functional near-infrared spectroscopy (fNIRS) 数据预处理

本教程属于陈锐CR整理撰写，如果对本教程有任何疑问及建议，欢迎与我联系和沟通。

微信公众号：BrainTechnology

微信助手ID：braintechnology

邮箱：science52brain@outlook.com

希望本手册教程能提供您一些分析思路，当然它可能并不难完全直接适用于您自己的近红外数据，但大体上代码是通用的，我相信您只需要进行一些并不困难的修改之后，您就可以很简单自如地开始着手处理您的数据了。

每次写教程都会花费了我很多的时间和精力，但还是公众号宗旨的那句话——让脑科学技术更简单。因此，在一定程度上来说，它或许能对初入心理学与神经科学领域的学生老师们一些帮助。

本教程介绍如何导入功能性近红外光谱（fNIRS）原始测量数据，并转换为相对的氧血红蛋白（HbO）及脱氧血红蛋白（HbR）浓度，查看平均波形及响应的地形表示。

```
import numpy as np
import matplotlib.pyplot as plt

import mne

raw_data = mne.io.read_raw_nirx('/Users/chenrui/Desktop/Participant-1', saturated =
'annotate' , preload = False , verbose = None)
raw_data.load_data()
```

```
Loading /Users/chenrui/Desktop/Participant-1
Reading 0 ... 23238  =      0.000 ...  2974.464 secs...
```

Measurement date	**November 02, 2019 13:16:16 GMT**
Experimenter	Unknown
Participant	P1
Digitized points	31 points
Good channels	56 fNIRS (CW amplitude)
Bad channels	None
EOG channels	Not available
ECG channels	Not available
Sampling frequency	7.81 Hz
Highpass	0.00 Hz
Lowpass	3.91 Hz
Filenames	Participant-1
Duration	00:49:34 (HH:MM:SS)

提供有意义的注释信息

1、将marker标记为有意义的名称
2、包括了关于每个刺激持续时间，在本实验的所有条件下都是5秒。
3、删除触发代码15，它表示实验的开始和结束，与我们的分析无关。

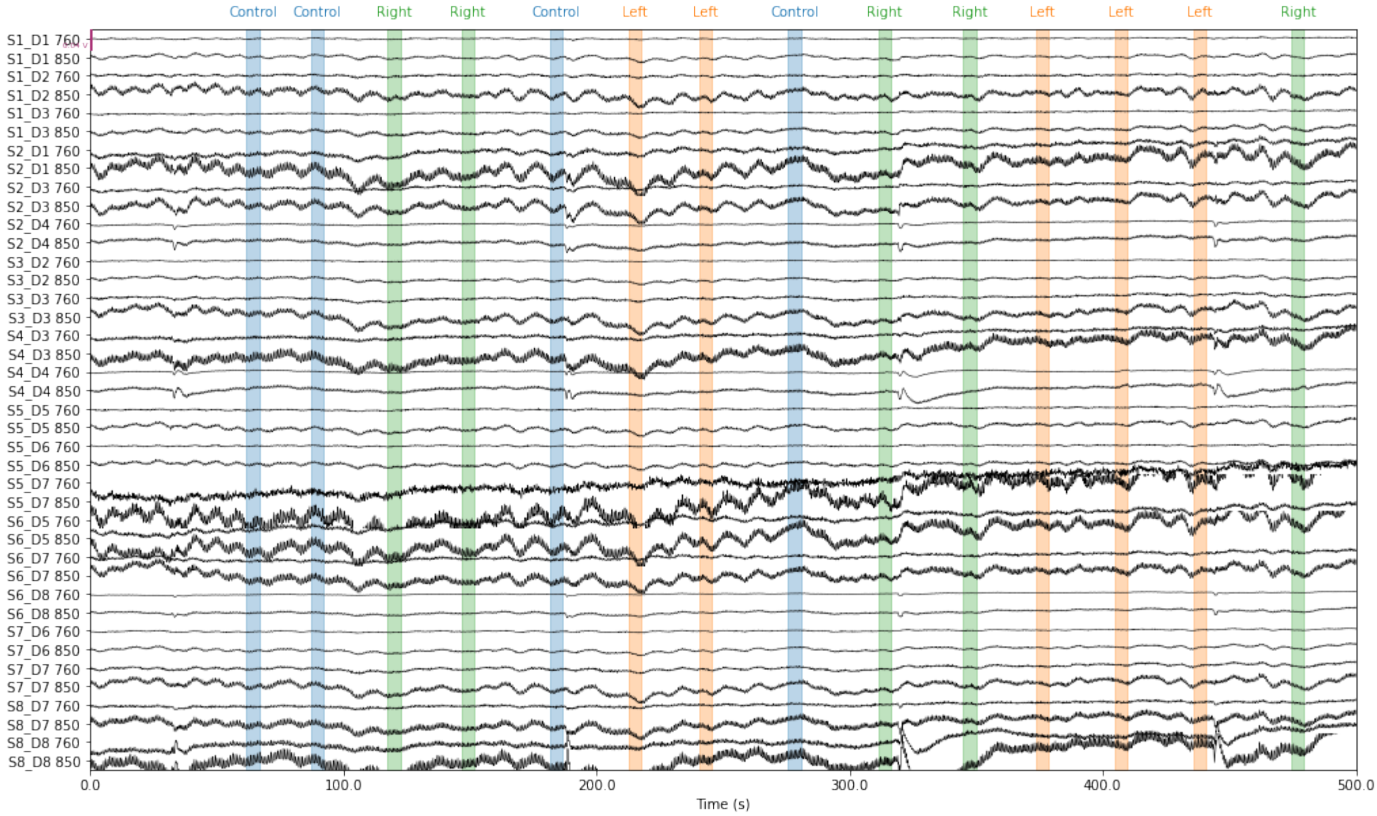
```
raw_data.annotations.set_durations(5)
raw_data.annotations.rename({'1.0': 'Control','2.0': 'Left','3.0': 'Right'})
unwanted = np.nonzero(raw_data.annotations.description == '15.0')
raw_data.annotations.delete(unwanted)
```

删除短通道

删除短通道，光源-探测器之间的距离小于1厘米。

```
picks = mne.pick_types(raw_data.info, fnirs=True)
dists = mne.preprocessing.nirs.source_detector_distances(raw_data.info, picks=picks)
raw_data.pick(picks[dists > 0.01])
raw_data.plot(n_channels=len(raw_data.ch_names), duration=500, show_scrollbars=False)
```

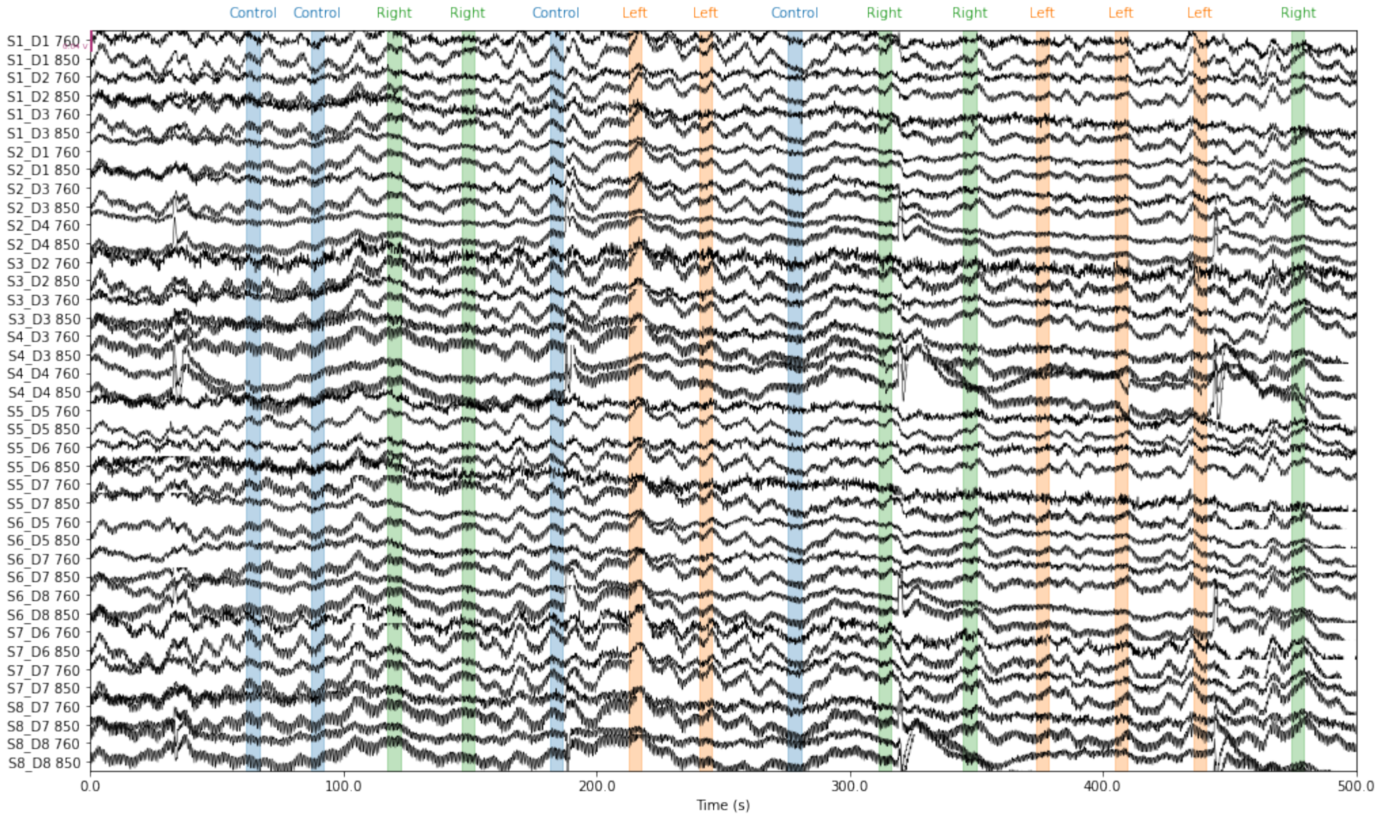
```
Using matplotlib as 2D backend.
```



从原始强度转换为光密度

将原始强度值转换为光密度。

```
raw_od = mne.preprocessing.nirs.optical_density(raw_data)
raw_od.plot(n_channels=len(raw_od.ch_names),duration=500, show_scrollbars=False)
```



评估数据质量

使用头皮耦合指数在头皮和optodes之间的方法。
在本例中，数据是干净的，耦合对所有的通道都可以。

```
sci = mne.preprocessing.nirs.scalp_coupling_index(raw_od)
fig, ax = plt.subplots()
ax.hist(sci)
ax.set(xlabel='Scalp Coupling Index', ylabel='Count', xlim=[0, 1])
```

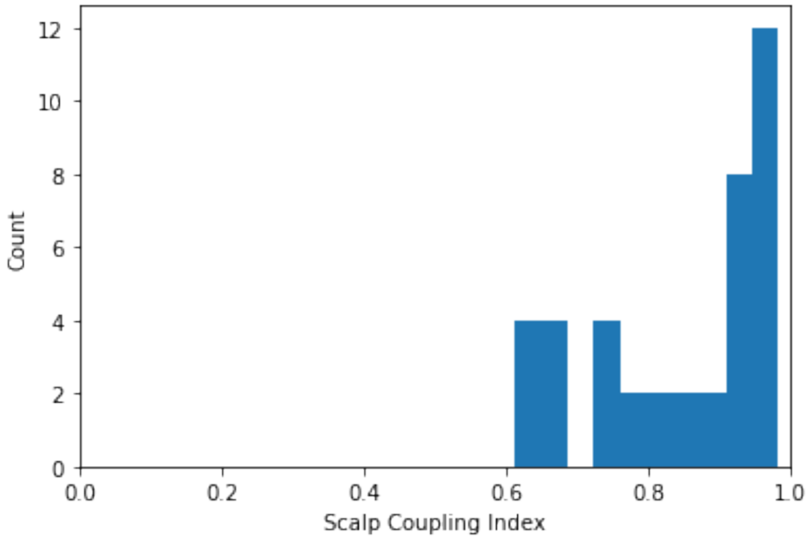
```
[Text(0.5, 0, 'Scalp Coupling Index'), Text(0, 0.5, 'Count'), (0.0, 1.0)]
```

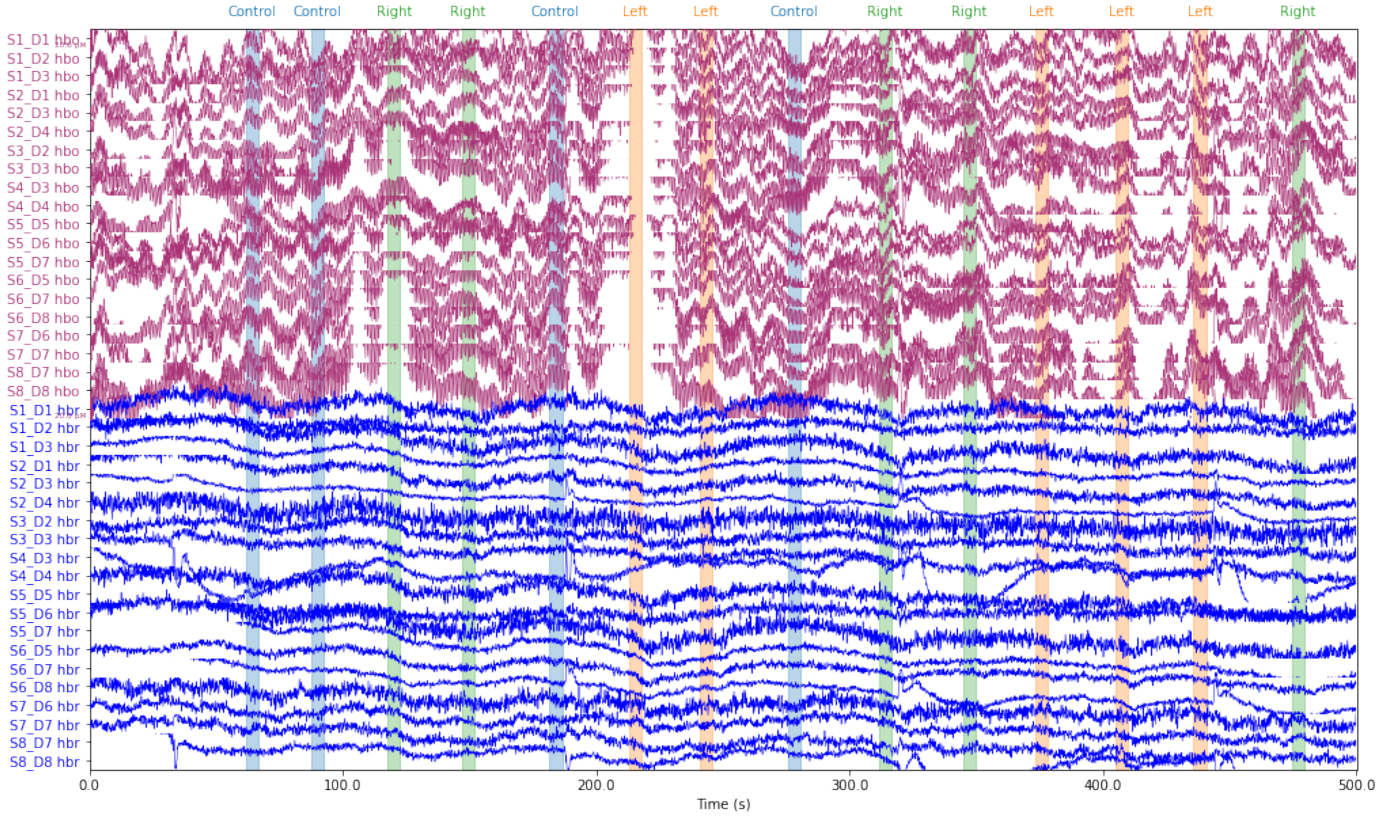
将耦合指数小于0.5的所有通道标记为“bads”（此数据集非常干净，因此没有通道被标记为坏）。

从光密度转换为血红蛋白

使用修正的比尔-朗伯定律将光密度数据转换为血红蛋白浓度。

```
raw_haemo = mne.preprocessing.nirs.beer_lambert_law(raw_od, ppf = 0.1)
raw_haemo.plot(n_channels=len(raw_haemo.ch_names), duration=500, show_scrollbars=False)
```





从信号中移除心率等信号，进行滤波

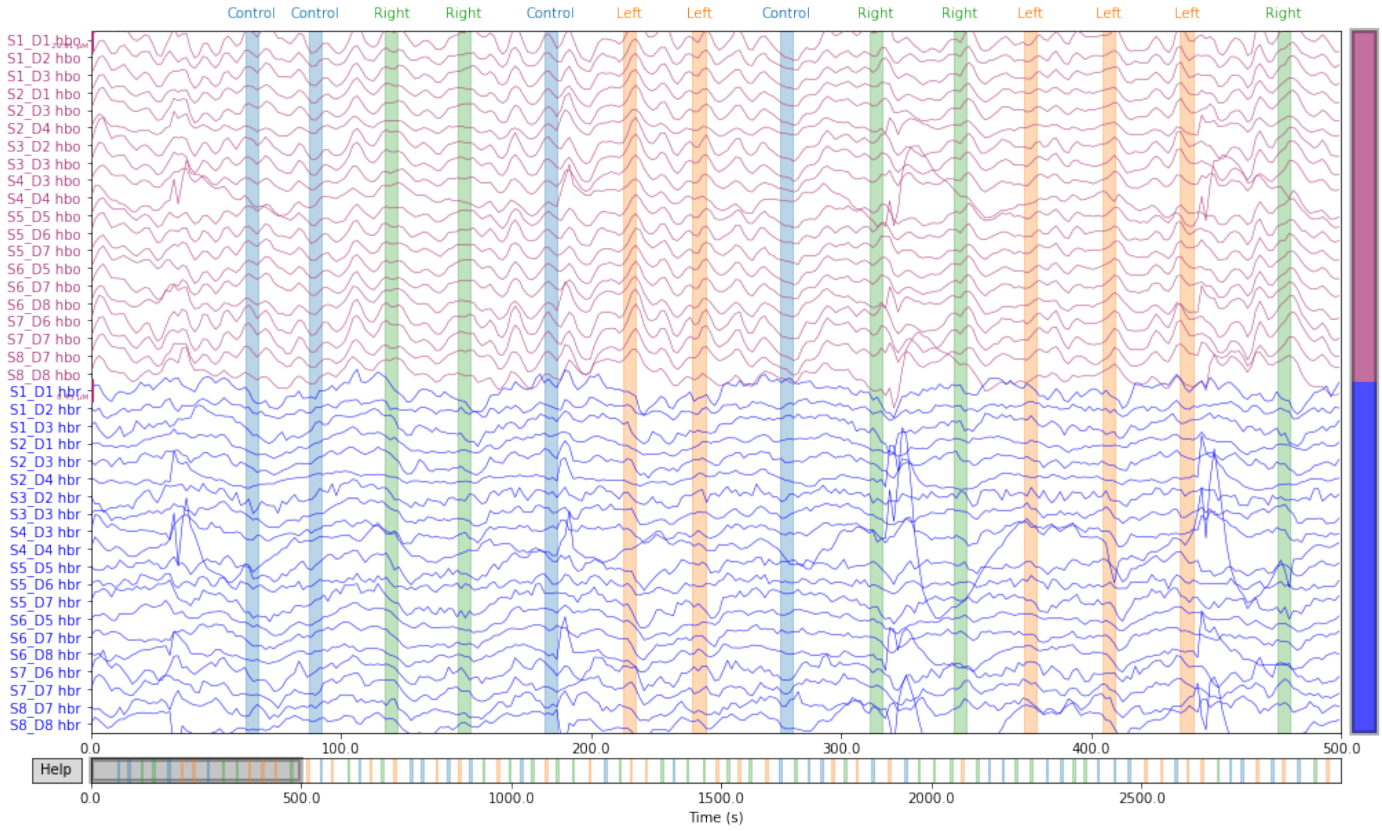
进行生理信号剔除，使用滤波方法

```
#fig = raw_haemo.plot_psd(average=True)
#fig.suptitle('Before filtering', weight='bold', size='x-large')
#fig.subplots_adjust(top=0.88)
raw_haemo = raw_haemo.filter(0.02, 0.2, h_trans_bandwidth=0.2, l_trans_bandwidth=0.02)
#fig = raw_haemo.plot_psd(average=True)
#fig.suptitle('After filtering', weight='bold', size='x-large')
#fig.subplots_adjust(top=0.88)
raw_haemo.plot(n_channels=len(raw_haemo.ch_names),
   duration=500, show_scrollbars=True,
      scalings='auto', clipping=None);
```

```
Filtering raw data in 1 contiguous segment
Setting up band-pass filter from 0.02 - 0.2 Hz

FIR filter parameters
---------------------
Designing a one-pass, zero-phase, non-causal bandpass filter:
- Windowed time-domain design (firwin) method
- Hamming window with 0.0194 passband ripple and 53 dB stopband attenuation
- Lower passband edge: 0.02
- Lower transition bandwidth: 0.02 Hz (-6 dB cutoff frequency: 0.01 Hz)
- Upper passband edge: 0.20 Hz
- Upper transition bandwidth: 0.20 Hz (-6 dB cutoff frequency: 0.30 Hz)
- Filter length: 1291 samples (165.248 sec)
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   2 out of   2 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   3 out of   3 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done   4 out of   4 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done  40 out of  40 | elapsed:    0.1s finished
```



分段

我们提取感兴趣的事件并将其可视化

```
events, event_dict = mne.events_from_annotations(raw_haemo)
#fig = mne.viz.plot_events(events, event_id=event_dict,sfreq=raw_haemo.info['sfreq'])
#fig.subplots_adjust(right=0.7)  # make room for the legend
```

```
Used Annotations descriptions: ['Control', 'Left', 'Right']
```

定义分段范围、伪迹阈值标准，基线校正，并提取感兴趣的分段。

```
reject_criteria = dict(hbo=80e-6)
tmin, tmax = -5, 15

epochs = mne.Epochs(raw_haemo, events, event_id=event_dict,
                    tmin=tmin, tmax=tmax,
                    reject=reject_criteria, reject_by_annotation=True,
                    proj=True, baseline=(None, 0), preload=True,
                    detrend=None, verbose=True)
#epochs.plot_drop_log() #日志可视化分段
```

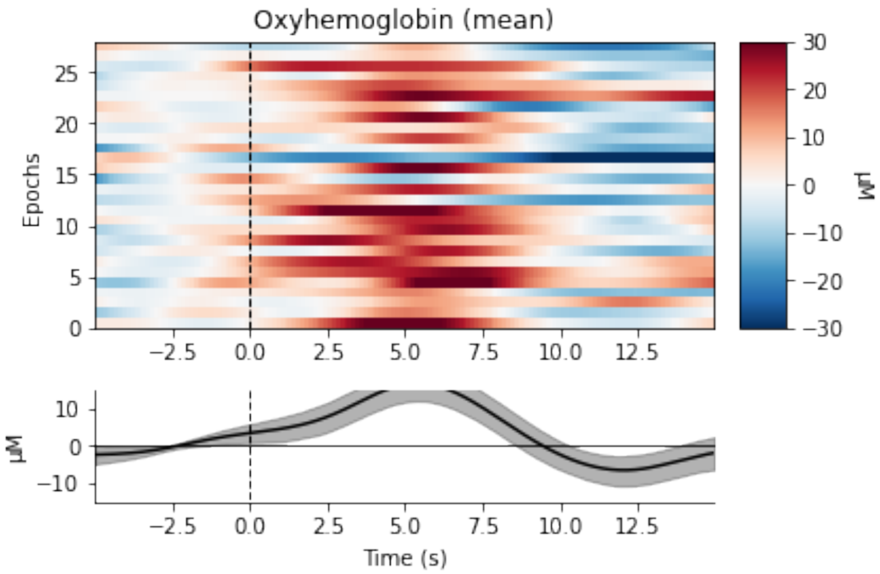
```
Not setting metadata
90 matching events found
Setting baseline interval to [-4.992, 0.0] sec
```

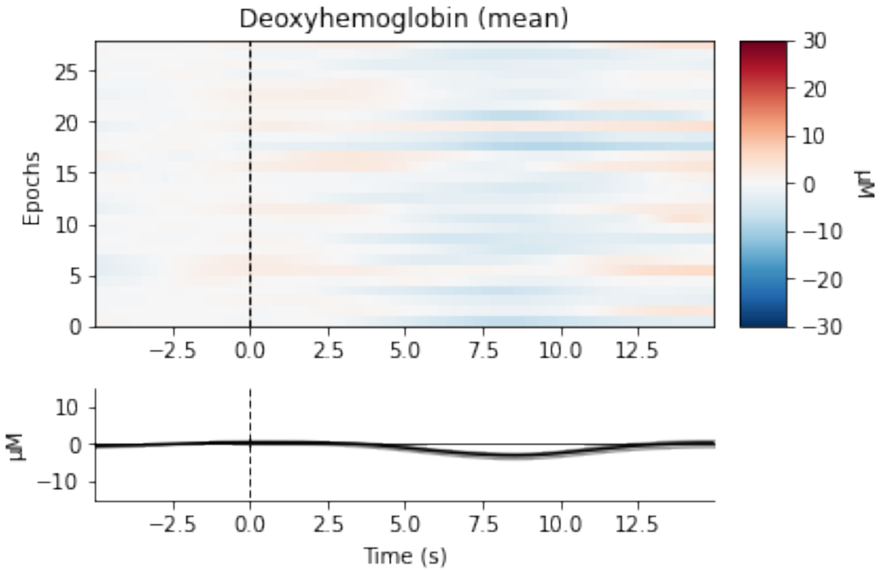
```
Applying baseline correction (mode: mean)
0 projection items activated
Using data from preloaded Raw for 90 events and 157 original time points ...
    Rejecting  epoch based on HBO : ['S4_D4 hbo']
    Rejecting  epoch based on HBO : ['S4_D4 hbo', 'S8_D8 hbo']
    Rejecting  epoch based on HBO : ['S4_D4 hbo']
    Rejecting  epoch based on HBO : ['S4_D4 hbo', 'S8_D8 hbo']
    Rejecting  epoch based on HBO : ['S7_D6 hbo']
    Rejecting  epoch based on HBO : ['S1_D1 hbo', 'S3_D3 hbo', 'S4_D4 hbo', 'S7_D6 hbo', 'S7_D7 hbo', 'S8_D8 hbo']
    Rejecting  epoch based on HBO : ['S7_D6 hbo']
    Rejecting  epoch based on HBO : ['S4_D4 hbo']
    Rejecting  epoch based on HBO : ['S4_D4 hbo', 'S8_D8 hbo']
    Rejecting  epoch based on HBO : ['S4_D4 hbo', 'S6_D8 hbo', 'S8_D8 hbo']
    Rejecting  epoch based on HBO : ['S4_D4 hbo', 'S8_D8 hbo']
11 bad epochs dropped
```

查看各试验中反应的一致性

```
epochs['Left'].plot_image(combine='mean', vmin=-30, vmax=30,
                             ts_args=dict(ylim=dict(hbo=[-15, 15],
                                                    hbr=[-15, 15])))
```

```
Not setting metadata
28 matching events found
No baseline correction applied
0 projection items activated
Not setting metadata
28 matching events found
No baseline correction applied
0 projection items activated
combining channels using "mean"
combining channels using "mean"
```

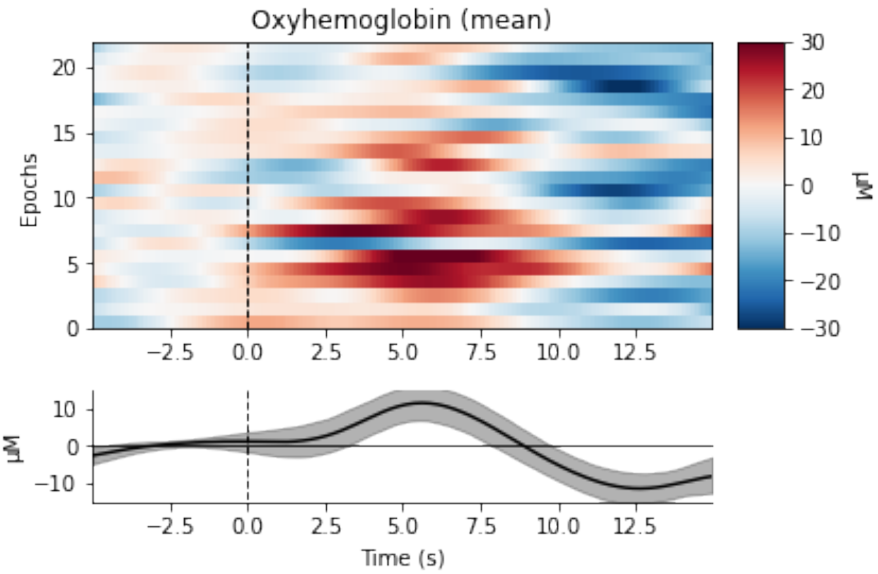


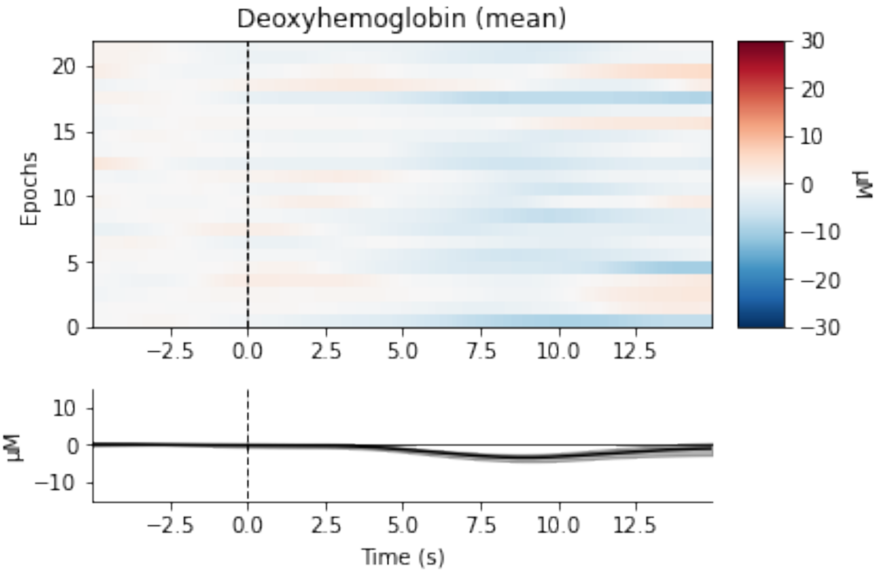


```
[<Figure size 432x288 with 3 Axes>, <Figure size 432x288 with 3 Axes>]
```

```
epochs['Right'].plot_image(combine='mean', vmin=-30, vmax=30,
                           ts_args=dict(ylim=dict(hbo=[-15, 15],
                                                  hbr=[-15, 15])))
```

```
Not setting metadata
22 matching events found
No baseline correction applied
0 projection items activated
Not setting metadata
22 matching events found
No baseline correction applied
0 projection items activated
combining channels using "mean"
combining channels using "mean"
```



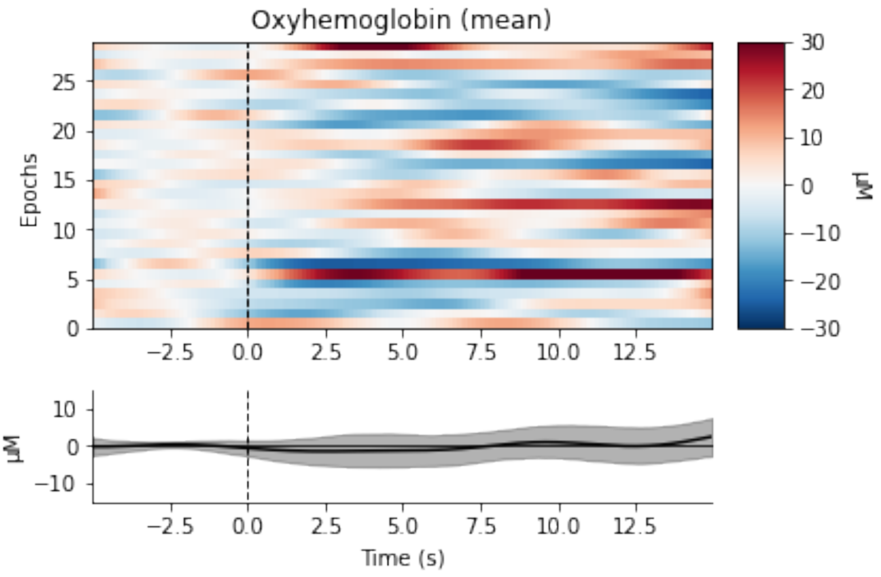


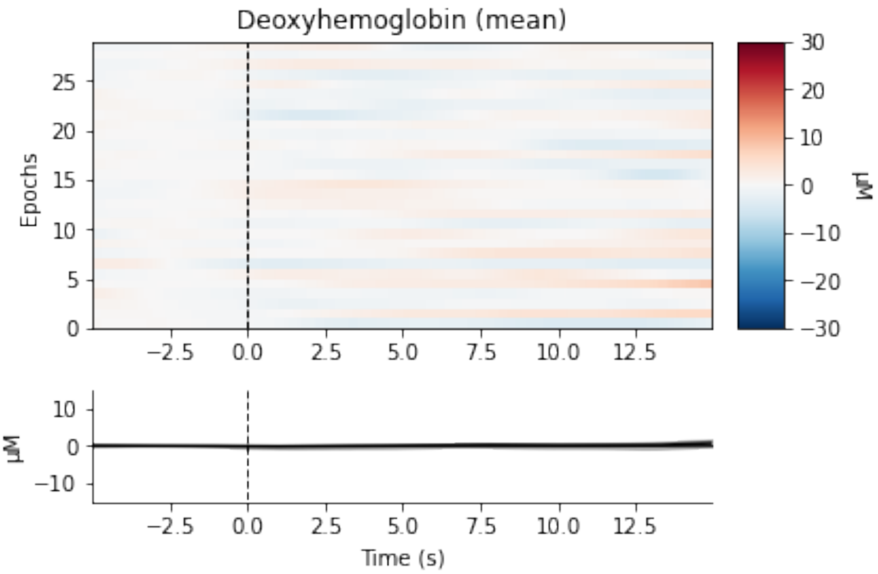
```
[<Figure size 432x288 with 3 Axes>, <Figure size 432x288 with 3 Axes>]
```

We can also view the epoched data for the control condition and observe that it does not show the expected morphology.

```
epochs['Control'].plot_image(combine='mean', vmin=-30, vmax=30,
                             ts_args=dict(ylim=dict(hbo=[-15, 15],
                                                    hbr=[-15, 15])))
```

```
Not setting metadata
29 matching events found
No baseline correction applied
0 projection items activated
Not setting metadata
29 matching events found
No baseline correction applied
0 projection items activated
combining channels using "mean"
combining channels using "mean"
```

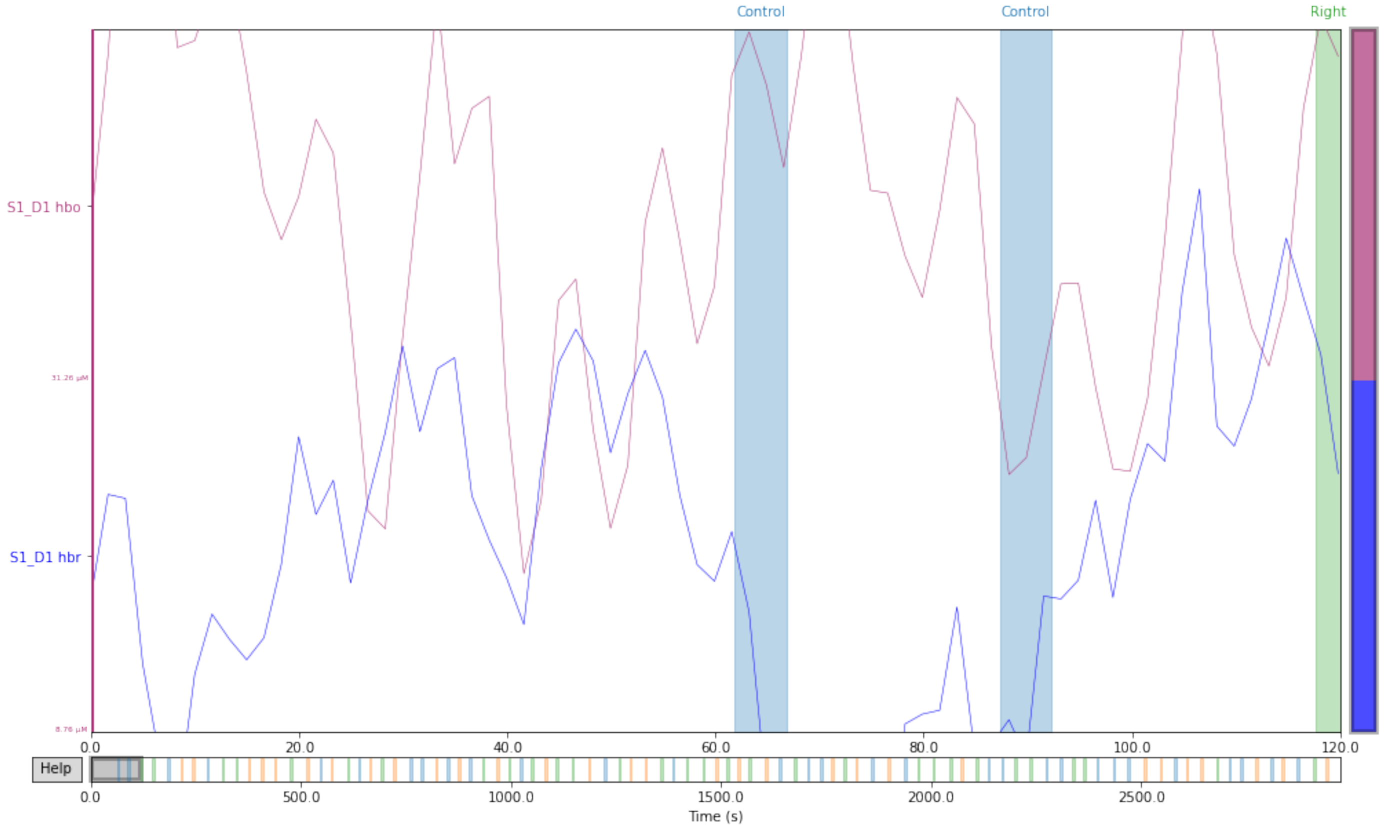




```
[<Figure size 432x288 with 3 Axes>, <Figure size 432x288 with 3 Axes>]
```

查看单通道的数据

```
single_haemo = raw_haemo.copy()
picks = mne.pick_channels_regexp(single_haemo.ch_names, regexp='S1_D1')
raw_haemo.plot(order=picks, n_channels=len(picks), duration=120, show_scrollbars=True,
      scalings='auto', clipping=None);
sep_ch = single_haemo.pick(picks)
```



绘制标准fNIRS响应图像

绘图HbO和HbR在同一图上，以说明两者之间的关系。

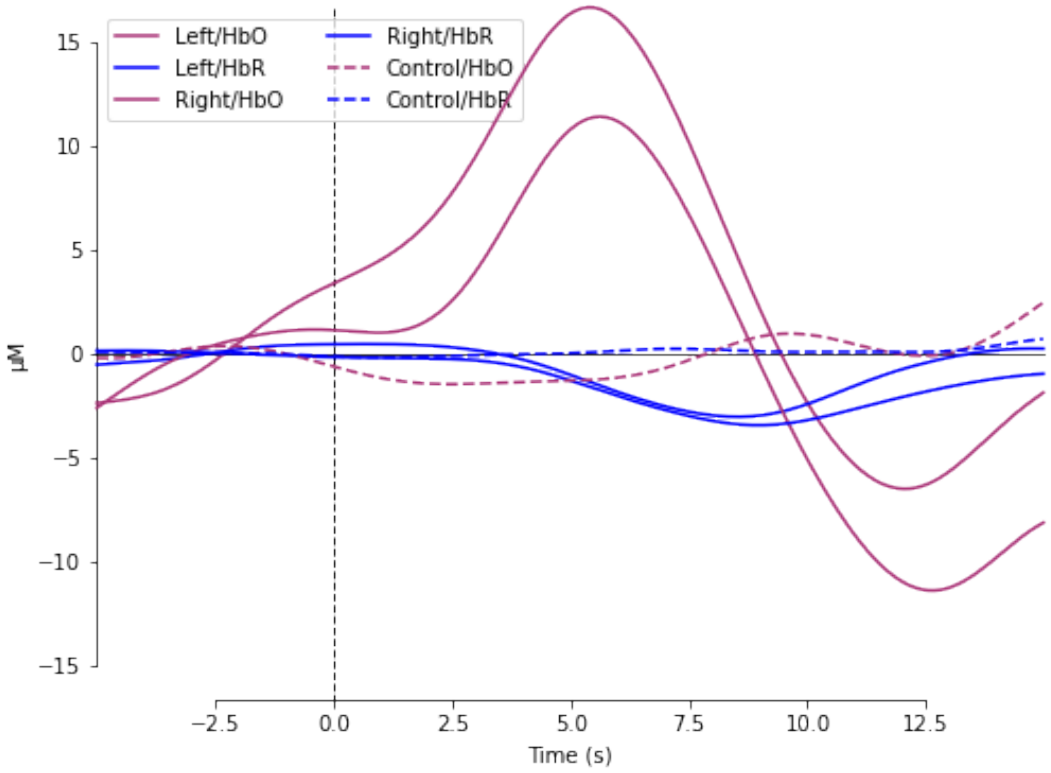
```
evoked_dict = {'Left/HbO': epochs['Left'].average(picks='hbo'),
               'Left/HbR': epochs['Left'].average(picks='hbr'),
               'Right/HbO': epochs['Right'].average(picks='hbo'),
               'Right/HbR': epochs['Right'].average(picks='hbr'),
               'Control/HbO': epochs['Control'].average(picks='hbo'),
               'Control/HbR': epochs['Control'].average(picks='hbr')}

# Rename channels until the encoding of frequency in ch_name is fixed
for condition in evoked_dict:
    evoked_dict[condition].rename_channels(lambda x: x[:-4])

color_dict = dict(HbO='#AA3377', HbR='b')
styles_dict = dict(Control=dict(linestyle='dashed'))

mne.viz.plot_compare_evokeds(evoked_dict, combine="mean", ci=0.95,
                             colors=color_dict, styles=styles_dict)
```

```
combining channels using "mean"
combining channels using "mean"
combining channels using "mean"
combining channels using "mean"
combining channels using "mean"
combining channels using "mean"
```



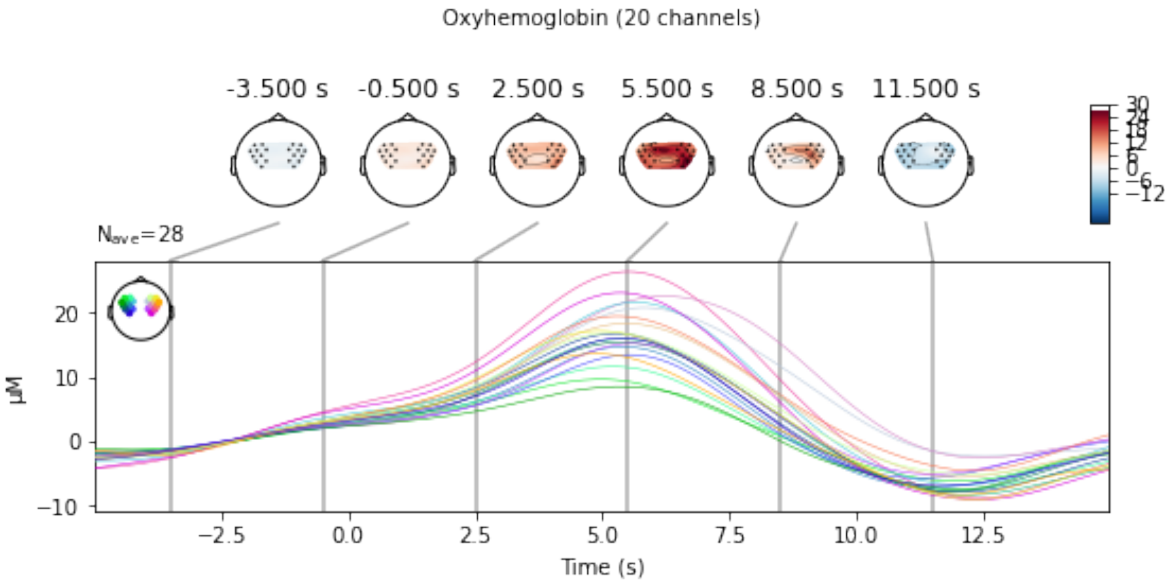
```
[<Figure size 576x432 with 1 Axes>]
```

查看地形图表示

查看整个响应过程中地形活动的变化。

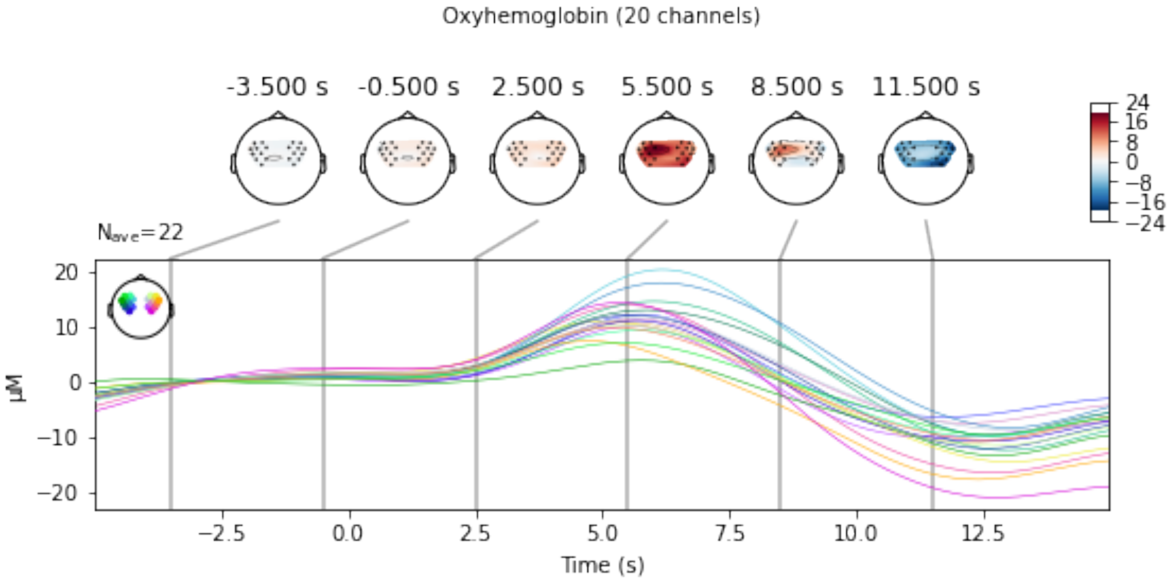
```
times = np.arange(-3.5, 13.2, 3.0)
topomap_args = dict(extrapolate='local')
epochs['Left'].average(picks='hbo').plot_joint(
    times=times, topomap_args=topomap_args)
```

```
No projector specified for this dataset. Please consider the method self.add_proj.
```



```python
times = np.arange(-3.5, 13.2, 3.0)
topomap_args = dict(extrapolate='local')
epochs['Right'].average(picks='hbo').plot_joint(
    times=times, topomap_args=topomap_args)
```

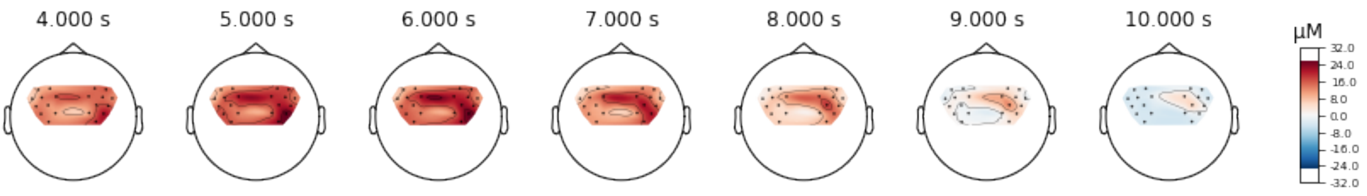
```
No projector specified for this dataset. Please consider the method self.add_proj.
```

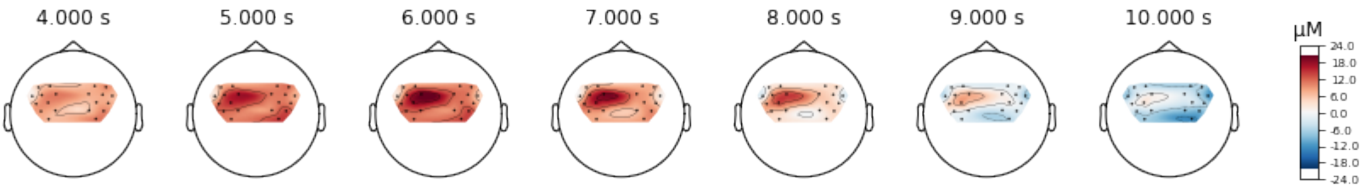


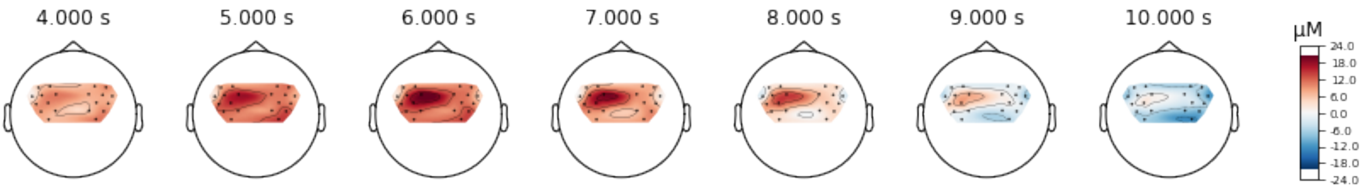
比较左右手

查看左右条件生成地形图活动的位置。

```
times = np.arange(4.0, 11.0, 1.0)
epochs['Left'].average(picks='hbo').plot_topomap(
    times=times, **topomap_args)
epochs['Right'].average(picks='hbo').plot_topomap(
    times=times, **topomap_args)
```

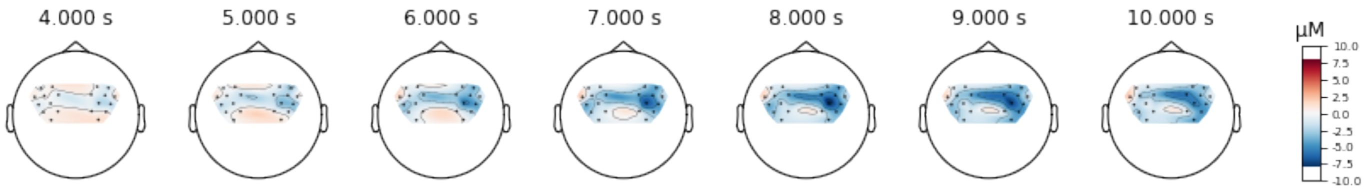


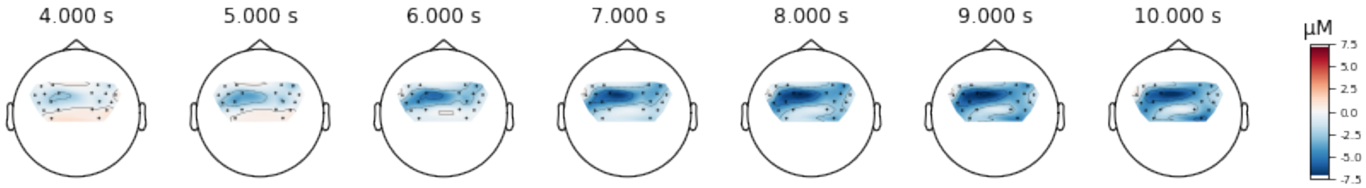


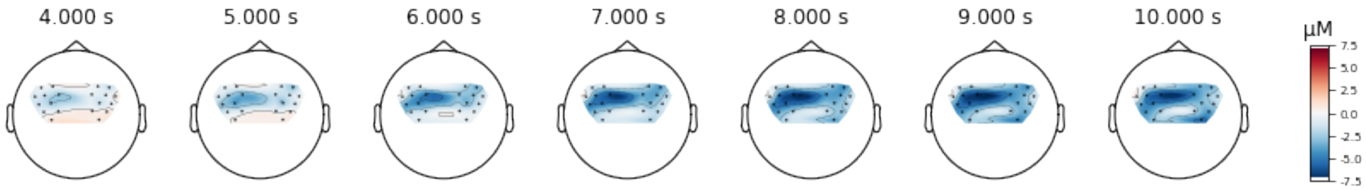


查看HbR

```
epochs['Left'].average(picks='hbr').plot_topomap(
    times=times, **topomap_args)
epochs['Right'].average(picks='hbr').plot_topomap(
    times=times, **topomap_args)
```



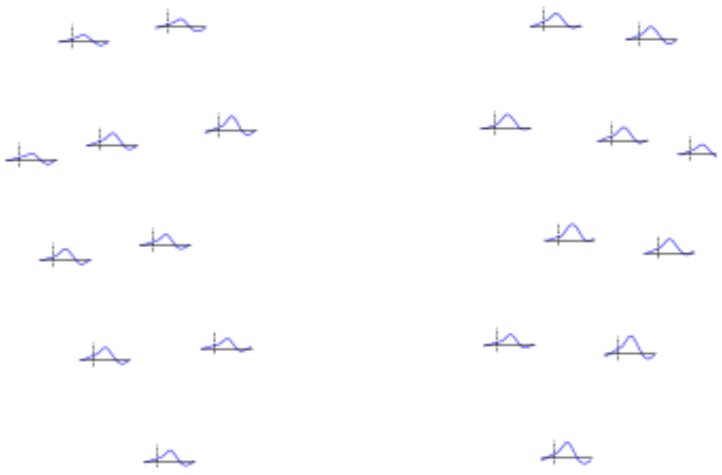




查看各个通道波形图

```
fig, axes = plt.subplots(nrows=1, ncols=1, figsize=(6, 4))
mne.viz.plot_evoked_topo(epochs['Left'].average(picks='hbo'), color='b',
                         axes=axes, legend=False)
mne.viz.plot_evoked_topo(epochs['Right'].average(picks='hbo'), color='r',
                         axes=axes, legend=False)

# Tidy the legend:
leg_lines = [line for line in axes.lines if line.get_c() == 'b'][:1]
leg_lines.append([line for line in axes.lines if line.get_c() == 'r'][0])
fig.legend(leg_lines, ['Left', 'Right'], loc='lower right')
```



```
<matplotlib.legend.Legend at 0x7fa00619ad60>
```

谢谢大家观看，如有帮助，来个喜欢或者关注吧！

本文作者：陈锐CR

博客地址 ： Chen Rui Blog

知乎地址 ：知乎专栏

B站地址 ：B站主页
书店地址 ：书店主页
简书地址 ：简书主页
CSDN地址 ：csdn主页
学术导航 ：导航主页