Санкт-Петербургский государственный университет
Прикладная математика и информатика
Динамические системы, эволюционные уравнения, экстремальные задачи
и математическая кибернетика

Степаненко Дмитрий Александрович

Библиотека программ обработки сигналов нейронной активности

Магистерская диссертация

Научный руководитель:
д. т. н., профессор Фрадков А. Л.

Рецензент:
в. н. с., к. ф.-м. н., Мельников А. А.

Санкт-Петербург
2019 г.

Saint-Petersburg State University
Applied Mathematics and Informatics
Dynamical systems, evolution equations, extremal problems and mathematical cybernetics

Dmitrii Stepanenko

Library of neural activity signal processing programs

Master's Thesis

Scientific supervisor:
Professor Alexander Fradkov

Reviewer:
leading researcher Alexander Melnikov

Saint-Petersburg
2019 г.

Содержание

1 Введение **4**

1.1 Актуальность темы исследования 4

1.2 Цель работы 4

1.3 Задачи 4

2 Обзор литературы **5**

3 Постановка задачи **6**

4 Библиотека программ **8**

4.1 Взаимосвязь модулей 8

4.2 Модуль полосовой фильтрации 10

4.3 Модуль вычисления коэффициентов CSP-фильтра 12

4.4 Модуль вычисления огибающей сигнала 14

4.5 Модуль расчета Движущейся волны ЭЭГ 19

4.6 Модуль отображения Движущейся волны ЭЭГ 25

4.7 Модуль обратной связи с использованием Движущейся волны ЭЭГ 28

4.8 Модуль обратной связи с использованием двух столбцов 30

4.9 Модуль связи с роботом 32

4.10 Модуль полосового частотного выделения 34

4.11 Модуль вычисления огибающей спектра 51

5 Заключение **53**

6 Список литературы **54**

1 Введение

1.1 Актуальность темы исследования

Заинтересованность в системах анализа биологических сигналов, получаемых неинвазивными методами, объясняется необходимостью создания простых и удобных в использовании методов отслеживания и управления физиологическими параметрами. Применимость таких систем очень велика, от лечебной терапии, до повышения мобильности людей с ограниченными возможностями.

1.2 Цель работы

Разработать и реализовать библиотеку программ для построения нейрообратной связи.

1.3 Задачи

Для достижения цели были поставлены следующие задачи:

- Выделить μ - ритм из одномерного сигнала, поступающего с датчика электроэнцефалографа, содержащего сумму колебаний, идущих от разных источников, расположенных по всему объему головного мозга. Содержащего в том числе и α - ритм, частота колебаний которого почти совпадает с частотой μ - ритма.

- Разработать интерфейс удобный для анализа многомерного сигнала, в частотной области, с возможностью отслеживания динамики обобщенных спектральных показателей.

- Реализовать систему вычисления и предъявления испытуемому Движущейся волны ЭЭГ в качестве интерфейса обратной связи.

- Проанализировать и модернизировать систему нейрообратной связи предыдущей версии, повысив функциональное значение модулей и прозрачность логической структуры всей системы. Кроме того, добавление комментариев и повышение качества кода.

2 Обзор литературы

Нейрообратная связь является частным случаем биологической обратной связи. Повышение интереса в научном сообществе к биологической обратной связи начинается во второй половине прошлого столетия [5]. Нейрообратная связь строится на открытии Ханса Бергера, в своей работе [6] еще в 1929 году он регистрирует колебания биопотенциала головы с частотой 8-12 Hz.

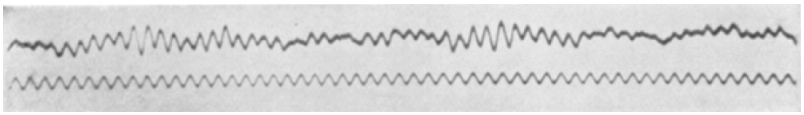


Рис. 1: Запись ЭЭГ сделанная Х. Бергером [6]

Изменение электрической активности мозга легло в основу метода нейрообратной связи. Ключевым моментом является неинвазивность подхода, поэтому в качестве метода регистрации биопотенциала выбирается электроэнцефалография. Кроме того, по сравнению с функциональной магнитно-резонансной томографией(фМРТ) и позитронно-эмиссионной томографией(ПЭТ), другими неинвазивными методами, электроэнцефалография имеет более высокое временное разрешение. Здесь стоит отметить, что существуют работы сочетающие различные методы регистрации, например, С. Фазли в своей работе [7] использовал спектроскопию в ближней инфракрасной области. А С. Уолдерт использовал магнитоэнцефалографию [8]. И все же классическим является метод электроэнцефалографии, который также использует и Фазли и Уолдерт. Этот же метод использует и Бланкерц при разработке мозг-компьютер интерфейса [9] с названием "Berlin".

Исследование электроэнцефалограммы ведется как во временной, так и в частотной областях [1]. Анализ ЭЭГ в частотной области опирается на обобщенные спектральные показатели, такие как средняя амплитуда спектра в диапазоне частот или частота максимальной по амплитуде гармоники из некоторого частотного диапазона. Также для анализа электроэнцефалограммы используют и вейвлет преобразование, переводящее сигнал во частотно-временное представление. На основе такого представления А. Е. Храмов разработал множество программ предназначенных как для

повышения качества сигнала [18], так и для различного анализа, например, оценки степени синхронизации [17]. И спектральный и вейвлет анализ направлены на изучение некоторых спектральных характеристик, для различных частотных диапазонов. Диапазоны обычно выбирают соответствующие ритмам головного мозга. Большинство работ сосредоточены на μ - ритме [13], но существуют и такие, где нейрообратная связь опирается на β - ритм [11], или их сочетание [10]. μ - ритм регистрируется над сенсомоторной корой головного мозга в состоянии покоя. Обычно, электроды электроэнцефалографа располагают одним из стандартных способов, покрывая исследуемую область, но вопрос об эффективном размещении еще не решен до конца, и Зих К. в своей работе [21] пишет об более эффективном расположении электродов для обнаружения μ-ритма. Какая либо двигательная активность или даже ее воображение уменьшает амплитуду μ - ритма. Задача построения нейрообратной связи здесь формулируется следующим образом: испытуемый должен научиться подавлять μ - ритм в некоторой области сенсомоторной коры по своему желанию. Интересная для нас обратная связь - это мобильный робот, т.е. уровень амплитуды μ - ритма в разных областях мозга отвечает за то куда повернет робот. Аналогичная идея использована в работе [22]. Обучение человека может быть довольно недолгим, согласно работе [12] В. Кирой сумел добиться качества управления компьютерным курсором с помощью α и β - ритмов на уровне 80% у 10 практически здоровых испытуемых за 2 недели обучения. Также, в центре работы [20] написанной Ли, лежит зависимость обучения различных языковых групп испытуемых, а именно говорящих на китайском и на английском языках. В работах Ли акцент сделан на использование мозг-компьютер интерфейса для повышения когнитивных способностей пожилых людей.

3 Постановка задачи

Испытуемый пытается контролировать биоэлектрическую активность своего мозга. Электроды электроэнцефалографа размещенные на его голове передают информацию о колебаниях биопотенциала головы в систему

обработки. На выходе системы обработки информация оформляется таким образом, чтобы испытуемый мог ее интерпретировать и контролировать. Кроме того, система должна быть настроена, для этого используются некоторые записанные непосредственно перед экспериментом выборки данных. Схема эксперимента представлена на Рис. 2.

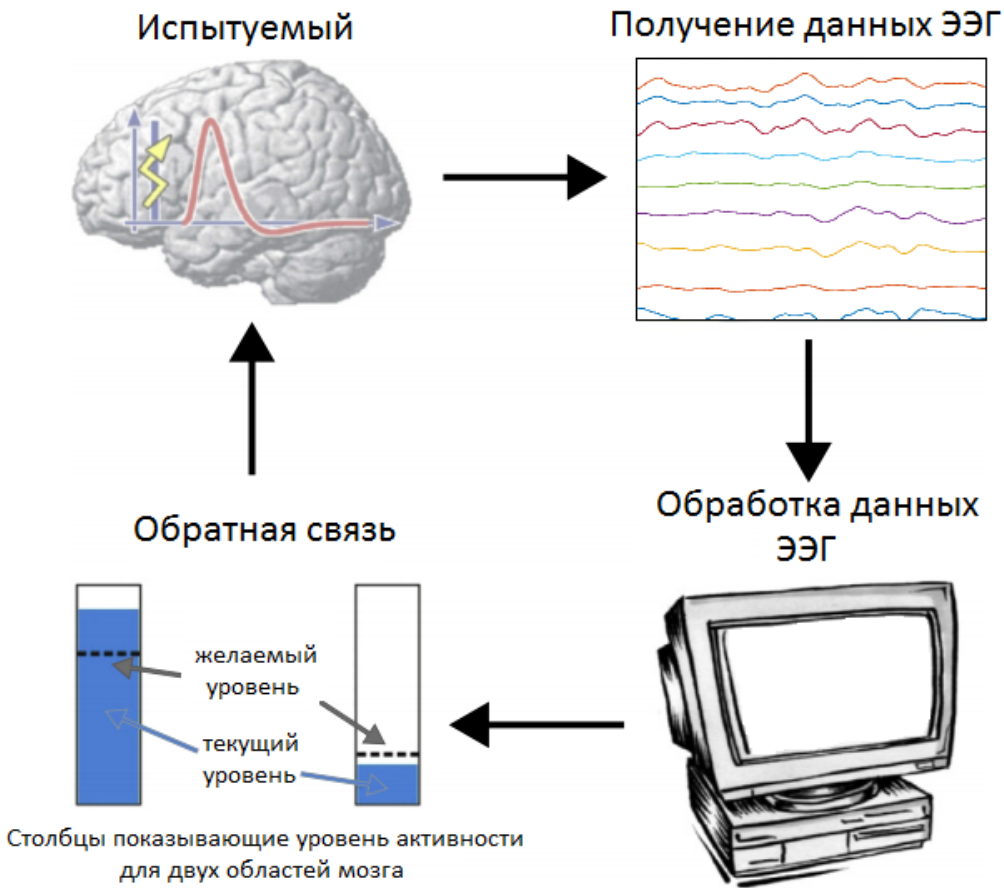


Рис. 2: Схема нейрообратной связи, адаптировано из [15]. Данные, записываемые электроэнцефалографом, шапочка которого надета на голове испытуемого, обрабатывается на вычислительном устройстве, и на основе этих обработанных данных вычисляется сигнал обратной связи, который показывается испытуемому. Испытуемый, пытается управлять изменениями сигнала обратной связи, и постепенно этому обучается.

4 Библиотека программ

4.1 Взаимосвязь модулей

Программный комплекс реализует все этапы построения нейро-обратной связи. *Модуль интерфейса* объединяет все остальные модули и связывает их в единую программу. Модули можно разделить на следующие группы:

- модули обработки сигнала ЭЭГ,
- модули отображения обратной связи,
- модули анализа сигнала ЭЭГ.

Рассмотрим группу модулей отвечающих за обработку сигнала ЭЭГ. Они, включаясь последовательно в различных комбинациях, позволяют гибко настроить систему преобразования сигнала от сырого, до желаемого. В сыром сигнале присутствуют шумы и различные артефакты, кроме этого для большинства людей в необработанном сигнале не так просто обнаружить характерные отличительные признаки для различных состояний мозга. Соответственно обработка должна начинаться с подавления шумов, например, наводки от проводов в 50 Hz и артефактов, таких как нерегулярные высокоамплитудные колебания вызванные движением глаз. Для этого применяется *модуль полосовой фильтрации*. Стоит отметить, что кроме подавления шумов *модуль полосовой фильтрации* применяется и для выделения диапазона частот интересного для исследования, например соответствующего μ-ритму. После этого сигналы в многомерном пространстве поворачиваются таким образом, чтобы между различными изучаемыми состояниями мозга наблюдалось бы наибольшее различие в дисперсии. Этот эффект достигается применением к многомерному сигналу CSP фильтра, коэффициенты которого рассчитываются специальным *модулем*. Кроме того на любом этапе обработке исследователем может быть использована огибающая сигнала, для этого просто подключается *модуль вычисления огибающей сигнала*.

После того как сигнал обработан его можно использовать, для отображения обратной связи пользователю. Библиотека программ предусматри-

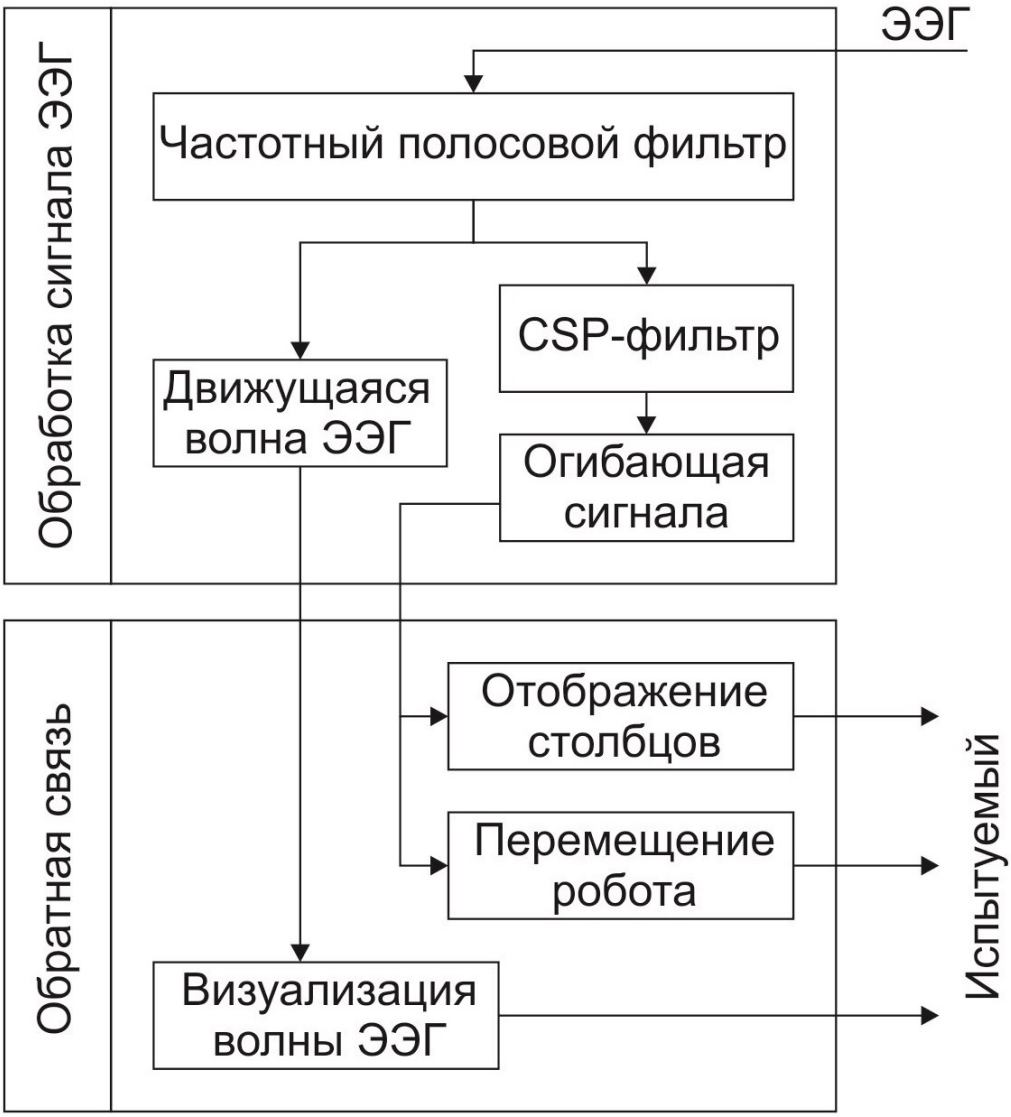


Рис. 3: Взаимосвязь модулей

вает несколько модулей для отображения обратной связи. Согласно изначальной идее испытуемый должен управлять роботом, но в ходе исследования понадобились и другие методы представления. Самый простой способ отображения обратной связи это столбец показывающий на сколько характерный признак записываемого сигнала совпадает с этой же характеристикой записанного сигнала. *Модуль обратной связи с использованием двух столбцов* показывает испытуемому два столбца которые он должен научиться изменять по своей воле. Другим способом отображения является *модуль отображения Движущейся волны ЭЭГ*. Здесь испытуемый пытается контролировать цветную картинку или векторное поле. Желаемый

результат при таком методе обратной связи еще исследуется, но удержание некоторой стабильно картины кажется вполне подходящим. Возвращаясь к управлению роботом упомянем *модуль связи с роботом*. Испытуемый наблюдая за мобильным роботом, учится контролировать его перемещения.

Перечисленные выше модули составляют полный цикл нейро-обратной связи. Схема связи этих модулей между собой показана на Рис. 3. Кроме них в библиотеке есть несколько модулей предназначенных для анализа данных. *Модуль* полосового частотного выделения позволяет исследовать энергию в выбираемом пользователем диапазоне частот. *Модуль вычисления огибающей спектра* облегчает исследователю восприятие спектра.

4.2 Модуль полосовой фильтрации

Позволяет выбрать настройки фильтрации. В зависимости от длины линии задержки можно выбрать либо полосовой фильтр Чебышева первого рода или фильтрацию путем присвоения какой то части спектра нулевых значений. Если система работает в реальном времени, то лучше использовать вариант фильтра с быстрым преобразованием Фурье. Или заранее настроить фильтр Чебышева, и получив коэффициенты использовать их в функции “filter”, чтобы не рассчитывать коэффициенты фильтра на каждом шаге обработке.

```
function [ out_data, coef_a, coef_b ] =
   bandpass_filter( in_data, f1, f2, sr, type )
%Частотный полосовой фильтр
%
% Аргументы :
% in_data - данные которые надо отфильтровать
% f1 - частота начала полосы выделения в Hz
% f2 - частота окончания полосы выделения в Hz
% sr - частота дискретизации
%
% Возвращаемые значения
```

```
% out_data - данные после фильтрации
% coef_a, coef_b - коэффициенты фильтра
%

switch type
    case 'fft'
        % Реализовация через
        % быстрое переобразование Фурье.

        % считаем преобразование Фурье
        f = fft(in_data);
        len = length(f);
        % зануляем часть спектра лежащую вне полосы
        f(1:round(len * f1 / sr)) = 0;
        % поскольку сигнал был вещественен,
        % то его спектр симетричен надо сохранить
        % это свойство, чтобы востановленный
        % сигнал был вещественен. Поэтому в
        % следующей строке есть добавка "+2"
        f(round(len*f2/sr):end-round(len*f2/sr)+2)=0;
        f(end - round(len * f1 / sr) + 2 : end) = 0;
        % обратное преобразование Фурье
        % возвращает отфильтрованные данные
        out_data = ifft(f);
    case 'cheb'
        % реализация через фильтр
        % Чебышева первого рода
        des_filt = designfilt('bandpassiir',...
            'FilterOrder',6, ...
            'PassbandFrequency1',f1,...
            'PassbandFrequency2',f2, ...
            'PassbandRipple',4,...
            'SampleRate',sr);
```

```
            % Коэффициенты фильтра
            [coef_b, coef_a] = tf(des_filt);

            % Отфильтрованный данные
            out_data = filter(des_filt, in_data);
end

end
```

4.3 Модуль вычисления коэффициентов CSP-фильтра

Различные ментальные задачи, например, представление движения правой рукой, левой рукой или непосредственное движение какой либо частью тела вызывают уменьшение амплитуды μ-ритма в определенной области головного мозга. Соответственно целая группа датчиков электроэнцефалографа содержат части изучаемого одномерного сигнала, кроме того на электроде находящемся ближе всего к интересующей нас области присутствуют дополнительные колебания не имеющие отношения к изучаемому процессу, например, μ-ритму. Для того чтобы сделать сигнал более информативным для анализа, применяют CSP-фильтр [4]. Common spatial pattern (CSP) - метод используемый в обработке сигналов для разделения многомерного сигнала на аддитивные компоненты, которые имеют максимальные различия в дисперсии между двумя окнами.

Описание алгоритма:

X_1 и X_2 временные ряды (окна размером t_1 и t_2) соответствующие различным ментальным задачам. N - количество каналов ЭЭГ.

$$C_{10} = \frac{X_1 X_1^*}{t_1}, \quad C_1 = C_{10} + \frac{\lambda \mathrm{tr} C_{10}}{N} I_N$$
$$C_{20} = \frac{X_2 X_2^*}{t_2}, \quad C_2 = C_{20} + \frac{\lambda \mathrm{tr} C_{20}}{N} I_N$$

Решая матричное уравнение $C_1 v = \alpha C_2 v$ можно найти собственные числа α и вектора v. Собственный вектор соответствующий наибольшему соб-

ственному числу принимается за вектор коэффициентов CSP-фильтра.

```
function [ V ] = calculate_coeff_csp_filter(data1,
   data2, lambda)
% Функция вычисляет коэффициенты
% пространственного фильтра.
%
% Аргументы :
% data1 - данные ЭЭГ соответствующие
%       первому типу активности
% data2 - данные ЭЭГ соответствующие
%       второму типу активности
% lambda - коэффициент усиления
%       диагонали матриц ковариации
%
% Возвращаемые значения :
% V - матрица коэффициентов csp-фильтра

% вычисление матриц ковариации
C10 = data1 * data1'/fix(size(data1,2));
C20 = data2 * data2'/fix(size(data2,2));

% количество каналов
nch = size(C10,1);
C1 = C10 + lambda * trace(C10) * eye(nch) / nch;
C2 = C20 + lambda * trace(C20) * eye(nch) / nch;

% решение матричного уравнения
[V, ~] = eig(C1,C2);

end
```

4.4 Модуль вычисления огибающей сигнала

Многомерные данные обработку которых ведет разрабатываемый программный комплекс состоят из набора одномерных сигналов. Во время исследования электроэнцефалограммы огибающая сигнала может оказаться более информативной чем сам сигнал. Например, при изучении периодических сигналов динамика которых велика. Полезность огибающей проявляется благодаря двум свойствам:

- Огибающая отражает динамику амплитудной модуляции сигнала.
- Площадь под фрагментом огибающей отражает мощность сигнала на этом фрагменте.

Отметим, что огибающая имеет и важный физиологический смысл [1]: она показывает степень синхронности в изменении постсинаптических потенциалов.

Описание алгоритма:

В общем случае огибающая определяется как модуль комплекснозначной функции $s_{env}(t) = s(t) + iH(s(t))$, где $s(t)$ - вещественный сигнал, а H - преобразование Гильберта. Преобразование Гильберта имеет довольно простую связь с преобразованием Фурье:

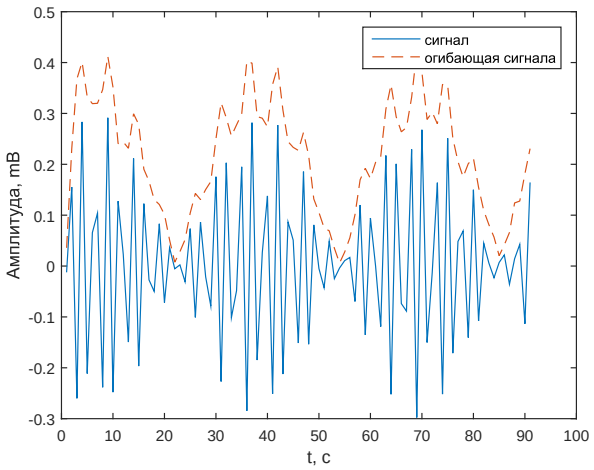
$$\mathcal{F}\big(H(u)\big)(\omega) = -i\text{sign}(\omega) \cdot \mathcal{F}(u)(\omega)$$

тогда, для огибающей:

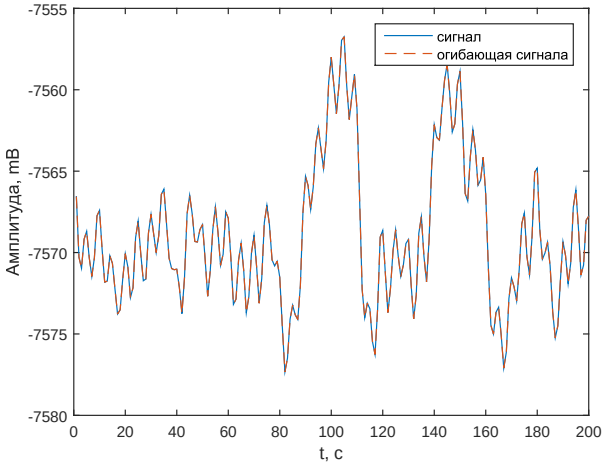
$$\begin{aligned} \mathcal{F}(s_{env})(\omega) &= \mathcal{F}(s)(\omega) + i\mathcal{F}\big(H(s)\big)(\omega) = \mathcal{F}(s)(\omega) - i^2\text{sign}(\omega) \cdot \mathcal{F}(s)(\omega) = \\ &= \mathcal{F}(s)(\omega) + \text{sign}(\omega) \cdot \mathcal{F}(s)(\omega) = (1 + \text{sign}(\omega))\mathcal{F}(s)(\omega) \end{aligned}$$

Здесь видно, что спектр огибающей на положительных частотах ω совпадает с удвоенным спектром сигнала, а на отрицательных он тождественный ноль.

Однако построенная таким образом огибающая не дает необходимый результат при анализе сигналов ЭЭГ. Если мы посмотрим на приведенные на Рис. 4 графики, то хорошо работающая на обычных амплитудно-модулированных сигналах огибающая начинает просто повторять форму



A. Сигнал $0.3\sin(t)\sin(100t+10)$



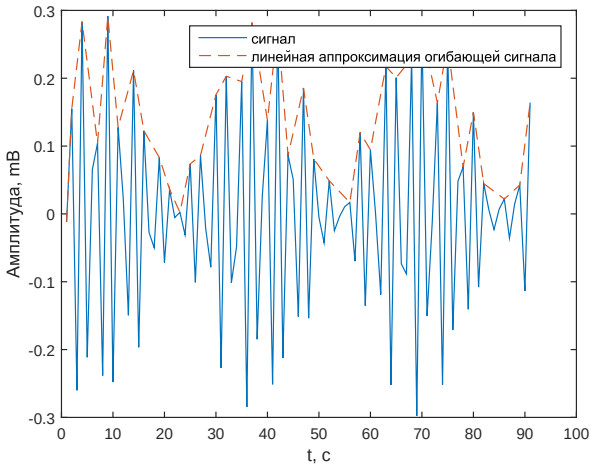
B. Одноканальный сигнал ЭЭГ

Рис. 4: Сравнение применимости огибающей полученной стандартным методом. На изображении **A** огибающая рассчитана для сигнала $0.3\sin(t)\sin(100t+10)$, который является амплитудно-модулированным и спектр его довольно узкий. Сигналы с электроэнцефалографа в сыром виде, представленные на **B**, обладают широким спектром, и стандартный метод не справляется с поставленной задачей, начиная просто повторять каждое колебание.

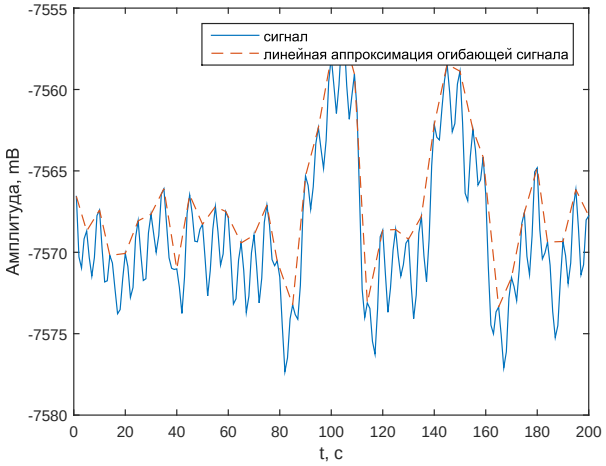
сигнала в каждом значении. Это объясняется тем, что такой метод рассчитан в лучшем случае на сигналы с узким спектром.

Линейная аппроксимация огибающей:

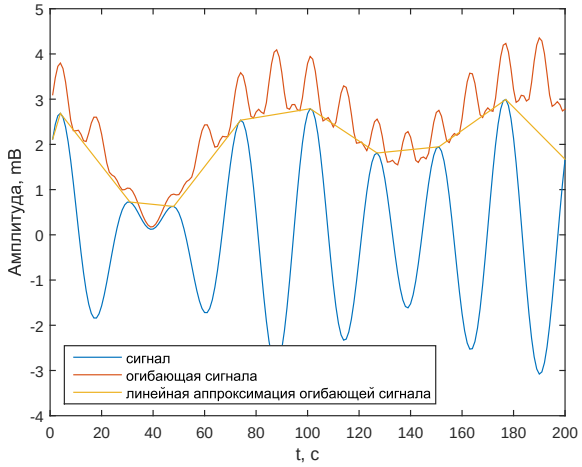
Как известно, верхняя огибающая сигнала проходит через все локальные максимумы. Поэтому используя производную мы с легкостью можем найти координаты этих максимумов и, не особо заботясь о гладкости, линейно интерполировать найденные точки (поскольку и искомый сигнал не был гладким). Назовем полученную функцию линейной аппроксимацией огибающей. Учитывая то, что согласно схеме (Рис.3) применение огибающей требуется уже после полосовой фильтрации получим и для метода с использованием преобразования Гильберта адекватные результаты. Однако здесь стоит отметить, что согласно работе Сметанина [10] при вычислении огибающей в аналогичной задаче в лучшем случае необходимо около 0,1с. Вычисление линейной аппроксимации огибающей требует того же времени, по той причине, что изучаемый ритм имеет частоту колебаний около 10 Hz.



A. Сигнал $0.3\sin(t)\sin(100t+10)$



B. Одноканальный сигнал ЭЭГ



C. Отфильтрованный в полосе 8-12 Hz сигнал ЭЭГ

Рис. 5: Сравнение применимости линейной аппроксимации огибающей (**A**, **B**) и сравнение огибающей с ее линейной аппроксимацией (на **C**). Видно что, линейная аппроксимация дает хороший результат и для сигналов с широким спектром (на **B**) и для амплитудно-модулированных сигналов с узким спектром (на **A**).

```
function [ out_h, out_l ] = envelopes( x, type )
% функция вычислет верхнюю и нижнюю
% огибающие входного сигнала
% Аргументы :
% x - одномерный сигнал
```

```
% type - тип алгоритма используемый для вычисления
   огибающей
%             возможные значения: 'hilbert', 'interp'
%
% Возвращаемые значения :
% out_h - вершняя огибающая
% out_l - нижняя огибающая

switch type
    case 'hilbert'
        % реализация через преобразование Гильберта
        out_h = abs(sqrt(x.^2 + hilbert(x).^2));
        out_l = -out_h;
    case 'interp'
        % реализация через линейную
        % аппроксимацию огибающей

        % вычисление производной
        dx = x(2:end) - x(1:end-1);


        % находим экстремумы
        z = [];
        for k = 2:length(dx)
            if dx(k-1) * dx(k)<0
                % здесь позиции экстремумов
                z = [z k];
            end
        end


        % Полученную сетку позиций экстремумов надо
        % разделить на максимумы и минимумы.
```

```
% Предположим, что первый из экстреммумов -
  максимум
% если это не так, то в конце поменяем
  местами.
% разделяем на максимумы
hz = z(1:2:end);
% и минимумы
lz = z(2:2:end);

% Для правильного интерполирования надо
  добавить
% начало и конец в сетку
if hz(1) ~= 1
    hz = [1 hz];
end
if hz(end) ~= length(x)
    hz = [hz length(x)];
end
if lz(1) ~= 1
    lz = [1 lz];
end
if lz(end) ~= length(x)
    lz = [lz length(x)];
end

% интерполируем максимумы и минимумы отдельно
%получая верхнюю и  нижнюю огибающие
out_h = interp1(hz,x(hz), [1:length(x)]);
out_l = interp1(lz,x(lz), [1:length(x)]);

% Поскольку проверки на кратность не было, то
   верхняя
```

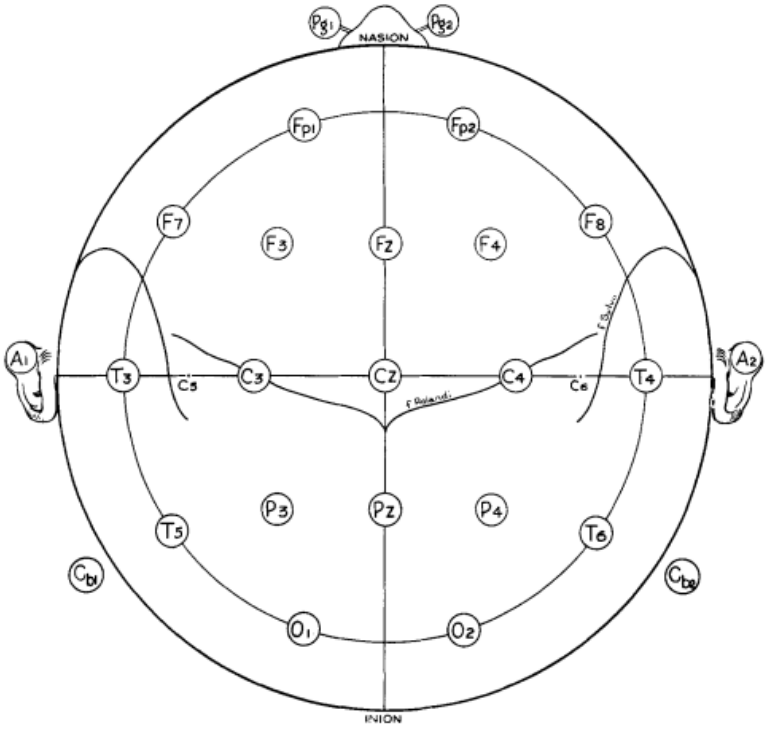
```
        % огибающаяя может превращаться в нижнюю и
          наоборот
        % в точках, где вторая производная обращается
           в ноль.
        % Чтобы избежать этого поменяем местами части
        % огибающих, в которых нижняя больше верхней.
        for k = 1 : length(x)
            if out_h(k) < out_l(k)
                tmp = out_h(k);
                out_h(k) = out_l(k);
                out_l(k) = tmp;
            end
        end
end
end
```

4.5 Модуль расчета Движущейся волны ЭЭГ

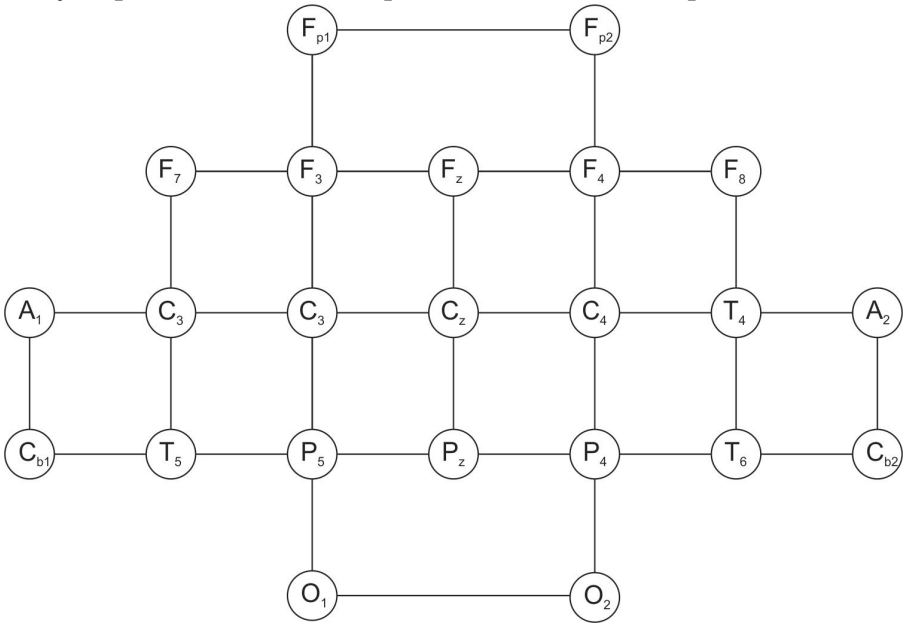
Рассмотрим данные поступающие с электроэнцефалогафа - это многомерный сигнал размерности, соответствующей количеству сенсоров электроэнцефалографа расположенных на голове испытуемого. В этом сигнале отражены колебания биопотенциала поверхности головы. Можно предположить, что существует некоторая движущаяся волна, переливы которой можно наблюдать вычислив фазовый сдвиг между электродами. По методике предложенной [2] можно представить в довольно хорошем виде данные о движении этой волны.

Описание алгоритма:

По голове испытуемого расположены электроды по какой либо общепринятой схеме, например, схема 10-20 (Рис. 6). Будем считать, что любую интересную для изучения схему можно преобразовать без изменения топологии к некоторому набору точек расположенных на плоскости с целочисленными координатами.



A. Международная система размещения электродов «10—20» [16]



B. Выровненная схема «10—20»

Рис. 6: Пример стандартного расположения электородов ЭЭГ (**A**) и его преобразование к сетке прямоугольной формы (**B**). Электроды F7, F3, C3, T3 не образуют прямоугольник, но этот четырех угольник без изменения топологии можно превратить в прямоугольник, тоже самое можно сказать и про остальные электроды. Схема «10—20» после такого преобразования изображена на **B**.

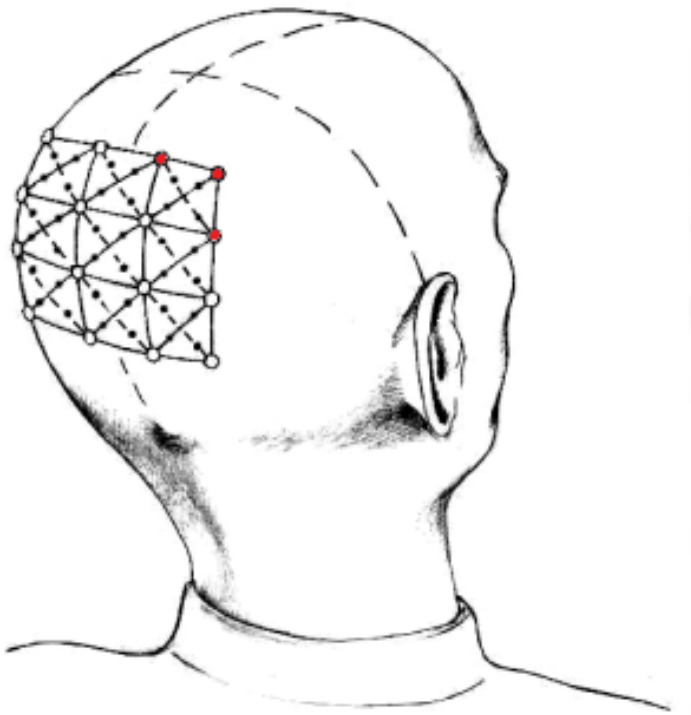


Рис. 7: Иллюстрация возможного расположения электродов электроэнцефалографа на голове испытуемого. Электроды образуют прямоугольную сетку. Три электрода выделенные красным цветом образуют прямоугольный треугольник. [2]

Рассмотрим любой электрод, если у него есть соседи по вертикали и горизонтали, то он вместе с соседями образует прямоугольный треугольник (см. рис. 7). Для каждого катета такого треугольника можно рассчитать фазовый сдвиг, как отклонение от центра максимума взаимной корреляции сигналов идущих с электродов, которые находятся в вершинах треугольника. Отметим, что количество данных для расчета взаимной корреляции здесь должно быть выбрано осознанно в зависимости от того процесса который изучается. В нашем случае речь идет про μ-ритм, т.е. размер линии задержки около 0.1 секунды. Эти сдвиги могут быть любого знака, и являются проекциями вектора указывающего на направление движения волны в данной точке. Для удобства перенесем этот вектор в центр окружности, вписанной в этот треугольник. Стоит отметить, что для квадрата, состоящего из четырех электродов, существуют четыре таких треугольника. Таким образом внутри каждого треугольника один вектор направления

распространения волны, а внутри квадрата их четыре.

```
function [ out_data ] = calculate_wave_eeg( data,
   chan_pos )
% Функция вычисляет векторное поле,
% изображающее движение волны ЭЭГ.
%
% Аргументы :
% data - многомерный сигнал ЭЭГ,
%       двумерный массив в котором
%       строки это одномерные сигналы
% chan_pos - двумерный массив, значения
%       в котором это номера каналов в порядке
%       их следования в data, а индексы для этих
%       значений выбираются в соответствии с
%       расположением датчиков. Если есть ячейка
%       массива которой не соответствует ни какой
%       канал, то туда следует записать "-1".
%       Пример такого массива
%       [-1  7   8  17  -1; %    Fp1 Fpz Fp2
%        3   4 13  20 21; % F7 F3 FZ F4 F 8
%        2   5  11 19 22; % T3 C3 Cz C4 T4
%        1   6  14 18 23; % T5 P3 Pz P4 T6
%       -1 10 12 15  -1;] %      O1 Oz O2
%
% Возвращаемые значения :
% out_data - массив комплексных чисел
%       соответствующих векторам направления
%       распространения волны. Полученная
%       матрица в два раза больше каждого
%       размера матрицы каналов, что
%       соответствует тому, что на каждый
%       канал не на краю приходится 4 вектора
%       сдвигов фазы. Их следует разнести
```

```
%       пространственно по диагоналям
%       от соответствующих датчиков.

% размер эпохи в сэмплах
len = size(data,2);

sz1 = size(chan_pos,1);
sz2 = size(chan_pos,2);

% буферный массив для данных длинной
% в одну эпоху
channels = zeros(sz1, sz2, len);

for k = 1 : sz1
    for p = 1 : sz2
        if chan_pos(k, p) ~= -1
            channels(k,p,:) = data(chan_pos(k, p),:);
        end
    end
end
% считаем разность фаз
% dvp_arr - delta_vertical_phase_array
dvp_arr = zeros(sz1-1,sz2);
% dhp_arr - delta_horizontal_phase_array
dhp_arr = zeros(sz1, sz2-1);
% w_v - weight_vertical
w_v = zeros(sz1-1,sz2);
% w_h - weight_horizontal
w_h = zeros(sz1,sz2-1);

% Сдвиг фаз между каждой парой
% ближайших датчиков, в каждой строке
tmp = 0;
```

```
for p = 1 : sz1-1
    for k = 1 : sz2
        [w_h(k,p), tmp] = max(abs(xcorr_pirs(...
            channels(k, p, :),channels(k, p+1, :))));
        if w_h(k,p) ~= 0
            dhp_arr(k,p) = tmp - len;
        end
    end
end

% Сдвиг фаз между каждой парой
% ближайших датчиков, в каждом столбце
tmp = 0;
for p = 1 : sz1
    for k = 1 : sz2-1
        [w_v(k,p), tmp] = max(abs(xcorr_pirs(...
            channels(k, p, :),channels(k+1, p, :))));
        if w_v(k,p) ~= 0
            dvp_arr(k,p) = tmp - len;
        end
    end
end

out_data = zeros(sz1*2-2, sz2*2-2);
% сдвиг от электрода вправо и вниз
out_data(1:2:end,1:2:end) =...
    dhp_arr(1:end-1,:)+i*dvp_arr(:,1:end-1);
% сдвиг от электрода влево и вниз
out_data(1:2:end,2:2:end) =...
    dhp_arr(1:end-1,:)+i*dvp_arr(:,2:end);
% сдвиг от электрода вправо и вверх
out_data(2:2:end,1:2:end) =...
    dhp_arr( 2:end, : )+i*dvp_arr(:,1:end-1);
```

```
% сдвиг от электрода влево и верх
out_data(2:2:end,2:2:end) =...
    dhp_arr( 2:end, : )+i*dvp_arr(:,2:end);

end

function [out] = xcorr_pirs(x,y)
% Функция вычисляет взаимную корреляцию
% используя коэффициент корелляции Пирсона
%
% Аргументы:
% x,y - два одномерных массива, между
%       которыми вычисляется взаимная корреляция
%
% Возвращаемое значение:
% out - массив длиной max(length(x),length(y))*2-1
%
x = x(:); y = y(:);
    if sum((x).^2)*sum((y).^2) ~= 0
        out = xcorr(x,y)/sqrt(sum((x).^2)*sum((y).^2)
          );
    else
        out = zeros(1,max(length(x),length(y))*2-1);
    end
end
```

4.6 Модуль отображения Движущейся волны ЭЭГ

Вектора описываемые в *модуле вычисления Движущейся волны ЭЭГ* образуют полную картину направлений движения волны ЭЭГ. Полученное векторное поле уже так можно отображать на экране испытуемому, но существует и другая форма представления этой информации. Вычислив дивергенцию по этому векторному полю, мы получим набор изолиний, ко-

торые например могут быть покрашены в соответствии с их высотной, что представляется более наглядно. Этот модуль, опирается на модуль расчета Движущейся волны и является частью интерфейса для него.

```
function [ ] = show_wave_eeg(vect_field, type )
% Функция для отображения движущейся
% волны ЭЭГ.
%
% Аргументы :
% vect_field - двумерный массив векторов,
%       векторов который возвращает
%       функция "calculate_wave_eeg".
% type - позволяет выбрать тип отображения,
%        возможные значения: 'field','div'
% chan_coord - вектор координат электродов
%

sz1 = size(vect_field, 1)/2+1;
sz2 = size(vect_field, 2)/2+1;

% массив размером sz1/2 на sz2/2
v0 = zeros(sz1,sz2);
s1 = sz1+1;
s2 = sz2+1;
for k = 1 : sz1
    v0(k,:) = k/s1 + i*[1:sz2]/s2;
end

v1 = v0(1:end-1,1:end-1);
v_base =...
    [(v1+0.3/s1+i*0.3/s2)  (v1+0.7/s1+i*0.3/s2);...
     (v1+0.3/s1+i*0.7/s2)  (v1+0.7/s1+i*0.7/s2)];

v_plot = vect_field+v_base;
```

```
% отрисовка квадрата ограничивающего
% рабочую область
plot([0 1 1 0 0],[0 0 1 1 0])
hold on
switch type
    case 'field'
        % векторное поле отображается как
        % набор векторов размещенных вблизи
        % соответствующих электродов

        % звездочками нарисован
        % массив электродов
        plot(v0,'r*')
        % все вектора рисуются по очереди
        for k = 1 : size(vect_field,1)
            for p = 1: size(vect_field,2)
                plot(real([v_plot(k,p) v_base(k,p)]),
                  ...
                    imag([v_plot(k,p) v_base(k,p)]),'
                      b')
            end
        end
    case 'div'
        % на основе векторного поля строится
        % карта стоков и истоков

        % вычисление дивиргенции
        div = divergence(real(v_plot),imag(v_plot));
        %ds - div_size
        ds1 = size(div,1);
        ds2 = size(div,2);
        % отображение карты высот
```

```
        contourf( [0:ds2-1] / (ds2-1), [0:ds1-1] /(
          ds1-1), div)
end
hold off

end
```

4.7 Модуль обратной связи с использованием Движущейся волны ЭЭГ

Модуль разработан для повышения "структурной ясности" алгоритма обратной связи. Он объединяет модули *вычисления* и *отображения* для Движущейся волны ЭЭГ в одну цельную систему, которую можно подключить и в режиме реального времени и в системе постобработки, необходимой для анализа.

Учитывая, что длина линии задержки для данного алгоритма довольно мала, то показ данных испытуемому в любом из вариантов приведенных в описаниях связанных модулей будет выглядеть слишком быстро меняющимся, поэтому требуется некоторое усреднение во времени полученных фазовых сдвигов.

Модуль может использоваться для отображения нейро-обратной связи. В данном случае испытуемый должен по изображению движущейся волны (ее источников и стоков) пытаться контролировать состояние мозга, например пытаться добиться неизменной во времени картины на экране или ее изменения по какому то закону.

```
function [ ] = feedback_wave_eeg( data, chan_pos,
  mean_buf, weight_coef )
% Функция обратной связи с
% использованием волны ЭЭГ.
%
% Аргументы :
% data - данные ЭЭГ со всех каналов
```

```
%       длиной в одну эпоху
% chan_pos - двумерный массив, значения
%       в котором это номера каналов в порядке
%       их следования в data, а индексы для этих
%       значений выбираются в соответствии с
%       расположением датчиков. Если есть ячейка
%       массива которой не соответствует ни какой
%       канал, то туда следует записать "-1".
%       Пример такого массива
%       [-1  7   8  17  -1; %    Fp1 Fpz Fp2
%        3  4 13  20 21; % F7 F3 FZ F4 F 8
%        2  5  11 19 22; % T3 C3 Cz C4 T4
%        1  6  14 18 23; % T5 P3 Pz P4 T6
%       -1 10 12 15  -1;] %      O1 Oz O2
% mean_buf - циклический буферный массив
%       для усреднения полученных векторoных
%       полей с нескольких эпох
% weight_coef - коэффициент масштабирующий
%       вектора при отображении
%

mean_len = length(mean_buf);

% вычисление векторного поля (волны ЭЭГ)
vf = calculate_wave_eeg(data,chan_pos);

% сдвиг циклического буфера
mean_buf(:,:,1:end-1) = mean_buf(:,:,2:end);
% запись в конец циклического буфера
mean_buf(:,:,end) = vf;

% вычисление среднего векторного поля
mean_vf = zeros(size(vf,1),size(vf,2));
```

```
for k = 1 : mean_len
    mean_vf = mean_vf + mean_buf(:,:,k);
end
mean_vf = mean_vf ./ mean_len;

% отрисовка векторного поля
show_wave_eeg(weight_coef.*mean_vf, 'div');

% в принудительном порядке вывести
% на экран изображение
drawnow

end
```

4.8 Модуль обратной связи с использованием двух столбцов

Модуль имеет два входных канала и показывает уровни амплитуд входных сигналов. Конечно возможен вариант подачи данных просто из двух каналов ЭЭГ, но такие сигналы сильно зависимы от посторонних колебаний. Поэтому лучше использовать CSP-фильтр, который по нескольким выбранным каналам, соответствующим некоторой области производит разложение на аддитивные составляющие. И сигналы на выходе CSP-фильтров настроенных на двух различных областях мозга можно уже использовать. Для повышения качества управления стоит для этих сигналов вычислять огибающую, и ее динамику уже показывать испытуемому. Итак, перед испытуемым на экране отображаются два столбца, уровень которых зависит от активности мозга в выбранных исследователем областях. Например, можно выбрать области сенсомоторной коры головного мозга, которые отвечают за контроль правой и левой руки. Некоторые выполняемые движения или представления движений вызывают понижение μ-ритма в этих областях: для правой руки левая область и наоборот. С помощью некоторого случайного поведения испытуемые учится контроли-

ровать высоту столбцов, т.е. уменьшать и увеличивать по своему желанию. Тут конечно есть рекомендация какое поведение следует брать случайным, например, воображение движения рукой.

```
function [ ] = two_bars_plot( left, right )
% Функция для показа испытуемому двух
% столбцов.
%
% Аргументы :
% left, right - сигналы обратной связи с помощью
%       которых вычисляются высоты столбцов.
%

% t_bar_fig - tag_bars_figure
% h_fig - hangle_figure
h_fig = findobj('Tag','t_bar_fig');
% если такого окна нет, то надо его создать
if ~ishandle(h_fig)
    h_fig = figure('Tag','t_bar_fig','NumberTitle',...
        'off','Name','Уровень активности');
    % h_l_bar - handle_left_bar
    h_l_bar = subplot(1,2,1);
    % h_r_bar - handle_right_bar
    h_r_bar = subplot(1,2,2);
    % отрисовка двух столбцов высотой 0
    bar1 = bar(h_l_bar,0);
    bar2 = bar(h_r_bar,0);
    % закрепление осей графиков
    h_l_bar.YLim = [-3 3];
    h_r_bar.YLim = [-3 3];
    h_l_bar.YLimMode = 'manual';
    h_r_bar.YLimMode = 'manual';
end
% изменение высот стобцов
```

```
bar1.YData = left;
bar2.YData = right;
% добавление надписей показывающих
% значение в соответствующих столбцах
text('Parent',left_bar,'String',num2str(left),...
    'Tag','t_bar_l_text','Position', [ 1 -2]);
text('Parent',right_bar,'String',num2str(right),...
    'Tag','t_bar_r_text','Position', [ 1 -2]);
end
```

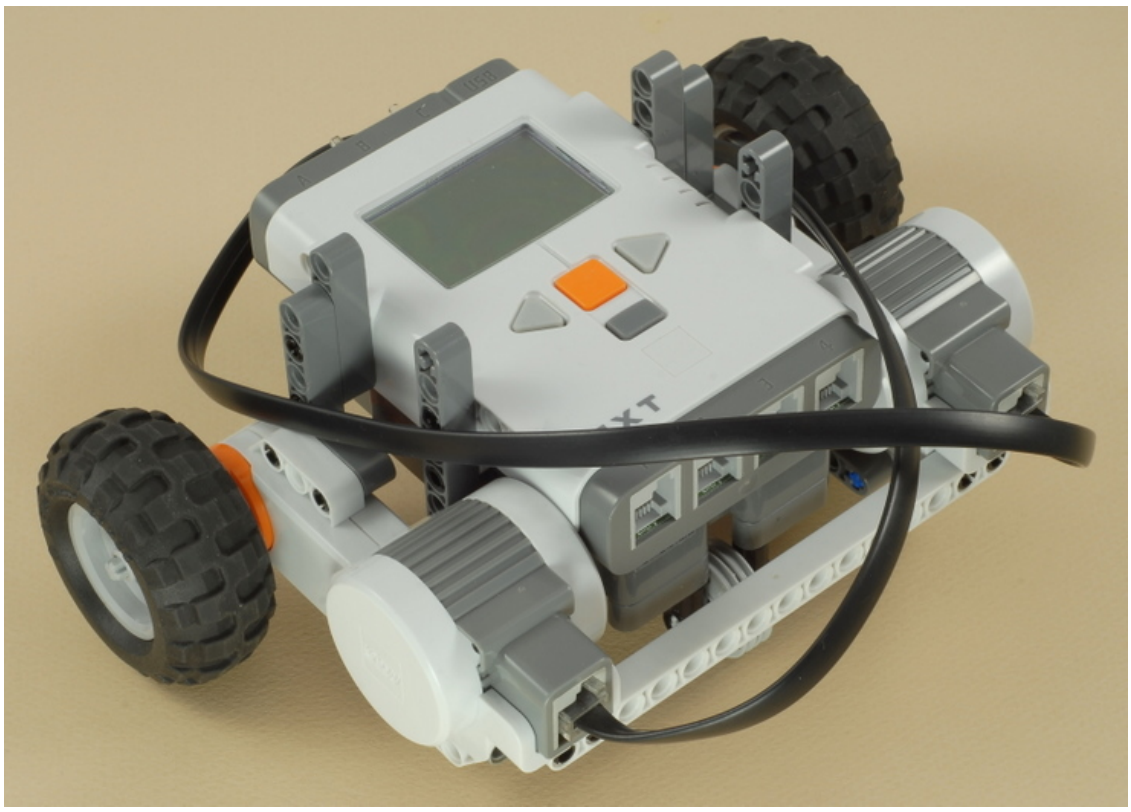


Рис. 8: Мобильный робот

4.9 Модуль связи с роботом

В этом модуле как и в *модуле обратной связи с использованием двух столбцов* есть два входа. Рекомендации по выбору данных следует взять из этого же модуля. Однако теперь объектом управления становиться не набор столбцов, а мобильный робот. Мобильный робот представлен на Рис:8

Этот модуль поддерживает соединение с роботом по каналу беспроводной связи Bluetooth и передает на него управление.

```
function [ ] = transmit_robot_bluetooth( left, right,
    l_const,r_const )
% Функция для передачи сигналов на робота
% с использованием bluetooth, до ее исполнения
% должна быть вызвана функция подключения
% к роботу.
%
% Аргументы :
% left, right - сигналы обратной связи с помощью
%       которых вычисляются управляющие
%       сигналы для левого и правого колес
%       робота соответственно.
% l_const, r_const - ограничения на уровень
%       сигналов left и right соответственно.
%       Эти константы лучше всего вычилсять
%       во время обучения csp-фильтра.
%

% bt_o - bluetooth_object
bt_o = findobj('Tag','t_robot');

% проверка на подключение
if ~strcmp(bt_o.Status,'closed');

    % если величина обратной связи в диапазоне,
    % то управляющий сигнал для левого колеса ch1=0.4
    if left > 0 && left < l_const
        ch1 = 0.4;
    else
        ch1 = 0;
    end
```

```
    % для правого аналогично
    if right > 0 && right < r_const
        ch2 = 0.4;
    else
        ch2 = 0;
    end

    % преобразование данных в формат
    % подходящий для робота nxt
    out1 = uint8((ch1+4)*32);
    out2 = uint8((ch2+4)*32);
    bytes = uint8([6 0 128 9 0 2 out1 out2]);

    % отправка данных по блютусу
    fwrite(self.bluetooth_connection,bytes);
end

end
```

4.10 Модуль полосового частотного выделения

Модуль предназначен для исследования записанного сигнала с электроэнцефалографа. Модуль позволяет отслеживать изменение во времени энергии соответствующей некоторому диапазону частот. Также этот параметр может быть представлен в нескольких вариантах: нормированный на длину диапазона, нормированный на полную энергию. В процессе исследования может потребоваться сохранить фильтр, выделяющий полосу частот для использования в последующей обработке. В интерфейсе модуля предусмотрена и эта возможность, позволяющая более точно настроить систему обработки сигналов на конкретного испытуемого.

```
function [] = interface_analysis_mean_freqence_diap(
  data,sample_rate)
% Функция создания всех элементов
```

```
% интерфейса
%
% Аргументы :
% data - данные ЭЭГ
% sample_rate - частота дискретизации
%

%-------------------------------------------------
% главное окно интерфейса

% h_fig_ia - handle_figure_interface_analysis
h_fig_ia = figure('Name','Анализ',...
    'NumberTitle','off','Tag','t_ifw');
% wsz - window_size
wsz = get(h_fig_ia, 'Position');
% вектор координат и размеров для
% правильного отображения элементов
% при масштабировании
% w_pos - window_positions
w_pos = [wsz(3) wsz(4) wsz(3) wsz(4)];
% функция изменения размера окна,
% чтобы элементы не уезжали
set(h_fig_ia,'SizeChangedFcn',@resize_fig_ia);


%-------------------------------------------------
% Выпадающее меню позволяющее
% выбрать канал для анализа

% h_t_p - handle_text_popup
h_t_p = uicontrol('Parent',h_fig_ia,'Style','Text',...
    'String','Канал ЭЭГ','Tag','t_t_p',...
    'Position',[0.01 0.95 0.09 0.041].*w_pos);
```

```
% h_p - handle_popup
h_p = uicontrol('Parent',h_fig_ia,'Style','popup',...
    'String', {'T5','T3','F7','F3','C3',...
    'P3','Fp1','Fpz','A1','O1','Cz',...
    'Oz','Fz','Pz','O2','A2','Fp2',...
    'P4','C4','F4','F8','T4','T6','AUX'},...
    'Position', [0.01 0.91 0.09 0.041].*w_pos,...
    'Callback', {@cb_popup,data,sample_rate},...
    'Tag','t_p');

%-------------------------------------------------
% выбор начала выделения данных

% h_t_pos - handle_text_position
h_t_pos = uicontrol('Parent',h_fig_ia,...
    'Style','Text','Tag','t_t_pos',...
    'String','Начало выделения данных',...
    'Position',[0.01 0.85 0.09 0.041].*w_pos);
% h_e_pos - handle_edit_position
h_e_pos = uicontrol('Parent',h_fig_ia,...
    'Style','Edit','String','1','Tag','t_e_pos',...
    'Position',[0.01 0.81 0.09 0.041].*w_pos,...
    'Callback',{@cb_e_pos,data,sample_rate});

%-------------------------------------------------
% выбор размера окна выделения

% h_t_ws - handle_text_window_size
h_t_ws = uicontrol('Parent',h_fig_ia,...
    'Style','Text','Tag','t_t_ws',...
    'String','Размер окна выделения',...
    'Position',[0.01 0.75 0.09 0.041].*w_pos);
% h_e_ws - handle_edit_window_size
```

```
h_e_ws = uicontrol('Parent',h_fig_ia,...
    'Style','Edit','String','1024','Tag','t_e_ws',...
    'Position',[0.01 0.71 0.09 0.041].*w_pos,...
    'Callback',{@cb_e_pos,data,sample_rate});

%-------------------------------------------------
% Выпадающее меню выбора типа нормировки

% h_t_p_type - handle_text_popup_type
h_t_p_type = uicontrol('Parent',h_fig_ia,...
    'Style','Text','String','Тип нормировки',...
    'Tag','t_t_p_type','Position',...
    [0.01 0.51 0.09 0.041].*w_pos);
% h_p_type - handle_popup_type
h_p_type = uicontrol('Parent',h_fig_ia,...
    'Style','popup',...
    'String', {'без нормировки','с нормировкой'},...
    'Position', [0.01 0.45 0.09 0.041].*w_pos,...
    'Callback', {@cb_b_pmd,data,sample_rate},...
    'Tag','t_p_type');


%-------------------------------------------------
% выбор начало частотного диапазона

% h_t_st_d - handle_text_start_diap
h_t_st_d = uicontrol('Parent',h_fig_ia,...
    'Style','Text','Tag','t_t_st_d',...
    'String','Начало частотного диапазона',...
    'Position',[0.01 0.41 0.09 0.041].*w_pos);
% h_e_st_d - handle_edit_start_diap
h_e_st_d = uicontrol('Parent',h_fig_ia,...
    'Style','Edit','String','8','Tag','t_e_st_d',...
```

```
    'Position',[0.01 0.35 0.09 0.041].*w_pos,...
    'Callback',{@calculate_fft,data,sample_rate});

%-------------------------------------------------
% выбор конца частотного диапазона

% h_t_en_d - handle_text_end_diap
h_t_en_d = uicontrol('Parent',h_fig_ia,...
    'Style','Text','Tag','t_t_en_d',...
    'String','Конец частотного диапазона',...
    'Position',[0.01 0.31 0.09 0.041].*w_pos);
% h_e_en_d - handle_edit_end_diap
h_e_en_d = uicontrol('Parent',h_fig_ia,...
    'Style','Edit','String','12','Tag','t_e_en_d',...
    'Position',[0.01 0.25 0.09 0.081].*w_pos,...
    'Callback',{@calculate_fft,data,sample_rate});

%-------------------------------------------------
% выбор размера скользящего окна

% h_t_len_f - handle_text_len_frame
h_t_len_f = uicontrol('Parent',h_fig_ia,...
    'Style','Text','Tag','t_t_len_f',...
    'String','Размер скользящего окна',...
    'Position',[0.01 0.21 0.09 0.041].*w_pos);
% h_e_len_f - handle_edit_len_frame
h_e_len_f = uicontrol('Parent',h_fig_ia,...
    'Style','Edit','String','1024',...
    'Tag','t_e_len_f','Position',...
    [0.01 0.15 0.09 0.081].*w_pos);

%-------------------------------------------------
% кнопка построить динамику
```

```
% средней для диапазона мощности

% h_b_pmd - handle_button_plot_mean_dynamics
% t_b_pmd - tag_button_plot_mean_dynamics
% cb_b_pmd - callback_button_plot_mean_dynamics
h_b_pmd = uicontrol('Parent',h_fig_ia,...
    'Style','Pushbutton','Tag','t_b_pmd',...
    'String','Среднее в диапазоне',...
    'Position',[0.01 0.05 0.09 0.081].*w_pos,...
    'Callback',{@cb_b_pmd,data,sample_rate});

%-------------------------------------------------
% Отрисовка "дисплея" для вывода данных
% с выбранного канала,  не требует масштабирования

% h_dp - handle_data_plot
% t_dp - tag_data_plot
h_dp = subplot('Position', [0.15 0.71 0.84 0.28],...
    'Tag','t_dp');
cb_e_pos([],[],data,sample_rate);

%-------------------------------------------------
% Отрисовка "дисплея" для вывода
% спектра, не требует масштабирования

% h_fp - handle_fft_plot
% t_fp - tag_fft_plot
h_fp = subplot('Position', [0.15 0.38 0.84 0.28],...
    'Tag','t_fp');
calculate_fft([],[],data,sample_rate);

%-------------------------------------------------
% Отрисовка "дисплея" для вывода
```

```
% изменения среднего в диапазоне частот,
% не требует масштабирования

% h_dfp - handle_diap_freq_plot
% t_dfp - tag_fft_plot
h_dfp = subplot('Position', [0.15 0.05 0.84 0.28],...
    'Tag','t_dfp');
cb_b_pmd([],[],data,sample_rate);

end

function [ ] = resize_fig_ia(src,callbackdata)
% Функция изменения размера окна
% отображения интерфейса

figsz = get(src, 'Position');
pos_vect = [figsz(3) figsz(4) figsz(3) figsz(4)];

h_p = findobj(gcbo,'Tag','t_p');
h_t_p = findobj(gcbo,'Tag','t_t_p');

h_e_pos = findobj(gcbo,'Tag','t_e_pos');
h_t_pos = findobj(gcbo,'Tag','t_t_pos');
h_e_ws = findobj(gcbo,'Tag','t_e_ws');
h_t_ws = findobj(gcbo,'Tag','t_t_ws');

h_e_st_d = findobj(gcbo,'Tag','t_e_st_d');
h_t_st_d = findobj(gcbo,'Tag','t_t_st_d');
h_e_en_d = findobj(gcbo,'Tag','t_e_en_d');
h_t_en_d = findobj(gcbo,'Tag','t_t_en_d');
h_e_len_f = findobj(gcbo,'Tag','t_e_len_f');
h_t_len_f = findobj(gcbo,'Tag','t_t_len_f');
h_b_pmd = findobj(gcbo,'Tag','t_b_pmd');
```

```
h_t_p_type = findobj(gcbo,'Tag','t_t_p_type');
h_p_type = findobj(gcbo,'Tag','t_p_type');



set(h_t_p,'Position',...
    [0.01 0.95 0.09 0.041].*pos_vect)
set(h_p,'Position',...
    [0.01 0.91 0.09 0.041].*pos_vect)
set(h_e_pos,'Position',...
    [0.01 0.81 0.09 0.041].*pos_vect)
set(h_t_pos,'Position',...
    [0.01 0.85 0.09 0.041].*pos_vect)
set(h_e_ws,'Position',...
    [0.01 0.71 0.09 0.041].*pos_vect)
set(h_t_ws,'Position',...
    [0.01 0.75 0.09 0.041].*pos_vect)

set(h_t_p_type,'Position',...
    [0.01 0.51 0.09 0.041].*pos_vect);
set(h_p_type,'Position',...
    [0.01 0.45 0.09 0.041].*pos_vect);
set(h_e_st_d,'Position',...
    [0.01 0.35 0.09 0.041].*pos_vect)
set(h_t_st_d,'Position',...
    [0.01 0.41 0.09 0.041].*pos_vect)
set(h_e_en_d,'Position',...
    [0.01 0.25 0.09 0.041].*pos_vect)
set(h_t_en_d,'Position',...
    [0.01 0.31 0.09 0.041].*pos_vect)
set(h_e_len_f,'Position',...
    [0.01 0.15 0.09 0.041].*pos_vect)
```

```
set(h_t_len_f,'Position',...
    [0.01 0.21 0.09 0.041].*pos_vect)
set(h_b_pmd,'Position',...
    [0.01 0.05 0.09 0.041].*pos_vect)
end

function [ ] = cb_popup( src, dataevent, data,
  sample_rate)
% Функция callback для выпадающего меню
% выбора канала для анализа
%
% Аргументы :
% data - данные ЭЭГ
% sample_rate - частота дискретизации
%

% Отрисовка "дисплея" для вывода данных
% с выбранного канала,
% не требует масштабирования
% h_dp - handle_data_plot
% t_dp - tag_data_plot
h_dp = subplot('Position', [0.15 0.71 0.84 0.28],...
    'Tag','t_dp');

% вызов функции callback для изменения позиции
cb_e_pos(src,dataevent,data,sample_rate);

end

function [ ] = cb_e_pos( src, dataevent, data,
  sample_rate )
% функция callback для изменения начала
% выделения данных
```

```
%
% Аргументы :
% data - данные ЭЭГ
% sample_rate - частота дискретизации
%

% Отрисовка "дисплея" для вывода данных
% с выбранного канала,
% не требует масштабирования
% h_dp - handle_data_plot
% t_dp - tag_data_plot
h_dp = subplot('Position', [0.15 0.71 0.84 0.28],...
    'Tag','t_dp');

% h_p - handle_poupe
h_p = findobj('Tag','t_p');
% h_e_pos - handle_edit_position
h_e_pos = findobj('Tag','t_e_pos');
% h_e_ws - handle_edit_windows_size
h_e_ws = findobj('Tag','t_e_ws');

% выбранный в выпадающем меню канал
x = data(h_p.Value,:);
% указанные в поле для ввода начало и
% длина выделения данных
pos = str2double(h_e_pos.String);
ws = str2double(h_e_ws.String);
% защитные ограничения
if pos < 1
    pos = 1;
end
if  ws < 0
    ws = 0;
```

```
end
if pos > length(x)
    pos = length(x);
end
if pos + ws > length(x)
    ws = length(x) - pos;
end

% отрисовка данных
plot(x)
hold on
% отрисовка вертикальной линии
% для начала выделения данных
plot([pos pos],[-1 1],'r')
% и для конца выделения данных
plot([pos+ws pos+ws],[-1 1],'r')
hold off

% вызов функции рассчета спектра
calculate_fft([],[],data,sample_rate);
end

function [ ] = cb_b_pmd( src, dataevent, data,
   sample_rate )
% функция callback для кнопки
% "Среднее в диапазоне"
%
% Аргументы :
% data - данные ЭЭГ
% sample_rate - частота дискретизации
%
```

```
% Отрисовка "дисплея" для вывода изменения
% среднего в диапазоне частот,
% не требует масштабирования

% h_dfp - handle_diap_freq_plot
% t_dfp - tag_fft_plot
h_dfp = subplot('Position', [0.15 0.05 0.84 0.28],...
    'Tag','t_dfp');

%h_p - handle_poupe
h_p = findobj('Tag','t_p');
%h_e_len_f - handle_edit_len_frame
h_e_len_f = findobj('Tag','t_e_len_f');
%h_e_st_d - handle_edit_start_diap
h_e_st_d = findobj('Tag','t_e_st_d');
%h_e_en_d - handle_edit_end_diap
h_e_en_d = findobj('Tag','t_e_en_d');
% h_p_type - handle_poupe_type
h_p_type = findobj('Tag', 't_p_type');

% выбранный в выпадающем меню
% тип нормировки
types = ['non','all'];
type = types(h_p_type.Value);
% выбранный в выпадающем меню канал
x = data(h_p.Value,:);
% указанные в соответствующих полях
% начало диапазона частот
st_d = str2double(h_e_st_d.String);
% конец диапазона частот
en_d = str2double(h_e_en_d.String);
% длина фрейма
len_f = str2double(h_e_len_f.String);
```

```
% защитные ограничения
if st_d < 0
    st_d = 0;
end
if en_d < 0
    en_d = 0;
end
if st_d > sample_rate / 2
    st_d = sample_rate / 2;
end
if en_d > sample_rate / 2
    en_d = sample_rate / 2;
end
if en_d < st_d
    en_d = st_d;
end
if  len_f < 0
    len_f = 0;
end
if  len_f > length(x)
    len_f = length(x);
end

% рассчет и построение динамику
% спектральных показателей
%mrfd - mean_range_frequence_dynamics
mrfd = mean_freq_diap(x,len_f,st_d,en_d,...
    sample_rate, type);
plot(mrfd)
end

function [ ] = calculate_fft(src, dataevent, data,
  sample_rate)
```

```
% Функция для вычисления и отображения
% спектра
%
% Аргументы :
% data - данные ЭЭГ
% sample_rate - частота дискретизации
%

% Отрисовка "дисплея" для вывода спектра,
% не требует масштабирования
% h_fp - handle_fft_plot
% t_fp - tag_fft_plot
h_fp = subplot('Position',  [0.15 0.38 0.84 0.28],...
    'Tag','t_fp');

% h_p - handle_poupe
h_p = findobj('Tag','t_p');
% h_e_pos - handle_edit_position
h_e_pos = findobj('Tag','t_e_pos');
% h_e_ws - handle_edit_windows_size
h_e_ws = findobj('Tag','t_e_ws');

% выбранный в выпадающем меню канал
x = data(h_p.Value,:);
% указанные в поле для ввода начало и
% длина выделения данных
pos = str2double(h_e_pos.String);
ws = str2double(h_e_ws.String);
% защитные ограничения
if pos < 1
    pos = 1;
end
if  ws < 0
```

```
    ws = 0;
end
if pos > length(x)
    pos = length(x);
end
if pos + ws > length(x)
    ws = length(x) - pos;
end

% h_e_st_d - handle_edit_start_diap
h_e_st_d = findobj('Tag','t_e_st_d');
% h_e_en_d - handle_edit_end_diap
h_e_en_d = findobj('Tag','t_e_en_d');

% указанные начало и конец частотного
% диапазона
st_d = str2double(h_e_st_d.String);
en_d = str2double(h_e_en_d.String);
%защитные ограничения
if st_d < 0
    st_d = 0;
end
if en_d < 0
    en_d = 0;
end
if st_d > sample_rate / 2
    st_d = sample_rate / 2;
end
if en_d > sample_rate / 2
    en_d = sample_rate / 2;
end
if en_d < st_d
    en_d = st_d;
```

```
end

% вычисление первой половины абсолютного
% значения спектра
afft = abs(fft(x(pos : pos + ws)));
afft = afft(1:round(end/2));

% отрисовка спектра
plot([0:length(afft)-1]...
    /length(afft)*sample_rate/2,afft)
hold on
% отрисовка вертикальных линий
% для начала частотного диапазона
plot([st_d st_d],[0 max(afft)],'r')
% и конца частотного диапазона
plot([en_d en_d],[0 max(afft)],'r')
hold off

end

function [ out ] = mean_freq_diap(x, len_frame,
  f_start_diap, f_end_diap, Fs, type)
% Функция для вычисления среднего
% значения в заданном чатотном диапазоне
%
% Аргументы :
% x - одномерный сигнал
% len_frame - длинна скользящего окна (фрейма)
% f_start_diap - начало частотного диапазона
% f_end_diap - конец частотного диапазона
% Fs - частота дискретизации
% type - тип нормировки, возможные варианты
%       'non' - без нормировки
```

```
%       'all' - деленное на среднее по всему спектру
%
% Возвращаемое значение :
% out - массив средних для частотного
%       диапазона значений длинной
%       в количество фреймов

% количество фреймов
num_frame = round(length(x)/len_frame) - 1;
out = zeros(1, num_frame);

for k = 1 : num_frame
    %
    frame = x((k-1) * len_frame +1: k * len_frame);
    afft_x = abs(fft(frame));
    afft_x(1) = 0;

    % выбор нормировки
    switch type
        case 'non'
            % нормировка отсутствует
            out(k) = mean(afft_x(round(...
                len_frame*f_start_diap/Fs):...
                round(len_frame*f_end_diap/Fs)));
        case 'all'
            % нормированное на среднее по всему
            % спектру
            out(k) = mean(afft_x(round(...
                len_frame*f_start_diap/Fs):...
                round(len_frame*f_end_diap/Fs)))...
                /mean(afft_x);
    end
end
```

```
end
```

4.11 Модуль вычисления огибающей спектра

Модуль может использоваться для повышения качества изображения спектра при исследованиях. Часто спектр слишком не гладкий, данный подход к построению спектральной огибающей хорош, тем что можно примерно определить количество пиков которые будут видны на спектре. Это достигается выбором порядка огибающей и соответственно многочлена линейного предсказания. Ниже приводятся некоторые аргументы за использование такой методики построения огибающей спектра.

Сигналы биологического происхождения могут быть приближены гармоническими сигналами [1]. Хорошо известны методы анализа сигналов в частотной области. Спектральная огибающая сигнала может быть описана функцией $\sigma/|A(z)|$, где A - многочлен не имеющий корней в единичном круге, а σ число. Коэффициенты многочлена A называются коэффициентами линейного предсказания. И могут быть вычислены по алгоритму Левинсона-Дурбина [3]. На Рис. 9 представлены спектральные огибающие для многочленов A порядка 2,10,20,50,100 для реального сигнала взятого из базы данных с кафедры Высшей нервной деятельности МГУ [14]. Как видно из Таблицы 1 ошибка вычисления обобщенных спектральных показателей [1] для диапазона частот от 8 до 13 Гц довольно мала.

Название	Порядок				
	2	10	20	50	100
Частота максимальной по амплитуде гармоники	0.00495	0.00105	0.00095	0.00095	0.00005
Средневзвешенная частота	1.35228	0.78331	0.31025	0.19985	0.18640

Таблица 1: Ошибка обобщенных спектральных показателей

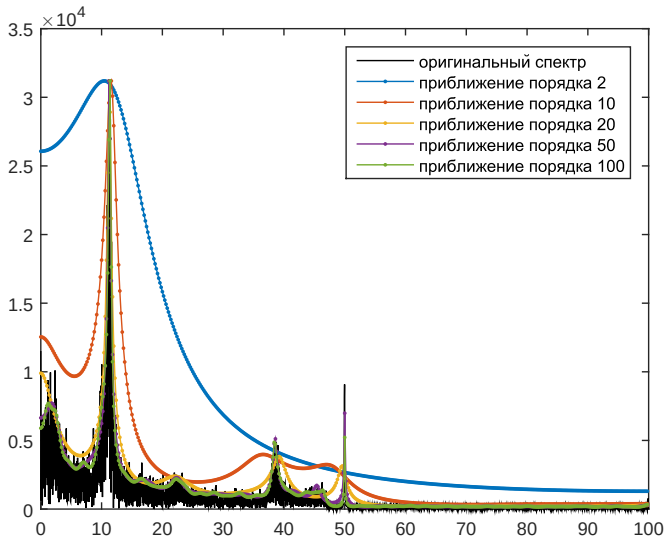


Рис. 9: Спектр сигнала и его огибающие

```
function [ out ] = spectrum_envelope( x, order, len )
% функция вычислет верхнюю
% огибающую спектра входного сигнала
% Аргументы :
% x - одномерный вещественный сигнал
% order - порядок многочлена линейного
%       предсказания
% len - длина возвращаемой огибающей
%
% Возвращаемые значения :
% out - вершняя огибающая половины спектра.
% т.к. сигнал веществен, то спектр симетричен

% вычисление коэффициентов линейного
% пердсказания по алгоритму
% Левинсона-Дурбина
Frame = length(x);
for k = 0:order
    r(k+1)=x(k+1:Frame)'*x(1:Frame-k);
end
```

```
E=r(1);
a=1;
for i = 1:order
    k=-a*r(i+1:-1:2)'/E;
    a=[a,0];
    a=a+k*a(i+1:-1:1);
    E=E*(1-k*k);
end

% равноотстоящие точки, расположенные
% на верхней половине единичной окружности
t = exp( i*[0:len/2-1]/len*2*pi);
% подставляются в многочлен, коэффициенты
% которого получены алгоритмом
% Левинсона-Дурбина
out = 1 ./ abs(polyval(a,t));

end
```

5 Заключение

В работе представлена разработанная библиотека программ для обработки нейронной активности на языке MATLAB. Библиотека включает в себя набор модулей позволяющих построить полный цикл нейрообратной связи. Были реализованы методы для повышения детектируемости μ-ритма. Разработан и интерфейс для анализа некоторых обобщенных спектральных показателей. Реализован набор модулей нейрообратной связи с использованием метода Движущейся волны ЭЭГ. Изучена предыдущая версия системы, и на ее основе разработана новая с четкой структурой, понятными связями между модулями и хорошей документацией. Получен патент [19] на программу формирующую сигнал необходимый для построения нейрообратной связи из сырых данных электроэнцефалограммы. Так-

же работа была представлена на конференции молодых ученых “Навигация и управление движением”, где в секции “Навигация и управление в робототехнических системах” была удостоена диплома третьей степени.

6 Список литературы

1. Кулаичев А.П. Компьютерная электрофизиология и функциональная диагностика: учебное пособие. 4-е изд., перераб. и доп. – М.: ФОРУМ: ИНФРА-М, 2007.

2. Белов Дмитрий Романович. Движущаяся волна ЭЭГ человека: диссертация ... доктора биологических наук: 03.03.01 [Место защиты: Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования "Санкт-Петербургский государственный университет"].- Санкт-Петербург, 2015.- 426 с.

3. Ramirez, M. A. A Levinson Algorithm Based on an Isometric Transformation of Durbin’s. IEEE Signal Processing Letters. 15: 99–102.

4. Zoltan J. K., Michael S. L. and Steven Z. Z. Spatial patterns underlying population differences in the background EEG. Brain topography. 1990. Vol. 2 (4) pp. 275-284.

5. Budzynski, T.H. Introduction to quantitative EEG and neurofeedback., Academic Press., 2009.

6. Berger, H. Uber das Elektroenkephalogramm des Menschen. Arch. Psychiatr. Nervenkr., 87, 527–570, 1929.

7. Fazli, S., Mehnert, J., Steinbrink, J., Curio, G., Villringer, A., Muller, KR., Blankertz, B. Enhanced performance by a hybrid NIRS-EEG brain computer interface. NeuroImage 59 (2012) 519–529.

8. Waldert, S., Preissl, H., Demandt, E., Braun, C., Birbaumer, N., Aertsen, A., Mehring, C. Hand movement direction decoded from MEG and EEG. The Journal of Neuroscience, January 23, 2008, 28(4):1000 –1008.

9. Blankertz, B., Dornhege, G., Schafer, C., Krepki, R., Kohlmorgen, J., Muller, KR., Kunzmann, V., Losch, F., Curio, G. Boosting bit rates and error detection for the classification of fast-paced motor commands based on single-trial EEG analysis. IEEE TRANSACTIONS ON NEURAL SYSTEMS AND REHABILITATION ENGINEERING, VOL. 11, NO. 2, JUNE 2003.

10. Smetanin, N., Volkova, K., Zabodaev, S., Lebedev, M., A., Ossadtchi, A., NFBLab—A Versatile Software for Neurofeedback and Brain-Computer Interface Research, Front. Neuroinform., 24 December 2018.

11. Vukelic, M., Gharabaghi, A., Oscillatory entrainment of the motor cortical network during motor imagery is modulated by the feedback modality, NeuroImage 111 (2015) 1–11.

12. Кирой, В., Н., Лазуренко, Д., М., Шепелев, И., Е., Миняева, Н., Р., Асланян, Е., В., Бахтин, О., М., Шапошников, Д., Г., Владимирский, Б., М., ИЗМЕНЕНИЕ СПЕКТРАЛЬНЫХ ХАРАКТЕРИСТИК ЭЭГ В ДИНАМИКЕ ТРЕНИНГОВ С НЕЙРООБРАТНОЙ СВЯЗЬЮ, ФИЗИОЛОГИЯ ЧЕЛОВЕКА, 2015, том 41, № 3, с. 50–62.

13. Сотников, П., И., Обзор методов обработки сигнала электроэнцефалограммы в интерфейсах мозг-компьютер, Инженерный вестник # 10, октябрь 2014.

14. Научные ресурсы: [Электронный ресурс]//Кафедра Высшей нервной деятельности МГУ. URL: http://neurobiology.ru/doc/index.php?ID=78. (Дата обращения: 12.05.19)

15. Scharnowski, F., Hutton, C., Josephs, O., Weiskopf, N., Rees, G., Improving visual perception through neurofeedback, J Neurosci 2012, 32:17830-17841.

16. Herbert. H., J., Report of the committee on methods of clinical examination in electroencephalography. Electroencephalography and Clinical Neurophysiology. 10 (2): 370–375 (1958)

17. ЖУРАВЛЕВ, М., О., ХРАМОВ, А., Е., МАКАРОВ, В., В., ПРОГРАММА ЭВМ ДЛЯ ОПРЕДЕЛЕНИЯ СТЕПЕНИ СИНХРОНИЗАЦИИ МЕЖДУ РАЗЛИЧНЫМИ КАНАЛАМИ ЭЭГ В УСРЕДНЕННЫХ ИНТЕРВАЛАХ ВРЕМЕНИ.//eLIBRARY.RU - Научная электронная библиотека. URL: https://elibrary.ru/item.asp?id=36940806. (Дата обращения: 12.05.19)

18. ГРУБОВ В. В., ХРАМОВ, А., Е., РУННОВА, А., Е., ЖУРАВЛЕВ, М., О., ПРОГРАММА ЭВМ ДЛЯ АВТОМАТИЗИРОВАННОГО УДАЛЕНИЯ АРТЕФАКТОВ РАЗЛИЧНЫХ ТИПОВ С СИГНАЛОВ ЭЭГ (AUTOMATED ARTIFACT REMOVAL).//eLIBRARY.RU - Научная электронная библиотека. URL: https://elibrary.ru/item.asp?id=36940806. (Дата обращения: 12.05.19)

19. Степаненко Д.А., Семенов Д.М., Плотников С.А., Фрадков А.Л. Программный комплекс для реализации нейрообратной связи: модуль обработки сигнала ЭЭГ. Свидетельство о государственной регистрации программы для ЭВМ, №2019610034 от 09.01.2019. – М.:Роспатент

20. Lee et al, A pilot randomized controlled trial using EEG-based brain–computer interface training for a Chinese-speaking group of healthy elderly, Clinical Interventions in Aging, 9 January 2015.

21. Zich, C., Vos, M.D., Kranczioch, C., Debener, S., Wireless EEG with individualized channel layout enables efficient motor imagery training, Clinical Neurophysiology (2014), doi: http://dx.doi.org/10.1016/j.clinph.2014.07.007

22. Stephygraph, L. R., Arunkumar, N., Brain-Actuated Wireless Mobile Robot Control Through an Adaptive Human–Machine Interface. Advances in Intelligent Systems and Computing, 537–549, (2015), doi:10.1007/978-81-322-2671-0_52.