Тропченко А.Ю., Тропченко А.А.

ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ. МЕТОДЫ ПРЕДВАРИТЕЛЬНОЙ ОБРАБОТКИ

Учебно-методическое пособие
по дисциплине "Методы обработки сигналов и изображений"

Санкт- Петербург

2005

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное агентство по образованию

Санкт-Петербургский государственный университет
информационных технологий, механики и оптики

Кафедра вычислительной техники

Тропченко А.Ю., Тропченко А.А.

ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ. МЕТОДЫ ПРЕДВАРИТЕЛЬНОЙ ОБРАБОТКИ

Учебно-методическое пособие
по дисциплине "Методы обработки сигналов и изображений"

Санкт- Петербург

2005

УДК 681.325:621.372
ББК 32.973

Учебно-методическое пособие является дополненной и переработанной версией изданного кафедрой вычислительной техники пособия Тропченко А.Ю. ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ. МЕТОДЫ ПРЕДВАРИТЕЛЬНОЙ ОБРАБОТКИ. Учебно-методическое пособие по дисциплине. – СПб: СПбГИТМО (ТУ), 1999. – 97 с.

В учебно-методическом пособии рассматриваются основные методы теории цифровой обработки сигналов, используемые при предварительной обработке сигналов различной физической природы. Материал пособия разбит на 6 разделов. В каждом разделе, кроме шестого, приведены краткие теоретические сведения. Задания, приведенные в шестом разделе, имеют своей целью выработать у студентов практические навыки применения основных положений теории цифровой обработки сигналов и ее методов. Пособие может быть использовано при подготовке бакалавров и магистров по направлению 55.28.00 “ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА”, а также инженеров по специальности 22.01 “ВЫЧИСЛИТЕЛЬНЫЕ МАШИНЫ, КОМПЛЕКСЫ, СИСТЕМЫ И СЕТИ” и аспирантов.

Санкт-Петербургский государственный университет
информационных технологий. механики и оптики,

2005

А.Ю. Тропченко,
А.А.Тропченко

2005

С о д е р ж а н и е

1. ОСНОВНЫЕ ПОНЯТИЯ ЦИФРОВОЙ ОБРАБОТКИ СИГНАЛОВ

Бурный прогресс вычислительной техники в последние десятилетия привел к широкому внедрению методов цифровой обработки информации практически во всех областях научных исследований и народно-хозяйственной деятельности. При этом среди различных применений средств вычислительной техники одно из важнейших мест занимают системы цифровой обработки сигналов (ЦОС), нашедшие использование при обработке данных дистанционного зондирования, медико-биологических исследований, решении задач навигации аэрокосмических и морских объектов, связи, радиофизики, цифровой оптики и в ряде других приложений [11].

Цифровая обработка сигналов (ЦОС) - это динамично развивающаяся область ВТ, которая охватывает как технические, так и программные средства. Родственными областями для цифровой обработки сигналов являются теория информации , в частности, теория оптимального приема сигналов и теория распознавания образов. При этом в первом случае основной задачей является выделение сигнала на фоне шумов и помех различной физической природы, а во втором - автоматическое распознавание, т.е. классификация и идентификация сигнала.

В теории информации под сигналом понимается материальный носитель информации. В цифровой же обработке сигналов под сигналом будем понимать его математическое описание, т.е. некоторую вещественную функцию, содержащую информацию о состоянии или поведении физической системы при каком-нибудь событии, которая может быть определена на непрерывном или дискретном пространстве изменения времени или пространственных координат.

В широком смысле под системами ЦОС понимают комплекс алгоритмических, аппаратных и программных средств. Как правило, системы содержат специализированные технические средства предварительной (или первичной) обработки сигналов и специальные технические средства для вторичной обработки сигналов. Средства предварительной обработки

предназначены для обработки исходных сигналов, наблюдаемых в общем случае на фоне случайных шумов и помех различной физической природы и представленных в виде дискретных цифровых отсчетов , с целью обнаружения и выделения (селекции) полезного сигнала, его пеленгования и оценки характеристик обнаруженного сигнала. Полученная в результате предварительной обработки полезная информация поступает в систему вторичной обработки для классификации, архивирования, структурного анализа и т.д. [5,9].

Основными процедурами предварительной обработки сигналов являются процедуры быстрых дискретных ортогональных преобразований (БДОП), реализуемых в различных функциональных базисах, процедуры линейной алгебры, линейной и нелинейной фильтрации.

Одним из важнейших требований к техническим средствам систем ЦОС является обеспечение режима обработки сигналов в реальном времени (в темпе их поступления) при приемлемых стоимостных и весо-габаритных характеристиках системы. Такое требование, по сути, влечет за собой исключительно жесткие требования к быстродействию прежде всего средств предварительной обработки сигналов, что связано с большой интенсивностью входного потока сигнала в системах ЦОС. Так, многие технические задачи радиолокации и цифровой обработки сигналов требуют быстродействия аппаратных средств порядка 10^9 – 10^{10} оп/сек для обеспечения обработки в реальном времени. Другой особенностью подобных задач является необходимость обработки больших объемов (от 1 Мбайта и более) сложно структурированных многомерных данных, что вносит дополнительные трудности в организацию вычислений.

Указанное быстродействие может быть достигнуто применительно к узкому классу решаемых задач и при минимальных стоимостных и весогабаритных характеристиках системы лишь при использовании специализированных вычислительных средств параллельно-конвейерной архитектуры [1,2,5].

1.1. Понятие о первичной и вторичной обработке сигналов

В задачах ЦОС выделяют этапы предварительной (первичной) и вторичной обработки сигналов. Это связано с тем, что в общем случае на входе системы ЦОС наблюдается смесь *V(t)* полезного сигнала *x(t)*, некоторого шума *n(t)* и различных помех разной природы *p(t)*:

$$V(t) = x(t) + n(t) + p(t), \qquad (1.1)$$

где *n(t)* является характеристикой самого техничного устройства, а *p(t)* - некоторое искажающее воздействие самой физической среды, в которой распространяется сигнал (например, затухание).

Важнейшей задачей предварительной обработки сигнала является подавление *n(t)* и *p(t)* (шума и помехи). Такая задача оптимального может быть решена только на основе использования избыточности представления исходного сигнала, а также имеющихся сведений о свойствах полезного сигнала, помехи и шума для увеличения вероятности правильного приема [7,11].

Вследствие того, что на вход приемного устройства системы поступает сумма полезного сигнала и помехи, вероятность правильного приема будет определяться отношением полезного сигнала к помехе. Для повышения вероятности правильного приема сигнала должна быть произведена предварительная обработка принятого сигнала, обеспечивающая увеличение отношения сигнал/помеха. Таким образом, средства предварительной обработки при приеме должны содержать два основных элемента (рис.1.1) : фильтр Ф, обеспечивающий улучшение отношения сигнал/помеха, и решающее устройство РУ, выполняющее главные функции приема (обнаружения, различения и восстановления сигналов).

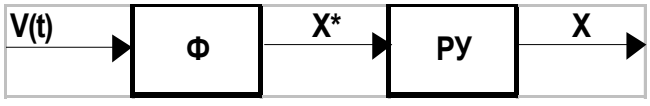


Рис.1.1. Структура оптимального приемного устройства

Известны следующие методы фильтрации, обеспечивающие улучшение соотношения сигнал/помеха:

- метод накопления;
- частотная фильтрация;
- корреляционный метод;
- согласованная фильтрация;
- нелинейная фильтрация.

Все эти методы основаны на использовании различий свойств полезного сигнала и помехи.

Кроме того, при предварительной обработке решается задача обнаружения сигнала и определения местоположения его источника. На этапе предварительной обработки в ряде случаев формируются также некоторые количественные оценки сигнала (амплитуда, частота, фаза).

Во входной смеси может и не быть полезного сигнала *x(t)*, поэтому на выходе системы предварительной обработки не будет никакого сигнала; следовательно, интенсивность потока данных на выходе будет ниже, чем на входе.

Система вторичной обработки сигнала предназначена для идентификации обнаруженного сигнала, его классификации и выдачи информации об обнаруженных сигналах оператору или формирования управляющего воздействия.

Характерной чертой предварительной обработки сигнала является постоянство алгоритма обработки при его достаточно высокой вычислительной сложности. Этап вторичной обработки характеризуется большей гибкостью используемых алгоритмов, необходимостью поддержки обмена с другим техническим средством или диалога с оператором. Поэтому системы вторичной обработки чаще всего строятся на основе программируемых вычислительных средств. Системы же предварительной обработки могут быть построены как на программируемых вычислительных средствах, так и на основе специальных вычислителей с жесткой логикой [5].

1.2. Технические средства комплекса обработки сигналов

Комплекс цифровой обработки сигналов содержит ЭВМ, специализированные устройства ввода и соответствующее программное обеспечение. В общем случае подобный комплекс должен также обеспечивать ввод, вывод и передачу сигналов различной физической природы. Общие требования к системам ЦОС представлены в таблице 1.1.

Таблица 1.1.

Основные требования к системам ЦОС

Параметр	**Значение параметра**	**Область применения**
Динамический диапазон обрабатываемых данных (бит)	Фиксированная точка: 8-16	Сжатие изображений, Радиолокация, Связь
	Плавающая точка: 16-32	Гидроакустика, Обработка речи, Обработка изображений
Емкость памяти буферных устройств (слов)	64-128 не менее 1К более 4К до 16К	Сжатие изображений, Радиолокация, Гидроакустика, Обработка изображений
Производительность (оп/сек)	10^8 10^9 10^9-10^{10} 10^{10}-10^{11}	Системы связи Обработка акустических сигналов Обработка изображений Радиолокация и гидроакустика

При этом особый интерес представляет обработка двумерных сигналов - изображений, получаемых от различных приемных устройств.

Многие задачи обработки изображений могут быть решены на современных персональных ЭВМ, если к скорости обработки не предъявляются высокие требования. В этом случае те или иные процедуры обработки изображений на

ПЭВМ реализуются путем создания специального программного обеспечения. Для обеспечения ввода изображения в реальном масштабе времени используются специализированные устройства ввода. К такому типу систем относятся системы IMAGE-3 и Microsight-2. Заметим, что в них обработка изображений производится на ПЭВМ не в реальном масштабе времени.

Для обработки сигнальной информации в реальном масштабе времени требуется производительность, превышающая производительность ПЭВМ. В этом случае необходимы специализированные устройства обработки. В настоящее время, согласно литературе, известны два типа систем обработки сигналов [6, 9].

Первый тип систем ЦОС предусматривает построение конструктивно законченного блока. Как правило, такой блок имеет модульную структуру и строится на базе специализированных СБИС (например, на основе БМК), что позволяет обеспечить аппаратную реализацию подлежащего исполнению алгоритма и оптимизировать структуру аппаратных средств под особенности алгоритма . К этому направлению можно отнести системы Series-151 и MaxVideo. В ряде случаев такие процессоры могут программироваться в целях выполнения тех или иных функций, как, например, WARP-процессор.

Отличительной чертой такой архитектуры является наличие отдельных магистралей ввода/вывода данных и возможность автономного функционирования. Блок со спецпроцессором при этом может быть выполнен в стандартном конструктиве типа VME, CAMAC, Multibys [5,6].

Такая система ЦОС допускает не только ввод, но и обработку изображений в реальном масштабе времени, поэтому подобный подход весьма эффективен при построении систем обработки видеоданных.

Второй тип систем ЦОС представляет собой ПЭВМ со специализированным сопроцессором в виде платы, подключаемой к магистрали ПЭВМ и конструктивно встраиваемый в ее корпус. Примером такой архитектуры могут служить наборы модулей фирмы Data Translation на базе сигнальных процессоров типа TMS и платы-акселераторы типа B008 фирмы INMOS на базе транспьютеров T800 [5,27] Указанные технические средства

ориентированы на использование в качестве периферийных спецпроцессоров для построения систем на базе IBM PC/AT. Спецпроцессор, входящий в эту систему, имеет, как правило, конвейерную структуру и может выполнять процедуры обработки изображений, требующие больших вычислительных затрат, в реальном масштабе времени. Настройка на выполнение тех или иных конкретных алгоритмов обработки видеоинформации производится программированием спецпроцессора, что увеличивает функциональную гибкость подобных систем и расширяет области их возможного применения.

На практике первый тип систем ЦОС наиболее часто используется в составе средств предварительной обработки сигналов, причем соответствующие вычислительные средства строятся по принципу операционного автомата с жесткой логикой. Такой подход связан с автономностью функционирования средств предварительной обработки от управляющей ЭВМ при неизменном алгоритме обработки и высокой интенсивности входного потока данных .

Второй тип систем используется, как правило, для систем, сочетающих средства предварительной (спецпроцессоры) и вторичной (ПЭВМ) обработки, когда требуется достаточно интенсивный обмен с оператором.

1.3. Основные типы алгоритмов цифровой обработки сигналов

Математические методы обработки сигналов можно подразделить на три группы, если в основу классификации положить принцип формирования отдельного элемента (отсчета) результата по некоторой совокупности элементов (отсчетов) исходного сигнала [21].

Точечные преобразования – в таких преобразованиях обработка каждого элемента исходных данных производится независимо от соседнего. Иначе говоря, значение каждого отсчета результата определяется как функция от одного отсчета исходного сигнала, причем номера отсчетов сигнала и результата одинаковы.

Иначе говоря, пусть требуется обработать вектор из n отсчетов сигнала:

$$X = [x_0 \quad x_1 \quad x_2 \quad \dots \quad x_n] \qquad (1.2)$$

и получить последовательность чисел:

$$Y = [y_0 \quad y_1 \quad y_2 \quad \dots \quad y_n], \qquad (1.3)$$

причем

$$y_i = f(x_i) \qquad (1.4)$$

Точечные преобразования достаточно просты и наименее громоздки с точки зрения вычислительных затрат. Если обрабатывается матрица размером N x N элементов отсчетов исходного двумерного сигнала, то вычислительная сложность процедуры точечных преобразований составит

$$Q_m = N^2 \text{ (БО)},$$

где под базовой операцией (БО) понимается операция вида (1.4).

Локальные преобразования - при локальных преобразованиях обеспечивается формирование каждого элемента матрицы или вектора результата как функции от некоторого множества соседних элементов матрицы или вектора отсчетов исходного сигнала, составляющих некоторую локальную окрестность. При этом полагается, что местоположение вычисляемого отсчета результата (или текущий индекс элемента) задается координатами (или текущими индексами) центрального элемента локальной окрестности. Для формирования следующего элемента матрицы результата выполняется смещение окрестности вдоль строки матрицы исходных данных или вдоль исходного вектора. Такая перемещаемая окрестность часто носит название окна сканирования. При обработке матрицы исходных данных после прохождения всей строки матрицы исходных данных окно сканирования смещается на одну строку и возвращается в начало следующей строки, после чего продолжается обработка. Просматриваемая при перемещении окна сканирования полоса строк матрицы носит название полосы сканирования. Иначе говоря, при такой обработке

$$y_i = F(\hat{X}) \; ; \; \hat{X} = \{x_{i-m/2}, \; x_{i-m/2+1}, \; \dots, \; x_i, \; \dots, \; x_{i+m/2-1}, \; x_{i+m/2}\}, \qquad (1.5)$$

где $i = 0,N-1$ - индекс отсчета результата , m - размер окна сканирования. Если $i<m/2$ или $i>N-m/2$, что имеет место на практике при обработке начальных и конечных отсчетов вектора исходного сигнала, то элементы вектора исходных данных с "недостающими" индексами полагаются равными нулю.

Вычислительная сложность локального преобразования составляет

$$Q_л = N^2 m^2 \quad (БО),$$

где под базовой операцией понимается выполнение заданного преобразования для отдельного отсчета исходных сигналов. Примером локальных преобразований могут служить апериодическая свертка или корреляция, а также процедуры ранговой фильтрации.

Глобальное преобразование предусматривает формирование каждого отсчета результата как функции от всей совокупности отсчетов исходного сигнала и некоторого множества меняющихся от одного отсчета результата к другому по определенному правилу коэффициентов, составляющих так называемое ядро преобразования. В случае обработки одномерного исходного сигнала глобальное преобразование можно определить как

$$Y_i = F(G_i, X); \qquad X = [x_0 \quad x_1 \quad x_2 \quad \dots \quad x_n], \qquad i = 0, N-1 \qquad (1.6)$$

где G_i – изменяемое ядро преобразования. Вычислительная сложность глобального преобразования в общем случае для случая обработки двумерного сигнала составляет

$$Q_г = N^4 \ (БО),$$

где под базовой операцией понимается выполнение заданного преобразования вида (1.6) для отдельного элемента исходных данных. Примером подобных преобразований могут служить дискретные ортогональные преобразования типа преобразования Фурье, Хартли, Адамара.

1.4. Линейные и нелинейные преобразования

Все преобразования ЦОС могут быть подразделены по своему типу на линейные и нелинейные преобразования [21,22, 28].

Пусть $x(t)$ - входная последовательность, а $y(t_1)$ - выходная последовательность, связанная со входной через некоторое функциональное преобразование T

$$y(t_1) = T[x(t)] \qquad (1.7)$$

Для линейных преобразований справедлив аддитивный закон :

$$T[ax_1(t) + bx_2(t)] = aT[x_1(t)] + bT[x_2(t)] = ay_1(t_1) + by_2(t_1) \ , \qquad (1.8)$$

где a и b – некоторые константы. Таким образом, линейное преобразование, применяемое к суперпозиции исходных сигналов эквивалентно по своему воздействию суперпозиции результатов преобразования каждого из сигналов. Свойство линейности является весьма важным для практических приложений, поскольку позволяет значительно упростить обработку различных сложных сигналов, являющихся суперпозицией некоторых элементарных сигналов. Так, в частности, за простейший элементарный сигнал может быть принят моногармонический сигнал $x(t)$, описываемый функцией:

$$x(t) = a\cos(2\pi\nu t - \varphi),$$

где a - амплитуда, $\nu = \frac{1}{T}$ - частота, T - период, , φ - начальная фаза.

$$\hat{x}(t) = \sum_{k=0}^{K} a_k \cos\left(2\pi\nu_k - \varphi_k\right)$$

Тогда более сложный полигармонический сигнал может быть записан как суперпозиция простейших моногармонических сигналов:

Важное место в цифровой обработке сигналов имеет некоторый идеализированный простейший импульсный сигнал, называемый дельта-функцией или единичным импульсом:

$$\delta(x) = \begin{cases} 0, x \neq 0 \\ 1, x = 0 \end{cases}$$

Согласно теории цифровой обработки сигналов, любой сигнал может быть представлен как суперпозиция взвешенных единичных импульсов следующим образом:

$$x(t) = \sum_{k=-\infty}^{\infty} x(t_k)\delta(t - t_k) \qquad (1.9)$$

где $x(t)$ – отсчет сигнала в некоторый момент времени.

Если на вход системы ЦОС, выполняющей линейное преобразование, поступает единичный импульс, то сигнал $h(t)$, снимаемый с выхода системы и являющийся откликом системы на единичный импульс, носит название импульсной характеристики (импульсного отклика) системы. Импульсный отклик является важнейшей характеристикой системы и позволяет описать ее как

“черный ящик”, задав реакцию системы на некоторый простейший эталонный сигнал.

Если *h(t)* конечна, то такие системы называются КИХ-системами, т.е. системами с конечной импульсной характеристикой. Если *h(t)* бесконечна, то это БИХ-системы, т.е. системы с бесконечной импульсной характеристикой. В цифровой обработке сигналов имеет смысл рассматривать только КИХ-системы, поскольку время обработки, т.е. реакции системы на входной сигнал должно быть конечно.

Подставив (1.9) в (1.7), получаем для линейных преобразований:

$$y(t_1) = T[x(t)] = T\left[\sum_{k=-\infty}^{\infty} x(t_k)\delta(t-t_k)\right] = \sum_{k=-\infty}^{\infty} x(t_k)T[\delta(t-t_k)] = \sum_{k=-\infty}^{\infty} x(t_k)h_k(t) \qquad (1.10)$$

Таким образом, для линейной системы результат обработки любого поступившего на вход сложного сигнала может быть определен как суперпозиция импульсных откликов системы на поступившие на вход единичные импульсы с соответствующей начальной задержкой и весом, определяемым весом соответствующего отсчета исходного сигнала.

Примерами линейных преобразований могут служить преобразования Фурье, Хартли, свертка и корреляция. К нелинейным преобразованиям относятся, в частности, многие алгоритмы распознавания, гистограммные преобразования и ранговая фильтрация.

1.5. Переход от непрерывных сигналов к дискретным

Процесс перехода от непрерывной области изменения аргумента (задания функции) к конечному множеству отдельных значений аргумента называется дискретизацией. Процесс перехода от непрерывной области изменения функции к конечному множеству определенных значений называется квантованием. Обычно полагается, что дискретизация и квантование выполняются с равными шагами, т.е. функция определена в равноотстоящих точках по оси абсцисс и по оси ординат. Переход от непрерывного сигнала к дискретному осуществляется с потерей информации. Восстановление непрерывного сигнала по дискретным значениям и устранение потерь информации зависит от параметров

дискретизации - т.е. шага дискретизации, способа восстановления сигнала и от свойств сигнала.

Условие, при котором возможно восстановление сигнала без потерь, определяется из теоремы Котельникова [22,28].

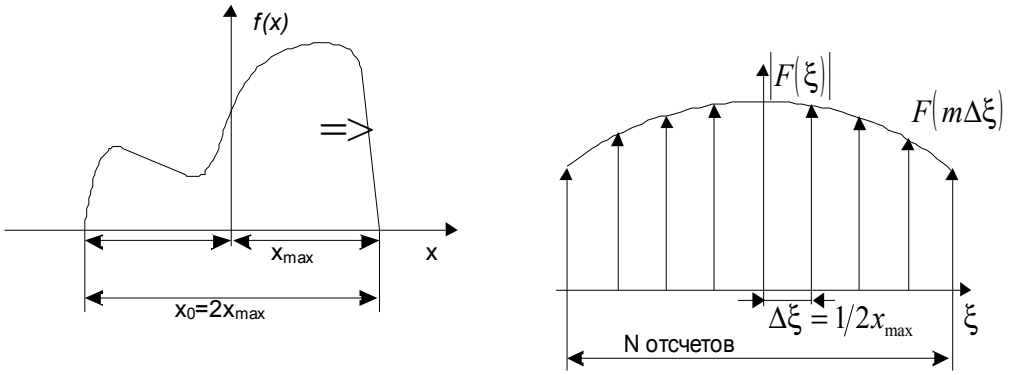
Пусть функции *f(x)* и *F(ξ)* связаны обратным преобразованием Фурье, т.е.

$$f(x) = \int_{-\xi_{max}}^{\xi_{max}} F(\xi) e^{j 2\pi \xi x} dx$$

Прямая формулировка теоремы Котельникова. Если функция *f(x)* имеет ограниченный спектр, локализованный в диапазоне $-\xi_{max} \le \xi \le \xi_{max}$, то она полностью определена путем задания отсчетов на наборе точек, отстоящих друг от друга на расстоянии $1/2\xi_{max}$.

Обратная формулировка теоремы Котельникова/ Если *f(x)* задана в ограниченной области $-x_{max} \le x \le x_{max}$, то ее спектр *F(v)* полностью определен набором отсчетов в точках, равноотстоящих друг от друга на расстоянии $1/2x_{max}$.

Поясним выбор шагов дискретизации по теореме Котельникова на рис.1.2.



ис.1.2. Выбор шага дискретизации по теореме Котельникова

При дискретизации согласно теореме Котельникова исходная функция *f(x)* может быть получена по ее дискретным значениям по формуле:

$$f(x) = \sum_{k=-\infty}^{\infty} f(k\Delta x) \frac{\sin 2\pi \nu \max(x - k\Delta x)}{\nu \max(x - k\Delta x)}, \qquad (1.11)$$

причем шаг дискретизации составляет

$$\Delta x = \frac{1}{2\nu\,\max} = \frac{\pi}{\omega\,\max}$$

Однако согласно теории Фурье-анализа конечной апериодической функции *f (x)* соответствует бесконечный спектр и наоборот, конечный спектр соответствует бесконечной исходной функции.

Поэтому для реальных сигналов условия теоремы Котельникова в строгом смысле слова не выполняются.

1. Все реальные сигналы ограничены во времени и имеют неограниченный спектр, т.е. $f_в = \infty$.
2. В соответствии с рядом Котельникова восстановление осуществляется по бесконечному числу отсчетов $(-\infty \le k \le \infty)$.
3. Поскольку сигнал восстанавливается по бесконечному числу отсчетов функций, то его восстановление осуществляется с бесконечной задержкой во времени.

Поэтому, чтобы получить конечный спектр, можно воспользоваться равенством Парсеваля :

$$\left|\int_{-\infty}^{\infty} f(x)dx\right|^2 \equiv \left|\int_{-\infty}^{\infty} F(\nu)d\nu\right|^2 (1-\varepsilon)\nu$$

и, зная, что исходная функция *f(x)* конечна, вычислить значение интеграла в левой части равенства, после чего, задав величину погрешности ε в определении интеграла в правой части, определить максимальную частоту. Исходя из максимальной частоты определяется число отсчетов. Если размер области задания исходной функции *f(x)* равен $X=2X_{max}$, а число отсчетов функции при дискретизации должен составлять N , то шаги дискретизации исходной функции и ее спектра составят:

$$\begin{cases} \Delta\nu \le \dfrac{1}{2x_{\max}}; \nu_{\max} = \dfrac{N}{2}\Delta\nu; \\ \Delta x \le \dfrac{1}{2\nu_{\max}}; x_{\max} = \dfrac{N}{2}\Delta x; \end{cases} \quad (1.12)$$

Итак, в результате дискретизации в соответствии с теоремой Котельникова от *x(t)* мы переходим к набору отсчетов или к вектору:

$$X = \{x_n\};\ X = \begin{bmatrix} x_0 & x_1 & x_2 & \dots & x_{N-1} \end{bmatrix}.\ n=0,\ N-1.$$

1.6. Циклическая свертка и корреляция

В дискретном виде линейные преобразования могут быть описаны в общем виде как векторно-матричные операции [12, 21]:

$$Y = B_N X \qquad (1.13)$$

где X - вектор отсчетов исходных данных, полученный в результате дискретизации непрерывного сигнала согласно теореме Котельникова, Y - вектор отсчетов результата, B_N - матрица размера $N \times N$, определяющая ядро выполняемого преобразования.

К числу подобных преобразований относится циклическая свертка последовательностей $X = \begin{bmatrix} x_0 & x_1 & x_2 & \dots & x_{N-1} \end{bmatrix}$ и $G = \begin{bmatrix} g_0 & g_1 & g_2 & \dots & g_{N-1} \end{bmatrix}$, в этом случае строится матрица ядра свертки:

$$B_N^C = G_N = \begin{vmatrix} g_0 & g_{N-1} & g_{N-2} & \cdots & g_1 \\ g_1 & g_0 & g_{N-1} & \cdots & g_2 \\ g_2 & g_1 & g_0 & \cdots & g_3 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ g_{N-1} & g_{N-2} & g_{N-3} & \cdots & g_0 \end{vmatrix}$$

Каждый элемент вектора Y может быть описан как:

$$y_m = \sum_{n=0}^{N-1} g_{m-n} x_n \ ;\ m = \overline{0, N-1} \qquad . \qquad (1.14)$$

Матрица ядра циклической взаимокорреляции может быть построена как транспонированная матрица ядра свертки, т.е. следующим образом:

$$K_N = B_N^K = \begin{vmatrix} g_0 & g_1 & g_2 & \cdots & g_{N-1} \\ g_{N-1} & g_0 & g_{N-1} & \cdots & g_{N-2} \\ g_{N-2} & g_{N-1} & g_0 & \cdots & g_{N-3} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ g_1 & g_2 & g_3 & \cdots & g_0 \end{vmatrix}$$

Поэтому каждый отсчет результата может быть записан как:

$$z_m = \sum_{n=0}^{N-1} g_{m+n} x_n \; ; \; m = \overline{0, N-1} \qquad (1.15)$$

причем $g_{m+n} = g_n$ для $m + n \geq N$.

1.7. Апериодическая свертка и корреляция

Апериодическая свертка и корреляция в отличие от циклической относятся к лассу локальных преобразований. При этом как правило полагается, что размер вектора исходных данных значительно больше размера ядра свертки, что приводит к следующему выражению для вычисления любого отсчета результата:

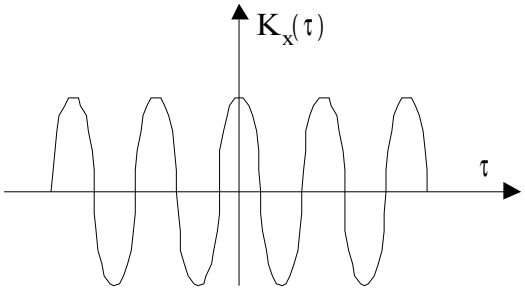
$$y_i = \sum_m g_m x_{i-m} \qquad (1.16)$$

Вычисление свертки и корреляции лежит в основе корреляционного метода подавления помех.

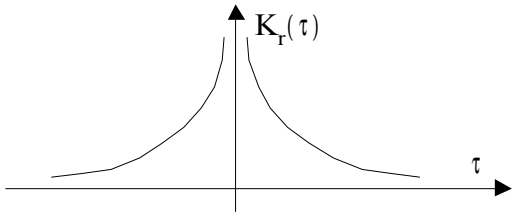
Сущность такого метода заключается в использовании различия между корреляционными функциями сигнала и помехи. Данный метод эффективен лишь в случае обработки периодических или квазипериодических сигналов.

Рассмотрим сущность метода на примере, когда полезный сигнал является гармоническим, а помеха - типа белого гауссова шума [22].

Автокорреляционная функция сигнала является тоже гармонической и имеет ту же частоту



Автокорреляционная функция помехи имеет вид



Метод автокорреляционного приема основан на анализе автокорреляционной функции принятого сигнала *y(t)=x(t)+p(t).*

$$K_y(\tau) = \lim_{T_x \to \infty} \frac{1}{T_x} \int_0^{T_x} y(t)y(t+\tau)dt =$$

$$= \lim_{T_x \to \infty} \frac{1}{T_x} \int_0^{T_x} [x(t)+r(t)][x(t+\tau)+r(t+\tau)]dt =$$

$$= \lim_{T_x \to \infty} \frac{1}{T_x} \int_0^{T_x} x(t)x(t+\tau)dt +$$

$$+ \lim_{T_x \to \infty} \frac{1}{T_x} \int_0^{T_x} x(t)r(t+\tau)dt +$$

$$+ \lim_{T_x \to \infty} \frac{1}{T_x} \int_0^{T_x} x(t+\tau)r(t)dt +$$

$$+ \lim_{T_x \to \infty} \frac{1}{T_x} \int_0^{T_x} r(t)r(t+\tau)dt.$$

Второе и третье слагаемые - взаимные корреляционные функции сигнала и помехи. Если сигнал и помеха взаимно независимы (типичный для практики случай), то второе и третье слагаемые равны 0 и, следовательно,

$$K_y(\tau) = K_x(\tau) + K_r(\tau),$$

т.е автокорреляционная функция принятого сигнала равна сумме автокорреляционных функций сигнала и помехи.

Метод корреляционного приема позволяет обнаружить полезный сигнал, который имеет мощность значительно меньшую, чем мощность помехи.

1.8. Двумерная апериодическая свертка и корреляция

Важное место среди операций линейной обработки сигналов занимает операция перемножения матриц одинаковой размерности. Такая операция имеет вид:

$$Z_N = B_N X_N \text{ , (1.17)}$$

где B_N и X_N исходные матрицы порядка *N*, а Z_N - результирующая матрица того же порядка. Каждый элемент матрицы Z_N формируется в соответствии с выражением:

$$z_{ij} = \sum_k x_{ij} b_{kj} \quad , \quad k, i, j = 1, N \qquad (1.18)$$

где x_{ik}, b_{kj} - элементы исходных матриц.

На основе операций (1.17) и (1.18) выполняется вычисление функций двумерной апериодической свертки/корреляции исходного двумерного сигнала (изображения) с двумерным ядром. Подобное преобразование часто используется как для удаления шумов, так и для выделения мелких объектов.

Математически двумерная апериодическая свертка может быть описана следующим образом:

$$y_{ij} = \sum_{n=1}^{M} \sum_{m-1}^{M} g_{mn} x_{i+m-1, j+n-1} \qquad (1.19)$$

где y_{ij} - отсчеты результатов вычислений (отсчеты свертки), g_{mn} - отсчеты весовой функции окна (ядра свертки) размерностью *M x M* отсчетов, причем *N >> M*. Очевидно, что размерность матрицы, описывающей двумерную свертку, равна *(N + M-1) x (N + M -1)* отсчетов. Поэтому матрица исходных данных также должна быть дополнена до указанного размера нулевыми элементами по краям кадра.

Отсчеты свертки формируются при перемещении окна вдоль строки исходного изображения. Для каждого положения окна формируется один отсчет свертки, после чего окно сдвигается на один элемент вдоль строки (т.е. на один столбец). Обработка начинается с элемента x исходного изображения. После прохождения i-й строки изображения *(i= 1,N)* окно смещается на одну строку вниз и возвращается к началу следующей (i + 1)-й строки изображения. По окончании обработки кадра изображения окно перемещается в исходное положение.

При вычислении свертки можно распараллелить вычисления по столбцам окна сканирования, выполняя параллельно вычисление произведений столбцов

ядра свертки $\hat{g}_n$ и столбцов $\hat{X}_{j+n-1}$ текущей полосы сканирования изображения. Поэтому для окончательного вычисления отсчета свертки достаточно сформировать сумму:

$$Y_j = \sum_n \hat{g}_n \hat{X}_{j+n-1} \qquad (1.20)$$

соответствующую отсчету свертки с номером *j*, расположенному в средней строке полосы шириной *M*.

Представляет интерес распараллеливание по разрядным срезам, поскольку в этом случае практически исключается операция умножения [17]. При разрядно-срезовой обработке данные должны быть представлены в формате с фиксированной точкой. Представим значение элемента изображения в следующем виде :

$$x_{mn} = \sum_q b_{mn}^q 2^{q-1}, \quad (1.21)$$

где b_{mn}^q - q-ый разряд x_{mn} *(q= 1,..., Q), (g=(1,Q))* где Q - разрядность данных.

С учетом этого процедура вычисления свертки принимает вид:

$$y_{ij} = \sum_q 2^{q-1} \sum_m \sum_n g_{mn} b_{i+m-1,\, j+n-1}^q = \sum_q^Q 2^{q-1} Y_q \qquad (1.22)$$

Таким образом, процедура двумерной апериодической свертки для одного положения окна сводится к *M x N x Q* операциям сложения и *(Q-1)* операциям сдвига. Умножение под знаком суммы сводится к операции вида :

$$g_{mn} b_{mn}^q = g_{mn} \text{ при } b_{mn}^q = 1 \text{ и } g_{mn} b_{mn}^q = 0 \text{ при } b_{mn}^q = 0.$$

Поэтому разрядно-срезовый алгоритм вычисления свертки или корреляции для одного положения окна может быть представлен в виде [17]:

начало;

$S := 0;$

для $q = 1, \ldots, Q$ *цикл:*

$S1 := 0;$

для $m = 1, \ldots, M$ *цикл:*

для $n = 1, \ldots, M$ *цикл:*

если $b_{i-m,\, j-n}^q = 1$, *то* $g_{mn} := g_{mn}$

иначе $g_{mn} := 0$;

$S1 := S1 + g_{mn}$;

конец цикла по n ;

конец цикла по m ;

$S = S + sh1_{q-1}(S1)$;

конец цикла по q ;

$y_{ij} = S$;

конец.

Физический смысл функций свертки и кросскорреляции состоит в том, что они являются количественной мерой совпадения (сходства) двух последовательностей *f(x)* и *g(x).* При этом наиболее полно мера сходства может быть определена по функции корреляции, в связи с чем функция взаимокорреляции может быть использована для распознавания сигналов. Если распознаваемый сигнал *f(t)* точно соответствует эталонному сигналу *g(t)*, то результирующий сигнал $\omega^k(t)$ принимает значение:

$\omega^k(t)$=max при *f(t)*≡ *g(t),*

что соответствует функции автокорреляции. Если сигналы отличаются, то $\omega^k(t)$ <max.

Кроме того, при обработке двумерных сигналов (изображений объектов) координаты максимума данной функции определяют центр тяжести исходного распознаваемого объекта, что позволяет определить и его местоположение (т.е. запеленговать объект). По этим причинам вычисление одномерной или двумерной корреляции лежит в основе целого ряда методов распознавания.

Таким образом, функции линейной апериодической свертки и корреляции полезны для распознавания сигналов заданной формы. На этом принципе работают корреляционные методы распознавания. Свертка определяется путем скольжения эталона по вектору исходного сигнала, и максимум функции будет тогда, когда исходный сигнал совпал с эталоном. Функция апериодической свертки, кроме того, оказывается полезной для удаления, например, низкочастотных помех.

2. ДИСКРЕТНЫЕ ОРТОГОНАЛЬНЫЕ ПРЕОБРАЗОВАНИЯ

2.1. Введение в теорию ортогональных преобразований

Две вещественные функции *g(x)* и *h(x),* заданные на конечном или бесконечном интервале *(a<x<b)*, называются ортогональными друг другу на этом интервале, если

$$\int_a^b g(x)h(x)dx = 0$$

При этом функции предполагаются конечными либо бесконечными, но обязательно с абсолютно сходящимся интегралом. Интеграл называется абсолютно сходящимся если

$$\int_a^\infty |f(x)|dx < \infty .$$

Система функций называется ортогональной на некотором интервале, если каждые две функции из этой системы ортогональны друг другу на этом интервале.

Пусть задана система функций

$$g_1(x), g_2(x), \ldots, g_n(x) \qquad (2.1)$$

ортогональная на некотором интервале *(a<x<b)*.

Может возникнуть задача о разложении произвольной функции *f(x)* на этом интервале в ряд по функциям (2.1), т.е. в ряд вида

$$f(x) = a_1 g_1(x) + a_2 g_2(x) + \ldots + a_n g_n(x) + \ldots = \sum_{n=1}^{\infty} a_n g_n(x), \qquad (2.2)$$

где a_n - числовые коэффициенты. При этом возникают вопросы: возможно ли разложение для любой функции *f(x)* и как найти коэффициенты a_n.

Будем считать для простоты все рассматриваемые функции, а также интервал конечным. Ответ на первый вопрос зависит от выбора системы, по которой мы будем производить разложение. Если разложение возможно для любой функции *f(x)*, то система функций называется полной.

Перейдем теперь к нахождению коэффициентов a_n разложения, причем будем считать, что ни одна из функций (2.1) не равна тождественно нулю. Для

этого умножим обе части уравнения (2.2) на $g_n(x)$ и проинтегрируем результат по интервалу $(a<x<b)$.

$$\int_a^b f(x)g_n(x)dx = a_1\int_a^b g_1(x)g_n(x) + a_2\int_a^b g_2(x)g_n(x) + \ldots + a_n\int_a^b g_n^2(x) + \ldots$$

В силу ортогональности системы (2.1), в правой части последнего равенства все интегралы равны нулю, за исключением интеграла от $g_n^2(x)$, и мы получаем формулу для коэффициентов

$$a_n = \frac{\int_a^b f(x)g_n(x)dx}{\int_a^b g_n^2(x)dx} \quad (n=1,2,3,\ \ldots).$$

2.2. Спектральное преобразование Фурье

Любой периодический сигнал $x_n(t)$ можно представить в виде ряда Фурье [22]:

$$x_П(t) = \frac{A_0}{2} + \sum_{k=1}^{\infty} A_k \cos(k\omega_0 t - \varphi_k), \quad (2.3)$$

где $A_0 = \frac{2}{T}\int_{t_1}^{t_1+T} x_n(t)dt$ - постоянная составляющая сигнала, $\omega_0 = 2\pi/T = 2\pi\nu$. Из выражения (2.3) следует, что периодический сигнал любой формы может быть представлен в виде суммы гармонических составляющих с различными амплитудами A_k , частотами $k\omega_0$ - и фазами φ_k. Благодаря этому можно перейти от представления сигнала во временной области к частотной, где $A_k = A_k(k\omega_0)$ - спектр амплитуд, $\varphi_k = \varphi_k(k\omega_0)$ - спектр фаз.

Эти спектры являются линейчатыми. Для любого сигнала $x(t)$ можно определить спектры амплитуд A_k и фаз φ_k .следующим образом:

$$A_k(k\omega_0) = \sqrt{a_k^2 + b_k^2};$$

$$\varphi_k(k\omega_0) = arctg\ b_k/a_k.$$

где, в свою очередь, величины a_k и b_k определяются следующим образом:

$$a_k = \frac{2}{T}\int_{t_1}^{t_1+T} x_П(t)\cos k\omega_0 t dt; \qquad b_k = \frac{2}{T}\int_{t_1}^{t_1+T} x_П(t)\sin k\omega_0 t dt;$$

Любой периодический сигнал бесконечен во времени, что на практике не осуществимо, поэтому периодический сигнал - математическая абстракция, и все рассмотренное выше не применимо к реальным сигналам.

Реальный сигнал ограничен во времени и, следовательно, является непериодическим. Однако, условно его можно рассматривать как периодический с периодом $T\to\infty$. Тогда $\omega_0=2\pi/T \to 0$, а спектры амплитуд и фаз становятся непрерывными (сплошными), сумма в разложении Фурье превращается в интеграл. В результате переходим к интегралу Фурье (обратное преобразование) [22]:

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

В формуле $s(j\omega)$ - спектральная плотность сигнала, определяющая как распределяются амплитуды и фазы по частотам непрерывного спектра. Иногда в задачах обработки сигналов ее называют фурье-образом или фурье-спектром сигнала.

От $s(j\omega)$ можно перейти к спектральной плотности амплитуд ($s(\omega)$) и фаз ($\varphi(\omega)$).

Для решения этой задачи используется прямое преобразование Фурье:

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

Важно отметить, что $s(\omega)$ -всегда убывающая функция, а $\varphi(\omega)$ - всегда неубывающая функция. Кроме того, $s(\omega)=s(-\omega)$ - четная функция; а $\varphi(\omega) = -\varphi(-\omega)$ - нечетная функция.

Таким образом, с точностью до постоянного коэффициента, прямое и обратное преобразование Фурье могут быть определены по соотношениям

- прямое преобразование Фурье:

$$F(\xi) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi\xi x} dx \qquad (2.4)$$

- обратное преобразование Фурье:

$$f(x) = \int_{\infty}^{\infty} F(\xi) e^{j2\pi\xi x} d\xi \qquad (2.5)$$

где ξ – некоторая пространственно-частотная спектральная координата (аналог координаты v для сигналов во временной области). Напомним основные общие свойства преобразования Фурье:

Инвариантность к линейному смещению. Это свойство преобразования Фурье позволяет получать неизменный квадрат Фурье-образа при перемещении исходного объекта вдоль осей координат, что имеет исключительно важное значение при обработке и распознавании образов:

$$f(x) \rightarrow F(\xi)$$

$$f(x-c) \rightarrow F_1(\xi) = F(\xi) e^{-j2\pi\xi c}$$

или $|F(\xi)| \equiv |F_1(\xi)|$, где c - произвольное смещение функции вдоль оси.

Теорема масштабов или теорема подобия. Теорема определяет характер изменения спектра при изменении масштаба сигнала

$$f(x) \rightarrow F(\xi)$$

$$f(mx) \rightarrow F_1(\xi) = F(\xi/m)$$

т.е. при растяжении функции $f(x)$ в m раз происходит сжатие в m раз ее Фурье-образа, и наоборот.

Теорема о свертке. Пускай требуется вычислить свертку функций $f(x)$ и $g(x)$, причем $F(\xi)$и $G(\xi)$ - соотвентственно их Фурье-образы:

$$\omega_c = f(x) \times g(x) = \int_{-\infty}^{\infty} f(x - x_1) g(x_1) dx_1$$

$$f(x) \rightarrow F(\xi); \; g(x) \rightarrow G(\xi)$$

Если функции $F(\xi)$ и $G(\xi)$ есть Фурье-образы функции $f(x)$ и ядра $g(x)$ соответственно, то $F(\xi)G(\xi)$ есть Фурье-образ свертки $\omega_c(x)$.

Теорема о корреляции. Если функции $F(\xi)$ и $G(\xi)$ есть Фурье-образы функции *f(x)* и ядра *g(x)* соответственно, то $F^*(\xi)G(\xi)$ есть Фурье образ корреляции $\omega_k(x)$ функций *f(x)* и *g(x)*.

Теоремы о свертке и корреляции свидетельствуют о возможности вычисления функций свертки и корреляции через преобразование Фурье. Тем самым на основе преобразования Фурье можно выполнять и распознавание, идентификацию сигналов и обнаружение координат источника сигналов.

Теорема Парсеваля или закон сохранения энергии. Эта теорема свидетельствует о том, что мощность исходного сигнала и мощность спектра сигнала одинаковы:

$$\left|\int_{-\infty}^{\infty} f(x)\,dx\right|^2 \equiv \left|\int_{-\infty}^{\infty} F(\xi)\,d\xi\right|^2$$

Преобразование Фурье является одним из важнейших ортогональных преобразований, используемых в цифровой обработке сигналов. Действительно, вполне физически ясен смысл перехода от временного описания исходного сигнала к его частотному описанию. Кроме того, двумерное преобразование Фурье описывает не что иное, как дифракцию электромагнитных и упругих волн в дальней зоне (дифракцию Фраунгофера) – т.е. на большом (по сравнению с размерами источника и длиной волны) расстоянии от источника [28].

2.3. Преобразование Хартли

Для одномерного случая прямое преобразование Хартли может быть определено как [3]

$$H(\xi) = \int_{-\infty}^{\infty} f(x)\left[\cos 2\pi\xi x + \sin 2\pi\xi x\right]dx \qquad (2.6)$$

и, соответственно, обратное преобра-зование Хартли

$$f(x) = \int_{-\infty}^{\infty} H(\xi)\left[\cos 2\pi\xi x + \sin 2\pi\xi x\right]d\xi \qquad (2.7)$$

Сравним эти выражения с (2.4.) и (2.5), разложив ядро по формуле Эйлера на действительную и мнимую части (т.е. sin и cos компоненты):

$$F(\xi) = \int_{-\infty}^{\infty} f(x)[\cos 2\pi\xi x - j\sin 2\pi\xi x]\,dx$$

$$f(x) = \int_{-\infty}^{\infty} F(\xi)[\cos 2\pi\xi x + j\sin 2\pi\xi x]\,d\xi \qquad (2.8)$$

Из анализа (2.6) - (2.8), можно сделать следующие выводы:

1) Преобразование Хартли является преобразованием с действительным ядром;

2) Прямое и обратное преобразование Хартли вычисляются идентично;

3) Квадрат модуля преобразования Фурье $|F(\nu)|^2$ равен:

$$|F(\nu)|^2 = \frac{H^2(\nu) + H^2(-\nu)}{2} \qquad (2.9)$$

4) Действительная и мнимая компоненты преобразования Фурье могут быть вычислены на основе преобразования Хартли весьма простым образом:

$$\mathrm{Re}\{F(\xi)\} = \frac{H(\xi) + H(-\xi)}{2} \qquad (2.10)$$

(2.11)

$$\mathrm{Im}\{F(\xi)\} = \frac{H(\xi) - H(-\xi)}{2}$$

5) Если *f(x)* - четная (т.е. *f(-x)=f(x)*), то:

$$H(\xi) \equiv \mathrm{Re}\{F(\xi)\}, \quad \mathrm{Im}\{F(\xi)\} \equiv 0.$$

Основные свойства преобразования Хартли соответствуют преобразованию Фурье:

1) Инвариантность к сдвигу (модуль $H^2(\xi) + H^2(\xi)$ - неизменен).

2) Так же, как и для преобразования Фурье, для преобразования Хартли справедливы следующие соотношения согласно теоремы масштабов:

$$f(x) \to H(\xi)$$

$$f(m_x x) \to H_1(\xi) = H(\xi/m_x)/m_x$$

3) Так же, как и для преобразования Фурье, для преобразования Хартли справедлива Теорема Парсеваля.

Отличие от Фурье - преобразования заключается в иной трактовке теоремы о свертке:

Если заданы функции *f(x)* и *g(x)*, причем *H(ξ)* и $G(\xi)$ - соответственно их спектры Хартли:

$$f(x) \to H(\xi),$$

$$g(x) \to G(\xi),$$

то их свертка вычисляется следующим образом [3]:

1) вычисляются функции $H(-\xi)$ и $G(-\xi)$;

2) формируется функция:

$$\Phi(\xi) = \frac{\left[\left(H(\xi) + H(-\xi)\right)G(\xi) + \left(H(\xi) - H(-\xi)\right)G(-\xi)\right]}{2}$$

3) Вычисляется преобразование Хартли от функции *Ф(ξ)*.

Очевидно, что если функция *g(x)* - четная, то:

$$G(\xi) \equiv G(-\xi)$$

$$\Phi(\xi) = H(\xi)G(\xi),$$

Если и функция *f(x)* - четная, то:

$$H(\xi) \equiv H(-\xi)$$

$$\Phi(\xi) = G(\xi)H(\xi)$$

Преобразование Хартли требует вычислений примерно вдвое меньшей сложности (поскольку его ядро действительная функция) и в то же время от его результата достаточно просто перейти к результату, эквивалентному результату преобразования Фурье. Поэтому на практике преобразование Хартли используется вместо преобразования Фурье в различных задачах ЦОС как некоторое искусственное синтетическое преобразование меньшей сложности, но обеспечивающее получение требуемого результата.

2.4. Дискретное преобразование Фурье

Перейдём от интегрального преобразования Фурье (2.3) к дискретному преобразованию Фурье (ДПФ), при условии что точки дискретизации выбраны согласно теоремы отсчётов (теоремы Котельникова) [22]:

$$F(\Delta\xi)=\sum_{k=0}^{N-1}x(k\Delta x)\delta(x-k\Delta x)*e^{-j2\pi k\Delta\xi\Delta x}=\sum_{k=0}^{N-1}x_k e^{-j2\pi\frac{2x_{\max}*k}{N}*\frac{1}{2x_{\max}}n}=$$

$$=k*\sum_{k=0}^{N-1}x_k*e^{-j\frac{2\pi}{N}kn}$$

$$\text{где}\begin{cases}\dfrac{2x_{\max}}{N}=\Delta x\\ \dfrac{1}{2x_{\max}}=\Delta\xi\end{cases}\qquad(2.11)$$

Тогда нетрудно получить,что

$$\Delta x*\Delta\xi=\frac{1}{N}$$

Подобным же образом можно получить и для обратного преобразования

$$x_m=k\sum F_n*e^{j\frac{2\pi}{N}nm}\qquad(2.12)$$

$$k=\frac{1}{\sqrt{N}}$$

Заметим, что происхождение множителя $k=\frac{1}{\sqrt{N}}$ связано с заменой при дискретизации согласно теоремы котельникова восставливающую функцию в (1.12) на “гребёнку” отсчётов.

Таким образом, в матричной форме:

$$F=\frac{1}{\sqrt{N}}E_N X\qquad(2.13)$$

$$E_N = \begin{vmatrix} W^0 & W^0 & W^0 & \dots & W^0 \\ W^0 & W^1 & W^2 & \dots & W^{N-1} \\ W^0 & W^2 & W^4 & \dots & W^{2(N-1)} \\ W^0 & W^3 & W^6 & \dots & W^{3(N-1)} \\ \dots & \dots & \dots & \dots & \dots \\ W^0 & W^{N-1} & W^{2(N-1)} & \dots & W^{(N-1)(N-1)} \end{vmatrix}$$

где $W^k = e^{-j2\pi k/N}$, а сама матрица ядра ДПФ носит название матрицы дискретных экспоненциальных функций (ДЭФ). При этом строки матрицы определяют набор ортогональных функций или базис разложения.

При выполнении преобразования Фурье строки матрицы ядра задают набор ортогональных функций, по которым выполняется разложение исходного сигнала. Каждый элемент вектора результата определяет вклад соответствующей ортогональной функции в формирование исходного сигнала.

Для преобразования Фурье, как и для любого ортогонального преобразования, матрица ядра преобразования E_N обратима (т.е. определитель отличен от “0”) , что позволяет выполнить как прямое, так и обратное преобразования:

$$\begin{cases} F = \dfrac{1}{\sqrt{N}} E_N X \\ X = \dfrac{1}{\sqrt{N}} E_N^{-1} F \end{cases} \qquad (2.14),$$

поскольку $\dfrac{1}{N} E_N * E_N^{-1} = \dfrac{1}{N} I_N = \begin{vmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \end{vmatrix}$

При этом матрица ядра обратного преобразования обладает свойством $E_N^{-1} \equiv E_N^*$, где E_N^* - эрмитово -сопряжённая матрица. Понятие эрмитово-сопряженной матрицы предусматривает, что матрица обратного преобразования является транспонированной по отношению к E_N и элементы её есть комплексно сопряжённые к W_{ij}.

Рассмотрим основные свойства матрицы ядра преобразования E_N. Коэффициенты такой матрицы обладают следующими свойствами:

1) цикличностью: $W^{k+N} \equiv W^k$ или $W^{-k} \equiv W^{N-k}$

2) мультипликативностью $W^{k+m} \equiv W^k * W^m$

Из указанных свойств следует, матрица E_N из N^2 элементов содержит только N попарно различных элементов.

3) симметричностью $W_{ij} \equiv W_{ji}$

Рассмотрим примеры матрицы E_N для некоторых N:

N=2

$$E_2 = \begin{vmatrix} 1 & 1 \\ 1 & -1 \end{vmatrix}$$

N=3

$$E_3 = \begin{vmatrix} W^0 & W^0 & W^0 \\ W^0 & W^1 & W^2 \\ W^0 & W^2 & \underline{W^1} \end{vmatrix}$$

$$W^4 \equiv W^{3+1} \equiv W^1$$

N=4

$$E_4 = \begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{vmatrix}$$

и, наконец, для *N=8*

$$E_8 = \begin{vmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \frac{\sqrt{2}}{2}(1-j) & -j & -\frac{\sqrt{2}}{2}(1+j) & -1 & \frac{\sqrt{2}}{2}(-1+j) & j & \frac{\sqrt{2}}{2}(1+j) \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ 1 & -\frac{\sqrt{2}}{2}(1+j) & j & \frac{\sqrt{2}}{2}(1-j) & -1 & \frac{\sqrt{2}}{2}(1+j) & -j & \frac{\sqrt{2}}{2}(-1+j) \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & \frac{\sqrt{2}}{2}(1-j) & j & \frac{\sqrt{2}}{2}(1+j) & -1 & \frac{\sqrt{2}}{2}(1-j) & -j & \frac{\sqrt{2}}{2}(-1-j) \\ 1 & j & -1 & -j & 1 & -j & -1 & -j \\ 1 & \frac{\sqrt{2}}{2}(1+j) & j & \frac{\sqrt{2}}{2}(-1+j) & -1 & -\frac{\sqrt{2}}{2}(1+j) & -j & \frac{\sqrt{2}}{2}(1-j) \end{vmatrix}$$

2.5. Дискретное преобразование Хартли

Дискретное преобразование Хартли имеет вид [3]:

- прямое преобразование

$$h_n = \sum_{k=0}^{N-1} x_k \cos\frac{2\pi kn}{N} + \sin\frac{2\pi kn}{N} = \sum_{k=0}^{N-1} x_k cas\frac{2\pi kn}{N} \qquad (2.15)$$

$$cas[\ldots] = \cos[\ldots] + \sin[\ldots]$$

- обратное преобразование

$$x_m = \sum_{n=0}^{N-1} h_n cas\frac{2\pi nm}{N}$$

Матрица ядра преобразования Хартли может быть записана как:

$$C_N = \begin{vmatrix} C_0 & C_0 & C_0 & \ldots & C_0 \\ C_0 & C_1 & C_2 & \ldots & C_{N-1} \\ C_0 & C_2 & C_4 & \ldots & C_{2(N-1)} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ C_0 & C_{N-1} & C_{2(N-1)} & \ldots & C_{(N-1)^2} \end{vmatrix}$$

где $C_k = cas\frac{2\pi k}{N}$, причем $C_0 \equiv 1$

Матрица ядра преобразования обладает следующими свойствами:

1) цикличностью $C_{k+N} \equiv C_k$; $C_{N-k} \equiv C_{-k}$;

2) отсутствием мультипликативности, т.е.: $C_{k+l} \neq C_k * C_l$

3) симметричностью $C_{ij} \equiv C_{ji}$

Отсюда следует, что обратная матрица $C_N^{-1} \equiv C_N$, поскольку $C_N \equiv C_N^T$.

4) из свойства цикличности следует, что в матрице C_N имеется лишь N различных между собой коэффициентов C_{ij} из N^2

Для $N=2$, $N=4$ и $N=8$ матрицы ядра преобразования будут иметь вид:

$$N=2 \qquad C_2 = \begin{vmatrix} 1 & 1 \\ 1 & -1 \end{vmatrix} = E_2$$

$$N=4 \qquad C_4 = \begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{vmatrix} \neq E_4$$

$$N = 8 \quad C_8 = \begin{vmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \sqrt{2} & 1 & 0 & -1 & -\sqrt{2} & -1 & 0 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & 0 & -1 & \sqrt{2} & -1 & 0 & 1 & -\sqrt{2} \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -\sqrt{2} & 1 & 0 & -1 & \sqrt{2} & -1 & 0 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 0 & -1 & -\sqrt{2} & -1 & 0 & 1 & \sqrt{2} \end{vmatrix}$$

Таким образом, в отличии от матрицы ДЭФ, матрица ядра преобразования Хартли содержит коэффициенты, большие единицы.

2.6. Двумерные дискретные преобразования Фурье и Хартли

Для обработки двумерных сигналов, в частности, изображений, широко используются двумерные спектральные преобразования. Двумерное дискретное преобразование Фурье имеет вид [12, 21]:

$$f_{k,l} = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} x_{m,n} e^{-j\frac{2\pi}{N}[km+\ln]} \qquad (2.16)$$

Поскольку ядро преобразования Фурье является разделимым по переменным интегрирования, то для выполнения двумерного ДПФ на практике используется строчно - столбцовый метод.

Действительно,

$$e^{-j\frac{2\pi}{N}[km+\ln]} = e^{-j\frac{2\pi}{N}km} e^{-j\frac{2\pi}{N}\ln}$$

откуда получаем с учетом (2.16):

$$f_{k,l} = \sum_{n=0}^{N-1} \left[\sum_{m=0}^{N-1} x_{mn} e^{-j\frac{2\pi}{N}km} \right] e^{-j\frac{2\pi}{N}\ln}$$

Введя обозначение

$$\hat{f}_{k,l} = \sum_{m=0}^{N-1} x_{mn} e^{-j\frac{2\pi}{N}km} \quad (2.17)$$

получим:

$$f_{k,l} = \sum_{n=0}^{N-1} \hat{f}_{k,l} e^{-j\frac{2\pi}{N}\ln} . \quad (2.18)$$

Согласно такому подходу, вначале выполняются одномерные ДПФ по строкам матрицы с отсчетами изображения, а затем выполняются одномерные ДПФ по столбцам матрицы промежуточных результатов, представляющих собой одномерные спектры по строкам изображения. В матричном виде это может быть записано следующим образом [12]:

$$F_N = [E_N X_N] E_N \qquad (2.19).$$

Из (2.19) следует, что двумерное ДПФ сводится вначале к одномерному ДПФ матрицы X по столбцам, а затем к одномерному ДПФ по строкам матрицы промежуточных результатов $\hat{F}_N$.

Сложность вычислений на первом этапе составит N раз по N^2 базовых операций, под которыми понимаются операции вычисления выражения под знаком суммы в (2.17) и (2.18), и столько же на втором, откуда: $Q_{ДПФ} = 2N^3$ *(Б.О.)*

В то же время непосредственные вычисления двумерного ДПФ по формуле (2.16) требуют вычислительных затрат: $Q^1_{ДПФ_2} = N^4 (Б.О.)$

Таким образом, строчно - столбцовый метод позволят существенно снизить вычислительную сложность алгоритма двумерного ДПФ с N^4 до $2N^3$ операций умножения. Вычисление одномерного ДПФ по каждой из координат преобразование выполняется на основе процедуры быстрого преобразования Фурье. Такой подход, однако, требует выполнения дополнительной процедуры транспонирования матрицы промежуточных результатов после выполнения обработки по одной из координат матрицы. При этом преобразование по следующей координате может выполняться только после того, как сформирована вся матрица промежуточных результатов и выполнено ее транспонирование.

Поэтому использование свойства разделимости ядра двумерного ДПФ по переменным суммирования приводит к снижению сложности вычислений *в N/2* раз.

Однако далеко не все ядра двумерных спектральных ортогональных преобразований в явном виде обладают свойством разделимости. К таким

преобразованиям относится, например, двумерное дискретное преобразование Хартли [3]:

$$H_{kl} = \sum_m \sum_n x_{mn} \left[\; cas(2\pi(km + \ln) / N \; \right] \qquad (2.20)$$

где $cas[\ldots] = \cos[\ldots] + \sin[\ldots]$

Рассмотрим применение специального или модифицированного преобразования Хартли, в котором искусственно выполнено разделение ядра по координатам [3]:

$$\hat{H}_{kl} = \sum_n^N \left[\; \sum_m^N x_{mn} cas(2\pi km \; / \; N) \; \right] cas(2\pi \ln / \; N) \qquad (2.21)$$

Очевидно, что выражение (2.21) может быть представлено в виде:

$$\hat{H}_{kl} = \sum_m \left[\cos(2\pi km \; / \; N) + \sin(2\pi km \; / \; N) \right] * \left\{ \sum_n x_{mn} \left[\cos(2\pi \ln / \; N) + \sin(2\pi \ln / \; N) \right] \right\}$$

Представляет интерес установить взаимосвязь между традиционным двумерным преобразованием Хартли (выражение (2.20)) и модифицированным преобразованием Хартли, определяемым согласно выражению (2.21). После ряда тригонометрических и алгебраических преобразований ядра в выражении (2.21) нетрудно получить [24]:

$$H_{kl} = \left[\hat{H}_{kl} + \hat{H}_{k,-l} + \hat{H}_{-k,l} - \hat{H}_{-k,-l} \right] / \; 2 \qquad (2.22)$$

где в силу цикличности ядра *(N-k)modN = -k; (N-l)modN = -l.*

От компонент спектра Хартли H_{kl} можно перейти к компонентам спектра Фурье, используя известное соотношение:

$$F_{kl} = \left\{ \left[H_{kl} + H_{-k,-l} \right] - j \left[H_{kl} - H_{-k,-l} \right] \right\} / \; 2$$

Если записать $H_{-k,-l}$ в виде, подобном выражению (2.22), то получим выражение, определяющее связь между компонентами спектра Фурье и компонентами модифицированного преобразования Хартли:

$$F_{kl} = \left\{ \left[\hat{H}_{k,-l} + \hat{H}_{-k,l} \right] - j \left[\hat{H}_{k,l} - \hat{H}_{-k,-l} \right] \right\} / \; 2 \qquad (2.23)$$

Следует подчеркнуть, что определение F_{kl} как через традиционное, так и через модифицированное преобразование Хартли требует вычисления только лишь H_{kl} или $H^{\wedge}_{kl}$ соответственно. Остальные требуемые слагаемые легко могут быть найдены путем перекомпоновки матрицы H_{kl} или $H^{\wedge}_{kl}$

Выражения (2.22) и (2.23) справедливы для любых действительных функций x_{mn}. В частных случаях, когда x_{mn} обладает симметрией относительно одной или двух координат, эти выражения существенно упрощаются.

Пусть $x_{mn} = x_{m,-n}$ Тогда $H^{\wedge}_{kl} = H^{\wedge}_{k,-l}$, $H^{\wedge}_{-k,-l} = H^{\wedge}_{-k,l}$, а формулы (2.22) и (2.23) приобретают вид:

$$H_{kl} = H^{\wedge}_{kl}$$

$$F_{kl} = \left\{\left[H^{\wedge}_{kl} + H^{\wedge}_{-k,l}\right] - j\left[H^{\wedge}_{k,l} - H^{\wedge}_{-k,l}\right]\right\} / 2$$

Если $x_{mn} = x_{-m,n} = x_{m,-n}$, то как преобразование Хартли, так и преобразование Фурье эквивалентны модифицированному преобразованию Хартли:

$$H_{kl} = H^{\wedge}_{kl}; F_{kl} = H^{\wedge}_{kl}$$

2.7. Ортогональные преобразования в диадных базисах

Ортогональные преобразования в диадных (или иначе говоря, двузначных знакопеременных) базисах определены для данных, представленных векторами длиной N= 2^M. К таким преобразованиям относятся преобразования Адамара, Пэли, Уолша, Трахтмана, Качмарджа и ряд других [19, 20, 23]. Матрица ядра любого из подобных преобразований содержит целочисленные коэффициенты из множества {-1; +1}. Очевидно, что при выполнении подобных преобразований существенно сокращается объем вычислений за счет исключения умножения в каждой базовой операции.

Матрица ядра преобразования Уолша - Адамара для N =2^m может быть описана как результат кронекеровского произведения m матриц ДЭФ E_2 размера 2x2 [8,12] :

$$A_N = A_{2^m} = E_2 \otimes E_2 \otimes E_2 \otimes \ldots \otimes E_2 = \left[E_2\right]^m = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}^m, \qquad (2.24)$$

где символ ⊗ - операция кронекеровского умножения векторов, в результате чего порождается матрица блочной структуры. Заметим, что операция кронекеровского умножения двух матриц и состоит в получении блочной матрицы, блоками которой является умноженная на соответствующий элемент правой матрицы левая матрица, т.е.:

$$A_4 = E_2 \otimes E_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \left|\begin{array}{cc:cc} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ \hdashline 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{array}\right|$$

аналогично можно получить и для N = 8:

$$A_8 = E_2 \otimes E_2 \otimes E_2 = A_4 \otimes E_2 = \left|\begin{array}{cc:cc:cc:cc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ \hdashline 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hdashline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ \hdashline 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right|$$

Матрица ядра Адамара обладает следующими свойствами:

1) цикличностью $a_{N+k} = a_k$; $a_{N-k} = a_{-k}$

2) мультипликативностью $a_{k+l} = a_k * a_l$

3) симметричностью $A_N = A_N^T$

В задачах ЦОС используются и другие, подобные Адамару, преобразования - Пэли, Уолша, Трахтмана и других. Ядра (матрицы) этих преобразований могут быть получены на основе матрицы ядра преобразования Адамара при определенном переупорядочении их строк. Поэтому перед рассмотрением указанных преобразований рассмотрим правила переупорядочения матриц, применяемые для формирования ядер таких преобразований.

Переупорядочим элементы вектора $X = \left[x_0, x_1, x_2, \ldots, x_{N-1}\right]$ таким образом, чтобы первые элементы вектора были бы чётными, а вторые - нечётными.

Повторим эту процедуру до тех пор, умножая каждый раз длину соответствующего вектора вдвое, пока длина подвектора не станет равной 2. В результате получится переупорядоченный вектор, элементы которого упорядочены по закону так называемой двоичной инверсии.

В частности, для *N=8* такое переупорядочение будет выполняться следующим образом:

$$X = \begin{vmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{vmatrix} \Rightarrow \begin{vmatrix} x_0 \\ x_2 \\ x_4 \\ x_6 \\ x_1 \\ x_3 \\ x_5 \\ x_7 \end{vmatrix} \Rightarrow \begin{vmatrix} x_0 \\ x_4 \\ x_2 \\ x_6 \\ x_1 \\ x_5 \\ x_3 \\ x_7 \end{vmatrix} = \hat{X}$$

Вектор $\hat{X}$ можно получить из *X*, если задать следующее правило формирования индекса текущего элемента вектора $\hat{X}$ из индекса элемента x_i исходного вектора *X*:

1) записывается двоичный код текущего индекса i;

2) двоичный код записывается в обратном порядке как рефлексивный или отраженный код (т.е. начиная с младших разрядов);

3) полученный код определяет индекс текущего элемента вектора $\hat{X}$;

4) переводим полученный индекс в десятичную систему.

Для рассмотренного выше случая (*N=8)* получим:

$$\begin{vmatrix} 000 \\ 001 \\ 010 \\ 011 \\ 100 \\ 101 \\ 110 \\ 111 \end{vmatrix} \Rightarrow \begin{vmatrix} 000 \\ 100 \\ 010 \\ 110 \\ 001 \\ 101 \\ 011 \\ 111 \end{vmatrix} \Rightarrow \begin{vmatrix} x_0 \\ x_4 \\ x_2 \\ x_6 \\ x_1 \\ x_5 \\ x_3 \\ x_7 \end{vmatrix}$$

Процедура перестановки в терминах векторно-матричных операций может быть записана как:

$$\hat{X} = S_N X , \qquad (2.25)$$

где S_N - перестановочная матрица. Для случая двоично-инверсной перестановки перестановочная матрица будет иметь вид:

$$S_8 = \begin{vmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{vmatrix}$$

Элементы вектора могут быть также переупорядочены по коду Грея:

$$\begin{vmatrix} 000 \\ 001 \\ 010 \\ 011 \\ 100 \\ 101 \\ 110 \\ 111 \end{vmatrix} \Rightarrow \begin{vmatrix} 000 \\ 001 \\ 011 \\ 010 \\ 110 \\ 111 \\ 101 \\ 100 \end{vmatrix}$$

В терминах векторно-матричных операций подобная перестановка элементов вектора может быть описана так:

$$\hat{X} = \Gamma_N X ,$$

где Γ_N - перестановочная матрица, в частности, для *N=8:*

$$\Gamma_8 = \begin{vmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{vmatrix}$$

Ядро преобразования Пэли можно получить из ядра преобразования Адамара, переупорядочив строки матрицы A_N по закону двоичной инверсии, в частности, для $N = 8$ получаем:

$$P_8 = \left|\begin{array}{cccccccc|c} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 000 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 001 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 010 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & 011 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 100 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 101 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 110 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & 111 \end{array}\right|$$

В общем случае правило перехода от матрицы Адамара к матрице Пэли может быть представлено как [8]:

$$A_N \xrightarrow{Дв.инверсия} P_N \qquad (2.26)$$

Если теперь переупорядочить строки матрицы ядра преобразования Пэли по коду Грея, то получим матрицу ядра преобразования Уолша[8]

$$W_8 = \left|\begin{array}{cccccccc|c} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 000 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & 001 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & 011 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 010 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 110 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & 111 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 & 101 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 100 \end{array}\right|$$

Или в общем виде:

$$A_N \xrightarrow{Дв.инв.} P_N \xrightarrow{Грей} W_N \qquad (2.27)$$

$$P_N \xrightarrow{Грей} W_N \qquad (2.28)$$

Очевидно, что свойства матриц P_N и W_N аналогичны свойствам матрицы A_N. Матрицы ядер преобразования Трахтмана и Качмарджа также могут быть получены на основе матрицы ядра Адамара, но при использовании обратной перестановки по коду Грея.

Заметим, что матрицы преобразований в указанных базисах отличаются порядком строк. Очевидно, что при обработке одного и того же вектора исходных данных X в различных базисах вектор результата будет содержать одинаковые по своей величине элементы, отличаясь для каждого преобразования лишь порядком их следования. Поэтому для всех подобных преобразований

можно использовать одну и ту же матрицу ядра (например, A_N), а для получения вектора F с нужным для каждого преобразования порядком следования элементов, переупорядочить элементы исходного вектора X. Так, для преобразования Пэли необходима двоично-инверсная перестановка, а для преобразования Уолша - двоично-инверсная перестановка и затем перестановка по коду Грея.

Если проанализировать все рассмотренные во второй главе преобразования, то можно придти к выводу, что их сущность состоит в разложении исходной функции на ряд чётных и нечётных составляющих, которые задаются строками матриц E_N, H_N, A_N, W_N и P_N. Так, для матриц E_N и H_N это набор гармонических функций *cos[…]* и *sin[..],* а для A_N, W_N и P_N - наборы прямоугольных знакопеременных функций.

Тем самым, в зависимости от вида исходной функции, в её составе будут определены отдельные компоненты (и их частотное значение), которые задаются строками матрицы ядра преобразования. Очевидно, что разложение по ортогональным функциям, задаваемым строками матрицы E_N ,позволяет определить частотный состав исходной функции (сигнала), что имеет вполне понятный физический смысл.

Точно также при разложении по функциям, задаваемым строками матриц A_N, W_N и P_N, можно определить, какой вклад вносит та или иная знакопеременная функция в состав исходного сигнала.

Очевидно, что исходный сигнал может быть разложен по различным ортогональным составляющим. При этом вклад таких составляющих в исходный сигнал будет различен для разных матриц ортогональных преобразований.

Согласно теореме Коруэна-Лоева [25], может быть определено оптимальное разложение исходной функции по набору ортогональных функций в смысле наименьшего числа отличных от нуля компонент такого разложения.

На практике для сигналов гармонической природы удобно использовать различные гармонические функции, т.е. для определения частотного состава сигнала выполнить преобразование Фурье или Хартли (ДПФ или БПФ). Действительно, для моногармонического сигнала лишь одна компонента

разложения по строкам матрицы E_N будет отлична от “0” (при условии, что f_0 кратно $k\pi$).

Для сигналов, описываемых знакопеременными функциями, близким к оптимальному является разложение по знакопеременным функциям типа Уолша, Адамара. Поэтому в задачах кодирования и распознавания речи, где для представления сигналов широко используется метод широтно-импульсной модуляции, удобно выполнять разложение по ортогональным функциям, задаваемым строками матриц A_N, W_N или P_N.

2.8. Понятие о Wavelet-преобразованиях. Преобразование Хаара

В ряде случаев оказывается более удобным в качестве базисов разложения использовать такие системы функций, для которых коэффициенты разложения учитывают поведение исходной функции лишь в нескольких близкорасположенных точках.

Использование такого базиса по своей сути означает переход от частотного анализа к масштабному, т.е. функция $f(x)$ анализируется с помощью некоторой “стандартной” математической функции, изменяемой по масштабу и сдвигу на некоторую величину.

Первое упоминание об этих функциях появилось в работах Хаара в 1909 году. В 30-е годы начались более детальные исследования возможностей представления сигналов с использованием базисных масштабируемых функций. Пол Леви, используя масштабируемую базисную функцию типа функции Хаара, исследовал разновидность случайного сигнала - броуновское движение. Он обнаружил преимущество в применении базисных функций Хаара перед функциями Фурье.

В период 60-х - 80-х годов Вейс и Кофман исследовали простейшие элементы функционального пространства, названные ими атомами, с целью обнаружить атомы для произвольной функции и найти "правила сборки", позволяющие реконструировать все элементы функционального пространства, используя эти атомы. В 1980 году Гроссман и Марлет определили такие функции как Wavelet-функции. В переводе с английского Wavelet – всплеск,

поэтому в отечественной литературе встречается термин «разложение по всплескам» наряду с вейвлет-анализом.

В конце 80-х г.г. Мейер и Добичи на основе исследований Марлета создали ортогональные базисы Wavelet-функций, которые и стали основой современных Wavelet-функций. Сходство Wavelet и Фурье преобразований состоит в следующем:

1. Оба они являются линейными преобразованиями и предназначены для обработки блоков данных, содержащих $log_2 N$ элементов.

2.Обратные матрицы ДПФ и DWT (discret wavelet transform) равны транспонированной, причем строки самих матриц содержат функции cos(x) и sin(x), а для DWT - более сложные базисные функции - wavelet.

Наиболее важное различие между этими двумя видами преобразований состоит в том, что отдельные функции wavelet локализованы в пространстве, а синусные и косинусные - нет. Благодаря этой особенности, DWT находит большое число применений, в том числе для сжатия данных, распознавания образов и подавления шумовой составляющей принимаемого сигнала.

Преобразование Wavelet состоит из неограниченного набора функций. Семейство Wavelet различают по тому, насколько компактны базисные функции в пространстве и насколько они гладки. Некоторые их них имеют фрактальную структуру. В каждом семействе они могут быть разбиты на подклассы по числу коэффициентов и уровню итераций. Чаще всего внутри семейства функция классифицируется по номеру моментов исчезновения.

Набор дискретных Wavelet-функций в общем виде может быть описан как

$$W_{[s,l]}(x) = 2^{-s/2} W(2^{-s}, x-l), \quad (2.29)$$

где s и l - целые числа, которые масштабируют и сдвигают материнскую функцию $W(x)$ для создания wavelet.

Индекс масштаба s показывает ширину wavelet, а индекс смещения l определяет ее позицию. Материнские функции масштабированы или растянуты коэффициентом, кратным степени 2 и приведены к целому. Таким образом, если известна материнская функция, то может быть построен и весь базис.

В свою очередь, само Wawelet-преобразование может быть записано в виде

[29]:

$$V(x) = \sum_{k=1}^{N-2} (-1)^k z_{k+l} W(2x+l) \quad (2.30)$$

Z_{k+l} – отсчеты исходного сигнала.

Метод разложения по всплескам широко используется для выделения шумовой компоненты при обработке данных.

К wavelet-подобным функциям относятся функции Хаара. Для $N=4$ и $N=8$ матрицы ядра преобразования имеют вид:

$$X_4 = \begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ -1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{vmatrix}$$

$$X_8 = \begin{vmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{vmatrix}$$

Основные свойства матрицы ядра преобразования Хаара состоят в следующем:

а) ее элементы не мультипликативны

б) матрица не симметрична, откуда следует, что для обратного преобразования матрицу ядра необходимо транспонировать, т.е.

$$X_N^{-1} = X_N^T$$

в) строки матрицы определяют переодические функции с периодом N.

Однако матрица ядра преобразования Хаара является не ортонормированной, т.е. для данного преобразования не выполняется теорема Парсеваля. Поэтому для выполнения теоремы Парсеваля требуется ввести дополнительную нормировку, такая нормировка заключается в умножении на $\sqrt{2}$

тех элементов вектора результатов, которые соответствуют строкам матрицы с нулевыми компонентами. Для строк, содержащих N/2 ненулевых элемента такое умножение выполняется один раз, для строк с N/4 ненулевыми элементами - два раза и так далее. Подобную нормировку необходимо выполнять как при выполнении прямого, так и обратного преобразования Хаара.

Близким к преобразованию Хаара является усеченное преобразование Адамара, отличающееся, по своей сути, лишь порядком следования строк матрицы ядра преобразования [13].

Итак, любое линейное преобразование может быть записано как векторно-матричная процедура:

$$F=kB_NX$$

где $X=[x_0,x_1,...,x_{N-1}]^T$ - вектор исходных данных;

B_N - квадратная матрица размера $N\,x\,N$ ядра преобразования;

$F=[f_0,f_1,\ldots,f_{N-1}]^T$ - вектор результатов.

Если для матрицы B_N существует обратная матрица B_N^{-1} (что справедливо для всех ортогональных преобразований), то существует и обратное преобразование:

$$X=kB_N^{-1}F.$$

Методы дискретных ортогональных преобразований в настоящее время широко используются при обработке сигналов. В таблице 2.2 приведены основные области применения спектральных методов и задачи, решаемые с их помощью [12].

Таблица 2.2

Задачи ЦОС, решаемые спектральными методами

Решаемые задачи	Область применения	Выполняемые преобразования
Формирование характеристик направленности антенных устройств	Радиофизика, радиолокация, гидроакустика	Двумерное ДПФ матрицы данных
Выделение сигнала на фоне шумов	Радиофизика, радиолокация, обработка изображений	Прямое ДПФ + сглаживание спектра + обратное ДПФ
Обнаружение и определение координат объекта на изображении	Радиофизика, гидроакустика, обработка изображений	Прямое ДПФ + согласованная фильтрация спектра + обратное ДПФ
Определение скорости движущегося объекта	Радиолокация, гидроакустика	Одномерное ДПФ временной последовательности сигнала
Сжатие и восстановление изображений	Системы передачи, обработки и архивации изображений	Двумерное косинусное прямое или обратное преобразование по окну
Восстановление изображения по проекциям	Томография	Преобразование Радона для векторов данных через ДПФ

3. БЫСТРЫЕ АЛГОРИТМЫ ОРТОГОНАЛЬНЫХ ПРЕОБРАЗОВАНИЙ

3.1. Вычислительная сложность ДПФ и способы её сокращения

Во второй главе мы пришли к выводу, что любое ортогональное преобразование в матричной форме может быть описано как процедура умножения вектора исходных данных на матрицу ядра (при прямом преобразовании)

$$F = kB_N X$$

или вектора результатов разложения исходного сигнала по тем или иным базисным ортогональным функциям на обратную матрицу (при обратном преобразовании):

$$X = k\,B_N^1 F$$

В общем случае вычислительная сложность такой процедуры составляет: $Q = N^2(БО)$, где отдельная базовая операция включает операцию умножения и сложения действительных чисел или те же операции умножения и сложения, но для комплексных чисел, в случае преобразования Фурье. С учётом того, что для комплексных чисел

$$A_1 * A_1 = (a + jb)(c + jd) = (ac - bd) + j(bc + bd)$$

сложность вычислительной процедуры ДПФ

$$Q_{ДПФ} = 4N^2 Б.О., \qquad (3.1)$$

если под Б.О. понимать те же операции, как и в базисах действительных функций.

В целях сокращения объёма вычислений можно выполнить формирование отсчётов вектора F с учётом вырождения (т.е. тривиальности) первой строки и первого столбца матрицы ДПФ:

$$\begin{cases} f_0 = \sum_{k=0}^{N-1} x_k \\ f_n = x_0 + \sum_{k=1}^{N-1} x_k * e^{2\pi kn/N}; n = \overline{1, N-1} \end{cases} \qquad (3.2)$$

Отметим особенности представления чётных и нечётных функций при ДПФ и ДПХ. Пусть исходный сигнал описан как суперпозиция чётной и нечётной

составляющей:

$$X = X_{sim} + X_{asim}$$

причем для каждой из составляющих можно записать

$$\begin{cases} X_{sim_n} = X_{sim_{-n}} = X_{sim_{N-n}} \\ X_{asim_N} = -X_{asim_{-N}} = -X_{asim_{N-n}} \end{cases} \quad (3.3)$$

С учётом такого представления сигнала можно записать, что

$$\begin{bmatrix} X_{sim_n} * \cos\frac{2\pi}{N} kn = X_{sim_{N-n}} * \cos\frac{2\pi}{N}(N-n)k \\ X_{asim_n} * \cos\frac{2\pi}{N} kn = -X_{asim_{N-n}} * \cos\frac{2\pi}{N}(N-n)k \end{bmatrix} \quad (3.4а)$$

$$\begin{bmatrix} X_{sim_n} * \sin\frac{2\pi}{N} k = -X_{sim_{N-n}} * \sin\frac{2\pi}{N}(N-n)k \\ X_{asim_n} * \sin\frac{2\pi}{N} k = X_{asim_{N-n}} * \sin\frac{2\pi}{N}(N-n)k \end{bmatrix} \quad (3.4б)$$

поэтому для чётного исходного сигнала “синусная” компонента спектра будет равна 0, а для нечётного сигнала - “косинусная” компонента спектра равна 0.

Поэтому спектр действительного сигнала в базисах Фурье и Хартли описывается чётной функцией и для его задания требуется только косинусная компонента, причём в силу чётности для задания спектра достаточно только $N/2$ отсчётов $\left\{ f_0 \div f_{N/2-1} \right\}$.

Кроме того, матрицы E_N и C_N обладают свойством симметрии и периодичности (цикличности) следования элементов. Из симметрии матриц указанных ядер следует, что в общем случае для действительных последовательностей (т.е. вектора X, содержащего только действительные элементы) может быть выполнено только для $N/2$ компонент спектра, (например, с номерами $\left\{ 0, N/2 - 1 \right\}$), поскольку компоненты “положительной” и “отрицательной” областей частот (компоненты с номерами $\left\{ 0, N/2 - 1 \right\}$ и $\left\{ N/2, N-1 \right\}$ имеют одинаковую амплитуду, но противоположные фазы. Поэтому

вторую половину компонент можно легко достроить из вычисленных.

Аналогично можно поступить и при вычислении ДПХ - cos-составляющая “положительных” и “отрицательных” спектральных компонент одинакова, а sin-составляющие для “положительных” и “отрицательных” имеют противоположные знаки.

3.2. Запись алгоритма БПФ в векторно-матричной форме

Однако матрицы указанных ортогональных преобразований позволяют значительно сократить объём вычислений, поскольку из N^2 элементов матрицы имеется лишь N (для ДПФ и ДПХ) различных значений.

Действительно, рассмотрим ДПФ для $N = 4$

$$F = \begin{vmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{vmatrix} = \begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{vmatrix} * \begin{vmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{vmatrix} = \begin{vmatrix} x_0 + x_1 + x_2 + x_3 \\ x_0 - jx_1 - x_2 + jx_3 \\ x_0 - x_1 + x_2 - x_3 \\ x_0 + jx_1 - x_2 - jx_3 \end{vmatrix} = \begin{vmatrix} (x_0 + x_2) + (x_1 + x_3) \\ (x_0 - x_2) - j(x_1 - x_3) \\ (x_0 + x_2) - (x_1 + x_3) \\ (x_0 - x_2) + j(x_1 - x_3) \end{vmatrix} = \begin{vmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{vmatrix}$$

После выделения встречающихся неоднократно блоков данных можно заметить, что $(x_0 + x_2)$ и $(x_0 - x_2)$ есть не что иное, как ДПФ для $N = 2$ и подвектора (x_0, x_2). Аналогичные выводы можно сделать и для других блоков данных, что позволяет записать :

$$|E_2| \begin{vmatrix} x_0 \\ x_2 \end{vmatrix} = \begin{vmatrix} x_0 + x_2 \\ x_0 - x_2 \end{vmatrix}; \; |E_2| \begin{vmatrix} x_1 \\ x_3 \end{vmatrix} = \begin{vmatrix} x_1 + x_3 \\ x_1 - x_3 \end{vmatrix}$$

Попробуем использовать как типовой элемент преобразования процедуру вида $E_2 * X_2$ и перекомпоновку векторов.

1) Переставим элементы исходного вектора, чтобы сформировать указанные подвектора. Очевидно, что для такой перестановки необходимо двоично-инверсное переупорядочивание вектора X.

Такая процедура может быть записана как операция умножения вектора X на перестановочную матрицу S_4^1.

$$\hat{X}^1 = S_4^1 X \rightarrow \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} * \begin{vmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{vmatrix} = \begin{vmatrix} x_0 \\ x_2 \\ x_1 \\ x_3 \end{vmatrix}$$

2) Запишем матрицу

$$\hat{E}_4 = (E_2 \otimes I_2^1) = \left[\begin{vmatrix}1 & 1\\ 1 & -1\end{vmatrix} \otimes \begin{vmatrix}1 & 0\\ 0 & 1\end{vmatrix}\right] = \begin{vmatrix}\begin{vmatrix}1 & 1\\ 1 & -1\end{vmatrix} & \\ & \begin{vmatrix}1 & 1\\ 1 & -1\end{vmatrix}\end{vmatrix}$$

и получим вектор $\hat{F}_1$ как:

$$\hat{F}^1 = \hat{E}_4 * \hat{X}^1 = \hat{E}_4 * \left[S_4^1 X\right] = \begin{vmatrix}\begin{vmatrix}1 & 1\\ 1 & -1\end{vmatrix} & \\ & \begin{vmatrix}1 & 1\\ 1 & -1\end{vmatrix}\end{vmatrix} = \begin{vmatrix}x_0 + x_2\\ x_0 - x_2\\ x_1 + x_3\\ x_1 - x_3\end{vmatrix}$$

$$\hat{F}^1 = \left[E_2 \otimes I_2\right] S_4^1 * X$$

3) Умножим вектор $\hat{F}_1$ на диагональную матрицу D_4^1:

$$\hat{F}^2 = D_4^1 * \hat{F}^1 = \begin{vmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & -j\end{vmatrix} * \begin{vmatrix}x_0 + x_2\\ x_0 - x_2\\ x_1 + x_3\\ x_1 - x_3\end{vmatrix} = \begin{vmatrix}x_0 + x_2\\ x_0 - x_2\\ x_1 + x_3\\ -j(x_1 - x_3)\end{vmatrix}$$

$$\hat{F}^2 = D_4^1\left[E_2 \otimes I_2\right] * (S_4^1 X)$$

4) Получим вектор $\hat{X}^2$, вновь переставив элементы вектора $\hat{F}^2$ по правилу двоичной инверсии с помощью матрицы S_4^1:

$$\hat{X}^2 = S_4^1 * \hat{F}^2 = \begin{vmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\end{vmatrix} * \begin{vmatrix}x_0 + x_2\\ x_0 - x_2\\ x_1 + x_3\\ -j(x_1 - x_3)\end{vmatrix} = \begin{vmatrix}x_0 + x_2\\ x_1 + x_3\\ x_0 - x_2\\ -j(x_1 - x_3)\end{vmatrix}$$

$$\hat{X}^2 = S_4^1\left[D_4^1 * (E_2 \otimes I_2)\right] * (S_4^1 X)$$

5) Умножим вектор $\hat{X}^2$ на матрицу $\hat{E}_4$:

$$\hat{F}^3 = \hat{E}_4 * \hat{X}^2 = \begin{vmatrix}\begin{vmatrix}1 & 1\\ 1 & -1\end{vmatrix} & \\ & \begin{vmatrix}1 & 1\\ 1 & -1\end{vmatrix}\end{vmatrix} * \begin{vmatrix}x_0 + x_2\\ x_1 + x_3\\ x_0 - x_2\\ -j(x_1 - x_3)\end{vmatrix} = \begin{vmatrix}(x_0 + x_2) + (x_1 + x_3)\\ (x_0 + x_2) - (x_1 + x_3)\\ (x_0 - x_2) - j(x_1 - x_3)\\ (x_0 - x_2) + j(x_1 - x_3)\end{vmatrix} = \begin{vmatrix}f_0\\ f_2\\ f_1\\ f_3\end{vmatrix}$$

$$\hat{F}^3 = \left[E_2 \otimes I_2\right] * S_4^1\left[D_4^1 * (E_2 \otimes I_2)\right] * (S_4^1 X)$$

6) Переставим элементы вектора $\hat{F}^3$ с помощью матрицы S_4^1:

$$F = S_4^1 * \hat{F}^3 = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{vmatrix} = \begin{vmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{vmatrix}$$

иначе говоря:

$$F = S_4^1 \left\{ \left[(E_2 \otimes I_2) D_4^1 \right] * S_4^1 \left[(E_2 \otimes I_2) D_4^0 \right] * S_4^1 X \right\} \qquad (3.5)$$

где на первой итерации диагональная матрица

$$D_4^0 = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix}$$

введена формально, для симметрии матричной записи обоих итераций.

Рассуждая аналогичным образом, можно придти к алгоритму, состоящему в выполнении последовательности итераций. На рис.3.1 приведен граф алгоритма БПФ, иллюстрирующий выполнение итеративнно - связанных вычислений по описанной схеме.

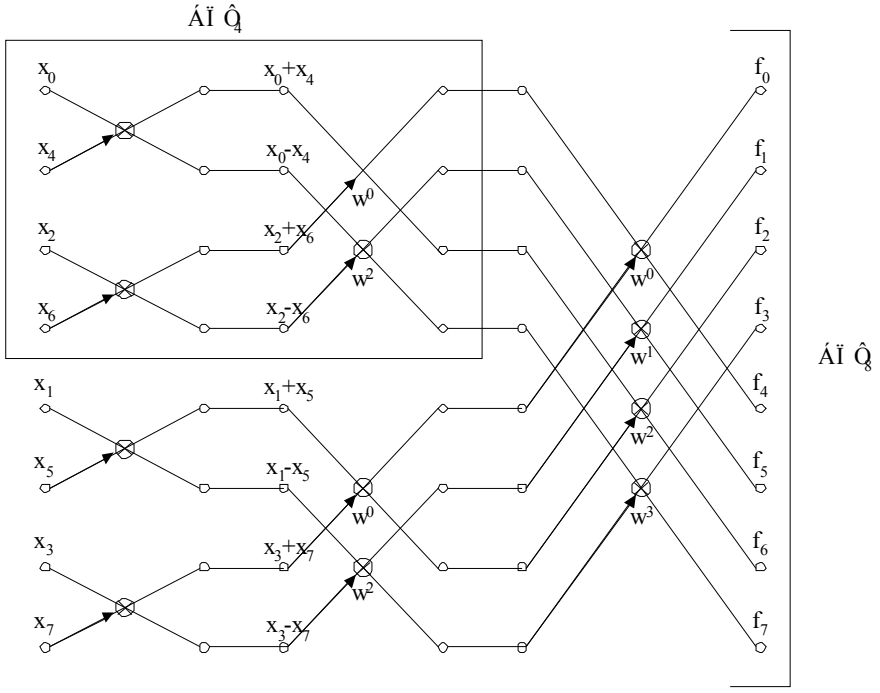


Рис.3.1. Граф процедуры БПФ для *N = 8.*

На рисунке обозначены:

$$W^0 \equiv 1 \equiv e^{-j\frac{2\pi \cdot 0}{8}};\ W^1 = e^{-j\frac{\pi}{4}} = \frac{\sqrt{2}}{2}(1-j);\ W^2 = e^{-j\frac{\pi}{2}} = -j;\ W^3 = e^{-j\frac{3\pi}{4}} = -\frac{\sqrt{2}}{2}(1+j)$$

Число итераций составляет $m = \log_2 N$ ($N = 2^m$). Перед первой итерацией выполняется r-ично инверсная перестановка элементов вектора исходных данных. В свою очередь каждая итерация сводится к схожим действиям:

1. Перестановка элементов вектора результатов предшествующей итерации;
2. Умножение вектора данных на диагональную матрицу D_N^m (m - номер итерации), причём $D_N^1 \equiv I_N$
3. Умножение вектора результатов (по п.1) на матрицу блочной структуры

$$\hat{E_1}(\hat{E_N} = E_2 \otimes I_2 \otimes I_2 \otimes I_2 \otimes I_2 \ldots)$$

4. Выполнение перестановок элементов вектора, содержащего результаты п.2.

В матричном виде это можно записать [8, 12]:

$$F^{(m)} = S_N^{(m)}(B_N^m \widetilde{S}_N^m F^{m-1}) \qquad (3.6),$$

где $F^{(0)} = \left[x_0, x_1, x_2 \ldots x_{N-1}\right]$ - вектор исходных данных,

$F^{(m)} = \left[f_0, f_1, f_2 \ldots f_{N-1}\right]$ - вектор результатов БПФ,

$\widetilde{S}_N^m$ - матрица перестановок вектора выходных данных (т.е. $F^{(m)}$),

S_N^m - матрица перестановок вектора входных данных (т.е. $F^{(m-1)}$).

В свою очередь, матрица $B_N^{(m)}$ является блочной матрицей, процедура формирования которой определяется выражением:

$$B_N^{(m)} = I_2^{M-m} \otimes \left[\oplus_{K=0}^{2^{m-1}} (E_2 D_2^{K2^{m-1}})\right] \qquad (3.7),$$

здесь I_2^{M-m} - единичная матрица порядка 2^{M-m}; $(m = \overline{1, M})$,

$\otimes$ - знак прямого произведения матриц (кронекеровского произведения),

$\oplus$ - знак прямой суммы матриц. Заметим, что прямой суммой матриц A_N и B_N является диагональная матрица C_{2N} на главной диагонали которой расположены блоки из исходных матриц:

$$C_{2N} = A_N \oplus B_N = \begin{vmatrix} A_N & \\ & B_N \end{vmatrix} \qquad (3.8).$$

$D_2^P = diag\left\{W_N^{(0)}, W_N^{(1P)}\right\}$ - диагональная матрица поворачивающих множителей ($P = K \cdot 2^{m-1}$),

$$E_2 = \begin{vmatrix} 1 & 1 \\ 1 & -1 \end{vmatrix}$$ - матрица ДЭФ порядка,

$$W_N^{iP} = \exp(-j\frac{2\pi iP}{N}); P = K \cdot 2^{m-1}; i = \overline{0,1}$$

Такое представление алгоритма БПФ в виде матричных операций носит в большей мере характер формального описания. Действительно, выполнение перестановок как операций умножения вектора на матрицу явно не является оптимальным путем их реализации, также как и описываемые через аналогичные операции умножения на поворачивающие множители.

Серьезным недостатком такого представления алгоритма БПФ является трудность перехода к программной реализации алгоритма.

Для практических целей более удобно описание алгоритма БПФ через систему рекуррентных выражений.

3.3. Представление алгоритма БПФ в виде рекурсивных соотношений

Вернемся к рассмотрению процедуры БПФ. Мы говорили, что на каждой ступени вычисляются группы однородных операций:

- умножение на матрицу D поворачивающих множителей;
- умножение на матрицу $\hat{E}_2$;
- перестановка элементов вектора результатов.

Основная сложность в записи алгоритма БПФ состоит именно в описании перестановок элементов вектора.

Если рассмотреть граф БПФ (например, для N=8, рис3.1), то можно увидеть, что на первой итерации как бы независимо отрабатывается 4 фрагмента исходного вектора, содержащие по 2 элемента: такие фрагменты назовем подвекторами. В данном случае это подвектор $X=\{x_0, x_4\}$; $X=\{x_1, x_5\}$; $X=\{x_2, x_6\}$; $X=\{x_3, x_7\}$. На второй итерации отрабатывается 2 подвектора по 4 элемента, а на третий - один вектор из восьми элементов.

Таким образом, суть алгоритма БПФ состоит в разбиении на подвектора по 2 элемента, их независимой обработке, и формированию из промежуточных подвекторов вдвое большей длины (очевидно, что число таких подвекторов уменьшается от итерации к итерации), что выполняется до тех пор, пока не

образуется один вектор длиной N элементов.

Подобная процедура может быть описана как [24]:

$$F^m = \begin{vmatrix} f_{r,m} \\ f_{k+1,m} \end{vmatrix} = \begin{vmatrix} 1 & 1 \\ 1 & -1 \end{vmatrix} \begin{vmatrix} W^0_{N_m} & 0 \\ 0 & W^k_{N_m} \end{vmatrix} \begin{vmatrix} f_{k,m-1} \\ f_{k+1,m-1} \end{vmatrix} \qquad (3.9)$$

где $l=2^{m-1}$, $m \in \overline{1, M}$, $k=(n)_{\mathrm{mod}\ l}$, $k \in \overline{0,2^{m-1}-1}$, $n \in (0, N-1)$ - текущий индекс элемента вектора, k - соответствует номеру Б0 при обработке подвекто-ра, $W^k_{N_m} = \exp(-j\frac{2\pi k}{N_m})$, $N = 2^m$ - длина подвектора, причём для m=0 вектор $F^{(0)} \equiv X$. Номер обрабатываемого на данной итерации блока данных определяется целой частью $(n)_{\mathrm{mod}\ l}$

Выражение (3.9) определяет обработку отдельного, но любого подвектора на каждой итерации из общего их числа $N / N_m = 2^{M-m}$. Индекс n определяется следующим образом:

$$n_i = N_{m-i} \quad (i = \overline{0,2^{M-m}-1}),$$

где n_i - начальный индекс для i-го подвектора.

В свою очередь, (3.9) может быть переписано в виде системы из двух уравнений:

$$\begin{cases} f_{k,m} = f_{k,m-1} + f_{k+l,m-1} \cdot W^K_{N_m} \\ f_{k+l,m} = f_{k,m-1} - f_{k+l,m-1} \cdot W^K_{N_m} \end{cases} \qquad (3.10)$$

Это выражение и соответствует базовой операции, для которой указаны правила формирования текущих индексов по “входным” и “выходным” бабочкам, а также правила вычисления поворачивающего множителя $W^k_{N_m}$

3.4. Алгоритмы БПФ с прореживанием по времени и по частоте

Быстрые алгоритмы БПФ являются “обратимыми”, т.е. вычисления по графу БПФ (см., например, рис. 3.1) могут выполняться как “слева направо”, так и “справа налево”.

В первом случае алгоритм БПФ основан на “бабочке” вида:

$$\begin{cases} P = A + W^k B \\ Q = A - W^k B \end{cases} \qquad (3.11)$$

и называется алгоритмом с прореживанием по времени (алгоритм Кули- Тьюки) [22].

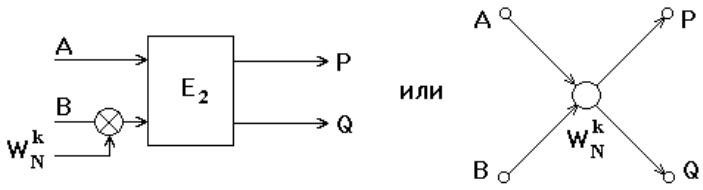
Во втором случае (при вычислении по графу “справа налево”) алгоритм БПФ производится на основе бабочки вида:

$$\begin{cases} P = A + B \\ Q = (A - B) \cdot W^k \end{cases} \qquad (3.12)$$

и называется алгоритмом БПФ с прореживанием по частоте (алгоритмом Сэнди-Тьюки). Такие алгоритмы были впервые опубликованы в середине 60-х годов [22].

Графы подобных базовых операций приведены на рис. 3.2, а) и б) соответственно.

а)



б)

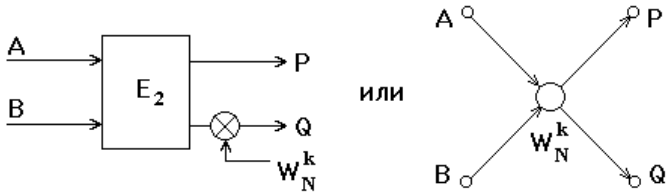


Рис. 3.2. Графы базовых операций БПФ с прореживанием по времени (а) и по частоте (б)

Заметим, что для алгоритма БПФ с прореживанием по частоте не выполняется двоично-инверсная перестановка элементов вектора X перед первой итерацией, но зато необходимо переставлять элементы векторов результата по закону двоичной инверсии.

3.5. Алгоритм БПФ по основанию r ($N = r^m$, $r \geq 3$)

До сих пор мы рассматривали БПФ только для случая, когда $N = 2^m$. На практике, однако, достаточно часто возникает необходимость вычисления ДПФ

при $N = r^m$, где r отлично от 2 (например, *N=125=5³, N= 625=5⁴; N=27* и т.д.).

Для таких случаев несколько позднее Сэнди и Радемахором были разработаны алгоритмы БПФ, основу которых составляют базовые операции “бабочка” следующего вида для алгоритмов с прореживанием по времени:

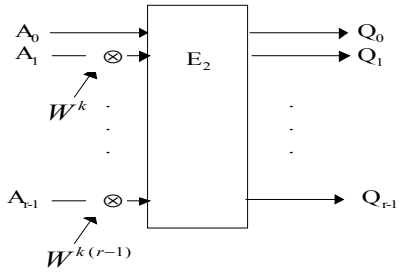
$$\begin{vmatrix} Q_0 \\ Q_1 \\ Q_2 \\ \dots \\ Q_{r-2} \\ Q_{r-1} \end{vmatrix} = \left[E_r\right] \cdot \begin{vmatrix} W^0 & & & & & \\ & W^{1K} & & & & \\ & & W^{2K} & & & \\ & & & \dots & & \\ & & & & W^{K(r-1)} & \end{vmatrix} \cdot \begin{vmatrix} A_0 \\ A_1 \\ A_2 \\ \dots \\ A_{r-2} \\ A_{r-1} \end{vmatrix} \qquad (3.13)$$

и для алгоритмов с прореживанием по частоте соответственно:

$$\begin{vmatrix} Q_0 \\ Q_1 \\ Q_2 \\ \dots \\ Q_{r-2} \\ Q_{r-1} \end{vmatrix} = \begin{vmatrix} W^0 & & & & & \\ & W^{1K} & & & & \\ & & W^{2K} & & & \\ & & & \dots & & \\ & & & & W^{K(r-1)} & \end{vmatrix} \cdot \left[E_r\right] \cdot \begin{vmatrix} A_0 \\ A_1 \\ A_2 \\ \dots \\ A_{r-2} \\ A_{r-1} \end{vmatrix} \qquad (3.14)$$

Графически каждая из представленных базовых операций может быть изображена так, как это указано на рис.3.3.

а)



б)

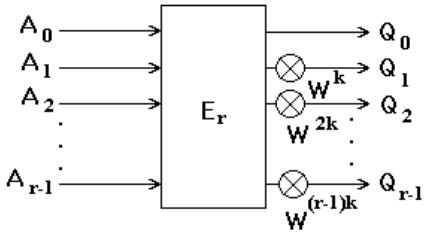


Рис. 3.3. Графы базовых операций БПФ по основанию *r*

Остановимся на правилах перестановки элементов в таком случае. На входе выполняется перестановка по закону r - ичной инверсии, т.е. на входе элементы вектора X объединяются в подвектора из r - элементов, причём шаг выборок

составит: $\Delta = r^{M-1}$

С помощью системы рекуррентных соотношений, подобных выражению (3.9), алгоритм БПФ для $N=r^M$ с прореживанием по времени можно описать следующим образом [24]:

$$F^m = \begin{vmatrix} f_{k,m} \\ f_{k+l,m} \\ f_{k+2l,m} \\ \dots \\ f_{k+(r-1)l,m} \end{vmatrix} = [E_r] \cdot \begin{vmatrix} W_{N_m}^0 & 0 & 0 & \dots & 0 \\ 0 & W_{N_m}^k & 0 & & 0 \\ 0 & 0 & W_{N_m}^{2k} & & 0 \\ & & \dots & \dots & 0 \\ 0 & 0 & 0 & & W_{N_m}^{(r-1)k} \end{vmatrix} \cdot \begin{vmatrix} f_{k,m-1} \\ f_{k+l,m-1} \\ f_{k+2l,m-1} \\ \dots \\ f_{k+(r-1)l,m-1} \end{vmatrix} \qquad (3.15)$$

где $E_r = \left\| \exp\left(-jsq2\pi / r\right) \right\|$; $s,q \in \overline{0,r-1}$ - матрица ДЭФ порядка r (т.е. ДПФ для вектора N сводится к ряду ДПФ размера r над блоками), $k=(n)_{\text{mod } l}$, n текущий индекс элемента вектора ($n = \overline{0,N-1}$), $k \in \overline{0,l-1}$,

$$l = r^{m-1},\ W_{N_m}^{pk} = \exp(-j\frac{2\pi}{N_m} \cdot pk),\ p \in \overline{0,r-1},\ N_m = r^m$$

Граф БПФ для $N=9$ имеет вид, представленный на рис.3.4

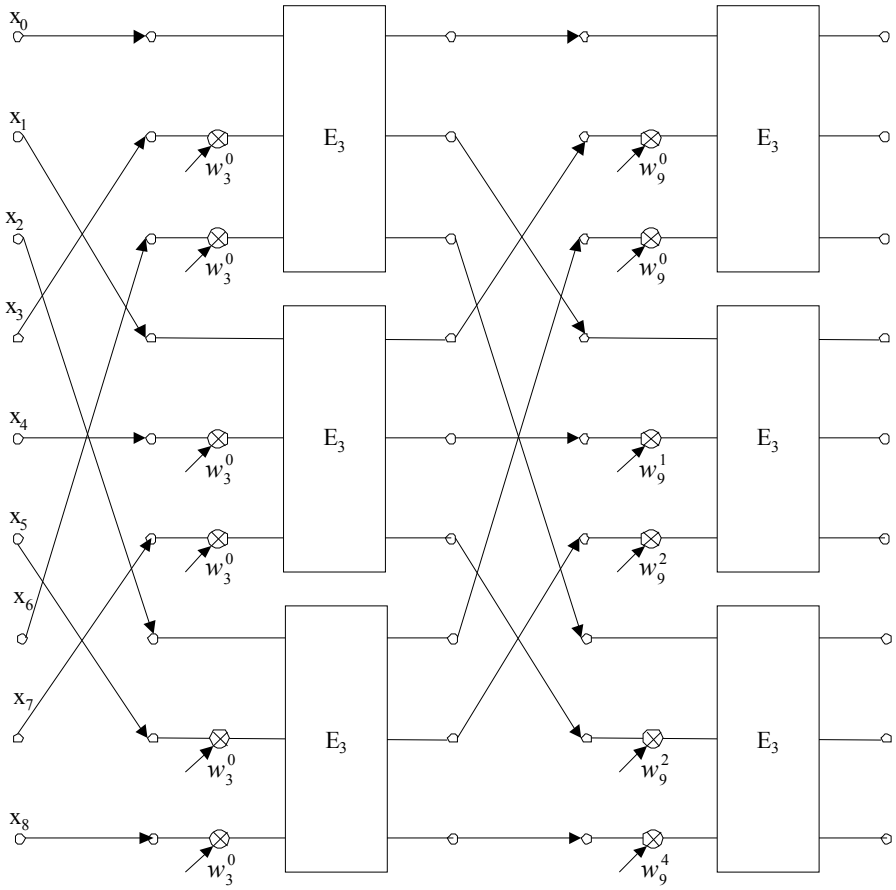


Рис.3.4. Граф БПФ

3.6. Вычислительная сложность алгоритмов БПФ

Рассмотрим вначале алгоритмы БПФ для $N = 2^m$ с прореживанием по времени.

Такой алгоритм является итерационным и включает $M = \log_2 N$ итераций, причем на каждой стадии выполняется N/2 базовых операций вида (3.11), откуда нетрудно получить, что общая трудоемкость алгоритма БПФ [22]:

$$Q = \left[\frac{N}{2} \log_2 N \right] q_{Бо} \qquad (3,16)$$

где $q_{БО}$ - сложность базовой операции.

В свою очередь, базовая операция требует для своего выполнения 1 операцию умножения и 2 операции сложения комплексных чисел или, в пересчете на операции с действительными числами, 4 операции умножения и 4 операции сложения действительных чисел.

Для ЭВМ предыдущих поколений, где операции умножения выполнялись главным образом программным (а не аппаратным) способом, особую важность имело прежде всего сокращение числа операций умножения как наиболее длительных.

В современных ЭВМ эта задача не столь актуальна, так как длительность практически всех операций близка, особенно в специализированных процессорах.

Поэтому будем полагать, что сложность базовой операции при работе с комплексными числами равна

$$q_{б_0 k} \approx 4 q_{Dr}$$

где $q_D = q_x + q_+$

Здесь и далее под q подразумевает сложность отдельной машинной команды, например, в числе тактов работы процессора, необходимой для выполнения конкретной арифметической операции. Следовательно, для БПФ сложность алгоритма можно определить как:

$$Q_2 = 2 \left[N \log_2 N \right] q_D \qquad (3.17)$$

При обработке вектора данных длиной $N = r^m$ на каждой из M итераций выполняется по N/r базовых операций, т.е.

Однако сложность базовой операции, как это можно видеть из (3.15), составит:

$$Q_1 = \left[\frac{N}{r}\log_2 N\right] q_{БО} \qquad (3.18)$$

$$q_{БО} \approx r^2 q_k$$

где $q_k = q_{УМН} + q_{СЛ}$ - для комплексных чисел, или, как и в случае $N = 2^m$, посчитаем, что $q_k = 4q_D$ и получим, подставив в (3.15) значения $q_{БО}$:

$$Q_r = 4N\left[\log_2 N\right] r^2 q_D \qquad (3.19)$$

В таблице приведены значения для времени выполнения ДПФ и БПФ по различным основаниям при условии, что $t_{сл} = t_{умн} = 100нсек.$

N		БПФ	ДПФ
16	2^4	$2{,}56\cdot10^{-5}$	$2{,}0\cdot10^{-4}$
25	5^2	$2\cdot10^{-4}$	$5\cdot10^{-4}$
27	3^3	$6{,}5\cdot10^{-5}$	$6{,}5\cdot10^{-5}$
32	2^5	$6{,}4\cdot10^{-5}$	$8\cdot10^{-4}$
81	3^4	$7{,}8\cdot10^{-4}$	$5{,}3\cdot10^{-3}$
125	5^3	$1{,}5\cdot10^{-3}$	$1{,}2\cdot10^{-3}$
128	2^7	$3{,}6\cdot10^{-4}$	$1{,}3\cdot10^{-2}$
512	2^9	$1{,}8\cdot10^{-3}$	$2\cdot10^{-1}$
625	5^4	10^{-2}	$3\cdot10^{-1}$
1024	2^{16}	$4{,}1\cdot10^{-3}$	$8{,}3\cdot10^{-1}$
16384	2^{14}	≈1 сек.	≈36 мин.

Поэтому с точки зрения сокращения вычислительных затрат выгодны алгоритмы БПФ по основанию 2 (или 4), причем с ростом основания r выигрыш уменьшается .

3.7. Выполнение БПФ для случаев $N \neq r^M$

Если $N = m_1 * m_2 * m_3 * m_4 ... m_M$, то в этом случае выполняется алгоритм БПФ по смешанному основанию. Он может быть реализован как алгоритм с прореживанием по частоте или как алгоритм с прореживанием по времени. В обоих случаях такой алгоритм выполняется за M итераций. Однако на каждой итерации выполняется различное число базовых операций, причём размерность матрицы ядра ДПФ базовой операции на каждой итерации различна и составляет

E_{mi}, а число БО соответственно на каждой итерации составляет - N/m_i, где до первой итерации элементы вектора переупорядочиваются с шагом N/m_i [8]. На рис. 3.5. приведен в качестве примера граф БПФ для $N = 6 = 3x2$.

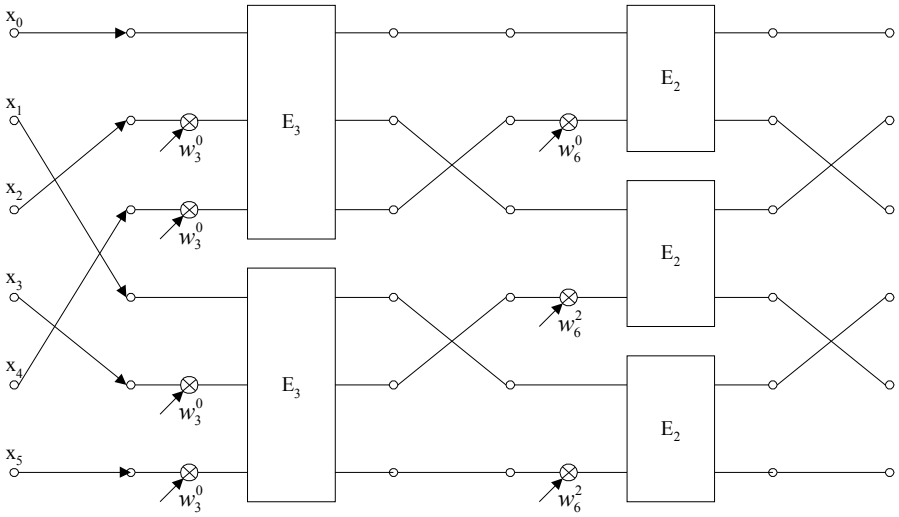


Рис.3.5. Граф БПФ для $N=6$

Если же N является простым числом и не может быть разложено на взаимно-простые множители, то в этом случае также можно использовать алгоритм быстрого преобразования, но особого вида - так называемый алгоритм Винограда [16]. Этот алгоритм позволит значительно сократить число операций умножения, но зато число операций сложения сокращается не столь существенно, причем требуются достаточно сложные процедуры перекомпоновки векторов промежуточных результатами.

Поэтому для современных ЭВМ, и особенно спецпроцессоров, когда $\Delta t_{умн} \approx \Delta t_{сл}$ (а в некоторых из них одинаково время любой операции), такие алгоритмы, как и БПФ по смешанному основанию, не приносят слишком большого выигрыша по времени.

С точки зрения экономии времени целесообразно дополнить вектор до $N = r^M$ (желательно $N = 2^M$ или $N = 4^M$) нулями. Однако такой прием приводит к некоторому искажению результата, что связано с введением так называемой оконной функции. Пусть исходных сигнал задан в виде вектора из N_1 отсчетов (рис.3.6.).

Для того, чтобы иметь возможность использовать алгоритмы БПФ дополним сигнал нулевыми отсчетами до общего числа отсчетов $N= 2^M$, причем N – ближайшее большее к N_1 .В этом случае исходный сигнал оказывается заданным с помощью функции

$$\hat{f}(x) = f(x) * g(x),$$

где *g(x)* – функция окна вида

$$g(x) = rect(\frac{X * \frac{N_1}{2}}{N_1})$$

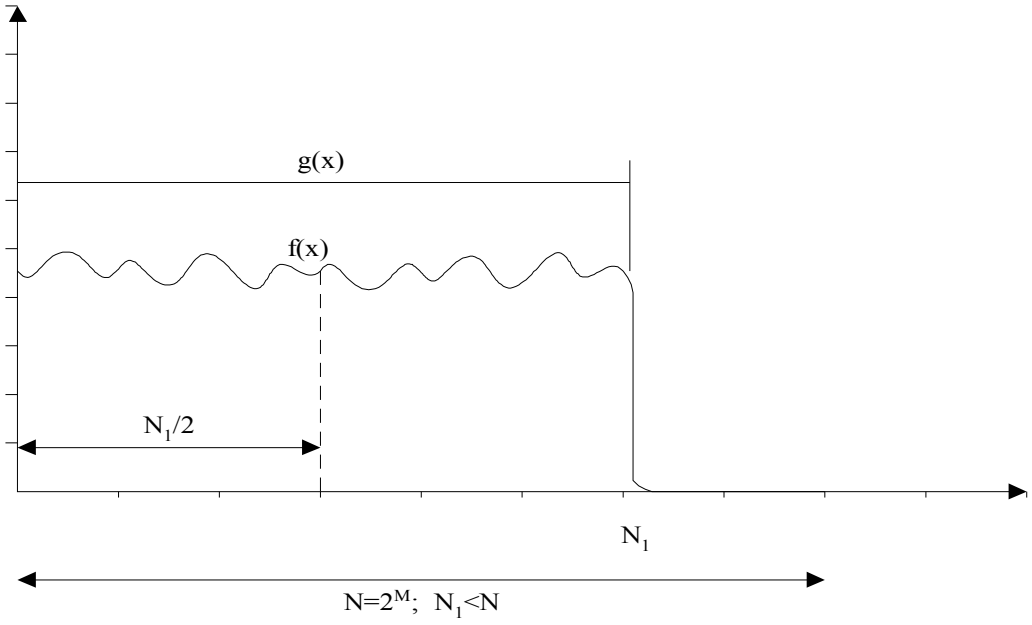


Рис.3.6. Проявление эффекта окна

Поэтому при вычислении преобразования Фурье от $\hat{f}(x)$ проявляется эффект, получивший название “эффекта окна”.

$$\hat{F}_{(\varsigma)} = \int \hat{f}(x) e^{-j2\pi\varsigma x} dx = \int [f(x) * g(x)] e^{-j2\pi\varsigma x} dx = F_{(\varsigma)} \otimes G_{(\varsigma)},$$

где $G_{(\zeta)} = \sin c(\zeta / N_1)$ при этом функция $G_{(\varsigma)}$ будет тем ближе к δ - функции, чем шире $rect[\frac{2x}{N_1 \Delta x}]$ или чем N_1 ближе к N. Проявляется “эффект окна” в

искажении периферийных областей спектра сигнала. Поэтому, во избежание проявления “эффекта окна” удобнее функцию $f(x)$ при переходе от N_1 к N сделать периодической с периодом N_1, дополнив вектор исходных данных до длины N, повторив первые $(N - N_1)$ отсчетов за последние элементы вектора X.

3.8. Быстрое преобразование Хартли

Как уже отмечалось в разделе 2.5., множители ядра преобразования Харли не обладают свойством мультипликативности. Это не позволяет достаточно просто записать выражения для итерации алгоритма БПХ в матричной форме.

Выполним выкладки для получения выражений, описывающих итерацию БПХ. Для этого возьмем за основу выражения для описания “бабочки” БПФ ($N = 2^m$) произвольной итерации согласно (3.10):

$$\begin{cases} f_{k,m} = f_{k,m-1} + f_{k+l,m-1} \cdot \left[\cos\dfrac{2\pi k}{N_m} - j \sin\dfrac{2\pi k}{N_m} \right] (1) \\ f_{k+l,m} = f_{k,m-1} - f_{k+l,m-1} \cdot \left[\cos\dfrac{2\pi k}{N_m} - j \sin\dfrac{2\pi k}{N_m} \right] \end{cases} \qquad (3.20)$$

здесь l=2^{m-1}, $k = (n) \bmod l$, $N = 2^m$.

Воспользуемся выражениями, связывающими отсчёты ДПФ и ДПХ [3]:

$$\begin{cases} f_{k,m-1} = (h_{k,m-1} + h_{p,m-1})/2 - j(h_{k,m-1} - h_{p,m-1})/2 & (*) \\ f_{k+l,m-1} = (h_{k+l,m-1} + h_{p+l,m-1})/2 - j(h_{k+l,m-1} - h_{p+l,m-1})/2 & (**) \\ f_{k,m} = (h_{k,m} + h_{p,m})/2 - j(h_{k,m} - h_{p,m})/2 & (***) \end{cases}$$

подставим (*), (**) и (***) в формулу (1) выражения (3.20) и получим:

$$\begin{aligned} &(h_{k,m} + h_{p,m}) - j(h_{k,m} - h_{p,m}) = \left|(h_{k,m-1} + h_{p,m-1}) - j(h_{k,m-1} - h_{p,m-1})\right| + \\ &+ \left[(h_{k+l,m-1} + h_{p+l,m-1}) - j(h_{k+l,m-1} - h_{p+l,m-1})\right] \cdot \left[\cos\frac{2\pi k}{N_m} - j \sin\frac{2\pi k}{N_m} \right] \end{aligned} \qquad (3.21)$$

В свою очередь, из выражения (3.21) можно получить выражения для

$$\left(h_{k,m} + h_{p,m}\right) = \left(h_{k,m-1} + h_{p,m-1}\right) + \left(h_{k+1,m-1} + h_{p+l,m-1}\right) \cos\frac{2\pi k}{N_m} - \left(h_{k+l,m-1} - h_{p+l,m-1}\right) \sin\frac{2\pi k}{N_m} \qquad (3.22)$$

действительной и мнимой частей:

Сложив выражения (3.22) и (3.23), можно получить:

$$h_{k,m} = h_{k,m-1} + h_{k+l,m-1} \cos\frac{2\pi k}{N_m} + h_{p+l,m-1} \sin\frac{2\pi k}{N_m}$$

$$h_{k,m} - h_{p,m} = \left(h_{k,m-1} - h_{p,m-1}\right) + \left(h_{k+l,m-1} + h_{p+l,m-1}\right)\sin\frac{2\pi k}{N_m} + \left(h_{k+l,m-1} + h_{p+l,m-1}\right)\cos\frac{2\pi k}{N_m} \qquad (3.23)$$

$$h_{k+m} = h_{k,m-1} + h_{k+l,m-1}\cdot\cos\frac{2\pi k}{N_m} + h_{p,m-1}\cdot\sin\frac{\pi k}{N_m} \qquad (3.24)$$

Проделав аналогичные выкладки, можно получить, что:

$$h_{k+l,m} = h_{k,m-1} - h_{k+l,m-1}\cos\frac{2\pi k}{N_m} - h_{p,m-1}\sin\frac{\pi k}{N_m} \qquad (3.25)$$

здесь $p = l+(l-k)_{mod\ l}$, $N_m = 2^m$, $e = r^{m-1}$, $k=(n)_{mod\ l}$, n - текущий индекс элемента вектора ($n = \overline{0, N-1}$).

Таким образом, при $N_m = 2^m$ “бабочка” БПХ подобна “бабочке” БПФ, но обладает следующим отличием: правило вычисления индекса элемента синусной компоненты иное, чем у индекса косинусного элемента. Индекс же элемента косинусной компоненты и “свободной” компоненты, а также индекса у результирующего элемента и аргументы косинуса и синуса вычисляются так же, как и для “бабочки” БПФ.

На рис. 3.7. приведен граф базовой операции бабочка для алгоритма БПХ.

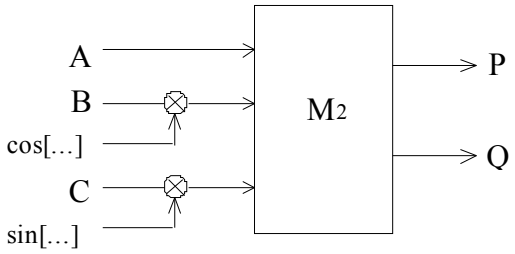


Рис.3.7. Граф базовой операции БПХ, где

$$\begin{cases} P = A + B \cdot \cos[\ldots] + C \cdot \sin[\ldots] \\ Q = A - B \cdot \cos[\ldots] - C \cdot \sin[\ldots] \end{cases}$$

Граф алгоритма БПХ для $N = 2^3$ приведен на рис.3.8.

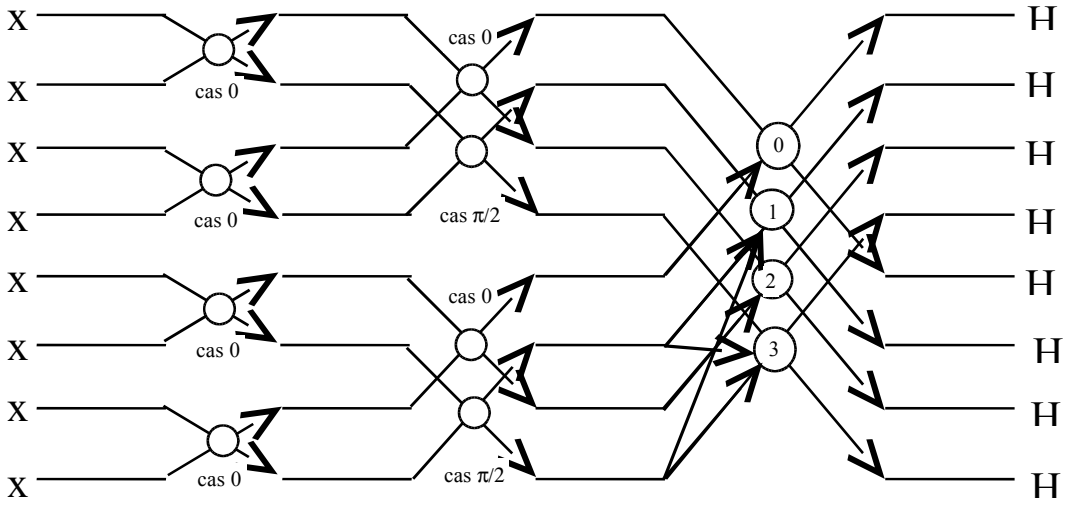


Рис.3.8. Граф алгоритма БПХ

Из представленного графа видно, что на третьей итерации при вычислении $H_1 \div H_5$ и $H_3 \div H_7$ в “бабочках” участвуют cos[...] и sin[...] компоненты с разными элементами.

3.9. Быстрое преобразование Адамара

Как отмечалось в главе 2, матрица ядра Адамара

$$A_N = E_2 \otimes E_2 \otimes E \ldots \otimes E_2 = \otimes_{i=2}^{M-1} E_2$$

Поэтому алгоритм быстрого преобразования Адамара может быть легко получен на основе алгоритма БПФ$_2$, если из него удалить операцию умножения на

матрицу D поворачивающих элементов. На основании этого из выражения (3.10) можно получить:

$$\begin{cases} a_{k,m} = a_{k,m-1} + a_{k+l,m-1} \\ a_{k+l,m} = a_{k,m-1} - a_{k+l,m-1} \end{cases} \qquad (3.26)$$

где, как и в (3.10) $l = 2^{m-1}$, $k = (n)_{mod\, l}$, n - текущий индекс вектора ($n = \overline{0, N-1}$).

Иначе говоря, “бабочка” в таком алгоритме имеет вид как это показано на рис.3.9

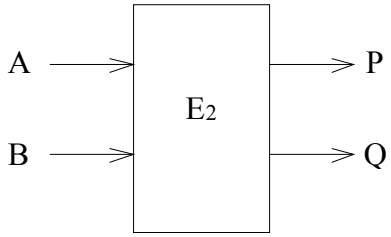


Рис.3.9. Бабочка алгоритма быстрого преобразования Адамара, где

$$\begin{cases} P = A + B \\ Q = A - B \end{cases}$$

Такая же точно “бабочка”, как и представленная на рис.3.7, служит основой алгоритмов быстрого преобразования Пэли и Уолша, отличие которых состоит лишь в упорядочении исходных данных перед первой итерацией (см. главу 2).

Граф быстрого алгоритма Уолша – Пэли для $N = 16$ приведен на рис.3.10. Сложность же алгоритма быстрых ортогональных преобразований Уолша - Адамера в числе Б0 алгоритма $Q_{БПФ}$ и $Q_{БПХ}$ для $N = 2^M$, однако сложность самой базовой операции значительно меньше и составит только 2 операции сложения, что значительно меньше чем для алгоритма БПХ (не говоря уже о БПФ). Поэтому можно записать, что:

$$Q_{БДОП} = \left[N \log_2 N \right] q_{сл}.$$

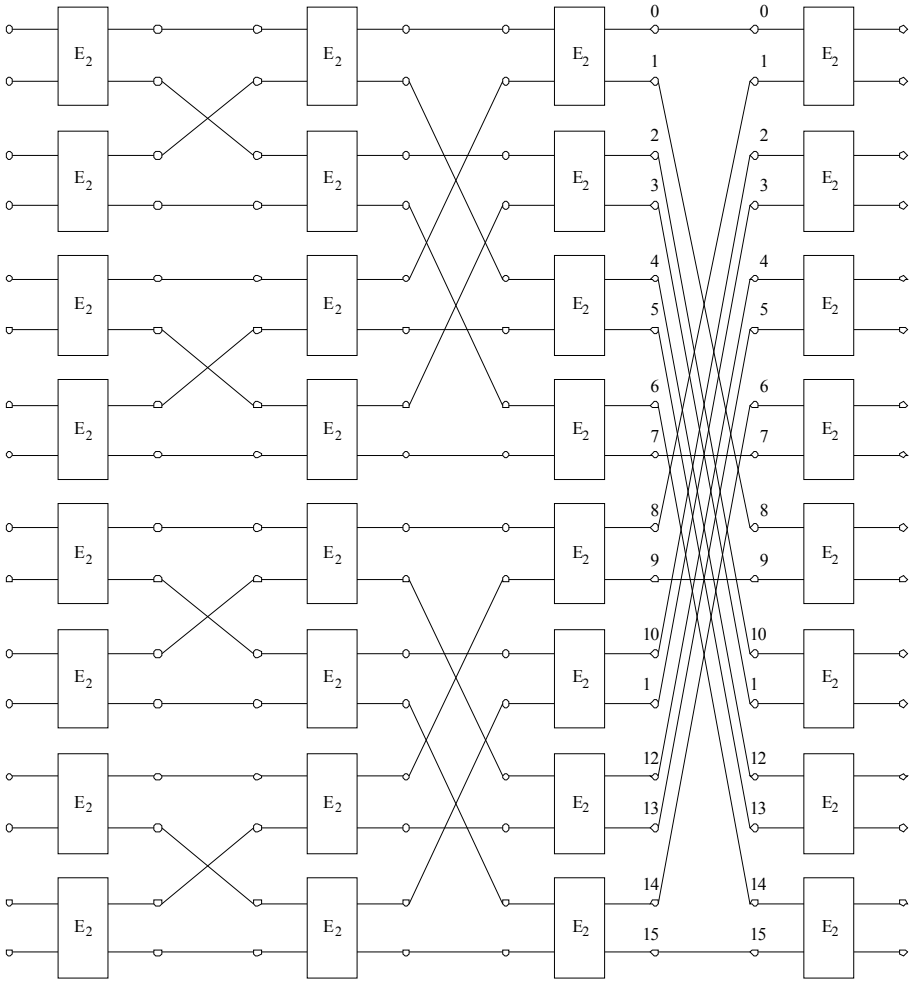


Рис.3.10. Граф алгоритма быстрого преобразования Уолша-Адамара для *N=16*

Быстрые алгоритмы БДОП Адамара-Уолша являются «обратимыми», т.е. вычисления по графу (см., например, рис.3.10) могут выполняться как «слева направо», так и «справа налево».

Заметим, что для алгоритма БДОП с прореживанием по частоте не выполняется двоично-инверсная перестановка элементов вектора X перед первой итерацией, но зато необходимо переставлять элементы векторов результата по закону двоичной инверсии.

На рис. 3.11 представлен граф быстрого преобразования Хаара. Нетрудно видеть, что он является частично вырожденным по отношению к графу БДОП Адамара-Уолша.

Сам алгоритм быстрого преобразования Хаара (за исключением итоговой нормировки) будет описываться выражением (3.26), за тем лишь исключением, что при обработке каждого блока данных выполняется только первая базовая операция (*k=0*) Для БП Хаара сложность отдельной базовой операции такая же,

как и у БПУ. Однако общее число базовых операций меньше, чем в алгоритмах БПХ и БПУ, и составляет всего *Q = (2N-1) БО*.

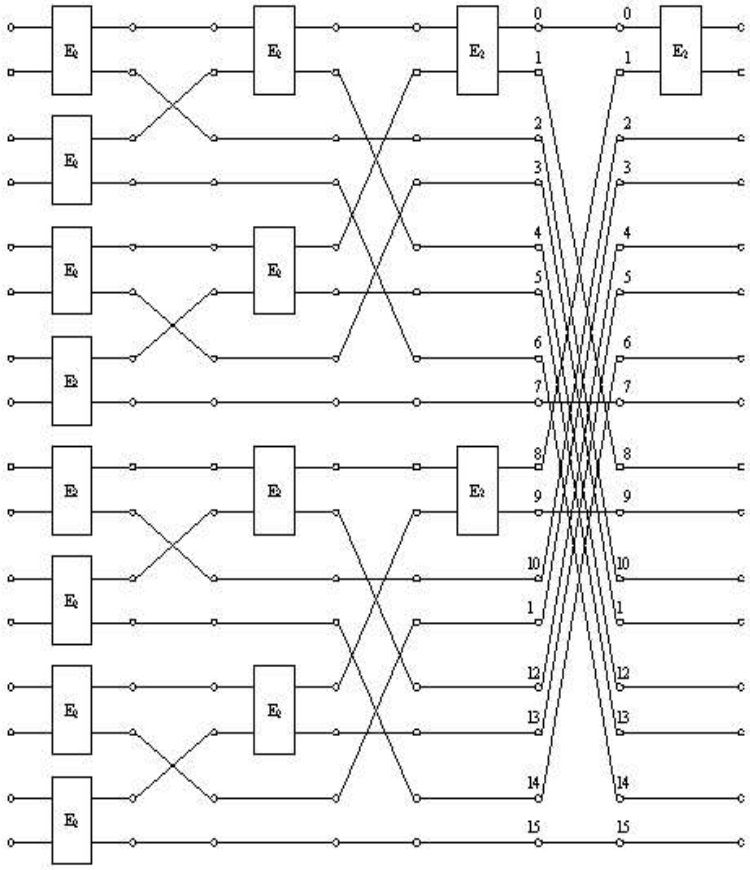


Рис.3.11. Граф быстрого преобразования Хаара для N =16.

4. ЛИНЕЙНАЯ ФИЛЬТРАЦИЯ СИГНАЛОВ ВО ВРЕМЕННОЙ И ЧАСТОТНОЙ ОБЛАСТЯХ

Основная задача линейной фильтрации - выделение полезного сигнала на фоне случайного шума в определенном диапазоне частотного спектра [21, 22].

Известны следующие методы линейной фильтрации, обеспечивающие улучшение соотношения сигнал/шум:

- метод накопления;
- частотная фильтрация;
- корреляционный метод;
- согласованная фильтрация;
- адаптивная фильтрация.

Все указанные методы улучшения отношения сигнал/шум могут быть эффективны при выполнении следующих условий:

1.Между помехой и полезным сигналом должны быть определенные различия.

2.Эти различия должны быть известны заранее.

Если одно из этих условий не выполняется, то нельзя ожидать положительного эффекта от применения любого из указанных методов.

Основные задачи, решаемые на основе линейной фильтрации, указаны в таблице 4.1.

4.1. Метод накопления

Метод накопления применим в том случае, если полезный сигнал в течении времени приема постоянен или является периодической функцией. Метод состоит в многократном повторении сигнала и суммировании отдельных его реализаций в устройстве обработки. Данный метод относится к группе точечных алгоритмов обработки сигналов.

Пусть полезный сигнал представлен двумя уровнями (рис.4.1).

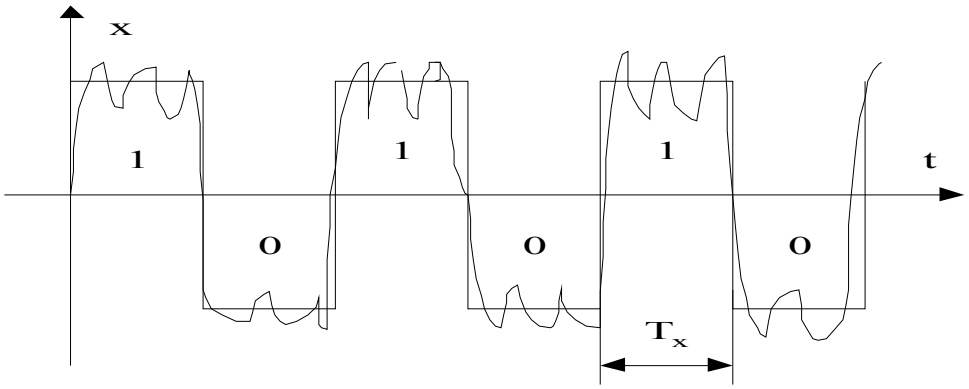


Рис.4.1. Входной двухуровневый сигнал

В интервале T_x сигнал постоянен. На интервале наблюдения T_x накапливается выборка значений принятого сигнала

$$y_1=x+n_1$$
$$y_2=x+n_2$$
$$\ldots\ldots\ldots\ldots$$
$$y_m=x+n_m$$

и эти значения суммируются:

$$Y=\sum_{i=1}^{n} y_i = nx+\sum_{i=1}^{n} r_i .$$

Введем два допущения:

1) отсчеты помехи n_i не зависят друг от друга;

2) помеха стационарна (ее характеристики не зависят от времени)

и определим $\left(\frac{P_x}{P_r}\right)_{вых}$ на выходе этого накопителя, т.е.

$$\left(\frac{P_x}{P_r}\right)_{вых} = \frac{(nx)^2}{(\sum_{i=1}^{n} r_i)^2} = \frac{(nx)^2}{nD_r} = \frac{n^2x^2}{nP_r} = \frac{nP_x}{P_r}. \qquad (4.1)$$

Таким образом, при перечисленных выше условиях, в результате n - кратного отсчета, отношение мощностей сигнала и помехи увеличивается в n раз. Временной интервал между отдельными отсчетами должен быть больше

Таблица 4.1.

Основные задачи ЦОС, решаемые на основе методов линейной фильтрации

Решаемые задачи	**Область применения**	**Выполняемые операции преобразования**
Выделение сигнала на фоне шумов	Радиофизика, радиолокация, обработка изображений	одно- и двумерная свертка
Обнаружение и определение координат объекта на изображении	Радиофизика, гидроакустика, обработка изображений	Вычисление двумерной функции корреляции с эталоном в скользящем режиме
Выделение мелких деталей и границ объекта	Обработка изображений	Двумерная свертка
Сокращение информационной избыточности изображений	Системы передачи, обработки и архивации изображений	Двумерная свертка со специальным ядром
Адаптация системы к изменению входного воздействия	Радиолокация, гидроакустика, системы технического зрения	Вычисление обратной взаимокорреляционной матрицы

интервала корреляции помехи τ_0^r. В противном случае выигрыш за счет накопления будет меньше значения, даваемого выражением (4.1).

За счет увеличения числа отсчетов m, т.е. времени передачи T_x, можно сколь угодно увеличивать отношения сигнал/помеха.

Если сигнал представляет периодическую функцию времени, то отсчеты нужно производить через интервалы, равные или кратные периоду этой функции. В таких случаях метод носит название метода синхронного или когерентного накопления. Эффект накопления такой же, как в случае постоянного сигнала.

Эффект накопления можно осуществить также за счет интегрирования входного сигнала в течении времени T_x. Такой метод получил название интегрального приема.

Интегральный прием целесообразно применять в случае, когда полезный сигнал постоянен (или квазипостоянен).

4.2. **Рекурсивные и не рекурсивные фильтры**

Одной из областей ЦОС, в основе выполнения процедур которой лежит вычисление выражений типа свертки, является цифровая линейная фильтрация

Цифровые фильтры по принципу выполнения фильтрации могут быть разделены на два класса - рекурсивные и не рекурсивные фильтры [21, 22].

Математически функционирование не рекурсивного фильтра описывается уравнением вида :

$$y_n = \sum_{k=0}^{N-1} h_k x_{n-k} \qquad (4.2)$$

Такой фильтр может рассматриваться как устройство без обратной связи, имеющее структуру, показанную на рис.4.2 :

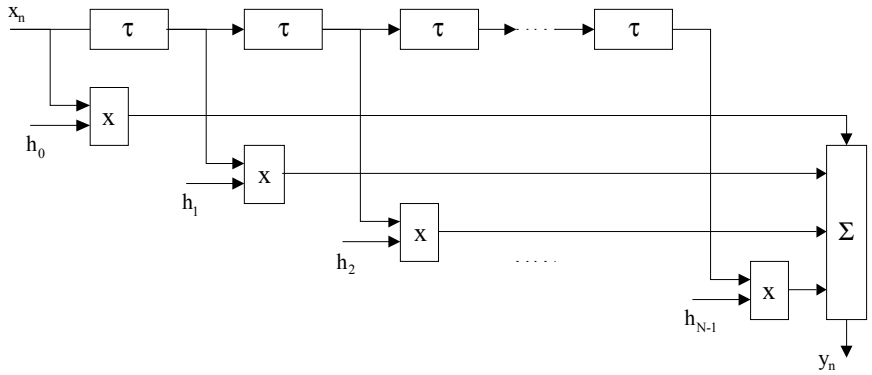


Рис. 4.2. Нерекурсивный линейный фильтр.

Работа же рекурсивного фильтра описывается выражением :

$$y_n = -\sum_{j=1}^{N-1} b_j y_{n-j} - \sum_{k=0}^{N-1} h_k x_{n-k}, \qquad (4.3)$$

где y_n - n-й отсчет выходного сигнала; x_n - n-й отсчет входного сигнала; h_k - коэффициенты фильтра.

Нетрудно видеть, что такой рекурсивный фильтр может рассматриваться как устройство с обратной связью.

4.3. Выбор метода вычисления свертки / корреляции

Таким образом, в основе работы не рекурсивного фильтра лежит вычисление свертки вектора исходных данных

$$X = \left[x_0, x_1, x_2, \ldots x_{N-1} \right]$$

и импульсной характеристики фильтра (ядра свертки)

$$G = \left[g_0, g_1, g_2, \ldots g_{M-1} \right],$$

где $N >> M$. Подобные вычисления могут быть вычислены как на основе прямого алгоритма (см. раздел 1.7), так и через спектральные дискретные преобразования Фурье и Хартли (см. разделы 2.1 и 2.2). Возникает естественный вопрос, какой метод предпочтителен, исходя из меньшего объёма вычислений, а следовательно, и времени, необходимого для реализации данного метода.

Если размер “окна” или ядра свёртки равен M , а длина вектора исходных данных, то для получения N отсчётов результата апериодической свёртки необходимо выполнить:

$$Q_{св} = MN\ (БО), \qquad (4.4)$$

где БО – базовая операция, включающая операцию умножения и сложения действительных чисел. Сложность БО составляет:

$$q_{БО} = q_{умн} + q_{сл} .$$

Отсюда можно получить, что время вычисления свертки по прямому алгоритму

$$T_{св} = Q_{св} * \Delta t = MN(q_{умн} + q_{сл)} * \Delta t$$

где Δt - длительность такта В.У.

При этом необходимо отметить, что задаваемые данные и результаты являются действительными числами, т.е. при расчёте прямого алгоритма $q_{БО}$ определяется суммой операций умножения и сложения действительных чисел.

При вычислении свёртки на основе спектрального преобразования выполняются следующие действия:

1. Прямые спектральные преобразования Фурье или Хартли исходного вектора.

2. При вычислении свёртки на основе ДПХ выполняются перекомпоновка полученного спектра (см. лекцию, т.е. формируется $\hat{H}_{(\varsigma)}$ и вычисляются $\left[H_{(\varsigma)} + H_{(-\varsigma)}\right]$ и $\left|H_{(\varsigma)} - H_{(-\varsigma)}\right|$.
3. Поэлементное перемножение спектров для ДПФ : $\Phi_{(\varsigma)} = H_{(\varsigma)} * G_{(\varsigma)}$ или вычисление в $\Phi_{(\varsigma)}$ согласно методике в разделе 2.2 - для ДПХ.
4. Обратное преобразование для функции $\Phi_{(\varsigma)}$ на основе ДПФ или ДПХ.

Таким образом, если свёртка при $N = 2^N$ вычисляется через ДПФ на основе быстрых алгоритмов:

$$\begin{aligned}\hat{Q}_{св} &= Q^1_{БПФ} + Q_{умн} + Q^{-1}_{БПФ} = 2Q_{БПФ} + Nq_{умн} = \left[N \log_2 N\right]q_{БО} + Nq_{умн} = \\ &= \left[N \log_2 N\right](2\hat{q}_{сл} + \hat{q}_{умн}) + N\hat{q}_{умн}\end{aligned} \quad (4.5)$$

где $q_{БО} = 2\hat{q}_{сл} + \hat{q}_{умн}$

При расчёте $\hat{Q}_{св}$ свёртки необходимо учитывать, что

$$q_{БО} = 2\hat{q}_{сл} + \hat{q}_{умн} \approx 2(\hat{q}_{сл} + \hat{q}_{умн}),$$

где $\hat{q}_{сл}$ и $\hat{q}_{умн}$ - сложность операций сложения и умножения комплексных чисел, причём

$$\hat{q}_{сл} = 2q_{сл}$$
$$\hat{q}_{умн} = 4q_{умн} + 2q_{сл}$$

аналогичных действий с действительными числами, т.е.

$$q_{БО} = 4(q_{умн} + 4q_{сл}) = 4(q_{умн} + q_{сл})$$

Тогда получим

$$\hat{Q}_{св} = [4N \log_2 N](q_{сл} + q_{умн}) + 4N(q_{сл} + q_{умн}) = 4N[1 + \log_2 N](q_{сл} + q_{умн}) \quad (4.6)$$
$$\hat{T}_{св} = 4N[1 + \log_2 N](q_{умн} + q_{сл})\Delta t$$

Отсюда нетрудно получить с учетом выражения (4.4), что прямой метод вычисления свёртки предпочтителен, если:

$$\underline{M < 4(1 + \log_2 N)} \quad (4.7)$$

Из (4.7) следует, например, что для $N = 1024$ предельный размер $M = 44$ для вычисления по прямому алгоритму, а для $N = 128$ предельный размер окна составляет $M = 32$

Если свёртка вычисляется на основе БПХ, то при том же числе БО их сложность умножается:

$$q_{БО} = 4q_{сл} + 2q_{умн} = 2(q_{умн} + q_{сл})$$

для действительных чисел, однако требуется выполнить операции по вычислению функции $\hat{\Phi}_{(\varsigma)}$, что потребует примерно $\approx 2N(2q_{сл} + q_{умн})$. С учетом сложности базовых операций при прямом и спектральном алгоритме можно получить, что прямой алгоритм предпочтительнее алгоритма вычисления свертки через БПХ, если :

$$M \le \frac{3}{2}[1 + \log_2 N] \quad (4.8)$$

Что же касается БДОП типа Уолша - Адамара, то базис этих функций для вычисления свёртки мало пригоден, поскольку в отличии от ДПХ для подобных преобразований нет простых формул для связи с ДПФ и не справедлива теорема о свёртке. Известны алгоритмы вычисления свёртки и на основе указанных ДОП [18, 19], однако отсутствие операции умножения при выполнении БДОП не компенсируется сложностью вычисления $\hat{\Phi}$.

Таким образом, от фильтрации во временной области выполняется переход к фильтрации в частотной области.

4.4. Выполнение фильтрации в частотной области

Идея частотной фильтрации основана на отличии спектров полезного сигнала и помехи. При этом используются линейные частотные фильтры, позволяющие подавлять помеху и улучшать тем самым соотношение сигнал/помеха. Параметры фильтра определяются спектральными характеристиками сигнала и помехи. На практике наиболее часто встречаются следующие случаи.

1) На вход фильтра поступает узкополосный сигнал и широкополосная помеха. В этом случае эффективен узкополосный фильтр с полосой пропускания $\Delta\omega_x$;

2) На вход фильтра поступает широкополосный сигнал и узкополосная помеха с шириной спектра $\Delta\omega_r$. Для подавления подобной помехи фильтр должен обеспечивать подавление помехи в полосе $\Delta\omega_r$;

3) На вход фильтра поступают периодический сигнал и широкополосная помеха.

Рассмотрим случай, когда полезный сигнал является гармоническим, а помеха типа белого шума. Для выделения полезного сигнала в этом случае должен быть использован узкополосный фильтр, настроенный на частоту сигнала. Отношение мощности сигнала к мощности помехи на выходе фильтра при этом

$$\left(\frac{P_x^*}{P_r}\right)_{\text{‰ħħ}} = \frac{(P_x)_{\text{‰›}}}{P_0\Delta\omega_{\text{тм}}} = \frac{(P_x)_{\text{‰›}}}{2\pi\Delta f_{\text{тм}}P_0}, \quad (4.9)$$

где P_0 - средняя мощность помехи, приходящаяся на единицу полосы; $\Delta\omega_ф$ - полоса пропускания фильтра. Как видно из выражения (4.9), отношение $\frac{P_x^*}{P_r}$ можно сделать сколь угодно большим за счет уменьшения полосы пропускания фильтра $\Delta f_ф$.

В реальных условиях полезный сигнал поступает лишь в течении определенного времени T_x и, следовательно, его спектр неограничен.

Известно, что практическая ширина спектра такого сигнала связана с его длительностью соотношением

$$\Delta f_ф T_x = \mu, \qquad (4.10)$$

где μ - постоянная, зависящая от формы сигнала. Обычно принимается $\mu \cong 1$.

Длительность сигнала T_x должна быть выбрана такой, чтобы его спектр был не шире полосы пропускания фильтра $\Delta f_x \le \Delta f_ф$.

Подставляя в (4.9) вместо $\Delta f_ф$ величину Δf_x, получаем

$$\left(\frac{P_x^*}{P_r}\right)_{вых} = \frac{\left(P_x\right)_{вх}}{P_0 \Delta\omega_ф} = \frac{\left(P_x\right)_{вх}}{2\pi P_0 \mu} T_x . \quad (4.11)$$

Формула (4.11) показывает, что увеличение отношения сигнал/помеха достигается за счет увеличения длительности сигнала T_x, т.е. времени наблюдения.

Таким образом, при частотной фильтрации улучшение отношения сигнал/помеха окупается ценой увеличения времени наблюдения сигнала.

Кроме указанных видов фильтров в частотной области может выполняться линейная фильтрация более общего вида, описываемая выражением типа свертки. Схема выполнения фильтрации сигнала в частотной области приведена на рис. 4.3. Обобщенно подобный метод частотной фильтрации иногда называют операционной частотной фильтрацией. Таким методом можно выполнять заданные математические преобразования исходного сигнала, например, дифференцирование или интегрирование [21, 28].

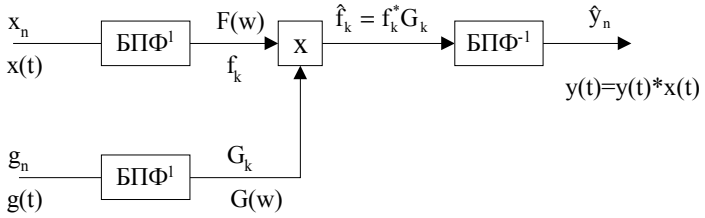


Рис.4.3. Схема операционной частотной фильтрации

Согласованные фильтры предназначены для выделения сигналов известной формы на фоне шумов. Критерием оптимальности таких фильтров является получение на выходе максимально возможного отношения амплитудного значения сигнала к действующему значению помехи. Реакция согласованного фильтра эквивалентна действию корреляционного приемника. Схема выполнения согласованной фильтрации приведена рис. 4.4.

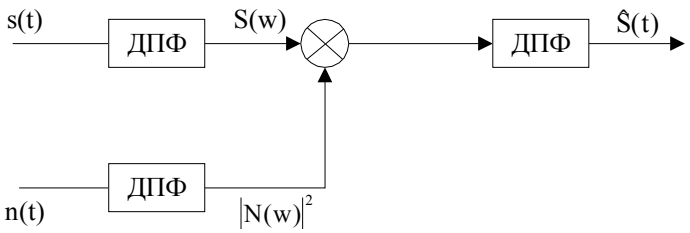


Рис. 4.4. Схема выполнения согласованной фильтрации

Для выделения известного сигнала S(t) из стационарного шума n(t) оптимальным является фильтр с передаточной функцией [21, 28]:

$$H(\omega)=\frac{S^{*}(\omega)}{\left|N(\omega)\right|^{2}} \qquad (4.12)$$

$S^{(*)}(\omega)$ - сопряженный спектр Фурье (или Фурье образ) исходного сигнала, $\left|N(\omega)\right|^{2}$ - спектральная плотность шума, $\omega=2\pi\nu$. Этот фильтр носит название согласованного фильтра.

Если нужно минимизировать среднеквадратичную ошибку восстановленного сигнала, то:

$$H(\omega)=\frac{1}{F(\omega)}\cdot\frac{\varphi_0\varphi_n^{-1}}{\varphi_0\varphi_n^{-1}+\left|F(\omega)\right|^{-2}} \qquad (4.13)$$

где $F(\omega)$ - спектр сигнала; φ_0 - спектр мощности восстанавливаемого сигнала, φ_n - спектр мощности шума.

При отсутствии шума для минимизации среднеквадратичной погрешности используется инверсный фильтр [28]:

$$H(\omega)=\frac{1}{F(\omega)}=\frac{F^{*}(\omega)}{\left|F(\omega)\right|^{2}} \qquad (4.14)$$

4.5. Адаптивные фильтры

Адаптивный фильтр - это фильтр, передаточная функция (или частотная характеристика) которого адаптируется, т.е. изменяется таким образом, чтобы пропустить без искажений полезные составляющие сигнала и ослабить нежелательные сигналы или помехи [26]. Схема адаптивного фильтра представлена на рис.4.5.

Подобный фильтр действует по принципу оценки статистических параметров сигнала и подстройки собственной передаточной функции таким образом, чтобы минимизировать некоторую целевую функцию. Эту функцию обычно формируют с помощью “эталонного” сигнала на задающем входе. Этот эталонный сигнал можно рассматривать как желаемый сигнал на выходе фильтра. Задача блока адаптации состоит в подстройке коэффициентов

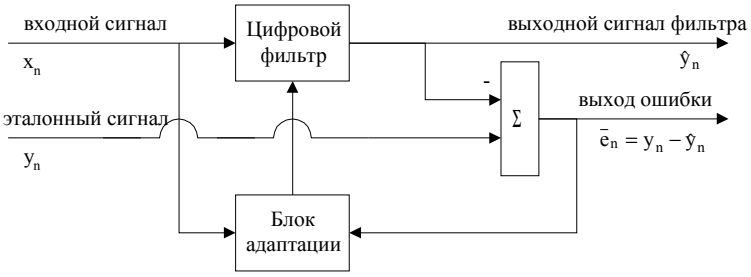
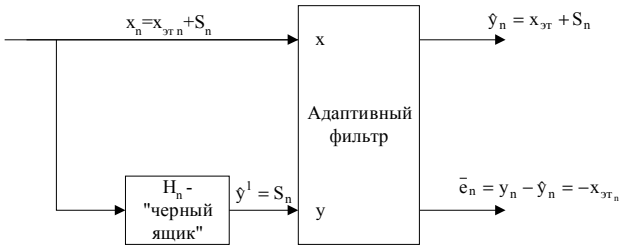


Рис.4.5. Адаптивный фильтр

цифрового фильтра таким образом, чтобы свести к минимуму разность $\hat{e}_n = \hat{y}_n - \hat{x}_n$, определяеющую ошибку в работе фильтра. Варианты включения адаптивного фильтра паказаны на рис. 4.6,а) и б).

а)



б)

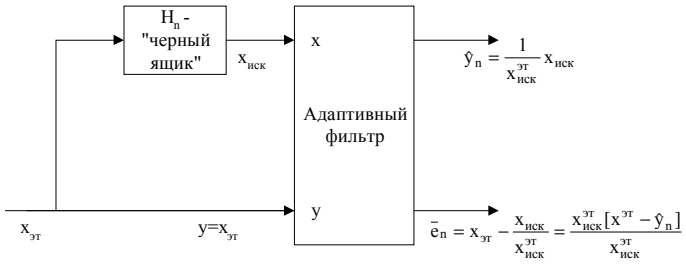


Рис.4.6. Два варианта включения адаптивного фильтра

Для варианта на рис.4.6,а) оптимальный импульсный отклик адаптивного фильтра равен импульсному отклику “черного ящика”:

$$H_{opt} = H_n$$

Фильтры первого рода - это фильтры для подавления шума. Здесь поступающий сигнал, являющийся смесью полезного сигнала с шумом (помехой)

приложен ко входу x_n , а от другого источника, не содержащего никаких составляющих полезного сигнала, поступает образец “мешающего” сигнала - т.е. шума или помехи, тогда на выходе фильтра :

$$x_n = S_n + x^{\wedge}_n ;$$

$$y_n = S_n ;$$

Для варианта на рис.4.6,б) оптимальный импульсный отклик фильтра равен обратному импульсному отклику “черного ящика”:

$$H_{opt} = H_n^{-1}$$

Примером адаптивного фильтра второго типа является фильтр для коррекции искажений при передаче данных по линии связи. В этом случае вход линии связи возбуждается известным сигналом, а искаженный сигнал поступает на вход *x*. Затем фильтр перестраивается с помощью подачи на вход *x*^ набора известных неискаженных сигналов.

4.6. Оптимальный фильтр Винера

Определим импульсный отклик не рекурсивного адаптивного фильтра, позволяющего минимизировать среднеквадратичную погрешность при выделении сигнала из случайного шума.

Адаптивный фильтр формирует оценку:

$$\bar{e} = y_n - \hat{y}_n \quad (4.15)$$

где $\bar{e}$ - ошибка, y_n - искомая величина, - оценка y_n .

$$\hat{y}_n = f(x_n, h_k) \qquad (4.17)$$

$\hat{y}_n$ - функция от последовательности входных сигналов x_n и набора весов фильтра h_k..

В свою очередь:

$$x_n = x_n + S_n \qquad (4.17)$$

где xn - исходный сигнал, S_n - белый шум с дисперсией δ_S^2 .

Зададим как меру оценки среднеквадратичную ошибку:

$$\left\{e_n^2\right\} = \overline{\left|\hat{y}_n - y_n\right|}^2 \qquad (4.18)$$

При этом выходной сигнал может быть представлен в виде:

$$\hat{y}_n = \sum_{k=0}^{N-1} x_{n-k} h_k = \sum_{k=0}^{N-1} h_k x_{n-k} \qquad (4.19)$$

или

$$\hat{y}_n = H^T X_n = X_n^T H \qquad (4.20)$$

где $X_n = [x_n \quad x_{n-1} \quad \dots \quad x_{n-N}]^T$, $H^T = [h_0 \quad h_1 \quad \dots \quad h_{N-1}]$.

Тогда:

$$\{e_n^2\} = \{y_n - H^T X_n\}^2 \qquad (4.21)$$

Для определения минимума найдем производную по H^T:

$$\frac{\partial e_n^2}{\partial H^T} = -2\left[\left(y_n - H^T X_n\right) X_n^T\right] \qquad (4.22)$$

Условие минимума:

$$\left[\left(y_n - H^T X_n\right) X_n^T\right] = 0$$

Откуда:

$$\left[y_n X_n^T\right] = \left[\left(H^T X_n\right) X_n^T\right] \qquad (4.23)$$

Если вектор H^T и вектор X не коррелированны, то:

$$E[y_n X_n^T] = H_{opt}^T E[X_n X_n^T] \qquad (4.24)$$

$E[X_n X_n^T] = R$ - матрица автокорреляции входной сигнальной последовательности (размером NxN), $[y_n X_n^T]$=P- вектор взаимной корреляции между входным сигналом x_n и оцениваемым параметром y_n длиной N.

В итоге получаем:

$$P^T = H_{opt}^T R$$

или

$$H_{opt}^T = P^T R^{-1} \qquad (4.25)$$

Выражение (4.25) получило название *уравнения Винера-Хопфа.* Оно определяет правило формирования импульсного отклика адаптивного фильтра, обеспечивающего минимальную среднеквадратичную ошибку.

Адаптивный рекурсивный фильтр, позволяющий минимизировать среднеквадрачную ошибку сигнала при выделении его на фоне случайного шума, получил название *фильтра Калмана* [26].

4.7. Методы обращения матриц

Для фильтра Винера-Хопфа при вычислении оптимального импульсного отклика фильтра необходимо выполнять обращение автокорреляционных или ковариационных матриц. Поэтому остановимся на методах обращения матриц, используемых при решении задач ЦОС.

Вычислительная сложность процедуры обращения матриц достаточно велика, поэтому предназначенные для решения задач ЦОС в реальном масштабе времени алгоритмы обращения должны допускать распараллеливание и конвейеризацию вычислений. По этим причинам для обращения матриц в задачах ЦОС часто используют методы обращения, основанные на процедуре исключения (алгоритмы Гаусса, Гаусса-Жордана) [13, 18]. Рассмотрим процедуру обращения матриц по методу Гаусса-Жордана.

Пусть для матрицы A_N порядка N требуется найти обратную матрицу Z_N:

$$Z_N = A_N^{-1}$$

Заметим, что для матрицы A_N может и не существовать обратная матрица. Для того, чтобы существовала матрица Z_N, необходимо и достаточно, чтобы матрица A_N была не вырожденной [4], т.е. чтобы определитель матрицы $det[A_N] \neq 0$.

Запишем матричное уравнение:

$$A_N Z_N = I_N$$

где I_N - единичная матрица. Сформируем матрицу вида:

$$\hat{A}_0 = [A_N \mid I_N]$$

размером $N \times 2N$ и за N итераций приведем ее к виду:

$$\hat{A}_N = [I_N \mid Z_N]$$

при помощи соотношений:

$$\begin{cases} b^k = a_{kk}^{k-1}, k = 1, N \\ a_{kj}^k = \dfrac{a_{kj}^{k-1}}{b^k}, j = \overline{k, 2N} \end{cases} \qquad (4.26)$$

$$\begin{cases} U_l^k = a_{lk}^{k-1}, l = \overline{1, N} \\ a_{lj}^k = a_{lj}^{k-1} - a_{kj}^k U_l^k, l \neq k \end{cases} \qquad (4.27)$$

причем $a^{k-1}_{lj} = \hat{a}_{lj}$. Здесь b^k - ведущий элемент, который может определяться по номеру итерации как *k*- тый элемент *k*- й строки, либо как максимальный элемент *k* -го столбца [4]. Выражение (4.26) определяет главный или ведущий элемент на каждой итерации и приведение (масштабирование) коэффициентов строки, содержащей главный элемент. Назовем такую строку ведущей. Выражение (4.27) определяет порядок исключения, т.е. вычитания ведущей строки из остальных строк матрицы.

Подобный алгоритм почти в *N* раз снижает вычислительные затраты по сравнению с обращением матрицы по формулам Крамера.

Тем не менее, вычислительная сложность процедуры обращения матрицы по выражениям (4.26) и (4.27) все же достаточно велика и составляет порядка N^3 базовых операций.

С целью дальнейшего снижения вычислительной сложности в ряде случаев используют приближенные итерационные процедуры обращения , например, по алгоритму Ньютона. Согласно такому алгоритму, для матрицы A_N выполняются последовательные приближения:

$$X_N^{j+1} = X_N^j (2X_N^{(0)} - A_N X_N^j), \quad j = 0,1,2... \quad (4.28)$$

где $X_N^{(0)}$ - произвольное начальное значение исходной матрицы. Если данная последовательность сходится, то ее пределом является $Z_N = A_N^{-1}$.

Для ряда практических применений, например, расчета оптимального импульсного отклика адаптивного фильтра, используют упрощенные алгоритмы, к которым относится алгоритм Гриффитса [7]:

$$Y^{(k)} = N_N^{(k)} X^{(k)}$$

$$N_N^{(k)} = R_N^{-1(k)} \qquad (4.29)$$

$$r_{mn} = x_m^{(k)} y_n^{(k)}$$

Вместо нахождения обратной матрицы при нахождении импульсного отклика фильтра Винера-Хопфа:

$$\hat{H} = R^{-1}$$

вычисляют матрицу:

$$\tilde{H}^{(k+1)} = \tilde{H}^{(k)} - \mu[I_N - R_N] \qquad (4.30)$$

где μ - некоторая действительная переменная:

$$\mu = \frac{1}{\sum_m (\overline{x_m})^2}$$

Однако при этом остается открытым вопрос о сходимости последовательности $\tilde{H}^{(k+1)}$, что требует в конкретном случае предварительного исследования такой сходимости.

5. АЛГОРИТМЫ НЕЛИНЕЙНОЙ ОБРАБОТКИ СИГНАЛОВ

Ранее рассматривались алгоритмы, относящиеся к типу линейных преобразований. Характерная черта этих преобразований состоит в том, что они могут быть описаны в терминах матричной алгебры. При этом, если $Y=B_N X$ и B_N является невыраженной, т.е. $det[B_N]\neq 0$, (замети, что такое требование всегда соблюдается для ортогональной матрицы B_N), то $X = B_N^{-1} Y$.

Это позволяет однозначно перейти от вектора *Y,* являющегося результатом преобразования, к вектору X . Поэтому, если результат обработки сигнала не удовлетворяет по тем или иным соображениям пользования, вновь можно получить исходный сигнал X и выбрать другое ядро преобразования B_N или заменить алгоритм.

Базовые операции любых линейных преобразований требуют выполнения операций умножения (деления) и сложения (вычитания) чисел. При выполнении линейных преобразований данные и результаты могут быть как в формате с фиксированной точкой, так и в формате с плавающей точкой. Однако при использовании формата с фиксированной точкой возникает необходимость увеличения разрядности данных в ходе обработки. Поясним это на примере.

Пусть вычисляется апериодическая свертка с ядром размером M=9 элементов при разрядности исходных данных и коэффициентов ядра, равной 8 бит (причем исходные данные – положительные числа, а коэффициенты ядра знакопеременны):

q{X}=8; (x>=0) q{G}=8 (включая знак)

Вычисление свертки производится по прямому алгоритму:

$$Y_n = \sum_{m=-4}^{4} x_{n-m} q_{n-m}$$

В этом случая разрядность любого частичного произведения составляет q {MPL}=15, а с учетом вычисления суммы таких произведений разрядность конечного результата равна q{Yn}=15+4=19=20 бит.

Еще сложнее обстоит дело с увеличением разрядности при использовании формата с фиксированной точкой при выполнении быстрых ортогональных преобразований Фурье и Хартли. Это связано с необходимостью последовательного выполнения операции умножения формируемого результата на различные поворачивающие множители. Поэтому при большой длине вектора разрядность результата становится недопустимо большой. Усечение же его может внести недопустимо большие погрешности или потерять значимость части спектральных коэффициентов при большем динамическом диапазоне сигнала.

Поэтому формат чисел с фиксированной точкой целесообразно использовать для вычисления свертки с малым ядром или для ортогональных преобразований Уолша-Адамара. Для циклической свертки при большом N и для реализации алгоритмов БПФ и БПХ целесообразно использование формата чисел с плавающей точкой. Однако такой формат существенно усложняет выполнение арифметических операций.

Помимо линейных преобразований для обработки сигналов в настоящее время используются так называемые нелинейные преобразования. К числу нелинейных преобразований в частности, относится [17, 21]:

- ранговая фильтрация;
- взвешенная ранговая фильтрация;
- гистограммные преобразования.

Указанные преобразования не могут быть описаны в терминах операций матричной алгебры, хотя, как и операции алгебраической свертки, рекурсивной и не рекурсивной фильтрации, они относятся к группе локальных преобразований и выполняются в “скользящем” режиме при последовательном перемещении окна сканирования размером M элементов вдоль вектора из N элементов (M<<N, M - нечетное).

Вторая характерная особенность указанных нелинейных алгоритмов состоит в том, что они могут быть применены к исходным данным только формата с фиксированной точкой при ограниченной разрядности операндов. Увеличения разрядности результатов при выполнении подобных преобразований не происходит, что существенно упрощает вычисления.

Кроме указанных особенностей, нелинейные преобразования обладают, к сожалению, негативным свойством – они необратимы, т.е. от результата обработки невозможно вернуться к исходным данным. Поэтому при выполнении нелинейной обработки приходится сохранять исходный сигнал во избежание риска его потери до получения удовлетворяющих пользователя результатов.

5.1. Ранговая фильтрация

Рассмотрим в начале выполнение ранговой фильтрации в одномерном случае. При ранговой фильтрации для i-го положения окна сканирования требуется выполнить следующие действия [21]:

1. Все элементы вектора X, попавшие в окно сканирования, переупорядочиваются в порядке возрастания:

$$\left\{ x_{i-(M/2+1)}; \ldots x_{i-(M/2+1)} \right\} \rightarrow \left\{ \hat{x}_{\min}; \ldots \hat{x}_{\max} \right\}$$

2. В качестве результата y_i выбирается значение: $y_i = \hat{x}_R$, где R – заданный номер позиции элемента в упорядоченном списке.

Производится смещение окна сканирования на одну позицию: $i=i+1$

3. Повторяются пункты 1-3.

Очевидно, что если $R=1$, то выполняется так называемая минимальная фильтрация, поскольку всегда в качестве результата выбирается наименьший элемент в окне; если же $R=M$, - то за результат выбирается максимальный элемент окна.

При $R=(M+1)/2$ выполняется медианная или срединная фильтрация. При ранговой фильтрации при размере окна M происходит искажение фильтруемого сигнала - высокочастотные компоненты сигнала с периодом $<R$ будут пропущены.

Для того, чтобы сигнал при фильтрации не был подвержен существенному искажению, необходимо, чтобы:

$$T_{\max} \geq R \ \ (\text{при } R = med \rightarrow T_{\min} \geq \frac{M+1}{2}). \qquad (5.1)$$

Если размер окна невелик, то ранговую фильтрацию можно выполнить посредством сортировки элементов окна. Однако при больших размерах окна и жестких ограничениях на время фильтрации сортировку выполняют с использованием специальных аппаратных средств - сортирующих сетей [14].

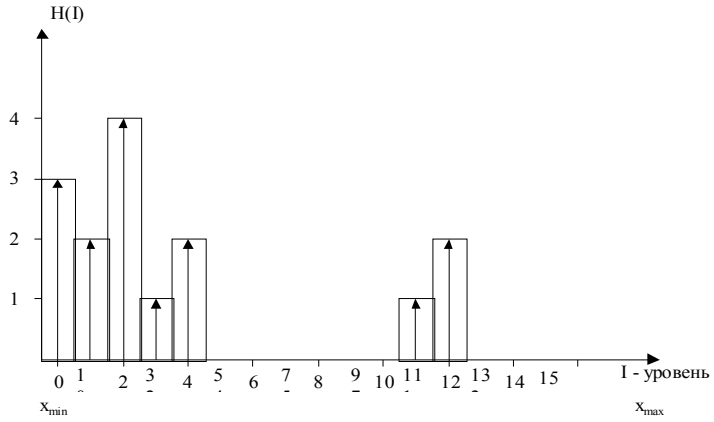
Возможны ещё два алгоритма выполнения РФ:

- гистограммный алгоритм;
- разрядно-срезовый алгоритм.

При гистограммном алгоритме строится гистограмма H(I), т.е. распределение числа элементов окна по уровням (т.е. по значениям):

$$R \leq \sum_{I=0} H(I) \qquad (5.2)$$

При выполнении ранговой фильтрации согласно (5.2) строится гистограмма и вычисляется её площадь (сумма значений), до тех пор, пока число элементов учтённых уровней в гистограмме не превысит значение ранга R. Последний учтённый уровень и определяет значение элемента заданного ранга. На рис.5.1. показан пример поиска элемента заданного ранга по построенной гистограмме.



$R_1 = 5 \rightarrow x = 1$
$R_2 = 7 \rightarrow x = 2$
$R_3 = 10 \rightarrow x = 3$
$R_4 = 12 \rightarrow x = 4$
$R_5 = 13 \rightarrow x = 11$

Рис.5.1. Гистограмма распределения элементов по уровням

Гистограммный алгоритм ранговой фильтрации для окна размером M x M может быть представлен в следующем виде [17]:

начало;

H(I):=0; *для* $\forall i = 0,2^Q - 1$

для $m = 1, M$ *цикл:*

для $n = 1, M$ *цикл:*

$I := B_{nm}; H(I) := H(I) + 1;$

конец цикла по n*;*

конец цикла по m*;*

$D := 0;$

для $i = 0,2^Q - 1$ *цикл:*

если $D \geq R$ *то выход из цикла;*

иначе $D := D + H(i)$*;*

конец цикла по i*;*

$I_R := i;$

конец

Здесь B_{mn}- значение отсчета с номером (m,n), Q - разрядность отсчетов. Поскольку фильтрация выполняется в скользящем режиме, то при перемещении окна вдоль строки изображения достаточно модифицировать имеющуюся гистограмму предшествующего положения окна, включив в нее элементы нового столбца и исключив элементы, соответствующие "ушедшему" столбцу. Модификация гистограммы сводится к следующим операциям:

1 выполнить *M* раз вычитание *H(I) := H(I) - 1* для элементов $I = B'_{mn}$ столбца, выходящего из окна сканирования;

1 выполнить *M* раз сложение *H(I) := H(I) + 1* для элементов $I = B_{mn}$ нового столбца полосы.

Очевидно, что такой подход позволяет значительно сократить время формирования гистограммы. Дальнейшее сокращение времени фильтрации может быть достигнуто путем сокращения времени поиска элемента заданного ранга. Для этого оказывается удобным строить не одну, а Q гистограмм, причем $H_1(I)$ строится только по старшему разряду, $H_2(I)$ - по двум старшим разрядам и так далее, до $H_Q(I)$, которая строится по всем разрядам элементов окна. В результате поиск по Q гистограммам организуется как поиск по Q -

уровневому бинарному дереву. Поиск результата начинается с первой гистограммы $H_1(I)$, по которой определяется старший разряд. Анализ каждой следующей гистограммы уточняет очередной младший разряд результата, в результате за Q циклов будет определен Q-разрядный результат фильтрации .

Разрядно-срезовый алгоритм РФ может быть интерпретирован как поиск необходимого значения по по двоичному дереву и сведен к следуещему(пусть разрядность данных – 4 бита) :

1. Анализируем биты старшего среза. Если “0” больше, чем R, то это значит, что элемент ранга R лежит среди отсчётов с нулевым старшим битом;
2. Анализируем второй срез, исключив при его рассмотрении те отсчеты x_m, для которых старший бит =1 (т.е. модифицируем второй срез). Если “0” во втором срезе больше, чем R, то это значит, что элемент заданного ранга лежит среди отсчетов со старшими битами “00”;
3. Анализируем третий срез, исключив при его рассмотрении те отсчеты x_m, у которых во втором модифицированном срезе бит =1, т.е. у которых старшие два бита равны “01”. Если “0” в третьем срезе больше, чем R, то это значит, что элемент заданного ранга лежит среди отсчетов со старшими битами “000”;
4. Анализируем четвертый срез, исключив при его рассмотрении те отсчеты x_m, у которых старшие биты равны ”001”. Если “0” в младшем срезе больше, чем R, то это значит, что элемент заданного ранга лежит среди отсчетов со старшими битами “0000”, иначе среди тех, у которых младший бит =1, т.е. x_m,=”0001”.

Заметим, что “1” в последнем младшем срезе указывает и местоположение, и число элементов заданного ранга

Подобный алгоритм получил название разрядно-срезового алгоритма со сквозным маскированием. Очевидно, что отдельно взятый разрядный срез не позволяет определить искомый элемент I_R. Для этого необходимо дополнительно иметь результаты анализа на предыдущих итерациях поиска. Поэтому в разрядно - срезовом алгоритме требуется, чтобы при переходе к анализу текущего разрядного среза были известны все элементы, исключенные из поиска на предыдущих этапах, т.е. требуется задание срезовой маски. В

результате обработки текущего разрядного среза определяется очередной разряд результата I_R и маска, содержащая всю информацию об исключенных элементах списка на q -ом этапе поиска.

Запишем подобный алгоритм (для фиксированного положения окна) [17]:

начало:

$D_0 = -R$;

для $q = 1, Q$ *цикл:*

$n_0 := 0$;

для $m = 1, M$ *цикл:*

для $n = 1, M$ *цикл:*

если $b_{mn} = 1$ *то* $n_0 := n_0$,

иначе $n_0 := n_0 + 1$;

конец цикла по n ;

конец цикла по m ;

$D_q := D_{q-1} + n_0$; ;

если $D_q \geq 0$, *то*

{

$d_q := 0$,

для $k = q, Q - 1$ *цикл:*

для $m = 1, M$ *цикл:*

для $n = 1, M$ *цикл:*

$b_{mn,k+1} := b_{mn,k+1} \cup b_{mn,q}$;

конец цикла по n ;

конец цикла по m ;

конец цикла по k;

$D_q := D_{q-1}$;

}

иначе

{

$d_q := 1$;

для $k = q, Q - 1$ *цикл:*

для $m = 1, M$ *цикл:*

для $n = 1, M$ *цикл:*

$b_{mn,k+1} := b_{mn,k+1} \cup \overline{b}_{mn,q}$;

конец цикла по n ;

конец цикла по m ;

конец цикла по k;

$\}$

конец цикла по q *;*

$I_R := d_1 d_2 d_3 \ldots d_Q$ *;*

конец.

Недостатком алгоритма со сквозным маскированием является необходимость после анализа текущего среза модифицировать все последующие, что резко увеличивает объем вычислений и практически не позволяет распараллелить вычисления по срезам. Указанный недостаток преодолевается путем ввода маски, воздействующей только на один последующий срез и не изменяющей другие разрядные срезы.

Такая маска S_q может быть найдена в соответствии с выражением [23]:

$$S_q = \overline{S_{q-1} \& (B_q \# d_{q-1})}, \qquad (5.3)$$

где B_q - q-й разрядный срез элементов окна, а символ # обозначает логическую операцию неравнозначности. Накапливаемая сумма D_q определяется в этом случае из соотношения:

$$D_q = \begin{cases} D_{q-1} + \Delta D_q, d_{q-1} = 1. \\ D_{q-1} - \Delta D_q, d_{q-1} = 0. \end{cases} \qquad (5.4)$$

$$\Delta D_q = \sum S_q \& \left(B_{q-1} \# B_q \right)$$

Такой подход и является основой алгоритма с последовательным маскированием. Таким образом, для обработки последующего разрядного среза B_q необходимо сформировать маску S_q и передать предыдущий разрядный срез B_{q-1}.

Возможны три варианта процедуры разрядно-срезовой ранговой фильтрации: четный, нечетный и четно-нечетный алгоритмы. На каждой q-й итерации четного алгоритма анализируются элементы q-го разрядного среза, имеющие четные значения, т.е. элементы, q-ый разряд которых равен нулю. Для нечетного алгоритма на q-й итерации учитываются те элементы, соответствующий q-й разряд которых равен единице, а в четно-нечетном алгоритме на различных итерациях могут учитываться как четные, так и

нечетные элементы .

В разрядно-срезовых алгоритмах каждый разрядный срез должен быть представлен как битовая строка длиной M^2 бит, что при размере окна более чем *5 x 5* превышает разрядность большинства существующих операционных элементов. Поэтому в разрядно-срезовых алгоритмах следует предусмотреть возможность обработки разрядных срезов по фрагментам (например по столбцам). В этом случае обрабатывается вектор длиной лишь *M* бит.

Модифицированный четный разрядно-срезовый алгоритм ранговой фильтрации, позволяющий оптимизировать конвейерный режим вычислений с возможностью обработки срезов по столбцам, может быть записан в следующем виде [23]:

Начало :

$D_0 := -R; I_R = 00000\ldots0; S_0 := 11111\ldots11;$

цикл для $q = 1, Q$ *:*

$D_q := D_{q-1};$

$\Delta := 0;$

цикл для $i = 1, M$

$\Delta := \Delta + \sum_q^1 (S_{q-1}^i \,\&\, \overline{B}_q^i);$

конец цикла по i

$D_q := D_q + \Delta;$

если $D_q >= 0$ *, то*

$\{ d_q := 0, D_q := D_{q-1};$

цикл для $i = 1, M$

$S_q^i := S_{q-1}^i \,\&\, \overline{B}_q^i$

конец цикла по i $\}$ *;*

иначе $\{ d_q := 1,$

цикл для $i = 1, M$

$S_q^i := S_{q-1}^i \,\&\, B_q^i$

конец цикла по i $\}$ *;*

конец цикла по q *;*

$I_R := d_1 d_2 d_3 \ldots d_Q;$

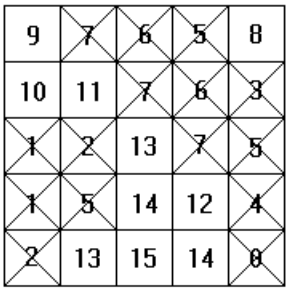
конец.

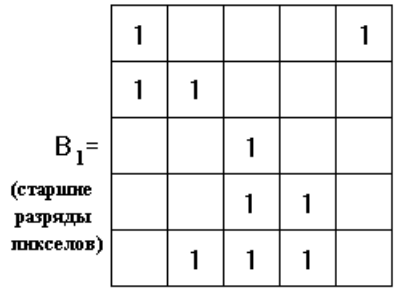
Здесь Q - разрядность данных; S_q - q-я разрядно-срезовая бинарная маска ; B_q - q-й разрядный срез матрицы данных ; $\sum_q^1$ - число единичных бит в маскированном разрядном срезе $\overline{B}_q, d_q$ - вычисленое значение q-го разряда результата; I_R - результат ранговой фильтрации.

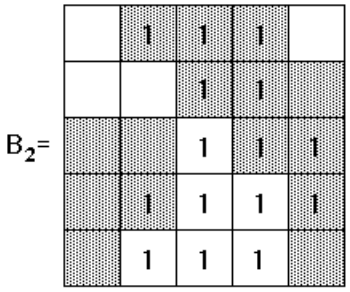
Следует отметить, что при использовании подобного алгоритма может быть определена не только величина элемента заданного ранга R, но и его местоположение в окне. Информация о местоположении элемента I_R содержится в бинарной маске S_q, сформированной на последней итерации алгоритма. Положение единичного бита (или единичных бит) определяет положение элементов заданного ранга R в окне сканирования.

Рассмотрим примеры выполнения ранговой фильтрации по разрядно-срезовому алгоритму со сквозным маскированием.

1. Пусть M×M=5×5; Q=4 и надо определить элемент ранга<u>R=13</u> (выполняется медианная фильтрация)



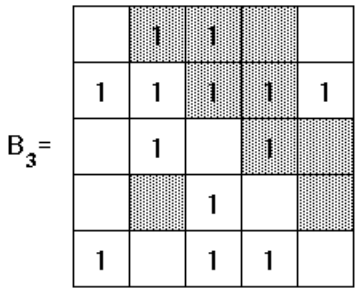


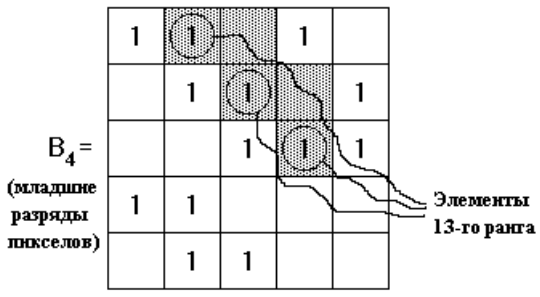


0) S_0=11...1111;
D_0=-13;
$Y_{i0}=I_R$;

1) $\Sigma_1=S_0\&\overline{B}_1=15$;
D_1=-13+15=2; $D_1>0; d_1=0$;
D_1=-13; $S_1=S_0\&B_1$;

2) $\Sigma_2=S_1\&\overline{B}_2$;
D_2=-13+6=--7;
D_2=-7; $d_2=1$; $D_2<0$;
D_2=-7; $S_2=S_1\&\overline{B}_2$;





3) $\Sigma_3=S_2\&\overline{B}_3$;

4) $\Sigma_4=S_3\&\overline{B}_4$;

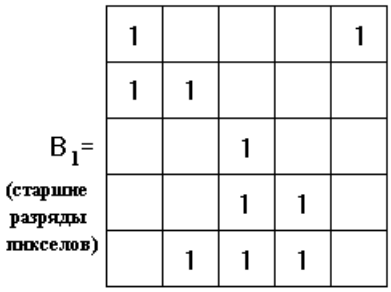
D_3=-7+4=-3; D_3=-3; d_3=1; D_3<0; D_4=-3+2=-1; d_4=1;

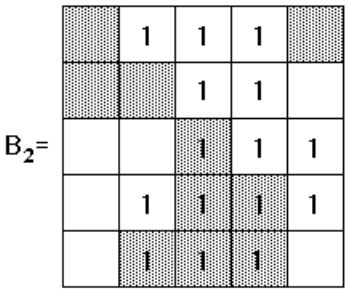
S_3= S_2& B_3; S_4=S_3& $\overline{B}_4$;

“1” в S4 показывает, где расположен I_R.

2. Пусть для того же окна надо найти элемент ранга <u>R=18</u>.

<u>R=18</u>; I_{R18}=10.



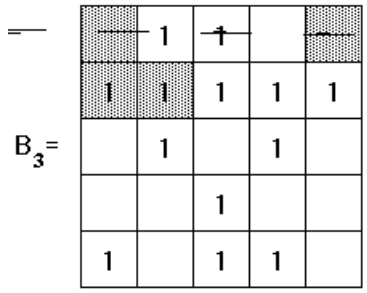


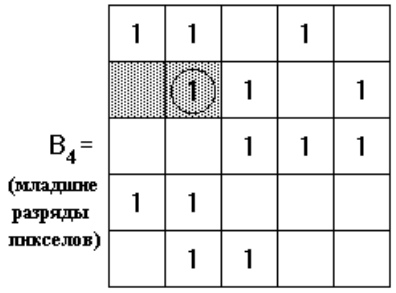
0) S_0=11...1111; 1) Σ_1=S_0& $\overline{B}_1$=15; 2) Σ_2=S_1& =4;

D_0=-18; D_1=-18+15=-3; D_1<0;d_1=1; D_2=-3+4=1;

D_1=-3; S_1= S_0& B_1; d_2=0; D_2>0; D_2=-3;

S_2=S_1& $\overline{B}_2$;





3) Σ_3=S_2& $\overline{B}_3$=2; 4) Σ_4=S_3& $\overline{B}_4$=1;

D_3=-3+2=-1; D_3<0; D_3=-1; d_3=1; D_4=-1+1=0; d_4=0; D_4=0;

S_3= S_2& B_3; S_4=S_3& $\overline{B}_4$;

“1” в S4 показывает, где расположен I_R.

5.2. Взвешенная ранговая фильтрация

Дальнейшим развитием метода ранговой фильтрации является процедура взвешенной ранговой фильтраци (ВРФ). При выполнении ВРФ задаётся вектор Z_M (вектор взвешивания). Каждый элемент Z_m $m \in \overline{1, M}$ указывает, сколько раз должен быть повторён элемент вектора, занимающий ту же позицию в окне сканирования (в неупорядоченном).Поэтому процедура ВРФ состоит из 2-х этапов [10, 15]:

1. “Взвешивание” элементов с помощью весовых коэффициентов;

2. Поиск элемента с рангом R.

При этом наиболее просто выполнение указанных процедур осуществляется разрядно-срезовыми алгоритмами. Разрядно-срезовый алгоритм взвешенной ранговой фильтрации аналогичен алгоритму ранговой фильтрации, за исключением операции модификации текущей суммы, которая принимает вид :

$$D_q = D_{q-1} + \sum_m \sum_n z_{mn}(s_{mn}^{q-1} \& \bar{b}_{mn}^{q}) \qquad (5.5)$$

Таким образом, ранговая фильтрация может рассматриваться как частный случай взвешенной ранговой фильтрации, когда $z_{mn} = 1$ для всех значений m и n. Если $z_{mn} = \{0; 1\}$, то выполняется процедура ранговой фильтрации с произвольной формой окна .

Согласно литературе, ВРФ приводит к меньшим искажениям мелких деталей изображений.

5.3. Скользящая эквализация гистограмм

Скользящая эквализация гистограмм является процедурой, обратной по своему алгоритму процедуре ранговой фильтрации. Элемент преобразованного изображения при скользящей эквализации определяется рангом центрального элемента окна исходного изображения.

Выполнение процедуры скользящей эквализации при использовании гистограммного алгоритма сводится к вычислению следующей суммы [17,21]:

$$R = \sum_{q=1}^{B_Ц} H(q) \qquad (5.6)$$

Здесь H(q) - q-ый отсчет гистограммы, R - ранг центрального элемента окна $B^{ц}$.

Алгоритм ранговой фильтрации при незначительном видоизменении базовой операции может быть сведен к алгоритму скользящей эквализации гистограмм. При выполнении такой процедуры элемент преобразованного изображения определяется рангом центрального элемента окна исходного изображения .

Однако важное отличие процедуры скользящей эквализации от ранговой и взвешенной ранговой фильтрации заключается в увеличении размеров окна

сканирования. При обработке двумерных сигналов (изображений) размер окна сканирования выбирается из условия

$$M^2 >= 2^Q,$$

Где Q – разрядность отсчетов сигнала (желательно при этом выбирать ближайшее значение M, при котором соблюдается указанное условие) .

Разрядно-срезовый алгоритм обратной ранговой фильтрации с послойным маскированием приобретает вид (для некоторого произвольного положения окна) [17]:

Начало :

$D_0 := -M^2$;

$b^Ц := d^1 d^2 d^3 ... d^Ц$;

$S_0 := 1111...1$;

для $q = 1, Q$ *цикл:*

$\Delta D := \sum_q^1 \overline{B}_q \& S_{q-1}$;

если $d_q^{\ddot{o}} = 0$, *то* $\left\{ D_q := D_{q-1} + \Delta D; S_q := S_{q-1} \& \overline{B}_q \right\}$

иначе $\left\{ D_q := D_{q-1}; S_q := S_{q-1} \& B_q \right\}$;

конец цикла по q ;

$R^Ц := abs(D_Q)$;

конец.

Здесь $B^ц$ - центральный элемент окна изображения. Заметим, что алгоритмы обратной ранговой фильтрации могут быть получены на базе алгоритмов как обычной, так и взвешенной ранговой фильтрации.

Ниже приведены примеры выполнения скользящей эквализации гистограмм.

Пример1. Выполнение эквализации на основе гистограммного алгоритма.

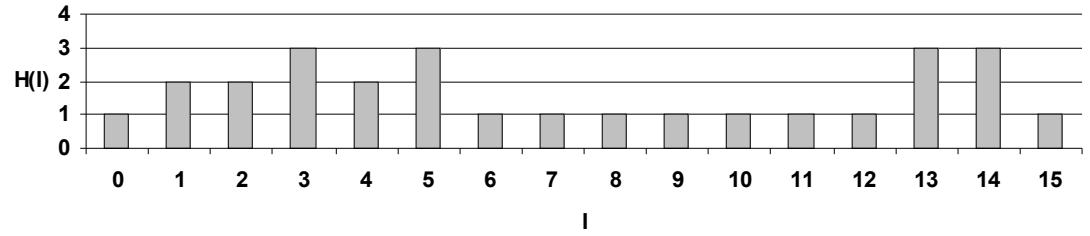
9	7	6	5	8
10	11	7	6	3
1	2	(13)	7	5
1	5	14	12	4
2	13	15	14	0

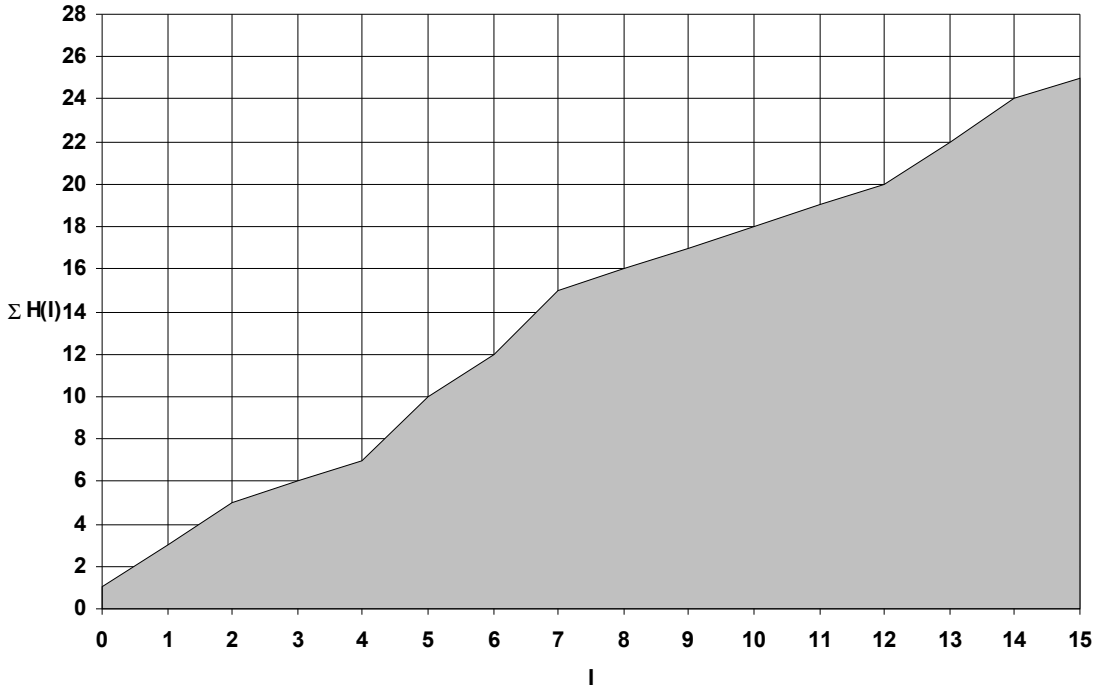
$B_{ср}$

I_R=13

R_{13} =22

y_{ij}=R_{13}=22





Пример 2. Эквализация на основе разрядно-срезового алгоритма с послойным маскированием.

9	7	6	5	8
10	11	7	6	3
1	2	(13)	7	5
1	5	14	12	4
2	13	15	14	0

0) D_0=-25

$b^{ц}$=1101

S_0=111...111

1) D_1=-25+10=-15

b_1=1; D_1=-25

$S_1=S_0\&B_1$

B_1 =

1				1
1	1			
		1		
		1	1	
	1	1	1	

2) D_2=-25+6=-19

b_2=1; D_2=-25

$S_2=S_1\&B_2$

B_2 =

	1	1	1	
		1	1	
		1	1	1
	1	1	1	1
	1	1	1	

B_3 =

	1	1		
1	1	1	1	1
	1		1	
		1		
1		1	1	

B_4 =

1	1		1	
	1	1		1
1		1	1	1
1	1			
	1	1		

3) D_3=-25+3=-22

b_3=0; $\underline{D_3}$=-22

$S_3=S_2\&B_3$

4) D_4=-22+2=-20

b_4=1; $D_4=D_3$

$S_4=S_2\&B_3$ $R=|D_4|=22$

5.4. Преобразование гистограмм распределения

Процедура модификации гистограммы обеспечивает изменение параметров гистограммы (глобально или локально) по различным законам [21]:

- по экспоненциальному закону (задается параметр α - показатель степени экспоненты);
- по закону гиперболической функции;
- по закону распределения Рэллея или распределения степени *2/3*;

- возможна также эквализация (лианеризация) и степенная интенсификация гистограммы.

Суть подобной процедуры иллюстрируется ниже на рис.5.2. На рис.5.2,а) приведена первоначальная гистограмма распределения яркостей (амплитуд) отсчетов изображения (пунктиром показаны желаемые распределения).

На рис.5.2,б) - гистограмма распределения яркостей после ее модификации для результирующего изображения.

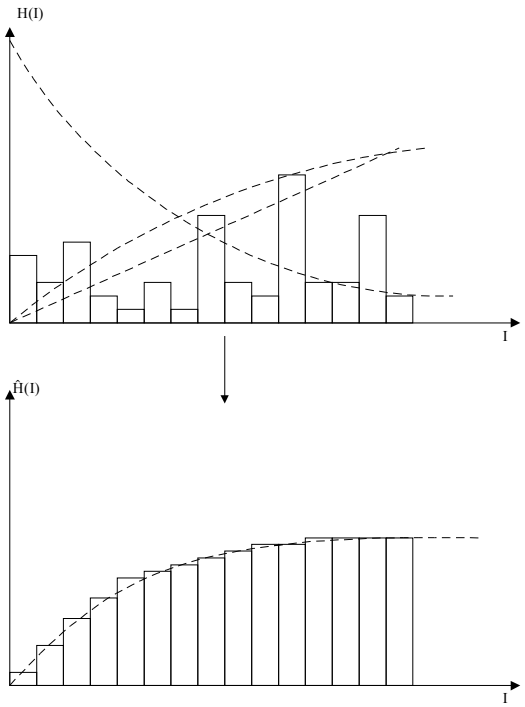


Рис.5.2. Глобальная эквализация гистограмм

При выполнении процедуры глобальной эквализации производится выравнивание гистограммы уровней яркости изображения на основе формирования гистограммы всего изображения.

$$H = [h_g], \quad g = 0, 255.$$

Преобразование выполняется в два этапа [21]:

1. Вычисляется таблица преобразования $T = [t_g]$, обеспечивающая выравнивание гистограммы H изображения :

$$t_k = \sum_{i=0}^{k} h_g \ , \qquad k = 0, 255 .$$

2. Элементы изображения преобразуются по таблице T и нормируются для приведения яркостей результирующего изображения в диапазон [0,255]:

$$y_{ij} = (255 * T[a_{ij}])/(I * J),$$

где $I \times J$ - размеры изображения A.

6. КОНТРОЛЬНЫЕ ВОПРОСЫ И ЗАДАЧИ

Вариант 1

1. Сигнал имеет частотный спектр, ограниченный частотой

   F_{max} = 10 КГц, причем разрешение по частоте составляет 100 Гц.

   В течении какого промежутка времени должен наблюдаться сигнал ? Через какие промежутки времени должны сниматься отсчеты сигнала ?

2. Заданы последовательности

   G = [0; 1; 2] и X = [0; 1; 2].

   Вычислить апериодическую свертку и корреляцию.

3. Выполнить двоично-инверсную перестановку вектора данных

   $X = [x_0 ; x_1; x_2; x_3; ..., x_{N-2}; x_{N-1}]^T$ (N = 16)

4. Проверить, является ли ортогональным ядро преобразования

   $f_k = \Sigma \; x_n \cos[2\pi kn/N]$ для N = 4.

5. Построить граф БПФ для N = 4 с прореживанием по частоте.

6. Показать, что с точки зрения обепечения минимума вычислительных затрат предпочтителен алгоритм БПФ по основанию 4.

7. Определить максимальный размер M окна сканирования, при котором предпочтителен прямой алгоритм вычисления свертки, если N = 2048, а исходные данные $X=[x_0 ; x_1; x_2; x_3; ..., x_{N-2}; x_{N-1}]$ и ядро свертки $G = [g_0, g_1,...,g_{M-1}]$ комплексные.

8. Построить функцию пропускания фильтра высоких частот с граничной частотой $f_{гр}$ = 100 Кгц, при условии, что число отсчетов спектра N=100 и разрешение по частоте Δf = 20 Кгц. Привести рисунок.

9. Задан вектор X =[0,0,1,1,2,3,2,1,1,0,0,0]. Определить вектор Y с отсчетами отфильтрованного сигнала при использовании нерекурсивного линейного фильтра с коэффициентами H = [1,4,1] ("краевыми" эффектами пренебречь).

10. Задана матрица взаимокорреляции:

$$R = \begin{vmatrix} 2 & 1 & -1 \\ 2 & 2 & -1 \end{vmatrix}$$

|-1 2 1|

Найти для нее обратную матрицу.

11. Для некоторого j-го положения окна сканирования нелинейным фильтром обрабатывается фрагмент вектора исходных данных

X(j) =[0, 1, 12, 1, 15, 13, 10, 3, 2, 2, 0, 15, 13].

Обработка выполняется методом ранговой фильтрации. Используя гистограммный алгоритм, найти элементы следующих рангов:

а) R = 9;

б) R = 5.

Вариант 2

1. Сигнал наблюдается в течении 10 сек., причем отсчеты сигнала снимаются через 10 мксек. Какова предельная частота сигнала F_{max} может быть зафиксирована ? Какое разрешение по частоте будет обеспечиваться в этом случае ?
2. Заданы последовательности

G = [1; 1; 2] и X = [0; 1; 2].

Вычислить циклическую свертку и корреляцию.

3. Выполнить перестановку вектора данных

$X = [x_0; x_1; x_2; x_3; ..., x_{N-2}; x_{N-1}]^T$

с троичным прореживанием (N = 9).

4. Проверить, является ли ортогональным ядро преобразования

$f_k = \Sigma\blacksquare\, x_n \sin[2\pi kn/N]$ для N = 4.

5. Построить граф БПФ для N = 9 с прореживанием по времени.
6. Определить вычислительную сложность алгоритма БПФ для $N = 3^6$ (в числе операций умножения действительных чисел).
7. Выполнить оценку вычислительной сложности разрядно-срезового алгоритма сверки/корреляции в сравнении с вычислительной сложностью прямого алгоритма свертки/корреляции.

8. 8. Построить функцию пропускания фильтра низких частот с граничной частотой $f_{гр}$ = 750 Кгц, при условии, что число отсчетов спектра N=100 и разрешение по частоте Δf = 50 Кгц. Привести рисунок.
9. Задан вектор X =[0,0,1,1,2,3,2,1,0,1,0,0]. Определить вектор Y с отсчетами отфильтрованного сигнала при использовании рекурсивного линейного фильтра с коэффициентами H = [1,3,1] и

   B = [-1/2, 1] ("краевыми" эффектами пренебречь).
10. Задана матрица взаимокорреляции:

$$R = \begin{vmatrix} 2 & 1 & 0 \\ 0 & 2 & -1 \\ -2 & -2 & 2 \end{vmatrix}$$

    Найти для нее обратную матрицу.
11. Для некоторого j-го положения окна сканирования нелинейным фильтром обрабатывается фрагмент вектора исходных данных

    X(j) =[0, 0, 1, 10, 0, 1, 1, 3, 2, 12, 2, 3, 2, 0, 0].

Обработка выполняется методом медианной фильтрации. Используя разрядно-срезовый алгоритм, найти медиану.

Вариант 3

1. Сигнал имеет частотный спектр, ограниченный частотой F_{max} = 10 КГц, и наблюдается в течение 0,1 сек.

   Каким будет в этом случае разрешение по частоте ? Через какие промежутки времени должны сниматься отсчеты сигнала ?
2. Заданы последовательности

   G = [1; 2; 0] и X = [0; 2; 1].

   Вычислить циклическую сверку и апериодическую корреляцию.
3. Построить перестановочную матрицу, с помощью которой можно было бы сгруппировать вначале нечетные элементы вектора, а затем четные (и те и другие - в порядке возрастания индексов) для N = 8.
4. Проверить, является ли ортогональным ядро преобразования

   $f_k = \Sigma\blacksquare\, x_n(-1)^{kn/N}$ для N = 4.

5. Построить граф БПФ для N = 6 с прореживанием по времени.
6. Определить вычислительную сложность алгоритма БПФ для $N = 2^{10}$ (в числе операций умножения действительных чисел).
7. Выполнить оценку вычислительной сложности разрядно-срезового алгоритма сверки/корреляции в сравнении с вычислительной сложностью прямого алгоритма свертки/корреляции.
8. Построить функцию пропускания фильтра низких частот с граничной частотой $f_{гр}$ = 750 Кгц, при условии, что число отсчетов спектра N=100 и разрешение по частоте Δf = 50 Кгц. Привести рисунок.
9. Задан вектор X =[0,0,1,1,2,3,2,1,0,1,0,0]. Определить вектор Y с осчетами отфильтрованного сигнала при использовании рекурсивного линейного фильтра с коэффициентами H = [1,3,1] и B = [-1/2, 1] ("краевыми эффектами пренебречь).
10. Задана матрица взаимокорреляции:

$$R = \begin{vmatrix} 2 & 1 & 0 \\ 0 & 2 & -1 \\ -2 & -2 & 2 \end{vmatrix}$$

Найти для нее обратную матрицу.

11. Для некоторого j-го положения окна сканирования нелинейным фильтром обрабатывается фрагмент вектора исходных данных

X(j) =[0, 0, 1, 10, 0, 1, 1, 3, 2, 12, 2, 3, 2, 0, 0].

Обработка выполняется методом медианной фильтрации. Используя разрядно-срезовый алгоритм, найти медиану.

ЛИТЕРАТУРА

1. . Аллен Дж. Архитектура процессоров для цифровой обработки сигналов.// ТИИЭР. -1986. -т.73.-N 5.-с.3-37.
2. Бандман О.Л., Миренков Н.Н., Седухин С.Г. и др.Специализированные процессоры для высокопроизводительной обработки данных.//-Новосибирск.: Наука. -1988. – 135 с.
3. Брейсуэлл Р. Преобразование Хартли. Теория и приложения. М.: Мир, 1990.
4. Воеводин В.В., Кузнецов Ю.А. Матрицы и вычисления.//-М,; Наука,-1984, -318 с.
5. Грачев В.А., Кухарев Г.А. Специализированные процессоры в системах пространственно-временной обработки сигналов. Обзор по данным отечественной и зарубежной печати.// -Л.: ЦНИИ "Румб",-1991, -57 с.
6. Губин А.В., Шарыпин Е.В. Архитектура персональных систем обработки изображений.// Системы и средства информатики. -Ежегодник.- Вып.1. -1989. -с.74-81.
7. Гусев В.Г. Системы пространственно-временной обработки гидроакустической информации //-Л.: Судостроение,-1988, -263 с.
8. Дагман Э.Е.,Кухарев Г.А. Быстрые дискретные ортогональные преобразования.//-Новосибирск: Наука, -1983, -232 с.
9. Дымков В.И., Синицын Е.В. Элементы концепции персональных систем обработки изображений.// В сб. "Системы и средства информатики".Вып.1. -1989.-с.74-81.
10. Клочков В.С., Романов Ю.Ф., Тропченко А.Ю., Юсупов К.М. Конвейерные разрядно-срезовые процессоры ранговой и взвешенной ранговой фильтрации изображений // Известия вузов СССР - Приборостроение, -1991, -т.34, -N.2, -с.24 - 29.
11. Кодирование и обработка изображений.//- М.: Наука , -1988. -175 с.
12. Кухарев Г.А.,Тропченко А.Ю.,Шмерко В.П. Систолические процессоры для обработки сигналов.// -Минск: Беларусь, -1988, -127 с.

13. Кухарев Г.А.,Тропченко А.Ю. Систолический процессор для обращения матриц.// Известия вузов СССР.-Приборостроение,-1990, -т.33, -N 11, -с.
14. Кучеренко К.И., Очин Е.Ф. Процессоры двумерной медианной фильтрации на основе сортирующих сетей.// Автометрия. - 1988. - N 21. - с.92-94.
15. Кучеренко К.И., Никитин А.В. О реализации взвешенной медианной фильтрации сигналов в реальном масштабе времени // "SIAP-89". - Рига. - 1989. - т.2 - с. 43-45.
16. Маккелан Д.Х., Рейдер Ч.М. Применение теории чисел в цифровой обработке сигналов. М.: Радио и связь, 1983.
17. Очин Е.Ф. Вычислительные системы обработки изображений.// - Л.: Энергоатомиздат. Ленингр. отд-ние. -1989. -136 с.
18. Параллельная обработка информации. Параллельные методы и средства распознавания образов. Т.2 -Киев: Наукова думка, -1985, - 279 с.
19. Параллельная обработка информации Т.3 -Киев: Наукова думка, -1986, -288 с.
20. Применение цифровой обработки сигналов.//Под.ред. А.Оппенгейма.-М.:Мир, -1980.-550 с.
21. Прэт У. Цифровая обработка изображений.//-М.:Мир,-1982, -т.1, - с. -т.2. - с.
22. Рабинер Л.,Гоулд Б. Теория и применения цифровой обработки сигналов.//-М.: Мир, -1978, - с.
23. Романов Ю.Ф., Тропченко А.Ю., Юсупов К.М. Систолические мультигистограммные и разрядно-срезовые процессоры ранговой фильтрации // Известия вузов СССР - Приборостроение, -1991, -т.34, -N.12, -с.26 – 30.
24. Сергиенко А.Б. Цифровая обработка сигналов. - СПб., "Питер". 2002, - 608 с.
25. Солонина А., Улахович Д., Яковлев Л. Алгоритмы и процессоры цифровой обработки сигналов. – СПб, БХВ-Петербург, 2001, 464 с.
26. Тропченко А.Ю., Романов Ю.Ф. Алгоритмы быстрого преобразования Хартли при различных основаниях и конвейерные структуры для их реализации // "Известия вузов-Приборостроение", 1993, т.36, N 4, стр.

27. Ту Дж., Гонсалес Р. Принципы распознавания образов. -М.; "Мир", 1978, - 368 с.
28. Уидроу Б., Стирнз С. Адаптивная обработка сигналов // -М.; "Радио и связь", 1989, - 440 с.
29. Фрумкин М.А. Систолические вычисления.// - М.: Наука, -1990. - 191 с.
30. Ярославский Л.П. Цифровая обработка сигналов в оптике и голографии: Введение в цифровую оптику.//- М.: Радио и связь. -1987.- 296 с.
31. Graps A.L. An introduction to Wavelets.// IEEE Computational Sciences and Engeneering. V.2, N.2, Summer 1995, p.p. 50-61.