

МОСКОВСКИЙ ИНСТИТУТ ЭЛЕКТРОННОЙ ТЕХНИКИ
Институт системной и программной инженерии
и информационных технологий (Институт СПИНТех)

**Лабораторный практикум по курсу
"Нейронные сети"**

Лабораторная работа 1.

Трудоёмкость алгоритма обработки данных.

Моделирование функций активации нейрона.

1. Основные теоретические сведения

Нейронная сеть головного мозга содержит 10^{10} - 10^{11} нейронов, при этом каждая нервная клетка связана в среднем с 10^3 - 10^4 других нейронов, образуя в целом около 60×10^{12} синаптических связей. Установлено, что совокупность нейронов центральной нервной системы в объеме 1 мм^3 формирует достаточно независимую локальную биологическую нейронную сеть (БНС), несущую определенную функциональную нагрузку. Быстродействие БНС на пять-шесть порядков ниже, чем быстродействие элементов современных ИС: БНС - 10^{-3} с., ИС - 10^{-9} с. При этом энергетическая эффективность БНС равна 10^{-16} Дж на операцию в секунду, тогда как энергетическая эффективность современной ЭВМ, осуществляющей операции на основе бинарной логики, составляет порядка 10^{-6} Дж на операцию в секунду.

Нейроны образуют два характерных типа соединений - *конвергентные*, в которых число нейронов предыдущего слоя превосходит число нейронов последующего слоя, и *дивергентные*, в которых контакты осуществляются со все возрастающим числом клеток последующих слоев иерархии. Сочетание конвергентных и дивергентных соединений обеспечивает многократное дублирование информационных каналов, что собственно и определяет высокую надежность БНС. При повреждении или исчезновении части нервных клеток сохранившиеся нейроны оказываются в состоянии поддерживать функционирование сети.

Выделяют три основных типа БНС, отличающихся структурой и назначением.

К БНС первого типа относятся *иерархические сети*, наиболее часто встречающиеся в сенсорных и двигательных путях. Передача информации в иерархических сетях осуществляется в процессе последовательного перехода от одного структурного уровня к другому.

К БНС второго типа относятся *локальные сети*, формируемые нейронами с ограниченными зонами влияния. Нейроны таких сетей производят переработку информации в пределах одного иерархического уровня. При этом с функциональной точки зрения локальная БНС представляет собой достаточно изолированную тормозящую или возбуждающую структуру.

БНС третьего типа принято считать *дивергентные сети с одним входом*. Базовый нейрон, находящийся в основании данной сети, может оказывать воздействие на целую совокупность нейронов, а потому дивергентные сети с одним входом являются эффективным согласующим инструментом в сложном переплетении нейронно-сетевых архитектур всех типов.

Принципиальные отличия в функционировании БНС и последовательных ЭВМ приведены в табл.1 [1].

Таблица 1.

Сравнение особенностей функционирования БНС и последовательных ЭВМ.

Параметры	Последовательная ЭВМ	БНС
Процессор	Сложный	Простой
	Высокоскоростной	Низкоскоростной
	Один или несколько	Большое количество
Память	Отделена от процессора	Интегрирована в процессор
	Локализованная	Распределенная
	Адресация не по содержанию	Адресация по содержанию
Вычисления	Централизованные	Распределенные
	Последовательные	Параллельные
	Хранимые программы	Самообучение
Надежность	Высокая уязвимость	Живучесть
Специализация	Численные и символьные операции	Проблемы восприятия

Отметим важнейшие свойства БНС:

1. обработка информации в БНС осуществляется в *параллельном режиме*. Каждый нейрон формирует свой выход только на основе своих входов и собственного внутреннего состояния под воздействием общих механизмов регуляции нервной системы;
2. БНС обладают способностью к комплексной обработке информации. К этой группе свойств относятся *ассоциативность* (сеть может восстанавливать полный образ по его части), способность к классификации, обобщению, абстрагированию и множество других;
3. функционирование БНС отличается высокой степенью самоорганизации. В процессе работы они самостоятельно, под воздействием внешней среды, обучаются решению разнообразных задач. Не существует никаких *принципиальных ограничений на сложность задач*, решаемых БНС. *Нервная система сама формирует алгоритмы своей деятельности*, уточняя и усложняя их в течение жизни;
4. БНС являются *аналоговыми системами*. Информация поступает в сеть по большому количеству каналов и кодируется по пространственному принципу: вид информации определяется номером нервного волокна, по которому она передается. Амплитуда входного воздействия кодируется плотностью нервных импульсов, передаваемых по волокну;

БНС обладают чрезвычайно высокой надежностью: выход из строя даже 10% нейронов в нервной системе не прерывает их работы.

Таким образом, за счет использования новой математической модели, а также её программного или аппаратного воплощения можно существенно повысить *эффективность вычислительного процесса* и его *масштабируемость* (англ. *scalability*).

В настоящей лабораторной работе изучается вопрос оценки трудоемкости алгоритма на примере прямой реализации преобразования Фурье непрерывных во времени сигналов, а также его эффективного, быстрого аналога - БПФ. В дополнение осуществляется программирование и визуализация набора функций активации нейрона.

1.1. Преобразование Фурье непрерывных во времени сигналов

Основная цель каждого преобразования состоит в том, чтобы сформулировать исходную проблему в альтернативной форме, с целью *упрощения ее решения* (снижения *трудоемкости обработки данных*). Это утверждение справедливо и для преобразования Фурье, являющегося мощным математическим инструментом для решения широкого круга физических, инженерных и иных технических задач.

В системах компьютерного зрения, медицинской визуализации (томографии) можно выделить три взаимосвязанных области, в которых использование преобразования Фурье является одной из важнейших процедур вычислительного процесса. Это обработка исходных измерительных данных, собственно реализация распознавания (реконструктивного алгоритма), а также анализ и фильтрация полученного изображения.

В рамках настоящего задания мы сосредоточим свое внимание на применении преобразования Фурье в теории сигналов и оценки эффективности алгоритма. В качестве данных при этом будут использованы гармонические сигналы как функции времени. В такой постановке преобразование Фурье формирует представление сигнала в новом базисе, а именно его временной спектр.

Фурье-преобразование сигнала, являющегося временной функцией, имеет вид

$$X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-j\omega t} dt. \quad (1)$$

Операция обращения

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) e^{j\omega t} d\omega \quad (2)$$

соответствует обратному преобразованию Фурье.

В выражениях (1–2) $x(t)$ – зависящий от времени сигнал, $X(\omega)$ – его фурье-образ. Фурье-образ и его обратное значение определены в предположении, что оба вышеприведенных интеграла существуют. Условия существования интегралов, т.е. вытекающие из этих условий требования к $x(t)$, равно как и к $X(\omega)$, анализируются в общих математических курсах.

1.2. Дискретное во времени преобразование Фурье

Методы цифровой обработки сигналов основаны на представлении исходного сигнала в виде массива его дискретных значений. Процесс дискретизации можно интерпретировать перемножение исходного непрерывного сигнала с последовательностью эквидистантных дельта-функций

$$a(t) = \sum_{n=-\infty}^{\infty} \delta(t - n\Delta), \quad (3)$$

с интервалом дискретизации Δ и частотой дискретизации $f_s = \frac{1}{\Delta}$. Для фурье-образа сигнала, представленного в виде выборочных данных $x_a(t) = x(t)a(t)$, будем иметь

$$X_a(\omega) = \int_{-\infty}^{\infty} x_a(t) e^{-j\omega t} dt = \int_{-\infty}^{\infty} x(t) a(t) e^{-j\omega t} dt. \quad (4)$$

Вычисление интеграла с учетом (3) соответствует дискретному во времени преобразованию Фурье

$$X_a(\omega) = \sum_{-\infty}^{\infty} x_n e^{-j\omega n \Delta} \quad (5)$$

с $x_n = x(n\Delta)$. Эквивалентное (2) обратное преобразование Фурье выражается в виде

$$x_n = \frac{\Delta}{2\pi} \int_{-\pi/\Delta}^{\pi/\Delta} X_a(\omega) e^{j\omega n \Delta} d\omega. \quad (6)$$

Взаимосвязь между $X_a(\omega)$ и $X(\omega)$ можно легко установить, используя теорему о свертке. Произведению двух сигналов во временной области соответствует свертка их фурье-образов в пространстве частот. Другими словами, справедливо следующее равенство

$$X_a(\omega) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\nu) A(\omega - \nu) d\nu, \quad (7)$$

в котором фурье-образ $a(t)$, записанный в форме разложения в ряд Фурье

$$a(t) = \frac{1}{\Delta} \sum_{-\infty}^{\infty} e^{j\frac{2\pi n}{\Delta} t}, \quad (8)$$

может быть представлен в виде

$$A(\omega) = \int_{-\infty}^{+\infty} a(t) e^{-j\omega t} dt = \frac{2\pi}{\Delta} \sum_{n=-\infty}^{\infty} \delta\left(\omega - \frac{2\pi n}{\Delta}\right). \quad (9)$$

С учетом (8–9), интеграл свертки (7) имеет вид

$$X_a(\omega) = \frac{1}{\Delta} \sum_{n=-\infty}^{\infty} X\left(\omega - \frac{2\pi n}{\Delta}\right). \quad (10)$$

Нетрудно видеть, что фурье-образ дискретизованного сигнала является периодическим с периодом $2\pi/\Delta$ и представляет собой периодически продолженную версию фурье-образа исходного непрерывного во времени сигнала. Отсюда следует, что применение дискретного преобразования Фурье к сигналам с неограниченной шириной спектра должно сопровождаться перекрытием смежных спектральных областей. Этот эффект получил название эффекта наложения (Aliasing). В практических приложениях устранение частотного наложения достигается низкочастотной фильтрацией исходного сигнала. При этом частота дискретизации f_s отфильтрованного сигнала должна быть по крайней мере в два раза выше граничной частоты фильтра f_c :

$$f_s > 2f_c. \quad (11)$$

Данное условие называется критерием Найквиста.

1.3. Конечное преобразование Фурье

На практике в качестве входной информации доступно лишь ограниченное количество дискретных отсчетов x_n ($n = 0, 1, \dots, N-1$). По этой причине вместо фурье-образа (5) вычисляется его конечномерный аналог

$$X^N(\omega) = \sum_{n=0}^{N-1} x_n e^{-j\omega n \Delta}, \quad (12)$$

где N соответствует количеству выборочных элементов. Данное преобразование известно как конечное преобразование Фурье.

Последствия использования конечного числа дискретных значений можно установить снова с помощью сверточной теоремы. Ограничение размерности выборки может быть интерпретировано как произведение дискретизованного сигнала с гребенчатой прямоугольной функцией

$$\omega_n = \begin{cases} 1 & n \in [0, N-1] \\ 0 & n \notin [0, N-1] \end{cases}. \quad (13)$$

Данному произведению, в свою очередь, соответствует свертка $X_a(\omega)$ с известным фурье-образом прямоугольного окна

$$W^N(\omega) = \sum_{n=-\infty}^{\infty} \omega_n e^{-j\omega n \Delta} = e^{-j\omega(N-1)\Delta} \frac{\sin(\omega N \Delta / 2)}{\sin(\omega \Delta / 2)} \quad (14)$$

в частотной области:

$$X^N(\omega) = \sum_{n=-\infty}^{\infty} x_n \omega_n e^{-j\omega n \Delta} = \frac{\Delta}{2\pi} \int_{-\pi/\Delta}^{\pi/\Delta} X_a(\nu) W^N(\omega - \nu) d\nu. \quad (15)$$

Свертка (15) приводит к искажениям $X_a(\omega)$, которые определяются конечной шириной так называемого главного лепестка и уровнем боковых лепестков $W^N(\omega)$. Оба эффекта зависят от формы и ширины выбранного окна. В общем виде справедливо: чем шире окно, т.е. чем больше дискретных отсчетов находится в распоряжении для последующей обработки, тем уже главный лепесток и слабее боковые лепестки, т.е. лучше частотная разрешающая способность.

Часто с целью достижения оптимального компромисса по ширине главного лепестка и высоты боковых лепестков вместо окна прямоугольной формы используют другие окна. Эффекты, связанные с выбором того или иного окна, будут рассматриваться позднее применительно к обработке изображений.

1.4. Дискретное преобразование Фурье

В цифровой обработке сигналов конечное преобразование Фурье рассчитывается лишь для некоторого ограниченного числа отсчетов в частотной. Обычно выбираются $\omega_k = \frac{2\pi k}{N\Delta}$, что приводит к замене конечного Фурье-преобразования (12) дискретным преобразованием Фурье (ДПФ)

$$X(k) = X^N(\omega_k) = \sum_{n=0}^{N-1} x_n e^{-j \frac{2\pi}{N} nk}, \quad (k = 0, 1, \dots, N-1). \quad (16)$$

Обратное ДПФ вводится в виде

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j \frac{2\pi}{N} nk}, \quad (n = 0, 1, \dots, N-1). \quad (17)$$

Часто приходится использовать очень ограниченный набор данных, так что применение ДПФ приводит к определению незначительного числа компонент спектра. Недостаток информации может быть в некоторой степени восполнен, если исходную выборку данных дополнить нулями. С помощью данной процедуры получают большее количество точек в частотной области и более гладкое представление. Следует, однако, учесть, что разрешение остается при этом неизменным.

1.5. Быстрое преобразование Фурье

Несмотря на то, что ДПФ в форме (16) позволяет вычислять фурье-образ сигнала в процессе компьютерного моделирования, тем не менее непосредственное использование данной процедуры не получило широкого распространения в силу *квадратичной зависимости требуемого количества машинных операций от длины вектора данных*. Лишь создание эффективных алгоритмов компьютерной реализации (16) сделало его доступным для широкого использования и обеспечило значительный вычислительный и технический прогресс в самых разных областях. Первый быстрый алгоритм решения (16) был разработан Кули и Тьюки (Cooley and Tuckey) в 1965 году [2]. Процедура, предложенная Кули и Тьюки, стала основой целого ряда вычислительных алгоритмов, известных сегодня как БПФ - быстрое преобразование Фурье (англ. FFT - Fast Fourier Transform).

БПФ-алгоритмы особенно эффективны, когда длины входных векторов равняются 2^i ($i \in N$). В этом случае количество требуемых вычислительных операций в противоположность ДПФ выражается как

$$Q_{FFT} \sim N \log_2 N, \quad (18)$$

в то время как для ДПФ

$$Q_{DFT} \sim N^2. \quad (19)$$

Это означает, например, для $N = 1024$, снижение трудоемкости вычислительных операций с коэффициентом

$$\frac{N \log_2(N)}{N^2} \approx \frac{1}{100}. \quad (20)$$

В том случае, если размерность выборки не равняется целой степени двойки, можно дополнить исходный вектор нулями до следующего требуемого алгоритмом значения. Существуют также так называемые b-БПФ-алгоритмы, работающие с длинами данных $b^i, i \in N$, а также еще более медленные процедуры, использующие смешанный базис, а потому доступные для применения при обработке данных произвольной длины.

2. Функции активации в нейроне (перцептроне)

Функция активации (активационная функция, функция возбуждения) – функция, вычисляющая выходной сигнал нейрона (перцептрона) $y = y(v)$. В качестве аргумента принимает сигнал v , получаемый на выходе входного сумматора Σ . Наиболее часто используются следующие функции активации:

1. Единичный скачок или пороговая функция, т.е. простая кусочно-линейная функция. Если входное значение меньше порогового $v \leq v_0$, то значение функции активации равно минимальному допустимому $y_{\min} = 0$, иначе – максимально допустимому. В большинстве случаев $y_{\max} = 1$.
2. Линейный порог или гистерезис, т.е. кусочно-линейная функция. Данная функция имеет два линейных участка, где функция активации тождественно равна минимально допустимому $v_0 = 0$ и максимально допустимому $v_0 = 1$ значениям, а также имеет участок, на котором функция строго монотонно возрастает.
3. Сигмоидная функция (англ. *sigmoid*), а именно монотонно возрастающая, всюду дифференцируемая S-образная нелинейная функция с насыщением. Сигмоидная функция позволяет усиливать слабые сигналы и не переходить в режим насыщения при наличии сильных сигналов. Примером сигмоидальной функции активации может служить логистическая функция, задаваемая следующим выражением:

$$y = \frac{1}{1 + \exp(-\alpha v)},$$

где α – параметр наклона сигмоидной функции активации. Изменяя этот параметр, можно построить функции с различной крутизной. Диапазон изменения значений функции также от 0 до 1

4. Еще одним примером сигмоидной функции активации является гиперболический тангенс, представляемый следующим выражением:

$$y = th \frac{v}{\alpha},$$

где α – это также параметр, влияющий на наклон тангенциальной функции.

Все эти функции применяются при моделировании нейронных сетей в пакете Matlab для создании архитектур нейронных сетей и их последующего моделирования, а именно, обучения и тестирования.

3. Задания

3.1. Реализация ДПФ в языке MATLAB

Пусть задан гармонический сигнал с некоторыми амплитудой и частотой. Изучить программу *Lab_1_1.m*, обеспечивающую диалоговое задание гармонического сигнала и его визуализацию, а также программу *Lab_1_2.m*, реализующую ДПФ такого сигнала и его восстановление с помощью обратного ДПФ. Пояснить работу программы, выбор частоты дискретизации и исчезновение оператора суммы при реализации прямого и обратного ДПФ в программе *Lab_1_2.m*.

3.2. Оценка трудоемкости обработки данных с помощью ДПФ и БПФ

Используя программу *Lab_1_2.m*, написать программу *Lab_1_3.m*, реализующую: а) дискретизацию и визуализацию функций синуса и косинуса с частотой 2 кГц в двух вариантах: для заданного интервала наблюдения и для заданного количества точек; б) вычислить фурье-образы исходных сигналов с помощью прямого вычисления ДПФ и с помощью ДПФ, реализованного в MATLAB (функция *fft*); в) визуально сравнить реальные и мнимые части фурье-образов и квадраты их модулей.

Построить график зависимости времени обработки исходных данных с помощью ДПФ и БПФ, варьируя размерность исходного массива 2^s от 128 ($s = 7$) до 4096 ($s = 12$) (если не происходит зависание вычислительного устройства).



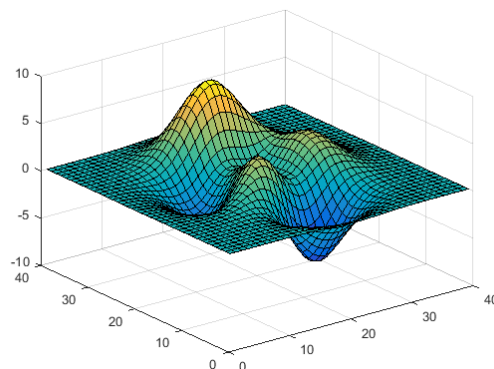
Рис. 1. Демонстрация трудоемкости вычислительного процесса (вычисление БПФ случайного сигнала размерности 2^s).

Указание: Для измерения производительности программ в MATLAB можно использовать следующие возможности:

- функцию *etime*, которая позволяет определять текущее время с точностью до сотых долей секунды. Имея два показателя времени t_1 и t_2 и подав команду *etime(t2,t1)*, получим время, прошедшее с момента t_1 до момента t_2 ;
- функцию *cputime*, позволяющую оценить время, расходуемое центральным процессором, для запуска и выполнения приложения. Например,

```
t = cputime;
surf(peaks(40));
e = cputime-t
e = 0.4667
```

- запуск таймера посредством вызова команд *tic* и *toc*
- ```
tic
```





```

A = rand(12000, 4400);
B = rand(12000, 4400);
toc
C = A' .* B';
toc
Elapsed time is 1.736832 seconds.
Elapsed time is 0.757567 seconds.

```

### 3.3. Программирование функций активации нейрона (перцептрона)

Написать программу-функцию, реализующую вычисление и отображение функций активации, представленных в разделе 2. Результат представить в виде  $m$ -функции, на вход которой поступает массив входных данных  $v$ , а также, если требуется, параметр  $\alpha$ , а в результате ее выполнения производится прорисовка требуемой функции активации.

### 3.4. Представление данных

Представьте результаты пп. 2.1-2.3 в виде матриц размерности  $N \times 2$  обучающего набора  $\{t_n, y_n\}$ , где  $t_n$  - вектор времени,  $y_n$  - вектор данных.

### 3.5. Производная сигмоидной функции

Вычислите (теоретически и численно) производную сигмоидной функции (п. 2.3) и представьте на графике.

## 4. Литература

1. Хайкин С. Нейронные сети. Полный курс – Изд-во Вильямс, Москва, 2006. - 1104с.
2. Рабинер Л. Гоулд Б. Теория и применение цифровой обработки сигналов. – Пер. с англ. – М: Мир, 1978. – 835с.