electrode due to the volume conduction effect. For instance, the latency jitter of P3 component distorts the P3 on electrodes Pz (where P3 is usually largest), but also its nearby electrodes (e.g., CPz, PO3, PO4). However, to best identify the latency variability, Pz should be used for latency estimation and ERP correction. And the correction should be applied on every electrode based on the latency variability estimated from Pz. If an electrode is not affected by such latency variability, the ERP waveform from this electrode would not be changed much. The button 'ReSync Globally' is for implementing such processing, i.e., it will correct the latency jitter effect on all electrodes, based on the jitter feature estimated from the prominent electrode(s). And this button will change the information in EEG data structure (it will change EEG.data) and offer

the user option to overwrite the current EEG data and generate a new one. After this operation, the EEG data have been changed (with latency jitter corrected).

11. To test the function of the button 'ReSync Globally'. We can do a two-step ReSync here. First, again select electrode Oz, marker (S 11, S 12, S 13). Set the ReSync time window to be '200, 400'. Then click 'ReSync Globally'. The data for all electrodes will be ReSynced and the progress will be shown on the Matlab command window with appearing electrode numbers. Once it is done, a window will be popped up for you to choose to overwrite the existing data and create a new set. After that, change the ReSync time window to be '400 800', and click the 'ReSync Globally' again. In this time, the data from all electrodes will again be ReSynced within the new time window and you can choose to overwrite the previous data or not. After this two steps, the ERP data from all electrodes has been corrected for jitter for both the time windows 200 – 400 ms and 400 – 800 ms, based on the jitter feature of Oz. You can examine the final effect by plotting the average ERPs from the current data set and the original data set. The effect is shown in Figure 5.

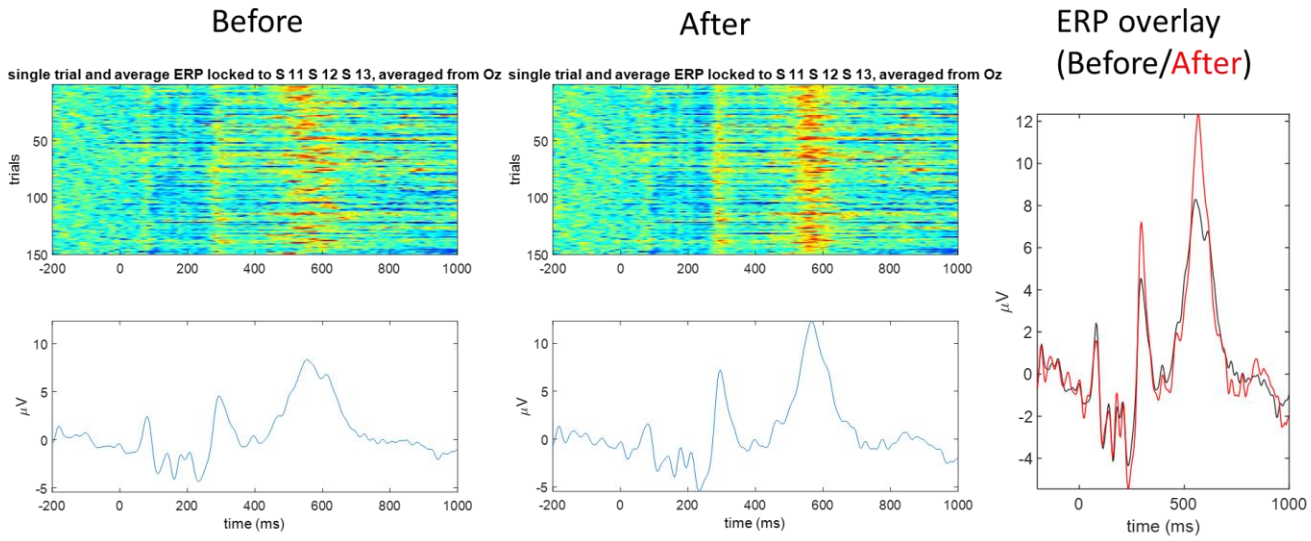

Figure 5. Correcting ERP jitter from two time windows iteratively (200 – 400ms and 400 – 800 ms).

IV. Resyncing ERP using script

1. Operating on the EEG dataset

With all relevant parameters specified, the ReSync processing can be easily turned into script for automatizing purpose. If you want to correct jitter in the single trial ERP data, all you need to specify is the 1) the marker(trigger) to which your ERP is elicited; 2) electrode(s) where the jittering component is strongest; and 3) time window (ReSync time window) where the jitter needed to be corrected. Please refer to the above material for how time window is specified. The time window need to be identified from the ERP waveform (Figure 2). For a typical study with a number of participant, the ReSync time

window can be universally determined from the grand average ERP. Individualized determination is also an option (see paper for a discussion on this issue).

Assuming all the parameters have been decided, the Matlab script goes like:

```
%load the sample data first
cfg = [];
cfg.epoch_twd = [-200,1000];%in millisecond
cfg.base_twd = [-200,0];
cfg.resync_twd = [200,400];
cfg.selected_elec = {'Oz'};
cfg.selected_marker = {'S 11','S 12','S 13'};
cfg.glb = 1;

EEG = RS_resyncERP(EEG,cfg);
```

Now, the EEG data has been ReSynced, based on the parameter specified. The variable 'data' under EEG has been changed (jitter corrected). The ReSynced related information is contained in EEG.ReSync. If you do multiple ReSycning, EEG.ReSync will contain multiple cell elements. The explanation of variables under EEG.ReSync is as follows:

cfg: configuration of parameters.
original_data: the original EEG trace fed to ReSync. This is the EEG trace averaged from the selected electrode(s).
resync_data: the ReSync processed EEG trace (jitter corrected).
original_ERP: the original ERP locked to the selected marker.
resync_ERP: ReSynced ERP.
t: the time coordinates for the ERP (with respect to stimulus onset).
est_latency: estimated jitter. Every value represent the lag of ERP component in a single trial with respect to the template, in the unit of the sampling data point.
decomp_ERPs: the decomposed ERP by GLM. This the internal information of GLM decomposition.
SNR, RLV, dominant_freq, and resync_flag were explained above.

Note that this processing has to be done after launching the EEGlab toobox and load the sample data. You can also add the ReSync toolbox directly into the Matlab path, so that you don't need to launch EEGLAB before using ReSync.

Now we only corrected the jitter with in 200-400 time window. Following the previous script, we can further correct the jitter within 400-800 (just like what we did in the previous section based on UI) by adding and running the following script.

```
cfg.resync_twd = [400,800];
EEG = RS_resyncERP(EEG,cfg);
```

Now the EEG data has been ReSynced again in 400-800 ms according to jitter in Oz. You can compare the original ERP and twice-ReSynced ERP by running the following lines. It will generate the same result as in Figure 5 (right).

```
figure;plot(EEG.ReSync{1}.t,EEG.ReSync{1}.original_ERP,'k');
hold on;plot(EEG.ReSync{2}.t,EEG.ReSync{2}.resync_ERP,'r');
```

Note that the two-step ReSyncing does not need to base on the same electrode (same as multi-step ReSyncing). You can change the selected electrode to other(s) before each ReSync step. Actually, changing selected electrode(s) should be the norm. For example, you can first correct the jitter of early ERP components based on occipital electrodes, and then correct the late component (e.g., P3) on central parietal electrode(s). In each ReSyncing step, all the electrodes will be corrected for jitter distortion (but based on the jitter feature from the selected electrode(s)).

2. Using ReSync as a general signal processing algorithm

The EEG trace from a single electrode is simply a time series. Given a time series with some known time markers on it, and assuming every marker will elicit a ERP-like activation with certain latency jitter, ReSync can be directly applied on this time series. All the information that are needed are: 1) data: an 1×p vector (time series) where p is the number of data points; 2) latencies: an 1×n vector where n is the number of markers indicting the stimulus onsets on the time series; 3) epoch time window: the time window encompassing the time marker where the average ERP (or activation waveform) is to be derived. All time windows are in unit of millisecond; 4) baseline time window; 5) ReSync time window: the time window where jitter is supposed to occur and to be corrected; 6) sampling rate: the sampling rate of the data. The following script is for directly applying ReSync on a one dimensional time series. The first part is for preparation of the parameters. We will use the EEG trace from Oz as our example. The second part is the application of ReSync on the time series.

```
%data and parameter preparation (remember to reload EEG data)
elec = {'Oz'};
data = mean(EEG.data(ismember({EEG.chanlocs.labels},elec),:),1);
marker = {'S 11','S 12','S 13'};
latencies =
round([EEG.event(ismember({EEG.event.type},marker)).latency]);
srate = EEG.srate;

%applying ReSync on a time series
cfg = [];
cfg.srate = srate;
cfg.latencies = latencies;
cfg.epoch_twd = [-200,1000];
cfg.base_twd = [-200,0];
cfg.resync_twd = [200,400];
results = ReSync(data, cfg);

%the results will be shown in a popped-up figure.
```