

# Laboratory Report No. 4

Student of group IU1-41M Derina Ekaterina

## 1 Цель работы

Целью данной лабораторной работы является исследование скрытых характеристик сигналов электроэнцефалографии на основе датасета записей ЭЭГ с приступами.

## 2 Ключевой навык

Ключевой получаемый навык - умение рассчитывать спектрограмму и скейлограмму сигнала и применять вейвлет преобразование.

## 3 Задание

В ходе выполнения лабораторной работы требуется:

1. Загрузите запись ЭЭГ из базы данных, выбрать любой файл EDF; 2. По аннотации из датасета определить, где в записи ЭЭГ указан приступ; 3. Построить график временной зависимости ЭЭГ в момент приступа (выбрать диапазон времени и масштаб так, чтобы было наглядно). Сделать усреднение всех каналов ЭЭГ в один. Удалить из сигнала все частоты выше 60 Гц. Для этого преобразованного сигнала:
  - 1) построить спектрограмму сигнала;
  - 2) построить вейвлет-преобразование (скейлограмму).

## 1 工作目的

本实验的目的是基于脑电图(EEG)数据集研究癫痫发作期间脑电信号的隐藏特征。

## 2 关键技能

获得的核心技能是计算信号的频谱图和尺度图，并应用小波变换。

## 3 实验任务

在实验过程中需要完成：

1. 从数据库中加载EEG记录，选择任意EDF格式文件；
2. 根据数据集中的标注确定EEG记录中癫痫发作的位置；
3. 绘制癫痫发作时EEG的时间依赖图（选择合适的时间范围和比例使其清晰可见）。将所有EEG通道平均为一个信号。去除信号中60Hz以上的频率。对于这个转换后的信号：
  - 1) 绘制信号的频谱图；
  - 2) 绘制小波变换(尺度图)。

## 4 Описание метода исследования

В ходе лабораторной работы после предварительной обработки ЭЭГ-сигнала (усреднение по каналам и фильтрация частот выше 60 Гц), производится спектральный анализ сигнала двумя методами[1]: спектрограммой и непрерывным вейвлет-преобразованием (CWT). Оба метода позволяют изучить, как энергетика сигнала распределена по частотам во времени, что особенно важно для детектирования кратковременных патологических состояний, таких как эпилептический приступ.

### 1. Построение спектрограммы сигнала

Спектрограмма строится на основе оконного преобразования Фурье (Short-Time Fourier Transform, STFT)[2]. В этом методе сигнал разбивается на перекрывающиеся окна, внутри которых выполняется обычное преобразование Фурье. Это позволяет получить спектральную информацию, локализованную во времени.

Математически STFT определяется как:

$$\text{STFT}_x(t, \omega) = \int_{-\infty}^{\infty} x(\tau) \cdot w(\tau - t) \cdot e^{-j\omega\tau} d\tau \quad (1)$$

где:

- $x(\tau)$  — исходный сигнал,
- $w(\tau - t)$  — оконная функция, сдвинутая по времени (например, окно Хэмминга),
- $\omega$  — угловая частота,
- $\tau$  — момент времени.

## 4 研究方法描述

在实验室工作中，经过EEG信号的预处理（通道平均和过滤60Hz以上频率）后，[1] 采用两种方法进行信号频谱分析：频谱图和连续小波变换(CWT)。这两种方法都可以研究信号能量如何随时间在频率上分布，这对于检测短暂的病理状态（如癫痫发作）特别重要。

### 1. 信号频谱图的构建

频谱图基于短时傅里叶变换(STFT)[2] 构建。在这种方法中，信号被分成重叠的窗口，在每个窗口内执行常规傅里叶变换。这样可以获得时间上局部化的频谱信息。

STFT的数学定义为：

$$\text{STFT}_x(t, \omega) = \int_{-\infty}^{\infty} x(\tau) \cdot w(\tau - t) \cdot e^{-j\omega\tau} d\tau \quad (1)$$

其中：

- $x(\tau)$  — 原始信号，
- $w(\tau - t)$  — 时间平移的窗函数（例如汉明窗），
- $\omega$  — 角频率，
- $\tau$  — 时间点。

频谱图是STFT模的平方：

$$\text{Spectrogram}_x(t, \omega) = |\text{STFT}_x(t, \omega)|^2 \quad (2)$$

这种二维表示显示了信号在每个时间点存在的频率分量。[3]

### 2. 小波变换(尺度图)

与STFT不同，连续小波变换(CWT)可以自适应地改变时间和频率分辨率，为高频分量提供高时间精度，为低频分量提供高频率精度。

CWT定义如下：

$$\text{CWT}_x(a, b) = \frac{1}{\sqrt{|a|}} \int_{-\infty}^{\infty} x(t) \cdot \psi^* \left( \frac{t - b}{a} \right) dt \quad (3)$$

其中：

- $x(t)$  — 待分析信号，

Спектрограмма получается как квадрат модуля STFT:

$$\text{Spectrogram}_x(t, \omega) = |\text{STFT}_x(t, \omega)|^2 \quad (2)$$

Это двумерное представление показывает, какие частотные компоненты присутствуют в сигнале в каждый момент времени[3].

## 2. Вейвлет-преобразование (скейлограмма)

В отличие от STFT, непрерывное вейвлет-преобразование (CWT) позволяет адаптивно изменять временное и частотное разрешение, обеспечивая высокую временную точность для высокочастотных компонентов и высокую частотную точность для низкочастотных.

CWT определяется следующим образом:

$$\text{CWT}_x(a, b) = \frac{1}{\sqrt{|a|}} \int_{-\infty}^{\infty} x(t) \cdot \psi^* \left( \frac{t-b}{a} \right) dt \quad (3)$$

где:

- $x(t)$  — анализируемый сигнал,
- $\psi(t)$  — вейвлет-функция (например, комплексный Морле),
- $\psi^*(t)$  — комплексно-сопряжённая функция,
- $a$  — масштаб (обратно пропорционален частоте),
- $b$  — временной сдвиг,
- $\frac{1}{\sqrt{|a|}}$  — нормирующий коэффициент, обеспечивающий сохранение энергии при изменении масштаба.

Результат отображается на скейлограмме, где по оси X — время, по оси Y — масштаб или эквивалентная частота, а цвет отражает энергию (амплитуду преобразования) на соответствующем масштабе и моменте времени:

- $\psi(t)$  — маловолновая функция (например, комплексный Морле),
- $\psi^*(t)$  — комплексно-сопряжённая функция,
- $a$  — масштаб (обратно пропорционален частоте),
- $b$  — временной сдвиг,
- $\frac{1}{\sqrt{|a|}}$  — нормирующий коэффициент, обеспечивающий сохранение энергии при изменении масштаба.

Результат в скейлограмме, где X-ось — время, Y-ось — масштаб или эквивалентная частота, а цвет отражает энергию (амплитуду преобразования) на соответствующем масштабе и моменте времени:

$$\text{Scalogram}_x(a, b) = |\text{CWT}_x(a, b)|^2 \quad (4)$$

Объединение двух методов может более полно анализировать динамику приступов эпилепсии: спектрограмма показывает высокочастотные частоты, а скейлограмма может выявить скрытые структуры и переключение активности мозга.

$$\text{Scalogram}_x(a, b) = |\text{CWT}_x(a, b)|^2 \quad (4)$$

Использование обеих методик в комплексе позволяет более полно проанализировать динамику приступа: спектрограмма показывает частоты с высокой амплитудой, а скейлограмма позволяет выявить скрытые структуры и переходы между режимами мозговой активности.

## 5 Ход работы

### 5.1 Установка библиотеки

Устанавливаем библиотеку pywavelets mne и zenodo-get.

```
1 !pip install pywavelets mne
2 !pip install zenodo-get
```

**Листинг 1** Установка библиотеки

### 5.2 Загрузка набора данных

Загружаем набор данных.

```
1 !zenodo_get "10.5281/zenodo.4940267"
   " -k -g eeg19.edf
2 data_from_raw_edf = mne.io.read_raw_edf(
   '/content/eeg19.edf', preload=True)
3 eeg_data, eeg_times = data_from_raw_edf.
   get_data(return_times=True)
```

**Листинг 2** Загрузка набора данных

Строим исходную ЭЭГ.

## 5 实验流程

### 5.1 安装库

安装pywavelets、mne和zenodo-get库。

```
1 !pip install pywavelets mne
2 !pip install zenodo-get
```

**代码 1** 安装库

### 5.2 加载数据集

加载数据集。

```
1 !zenodo_get "10.5281/zenodo.4940267"
   " -k -g eeg19.edf
2 data_from_raw_edf = mne.io.read_raw_edf(
   '/content/eeg19.edf', preload=True)
3 eeg_data, eeg_times = data_from_raw_edf.
   get_data(return_times=True)
```

**代码 2** 加载数据集

绘制原始EEG。

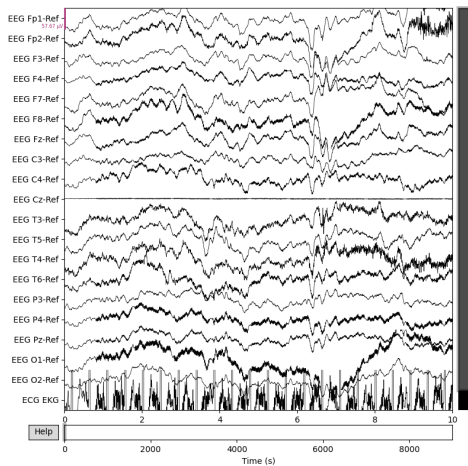


Рис. 1 Исходная ЭЭГ

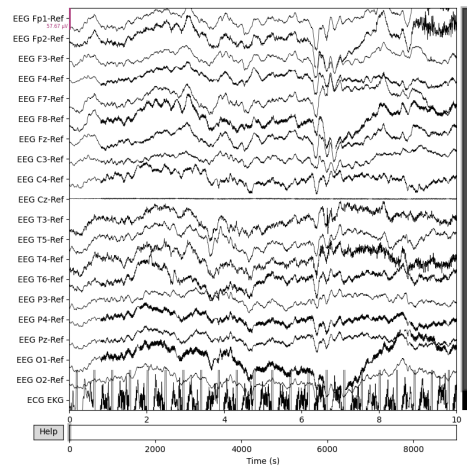


图 1 原始脑电图

Удаляем данные последнего канала ввиду неинформативности в контексте данной задачи.

```
1 eeg_data = np.delete(eeg_data, [
    len(eeg_data)-1], axis = 0)
```

**Листинг 3** Удаляем ненужные сигналы из массива данных ЭЭГ

Построим график временной зависимости ЭЭГ в момент приступа (начало приступа 5800 отсчет).

```
1 t_index_begin = np.where(eeg_times >
    5500 - 480)[0][0]
2 t_index_end = np.where(eeg_times > 5800
    + 480)[0][0]
3 t = eeg_times[t_index_begin:t_index_end]
4 T = t[-1] - t[0]
5 N = len(t)
6 for i in range(
    len(data_from_raw_edf.ch_names)-2):
7     plt.plot(t, eeg_data[i,
        t_index_begin:t_index_end],
        linewidth = 0.1)
8 plt.show()
```

**Листинг 4** Построение графика временной зависимости ЭЭГ в момент приступа

Удаляем данные последнего канала, потому что в заданном контексте не информативны.

```
1 eeg_data = np.delete(eeg_data, [
    len(eeg_data)-1], axis = 0)
```

**код 3** Удаляем лишние сигналы

Нарисуем график зависимости ЭЭГ от времени во время приступа (приступ начался около 5800 точек отсчета).

```
1 t_index_begin = np.where(eeg_times >
    5500 - 480)[0][0]
2 t_index_end = np.where(eeg_times > 5800
    + 480)[0][0]
3 t = eeg_times[t_index_begin:t_index_end]
4 T = t[-1] - t[0]
5 N = len(t)
6 for i in range(
    len(data_from_raw_edf.ch_names)-2):
7     plt.plot(t, eeg_data[i,
        t_index_begin:t_index_end],
        linewidth = 0.1)
8 plt.show()
```

**код 4** Нарисовать график зависимости ЭЭГ от времени во время приступа

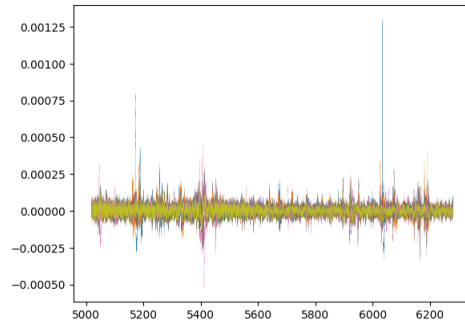


Рис. 2 Временная зависимость ЭЭГ в момент приступа

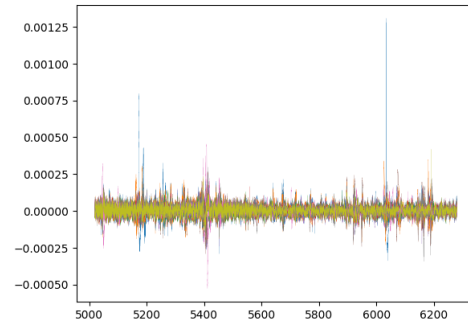


图 2 癫痫期间脑电时间依赖图

Сделаем простой усредненный сигнал - все каналы в один массив.

```
1 y=[]
2 for i in range(len(eeg_data)-2):
3     y += eeg_data[i, t_index_begin:
4         t_index_end]
5 y = y/(len(eeg_data)-1)
6 plt.plot(t,y)
```

Листинг 5 Усреднение сигнала

创建一个简单的平均信号 - 将所有通道合并为一个数组。

```
1 y=[]
2 for i in range(len(eeg_data)-2):
3     y += eeg_data[i, t_index_begin:
4         t_index_end]
5 y = y/(len(eeg_data)-1)
6 plt.plot(t,y)
```

代码 5 信号平均

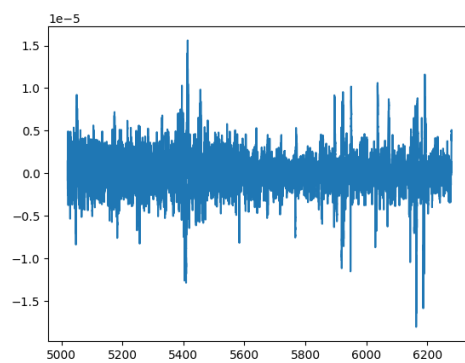


Рис. 3 Усредненный сигнал

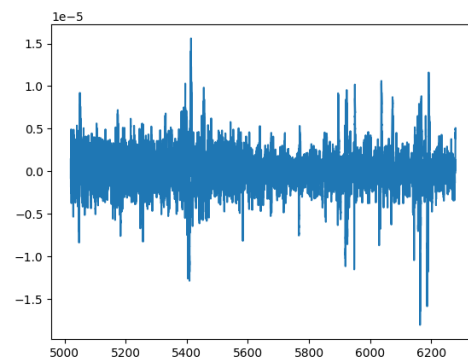


图 3 平均信号

## 5.3 Фильтрация сигнала

Фильтр Баттерфорда для вырезания полосы частот.

```
1 from scipy import signal
2 def butter_bandpass_filter(data, lowcut,
3     highcut, fs, order=5):
```

## 5.3 信号滤波

巴特沃斯滤波器用于切除频带。

```
1 from scipy import signal
2 def butter_bandpass_filter(data, lowcut,
3     highcut, fs, order=5):
4     nyquist = 0.5 * fs
5     low = lowcut / nyquist
6     high = highcut / nyquist
```

```

3     nyquist = 0.5 * fs
4     low = lowcut / nyquist
5     high = highcut / nyquist
6     b, a = signal.butter(order, [low,
7                               high], btype='band
8                               ')
9     filtered_data = signal.lfilter(b, a,
10    data)
11     return filtered_data
12
13 fs = len(t)/(T)
14 lowcut = 1
15 highcut = 60
16 y_filt = np.apply_along_axis(
17     butter_bandpass_filter, axis=0, arr
18     =y, lowcut=lowcut, highcut=highcut,
19     fs=fs)
20 plt.plot(t,y, t, y_filt)

```

Листинг 6 Фильтр Баттерфорда

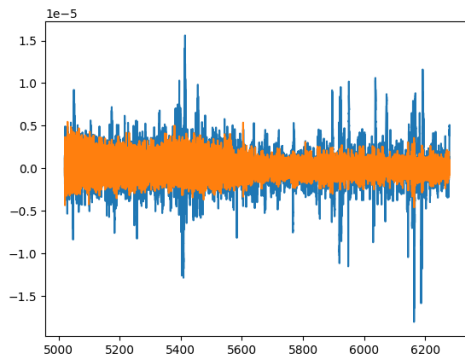


Рис. 4 Усредненный сигнал после фильтрации

## 5.4 Спектр Фурье

```

1 from scipy.fft import fft, fftfreq
2
3 y = y_filt
4 yfft = fft(y)
5 xf = fftfreq(N, T/N)[:N//2]
6 yf = np.abs(yfft[0:N//2])
7
8 f_viewmax = 60
9 if len(np.where(xf > f_viewmax)[0]) ==
10 0:
11     index_f_viewmax = len(xf)
12 else:
13     index_f_viewmax = np.where(xf >
14     f_viewmax)[0][0]
15 print('chislo tochek v diapazone
16     preobrazovaniya Fur'e:', len(xf))
17 print('maksimal'naya chastota', xf[-1])
18 plt.figure(figsize=(10, 5))

```

```

6     b, a = signal.butter(order, [low,
7                               high], btype='band
8                               ')
9     filtered_data = signal.lfilter(b, a,
10    data)
11     return filtered_data
12
13 fs = len(t)/(T)
14 lowcut = 1
15 highcut = 60
16 y_filt = np.apply_along_axis(
17     butter_bandpass_filter, axis=0, arr
18     =y, lowcut=lowcut, highcut=highcut,
19     fs=fs)
20 plt.plot(t,y, t, y_filt)

```

代码 6 巴特沃斯滤波器

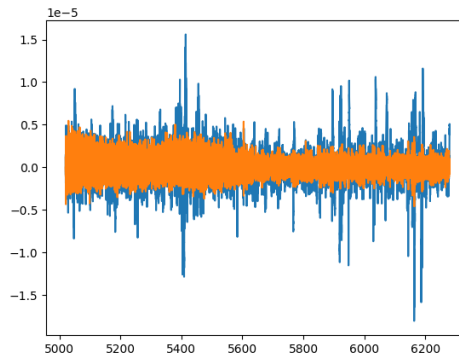


图 4 滤波后的平均信号

## 5.4 傅里叶频谱

```

1 from scipy.fft import fft, fftfreq
2
3 y = y_filt
4 yfft = fft(y)
5 xf = fftfreq(N, T/N)[:N//2]
6 yf = np.abs(yfft[0:N//2])
7
8 f_viewmax = 60
9 if len(np.where(xf > f_viewmax)[0]) ==
10 0:
11     index_f_viewmax = len(xf)
12 else:
13     index_f_viewmax = np.where(xf >
14     f_viewmax)[0][0]
15 print('Number of points in Fourier
16     transform range:', len(xf))
17 print('Maximum frequency', xf[-1])
18 plt.figure(figsize=(10, 5))
19 plt.plot(xf[0:index_f_viewmax], yf[0:
20     index_f_viewmax], label='FFT
21     transform')
22 plt.xlabel('Frequency [Hz]
23     ', fontsize=12)

```

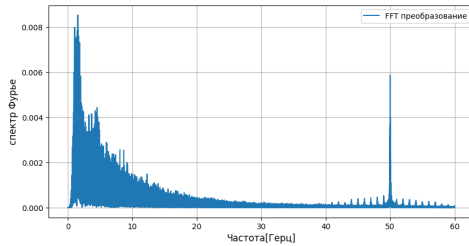
```

16 plt.plot(xf[0:index_f_viewmax], yf[0:
    index_f_viewmax], label='FFT
    преобразование')
17 plt.xlabel('chastota[Hz]', fontsize=12)
18 plt.ylabel('spektr Fur'e', fontsize=12)
19 plt.grid()
20 plt.legend()
21 plt.show()

```

**Листинг 7** Быстрое преобразование Фурье

Число точек в диапазоне преобразования Фурье: 161280  
Максимальная частота 127.99960317337293



**Рис. 5** FFT преобразование

## 5.5 Вейвлет-преобразование

```

1 wavlist_continuous = pywt.wavelist(kind=
    'continuous')
2 wavlist_discrete = pywt.wavelist(kind='
    discrete')

```

**Листинг 8** Перечень вейвлетов

Изобразим график зависимости частоты и масштаба.

```

1 scale_max = 300
2 scale_min = 3
3 scales = np.linspace(scale_min,
    scale_max, num = 25, endpoint=True)
4
5 wavelet_core = 'morl'
6 fs = len(t)/(T)
7 dt = 1/ fs
8 coef, freqs = pywt.cwt(y, scales,
    wavelet_core, sampling_period = dt)
9
10 plt.figure(figsize=(7, 7))
11 plt.grid()
12 plt.yticks(np.arange(min(freqs),
    max(freqs), (max(freqs) -
    min(freqs))/10))
13 plt.xticks(np.arange(min(scales),
    max(scales), (max(scales) -
    min(scales))/10))
14 plt.ylabel('chastota[Hz]', fontsize=12)

```

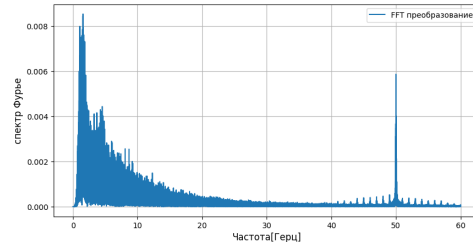
```

18 plt.ylabel('Fourier spectrum
    ', fontsize=12)
19 plt.grid()
20 plt.legend()
21 plt.show()

```

**код 7** Быстрое преобразование Фурье

Number of points in Fourier transform  
range: 161280 Maximum frequency  
127.99960317337293



**код 7** Быстрое преобразование Фурье

## 5.5 小波变换

```

1 wavlist_continuous = pywt.wavelist(kind=
    'continuous')
2 wavlist_discrete = pywt.wavelist(kind='
    discrete')

```

**код 8** 小波函数列表

绘制频率与尺度的关系图。

```

1 scale_max = 300
2 scale_min = 3
3 scales = np.linspace(scale_min,
    scale_max, num = 25, endpoint=True)
4
5 wavelet_core = 'morl'
6 fs = len(t)/(T)
7 dt = 1/ fs
8 coef, freqs = pywt.cwt(y, scales,
    wavelet_core, sampling_period = dt)
9
10 plt.figure(figsize=(7, 7))
11 plt.grid()
12 plt.yticks(np.arange(min(freqs),
    max(freqs), (max(freqs) -
    min(freqs))/10))
13 plt.xticks(np.arange(min(scales),
    max(scales), (max(scales) -
    min(scales))/10))
14 plt.ylabel('Frequency [Hz]
    ', fontsize=12)
15 plt.xlabel('Scale in wavelet transform
    ', fontsize=12)
16 plt.plot(scales, freqs, '-.')

```

**код 9** 频率与尺度的关系图

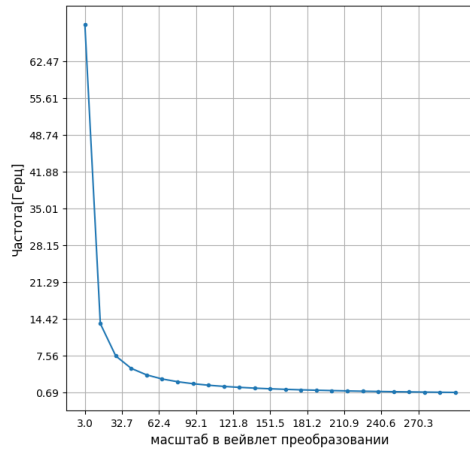


```

15 plt.xlabel('masshtab v veivlet
    preobrazovanii', fontsize=12)
16 plt.plot(scales, freqs, '-.')

```

**Листинг 9** Вейвлет-преобразование



**Рис. 6** График зависимости частоты и масштаба

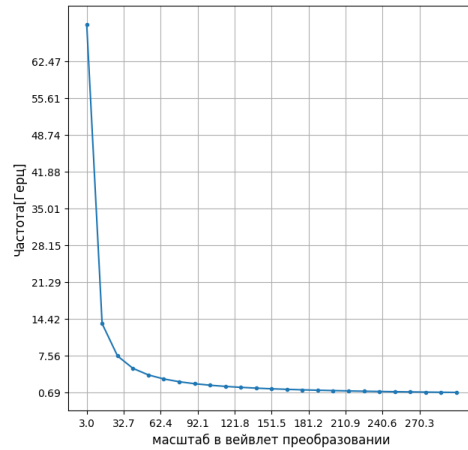
Построим скейлограмму.

```

1 plt.figure(figsize=(10, 5))
2 plt.imshow(abs(coef), extent=[t[0], t
    [-1], max(scales),
    min(scales)], interpolation='
    bilinear', cmap='plasma', aspect='
    auto')
3 #plt.imshow(abs(coef), interpolation='
    bilinear', cmap='plasma', aspect='
    auto')
4 plt.gca().invert_yaxis ()
5 plt.yticks(np.arange(min(scales),
    max(scales), (max(scales) -
    min(scales))/10))
6 plt.ylabel('masshtab ', fontsize=12)
7 plt.xlabel('vremya ', fontsize=12)
8 plt.show()

```

**Листинг 10** Скейлограмма



**图 6** 频率与尺度关系图

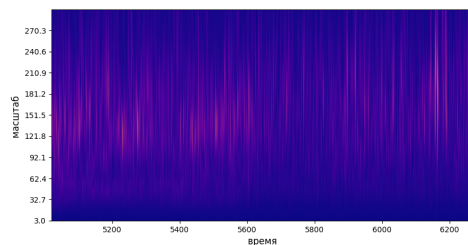
绘制尺度图。

```

1 plt.figure(figsize=(10, 5))
2 plt.imshow(abs(coef), extent=[t[0], t
    [-1], max(scales),
    min(scales)], interpolation='
    bilinear', cmap='plasma', aspect='
    auto')
3 plt.gca().invert_yaxis ()
4 plt.yticks(np.arange(min(scales),
    max(scales), (max(scales) -
    min(scales))/10))
5 plt.ylabel('Scale', fontsize=12)
6 plt.xlabel('Time', fontsize=12)
7 plt.show()

```

**代码 10** 尺度图



**图 7** 尺度图

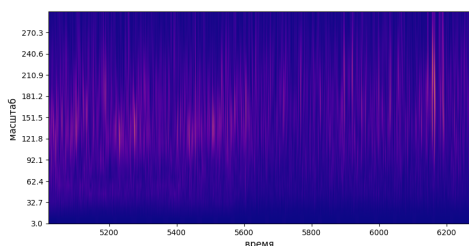


Рис. 7 Скейлограмма

## 6 Заключение

В ходе выполнения лабораторной работы были исследованы скрытые характеристики сигналов ЭЭГ на основе данных, содержащих приступы. Была загружена запись в формате EDF и определён интервал приступа по аннотациям из датасета. Для наглядного анализа построен график ЭЭГ во временной области, выполнено усреднение сигналов по каналам и применена фильтрация для удаления высокочастотных компонент (выше 60 Гц).

Основное внимание было уделено анализу преобразованного сигнала с помощью спектральных методов. Построены спектрограмма и скейлограмма, что позволило визуализировать изменение частотных характеристик сигнала во времени. В процессе работы был освоен навык применения спектрального и вейвлет-преобразования.

## References

- [1] Shoeb A. *Application of Machine Learning to Epileptic Seizure Onset Detection and Treatment*. PhD Thesis, Massachusetts Institute of Technology. (2009)
- [2] Truong N. D., Nguyen A. D., Kuhlmann L., Bonyadi M. R., Yang J., Ippolito S., Kavehei O. *Convolutional Neural Networks for Seizure Prediction Using EEG Signals*. IEEE Transactions on Biomedical Engineering, 65(9): 2108–2118. (2018)

## 6 结论

在本次实验工作中，我们研究了包含癫痫发作的EEG数据的隐藏特征。加载了EDF格式的记录，并根据数据集中的标注确定了癫痫发作的时间间隔。为了直观分析，绘制了时域EEG图，执行了通道信号平均，并应用了滤波去除60Hz以上的高频分量。

主要关注点是通过频谱方法分析转换后的信号。构建了频谱图和尺度图，这使我们能够可视化信号频率特性随时间的变化。在工作过程中，掌握了应用频谱和小波变换的技能，这些技能也能在考虑的EEG信号数据上识别癫痫发作。

- [3] Acharya U. R., Oh S. L., Hagiwara Y., Tan J. H., Adeli H. *Deep Convolutional Neural Network for the Automated Detection and Diagnosis of Seizure Using EEG Signals*. Computers in Biology and Medicine, 100: 270–278. (2018)